Généralisations de la théorie PAC-bayésienne pour l'apprentissage inductif, l'apprentissage transductif et l'adaptation de domaine

Thèse

Pascal Germain

Doctorat en informatique
Philosophiæ doctor (Ph.D.)

Québec, Canada

Résumé

En apprentissage automatique, l'approche PAC-bayésienne permet d'obtenir des garanties statistiques sur le risque de votes de majorité pondérés de plusieurs classificateurs (nommés *votants*). La théorie PAC-bayésienne «classique», initiée par McAllester (1999), étudie le cadre d'apprentissage *inductif*, sous l'hypothèse que les exemples d'apprentissage sont générés de manière indépendante et qu'ils sont identiquement distribués (*i.i.d.*) selon une distribution de probabilité inconnue mais fixe. Les contributions de la thèse se divisent en deux parties.

Nous présentons d'abord une analyse des votes de majorité, fondée sur l'étude de la *marge* comme variable aléatoire. Il en découle une conceptualisation originale de la théorie PAC-bayésienne. Notre approche, très générale, permet de retrouver plusieurs résultats existants pour le cadre d'apprentissage inductif, ainsi que de les relier entre eux. Nous mettons notamment en lumière l'importance de la notion d'*espérance de désaccord* entre les votants.

Bâtissant sur une compréhension approfondie de la théorie PAC-bayésienne, acquise dans le cadre inductif, nous l'étendons ensuite à deux autres cadres d'apprentissage. D'une part, nous étudions le cadre d'apprentissage *transductif*, dans lequel les descriptions des exemples à classifier sont connues de l'algorithme d'apprentissage. Dans ce contexte, nous formulons des bornes sur le risque du vote de majorité qui améliorent celles de la littérature. D'autre part, nous étudions le cadre de l'*adaptation de domaine*, dans lequel la distribution génératrice des exemples étiquetés de l'échantillon d'entraînement diffère de la distribution générative des exemples sur lesquels sera employé le classificateur. Grâce à une analyse théorique – qui se révèle être la première approche PAC-bayésienne de ce cadre d'apprentissage –, nous concevons un algorithme d'apprentissage automatique dédié à l'adaptation de domaine. Nos expérimentations empiriques montrent que notre algorithme est compétitif avec l'état de l'art.

Abstract

In machine learning, the PAC-Bayesian approach provides statistical guarantees on the risk of a weighted majority vote of many classifiers (named *voters*). The "classical" PAC-Bayesian theory, initiated by McAllester (1999), studies the *inductive* learning framework under the assumption that the learning examples are independently generated and are identically distributed (*i.i.d.*) according to an unknown but fixed probability distribution. The thesis contributions are divided in two major parts.

First, we present an analysis of majority votes based on the study of the *margin* as a random variable. It follows a new conceptualization of the PAC-Bayesian theory. Our very general approach allows us to recover several existing results for the inductive PAC-Bayesian framework, and link them in a whole. Among other things, we highlight the notion of *expected disagreement* between the voters.

Building upon an improved understanding of the PAC-Bayesian theory, gained by studying the inductive framework, we then extend it to two other learning frameworks. On the one hand, we study the *transductive* framework, where the learning algorithm knows the description of the examples to be classified. In this context, we state risk bounds on majority votes that improve those from the current literature. On the other hand, we study the *domain adaptation* framework, where the generating distribution of the labelled learning examples differs from the generating distribution of the examples to be classified. Our theoretical analysis is the first PAC-Bayesian approach of this learning framework, and allows us to conceive a new machine learning algorithm for domain adaptation. Our empirical experiments show that our algorithm is competitive with other state-of-the-art algorithms.

Table des matières

Liste des tableaux

Liste des figures

Liste des algorithmes

[The] question of whether Machines Can Think [...] is about as relevant as the question of whether Submarines Can Swim.

The threats to computing science,
Edsger W. Dijkstra (1984)

RACHAEL : It seems you feel our work is not a benefit to the public.
DECKARD : Replicants are like any other machine. They're either a benefit or a hazard.

Blade Runner, réalisation de Ridley Scott (1982),
d'après *Do Androids Dream of Electric Sheep ?*,
Philip K. Dick (1964)

– Vous n'aimez pas le travail ? dit l'antiquitaire.
– C'est horrible, dit Colin. Ca rabaisse l'homme au rang de la machine.

L'écume des jours, Boris Vian (1947)

Remerciements

L'histoire de ces 2^8 pages commence le 2 décembre 2005, quand ce courriel fut déposé sans bruit dans ma boîte de messagerie :

> Je tiens d'abord à vous féliciter de votre excellente prestation à vos deux premiers examens d'analyse d'algorithmes [...]. Ainsi, si ça vous intéresse d'acquérir une expérience en recherche scientifique, je vous invite à venir rencontrer François Laviolette pour discuter des programmes de bourses du CRSNG (principal organisme subventionnant les activités de recherche au Canada) qui vous permettent de faire un stage de recherche [...]. En espérant que vous considèrerez la possibilité de faire de la recherche scientifique au cours de votre carrière.
>
> meilleures salutations,
> -mario

C'est donc à l'été 2006 que j'entrepris un stage de recherche 1er cycle et que je commençai mes collaborations avec François Laviolette (qui allait vite devenir mon directeur de recherche), Mario Marchand (qui allait devenir mon codirecteur) et Alexandre Lacasse (qui commençait son doctorat, et qui fut un confrère essentiel). Je ne présageais que vaguement poursuivre mes études à la maîtrise, et nullement au doctorat. Mais, comme le temps passe vite – c'est une vérité intemporelle – voilà que je dépose ma thèse ! Comme beaucoup d'histoires d'étudiants gradués, la mienne fut parsemée de nombreux apprentissages, d'essais et d'erreurs, de découragement ponctuel et de persévérance, de caféine et d'espoir. Bien sûr, l'objectif de ces remerciements n'est pas de relater cette histoire, mais de souligner la contribution de ses protagonistes. Car, si j'ai passé toutes ces années au sein de la même équipe de recherche, c'est beaucoup grâce à la qualité des gens que j'ai eu la chance d'y côtoyer.

Avant toutes choses, je suis extrêmement reconnaissant à François Laviolette et Mario Marchand pour la confiance qu'ils ont toujours témoignée à mon égard, et pour toutes les grandes et petites choses que j'ai apprises d'eux. Pour n'en nommer que deux seules, je remercie François de m'avoir permis de développer mon goût de l'élégance des démonstrations mathématiques et Mario de m'avoir montré la rigueur dans l'écriture scientifique. Outre mes directeurs, je remercie grandement Francis Bach, Claude-Guy Quimper et Luc Lamontagne d'avoir accepté de former le jury de ma thèse.

Plusieurs des contributions contenues dans cette thèse sont le fruit de collaborations avec des chercheurs d'autres horizons. J'ai profité de l'expertise d'Emilie Morvant et d'Amaury

Habrad en adaptation de domaine. Notre bonne entente et nos six heures de décalage horaire ont permis à notre article «PBDA» de s'écrire par delà nos cycles de sommeils respectifs. J'ai aussi entrepris une stimulante collaboration avec Luc Bégin, qui, de son regard perspicace de physicien, m'a amené à mieux décortiquer les fondements de la théorie PAC-bayésienne. Cela a notamment mené à nos travaux sur l'apprentissage transductif. Plus près de moi (à environ deux mètres du dossier de ma chaise), mon collègue Jean-Francis Roy m'a accompagné dans plusieurs réalisations, notamment celle de l'article de revue «Risk Bounds for the Majority Vote» (connu à l'interne sous le nom de code du «Never Ending Paper»), dont la rédaction s'est étendue sur plusieurs années et en plusieurs lieux, jusqu'au *Laundromat Café* de Reykjavik. Enfin, bien que nos publications laissent peu de traces de nos nombreux échanges, Alexandre Lacoste a été au quotidien un collègue important de mon doctorat, tant par ses idées foisonnantes de chercheur que par sa bonne humeur contagieuse. Je tiens à souligné qu'il est l'auteur des librairies dont je me suis servis pour exécuter la plupart de mes expérimentations sur les superordinateurs *Colosse* (Calcul Québec) et *SciNet* (Calcul Canada).

Je salue aussi tous les autres *Graaliens* que je n'ai pas encore nommés dans ce texte et qui ont fait du *Groupe de recherche en apprentissage automatique de Laval* un environnement de travail si agréable : Hana Ajakan, Sébastien Boisvert, Francis Brochu, Alexandre Drouin, Mathieu Dumoulin, Sébastien Giguère, Amélie Rolland, Sara Shanian, Prudencio Tossou, Francis Turgeon-Boutin, Brice Zirakiza. Chacun à leur manière, les sympathiques membres du *GRAAL* ont été d'excellents compagnons de route ; certains pour collaborer à des travaux ou projets, certains pour échanger des idées de recherche, certains pour partager un café, certains pour débattre d'enjeux sociaux, certains pour fêter l'acceptation d'un article, certains pour partager des anecdotes de conférences, et plusieurs pour (presque) tout cela à la fois.

Il m'importe de remercier ceux qui contribuent à soutenir un environnement propice à la recherche académique à l'échelle locale. Je pense d'abord au personnel administratif du département (Lynda Goulet, Denise Thibault, Alexandra Côté, Marie-Noelle Morin, ...) pour leur serviabilité, leur efficacité et leur bon sens. Je pense aussi aux techniciens de *Colosse* pour leur dévouement, lequel m'a parfois permis d'ajouter des résultats empiriques à mes articles quelques jours avant leur date de tombée. Plus généralement, je me sais redevable aux contribuables canadiens et québécois de m'avoir versé, par le biais des organismes subventionnaires, un peu de leurs impôts en bourses d'étude et de recherche.

Enfin, en cet ultime paragraphe de ces sincères remerciements, je souligne l'apport implicite à cette thèse de mon ami Germain et de ma sœur Anne, qui m'ont hébergé alors que j'avais besoin de changer occasionnellement de lieu lors de la rédaction. Aussi, de la même manière que je l'ai fait par le passé (voir Germain, 2009) – je n'ai pas trouvé de meilleures formules – je remercie avec amour et admiration Thérèse et Raymond, parents indéfectibles dans le soutien qu'ils apportent à leurs enfants. Finalement, je réserve cette toute dernière phrase à ma filleule Luz, qui m'aide à me remémorer l'importance de s'émerveiller.

Avant-propos

Une grande partie du contenu de cette thèse a fait l'objet de publications dans des comptes rendus de conférences évalués par les pairs (Germain et al., 2013; Bégin et al., 2014), d'ateliers de travail (Germain et al., 2012b, 2014) ou d'articles de revues (Germain et al., 2015b). Ces articles sont le fruit d'étroites collaborations avec mes collègues. Les auteurs y sont d'ailleurs cités en ordre alphabétique, car il est souvent difficile d'attribuer le titre de «premier auteur» à l'un ou l'autre des collaborateurs. Cela étant dit, le corps de la thèse met l'accent sur les idées et résultats dont je peux m'attribuer ou partager la paternité. Chaque chapitre concerné débute par une référence aux articles qui lui sont liés. Un travail a été fait pour unifier les concepts et les notations, afin que s'en dégage une vue d'ensemble.

Autres collaborations

Les manuscrits de trois autres articles auxquels j'ai collaboré au cours de mon doctorat sont joints en annexe. Ces derniers n'ont pas été inclus dans le corps de la thèse pour deux raisons principales :

1. Le premier article (annexe C) fait l'objet d'une présentation détaillée dans une autre thèse (Shanian, 2012), et j'ai préféré me concentrer sur mes contributions plus récentes ;
2. Les deux articles suivants (annexes D et E) ne s'inscrivent pas dans l'axe «PAC-bayésien» de la thèse, mais découlent de questionnements rencontrés lors de l'étude de cadres d'apprentissage abordés lors de mes recherches.

Annexe C : A PAC-Bayes Sample Compression Approach to Kernel Methods (Germain, Lacoste, Laviolette, Marchand et Shanian, 2011)

Cet article de conférence constitue le travail au cœur de la thèse de Sara Shanian (Shanian, 2012). Dans la première année de mon doctorat, Alexandre Lacoste et moi avons collaboré avec Sara Shanian – nous partagions les mêmes superviseurs, François Laviolette et Mario Marchand – sur plusieurs aspects de ce travail, principalement pour simplifier les démonstrations des théorèmes et implémenter les algorithmes d'apprentissage.

Annexe D : A Pseudo-Boolean Set Covering Machine
(Germain, Giguère, Roy, Zirakiza, Laviolette et Quimper, 2012a)

Cet article de conférence a été amorcé par quatre étudiants (Sébastien Giguère, Jean-Francis Roy, Brice Zirakiza et moi-même) dans le cadre d'un travail de session pour le cours *Optimisation combinatoire*. Ces quatre étudiants ont contribué également à la conception de la méthode, à l'implémentation, aux expérimentations et à l'écriture de l'article. Les professeurs François Laviolette et Claude-Guy Quimper ont supervisé le projet et révisé l'article.

Annexe E : Domain-Adversarial Neural Networks
(Ajakan, Germain, Larochelle, Laviolette et Marchand, 2014)

L'idée de l'algorithme d'apprentissage présentée dans ce compte-rendu d'atelier est née d'une collaboration entre les professeurs François Laviolette, Hugo Larochelle et Mario Marchand. J'ai rédigé la section théorique de l'article, et j'ai collaboré avec Hana Ajakan (étudiante à la maîtrise) à la réalisation des expérimentations empiriques.

Première partie

Mise en contexte

Chapitre 1

Introduction

Au cours des dernières décennies, le recours aux systèmes informatiques s'est répandu dans la majorité des sphères de l'activité humaine. En tant que machine programmable, l'ordinateur a permis d'automatiser plusieurs tâches existantes. Il a aussi permis d'accomplir plusieurs tâches difficilement envisageables pour un humain, notamment grâce à sa grande capacité de traitement de l'information. Ces multiples usages nécessitent beaucoup de travail en amont ; traditionnellement, lorsqu'un individu désire utiliser un ordinateur pour accomplir une tâche, il doit le programmer. En plus de reposer sur une expertise technique, ce travail de programmation requiert parfois une connaissance approfondie d'un domaine particulier. À titre d'exemple, la conception d'un programme qui permet de poser un diagnostic médical requiert l'intervention d'un médecin, et les connaissances d'un ornithologue sont requises pour concevoir un programme qui identifie un oiseau sur une photographie. De même, on peut aisément énoncer d'autres situations exigeant l'aide d'un autre type de spécialiste : traducteur, pilote, musicien, ingénieur, et ainsi de suite...

L'***apprentissage automatique*** (ou, en anglais, *machine learning*) permet d'aborder ce type de problèmes sous un angle différent. Plutôt que de créer un nouveau programme de manière explicite, un algorithme d'apprentissage automatique permet d'«entraîner» le système informatique à effectuer cette tâche en lui fournissant une série d'exemples propres à ce qu'il doit «apprendre». Ainsi, un assistant de diagnostic médical peut être entraîné à l'aide d'une banque de dossiers médicaux de patients, un outil d'identification d'oiseaux à l'aide d'images préalablement identifiés, un système de traduction anglais-français à l'aide de documents rédigés dans les deux langues, etc.

1.1 Le problème de la classification et les votes de majorité

Dans le contexte de la classification, un ***algorithme d'apprentissage*** est entraîné sur un *échantillon de données* contenant une série d'***exemples***. Chaque exemple est caractérisé par une ***description*** et une ***étiquette***. L'objectif de l'algorithme d'apprentissage consiste à gé-

néraliser l'information contenue dans l'échantillon de données afin de créer un ***classificateur***, c'est-à-dire une fonction qui prend en entrée la description d'un exemple et prédit son étiquette. Un bon classificateur est caractérisé par un faible ***risque*** (ou ***erreur de généralisation***), c'est-à-dire une faible probabilité de classifier incorrectement un exemple qui n'a pas servi au processus d'apprentissage. Une préoccupation centrale de la théorie statistique de l'apprentissage est donc de développer des garanties sur le risque d'un classificateur.

Un ***vote de majorité*** consiste en une agrégation de plusieurs classificateurs – les ***votants*** – selon une pondération déterminée par un algorithme d'apprentissage. Plusieurs algorithmes d'apprentissage construisent explicitement des votes de majorité, telles les méthodes d'ensembles comme le *Boosting* (Freund et Schapire, 1997; Schapire et Singer, 1999), le *Bagging* (Breiman, 1996) et les *Random Forests* (Breiman, 2001). D'autres algorithmes construisent des classificateurs en apparence de natures très différentes, mais qui peuvent s'exprimer comme des votes de majorité à l'aide d'astuces mathématiques. C'est le cas des méthodes à noyaux (voir Langford et Shawe-Taylor, 2002) – famille à laquelle appartient le *SVM* (Cortes et Vapnik, 1995) – et de certains *réseaux de neurones* (voir McAllester, 2013).

1.2 Présentation de la théorie PAC-bayésienne

La ***théorie PAC-bayésienne***, initiée par McAllester (1999), permet d'obtenir des garanties sur le risque des votes de majorité. Plusieurs auteurs ont depuis contribué à améliorer et à généraliser le théorème de McAllester, dont Seeger (2002); Langford et Shawe-Taylor (2002); Maurer (2004); Lacasse et al. (2006); Catoni (2007); Germain et al. (2009a); Tolstikhin et Seldin (2013). Bien que possédant des formulations légèrement différentes, tous les résultats PAC-bayésiens partagent une même approche statistique de l'apprentissage automatique. En effet, la théorie PAC-bayésienne énonce des garanties *probablement approximativement correctes* (PAC) sur l'erreur de généralisation des classificateurs en incorporant, comme le font les approches bayésiennes, certaines connaissances *a priori* du problème d'apprentissage.

Précisons que l'***apprentissage PAC*** est une approche initiée par Valiant (1984) qui consiste à affirmer, avec une certaine probabilité «$1-\delta$» – d'où le terme «probablement» – qu'un classificateur h possède un risque inférieur à une certaine valeur «ϵ» – d'où le terme «approximativement correct». Sommairement, une borne de type PAC possède la forme suivante :

$$\Pr\Big(R_D(h) \leq \epsilon\Big) \;\geq\; 1-\delta\,,$$

où $R_D(h)$ représente le *risque* du classificateur h sur la distribution de données D.

Les bornes PAC les plus simples à obtenir sont basées sur l'acuité du classificateur sur un ***échantillon de test*** constitué d'exemples étiquetés qui n'ont pas servi au processus d'apprentissage, car le risque du classificateur sur l'échantillon de test s'avère un estimateur non biaisé de son «vrai» risque. Une particularité de la théorie PAC-bayésienne est qu'il n'est pas

nécessaire de recourir à un échantillon de test pour calculer des bornes sur le risque. En effet, elles se calculent à partir de l'échantillon d'entraînement duquel le classificateur est issu, et de quelques autres éléments que nous définirons rigoureusement au chapitre 4.

Très sommairement, une borne PAC-bayésienne possède habituellement la forme suivante :

$$\Pr\left(\Delta\Big(R_D(h_{post}), R_S(h_{post})\Big) \;\leq\; \frac{\mathrm{KL}(post \,\|\, prior) + \epsilon}{|S|}\right) \;\geq\; 1-\delta\,, \tag{1.1}$$

où le terme $\mathrm{KL}(post \,\|\, prior)$ mesure la divergence entre les connaissances *a priori* et *a posteriori* du problème d'apprentissage. Les connaissances *a posteriori* déterminent la nature du classificateur h_{post}, duquel on veut estimer le risque. Enfin, le terme $\Delta(R_D(h_{post}), R_S(h_{post}))$ est une «mesure de distance» entre

— le «vrai» *risque* $R_D(h_{post})$ du classificateur h_{post} sur la distribution de données D ; et

— le ***risque empirique*** $R_S(h_{post})$, c'est-à-dire le ratio d'erreurs que le classificateur h_{post} effectue sur l'échantillon d'entraînement S (contenant $|S|$ exemples).

Un choix possible de «mesure de distance» est la fonction quadratique $\Delta(q,p) \coloneqq (q-p)^2$, ce qui permet de réécrire l'équation (1.1) sous la forme

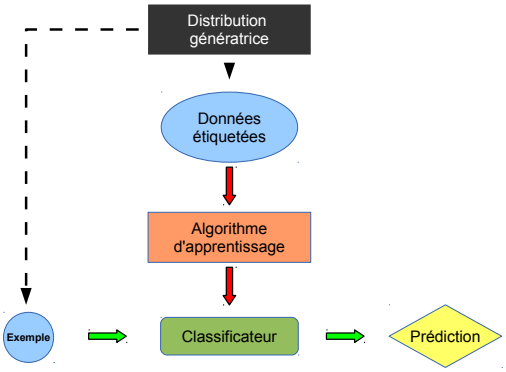
$$\Pr\left(R_D(h_{post}) \;\leq\; R_S(h_{post}) + \sqrt{\frac{\mathrm{KL}(post \,\|\, prior) + \epsilon}{|S|}}\right) \;\geq\; 1-\delta\,.$$

On obtient ainsi une garantie sur l'acuité d'un classificateur sans avoir à l'évaluer sur un échantillon de test. Il s'agit d'un avantage de la théorie PAC-bayésienne, particulièrement dans un contexte où il est coûteux (en temps ou en argent) d'obtenir des données étiquetées.

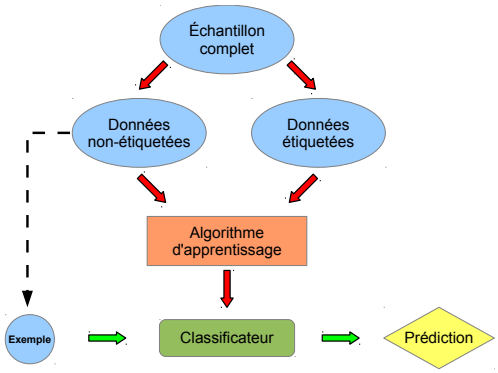
L'étude de la théorie PAC-bayésienne permet aussi le développement de nouvelles méthodes pour appréhender les problèmes d'apprentissage automatique ; chaque nouvelle borne PAC-bayésienne ouvrant la voie à la conception d'algorithmes d'apprentissage. En effet, en optimisant l'expression mathématique des bornes, on obtient le classificateur qui minimise la garantie théorique sur le risque. Les travaux de maîtrise ayant précédé cette thèse (Germain, 2009) ont montré, dans le cadre inductif, qu'une telle approche permet de découvrir des algorithmes d'apprentissage compétitifs avec l'état de l'art. Dans un contexte où l'apprentissage automatique est appelé à résoudre un nombre grandissant de problèmes de natures diverses, la théorie PAC-bayésienne peut donc servir de guide orientant l'approche adoptée – en lieu et place de stratégies *ad hoc* qui possèdent parfois peu de fondements théoriques.

1.3 Description de certains cadres d'apprentissage

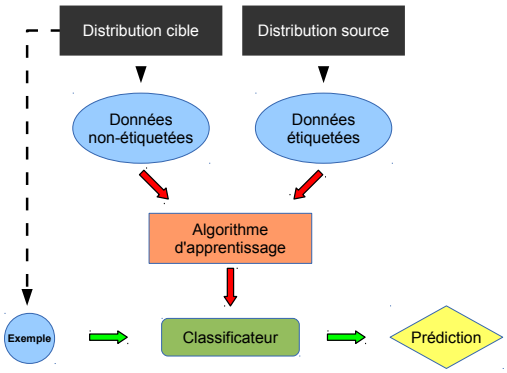
La théorie PAC-bayésienne «classique» étudie le cadre d'apprentissage inductif. Dans cette thèse, nous présentons une théorie PAC-bayésienne générale qui permet de retrouver plusieurs résultats connus pour l'apprentissage inductif. Nous élargissons ensuite cette théorie aux cadres de l'apprentissage transductif et de l'adaptation de domaine.



(a) Apprentissage inductif



(b) Apprentissage transductif



(c) Adaptation de domaine

FIGURE 1.1 – Cadres d'apprentissage étudiés dans la thèse.

Le texte qui suit présente chacun de ces cadres d'apprentissage. La figure 1.1 en synthétise les aspects principaux. Précisons que ces «paradigmes» ne sont pas mutuellement exclusifs. Les praticiens de l'apprentissage automatique rencontrent parfois des problèmes que l'on peut modéliser à l'aide de plus d'un cadre d'apprentissage. Cela dit, il convient de les étudier séparément pour bien en cerner les particularités.

1.3.1 L'apprentissage inductif

Dans le cadre d'***apprentissage inductif***, on fait l'hypothèse que les exemples d'apprentissage sont générés de manière *indépendante* et qu'ils sont *identiquement distribués* (***i.i.d.***) selon une distribution de probabilité inconnue. Ainsi, l'***échantillon d'entraînement*** consiste en un nombre fini d'observations issues de cette ***distribution génératrice*** de données. Dans ce contexte, un algorithme d'apprentissage reçoit en entrée cet échantillon d'entraînement et retourne un classificateur. Un tel classificateur doit prédire l'étiquette des exemples provenant de la même distribution génératrice, mais qui n'ont pas nécessairement servi au processus d'apprentissage.

Exemple de problème d'apprentissage inductif. Imaginons que nous désirons «entraîner» un ordinateur à reconnaître l'écriture manuscrite. On collecte d'abord des images des 26 lettres de l'alphabet écrites à la main par divers individus. On constitue ainsi un *échantillon d'entraînement*, en prenant soin d'associer chaque image à une des 26 étiquettes possibles (de «A» à «Z»). On exécute ensuite l'algorithme d'apprentissage sur cet échantillon d'entraînement pour obtenir un classificateur. Ce classificateur peut ensuite être utilisé pour reconnaître des caractères écrits par de nouveaux individus.

Par l'hypothèse *i.i.d.*, on suppose que la probabilité que le classificateur rencontre une image précise est la même que la probabilité cette image appartienne à l'échantillon d'entraînement. Ainsi, la population qui «génère» les caractères manuscrits doit demeurer inchangée.

1.3.2 L'apprentissage transductif

Le cadre d'***apprentissage transductif*** correspond à la situation où l'algorithme d'apprentissage a accès à un échantillon de données (de taille finie), appelé ***échantillon complet***, qui contient à la fois des exemples étiquetés et des exemples non étiquetés. Dans ce contexte, l'objectif de l'algorithme est de produire un classificateur dont le rôle est ensuite de prédire l'étiquette des exemples non étiquetés de l'échantillon complet, et uniquement de ceux-là. Cela diffère du cadre d'apprentissage inductif, où le classificateur appris doit posséder un faible risque sur n'importe quel exemple généré par une distribution de données.

Dans le cadre d'apprentissage transductif, nous ne faisons aucune hypothèse sur la distribution qui a généré l'échantillon complet. Cependant, nous considérons que le choix des exemples étiquetés parmi l'échantillon complet est effectué au ***hasard uniforme sans remise***.[1]

Exemple de problème d'apprentissage transductif. Imaginons que nous disposons d'un nombre N (fini) d'articles de journaux à catégoriser par sujet (politique, sport, culture, etc.). Parmi ces N articles, nous en choisissons m aléatoirement et nous les catégorisons à la main. Nous fournissons ensuite à un algorithme d'apprentissage transductif les textes des N articles, accompagnés de leur catégorie pour m d'entre eux, dans le but d'obtenir un classificateur prédisant la catégorie des $N-m$ articles restants.

1.3.3 L'adaptation de domaine

Le cadre d'apprentissage de l'***adaptation de domaine*** se distingue du cadre inductif par le fait l'échantillon de données étiquetées utilisé lors du processus d'entraînement (nommé ***échantillon source***) provient d'un problème différent de celui sur lequel sera utilisé le classificateur. L'algorithme d'apprentissage a seulement accès à un ***échantillon cible*** non éti-

1. Le cadre d'apprentissage transductif étudié dans cette thèse correspond au «cadre transductif de type 1» défini par Vapnik (1998). Nous discutons des distinctions entre le «type 1» et le «type 2» à la section 5.1.

queté provenant du problème d'intérêt. Bien entendu, l'échantillon source et l'échantillon cible doivent provenir de problèmes similaires pour qu'il soit possible de s'adapter de l'un à l'autre.

Exemple de problème d'adaptation de domaine. Imaginons que l'on possède deux échantillons de textes critiques concernant des livres et des films. Chaque critique de livre est accompagnée d'une appréciation (une côte numérique). Cependant, aucune appréciation n'est disponible pour les critiques de films. À partir de ces données, on désire entraîner un classificateur à prédire l'appréciation d'un film à partir de sa critique. Lors de l'apprentissage, il faudra non seulement minimiser l'erreur sur la classification de livres (la *source*), mais aussi «s'adapter» à la tâche de classification de film (la *cible*).

1.4 Contributions de la thèse

La thèse se consacre, à la lumière de la théorie PAC-bayésienne, à l'étude des trois cadres d'apprentissage automatique présentés à la section 1.3 ci-haut : l'apprentissage inductif, l'apprentissage transductif et l'adaptation de domaine. Nous distinguons deux grands volets aux contributions de la thèse, correspondant aux parties II et III du document. Les contributions de chacune de ces deux parties sont résumées dans les deux prochaines sous-sections.

Partie II : Le cadre d'apprentissage inductif revisité

La théorie PAC-bayésienne «classique» étudie le cadre d'apprentissage inductif. La littérature à ce sujet contient de nombreux résultats intéressants. Les contributions de la thèse dans ce domaine ne résident pas dans l'apport de nouveaux résultats algorithmiques ou théoriques (sauf quelques améliorations mineures aux expressions de certaines bornes sur le risque), mais plutôt dans l'originalité de l'approche suggérée. Cette approche, très générale, s'avère une nouvelle conceptualisation de la théorie PAC-bayésienne. Elle permet d'abord de relier entre eux de nombreux théorèmes PAC-bayésiens existants, qui ont été démontrés de manière indépendante lorsqu'ils ont été énoncés pour la première fois (notamment par McAllester, 1999, 2003a; Langford et Seeger, 2001; Catoni, 2007), en les exprimant comme des corollaires d'un même théorème PAC-bayésien général. De manière plus importante, l'approche adoptée fait ressortir clairement les «concepts fondamentaux» de la théorie PAC-bayésienne. C'est grâce à cette compréhension que nous pouvons, dans la suite de la thèse, intervenir sur l'un ou l'autre de ces «concepts» pour adapter la théorie à de nouveaux cadres d'apprentissage.

Plus spécifiquement, les contributions du premier volet de la thèse incluent :

— Une étude des votes de majorité basée sur la *marge* comme variable aléatoire. Cela nous permet d'exprimer la «$\mathcal{C}$-borne» de Lacasse et al. (2006); Lacasse (2010) de manière élégante et intuitive, en faisant ressortir notamment l'importance d'une forme de

désaccord[2] entre les différents votants du vote de majorité.

— La démonstration d'un théorème PAC-bayésien général pour toute *espérance de perte*[3]. Ce théorème fait ressortir le rôle fondamental de la loi binomiale dans le cadre inductif[4], de même que l'influence du choix d'une «fonction de distance» entre le *risque empirique* et l'*erreur de généralisation*. En effet, en variant cette «fonction de distance», nous retrouvons certains des théorèmes PAC-bayésiens les mieux connus dans la littérature.

— L'énonciation du concept de «votant jumelé», à partir duquel nous bornons deux *espérances de perte* d'un même vote de majorité *simultanément* à l'aide d'un théorème PAC-bayésien faisant intervenir une loi de probabilité *trinomiale*. Cette technique originale nous permet de retrouver la borne PAC-bayésienne de la «$\mathcal{C}$-borne».

Partie III : Au-delà du cadre d'apprentissage inductif

En s'appuyant sur l'étude des votes de majorité et de la théorie PAC-bayésienne présentée dans la partie II, nous généralisons la théorie PAC-bayésienne à deux autres cadres d'apprentissage. Cela nous permet d'énoncer de nouvelles bornes PAC-bayésiennes grâce auxquelles nous obtenons des garanties de généralisations et nous concevons de nouveaux algorithmes d'apprentissage.

Contributions au cadre d'apprentissage transductif :

— Démonstration d'un nouveau théorème PAC-bayésien pour l'apprentissage transductif, qui se distingue par la possibilité d'être facilement calculable avec toute «fonction de distance» convexe entre le *risque empirique* et l'*erreur de généralisation*.

— Conception d'une «fonction de distance» pour le cadre d'apprentissage transductif.

— Formulation d'une nouvelle borne PAC-bayésienne sur le risque des classificateurs transductifs qui se révèle beaucoup plus serrée que le résultat existant dans la littérature (due à Derbeko et al., 2004).

— Étude empirique du comportement des nouvelles bornes PAC-bayésiennes transductives obtenues avec différentes «fonctions de distances».

Contributions au cadre de l'adaptation de domaine :

— Définition d'une nouvelle notion de *divergence* entre les distributions source cible qui dépend d'une pondération de votants, ce qui ouvre la voie à la première analyse PAC-bayésienne de l'adaptation de domaine.

2. Plus loin dans la thèse, cette notion de *désaccord* se révèle un élément clé dans la formulation d'une nouvelle mesure de divergence entre distributions pour l'adaptation de domaine (chapitre 6).

3. Sommairement, une *espérance de perte* est une généralisation du risque d'un classificateur (voir la définition 2.2 plus loin).

4. C'est en remplaçant la loi binomiale par la loi hypergéométrique que nous formulons ensuite nos résultats pour le cadre transductif (chapitre 5).

— Formulation d'une nouvelle borne sur le risque cible pour l'adaptation de domaine.

— Conception d'un algorithme d'apprentissage PAC-bayésien pour l'adaptation de domaine, nommé *PBDA*, qui minimise la spécialisation aux *classificateurs linéaires* de la borne sur le risque cible. Nos expérimentations empiriques montrent que PBDA est compétitif avec d'autres algorithmes appartenant à l'état de l'art en adaptation de domaine (Bruzzone et Marconcini, 2010; Chen et al., 2011) sur des données réelles.

1.5 Organisation de la thèse

La thèse compte huit chapitres regroupés en quatre grandes parties, suivis d'une série d'annexes.

Partie I : Mise en contexte

1. La présente «Introduction».
2. Le chapitre «Notions d'apprentissage automatique» présente les notions et définitions générales qui sont évoquées dans la thèse, ainsi que quelques algorithmes d'apprentissage.

Partie II : Le cadre d'apprentissage inductif revisité

3. Dans le chapitre «Étude des votes de majorité», nous posons les bases nécessaires à l'analyse PAC-bayésienne de la thèse. Nous y formulons les notions de *risque de Gibbs* et d'*espérance de désaccord* en fonction de la *marge* du vote de majorité. De même, nous discutons de deux bornes sur le risque du vote de majorité : la «borne du facteur deux» et la «$\mathcal{C}$-borne».
4. Dans chapitre «Théorie PAC-bayésienne pour l'apprentissage inductif», nous présentons la conceptualisation de la théorie PAC-bayésienne propre à cette thèse. À partir d'un théorème très général, nous obtenons plusieurs bornes connues sur le risque de Gibbs. Nous expliquons aussi comment calculer certaines bornes sur le risque du vote de majorité en se basant sur l'étude présentée au chapitre 3.

Partie III : Au-delà du cadre d'apprentissage inductif

5. Dans le chapitre «Théorie PAC-bayésienne pour l'apprentissage transductif», nous présentons un nouveau théorème qui permet de borner le risque sur l'échantillon de données complet dans le cadre transductif. Nous montrons que ce théorème permet d'améliorer considérablement les bornes existantes, et nous proposons une comparaison empirique des différentes déclinaisons des bornes dérivées du théorème.
6. Dans le chapitre «Théorie PAC-bayésienne pour l'adaptation de domaine», nous introduisons la première analyse de l'adaptation de domaine PAC-bayésienne, basée sur une

nouvelle notion de divergence entre les distributions source et cible. Cette analyse nous mène à la conception d'un algorithme d'apprentissage, que nous évaluons empiriquement sur des données jouets et des données réelles.

Partie IV : Perspectives

7. Le chapitre «Aperçu des autres contributions» aborde brièvement les autres travaux auxquels nous avons collaboré au cours du doctorat et qui ne sont pas couverts par les chapitres précédents de la thèse. Ce faisant, nous référons le lecteur aux articles où sont détaillées ces «autres contributions».
8. Le chapitre «Conclusion et travaux futurs» présente une récapitulation concise des contributions de la thèse et discute de certaines perspectives de recherche.

Annexes

A. L'annexe «Résultats mathématiques» contient certains lemmes et théorèmes auxquels nous référons dans le corps de la thèse.
B. L'annexe «Détails des expérimentations» contient des liens vers le code source de certains algorithmes développés au cours de la thèse, l'explication du calcul des bornes PAC-bayésiennes lors des expérimentations avec l'algorithme AdaBoost, ainsi que des résultats supplémentaires obtenus dans le cadre transductif.
C. L'annexe «A PAC-Bayes Sample Compression Approach to Kernel Methods» contient le texte original d'un article publié dans le cadre de la conférence *ICML* en 2011. Nous avons aussi inclus le matériel supplémentaire accompagnant l'article.
D. L'annexe «A Pseudo-Boolean Set Covering Machine» contient le texte original d'un article publié dans le cadre de la conférence *Principles and Practice of Constraint Programming* en 2012.
E. L'annexe «Domain-Adversarial Neural Networks» contient le texte original d'un article soumis à la conférence *ICML* en février 2015 (en cours d'évaluation au moment du dépôt de la thèse).

Chapitre 2

Notions d'apprentissage automatique

Résumé. Ce chapitre présente la plupart des concepts de base nécessaires à l'étude de l'apprentissage automatique d'un point de vue statistique, tout en présentant la notation et la terminologie utilisées tout au long de la thèse. Nous y décrivons aussi sommairement quelques algorithmes d'apprentissage (SVM, SCM et AdaBoost) auxquels il sera fait allusion par la suite dans les chapitres subséquents.

2.1 Le problème de l'apprentissage automatique

Un algorithme d'apprentissage permet d'«entraîner» un système informatique à effectuer une tâche en «observant» une série d'exemples propres à cette tâche.

2.1.1 Les exemples d'apprentissage et les échantillons de données

On représente un ***exemple*** par une paire description-étiquette (x, y). La ***description*** de l'exemple appartient à l'***espace d'entrée*** $\mathcal{X}$ et l'***étiquette*** appartient à l'***espace de sortie*** $\mathcal{Y}$. Ainsi,

$$(x, y) \in \mathcal{X} \times \mathcal{Y}\,.$$

Les espaces $\mathcal{X}$ et $\mathcal{Y}$ peuvent être de natures variées selon le problème étudié. Dans la plupart des situations étudiées empiriquement, la ***description*** d'un exemple est donnée par un vecteur d'attributs à valeurs réelles, c'est-à-dire $\mathcal{X} \subseteq \mathbb{R}^n$, où n correspond au nombre d'attributs décrivant un exemple.[1] Lorsque les expressions mathématiques ne sont valides que pour un espace d'entrée vectoriel, nous représentons les exemples par une variable en caractère gras

1. Souvent, même si les descriptions des exemples ne sont pas *explicitement* des vecteurs (par exemple, des images, des sons ou des textes), il existe des méthodes simples pour convertir ces descriptions en vecteurs de nombres réels. Après tout, une représentation informatique est toujours un vecteur de nombres binaires !

(nous écrirons donc $\mathbf{x} \in \mathbb{R}^n$). Dans cette situation, les exemples possèdent donc la forme

$$(\mathbf{x}, y) \in \mathbb{R}^n \times \mathcal{Y} . \tag{2.1}$$

Un ***échantillon de données*** (aussi appelé *ensemble de données*) est une collection de plusieurs exemples issus de l'observation d'un même phénomène. Dans ce document, nous désignons habituellement un échantillon de données par l'ensemble S :

$$S \coloneqq \{(x_i, y_i)\}_{i=1}^m = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})^m ,$$

où m correspond au nombre d'exemples contenus dans S.

2.1.2 La distribution génératrice des données

Afin d'étudier l'apprentissage d'un point de vue statistique, on représente le «phénomène» qui génère les exemples propres à la tâche étudiée par une distribution de probabilité. Cette ***distribution génératrice*** des données, que nous notons D, est définie sur l'espace $\mathcal{X} \times \mathcal{Y}$. On adopte généralement l'hypothèse que chaque exemple observé est une réalisation *indépendante* et *identiquement distribuée* (***i.i.d.***) d'une variable aléatoire régie par la distribution génératrice des données. Autrement dit, chaque exemple est généré par la même distribution de probabilité D, et la probabilité d'observer cet exemple n'est pas influencée par les observations précédentes.

En acceptant cette hypothèse, on peut affirmer qu'un échantillon de données S est une collection de m réalisations générées *i.i.d.* selon D :

$$S \coloneqq \{(x_i, y_i)\}_{i=1}^m \sim D^m .$$

C'est une hypothèse forte, mais il s'agit souvent de l'unique hypothèse sur laquelle repose la théorie statistique de l'apprentissage automatique telle que présentée dans cette thèse.

2.1.3 Le problème de la classification

Lorsque l'espace de sortie $\mathcal{Y}$ est un ensemble (non ordonné) de valeurs discrètes, le problème d'apprentissage en est un de ***classification***. Un ***classificateur***, que nous désignons par $h(\cdot)$, reçoit en entrée la description d'un exemple et prédit l'étiquette associée à cette description. Ainsi, un classificateur est une fonction de la forme

$$h : \mathcal{X} \to \mathcal{Y} . \tag{2.2}$$

Dans la présente thèse, nous étudions le problème de la ***classification binaire***, où l'espace de sortie $\mathcal{Y}$ contient seulement deux éléments. Par convention, nous représentons les ***étiquettes*** par $\mathcal{Y} \coloneqq \{-1, +1\}$. Nous disons qu'une étiquette est soit négative (-1), soit positive $(+1)$. Dans ce contexte, un ***classificateur binaire*** est une fonction de la forme

$$h : \mathcal{X} \to \{-1, +1\} .$$

Le ***risque*** (ou l'***erreur de généralisation***) $R_D(h)$ d'un classificateur consiste en la probabilité qu'il classifie incorrectement un exemple généré par la distribution de probabilité D :

$$R_D(h) \stackrel{\text{def}}{=} \Pr_{(x,y)\sim D}\big(h(x) \neq y\big) \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D} \mathtt{I}\big[h(x) \neq y\big]\,,$$

où $\mathtt{I}[\,\cdot\,]$ est la ***fonction indicatrice*** : $\mathtt{I}[a] = 1$ si l'énoncé a est vrai ; $\mathtt{I}[a] = 0$ sinon.

Le ***risque empirique*** $R_S(h)$ d'un classificateur sur un échantillon de données $S \sim D^m$, comprenant m exemples, est le ratio d'erreurs sur cet échantillon :

$$R_S(h) \stackrel{\text{def}}{=} \frac{1}{m}\sum_{i=1}^{m} \mathtt{I}\big[h(x_i) \neq y_i\big]\,.$$

Dans la plupart des situations, le problème à résoudre en apprentissage automatique consiste à trouver un classificateur $h(\cdot)$ possédant une faible erreur de généralisation $R_D(h)$, tout en ignorant la nature de la distribution génératrice D. Les observations contenues dans l'échantillon d'entraînement S constituent la principale source d'information (sinon la seule) que l'on possède sur la distribution D.

2.1.4 Le problème de la régression

Dans un problème de régression, l'espace de sortie est constitué de nombres réels (c'est-à-dire $\mathcal{Y} \subseteq \mathbb{R}$). Un ***régresseur*** est une fonction, que nous notons $f(\cdot)$. Il s'agit d'une généralisation du concept de classificateur : un régresseur est une fonction qui reçoit en entrée une description d'exemple et retourne un nombre réel,

$$f : \mathcal{X} \to \overline{\mathcal{Y}}\,, \qquad \text{où} \;\; \overline{\mathcal{Y}} \subseteq \mathbb{R}\,.$$

Le problème de la régression n'est pas directement abordé dans cette thèse. Cependant, nous utilisons des régresseurs à valeur de sortie $\overline{\mathcal{Y}} \coloneqq [-1, +1]$ dans le contexte de la classification binaire, où l'espace de sortie est $\mathcal{Y} \coloneqq \{-1, +1\}$. Plus précisément, considérant un régresseur $f : \mathcal{X} \to [-1, 1]$ et un exemple $(x, y) \in \mathcal{X} \times \{-1, +1\}$, le signe de la sortie du régresseur (noté $\tilde{y}$ ci-bas) est interprété comme la prédiction quant à l'étiquette de l'exemple :

$$\tilde{y} \;=\; \operatorname{sgn}\Big[\, f(x) \,\Big] \;=\; \begin{cases} +1 \text{ si } f(x) > 0\,, \\ -1 \text{ sinon.} \end{cases}$$

De plus, on interprète généralement l'amplitude $|f(x)| \in [0, 1]$ comme une «mesure de confiance» du régresseur envers cette prédiction. Selon cette interprétation, plus $|f(x)|$ est près de 1, plus le régresseur est «confiant» que l'étiquette associée à la description x est $\tilde{y}$. Inversement, si $|f(x)|$ est près de 0, le régresseur est «incertain» quant à sa prédiction.

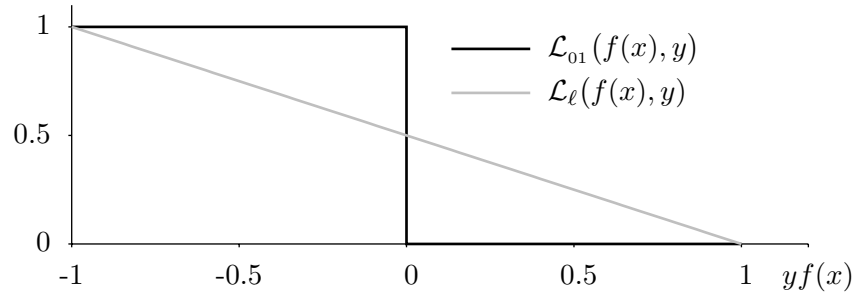


FIGURE 2.1 – Comparaison de la *perte zéro-un* $\mathcal{L}_{01}(f(x), y)$ et de la *perte linéaire* $\mathcal{L}_{\ell}(f(x), y)$, sur un exemple $(x, y) \in \mathcal{X} \times \{-1, +1\}$, en fonction de la valeur $yf(x)$.

2.1.5 Les fonctions de pertes et leur espérance

Lorsque les régresseurs sont utilisés pour résoudre une tâche de classification binaire, le recours à une ***fonction de perte*** $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to \mathbb{R}^+$ permet de quantifier la justesse de la prédiction d'un régresseur. Dans cette situation, nous utilisons fréquemment (mais pas exclusivement) la *fonction de perte zéro-un* et la *fonction de perte linéaire*, dont les valeurs se situent entre 0 et 1, qui sont définies comme suit.

Définition 2.1. Soit un exemple $(x, y) \in \mathcal{X} \times \{-1, +1\}$ et un régresseur $f : \mathcal{X} \to [-1, 1]$.

1. La fonction de ***perte zéro-un*** $\mathcal{L}_{01} : [-1, 1] \times \{-1, 1\} \to [0, 1]$ est définie par

$$\mathcal{L}_{01}(f(x), y) \stackrel{\text{def}}{=} \mathtt{I}\big[\, yf(x) \leq 0 \,\big] = \begin{cases} 1 & \text{si } \; yf(x) \leq 0\,, \\ 0 & \text{sinon.} \end{cases}$$

2. La fonction de ***perte linéaire*** $\mathcal{L}_{\ell} : [-1, 1] \times \{-1, 1\} \to [0, 1]$ est définie par

$$\mathcal{L}_{\ell}(f(x), y) \stackrel{\text{def}}{=} \frac{1}{2}\Big(1 - y \cdot f(x)\Big)\,.$$

La figure 2.1 compare la perte zéro-un et la perte linéaire. On remarque que, contrairement à la fonction de perte zéro-un, la fonction de perte linéaire prend en considération la «confiance» du régresseur envers sa prédiction (c'est-à-dire de l'amplitude $|f(x)|$).

Remarque. Lorsqu'un régresseur s'abstient de prédire une étiquette, c'est-à-dire lorsque $f(x)$ retourne exactement 0, la fonction de perte zéro-un vaut 1. Ceci est une convention adoptée dans ce document, et il y aurait d'autres choix possibles. Par exemple, nous pourrions déclarer que la perte vaut $\frac{1}{2}$ lorsque $f(x) = 0$. Ce choix a une influence négligeable sur les résultats présentés dans cette thèse, car les régresseurs utilisés en pratique ne retournent que très rarement 0.

Notons que les deux fonctions de pertes présentées par la définition 2.1 sont également valides lorsque le régresseur est un classificateur binaire (c'est-à-dire que sa valeur de sortie est soit -1,

soit $+1$). Dans ce cas particulier, la perte zéro-un et la perte linéaire sont équivalentes, car pour tout classificateur $h : \mathcal{X} \to \{-1, +1\}$, on a

$$\mathcal{L}_\ell\big(h(x), y\big) \;=\; \mathcal{L}_{01}\big(h(x), y\big) \qquad \forall\, y \in \{-1, +1\}\,. \tag{2.3}$$

En effet, si $h(x) \neq y$, alors $\mathcal{L}_\ell\big(h(x), y\big) = 1$. Sinon, on a $h(x) = y$ et $\mathcal{L}_\ell\big(h(x), y\big) = 0$.

Définissons maintenant l'espérance de perte d'un régresseur sur un échantillon de données et sur une distribution génératrice.

Définition 2.2. Soit un régresseur $f : \mathcal{X} \to \overline{\mathcal{Y}}$ et une fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0, 1]$.

1. L'***espérance de perte*** sur une distribution génératrice de données D est

$$\mathbb{E}_D^{\mathcal{L}}(f) \;\stackrel{\text{def}}{=}\; \mathop{\mathbf{E}}_{(x,y)\sim D} \mathcal{L}\big(f(x), y\big)\,.$$

2. L'***espérance de perte empirique*** sur un échantillon de données $S \coloneqq \{(x_i, y_i)\}_{i=1}^m$ est

$$\mathbb{E}_S^{\mathcal{L}}(f) \;\stackrel{\text{def}}{=}\; \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}\big(f(x_i), y_i\big)\,.$$

En adoptant ces définitions, on constate que le risque d'un classificateur binaire (qui, rappelons-le, est un cas particulier de régresseur) est équivalent à son espérance de perte zéro-un. En effet, par la constatation faite à l'équation (2.3), on a

$$R_{D'}(h) \;=\; \mathbb{E}_{D'}^{\mathcal{L}_{01}}(h)\,. \tag{2.4}$$

Remarque sur la notation de l'équation (2.4). Dans l'équation précédente, on écrit D' pour signifier que l'expression est à la fois valide pour l'espérance de perte sur la distribution génératrice (auquel cas $D' \coloneqq D$) et pour l'espérance de perte empirique (auquel cas $D' \coloneqq S$). Dans le second cas, nous faisons un léger abus de notation : il faut interpréter la notation $\mathbb{E}_S^{\mathcal{L}}(\cdot)$ comme l'espérance de perte sur un tirage avec remise d'un exemple dans l'échantillon de données S. Autrement dit, en notant $\mathtt{U}[S]$ la distribution uniforme S, on a

$$R_S(h) \;=\; \mathbb{E}_S^{\mathcal{L}_{01}}(h) \;=\; \mathop{\mathbf{E}}_{(x,y)\sim \mathtt{U}[S]} \mathcal{L}\big(f(x), y\big)\,.$$

Nous réutiliserons cette notation raccourcie tout au long de la thèse.

Spécifions aussi que le recours à la notation «$\mathbb{E}$» pour désigner l'espérance de perte, introduite par la définition 2.2, est particulier à ce document. Ce choix de notation permet de simplifier les expressions de la théorie PAC-bayésienne générale introduite au chapitre 4.

2.1.6 Les algorithmes d'apprentissage et le défi de la généralisation

Dans le contexte de la classification (ou de la régression), un **algorithme d'apprentissage** A reçoit en entrée un ***échantillon d'entraînement*** S et construit un classificateur $h(\cdot)$.[2] Ainsi,

$$A_p(S) \longrightarrow h\,,$$

où nous représentons par p les ***hyperparamètres*** de l'algorithme : un tuple de paramètres qu'on doit fixer préalablement l'exécution de l'algorithme et qui conditionne le «comportement» de ce dernier.

Le défi d'un algorithme d'apprentissage est de généraliser l'information contenue dans l'échantillon d'entraînement S afin de produire un classificateur de faible risque $R_D(h)$, c'est-à-dire possédant une faible probabilité de prédire incorrectement l'étiquette d'un exemple généré par la distribution D. Il s'agit d'une tâche difficile, étant donné que cette distribution génératrice est inconnue. Il est donc impossible de connaître la valeur exacte de l'erreur de généralisation $R_D(h)$ d'un classificateur (ou, autrement dit, le «vrai» risque du classificateur).

Il est bien connu chez les praticiens de l'apprentissage automatique qu'un algorithme d'apprentissage automatique ne doit pas uniquement chercher à minimiser le risque empirique $R_S(h)$. En effet, un algorithme misant sur cette stratégie s'expose au problème du ***surapprentissage***, c'est-à-dire que le classificateur retourné par l'algorithme «concentre ses efforts» sur les exemples d'entraînement au risque de nuire à la généralisation. Pour illustrer ce phénomène, considérons un cas extrême où un algorithme d'apprentissage, après avoir observé un échantillon d'entraînement $S \coloneqq \{(x_i, y_i)\}_{i=1}^m \sim D^m$, construit le classificateur $h_{\text{bête}}(\cdot)$ défini comme suit :

$$h_{\text{bête}}(x) \stackrel{\text{def}}{=} \begin{cases} y_i & \text{si } (x_i, y_i) \in S\,, \\ -1 & \text{sinon.} \end{cases}$$

Le risque empirique du classificateur $h_{\text{bête}}(\cdot)$ est nul – c'est-à-dire $R_S(h_{\text{bête}}) = 0$ –, mais son erreur de généralisation $R_D(h_{\text{bête}})$ est vraisemblablement très élevée, car $h_{\text{bête}}(\cdot)$ prédit une étiquette négative (-1) pour tous les exemples qui n'appartiennent pas à l'échantillon d'entraînement S.

Il importe donc d'incorporer aux algorithmes d'apprentissage un (ou plusieurs) mécanisme(s) permettant de généraliser l'information contenue dans les données d'entraînement ; autrement dit, les «aider» à bien «apprendre». Beaucoup d'algorithmes d'apprentissage résolvent un problème d'optimisation qui exprime un compromis entre la minimisation de l'espérance d'une fonction de perte empirique (calculée sur un échantillon d'entraînement) et un certain régularisateur. Sommairement, ce type d'algorithmes s'expriment sous une forme mathématique

2. Par souci de simplicité, cette introduction présente la plupart des concepts à l'aide de classificateurs $h : \mathcal{X} \to \{-1, 1\}$, mais tout est directement généralisable aux régresseurs $f : \mathcal{X} \to [-1, 1]$.

semblable à

$$A_p(S) \longrightarrow \underset{h \in \mathcal{H}}{\operatorname{argmin}} \Bigg[\underbrace{\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\big(h(x_i), y_i\big)}_{\textbf{\textit{espérance de perte}}} + \underbrace{\operatorname{Reg}_{(p)}\Big(h\Big)}_{\textbf{\textit{régularisateur}}} \Bigg], \tag{2.5}$$

où $\mathcal{H}$ est la famille de tous les classificateurs que peut construire l'algorithme d'apprentissage [3], et où $\operatorname{Reg}_{(p)}(h)$ représente une fonction de régularisation dont l'expression peut dépendre des hyperparamètres p. La fonction de régularisation a généralement pour rôle de limiter la «complexité» du classificateur $h(\cdot)$ choisi par l'algorithme d'apprentissage.

L'algorithme ***SVM*** (décrit à la section 2.2 ci-bas) s'exprime comme un problème d'optimisation de la forme présentée par l'équation (2.5). D'autres algorithmes d'apprentissage incorporent un processus de régularisation de manière plus implicite. C'est le cas de l'algorithme ***SCM*** (section 2.3), qui limite la complexité du classificateur en s'assurant que celui-ci est exprimable avec un petit sous-ensemble des exemples d'entraînement. Quant à l'algorithme ***AdaBoost*** (section 2.4.2), il minimise une espérance de perte empirique sans terme de régularisation, mais il évite le surapprentissage en construisant un vote de majorité de classificateurs «faibles» et en fixant un nombre d'itérations maximal à la procédure d'optimisation. Notons qu'une stratégie similaire est fréquemment utilisée dans divers algorithmes d'apprentissage appartenant à la famille des ***réseaux de neurones*** (abordés dans l'article présenté à l'annexe E). Cette stratégie consiste à minimiser, par descente de gradient, une fonction de coût sur l'ensemble d'entraînement, et d'arrêter la procédure d'optimisation lorsque le risque calculé sur un échantillon de données distinct (nommé l'*ensemble de validation*) est minimal.

2.2 Les classificateurs linéaires, l'algorithme SVM et l'astuce du noyau

En considérant que chaque exemple d'apprentissage s'avère un point dans un espace d'entrée vectoriel $\mathcal{X} := \mathbb{R}^n$ – voir l'équation (2.1) –, un classificateur peut être représenté comme une frontière de décision dans ce même espace. En apprentissage automatique, il est fréquent de recourir à un ***classificateur linéaire***, c'est-à-dire une frontière de décision qui consiste en un hyperplan séparant l'espace $\mathbb{R}^n$ en deux régions distinctes, tel qu'illustré par la figure 2.2.

Un classificateur linéaire $h_{\mathbf{w}} : \mathbb{R}^n \to \{-1, +1\}$ est représenté par un vecteur $\mathbf{w} \in \mathbb{R}^n$ orthogonal à la frontière de décision. Étant donné un exemple $(\mathbf{x}, y) \in \mathbb{R}^n \times \{-1, +1\}$, la prédiction $h_{\mathbf{w}}(\mathbf{x})$ est donnée par le signe du ***produit scalaire*** entre les vecteurs $\mathbf{w}$ et $\mathbf{x}$:

$$h_{\mathbf{w}}(\mathbf{x}) \overset{\text{def}}{=} \operatorname{sgn}\big[\, \mathbf{w} \cdot \mathbf{x} \,\big]. \tag{2.6}$$

3. Cet ensemble (fini ou infini) $\mathcal{H}$ est souvent nommé la *classe d'hypothèses* dans la littérature.

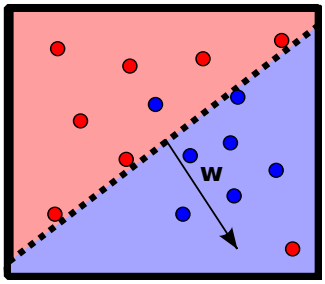


FIGURE 2.2 – Exemple de classificateur linéaire sur un ensemble jouet à deux dimensions. Les points bleus représentent des exemples positifs et les points rouges des exemples négatifs (chaque exemple est donc décrit par un vecteur dans $\mathbb{R}^2$). Le vecteur $\mathbf{w}$ est orthogonal à l'hyperplan correspondant à la frontière de décision (représentée par la ligne pointillée).

La quantité $y\,\mathbf{w}\cdot\mathbf{x}$ est nommée la ***marge fonctionnelle*** du classificateur sur cet exemple [4] ; elle est positive si l'exemple est bien classifié (c'est-à-dire lorsque $h_{\mathbf{w}}(\mathbf{x}) = y$), et négative sinon (lorsque $h_{\mathbf{w}}(\mathbf{x}) \neq y$). De même, la quantité $y\frac{\mathbf{w}\cdot\mathbf{x}}{\|\mathbf{w}\|}$ correspond à la ***marge géométrique*** du classificateur sur l'exemple. La valeur absolue de la marge géométrique correspond à la distance euclidienne dans l'espace $\mathbb{R}^n$ entre le point $\mathbf{x}$ et l'hyperplan décrit par le vecteur $\mathbf{w}$.

Un échantillon de données S est dit ***linéairement séparable*** s'il existe un classificateur linéaire qui classifie correctement tous les exemples de S.

Remarque. L'hyperplan décrit par le classificateur linéaire de l'équation (2.6) passe nécessairement par l'origine de l'espace vectoriel $\mathbb{R}^n$. Afin de se départir de cette contrainte, il est commun d'introduire un biais $b \in \mathbb{R}$ dans l'équation du classificateur linéaire, qui devient alors

$$h_{\mathbf{w},b}(\mathbf{x}) \stackrel{\text{def}}{=} \operatorname{sgn}[\,\mathbf{w}\cdot\mathbf{x} + b\,]\,.$$

La frontière de décision d'un classificateur linéaire *avec biais* consiste en un hyperplan dans l'espace vectoriel $\mathbb{R}^n$ qui ne passe pas nécessairement à l'origine.

On peut représenter un classificateur *avec biais* $h_{\mathbf{w},b}(\cdot)$ par un classificateur linéaire *sans biais* $h_{\mathbf{w}'}(\cdot)$ en «augmentant» l'espace d'entrée d'une dimension. En effet, en posant

$$\mathbf{x}' \coloneqq (\mathbf{x}, 1) \in \mathbb{R}^{n+1} \quad \text{et} \quad \mathbf{w}' \coloneqq (\mathbf{w}, b) \in \mathbb{R}^{n+1}\,,$$

on a

$$h_{\mathbf{w}'}(\mathbf{x}') = \operatorname{sgn}[\,\mathbf{w}'\cdot\mathbf{x}'] = \operatorname{sgn}[\,(\mathbf{w},b)\cdot(\mathbf{x},1)] = \operatorname{sgn}[\,\mathbf{w}\cdot\mathbf{x}+b\,] = h_{\mathbf{w},b}(\mathbf{x})\,.$$

Pour cette raison, dans la suite de ce document, nous présentons les équations sans biais lorsqu'il est question de classificateurs linéaires. Ce choix nous permet d'alléger la notation et de mettre l'accent sur les concepts fondamentaux.

4. Notons que la *marge géométrique* et la *marge fonctionnelle* d'un classificateur linéaire sont des concepts différents de la *marge d'un vote de majorité* qui est étudié en profondeur au chapitre 3, bien qu'il existe des liens entre ces trois quantités et que l'interprétation que l'on en fait habituellement est similaire.

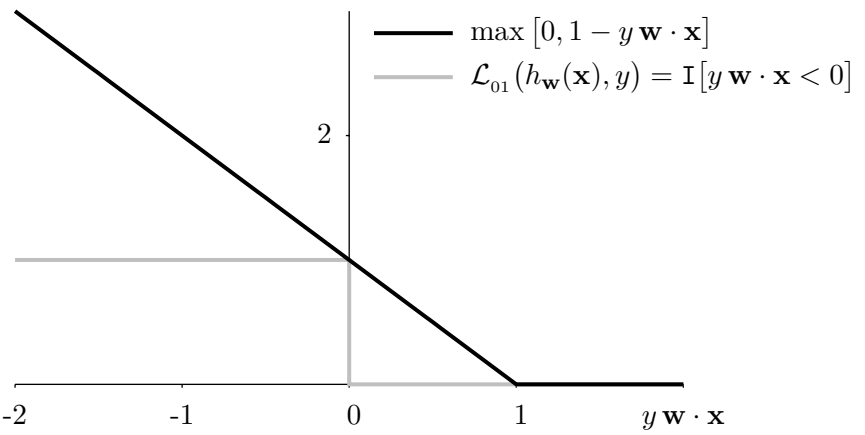


FIGURE 2.3 – Comparaison de la *perte hinge* et de la *perte zéro-un* d'un classificateur linéaire $h_{\mathbf{w}}(\cdot)$ en termes de sa *marge fonctionnelle* $y\,\mathbf{w}\cdot\mathbf{x}$.

2.2.1 Le SVM dans l'espace des exemples.

Plusieurs algorithmes d'apprentissage construisent des classificateurs linéaires. C'est le cas des *Machines à vecteurs de support* de Boser et al. (1992); Cortes et Vapnik (1995), que l'on désigne habituellement par ***SVM*** (de l'anglais *Support Vector Machines*). Étant donné un échantillon d'entraînement $S \coloneqq \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$ de m exemples et un hyperparamètre $C \in \mathbb{R}^+$, le SVM résout un problème d'optimisation afin de trouver le vecteur $\mathbf{w} \in \mathbb{R}^n$ qui minimise la fonction de coût suivante :

$$C \sum_{i=1}^{m} \underbrace{\max\left[0,\, 1 - y_i\,\mathbf{w}\cdot\mathbf{x}_i\right]}_{\textbf{\textit{perte hinge}}} + \underbrace{\frac{1}{2}\|\mathbf{w}\|^2}_{\textbf{\textit{régularisateur}}\ \ell_2}. \tag{2.7}$$

On nomme la fonction de perte du SVM la *perte hinge* (ou, en anglais, *hinge loss*). Comme illustrée par la figure 2.3, la perte hinge repose sur la *marge fonctionnelle* du classificateur linéaire et exprime une relaxation convexe de la perte zéro-un.

Lorsque l'échantillon d'entraînement est *linéairement séparable*, minimiser la perte hinge sans tenir compte du régularisateur – ce qui équivaut à choisir une valeur de C très élevée dans l'équation (2.7) – permet de trouver le classificateur linéaire $h_{\mathbf{w}^*}(\cdot)$ correspondant à l'hyperplan $\mathbf{w}^*$ de *marge géométrique* maximale. Cet hyperplan $\mathbf{w}^*$ est unique et il est donné par

$$\mathbf{w}^* \coloneqq \operatorname*{argmax}_{\mathbf{w}\in\mathbb{R}^n} \left[\min_{(\mathbf{x}_i, y_i)\in S} \left(\frac{y_i\,\mathbf{w}\cdot\mathbf{x}_i}{\|\mathbf{w}\|} \right) \right]. \tag{2.8}$$

Autrement dit, de tous les classificateurs linéaires possédant un risque empirique nul sur l'échantillon d'entraînement S, le classificateur $h_{\mathbf{w}^*}(\cdot)$ correspondant à l'équation (2.8) est celui dont la distance euclidienne des exemples les plus près de la frontière de décision (d'où le «min») est maximale (d'où le «argmax»). Notons que l'équation (2.8) correspond au problème

résolu par l'algorithme d'apprentissage du *SVM à marge rigide* initialement énoncé par Boser et al. (1992).

Dans l'équation (2.7), l'hyperparamètre C détermine un compromis entre la minimisation de l'espérance de perte empirique – ce qui équivaut à la recherche de l'hyperplan de marge géométrique maximale – et la minimisation du régularisateur $\|\mathbf{w}\|^2$. C'est pourquoi l'algorithme d'apprentissage correspondant est nommé le *SVM à marge floue* (Cortes et Vapnik, 1995). La ***norme euclidienne*** (aussi appelée la *norme* ℓ_2) du vecteur $\mathbf{w}$ est un régularisateur fréquemment utilisé dans les algorithmes d'apprentissage afin de s'assurer que l'algorithme ne «surapprend» pas les données.

Il est possible d'exprimer le problème d'optimisation du SVM, c'est-à-dire la minimisation de l'équation (2.7), sous la forme d'un problème quadratique. La solution trouvée par le SVM est donc unique (pour un échantillon d'entraînement S donné et un hyperparamètre C fixe). Les bons résultats empiriques et la rapidité d'exécution des algorithmes résolvant ce problème en font un algorithme d'apprentissage très populaire – comme en témoigne notamment les travaux de Platt (1999); Joachims (1999); Bordes et al. (2005); Chang et Lin (2011) – surtout lorsque le SVM est jumelé avec l'astuce du noyau décrite dans la suite de ce texte.

2.2.2 L'astuce du noyau

Les classificateurs linéaires dans l'espace d'entrée ne conviennent pas à tous les problèmes d'apprentissage. [5] L'***astuce du noyau*** est une méthode grandement utilisée afin d'exprimer des frontières de décisions plus complexes qu'un simple hyperplan en les exprimant comme des classificateurs linéaires dans un espace «augmenté». Cette astuce permet aussi de définir des classificateurs alors que l'espace d'entrée $\mathcal{X}$ n'est pas un espace vectoriel. Par exemple, la description d'un exemple peut prendre la forme d'un graphe, d'une chaîne de caractères, d'une image, etc. (voir Shawe-Taylor et Cristianini, 2004).

Remarque sur la présentation des concepts. Nous ne présentons pas ici tous les détails et interprétations liés à la théorie des noyaux. [6] Nous proposons plutôt un aperçu intuitif des concepts auxquels nous ferons appel par la suite, spécialement lors de la conception d'un algorithme d'apprentissage pour l'adaptation de domaine au chapitre 6.

Introduction aux variables duales. Pour expliquer l'astuce du noyau, considérons en premier lieu un problème de classification binaire dont l'espace d'entrée est constitué de vecteurs à valeurs réelles. Ainsi, $\mathcal{X} \coloneqq \mathbb{R}^n$ et $\mathcal{Y} \coloneqq \{-1, +1\}$. Voyons comment exprimer

5. Par exemple, un classificateur linéaire ne peut classifier correctement le problème du «ou exclusif», correspondant à l'échantillon de données $\{(\,(-1,1),\,1\,),\,(\,(1,1),\,-1\,),\,(\,(-1,-1),\,-1\,),\,(\,(1,-1),\,1\,)\} \subset \mathbb{R}^2 \times \{-1,1\}$.

6. Pour un exposé plus approfondi, nous conseillons les livres de référence de Shawe-Taylor et Cristianini (2004); Mohri et al. (2012).

un vecteur $\mathbf{w} \in \mathbb{R}^n$, caractérisant un classificateur linéaire $h_{\mathbf{w}} : \mathbb{R}^n \to \{-1, +1\}$, par une combinaison linéaire des exemples d'apprentissage.

Considérons un échantillon de données $S' \in (\mathbb{R}^n \times \{-1, 1\})^m$ contenant m exemples. Supposons pour l'instant que les exemples de S' sont décrits par des vecteurs qui engendrent l'espace $\mathbb{R}^n$. Autrement dit, tout vecteur de $\mathbb{R}^n$ peut être exprimé comme une combinaison linéaire des vecteurs $\{\mathbf{x}_i\}_{i=1}^m$. En particulier, nous pouvons représenter le vecteur $\mathbf{w}$ par un vecteur de ***variables duales*** $\boldsymbol{\alpha} \coloneqq (\alpha_1, \alpha_2, \ldots, \alpha_m) \in \mathbb{R}^m$:

$$\mathbf{w} \;=\; \sum_{i=1}^{m} \alpha_i \, \mathbf{x}_i \,. \tag{2.9}$$

Cette représentation permet de réécrire la fonction de prédiction du classificateur linéaire $h_{\mathbf{w}}(\cdot)$ – équation (2.6) – en termes des variables duales $\boldsymbol{\alpha}$ et des exemples $\{\mathbf{x}_i\}_{i=1}^m$. Ainsi,

$$h_{\mathbf{w}}(\mathbf{x}) \;=\; \operatorname{sgn}\left[\left(\sum_{i=1}^{m} \alpha_i \, \mathbf{x}_i\right) \cdot \mathbf{x}\right] \;=\; \operatorname{sgn}\left[\sum_{i=1}^{m} \alpha_i \mathbf{x}_i \cdot \mathbf{x}\right] \;=\; \operatorname{sgn}\left[\sum_{i=1}^{m} \alpha_i \, \langle \mathbf{x}_i, \mathbf{x} \rangle_{\mathbb{R}^n}\right], \tag{2.10}$$

où la dernière égalité introduit un simple changement de notation ; on note $\langle \cdot, \cdot \rangle_{\mathbb{R}^n}$ le ***produit scalaire*** dans l'espace vectoriel $\mathbb{R}^n$.

Produit scalaire dans un espace «augmenté». Considérons maintenant un nouveau problème de classification binaire, où l'espace d'entrée $\mathcal{X}$ n'est pas limité aux vecteurs de valeurs réelles (ainsi, dans le texte qui suit, les descriptions «$x \in \mathcal{X}$» ne sont plus représentées par des variables en caractères gras). Cela dit, nous considérons toujours que l'espace de sortie $\mathcal{Y} \coloneqq \{-1, +1\}$ est binaire.

Considérons un échantillon de données $S \in (\mathcal{X} \times \{-1, 1\})^m$ comprenant m exemples, ainsi qu'une fonction $\phi : \mathcal{X} \to \mathcal{Z}$ qui transforme un exemple $x \in \mathcal{X}$ en un élément $\phi(x) \in \mathcal{Z}$ d'un nouvel espace «augmenté». Rappelons que, contrairement aux équations (2.9) et (2.10), l'espace d'entrée $\mathcal{X}$ n'est pas nécessairement un espace vectoriel. Il en va de même pour l'espace «augmenté» $\mathcal{Z}$, pourvu que le produit scalaire $\langle \cdot, \cdot \rangle_{\mathcal{Z}}$ soit défini.

Prenons un élément $w \in \mathcal{Z}$ qui admet une représentation à partir des exemples transformés $\{\phi(x_i)\}_{i=1}^m$ et d'un vecteur de *variables duales* $\boldsymbol{\alpha} \coloneqq (\alpha_1, \alpha_2, \ldots, \alpha_m) \in \mathbb{R}^m$, tel que

$$w \;=\; \sum_{i=1}^{m} \alpha_i \, \phi(x_i) \,. \tag{2.11}$$

Le w considéré ici n'est pas n'importe quel élément de l'espace «augmenté» $\mathcal{Z}$: il doit s'exprimer comme une combinaison linéaire des éléments $\{\phi(x_i)\}_{i=1}^m$. Nous verrons, au terme de cette section, que le *théorème du représentant* garantit que les éléments de $\mathcal{Z}$ qui nous intéressent possèdent cette propriété (lorsque nous travaillons avec certains algorithmes d'apprentissage).

Le produit scalaire étant (par hypothèse) défini dans l'espace $\mathcal{Z}$, l'élément w de l'équation (2.11) définit un classificateur linéaire $h_w : \mathcal{Z} \to \{-1, 1\}$ dans cet espace.[7] Par la méthode employée à l'équation (2.10), ce classificateur linéaire s'exprime par les variables duales $\boldsymbol{\alpha} \in \mathbb{R}^m$:

$$\begin{aligned} h_w(\,\phi(x)\,) &= \operatorname{sgn}\Big[\ \langle w, \phi(x)\rangle_{\mathcal{Z}}\ \Big] \\ &= \operatorname{sgn}\left[\left\langle \left(\sum_{i=1}^{m} \alpha_i \phi(x_i)\right),\ \phi(x)\right\rangle_{\mathcal{Z}}\right] \\ &= \operatorname{sgn}\left[\sum_{i=1}^{m} \alpha_i \left\langle \phi(x_i), \phi(x)\right\rangle_{\mathcal{Z}}\right]. \end{aligned} \tag{2.12}$$

On constate que, pour calculer la prédiction $h_w(x)$, il n'est pas nécessaire de connaître ni la nature de l'espace $\mathcal{Z}$, ni celle de la fonction $\phi(\cdot)$ associée. En effet, il suffit de pouvoir calculer le produit scalaire $\langle \phi(x_i), \phi(x)\rangle_{\mathcal{Z}}$.

Fonctions et matrices de noyaux. On désigne par le terme ***fonction de noyau*** une fonction $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ qui exprime un produit scalaire entre la description de deux exemples dans un certain espace $\mathcal{Z}$. À partir de l'équation (2.12), avec $k(x, x') \coloneqq \langle \phi(x), \phi(x')\rangle_{\mathcal{Z}}$, on exprime le classificateur $h_w : \mathcal{Z} \to \{-1, 1\}$ par le classificateur dual $h_{\boldsymbol{\alpha}} : \mathcal{X} \to \{-1, 1\}$:

$$h_{\boldsymbol{\alpha}}(x) \ \stackrel{\text{def}}{=}\ \operatorname{sgn}\left[\sum_{i=1}^{m} \alpha_i k(x_i, x)\right]. \tag{2.13}$$

Pour un échantillon de données $S \coloneqq \{(x_i, y_i)\}_{i=1}^{m}$, la ***matrice de noyau*** K, de taille $m \times m$, contient les valeurs de la fonction de noyau pour chaque paire d'exemples appartenant à S :

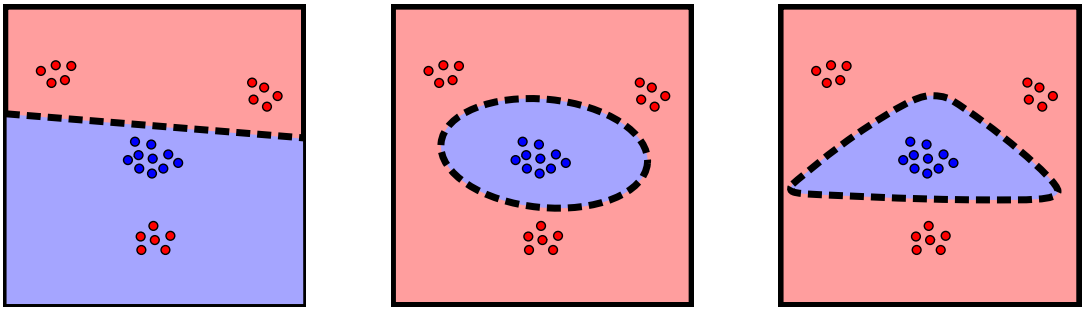
$$K_{ij} \ \stackrel{\text{def}}{=}\ k(x_i, x_j) \quad \text{pour } i, j \in \{1, \ldots, m\}. \tag{2.14}$$

La matrice K est une ***matrice semi-définie positive*** lorsque pour tout $\mathbf{z} \in \mathbb{R}^m$ on a $\mathbf{z}^T K \mathbf{z} \geq 0$. Autrement dit,

$$\sum_{i=1}^{m}\sum_{j=1}^{m} z_i z_j K_{ij} \geq 0, \quad \forall z_1, \ldots, z_m \in \mathbb{R}. \tag{2.15}$$

Une fonction de noyau dont la matrice de similarité associée est toujours semi-définie positive est nommée un ***noyau semi-défini positif*** (certains auteurs parlent aussi de *noyau de Mercer*). Autrement dit, le noyau $k(\cdot, \cdot)$ est noyau semi-défini positif lorsque, pour tout échantillon de données $S \in (\mathcal{X} \times \mathcal{Y})^m$, la matrice de similarité K correspondante respecte l'équation (2.15).

7. Lorsque $\mathcal{X} \subseteq \mathbb{R}^n$, l'espace $\mathcal{Z}$ peut généralement être exprimé comme un espace vectoriel $\mathbb{R}^{n'}$, avec $n' \gg n$. Notons que Schölkopf et al. (2001) présente une définition plus générale où $\mathcal{Z}$ est un ensemble de fonctions $\mathcal{X} \to \mathbb{R}$, que nous reprenons par le théorème du représentant présenté en annexe (théorème A.9).



(a) Noyau linéaire
$k(\mathbf{x}_i, \mathbf{x}_j) \coloneqq \mathbf{x}_i \cdot \mathbf{x}_j$

(b) Noyau polynomial (degré 2)
$k(\mathbf{x}_i, \mathbf{x}_j) \coloneqq (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^2$

(c) Noyau RBF
$k(\mathbf{x}_i, \mathbf{x}_j) \coloneqq e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|}$

FIGURE 2.4 – Frontières de décision, dans l'espace d'entrée $\mathcal{X}$, générée à l'aide de différentes fonctions de noyaux sur un échantillon de données à deux dimensions ($\mathcal{X} \coloneqq \mathbb{R}^2$). Chacune de ces frontières de décisions s'exprime comme un hyperplan dans l'espace «augmenté» (noté $\mathcal{Z}$ dans ce document) représenté *implicitement* par la fonction de noyau. Remarquons que le noyau linéaire exprime un hyperplan dans l'espace des exemples, au même titre que le classificateur linéaire de l'équation (2.6). Le noyau RBF est paramétré par une constante $\gamma > 0$.

Il est montré dans Schölkopf et al. (2001) qu'un noyau $k(\cdot, \cdot)$ noyau semi-défini positif correspond nécessairement à un produit scalaire dans un «certain» espace. Ainsi, une fonction de noyau effectue *implicitement* le produit scalaire entre deux exemples dans l'espace $\mathcal{Z}$.

La figure 2.4 illustre quelques exemples de frontières de décisions obtenues avec des fonctions de noyaux fréquemment utilisées dans la littérature.

Le théorème du représentant. Plusieurs algorithmes d'apprentissage construisant un classificateur linéaire $h_{\mathbf{w}}(\cdot)$ – dont le SVM décrit par l'équation (2.7) – résolvent un problème d'optimisation qui s'exprime comme une combinaison linéaire d'une fonction de perte dépendant de la marge (géométrique ou fonctionnelle) du classificateur linéaire et d'un régularisateur dépendant de la norme du vecteur $\mathbf{w}$. Dans plusieurs cas, il est possible de formuler ce problème d'optimisation en référant au vecteur $\mathbf{w}$ par des opérations de ***produit scalaire***. Puisque la fonction de noyau $k(\cdot, \cdot)$ représente justement un produit scalaire dans un espace «augmenté» $\mathcal{Z}$, ledit problème d'optimisation peut s'exprimer en termes des variables duales $\boldsymbol{\alpha}$. Ce faisant, chaque occurrence du produit scalaire est remplacée par un appel à la fonction de noyau et le problème d'optimisation obtenu ne contient aucune référence *explicite* au vecteur $\mathbf{w}$.

À titre d'exemple, considérons un noyau $k(\cdot, \cdot)$ associé à un espace «augmenté» $\mathcal{Z}$ et un échantillon d'entraînement $S \coloneqq \{(x_i, y_i)\}_{i=1}^m$. La ***marge fonctionnelle*** sur un exemple (x, y) du classificateur $h_{\boldsymbol{\alpha}}(\cdot)$ – lequel est exprimé par l'équation (2.13) et correspond au classificateur dual $h_w(\phi(\cdot))$ de l'équation (2.12) – peut s'écrire ainsi (nous reproduisons une démarche très

similaire à celle employée pour transformer l'équation (2.13) à l'équation (2.12)) :

$$\begin{aligned} y \langle w, \phi(x) \rangle_{\mathcal{Z}} &= y \sum_{i=1}^{m} \alpha_i \langle \phi(x_i), \phi(x) \rangle_{\mathcal{Z}} \\ &= y \sum_{i=1}^{m} \alpha_i k(x_i, x) \,. \end{aligned} \tag{2.16}$$

De même, la norme de w dans l'espace $\mathcal{Z}$ correspond à

$$\begin{aligned} \|w\| = \sqrt{\langle w, w \rangle_{\mathcal{Z}}} &= \sqrt{\Big\langle \sum_{i=1}^{m} \alpha_i \, \phi(x_i), \sum_{j=1}^{m} \alpha_j \, \phi(x_h) \Big\rangle_{\mathcal{Z}}} \\ &= \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{Z}}} \\ &= \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j k(x_i, x_j)} \,. \end{aligned} \tag{2.17}$$

Il est donc possible, en s'inspirant des deux équations ci-dessus, de réécrire plusieurs problèmes d'optimisations impliquant un classificateur linéaire en termes d'une fonction de noyau et de variables duales. Cependant, contrairement à ce que nous avons supposé depuis l'équation (2.11), les éléments $\{\phi(x_i)\}_{i=1}^{m}$ qui sont manipulés implicitement par le noyau $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{Z}}$ ne génèrent possiblement qu'un sous-ensemble de l'espace $\mathcal{Z}$. Autrement dit, l'équation (2.13) ne permet pas, en ajustant les variables duales $\boldsymbol{\alpha}$, d'exprimer tous les classificateurs linéaires dans l'espace $\mathcal{Z}$. Malgré cela, le ***théorème du représentant*** (théorème A.9, en annexe) nous assure que, si un algorithme d'apprentissage minimise une fonction objectif respectant certaines conditions – voir les équations (A.1) et (A.2) en annexe –, alors la solution du problème d'optimisation admet une représentation en termes du noyau et des variables duales. Le recours à l'astuce du noyau ne requiert donc pas la connaissance de l'espace «augmenté» $\mathcal{Z}$ associé à la fonction de noyau $k(\cdot, \cdot)$; il est suffisant de savoir que ce noyau est semi-défini positif – voir l'équation (2.15) – et que la fonction objectif possède la forme appropriée – dictée par le *théorème du représentant* – pour garantir que l'algorithme d'apprentissage, tirant profit de l'astuce du noyau, converge vers la solution optimale.

Précisons que le théorème A.9 (en annexe) correspond au le théorème du représentant dans la forme présentée par Schölkopf et al. (2001). Ce théorème s'applique notamment au SVM, mais aussi à une famille d'algorithmes d'apprentissage utilisant des noyaux pour construire un classificateur linéaire exprimé à l'aide de variables duales. Il s'agit d'un résultat très «puissant» : il permet d'utiliser des algorithmes d'apprentissage construisant des classificateurs linéaires, en apparence plutôt limités, pour construire des frontières de décision complexes.

2.2.3 Le SVM dans l'espace des noyaux

Montrons maintenant comment adapter le problème d'optimisation du ***SVM*** pour tirer profit de l'***astuce du noyau***.

Considérons un échantillon d'entraînement $S \coloneqq \{(x_i, y_i)\}_{i=1}^m$ et un ***noyau semi-défini positif*** $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Grâce aux équations (2.16) et (2.17), nous transformons la fonction objectif du SVM – équation (2.7), page 21 – comme suit :

$$\begin{aligned} & C \sum_{i=1}^{m} \max\Big[0,\, 1 - y_i \left\langle w, \phi(x_i) \right\rangle_{\mathcal{Z}} \Big] + \frac{1}{2} \left\langle w, w \right\rangle_{\mathcal{Z}} \\ & \quad = \quad C \sum_{i=1}^{m} \max\Big[0,\, 1 - y_i \sum_{j=1}^{m} \alpha_j k(x_j, x_i)\Big] + \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \\ & \quad = \quad C \sum_{i=1}^{m} \max\Big[0,\, 1 - y_i \sum_{j=1}^{m} \alpha_j K_{ji}\Big] + \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j K_{ji}\,, \end{aligned} \tag{2.18}$$

où K est la matrice de noyau telle que présentée par l'équation (2.14).

Le ***théorème du représentant*** (théorème A.9), présenté en annexe et décrit à la sous-section précédente, nous assure que le minimum du problème d'optimisation posé par l'équation (2.18) coïncide avec le minimum de la fonction d'optimisation correspondante dans l'espace «augmenté» $\mathcal{Z}$. Lorsqu'employé avec une fonction de noyau, l'algorithme du SVM consiste donc à trouver le vecteur de ***variables duales*** $\boldsymbol{\alpha} \in \mathbb{R}^m$ qui minimise l'équation (2.18). Ce faisant, on obtient un classificateur $h_{\boldsymbol{\alpha}}(\cdot)$ dont la prédiction est calculée à l'aide de l'équation (2.13).

2.3 L'apprentissage par compression d'échantillons et l'algorithme SCM

Considérons un classificateur obtenu par l'exécution d'un algorithme d'apprentissage sur l'échantillon d'entraînement S contenant m exemples. La théorie de la ***compression d'échantillons*** (Floyd et Warmuth, 1995) s'intéresse aux classificateurs qu'il est possible d'exprimer par un sous-ensemble $S_{\mathbf{i}}$ de l'échantillon d'entraînement et un message σ contenant de l'information supplémentaire.

L'ensemble $S_{\mathbf{i}}$ est appelé ***ensemble de compression*** et le vecteur $\mathbf{i}$ réfère aux indices des exemples de l'échantillon d'entraînement S. De plus, on désigne par I l'ensemble des 2^m vecteurs d'indices possibles. Ainsi,

$$\begin{aligned} \mathbf{i} &\stackrel{\text{def}}{=} \langle i_1, i_2, \ldots, i_m \rangle \in I \qquad \text{avec } 1 \leq i_1 < i_2 < \ldots < i_{|\mathbf{i}|} \leq m\,, \\ S_{\mathbf{i}} &\stackrel{\text{def}}{=} \Big\{(x_{i_1}, y_{i_1}), (x_{i_2}, y_{i_2}), \ldots, (x_{i_{|\mathbf{i}|}}, y_{i_{|\mathbf{i}|}})\Big\} \subseteq S\,. \end{aligned}$$

Le ***message*** σ est choisi parmi un ensemble $\Sigma_{\mathbf{i}}$ prédéterminé selon les messages pouvant être associés à l'ensemble $S_{\mathbf{i}}$:

$$\sigma \ \in \ \Sigma_{\mathbf{i}} \, .$$

Le terme ***classificateur comprimé*** désigne un classificateur $h^{\sigma}_{\mathbf{i}} : \mathcal{X} \to \mathcal{Y}$ qu'il est possible d'exprimer uniquement à partir de l'ensemble de compression $S_{\mathbf{i}}$ et le message σ. On suppose donc qu'il existe une ***fonction de reconstruction*** $\mathcal{R}(\cdot, \cdot)$ telle que

$$h^{\sigma}_{\mathbf{i}} \ = \ \mathcal{R}(S_{\mathbf{i}}, \sigma) \, .$$

Comme expliqué dans les prochaines sous-sections, la recherche d'un classificateur exprimable à l'aide d'un petit ensemble de compression est une méthode de régularisation menant à des algorithmes d'apprentissage efficaces. De plus, il existe des contextes où il est souhaitable d'exprimer un classificateur à l'aide de seulement quelques exemples représentatifs du problème d'apprentissage. En effet, cette caractéristique peut notamment encourager l'adoption d'algorithmes d'apprentissage par des chercheurs d'autres domaines qui désirent comprendre la règle de classification déduite par l'algorithme, car un classificateur comprimé est généralement interprétable par un individu non initié aux techniques sophistiquées propres à l'apprentissage automatique.

2.3.1 Un théorème pour l'apprentissage par compression d'échantillons

Le théorème 2.3, énoncé par Marchand et Sokolova (2005) et présenté ci-bas, exprime une borne supérieure sur le risque d'un classificateur comprimé. Cette garantie de généralisation dépend principalement de la taille de l'ensemble de compression $|\mathbf{i}|$ et du nombre d'erreurs k du classificateur sur les exemples n'appartenant pas à l'ensemble de compression. Le théorème 2.3 requiert aussi une distribution $P_{\Sigma_{\mathbf{i}}}$ sur l'ensemble des messages possibles, telle que

$$\sum_{\sigma \in \Sigma_{\mathbf{i}}} P_{\Sigma_{\mathbf{i}}}(\sigma) \ \leq \ 1 \, .$$

Lors de l'application du théorème, cette distribution permet d'accorder plus d'importance aux messages envers lesquels on a confiance. Typiquement, on définit la distribution $P_{\Sigma_{\mathbf{i}}}$ en fonction de la longueur des messages, accordant davantage de poids aux messages courts.

Théorème 2.3 (Marchand et Sokolova, 2005)**.** *Pour toute fonction de reconstruction $\mathcal{R}$, construisant un classificateur comprimé $h^{\sigma}_{\mathbf{i}}$ à partir d'un ensemble de compression $S_{\mathbf{i}}$ et d'un message $\sigma \in \Sigma_{\mathbf{i}}$, pour toute distribution de messages $P_{\Sigma_{\mathbf{i}}}$, pour tout $\delta \in (0, 1]$, on a, avec probabilité au moins $1 - \delta$ sur le choix de $S \sim D^m$,*

$$\forall \, \mathbf{i} \in I, \, \forall \sigma \in \Sigma_{\mathbf{i}} : \\ R(h^{\sigma}_{\mathbf{i}}) \ \leq \ 1 - \exp\left(\frac{-1}{m - |\mathbf{i}| - k} \left[\ln \binom{m}{|\mathbf{i}|} + \ln \binom{m - |\mathbf{i}|}{k} + \ln \left(\frac{1}{P_{\Sigma_{\mathbf{i}}}(\sigma) \cdot \xi(|\mathbf{i}|) \cdot \xi(k) \cdot \delta} \right) \right] \right) ,$$

où $k \coloneqq (m - |\mathbf{i}|) \cdot R_{S \setminus S_{\mathbf{i}}}(h^{\sigma}_{\mathbf{i}})$, et où $\xi(a) \stackrel{\text{def}}{=} \frac{6}{\pi^2}(a+1)^{-2}$.

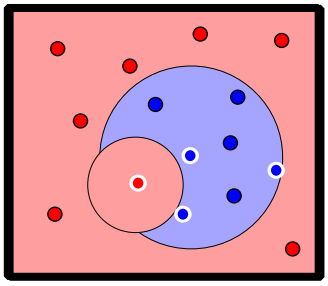


FIGURE 2.5 – Exemple de classificateur dépendant des données constitué d'une conjonction de deux boules. Les exemples positifs et négatifs sont respectivement représentés en bleu et en rouge. L'ensemble de compression est formé des quatre exemples à la bordure blanche.

La borne sur le risque d'un classificateur comprimé découlant du théorème 2.3 croit avec l'augmentation du nombre d'erreurs k sur les exemples n'appartenant pas à l'ensemble de compression ainsi qu'avec l'augmentation de la taille de l'ensemble de compression $|\mathbf{i}|$. Cette théorie suggère qu'un algorithme d'apprentissage doit favoriser les classificateurs qui, tout en possédant un risque empirique faible, s'expriment par un petit nombre d'exemples d'apprentissage.

2.3.2 L'algorithme SCM

Peu d'algorithmes sont conçus en accord avec le paradigme de l'apprentissage par compression d'échantillons. C'est toutefois le cas des *Machines à couverture d'ensembles* (ou ***SCM***, de l'anglais *Set Covering Machines*) suggéré par Marchand et Shawe-Taylor (2002). Un SCM s'avère une ***conjonction*** ou une ***disjonction*** de classificateurs qui dépendent des données. Ces formules booléennes sont représentées par un ensemble de classificateurs, que nous désignons par $\mathcal{B}$:

$$\begin{aligned} h^{\wedge}_{\mathcal{B}}(x) &\stackrel{\text{def}}{=} \bigwedge_{h\in\mathcal{B}} h(x) \quad \text{pour la conjonction,} \\ h^{\vee}_{\mathcal{B}}(x) &\stackrel{\text{def}}{=} \bigvee_{h\in\mathcal{B}} h(x) \quad \text{pour la disjonction,} \end{aligned} \tag{2.19}$$

en adoptant la convention VRAI $\equiv +1$ et FAUX $\equiv -1$.

En considérant l'ensemble de classificateurs complémentaires $\overline{\mathcal{B}} \stackrel{\text{def}}{=} \{\overline{h} \,|\, \overline{h}(x) = -h(x), h \in \mathcal{B}\}$, on constate qu'il est possible d'exprimer un classificateur par disjonction sur $\mathcal{B}$ comme le complément (ou la négation) d'un classificateur par conjonction sur $\overline{\mathcal{B}}$:

$$h^{\vee}_{\mathcal{B}}(x) \;=\; \bigvee_{h\in\mathcal{B}} h(x) \;=\; -\bigwedge_{h\in\mathcal{B}} \Big(-h(x)\Big) \;=\; -\bigwedge_{\overline{h}\in\overline{\mathcal{B}}} \overline{h}(x) \;=\; -h^{\wedge}_{\overline{\mathcal{B}}}(x)\,.$$

Ainsi, sans perte de généralité, la discussion qui suit traite seulement des classificateurs par conjonction.

Algorithme 1 CONSTRUIRE_SCM(*données S, classificateurs $\mathcal{H}$, pénalité p, critère d'arrêt s*)

Initialiser $\mathcal{N} \leftarrow \{x \,|\, (x,-1) \in S\}$ et $\mathcal{P} \leftarrow \{x \,|\, (x,+1) \in S\}$.
Initialiser $t \leftarrow 0$.
tant que $(t < s)$ **ou** $(\mathcal{N} \neq \emptyset)$ **faire**
  $t \leftarrow t + 1$.
  Sélectionner le meilleur classificateur selon la fonction (2.20) :

$$h_t \coloneqq \operatorname*{argmax}_{h \in \mathcal{H}} \{\, U^p(h, \mathcal{N}, \mathcal{P}) \,\} .$$

  Retirer les exemples couverts :

$$\mathcal{N} \leftarrow \{x \in \mathcal{N} \,|\, h_t(x) \neq -1\} \; ; \quad \mathcal{P} \leftarrow \{x \in \mathcal{P} \,|\, h_t(x) \neq -1\} .$$

fin tant que
retourner $\mathcal{B} \coloneqq \{h_1, \ldots, h_t\}$ correspondant au classificateur $h^{\wedge}_{\mathcal{B}}$ défini à l'équation (2.19).

On dit qu'un classificateur appartenant à une conjonction *couvre un exemple* lorsqu'il le classifie négativement. On constate qu'une conjonction de classificateurs booléens $h^{\wedge}_{\mathcal{B}}$ classifie un exemple x négativement si et seulement si au moins un classificateur dans $\mathcal{B}$ couvre x.

Étant donné un échantillon d'entraînement S et un ensemble de classificateurs $\mathcal{H}$, l'algorithme du SCM (voir l'algorithme 1) est une procédure gloutonne qui permet de sélectionner un petit sous-ensemble $\mathcal{B} \subseteq \mathcal{H}$ de classificateurs de telle sorte qu'un nombre élevé d'exemples négatifs de S soient couverts par au moins un classificateur appartenant à $\mathcal{B}$. À l'initialisation, les ensembles $\mathcal{N}$ et $\mathcal{P}$ contiennent respectivement les descriptions des exemples négatifs et positifs de S. À chaque étape, l'algorithme ajoute à la conjonction un classificateur qui couvre un nombre élevé d'exemples de $\mathcal{N}$ et un nombre faible d'exemples de $\mathcal{P}$, puis il retire les exemples couverts des ensembles $\mathcal{N}$ et $\mathcal{P}$. Le compromis entre le nombre d'exemples négatifs couverts et le nombre d'exemples positifs couverts est attribuable à un *paramètre de pénalité* $p \in [0, \infty[$. Plus précisément, le classificateur choisi à chaque itération est celui qui maximise la fonction de coût suivante :

$$U^p(h, \mathcal{N}, \mathcal{P}) \stackrel{\text{def}}{=} \big|\, \{x \in \mathcal{N} \,|\, h(x) = -1\} \,\big| \; - \; p \cdot \big|\, \{x \in \mathcal{P} \,|\, h(x) = -1\} \,\big| . \tag{2.20}$$

En plus du paramètre de pénalité, le SCM requiert un *critère d'arrêt* $s \in \mathbb{N}^*$ qui fait office de régularisateur en limitant la taille de la conjonction (c'est-à-dire en imposant $|\mathcal{B}| \leq s$).

Les expérimentations réalisées par Marchand et Shawe-Taylor (2002) utilisent des *boules dépendantes des données* comme ensemble de classificateurs pour créer un SCM. Chaque boule $h^{\odot}_{i,j} \in \mathcal{H}$ est caractérisée par deux exemples d'entraînement, soit un centre $(x_i, y_i) \in S$ et une bordure $(x_j, y_j) \in S$:

$$h^{\odot}_{i,j}(x) \stackrel{\text{def}}{=} \begin{cases} +y_i \text{ si } d(x, x_i) \leq d(x_i, x_j) + \epsilon \cdot y_i \,, \\ -y_i \text{ sinon,} \end{cases} \tag{2.21}$$

où $\epsilon > 0$ est un nombre arbitrairement petit et $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ est une mesure de distance entre deux exemples.

2.3.3 Évidences empiriques de l'efficacité du SCM

Bien que la fonction $d(\cdot, \cdot)$ puisse être de nature variée, les expérimentations de Marchand et Shawe-Taylor (2002) se concentrent sur la distance euclidienne. Il en ressort que l'algorithme SCM construit des classificateurs dont les risques rivalisent souvent avec ceux construits par les SVM utilisant un noyau RBF (tel que présenté à la figure 2.4c, page 25). Par sa nature, le SCM est apte à produire des classificateurs beaucoup plus compacts que le SVM, c'est-à-dire des classificateurs exprimables par peu de données.

Les bornes provenant de la théorie de la compression d'échantillons (comme celle présentée au théorème 2.3) s'avèrent d'excellents critères de sélection de modèle pour le paramètre de pénalité p et le critère d'arrêt s du SCM. En effet, la disjonction (ou la conjonction) de boules sélectionnée par la borne et celle sélectionnée par *validation croisée*[8] possèdent des risques comparables. Il est exceptionnel qu'une borne sur le risque se révèle aussi appropriée, ce qui semble confirmer l'existence d'une adéquation entre l'algorithme du SCM et la théorie de la compression d'échantillons.

Travaux connexes éffectués pendant le doctorat

L'article Germain et al. (2012a), coécrit au cours du doctorat, est consacré à l'étude de l'algorithme SCM. Les résultats contenus dans cet article montrent que l'heuristique de recherche employée par l'algorithme du SCM, bien qu'il ne minimise pas directement l'expression de la borne exprimée par le théorème 2.3, trouve presque toujours une solution près du minimum suggéré par la borne. Il s'agit d'un résultat surprenant, puisque l'algorithme (glouton) du SCM est très rapide d'exécution, alors que la recherche du classificateur qui minimise la borne est un problème NP-complet.

Nous donnons un aperçu de cet article à la section 7.3 (page 156) et le texte complet du manuscrit est reproduit à l'annexe D (page 201).

2.4 Les votes de majorité et l'algorithme AdaBoost

Un vote de majorité est une agrégation de plusieurs classificateurs. Dans cette thèse, nous désignons les classificateurs formant le vote de majorité par le terme ***votants***.

Plus formellement, un ***vote de majorité*** est un classificateur caractérisé par un ensemble (fini ou infini) de votants. Nous représentons un vote de majorité par une distribution Q sur

8. La technique de la *validation croisée* est présentée plus loin dans ce chapitre (section 2.5.2).

un ensemble de votants $\mathcal{H}$. Étant donné un espace de sortie $\mathcal{Y}$ comportant un nombre (fini) quelconque d'étiquettes possibles, un vote de majorité $h_Q : \mathcal{X} \to \mathcal{Y}$ retourne l'étiquette la plus probable selon la pondération Q :

$$h_Q(x) \stackrel{\text{def}}{=} \operatorname*{argmax}_{y \in \mathcal{Y}} \left(\mathop{\mathbf{E}}_{h \sim Q} \mathtt{I}[h(x) = y] \right). \tag{2.22}$$

Dans cette thèse, nous étudions plus spécifiquement le cas de la classification binaire, présenté ci-dessous.

2.4.1 Le cas de la classification binaire

Nous notons $B_Q(\cdot)$ le vote de majorité dans le contexte de la ***classification binaire***, c'est-à-dire lorsque les votants possèdent la forme $h : \mathcal{X} \to \{-1, +1\}$ et que l'espace de sortie est $\mathcal{Y} \coloneqq \{-1, +1\}$. Nous réécrivons alors l'équation (2.22), exprimant la fonction de classification du vote de majorité, ainsi :

$$B_Q(x) \stackrel{\text{def}}{=} \operatorname{sgn}\left[\mathop{\mathbf{E}}_{h \sim Q} h(x) \right] = \begin{cases} +1 \text{ si } \mathop{\mathbf{E}}_{h \sim Q} h(x) > 0\,, \\ -1 \text{ sinon.} \end{cases} \tag{2.23}$$

Le chapitre 3 est consacré à l'étude de tels votes de majorité, que nous nommons aussi ***classificateur de Bayes***. Notre étude repose sur la ***marge du vote de majorité***. La marge d'un vote de majorité B_Q sur un exemple (x, y) est exprimée par

$$M_Q(x, y) \stackrel{\text{def}}{=} y \mathop{\mathbf{E}}_{h \sim Q} h(x)\,. \tag{2.24}$$

Nous remarquons que $M_Q(x, y)$ est positif si l'exemple est bien classifié (c'est-à-dire lorsque $B_Q(x) = y$), et négatif sinon (lorsque $B_Q(x) \neq y$).

Les définitions des équations (2.23) et (2.24) seront généralisées au chapitre 3 afin d'étudier les votes de majorité dont les votants sont des régresseurs de la forme $f : \mathcal{X} \to [-1, 1]$.

2.4.2 L'algorithme AdaBoost

L'algorithme d'apprentissage ***AdaBoost*** (Freund et Schapire, 1997; Schapire et Singer, 1999; Schapire, 2003) construit un vote de majorité de manière itérative. Au fil des itérations, il maintient une distribution de poids sur les exemples d'entraînement de telle sorte que les exemples mal classifiés voient leur poids augmenter, alors que les exemples bien classifiés voient leur poids diminuer. À chaque itération, l'algorithme sélectionne un votant ayant un faible risque sur la distribution des exemples pondérés. De cette manière, le nouveau votant tend à compenser les erreurs empiriques effectuées par les votants sélectionnés précédemment.

L'algorithme 2 présente le pseudo-code d'AdaBoost, inspiré de celui proposé par Schapire (2003). L'hyperparamètre $t_{\max}$ correspond au nombre total d'itérations exécutées par l'algorithme. Nous désignons par W_t la distribution de poids sur les exemples à l'itération t. La

Algorithme 2 AdaBoost(*données S, classificateurs $\mathcal{H}$, nombre d'itérations $t_{\max}$*)

Initialiser $W_1(i) \leftarrow 1/m$ pour $i = 1, \ldots, m$ (avec $m \coloneqq |S|$).
pour $t = 1, \ldots, t_{\max}$ **faire**
Sélectionner le classificateur de risque empirique pondéré minimal :

$$h_t \;\leftarrow\; \operatorname*{argmin}_{h \in \mathcal{H}} \Big[r_t(h) \Big] , \qquad \text{avec } r_t(h) \;\coloneqq\; \sum_{i=1}^{m} W_t(i)\; \mathtt{I}\big[h_t(x_i) \neq y_i \big] . \tag{2.25}$$

Calculer le poids du classificateur dans le vote de majorité :

$$\alpha_t \;\leftarrow\; \frac{1}{2} \ln \left[\frac{1 - r_t(h_t)}{r_t(h_t)} \right] . \tag{2.26}$$

Mettre à jour les poids des exemples (Z_t est une constante de normalisation) :

$$W_{t+1}(i) \;\leftarrow\; \frac{W_t(i) \cdot \exp\Big(- \alpha_t y_i h_t(x_i) \Big)}{Z_t} . \tag{2.27}$$

fin pour

retourner $B_{\boldsymbol{\alpha}}(\cdot) \;\stackrel{\text{def}}{=}\; \operatorname{sgn} \left[\sum_{t=1}^{t_{\max}} \alpha_t \, h_t(\cdot) \right] .$

ligne (2.25) correspond au choix du votant $h_t(\cdot)$ de plus faible risque pondéré. Notons que certaines implémentations d'AdaBoost remplacent le «argmin» par l'appel à un algorithme d'apprentissage auxiliaire acceptant un échantillon pondéré d'exemples (généralement désigné par le terme *weak learner* dans la littérature anglaise).

Il est montré dans Schapire et Singer (1999) que l'algorithme AdaBoost minimise une borne supérieure sur le risque empirique du vote de majorité, qui est donnée par la fonction de *perte exponentielle* :

$$R_S(B_Q) \;=\; \frac{1}{m} \sum_{i=1}^{m} \underbrace{\mathtt{I}\Big[B_{\boldsymbol{\alpha}}(x_i) \neq y_i \Big]}_{\textbf{\textit{perte zéro-un}}} \;\leq\; \frac{1}{m} \sum_{i=1}^{m} \underbrace{\exp\left(-y_i \sum_{t=1}^{t_{\max}} \alpha_t h_t(x_i) \right)}_{\textbf{\textit{perte exponentielle}}} .$$

La figure 2.6 donne un aperçu de la perte exponentielle mesurée sur un exemple (x, y) donné, qui s'avère une relaxation convexe de la perte zéro-un. Il est important de préciser que la perte exponentielle minimisée par AdaBoost est calculée à partir des poids $\boldsymbol{\alpha} \coloneqq (\alpha_1, \ldots, \alpha_{t_{\max}})$ déterminés par l'équation (2.26), qui ne forment pas une distribution. La perte exponentielle minimisée par l'algorithme AdaBoost est donc calculée sur une notion de *marge non normalisée du vote de majorité*. Cela dit, le classificateur $B_{\boldsymbol{\alpha}}(\cdot)$ retourné par AdaBoost peut être converti en un classificateur équivalent $B_Q(\cdot)$, caractérisé par une distribution de poids Q sur $\mathcal{H}$, en

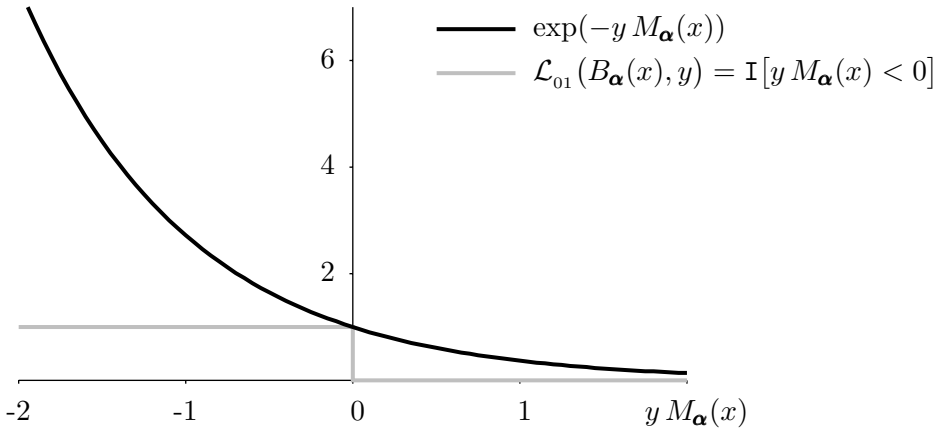


FIGURE 2.6 – Comparaison des valeurs de la *perte exponentielle* et de la *perte zéro-un* d'un vote de majorité $B_{\boldsymbol{\alpha}}$ retourné par l'algorithme *AdaBoost* en termes de sa *marge non normalisée* $y\,M_{\boldsymbol{\alpha}}(x)$, avec $M_{\boldsymbol{\alpha}}(x) \stackrel{\text{def}}{=} \sum_{t=1}^{t_{\max}} \alpha_t h_t(x)$.

posant

$$Q(h_t) \;\coloneqq\; \frac{\alpha_t}{\sum_{i=1}^{t_{\max}} \alpha_i}\,, \quad \text{et} \quad Q(h) \;\coloneqq\; 0 \text{ pour } h \in \mathcal{H} \setminus \{h_1, \ldots, h_{t_{\max}}\}\,. \tag{2.28}$$

Le classificateur $B_Q(\cdot)$ ainsi exprimé respecte la forme de l'équation (2.23). Cette méthode de normalisation des poids est employée lors des expérimentations empiriques présentées par les chapitres 3, 4 et 5.

2.4.3 Les souches de décision

Les ***souches de décision*** (ou, en anglais, *decisions stumps*) sont des classificateurs très simples souvent utilisés comme votants de pair avec l'algorithme AdaBoost.

Pour classifier un exemple $\mathbf{x} \coloneqq (a_1, a_2, \ldots, a_n) \in \mathbb{R}^n$, une souche de décision $h_{i,t,d}$ considère la valeur d'un seul attribut a_i, où $i \in \{1, \ldots, n\}$. La sortie du classificateur est déterminée par la valeur de seuil $t \in \mathbb{R}$ et la direction $d \in \{-1, +1\}$:

$$h_{i,t,d}(\mathbf{x}) \;\stackrel{\text{def}}{=}\; \begin{cases} +d & \text{si } a_i > t \\ -d & \text{sinon.} \end{cases} \tag{2.29}$$

La figure 2.7 illustre que, même si les souches de décision sont des classificateurs excessivement simples, leur combinaison au sein d'un vote de majorité permet d'exprimer des frontières de décision d'une certaine complexité.

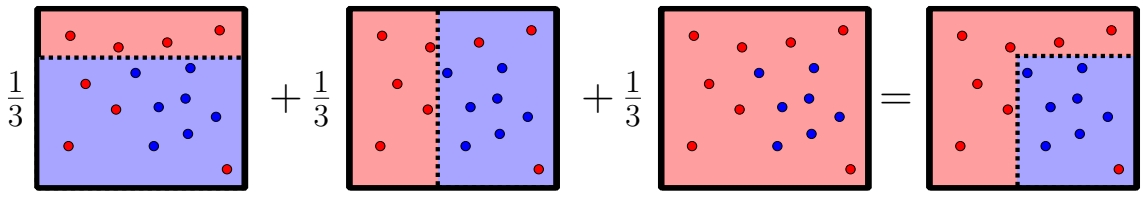


FIGURE 2.7 – Exemple de vote de majorité comportant trois votants. Les deux premiers votants (à partir de la gauche) sont des souches de décision. Le troisième votant est un classificateur «trivial». Les trois votants possèdent le même poids ($\frac{1}{3}$), ce qui permet d'obtenir le vote de majorité illustré à la droite du symbole d'égalité.

2.5 La sélection de modèle

La section précédente a introduit trois algorithmes d'apprentissage : le SVM, le SCM et AdaBoost. Il en existe beaucoup d'autres. Quiconque désire utiliser l'apprentissage automatique pour résoudre un problème doit donc choisir quel algorithme exécuter sur l'échantillon d'entraînement contenant les données propres à la tâche qu'il veut résoudre. De plus, la plupart des algorithmes requièrent un choix d'hyperparamètres. Par exemple, le SVM nécessite une valeur de C (voir l'équation (2.7), page 21), le SCM une combinaison de valeurs de p et s (voir l'algorithme 1, page 30), et AdaBoost une valeur t_{max} (voir l'algorithme 2, page 33).

Typiquement, un praticien confronté à un problème d'apprentissage exécute plusieurs algorithmes différents, chacun d'eux plusieurs fois en variant les valeurs d'hyperparamètres. Puisque chaque combinaison d'un algorithme et d'un (ou plusieurs) hyperparamètre(s) produit un classificateur distinct, il est confronté au problème de la ***sélection de modèle*** :

> Parmi un ensemble de classificateurs, comment sélectionner celui qui sera par la suite utilisé pour classifier de nouveaux exemples ?

Le texte qui suit présente sommairement deux méthodes utilisées couramment par les praticiens de l'apprentissage automatique pour la sélection de modèle.

2.5.1 Sélection de modèle par ensemble de validation

La méthode de sélection de modèle la plus simple est certainement le recours à un ***ensemble de validation***. Étant donné un échantillon d'entraînement S, on le divise en deux sous ensembles disjoints : un nouvel échantillon d'entraînement $S' \subset S$ et un ensemble de validation $V \coloneqq S \setminus S'$.[9] On exécute ensuite le (ou les) algorithme(s) d'apprentissage avec une variété d'hyperparamètres. Pour chaque classificateur $h(\cdot)$ ainsi obtenu, on calcule le ***risque de validation*** $R_V(h)$. Finalement, on sélectionne le classificateur dont le risque de validation est minimal.

9. Une pratique rependue est de diviser l'échantillon d'entraînement S de sorte que S' contient 80% à 90% des exemples (et donc V en contient 10% à 20%.). Cela étant dit, il s'agit d'un choix qui varie fortement en fonction du problème et du praticien.

Cette méthode simple implique que l'on prive l'algorithme d'apprentissage des exemples de l'ensemble de validation V. Cela s'avère problématique lorsqu'on possède peu d'exemples d'entraînement. Pour pallier cet inconvénient, on a recours à la méthode de la *validation croisée*, qui permet alors d'utiliser toute l'information de S pour construire le classificateur.

2.5.2 Sélection de modèle par validation croisée

La ***validation croisée*** k-fois est une méthode couramment utilisée pour sélectionner les hyperparamètres d'un algorithme d'apprentissage. Elle consiste à partitionner l'échantillon d'entraînement S en k sous-ensembles disjoints $V_1, V_2, \ldots, V_k$ de tailles semblables. Pour chaque combinaison d'hyperparamètres p à évaluer, on exécute l'algorithme d'apprentissage un nombre k de «fois». À chaque «fois» $i \in \{1, \ldots, k\}$, un sous-ensemble distinct V_i est réservé pour *valider* le classificateur $h_i(\cdot)$ entraîné avec les exemples $S \setminus V_i$:

$$A_p\left(S \setminus V_i\right) \longrightarrow h_i\,.$$

On calcule ensuite le ***risque de validation croisée*** associé à ces hyperparamètres, c'est-à-dire la moyenne des risques calculés pour chaque sous-ensemble de validation :

$$R_{CV}(p) \stackrel{\text{def}}{=} \frac{1}{k}\sum_{i=1}^{k} R_{V_i}(h_i)\,.$$

Parmi toutes les combinaisons d'hyperparamètres évaluées, on retient celle possédant le risque de validation croisée minimal. Ces hyperparamètres p^* sont alors utilisés pour entraîner l'algorithme sur la totalité de l'échantillon d'entraînement S, ce qui fournit le classificateur final :

$$A_{p^*}(S) \longrightarrow h^*\,.$$

Cette méthode tente d'utiliser au maximum les données disponibles pour sélectionner les hyperparamètres adéquatement. En pratique, elle se révèle d'une grande utilité, mais elle possède deux inconvénients. Premièrement, son utilisation ralentit considérablement le processus d'apprentissage. Deuxièmement, bien que l'on suppose qu'une combinaison d'hyperparamètres possédant un faible risque de validation croisée permet d'obtenir un bon classificateur lorsque l'algorithme d'apprentissage est exécuté sur toutes les données d'entraînement, la validation croisée ne fournit aucune garantie sur l'erreur de généralisation du classificateur final.

2.6 Les garanties de généralisation

Si les méthodes de sélection de modèle guident le praticien dans le choix d'algorithme d'apprentissage et d'hyperparamètres à utiliser pour résoudre un problème en particulier, elles fournissent peu de garanties rigoureuses que le classificateur sélectionné prédira adéquatement l'étiquette de nouveaux exemples (que l'on ne connaît pas au moment de l'apprentissage). Autrement dit, un praticien est confronté aux questions suivantes :

Lorsqu'on a sélectionné un classificateur, comment savoir s'il possède un faible risque ? Autrement dit, comment fournir des garanties qu'il se «comportera bien» lorsqu'on l'utilisera pour classifier de «nouveaux» exemples ?

Pour répondre à ces questions, on cherche à obtenir des ***garanties de généralisation*** (ou *bornes sur le risque*). Cette notion est au cœur de la présente thèse, puisque l'objectif premier de la théorie PAC-bayésienne est de formuler des garanties de généralisation obtenues à partir du risque empirique d'un classificateur (sur l'échantillon d'entraînement). [10] Cela étant dit, il est souvent plus simple d'obtenir des garanties de généralisation par le recours à un *échantillon de test*, comme nous décrivons dans la sous-section suivante.

2.6.1 Garanties de généralisation à partir de l'échantillon de test

Une pratique courante consiste à estimer le risque en calculant le risque empirique sur un ***échantillon de test*** T, contenant m_t exemples qui n'ont pas servi au processus d'entraînement. Comme on suppose que les exemples contenus dans T sont générés de manière *i.i.d.* selon la distribution génératrice D, le nombre d'erreurs sur l'échantillon de données de test est une variable aléatoire qui suit une ***loi binomiale***.

Plus précisément, considérons une distribution génératrice D sur $\mathcal{X} \times \{-1, 1\}$ et un classificateur binaire $h : \mathcal{X} \to \{-1, 1\}$. Le risque de ce classificateur, c'est-à-dire son espérance de perte zéro-un sur la distribution D, est $R_D(h)$. Associons le classificateur $h(\cdot)$ à une variable aléatoire binomiale X_h de m_t essais avec probabilité de succès $R_D(h)$, ce que nous désignons ainsi :

$$X_h \sim B\Big(m_t, R_D(h)\Big) .$$

Du point de vue de la variable binomiale X_h, un *succès* correspond à une *erreur* de prédiction du classificateur $h(\cdot)$. Conséquemment, la probabilité d'observer spécifiquement k erreurs parmi m_t exemples est donnée par

$$\begin{aligned} \Pr_{T \sim D^{m_t}} \left(R_T(h) = \frac{k}{m_t} \right) &= \Pr_{X_h \sim B(m_t, R_D(h))} \Big(X_h = k \Big) \\ &= \binom{m_t}{k} \Big(R_D(h) \Big)^k \Big(1 - R_D(h) \Big)^{m_t - k} . \end{aligned}$$

Nous présentons ici une borne supérieure sur le risque d'un classificateur obtenu à partir d'un échantillon de test, telle que suggérée par Langford (2005). Remarquons d'abord que la fonction de répartition de la queue de la loi binomiale, notée ici $\mathrm{Bin}(m', k, r)$, exprime la probabilité qu'un classificateur de vrai risque r fasse jusqu'à k erreurs parmi m' exemples de

10. Idéalement, nous voudrions que ces garanties de généralisation soient aussi de bons critères de sélection de modèle. Cet aspect n'est pas abordé dans cette thèse, bien que les bornes dérivées de la théorie PAC-bayésienne se révèlent, dans certains contextes, de bons critères de sélection de modèle (voir McAllester, 2003a; Langford, 2005; Ambroladze et al., 2006; Parrado-Hernández et al., 2012).

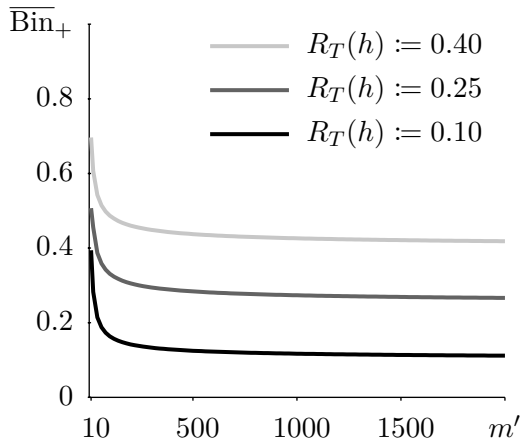


FIGURE 2.8 – Valeurs de la borne sur l'échantillon de test $\overline{\text{Bin}}_+(m', m' R_T(h), \delta)$ en fonction du nombre d'exemples de test $m' := |T|$ (avec $\delta := \frac{1}{20}$) pour des risques empiriques observés $R_T(h) \in \{0.10, 0.25, 0.40\}$.

test :

$$\text{Bin}(m', k, r) \stackrel{\text{def}}{=} \Pr_{X \sim B(m', r)} \Big(X \leq k \Big) = \sum_{i=0}^{k} \binom{m'}{i} r^i (1-r)^{m'-i} .$$

Nous nous intéressons maintenant à l'inverse de la queue de la binomiale, $\overline{\text{Bin}}_+(m', k, \delta)$, c'est-à-dire la plus grande valeur de risque r telle que la probabilité est au moins δ de faire jusqu'à k erreurs parmi m' exemples.

Théorème 2.4 (Langford, 2005)**.** *Pour toute distribution D sur $\mathcal{X} \times \{-1, 1\}$, pour tout classificateur $h : \mathcal{X} \to \{-1, 1\}$ de risque $R_D(h)$ et pour tout $\delta \in (0, 1]$, on a, avec probabilité au moins $1 - \delta$ sur le choix de $T \sim D^{m'}$,*

$$R_D(h) \leq \overline{\text{Bin}}_+\Big(m', \; m' R_T(h), \; \delta \Big) , \tag{2.30}$$

où $\overline{\text{Bin}}_+$ correspond à l'inverse de la queue de la binomiale :

$$\overline{\text{Bin}}_+\Big(m', k, \delta \Big) \stackrel{\text{def}}{=} \max \left\{ r \in [0, 1] \;\middle|\; \text{Bin}(m', k, r) \geq \delta \right\} .$$

Ainsi, l'inverse de la queue de la binomiale nous donne une borne supérieure sur le risque d'un classificateur à partir du risque observé sur l'échantillon de test. À l'aide d'une méthode similaire, on peut aussi exprimer une borne inférieure sur le risque (voir Langford, 2005).

Comme l'illustre la figure 2.8, il est possible d'obtenir une valeur de borne très serrée grâce au théorème 2.4. Cependant, la borne est valide *avec probabilité au moins* $1 - \delta$ seulement si on l'évalue sur l'échantillon de test T avec un seul classificateur. Pour obtenir plusieurs bornes simultanément valides pour k classificateurs $h_1, \ldots, h_k$ à partir d'un seul ensemble T

avec probabilité au moins $1-\delta$, il faut calculer chacune des k bornes en utilisant dans l'équation (2.30) un paramètre de confiance $\delta := \frac{\delta}{k}$ (en vertu de la borne de l'union). Ceci entraîne une importante dégradation des bornes.

2.6.2 Garanties de généralisation à partir l'échantillon d'entraînement

Cette thèse est dédiée à l'étude de la ***théorie PAC-bayésienne*** (initiée par McAllester, 1999), qui permet d'obtenir des garanties de généralisation en évitant le recours à un échantillon de test, contrairement à la borne découlant du théorème 2.4. En effet, comme nous avons brièvement discuté à la sous-section 1.2 de l'introduction (page 4), les bornes PAC-bayésiennes reposent sur le *risque empirique* (et sur quelques autres éléments que nous définissons rigoureusement au chapitre 4).

Nous le verrons en détail dans les chapitres suivants, malgré le fait que les bornes sur le risque exprimées à l'aide d'un échantillon test possèdent généralement de plus petites valeurs, l'approche PAC-bayésienne possède deux avantages notables :

1. Les bornes PAC-bayésiennes permettent de calculer des garanties de généralisations (des bornes sur le risque d'un classificateur) statistiquement valides tout en mettant à profit *tous les exemples étiquetés* à notre disposition lors du processus d'apprentissage. Dans certains contextes où les exemples sont rares ou coûteux à étiqueter (en temps ou en argent), cela peut faire une grande différence.
2. Les bornes PAC-bayésiennes expriment, par leur expression mathématique, un compromis entre diverses quantités. Ainsi, elles nous enseignent ce que l'on doit prendre en compte pour favoriser un «bon» apprentissage à partir d'un échantillon d'entraînement. Conséquemment, chaque borne PAC-bayésienne est potentiellement une source d'inspiration pour la conception d'un nouvel algorithme d'apprentissage. Un tel algorithme consiste essentiellement à minimiser l'expression mathématique de la borne à l'aide d'une procédure d'optimisation appropriée.

Idéalement, nous voudrions que ces garanties de généralisation soient aussi de bons critères de sélection de modèle. Cet aspect n'est pas abordé dans cette thèse, malgré le fait que les bornes dérivées de la théorie PAC-bayésienne se révèlent, dans certains contextes, de bons critères de sélection de modèle (voir McAllester, 2003a; Langford, 2005; Ambroladze et al., 2006; Parrado-Hernández et al., 2012).

Mentionnons que la théorie PAC-bayésienne n'est pas la seule approche permettant d'exprimer des garanties de généralisation à partir de l'échantillon d'entraînement. Les autres théories souvent citées dans la littérature propre à l'apprentissage automatique sont notamment la *dimension VC* (Vapnik et Chervonenkis, 1971) et la *complexité de Rademacher* (Koltchinskii et Panchenko, 2000). Nous n'abordons pas ces approches dans la présente thèse. Pour une introduction à ces deux théories, nous suggérons la lecture de Mohri et al. (2012, chapitre 3) ou Shalev-Shwartz et Ben-David (2014, chapitres 6 et 26).

Deuxième partie

Le cadre d'apprentissage inductif revisité

Chapitre 3

Étude des votes de majorité

Résumé. Ce chapitre se consacre à l'étude des classificateurs au centre de la théorie PAC-bayésienne : les votes de majorité. Nous y présentons d'abord les concepts de risque de Bayes, de risque de Gibbs et d'espérance de désaccord. Nous exprimons ensuite ces quantités en termes de la marge du vote de majorité. En considérant cette marge comme une variable aléatoire, nous démontrons la $\mathcal{C}$-borne (présentée initialement par Lacasse et al., 2006) de manière élégante et intuitive, en faisant ressortir le rôle joué par l'espérance de désaccord entre les votants. La $\mathcal{C}$-borne permet de borner supérieurement le risque d'un vote de majorité.

3.1 Le risque de Bayes

Comme nous l'avons introduit à la section 2.4 (page 31), nous représentons un ***vote de majorité*** par une distribution Q sur un ensemble (possiblement infini) de *votants* $\mathcal{H}$. Dans ce chapitre, nous étudions la classification binaire, où l'espace de sortie est

$$\mathcal{Y} \;\coloneqq\; \{-1, +1\}\,.$$

De plus, nous considérons que chaque ***votant*** $f \in \mathcal{H}$ est un régresseur de la forme

$$f \;:\; \mathcal{X} \to [-1, +1]\,.$$

Étant donné la description d'un exemple $x \in \mathcal{X}$, nous interprétons le signe de $f(x)$ comme la prédiction du votant $f(\cdot)$ quant à l'étiquette de x, et l'amplitude $|f(x)|$ comme une mesure de confiance envers cette prédiction. Le classificateur de *vote de majorité* associé à ces régresseurs, que l'on nomme aussi le ***classificateur de Bayes*** $B_Q : \mathcal{X} \to \{-1, +1\}$, est défini par l'équation (3.1) ci-bas. Notons qu'il s'agit d'une généralisation de l'équation (2.23), présentée en introduction pour les votes de majorité dont les votants sont des classificateurs binaires.

$$B_Q(x) \;\stackrel{\text{def}}{=}\; \operatorname{sgn}\left(\mathop{\mathbf{E}}_{f\sim Q} f(x)\right). \tag{3.1}$$

La présente thèse se consacre à la recherche de garanties, les plus précises possibles, sur le ***risque*** du classificateur de Bayes $B_Q(\cdot)$. Nous nommons cette quantité le *risque de Bayes*, en opposition au *risque de Gibbs* défini à la section 3.2 (plus bas).

Définition 3.1. Soit une distribution Q sur un ensemble de votants. Le ***risque de Bayes*** $R_{D'}(B_Q)$, aussi nommé le ***risque du vote de majorité***, correspond à l'espérance de la perte zéro-un (voir les définitions 2.1 et 2.2) du classificateur $B_Q(\cdot)$ sur D',

$$\begin{aligned} R_{D'}(B_Q) \stackrel{\text{def}}{=} \mathbb{E}^{\mathcal{L}_{01}}_{D'}(B_Q) &= \mathop{\mathbf{E}}_{(x,y)\sim D'} \mathtt{I}\Big[B_Q(x) \neq y\Big] \\ &= \mathop{\mathbf{E}}_{(x,y)\sim D'} \mathtt{I}\Big[\mathop{\mathbf{E}}_{f\sim Q} y \cdot f(x) \leq 0\Big]\,. \end{aligned}$$

Rappelons que la notation D' signifie que la définition est valide à la fois pour le ***risque*** sur la distribution génératrice D (que l'on nomme aussi ***erreur de généralisation***) et le ***risque empirique*** calculé sur un échantillon d'entraînement $S \coloneqq \{(x_i, y_i)\}_{i=1}^m$. Dans le dernier cas, la définition 3.1 est donc équivalente à

$$\begin{aligned} R_S(B_Q) \stackrel{\text{def}}{=} \mathbb{E}^{\mathcal{L}_{01}}_{S}(B_Q) &= \sum_{i=1}^{m} \mathtt{I}\Big[B_Q(x_i) \neq y_i\Big] \\ &= \sum_{i=1}^{m} \mathtt{I}\Big[\mathop{\mathbf{E}}_{f\sim Q} y_i \cdot f(x_i) \leq 0\Big]\,. \end{aligned}$$

Le risque empirique $R_S(B_Q)$, calculé sur un échantillon $S \sim D^m$, est bien sûr un élément très important pour estimer la valeur sur le «vrai» risque $R_D(B_Q)$. Dans les prochaines sections de ce chapitre, nous verrons qu'il existe d'autres quantités utiles pour formuler des garanties statistiques sur l'erreur de généralisation.

3.2 Le risque de Gibbs

Le ***classificateur de Gibbs*** $G_Q(\cdot)$ est un classificateur *stochastique*[1] relié au vote de majorité. Pour classifier un exemple $x \in \mathcal{X}$, le classificateur de Gibbs pige aléatoirement un votant $\widehat{f}$ parmi $\mathcal{H}$, en accord avec la distribution de probabilité Q. Ensuite, la sortie de $G_Q(x)$ dépend à la fois du signe du votant $\widehat{f}(x)$ et de sa «confiance», c'est-à-dire de l'amplitude $|\widehat{f}(x)|$:

$$G_Q(x) \stackrel{\text{def}}{=} \begin{cases} \operatorname{sgn}\Big[\widehat{f}(x)\Big] & \text{avec probabilité } \dfrac{|\widehat{f}(x)|+1}{2}\,, \\ -\operatorname{sgn}\Big[\widehat{f}(x)\Big] & \text{sinon}\,. \end{cases}$$

Donc, la valeur retournée est $G_Q(x) = +1$ avec probabilité 1 lorsque $\widehat{f}(x) = 1$. Cette probabilité décroît progressivement avec la valeur de $\widehat{f}(x)$ pour atteindre $\frac{1}{2}$ lorsque $\widehat{f}(x) = 0$,

1. Contrairement aux autres classificateurs définis dans ce document, le classificateur de Gibbs ne respecte pas l'équation (2.2) (page 14). En effet, ce n'est pas une fonction, car sa sortie est stochastique : elle peut varier d'une fois à l'autre pour un même $x \in \mathcal{X}$.

et la probabilité devient nulle lorsque $\widehat{f}(x) = -1$. Dans cette dernière situation, on a donc nécessairement $G_Q(x) = -1$. Ainsi, le classificateur de Gibbs est une variable aléatoire telle que

$$\begin{aligned} \Pr\Big(G_Q(x) = +1\Big) &= \frac{1}{2}\left(\mathop{\mathbf{E}}_{f\sim Q} f(x) + 1\right), \\ \Pr\Big(G_Q(x) = -1\Big) &= \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{f\sim Q} f(x)\right). \end{aligned} \tag{3.2}$$

Le risque de ce classificateur stochastique est nommé le *risque de Gibbs*. La définition suivante découle directement de la définition du classificateur de Gibbs et de l'équation (3.2).

Définition 3.2. Soit une distribution Q sur un ensemble de votants. Le ***risque de Gibbs*** $R_{D'}(G_Q)$ correspond à l'espérance de la perte linéaire du classificateur Gibbs $G_Q(\cdot)$ sur D',

$$R_{D'}(G_Q) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}^{\mathcal{L}_\ell}_{D'}(f) = \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{(x,y)\sim D'} \mathop{\mathbf{E}}_{f\sim Q} y\cdot f(x)\right).$$

En pratique, nous utilisons rarement le classificateur de Gibbs pour résoudre des problèmes d'apprentissage automatique. Toutefois, le risque de Gibbs est un outil théorique important dans l'étude des votes de majorité. En effet, comme nous le verrons à la section 3.5, le risque de Bayes et le risque de Gibbs sont reliés.

3.3 L'espérance de désaccord

Définissons maintenant l'*espérance de désaccord* d'un ensemble de votants selon une distribution Q. Lorsque les votants sont des classificateurs binaires (retournant -1 ou $+1$), l'espérance de désaccord correspond à la probabilité que deux classificateurs pigés selon Q ne retournent pas la même valeur. Autrement dit,

$$\begin{aligned} \Pr_{\substack{(x,\cdot)\sim D'\\ h_1,h_2\sim Q}}\Big(h_1(x) \neq h_2(x)\Big) &= \mathop{\mathbf{E}}_{(x,\cdot)\sim D'} \mathop{\mathbf{E}}_{h_1,h_2\sim Q} \mathtt{I}\Big[h_1(x) \neq h_2(x)\Big] \\ &= \mathop{\mathbf{E}}_{(x,\cdot)\sim D'} \mathop{\mathbf{E}}_{h_1\sim Q} \mathop{\mathbf{E}}_{h_2\sim Q} \mathtt{I}\Big[h_1(x) \neq h_2(x)\Big] \\ &= \mathop{\mathbf{E}}_{(x,\cdot)\sim D'} \mathop{\mathbf{E}}_{h_1\sim Q} \mathop{\mathbf{E}}_{h_2\sim Q} \mathtt{I}\Big[h_1(x)\cdot h_2(x) \neq 1\Big] \\ &= \mathop{\mathbf{E}}_{(x,\cdot)\sim D'} \mathop{\mathbf{E}}_{h_1\sim Q} \mathop{\mathbf{E}}_{h_2\sim Q} \mathcal{L}_{01}\big(\, h_1(x)\cdot h_2(x)\,,\, 1\,\big), \end{aligned} \tag{3.3}$$

où $(x,\cdot) \sim D'$ signifie que x est généré suivant la ***distribution marginale*** de D' sur l'espace d'entrée $\mathcal{X}$. Ainsi, les étiquettes des exemples ne sont pas requises pour calculer l'espérance de désaccord.

La définition 3.3 généralise la notion de désaccord aux votants à valeurs réelles. Notons que l'équation (3.4) ci-bas est calquée sur l'équation (3.3), en substituant les classificateurs (binaires) $h_1(\cdot), h_2(\cdot)$ par les régresseurs $f_1(\cdot), f_2(\cdot)$, de même que la perte zéro-un $\mathcal{L}_{01}(\cdot,\cdot)$ par la perte linéaire $\mathcal{L}_\ell(\cdot,\cdot)$.

Définition 3.3. Soit une distribution de probabilité Q sur un ensemble de votants. L'***espérance de désaccord*** $d_Q^{D'}$ sur D' correspond à

$$\begin{aligned} d_Q^{D'} &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,\cdot)\sim D'} \mathop{\mathbf{E}}_{f_1\sim Q} \mathop{\mathbf{E}}_{f_2\sim Q} \mathcal{L}_\ell\left(\, f_1(x)\cdot f_2(x)\,,\, 1\,\right) \\ &= \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{(x,\cdot)\sim D'} \mathop{\mathbf{E}}_{f_1\sim Q} \mathop{\mathbf{E}}_{f_2\sim Q} 1\cdot f_1(x)\cdot f_2(x)\right) \\ &= \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{(x,\cdot)\sim D'}\left[\mathop{\mathbf{E}}_{f\sim Q} f(x)\right]^2\right). \end{aligned} \tag{3.4}$$

La notion de désaccord intervient dans plusieurs résultats présentés dans cette thèse. En effet, cette quantité contient certaines informations sur un vote de majorité qu'on ne peut pas déduire du risque de Bayes et du risque de Gibbs. Qui plus est, comme le désaccord se calcule uniquement à partir des descriptions des exemples, nous pouvons exploiter cette information dans les cadres d'apprentissage semi-supervisé ou transductif, c'est-à-dire en présence d'exemples non étiquetés.

3.4 La marge du vote de majorité et ses moments

La plupart des résultats du présent chapitre sont obtenus en considérant la marge d'un vote de majorité comme une variable aléatoire. La définition suivante clarifie ce concept.

Définition 3.4. Soit une distribution de probabilité Q sur un ensemble de votants. La ***marge du vote de majorité*** $B_Q(\cdot)$ sur un exemple $(x, y) \in \mathcal{X} \times \mathcal{Y}$ est donnée par

$$M_Q(x,y) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{f\sim Q} y\cdot f(x)\,.$$

De plus, nous désignons par $M_Q^{D'}$ la variable aléatoire qui donne la valeur de la marge du vote de majorité sur exemple généré par la distribution D'.

Le ***premier moment de la marge*** correspond au premier moment de la variable $M_Q^{D'}$,

$$\mu_1(M_Q^{D'}) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\,, \tag{3.5}$$

le ***deuxième moment de la marge*** correspond au deuxième moment de la variable $M_Q^{D'}$,

$$\mu_2(M_Q^{D'}) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim D'}\left[M_Q(x,y)\right]^2, \tag{3.6}$$

et la ***variance de la marge*** correspond à la variance de la variable $M_Q^{D'}$,

$$\begin{aligned} \mathrm{Var}(M_Q^{D'}) &\stackrel{\text{def}}{=} \mathop{\mathbf{Var}}_{(x,y)\sim D'}\left(M_Q(x,y)\right) \\ &= \mu_2(M_Q^{D'}) - \left(\mu_1(M_Q^{D'})\right)^2. \end{aligned} \tag{3.7}$$

On peut maintenant réécrire le risque de Bayes (définition 3.1) en fonction de marge :

$$R_{D'}(B_Q) \;=\; \Pr_{(x,y)\sim D'}\Big(M_Q(x,y)\leq 0\Big)\,. \tag{3.8}$$

De plus, le risque de Gibbs (définition 3.2) est directement lié au premier moment de la marge,

$$\begin{aligned} R_{D'}(G_Q) \;&=\; \mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}^{\mathcal{L}_\ell}_{D'}(f) \;=\; \frac{1}{2}\left(1-\mathop{\mathbf{E}}_{(x,y)\sim D'}\mathop{\mathbf{E}}_{f\sim Q} y\cdot f(x)\right)\\ &=\; \frac{1}{2}\Big(1-\mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\Big)\\ &=\; \frac{1}{2}\Big(1-\mu_1(M_Q^{D'})\Big)\,. \end{aligned} \tag{3.9}$$

Enfin, l'espérance de désaccord (définition 3.3) est liée au deuxième moment de la marge. En effet, on a

$$\begin{aligned} \mu_2(M_Q^{D'}) \;&=\; \mathop{\mathbf{E}}_{(x,y)\sim D'}\Big[M_Q(x,y)\Big]^2 \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D'} y^2\cdot\Big[\mathop{\mathbf{E}}_{f\sim Q} f(x)\Big]^2\\ &=\; \mathop{\mathbf{E}}_{(x,\cdot)\sim D'}\Big[\mathop{\mathbf{E}}_{f\sim Q} f(x)\Big]^2, \end{aligned}$$

et donc

$$\begin{aligned} d_Q^{D'} \;&=\; \frac{1}{2}\left(1-\mathop{\mathbf{E}}_{(x,\cdot)\sim D'}\left[\mathop{\mathbf{E}}_{f\sim Q} f(x)\right]^2\right)\\ &=\; \frac{1}{2}\Big(1-\mu_2(M_Q^{D'})\Big)\,. \end{aligned} \tag{3.10}$$

Puisque $0 \leq \mu_2(M_Q^{D'}) \leq 1$, l'équation (3.10) met en évidence que $0 \leq d_Q^{D'} \leq 1/2$. La proposition suivante améliore la borne supérieure « $d_Q^{D'} \leq 1/2$ » en exploitant la variance de la marge, notée $\mathrm{Var}(M_Q^{D'})$.

Proposition 3.5. *Pour toute distribution de probabilité Q sur un ensemble de votants et pour toute distribution D' sur $\mathcal{X}\times\mathcal{Y}$, on a*

$$d_Q^{D'} \;\leq\; 2\cdot R_{D'}(G_Q)\cdot\big(1-R_{D'}(G_Q)\big) \;\leq\; \frac{1}{2}\,.$$

De plus, si $d_Q^{D'}=\frac{1}{2}$ alors $R_{D'}(G_Q)=\frac{1}{2}$.

Démonstration. Étant donné que la variance de la marge $\mathrm{Var}(M_Q^{D'})$ est non négative, on déduit de l'équation (3.7) que

$$\mu_2(M_Q^{D'}) \;\geq\; \Big(\mu_1(M_Q^{D'})\Big)^2\,,$$

ce qui implique, par les équations (3.9) et (3.10),

$$1-2\cdot d_Q^{D'} \;\geq\; \big(1-2\cdot R_{D'}(G_Q)\big)^2\,.$$

Par une simple réorganisation des termes, nous obtenons comme désiré

$$d_Q^{D'} \quad \leq \quad 2 \cdot R_{D'}(G_Q) \cdot \left(1 - R_{D'}(G_Q)\right) .$$

Enfin, on note que la fonction $F(x) \coloneqq 2x(1-x)$ décrit une parabole dont l'unique maximum se situe au point $F(\frac{1}{2}) = \frac{1}{2}$, ce qui permet d'affirmer $d_Q^{D'} = \frac{1}{2} \Rightarrow R_{D'}(G_Q) = \frac{1}{2}$. □

La proposition 3.5 indique que l'espérance de désaccord $d_Q^{D'}$ possède nécessairement une valeur faible lorsque le risque de Gibbs $R_{D'}(G_Q)$ est très faible (près de 0) ou très grand (près de 1). Nous utiliserons ce résultat au chapitre 4 lorsque nous présenterons une borne PAC-bayésienne reposant à la fois sur le risque de Gibbs et l'espérance de désaccord (voir la section 4.5.6).

3.5 La borne du facteur deux

Il est bien connu dans la littérature PAC-bayésienne que le risque de Bayes est inférieur ou égal à deux fois le risque de Gibbs (voir Langford et Shawe-Taylor, 2002; McAllester, 2003b; Germain et al., 2009a). Il existe plusieurs façons de démontrer ce résultat, et celle introduite dans la démonstration de la proposition 3.6 ci-bas met en lumière le rôle de la marge dans la relation liant les deux risques. En effet, nous procédons en bornant supérieurement le premier moment de la marge à l'aide de l'***inégalité de Markov*** (lemme A.1, en annexe). Dans ce document, nous référons souvent à ce résultat sous le vocable de ***borne du facteur deux***.

Remarque sur la démonstration de la proposition 3.6. Dans la littérature PAC-bayésienne, l'*inégalité de Markov* n'est pas fréquemment évoquée pour démontrer la *borne du facteur deux* ci-dessous. Nous empruntons cette idée à Lacasse (2010), mais nous l'adaptons à notre analyse centrée sur la marge du vote de majorité. Comme nous verrons à la sous-section 3.6, notre démonstration de la proposition 3.6 permet de comprendre en quoi la $\mathcal{C}$*-borne* est une extension naturelle de la *borne du facteur deux*.

Proposition 3.6 (Borne du facteur deux)**.** *Pour toute distribution de probabilité Q sur un ensemble de votants et pour toute distribution D' sur $\mathcal{X} \times \mathcal{Y}$, on a*

$$R_{D'}(B_Q) \;\leq\; 2 \cdot R_{D'}(G_Q) .$$

Démonstration. Le résultat est obtenu à partir de l'équation (3.8), sur laquelle nous appliquons l'inégalité de Markov (lemme A.1, avec $X \coloneqq 1 - M_Q(x,y)$ et $a \coloneqq 1$) :

$$\begin{aligned}
R_{D'}(B_Q) &= \Pr_{(x,y)\sim D'}\Big(M_Q(x,y) \leq 0\Big) \\
&= \Pr_{(x,y)\sim D'}\Big(1 - M_Q(x,y) \geq 1\Big) \\
&\leq \mathop{\mathbf{E}}_{(x,y)\sim D'}\Big(1 - M_Q(x,y)\Big) && \langle\ \textit{Inégalité de Markov}\ \rangle \\
&= 1 - \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y) \\
&= 1 - \mu_1(M_Q^{D'}) \\
&= 2 \cdot R_{D'}(G_Q)\,.
\end{aligned}$$

La dernière égalité est une conséquence directe de l'équation (3.9). □

La démonstration de la proposition 3.6 ci-haut permet constater qu'on obtient une borne du risque de Bayes $R_{D'}(B_Q)$ en considérant seulement le premier moment de la marge $\mu_1(M_Q^{D'})$. Il parait donc naturel d'adapter ce résultat pour prendre en compte les moments supérieurs.

3.6 La $\mathcal{C}$-borne

Reprenons l'idée de la démonstration de la proposition 3.6, mais en utilisant cette fois-ci l'***inégalité de Cantelli-Tchebychev***[2] (théorème A.3, en annexe) au lieu de l'inégalité de Markov. Cela permet de tenir compte des deux premiers moments de la marge, $\mu_1(M_Q^{D'})$ et $\mu_2(M_Q^{D'})$. Nous nommons ce résultat la ***$\mathcal{C}$-borne***.

Notons que la $\mathcal{C}$-borne fut initialement présentée par Lacasse et al. (2006); Lacasse (2010), sous une forme légèrement différente.[3] On peut en effet formuler la $\mathcal{C}$-borne de plusieurs manières, en exploitant les liens entre la variance de la marge, les moments de la marge, le risque de Gibbs et l'espérance de désaccord. Le théorème 3.7 (ci-bas) présente trois formulations de la $\mathcal{C}$-borne, chacune d'elle mettant en lumière une certaine propriété ou un certain comportement de la borne. La figure 3.1 (page 51) illustre ces comportements.

2. L'*inégalité de Cantelli-Tchebychev* fournit une borne supérieure sur la valeur de la variable aléatoire (dans la littérature anglaise, on la désigne parfois par *one-sided Chebyshev inequality*). Nous préférons ce résultat à celui, connu sous le nom d'*inégalité de Bienaymé-Tchebychev* ou simplement d'*inégalité de Tchebychev*, qui fournit à la fois une borne inférieure et supérieure (d'où le terme *two-sided Chebyshev inequality* en anglais), car la borne supérieure obtenue par l'*inégalité de Cantelli-Tchebychev* est légèrement plus serrée que celle obtenue par l'*inégalité de Bienaymé-Tchebychev*.

3. Le théorème A.10, en annexe, présente la $\mathcal{C}$-borne dans la forme employée par Lacasse (2010). Cette dernière est formulée pour le cas où les votants sont des classificateurs binaires (nous l'avons généralisée aux régresseurs) et en fonction de la variable aléatoire W_Q – présentée par l'équation (A.3) – au lieu de M_Q.

Théorème 3.7 (La $\mathcal{C}$-borne)**.** *Pour toute distribution de probabilité Q sur un ensemble de votants et pour toute distribution D' sur $\mathcal{X}\times\mathcal{Y}$, si $\mu_1(M_Q^{D'}) > 0$ (c'est-à-dire $R_{D'}(G_Q) < 1/2$), on a*

$$R_{D'}(B_Q) \;\leq\; \mathcal{C}_Q^{D'},$$

où

$$\mathcal{C}_Q^{D'} \stackrel{\text{def}}{=} \underbrace{\frac{\mathrm{Var}(M_Q^{D'})}{\mu_2(M_Q^{D'})}}_{\textbf{Première forme}} = \underbrace{1 - \frac{\left(\mu_1(M_Q^{D'})\right)^2}{\mu_2(M_Q^{D'})}}_{\textbf{Deuxième forme}} = \underbrace{1 - \frac{\left(1 - 2\cdot R_{D'}(G_Q)\right)^2}{1 - 2\cdot d_Q^{D'}}}_{\textbf{Troisième forme}}.$$

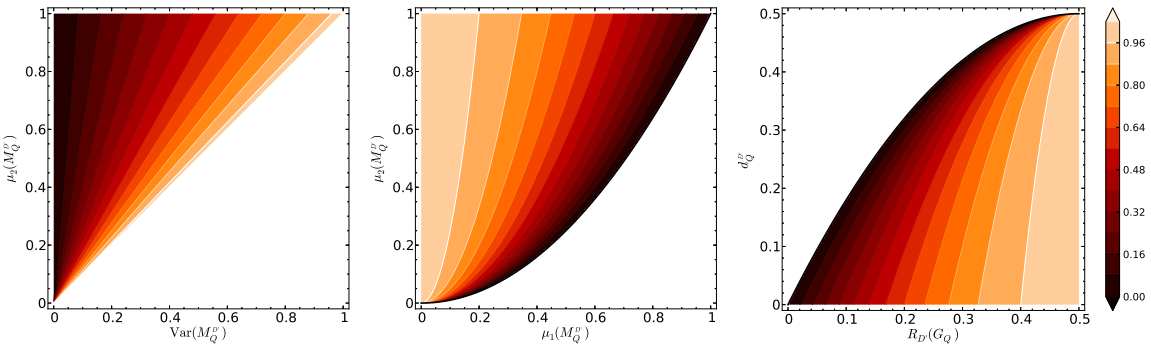
Démonstration. Le résultat est obtenu à partir de l'équation (3.8), sur laquelle nous appliquons l'inégalité de Cantelli-Tchebychev (théorème A.3), avec

$$X \;\coloneqq\; -M_Q(x,y) \quad \text{et} \quad a \;\coloneqq\; \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\,.$$

Ainsi,

$$\begin{aligned}
R_{D'}(B_Q) &= \Pr_{(x,y)\sim D'}\Big(M_Q(x,y) \leq 0\Big) \\
&= \Pr_{(x,y)\sim D'}\left(-M_Q(x,y) + \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y) \geq \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\right) \\
&\leq \frac{\mathop{\mathbf{Var}}_{(x,y)\sim D'}(M_Q(x,y))}{\mathop{\mathbf{Var}}_{(x,y)\sim D'}(M_Q(x,y)) + \left(\mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\right)^2} \quad \langle \textit{ Inégalité de Cantelli-Tchebychev} \rangle \\
&= \frac{\mathrm{Var}(M_Q^{D'})}{\mu_2(M_Q^{D'}) \;-\left(\mu_1(M_Q^{D'})\right)^2 + \left(\mu_1(M_Q^{D'})\right)^2} \;=\; \frac{\mathrm{Var}(M_Q^{D'})}{\mu_2(M_Q^{D'})} \qquad (3.11) \\
&= \frac{\mu_2(M_Q^{D'}) \;-\left(\mu_1(M_Q^{D'})\right)^2}{\mu_2(M_Q^{D'})} \\
&= 1 - \frac{\left(\mu_1(M_Q^{D'})\right)^2}{\mu_2(M_Q^{D'})} \qquad (3.12) \\
&= 1 - \frac{\left(1 - 2\cdot R_{D'}(G_Q)\right)^2}{1 - 2\cdot d_Q^{D'}}\,. \qquad (3.13)
\end{aligned}$$

La ligne (3.11) présente la *première forme* de $\mathcal{C}_Q^{D'}$, tandis que la ligne (3.12) présente la *deuxième forme* de $\mathcal{C}_Q^{D'}$. Ces résultats découlent directement des équations (3.5), (3.6) et (3.7), c'est-à-dire des définitions de $\mu_1(M_Q^{D'})$, $\mu_2(M_Q^{D'})$, et $\mathrm{Var}(M_Q^{D'})$. La *troisième forme* de $\mathcal{C}_Q^{D'}$ est obtenue à la ligne (3.13), à l'aide des équations (3.9) et (3.10). □



(a) Première forme (b) Deuxième forme (c) Troisième forme

FIGURE 3.1 – Courbes de niveaux de la $\mathcal{C}$-borne.

La première forme de la $\mathcal{C}$-borne met en évidence que sa valeur est toujours positive, car la variance et le second moment de la marge ne peuvent pas être négatifs. Similairement, la deuxième forme montre que sa valeur n'est jamais supérieure à 1.

La troisième forme de la $\mathcal{C}$-borne, de même que la figure 3.1c, nous renseigne sur son comportement en fonction du risque de Gibbs et de l'espérance de désaccord. Pour une espérance de désaccord constante, la $\mathcal{C}$-borne décroît lorsque le risque de Gibbs diminue. Or, pour un risque de Gibbs constant, la $\mathcal{C}$-borne décroît lorsque le désaccord augmente. Ainsi, la $\mathcal{C}$-borne suggère que, pour obtenir un faible risque de Bayes, on doit effectuer un compromis entre ces deux quantités lors du choix de la distribution Q pondérant les votants. Cette information n'est pas prise en compte par la borne du facteur deux (proposition 3.6), qui repose seulement sur le risque de Gibbs, c'est-à-dire uniquement sur le premier moment de la marge du vote de majorité.

3.7 Les conditions d'optimalité de la $\mathcal{C}$-borne

Attardons-nous maintenant à l'étude des conditions sous lesquelles la $\mathcal{C}$-borne est *optimale*. Nous disons ici que la borne est *optimale* lorsque, considérant uniquement les deux premiers moments de la marge d'un vote de majorité B_Q, il est possible que la valeur donnée par la $\mathcal{C}$-borne soit égale au risque de Bayes. Autrement dit, il est possible d'avoir $\mathcal{C}_Q^{D'} = R_{D'}(B_Q)$.

Dans le texte qui suit, nous étudions le comportement de la $\mathcal{C}$-borne en fonction d'une variable aléatoire «abstraite» M, représentant une distribution de marges dont les deux premiers moments sont respectivement $\mu_1(M)$ et $\mu_2(M)$. De cette manière, nous faisons abstraction de la distribution Q sur $\mathcal{H}$, ainsi que la distribution D' sur $\mathcal{X} \times \mathcal{Y}$. En notant B_M le vote de majorité relié aux marges M, nous avons, par l'équation (3.8),

$$R(B_M) \stackrel{\text{def}}{=} \Pr\left(M \leq 0\right). \tag{3.14}$$

Aussi, $R(B_M)$ est borné supérieurement par $\mathcal{C}_M$ (la $\mathcal{C}$-borne énoncée par le théorème 3.7) :

$$\mathcal{C}_M \stackrel{\text{def}}{=} 1 - \frac{\big(\,\mu_1(M)\,\big)^2}{\mu_2(M)}\,. \tag{3.15}$$

La proposition 3.8 ci-bas précise sous quelles conditions la $\mathcal{C}$-borne est atteinte. Notons que cela survient lorsqu'il existe une variable aléatoire, possédant les mêmes deux premiers moments que la distribution de la marge, telle que l'inégalité de Cantelli-Tchebychev (théorème A.3) est atteinte.

Proposition 3.8. *Soit M une variable aléatoire représentant la marge d'un vote de majorité. Alors il existe une variable aléatoire $\widetilde{M}$ telle que*

$$\mu_1(\widetilde{M}) = \mu_1(M)\,, \quad \mu_2(\widetilde{M}) = \mu_2(M) \quad \textit{et} \quad \mathcal{C}_{\widetilde{M}} = \mathcal{C}_M = R(B_{\widetilde{M}}) \tag{3.16}$$

si et seulement si

$$0 \;<\; \mu_2(M) \;\leq\; \mu_1(M)\,. \tag{3.17}$$

Démonstration. Montrons d'abord que la ligne (3.17) implique la ligne (3.16). On suppose M tel que $0 < \mu_2(M) \leq \mu_1(M)$, et on considère que $\widetilde{M}$ est concentré en deux points :

$$\widetilde{M} \;:=\; \begin{cases} 0 & \text{avec probabilité } \mathcal{C}_M = 1 - \dfrac{\big(\mu_1(M)\big)^2}{\mu_2(M)}\,, \\[2ex] \dfrac{\mu_2(M)}{\mu_1(M)} & \text{avec probabilité } 1 - \mathcal{C}_M = \dfrac{\big(\mu_1(M)\big)^2}{\mu_2(M)}\,. \end{cases}$$

Nous avons bien $\mu_1(\widetilde{M}) = \mu_1(M)$ et $\mu_2(\widetilde{M}) = \mu_2(M)$, car

$$\mu_1(\widetilde{M}) = \frac{\big(\mu_1(M)\big)^2}{\mu_2(M)}\left[\frac{\mu_2(M)}{\mu_1(M)}\right] = \mu_1(M) \quad \text{et} \quad \mu_2(\widetilde{M}) = \frac{\big(\mu_1(M)\big)^2}{\mu_2(M)}\left[\frac{\mu_2(M)}{\mu_1(M)}\right]^2 = \mu_2(M)\,.$$

De plus, il découle de l'équation (3.15) que $\mathcal{C}_{\widetilde{M}} = \mathcal{C}_M$. Finalement, par le choix de $\widetilde{M}$, on a $\Pr(\widetilde{M} = 0) = \mathcal{C}_M$ (car $\frac{\mu_2(M)}{\mu_1(M)} > 0$), et l'équation (3.14) permet de conclure

$$R(B_{\widetilde{M}}) \;=\; \Pr\left(\widetilde{M} \leq 0\right) \;=\; \mathcal{C}_M\,.$$

Maintenant, montrons que la ligne (3.16) implique la ligne (3.17). On suppose $\widetilde{M}$ tel que les égalités de la ligne (3.16) sont respectées. Par la proposition 3.6 et l'équation (3.9), on a $\mathcal{C}_M \leq 1 - \mu_1(M)$, car

$$\mathcal{C}_M \;=\; R(B_{\widetilde{M}}) \;\leq\; 2 \times \frac{1}{2}\Big(1 - \mu_1(\widetilde{M})\Big) \;=\; 1 - \mu_1(\widetilde{M}) \;=\; 1 - \mu_1(M)\,.$$

Ainsi, par la définition de $\mathcal{C}_M$, on obtient

$$1 - \frac{\big(\,\mu_1(M)\,\big)^2}{\mu_2(M)} \;\leq\; 1 - \mu_1(M) \quad \Longleftrightarrow \quad 0 \;<\; \mu_2(M) \;\leq\; \mu_1(M)\,.$$

□

Nous avons montré à la section 3.4 qu'il existe plusieurs liens entre les moments de la marge, le risque de Gibbs et l'espérance de désaccord. La prochaine proposition exploite ces liens pour énoncer des expressions équivalentes à la ligne (3.17) de la proposition 3.8. Ce faisant, nous obtenons trois conditions (équivalentes) sous lesquelles la $\mathcal{C}$-borne est optimale.

Proposition 3.9. *Pour toute distribution Q sur un ensemble de votants et pour toute distribution D' sur $\mathcal{X} \times \mathcal{Y}$, si $\mu_1(M_Q^{D'}) > 0$ (c'est-à-dire si $R_{D'}(G_Q) < 1/2$), alors les trois expressions suivantes sont équivalentes :*

(i) $\mu_2(M_Q^{D'}) \;\leq\; \mu_1(M_Q^{D'})$;

(ii) $R_{D'}(G_Q) \;\leq\; d_Q^{D'}$;

(iii) $\mathcal{C}_Q^{D'} \;\leq\; 2\, R_{D'}(G_Q)$.

Démonstration. Par les équations (3.9) et (3.10), nous obtenons directement $(i) \Leftrightarrow (ii)$. Nous démontrons $(iii) \Leftrightarrow (ii)$ en exprimant $\mathcal{C}_Q^{D'}$ dans sa troisième forme (voir le théorème 3.7) :

$$\mathcal{C}_Q^{D'} \;=\; 1 - \frac{\left(1 - 2\, R_{D'}(G_Q)\right)^2}{1 - 2\, d_Q^{D'}} \;\leq\; 2R_{D'}(G_Q) \quad\Longleftrightarrow\quad R_{D'}(G_Q) \;\leq\; d_Q^{D'}\,.$$

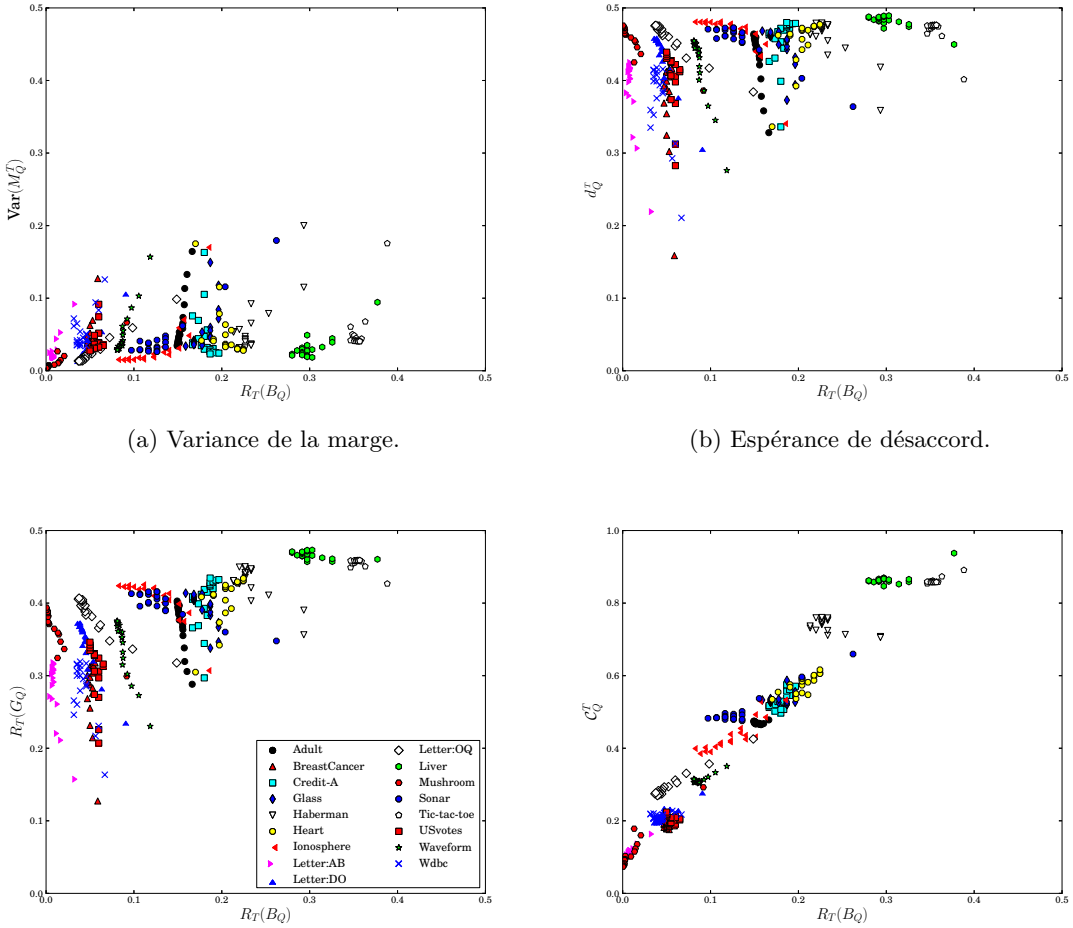
□

Nous déduisons des propositions 3.8 et 3.9(iii) un résultat très intéressant sur la relation qu'entretiennent les deux bornes sur le risque de Bayes présentées plus haut : La $\mathcal{C}$-borne est optimale si est seulement si sa valeur est inférieure ou égale à la borne du facteur deux. Rappelons que la borne du facteur deux, présentée par la proposition 3.6, s'avère un résultat «classique» de la littérature propre à la théorie PAC-bayésienne.

3.8 Étude empirique du comportement de la $\mathcal{C}$-borne

Dans Lacasse et al. (2006) et Lacasse (2010), des expérimentations empiriques montrent que la valeur de la $\mathcal{C}$-borne est un meilleur indicateur du risque du vote de majorité que la borne du facteur deux. Pour obtenir ces résultats, les auteurs construisent des votes de majorité à l'aide de l'algorithme d'apprentissage ***AdaBoost*** sur plusieurs échantillons de données provenant d'une grande variété problèmes réels. Nous effectuons ici des expérimentations similaires afin de montrer la relation qu'il existe entre le risque de Bayes $R_{D'}(B_Q)$ et les différentes quantités apparaissant dans les trois formes de la $\mathcal{C}$-borne présentées par le théorème 3.7.

Plus précisément, nous étudions comment les valeurs de $\mathrm{Var}(M_Q^{D'})$, $d_Q^{D'}$, $R_{D'}(G_Q)$ et $\mathcal{C}_Q^{D'}$ sont respectivement corrélés avec $R_{D'}(B_Q)$. Ces quatre quantités interviennent dans le calcul de la première ou de la troisième forme de la $\mathcal{C}$-borne. Nous ne présentons pas les résultats obtenus à l'aide des moments $\mu_1(M_Q^{D'})$ et $\mu_2(M_Q^{D'})$, intervenant dans le calcul de la deuxième forme de la $\mathcal{C}$-borne, puisque la relation entre $\mu_1(M_Q^{D'})$ et $R_{D'}(G_Q)$, de même que la relation entre $\mu_2(M_Q^{D'})$ et $d_Q^{D'}$, est linéaire.



(a) Variance de la marge.

(b) Espérance de désaccord.

(c) Risque de Gibbs.

(d) $\mathcal{C}$-borne.

FIGURE 3.2 – Valeur de $R_T(B_Q)$ en fonction de $\mathrm{Var}(M_Q^T)$, d_Q^T , $R_T(G_Q)$ et $\mathcal{C}_Q^T$ respectivement.

Les résultats présentés par la figure 3.2 sont obtenus en exécutant l'algorithme *AdaBoost*, dans la version décrite à la section 2.4.2. L'ensemble de votants $\mathcal{H}$ utilisé est constitué de *souches de décision* (la méthodologie utilisée pour créer cet ensemble de votants est décrite à l'annexe B.2). La distribution Q correspond au vote de majorité construit par AdaBoost, en normalisant les poids donnés par l'algorithme, comme il est montré à l'équation (2.28).

Nous exécutons l'algorithme d'apprentissage sur dix-sept échantillons de données provenant du «UCI Machine Learning Repository» (Bache et Lichman, 2013). Chaque échantillon de données est séparé en deux sous-ensembles : un échantillon d'entraînement S et un échantillon de test T. Nous exécutons l'algorithme AdaBoost sur l'échantillon d'entraînement S pendant 100 itérations. Nous calculons les valeurs de $\mathrm{Var}(M_Q^T)$, d_Q^T, $R_T(G_Q)$ et $\mathcal{C}_Q^T$, sur l'échantillon de test T, à toutes les 5 itérations. Conséquemment, nous étudions 20 votes de majorité par échantillon de données, chacun correspondant à un point sur les graphiques de la figure 3.2.

En examinant les figures 3.2a et 3.2b, nous ne remarquons pas de corrélation entre $R_T(B_Q)$ et $\mathrm{Var}(M_Q^T)$, ou entre $R_T(B_Q)$ et d_Q^T. nous pouvons seulement affirmer que, généralement,

$$R_T(B_Q) > \mathrm{Var}(M_Q^T) \quad \text{et} \quad R_T(B_Q) < d_Q^T .$$

En examinant la figure 3.2c, nous remarquons que tous les classificateurs produits respectent l'inégalité

$$R_T(B_Q) < R_T(G_Q) .$$

Rappelons que, par la *borne du facteur deux* (proposition 3.6), on a nécessairement

$$R_T(B_Q) \leq 2\, R_T(G_Q) .$$

À la lumière de nos expérimentations, la borne du facteur deux ne semble donc pas très précise. De plus, la figure 3.2c ne montre pas de corrélation évidente entre $R_T(B_Q)$ et $R_T(G_Q)$.

À l'inverse, la figure 3.2d montre une forte corrélation entre $\mathcal{C}_Q^T$ et $R_T(B_Q)$. En fait, la relation est presque linéaire. De plus, la valeur de $\mathcal{C}_Q^T$ est généralement très près de la valeur de $R_T(B_Q)$. Cela semble constituer une évidence empirique que la $\mathcal{C}$-borne reflète le comportement du risque de Bayes, tandis que les quantités contenues dans la $\mathcal{C}$-borne – analysées par les figures 3.2a , 3.2b et 3.2c – sont insuffisantes lorsque considérées individuellement. Insistons sur le fait que cela inclut le risque de Gibbs, qui est souvent utilisé pour borner le risque de Bayes par la borne du facteur deux (proposition 3.6).

Enfin, soulignons que Roy et al. (2011) ont conçu un algorithme d'apprentissage qui minimise explicitement l'expression de la $\mathcal{C}$-borne. Les excellents résultats empiriques ainsi obtenus renforcent l'idée que la $\mathcal{C}$-borne est un outil approprié pour l'étude des votes de majorité.

3.9 Synthèse des contributions du chapitre

Dans ce chapitre, nous avons étudié le vote de majorité en considérant la marge sur un exemple $(x, y) \sim D'$ comme la réalisation d'une variable aléatoire $M_Q^{D'}$. Nous avons exprimé le risque de Gibbs $R_{D'}(G_Q)$ en termes du premier moment de la marge, et l'espérance de désaccord $d_Q^{D'}$ en termes du deuxième moment de la marge $\mu_2(M_Q^{D'})$.

Notre approche exhibe que la *borne du facteur deux*, couramment invoquée pour borner supérieurement le risque de Bayes $R_{D'}(B_Q)$ dans la théorie PAC-bayésienne, ne prend en compte que le premier moment de la marge, par l'entremise de l'inégalité de Markov. Il devient alors naturel de démontrer la $\mathcal{C}$-borne de Lacasse et al. (2006) à l'aide des deux premiers moments de la marge et de l'inégalité de Cantelli-Tchebychev.

Enfin, nous avons présenté certaines conditions d'optimalité de la $\mathcal{C}$-borne. Par cette analyse théorique, ainsi que par quelques résultats empiriques, nous réaffirmons son importance dans la compréhension des votes de majorité en apprentissage automatique.

Chapitre 4

Théorie PAC-bayésienne pour l'apprentissage inductif

Le contenu de ce chapitre se retrouve dans un article publié dans la revue *Journal of Machine Learning Research* (Germain et al., 2015b).

Résumé. Ce chapitre présente plusieurs résultats PAC-bayésiens propres au cadre d'apprentissage inductif. Même si bon nombre de ces résultats proviennent de la littérature, nous montrons qu'ils constituent plusieurs déclinaisons d'un théorème PAC-bayésien général, valide pour toute espérance de perte. Nous présentons aussi le concept de *votant jumelé*, qui permet de borner l'espérance de désaccord et d'obtenir des garanties de généralisation basées sur la $\mathcal{C}$-borne du théorème 3.7.

4.1 Éléments de la théorie PAC-bayésienne

Dans le contexte de l'apprentissage PAC-bayésien, un ***algorithme d'apprentissage*** peut être vu comme une fonction qui reçoit en entrée un ***échantillon d'entraînement*** S, un ***ensemble de votants*** $\mathcal{H}$ (discret ou continu), ainsi qu'une distribution *a priori* P sur $\mathcal{H}$, et retourne une distribution *a posteriori* Q sur $\mathcal{H}$. On appelle aussi la distribution P le ***prior*** et la distribution Q le ***posterior***. Le *prior* P permet d'«encoder» les informations que l'on possède sur le problème avant l'apprentissage. Selon le contexte, le *posterior* Q décrit le classificateur de Gibbs G_Q, ou encore le vote de majorité associé B_Q.

Dans le cadre d'***apprentissage inductif***, on fait l'hypothèse que les exemples de l'échantillon d'apprentissage S sont générés de manière indépendante et qu'ils sont identiquement distribués (***i.i.d.***) selon la distribution génératrice D sur $\mathcal{X} \times \mathcal{Y}$. Ainsi,

$$S \coloneqq \{(x_i, y_i)\}_{i=1}^{m} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \sim D^m .$$

De plus, la théorie PAC-bayésienne décrite dans ce chapitre nécessite que les votants de l'ensemble $\mathcal{H}$ et la distribution *a priori* P soient choisis indépendamment des exemples de l'échantillon d'entraînement S. Si ces conditions sont respectées, de même que l'hypothèse que les exemples sont générés de manière *i.i.d.*, la ***théorie PAC-bayésienne*** «classique» permet de borner le risque du classificateur de Gibbs G_Q,

$$R_D(G_Q) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}_\ell}(f) \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D} \mathop{\mathbf{E}}_{f\sim Q} \mathcal{L}_\ell(f(x),y)\,,$$

à partir de deux «ingrédients» principaux :

a) le ***risque empirique*** du classificateur G_Q, c'est-à-dire le risque de Gibbs observé sur un échantillon d'apprentissage

$$R_S(G_Q) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}_\ell}(f) \;=\; \frac{1}{m}\sum_{i=1}^{m} \mathop{\mathbf{E}}_{f\sim Q} \mathcal{L}_\ell(f(x_i),y_i)\,;$$

b) la ***divergence Kullback-Leibler*** entre les distributions Q et P, mesurant la «distance» entre le *prior* et le *posterior*

$$\mathrm{KL}(Q\|P) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{f\sim Q} \ln \frac{Q(f)}{P(f)}\,. \tag{4.1}$$

Rappelons que toute borne sur le risque du classificateur de Gibbs peut être convertie directement en borne sur le risque du vote de majorité (ou risque de Bayes) à l'aide de la borne du facteur deux (proposition 3.6) :

$$R_D(B_Q) \;\leq\; 2\cdot R_D(G_Q)\,.$$

La section 4.2 ci-bas présente la théorie PAC-bayésienne dans un contexte plus général. En effet, plutôt que d'étudier seulement le risque de Gibbs, nous désirons borner toute espérance de perte pondérée selon un *posterior* Q.

4.2 Théorie PAC-bayésienne générale

Cette section présente un premier théorème PAC-bayésien, qui se révèle la base de plusieurs résultats présentés dans ce document. Étant donné une fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0,1]$, notre théorie générale permet de borner l'espérance de perte sur la distribution génératrice des données D, c'est-à-dire

$$\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}}(f) \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D} \mathop{\mathbf{E}}_{f\sim Q} \mathcal{L}(f(x),y)\,,$$

à partir de l'espérance de perte empirique

$$\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}}(f) \;=\; \frac{1}{m}\sum_{i=1}^{m} \mathop{\mathbf{E}}_{f\sim Q} \mathcal{L}(f(x_i),y_i)\,.$$

Cette théorie générale permet notamment de retrouver des bornes sur le risque de Gibbs en utilisant la fonction de perte linéaire – autrement dit en fixant $\mathcal{L} := \mathcal{L}_\ell$ – comme il est montré à la section 4.3. Cette même théorie permet, à la section 4.4, de borner l'espérance de désaccord.

4.2.1 Espérance de perte zéro-un et loi binomiale

Une des clés de la démonstration du théorème PAC-bayésien dans le cadre d'apprentissage inductif est le fait que l'espérance de ***perte zéro-un*** (définition 2.1, page 16) s'exprime à l'aide d'une ***loi binomiale***. Afin d'expliquer cela, considérons une distribution de données D sur $\mathcal{X} \times \{-1, 1\}$ et un classificateur binaire $h : \mathcal{X} \to \{-1, 1\}$. L'espérance de perte zéro-un de $h(\cdot)$ sur D est $R_D(h) = \mathbb{E}_D^{\mathcal{L}_{01}}(h)$. Associons le classificateur h à une variable aléatoire binomiale X_h de m essais avec probabilité de succès $R_D(h)$, ce que nous désignons ainsi :

$$X_h \sim B\big(m, R_D(h)\big)\,.$$

Conséquemment, grâce à la fonction de masse de la loi binomiale, la probabilité d'observer spécifiquement k erreurs parmi m exemples est donnée par

$$\Pr_{S\sim D^m}\Big(R_S(h) = \tfrac{k}{m}\Big) \;=\; \Pr_{X_h \sim B(m,\, R_D(h))}\Big(X_h = k\Big) \;=\; \binom{m}{k}\Big(R_D(h)\Big)^k\Big(1 - R_D(h)\Big)^{m-k},$$

pour tout $k \in \{0, \ldots, m\}$.

Grâce à un résultat exprimé par Maurer (2004) – voir le lemme A.12 en annexe – nous généralisons ce principe aux votants à valeur réelle $f : \mathcal{X} \to [-1, 1]$ et à toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0, 1]$. Le lemme 4.1 ci-dessous présente ce résultat, faisant intervenir la loi binomiale, dans une forme qui permettra de l'invoquer facilement dans la démonstration du théorème PAC-bayésien.

Lemme 4.1. *Soit une distribution D sur $\mathcal{X} \times \mathcal{Y}$, une fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0, 1]$, un votant $f : \mathcal{X} \to \overline{\mathcal{Y}}$ d'espérance de perte $\mathbb{E}_D^{\mathcal{L}}(f)$ et une fonction convexe $g : [0, 1] \to \mathbb{R}$. On a*

$$\mathop{\mathbf{E}}_{S\sim D^m} g\Big(\mathbb{E}_S^{\mathcal{L}}(f)\Big) \;\le\; \sum_{k=0}^{m}\binom{m}{k}\Big(\mathbb{E}_D^{\mathcal{L}}(f)\Big)^k\Big(1 - \mathbb{E}_D^{\mathcal{L}}(f)\Big)^{m-k} \times g\Big(\tfrac{k}{m}\Big)\,.$$

Démonstration. Considérons une variable aléatoire X_f, suivant une distribution binomiale de m essais, et dont la probabilité de succès est $R \coloneqq \mathbb{E}_D^{\mathcal{L}}(f)$. Ainsi, $X_f \sim B(m, R)$. Comme $g(\cdot)$ est une fonction convexe, le lemme A.12 (Maurer, 2004) montre que

$$\begin{aligned}
\mathop{\mathbf{E}}_{S\sim D^m} g\Big(\mathbb{E}_S^{\mathcal{L}}(f)\Big) \;&\le\; \mathop{\mathbf{E}}_{X_f \sim B(m,R)} g\Big(\tfrac{1}{m}X_f\Big) \\
&=\; \sum_{k=0}^{m}\Pr_{X_f\sim B(m,R)}\Big(X_f = k\Big) \times g\Big(\tfrac{1}{m}X_f\Big) \\
&=\; \sum_{k=0}^{m}\binom{m}{k}\Big(R\Big)^k\Big(1 - R\Big)^{m-k} \times g\Big(\tfrac{k}{m}\Big)\,.
\end{aligned}$$

La dernière égalité découle de la fonction de masse de la loi binomiale. □

4.2.2 Théorème général pour fonctions de perte à valeurs réelles

Le théorème 4.4 présenté dans cette section est inspiré du théorème général présenté dans Germain et al. (2009a). L'apport de ce dernier est d'exprimer une conception générale de la théorie PAC-bayésienne, à l'aide d'une fonction de «distance» entre la perte empirique et la perte sur D. Dans la présente thèse, nous référons à ce type de fonction par le terme Δ-fonction.

Définition 4.2. Une ***Δ-fonction*** désigne une fonction convexe $\Delta : [0,1] \times [0,1] \to \mathbb{R}$.

Une étape clé de la plupart des démonstrations PAC-bayésiennes est synthétisée par l'***inégalité du changement de mesure*** ci-bas.

Remarque sur la démonstration du lemme 4.3. Notre démonstration de l'*inégalité de changement de mesure* est fortement inspirée de celles de Seldin et Tishby (2010) et McAllester (2013), à l'exception du détail mentionné en note de bas de page. Notons que le même résultat est dérivé de l'*inégalité de Fenchel* dans Banerjee (2006) et de la *formule variationnelle de Donsker-Varadhan pour l'entropie relative* dans Seldin et al. (2012); Tolstikhin et Seldin (2013).

Lemme 4.3 (Inégalité de changement de mesure)**.** *Pour tout ensemble $\mathcal{H}$, pour toutes distributions P et Q sur $\mathcal{H}$, et pour toute fonction mesurable $\phi : \mathcal{H} \to \mathbb{R}$, on a*

$$\mathop{\mathbf{E}}_{f\sim Q} \phi(f) \;\leq\; \mathrm{KL}(Q\|P) + \ln\left(\mathop{\mathbf{E}}_{f\sim P} e^{\phi(f)}\right).$$

Démonstration. Le résultat est obtenu par la définition de la divergence Kullback-Leibler (équation (4.1)) et l'inégalité de Jensen (lemme A.2, en annexe) sur la fonction concave $\ln(\cdot)$:

$$\begin{aligned}
\mathop{\mathbf{E}}_{f\sim Q} \phi(f) &= \mathop{\mathbf{E}}_{f\sim Q} \ln e^{\phi(f)} = \mathop{\mathbf{E}}_{f\sim Q} \ln\left(\frac{Q(f)}{P(f)} \cdot \frac{P(f)}{Q(f)} \cdot e^{\phi(f)}\right) \\
&= \mathop{\mathbf{E}}_{f\sim Q} \ln\left(\frac{Q(f)}{P(f)}\right) + \mathop{\mathbf{E}}_{f\sim Q} \ln\left(\frac{P(f)}{Q(f)} \cdot e^{\phi(f)}\right) \\
&= \mathrm{KL}(Q\|P) + \mathop{\mathbf{E}}_{f\sim Q} \ln\left(\frac{P(f)}{Q(f)} \cdot e^{\phi(f)}\right) && \langle\ \textit{Définition de } \mathrm{KL}(Q\|P)\ \rangle \\
&\leq \mathrm{KL}(Q\|P) + \ln\left(\mathop{\mathbf{E}}_{f\sim Q} \frac{P(f)}{Q(f)} \cdot e^{\phi(f)}\right) && \langle\ \textit{Inégalité de Jensen}\ \rangle \\
&\leq \mathrm{KL}(Q\|P) + \ln\left(\mathop{\mathbf{E}}_{f\sim P} e^{\phi(f)}\right).
\end{aligned}$$

Notons que la dernière inégalité, où l'on transforme l'espérance sur Q en espérance sur P, devient une égalité lorsque les distributions Q et P possèdent le même support. [1] □

1. Dans les démonstrations de Seldin et Tishby (2010) et McAllester (2013), cette inégalité est présentée

Nous possédons maintenant tous les éléments nécessaires à la compréhension du théorème 4.4. Ce résultat est encore plus général que celui présenté dans Germain et al. (2009a), qui s'intéresse uniquement à la fonction de perte linéaire, et donc qui exprime une borne sur le risque de Gibbs (voir la définition 3.2).

Remarque sur la démonstration du théorème 4.4. Bien que notre démonstration du théorème PAC-bayésien regroupe tous les éléments des démonstrations PAC-bayésiennes habituelles, un travail particulier a été fait pour en simplifier la démarche. Il en ressort une démonstration «ordonnée», c'est-à-dire qu'elle se présente comme une succession d'étapes. Chacune de ces étapes illustre une idée au cœur de la théorie PAC-bayésienne. Nous mettons ainsi en évidence chacune des approximations requises pour obtenir le résultat final. En plus de faciliter (à notre avis) la compréhensibilité de la démonstration, cela a l'avantage d'en exhiber la «modularité». En effet, la plupart des variations du théorème PAC-bayésien présentées plus loin dans la thèse sont obtenues en modifiant seulement une étape de la démonstration ci-dessous. Autrement dit, on peut intervenir à l'une ou l'autre des étapes de la démonstration pour adapter le théorème selon nos besoins. Ce sera notamment le cas dans le cadre d'apprentissage transductif discuté au chapitre 5.

Théorème 4.4. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X}\to\overline{\mathcal{Y}}$, pour toute fonction de perte $\mathcal{L}:\overline{\mathcal{Y}}\times\mathcal{Y}\to[0,1]$, pour toute distribution P sur $\mathcal{H}$, pour tout $\delta\in(0,1]$, et pour toute fonction convexe $\Delta:[0,1]\times[0,1]\to\mathbb{R}$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$,*

$$\forall\, Q \text{ sur } \mathcal{H}: \quad \Delta\Big(\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f),\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)\Big) \;\le\; \frac{1}{m}\Big[\mathrm{KL}(Q\|P)+\ln\frac{\mathcal{I}_\Delta(m)}{\delta}\Big], \tag{4.2}$$

où

$$\mathcal{I}_\Delta(m) \;\stackrel{\text{def}}{=}\; \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\binom{m}{k}r^k(1-r)^{m-k}e^{m\Delta(\frac{k}{m},r)}\right], \tag{4.3}$$

et où $\mathrm{KL}(Q\|P)$ *est la divergence Kullback-Leibler définie par l'équation* (4.1).

Démonstration. Nous désirons borner supérieurement la quantité $\Delta\Big(\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f),\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)\Big)$.

Nous appliquons d'abord l'inégalité de Jensen (lemme A.2, en annexe) sur la fonction convexe $\Delta(\cdot,\cdot)$, puis l'inégalité du changement de mesure (lemme 4.3) avec $\phi(f)\coloneqq m\cdot\Delta\Big(\mathbb{E}_S^{\mathcal{L}}(f),\mathbb{E}_D^{\mathcal{L}}(f)\Big)$.

comme une égalité. Cependant, ce n'est pas toujours vrai. Par exemple, si $\mathcal{H}\coloneqq\{f_1,f_2\}$, avec $Q(f_1)\coloneqq 1$, $Q(f_2)\coloneqq 0$, $P(f_1)\coloneqq\frac{1}{2}$ et $P(f_2)\coloneqq\frac{1}{2}$, on a

$$\mathop{\mathbf{E}}_{f\sim Q}\frac{P(f)}{Q(f)}\cdot e^{\phi(f)} \;=\; P(f_1)\cdot e^{\phi(f_1)} \;<\; P(f_1)\cdot e^{\phi(f_1)}+P(f_2)\cdot e^{\phi(f_2)} \;=\; \mathop{\mathbf{E}}_{f\sim P}e^{\phi(f)}\,.$$

On obtient, pour tout ensemble S généré par la distribution D,

$$
\begin{aligned}
&\forall\, Q \text{ sur } \mathcal{H} : \\
&\qquad m \cdot \Delta\Big(\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}}(f), \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}}(f) \Big) \\
&\qquad\qquad \leq \quad \mathop{\mathbf{E}}_{f\sim Q} m \cdot \Delta\Big(\mathbb{E}_S^{\mathcal{L}}(f), \mathbb{E}_D^{\mathcal{L}}(f) \Big) && \langle \textit{ Inégalité de Jensen } \rangle \\
&\qquad\qquad \leq \quad \mathrm{KL}(Q\|P) + \ln \Bigg(\underbrace{\mathop{\mathbf{E}}_{f\sim P} e^{m\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f), \mathbb{E}_D^{\mathcal{L}}(f)\right)}}_{X_P(S)} \Bigg) . && \langle \textit{ Changement de mesure } \rangle
\end{aligned}
$$

Considérons maintenant la variable aléatoire $X_P(S) \coloneqq \mathbf{E}_{f\sim P}\, e^{m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f), \mathbb{E}_D^{\mathcal{L}}(f)\right)}$. Par l'inégalité de Markov (lemme A.1, en annexe), on a $\Pr\limits_{S\sim D^m}\left(X_P(S) \leq \frac{1}{\delta} \mathop{\mathbf{E}}\limits_{S'\sim D^m} X_P(S') \right) \geq 1-\delta$.

Ainsi, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$,

$$
\forall\, Q \text{ sur } \mathcal{H} : \quad m \cdot \Delta\Big(\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}}(f), \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}}(f) \Big) \;\leq\; \mathrm{KL}(Q\|P) + \ln \frac{1}{\delta} \mathop{\mathbf{E}}_{S'\sim D^m} X_P(S') . \tag{4.4}
$$

Comme la distribution *a priori* P est indépendante de S', on peut inverser l'ordre des espérances dans le terme de droite. Cela nous permet ensuite d'appliquer le lemme 4.1 avec la fonction convexe $g(\cdot) \coloneqq e^{m\Delta(\,\cdot\,, \mathbb{E}_D^{\mathcal{L}}(f))}$:

$$
\begin{aligned}
&\mathop{\mathbf{E}}_{S'\sim D^m} X_P(S') && (4.5)\\
&\quad = \mathop{\mathbf{E}}_{S'\sim D^m} \mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta(\mathbb{E}_{S'}^{\mathcal{L}}(f), \mathbb{E}_D^{\mathcal{L}}(f))} \\
&\quad = \mathop{\mathbf{E}}_{f\sim P} \mathop{\mathbf{E}}_{S'\sim D^m} e^{m\cdot\Delta(\mathbb{E}_{S'}^{\mathcal{L}}(f), \mathbb{E}_D^{\mathcal{L}}(f))} && \langle \textit{ Inversion des espérances } \rangle \\
&\quad \leq \mathop{\mathbf{E}}_{f\sim P} \sum_{k=0}^{m} \binom{m}{k} \Big(\mathbb{E}_D^{\mathcal{L}}(f)\Big)^k \Big(1-\mathbb{E}_D^{\mathcal{L}}(f)\Big)^{m-k} e^{m\cdot\Delta(\frac{k}{m}, \mathbb{E}_D^{\mathcal{L}}(f))} && \langle \textit{ Lemme 4.1 } \rangle \\
&\quad \leq \sup_{r\in[0,1]} \left[\sum_{k=0}^{m} \binom{m}{k} r^k (1-r)^{m-k} e^{m\Delta(\frac{k}{m}, r)} \right] \\
&\quad = \mathcal{I}_\Delta(m) .
\end{aligned}
$$

Le résultat est alors obtenu en substituant $\mathbf{E}_{S'\sim D^m} X_P(S')$ par $\mathcal{I}_\Delta(m)$ dans l'équation (4.4), et en divisant par m des deux côtés de l'inégalité. □

Nous possédons maintenant un outil – le théorème général – permettant d'exprimer plusieurs variantes de bornes PAC-bayésiennes. Il suffit de choisir une Δ-fonction et une fonction de perte $\mathcal{L}$.

Le choix de Δ-fonction influence à la fois le terme de gauche de l'inégalité (4.2) et le terme de droite, en déterminant la valeur du terme $\mathcal{I}_\Delta(m)$. Notons qu'il n'est pas toujours possible de calculer la valeur exacte de ce terme. En effet, comme le montre l'équation (4.3), la valeur

de $\mathcal{I}_\Delta(m)$ repose sur le *supremum* d'une expression selon la variable continue $r \in [0,1]$. En pratique, il faut soit borner supérieurement $\mathcal{I}_\Delta(m)$, ou encore utiliser une Δ-fonction qui rend l'expression à l'intérieur du *supremum* indépendante de la variable r. Comme nous le verrons dans la suite de ce chapitre, les théorèmes connus de la théorie PAC-bayésienne correspondent à des choix de Δ-fonction qui rendent le terme $\mathcal{I}_\Delta(m)$ facilement calculable ou estimable.

Travaux connexes éffectués pendant le doctorat

Au cours du doctorat, nous avons collaboré à développer des théorèmes PAC-bayésiens ne contenant pas de terme «KL($Q\|P$)». Pour parvenir à exprimer de tels résultats, nous devons contraindre la distribution Q respecter une contrainte bien particulière. Nous décrivons cette approche à la section 7.1 (page 144).

Nous avons aussi étudié les votes de majorité où les votants sont exprimés à l'aide des exemples de l'échantillon d'entraînement. Plus précisément, chaque votant est considéré comme un *classificateur comprimé* (tel que décrit à la section 2.3, page 27). Nous décrivons cette approche à la section 7.2 (page 148).

Ces deux aspects sont aussi au cœur de l'article Germain et al. (2011), inclus en annexe C de la thèse.

4.2.3 Une première Δ-fonction : la divergence Kullback-Leibler entre deux distributions de Bernoulli

Pour retrouver une borne PAC-bayésienne similaire à celle proposée par Langford et Seeger (2001); Seeger (2002); Langford (2005), nous utilisons comme Δ-fonction la ***divergence Kullback-Leibler*** entre deux distributions de Bernoulli de probabilité de succès p et q respectivement :

$$\mathrm{kl}(q,p) \stackrel{\text{def}}{=} q \ln \frac{q}{p} + (1-q) \ln \frac{1-q}{1-p} \,. \tag{4.6}$$

Par le corollaire A.5 (en annexe), on sait que $\mathrm{kl}(q,p)$ est une fonction convexe.

Il est intéressant de constater que l'on peut réécrire l'équation (4.6) en termes de l'***entropie*** $H(q)$ d'une distribution de Bernoulli et de l'*entropie conjointe* $H(q,p)$ de deux distributions de Bernoulli (Cover et Thomas, 1991) :

$$\begin{aligned} \mathrm{kl}(q,p) &= H(q,p) - H(q) \,, \\ H(q) &\stackrel{\text{def}}{=} -q \ln q - (1-q) \ln(1-q) \,, \\ H(q,p) &\stackrel{\text{def}}{=} -q \ln p - (1-q) \ln(1-p) \,. \end{aligned} \tag{4.7}$$

En partant de ces définitions, et constatant que $e^{mH(\frac{k}{m},r)} = \frac{1}{r^k(1-r)^{m-k}}$, on obtient

$$\begin{aligned}\mathcal{I}_{\mathrm{kl}}(m) &= \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\binom{m}{k}r^k(1-r)^{m-k}e^{mH(\frac{k}{m},r)-mH(\frac{k}{m})}\right]\\ &= \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\binom{m}{k}e^{-mH(\frac{k}{m})}\right]\\ &= \sum_{k=0}^{m}\psi(k,m)\,, \end{aligned} \tag{4.8}$$

où

$$\psi(a,b) \stackrel{\text{def}}{=} \binom{b}{a}\left(\frac{a}{b}\right)^a\left(1-\frac{a}{b}\right)^{b-a} \;\leq\; 1\,.$$

L'inégalité $\psi(a,b)\leq 1$ est démontrée en annexe par le lemme A.6. Il en découle directement que $\mathcal{I}_{\mathrm{kl}}(m)\leq m+1$. Cette borne est cependant très approximative, car Maurer (2004) a démontré [2]

$$\sqrt{m} \;\leq\; \mathcal{I}_{\mathrm{kl}}(m) \;\leq\; 2\sqrt{m}\,, \tag{4.9}$$

en utilisant l'approximation de Stirling du coefficient binomial.

Le corollaire suivant est une spécialisation du théorème 4.4, avec $\Delta(q,p)\coloneqq \mathrm{kl}(q,p)$. Ce résultat est valide pour toute fonction de perte $\mathcal{L}:\overline{\mathcal{Y}}\times\mathcal{Y}\to[0,1]$. En utilisant à la fonction de perte linéaire, nous retrouverons, à la section 4.3, le théorème PAC-bayésien de Langford et Seeger (2001). Cela dit, le corollaire 4.5 sera aussi un outil essentiel à la section 4.4 pour formuler des garanties sur l'espérance de désaccord.

Corollaire 4.5. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X}\to\overline{\mathcal{Y}}$, pour toute fonction de perte $\mathcal{L}:\overline{\mathcal{Y}}\times\mathcal{Y}\to[0,1]$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta\in(0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$,*

$$\forall\, Q \text{ sur } \mathcal{H}: \quad \mathrm{kl}\left(\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f),\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)\right) \;\leq\; \frac{1}{m}\left[\mathrm{KL}(Q\|P)+\ln\frac{2\sqrt{m}}{\delta}\right].$$

Démonstration. Le résultat découle directement du théorème 4.4, avec $\Delta(q,p)\coloneqq\mathrm{kl}(q,p)$ et l'équation (4.9). □

Pour une fonction de perte $\mathcal{L}(\cdot,\cdot)$ donnée, le corollaire 4.5 permet donc de borner la perte généralisée $\mathbf{E}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)$ à partir de l'espérance de perte empirique $\mathbf{E}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f)$. Par contre, la borne obtenue ne possède pas de forme explicite. Pour obtenir une borne sous forme explicite, l'***inégalité de Pinsker*** se révèle fort utile :

$$2(q-p)^2 \;\leq\; \mathrm{kl}(q,p)\,. \tag{4.10}$$

2. En réalité, Maurer (2004) a démontré $\mathcal{I}_{\mathrm{kl}}(m)\leq 2\sqrt{m}$ pour $m\geq 8$, ainsi que $\mathcal{I}_{\mathrm{kl}}(m)\geq\sqrt{m}$ pour $m\geq 2$. Cependant, en effectuant les calculs exacts, on réalise que ces deux inégalités sont vraies pour $m\in\{1,2,3,4,5,6,7\}$.

L'équation (4.10) permet d'obtenir le corollaire suivant. En utilisant le corollaire 4.6 de pair avec la fonction de perte linéaire, nous retrouverons, à la section 4.3, le théorème PAC-bayésien de McAllester (1999, 2003a).

Corollaire 4.6. *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to \overline{\mathcal{Y}}$, pour toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0,1]$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta \in (0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S \sim D^m$,*

$\forall\, Q$ *sur* $\mathcal{H}$:

$$\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}}(f) - \sqrt{\tfrac{1}{2m}\Big[\mathrm{KL}(Q\|P) + \ln \tfrac{2\sqrt{m}}{\delta}\Big]} \;\le\; \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}}(f) \;\le\; \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}}(f) + \sqrt{\tfrac{1}{2m}\Big[\mathrm{KL}(Q\|P) + \ln \tfrac{2\sqrt{m}}{\delta}\Big]}.$$

Démonstration. Les deux inégalités (borne inférieure et borne supérieure) sont obtenues en appliquant l'inégalité de Pinsker – équation (4.10) – sur le corollaire 4.6. □

Les corollaires 4.5 et 4.6 expriment à la fois une borne inférieure et une borne supérieure sur l'espérance de perte pondérée $\mathbf{E}_{f\sim Q}\, \mathbb{E}_D^{\mathcal{L}}(f)$. Insistons sur le fait que le corollaire 4.5 donne de meilleures garanties statistiques, car l'utilisation de l'inégalité de Pinsker pour obtenir le corollaire 4.6 entraîne un «relâchement» des bornes. Néanmoins, nous préférons parfois la simplicité de l'énoncé de ce dernier.

Dans tous les cas, les bornes inférieures et supérieures convergent vers $\mathbf{E}_{f\sim Q}\, \mathbb{E}_D^{\mathcal{L}}(f)$ lorsque le nombre d'exemples d'entraînement tend vers l'infini. Ceci est un comportement souhaitable pour une borne sur l'espérance de perte. Plus précisément, peu importe la nature de $\mathcal{H}$, $\mathcal{L}$, Q, P et δ, si les m exemples de l'échantillon d'entraînement S sont générés de manière *i.i.d.* par la distribution D (c'est-à-dire $S \sim D^m$), on a

$$\lim_{|S|\to\infty} \left[\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}}(f) \right] \;=\; \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}}(f)\,. \tag{4.11}$$

Autrement dit, l'espérance de perte empirique converge vers l'espérance de perte «réelle». De plus, le terme de complexité présent dans la borne du corollaire 4.6 devient nul lorsque m tend vers l'infini :

$$\lim_{m\to\infty} \left[\sqrt{\tfrac{1}{2m}\Big[\mathrm{KL}(Q\|P) + \ln \tfrac{2\sqrt{m}}{\delta}\Big]} \right] \;=\; 0\,. \tag{4.12}$$

L'équation (4.12) permet aussi de constater que le taux de convergence de la borne du corollaire 4.6 est de l'ordre de $\frac{1}{\sqrt{m}}$ en fonction de la taille de l'échantillon d'entraînement m.

4.2.4 Une deuxième Δ-fonction : une fonction «hyperparamétrée»

Explorons un autre type de Δ-fonction, permettant d'obtenir une borne PAC-bayésienne semblable à celle suggérée par Catoni (2007).

Dans Germain et al. (2009a); Germain (2009), nous avons initialement obtenu la Δ-fonction de l'équation (4.13) ci-bas en cherchant une fonction, notée $\Delta_c(q,p)$, qui soit linéaire en son

premier argument q (correspondant à l'espérance de perte empirique dans les bornes PAC-bayésiennes) et convexe en son deuxième argument p (correspondant à l'espérance de perte sur la distribution génératrice). Plus précisément, notre volonté était de formuler une Δ-fonction possédant la forme

$$\Delta_c(q,p) \;\coloneqq\; \mathcal{F}(p) - c \cdot q\,,$$

où $c > 0$ est un paramètre et $\mathcal{F}(\cdot)$ est une fonction convexe. Nous devions alors choisir cette fonction $\mathcal{F}(\cdot)$ judicieusement. La démonstration du corollaire 4.7 (plus bas) permet de constater que le choix

$$\mathcal{F}(p) \;\coloneqq\; -\ln[\,1 - (1 - e^{-c}) \cdot p\,]$$

possède la propriété intéressante que $\mathcal{I}_{\Delta_c}(m) = 1$ pour tout $m \in \mathbb{N}^*$ (pour plus de détails, voir Germain, 2009, section 3.1.3).

Ainsi, nous définissons

$$\Delta_c(q,p) \;\stackrel{\text{def}}{=}\; -\ln[\,1 - (1 - e^{-c}) \cdot p\,] - c \cdot q\,. \tag{4.13}$$

Comme souligné par Lever et al. (2013), on peut réécrire l'équation (4.13) ainsi :

$$\Delta_c(q,p) \;=\; -\frac{1}{m} \ln\Big[\, M_X(-c) \,\Big] - c \cdot q\,,$$

où $M_X(t) \stackrel{\text{def}}{=} (1 - p + pe^t)^m$ est la *fonction génératrice des moments* d'une variable aléatoire binomiale de m essais avec probabilité de succès p.

Le corollaire 4.7 ci-bas spécialise théorème 4.4 au cas $\Delta(q,p) \coloneqq \Delta_c(q,p)$. Ainsi, en utilisant le corollaire 4.7 de pair avec la fonction de perte linéaire, nous retrouverons, à la section 4.3, le théorème PAC-bayésien de Catoni (2007).

Corollaire 4.7. *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to \overline{\mathcal{Y}}$, pour toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0,1]$, pour toute distribution P sur $\mathcal{H}$, pour tout nombre réel $c > 0$, et pour tout $\delta \in (0,1]$, on a, avec probabilité au moins $1 - \delta$ sur le choix de $S \sim D^m$,*

$$\forall\, Q \text{ sur } \mathcal{H} :$$

$$\begin{aligned}
\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}^{\mathcal{L}}_D(f) \;&\leq\; \frac{1}{1 - e^{-c}} \left[1 - \exp\left\{ -c \cdot \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}^{\mathcal{L}}_S(f) - \frac{\mathrm{KL}(Q\|P) + \ln\frac{1}{\delta}}{m} \right\} \right] && (4.14)\\
&\leq\; \frac{1}{1 - e^{-c}} \left[c \cdot \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}^{\mathcal{L}}_S(f) + \frac{\mathrm{KL}(Q\|P) + \ln\frac{1}{\delta}}{m} \right]. && (4.15)
\end{aligned}$$

Démonstration. Le résultat découle du théorème 4.4, avec $\Delta(q,p) \coloneqq \Delta_c(q,p)$ tel que défini par l'équation (4.13). Simplifions d'abord la valeur de $\mathcal{I}_{\Delta_c}(m)$:

$$\begin{aligned}
\mathcal{I}_{\Delta_c}(m) &= \sup_{r\in[0,1]} \left[\sum_{k=0}^{m} \binom{m}{k} r^k (1-r)^{m-k} e^{m\Delta_c(\frac{k}{m},r)} \right] \\
&= \sup_{r\in[0,1]} \left[\sum_{k=0}^{m} \binom{m}{k} r^k (1-r)^{m-k} \frac{e^{-c\cdot k}}{[1-(1-e^{-c})\cdot r]^m} \right] \\
&= \sup_{r\in[0,1]} \left[\sum_{k=0}^{m} \binom{m}{k} (r\cdot e^{-c})^k (1-r)^{m-k} \frac{1}{[1-(1-e^{-c})\cdot r]^m} \right] \\
&= \sup_{r\in[0,1]} \left[\frac{[r\cdot e^{-c}+(1-r)]^m}{[1-(1-e^{-c})\cdot r]^m} \right] \;=\; \sup_{r\in[0,1]} [\,1\,] \;=\; 1\,.
\end{aligned}$$

Notons que le passage à la dernière ligne ci-dessus découle du binôme de Newton :

$$\sum_{k=0}^{m} \tbinom{m}{k} x^k y^{m-k} \;=\; (x+y)^m\,.$$

En insérant ce résultat dans l'énoncé du théorème 4.4, on obtient

$$-\ln\Big[1-(1-e^{-c})\cdot \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}}(f)\Big] - c\cdot \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}}(f) \;\le\; \frac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln\frac{1}{\delta}\right],$$

et la ligne (4.14) de l'énoncé est obtenue en isolant le terme $\mathbf{E}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)$. La ligne (4.15) est ensuite obtenue de la ligne (4.14) par l'inégalité $1-e^{-x} \le x$. □

Contrairement aux corollaires 4.5 et 4.6, le corollaire 4.7 exprime uniquement une borne supérieure sur l'espérance de perte $\mathbf{E}_{f\sim Q}\,\mathbb{E}_D^{\mathcal{L}}(f)$. Bien entendu, la valeur du paramètre c influence la valeur de cette borne. Il est montré dans Lacasse (2010) que, du point de vue de la minimisation de la valeur de la borne, le paramètre c optimal permet d'obtenir une borne supérieure égale (à un minuscule coefficient près) à celle obtenue par le corollaire 4.5 et la Δ-fonction $\mathrm{kl}(\cdot,\cdot)$. Ainsi, nous préférons toujours le corollaire 4.5 lorsque l'objectif est d'obtenir la meilleure garantie statistique possible sur l'espérance de perte.

Toutefois, la borne sur l'espérance de perte exprimée par le corollaire 4.7 s'avère utile pour la conception d'algorithmes d'apprentissage. En effet, le paramètre c devient un hyperparamètre de l'algorithme modifiant son comportement. Une grande valeur de c permet d'augmenter l'influence de l'espérance de perte empirique $\mathbf{E}_{f\sim Q}\,\mathbb{E}_S^{\mathcal{L}}(f)$. À l'inverse, une petite valeur de c valorise le terme $\mathrm{KL}(Q\|P)$, et ainsi les votes de majorité (distributions Q sur $\mathcal{H}$) semblables à celui spécifié *a priori* (distribution P sur $\mathcal{H}$).

Soulignons qu'avec une valeur de paramètre c bien choisie (en fonction de m), la borne supérieure du corollaire 4.7 converge vers l'espérance de perte. En effet, avec $c \coloneqq \frac{1}{\sqrt{m}}$, l'équa-

tion (4.15) devient, lorsque le nombre d'exemples d'entraînement $m = |S|$ est très grand,

$$\lim_{m\to\infty}\left[\frac{1}{\sqrt{m}\left(1-e^{-\frac{1}{\sqrt{m}}}\right)}\cdot\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f)+\frac{\mathrm{KL}(Q\|P)+\ln\frac{1}{\delta}}{m\left(1-e^{-\frac{1}{\sqrt{m}}}\right)}\right] \;=\; 1\cdot\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)+0 \qquad (4.16)$$
$$\;=\; \mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)\,.$$

Puisque la valeur de l'équation (4.15) est plus grande ou égale à la valeur de l'équation (4.14), cette dernière donne aussi une borne qui converge vers l'espérance de perte «réelle» avec un paramètre $c \coloneqq \frac{1}{\sqrt{m}}$. L'équation (4.16) met en évidence que, lorsque l'espérance de perte est nulle (ou très faible), la borne supérieure donnée par le corollaire 4.7 a un taux de convergence de l'ordre de $\frac{1}{m}$ en fonction de la taille de l'échantillon d'entraînement m.

4.3 Théorie PAC-bayésienne pour le risque de Gibbs

Dans cette section, nous présentons un théorème qui permet d'obtenir une garantie sur le risque de Gibbs. Il s'agit du type de résultats le mieux connu de la théorie PAC-bayésienne. Nous expliquons ensuite en détail la procédure pour convertir ce résultat en une borne supérieure du risque de Bayes.

4.3.1 Bornes sur le risque de Gibbs

Le corollaire 4.8 ci-bas présente trois spécialisations du théorème PAC-bayésien général, démontré à la section précédente, pour le risque du classificateur de Gibbs $R_D(G_Q)$.

Corollaire 4.8. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X}\to[-1,1]$, pour toute distribution P sur $\mathcal{H}$, pour tout nombre réel $c>0$, et pour tout $\delta\in(0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$,*

$\forall\, Q$ *sur* $\mathcal{H}$:

(a) $\mathrm{kl}\Big(R_S(G_Q), R_D(G_Q)\Big) \;\le\; \frac{1}{m}\left[\mathrm{KL}(Q\|P)+\ln\frac{2\sqrt{m}}{\delta}\right],$ *(Langford et Seeger, 2001)*

(b) $R_D(G_Q) \;\le\; R_S(G_Q)+\sqrt{\frac{1}{2m}\left[\mathrm{KL}(Q\|P)+\ln\frac{2\sqrt{m}}{\delta}\right]},$ *(McAllester, 1999, 2003a)*

(c) $R_D(G_Q) \;\le\; \frac{1}{1-e^{-c}}\left(c\cdot R_S(G_Q)+\frac{1}{m}\left[\mathrm{KL}(Q\|P)+\ln\frac{1}{\delta}\right]\right).$ *(Catoni, 2007)*

Démonstration. Le résultat (a) est obtenu directement du corollaire 4.5, avec la fonction de perte linéaire $\mathcal{L} \coloneqq \mathcal{L}_\ell$ et la définition du risque de Gibbs (définition 3.2). De même, les résultats (b) et (c) sont obtenus directement des corollaires 4.6 et 4.7 respectivement. □

Nous retrouvons donc par le corollaire 4.8 trois formes très similaires à des théorèmes PAC-bayésiens connus dans la littérature :

(a) Le théorème de Langford et Seeger (2001); Seeger (2002) permet de formuler les bornes les plus serrées, en plus d'exprimer à la fois une borne inférieure et supérieure du risque de Gibbs. Cependant, il ne s'exprime pas sous une forme explicite.

(b) Le théorème de McAllester (1999, 2003a) possède l'avantage de s'exprimer de manière très simple.

(c) Le théorème de Catoni (2007) est paramétré par une constante c. Cet aspect en fait un outil de prédilection lors de la conception d'algorithmes d'apprentissage, car la valeur du paramètre c permet d'ajuster le compromis entre la minimisation du risque empirique $R_S(G_Q)$ et de la mesure de complexité $\mathrm{KL}(Q\|P)$ (voir Germain et al., 2009a; Germain, 2009).

Ces trois formulations du théorème PAC-bayésien donnent des bornes sur le risque de Gibbs $R_D(G_Q)$ qui sont consistantes : leurs valeurs convergent vers le risque de Gibbs quand le nombre m d'exemples d'entraînement augmente. Ceci découle directement de la convergence des bornes sur l'espérance de perte présentées par les corollaires 4.5, 4.6 et 4.7, comme le montre les équations (4.11), (4.12) et (4.16).

4.3.2 Borne sur le risque de Bayes

Chacune des bornes supérieures sur le risque du classificateur de Gibbs G_Q exprimées par le corollaire 4.8 de la section précédente peut directement être convertie en borne sur le risque du vote de majorité correspondant B_Q en appliquant la ***borne du facteur deux*** énoncée par la proposition 3.6. Nous disons que la borne obtenue est ***triviale*** lorsqu'elle est supérieure à 1.

Le résultat suivant synthétise la procédure permettant de borner le risque du vote de majorité $R_D(B_Q)$ à l'aide de la forme de Langford et Seeger (2001). Il s'agit d'une conséquence directe du corollaire 4.8(a), combinée avec la proposition 3.6.

Borne du vote de majorité 4.9. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X}\to[-1,1]$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta\in(0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$,*

$$\forall\, Q \text{ sur } \mathcal{H}: \quad R_D(B_Q) \;\leq\; 2\cdot\sup\mathcal{R}^{\delta}_{Q,S}\,,$$

où

$$\mathcal{R}^{\delta}_{Q,S} \;\stackrel{\text{def}}{=}\; \left\{ r\in[0,\tfrac{1}{2}] \;\middle|\; \mathrm{kl}(R_S(G_Q),\,r) \;\leq\; \frac{1}{m}\left[\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta}\right]\right\}. \tag{4.17}$$

La valeur de la borne 4.9 sur $R_D(B_Q)$ n'est pas donnée par une forme explicite. Pour la calculer, nous trouvons la valeur de $r=\sup\mathcal{R}^{\delta}_{Q,S}$ en résolvant l'équation

$$\mathrm{kl}(R_S(G_Q),\,r) \;=\; \tfrac{1}{m}\left[\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta}\right] \qquad (\text{avec } R_S(G_Q)\leq r\leq\tfrac{1}{2}\,).$$

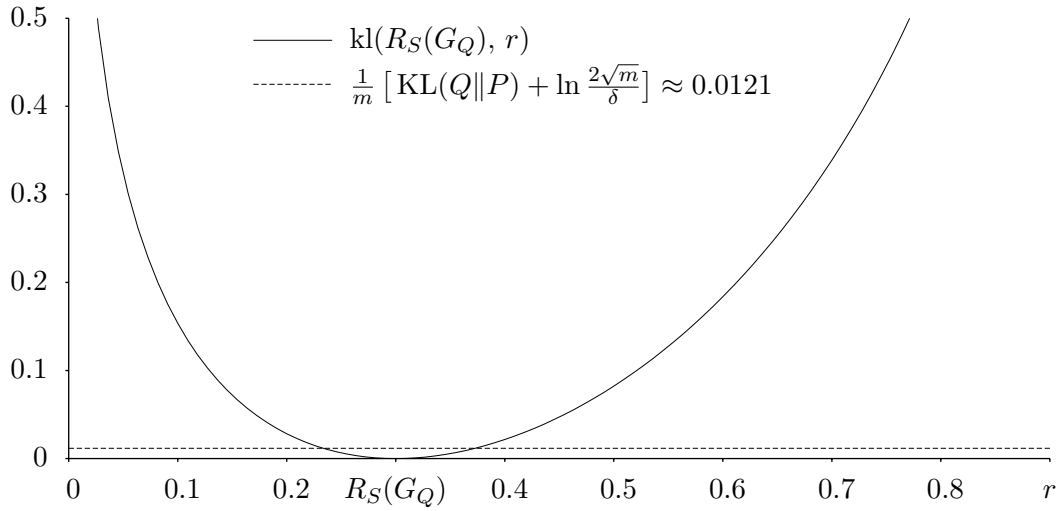


FIGURE 4.1 – Illustration du calcul de la borne 4.9. On suppose $\mathrm{KL}(Q\|P) = 5$, $m = 1000$ et $\delta = 0.05$. Si on observe un risque de Gibbs empirique $R_S(G_Q) = 0.30$, alors $\mathcal{R}^{\delta}_{Q,S} \approx [0.232, 0.374]$ (sur la figure, les intersections entre les deux lignes correspondent aux limites de l'intervalle $\mathcal{R}^{\delta}_{Q,S}$). On peut donc conclure, avec un taux de confiance de 95%, que $R_D(B_Q) \lesssim 2 \times 0.374 = 0.748$.

Il s'agit d'une tâche facile à exécuter à l'aide d'un algorithme de recherche de zéros d'une fonction (comme les méthodes de Newton ou de Brent), car le terme de gauche de l'égalité ci-haut est convexe fonction de r et le terme de droite est constant. La figure 4.1 illustre un exemple de calcul de la borne 4.9. Notons que l'intervalle $\mathcal{R}^{\delta}_{Q,S}$ de l'équation (4.17) a $\frac{1}{2}$ comme valeur maximale, car la borne 4.9 est triviale lorsque $\sup \mathcal{R}^{\delta}_{Q,S} = \frac{1}{2}$. Autrement dit, la borne nous indique $R_D(B_Q) \leq 1$.

4.4 Théorie PAC-bayésienne pour l'espérance de désaccord

La borne du vote de majorité 4.9 de la section précédente utilise la borne du facteur deux (proposition 3.6). Or, nous avons vu que, sous les conditions exprimées par la proposition 3.9, la $\mathcal{C}$-borne (théorème 3.7) permet d'obtenir une borne plus serrée. Afin de borner supérieurement la valeur de la $\mathcal{C}$-borne sur une distribution de données D à partir de sa valeur empirique sur un échantillon $S \sim D^m$, nous présentons dans cette section une stratégie pour obtenir une garantie sur l'espérance de désaccord. Pour ce faire, nous définissons la notion de *votants jumelés*.

4.4.1 Votant jumelé

Les votants étudiés précédemment sont des régresseurs qui retournent un nombre réel appartenant à l'intervalle $[-1, 1]$. Nous allons maintenant élargir le concept de votants. Les votants

jumelés définis ci-bas retournent un tuple appartenant à $[-1,1] \times [-1,1]$. Nous définissons ensuite une fonction de perte pour ces votants jumelés, de la forme $[-1,1]^2 \times \{-1,1\} \to [0,1]$.

Définition 4.10. Soit deux votants $f_i : \mathcal{X} \to [-1,1]$ et $f_j : \mathcal{X} \to [-1,1]$. Le ***votant jumelé*** associé à f_i et f_j est une fonction de la forme $f_{ij} : \mathcal{X} \to [-1,1]^2$ qui est définie par

$$f_{ij}(x) \stackrel{\text{def}}{=} \langle\, f_i(x), f_j(x) \,\rangle\,.$$

Étant donné un ensemble de votants $\mathcal{H}$ pondéré par une distribution Q, nous définissons l'*ensemble des votants jumelés*

$$\mathcal{H}^2 \stackrel{\text{def}}{=} \{f_{ij} \mid f_i, f_j \in \mathcal{H}\}\,. \tag{4.18}$$

De plus, pour chaque distribution Q sur $\mathcal{H}$, nous notons Q^2 la distribution sur $\mathcal{H}^2$ définie par

$$Q^2(f_{ij}) \stackrel{\text{def}}{=} Q(f_i) \cdot Q(f_j)\,. \tag{4.19}$$

La définition suivante présente une fonction de perte quantifiant le désaccord entre deux votants constituant un votant jumelé. La valeur de la fonction sera près de 1 lorsque les deux votants sont fortement en désaccord, et près de 0 sinon.

Définition 4.11. Soit un exemple $(x,y) \in \mathcal{X} \times \{-1,+1\}$ et un votant jumelé $f_{ij} : \mathcal{X} \to [-1,1]^2$, formé des deux votants $f_i : \mathcal{X} \to [-1,1]$ et $f_j : \mathcal{X} \to [-1,1]$. Nous définissons la ***fonction de perte*** $\mathcal{L}_d : [-1,1]^2 \times \{-1,1\} \to [0,1]$ comme suit :

$$\mathcal{L}_d(f_{ij}(x),\, y) \stackrel{\text{def}}{=} \mathcal{L}_\ell(f_i(x) \cdot f_j(x)\,,\, 1)\,.$$

Cette nouvelle fonction de perte permet d'énoncer, à la section 4.4.2, un théorème PAC-bayésien sur l'espérance de désaccord d'un ensemble de votants (voir définition 3.3, page 45). En effet, ce nouveau résultat repose sur le fait qu'il est possible de réécrire l'espérance de désaccord $d_Q^{D'}$ comme l'espérance de la perte $\mathcal{L}_d$ d'un ensemble de votants jumelés, pondéré par Q^2, sur une distribution de données D'. À partir des définitions 3.3 et 4.11, on obtient

$$\begin{aligned} d_Q^{D'} &= \mathop{\mathbf{E}}_{(x,\cdot)\sim D'} \mathop{\mathbf{E}}_{f_i\sim Q} \mathop{\mathbf{E}}_{f_j\sim Q} \mathcal{L}_\ell\left(\, f_i(x)\cdot f_j(x)\,,\, 1\,\right) \\ &= \mathop{\mathbf{E}}_{(x,\cdot)\sim D'} \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathcal{L}_d(f_{ij}(x),\, y) \\ &= \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathbb{E}_{D'}^{\mathcal{L}_d}\Big(f_{ij}\Big)\,, \end{aligned} \tag{4.20}$$

où la dernière ligne est une simple réécriture utilisant la notation de l'espérance de perte introduite par la définition 2.2 (page 17).

4.4.2 Borne sur l'espérance de désaccord

Les votants jumelés et la fonction de perte définis plus haut nous permettent, en s'inspirant du corollaire 4.8 bornant le risque de Gibbs, de démontrer le corollaire 4.12 ci-bas. Ce dernier borne l'espérance de désaccord d_Q^D sur une distribution génératrice D à partir de sa contrepartie empirique d_Q^S. Le corollaire 4.12 se décline en trois différentes formes, calquées sur les trois formes des bornes sur le risque de Gibbs du corollaire 4.8.

Corollaire 4.12. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X}\to[-1,1]$, pour toute distribution P sur $\mathcal{H}$, pour tout nombre réel $c>0$, et pour tout $\delta\in(0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$,*

$\forall\, Q$ *sur* $\mathcal{H}$:

(a) $\operatorname{kl}\Big(d_Q^S,\, d_Q^D\Big) \;\le\; \frac{1}{m}\left[2\,\mathrm{KL}(Q\|P)+\ln\frac{2\sqrt{m}}{\delta}\right],$

(b) $d_Q^S-\sqrt{\frac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\frac{2\sqrt{m}}{\delta}\right]} \;\le\; d_Q^D \;\le\; d_Q^S+\sqrt{\frac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\frac{2\sqrt{m}}{\delta}\right]},$

(c) $d_Q^D \;\le\; \dfrac{1}{1-e^{-c}}\left(c\cdot d_Q^S+\frac{1}{m}\left[2\,\mathrm{KL}(Q\|P)+\ln\frac{1}{\delta}\right]\right).$

Démonstration. Considérons l'ensemble de votants jumelés $\mathcal{H}^2$ pondéré par la distribution Q^2, conformément aux équations (4.18) et (4.19). Considérons également une distribution P^2 sur $\mathcal{H}^2$ définie par $P^2(f_{ij}) \stackrel{\text{def}}{=} P(f_i)\cdot P(f_j)$.

Le résultat (a) est obtenu en considérant le corollaire 4.5 avec $\mathcal{H}\coloneqq\mathcal{H}^2$, $Q\coloneqq Q^2$ et $P\coloneqq P^2$, de même que la fonction de perte $\mathcal{L}\coloneqq\mathcal{L}_d$. Ainsi,

$$\forall\, Q^2 \text{ sur } \mathcal{H}^2 :\; \operatorname{kl}\Big(\mathop{\mathbf{E}}_{f_{ij}\sim Q^2}\mathbb{E}_S^{\mathcal{L}_d}(f_{ij}),\, \mathop{\mathbf{E}}_{f_{ij}\sim Q^2}\mathbb{E}_D^{\mathcal{L}_d}(f_{ij})\Big) \;\le\; \frac{1}{m}\Big[\mathrm{KL}(Q^2\|P^2)+\ln\frac{2\sqrt{m}}{\delta}\Big]. \tag{4.21}$$

Par l'équation (4.20), on obtient

$$\mathop{\mathbf{E}}_{f_{ij}\sim Q^2}\mathbb{E}_D^{\mathcal{L}_d}\Big(f_{ij}\Big) \;=\; d_Q^D \quad\text{ et }\quad \mathop{\mathbf{E}}_{f_{ij}\sim Q^2}\mathbb{E}_S^{\mathcal{L}_d}\Big(f_{ij}\Big) \;=\; d_Q^S\,. \tag{4.22}$$

De plus, on calcule

$$\begin{aligned}
\mathrm{KL}(Q^2\|P^2) &= \mathop{\mathbf{E}}_{f_{ij}\sim Q^2}\ln\frac{Q^2(f_{ij})}{P^2(f_{ij})} \;=\; \mathop{\mathbf{E}}_{f_{ij}\sim Q^2}\ln\frac{Q(f_i)\cdot Q(f_j)}{P(f_i)\cdot P(f_j)}\\
&= \mathop{\mathbf{E}}_{f_{ij}\sim Q^2}\left[\ln\frac{Q(f_i)}{P(f_i)}+\ln\frac{Q(f_j)}{P(f_j)}\right]\\
&= 2\cdot\mathrm{KL}(Q\|P)\,,
\end{aligned} \tag{4.23}$$

et on obtient le résultat (a) en insérant les lignes (4.22) et (4.23) dans l'équation (4.21).

Le résultat (b) découle de l'application de l'inégalité de Pinsker sur (a). En effet, l'équation (4.10) nous donne : $2\left(d_Q^S - d_Q^D\right)^2 \leq \frac{1}{m}\left[2\,\mathrm{KL}(Q\|P) + \ln\frac{2\sqrt{m}}{\delta}\right]$.

Le résultat (c) est obtenu similairement au résultat (a), mais en exploitant le corollaire 4.7, c'est-à-dire en insérant les lignes (4.22) et (4.23) dans l'équation suivante :

$$\forall\, Q^2 \text{ sur } \mathcal{H}^2 : \quad \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathbb{E}_D^{\mathcal{L}_d}(f_{ij}) \;\leq\; \frac{1}{1-e^{-c}}\left[c \cdot \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathbb{E}_S^{\mathcal{L}_d}(f_{ij}) + \frac{\mathrm{KL}(Q^2\|P^2) + \ln\frac{1}{\delta}}{m}\right].$$

□

Puisque les trois versions du corollaire ci-dessus sont fortement inspirées des trois résultats pour le risque de Gibbs présentés par le corollaire 4.8, les remarques formulées à la section 4.3.1 sont valables pour les bornes sur l'espérance de désaccord obtenues grâce au corollaire 4.12. Ainsi, la forme (a) permet d'obtenir les bornes les plus serrées sur l'espérance de désaccord, tandis que la forme (b) est plus simple à calculer et que la (c) est paramétrée par une constante «c» permettant de modifier l'influence de l'espérance de désaccord empirique d_Q^S par rapport au terme $\mathrm{KL}(Q\|P)$. Aussi, les trois formes convergent vers l'espérance de désaccord d_Q^D lorsque le nombre d'exemples d'entraînement m est très grand.

Remarque sur les bornes inférieures et supérieures. Les formes (a) et (b) du corollaire 4.12 expriment à la fois une borne *inférieure* et *supérieure* de l'espérance de désaccord. Notons que, dans la présentation du corollaire 4.8(b) bornant le risque de Gibbs, nous écrivons uniquement la borne supérieure du risque de Gibbs. C'est que, comme nous le montrons plus loin (sous-section 4.4.3 ci-bas), la borne inférieure de l'espérance de désaccord peut servir à borner le risque d'un vote de majorité à l'aide de la $\mathcal{C}$-borne. De plus, le corollaire 4.12(b) est utilisé au chapitre 6, portant sur l'adaptation de domaine.

4.4.3 Une nouvelle borne sur le risque de Bayes

Nous présentons maintenant une borne sur le risque du vote de majorité $R_D(B_Q)$ obtenue en bornant supérieurement le risque de Gibbs à l'aide du corollaire 4.8(a) et en bornant inférieurement l'espérance de désaccord à l'aide du corollaire 4.12(a). Ces deux bornes sont combinées pour obtenir une borne supérieure de la troisième forme de $\mathcal{C}_Q^D$ (théorème 3.7, page 49) et par conséquent une borne supérieure de $R_D(B_Q)$.

Chacune des deux bornes intervenant dans le calcul de la borne 4.13 est calculée avec un paramètre de confiance $\delta/2$. En vertu la borne de l'union, cela est nécessaire afin que les deux

bornes soient *simultanément* valides *avec probabilité au moins* $1-\delta$.

Borne du vote de majorité 4.13. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X}\to[-1,1]$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta\in(0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$,*

$$\forall\, Q \text{ sur } \mathcal{H}: \quad R_D(B_Q) \;\leq\; \mathcal{C}_Q^D \;\leq\; 1-\frac{\left(1-2\cdot\sup\mathcal{R}_{Q,S}^{\delta/2}\right)^2}{1-2\cdot\inf\mathcal{D}_{Q,S}^{\delta/2}}\,,$$

où

$$\mathcal{R}_{Q,S}^{\delta/2} \;\stackrel{\text{def}}{=}\; \left\{ r\in[0,\tfrac{1}{2}] \;\middle|\; \mathrm{kl}(R_S(G_Q),\,r) \;\leq\; \frac{1}{m}\left[\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]\right\},$$
$$\mathcal{D}_{Q,S}^{\delta/2} \;\stackrel{\text{def}}{=}\; \left\{ d\in[0,\tfrac{1}{2}] \;\middle|\; \mathrm{kl}(d_Q^S,\,d) \;\leq\; \frac{1}{m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]\right\}.$$

Notons que la valeur maximale de l'intervalle $\mathcal{R}_{Q,S}^{\delta/2}$ est $\frac{1}{2}$, car la $\mathcal{C}$-borne n'est pas valide lorsque $R_D(G_Q)\geq\frac{1}{2}$ (en vertu du théorème 3.7). Lorsque $\sup\mathcal{R}_{Q,S}^{\delta/2}=\frac{1}{2}$, on a $1-2\cdot\sup\mathcal{R}_{Q,S}^{\delta/2}=0$, et la borne 4.13 est $R_D(B_Q)\leq\mathcal{C}_Q^D\leq 1$, ce qui est trivialement vrai.

Pour calculer la valeur de la borne 4.13, nous trouvons d'abord la valeur de $r=\sup\mathcal{R}_{Q,S}^{\delta/2}$ en calculant les zéros de l'équation

$$\mathrm{kl}(R_S(G_Q),\,r) \;=\; \tfrac{1}{m}\left[\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right] \qquad (\text{avec } R_S(G_Q)\leq r\leq\tfrac{1}{2})\,,$$

à l'aide de la même procédure employée pour le calcul de la borne 4.9 (voir la section 4.3.2).

En procédant de manière similaire, on trouve la valeur de $d=\inf\mathcal{D}_{Q,S}^{\delta/2}$ en résolvant l'équation

$$\mathrm{kl}(d_Q^S,\,d) \;=\; \tfrac{1}{m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right] \qquad (\text{avec } 0\leq d\leq d_Q^S)\,.$$

Ainsi, nous obtenons, avec probabilité au moins $1-\delta$:

$$R_D(B_Q) \;\leq\; 1-\frac{(1-2\cdot r)^2}{1-2\cdot d}\,.$$

Cette méthode requiert deux approximations, correspondant au calcul de r et au calcul de d. La prochaine section présente une technique pour obtenir ces deux quantités *simultanément.*

4.5 Théorie PAC-bayésienne pour borner directement la $\mathcal{C}$-borne

La borne 4.13 de la section précédente est obtenue en combinant deux résultats : une borne supérieure du risque de Gibbs et une borne inférieure de l'espérance de désaccord. Dans cette section, nous expliquons qu'il est possible de borner directement la $\mathcal{C}$-borne. Pour ce faire,

nous définissons les concepts d'*espérance d'erreur conjointe* et d'*espérance de succès conjoint* de votants jumelés, et leur fonction de perte associée. Nous utilisons ensuite ces concepts pour exprimer un nouveau théorème PAC-bayésien qui permet de borner *simultanément* deux espérances de perte de votants jumelés. Ce nouveau résultat permet, à la fin de la section, d'obtenir une nouvelle borne supérieure sur le risque du vote de majorité.

4.5.1 Erreur conjointe et succès conjoint

Nous avons présenté à la définition 3.3 l'espérance de désaccord. Nous présentons maintenant deux concepts reliés : l'espérance de succès conjoint et l'espérance d'erreur conjointe. Lorsque les votants sont des classificateurs binaires (retournant -1 ou $+1$), l'espérance de succès conjoint correspond à la probabilité que deux classificateurs pigés selon Q prédisent tous les deux la bonne étiquette,

$$\begin{aligned}\Pr_{\substack{(x,\cdot)\sim D'\\ h_1,h_2\sim Q}}\Big((h_1(x)=y)\wedge(h_2(x)=y)\Big) &= \mathop{\mathbf{E}}_{h_1\sim Q}\mathop{\mathbf{E}}_{h_2\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim D'}\mathtt{I}\big[\,h_1(x)=y\,\big]\times\mathtt{I}\big[\,h_2(x)=y\,\big]\\ &= \mathop{\mathbf{E}}_{h_1\sim Q}\mathop{\mathbf{E}}_{h_2\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim D'}\mathcal{L}_{01}\big(h_1(x),y\big)\times\mathcal{L}_{01}\big(h_2(x),y\big)\,,\end{aligned}$$

tandis que l'espérance d'erreur conjointe correspond à la probabilité que deux classificateurs pigés selon Q prédisent tous les deux la mauvaise étiquette,

$$\begin{aligned}\Pr_{\substack{(x,\cdot)\sim D'\\ h_1,h_2\sim Q}}\Big((h_1(x)\neq y)\wedge(h_2(x)\neq y)\Big) &= \mathop{\mathbf{E}}_{h_1\sim Q}\mathop{\mathbf{E}}_{h_2\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim D'}\mathtt{I}\big[\,h_1(x)\neq y\,\big]\times\mathtt{I}\big[\,h_2(x)\neq y\,\big]\\ &= \mathop{\mathbf{E}}_{h_1\sim Q}\mathop{\mathbf{E}}_{h_2\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim D'}\Big[1-\mathcal{L}_{01}(h_1(x),y)\Big]\times\Big[1-\mathcal{L}_{01}(h_2(x),y)\Big]\,.\end{aligned}$$

La définition suivante généralise ces concepts aux votants à valeurs réelles.

Définition 4.14. Soit une distribution de probabilité Q sur un ensemble de votants. L'***espérance d'erreur conjointe*** $e_Q^{D'}$ sur D' et l'***espérance de succès conjoint*** $s_Q^{D'}$ sur D' correspondent respectivement à

$$\begin{aligned}e_Q^{D'} &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{f_1\sim Q}\mathop{\mathbf{E}}_{f_2\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim D'}\mathcal{L}_\ell(f_1(x),y)\cdot\mathcal{L}_\ell(f_2(x),y)\,,\\ s_Q^{D'} &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{f_1\sim Q}\mathop{\mathbf{E}}_{f_2\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim D'}\Big[1-\mathcal{L}_\ell(f_1(x),y)\Big]\cdot\Big[1-\mathcal{L}_\ell(f_2(x),y)\Big]\,.\end{aligned}$$

Nous définissons maintenant des fonctions de pertes pour votants jumelés associées à l'erreur conjointe et au succès conjoint, de manière similaire à ce que nous avons fait pour l'espérance de désaccord à la section 4.4.1 (voir la définition 4.11).

Définition 4.15. Soit un exemple $(x, y) \in \mathcal{X} \times \{-1, +1\}$ et un votant jumelé $f_{ij} : \mathcal{X} \to [-1, 1]^2$, formé des deux votants $f_i : \mathcal{X} \to [-1, 1]$ et $f_j : \mathcal{X} \to [-1, 1]$. Les ***fonctions de pertes*** $\mathcal{L}_e$ et $\mathcal{L}_s$ sont définies comme suit :

$$\begin{aligned} \mathcal{L}_e(f_{ij}(x),\, y) &\stackrel{\text{def}}{=} \mathcal{L}_\ell(f_i(x), y) \cdot \mathcal{L}_\ell(f_j(x), y)\,, \\ \mathcal{L}_s(f_{ij}(x),\, y) &\stackrel{\text{def}}{=} \Big[1 - \mathcal{L}_\ell(f_i(x), y)\Big] \cdot \Big[1 - \mathcal{L}_\ell(f_j(x), y)\Big]\,. \end{aligned}$$

Tout comme, à la section 4.4.1, nous avons écrit l'espérance de désaccord $d_Q^{D'}$ en fonction de l'espérance de la perte $\mathcal{L}_d$, il est possible d'écrire l'espérance d'erreurs conjointe $e_Q^{D'}$ en fonction de l'espérance de la perte $\mathcal{L}_e$, de même que l'espérance de succès conjoint $e_Q^{D'}$ en fonction de l'espérance de la perte $\mathcal{L}_d$. En suivant la démarche détaillée à l'équation (4.20), et en utilisant la notation introduite par la définition 2.2 (page 17), on conclut

$$e_Q^{D'} = \mathop{\mathbf{E}}_{f_{ij} \sim Q^2} \mathbb{E}_{D'}^{\mathcal{L}_e}\Big(f_{ij}\Big) \quad \text{et} \quad s_Q^{D'} = \mathop{\mathbf{E}}_{f_{ij} \sim Q^2} \mathbb{E}_{D'}^{\mathcal{L}_s}\Big(f_{ij}\Big)\,. \tag{4.24}$$

Notons qu'il est facile d'adapter le corollaire 4.12 afin d'énoncer des garanties PAC-bayésiennes sur l'espérance d'erreur conjointe e_Q^D ou l'espérance de succès conjoint s_Q^D à partir de leur contrepartie empirique, respectivement e_Q^S ou s_Q^S. Il suffit de remplacer, dans l'énoncé corollaire 4.12, $\mathrm{kl}(d_Q^S, d_Q^D)$ par $\mathrm{kl}(e_Q^S, e_Q^D)$, ou encore par $\mathrm{kl}(s_Q^S, s_Q^D)$. De même, la démonstration de ces résultats s'effectue en suivant une démarche quasi identique à celle de la démonstration du corollaire 4.12, en exploitant l'une des deux égalités de l'équation (4.24), au lieu de celle de l'équation (4.20).

Dans les pages qui suivent, nous verrons plutôt comment borner *simultanément* deux espérances de pertes d'un même vote de majorité. La méthode proposée ci-bas s'adapte à toute paire parmi le triplet d_Q^D, e_Q^D et s_Q^D. Nous concentrerons notre étude sur le couple formé de d_Q^D et e_Q^D, car cela nous permet, comme il est expliqué plus loin, de borner directement la $\mathcal{C}$-borne. Pour ce faire, nous devons d'abord énoncer quelques propriétés qui unissent les espérances d'erreur conjointe, de succès conjoint et de désaccord.

En exploitant la définition 2.1 de la fonction de perte linéaire et la définition 3.4 de la marge (pages 16 et 46), on peut écrire les expressions $e_Q^{D'}$ et $s_Q^{D'}$, de la définition 4.14, en fonction des deux premiers moments de la marge :

$$\begin{aligned} e_Q^{D'} &= \mathop{\mathbf{E}}_{f_1 \sim Q} \mathop{\mathbf{E}}_{f_2 \sim Q} \mathop{\mathbf{E}}_{(x,y) \sim D'} \frac{1}{2}\Big(1 - y \cdot f_1(x)\Big) \cdot \frac{1}{2}\Big(1 - y \cdot f_2(x)\Big) \\ &= \mathop{\mathbf{E}}_{(x,y) \sim D'} \frac{1}{2}\Big(1 - M_Q(x, y)\Big) \cdot \frac{1}{2}\Big(1 - M_Q(x, y)\Big) \\ &= \frac{1}{4}\Big(1 - 2 \mathop{\mathbf{E}}_{(x,y) \sim D'} M_Q(x, y) + \mathop{\mathbf{E}}_{(x,y) \sim D'} \big[M_Q(x, y)\big]^2\Big) \\ &= \frac{1}{4}\Big(1 - 2 \cdot \mu_1(M_Q^{D'}) + \mu_2(M_Q^{D'})\Big)\,, \end{aligned} \tag{4.25}$$

et

$$
\begin{aligned}
s_Q^{D'} &= \mathop{\mathbf{E}}_{f_1\sim Q}\mathop{\mathbf{E}}_{f_2\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim D'}\frac{1}{2}\Big(1+y\cdot f_1(x)\Big)\cdot\frac{1}{2}\Big(1+y\cdot f_2(x)\Big)\\
&= \mathop{\mathbf{E}}_{(x,y)\sim D'}\frac{1}{2}\Big(1+M_Q(x,y)\Big)\cdot\frac{1}{2}\Big(1+M_Q(x,y)\Big)\\
&= \frac{1}{4}\Big(1+2\mathop{\mathbf{E}}_{(x,y)\sim D'}M_Q(x,y)+\mathop{\mathbf{E}}_{(x,y)\sim D'}\left[M_Q(x,y)\right]^2\Big)\\
&= \frac{1}{4}\Big(1+2\cdot\mu_1(M_Q^{D'})+\mu_2(M_Q^{D'})\Big)\,. \qquad (4.26)
\end{aligned}
$$

En additionnant les équations (4.25) et (4.26), on obtient

$$e_Q^{D'}+s_Q^{D'} = \frac{1}{2}\Big(1+\mu_2(M_Q^{D'})\Big)\,, \qquad (4.27)$$

et en les soustrayant, on obtient

$$e_Q^{D'}-s_Q^{D'} = \mu_1(M_Q^{D'})\,. \qquad (4.28)$$

En combinant l'équation (4.27) avec l'expression de l'espérance de désaccord $d_Q^{D'}$ de l'équation (3.10), c'est-à-dire $d_Q^{D'}=\frac{1}{2}(1-\mu_2(M_Q^{D'}))$, on constate que

$$e_Q^{D'}+s_Q^{D'}+d_Q^{D'} = 1\,. \qquad (4.29)$$

Autrement dit, pour un vote de majorité donné, la somme des espérances d'erreur conjointe, de succès conjoint et de désaccord vaut toujours 1. Remarquons que, dans le cas particulier où les votants sont des classificateurs binaires, ce phénomène se comprend intuitivement. En effet, étant donné un exemple $(x,y)\in\mathcal{X}\times\{-1,1\}$ et deux classificateurs binaires $h_1,h_2:\mathcal{X}\to\{-1,1\}$, nous sommes dans l'une des trois situations suivantes :

1. Les deux classificateurs prédisent la mauvaise étiquette : $h_1(x)=h_2(x)\neq y$,
2. Les deux classificateurs prédisent la bonne étiquette : $h_1(x)=h_2(x)=y$,
3. Les deux classificateurs prédisent des étiquettes différentes : $h_1(x)\neq h_2(x)$.

Notons finalement qu'en combinant les équations (4.28) et (4.29) avec l'expression du risque de Gibbs en fonction du premier moment de la marge – équation (3.9), page 47 – on a

$$R_{D'}(G_Q) = \frac{1}{2}\Big(1-\mu_1(M_Q^{D'})\Big) = \frac{1}{2}\left(1-s_Q^{D'}+e_Q^{D'}\right) = \frac{1}{2}\left(2e_Q^{D'}+d_Q^{D'}\right), \qquad (4.30)$$

ce qui permet de réécrire la troisième forme de la $\mathcal{C}$-borne – théorème 3.7, page 49 – sous la forme suivante :

$$\mathcal{C}_Q^{D'} = 1-\frac{\Big(1-(2e_Q^{D'}+d_Q^{D'})\Big)^2}{1-2d_Q^{D'}}\,. \qquad (4.31)$$

L'équation précédente nous permettra, à la sous-section 4.5.6, de borner $\mathcal{C}_Q^D$ en bornant simultanément e_Q^D et d_Q^D. Plus précisément, nous majorerons le numérateur de la fraction de l'équation (4.31) en bornant supérieurement e_Q^D et d_Q^D, et nous minorerons le dénominateur en bornant inférieurement d_Q^D.

4.5.2 Paire d'espérances de perte et loi trinomiale

À la section 4.2.1, nous avons expliqué que le théorème 4.4 repose en grande partie sur la possibilité d'exprimer l'espérance de perte d'un votant à l'aide d'une *loi binomiale*. Voyons maintenant comment exprimer une paire d'espérances de pertes, en l'occurrence $\mathcal{L}_d$ et $\mathcal{L}_e$, à l'aide d'une ***loi trinomiale***, c'est-à-dire une loi multinomiale à trois résultats possibles.

Pour simplifier l'explication, considérons deux classificateurs $h_i, h_j : \mathcal{X} \to \{-1, 1\}$, et le votant jumelé correspondant $h_{ij} : \mathcal{X} \to \{-1, 1\}^2$. Considérons aussi une distribution de données D. L'erreur conjointe de h_{ij} sur D est $\mathbb{E}_D^{\mathcal{L}_e}(h_{ij})$, le succès conjoint est $\mathbb{E}_D^{\mathcal{L}_s}(h_{ij})$ et le désaccord est $\mathbb{E}_D^{\mathcal{L}_d}(h_{ij})$.

Associons au votant jumelé h_{ij} une variable aléatoire X'_{ij} à trois valeurs possibles : $(1, 0)$, $(0, 1)$ et $(0, 0)$. Attribuons aux événements consistant à observer chacune de ces trois valeurs les probabilités suivantes :

$$\begin{aligned}
\Pr\Big(X'_{ij} = (1,0)\Big) &:= \mathbb{E}_D^{\mathcal{L}_e}(h_{ij})\,, \\
\Pr\Big(X'_{ij} = (0,1)\Big) &:= \mathbb{E}_D^{\mathcal{L}_d}(h_{ij})\,, \\
\Pr\Big(X'_{ij} = (0,0)\Big) &:= \mathbb{E}_D^{\mathcal{L}_s}(h_{ij}) \;=\; 1 - \mathbb{E}_D^{\mathcal{L}_e}(h_{ij}) - \mathbb{E}_D^{\mathcal{L}_d}(h_{ij})\,.
\end{aligned}$$

Notons qu'il est possible de calculer la dernière probabilité à partir des deux premières, ce qui est une conséquence directe de l'équation (4.29).

Considérons maintenant l'expérience aléatoire consistant à générer aléatoirement un échantillon S de m exemples (*i.i.d.* selon la distribution D) et de compter le nombre X_{ij}^e d'erreurs conjointes ainsi que le nombre X_{ij}^d de désaccords des classificateurs h_i et h_j parmi S. Ainsi,

$$X_{ij}^e := \Big|\{(x, y) \in S \mid h_i(x) = h_j(x) \neq y\}\Big| \quad \text{et} \quad X_{ij}^d := \Big|\{(x, y) \in S \mid h_i(x) \neq h_j(x)\}\Big|\,.$$

Cette expérience équivaut à additionner m réalisations de la variable X'_{ij} pour former le couple (X_{ij}^e, X_{ij}^d), où X_{ij}^e s'avère le nombre d'occurrences de la valeur $(1, 0)$ et X_{ij}^d le nombre d'occurrences de la valeur $(0, 1)$. De ce fait, le couple (X_{ij}^e, X_{ij}^d) est une variable aléatoire qui suit une *loi trinomiale* que nous désignons ainsi :

$$(X_{ij}^e, X_{ij}^d) \sim T\Big(m, \mathbb{E}_D^{\mathcal{L}_e}(h_{ij}), \mathbb{E}_D^{\mathcal{L}_d}(h_{ij})\Big)\,.$$

Conséquemment, par la fonction de masse de la loi trinomiale, on peut écrire la probabilité d'observer spécifiquement k_e erreurs conjointes et k_d désaccords parmi m exemples par une expression combinatoire :

$$\begin{aligned}
&\Pr_{S\sim D^m}\left(\left(\mathbb{E}_D^{\mathcal{L}_e}(h_{ij}),\mathbb{E}_D^{\mathcal{L}_d}(h_{ij})\right)=\left(\tfrac{k_e}{m},\tfrac{k_d}{m}\right)\right) \qquad (4.32)\\
&\quad= \Pr_{(X_{ij}^e,X_{ij}^d)\sim T(m,\mathbb{E}_D^{\mathcal{L}_e}(h_{ij}),\mathbb{E}_D^{\mathcal{L}_d}(h_{ij}))}\left(X_{ij}^e=k_e\wedge X_{ij}^d=k_d\right)\\
&\quad= \binom{m}{k_e}\binom{m-k_e}{k_d}\left(\mathbb{E}_D^{\mathcal{L}_e}(h_{ij})\right)^{k_e}\left(\mathbb{E}_D^{\mathcal{L}_d}(h_{ij})\right)^{k_d}\left(1-\mathbb{E}_D^{\mathcal{L}_e}(h_{ij})-\mathbb{E}_D^{\mathcal{L}_d}(h_{ij})\right)^{m-k_e-k_d},
\end{aligned}$$

pour tout $k_e\in\{0,..,m\}$ et pour tout $k_d\in\{0,..,m-k_e\}$.

La section 4.5.3 ci-dessous présente un lemme permettant d'appliquer ce principe aux votants à valeur réelle $f:\mathcal{X}\to[-1,1]$.

4.5.3 Généralisation des lemmes A.12 (Maurer, 2004) et 4.1

Le lemme A.12 (en annexe), démontré par Maurer (2004), permet d'appliquer le théorème PAC-bayésien à des votants à valeurs réelles en exprimant l'espérance de perte du votant par une ***loi binomiale*** (à deux valeurs possibles). À la section 4.2.1 (page 59), nous avons présenté le lemme 4.1, qui formule le résultat de Maurer de manière à pouvoir l'utiliser convenablement dans la démonstration du théorème PAC-bayésien général.

Par le lemme 4.16 ci-dessous, nous généralisons le lemme A.12 à la ***loi trinomiale*** (à trois valeurs possibles). Par la suite, le lemme 4.17 spécialise ce résultat dans une forme adaptée à nos démonstrations PAC-bayésiennes.

Remarque sur la démonstration du lemme 4.16. La démonstration du lemme ci-dessous s'inspire de la démonstration effectuée par Maurer (2004), mais contient plus de détails afin de faciliter sa compréhension. Il est intéressant de constater que, suivant la même technique de démonstration, il est possible de généraliser encore davantage le résultat de Maurer (2004), pour englober des variables aléatoires suivant une loi multinomiale possédant un nombre (fini) quelconque de valeurs possibles. Enfin, notons qu'une partie plus technique de la démonstration qui suit se retrouve en annexe (lemme A.14).

Lemme 4.16. *Soit une variable aléatoire (X,Y) dont la valeur appartient à $[0,1]^2$, telle que $X+Y \leq 1$, et d'espérance $(\mu_X,\mu_Y) \coloneqq (\mathbf{E}(X),\mathbf{E}(Y))$. Étant donné n observations de (X,Y), notons $\mathbf{X} \coloneqq (X_1,\dots,X_n) \in [0,1]^n$ le vecteur des valeurs observées de X, et $\mathbf{Y} \coloneqq (Y_1,\dots,Y_n) \in [0,1]^n$ le vecteur des valeurs observées de Y.*

Considérons une variable aléatoire (X',Y') à trois valeurs possibles, $(1,0)$, $(0,1)$ et $(0,0)$, dont les espérances sont respectivement μ_X, μ_Y et $1-\mu_X-\mu_Y$. Notons $\mathbf{X}',\mathbf{Y}' \in \{0,1\}^n$ les vecteurs contenant n observations indépendantes de (X',Y').

Soit une fonction convexe $\mathcal{F} : [0,1]^n \times [0,1]^n \to \mathbb{R}$, alors

$$\mathbf{E}\left[\mathcal{F}(\mathbf{X},\mathbf{Y})\right] \;\leq\; \mathbf{E}\left[\mathcal{F}(\mathbf{X}',\mathbf{Y}')\right].$$

Démonstration. Soit deux vecteurs $\mathbf{x} \coloneqq (x_1,\dots,x_n)$, $\mathbf{y} \coloneqq (y_1,\dots,y_n) \in [0,1]^n$. On écrit

$$(\mathbf{x},\mathbf{y}) \;\coloneqq\; \big((x_1,y_1),(x_2,y_2),\dots,(x_n,y_n)\big) \;\in\; ([0,1]\times[0,1])^n.$$

Soit $H \coloneqq \{(1,0),(0,1),(0,0)\}$. Le lemme A.14 (en annexe) montre que tout point $(\mathbf{x},\mathbf{y})$ peut être exprimé comme une combinaison convexe des points extrêmes $\boldsymbol{\eta} \coloneqq (\eta_1,\eta_2,\dots,\eta_n) \in H^n$:

$$(\mathbf{x},\mathbf{y}) \;=\; \sum_{\boldsymbol{\eta}\in H^n} \left[\prod_{i=1}^{n} \mathcal{G}_{\eta_i}(x_i,y_i)\right] \cdot \boldsymbol{\eta}\,, \tag{4.33}$$

où

$$\mathcal{G}_{\eta_i}(x_i,y_i) \;\coloneqq\; \begin{cases} x_i & \text{si } \eta_i = (1,0)\,, \\ y_i & \text{si } \eta_i = (0,1)\,, \\ 1-x_i-y_i & \text{sinon.} \end{cases}$$

La convexité de la fonction $\mathcal{F}$ implique

$$\mathcal{F}(\mathbf{x},\mathbf{y}) \;\leq\; \sum_{\boldsymbol{\eta}\in H^n} \left[\prod_{i=1}^{n} \mathcal{G}_{\eta_i}(x_i,y_i)\right] \cdot \mathcal{F}(\boldsymbol{\eta})\,. \tag{4.34}$$

Remarque $(*)$: On a l'égalité à l'équation (4.34) si $(\mathbf{x},\mathbf{y}) \in H^n = \{(1,0),(0,1),(0,0)\}^n$, car les éléments de la somme sont égaux à $1\cdot\mathcal{F}(\boldsymbol{\eta})$ seulement pour $\boldsymbol{\eta} = (\mathbf{x},\mathbf{y})$ (et $0\cdot\mathcal{F}(\boldsymbol{\eta})$ pour tout $\boldsymbol{\eta} \in H^n \setminus \{(\mathbf{x},\mathbf{y})\}$).

Étant donné que les réalisations de la variable aléatoire (X,Y) sont indépendantes, nous avons

$$\begin{aligned}
\mathbf{E}\left[\mathcal{F}(\mathbf{X},\mathbf{Y})\right] \;&\leq\; \mathbf{E}\left[\sum_{\boldsymbol{\eta}\in H^n} \left[\prod_{i=1}^{n} \mathcal{G}_{\eta_i}(X_i,Y_i)\right] \cdot \mathcal{F}(\boldsymbol{\eta})\right] \\
&=\; \sum_{\boldsymbol{\eta}\in H^n} \mathbf{E}\left[\prod_{i=1}^{n} \mathcal{G}_{\eta_i}(X_i,Y_i)\right] \cdot \mathcal{F}(\boldsymbol{\eta}) \\
&=\; \sum_{\boldsymbol{\eta}\in H^n} \left[\prod_{i=1}^{n} \mathcal{G}_{\eta_i}\big(\mathbf{E}(X_i),\mathbf{E}(Y_i)\big)\right] \cdot \mathcal{F}(\boldsymbol{\eta}) \\
&=\; \sum_{\boldsymbol{\eta}\in H^n} \left[\prod_{i=1}^{n} \mathcal{G}_{\eta_i}(\mu_X,\mu_Y)\right] \cdot \mathcal{F}(\boldsymbol{\eta})\,.
\end{aligned}$$

Étant donné que $(\mathbf{X}', \mathbf{Y}') \in H^n$, par la remarque $(*)$, on a

$$\mathbf{E}\left[\mathcal{F}(\mathbf{X}', \mathbf{Y}')\right] \;=\; \sum_{\boldsymbol{\eta} \in H^n} \left[\prod_{i=1}^{n} \mathcal{G}_{\eta_i}(\mu_X, \mu_Y)\right] \cdot \mathcal{F}(\boldsymbol{\eta})\,.$$

Ainsi, $\mathbf{E}\left[\mathcal{F}(\mathbf{X}, \mathbf{Y})\right] \;\leq\; \mathbf{E}\left[\mathcal{F}(\mathbf{X}', \mathbf{Y}')\right]$. □

Intéressons-nous maintenant à l'espérance d'une fonction qui dépend à la fois de l'erreur conjointe d'un votant jumelé et de son désaccord, c'est-à-dire au terme de gauche de l'équation (4.35) ci-bas. Une fois cette espérance bornée par l'espérance d'une variable aléatoire distribuée selon une loi trinomiale (grâce au lemme 4.16), nous retrouvons le «cas binaire» de la section 4.5.2. Nous évaluons ensuite l'espérance à l'aide d'une expression combinatoire, conformément à l'équation (4.32), ce qui permet d'obtenir le terme de droite de l'équation (4.35).

Lemme 4.17. *Soit une distribution D sur $\mathcal{X} \times \{-1, 1\}$, un votant jumelé $f_{ij} : \mathcal{X} \to [-1, 1]^2$ d'erreur conjointe $\mathbb{E}_D^{\mathcal{L}_e}(f_{ij})$ et de désaccord $\mathbb{E}_D^{\mathcal{L}_d}(f_{ij})$, ainsi qu'une fonction convexe $g : [0, 1]^2 \to \mathbb{R}$. On a*

$$\begin{aligned}
&\mathop{\mathbf{E}}_{S \sim D^m} g\left(\mathbb{E}_S^{\mathcal{L}_e}(f_{ij}), \mathbb{E}_S^{\mathcal{L}_d}(f_{ij})\right) \qquad (4.35)\\
&\leq \sum_{k_1=0}^{m} \sum_{k_2=0}^{m-k_1} \binom{m}{k_1}\binom{m-k_1}{k_2}\left(\mathbb{E}_D^{\mathcal{L}_e}(f_{ij})\right)^{k_1}\left(\mathbb{E}_D^{\mathcal{L}_d}(f_{ij})\right)^{k_2}\left(1-\mathbb{E}_D^{\mathcal{L}_e}(f_{ij})-\mathbb{E}_D^{\mathcal{L}_d}(f_{ij})\right)^{m-k_1-k_2} \times g\Big(\tfrac{k_1}{m}, \tfrac{k_2}{m}\Big).
\end{aligned}$$

Démonstration. Considérons une variable aléatoire X'_{ij} à trois valeurs possibles. La probabilité d'observer la première valeur est $R_e \coloneqq \mathbb{E}_D^{\mathcal{L}_e}(f_{ij})$, la probabilité d'observer la seconde valeur est $R_d \coloneqq \mathbb{E}_D^{\mathcal{L}_d}(f_{ij})$ et la probabilité d'observer la dernière valeur est $1 - R_e - R_d$. Étant donné m observations de X'_{ij}, notons X_{ij}^e le nombre de fois que la première valeur est observée, puis notons X_{ij}^d le nombre de fois que la seconde valeur est observée. Ainsi, le couple (X_{ij}^e, X_{ij}^d) suit une loi trinomiale :

$$(X_{ij}^e, X_{ij}^d) \;\sim\; T\Big(m,\, R_e,\, R_d\Big)\,.$$

Comme $g(\cdot, \cdot)$ est une fonction convexe, le lemme 4.16 nous permet d'obtenir la borne supérieure suivante :

$$\begin{aligned}
&\mathop{\mathbf{E}}_{S \sim D^m} g\left(\mathbb{E}_S^{\mathcal{L}_e}(f_{ij}), \mathbb{E}_S^{\mathcal{L}_d}(f_{ij})\right)\\
&\qquad\leq \mathop{\mathbf{E}}_{(X_{ij}^e, X_{ij}^d) \sim T(m,\, R_e,\, R_d)} g\Big(\tfrac{1}{m}X_{ij}^e,\, \tfrac{1}{m}X_{ij}^d\Big)\\
&\qquad= \sum_{k_1=0}^{m} \sum_{k_2=0}^{m-k_1} \Pr_{(X_{ij}^e, X_{ij}^d) \sim T(m,\, R_e,\, R_d)}\left(X_{ij}^e = k_1 \wedge X_{ij}^d = k_2\right)\, g\Big(\tfrac{X_{ij}^e}{m},\, \tfrac{X_{ij}^d}{m}\Big)\\
&\qquad= \sum_{k_1=0}^{m} \sum_{k_2=0}^{m-k_1} \tbinom{m}{k_1}\tbinom{m-k_1}{k_2}(R_e)^{k_1}\,(R_d)^{k_2}(1-R_e-R_d)^{m-k_1-k_2} \times g\Big(\tfrac{k_1}{m},\, \tfrac{k_2}{m}\Big)\,.
\end{aligned}$$

Notons que la dernière égalité découle de la fonction de masse de la loi trinomiale, comme nous l'avons présentée à l'équation (4.32). □

Nous avons maintenant présenté les concepts nécessaires permettant d'énoncer un théorème PAC-bayésien qui borne *simultanément* une paire d'espérances de perte.

4.5.4 Théorème PAC-bayésien pour deux pertes de votants jumelés

Le théorème 4.19 ci-dessous permet de borner *simultanément* l'espérance de succès conjoint et l'espérance de désaccord d'un même vote de majorité.

Comme le théorème présenté en début de chapitre (théorème 4.4, page 61), le théorème 4.19 est présenté sous une forme générale qui est valide pour une famille de fonctions de «distance», que nous désignons par le terme $\boldsymbol{\Delta_2}$-fonction.

Définition 4.18. Une ***$\boldsymbol{\Delta_2}$-fonction*** désigne une fonction convexe $\boldsymbol{\Delta_2} : [0,1]^2 \times [0,1]^2 \to \mathbb{R}$.

Étant donné que nous désirons borner deux espérances de perte, une $\boldsymbol{\Delta_2}$-fonction dépend de quatre arguments : une paire d'espérances de perte empiriques $(e_Q^S,\, d_Q^S)$ évaluées sur un échantillon d'entraînement, ainsi qu'une paire d'espérances de perte «réelles» $(e_Q^D,\, d_Q^D)$ correspondant à l'espérance d'erreur conjointe et l'espérance de désaccord sur la distribution génératrice D.

Remarque sur la démonstration du théorème 4.19. La démonstration du théorème ci-dessous reprend les grandes étapes de la démonstration du théorème 4.4 (page 61), en l'adaptant notamment aux concepts de $\boldsymbol{\Delta_2}$-fonction et de variable aléatoire suivant une loi trinomiale. Nous réutilisons aussi quelques astuces de la démonstration du corollaire 4.12 (page 72) propre aux distributions de votants jumelés.

Théorème 4.19. *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to [-1,1]$, pour toute distribution P sur $\mathcal{H}$, pour tout $\delta \in (0,1]$, et pour toute fonction convexe $\boldsymbol{\Delta_2} : [0,1]^2 \times [0,1]^2 \to \mathbb{R}$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S \sim D^m$,*

$$\forall\, Q \text{ sur } \mathcal{H} : \quad \boldsymbol{\Delta_2}\Big(e_Q^S,\, d_Q^S\,;\, e_Q^D,\, d_Q^D\Big) \;\leq\; \frac{1}{m}\left[2\,\mathrm{KL}(Q\|P) + \ln\frac{\mathcal{I}_{\boldsymbol{\Delta_2}}(m)}{\delta}\right],$$

où

$$\mathcal{I}_{\boldsymbol{\Delta_2}}(m) \;\stackrel{\text{def}}{=}\; \sup_{\substack{r,s\in[0,1]\\ r+s\leq 1}} \left[\sum_{k_1=0}^{m}\sum_{k_2=0}^{m-k_1}\binom{m}{k_1}\binom{m-k_1}{k_2} r^{k_1}\, s^{k_2}\, (1-r-s)^{m-k_1-k_2}\, e^{m\boldsymbol{\Delta_2}\left(\frac{k_1}{m},\,\frac{k_2}{m}\,;\,r,\,s\right)}\right].$$

Démonstration. Soit l'ensemble de votants jumelés $\mathcal{H}^2 \stackrel{\text{def}}{=} \{f_{ij} \mid f_i, f_j \in \mathcal{H}\}$ et la distribution P^2 sur $\mathcal{H}^2$ définie par $P^2(f_{ij}) \stackrel{\text{def}}{=} P(f_i) \cdot P(f_j)$.

Nous désirons borner supérieurement la quantité $\boldsymbol{\Delta_2}\Big(\, e_Q^S,\, d_Q^S\,;\, e_Q^D,\, d_Q^D\,\Big)$.

Nous appliquons d'abord l'inégalité de Jensen (lemme A.2) sur la fonction convexe $\boldsymbol{\Delta_2}(\cdot,\cdot\,;\,\cdot,\cdot)$, puis l'inégalité du changement de mesure (lemme 4.3, page 60) sur la fonction $\phi : \mathcal{H}^2 \to \mathbb{R}$ suivante (définie sur l'ensemble des votants jumelés) :

$$\begin{aligned}\phi(f) &\coloneqq m \cdot \boldsymbol{\Delta_2}\Big(\mathbb{E}_S^{\mathcal{L}_e}(f_{ij}),\, \mathbb{E}_S^{\mathcal{L}_d}(f_{ij})\,;\, \mathbb{E}_D^{\mathcal{L}_e}(f_{ij}),\, \mathbb{E}_D^{\mathcal{L}_d}(f_{ij})\Big)\\ &= m \cdot \boldsymbol{\Delta_2}\Big(e_{ij}^S,\, d_{ij}^S\,;\, e_{ij}^D,\, d_{ij}^D\Big)\,,\end{aligned}$$

où nous utilisons les notations abrégées $e_{ij}^{D'} \stackrel{\text{def}}{=} \mathbb{E}_{D'}^{\mathcal{L}_e}(f_{ij})$ et $d_{ij}^{D'} \stackrel{\text{def}}{=} \mathbb{E}_{D'}^{\mathcal{L}_d}(f_{ij})$.

Ainsi, pour tout ensemble S généré par la distribution D, on obtient

$\forall\, Q$ sur $\mathcal{H}$:

$$\begin{aligned}& m \cdot \boldsymbol{\Delta_2}\Big(\, e_Q^S,\, d_Q^S\,;\, e_Q^D,\, d_Q^D\,\Big)\\ &\quad= m \cdot \boldsymbol{\Delta_2}\Big(\mathop{\mathbf{E}}_{f_{ij}\sim Q^2} e_{ij}^S,\, \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} d_{ij}^S;\, \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} e_{ij}^D,\, \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} d_{ij}^D\Big) && \langle\ \textit{Définition 4.14}\ \rangle\\ &\quad\le \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} m \cdot \boldsymbol{\Delta_2}\Big(e_{ij}^S,\, d_{ij}^S\,;\, e_{ij}^D,\, d_{ij}^D\Big) && \langle\ \textit{Inégalité de Jensen}\ \rangle\\ &\quad\le \mathrm{KL}(Q^2\|P^2) + \ln\Big(\mathop{\mathbf{E}}_{f_{ij}\sim P^2} e^{m\boldsymbol{\Delta_2}\left(e_{ij}^S, d_{ij}^S\,;\, e_{ij}^D, d_{ij}^D\right)}\Big) && \langle\ \textit{Changement de mesure}\ \rangle\\ &\quad= 2\cdot\mathrm{KL}(Q\|P) + \ln\Big(\underbrace{\mathop{\mathbf{E}}_{f_{ij}\sim P^2} e^{m\boldsymbol{\Delta_2}\left(e_{ij}^S, d_{ij}^S\,;\, e_{ij}^D, d_{ij}^D\right)}}_{X_{P^2}(S)}\Big)\,. && \langle\ \textit{Équation (4.23)}\ \rangle\end{aligned}$$

Considérons maintenant la variable aléatoire $X_{P^2}(S) \coloneqq \mathbf{E}_{f_{ij}\sim P^2}\, e^{m\boldsymbol{\Delta_2}\left(e_{ij}^S, d_{ij}^S\,;\, e_{ij}^D, d_{ij}^D\right)}$, qui correspond à l'expression désignée ci-haut par une accolade horizontale. Par l'inégalité de Markov (lemme A.1, en annexe), on a $\Pr_{S\sim D^m}\left(X_{P^2}(S) \le \frac{1}{\delta}\, \mathop{\mathbf{E}}_{S'\sim D^m} X_{P^2}(S')\right) \ge 1-\delta\,.$

Ainsi, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$,

$$\forall\, Q \text{ sur } \mathcal{H} : \quad m \cdot \boldsymbol{\Delta_2}\Big(\, e_Q^S,\, d_Q^S\,;\, e_Q^D,\, d_Q^D\,\Big) \;\le\; 2\cdot\mathrm{KL}(Q\|P) + \ln\frac{1}{\delta}\, \mathop{\mathbf{E}}_{S'\sim D^m} X_{P^2}(S')\,. \tag{4.36}$$

Comme la distribution *a priori* P^2 est indépendante de S', on peut inverser l'ordre des espérances dans le terme de droite. Cela nous permet ensuite d'appliquer, à la ligne $(\star)$ ci-bas, le lemme 4.17 avec la fonction convexe $g(\cdot,\cdot) \coloneqq e^{m\boldsymbol{\Delta_2}(\,\cdot\,,\,\cdot\,;\, e_{ij}^D,\, d_{ij}^D)}$:

$$
\begin{aligned}
&\mathop{\mathbf{E}}_{S'\sim D^m} X_{P^2}(S') \\
&= \mathop{\mathbf{E}}_{S'\sim D^m} \mathop{\mathbf{E}}_{f_{ij}\sim P^2} e^{m\boldsymbol{\Delta_2}\left(e^S_{ij}, d^S_{ij}\,;\, e^D_{ij}, d^D_{ij}\right)} \\
&= \mathop{\mathbf{E}}_{f_{ij}\sim P^2} \mathop{\mathbf{E}}_{S'\sim D^m} e^{m\boldsymbol{\Delta_2}\left(e^S_{ij}, d^S_{ij}\,;\, e^D_{ij}, d^D_{ij}\right)} \qquad \langle\ \textit{Inversion des espérances}\ \rangle \\
&\leq \mathop{\mathbf{E}}_{f_{ij}\sim P^2}\left[\sum_{k_1=0}^{m}\sum_{k_2=0}^{m-k_1}\tbinom{m}{k_1}\tbinom{m-k_1}{k_2}\left(e^D_{ij}\right)^{k_1}\left(d^D_{ij}\right)^{k_2}\left(1-e^D_{ij}-d^D_{ij}\right)^{m-k_1-k_2} e^{m\boldsymbol{\Delta_2}\left(\frac{k_1}{m},\frac{k_2}{m}\,;\, e^D_{ij}, d^D_{ij}\right)}\right] \quad (\star) \\
&\leq \sup_{\substack{r,s\in[0,1]\\ r+s\leq 1}}\left[\sum_{k_1=0}^{m}\sum_{k_2=0}^{m-k_1}\tbinom{m}{k_1}\tbinom{m-k_1}{k_2}\, r^{k_1}\, s^{k_2}\ (1-r-s)^{m-k_1-k_2}\, e^{m\boldsymbol{\Delta_2}\left(\frac{k_1}{m},\frac{k_2}{m}\,;\, r, s\right)}\right] \\
&= \mathcal{I}_{\boldsymbol{\Delta_2}}(m)\,.
\end{aligned}
$$

Le résultat est alors obtenu en substituant $\mathbf{E}_{S'\sim D^m} X_{P^2}(S')$ par $\mathcal{I}_{\boldsymbol{\Delta_2}}(m)$ dans l'équation (4.36), et en divisant par m des deux côtés de l'inégalité. □

Il est maintenant possible de spécialiser le théorème 4.19 en utilisant une $\boldsymbol{\Delta_2}$-fonction de notre choix.

4.5.5 Une Δ_2-fonction basée sur la divergence Kullback-Leibler

Dans le texte qui suit, nous spécialisons le théorème 4.19 à une $\boldsymbol{\Delta_2}$-fonction inspirée de la Δ-fonction $\mathrm{kl}(q,p)$, présentée à l'équation (4.6) et exploitée dans plusieurs théorèmes PAC-bayésiens depuis Seeger (2002). Rappelons que la fonction $\mathrm{kl}(q,p)$ est définie comme la *divergence Kullback-Leibler* entre deux distributions de Bernoulli. Chacune de ces distributions représente donc une variable aléatoire à *deux valeurs possibles*, dont les probabilités sont q et $1-q$ pour la première distribution, et p et $1-p$ pour la deuxième distribution.

Considérons maintenant la ***divergence Kullback-Leibler*** entre deux distributions de probabilités de variables aléatoires à *trois valeurs possibles*. Notons q_1, q_2 et $1-q_1-q_2$ les probabilités associées à chacune des trois valeurs pour la première distribution ; notons p_1, p_2 et $1-p_1-p_2$ les probabilités associées à chacune des trois valeurs pour la deuxième distribution. La divergence Kullback-Leibler entre ces deux distributions est

$$
\mathrm{kl}_{\boldsymbol{2}}(q_1, q_2\,;\, p_1, p_2) \stackrel{\text{def}}{=} q_1 \ln\frac{q_1}{p_1} + q_2\ln\frac{q_2}{p_2} + (1-q_1-q_2)\ln\frac{1-q_1-q_2}{1-p_1-p_2}\,. \tag{4.37}
$$

Précisons que la fonction $\mathrm{kl}_{\boldsymbol{2}}(q_1, q_2\,;\, p_1, p_2)$ est convexe, comme le démontre le corollaire A.5 (en annexe).

Il été démontré par Younsi (2012) que $\mathcal{I}_{\mathrm{kl}_{\mathbf{2}}}(m) = \mathcal{I}_{\mathrm{kl}}(m) + m$. En combinant ce résultat avec l'équation (4.9) – due à Maurer (2004) – on obtient

$$\sqrt{m} + m \;\leq\; \mathcal{I}_{\mathrm{kl}_{\mathbf{2}}}(m) \;=\; \mathcal{I}_{\mathrm{kl}}(m) + m \;\leq\; 2\sqrt{m} + m\,, \tag{4.38}$$

ce qui permet de formuler le corollaire suivant.

Corollaire 4.20. *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to [-1, 1]$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta \in (0, 1]$, on a, avec probabilité au moins $1 - \delta$ sur le choix de $S \sim D^m$,*

$$\forall\, Q \text{ sur } \mathcal{H} : \quad \mathrm{kl}_{\mathbf{2}}\Big(\, e_Q^S,\, d_Q^S \,;\, e_Q^D,\, d_Q^D \,\Big) \;\leq\; \frac{1}{m}\left[2\,\mathrm{KL}(Q\|P) + \ln\frac{2\sqrt{m}+m}{\delta}\right]. \tag{4.39}$$

Démonstration. Le résultat découle directement du théorème 4.19, avec $\Delta_{\mathbf{2}} \coloneqq \mathrm{kl}_{\mathbf{2}}$ (le corollaire A.5, en annexe, montre que $\mathrm{kl}_{\mathbf{2}}$ est une fonction convexe). De plus, par l'équation (4.38), on a que $\mathcal{I}_{\mathrm{kl}_{\mathbf{2}}}(m) \leq 2\sqrt{m} + m$. □

Le corollaire 4.20 permet de borner la $\mathcal{C}$-borne, comme nous le montrons à la section 4.5.6 ci-bas.

4.5.6 Une borne sur le risque de Bayes via la $\mathcal{C}$-borne

Le corollaire 4.20 permet de borner *simultanément* l'espérance d'erreur conjointe e_Q^D et l'espérance de désaccord d_Q^D à partir de leur contrepartie empirique e_Q^S et d_Q^S, mesurée sur un échantillon d'entraînement S. Plus précisément, l'inégalité (4.39) définit un ensemble de valeurs possibles (avec probabilité au moins $1 - \delta$) pour la paire (e_Q^D, d_Q^D). Pour borner supérieurement la valeur $\mathcal{C}_Q^D$, nous cherchons parmi cet ensemble la paire de valeurs maximisant la $\mathcal{C}$-borne sous la forme donnée par l'équation (4.31), c'est-à-dire

$$\mathcal{C}_Q^D \;=\; 1 - \frac{\Big(1 - (2e_Q^D + d_Q^D)\Big)^2}{1 - 2d_Q^D}\,.$$

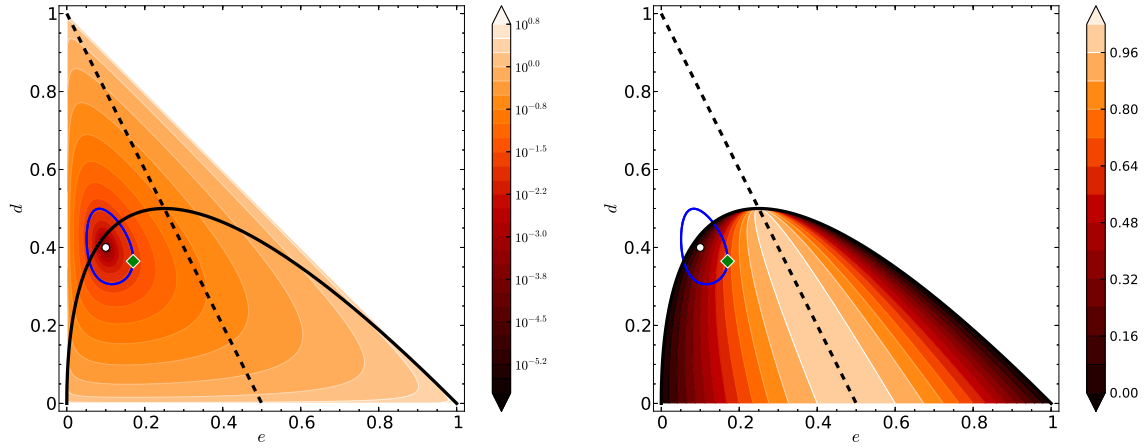
Rappelons que la $\mathcal{C}$-borne est seulement valide lorsque $R_D(G_Q) \leq 0.5$ (voir théorème 3.7, page 49). Conséquemment, cette méthode s'applique lorsque la paire (e_Q^D, d_Q^D) est telle que

$$R_D(G_Q) \;=\; \frac{1}{2}\left(2e_Q^D + d_Q^D\right) \;\leq\; \frac{1}{2}\,, \tag{4.40}$$

où l'égalité provient de l'équation (4.30).

Finalement, la proposition 3.5 (page 47), avec l'équation (4.30), nous permet de restreindre l'ensemble des paires (e_Q^D, d_Q^D) possibles :

$$\left.\begin{array}{rcl} d_Q^D & \leq & 2 \cdot R_D(G_Q) \cdot (1 - R_D(G_Q)) \\ & = & 2 \cdot (e_Q^D + \frac{1}{2} d_Q^D) \cdot (1 - (e_Q^D + \frac{1}{2} d_Q^D)) \end{array}\right\} \iff \quad d_Q^D \;\leq\; 2 \cdot \left(\sqrt{e_Q^D} - e_Q^D\right). \tag{4.41}$$



(a) Courbes de niveau de $\mathrm{kl}_{\mathbf{2}}(0.1, 0.4; e, d)$.

(b) Courbes de niveau de $F_{\mathcal{C}}(e, d)$.

FIGURE 4.2 – Illustration du calcul de la borne 4.21. On suppose $\mathrm{KL}(Q\|P) := 5$, $m := 1000$ et $\delta := 0.05$. Si on observe une espérance d'erreur conjointe empirique $e_Q^S := 0.10$ et une espérance de désaccord empirique $d_Q^S := 0.40$ (le risque de Gibbs empirique vaut donc $R_S(G_Q) = 0.1 + \frac{1}{2} \cdot 0.4 = 0.30$), alors nous devons maximiser $F_{\mathcal{C}}(e, d)$ à l'intérieur du domaine $\mathcal{A}_{Q,S}^{\delta}$ définit par trois contraintes : $\mathrm{kl}_{\mathbf{2}}(0.1, 0.4\,;\, e, d) \leq \frac{1}{m}\big[2 \cdot \mathrm{KL}(Q\|P) + \ln \frac{2\sqrt{m}+m}{\delta}\big] \approx 0.02$ (ovale bleu), $d \leq 2(\sqrt{e} - e)$ (courbe noire) et $2e + d < 1$ (ligne noire pointillée). Ainsi, la borne obtenue est $R_D(B_Q) \leq 0.679$ (correspondant au couple $(e, d) \approx (0.17, 0.36)$, indiqué par le losange vert).

Ces considérations nous permettent de formuler la borne suivante sur le risque du vote de majorité $R_D(B_Q)$.

Borne du vote de majorité 4.21. *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to [-1, 1]$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta \in (0, 1]$, on a, avec probabilité au moins $1 - \delta$ sur le choix de $S \sim D^m$,*

$$\forall\, Q \text{ sur } \mathcal{H} : \quad R_D(B_Q) \;\leq\; \mathcal{C}_Q^D \;\leq \sup_{(e,d) \in \mathcal{A}_{Q,S}^{\delta}} \left[1 - \frac{\big(1 - (2e + d)\big)^2}{1 - 2d} \right],$$

où $\mathcal{A}_{Q,S}^{\delta} \stackrel{\text{def}}{=}$

$$\left\{ (e, d) \in [0, \tfrac{1}{2}]^2 \;\middle|\; \mathrm{kl}_{\mathbf{2}}(e_Q^S\, d_Q^S; e, d) \leq \frac{1}{m}\Big[2\,\mathrm{KL}(Q\|P) + \ln \tfrac{2\sqrt{m}+m}{\delta}\Big],\; d \leq 2(\sqrt{e} - e),\; 2e + d < 1 \right\}.$$

Pour calculer la valeur de la borne 4.21 nous considérons une fonction de deux variables :

$$F_{\mathcal{C}}(e, d) \;\stackrel{\text{def}}{=}\; 1 - \frac{\big[\,1 - (2e + d)\,\big]^2}{1 - 2d}\,, \tag{4.42}$$

avec $e \in [0, 1]$ et $d \in [0, \frac{1}{2}]$. La proposition A.13 (en annexe) montre que la fonction $F_{\mathcal{C}}(e, d)$ est concave.

Ainsi, la valeur de la borne supérieure de $R_D(B_Q)$ est obtenue en maximisant la valeur de $F_{\mathcal{C}}(e,d)$ sur le domaine $\mathcal{A}^{\delta}_{Q,S}$, qui est à la fois borné et convexe. Une procédure d'optimisation possible s'avère de décomposer la fonction $F_{\mathcal{C}}(e,d)$ en deux fonctions imbriquées

$$\sup_{(e,d)\in\mathcal{A}^{\delta}_{Q,S}}\Big[F_{\mathcal{C}}(e,d)\Big] \;=\; \sup_{d:(\cdot,d)\in\mathcal{A}^{\delta}_{Q,S}}\Big[F^{*}_{\mathcal{C}}(d)\Big], \quad \text{avec } F^{*}_{\mathcal{C}}(d) \stackrel{\text{def}}{=} \sup_{e:(e,d)\in\mathcal{A}^{\delta}_{Q,S}}\Big[F_{\mathcal{C}}(e,d)\Big],$$

et d'implémenter la maximisation de $F_{\mathcal{C}}(e,d)$ en utilisant un algorithme d'optimisation en une dimension deux fois. La figure 4.2 illustre un exemple de calcul de la borne PAC-borne 4.21.

4.6 Comparaison empirique des bornes inductives sur le risque de Bayes

Nous présentons maintenant une étude empirique des trois bornes sur le risque du vote de majorité contenues dans ce chapitre, c'est-à-dire les bornes 4.9, 4.13 et 4.21. Rappelons les caractéristiques principales de chacune de ces bornes du risque du vote de majorité :

— La borne du vote de majorité 4.9 (page 69) correspond au résultat «classique» de la théorie PAC-bayésienne (Langford et Seeger, 2001). Par le biais de la *borne du facteur deux* (proposition 3.6), la borne sur le risque de Bayes est obtenue en multipliant par deux la borne sur le risque de Gibbs.

— La borne du vote de majorité 4.13 (page 74) est obtenue par le biais de la $\mathcal{C}$*-borne* (théorème 3.7), en ayant recours à deux estimations PAC-bayésiennes : la première est une borne supérieure sur le risque de Gibbs et la deuxième est une borne inférieure sur l'espérance de désaccord.

— La borne du vote de majorité 4.21 (page 86) est aussi obtenue par le biais de la $\mathcal{C}$*-borne* (théorème 3.7), mais repose sur une seule estimation PAC-bayésienne. Ce faisant, on borne *simultanément* l'espérance d'erreur conjointe et l'espérance de désaccord.

Afin de comparer empiriquement ces trois bornes du vote de majorité, nous utilisons l'algorithme d'apprentissage ***AdaBoost*** – tel que décrit à la section 2.4.2 (page 32)– en utilisant ensemble de votants $\mathcal{H}$ constitué de *souches de décision*.

Nous effectuons l'expérimentation sur l'échantillon de données «Mushrooms», provenant du «UCI Machine Learning Repository» (Bache et Lichman, 2013), qui contient 8124 exemples. Nous exécutons l'algorithme AdaBoost pour 60 itérations sur un échantillon d'entraînement S contenant 4062 exemples. Nous calculons, à chaque itération, chacune des trois bornes sur le risque du vote de majorité, ainsi que la valeur de l'espérance de désaccord d^{S}_{Q}, du risque de Gibbs $R_S(G_Q)$ et de la $\mathcal{C}$-borne $\mathcal{C}^{S}_{Q}$ sur l'échantillon d'entraînement S. Enfin, nous calculons aussi le risque $R_T(B_Q)$ sur l'échantillon de test T, constitué des 4062 exemples de l'échantillon de données «Mushrooms» qui n'ont pas servi à l'entraînement. Notons que, pour tous ces calculs, nous considérons une distribution *a priori* P sur $\mathcal{H}$ uniforme (c'est-à-dire $P(h) := \frac{1}{|\mathcal{H}|}$

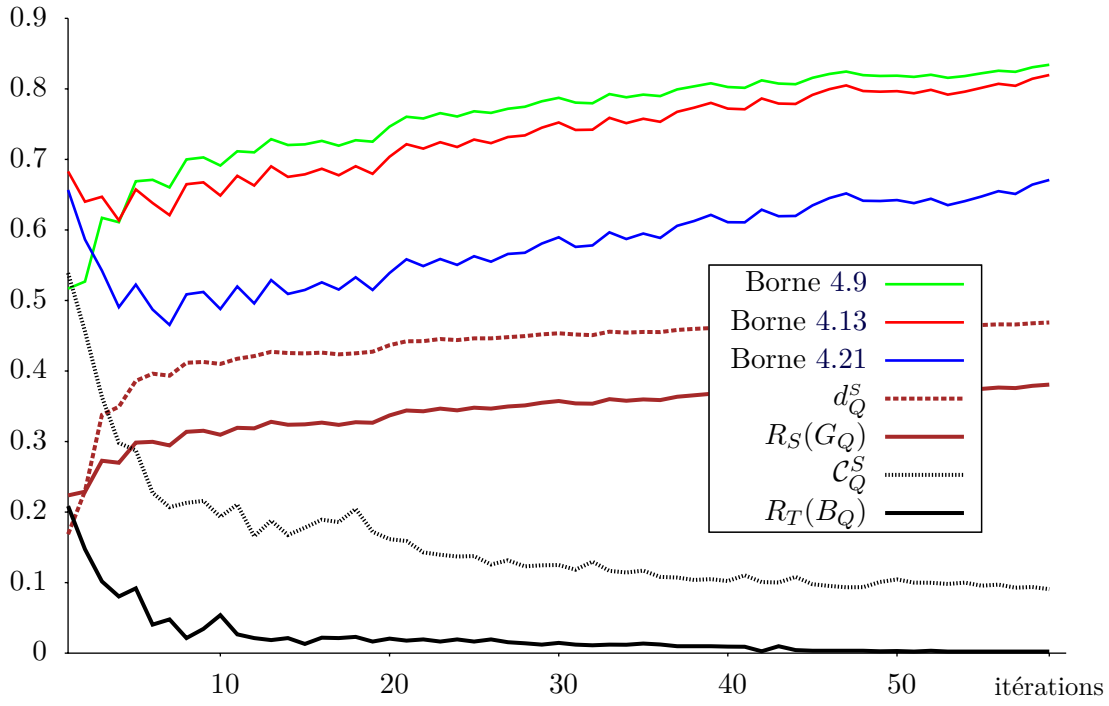


FIGURE 4.3 – Comparaison des bornes sur le risque du vote de majorité pendant les 60 itérations de l'algorithme *AdaBoost*.

pour tout $h \in \mathcal{H}$). La distribution *a posteriori* Q correspond au vote de majorité construit par AdaBoost, en normalisant les poids donnés par l'algorithme, comme il est montré à l'équation (2.28). L'annexe B.2 présente plus en détail la procédure expérimentale utilisée.

La figure 4.3 illustre les résultats obtenus. Nous constatons que les deux bornes basées sur la $\mathcal{C}$-borne (bornes 4.13 et 4.21) sont inférieures à la borne basée sur le facteur deux (borne 4.9) après quelques itérations. Nous constatons aussi qu'il y a un avantage certain à borner directement la $\mathcal{C}$-borne (borne 4.13), plutôt que de borner séparément le risque de Gibbs et l'espérance de désaccord (borne 4.21).

Malheureusement, aucune des trois bornes obtenues ne reflète le comportement du risque du vote de majorité adéquatement. Malgré le fait que $\mathcal{C}_Q^S$ diminue au fil des itérations, les bornes PAC-bayésiennes augmentent rapidement. Plus précisément, la borne 4.9, basée sur la borne du facteur deux, dégrade dès la première itération. Cela s'explique par le fait que le risque de Gibbs $R_S(G_Q)$ augmente au fil des itérations (ce qui est un comportement fréquemment observé lors de l'algorithme AdaBoost). L'augmentation du risque de Gibbs $R_S(G_Q)$ est en partie compensée par l'augmentation de l'espérance de désaccord d_Q^S dans le cas des bornes 4.13 et 4.21, basées sur la $\mathcal{C}$-borne. Cependant, ces deux bornes commencent à se dégrader après 8 itérations. Dans ce cas-ci, cela s'explique par le fait que le dénominateur de la $\mathcal{C}$-borne s'approche de 0 (voir théorème 3.7). Autrement dit, la borne inférieure de l'espérance

de désaccord est près de 0.5, ce qui entraîne une dégradation de la borne.

4.7 Synthèse des contributions du chapitre

La principale contribution de ce chapitre se situe dans la conceptualisation de la théorie PAC-bayésienne, sur laquelle reposent plusieurs résultats qui sont présentés dans les chapitres à venir (partie III de la thèse).

Nous avons commencé le chapitre en présentant un théorème PAC-bayésien pour le cadre d'apprentissage inductif (théorème 4.4, page 61). Ce théorème, se voulant très général, possède les caractéristiques suivantes :

— Il est valide pour toute espérance de perte de la forme $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0,1]$;

— La «distance» entre l'espérance de perte empirique et l'espérance de perte sur la distribution génératrice est exprimée par une fonction convexe $\Delta : [0,1] \times [0,1] \to \mathbb{R}$, que l'on nomme dans la thèse Δ-*fonction* ;

— L'influence du choix de Δ-fonction, ainsi que l'importance de la loi binomiale dans le cadre inductif, est mis en évidence par le terme $\mathcal{I}_\Delta(m)$;

— La technique de démonstration employée pour prouver le théorème facilite sa compréhension.

Nous avons ensuite montré par quelles Δ-fonction nous retrouvons, une fois le théorème spécialisé au *risque de Gibbs* $R_D(G_Q)$, les théorèmes PAC-bayésiens de McAllester; Langford et Seeger; Catoni (corollaire 4.8, page 68). Au passage, nous illustrons le calcul d'une borne sur le *risque de Bayes* $R_D(B_Q)$ à l'aide du théorème de Langford et Seeger et de la *borne du facteur deux* (borne 4.9, page 69).

Ensuite, par la définition du concept de *votant jumelé* f_{ij} et d'une fonction de perte $\mathcal{L}_d$ appropriée (définitions 4.10 et 4.11, page 71), nous avons formulé des garanties de généralisation sur l'*espérance de désaccord* d_Q^D (corollaire 4.12, page 72). Cela nous permet de calculer une première borne sur le *risque de Bayes* $R_D(B_Q)$ à l'aide de la $\mathcal{C}$-*borne* (borne 4.13, page 74)

Le concept de *votant jumelé* nous a aussi permis de formuler un théorème PAC-bayésien pour borner *simultanément* deux espérances de perte (théorème 4.19, page 82). Pour arriver à ces fins, nous avons dû :

— Introduire les notions d'*erreur conjointe* e_Q^D et de *succès conjoint* s_Q^D d'un vote de majorité, ainsi que les fonctions de pertes $\mathcal{L}_e$ et $\mathcal{L}_d$ associées (définitions 4.14 et 4.15, page 75) ;

— Relier les espérances d'*erreur conjointe*, de *succès conjoint* et de *désaccord* par une loi de probabilité *trinomiale* (section 4.5.2, page 78) ;

— Généraliser le résultat de Maurer, qui permet d'appliquer le théorème PAC-bayésien à des votants à valeurs réelles, aux *votants jumelés* et à la loi *trinomiale* (lemmes 4.16 et 4.17, page 79).

Enfin, ce théorème 4.19 nous a permis de calculer une deuxième borne sur le *risque de Bayes* $R_D(B_Q)$ à l'aide de la $\mathcal{C}$*-borne* (borne 4.21, page 86). À la section 4.6, nous avons montré empiriquement l'amélioration qu'apporte cette garantie de généralisation sur les votes de majorité construits par l'algorithme *AdaBoost*, bien qu'elle se révèle encore imparfaite pour la sélection de modèle.

Troisième partie

Au-delà du cadre d'apprentissage inductif

Chapitre 5

Théorie PAC-bayésienne pour l'apprentissage transductif

Le contenu de ce chapitre a fait l'objet d'un article publié dans le cadre de la conférence *AISTATS* (Bégin et al., 2014).

Résumé. Dans le cadre d'apprentissage transductif (décrit brièvement à la section 1.3.2, page 7), on fait l'hypothèse que le choix des exemples étiquetés parmi l'échantillon complet est effectué au *hasard uniforme sans remise*. Nous énonçons un nouveau théorème PAC-bayésien basé sur cette hypothèse, en remplaçant la loi de probabilité binomiale – intervenant dans théorie inductive – par la loi hypergéométrique. Le théorème PAC-bayésien transductif ainsi obtenu se démarque de son pendant inductif par le fait que toute Δ-fonction mène à une borne facilement calculable sans aucune estimation. Nous présentons aussi une nouvelle borne obtenue grâce à une Δ-fonction conçue en prenant en considération la spécificité du cadre transductif. Ce nouveau résultat permet de formuler des garanties de généralisation beaucoup plus précises que la borne PAC-bayésienne pour apprentissage transductif de Derbeko et al. (2004). Enfin, nous concluons ce chapitre par une étude empirique du comportement de diverses bornes transductives dérivées de notre théorème général.

5.1 Description du cadre d'apprentissage transductif

Dans le cadre d'apprentissage inductif, étudié au chapitre 4, nous avons fait l'hypothèse que les exemples d'apprentissage sont générés de manière *i.i.d.* par une distribution D. Cette hypothèse n'est pas toujours réaliste. Dans le cadre d'apprentissage transductif (introduit par Vapnik, 1998), nous ne faisons aucune hypothèse sur la manière dont les exemples d'apprentissage sont générés. Nous considérons que les exemples proviennent d'un échantillon de données, noté Z et nommé l'***échantillon complet***, contenant un nombre N (fini) d'exemples :

$$Z \coloneqq \{(x_i, y_i)\}_{i=1}^{N} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\} \in (\mathcal{X} \times \mathcal{Y})^N .$$

L'***échantillon d'entraînement*** $S \coloneqq \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \subset Z$ est obtenu par un tirage aléatoire *sans remise* de m exemples parmi Z (avec $m < N$). Nous notons les exemples restants par $U \coloneqq Z \setminus S$. En plus de l'échantillon d'entraînement S, un algorithme d'apprentissage transductif a typiquement accès à l'***échantillon non étiqueté*** $U_{\mathcal{X}}$, constitué des $N-m$ descriptions des exemples de U :

$$U_{\mathcal{X}} \stackrel{\text{def}}{=} \{x_i \mid (x_i, \cdot) \in U\} \;=\; \{x_{m+1}, x_{m+2}, \ldots, x_N\}\,. \tag{5.1}$$

L'objectif de l'algorithme d'apprentissage transductif est alors de construire un ***classificateur transductif*** de la forme

$$h : Z_{\mathcal{X}} \to \mathcal{Y}\,, \tag{5.2}$$

où l'ensemble $Z_{\mathcal{X}}$ contient descriptions des exemples de l'échantillon complet Z. Nous désirons que le risque du classificateur transductif $h(\cdot)$ sur l'échantillon complet Z soit minimal. Ce risque, noté $R_Z(h)$, est donné par

$$R_Z(h) \;=\; \frac{1}{N}\sum_{i=1}^{N} \mathtt{I}\big[h(x_i) \neq y_i\big]\,.$$

Insistons sur le fait que ce classificateur transductif $h(\cdot)$ sera seulement utilisé pour prédire l'étiquette des exemples contenus dans Z. Ainsi, $h(\cdot)$ est défini sur l'espace $Z_{\mathcal{X}}$ uniquement. Cela contraste avec les classificateurs inductifs, qui sont définis sur tout l'espace d'entrée $\mathcal{X}$.

Remarque sur les hypothèses de travail. Il convient de faire deux précisions pour bien situer nos travaux par rapport à ceux de Vapnik (1998), qui sont souvent cités comme point de départ dans la littérature sur l'apprentissage transductif.

1. Le cadre d'apprentissage que nous adoptons correspond au «cadre transductif de type 1» de Vapnik (1998), caractérisé par l'hypothèse que les exemples (étiquetés) de l'*échantillon d'entraînement* proviennent d'un tirage *sans remise* parmi l'*échantillon complet*. Nous ne supposons pas l'existence d'une distribution génératrice, comme nous l'avons fait pour le cadre inductif au chapitre 4.
   À l'inverse, dans le «cadre transductif de type 2» de Vapnik (1998), les exemples (étiquetés et non étiquetés) proviennent d'observations *i.i.d.* d'une distribution génératrice de données. Ainsi, l'unique caractéristique qui distingue ce second cadre du cadre d'apprentissage inductif est que l'algorithme d'apprentissage «connaît» les descriptions des exemples que le classificateur transductif devra étiqueter.
2. Dans les travaux de Vapnik (1998), le domaine d'un classificateur transductif est l'ensemble $U_{\mathcal{X}}$, tel que défini par l'équation (5.1). Cependant, nos bornes PAC-bayésiennes nécessiteront la connaissance du risque sur l'échantillon d'entraînement $R_S(G_Q)$. Le domaine de nos classificateurs transductifs est donc $Z_{\mathcal{X}}$, comme le spécifie l'équation (5.2).

5.2 Théorème général

Comme nous l'avons vue au début du chapitre 4, dans le cadre d'apprentissage inductif, l'hypothèse que les données sont générées de manière *i.i.d.* nous permet d'exprimer le nombre d'erreurs fait sur m exemples générés par la distribution D par un classificateur binaire $h : \mathcal{X} \to \{-1, 1\}$ comme une variable aléatoire suivant une *loi binomiale*. Voyons maintenant comment adapter le théorème PAC-bayésien au cadre transductif, où nous ne faisons pas l'«hypothèse *i.i.d.*».

5.2.1 Espérance de perte et loi hypergéométrique

Dans le cadre transductif, comme les exemples proviennent d'un tirage *sans remise* parmi l'échantillon complet Z, la probabilité d'effectuer k erreurs parmi m exemples suit une ***loi hypergéométrique*** de m tirages parmi une population de taille $N \coloneqq |Z|$ contenant $N \cdot R_Z(h)$ succès. En notant $[Z]^m$ la distribution de probabilité uniforme sur tous les sous-ensembles de Z de tailles m, et donc $S \sim [Z]^m$ l'expérience qui consiste en un tirage au ***hasard uniforme sans remise*** d'un échantillon $S \subset Z$ de m exemples, on obtient

$$\Pr_{S \sim [Z]^m} \left(R_S(h) = \tfrac{k}{m} \right) \;=\; \frac{\binom{N \cdot R_Z(h)}{k} \binom{N - N \cdot R_Z(h)}{m-k}}{\binom{N}{m}} \,,$$

pour tout $k \in \{\, \max[0, N \cdot R_Z(h) + m - N], \ldots, \min[m, N \cdot R_Z(h)] \,\}$.

Cette constatation est au centre de notre théorème PAC-bayésien pour l'apprentissage transductif (théorème 5.1, plus bas). En effet, sa démonstration suit essentiellement les étapes de la démonstration du théorème 4.4 telle que présentée dans le cadre inductif (page 61).

Précisons cependant que, dans le théorème 5.1, l'ensemble de votants $\mathcal{H}$ est constitué de classificateurs binaires de la forme $h : \mathcal{X} \to \{-1, 1\}$, alors que le théorème 4.4 considère le cas plus général où $\mathcal{H}$ est constitué de votants à valeurs réelles $f : \mathcal{X} \to [-1, 1]$. Cela s'explique par le fait que dans le cas inductif, le lemme A.12 (Maurer, 2004) permet de ramener le cas (plus général) des votants à valeurs réelles au cas des votants binaires (voir la section 4.2.1, page 59). Il n'existe pas, à notre connaissance, de tel résultat s'appliquant au cadre transductif.

5.2.2 Théorème général pour l'apprentissage transductif

Le théorème suivant constitue le fondement de ce chapitre. Il permet de borner le risque de Gibbs sur l'échantillon complet $R_Z(G_Q)$,

$$R_Z(G_Q) \;=\; \mathop{\mathbf{E}}_{h \sim Q} R_Z(h) \;=\; \frac{1}{N} \sum_{i=1}^{N} \mathop{\mathbf{E}}_{h \sim Q} \mathtt{I}\big[h(x_i) \neq y_i\big] \,,$$

à partir du risque de Gibbs empirique $R_S(G_Q)$, où $S \sim [Z]^m$ est un échantillon d'entraînement.

La formulation du théorème 5.1 ci-bas est semblable au théorème inductif (théorème 4.4, page 61), où la «distance» entre le risque empirique et le risque sur la distribution génératrice – ce dernier étant remplacé par le risque sur l'échantillon complet dans le cadre transductif – est exprimée par une Δ-fonction.

Théorème 5.1. *Pour tout échantillon de données Z contenant N exemples, pour tout ensemble $\mathcal{H}$ de classificateurs $\mathcal{X} \to \{-1, 1\}$, pour toute distribution P sur $\mathcal{H}$, pour tout $\delta \in (0, 1]$, et pour toute fonction convexe $\Delta : [0, 1] \times [0, 1] \to \mathbb{R}$, on a, avec probabilité au moins $1 - \delta$ sur le choix S de m exemples parmi Z,*

$$\forall\, Q \text{ sur } \mathcal{H} : \quad \Delta\Big(R_S(G_Q), R_Z(G_Q)\Big) \;\leq\; \frac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln\frac{\mathcal{T}_\Delta(m, N)}{\delta}\right],$$

où

$$\mathcal{T}_\Delta(m, N) \;\stackrel{\text{def}}{=}\; \max_{K=0\ldots N}\left[\sum_{k\in\mathcal{K}_{m,N,K}} \frac{\binom{K}{k}\binom{N-K}{m-k}}{\binom{N}{m}} e^{m\Delta\left(\frac{k}{m}, \frac{K}{N}\right)}\right], \tag{5.3}$$

et $\mathcal{K}_{m,N,K} \;\stackrel{\text{def}}{=}\; \{\, \max[0, K+m-N], \ldots, \min[m, K] \,\}$.

Démonstration. Nous désirons borner supérieurement la quantité $\Delta\big(R_S(G_Q), R_Z(G_Q)\big)$.

Nous appliquons d'abord l'inégalité de Jensen (lemme A.2, en annexe) sur la fonction convexe $\Delta(\cdot, \cdot)$, puis l'inégalité du changement de mesure (lemme 4.3) avec $\phi(f) \coloneqq m \cdot \Delta\big(R_S(h), R_Z(h)\big)$. On obtient, pour tout ensemble $S \subseteq Z$,

$$\begin{aligned}
&\forall\, Q \text{ sur } \mathcal{H} : \\
&\quad m \cdot \Delta\big(R_S(G_Q), R_Z(G_Q)\big) \\
&\qquad = \; m \cdot \Delta\Big(\mathop{\mathbf{E}}_{h\sim Q} R_S(h), \mathop{\mathbf{E}}_{h\sim Q} R_Z(h)\Big) \\
&\qquad \leq \; \mathop{\mathbf{E}}_{h\sim Q} m \cdot \Delta\Big(R_S(h), R_Z(h)\Big) && \langle \textit{ Inégalité de Jensen } \rangle \\
&\qquad \leq \; \mathrm{KL}(Q\|P) + \ln\Bigg(\underbrace{\mathop{\mathbf{E}}_{h\sim P} e^{m\Delta\left(R_S(h), R_Z(h)\right)}}_{X_P(S)}\Bigg). && \langle \textit{ Changement de mesure } \rangle
\end{aligned}$$

Considérons maintenant la variable aléatoire $X_P(S) \coloneqq \mathbf{E}_{h\sim P}\, e^{m\cdot\Delta(\mathbb{E}^{\mathcal{L}}_S(h), \mathbb{E}^{\mathcal{L}}_D(h))}$. Par l'inégalité de Markov (lemme A.1, en annexe), on a $\Pr_{S\sim[Z]^m}\left(X_P(S) \;\leq\; \frac{1}{\delta} \mathop{\mathbf{E}}_{S'\sim[Z]^m} X_P(S')\right) \geq 1 - \delta$.

Ainsi, avec probabilité au moins $1 - \delta$ sur le choix de m exemples pigés *sans remise* parmi Z (noté $S \sim [Z]^m$), on a

$$\forall\, Q \text{ sur } \mathcal{H} : \quad m \cdot \Delta\Big(\mathop{\mathbf{E}}_{h\sim Q} R_S(h), \mathop{\mathbf{E}}_{h\sim Q} R_Z(h)\Big) \;\leq\; \mathrm{KL}(Q\|P) + \ln\frac{1}{\delta} \mathop{\mathbf{E}}_{S'\sim[Z]^m} X_P(S'). \tag{5.4}$$

Comme la distribution *a priori* P est indépendante de S', on peut inverser l'ordre des espérances dans le terme de droite de l'équation (5.4). Ensuite, considérant que le nombre d'erreurs $mR_S(h)$ suit une loi hypergéométrique de m tirage parmi une population de taille N

contenant $NR_Z(h)$ succès, on obtient

$$\begin{aligned}
\mathop{\mathbf{E}}_{S'\sim[Z]^m} X_P(S') &= \mathop{\mathbf{E}}_{S\sim[Z]^m}\mathop{\mathbf{E}}_{h\sim P} e^{m\Delta(R_S(h),R_Z(h))} \\
&= \mathop{\mathbf{E}}_{h\sim P}\mathop{\mathbf{E}}_{S\sim[Z]^m} e^{m\Delta(R_S(h),R_Z(h))} && \langle\ \textit{Inversion des espérances}\ \rangle \\
&= \mathop{\mathbf{E}}_{h\sim P}\sum_{k=0}^{m}\Pr_{S\sim[Z]^m}\left(R_S(h)=\tfrac{k}{m}\right)e^{m\Delta(\frac{k}{m},R_Z(h))} \\
&= \mathop{\mathbf{E}}_{h\sim P}\sum_{k=\max[0,NR_Z(h)-N+m]}^{\min[m,NR_Z(h)]}\frac{\binom{NR_Z(h)}{k}\binom{N-NR_Z(h)}{m-k}}{\binom{N}{m}}e^{m\Delta(\frac{k}{m},R_Z(h))} \\
&\leq \max_{K=0\ldots N}\left[\sum_{k=\max[0,K-N+m]}^{\min[m,K]}\frac{\binom{K}{k}\binom{N-K}{m-k}}{\binom{N}{m}}e^{m\Delta(\frac{k}{m},\frac{K}{N})}\right] \\
&= \mathcal{T}_\Delta(m,N)\,.
\end{aligned}$$

Le résultat est alors obtenu en substituant $\mathbf{E}_{S'\sim[Z]^m} X_P(S')$ par $\mathcal{T}_\Delta(m)$ dans l'équation (5.4), et en divisant par m des deux côtés de l'inégalité. □

La principale différence entre les théorèmes PAC-bayésiens pour le cadre d'apprentissage inductif et transductif (théorèmes 4.4 et 5.1 respectivement) est que le terme $\mathcal{I}_\Delta(m)$ du théorème inductif (dérivé de la loi binomiale) est remplacé par le terme $\mathcal{T}_\Delta(m)$ dans le théorème transductif (dérivé de la loi hypergéométrique). Nous discutons de la différence entre ces deux termes dans la sous-section suivante.

5.2.3 Choix de Δ-fonctions dans le cadre inductif et dans le cadre transductif

Il y a une forte similitude entre l'expression $\mathcal{I}_\Delta(m)$ du théorème 4.4 (cas inductif) et l'expression $\mathcal{T}_\Delta(m,N)$ du théorème 5.1 (cas transductif). Cependant, pour calculer la valeur de $\mathcal{I}_\Delta(m)$, on doit calculer le supremum d'une expression possiblement non convexe selon une variable *continue* $r\in[0,1]$. Cela restreint le choix de Δ-fonctions dans le cas inductif. C'est pourquoi nous avons présenté seulement deux cas particuliers à la section 4.2 : la fonction $\mathrm{kl}(q,p)$ de l'équation (4.6) et la fonction $\Delta_c(q,p)$ de l'équation (4.13).

Dans le cas transductif, l'expression $\mathcal{T}_\Delta(m,N)$ peut être calculée pour toutes les Δ-fonctions – c'est-à-dire pour toutes les fonctions convexes définies sur $[0,1]\times[0,1]$ –, car sa valeur est donnée par le maximum d'une expression selon une variable *discrète* $K\in\{0,1,2,\ldots,N\}$. Cela ouvre la voie à l'utilisation d'une grande variété de Δ-fonctions dans le cadre transductif.

Cependant, la complexité en temps de calcul de l'expression $\mathcal{T}_\Delta(m,N)$ appartient à $O(m\cdot N)$, car ce calcul requiert N sommations comportant au plus m termes chacune. En pratique, cela peut constituer un obstacle computationnel si N est grand. Pour cette raison, il existe des contextes où il est souhaitable de borner $\mathcal{T}_\Delta(m,N)$ par une expression analytique.

5.3 Une Δ-fonction pour le cadre d'apprentissage transductif

Cette section présente une Δ-fonction conçue avec l'objectif d'exprimer des bornes PAC-bayésiennes pour le cadre d'apprentissage transductif dont le terme $\mathcal{T}_\Delta(m,N)$ possède une forme analytique. Nous nous inspirons des propriétés que possède la Δ-fonction $\mathrm{kl}(\cdot,\cdot)$ dans le cadre inductif, afin de retrouver des propriétés similaires dans le cadre transductif.

5.3.1 Une Δ-fonction connue : la divergence Kullback-Leibler entre deux distributions de Bernoulli

Un premier choix naturel de Δ-fonction est la ***divergence Kullback-Leibler*** entre deux distributions de Bernoulli $\mathrm{kl}(\cdot,\cdot)$, sachant qu'elle mène à la borne PAC-bayésienne la plus serrée dans le cas inductif. Rappelons l'expression de cette Δ-fonction :

$$\mathrm{kl}(q,p) \stackrel{\text{def}}{=} q\ln\frac{q}{p} + (1-q)\ln\frac{1-q}{1-p}\,. \tag{5.5}$$

En insérant l'équation (5.5) dans l'équation (5.3), on obtient

$$\begin{aligned}\mathcal{T}_{\mathrm{kl}}(m,N) &= \max_{K=0\ldots N}\left[\sum_{k\in\mathcal{K}_{m,N,K}}\frac{\binom{K}{k}\binom{N-K}{m-k}}{\binom{N}{m}}e^{m\left[\frac{k}{m}\ln\frac{k/m}{K/N}+(1-\frac{k}{m})\ln\frac{1-k/m}{1-K/N}\right]}\right]\\ &= \max_{K=0\ldots N}\left[\sum_{k\in\mathcal{K}_{m,N,K}}\frac{\binom{K}{k}\binom{N-K}{m-k}}{\binom{N}{m}}\left(\frac{k/m}{K/N}\right)^k\left(\frac{1-k/m}{1-K/N}\right)^{m-k}\right].\end{aligned} \tag{5.6}$$

Malheureusement, l'expression de $\mathcal{T}_{\mathrm{kl}}(m,N)$ obtenue ne se simplifie pas aussi facilement que l'expression analogue $\mathcal{I}_{\mathrm{kl}}(m)$ obtenue dans le cadre inductif (voir l'équation (4.8), page 64). Le temps requis pour effectuer le calcul de l'équation (5.6) dépend donc de l'ordre de grandeur de m et de N. Afin d'éviter d'éventuels obstacles liés au temps de calcul en présence de très grands échantillons de données, la section suivante présente une Δ-fonction possédant des propriétés intéressantes lorsqu'elle est utilisée dans le cadre d'apprentissage transductif.

5.3.2 Conception d'une Δ-fonction pour le cadre transductif

Nous désirons «concevoir» une Δ-fonction pour le cadre transductif tel que le terme $\mathcal{T}_\Delta(m,N)$ associé soit estimable facilement par une expression analytique. Pour ce faire, inspirons-nous de l'observation du terme $\mathcal{I}_{\mathrm{kl}}(m)$ propre au cadre inductif. Rappelons que dans le contexte inductif, nous avons

$$\mathcal{I}_{\mathrm{kl}}(m) = \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\binom{m}{k}e^{-mH(\frac{k}{m})}\right] = \sum_{k=0}^{m}\psi(k,m)\,, \tag{5.7}$$

où $H(q) \stackrel{\text{def}}{=} -q\ln q - (1-q)\ln(1-q)$ correspond à l'***entropie*** d'une distribution de Bernoulli de probabilité de succès q, et où nous définissons

$$\psi(a,b) \stackrel{\text{def}}{=} \binom{b}{a}\left(\frac{a}{b}\right)^a\left(1-\frac{a}{b}\right)^{b-a} = \binom{b}{a}e^{-b\,H(\frac{a}{b})}\,.$$

Le lemme A.6 (en annexe) montre que

$$\frac{1}{b+1} \;\leq\; \psi(a,b) \;\leq\; 1\,. \tag{5.8}$$

Ainsi, dans le cadre inductif, l'utilisation de la Δ-fonction $\mathrm{kl}(\cdot,\cdot)$ fait «disparaître» le *supremum* sur la variable $r \in [0,1]$. Cela permet d'exprimer simplement l'équation (5.7) comme une somme de $m+1$ termes $\psi(\cdot,\cdot)$, et d'obtenir directement $\mathcal{I}_{\mathrm{kl}}(m) \leq m+1$.

Nous cherchons une Δ-fonction, que nous désignons $\Delta_?(\cdot,\cdot)$ pour l'instant, possédant des propriétés similaires dans le cadre transductif. Nous désirons pouvoir borner chaque terme de la somme de l'équation (5.3), c'est-à-dire chaque terme de la forme

$$\frac{\binom{K}{k}\binom{N-K}{m-k}}{\binom{N}{m}}e^{m\Delta_?(\frac{k}{m},\frac{K}{N})}\,, \tag{5.9}$$

par une expression constante. Pour ce faire, associons chacun des trois coefficients binomiaux $\binom{b}{a}$ de l'équation (5.9) à une fonction $\psi(a,b)$, que nous savons borner grâce à l'équation (5.8) :

$$\psi(k,K) = \binom{K}{k}e^{-K\cdot H(\frac{k}{K})} \leq 1\,, \tag{5.10}$$

$$\psi(m{-}k,N{-}K) = \binom{N-K}{m-k}e^{-(N-K)\cdot H(\frac{m-k}{N-K})} \leq 1\,, \tag{5.11}$$

$$\psi(m,N) = \binom{N}{m}e^{m\Delta(\frac{k}{m},\frac{K}{N})} \geq \frac{1}{N+1}\,. \tag{5.12}$$

Cherchons maintenant la fonction $\Delta_?(\cdot,\cdot)$ telle que l'équation (5.9) s'exprime seulement une combinaison des termes $\psi(k,K)$, $\psi(m{-}k,N{-}K)$ et $\psi(m,N)$. On a

$$\begin{aligned}
\frac{\binom{K}{k}\binom{N-K}{m-k}}{\binom{N}{m}}e^{m\Delta_?(\frac{k}{m},\frac{K}{N})} &= \frac{\psi(k,K)\,\psi(m{-}k,N{-}K)}{\psi(m,N)} \\
&= \frac{\binom{K}{k}\binom{N-K}{m-k}}{\binom{N}{m}}\;\frac{e^{-K\cdot H(\frac{k}{K})}\,e^{-(N-K)\cdot H(\frac{m-k}{N-K})}}{e^{-N\cdot H(\frac{m}{N})}} \\
&= \frac{\binom{K}{k}\binom{N-K}{m-k}}{\binom{N}{m}}\;e^{N\cdot H(\frac{m}{N})-K\cdot H(\frac{k}{K})-(N-K)\cdot H(\frac{m-k}{N-K})}\,.
\end{aligned}$$

L'égalité ci-dessus est respectée lorsque

$$m\Delta_?(\tfrac{k}{m},\tfrac{K}{N}) = N\cdot H(\tfrac{m}{N}) - K\cdot H(\tfrac{k}{K}) - (N{-}K)\cdot H(\tfrac{m-k}{N-K})\,.$$

Comme, pour un problème d'apprentissage donné, les tailles m et N des échantillons de données sont fixes, nous allons paramétrer la Δ-fonction par une constante $\beta \coloneqq \frac{m}{N} \in (0,1)$. Quelques manipulations algébriques nous permettent d'exprimer cette nouvelle ***Δ-fonction*** $\Delta_\beta(q,p)$ par l'équation suivante :

$$\Delta_\beta(q,p) \stackrel{\text{def}}{=} \frac{H(\beta) - pH(\beta\frac{q}{p}) - (1-p)H(\beta\frac{1-q}{1-p})}{\beta}\,. \tag{5.13}$$

Comme requis, la fonction $\Delta_\beta : [0,1]\times[0,1] \to \mathbb{R}$ est convexe en ses deux arguments (pour un β fixe). Nous notons l'expression $\Delta_{\beta:\frac{m}{N}}(q,p)$ pour spécifier que $\beta \coloneqq \frac{m}{N}$ lorsque nécessaire.

Par construction, en insérant l'équation (5.13) dans l'équation (5.3) avec $\Delta \coloneqq \Delta_{\beta:\frac{m}{N}}$, on obtient l'équation suivante (nous écrivons $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$ au lieu de $\mathcal{T}_{\Delta_{\beta:\frac{m}{N}}}(m,N)$ afin de simplifier la notation).

$$\mathcal{T}_{\beta:\frac{m}{N}}(m,N) \;=\; \max_{K=0\ldots N}\left[\sum_{k\in\mathcal{K}_{m,N,K}} \frac{\psi(k,K)\,\psi(m-k,N-K)}{\psi(m,N)}\right]. \tag{5.14}$$

Nous pouvons maintenant utiliser les inégalités des équations (5.10), (5.11) et (5.12) pour borner individuellement chaque terme de la sommation :

$$\frac{\psi(k,K)\,\psi(m-k,N-K)}{\psi(m,N)} \;\le\; \frac{1\times 1}{\frac{1}{N+1}} \;=\; N+1\,.$$

Finalement, nous bornons supérieurement chaque terme de la somme de l'équation (5.14) par $N+1$. Nous obtenons l'expression analytique suivante :

$$\mathcal{T}_{\beta:\frac{m}{N}}(m,N) \;\le\; \max_{K=0\ldots N}\sum_{k\in\mathcal{K}_{m,N,K}} N+1 \;\le\; \sum_{k=0}^{m} N+1 \;=\; (m+1)(N+1)\,. \tag{5.15}$$

Nous avons donc formulé la Δ-fonction $\Delta_{\beta:\frac{m}{N}}(\cdot,\cdot)$ pour le cadre transductif de telle sorte que le terme $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$ se borne facilement par $(m+1)(N+1)$. Cependant, ce terme augmente avec la taille de l'échantillon complet N. Il s'agit d'un comportement qui n'est pas optimal, comme le montre le théorème 5.2 présenté dans la sous-section ci-dessous.

5.3.3 Une meilleure expression analytique pour borner $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$

Le théorème 5.2 ci-bas présente une borne analytique sur le terme $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$ dont la valeur peut être considérablement inférieure à celle donnée par l'équation (5.15). Notons que certaines parties de la démonstration qui suit sont fournies en annexe (par les lemmes A.7, A.16 et A.17), afin d'alléger la lecture.

Théorème 5.2. *Soit m et N des entiers tels que $20 \le m \le N-20$, alors*

$$\mathcal{T}_{\beta:\frac{m}{N}}(m,N) \;\le\; 3\,\ln(m)\sqrt{m(1-\tfrac{m}{N})}\,. \tag{5.16}$$

Démonstration. Étant donné des valeurs m, N, K fixées, considérons $k^- \coloneqq \max[0, K{+}m{-}N]$, $k^+ \coloneqq \min[m, K]$, $\mathcal{K}^*_{m,N,K} \coloneqq \mathcal{K}_{m,N,K} \setminus \{k^-, k^+\}$, et

$$F(k) \;\coloneqq\; \frac{\psi(k,K)\,\psi(m{-}k, N{-}K)}{\psi(m,N)}\,.$$

On remarque :

$$\mathcal{T}_{\beta:\frac{m}{N}}(m,N) \;=\; \max_{K=0\ldots N}\left[\sum_{k\in\mathcal{K}_{m,N,K}} F(k)\right]. \tag{5.17}$$

Le lemme A.16 (en annexe) montre que

$$F(k^-) + F(k^+) \;\leq\; 2\,e^{\frac{1}{6\times 20}}\sqrt{2\pi\, m(1-\tfrac{m}{N})}\,. \tag{5.18}$$

Aussi, par le lemme A.7 et quelques manipulations arithmétiques, on obtient

$$F(k) \;<\; \frac{\gamma}{\sqrt{2\pi}}\sqrt{m(1-\tfrac{m}{N})\left(\tfrac{1}{k}+\tfrac{1}{K-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{N-K-m+k}\right)} \qquad \forall k\in\mathcal{K}^*_{m,N,K}\,, \tag{5.19}$$

où $\gamma \coloneqq e^{\frac{1}{12}\left[\frac{1}{K}+\frac{1}{N-K}+\frac{1}{m}+\frac{1}{N-m}\right]} \leq e^{\frac{1}{12}\left[2+\frac{2}{20}\right]}$, car $m \geq 20$ et $N{-}m \geq 20$.

Le lemme A.17 montre que

$$\sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{K-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{N-K-m+k}\right)} \;\leq\; 2\,[1+\ln(m)]\,,$$

ce qui, combiné avec l'équation (5.19), donne

$$\sum_{k\in\mathcal{K}^*_{m,N,K}} F(k) \;\leq\; \frac{\gamma}{\sqrt{2\pi}}\sqrt{m(1-\tfrac{m}{N})}\times 2\,[1+\ln(m)]\;, \tag{5.20}$$

Ainsi, par les équations (5.18) et (5.20), on a

$$\sum_{k\in\mathcal{K}_{m,N,K}} F(k) \;\leq\; \sqrt{m(1-\tfrac{m}{N})}\times C(m)\,, \tag{5.21}$$

avec $C(m) \coloneqq 2\,e^{\frac{1}{6\times 20}}\sqrt{2\pi} + \frac{\gamma}{\sqrt{2\pi}}2[1+\ln(m)]$. Pour $m \geq 20$, on vérifie que $C(m) \leq 3\ln(m)$. Comme l'équation (5.21) ne dépend pas de K, en l'insérant dans l'équation (5.17), on obtient le résultat. □

La figure 5.1 compare la valeur de la borne sur le terme $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$, donnée par le théorème 5.2, à sa valeur exacte. La borne diminue lorsque le ratio m/N atteint un certain seuil, ce qui reflète le comportement de la valeur exacte du terme $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$, contrairement à la borne formulée par l'équation (5.15). Bien que la borne puisse sembler élevée, le terme $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$ se trouve au sein d'un logarithme, qui est lui-même divisé par m, dans le théorème PAC-bayésien (théorème 5.1). Ainsi, notre résultat entraîne une très faible dégradation de la borne sur le risque, au profit d'une simplification considérable du calcul à effectuer.

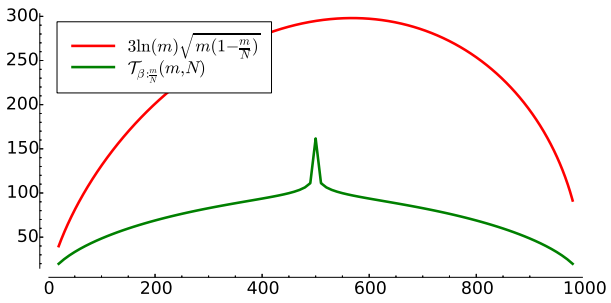


FIGURE 5.1 – Comparaison du terme $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$ et de sa borne supérieure (théorème 5.2), pour un échantillon complet de taille $N \coloneqq 1000$, en fonction de la taille de l'échantillon d'entraînement m.

5.4 Nouvelles bornes PAC-bayésiennes

Dans cette section, nous présentons deux bornes transductives dérivées du théorème transductif général (théorème 5.1). Nous montrons que ces bornes convergent vers les bornes exprimées pour le cadre d'apprentissage inductif lorsque la taille de l'échantillon complet tend vers l'infini, et qu'elles permettent d'obtenir de meilleures garanties sur le risque de Gibbs que la borne formulée par Derbeko et al. (2004). Enfin, nous expliquons que ces bornes transductives sur le risque de Gibbs permettent d'obtenir d'excellentes bornes sur le risque de Bayes en utilisant la $\mathcal{C}$-borne présentée au chapitre 3.

5.4.1 Deux bornes pour le cadre d'apprentissage transductif

La première borne – corollaire 5.3(a), ci-bas – est obtenue exploitant la fonction $\Delta_\beta(\cdot,\cdot)$ présentée par l'équation (5.13), ainsi que la borne supérieure du terme $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$ donnée par le théorème 5.2.

Pour obtenir la deuxième borne – corollaire 5.3(b) –, nous avons aussi besoin de l'observation suivante, qui est démontrée en annexe par le lemme A.15 : l'expression de la Δ-fonction $\Delta_\beta(\cdot,\cdot)$ – définie par l'équation (5.13) – peut être réécrite comme une somme de deux divergences $\mathrm{kl}(\cdot,\cdot)$. En effet,

$$\Delta_\beta(q,p) \;=\; \mathrm{kl}(q,p) + \tfrac{1-\beta}{\beta}\,\mathrm{kl}\Big(\tfrac{p-\beta q}{1-\beta}, p\Big)\,. \tag{5.22}$$

Il est intéressant de constater que la formulation de $\Delta_\beta(q,p)$ exprimée par l'équation (5.22) apparaît dans la démonstration du théorème transductif de Derbeko et al. (2004). Cependant, comme nous verrons à la section 5.4.3, nous obtenons des bornes plus serrées grâce, entre autres, à la formulation de l'équation (5.13).

Corollaire 5.3. *Pour tout échantillon de données Z contenant $N \geq 42$ exemples, pour tout ensemble $\mathcal{H}$ de classificateurs $\mathcal{X} \to \{-1, 1\}$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta \in (0, 1]$, on a, avec probabilité au moins $1 - \delta$ sur le choix S de m exemples parmi Z tel que $20 \leq m \leq N-20$,*

$\forall\, Q$ *sur* $\mathcal{H}$:

(a) $\Delta_{\beta:\frac{m}{N}}\Big(R_S(G_Q), R_Z(G_Q)\Big) \leq \dfrac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln \frac{3\ln(m)\sqrt{m(1-\frac{m}{N})}}{\delta}\right],$

(b) $R_Z(G_Q) \leq R_S(G_Q) + \sqrt{\dfrac{1-\frac{m}{N}}{2m}\left[\mathrm{KL}(Q\|P) + \ln \frac{3\ln(m)\sqrt{m(1-\frac{m}{N})}}{\delta}\right]}.$

Démonstration. Le résultat (a) est obtenu par le théorème 5.1, avec $\Delta(q, p) \coloneqq \Delta_{\beta:\frac{m}{N}}(q, p)$, et en bornant $\mathcal{T}_{\beta:\frac{m}{N}}(m, N)$ à l'aide du théorème 5.2.

Le résultat (b) est obtenu grâce à l'inégalité de Pinsker (équation (4.10), page 64). Plus spécifiquement, on a

$$\begin{aligned}
\Delta_{\beta:\frac{m}{N}}(q, p) &= \mathrm{kl}(q, p) + (\tfrac{N}{m} - 1)\,\mathrm{kl}\Big(\tfrac{p - \frac{m}{N}q}{1-\frac{m}{N}}, p\Big) && \langle\ \textit{Équation (5.22)}\ \rangle \\
&\geq 2(q-p)^2 + 2(\tfrac{N}{m} - 1)\Big(\tfrac{p - \frac{m}{N}q}{1-\frac{m}{N}} - p\Big)^2 && \langle\ \textit{Inégalité de Pinsker}\ \rangle \\
&= \tfrac{2(q-p)^2}{1-\frac{m}{N}}\,,
\end{aligned}$$

puis, en insérant cette inégalité dans l'équation (a), on obtient

$$\frac{2\big(R_S(G_Q) - R_Z(G_Q)\big)^2}{1-\frac{m}{N}} \leq \frac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln \frac{3\ln(m)\sqrt{m(1-\frac{m}{N})}}{\delta}\right],$$

et il suffit d'isoler $R_Z(G_Q)$ pour formuler le résultat (b). □

Les deux bornes pour l'apprentissage transductif présentées par le corollaire 5.3 partagent plusieurs similarités avec les bornes inductives présentées par le corollaire 4.8 (page 68). Comme dans le cadre inductif, la borne du corollaire 5.3(a) – inspirée de la borne inductive de Seeger (2002) – est celle permettant de formuler les meilleures garanties de généralisation (c'est-à-dire les bornes dont les valeurs sont les plus basses). D'un autre côté, la borne du corollaire 5.3(b) – inspirée de la borne inductive de McAllester (2003a) – possède une forme explicite qui facilite son calcul.

5.4.2 Convergence des bornes transductives vers les bornes inductives

Dans le contexte où l'échantillon complet Z est de cardinalité infinie, la probabilité de tirer un exemple (x, y) *sans remise* parmi Z équivaut à la probabilité de tirer ce même exemple *avec remise* parmi Z. Dans ce cas particulier, le cadre transductif se confond avec le cadre inductif, car on peut considérer que (x, y) est générée de manière *i.i.d.* par une distribution de

probabilité uniforme sur l'ensemble Z. Considérant cela, il est intéressant de constater que les bornes transductives du corollaire 5.3 convergent (à un petit facteur près expliqué ci-dessous) vers leur équivalent inductif – soit les bornes du corollaire 4.8 (page 68) – lorsque la taille N de l'échantillon complet devient très grande.

Remarquons d'abord que, lorsque $N \to \infty$, le terme de «complexité» présent du côté de gauche du corollaire 5.3(a),

$$\tfrac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln \tfrac{3\ln(m)\sqrt{m(1-\frac{m}{N})}}{\delta}\right],$$

devient *très près* du terme de «complexité» présent du côté de gauche du corollaire 4.8(a),

$$\tfrac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln \tfrac{2\sqrt{m}}{\delta}\right],$$

au petit facteur près que $\ln[3\ln(m)\sqrt{m(1-m/N)}]$ tend vers $\ln[3\ln(m)\sqrt{m}]$ au lieu de $\ln[2\sqrt{m}]$. Ceci a un effet minime, car ce terme est divisé par m dans l'expression de la borne.

De plus, grâce à l'équation (5.22) – employée avec $\beta \coloneqq \frac{m}{N}$ – on obtient

$$\Delta_{\beta:\frac{m}{N}}(q,p) \;=\; \mathrm{kl}(q,p) + \tfrac{1-m/N}{m/N}\,\mathrm{kl}\Big(\tfrac{p-q\,m/N}{1-m/N}, p\Big).$$

Cette dernière formulation permet de constater que, lorsque $N \to \infty$ et que m est fini, on a $\Delta_{\beta:\frac{m}{N}}(q,p) = \mathrm{kl}(q,p)$. Autrement dit, on retrouve la Δ-fonction $\mathrm{kl}(q,p)$ propre au cadre inductif lorsque l'échantillon complet est de taille infinie. Ainsi, lorsque $N \gg m$, les termes de «distances» entre le risque empirique et l'erreur de généralisation des corollaires 5.3(a) et 4.8(a) se confondent :

$$\Delta_{\beta:\frac{m}{N}}\Big(R_S(G_Q), R_Z(G_Q)\Big) \;\simeq\; \mathrm{kl}\Big(R_S(G_Q), R_Z(G_Q)\Big).$$

On constate don que la borne transductive du corollaire 5.3(a) se révèle une généralisation de la borne inductive de Seeger (2002). Similairement, la borne du corollaire 5.3(b) généralise la borne inductive de McAllester (2003a), car le facteur multiplicatif $\frac{1-\frac{m}{N}}{2m}$ devient $\frac{1}{2m}$ lorsque $N \gg m$.

5.4.3 Comparaison avec des travaux antérieurs

La borne transductive exprimée par le corollaire 5.3(b) peut être analysée comme une amélioration des travaux antérieurs de Derbeko et al. (2004). Notons d'abord que le théorème 18 de Derbeko et al. (2004), auquel nous comparons notre résultat, contient une petite erreur. Dans la dernière partie de la démonstration de Derbeko et al., *pour obtenir une borne explicite*, l'expression suivante est bornée inférieurement (voir l'équation (17) de Derbeko et al., 2004) :

$$\Delta_{\beta:\frac{m}{N}}\Big(R_S(G_Q), R_Z(G_Q)\Big) \;= \\ \mathrm{kl}\Big(R_S(G_Q), R_Z(G_Q)\Big) + (\tfrac{N}{m}-1)\,\mathrm{kl}\left(\tfrac{R_Z(G_Q)-\frac{m}{N}R_S(G_Q)}{1-\frac{m}{N}}\,,\, R_Z(G_Q)\right). \quad (5.23)$$

La borne inférieure de l'équation (5.23) est obtenue en appliquant deux fois, sur chacune des fonctions $\mathrm{kl}(\cdot,\cdot)$, l'inégalité

$$\mathrm{kl}(q,p) \;\geq\; \frac{(q-p)^2}{2p}\,. \tag{5.24}$$

Cependant, l'inégalité de l'équation (5.24) est seulement valide lorsque $q < p$ (voir McAllester, 2003b), et ne peut donc pas être appliquée sur le deuxième terme de l'équation (5.23). En effet, si $R_S(G_Q) \leq R_Z(G_Q)$ et $\frac{m}{N} \in [0,1)$, on a nécessairement

$$\begin{aligned}\frac{R_Z(G_Q) - \frac{m}{N}R_S(G_Q)}{1-\frac{m}{N}} &\geq \frac{R_Z(G_Q) - \frac{m}{N}R_Z(G_Q)}{1-\frac{m}{N}} \\ &= \frac{(1-\frac{m}{N})\,R_Z(G_Q)}{1-\frac{m}{N}} \\ &= R_Z(G_Q)\,.\end{aligned}$$

Cette erreur peut être corrigée en appliquant l'inégalité de Pinsker au lieu de l'inégalité (5.24), comme nous le suggérons dans la démonstration du théorème 5.4 suivant. [1]

Théorème 5.4 (Version corrigée de Derbeko et al., 2004)**.** *Pour tout échantillon de données Z contenant N exemples, pour tout ensemble $\mathcal{H}$ de classificateurs $\mathcal{X} \to \{-1,1\}$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta \in (0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix S de m exemples parmi Z,*

$$\forall\, Q \text{ sur } \mathcal{H}: \quad R_Z(G_Q) \;\leq\; R_S(G_Q) + \sqrt{\frac{1-\frac{m}{N}}{2(m-1)}\Big[\mathrm{KL}(Q\|P) + \ln\tfrac{m}{\delta} + 7\ln(N+1)\Big]}\,.$$

Démonstration. Nous utilisons ici les notations $R_S \coloneqq R_S(G_Q)$ et $R_Z \coloneqq R_Z(G_Q)$. Nous reprenons la démonstration à partir de l'équation (17) de Derbeko et al. (2004) :

$$\mathrm{kl}(R_S,R_Z) + \frac{1-\frac{m}{N}}{\frac{m}{N}}\,\mathrm{kl}\left(\frac{R_Z - \frac{m}{N}R_S}{1-\frac{m}{N}}, R_Z\right) - \frac{7}{m}\log(N+1) \;\leq\; \frac{\mathrm{KL}(Q\|P) + \ln\frac{m}{\delta}}{m-1}\,.$$

En appliquant deux fois l'inégalité de Pinsker (équation (4.10), page 64), on obtient

$$\begin{aligned}\mathrm{kl}(R_S,R_Z) + \frac{1-\frac{m}{N}}{\frac{m}{N}}\,\mathrm{kl}\left(\frac{R_Z - \frac{m}{N}R_S}{1-\frac{m}{N}}, R_Z\right) &\geq 2(R_S-R_Z)^2 + 2\Big(\tfrac{N}{m}-1\Big)\left(\tfrac{R_Z - \frac{m}{N}R_S}{1-\frac{m}{N}} - R_Z\right)^2 \\ &= \tfrac{2(R_S-R_Z)^2}{1-\frac{m}{N}}\,.\end{aligned}$$

Alors, le résultat final est obtenu en isolant R_Z dans l'expression

$$\frac{2(R_S-R_Z)^2}{1-\frac{m}{N}} - \frac{7}{m}\log(N+1) \;\leq\; \frac{\mathrm{KL}(Q\|P) + \ln\frac{m}{\delta}}{m-1}\,.$$

□

1. Dans Derbeko et al. (2004), les bornes transductives sont exprimées sur le risque des exemples non étiquetés $R_U(G_Q)$, alors que nous présentons le théorème 5.4 sous la forme d'une borne sur $R_Z(G_Q)$. Nous faisons ce choix afin d'être cohérents avec les autres résultats présentés dans le présent chapitre. Cela étant dit, il y a une relation directe entre les deux quantités. puisque $U = Z \backslash S$ et $R_Z(h) = \frac{1}{N}\big[mR_S(h) + (N-m)R_U(h)\big]$.

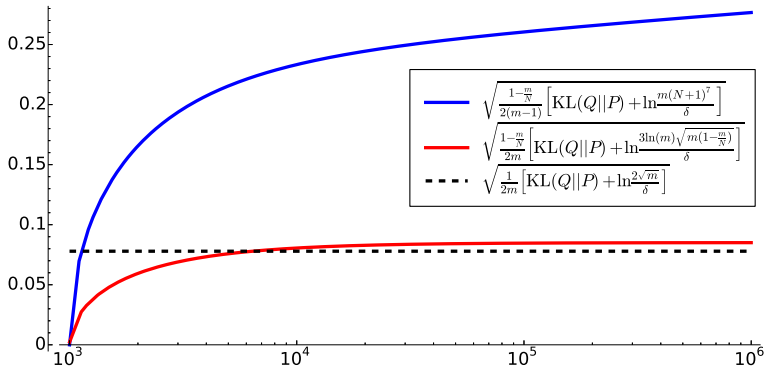


FIGURE 5.2 – Comparaison des termes de complexité des bornes transductives en fonction de la cardinalité de l'échantillon complet. On considère 1000 exemples étiquetés ($m \coloneqq 10^3$), de même que $\mathrm{KL}(Q\|P) \coloneqq 5$ et $\delta \coloneqq 0.05$, et on fait varier le nombre d'exemples de l'échantillon complet sur une échelle logarithmique ($N \in [10^3, 10^6]$). La courbe bleue correspond au terme de complexité de la borne de Derbeko et al. (2004) – théorème 5.4 – tandis que la courbe rouge correspond au terme de complexité de notre borne – corollaire 5.3(b). Aux fins de comparaison, la droite pointillée montre la valeur du terme de complexité de la borne *inductive* de McAllester (2003a) – corollaire 4.8(b) – lorsque $m \coloneqq 10^3$.

La principale différence entre la borne du corollaire 5.3(b) et la borne du théorème 5.4 est que le terme de complexité $\ln(m) + 7\ln(N+1) = \ln\left[m(N+1)^7\right]$, présent dans le résultat de Derbeko et al., est remplacé par le terme $\ln\left[3\ln(m)\sqrt{m(1-\frac{m}{N})}\right]$ grâce au théorème 5.2. Notre résultat permet donc d'exprimer des bornes beaucoup plus serrées, surtout lorsque N est grand. D'ailleurs, contrairement au comportement de la borne du corollaire 5.3(b), la borne de Derbeko et al. diverge lorsque N augmente et que m demeure constant. Ce comportement, illustré par la figure 5.2, n'est pas souhaitable. En effet, comme décrit précédemment (section 5.4.2), le cadre transductif se confond avec le cadre inductif lorsque l'échantillon complet Z contient une infinité d'exemples.

5.4.4 Bornes sur le risque des votes de majorité

Dans ce chapitre, nous avons formulé des garanties sur le risque de Gibbs d'un classificateur transductif. Nous avons démontré, aux chapitres 3 et 4, qu'une borne sur le risque de Gibbs peut être directement convertie en borne sur le risque de Bayes à l'aide de la *borne du facteur deux* ou la *$\mathcal{C}$-borne*. Ces deux bornes sont également valides dans le cadre transductif, comme nous l'expliquons dans les paragraphes suivants.

Rappelons que, dans le cadre d'apprentissage inductif, la ***borne du facteur deux*** est obtenue en appliquant l'***inégalité de Markov*** (lemme A.1, en annexe) sur la variable aléatoire M_Q^D, qui correspond à la marge du vote de majorité sur un tirage *i.i.d.* selon la distribution génératrice des données D. L'*inégalité de Markov* étant valide pour toute variable aléatoire, on peut aussi l'appliquer sur la marge sur un tirage sans remise dans l'échantillon complet Z. En no-

tant cette variable aléatoire M_Q^Z, reprenons la démonstration de la proposition 3.6 (page 48). On obtient

$$\begin{aligned} R_Z(B_Q) &= \Pr_{(x,y)\sim[Z]}\Big(M_Q(x,y) \leq 0\Big) \\ &\leq \mathop{\mathbf{E}}_{(x,y)\sim[Z]}\Big(1 - M_Q(x,y)\Big) && \langle \textit{ Inégalité de Markov } \rangle \\ &= 1 - \mu_1(M_Q^Z) \\ &= 2 \cdot R_Z(G_Q)\,. \end{aligned}$$

De manière similaire, dans le cadre d'apprentissage inductif, la ***$\mathcal{C}$-borne*** est obtenue en appliquant l'***inégalité de Cantelli-Tchebychev*** (théorème A.3, en annexe) sur la variable aléatoire M_Q^D. Dans le cadre transductif, nous reprenons la démonstration du théorème 3.7 (page 49) pour obtenir une borne qui dépend du risque de Gibbs et de l'espérance de désaccord sur l'échantillon complet :

$$\begin{aligned} R_Z(B_Q) &= \Pr_{(x,y)\sim[Z]}\Big(M_Q(x,y) \leq 0\Big) \\ &= \Pr_{(x,y)\sim[Z]}\left(-M_Q(x,y) + \mathop{\mathbf{E}}_{(x,y)\sim[Z]} M_Q(x,y) \geq \mathop{\mathbf{E}}_{(x,y)\sim[Z]} M_Q(x,y)\right) \\ &\leq \frac{\mathop{\mathbf{Var}}_{(x,y)\sim[Z]}(M_Q(x,y))}{\mathop{\mathbf{Var}}_{(x,y)\sim[Z]}(M_Q(x,y)) + \left(\mathop{\mathbf{E}}_{(x,y)\sim[Z]} M_Q(x,y)\right)^2} && \langle \textit{ Inégalité de Cantelli-Tchebychev} \rangle \\ &\vdots \\ &= 1 - \frac{\Big(\mu_1(M_Q^Z)\Big)^2}{\mu_2(M_Q^Z)} \\ &= 1 - \frac{\Big(1 - 2 \cdot R_Z(G_Q)\Big)^2}{1 - 2 \cdot d_Q^Z}\,. \end{aligned}$$

Les deux dernières inégalités nous permettent de convertir la borne sur le risque de Gibbs du corollaire 5.3 en une borne sur le risque de Bayes. Notons également qu'il n'est pas requis de formuler une borne sur l'espérance de désaccord d_Q^Z dans le cadre transductif, car nous connaissons sa valeur exacte. En effet, les étiquettes ne sont pas utilisées lors du calcul de d_Q^Z :

$$d_Q^Z \stackrel{\text{def}}{=} \frac{1}{2}\left(1 - \sum_{i=1}^{N}\left[\mathop{\mathbf{E}}_{f\sim Q} h(x_i)\right]^2\right). \tag{5.25}$$

Ces considérations nous permettent d'énoncer les deux bornes du vote de majorité suivantes.

Borne du vote de majorité 5.5. *Pour tout échantillon de données Z contenant $N \geq 42$ exemples, pour tout ensemble $\mathcal{H}$ de classificateurs $\mathcal{X} \to \{-1, 1\}$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta \in (0, 1]$, on a, avec probabilité au moins $1 - \delta$ sur le choix S de m exemples parmi Z tel que $20 \leq m \leq N-20$,*

$\forall\, Q$ *sur* $\mathcal{H}$:

$$\textit{(i)}\ \underbrace{R_Z(B_Q) \ \leq\ 2 \cdot \sup \mathcal{R}^{\delta,\,\beta}_{Q,S}}_{\textbf{borne du facteur deux}}, \quad \textit{et} \quad \textit{(ii)}\ \underbrace{R_Z(B_Q) \ \leq\ 1 - \frac{\left(1 - 2 \cdot \sup \mathcal{R}^{\delta,\,\beta}_{Q,S}\right)^2}{1 - 2 \cdot d^Z_Q}}_{\mathcal{C}\textbf{-borne}}.$$

où

$$\mathcal{R}^{\delta,\,\beta}_{Q,S} \ \stackrel{\text{def}}{=}\ \left\{ r \in [0, \tfrac{1}{2}] \ \middle|\ \Delta_{\beta:\frac{m}{N}}(R_S(G_Q),\, r) \ \leq\ \frac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln \frac{3\ln(m)\sqrt{m(1-\frac{m}{N})}}{\delta}\right] \right\}.$$

Remarquons que la borne 5.5(i) – basée sur la borne du facteur deux – est le pendant transductif de la borne 4.9 dans le cadre inductif (page 69). De même, la borne 5.5(ii) – basée sur la $\mathcal{C}$-borne – est le pendant transductif de la borne 4.13 dans le cadre inductif (page 74).

Insistons sur le fait que, contrairement aux bornes obtenues pour le cadre inductif, l'***espérance de désaccord*** n'a pas besoin d'être bornée dans le cadre transductif, puisque nous connaissons sa valeur exacte sur l'échantillon complet Z. Ceci fait de la $\mathcal{C}$-borne un outil puissant dans le cadre transductif, car la dégradation de la borne, provenant de l'estimation de l'espérance de désaccord, est évitée.

5.5 Étude empirique

Dans cette section, nous présentons deux types d'expérimentations. D'abord, nous explorons l'influence du choix de différentes Δ-fonctions sur la valeur de la borne obtenue par le théorème 5.1. Ensuite, nous comparons les valeurs des différentes bornes transductives calculées sur des échantillons de données réelles.

5.5.1 Étude de différentes Δ-fonctions

Le théorème 5.1 permet de calculer facilement une borne PAC-bayésienne transductive à l'aide de la Δ-fonction de notre choix. Ce choix permet d'obtenir des bornes plus ou moins serrées selon les valeurs des autres paramètres intervenant dans l'expression de la borne : le risque empirique $R_S(G_Q)$, la divergence $\mathrm{KL}(Q\|P)$, la taille de l'échantillon étiqueté m, la taille de l'échantillon complet N, et (de manière moins importante) le paramètre de confiance δ.

Description de l'expérimentation. Nous comparons les comportements de cinq Δ-fonctions. Les trois premières ont déjà été présentées. Il s'agit de :

1. la ***divergence Kullback-Leibler*** $\mathrm{kl}(q,p)$ entre deux distributions de Bernoulli, présentée par l'équation (4.6) ;
2. la ***fonction*** $\Delta_{\beta:\frac{m}{N}}(q,p)$ introduite dans ce chapitre à l'équation (5.13) ;
3. la borne inférieure de $\mathrm{kl}(q,p)$ donnée par l'inégalité de Pinsker – voir l'équation (4.10) – que nous nommons la ***distance quadratique*** : $\Delta_{V2}(q,p) \stackrel{\text{def}}{=} 2\,(q-p)^2\,.$

Rappelons que l'inégalité de Pinsker intervient dans les bornes PAC-bayésiennes inductives inspirées par McAllester (2003a), comme celle présentée par le corollaire 4.8(b).

Nous avons expérimenté plusieurs autres candidats de Δ-fonctions, et nous en présentons ici deux qui se démarquent à la fois par la simplicité de leur expression et leur bon comportement empirique. Il s'agit de :

4. la ***distance de variation*** $\Delta_{V1}(q,p) \stackrel{\text{def}}{=} 2\,|q-p|$;
5. la ***discrimination triangulaire*** $\Delta_{TR}(q,p) \stackrel{\text{def}}{=} \frac{(q-p)^2}{q+p} + \frac{(q-p)^2}{2-q-p}\,.$

Ces deux dernières fonctions s'avèrent des divergences bien connues dans la littérature (voir par exemple Topsøe, 2000).

La figure 5.3 compare les cinq Δ-fonctions (kl, $\Delta_{\beta:\frac{m}{N}}$, Δ_{V2}, Δ_{V1} et Δ_{TR}) étant donné un risque empirique fixé à $R_S(G_Q) := 0.2$, une divergence de $\mathrm{KL}(Q\|P) := 5$ et un paramètre de confiance $\delta := 0.05$, tout en variant la taille de l'échantillon complet N et le ratio d'exemples étiquetés $\frac{m}{N}$. Plus précisément, nous comparons neuf paires de valeurs (N,m), avec $N \in \{200, 500, 500\}$ et $m \in \{\frac{1}{10}N, \frac{1}{2}N, \frac{9}{10}N\}$.

Pour chaque combinaison de paramètres, la borne supérieure du risque de Gibbs sur l'échantillon complet $R_Z(G_Q)$ est calculée en trouvant la valeur r telle que

$$\Delta\big(R_S(G_Q),\, r\big) \;=\; \frac{1}{m}\left[\mathrm{KL}(Q\|P) + \ln\frac{\mathcal{T}_\Delta(m,N)}{\delta}\right] \qquad (\text{avec } R_S(G_Q) \leq r\,)\,, \tag{5.26}$$

où la valeur exacte de l'expression $\mathcal{T}_\Delta(m,N)$ est calculée conformément à l'équation (5.3) pour toutes les Δ-fonctions. Notamment, les valeurs de $\mathcal{T}_{\mathrm{kl}}(m,N)$ et $\mathcal{T}_{\beta:\frac{m}{N}}(m,N)$ sont respectivement données par les équations (5.6) et (5.14).

Remarquons que la valeur de r obtenue par l'équation (5.26) correspond bien à la borne supérieure recherchée, puisque l'expression $\Delta\big(R_S(G_Q),\, r\big)$ est strictement croissante en termes de r sur l'intervalle $r \in [R_S(G_Q), 1]$ pour les cinq Δ-fonctions évaluées. Cela se vérifie en calculant que la dérivée première $\frac{\partial}{\partial r}\Delta\big(R_S(G_Q),\, r\big)$ est positive pour $r \in [R_S(G_Q), 1]$.

Interprétation des résultats. La figure 5.3 montre que, une fois la Δ-fonction choisie, la valeur de la borne obtenue dépend du compromis entre le taux de croissance de l'expression $\Delta\big(R_S(G_Q), r\big)$ et la valeur du terme de droit de l'équation (5.26), qui est influencée par la valeur de $\mathcal{T}_\Delta(m,N)$. Il est surprenant de constater que la divergence $\mathrm{kl}(q,p)$ est souvent

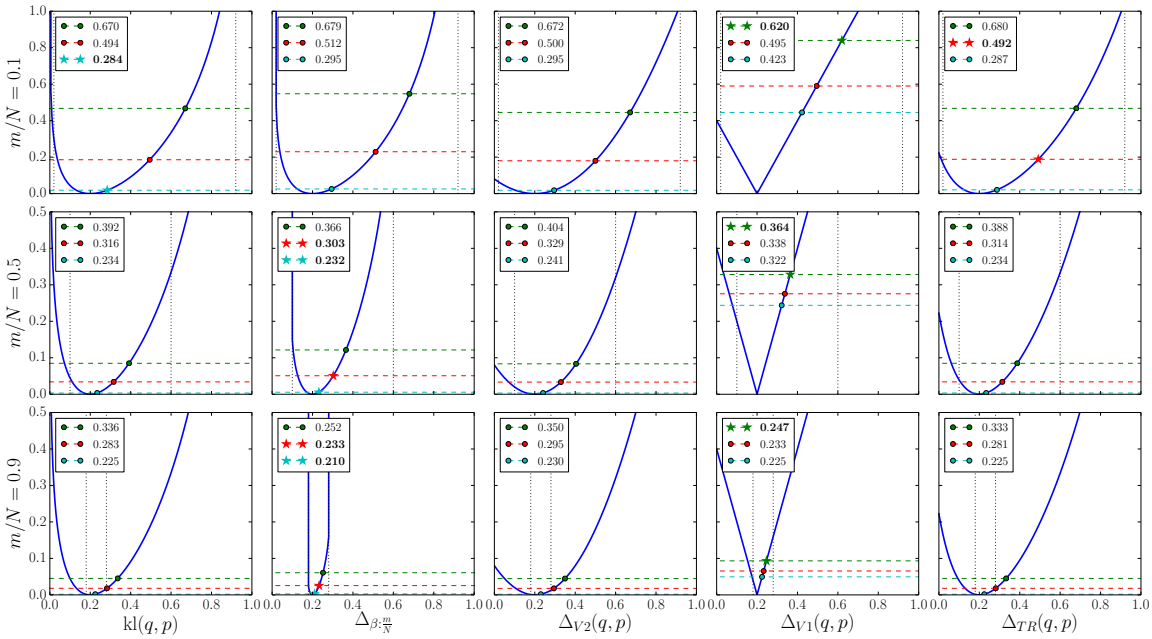


FIGURE 5.3 – Étude du comportement des bornes transductives obtenues par le théorème 5.1. Tous les graphiques considèrent des valeurs fixes de $R_S(G_Q) := 0.2$, $\mathrm{KL}(Q\|P) := 5$ et $\delta := 0.05$. Les trois graphiques de chaque colonne partagent la même Δ-fonction, tandis que les cinq graphiques de chaque ligne partagent un même ratio $\frac{m}{N}$. On interprète un graphique comme suit : Les deux lignes pointillées verticales montrent la valeur minimale et maximale de $R_Z(G_Q)$ – voir l'équation (5.27). La courbe bleue correspond à la fonction $\Delta(0.2, r)$. Chaque ligne horizontale correspond à la valeur donnée par $\frac{1}{m}[\mathrm{KL}(Q\|P)+\ln\frac{\mathcal{T}_\Delta(m,N)}{\delta}]$ pour trois valeurs de N : $N := 200$ (ligne verte), $N := 500$ (ligne rouge), et $N := 5000$ (ligne cyan). Sur chacune de ces trois lignes, la position des marqueurs montre la valeur des bornes – c'est-à-dire la valeur de r résolvant l'équation (5.26). Finalement, la valeur des bornes est indiquée dans la légende du graphique, et un marqueur étoilé indique la borne minimale obtenue pour toutes les Δ-fonctions.

moins précise que les autres Δ-fonctions dans le cadre transductif, alors qu'elle permet d'obtenir les meilleures bornes PAC-bayésiennes connues dans le cadre inductif. Nous remarquons que la divergence $\mathrm{kl}(q, p)$, la distance quadratique $\Delta_{V2}(q, p)$ et la discrimination triangulaire $\Delta_{TR}(q, p)$ performent similairement dans les conditions de notre expérimentation. La distance de variation $\Delta_{V1}(q, p)$ donne les plus petites bornes sur de petits échantillons de données, mais perd son avantage lorsque N devient grand.

La fonction $\Delta_{\beta:\frac{m}{N}}(q, p)$ – introduite à la section 5.3 – performe bien lorsque m est la moitié de N, et permet d'obtenir d'excellentes bornes lorsque m est près de N (c'est-à-dire dans la situation où il y a une faible quantité d'exemples non étiquetés). Ce phénomène s'explique par le fait que la fonction $\Delta_{\beta:\frac{m}{N}}(q, p)$ s'ajuste au ratio $\frac{m}{N}$. Ainsi, la valeur de la borne obtenue

Échantillons de données			Classificateur de Gibbs						Vote de majorité			
			Risques		Versus Derbeko		Théorème 5.1		Risques		Borne 5.5	
Nom	N	m/N	$R_S(G_Q)$	$R_Z(G_Q)$	Co 5.3(b)	Thm 5.4	kl	$\Delta_{\beta;\frac{m}{N}}$	$R_S(B_Q)$	$R_Z(B_Q)$	Facteur 2	$\mathcal{C}$-borne
car	1728	0.1	0.193	0.194	**0.555**	0.793	**0.527**	0.546	0.105	0.159	1.092	-
		0.5	0.179	0.181	**0.418**	0.496	0.418	**0.415**	0.115	0.125	0.830	**0.819**
letter_AB	1555	0.1	0.146	0.149	**0.469**	0.718	**0.437**	0.457	0.000	0.017	**0.914**	0.961
		0.5	0.171	0.171	**0.402**	0.485	0.401	**0.399**	0.000	0.001	0.797	**0.626**
mushroom	8124	0.1	0.202	0.202	**0.486**	0.609	**0.471**	0.482	0.000	0.000	**0.964**	0.966
		0.5	0.205	0.205	**0.439**	0.479	0.438	0.438	0.000	0.000	0.875	**0.546**
nursery	12959	0.1	0.169	0.168	**0.404**	0.504	**0.389**	0.399	0.009	0.016	0.798	**0.692**
		0.5	0.167	0.168	**0.357**	0.391	0.356	0.356	0.010	0.012	0.711	**0.379**
optdigits	3823	0.1	0.208	0.213	**0.533**	0.703	**0.513**	0.527	0.000	0.077	1.055	-
		0.5	0.210	0.211	**0.460**	0.516	0.460	**0.458**	0.026	0.042	0.917	**0.793**
pageblock	5473	0.1	0.199	0.201	**0.495**	0.642	**0.476**	0.490	0.048	0.063	**0.979**	0.992
		0.5	0.208	0.208	**0.448**	0.497	0.448	**0.447**	0.057	0.059	0.894	**0.697**
pendigits	7494	0.1	0.209	0.210	**0.499**	0.629	**0.481**	0.495	0.023	0.051	**0.989**	0.997
		0.5	0.215	0.215	**0.457**	0.500	**0.455**	0.456	0.041	0.045	0.912	**0.706**
segment	2310	0.1	0.206	0.207	**0.558**	0.769	**0.533**	0.550	0.000	0.059	1.101	-
		0.5	0.206	0.206	**0.462**	0.532	0.462	**0.460**	0.014	0.016	0.920	**0.834**
spambase	4601	0.1	0.222	0.227	**0.553**	0.708	**0.535**	0.548	0.115	0.161	1.096	-
		0.5	0.225	0.226	**0.488**	0.539	0.489	**0.486**	0.137	0.143	0.973	**0.961**

TABLE 5.1 – Comparaison des bornes transductives sur des données réelles.

est toujours à l'intérieur de l'intervalle des risques possibles sur l'échantillon complet :

$$\frac{m}{N} R_S(G_Q) \;\leq\; R_Z(G_Q) \;\leq\; \frac{m}{N} R_S(G_Q) + \frac{N-m}{N}\,. \tag{5.27}$$

Dans l'équation (5.27), la borne inférieure est obtenue en supposant que le risque sur les $N-m$ exemples non étiquetés est 0 (le «meilleur cas»), et la borne supérieure est obtenue en supposant que le risque sur les $N-m$ exemples non étiquetés est 1 (le «pire cas»).

Résultats supplémentaires. L'annexe B.3 de ce document présente des graphiques similaires à ceux de la figure 5.3, mais obtenus en considérant des risques plus petits, c'est-à-dire $R_S(G_Q) \coloneqq 0.1$ et $R_S(G_Q) \coloneqq 0.01$.

5.5.2 Comparaisons des bornes sur des données réelles

Nous comparons maintenant les valeurs des bornes transductives sur neuf échantillons de données réelles.

Description de l'expérimentation. Les neuf échantillons de données proviennent du «UCI Machine Learning Repository» (Bache et Lichman, 2013). Il s'agit tous de problèmes de classification binaire. Les N exemples de chacun de ces échantillons de données forment l'échantillon complet Z, et nous sélectionnons au hasard m exemples parmi N (sans remplacement) pour former un échantillon d'entraînement S, en répétant les expérimentations deux fois : une première fois avec un ratio m/N de 0.1 et une seconde fois avec un ratio m/N de 0.5. Lors de chacune de ces expérimentations, un vote de majorité de souches de décisions est obtenu en exécutant l'algorithme d'apprentissage ***AdaBoost*** sur l'échantillon d'entraînement S avec un nombre fixe de 200 itérations. L'algorithme AdaBoost utilisé est décrit à la section 2.4.2, et plus de détails sur le protocole expérimental figurent à l'annexe B.2.

Pour chacune des expérimentations (c'est-à-dire pour chaque échantillon de données et chaque ratio m/N), nous comparons à la fois les bornes sur le risque de Gibbs $R_Z(G_Q)$ et les bornes sur le risque de Bayes $R_Z(B_Q)$. Toutes les bornes sont calculées avec un paramètre de confiance $\delta := 0.05$. Les résultats obtenus sont présentés par le tableau 5.1.

Comparaison avec les résultats antérieurs. La colonne intitulée «Versus Derbeko» compare la borne de Derbeko et al. (2004) – dans la version corrigée présentée par le théorème 5.4 – avec la borne énoncée par le corollaire 5.3(b). Cela confirme numériquement que notre résultat améliore substantiellement celui de Derbeko et al.

Comparaison des bornes sur le risque de Gibbs. La colonne «Théorème 5.1» compare les bornes obtenues par le théorème général – présenté à la section 5.2 – lorsque utilisé avec les Δ-fonctions $\mathrm{kl}(q,p)$ et $\Delta_{\beta:\frac{m}{N}}(q,p)$. On remarque que la première fonction permet d'obtenir de meilleurs résultats dans le régime où l'échantillon d'entraînement est petit (ici, $m/N = 0.1$), tandis que la deuxième fonction – introduite à la section 5.3 – se démarque lorsque la taille de l'échantillon d'entraînement augmente (ici, $m/N = 0.5$). Ces observations vont dans le même sens que celles faites par la figure 5.3.

Il est intéressant de constater que, pour chacune des expérimentations, les valeurs des risques de Gibbs sur l'échantillon d'entraînement $R_S(G_Q)$ et sur l'échantillon complet $R_Z(G_Q)$ sont similaires. De même, pour un même échantillon de données, les valeurs des risques de Gibbs obtenus en exécutant AdaBoost sur 10% et 50% des données sont étonnamment semblables. Cela s'explique par le fait qu'AdaBoost ne minimise pas le risque de Gibbs, mais se concentre plutôt sur la minimisation du risque de Bayes (par le biais de la fonction de perte exponentielle, telle que décrite à la section 2.4.2).

Comparaison des bornes sur le risque de Bayes. La colonne «Borne 5.5» compare les deux bornes sur le risque du vote de majorité (autrement dit, le risque de Bayes), présentées à la section 5.4.4. Notons qu'un tiret dans la dernière colonne indique les cas où la $\mathcal{C}$-borne ne peut pas être calculée, puisque la borne sur le risque de Gibbs est supérieure à 0.5.

On remarque que l'utilisation de la $\mathcal{C}$-borne permet d'améliorer les résultats lorsque la taille de l'échantillon d'entraînement est suffisamment grande (ici, $m/N = 0.5$). Cette amélioration s'avère parfois spectaculaire : sur les échantillons de données «mushroom» et «nursery», on améliore la borne de plus de 30%.

5.6 Synthèse des contributions du chapitre

Dans ce chapitre, nous avons présenté une étude PAC-bayésienne du cadre d'apprentissage transductif. Plus précisément, nous avons porté notre attention sur le «cadre transductif de type 1», tel que présenté par Vapnik (1998). Tout au long du chapitre, nous avons pris soin

de mettre en évidence les similitudes et les particularités du cadre transductif par rapport au cadre inductif étudié dans les chapitres précédents.

Notre première contribution est l'énonciation d'un théorème PAC-Bayésien général pour le cadre transductif (théorème 5.1, page 96). Ce théorème possède les caractéristiques suivantes :

— Similairement au théorème général pour le cadre inductif introduit au chapitre 4, la «distance» entre l'espérance de perte empirique et l'espérance de perte sur l'échantillon complet est exprimée par une fonction convexe $\Delta : [0,1] \times [0,1] \to \mathbb{R}$, que l'on nomme Δ-*fonction* dans la thèse ;

— L'influence du choix de la Δ-fonction, ainsi que l'importance de la *loi hypergéométrique* dans le cadre transductif, est mise en évidence par le terme $\mathcal{T}_\Delta(m, N)$, qui remplace le terme $\mathcal{I}_\Delta(m)$ (ce dernier appartenant au théorème PAC-bayésien dans le cadre inductif). Le terme «transductif» $\mathcal{T}_\Delta(m, N)$ se distingue du terme «inductif» $\mathcal{I}_\Delta(m)$ par le fait qu'il est facilement calculable (c'est-à-dire qu'il ne requiert aucune approximation) pour tout choix de Δ-fonction.

Nous avons aussi présenté une Δ-fonction spécialement conçue pour le cadre d'apprentissage transductif (équation (5.13), page 100), que l'on note Δ_β. En insérant cette dernière dans le théorème transductif général, et en bornant le terme $\mathcal{T}_{\beta:\frac{m}{N}}(m, N)$ par l'expression $3\ln(m)\sqrt{m(1-\frac{m}{N})}$ qui demande moins de temps de calcul (théorème 5.2, page 100), nous avons exprimé deux nouvelles bornes PAC-bayésiennes (corollaire 5.3, page 102) :

— La borne du corollaire 5.3(a) possède une forme implicite et converge vers la borne PAC-bayésienne inductive de Seeger (2002) lorsque la taille N de l'échantillon complet devient très grande (section 5.4.2, page 103) ;

— La borne du corollaire 5.3(b) possède une forme explicite et converge vers la borne PAC-bayésienne inductive de McAllester (2003a) lorsque la taille N de l'échantillon complet devient très grande (section 5.4.2, page 103) ;

— Ces deux bornes possèdent des valeurs beaucoup plus basses que la borne PAC-bayésienne pour l'apprentissage transductif suggérée par Derbeko et al. (2004) (section 5.4.3, page 104).

Ensuite, nous avons montré que le recours à la $\mathcal{C}$-borne, présentée au chapitre 3, permet d'améliorer les bornes sur le risque du vote de majorité – c'est-à-dire les bornes sur le risque de Bayes $R_D(B_Q)$ – en profitant du fait que l'espérance de désaccord d_Q^z est connue dans le cadre transductif (borne 5.5, page 107).

Finalement, nous avons illustré empiriquement l'influence du choix de différentes Δ-fonctions au sein du théorème transductif général (section 5.5.1, page 108), ainsi que les améliorations des valeurs des bornes sur le risque de classificateurs obtenus avec l'algorithme AdaBoost sur des données réelles (section 5.5.2, page 111).

Chapitre 6

Théorie PAC-bayésienne pour l'adaptation de domaine

Une partie du contenu de ce chapitre a fait l'objet d'une publication dans le cadre de la conférence *International Conference on Machine Learning* (Germain et al., 2013). En outre, nous présentons certaines améliorations obtenues depuis la publication initiale, qui ont été partiellement présentées lors du *Workshop on Transfer and Multi-Task Learning* (Germain et al., 2014) et qui font l'objet d'un article soumis pour évaluation à la revue *Journal of Machine Learning Research* (voir Germain et al., 2015a).

Résumé. Les travaux présentés dans ce chapitre constituent la première analyse PAC-bayésienne du cadre en adaptation de domaine (décrit brièvement à la section 1.3.3, page 7). Une notion importante de l'adaptation de domaine est celle de *divergence* entre la distribution source et la distribution cible. Notre première contribution est la définition d'une mesure divergence qui dépend d'une pondération de votants, appropriée à l'analyse PAC-bayésienne. De cette analyse, nous déduisons un algorithme d'apprentissage qui minimise, par descente en gradient, l'expression d'une borne supérieure sur le risque cible. Des expérimentations empiriques sur des données réelles montrent que ce premier algorithme PAC-bayésien pour adaptation de domaine se révèle compétitif avec les autres méthodes suggérées dans la littérature pour ce cadre d'apprentissage.

6.1 Description du cadre de l'adaptation de domaine

Les cadres d'apprentissage inductif et transductif étudiés dans les chapitres précédents considèrent tous deux que les données d'entraînement et les données qu'il faut classifier sont générées par la même distribution de données. En effet, dans le cadre inductif (chapitre 4), on fait l'hypothèse que toutes les données sont générées de manière *i.i.d.* selon la distribution D. De

même, dans le cadre transductif (chapitre 5), il est supposé que les données proviennent d'un tirage sans remise parmi l'échantillon complet Z. Le cadre de l'***adaptation de domaine*** diffère par le fait que l'on considère que les données étiquetées disponibles pour l'entraînement sont générées par une ***distribution source*** D_{S} sur $\mathcal{X} \times \mathcal{Y}$ – représentant le *domaine source* – légèrement différente de la ***distribution cible*** D_{T} sur $\mathcal{X} \times \mathcal{Y}$ – représentant le *domaine cible* – sur laquelle sera utilisé le classificateur.

Plus précisément, nous considérons dans ce chapitre qu'un algorithme d'apprentissage reçoit en entrée un ***échantillon source*** *étiqueté* $S \coloneqq \{(x_1^{\mathrm{S}}, y_1^{\mathrm{S}}), (x_2^{\mathrm{S}}, y_2^{\mathrm{S}}), \ldots, (x_m^{\mathrm{S}}, y_m^{\mathrm{S}})\} \sim (D_{\mathrm{S}})^m$ et un ***échantillon cible*** *non étiqueté* $T \coloneqq \{(x_1^{\mathrm{T}}, \cdot), (x_2^{\mathrm{T}}, \cdot), \ldots, (x_m^{\mathrm{T}}, \cdot)\} \sim (D_{\mathrm{T}})^m$, chacun contenant m exemples. [1] L'objectif de l'algorithme d'apprentissage est de produire un classificateur dont la probabilité de classifier incorrectement un exemple provenant de la distribution cible est faible. Dans le contexte des votes de majorité, étant donné un ensemble de votants $\mathcal{H}$, un tel algorithme doit déterminer la pondération Q sur $\mathcal{H}$ afin de réduire $R_{D_{\mathrm{T}}}(B_Q)$, c'est-à-dire le risque de Bayes sur la distribution cible. Conformément à la définition 3.1, nous avons

$$R_{D_{\mathrm{T}}}(B_Q) \;=\; \mathbb{E}^{\mathcal{L}_{01}}_{D_{\mathrm{T}}}(B_Q) \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D_{\mathrm{T}}} \mathtt{I}\Big[\mathop{\mathbf{E}}_{f\sim Q} y \cdot f(x) \leq 0\Big]\,.$$

La *capacité d'adaptation* d'un algorithme sur un problème d'adaptation de domaine dépend grandement de la similarité entre la distribution source D_{S} et la distribution cible D_{T}. Nous devons donc supposer que D_{S} et D_{T} sont *semblables*, selon un certain critère que l'on se doit de déterminer afin d'étudier le problème d'adaptation de domaine. Ainsi, la section 6.2 présente une première divergence entre domaines, suggérée par Ben-David et al. (2006, 2010), qui permet de quantifier la similarité entre deux distributions de données, et par le biais de laquelle on peut lier le risque sur l'échantillon cible au risque sur l'échantillon source.

6.2 La $\mathcal{H}\Delta\mathcal{H}$-distance : définition et discussion

La majorité des travaux étudiant le problème de l'adaptation de domaine reposent sur le concept de ***divergence entre les distributions*** source et cible. Plusieurs mesures de divergences ont été suggérées par différents auteurs : Ben-David et al. (2006, 2010); Mansour et al. (2009a); Li et Bilmes (2007); Zhang et al. (2012).

6.2.1 Présentation de la $\mathcal{H}\Delta\mathcal{H}$-distance

Nous présentons ici la $\mathcal{H}\Delta\mathcal{H}$-distance (Ben-David et al., 2006, 2010), car il s'agit de la source d'inspiration pour la théorie PAC-bayésienne développée ensuite à la section 6.3. Notons qu'il

1. Pour des fins de simplicité, nous considérons ici que l'échantillon source et l'échantillon cible ont la même cardinalité. Par contre, il est possible de généraliser les résultats afin de considérer des échantillons de cardinalités distinctes.

s'agit d'une pseudo-métrique. [2]

Définition 6.1 (Ben-David et al., 2006, 2010)**.** La ***$\mathcal{H}\Delta\mathcal{H}$-distance*** entre les distributions D_{S} et D_{T} sur l'espace de classificateurs binaires $\mathcal{H}$ est définie par

$$d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}}) \stackrel{\text{def}}{=} 2 \sup_{h,h' \in \mathcal{H}} \left| \mathop{\mathbf{E}}_{(x^{\mathrm{S}},\cdot)\sim D_{\mathrm{S}}} \mathtt{I}[h(x^{\mathrm{S}}) \neq h'(x^{\mathrm{S}})] - \mathop{\mathbf{E}}_{(x^{\mathrm{T}},\cdot)\sim D_{\mathrm{T}}} \mathtt{I}[h(x^{\mathrm{T}}) \neq h'(x^{\mathrm{T}})] \right|.$$

La mesure $d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}})$ ne repose pas sur les étiquettes des exemples sources et cibles, mais seulement sur leur description. Autrement dit, $d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}})$ mesure la «distance» entre les distributions marginales de D_{S} et D_{T} sur l'espace d'entré $\mathcal{X}$. Cette distance est évaluée en regard aux classificateurs de $\mathcal{H}$. La $\mathcal{H}\Delta\mathcal{H}$-distance est faible lorsque, pour tout couple de classificateurs $(h, h') \in \mathcal{H}^2$, la probabilité que $h(\cdot)$ soit en désaccord avec $h'(\cdot)$ sur les exemples de la distribution source est similaire à la probabilité qu'ils soient en désaccord sur la distribution cible.

Notons que Mansour et al. (2009a) généralise la $\mathcal{H}\Delta\mathcal{H}$-distance aux régresseurs (dont la sortie est une valeur réelle). En nous inspirant de ces travaux, nous généralisons $d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}})$ pour le cas où $\mathcal{H}$ est un ensemble de votants $\mathcal{X} \to [-1, 1]$ en utilisant notre concept de ***votant jumelé*** (définition 4.11, page 71) et de ***fonction de perte $\mathcal{L}_d$*** (définition 4.11, page 71) :

$$\begin{aligned} d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}}) &\stackrel{\text{def}}{=} 2 \sup_{f_{ij} \in \mathcal{H}^2} \left| \mathop{\mathbf{E}}_{(x^{\mathrm{S}},\cdot)\sim D_{\mathrm{S}}} \mathcal{L}_d(f_{ij}(x^{\mathrm{S}}), \cdot) - \mathop{\mathbf{E}}_{(x^{\mathrm{T}},\cdot)\sim D_{\mathrm{T}}} \mathcal{L}_d(f_{ij}(x^{\mathrm{T}}), \cdot) \right| \\ &= 2 \sup_{f_{ij} \in \mathcal{H}^2} \left| \mathbb{E}^{\mathcal{L}_d}_{D_{\mathrm{S}}}\Big(f_{ij}\Big) - \mathbb{E}^{\mathcal{L}_d}_{D_{\mathrm{T}}}\Big(f_{ij}\Big) \right|. \end{aligned} \tag{6.1}$$

Par la présence du *supremum* sur $\mathcal{H}^2$, la pseudo-distance $d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}})$ ne dépend que du «pire cas». Nous proposons, à la section 6.3, une divergence entre domaines inspirée de l'équation (6.3) qui s'apparente à une étude du «cas moyen». Mais d'abord, la sous-section qui suit explique comment Ben-David et al. (2006, 2010) fait intervenir a $\mathcal{H}\Delta\mathcal{H}$-distance pour borner le risque sur l'ensemble cible.

6.2.2 Une borne pour l'adaptation de domaine

Après avoir défini la $\mathcal{H}\Delta\mathcal{H}$-distance, Ben-David et al. (2006, 2010) démontre le théorème 6.2 ci-bas, reliant le risque d'un classificateur $h \in \mathcal{H}$ sur la distribution cible $R_{D_{\mathrm{T}}}(h)$ et son risque sur la distribution source $R_{D_{\mathrm{S}}}(h)$. Ce théorème révèle que si la valeur de $d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}})$ est faible, et que l'adaptation est possible (c'est-à-dire qu'il existe au moins un classificateur de l'ensemble $\mathcal{H}$ qui classifie correctement à la fois les exemples sources et les exemples cibles), alors le risque source $R_{D_{\mathrm{S}}}(h)$ est un bon indicateur du risque cible $R_{D_{\mathrm{T}}}(h)$.

2. Contrairement à une *métrique*, deux points ne sont pas nécessairement discernables par la *pseudo-métrique* : on peut avoir $d(x, x') = 0$ pour des valeurs distinctes $x \neq x'$. Toutefois, à l'instar d'une métrique, la pseudo-métrique est symétrique et respecte l'inégalité du triangle. De plus, on a nécessairement $d(x, x) = 0$.

Théorème 6.2 (Ben-David et al., 2006, 2010)**.** *Étant donné un ensemble de classificateurs binaires $\mathcal{H}$ et deux distributions de données D_{S} et D_{T}, on a*

$$\forall\, h \in \mathcal{H} : \quad R_{D_{\mathrm{T}}}(h) \;\leq\; R_{D_{\mathrm{S}}}(h) + \tfrac{1}{2}\, d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}}) + R_{D_{\mathrm{S}}}(h^*) + R_{D_{\mathrm{T}}}(h^*)\,,$$

où $h^*(\cdot)$ est le classificateur qui minimise le **risque conjoint** *sur les distributions D_{S} et D_{T} :*

$$h^* \coloneqq \operatorname*{argmin}_{h\in\mathcal{H}} \Big[R_{D_{\mathrm{S}}}(h) + R_{D_{\mathrm{T}}}(h) \Big]\,. \tag{6.2}$$

La borne sur le risque cible donnée par le théorème 6.2 est donc tributaire du meilleur *risque conjoint* sur la distribution source et la distribution cible, c'est-à-dire $R_{D_{\mathrm{S}}}(h^*) + R_{D_{\mathrm{T}}}(h^*)$. Cette mesure n'est pas estimable dans le contexte où nous ignorons les étiquettes des exemples cibles. Il est généralement admis que cette quantité est faible lorsque, pour un problème donné, l'adaptation est possible. Conséquemment, en supposant que $R_{D_{\mathrm{S}}}(h^*) + R_{D_{\mathrm{T}}}(h^*)$ possède une valeur négligeable, et puisque la mesure $d_{\mathcal{H}\Delta\mathcal{H}}(D_{\mathrm{S}}, D_{\mathrm{T}})$ dépend uniquement de la nature de $\mathcal{H}$, nous parvenons à l'observation suivante :

> Étant donné un ensemble de classificateurs $\mathcal{H}$ fixé, le classificateur $h \in \mathcal{H}$ minimisant la borne sur $R_{D_{\mathrm{T}}}(h)$ donnée par le théorème 6.2 est simplement le classificateur $h(\cdot)$ minimisant le risque source $R_{D_{\mathrm{S}}}(h)$.

Le théorème 6.2 présente une borne sur $R_{D_{\mathrm{T}}}(h)$ qui dépend de quantités qui sont inconnues en pratique (c'est-à-dire les distributions D_{T} et D_{S}), et sa valeur ne peut donc pas être calculée. Les travaux de Ben-David et al. (2010) présentent aussi une borne supérieure de $R_{D_{\mathrm{T}}}(h)$ qui est calculable à partir d'échantillons empiriques $S \sim (D_{\mathrm{S}})^m$ et $T \sim (D_{\mathrm{T}})^m$ (valide avec probabilité $1-\delta$). Cette borne dépend de la *VC-Dimension* de l'espace $\mathcal{H}$ (basée sur la théorie de Vapnik, 1998), c'est-à-dire de la complexité de l'espace de classificateurs.

Dans la section qui suit, nous définissons une nouvelle divergence entre domaines inspirée de la $\mathcal{H}\Delta\mathcal{H}$-distance, mais dont la valeur dépend d'une pondération Q sur $\mathcal{H}$. Nous tirons ensuite profit de la théorie PAC-bayésienne pour développer une borne supérieure sur le risque d'un classificateur de Gibbs sur la distribution de données cible.

Travaux connexes éffectués pendant le doctorat

Parallèlement à l'approche PAC-bayésienne présentée dans ce chapitre, nous avons proposé une autre approche basée sur l'apprentissage d'une représentation commune aux exemples sources et cibles qui minimise *explicitement* l'expression du théorème 6.2 (à quelques détails près). Le fruit de ce travail est présenté brièvement par la section 7.4 et il est décrit en profondeur dans l'article de l'annexe E.

6.3 Une nouvelle divergence entre domaines et borne sur le risque cible

Dans cette section, nous définissons d'abord le concept de *désaccord entre distributions*, basé sur la notion d'espérance de désaccord entre votants énoncée au chapitre 3. Nous montrons ensuite que cette mesure permet de borner le risque de Gibbs sur l'échantillon cible à partir du risque de Gibbs sur l'échantillon source.

6.3.1 Définition du désaccord entre deux distributions

Nous définissons une pseudo-métrique portant sur l'écart entre les espérances de désaccord (tel que présenté à par la définition 3.3, page 45) mesurées sur les domaines source et cible, qui sont respectivement exprimées par

$$d_Q^{D_S} = \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{(x^S,\cdot)\sim D_S}\left[\mathop{\mathbf{E}}_{f\sim Q} f(x^S)\right]^2\right),$$
$$d_Q^{D_T} = \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{(x^T,\cdot)\sim D_T}\left[\mathop{\mathbf{E}}_{f\sim Q} f(x^T)\right]^2\right),$$

où Q est une distribution sur un ensemble de votants $\mathcal{H}$.

La définition de l'*espérance de désaccord* ci-bas est inspirée de l'équation (6.1), mais nous étudions la moyenne des désaccords (selon la distribution Q) plutôt que le *supremum*.

Définition 6.3. Soit deux distributions D_S et D_T sur $\mathcal{X} \times \mathcal{Y}$ et une distribution Q sur un ensemble de votants $\mathcal{H}$. Le ***désaccord entre les distributions*** D_S et D_T selon la pondération Q est défini par

$$\mathrm{dis}_Q(D_S, D_T) \stackrel{\text{def}}{=} \left| d_Q^{D_T} - d_Q^{D_S} \right|.$$

Le désaccord empirique entre deux échantillons de données S et T est donné par

$$\mathrm{dis}_Q(S, T) \stackrel{\text{def}}{=} \left| d_Q^{T} - d_Q^{S} \right|.$$

Comme la $\mathcal{H}\Delta\mathcal{H}$-distance de la définition 6.1, la notion de désaccord entre les distributions $\mathrm{dis}_Q(D_S, D_T)$ ne dépend pas des étiquettes des exemples, mais seulement de leur espace d'entrée. Contrairement à la $\mathcal{H}\Delta\mathcal{H}$-distance, la valeur de $\mathrm{dis}_Q(D_S, D_T)$ dépend d'une pondération Q des votants (et non seulement d'une valeur supérieure). Cela ouvre la porte à une analyse PAC-bayésienne de l'adaptation de domaine.

6.3.2 Une borne du risque de Gibbs sur la distribution cible

Le théorème 6.4 ci-bas est inspiré du théorème 6.2 (Ben-David et al., 2006, 2010). Cependant, alors que ce dernier borne le risque $R_{D_T}(h)$ de chaque votant $h \in \mathcal{H}$ individuellement, le

théorème 6.4 s'applique au risque de Gibbs $R_{D_\mathrm{T}}(G_Q)$, c'est-à-dire à la moyenne de tous les votants de $\mathcal{H}$ selon la pondération Q.

Mentionnons que le théorème 6.4 est une version améliorée du premier théorème sur le risque de Gibbs en adaptation de domaine publié dans Germain et al. (2013). Le recours à l'espérance d'erreur conjointe (définition 4.14, page 75) permet en effet d'exprimer une borne plus serrée et plus facilement interprétable.

Théorème 6.4. *Étant donné un ensemble de votants $\mathcal{H}$ et deux distributions de données D_S et D_T, on a*

$$\forall\, Q \text{ sur } \mathcal{H}: \quad R_{D_\mathrm{T}}(G_Q) \;\leq\; R_{D_\mathrm{S}}(G_Q) + \frac{1}{2}\,\mathrm{dis}_Q(D_\mathrm{S}, D_\mathrm{T}) + \xi_Q(D_\mathrm{S}, D_\mathrm{T})\,, \tag{6.3}$$

où $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ est la différence entre les **espérances d'erreur conjointe** *calculées sur les distributions source et cible :*

$$\xi_Q(D_\mathrm{S}, D_\mathrm{T}) \;\stackrel{\text{def}}{=}\; \left| e_Q^{D_\mathrm{T}} - e_Q^{D_\mathrm{S}} \right|. \tag{6.4}$$

Démonstration. En vertu de l'équation (4.30), pour toute distribution de données D' sur $\mathcal{X} \times \mathcal{Y}$, nous avons

$$R_{D'}(G_Q) \;=\; \frac{1}{2} d_Q^{D'} + e_Q^{D'}\,,$$

Ainsi,

$$\begin{aligned} R_{D_\mathrm{T}}(G_Q) - R_{D_\mathrm{S}}(G_Q) \;&=\; \frac{1}{2}\left(d_Q^{D_\mathrm{T}} - d_Q^{D_\mathrm{S}}\right) + \left(e_Q^{D_\mathrm{T}} - e_Q^{D_\mathrm{S}}\right) \\ &\leq\; \frac{1}{2}\left| d_Q^{D_\mathrm{T}} - d_Q^{D_\mathrm{S}} \right| + \left| e_Q^{D_\mathrm{T}} - e_Q^{D_\mathrm{S}} \right| \\ &=\; \frac{1}{2}\,\mathrm{dis}_Q(D_\mathrm{S}, D_\mathrm{T}) + \xi_Q(D_\mathrm{S}, D_\mathrm{T})\,. \end{aligned}$$

□

Similairement à la borne sur le risque cible de Ben-David et al. (2006, 2010) présentée par le théorème 6.2, la borne du théorème 6.4 dépend principalement du risque sur la distribution source et d'une mesure de «distance» entre les distributions source et cible. Cependant, insistons sur le fait que le théorème de Ben-David et al. (2006, 2010) borne le risque d'*un seul* classificateur $h \in \mathcal{H}$, alors que le nôtre borne le risque de Gibbs, c'est-à-dire le risque moyen de l'ensemble des votants (selon une pondération Q).

6.3.3 Sur le terme d'adaptation ξ_Q

Rappelons que la borne de Ben-David et al. (2006) – théorème 6.2 – dépend du meilleur risque conjoint d'un classificateur $h^* \in \mathcal{H}$, défini par l'équation (6.2). Ce risque conjoint est supposé faible lorsque l'adaptation entre domaines est possible. Dans le cas de la borne du théorème 6.4, cette notion est remplacée par la différence des espérances d'erreur conjointe,

exprimée par $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$. Tout comme le meilleur risque conjoint, la valeur de $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ est une quantité difficilement estimable (étant donné qu'elle repose entre autres sur les étiquettes de la distribution cible) qui est faible lorsque l'adaptation est possible. En particulier, on a $\xi_Q(D_\mathrm{S}, D_\mathrm{T}) = 0$ lorsque $D_\mathrm{S} = D_\mathrm{T}$, c'est-à-dire lorsque les distributions source et cible se confondent.

La proposition suivante présente une borne supérieure sur $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ qui ne repose pas sur l'espérance d'erreur conjointe sur la distribution cible $e_Q^{D_\mathrm{T}}$, mais seulement l'espérance d'erreur conjointe sur la distribution source $e_Q^{D_\mathrm{S}}$ et sur la divergence chi-carrée entre les distributions source et cible $\chi^2(D_\mathrm{T} \| D_\mathrm{S})$.

Proposition 6.5. *Étant donné un ensemble de votants $\mathcal{H}$ et deux distributions de données D_S et D_T qui partagent un même support, alors*

$$\forall\, Q \text{ sur } \mathcal{H} : \quad \xi_Q(D_\mathrm{S}, D_\mathrm{T}) \;\leq\; \sqrt{\chi^2(D_\mathrm{T} \| D_\mathrm{S})\, e_Q^{D_\mathrm{S}}}\,,$$

où $\chi^2(D_\mathrm{T} \| D_\mathrm{S})$ est la **divergence chi-carrée** *entre la distribution source et la distribution cible, qui peut être écrite de plusieurs manières équivalentes :*

$$\chi^2(D_\mathrm{T} \| D_\mathrm{S}) \overset{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \left(\frac{D_\mathrm{T}(x,y)}{D_\mathrm{S}(x,y)} - 1 \right)^2 = \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \left(\frac{D_\mathrm{T}(x,y)}{D_\mathrm{S}(x,y)} \right)^2 - 1 = \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{T}} \frac{D_\mathrm{T}(x,y)}{D_\mathrm{S}(x,y)} - 1\,.$$

Démonstration. Si les distributions D_T et D_S ont le même support, alors nous pouvons borner supérieurement $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ par l'inégalité de Cauchy-Schwarz (lemme A.8, en annexe) :

$$\begin{aligned}
\xi_Q(D_\mathrm{S}, D_\mathrm{T}) \;&=\; \left| e_Q^{D_\mathrm{T}} - e_Q^{D_\mathrm{S}} \right| \\
&= \left| \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathbb{E}_{D_\mathrm{T}}^{\mathcal{L}_e}\big(f_{ij}\big) - \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathbb{E}_{D_\mathrm{S}}^{\mathcal{L}_e}\big(f_{ij}\big) \right| \\
&= \left| \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{T}} \mathcal{L}_e(f_{ij}(x), y) - \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \mathcal{L}_e(f_{ij}(x), y) \right| \\
&= \left| \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \frac{D_\mathrm{T}(x,y)}{D_\mathrm{S}(x,y)} \mathcal{L}_e(f_{ij}(x), y) - \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \mathcal{L}_e(f_{ij}(x), y) \right| \\
&= \left| \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \left(\frac{D_\mathrm{T}(x,y)}{D_\mathrm{S}(x,y)} - 1 \right) \mathcal{L}_e(f_{ij}(x), y) \right| \\
&= \sqrt{\left| \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \left(\frac{D_\mathrm{T}(x,y)}{D_\mathrm{S}(x,y)} - 1 \right) \mathcal{L}_e(f_{ij}(x), y) \right|^2} \\
&\leq \sqrt{\mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \left(\frac{D_\mathrm{T}(x,y)}{D_\mathrm{S}(x,y)} - 1 \right)^2 \times \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \Big(\mathcal{L}_e(f_{ij}(x), y) \Big)^2} \qquad \langle \textit{ Cauchy-Schwarz } \rangle \\
&\leq \sqrt{\mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \left(\frac{D_\mathrm{T}(x,y)}{D_\mathrm{S}(x,y)} - 1 \right)^2 \times \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathop{\mathbf{E}}_{(x,y)\sim D_\mathrm{S}} \mathcal{L}_e(f_{ij}(x), y)}
\end{aligned}$$

$$= \sqrt{\mathop{\mathbf{E}}_{(x,y)\sim D_{\mathrm{S}}}\left(\frac{D_{\mathrm{T}}(x,y)}{D_{\mathrm{S}}(x,y)} - 1\right)^2 \times e_Q^{D_{\mathrm{S}}}} \;=\; \sqrt{\chi^2(D_{\mathrm{T}} \| D_{\mathrm{S}})\, e_Q^{D_{\mathrm{S}}}}\,. \qquad \square$$

La proposition 6.5 met en évidence que le terme non calculable $\xi_Q(D_{\mathrm{S}}, D_{\mathrm{T}})$ du théorème 6.4 est faible lorsque les distributions source D_{S} et cible D_{T} sont semblables. En effet, dans une telle situation, la divergence chi-carrée $\chi^2(D_{\mathrm{T}} \| D_{\mathrm{S}})$ possède une petite valeur.

Il est envisageable, en se basant sur la proposition 6.5, d'estimer la valeur de $\xi_Q(D_{\mathrm{S}}, D_{\mathrm{T}})$ à partir d'échantillons d'entraînement source $S \sim (D_{\mathrm{S}})^m$ et cible $T \sim (D_{\mathrm{T}})^m$. En effet, les méthodes développées à la section 4.4 pour borner l'espérance de désaccord peuvent facilement être utilisées pour borner l'espérance d'erreur conjointe $e_Q^{D_{\mathrm{S}}}$ à partir de sa contrepartie empirique e_Q^S. Toutefois, afin d'estimer le terme $\chi^2(D_{\mathrm{T}} \| D_{\mathrm{S}})$, il est nécessaire de faire une hypothèse sur la distribution des étiquettes des données cibles par rapport aux étiquettes des données sources. Cette approche n'est pas explorée dans cette thèse.

6.4 Une analyse PAC-bayésienne de l'adaptation de domaine

Rappelons que le théorème 6.4, présenté à la section précédente, permet de borner le risque cible $R_{D_{\mathrm{T}}}(G_Q)$ par l'entremise, notamment, du risque source $R_{D_{\mathrm{S}}}(G_Q)$ et du désaccord entre les distributions $\mathrm{dis}_Q(D_{\mathrm{S}}, D_{\mathrm{T}})$. Les quantités $R_{D_{\mathrm{S}}}(G_Q)$ et $\mathrm{dis}_Q(D_{\mathrm{S}}, D_{\mathrm{T}})$ reposent sur les distributions de données D_{S} et D_{T}, qui ne sont pas connues en pratique. Dans cette section, nous utilisons la théorie PAC-bayésienne pour borner le risque cible $R_{D_{\mathrm{T}}}(G_Q)$ à l'aide du risque source empirique $R_S(G_Q)$ et du désaccord empirique $\mathrm{dis}_Q(S, T)$. Ces quantités sont mesurées sur un échantillon source S (contenant des exemples étiquetés) et un échantillon cible T (contenant des exemples non étiquetés) fournis à un algorithme d'apprentissage. On considère que ces deux échantillons sont générés de manière *i.i.d.* par les distributions D_{S} et D_{T} respectivement.

Nous présentons d'abord deux théorèmes PAC-bayésiens permettant d'obtenir une garantie sur la valeur du désaccord entre les distributions source et cible à partir d'échantillons de données (section 6.4.1). Ensuite, nous présentons une borne PAC-bayésienne sur le risque cible (section 6.4.2).

6.4.1 Deux théorèmes PAC-bayésiens pour le désaccord entre les distributions

Les deux théorèmes suivants permettent de borner supérieurement le désaccord entre deux distributions, présenté par la définition 6.3.

Tout d'abord, le théorème 6.6 ci-dessous exploite le fait que la pseudo-métrique $\mathrm{dis}_Q(D_{\mathrm{S}}, D_{\mathrm{T}})$ est exprimée à partir de deux espérances de désaccord (sur la distribution source $d_Q^{D_{\mathrm{S}}}$ et sur

la distribution cible $d_Q^{D_\mathrm{T}}$). Nous utilisons le corollaire 4.12 (page 72), qui permet de borner l'espérance de désaccord d'un vote de majorité, séparément sur $d_Q^{D_\mathrm{S}}$ et $d_Q^{D_\mathrm{T}}$.

Théorème 6.6. *Pour toute paire de distributions D_S et D_T sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X}\to[-1,1]$, pour toute distribution P sur $\mathcal{H}$, et pour tout $\delta\in(0,1]$, avec probabilité au moins $1-\delta$ sur le choix de $S\times T\sim(D_\mathrm{S}\times D_\mathrm{T})^m$, on a*

$$\forall\, Q \text{ sur } \mathcal{H}: \quad \mathrm{dis}_Q(D_\mathrm{S},D_\mathrm{T}) \;\le\; \mathrm{dis}_Q(S,T) + \sqrt{\frac{2}{m}\left[2\,\mathrm{KL}(Q\|P)+\ln\frac{4\sqrt{m}}{\delta}\right]}\,.$$

Démonstration. On désire borner supérieurement

$$\mathrm{dis}_Q(D_\mathrm{S},D_\mathrm{T}) \;=\; \left|\, d_Q^{D_\mathrm{T}} \;-\; d_Q^{D_\mathrm{S}} \,\right| \;=\; \max\left\{d_Q^{D_\mathrm{T}}-d_Q^{D_\mathrm{S}},\, d_Q^{D_\mathrm{S}}-d_Q^{D_\mathrm{T}}\right\}.$$

Pour ce faire, bornons séparément $d_Q^{D_\mathrm{T}}$ et $d_Q^{D_\mathrm{S}}$ à l'aide du corollaire 4.12(b). En utilisant $\delta\coloneqq\delta/2$ pour les deux bornes, la borne de l'union permet d'obtenir le résultat suivant.

Avec probabilité au moins $1-\delta$ sur le choix de $S\times T\sim(D_\mathrm{S}\times D_\mathrm{T})^m$, on a

$\forall\, Q$ sur $\mathcal{H}$:

$$\begin{aligned}
d_Q^T-\sqrt{\tfrac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]} \;&\le\; d_Q^{D_\mathrm{T}} \;\le\; d_Q^T+\sqrt{\tfrac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]},\\
d_Q^S-\sqrt{\tfrac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]} \;&\le\; d_Q^{D_\mathrm{S}} \;\le\; d_Q^S+\sqrt{\tfrac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]}.
\end{aligned}$$

Ainsi,

$$\begin{aligned}
\mathrm{dis}_Q(D_\mathrm{S},D_\mathrm{T}) \;&\le\; \max\left\{d_Q^T-d_Q^S+2\sqrt{\tfrac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]},\right.\\
&\qquad\qquad \left. d_Q^S-d_Q^T+2\sqrt{\tfrac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]}\right\}\\
&=\; \max\left\{d_Q^T-d_Q^S,\, d_Q^S-d_Q^T\right\}+2\sqrt{\tfrac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]}\\
&=\; \mathrm{dis}_Q(S,T)+2\sqrt{\tfrac{1}{2m}\left[2\,\mathrm{KL}(Q\|P)+\ln\tfrac{2\sqrt{m}}{\delta/2}\right]},
\end{aligned}$$

et le résultat désiré est obtenu par quelques manipulations arithmétiques élémentaires. □

Le théorème 6.7 ci-dessous présente un deuxième résultat permettant de borner le désaccord entre distributions. Ce dernier nécessite une démonstration un peu plus longue que celle du précédent théorème 6.6. Au lieu de borner séparément les espérances de désaccord sur la distribution source et la distribution cible, la démonstration du théorème 6.7 fait appel à une fonction de perte qui prend en compte simultanément les deux distributions. Ainsi, nous devons définir un nouveau concept de ***votants jumelés***, similaire à celui de la définition 4.10

(page 71), mais qui opère sur une paire d'exemples source-cible $(x^{\mathrm{S}}, x^{\mathrm{T}})$ provenant de la distribution jointe $D_{\mathrm{S}} \times D_{\mathrm{T}}$.

Théorème 6.7. *Pour toute paire de distributions D_{S} et D_{T} sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to [-1,1]$, pour toute distribution P sur $\mathcal{H}$, pour tout nombre réel $a > 0$ et pour tout $\delta \in (0,1]$, avec probabilité au moins $1-\delta$ sur le choix de $S \times T \sim (D_{\mathrm{S}} \times D_{\mathrm{T}})^m$, on a*

$$\forall\, Q \text{ sur } \mathcal{H}: \quad \mathrm{dis}_Q(D_{\mathrm{S}}, D_{\mathrm{T}}) \;\leq\; \frac{2a}{1-e^{-2a}}\left[\mathrm{dis}_Q(S,T) + \frac{2\,\mathrm{KL}(Q\|P) + \ln\frac{2}{\delta}}{m \times a} + 1\right] - 1\,.$$

Démonstration. Dans cette démonstration, on considère la distribution de données jointe $D_{\mathrm{S}} \times D_{\mathrm{T}}$. On définit un ensemble de votants jumelés $\mathcal{H}^2 \stackrel{\text{def}}{=} \{\widehat{f_{ij}} \mid f_i, f_j \in \mathcal{H}\}$, de sorte que chaque votant jumelé $\widehat{f_{ij}} : \mathcal{X}^2 \to [-1,1]^4$ est composé de deux votants $f_i, f_j : \mathcal{X} \to [-1,1]$ et, étant donné une paire d'exemples source-cible $(x^{\mathrm{S}}, x^{\mathrm{T}}) \sim D_{\mathrm{S}} \times D_{\mathrm{T}}$, retourne un tuple de quatre éléments :

$$\widehat{f_{ij}}(x^{\mathrm{S}}, x^{\mathrm{T}}) \;\stackrel{\text{def}}{=}\; \langle\, f_i(x^{\mathrm{S}}), f_j(x^{\mathrm{S}}), f_i(x^{\mathrm{T}}), f_j(x^{\mathrm{T}}) \,\rangle\,. \tag{6.5}$$

Nous considérons aussi des *prior* et *posterior* de la forme $P^2(\widehat{f_{ij}}) \;\stackrel{\text{def}}{=}\; P(f_i) \cdot P(f_j)$ et $Q^2(\widehat{f_{ij}}) \;\stackrel{\text{def}}{=}\; Q(f_i) \cdot Q(f_j)$.

On désire borner supérieurement

$$\mathrm{dis}_Q(D_{\mathrm{S}}, D_{\mathrm{T}}) \;=\; \left|\, d_Q^{D_{\mathrm{T}}} \,-\, d_Q^{D_{\mathrm{S}}} \,\right| \;=\; \max\{d_1,\, d_2\}\,,$$

où $d_1 \stackrel{\text{def}}{=} d_Q^{D_{\mathrm{S}}} - d_Q^{D_{\mathrm{T}}}$ et $d_2 \stackrel{\text{def}}{=} d_Q^{D_{\mathrm{T}}} - d_Q^{D_{\mathrm{S}}}$. Notons respectivement $d_1^* \stackrel{\text{def}}{=} d_Q^S - d_Q^T$ et $d_2^* \stackrel{\text{def}}{=} d_Q^T - d_Q^S$ leur pendant empirique.

Débutons en bornant supérieurement d_1. Pour ce faire, nous définissons une nouvelle fonction de perte pour les votants jumelés définis à la ligne (6.5) :

$$\mathcal{L}_1(\widehat{f_{ij}}(x^{\mathrm{S}}, x^{\mathrm{T}}), y) \;\stackrel{\text{def}}{=}\; \frac{\mathcal{L}_d(f_{ij}(x^{\mathrm{S}}), y) - \mathcal{L}_d(f_{ij}(x^{\mathrm{T}}), y) + 1}{2}\,,$$

où $\mathcal{L}_d$ est la fonction de perte de la définition 4.11 (notons que la valeur de y n'influence la valeur de perte). La fonction de perte $\mathcal{L}_1$ est conçue de telle sorte que son espérance sur la distribution jointe $D_{\mathrm{S}} \times D_{\mathrm{T}}$ coïncide, à une transformation linéaire près, avec d_1. En effet, on sait par l'équation (4.20) que $d_Q^{D'} = \mathbf{E}_{f_{ij} \sim Q^2} \mathbb{E}_{D'}^{\mathcal{L}_d}\big(f_{ij}\big)$, et donc

$$\mathop{\mathbf{E}}_{\widehat{f_{ij}} \sim Q^2} \mathbb{E}_{D_{\mathrm{S}} \times D_{\mathrm{T}}}^{\mathcal{L}_1}\big(\widehat{f_{ij}}\big) \;=\; \frac{d_Q^{D_{\mathrm{S}}} - d_Q^{D_{\mathrm{T}}} + 1}{2} \;=\; \frac{d_1 + 1}{2}\,.$$

De même, nous obtenons que l'espérance empirique de la perte $\mathcal{L}_1$ sur $S \times T$ est

$$\mathop{\mathbf{E}}_{\widehat{f_{ij}} \sim Q^2} \mathbb{E}_{S \times T}^{\mathcal{L}_1}\big(\widehat{f_{ij}}\big) \;=\; \frac{d_Q^S - d_Q^T + 1}{2} \;=\; \frac{d_1^* + 1}{2}\,.$$

Ainsi, par le corollaire 4.7 (page 66), avec la fonction de perte $\mathcal{L} \coloneqq \mathcal{L}_1$ et en posant $c \coloneqq 2a$ et $\delta \coloneqq \delta/2$, on obtient, avec probabilité au moins $1 - \delta/2$ sur le choix de $S \times T \sim (D_{\mathrm{S}} \times D_{\mathrm{T}})^m$,

$$\forall\, Q \text{ sur } \mathcal{H} : \quad \frac{d_1 + 1}{2} \;\leq\; \frac{1}{1 - e^{-(2a)}} \left[(2a) \cdot \frac{d_1^* + 1}{2} + \frac{2\,\mathrm{KL}(Q\|P) + \ln \frac{1}{\delta/2}}{m} \right],$$

où nous avons appliqué l'égalité $\mathrm{KL}(Q^2\|P^2) = 2\,\mathrm{KL}(Q\|P)$ démontrée à l'équation (4.23). Il suffit d'isoler d_1 pour obtenir la borne supérieure désirée :

$$\forall\, Q \text{ sur } \mathcal{H} : \quad d_1 \;\leq\; \frac{2a}{1 - e^{-2a}} \left[d_1^* + \frac{2\,\mathrm{KL}(Q\|P) + \ln \frac{2}{\delta}}{m} + 1 \right] - 1\,. \tag{6.6}$$

Afin de borner supérieurement d_2 à partir de d_2^*, nous reproduisons la même démarche en utilisant cette fois la fonction de perte suivante :

$$\mathcal{L}_2(\widehat{f_{ij}}(x^{\mathrm{S}}, x^{\mathrm{T}}), y) \;\stackrel{\text{def}}{=}\; \frac{\mathcal{L}_d(f_{ij}(x^{\mathrm{T}}), y) - \mathcal{L}_d(f_{ij}(x^{\mathrm{S}}), y) + 1}{2}\,.$$

Par le corollaire 4.7, avec la fonction de perte $\mathcal{L} \coloneqq \mathcal{L}_2$ et en posant $c \coloneqq 2a$ et $\delta \coloneqq \delta/2$, on obtient, avec probabilité au moins $1 - \delta/2$ sur le choix $S \times T \sim (D_{\mathrm{S}} \times D_{\mathrm{T}})^m$,

$$\forall\, Q \text{ sur } \mathcal{H} : \quad d_2 \;\leq\; \frac{2a}{1 - e^{-2a}} \left[d_2^* + \frac{2\,\mathrm{KL}(Q\|P) + \ln \frac{2}{\delta}}{m} + 1 \right] - 1\,. \tag{6.7}$$

Par la borne de l'union, les inégalités des lignes (6.6) et (6.7) sont respectées simultanément avec probabilité au moins $1 - \delta$. En les combinant, on obtient

$$\begin{aligned}
\forall\, Q \text{ sur } \mathcal{H} : \quad \mathrm{dis}_Q(D_{\mathrm{S}}, D_{\mathrm{T}}) \;&=\; \max\{d_1,\, d_2\} \\
&\leq\; \max\left\{ \tfrac{2a}{1-e^{-2a}} \left[d_1^* + \tfrac{2\,\mathrm{KL}(Q\|P) + \ln \frac{2}{\delta}}{m} + 1 \right] - 1\,, \right. \\
&\qquad\qquad \left. \tfrac{2a}{1-e^{-2a}} \left[d_2^* + \tfrac{2\,\mathrm{KL}(Q\|P) + \ln \frac{2}{\delta}}{m} + 1 \right] - 1 \right\} \\
&=\; \tfrac{2a}{1-e^{-2a}} \left[\max\{d_1^*, d_2^*\} + \tfrac{2\,\mathrm{KL}(Q\|P) + \ln \frac{2}{\delta}}{m} + 1 \right] - 1 \\
&=\; \tfrac{2a}{1-e^{-2a}} \left[\mathrm{dis}_Q(S, T) + \tfrac{2\,\mathrm{KL}(Q\|P) + \ln \frac{2}{\delta}}{m} + 1 \right] - 1\,,
\end{aligned}$$

ce qui conclut la démonstration. □

On remarque que le théorème 6.6 possède une forme similaire au théorème PAC-bayésien de McAllester (1999, 2003a) – voir le corollaire 4.8(b) – tandis que la forme du théorème 6.7 s'apparente à celui de Catoni (2007) – voir le corollaire 4.8(c). Ce dernier possède un paramètre, nommé ici α, qui en fait un outil approprié à la création d'un algorithme d'apprentissage. C'est pourquoi nous travaillons avec le théorème 6.7 pour la suite de ce chapitre.

6.4.2 Une nouvelle borne pour l'adaptation de domaine

Notre objectif est maintenant d'énoncer une borne PAC-bayésienne pour l'adaptation de domaine que nous pouvons minimiser par un algorithme d'apprentissage. Prenons comme point de départ la borne sur le risque cible $R_{D_\mathrm{T}}(G_Q)$, présentée par le théorème 6.4. Cette borne repose sur trois quantités :

1. Le risque $R_{D_\mathrm{S}}(G_Q)$ sur la distribution source.
2. Le désaccord $\mathrm{dis}_Q(D_S, D_T)$ entre les distributions source et cible.
3. La différence entre l'espérance d'erreur conjointe entre les distributions source et cible, notée $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$.

Ces trois quantités sont tributaires des distributions génératrices D_S et D_T, qui nous sont inconnues. Nous désirons estimer, au meilleur de nos capacités, la borne sur le risque cible $R_{D_\mathrm{T}}(G_Q)$ du théorème 6.4 à partir d'un échantillon source $S \coloneqq \{(\mathbf{x}_i^\mathrm{S}, y_i^\mathrm{S})\}_{i=1}^m \sim (D_\mathrm{S})^m$ et d'un échantillon cible $T \coloneqq \{(\mathbf{x}_i^\mathrm{T}, \cdot)\}_{i=1}^m \sim (D_\mathrm{T})^m$. Nous disposons de bornes PAC-bayésiennes permettant d'obtenir des garanties sur les deux premières quantités, $R_{D_\mathrm{S}}(G_Q)$ et $\mathrm{dis}_Q(D_S, D_T)$, à partir des échantillons de données S et T. Cependant, comme discuté précédemment à la sous-section 6.3.3, il est difficile d'estimer la troisième quantité, puisque $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ repose sur les étiquettes de la distribution cible. Ainsi, le théorème 6.8 ci-bas combine une borne sur le risque source $R_{D_\mathrm{S}}(G_Q)$ et le désaccord $\mathrm{dis}_Q(D_S, D_T)$ afin d'obtenir une borne sur le risque cible $R_{D_\mathrm{T}}(G_Q)$ à partir d'observations empiriques $R_S(G_Q)$ et $\mathrm{dis}_Q(S, T)$, tout en considérant le terme $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ comme un inconnu.

Théorème 6.8. *Pour toute paire de distributions D_S et D_T sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to [-1, 1]$, pour toute distribution P sur $\mathcal{H}$, pour tous nombres réels $c > 0$ et $a > 0$, et pour tout $\delta \in (0, 1]$, avec probabilité au moins $1-\delta$ sur le choix de $S \times T \sim (D_\mathrm{S} \times D_\mathrm{T})^m$, on a*

$$\forall\, Q \text{ sur } \mathcal{H} :$$

$$R_{D_\mathrm{T}}(G_Q) \;\le\; c' \cdot R_S(G_Q) + a' \cdot \mathrm{dis}_Q(S, T) + \left(\frac{c'}{c} + \frac{a'}{a}\right) \frac{\mathrm{KL}(Q\|P) + \ln \frac{3}{\delta}}{m} + \xi_Q(D_\mathrm{S}, D_\mathrm{T}) + \frac{a'-1}{2}\,,$$

$$\text{où} \quad a' \coloneqq \frac{2a}{1 - e^{-2a}} \quad \text{et} \quad c' \coloneqq \frac{c}{1 - e^{-c}}\,.$$

Démonstration. Le corollaire 4.8(c), avec $\delta \coloneqq \delta/3$, nous donne

$$R_{D_\mathrm{S}}(G_Q) \;\le\; c' \left[R_S(G_Q) + \frac{\mathrm{KL}(Q\|P) + \ln \frac{3}{\delta}}{m \times c} \right].$$

Le théorème 6.7, avec $\delta \coloneqq 2\delta/3$, nous donne

$$\mathrm{dis}_Q(D_\mathrm{S}, D_\mathrm{T}) \le\; a' \left[\mathrm{dis}_Q(S, T) + \frac{2\,\mathrm{KL}(Q\|P) + \ln \frac{3}{\delta}}{m \times a} + 1 \right] - 1\,.$$

Par la borne de l'union (car $\delta/3 + 2\delta/3 = \delta$), on obtient le résultat désiré en insérant les deux dernières équations dans l'expression du théorème 6.4 et en réorganisant les termes. Notons que utilisons l'inégalité $2\,\mathrm{KL}(Q\|P) + \ln\frac{3}{\delta} < 2\big[\mathrm{KL}(Q\|P) + \ln\frac{3}{\delta}\big]$. □

Rappelons que l'objectif poursuivi dans le présent chapitre est de concevoir un algorithme d'apprentissage reposant sur la borne d'adaptation de domaine. C'est pourquoi avons exprimé un théorème permettant d'ajuster le compromis entre les différentes quantités à l'aide de paramètres. C'est précisément le rôle que tiennent les valeurs de c' et a' dans la conception de l'algorithme présenté dans la prochaine section.

Nous avons donc fait le choix, pour l'énoncé du théorème 6.8, de combiner la borne sur le risque de Gibbs $R_{D_\mathrm{S}}(G_Q)$ du corollaire 4.8(c) et la borne du désaccord entre les distributions $\mathrm{dis}_Q(D_\mathrm{S}, D_\mathrm{T})$ provenant du théorème 6.7. D'autres choix sont possibles : on peut faire appel les corollaires 4.8(a) ou 4.8(b) pour borner $R_{D_\mathrm{S}}(G_Q)$ à partir de $R_S(G_Q)$ et $\mathrm{KL}(Q\|P)$, ainsi que le théorème 6.6 pour borner $\mathrm{dis}_Q(D_\mathrm{S}, D_\mathrm{T})$ à partir de $\mathrm{dis}_Q(S, T)$ et $\mathrm{KL}(Q\|P)$. Chaque combinaison d'une borne sur $R_{D_\mathrm{S}}(G_Q)$ et d'une borne sur $\mathrm{dis}_Q(D_\mathrm{S}, D_\mathrm{T})$ exprime un certain compromis entre le risque empirique sur l'échantillon source $R_S(G_Q)$, le désaccord empirique entre les distributions $\mathrm{dis}_Q(D_\mathrm{S}, D_\mathrm{T})$, la divergence $\mathrm{KL}(Q\|P)$, ainsi que le terme $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ provenant de la borne d'adaptation de domaine (théorème 6.4).

Pour énoncer le théorème 6.8, nous avons aussi fait le choix de ne pas estimer la valeur du terme $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$. Ce faisant, cette valeur demeure inconnue et il est impossible de calculer empiriquement une borne sur le risque cible à l'aide du théorème. Insistons à nouveau sur le fait que l'objectif est ici de concevoir un algorithme d'apprentissage. Puisque cet algorithme sera exécuté à dessein sur des problèmes propices à l'adaptation de domaine, nous supposons que de la valeur de $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ sera négligeable par rapport aux valeurs des termes $R_{D_\mathrm{S}}(G_Q)$ et $\mathrm{dis}_Q(D_S, D_T)$. Cette supposition est supportée par le résultat de la proposition 6.5, qui montre que la valeur de $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ est faible lorsque les distributions sources et cibles sont «semblables» d'après la divergence $\chi^2(D_\mathrm{T}\|D_\mathrm{S})$. En outre, les résultats empiriques présentés à la fin de ce chapitre nous suggèrent que ce choix est judicieux. Notons toutefois qu'il serait pertinent, lors de travaux futurs, d'explorer d'autres stratégies où la valeur de $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ est estimée.

6.5 Algorithme d'apprentissage pour classificateurs linéaires

Dans cette section, nous concevons un algorithme d'apprentissage pour l'adaptation de domaine basé sur la borne du théorème 6.8. Notre algorithme, qui construit un classificateur linéaire, est une extension à l'adaptation de domaine de l'algorithme PBGD présenté dans le mémoire de maîtrise ayant précédé cette thèse (Germain, 2009), ainsi que dans Germain

et al. (2009a). Dans sa version originale, l'algorithme PBGD est dédié au cadre d'apprentissage inductif.

Nous présentons d'abord un résumé de la spécialisation de la théorie PAC-bayésienne aux votes de majorité (section 6.5.1) et de l'algorithme PBGD [3] (section 6.5.2), avant de présenter notre algorithme d'adaptation de domaine (sections 6.5.3 à 6.5.6).

6.5.1 Spécialisation de la théorie PAC-bayésienne aux classificateurs linéaires

Rappelons d'abord que la section 2.2 (page 19) de cette thèse présente le concept de classificateurs linéaires. La frontière de décision d'un ***classificateur linéaire*** $h_{\mathbf{w}}(\cdot)$ s'avère un hyperplan qui sépare l'espace d'entrée des exemples en deux. Nous exprimons la fonction de classification de $h_{\mathbf{w}}(\cdot)$ ainsi :

$$h_{\mathbf{w}}(\mathbf{x}) \stackrel{\text{def}}{=} \operatorname{sgn}\left[\mathbf{w} \cdot \mathbf{x}\right],$$

où $\mathbf{w} \in \mathbb{R}^d$ est un vecteur de poids, orthogonal à la frontière de décision, dans un espace à d dimensions.

Pour simplifier la présentation, nous supposons pour l'instant que l'espace d'entrée des exemples étudiés est $\mathcal{X} \coloneqq \mathbb{R}^d$. Autrement dit, la description d'un exemple est un vecteur de d nombres réels ; chaque exemple a la forme

$$(\mathbf{x}, y) \in \mathbb{R}^d \times \{-1, 1\}.$$

Tout au long de cette section, nous considérons un ensemble de votants formé de *tous les* classificateurs linéaires sur un espace de nombres réels à d dimensions :

$$\mathcal{H} \coloneqq \left\{ h_{\mathbf{v}} \;\middle|\; \mathbf{v} \in \mathbb{R}^d \right\}.$$

En restreignant les distributions *a priori* et *a posteriori* sur $\mathcal{H}$ à des distributions gaussiennes, Langford et Shawe-Taylor (2002); Ambroladze et al. (2006) ont spécialisé la théorie PAC-bayésienne afin de borner l'erreur de généralisation d'un classificateur linéaire. Plus précisément, un classificateur linéaire $h_{\mathbf{w}}(\cdot)$ est représenté par un vote de majorité de l'ensemble continu des classificateurs linéaires $\mathcal{H}$ pondéré par une distribution $Q_{\mathbf{w}}$ paramétrée ainsi :

$$Q_{\mathbf{w}}(h_{\mathbf{v}}) \stackrel{\text{def}}{=} \left(\frac{1}{\sqrt{2\pi}}\right)^d e^{-\frac{1}{2}\|\mathbf{v}-\mathbf{w}\|^2}, \quad \text{pour tout } \mathbf{v} \in \mathbb{R}^d. \tag{6.8}$$

Autrement dit, la distribution $Q_{\mathbf{w}}$ est distribuée selon une gaussienne isotrope de variance unité centrée sur le vecteur $\mathbf{w}$.

3. Pour une description détaillée de l'algorithme PBGD, le lecteur est invité à consulter le chapitre 4 de Germain (2009) ou, pour une description plus succincte (et en anglais), l'article de conférence Germain et al. (2009a).

Une propriété intéressante (voir Langford et Shawe-Taylor, 2002; Langford, 2005; Germain, 2009) de la distribution gaussienne $Q_{\mathbf{w}}$ est que, pour tout exemple $\mathbf{x} \in \mathbb{R}^d$, le classificateur de ***vote de majorité*** (ou ***classificateur de Bayes***) $B_{Q_{\mathbf{w}}}(\mathbf{x})$ – voir l'équation (3.1) – équivaut au classificateur linéaire $h_{\mathbf{w}}(\mathbf{x})$. Autrement dit,

$$\forall \mathbf{x}, \mathbf{w} \in \mathbb{R}^d : \quad B_{Q_{\mathbf{w}}}(\mathbf{x}) \;=\; \operatorname{sgn}\left[\mathop{\mathbf{E}}_{h_{\mathbf{v}} \sim Q_{\mathbf{w}}} h_{\mathbf{v}}(\mathbf{x})\right] \;=\; h_{\mathbf{w}}(\mathbf{x})\,. \tag{6.9}$$

Conséquemment, pour toute distribution D', le risque du classificateur linéaire $R_{D'}(h_{\mathbf{w}})$ est égal au risque de Bayes $R_{D'}(B_{Q_{\mathbf{w}}})$.

Quant au ***risque de Gibbs*** $R_{D'}(G_{Q_{\mathbf{w}}})$, Langford et Shawe-Taylor (2002) ont montré qu'il s'exprime par

$$R_{D'}(G_{Q_{\mathbf{w}}}) \;=\; \mathop{\mathbf{E}}_{(\mathbf{x},y)\sim D'} \mathop{\mathbf{E}}_{h_{\mathbf{v}} \sim Q_{\mathbf{w}}} \mathtt{I}(h_{\mathbf{v}}(\mathbf{x}) \neq y) \;=\; \mathop{\mathbf{E}}_{(\mathbf{x},y)\sim D'} \Phi\left(y \frac{\mathbf{w}\cdot\mathbf{x}}{\|\mathbf{x}\|}\right), \tag{6.10}$$

où

$$\Phi(a) \;\stackrel{\text{def}}{=}\; \frac{1}{2}\left[1 - \operatorname{erf}\left(\frac{a}{\sqrt{2}}\right)\right],$$

et $\operatorname{erf}(\cdot)$ est la *fonction d'erreur de Gauss*, définie par

$$\operatorname{erf}(z) \;\stackrel{\text{def}}{=}\; \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} \mathrm{d}t\,.$$

De même, considérant un *prior* $P_{\mathbf{0}}$ décrit par une gaussienne isotrope de variance unité et centrée à l'origine,

$$P_{\mathbf{0}}(h_{\mathbf{v}}) \;\stackrel{\text{def}}{=}\; \left(\frac{1}{\sqrt{2\pi}}\right)^d e^{-\frac{1}{2}\|\mathbf{v}\|^2}, \quad \text{pour tout } \mathbf{v} \in \mathbb{R}^d\,, \tag{6.11}$$

la ***divergence Kullback-Leibler*** entre $Q_{\mathbf{w}}$ et $P_{\mathbf{0}}$ s'exprime simplement par

$$\mathrm{KL}(Q_{\mathbf{w}} \| P_{\mathbf{0}}) \;=\; \tfrac{1}{2}\|\mathbf{w}\|^2\,. \tag{6.12}$$

Dans le cadre d'apprentissage inductif, nous pouvons utiliser ces résultats pour obtenir une borne sur le risque d'un classificateur linéaire à l'aide d'une des trois versions du corollaire 4.8 (page 68). Nous présentons, par le corollaire 6.9 ci-dessous, la spécialisation du théorème PAC-bayésien de Catoni (2007) aux classificateurs linéaires.

Corollaire 6.9. *Pour toute distribution D sur $\mathbb{R}^d \times \{-1, 1\}$, pour tout nombre réel $c > 0$, et pour tout $\delta \in (0, 1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S \coloneqq \{(\mathbf{x}_i, y_i)\}_{i=1}^m \sim D^m$,*

$$\forall\, \mathbf{w} \in \mathbb{R}^d : \quad R_D(h_{\mathbf{w}}) \;\leq\; \frac{1}{1 - e^{-c}} \cdot \frac{2}{m}\left[c \cdot \sum_{i=1}^m \Phi\left(y_i \frac{\mathbf{w}\cdot\mathbf{x}_i}{\|\mathbf{x}_i\|}\right) + \frac{\|\mathbf{w}\|^2}{2} + \ln \tfrac{1}{\delta}\right].$$

Démonstration. Le résultat est une spécialisation du corollaire 4.8(c) aux classificateurs linéaires, en utilisant le *posterior* centré sur le vecteur $\mathbf{w}$ de l'équation (6.8) et le *prior* centré à l'origine de l'équation (6.11). Nous substituons les valeurs de $R_S(G_{Q_\mathbf{w}})$ et $\mathrm{KL}(Q_\mathbf{w}\|P_\mathbf{0})$ par les expressions tirées des équations (6.10) et (6.12) respectivement. Enfin, nous utilisons l'égalité de l'équation (6.9) et le fait que le risque de Bayes est borné supérieurement par deux fois le risque de Gibbs (proposition 3.6, page 48) : $R_D(h_\mathbf{w}) = R_D(B_{Q_\mathbf{w}}) \leq 2 \cdot R_D(G_{Q_\mathbf{w}})$. □

Le corollaire 6.9 permet de borner le risque d'un classificateur obtenu par n'importe quel algorithme d'apprentissage construisant un classificateur linéaire. Dans cet esprit, Ambroladze et al. (2006); Parrado-Hernández et al. (2012) utilisent la spécialisation de la théorie PAC-bayésienne aux classificateurs linéaires pour obtenir des garanties sur le risque des SVM et effectuer de la sélection de modèle.

De même, puisque la borne PAC-bayésienne est valide uniformément pour tous les classificateurs linéaires (ce que signifie le «$\forall \mathbf{w} \in \mathbb{R}^d$»), le corollaire 6.9 suggère un algorithme d'apprentissage dont l'objectif est de minimiser l'expression de la borne. C'est la stratégie adoptée par l'algorithme PBGD présenté dans la suite du texte.

6.5.2 L'algorithme PBGD

Dans Germain et al. (2009a); Germain (2009), nous avons présenté un algorithme d'apprentissage pour le cadre inductif, nommé ***PBGD*** (de l'anglais *PAC-Bayesian Gradient Descent*), basé sur la minimisation d'une borne PAC-bayésienne spécialisée aux classificateurs linéaires. Les meilleurs résultats sont obtenus en minimisant la borne de Catoni (2007), dans la forme donnée par le corollaire 6.9.

Étant donné un échantillon $S \coloneqq \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ et un hyperparamètre $C > 0$, l'algorithme d'apprentissage PBGD effectue une descente de gradient pour trouver un vecteur de poids optimal $\mathbf{w}$ qui minimise l'expression suivante :

$$C\, m\, R_S(G_{Q_\mathbf{w}}) + \mathrm{KL}(Q_\mathbf{w}\|P_\mathbf{0}) \;=\; C \sum_{i=1}^{m} \Phi\left(y_i \frac{\mathbf{w}\cdot\mathbf{x}_i}{\|\mathbf{x}_i\|}\right) + \frac{\|\mathbf{w}\|^2}{2}\,. \tag{6.13}$$

Le vecteur $\mathbf{w}$ minimisant l'équation (6.13) minimise aussi la borne supérieure $R_D(h_\mathbf{w})$ du corollaire 6.9, pour $c \coloneqq C$, car tous les autres termes de la borne sont constants ; ils peuvent modifier la valeur de la borne, mais ne modifient pas la position du minimum (c'est-à-dire la valeur du vecteur $\mathbf{w}$ optimal).

L'équation (6.13) met en évidence que l'algorithme PBGD cherche un compromis entre le taux d'erreur empirique – exprimé par la ***fonction de perte*** $\Phi(\cdot)$ – et la complexité du vote de majorité appris – mesurée par la ***norme euclidienne*** $\|\mathbf{w}\|^2$. Ce compromis peut être contrôlé en choisissant la valeur de l'hyperparamètre C. Un inconvénient de l'algorithme PBGD, mentionné dans Germain (2009), est que la fonction objectif donnée par l'équation (6.13) n'est pas

convexe. L'implémentation de la descente de gradient nécessite donc de nombreux redémarrages aléatoires d'explorer convenablement l'espace des solutions. Cela dit, nous avons réalisé une étude empirique approfondie qui a permis de constater que PBGD est aussi performant (tout en étant plus rapide) en remplaçant la fonction de perte $\Phi(\cdot)$ par sa relaxation convexe :

$$\begin{aligned} \Phi_{\text{cvx}}(a) &\stackrel{\text{def}}{=} \max\left[\Phi(a), \tfrac{1}{2} - \tfrac{a}{\sqrt{2\pi}}\right] \\ &= \begin{cases} \frac{1}{2} - \frac{a}{\sqrt{2\pi}} & \text{si } a \leq 0 \\ \Phi(a) & \text{sinon.} \end{cases} \end{aligned} \tag{6.14}$$

La fonction objectif minimisée par la version convexifiée de l'algorithme PBGD devient alors

$$C \sum_{i=1}^{m} \underbrace{\Phi_{\text{cvx}}\left(y_i \frac{\mathbf{w} \cdot \mathbf{x}_i}{\|\mathbf{x}_i\|}\right)}_{\textbf{\textit{perte}}\ \Phi_{\text{cvx}}(\cdot)} + \underbrace{\frac{1}{2}\|\mathbf{w}\|^2}_{\textbf{\textit{régularisateur}}\ \ell_2}. \tag{6.15}$$

Il est intéressant de constater que la fonction objectif de l'équation (6.15) est semblable à celle de l'algorithme d'apprentissage SVM, telle que présentée par l'équation (2.7) (page 21). Le régularisateur ℓ_2 (la norme euclidienne $\|\mathbf{w}\|^2$) est exactement le même que pour les algorithmes PBGD et SVM. De même la fonction de perte convexifiée $\Phi_{\text{cvx}}(\cdot)$ possède une forme similaire à la ***perte hinge*** intervenant dans le problème du SVM (cela est mis en évidence par la figure 6.1, page 134). Finalement, un hyperparamètre C permet d'ajuster de compromis entre la minimisation de la perte empirique et la fonction de régularisation dans les deux cas.

De fait, lorsqu'évalué sur des échantillons de données réelles, les performances empiriques de l'algorithme PBGD se comparent avantageusement aux résultats de l'algorithme SVM, comme nous l'avons montré dans Germain et al. (2009a); Germain (2009) par des expérimentations détaillées. Bien que ces dernières concernent la minimisation de la fonction objectif *non convexifiée* de l'équation (6.13), la minimisation de la fonction objectif *convexifiée* – équation (6.15) – a produit d'aussi bons résultats empiriques au cours de nos expérimentations ultérieures.

Encouragés par les succès de l'algorithme PBGD, nous proposons, dans les sous-sections suivantes, de s'inspirer de ce dernier pour la conception d'un algorithme d'adaptation de domaine. Plus précisément, notre approche consiste à minimiser la borne PAC-bayésienne exprimée par le théorème 6.8, que nous spécialisons aux classificateurs linéaires.

6.5.3 Spécialisation de la borne d'adaptation de domaine aux classificateurs linéaires

Supposons que nous possédons un échantillon source (étiqueté) $S \coloneqq \{(\mathbf{x}_i^{\text{S}}, y_i^{\text{S}})\}_{i=1}^m$ et un échantillon cible (non étiqueté) $T \coloneqq \{(\mathbf{x}_i^{\text{T}}, \cdot)\}_{i=1}^m$, contenant chacun m observations générées de manière *i.i.d.* par les distributions D_{S} et D_{T} respectivement. Notre objectif est alors de concevoir un algorithme minimisant la borne découlant du théorème 6.8.

Nous fondons notre algorithme sur la spécialisation de la théorie PAC-bayésienne aux classificateurs linéaires décrite par les sections 6.5.1 et 6.5.2. Ainsi, nous exprimons un classificateur linéaire $h_{\mathbf{w}}(\cdot)$ par le vote de majorité $B_{Q_{\mathbf{w}}}(\cdot)$, où le *posterior* $Q_{\mathbf{w}}$ est une distribution gaussienne isotrope centrée sur le vecteur $\mathbf{w}$. Les équations (6.10) et (6.12) donnent les expressions de $R_S(G_{Q_{\mathbf{w}}})$ et $\mathrm{KL}(Q_{\mathbf{w}} \| P_{\mathbf{0}})$ en fonction du vecteur $\mathbf{w}$.

Attardons-nous d'abord au calcul du terme $\mathrm{dis}_{Q_{\mathbf{w}}}(S,T) = \left| \, d^T_{Q_{\mathbf{w}}} \, - \, d^S_{Q_{\mathbf{w}}} \, \right|$, correspondant au désaccord empirique entre les échantillons S et T (voir la définition 6.3) selon la distribution gaussienne $Q_{\mathbf{w}}$. Puisque les classificateurs linéaires $h_{\mathbf{v}} \in \mathcal{H}$ sont des classificateurs binaires, nous calculons, en nous basant sur l'équation (3.3), l'espérance de désaccord $d^{D'}_{Q_{\mathbf{w}}}$ sur une distribution D' par

$$
\begin{aligned}
d^{D'}_{Q_{\mathbf{w}}} &= \mathop{\mathbf{E}}_{(\mathbf{x},\cdot)\sim D'} \mathop{\mathbf{E}}_{h_{\mathbf{v}}\sim Q_{\mathbf{w}}} \mathop{\mathbf{E}}_{h_{\mathbf{v}'}\sim Q_{\mathbf{w}}} \mathtt{I}[h_{\mathbf{v}}(\mathbf{x}) \neq h_{\mathbf{v}'}(\mathbf{x})] \\
&= \mathop{\mathbf{E}}_{(\mathbf{x},\cdot)\sim D'} \mathop{\mathbf{E}}_{h_{\mathbf{v}}\sim Q_{\mathbf{w}}} \mathop{\mathbf{E}}_{h_{\mathbf{v}'}\sim Q_{\mathbf{w}}} \Big(\mathtt{I}[h_{\mathbf{v}}(\mathbf{x}) = 1]\, \mathtt{I}[h_{\mathbf{v}'}(\mathbf{x}) = -1] + \mathtt{I}[h_{\mathbf{v}}(\mathbf{x}) = -1]\, \mathtt{I}[h_{\mathbf{v}'}(\mathbf{x}) = 1] \Big) \\
&= 2 \mathop{\mathbf{E}}_{(\mathbf{x},\cdot)\sim D'} \mathop{\mathbf{E}}_{h_{\mathbf{v}}\sim Q_{\mathbf{w}}} \mathop{\mathbf{E}}_{h_{\mathbf{v}'}\sim Q_{\mathbf{w}}} \mathtt{I}[h_{\mathbf{v}}(\mathbf{x}) = 1]\, \mathtt{I}[h_{\mathbf{v}'}(\mathbf{x}) = -1] \\
&= 2 \mathop{\mathbf{E}}_{(\mathbf{x},\cdot)\sim D'} \mathop{\mathbf{E}}_{h_{\mathbf{v}}\sim Q_{\mathbf{w}}} \mathtt{I}[h_{\mathbf{v}}(\mathbf{x}) = 1] \mathop{\mathbf{E}}_{h_{\mathbf{v}'}\sim Q_{\mathbf{w}}} \mathtt{I}[h_{\mathbf{v}'}(\mathbf{x}) = -1] \\
&= 2 \mathop{\mathbf{E}}_{(\mathbf{x},\cdot)\sim D'} \Phi\left(\tfrac{\mathbf{w}\cdot\mathbf{x}}{\|\mathbf{x}\|}\right) \Phi\left(-\tfrac{\mathbf{w}\cdot\mathbf{x}}{\|\mathbf{x}\|}\right).
\end{aligned}
$$

La dernière égalité découle de l'équation (6.10). Ainsi,

$$
\begin{aligned}
\mathrm{dis}_{Q_{\mathbf{w}}}(S,T) &= \left| \, d^T_{Q_{\mathbf{w}}} \, - \, d^S_{Q_{\mathbf{w}}} \, \right| \\
&= \left| \frac{1}{m} \sum_{i=1}^{m} \Phi_{\mathrm{dis}}\left(\frac{\mathbf{w}\cdot\mathbf{x}^{\mathrm{S}}_i}{\|\mathbf{x}^{\mathrm{S}}_i\|}\right) - \frac{1}{m} \sum_{i=1}^{m} \Phi_{\mathrm{dis}}\left(\frac{\mathbf{w}\cdot\mathbf{x}^{\mathrm{T}}_i}{\|\mathbf{x}^{\mathrm{T}}_i\|}\right) \right|, \qquad (6.16)
\end{aligned}
$$

où

$$
\Phi_{\mathrm{dis}}(a) \stackrel{\mathrm{def}}{=} 2\,\Phi(a)\,\Phi(-a)\,.
$$

Nous possédons maintenant toutes les connaissances nécessaires pour énoncer la spécialisation du théorème 6.8 aux classificateurs linéaires.

Corollaire 6.10. *Pour toute paire de distributions D_{S} et D_{T} sur $\mathbb{R}^d \times \{-1,1\}$, pour tous nombres réels $c > 0$ et $a > 0$, et pour tout $\delta \in (0,1]$, avec probabilité au moins $1-\delta$ sur le choix de $S \times T \sim (D_{\mathrm{S}} \times D_{\mathrm{T}})^m$ — en écrivant $S \coloneqq \{(\mathbf{x}^{\mathrm{S}}_i, y^{\mathrm{S}}_i)\}_{i=1}^m$ et $T \coloneqq \{(\mathbf{x}^{\mathrm{T}}_i, \cdot)\}_{i=1}^m$ — on a*

$\forall\, \mathbf{w} \in \mathbb{R}^d$:

$$
R_D(h_{\mathbf{w}}) \;\leq\; \frac{2}{m}\left[c' \cdot \sum_{i=1}^{m} \Phi\left(y^{\mathrm{S}}_i \tfrac{\mathbf{w}\cdot\mathbf{x}^{\mathrm{S}}_i}{\|\mathbf{x}^{\mathrm{S}}_i\|} \right) + a' \cdot \left| \sum_{i=1}^{m} \Phi_{\mathrm{dis}}\Big(\tfrac{\mathbf{w}\cdot\mathbf{x}^{\mathrm{S}}_i}{\|\mathbf{x}^{\mathrm{S}}_i\|}\Big) - \Phi_{\mathrm{dis}}\Big(\tfrac{\mathbf{w}\cdot\mathbf{x}^{\mathrm{T}}_i}{\|\mathbf{x}^{\mathrm{T}}_i\|}\Big) \right| + \Big(\tfrac{c'}{c} + \tfrac{a'}{a}\Big)\Big(\tfrac{\|\mathbf{w}\|^2}{2} + \ln\tfrac{3}{\delta}\Big) \right]
$$
$$
+\, 2\,\xi_{Q_{\mathbf{w}}}(D_{\mathrm{S}}, D_{\mathrm{T}}) + a' - 1\,,
$$

où $a' \coloneqq \frac{2a}{1-e^{-2a}}$, *et* $c' \coloneqq \frac{c}{1-e^{-c}}$.

Démonstration. Le résultat est une spécialisation du théorème 6.8 aux classificateurs linéaires, et s'inspire fortement de la démonstration du corollaire 6.9. Il s'agit donc d'une conséquence directe des équations (6.9), (6.10) et (6.12), ainsi que de la *borne du facteur deux* (proposition 3.6). De plus, nous substituons la valeur de $\mathrm{dis}_{Q_\mathbf{w}}(S,T)$ par l'égalité de l'équation (6.16). □

La sous-section suivante présente un algorithme directement basé sur le résultat du corollaire 6.10.

6.5.4 L'algorithme PBDA

Comme nous l'avons expliqué précédemment, la quantité $\xi_Q(D_\mathrm{S}, D_\mathrm{T})$ n'est pas calculable dans le contexte où aucune étiquette d'exemples cibles n'est à notre disposition. En posant l'hypothèse qu'il s'agit d'une quantité négligeable lorsque l'adaptation du domaine source vers le domaine cible est possible, nous suggérons d'optimiser l'expression de la borne sur le risque cible $R_D(h_\mathbf{w})$, donnée par le corollaire 6.10, en trouvant le vecteur $\mathbf{w}$ minimisant l'expression suivante :

$$\begin{aligned} & C\,m\,R_S(G_{Q_\mathbf{w}}) + A\,m\,\mathrm{dis}_{Q_\mathbf{w}}(S,T) + \mathrm{KL}(Q_\mathbf{w}\|P_\mathbf{0}) \\ & \quad = \quad C\sum_{i=1}^{m}\Phi\left(y_i^\mathrm{S}\frac{\mathbf{w}\cdot\mathbf{x}_i^\mathrm{S}}{\|\mathbf{x}_i^\mathrm{S}\|}\right) + A\left|\sum_{i=1}^{m}\Phi_\mathrm{dis}\left(\frac{\mathbf{w}\cdot\mathbf{x}_i^\mathrm{S}}{\|\mathbf{x}_i^\mathrm{S}\|}\right) - \Phi_\mathrm{dis}\left(\frac{\mathbf{w}\cdot\mathbf{x}_i^\mathrm{T}}{\|\mathbf{x}_i^\mathrm{T}\|}\right)\right| + \frac{\|\mathbf{w}\|^2}{2}\,, \end{aligned} \tag{6.17}$$

où A et C sont des constantes positives que deviendront les hyperparamètres de notre algorithme d'apprentissage. Notons que pour n'importe quelles valeurs de A et C, il est possible de retrouver des valeurs de a et c correspondante, de telle sorte le vecteur $\mathbf{w}$ minimisant l'expression du corollaire 6.10 est le même qui minimise la fonction objectif donnée par l'équation (6.17).

Le problème d'optimisation. L'équation (6.17) est fortement non convexe. Afin de rendre le problème d'optimisation plus facile à résoudre, nous remplaçons la fonction de perte $\Phi(\cdot)$ par sa relaxation convexe $\Phi_\mathrm{cvx}(\cdot)$, donnée par l'équation (6.14). Le nouveau problème d'optimisation consiste à trouver le vecteur $\mathbf{w}$ minimisant la fonction $G(\mathbf{w})$ suivante :

$$G(\mathbf{w}) \;\coloneqq\; C\sum_{i=1}^{m}\Phi_\mathrm{cvx}\left(y_i^\mathrm{S}\frac{\mathbf{w}\cdot\mathbf{x}_i^\mathrm{S}}{\|\mathbf{x}_i^\mathrm{S}\|}\right) + A\left|\sum_{i=1}^{m}\Phi_\mathrm{dis}\left(\frac{\mathbf{w}\cdot\mathbf{x}_i^\mathrm{S}}{\|\mathbf{x}_i^\mathrm{S}\|}\right) - \Phi_\mathrm{dis}\left(\frac{\mathbf{w}\cdot\mathbf{x}_i^\mathrm{T}}{\|\mathbf{x}_i^\mathrm{T}\|}\right)\right| + \frac{\|\mathbf{w}\|^2}{2}\,, \tag{6.18}$$

où, comme nous avons déjà défini,

$$\Phi(a) \overset{\text{def}}{=} \tfrac{1}{2}\left[1-\mathrm{erf}\left(\tfrac{a}{\sqrt{2}}\right)\right],\ \Phi_\mathrm{cvx}(a) \overset{\text{def}}{=} \max\left[\Phi(a), \tfrac{1}{2}-\tfrac{a}{\sqrt{2\pi}}\right] \text{ et } \Phi_\mathrm{dis}(a) \overset{\text{def}}{=} 2\,\Phi(a)\,\Phi(-a)\,.$$

La figure 6.1 illustre les trois fonctions $\Phi(\cdot)$, $\Phi_\mathrm{cvx}(\cdot)$ et $\Phi_\mathrm{dis}(\cdot)$ intervenants dans le problème d'optimisation. Comme mentionné à la sous-section 6.5.2, la fonction de perte convexifiée $\Phi_\mathrm{cvx}(\cdot)$ possède une forme similaire à la ***perte hinge*** intervenant dans le problème du SVM (voir la figure 2.3, page 21).

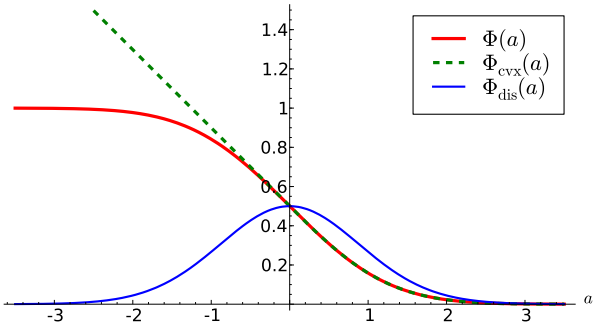


FIGURE 6.1 – Fonctions $\Phi(\cdot)$, $\Phi_{\text{cvx}}(\cdot)$ et $\Phi_{\text{dis}}(\cdot)$.

Procédure de minimisation. Par des calculs assez simples, on trouve que le gradient de la fonction $G(\mathbf{w})$ est exprimé par

$$\nabla G(\mathbf{w}) \;=\; \mathbf{w} + C\sum_{i=1}^{m}\Phi'_{\text{cvx}}\left(\frac{y_i^{\text{S}}\mathbf{w}\cdot\mathbf{x}_i^{\text{S}}}{\|\mathbf{x}_i^{\text{S}}\|}\right)\frac{y_i^{\text{S}}\mathbf{x}_i^{\text{S}}}{\|\mathbf{x}_i^{\text{S}}\|} + s'\times A\left[\sum_{i=1}^{m}\Phi'_{\text{dis}}\Big(\frac{\mathbf{w}\cdot\mathbf{x}_i^{\text{T}}}{\|\mathbf{x}_i^{\text{T}}\|}\Big)\frac{\mathbf{x}_i^{\text{T}}}{\|\mathbf{x}_i^{\text{T}}\|} - \Phi'_{\text{dis}}\Big(\frac{\mathbf{w}\cdot\mathbf{x}_i^{\text{S}}}{\|\mathbf{x}_i^{\text{S}}\|}\Big)\frac{\mathbf{x}_i^{\text{S}}}{\|\mathbf{x}_i^{\text{S}}\|}\right],$$

où $\Phi'_{\text{cvx}}(a)$ et $\Phi'_{\text{dis}}(a)$ sont respectivement les dérivées des fonctions $\Phi_{\text{cvx}}(\cdot)$ et $\Phi_{\text{dis}}(\cdot)$ évaluées au point a, que l'on peut exprimer ainsi :

$$\Phi'_{\text{cvx}}(a) \;=\; \begin{cases} -\frac{1}{\sqrt{2\pi}} & \text{si } a \le 0 \\ -\frac{1}{\sqrt{2\pi}}\cdot\exp\left(-\frac{a^2}{2}\right) & \text{sinon}\,, \end{cases} \tag{6.19}$$

$$\Phi'_{\text{dis}}(a) \;=\; -\sqrt{\frac{2}{\pi}}\cdot\text{erf}\left(\frac{a}{\sqrt{2}}\right)\cdot\exp\left(-\frac{a^2}{2}\right). \tag{6.20}$$

De plus,

$$s' \;\coloneqq\; \text{sgn}\left[\sum_{i=1}^{m}\Phi_{\text{dis}}\left(\frac{\mathbf{w}\cdot\mathbf{x}_i^{\text{S}}}{\|\mathbf{x}_i^{\text{S}}\|}\right) - \Phi_{\text{dis}}\left(\frac{\mathbf{w}\cdot\mathbf{x}_i^{\text{T}}}{\|\mathbf{x}_i^{\text{T}}\|}\right)\right].$$

Nous nommons ***PBDA*** (de l'anglais *PAC-Bayesian Domain Adaptation*) l'algorithme d'adaptation de domaine qui consiste à minimiser la fonction $G(\mathbf{w})$ par la méthode de descente en gradient.[4] Cette tâche d'optimisation demeure non convexe, malgré la relaxation de la fonction objectif par laquelle nous avons transformé l'équation (6.17) en l'équation (6.18), puisque la fonction $\Phi_{\text{dis}}(\cdot)$ est quasi concave. Cependant, notre étude empirique montre qu'il n'est pas nécessaire d'effectuer plusieurs redémarrages aléatoires de la descente en gradient pour trouver une solution convenable.

6.5.5 L'algorithme PBDA dans l'espace dual

En appliquant l'***astuce du noyau*** – décrite à la section 2.2.2 (page 22) – à PBDA, nous pouvons travailler avec le vecteur de poids dual $\boldsymbol{\alpha} \in \mathbb{R}^{2m}$, correspondant à un classificateur linéaire dans un espace «augmenté».

4. Notre implémentation de PBDA utilise la méthode de descente en gradient de *Broyden-Fletcher-Goldfarb-Shanno (BFGS)* implémentée dans la librairie Python *scipy* (`http://www.scipy.org/`).

Considérons une ***fonction de noyau*** $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, un échantillon source $S \coloneqq \{(\mathbf{x}_i^{\mathrm{S}}, y_i^{\mathrm{S}})\}_{i=1}^m$ et un échantillon cible $T \coloneqq \{(\mathbf{x}_i^{\mathrm{T}}, \cdot)\}_{i=1}^m$. En s'inspirant de l'équation (2.13), on exprime un classificateur comme une combinaison linéaire des produits scalaires donnés par les exemples des échantillons S et T :

$$h_{\boldsymbol{\alpha}}(x) \overset{\text{def}}{=} \operatorname{sgn}\left[\sum_{i=1}^{m} \alpha_i k(x_i^{\mathrm{S}}, x) + \sum_{i=1}^{m} \alpha_{i+m} k(x_i^{\mathrm{T}}, x)\right].$$

Dans le texte qui suit, nous notons K la ***matrice de noyau*** de taille $2m \times 2m$ telle que

$$K_{i,j} \overset{\text{def}}{=} k(x_i, x_j)\,,$$

où

$$x_{\#} \coloneqq \begin{cases} x_{\#}^{\mathrm{S}} & \text{si } \# \leq m \\ x_{\#-m}^{\mathrm{T}} & \text{sinon.} \end{cases}$$

Avec cette définition, la fonction objectif donnée par l'équation (6.18) peut être réécrite en termes du vecteur de variables duales $\boldsymbol{\alpha} \coloneqq (\alpha_1, \alpha_2, \ldots \alpha_{2m})$:

$$\begin{aligned} G(\boldsymbol{\alpha}) \coloneqq C \sum_{i=1}^{m} \Phi_{\text{cvx}}\left(y_i^{\mathrm{S}} \frac{\sum_{j=1}^{2m} \alpha_j K_{i,j}}{\sqrt{K_{i,i}}}\right) \\ + A\left|\sum_{i=1}^{m}\left[\Phi_{\text{dis}}\left(\frac{\sum_{j=1}^{2m} \alpha_j K_{i,j}}{\sqrt{K_{i,i}}}\right) - \Phi_{\text{dis}}\left(\frac{\sum_{j=1}^{2m} \alpha_j K_{i+m,j}}{\sqrt{K_{i+m,i+m}}}\right)\right]\right| + \frac{1}{2}\sum_{i=1}^{2m}\sum_{j=1}^{2m} \alpha_i \alpha_j K_{i,j}\,. \end{aligned} \tag{6.21}$$

Le ***théorème du représentant*** (théorème A.9, en annexe) assure que le minimum de la fonction objectif $G(\cdot)$, en fonction des variables duales $\boldsymbol{\alpha}$, coïncide avec le minimum de la fonction d'optimisation originale exprimée par l'équation (6.18) dans l'espace induit par le noyau $k(\cdot, \cdot)$. L'algorithme ***PBDA*** dans l'espace dual consiste donc à trouver, par descente en gradient, le vecteur $\boldsymbol{\alpha}$ minimisant la fonction objectif $G(\boldsymbol{\alpha})$.

Le gradient de l'équation (6.21) est donné par le vecteur $\nabla G(\boldsymbol{\alpha}) \coloneqq (\alpha'_1, \alpha'_2, \ldots \alpha'_{2m})$, avec

$$\begin{aligned} \alpha'_{\#} \coloneqq C \sum_{i=1}^{m} \Phi'_{\text{cvx}}\left(y_i^{\mathrm{S}} \frac{\sum_{j=1}^{2m} \alpha_j K_{i,j}}{\sqrt{K_{i,i}}}\right) \frac{y_i^{\mathrm{S}} K_{i,\#}}{\sqrt{K_{i,i}}} + \sum_{j=1}^{2m} \alpha_i K_{i,\#} \\ + s' \times A\left(\sum_{i=1}^{m}\left[\Phi'_{\text{dis}}\left(\frac{\sum_{j=1}^{2m} \alpha_j K_{i,j}}{\sqrt{K_{i,i}}}\right) \frac{K_{i,\#}}{\sqrt{K_{i,i}}} - \Phi'_{\text{dis}}\left(\frac{\sum_{j=1}^{2m} \alpha_j K_{i+m,j}}{\sqrt{K_{i+m,i+m}}}\right) \frac{K_{i+m,\#}}{\sqrt{K_{i+m,i+m}}}\right]\right), \end{aligned}$$

où les définitions de $\Phi'_{\text{cvx}}(a)$ et $\Phi'_{\text{dis}}(a)$ sont données par les équations (6.19) et (6.20). De plus,

$$s' \coloneqq \operatorname{sgn}\left(\sum_{i=1}^{m}\left[\Phi_{\text{dis}}\left(\frac{\sum_{j=1}^{2m} \alpha_j K_{i,j}}{\sqrt{K_{i,i}}}\right) - \Phi_{\text{dis}}\left(\frac{\sum_{j=1}^{2m} \alpha_j K_{i+m,j}}{\sqrt{K_{i+m,i+m}}}\right)\right]\right).$$

Tout comme le SVM, l'algorithme d'apprentissage PBDA peut être utilisé à la fois pour construire une frontière de décision linéaire dans l'espace des exemples ou un classificateur plus complexe par l'utilisation d'une fonction de noyau.

6.5.6 Sélection de modèle par validation croisée inverse

L'algorithme PBDA possède deux hyperparamètres (A et C). En variant les valeurs de ces hyperparamètres, nous pouvons modifier grandement la fonction objectif minimisée, et ainsi influencer le choix du classificateur retourné par l'algorithme. Une question cruciale en adaptation de domaine est celle de la ***sélection de modèle*** :

> Comment choisir les valeurs des hyperparamètres alors que nous ne possédons aucun exemple étiqueté provenant de la distribution cible ?

Dans les expérimentations présentées à la section 6.6, nous utilisons la ***validation croisée inverse***, suggérée par Zhong et al. (2010). Pour appliquer la *validation croisée inverse k-fois*, on divise d'abord en k sous-ensembles l'échantillon source (étiqueté) et l'échantillon cible (non étiqueté), de telle sorte que $S = S_1 \cup \ldots \cup S_k$ et $T = T_1 \cup \ldots \cup T_k$. Pour un algorithme d'adaptation de domaine et un tuple d'hyperparamètres donné, on calcule le *risque de validation croisée inverse* comme suit.

Pour chaque «fois» $i \in \{1, \ldots, k\}$:

1. L'algorithme est exécuté avec $S \setminus S_i$ comme échantillon source (étiqueté) et $T \setminus T_i$ comme échantillon cible (non étiqueté) pour obtenir le classificateur $h_i(\cdot)$.
2. Le même algorithme est exécuté avec $\{(x, h_i(x))\}_{x \in T \setminus T_i}$ comme échantillon source (étiqueté par le classificateur $h_i(\cdot)$) et $S \setminus S_i$ comme échantillon cible (en ignorant les étiquettes) pour obtenir le *classificateur inverse* $h'_i(\cdot)$.
3. Le risque du *classificateur inverse* $h'_i(\cdot)$ est calculé sur le sous-ensemble S_i.

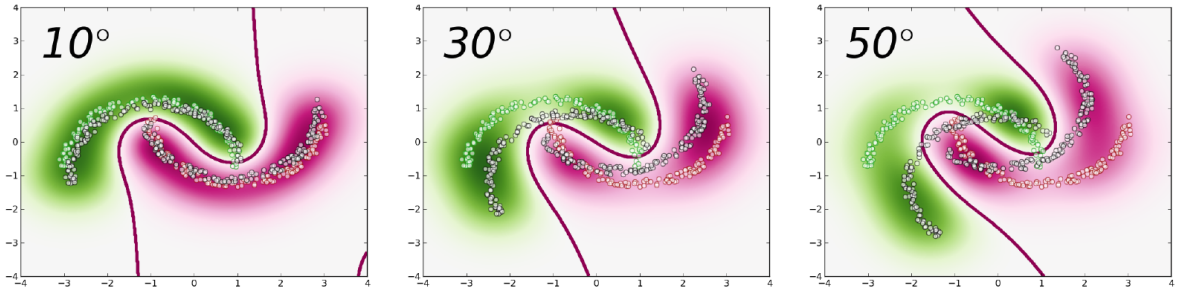
Après avoir répété la procédure pour les k «fois», on calcule risque de validation croisée inverse en effectuant la moyenne des risques des classificateurs inverses $h'_1(\cdot), \ldots, h'_k(\cdot)$. On répète la procédure avec tous les tuples d'hyperparamètres candidats, et on exécute finalement l'algorithme sur la totalité des échantillons source S et cible T, en utilisant le tuple d'hyperparamètres ayant mené au risque de validation croisée inverse minimal.

6.6 Expérimentations

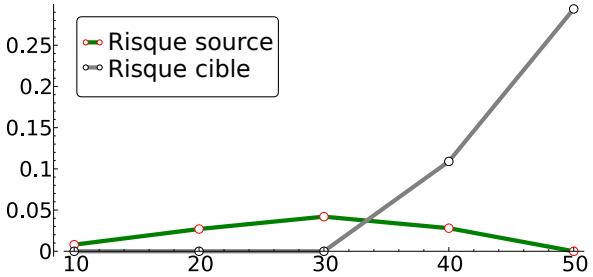
Nous avons évalué l'algorithme PBDA sur un échantillon de données jouet, ainsi que sur un échantillon de données réelles communément utilisé en adaptation de domaine.

6.6.1 Problème jouet : les lunes jumelles

Nous considérons comme domaine source le problème des *lunes jumelles*, qui s'avère un problème jouet de classification binaire classique. Les exemples d'apprentissage sont divisés en deux lunes distinctes (voir la figure 6.2). Nous étudions cinq domaines cibles différents, obtenus par la rotation du domaine source de 10, 20, 30, 40, puis 50 degré. Pour chaque problème, nous



(a) Frontière de décision du classificateur à la suite de l'apprentissage. Les points verts et les points rouges correspondent aux exemples de l'échantillon source (les couleurs indiquent les étiquettes des exemples), tandis que les points gris correspondent aux exemples de l'échantillon cible (non étiquetté).



(b) Évolution du risque source et du risque cible en fonction de l'angle de rotation de l'échantillon cible (moyenné sur 10 tirages des échantillons source et cible).

FIGURE 6.2 – Comportement de l'algorithme PBDA sur le problème des lunes jumelles en fonction de l'angle de rotation de l'échantillon cible. Les paramètres sont fixés à $A = 1$ et $C = 1$.

générons un échantillon source et un échantillon cible de 300 exemples (dont 150 exemples positifs et 150 exemples négatifs). Nous utilisons un noyau RBF de variance unitaire (autrement dit, $k(\mathbf{x}_i, \mathbf{x}_j) \coloneqq e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|}$, avec $\gamma \coloneqq 1$).

La figure 6.2a représente les frontières de décisions apprises sur les échantillons de données correspondant à une rotation de 10, 30 et 50 degré. Nous constatons que l'algorithme PBDA possède une bonne capacité d'adaptation sur ce problème jouet, même lorsque le problème est difficile (une rotation du domaine cible de 50 degré). Ce comportement suggère que notre divergence entre les domaines $\mathrm{dis}_Q(S, T)$ est un bon régularisateur dans un contexte d'adaptation de domaine.

Le graphique de la figure 6.2b illustre les risques du classificateur mesurés à la fois sur l'échantillon source et l'échantillon cible, moyenné sur 10 tirages aléatoires des échantillons de données. Il apparaît que PBDA réalise un compromis que nous pouvons expliquer intuitivement ainsi : l'algorithme «accepte» de perdre en performance sur le domaine source pour maintenir sa performance sur le domaine cible lorsque les domaines sont semblables (de 10 à 30 degré de rotation), et préfère se concentrer sur la minimisation du risque cible quand les domaines s'éloignent (40 et 50 degré).

source → cible	PBGD	SVM	DASVM	CODA	PBDA
books→dvd	**0.174**	0.179	0.193	0.181	0.183
books→electronics	0.275	0.290	**0.226**	0.232	0.263
books→kitchen	0.236	0.251	**0.179**	0.215	0.229
dvd→books	**0.192**	0.203	0.202	0.217	0.197
dvd→electronics	0.256	0.269	**0.186**	0.214	0.241
dvd→kitchen	0.211	0.232	0.183	**0.181**	0.186
electronics→books	0.268	0.287	0.305	0.275	**0.232**
electronics→dvd	0.245	0.267	**0.214**	0.239	0.221
electronics→kitchen	**0.127**	0.129	0.149	0.134	0.141
kitchen→books	0.255	0.267	0.259	**0.247**	**0.247**
kitchen→dvd	0.244	0.253	**0.198**	0.238	0.233
kitchen→electronics	0.235	0.149	0.157	0.153	**0.129**
Moyenne	0.226	0.231	**0.204**	0.210	0.208

TABLE 6.1 – Risque cible sur l'échantillon de données *Amazon reviews*. B, D, E, K correspondent respectivement aux domaines *books, DVDs, electronics, kitchen*.

6.6.2 Données réelles : appréciation de produits

Nous considérons l'échantillon de données *Amazon reviews* (Blitzer et al., 2006a) constitué d'*avis de consommateurs* portant sur quatre types de produits issus du site web *Amazon.com* : *books, dvd disks, electronics, kitchen appliances*. Chaque *avis* fait office d'exemple d'apprentissage. Le texte d'un commentaire à propos d'un produit (la description de l'exemple) est associé à une évaluation, c'est-à-dire une cote de 1 à 5 «étoiles» attribuée par un consommateur (l'étiquette de l'exemple). À l'origine, les descriptions des exemples sont fournies sous la forme de vecteurs de taille 100 000 approximativement. Chaque attribut de ces vecteurs réfère au nombre d'occurrences d'un unigramme ou d'un bigramme dans les textes des commentaires rédigés par les consommateurs. Pour simplifier la tâche de classification, nous suivons un protocole similaire à celui proposé par Chen et al. (2011). Les deux étiquettes possibles sont «+1» pour les produits ayant au moins 4 étoiles et «−1» pour ceux ayant au plus 3 étoiles. Un domaine correspond à un type de produit, ce qui permet de formuler 12 tâches d'adaptation de domaine distinctes en combinant les quatre types de produits en couples constitués d'un domaine source et d'un domaine cible (comme le domaine source et le domaine cible sont distincts dans chacune des tâches, nous avons bien $4 \times 3 = 12$ possibilités). La dimension des données est réduite de la manière suivante : étant donné un couple de domaines source-cible, Chen et al. (2011) ont uniquement gardé les attributs qui apparaissent au moins 10 fois dans les deux domaines (il reste environ 40 000 attributs), puis ont appliqué une pondération de type *tf-idf* sur les vecteurs ainsi obtenus.

D'une part, nous avons comparé les résultats obtenus par l'algorithme PBDA sur ces 12 tâches avec deux autres algorithmes d'apprentissage conçus pour le cadre d'apprentissage inductif :

— L'algorithme ***PBGD*** (Germain et al., 2009a; Germain, 2009), dans sa version convexifiée décrite à la section 6.5.2.

— Le ***SVM*** décrit à la section 2.2.1. [5]

Puisque les algorithmes PBGD et SVM ne sont pas conçus pour s'adapter à un domaine cible, nous les avons entraînés en utilisant seulement l'échantillon de données source.

D'autre part, nous avons aussi comparé nos résultats avec deux algorithmes d'apprentissage conçus pour l'adaptation de domaine :

— L'algorithme ***DASVM*** de Bruzzone et Marconcini (2010), qui maximise itérativement une notion de marge sur des exemples cibles étiquetés par l'algorithme lui-même. [6]

— L'algorithme ***CODA*** de Chen et al. (2011), qui utilise une méthode dite de «coapprentissage» en sélectionnant itérativement des attributs cibles reliés à l'échantillon d'entraînement. [7] Notons que CODA a obtenu les meilleurs résultats sur l'échantillon de données *Amazon reviews* selon la comparaison présentée par Chen et al. (2011).

À l'instar de PBDA, les algorithmes DASVM et CODA s'entraînent simultanément sur un échantillon de données source et un échantillon de données cible.

Lors de cette expérimentation, les cinq algorithmes d'apprentissage utilisent un noyau linéaire et considèrent 2 000 exemples sources étiquetés et 2 000 exemples cibles non étiquetés. Les hyperparamètres sont sélectionnés parmi une grille prédéfinie, par une ***validation croisée*** 5-*fois* sur l'échantillon source pour PBGD et SVM, et par une ***validation croisée inverse*** 5-*fois*, à l'aide des échantillons source et cible, pour CODA, DASVM et PBDA. Rappelons que la méthode de la *validation croisée* a été introduite à la section 2.5.2 (page 36) et la *validation croisée inverse* à la section 6.5.6 (page 136). Nous évaluons les classificateurs ainsi obtenus sur les échantillons de test cibles proposés par Chen et al. (2011) – qui contiennent entre 3 000 et 6 000 exemples chacun – puis nous reportons les résultats dans la table 6.1.

Nous remarquons que les trois algorithmes conçus par l'adaptation de domaine (DASVM, CODA et PBDA) obtiennent les meilleurs résultats en moyenne que les algorithmes conçus pour le cadre d'apprentissage inductif (PBGD et SVM). L'algorithme PBDA est en moyenne plus performant que CODA, mais moins performant que DASVM. Cependant, PBDA s'avère un algorithme d'adaptation de domaine compétitif avec l'état de l'art : les résultats de PBDA ne sont pas significativement différents de ceux obtenus par CODA et DASVM. De plus, notons que le temps d'exécution de PBDA est substantiellement plus court que celui de CODA et de DASVM. Tel que décrit plus haut, ces deux algorithmes utilisent une méthode itérative, alternant entre l'optimisation de deux critères. Ainsi, un avantage de PBDA est d'optimiser conjointement les termes de notre borne en une seule étape.

5. Nous avons utilisé le logiciel SVM-light (Joachims, 1999) : `http://svmlight.joachims.org/`
6. L'implémentation utilise le logiciel LibSVM (Chang et Lin, 2011).
7. L'implémentation de CODA est disponible : `http://www1.cse.wustl.edu/~mchen/code/coda.tar`

6.6.3 Conclusion

Rappelons que PBDA est le premier algorithme pour l'adaptation de domaine basé sur la théorie PAC-bayésienne. Nos résultats empiriques montrent qu'il s'agit d'une approche judicieuse. Plusieurs variations sont envisageables. Notamment, nous pourrions reformuler la borne sur le risque cible du théorème 6.8 en bornant l'espérance de désaccord différemment (par le théorème 6.6, par exemple) ou en estimant le terme $\xi_Q(D_S, D_T)$ (ce qui serait possible à l'aide de la proposition 6.5, en posant des hypothèses supplémentaires sur la nature de la distribution cible). Ces possibilités permettraient de formuler d'autres algorithmes d'apprentissage, qui exprimeraient des compromis légèrement différents de ceux en jeu au sein de PBDA. Nous réservons l'exploration de ces possibilités pour des travaux futurs.

6.7 Synthèse des contributions du chapitre

Dans ce chapitre, nous avons proposé une première approche PAC-bayésienne de l'adaptation de domaine. En commençant par une analyse théorique du problème, nous avons progressé vers la conception d'un algorithme d'apprentissage.

D'abord inspirés par la $\mathcal{H}\Delta\mathcal{H}$-distance de Ben-David et al. (2006, 2010), nous avons suggéré une nouvelle notion de distance entre les distributions source et cible (définition 6.3, page 119). Cette pseudo-métrique, que nous nommons le *désaccord entre les distributions* $\mathrm{dis}_Q(D_S, D_T)$, se distingue de la $\mathcal{H}\Delta\mathcal{H}$-distance par le fait qu'elle prend en considération une distribution Q sur l'ensemble des votants $\mathcal{H}$.

Nous énonçons ensuite, par le théorème 6.4 (page 120), une borne sur le risque de Gibbs calculé sur la distribution cible $R_{D_T}(G_Q)$. Cette dernière dépend principalement du risque calculé sur la distribution source $R_{D_S}(G_Q)$ et du désaccord entre les distributions $\mathrm{dis}_Q(D_S, D_T)$. En combinant les bornes PAC-bayésiennes de chacun de ces deux termes, nous exprimons par le théorème 6.7 (page 124) une garantie sur $R_{D_T}(G_Q)$ qui est obtenue à partir d'échantillons d'entraînement source $S \sim (D_S)^m$ et cible $T \sim (D_T)^m$.

Ce théorème PAC-bayésien pour l'adaptation de domaine est à la base de la conception d'un nouvel algorithme d'apprentissage, nommé PBDA (section 6.5, page 127). Inspiré de l'algorithme «inductif» PBGD présenté dans Germain (2009); Germain et al. (2009a), l'algorithme «adaptatif» PBDA construit des classificateurs linéaires dans l'espace des exemples ou, par le recours à l'*astuce du noyau*, dans un espace de dimension plus élevée.

Par des expérimentations (section 6.6, page 136) sur le problème jouet des deux lunes jumelles, nous avons illustré la capacité d'adaptation de PBDA. De même, une comparaison avec d'autres algorithmes conçus pour l'adaptation de domaine sur les données *Amazon Reviews* a montré que PBDA possède des performances semblables à l'état de l'art.

Quatrième partie

Perspectives

Chapitre 7

Aperçu des autres contributions

Tel qu'annoncé dans l'avant-propos (page xix), certains travaux effectués lors du doctorat n'ont pas été couverts par les chapitres précédents de cette thèse, soit parce qu'ils sont issus de collaborations avec d'autres étudiants et ont fait (ou feront) l'objet d'autres thèses, soit parce qu'ils ne s'inscrivent pas dans l'axe «PAC-bayésien» de la thèse, mais découlent de réflexions provoquées par l'étude des divers cadres d'apprentissage. Par le présent chapitre, nous voulons donner un aperçu de ces autres contributions et les mettre en relation avec les sujets abordés précédemment.

Le lecteur qui désire en apprendre davantage sur les sujets abordés dans les prochaines sections est invité à consulter les annexes C, D et E, qui contiennent les manuscrits se rapportant à ces «autres contributions», de même que l'article de revue Germain et al. (2015b). Nous expliquons dans les paragraphes suivants en quoi chacun de ces documents est lié à nos «autres contributions».

Annexe C : «A PAC-Bayes Sample Compression Approach to Kernel Methods» (Germain et al., 2011).

Cet article constitue le travail au cœur de la thèse de Sara Shanian (Shanian, 2012). Nous avons néanmoins collaboré à plusieurs aspects de la théorie qui y est présentée. Parmi eux, nous abordons à la section 7.1 un stratagème qui permet d'énoncer des théorèmes PAC-bayésiens exempts de l'habituel terme de complexité «$\mathrm{KL}(Q\|P)$». Aussi, nous présentons à la section 7.2 les bases d'une théorie permettant d'étudier des votes de majorité de classificateurs comprimés. Comme expliqué brièvement à la sous-section 7.2.4, cette approche ouvre la voie à l'apprentissage des mesures de similarité.

Annexe D : «A Pseudo-Boolean Set Covering Machine» (Germain et al., 2012a).

Cet article décrit les travaux, présentés brièvement à la section 7.3, qui nous ont permis d'évaluer empiriquement l'efficacité de l'algorithme SCM en minimisant directement la borne provenant de la théorie de la compression d'échantillons.

Annexe E : «Domain-Adversarial Neural Networks» (Ajakan et al., 2014).

Cet article décrit les travaux, abordés brièvement à la section 7.4, par lesquels nous avons intégré les idées provenant de la théorie de l'adaptation de domaine de Ben-David et al. (2006, 2010) à un *réseau de neurones*. Rappelons que la théorie de Ben-David et al. a été présentée à la section 6.2 (page 116).

Autre article important : «Risk Bounds for the Majority Vote : From a PAC-Bayesian Analysis to a Learning Algorithm» (Germain et al., 2015b).

Cet article de revue contient plusieurs éléments présentés par la partie II de la thèse (chapitres 3 et 4), c'est pourquoi nous ne l'avons pas inclus en annexe à la thèse. Mentionnons toutefois qu'il s'agit d'un document de 74 pages contenant une introduction pédagogique à la théorie PAC-bayésienne ainsi qu'une présentation de nombreux résultats plus avancés. Bien que ces résultats ont fait l'objet d'autres publications de notre groupe de recherche, l'article Germain et al. (2015b) les présente dans un tout unifié. Nous suggérons fortement ledit article aux lecteurs qui cherchent une référence de langue anglaise sur la théorie PAC-bayésienne. Notons enfin que nous reprenons aux sections 7.1 et 7.2 ci-bas l'approche adoptée dans cet article de revue, afin donner un aperçu de certaines contributions, tout en adaptant la notation pour se conformer à la notation propre aux chapitres précédents.

7.1 Théorèmes PAC-bayésiens sans terme $\mathrm{KL}(Q\|P)$

Toutes les variantes des théorèmes PAC-bayésiens que nous avons présentées au cours de la thèse contiennent un terme $\mathrm{KL}(Q\|P)$, qui correspond à la ***divergence Kullback-Leibler*** entre la distribution *a priori* P et la distribution *a posteriori* Q (voir l'équation (4.1), page 58). Nous pourrions croire qu'il s'agit d'une caractéristique intrinsèque à la théorie PAC-bayésienne.

Cependant, certaines approches bien particulières permettent d'énoncer des théorèmes PAC-bayésiens qui ne contiennent pas ce terme «$\mathrm{KL}(Q\|P)$» (voir, outre l'approche décrite dans le texte qui suit, Catoni, 2007; Lever et al., 2013). L'approche présentée ici fut publiée pour la première fois dans l'article Germain et al. (2011), que nous incluons à l'annexe C (voir plus spécifiquement la section C.2.1, page 182). Elle a ensuite paru dans la thèse de Sara Shanian (Shanian, 2012), de même que reprise dans Roy et al. (2011). Pour donner un aperçu de cette approche, nous adoptons la présentation de Germain et al. (2015b, section 5) qui contient une vision simplifiée du résultat. [1]

1. Précisons que je suis l'auteur principal de cette réécriture de la théorie.

7.1.1 Distributions alignées et inégalité de changement de mesure

Considérons le cadre PAC-bayésien «classique», que nous avons approfondi au chapitre 4, dans lequel nous travaillons avec un ensemble de votants $\mathcal{H}$, une distribution *a posteriori* Q sur $\mathcal{H}$, ainsi qu'une distribution *a priori* P sur $\mathcal{H}$. Nous verrons que nous pouvons exprimer des théorèmes PAC-bayésiens sans terme $\mathrm{KL}(Q\|P)$ si l'ensemble $\mathcal{H}$ est *auto-complémenté* et que la distribution Q est *alignée sur* la distribution P, deux concepts introduits par les définitions suivantes.

Définition 7.1. Un ensemble de votants $\mathcal{H}$ est *auto-complémenté* si

$$-f \in \mathcal{H}\,, \qquad \forall f \in \mathcal{H}\,.$$

Définition 7.2. Une distribution Q sur un ensemble auto-complémenté $\mathcal{H}$ est *alignée* sur une distribution *a priori* P lorsque

$$Q(f) + Q(-f) \;=\; P(f) + P(-f)\,, \qquad \forall f \in \mathcal{H}\,.$$

Le prochain résultat (lemme 7.3, ci-bas) présente le résultat central à notre approche. Il s'agit d'une réécriture de l'***inégalité du changement de mesure*** – introduite par le lemme 4.3 (page 60) – que nous spécialisons aux distributions Q alignées. Alors que le terme $\mathrm{KL}(Q\|P)$ «apparaît» habituellement à cette étape des démonstrations PAC-bayésienne, il n'en est rien lorsque la distribution Q est alignée sur une distribution P.

Lemme 7.3 (Inégalité de changement de mesure pour distributions alignées)**.** *Pour tout ensemble $\mathcal{H}$ auto-complémenté, pour toute distribution P sur $\mathcal{H}$, pour toute distribution Q alignée sur P, et pour toute fonction mesurable $\phi : \mathcal{H} \to \mathbb{R}$ telle $\phi(f) = \phi(-f)$ pour tout $f \in \mathcal{H}$, on a*

$$\mathop{\mathbf{E}}_{f\sim Q} \phi(f) \;\le\; \ln\left(\mathop{\mathbf{E}}_{f\sim P} e^{\phi(f)}\right).$$

Démonstration. Montrons d'abord que l'on peut changer l'espérance sur Q en une espérance sur P, c'est-à-dire $\mathbf{E}_{f\sim Q}\,\phi(f) = \mathbf{E}_{f\sim P}\,\phi(f)$, en utilisant le fait que $\phi(f) = \phi(-f)$ pour tout $f \in \mathcal{H}$ et que Q est aligné sur P :

$$\begin{aligned}
2 \cdot \mathop{\mathbf{E}}_{f\sim Q} \phi(f) \;&=\; \int_{\mathcal{H}} df\; Q(f)\,\phi(f) + \int_{\mathcal{H}} df\; Q(-f)\,\phi(-f) \\
&=\; \int_{\mathcal{H}} df\; Q(f)\,\phi(f) + \int_{\mathcal{H}} df\; Q(-f)\,\phi(f) \\
&=\; \int_{\mathcal{H}} df\; \Big(Q(f) + Q(-f)\Big)\,\phi(f) \\
&=\; \int_{\mathcal{H}} df\; \Big(P(f) + P(-f)\Big)\,\phi(f) \\
&=\; \int_{\mathcal{H}} df\; P(f)\,\phi(f) + \int_{\mathcal{H}} df\; P(-f)\,\phi(f)
\end{aligned}$$

$$= \int_{\mathcal{H}} df\ P(f)\,\phi(f) + \int_{\mathcal{H}} df\ P(-f)\,\phi(-f)$$
$$= 2 \cdot \mathop{\mathbf{E}}_{f\sim P} \phi(f)\,.$$

Ensuite, le résultat découle d'une simple application de l'inégalité de Jensen (lemme A.2, en annexe) :

$$\begin{aligned}
\mathop{\mathbf{E}}_{f\sim Q} \phi(f) &= \mathop{\mathbf{E}}_{f\sim P} \phi(f) \\
&= \mathop{\mathbf{E}}_{f\sim P} \ln e^{\phi(f)} \\
&\leq \ln\left(\mathop{\mathbf{E}}_{f\sim P} e^{\phi(f)}\right). \qquad \langle\ \textit{Inégalité de Jensen}\ \rangle
\end{aligned}$$

□

7.1.2 Nouveaux théorèmes PAC-bayésiens

L'inégalité du changement de mesure du lemme 7.3 permet de reformuler le théorème PAC-bayésien général pour le cadre inductif, correspondant au théorème 4.4 (page 61). Par souci de simplicité, nous considérons seulement la perte linéaire dans le théorème 7.4 suivant (voir Germain et al., 2015b, annexe B, pour un résultat beaucoup plus général).

Théorème 7.4. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ auto-complémenté de votants $\mathcal{X} \to \overline{\mathcal{Y}}$, pour toute distribution P sur $\mathcal{H}$, pour tout $\delta \in (0,1]$, et pour toute fonction convexe $\Delta : [0,1]\times[0,1] \to \mathbb{R}$ telle que $\Delta(q,p) = \Delta(1-q,1-p)$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S \sim D^m$,*

$$\forall\, Q \text{ aligné sur } P: \quad \Delta\Big(\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}_\ell}(f), \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}_\ell}(f)\Big) \ \leq\ \frac{1}{m}\left[\ln \frac{\mathcal{I}_\Delta(m)}{\delta}\right],$$

où $\mathcal{I}_\Delta(m)$ est défini par l'équation (4.3), page 61.

Démonstration. La démonstration suit les mêmes étapes que la démonstration du théorème 4.4, en utilisant la perte linéaire $\mathcal{L} \coloneqq \mathcal{L}_\ell$ et en remplaçant l'inégalité du changement de mesure (lemme 4.3) par l'inégalité de changement de mesures pour distributions alignées (lemme 7.3), avec

$$\phi(f) \ \coloneqq\ m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}_\ell}(f), \mathbb{E}_D^{\mathcal{L}_\ell}(f)\right).$$

Précisons que la fonction $\phi(f)$ ci-dessus possède la propriété $\phi(f) = \phi(-f)$ requise par le lemme 7.3, car $\Delta(q,p) = \Delta(1-q,1-p)$ et $\mathbb{E}_{D'}^{\mathcal{L}_\ell}(\cdot)$ est une fonction linéaire. On a donc

$$\Delta\left(\mathbb{E}_S^{\mathcal{L}_\ell}(f), \mathbb{E}_D^{\mathcal{L}_\ell}(f)\right) = \Delta\left(1-\mathbb{E}_S^{\mathcal{L}_\ell}(-f), 1-\mathbb{E}_D^{\mathcal{L}_\ell}(-f)\right) = \Delta\left(\mathbb{E}_S^{\mathcal{L}_\ell}(-f), \mathbb{E}_D^{\mathcal{L}_\ell}(-f)\right).$$

La suite de la démonstration suit exactement les mêmes étapes que la démonstration du théorème 4.4. □

Le théorème 7.4 peut être spécialisé à plusieurs Δ-fonctions. Notons toutefois qu'elles doivent respecter la propriété $\Delta(q,p) = \Delta(1-q,1-p)$ pour tout $q,p \in [0,1]$, ce qui disqualifie la fonction $\Delta_c(q,p)$ de l'équation (4.13) (page 66). Toutefois, la fonction $\mathrm{kl}(q,p)$ – équation (4.6), page 63 – possède ladite propriété. Il en est de même de la fonction $2\,(q-p)^2$ provenant de l'inégalité de Pinsker de l'équation (4.10) (page 64). Ces observations permettent d'exprimer le corollaire suivant, qui présente deux bornes sur le risque de Gibbs qui ne dépendent pas du terme $\mathrm{KL}(Q\|P)$. Ces nouvelles bornes sont fortement inspirées de celles du corollaire 4.8 (page 68).

Corollaire 7.5. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble auto-complémenté $\mathcal{H}$ de votants $\mathcal{X} \to [-1,1]$, pour toute distribution P sur $\mathcal{H}$ et pour tout $\delta \in (0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S \sim D^m$,*

$\forall\, Q$ *aligné sur* P :

(a) $\mathrm{kl}\Big(R_S(G_Q), R_D(G_Q)\Big) \leq \frac{1}{m}\left[\ln \frac{2\sqrt{m}}{\delta}\right]$,

(b) $R_D(G_Q) \leq R_S(G_Q) + \sqrt{\frac{1}{2m}\left[\ln \frac{2\sqrt{m}}{\delta}\right]}$.

Démonstration. Le résultat (a) découle du théorème 7.4, car $R_{D'}(G_Q) = \mathbf{E}_{f\sim Q}\mathbb{E}^{\mathcal{L}_\ell}_{D'}(f)$ et $\mathrm{kl}(q,p) = \mathrm{kl}(1-q,1-p)$. Le résultat (b) est obtenu à partir du résultat (a), en appliquant l'inégalité de Pinsker (équation (4.10), page 64). □

L'article Germain et al. (2015b) présente des théorèmes sans terme $\mathrm{KL}(Q\|P)$ semblables à celles du corollaire 7.5, mais pour l'espérance de désaccord. Il s'agit donc essentiellement des bornes du corollaire 4.12(a,b) spécialisées pour distributions Q alignées sur P. Les théorèmes PAC-bayésiens sans terme $\mathrm{KL}(Q\|P)$ de l'article Germain et al. (2009a) – annexe C de la thèse – sont valides pour des fonctions de pertes plus complexes, qui peuvent s'exprimer à l'aide d'un tuple de votants de cardinalité quelconque. Cela dit, ces résultats reposent essentiellement sur les idées que nous avons présentées dans cette section.

7.1.3 Conception d'algorithmes d'apprentissage

Les bornes PAC-bayésiennes sans terme $\mathrm{KL}(Q\|P)$ sont utilisées dans Germain et al. (2011); Roy et al. (2011); Shanian (2012); Germain et al. (2015b) comme inspirations menant à la conception d'algorithmes d'apprentissage. Contrairement aux algorithmes PBGD et PBDA présentés à la section 6.5 (page 127), où le terme $\mathrm{KL}(Q\|P)$ devient un régularisateur dans la fonction objective à minimiser, les algorithmes ainsi obtenus possèdent un autre type de régularisateur. Ce dernier n'apparaît pas directement dans l'expression de la fonction objective que les algorithmes optimisent. En effet, le nouveau régularisateur prend la forme d'une contrainte sur les valeurs admises par distribution Q, provenant du fait que Q doit demeurer *aligné* sur la distribution P.

7.2 Apprentissage PAC-bayésien et compression d'échantillons

Nous avons brièvement présenté la théorie de la ***compression d'échantillons*** à la section 2.3 (page 27). Rappelons que cette théorie étudie les classificateurs qui sont exprimés à l'aide des exemples d'entraînement. À première vue, cette approche semble incompatible avec l'approche PAC-bayésienne. En effet, tous les théorèmes PAC-bayésiens que nous avons présentés dans les chapitres précédents nécessitent que l'ensemble des votants $\mathcal{H}$ soit défini *a priori*, c'est-à-dire avant de prendre en considération les exemples d'apprentissage.

Dans cette section, nous abordons une théorie PAC-bayésienne où les votants peuvent être des classificateurs exprimés à l'aide d'un ensemble de compression. Les premiers résultats menant à cette théorie sont attribuables à Laviolette et al. (2005); Laviolette et Marchand (2005, 2007). Les sous-sections 7.2.1 et 7.2.2 présentent une version simplifiée de cette théorie, qui a été développée dans Germain et al. (2015b, section 6). Comme nous l'abordons à la sous-section 7.2.3, cette théorie, bien que simplifiée, permet entre autres d'énoncer des bornes sur les classificateurs linéaires reposant sur une fonction de noyau $k(\cdot, \cdot)$. Cette théorie a aussi mené à la conception d'algorithmes d'apprentissage où la fonction $k(\cdot, \cdot)$ peut être une mesure de similarité quelconque entre deux exemples (et non seulement un noyau semi-défini positif). Aussi, nous donnons un aperçu à la sous-section 7.2.4 de la théorie plus avancée développée dans l'article Germain et al. (2011), que nous incluons à l'annexe C. Cette approche ouvre la voie à la conception d'algorithmes qui apprennent la fonction de similarité.

7.2.1 Une vision simplifiée de la compression d'échantillons

Le texte qui suit présente les définitions de base avec lesquelles nous pouvons élaborer une théorie PAC-bayésienne pour les votes de majorité dont les votants dépendent des exemples d'apprentissage. Nous utilisons le terme ***votant comprimé*** pour désigner un votant qui, comme tout ***classificateur comprimé*** (voir la section 2.3, page 27) peut être *reconstruit* par une fonction $\mathcal{R}(S_{\mathbf{i}}, \boldsymbol{\sigma})$, où $S_{\mathbf{i}}$ est un sous-ensemble de l'échantillon d'entraînement S et $\boldsymbol{\sigma}$ est un *message* qui contient de l'information supplémentaire nécessaire pour décrire ce votant/classificateur.

Étudions maintenant une situation simplifiée de la théorie de la compression. La méthode permettant de déduire des bornes PAC-bayésiennes pour votants comprimés que nous présentons ci-bas se généralise au cadre général présenté à la section 2.3, mais la notation y est nettement plus complexe. Dans notre cadre simplifié, nous considérons que les votants comprimés s'expriment par un ***ensemble de compression*** d'au plus λ exemples (contenant possiblement des répétitions) et un ***message*** que nous représentons par une séquence de «-1» et «$+1$». La distribution Q n'est pas directement définie sur l'ensemble de votants comprimés, mais

plutôt sur $I_\lambda \times \Sigma_\lambda$, où

$$I_\lambda \stackrel{\text{def}}{=} \left\{ \langle i_1, i_2, .., i_l \rangle \in \{1, .., m\}^l \;\middle|\; l \in \{1, .., \lambda\} \right\} \quad \text{et} \quad \Sigma_\lambda \stackrel{\text{def}}{=} \left\{ -1, 1 \right\}^\lambda . \tag{7.1}$$

Autrement dit, $Q(\mathbf{i}, \boldsymbol{\sigma})$ correspond au poids associé au votant comprimé donné par la ***fonction de reconstruction*** $\mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma})$, c'est-à-dire le votant comprimé dont l'ensemble de compression est $\mathbf{i}\langle i_1, i_2, \ldots, i_{|\mathbf{i}|} \rangle \in I_\lambda$ et le message est $\boldsymbol{\sigma} = \langle \sigma_1, \sigma_2, \ldots, \sigma_\lambda \rangle \in \Sigma_\lambda$. La distribution *a priori* P est, elle aussi, une distribution sur $I_\lambda \times \Sigma_\lambda$. De cette manière, la distribution P peut être définie avant de «voir» l'échantillon d'entraînement S. L'ensemble des votants est seulement construit lorsque S nous est donné. Ce dernier correspond à

$$\mathcal{H}^\mathcal{R}_{S,\lambda} \stackrel{\text{def}}{=} \left\{ \mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma}) \mid \mathbf{i} \in I_\lambda, \boldsymbol{\sigma} \in \Sigma_\lambda \right\} .$$

Ainsi, étant donné un échantillon d'entraînement S, une fonction de reconstruction $\mathcal{R}(\cdot, \cdot)$ et une distribution Q sur $I_\lambda \times \Sigma_\lambda$, le ***classificateur de Bayes*** – initialement exprimé, pour le cas de votants «non comprimés», par l'équation 3.1 (page 43) – correspond à

$$B_{Q,S}(x) \stackrel{\text{def}}{=} \operatorname{sgn} \left[\mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim Q} \mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma}) \right] .$$

Nous définissons aussi le ***risque de Bayes*** $R_{D'}(B_{Q,S})$ et le ***risque de Gibbs*** $R_{D'}(G_{Q,S})$ – voir définitions 3.1 (page 44) et 3.2 (page 45) – associés à une distribution Q sur $I_\lambda \times \Sigma_\lambda$ tel que

$$\begin{aligned} R_{D'}(B_{Q,S}) &\stackrel{\text{def}}{=} \mathbb{E}^{\mathcal{L}_{01}}_{D'}(B_{Q,S}) , \\ R_{D'}(G_{Q,S}) &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim Q} \mathbb{E}^{\mathcal{L}_\ell}_{D'}(\mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma})) , \end{aligned} \tag{7.2}$$

où D' représente une distribution de données.

7.2.2 Théorème PAC-bayésien pour espérance de votants comprimés

Une étape cruciale de la démonstration des théorèmes PAC-bayésiens dans le contexte de l'apprentissage par compression d'échantillons réside dans l'observation suivante :

> Alors que le risque empirique d'un votant $h(\cdot)$ indépendant des données s'exprime par une variable aléatoire suivant une ***loi binomiale*** de m essais avec probabilité de succès $R_D(h)$ – voir la section 4.2.1 (page 59) –, le risque du votant comprimé $h^{\boldsymbol{\sigma}}_\mathbf{i}(\cdot) \coloneqq \mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma})$ est biaisé sur les exemples de $S_\mathbf{i}$.

Cependant, le nombre d'erreurs $(m - |\mathbf{i}|) \cdot R_{S \setminus S_\mathbf{i}}(h^{\boldsymbol{\sigma}}_\mathbf{i})$ sur les données qui n'appartiennent pas à l'ensemble de compression se révèle être une variable aléatoire suivant une loi binomiale de $m - |\mathbf{i}|$ essais avec probabilité de succès $R_D(h^{\boldsymbol{\sigma}}_\mathbf{i})$:

$$\Pr_{S\sim D^m} \left(R_{S\setminus S_\mathbf{i}}(h^{\boldsymbol{\sigma}}_\mathbf{i}) = \frac{k}{m - |\mathbf{i}|} \right) = \binom{m - |\mathbf{i}|}{k} \cdot \left(R(h^{\boldsymbol{\sigma}}_\mathbf{i}) \right)^k \cdot \left(1 - R(h^{\boldsymbol{\sigma}}_\mathbf{i}) \right)^{m - |\mathbf{i}| - k} . \tag{7.3}$$

Cette observation mène au lemme 7.6 ci-dessous, qui regroupe les étapes importantes qui différencient la théorie PAC-bayésienne pour votants comprimés de la théorie «classique». Lorsque nous désirerons énoncer un théorème PAC-bayésien en présence de votants comprimés, le rôle du lemme 7.6 sera essentiellement de remplacer l'étape de la démonstration du théorème 4.4 (page 61) correspondant à l'équation (4.5).

Par souci de simplicité, le résultat suivant ne couvre que la perte linéaire et la Δ-fonction propre au théorème PAC-bayésien de McAllester (1999, 2003a), c'est-à-dire

$$\Delta(q, p) \;\coloneqq\; 2(q-p)^2 \,.$$

Lemme 7.6. *Soit $\mathcal{R}(\cdot,\cdot)$ une fonction de reconstruction qui produit des votants comprimés dont la taille de l'ensemble de compression est au plus λ (avec $\lambda < m$). Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$ et pour toute distribution* a priori *P sur $I_\lambda \times \Sigma_\lambda$, on a*

$$\mathop{\mathbf{E}}_{S\sim D^m} \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P} e^{(m-\lambda)\cdot 2\left(\mathbb{E}_S^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))-\mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)^2} \;\;\leq\;\; e^{4\,\lambda}\cdot 2\sqrt{m-\lambda}\,,$$

Démonstration. Comme le choix de $(\mathbf{i},\boldsymbol{\sigma})$ selon la distribution *a priori* P est indépendant du choix de S, on a

$$\begin{aligned}
&\mathop{\mathbf{E}}_{S\sim D^m} \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P} e^{(m-\lambda)\cdot 2\cdot\left(\mathbb{E}_S^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))-\mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)^2} \\
&\qquad = \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P} \mathop{\mathbf{E}}_{S\sim D^m} e^{(m-\lambda)\cdot 2\cdot\left(\mathbb{E}_S^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))-\mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)^2} \\
&\qquad = \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P} \mathop{\mathbf{E}}_{S_{\mathbf{i}}\sim D^\lambda} \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{m-\lambda}} e^{(m-\lambda)\cdot 2\cdot\left(\mathbb{E}_S^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))-\mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)^2}. \qquad (7.4)
\end{aligned}$$

Réécrivons maintenant la perte empirique d'un votant comprimé comme une combinaison de sa perte sur l'ensemble de compression $S_{\mathbf{i}}$ et sa perte sur les autres exemples $S_{\mathbf{i}^c} \stackrel{\text{def}}{=} S \setminus S_{\mathbf{i}}$:

$$\mathbb{E}_S^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) \;\;=\;\; \frac{1}{m}\left[\lambda\cdot\mathbb{E}_{S_{\mathbf{i}}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) + (m-\lambda)\cdot\mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right].$$

Puisque $0 \leq \mathbb{E}_{D'}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) \leq 1$, on obtient, par une succession de manipulations algébriques,

$$\begin{aligned}
&(m-\lambda)\cdot 2\cdot\left(\mathbb{E}_S^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)^2 \\
&\quad= (m-\lambda)\cdot 2\cdot\left(\tfrac{1}{m}\left[\lambda\cdot\mathbb{E}_{S_{\mathbf{i}}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) + (m-\lambda)\cdot\mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right] - \mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)^2 \\
&\quad= (m-\lambda)\cdot 2\cdot\left(\tfrac{\lambda}{m}\left[\mathbb{E}_{S_{\mathbf{i}}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right] + \left[\mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right]\right)^2 \\
&\quad= (m-\lambda)\cdot 2\cdot\Big(\left(\tfrac{\lambda}{m}\right)^2\left[\mathbb{E}_{S_{\mathbf{i}}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right]^2 + \left[\mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right]^2 \\
&\qquad\qquad + \tfrac{2\lambda}{m}\left[\mathbb{E}_{S_{\mathbf{i}}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right]\left[\mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right]\Big) \\
&\quad\leq (m-\lambda)\cdot 2\cdot\left(\left(\tfrac{\lambda}{m}\right)^2 + \left[\mathbb{E}_{S_{\mathbf{i}^c}}^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right]^2 + \tfrac{2\lambda}{m}\right)
\end{aligned}$$

$$
\begin{aligned}
&= 2\lambda \cdot \Big(2 - \tfrac{\lambda}{m} - \big(\tfrac{\lambda}{m}\big)^2\Big) + (m-\lambda)\cdot 2 \cdot \big[\mathbb{E}^{\mathcal{L}_\ell}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}^{\mathcal{L}_\ell}_{D}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\big]^2 \\
&\leq 4\lambda + (m-\lambda)\cdot 2 \cdot \big[\mathbb{E}^{\mathcal{L}_\ell}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}^{\mathcal{L}_\ell}_{D}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\big]^2 \\
&\leq 4\lambda + (m-\lambda)\cdot \mathrm{kl}\big(\mathbb{E}^{\mathcal{L}_\ell}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})),\, \mathbb{E}^{\mathcal{L}_\ell}_{D}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\big)\,, \qquad (7.5)
\end{aligned}
$$

où la dernière inégalité découle de l'inégalité de Pinsker, c'est-à-dire $2\,(q-p)^2 \leq \mathrm{kl}(q,p)$.

On remarque à la ligne (7.5) que $\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})$ ne dépend pas des exemples contenus dans $S_{\mathbf{i}^c}$. Ainsi, du point de vue de $S_{\mathbf{i}^c}$, le votant $\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})$ n'est pas un votant comprimé. Considérant cela, on peut appliquer le lemme 4.1 (page 59), en remplaçant $S \sim D^m$ par $S_{\mathbf{i}^c} \sim D^{m-\lambda}$, et f par $\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})$. En combinant cela avec les résultats intermédiaires des équations (7.4) et (7.5), on obtient

$$
\begin{aligned}
&\mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}}\sim D^\lambda}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{m-\lambda}} e^{(m-\lambda)\cdot 2\cdot\left(\mathbb{E}^{\mathcal{L}_\ell}_{S}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})) - \mathbb{E}^{\mathcal{L}_\ell}_{D}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)^2} \\
\leq\ & e^{4\lambda}\cdot \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}}\sim D^\lambda}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{m-\lambda}} e^{(m-\lambda)\cdot \mathrm{kl}\left(\mathbb{E}^{\mathcal{L}_\ell}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})),\, \mathbb{E}^{\mathcal{L}_\ell}_{D}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)} && \langle\ \textit{Équation (7.5)}\ \rangle \\
\leq\ & e^{4\lambda}\cdot \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}}\sim D^\lambda} \sum_{k=0}^{m-\lambda} \binom{m-\lambda}{k} \Big(\mathbb{E}^{\mathcal{L}_\ell}_{D}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\Big)^k \Big(1-\mathbb{E}^{\mathcal{L}_\ell}_{D}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\Big)^{m-\lambda-k} && \langle\ \textit{Lemme 4.1}\ \rangle \\
&\qquad\qquad \times e^{(m-\lambda)\cdot \mathrm{kl}\left(\mathbb{E}^{\mathcal{L}_\ell}_{S_{\mathbf{i}^c}}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma})),\, \mathbb{E}^{\mathcal{L}_\ell}_{D}(\mathcal{R}(S_{\mathbf{i}},\boldsymbol{\sigma}))\right)} \\
\leq\ & e^{4\lambda}\cdot \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}}\sim D^\lambda} \sup_{r\in[0,1]} \left[\sum_{k=0}^{m-\lambda} \binom{m-\lambda}{k} r^k (1-r)^{m-\lambda-k} e^{(m-\lambda)\cdot \mathrm{kl}\left(\frac{k}{m-\lambda}, r\right)}\right] \\
=\ & e^{4\lambda}\cdot \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim P}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}}\sim D^\lambda} \mathcal{I}_{\mathrm{kl}}(m-\lambda) && \langle\ \textit{Équation (4.3), page 61, avec } m \coloneqq m-\lambda \textit{ et } \Delta \coloneqq \mathrm{kl}\ \rangle \\
=\ & e^{4\lambda}\cdot \mathcal{I}_{\mathrm{kl}}(m-\lambda) \\
\leq\ & e^{4\lambda}\cdot 2\sqrt{m-\lambda}\,. && \langle\ \textit{Équation (4.9), page 64, avec } m \coloneqq m-\lambda\ \rangle
\end{aligned}
$$

Rappelons que la dernière inégalité, qui nous permet d'obtenir le résultat désiré, provient de Maurer (2004). □

À partir du lemme 7.6, il devient facile de formuler un théorème PAC-bayésien qui borne le risque de Gibbs dans un contexte de votants comprimés. Le théorème 7.7 suivant est donc une généralisation du résultat de McAllester (1999, 2003a), que nous avons présenté par le corollaire 4.8(b) (page 68). En effet, nous retrouvons le résultat original lorsque $\lambda \coloneqq 0$.

Théorème 7.7. *Soit $\mathcal{R}(\cdot,\cdot)$ une fonction de reconstruction qui produit des votants comprimés dont la taille de l'ensemble de compression est au plus λ (avec $\lambda < m$). Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$ et pour toute distribution* a priori *P sur $I_\lambda \times \Sigma_\lambda$, et pour tout $\delta \in (0,1]$, on a, avec probabilité au moins $1-\delta$ sur le choix de $S \sim D^m$,*

$$
\forall\, Q \text{ sur } I_\lambda \times \Sigma_\lambda : \quad R_D(G_{Q,S}) \leq R_S(G_{Q,S}) + \sqrt{\frac{1}{2(m-\lambda)}\left[\mathrm{KL}(Q\|P) + 4\lambda + \ln\frac{2\sqrt{m-\lambda}}{\delta}\right]}\,.
$$

Démonstration. Nous suivons les mêmes étapes que lors de la démonstration du théorème 4.4 (page 61), avec $m \coloneqq m - \lambda$, $f \coloneqq \mathcal{R}(S_{\mathbf{i}}, \boldsymbol{\sigma})$, $\mathcal{L} \coloneqq \mathcal{L}_\ell$ et $\Delta(q,p) = 2(q-p)^2$. À la fin de la démonstration, nous remplaçons l'équation (4.5) par le résultat du lemme 7.6. Nous obtenons ainsi, au lieu de l'équation (4.2),

$$\forall\, Q \text{ sur } I_\lambda \times \Sigma_\lambda : \; 2\Big(\mathbb{E}_S^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}}, \boldsymbol{\sigma})) - \mathbb{E}_D^{\mathcal{L}_\ell}(\mathcal{R}(S_{\mathbf{i}}, \boldsymbol{\sigma}))\Big)^2 \leq \frac{1}{m-\lambda}\left[\mathrm{KL}(Q\|P) + \ln \frac{e^{4\,\lambda} \cdot 2\sqrt{m-\lambda}}{\delta}\right],$$

et le résultat est obtenu par une simple réorganisation des termes, en utilisant la définition du risque de Gibbs de l'équation (7.2). □

7.2.3 Vote de majorité de fonctions de similarités

Nous allons maintenant montrer que la théorie présentée par les sous-sections 7.2.1 et 7.2.2 permet d'exprimer un classificateur linéaire comme un vote de majorité de votants comprimés. Nous verrons ensuite que ce résultat est en fait plus général, et permet de remplacer le noyau des classificateurs linéaires par une mesure de similarité quelconque.

Considérons d'abord une ***fonction de noyau*** $k : \mathcal{X} \times \mathcal{X} \to [-1, 1]$ et un échantillon d'entraînement $S \coloneqq \{(x_i, y_i)\}_{i=1}^m$. Comme présenté par la section 2.2 – voir équation (2.13), page 24 –, on exprime un classificateur linéaire dans l'espace *induit* par le noyau à l'aide d'un vecteur de ***variables duales*** $\boldsymbol{\alpha} \in \mathbb{R}^n$. On obtient ainsi un classificateur $h_{\boldsymbol{\alpha}}(\cdot)$ dont la fonction de classification est

$$h_{\boldsymbol{\alpha}}(x) \;\stackrel{\text{def}}{=}\; \mathrm{sgn}\left[\sum_{i=1}^m \alpha_i k(x_i, x)\right]. \tag{7.6}$$

Il est facile de réécrire l'équation (7.6) comme un ***vote de majorité***. Pour ce faire, définissons un ensemble de $2m$ votants $\mathcal{H}_S^k$ tel que

$$\mathcal{H}_S^k \;\stackrel{\text{def}}{=}\; \Big\{k(x_1,\cdot), k(x_2,\cdot), \ldots, k(x_m,\cdot), -k(x_1,\cdot), -k(x_2,\cdot), \ldots, -k(x_m,\cdot)\Big\}, \tag{7.7}$$

et une distribution $Q_{\boldsymbol{\alpha}}$ sur $\mathcal{H}_S^k$ telle que, pour tout $i \in \{1, \ldots, m\}$,

$$Q_{\boldsymbol{\alpha}}\Big(\, k(x_i,\cdot)\,\Big) \coloneqq \begin{cases} \dfrac{\alpha_i}{\sum_{j=1}^m \alpha_j} & \text{si } \alpha_i \geq 0 \\ 0 & \text{sinon,} \end{cases} \qquad \text{et} \qquad Q_{\boldsymbol{\alpha}}\Big(\, -k(x_i,\cdot)\,\Big) \coloneqq \begin{cases} 0 & \text{si } \alpha_i \geq 0 \\ \dfrac{\alpha_i}{\sum_{j=1}^m \alpha_j} & \text{sinon.} \end{cases}$$

Nous obtenons un vote de majorité dont la fonction de classification est équivalente à celle du classificateur de l'équation (7.6), car

$$\begin{aligned} B_{Q_{\boldsymbol{\alpha}}}(x) &= \mathrm{sgn}\left[\mathop{\mathbf{E}}_{h\sim Q_{\boldsymbol{\alpha}}} h(x)\right] \\ &= \mathrm{sgn}\left[\sum_{i=1}^m \Big[Q_{\boldsymbol{\alpha}}(k(x_i,\cdot)) - Q_{\boldsymbol{\alpha}}(-k(x_i,\cdot))\Big] k(x_i, x)\right] \\ &= \mathrm{sgn}\left[\sum_{i=1}^m \alpha_i k(x_i, x)\right] \\ &= h_{\boldsymbol{\alpha}}(x)\,. \end{aligned}$$

Nous ne pouvons pas utiliser les théorèmes PAC-bayésiens démontrés dans les chapitres précédents pour borner le risque de ce vote de majorité. En effet, les votants ne sont pas déterminés *a priori*. Autrement dit, chacun des votants $k(x_i, \cdot) \in \mathcal{H}_S^k$ est en fait un votant comprimé dont l'ensemble de compression (de taille 1) est $\{x_i\} = S_{\langle i \rangle} \subseteq S$.

Considérons la théorie de la compression d'échantillons de la section 7.2.1. Par l'équation (7.1), dans le cas particulier où les votants possède un ensemble de compression de taille 1, on a

$$I_1 \,=\, \Big\{\langle i_1\rangle, \langle i_2\rangle, \ldots, \langle i_m\rangle\Big\} \quad \text{et} \quad \Sigma_1 \,=\, \Big\{-1, 1\Big\}\,.$$

Nous adoptons la fonction de reconstruction suivante. Étant donné un exemple $x_i \in S$ formant un ensemble de compression et un message $\sigma \in \{-1, 1\}$, on a

$$\mathcal{R}\left(\{x_i\}, \{\sigma\}\right) \stackrel{\text{def}}{=} \sigma\, k(x_i, \cdot)\,. \tag{7.8}$$

Cette fonction de reconstruction permet d'exprimer tous les votants comprimés de l'ensemble $\mathcal{H}_S^k$ défini par l'équation (7.7). Nous pouvons donc mettre à profit le théorème 7.7 pour borner le risque du classificateur de Gibbs exprimé par la distribution $Q_{\boldsymbol{\alpha}}$ sur $\mathcal{H}_S^k$. De ce résultat, nous bornons le risque de Bayes par le biais de la ***borne du facteur deux*** (proposition 3.6, page 151), comme le montre le corollaire 7.8 ci-bas.

Corollaire 7.8. *Soit $k : \mathcal{X} \times \mathcal{X} \to [-1, 1]$ une fonction de similarité. Pour toute distribution D sur $\mathcal{X} \times \{-1, 1\}$ et pour toute distribution* a priori *$P_{\boldsymbol{\alpha}}$ sur $I_1 \times \Sigma_1$, et tout $\delta \in (0, 1]$, on a, avec probabilité au moins $1 - \delta$ sur le choix de $S \sim D^m$,*

$$\forall\, Q_{\boldsymbol{\alpha}} \text{ sur } I_1 \times \Sigma_1 : \quad R_D(B_{Q_{\boldsymbol{\alpha}}}) \,\leq\, 2\, R_S(G_{Q_{\boldsymbol{\alpha}}}) + \sqrt{\frac{2}{m-1}\left[\mathrm{KL}(Q_{\boldsymbol{\alpha}} \| P_{\boldsymbol{\alpha}}) + 4 + \ln \frac{2\sqrt{m-1}}{\delta}\right]}\,.$$

Démonstration. Le résultat est une simple application du théorème 7.7, avec $\lambda \coloneqq 1$ et la fonction de reconstruction de l'équation (7.8), ainsi que de la *borne du facteur deux* (proposition 3.6). □

Puisque les classificateurs $B_{Q_{\boldsymbol{\alpha}}}(\cdot)$ et $h_{\boldsymbol{\alpha}}(\cdot)$ sont équivalents, et donc $R_D(B_{Q_{\boldsymbol{\alpha}}}) = R_D(h_{\boldsymbol{\alpha}})$, le corollaire 7.8 borne le risque d'un classificateur linéaire de la forme donnée par l'équation (7.6). Cependant, la fonction $k(\cdot, \cdot)$ n'est pas limitée ici à être un ***noyau semi-défini positif*** – voir l'équation (2.15), page 24 –, mais peut-être n'importe quelle *fonction de similarité* entre deux exemples. Ainsi, les algorithmes d'apprentissage qui sont conçus à partir de cette théorie sont davantage polyvalents que les algorithmes à noyaux «traditionnels», tel le SVM. C'est le cas de l'algorithme *PBSC* (Germain et al., 2011; Shanian, 2012, voir aussi la section C.2.3 en annexe) et l'algorithme *MinCq* (Roy et al., 2011; Germain et al., 2015b).

7.2.4 Vers l'apprentissage des fonctions de similarités

La théorie PAC-bayésienne présentée dans l'article Germain et al. (2011) – reproduit à l'annexe C – possède un niveau de complexité plus élevé que celle présentée au cours des pages précédentes. Toutefois, cette approche plus générale ouvre la voie à la conception d'algorithmes qui *apprennent* la fonction de similarité, au lieu de simplement utiliser une fonction $k(\cdot,\cdot)$ définie *a priori*. Nous n'avons pas exploré cette avenue de recherche au cours de la thèse, mais nous croyons pertinent de donner un aperçu des possibilités qu'elle offre.

Ensemble continu de votants comprimés

Considérons une fonction de similarité $k : \mathcal{X} \times \mathcal{X} \to [-1, 1]$ et un échantillon d'entraînement $S \coloneqq \{(x_i, y_i)\}_{i=1}^m$. Définissons un vote de majorité dont l'ensemble de votants $\overline{\mathcal{H}_S^k}$ contient une *infinité* de votants comprimés. Chacun de ces votants $h_{\langle i \rangle}^{(\sigma,s)} \in \overline{\mathcal{H}_S^k}$ est caractérisé par un ensemble de compression $S_{\langle i \rangle} = \{x_i\} \subseteq S$ de taille 1 et un message (σ, s) constitué d'un nombre réel $\sigma \in [0, 1]$ et d'un signe $s \in \{+, -\}$. Autrement dit,

$$\overline{\mathcal{H}_S^k} \stackrel{\text{def}}{=} \left\{ h_{\langle i \rangle}^{(\sigma,s)}(\cdot) \;\middle|\; \langle i \rangle \in \{\langle 1 \rangle, \langle 2 \rangle, \ldots, \langle m \rangle\} \text{ et } (\sigma, s) \in [0, 1] \times \{+, -\} \right\}. \tag{7.9}$$

Un votant $h_{\langle i \rangle}^{(\sigma,s)} \in \overline{\mathcal{H}_S^k}$ est régi par la fonction de classification suivante. Étant donné un exemple $x \in \mathcal{X}$, on a

$$h_{\langle i \rangle}^{(\sigma,+)}(x) \stackrel{\text{def}}{=} \begin{cases} +1 \text{ si } (2\sigma - 1) < k(x_i, x)\,, \\ -1 \text{ sinon.} \end{cases} \tag{7.10}$$

Remarquons que le signe ($+$ et $-$) du message caractérise une paire de classificateurs complémentaires, puisque $h_{\langle i \rangle}^{(\sigma,+)} = -h_{\langle i \rangle}^{(\sigma,-)}$. La fonction de reconstruction $\mathcal{R}(\cdot,\cdot)$ permettant de retrouver chacun des votants comprimés à partir des valeurs de i, σ et s est définie ainsi :

$$\begin{aligned} \mathcal{R}\left(\{x_i\}, (\sigma, +)\right) &\stackrel{\text{def}}{=} h_{\langle i \rangle}^{(\sigma,+)} \\ \mathcal{R}\left(\{x_i\}, (\sigma, -)\right) &\stackrel{\text{def}}{=} h_{\langle i \rangle}^{(\sigma,-)} = -h_{\langle i \rangle}^{(\sigma,+)}\,, \end{aligned}$$

Distributions fortement alignées.

Voyons d'abord comment, en considérant un vote de majorité dont la distribution (continue) Q sur $\overline{\mathcal{H}_S^k}$ est contrainte à être *fortement alignée* (définition 7.9 ci-dessous), nous pouvons retrouver une fonction de classification habituellement associée aux classificateurs à noyaux (c'est-à-dire respectant l'équation (7.6) de la sous-section précédente).

Définition 7.9. La distribution Q est *fortement alignée* sur une distribution P lorsque, en plus d'être *alignée* sur P (voir la définition 7.2, page 145), la distribution Q possède la caractéristique suivante : pour chaque ensemble de compression $\{x_i\}$ et signe s, la probabilité de choisir un votant selon Q est uniforme pour toute valeur de σ.

Lorsqu'une distribution Q est fortement alignée, il possible de la caractériser par seulement $2m$ valeurs, que l'on désigne par $Q_1, \ldots, Q_{2m}$:

$$\begin{aligned} Q_i &\stackrel{\text{def}}{=} Q(h_{\langle i \rangle}^{(\sigma,+)}) \,, \\ Q_{m+i} &\stackrel{\text{def}}{=} Q(h_{\langle i \rangle}^{(\sigma,-)}) \,, \end{aligned} \qquad \forall\, i \in \{1, \ldots, m\},\, \sigma \in [0,1] \,.$$

De manière équivalente, on a

$$\begin{aligned} Q_i &= \int_0^1 Q(h_{\langle i \rangle}^{(\sigma,+)})\, d\sigma \,, \\ Q_{m+i} &= \int_0^1 Q(h_{\langle i \rangle}^{(\sigma,-)})\, d\sigma \,, \end{aligned} \qquad \forall\, i \in \{1, \ldots, m\} \,.$$

Similairement ce que nous avons fait à la section 7.2.3, nous transformons le vote de majorité en classificateur caractérisé par une combinaison linéaire de mesures de similarité par l'identité suivante :

$$\alpha_i \;\coloneqq\; Q_i - Q_{m+i} \quad \text{pour } i \in \{1, \ldots, m\} \,.$$

Ainsi, le classificateur de Bayes associé à une distribution Q fortement alignée est équivalent au classificateur linéaire exprimé par le vecteur $\boldsymbol{\alpha} \in \mathbb{R}^m$, c'est-à-dire

$$B_Q(x) \;=\; \operatorname{sgn}\left(\mathop{\mathbf{E}}_{h \sim Q} h(x) \right) \;=\; \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i k(x_i, x) \right) ,$$

puisque

$$\begin{aligned} \mathop{\mathbf{E}}_{h \sim Q} h(x) &= \sum_{i=1}^{m} \int_0^1 \left[Q\left(h_{\langle i \rangle}^{(\sigma,+)}\right) \cdot h_{\langle i \rangle}^{(\sigma,+)}(x) + Q\left(h_{\langle i \rangle}^{(\sigma,-)}\right) \cdot h_{\langle i \rangle}^{(\sigma,-)}(x) \right] d\sigma \\ &= \sum_{i=1}^{m} \int_0^1 \left[Q_i - Q_{m+i} \right] \cdot h_{\langle i \rangle}^{(\sigma,+)}(x)\, d\sigma \\ &= \sum_{i=1}^{m} \alpha_i \int_0^1 h_{\langle i \rangle}^{(\sigma,+)}(x)\, d\sigma \\ &= \sum_{i=1}^{m} \alpha_i \left[\int_0^{\frac{1}{2}k(x_i,x)+\frac{1}{2}} (+1)\, d\sigma + \int_{\frac{1}{2}k(x_i,x)+\frac{1}{2}}^{1} (-1)\, d\sigma \right] \\ &= \sum_{i=1}^{m} \alpha_i k(x_i, x) \,. \end{aligned}$$

Nous ne présentons pas ici les théorèmes PAC-bayésiens sur le risque des classificateurs linéaires obtenus à l'aide des distributions fortement alignées. Le lecteur est invité à consulter l'annexe C, où cette théorie est abordée en détail. Notons que cette approche peut sembler inutilement compliquée si le seul objectif est de retrouver un résultat semblable à celui présenté, par le biais d'une théorie plus simple, à la sous-section 7.2.3. Cependant, comme nous l'expliquons dans la suite du texte, l'utilisation de messages continus ouvre la voie à d'autres possibilités.

Distributions non fortement alignées

Le fait de considérer seulement des votes de majorité dont la distribution Q est *fortement alignée* sur une distribution P (définition 7.9) est une autre contrainte nécessaire à l'algorithme d'apprentissage *PBSC* de l'article Germain et al. (2011) de l'annexe C. Cependant, cette contrainte n'est pas imposée par le théorème sur par le théorème PAC-bayésien sur lequel l'algorithme repose. [2] Ce théorème est valide pour toute distribution Q sur l'ensemble continu de votants $\overline{\mathcal{H}_S^k}$ donnée par l'équation (7.9).

Permettre à un algorithme d'apprentissage de considérer une classe plus large de distributions Q sur $\overline{\mathcal{H}_S^k}$ permettrait de choisir le vote de majorité $B_Q(\cdot)$ parmi une classe beaucoup plus grande de classificateurs. Un tel algorithme posséderait, en quelque sorte, le pouvoir d'apprendre la fonction de similarité selon les exemples d'apprentissage. Pour comprendre cela, il faut voir que l'équation (7.10), définissant la fonction de classification de chacun des votants de $\overline{\mathcal{H}_S^k}$, repose notamment sur le message $\sigma \in [0, 1]$. En ne contraignant pas Q à être une distribution fortement alignée, il serait possible de pondérer la fonction $k(\cdot, \cdot)$ d'une multitude de manières.

Cela dit, afin qu'une telle approche demeure «algorithmiquement viable», il faut parvenir à exprimer la distribution Q par un nombre relativement restreint de variables. De plus, alors que le problème d'optimisation résolu par l'algorithme *PBSC* est de nature quadratique, il est risqué d'obtenir des fonctions objectives difficiles à minimiser en considérant des distributions non fortement alignées sur l'ensemble de votants $\overline{\mathcal{H}_S^k}$. En particulier, le problème d'optimisation risque d'être fortement non convexe.

En résumé, il s'agit d'une avenue de recherche riche en possibilités, mais qui s'annonce ardue.

7.3 Étude empirique de l'efficacité de l'algorithme SCM

L'algorithme ***SCM*** (Marchand et Shawe-Taylor, 2002) – détaillé à la section 2.3.2 (page 29) – est un des rares algorithmes d'apprentissage motivé par la théorie de la compression d'échantillons. Il s'agit d'une stratégie gloutonne visant à construire une conjonction (ou une disjonc-

2. Voir plus spécifiquement le théorème C.1 (page 181, en annexe), qui justifie la première version de l'algorithme.

tion) de classificateurs qui tend à minimiser la borne sur le risque présentée par le théorème 2.3 (page 28).

Le SCM ne trouve pas nécessairement le minimum absolu du problème d'optimisation donné par la borne, car ce problème combinatoire est NP-complet. Bien que les travaux de Marchand et Shawe-Taylor (2002) ont montré que cet algorithme d'apprentissage produit des classificateurs de faible risque sur des échantillons de données réelles, ils ne donnent aucune indication si la solution trouvée par la stratégie gloutonne est, en pratique, une bonne approximation de la solution optimale.

Afin d'explorer cette question, nous avons formulé le problème de la minimisation de la borne du théorème 2.3, spécialisé aux conjonctions de *boules dépendantes des données* (comme dans les expérimentations de Marchand et Shawe-Taylor, 2002), comme un problème d'optimisation pseudo-booléen. En utilisant des solveurs pseudo-booléens modernes, il nous a été possible de trouver le classificateur (autrement dit, la conjonction de boules) qui minimise réellement la borne qui a inspiré le SCM. Ces expérimentations ont été menées à l'aide de plusieurs échantillons de données réelles. Elles permettent de conclure que l'heuristique de recherche employée par l'algorithme du SCM, bien qu'il ne minimise pas directement l'expression de la borne exprimée par le théorème 2.3, trouve presque toujours une solution près du minimum suggéré par la borne. Il s'agit d'un résultat surprenant, puisque l'algorithme (glouton) du SCM est très rapide d'exécution.

L'annexe D (page 201) présente un article consacré à cette étude, qui fut coécrit au cours de la thèse (Germain et al., 2012a). Il s'agit de résultats empiriques qui viennent renforcer l'idée que l'algorithme SCM (en particulier) et la théorie de la compression d'échantillons (en général) sont des approches qui conviennent bien à la résolution de problèmes en apprentissage automatique.

7.4 Apprentissage des représentations et adaptation de domaine

La théorie PAC-bayésienne pour l'adaptation de domaine présentée au chapitre 6 est inspirée des travaux de Ben-David et al. (2006, 2010). Plus précisément, nous avons suggéré une mesure de ***divergence entre les distributions*** source et cible, inspirée de la $\mathcal{H}\Delta\mathcal{H}$-distance de Ben-David et al., qui prend en compte une pondération des votants. Nous avons ainsi formulé de nouvelles garanties pour le cadre de l'adaptation de domaine.

Parallèlement à la réalisation des travaux présentés au chapitre 6, nous avons proposé une autre approche «non PAC-Bayésienne», dont fait état l'article présenté par l'annexe E. Cette approche est directement motivée par l'observation suivante :

La théorie de Ben-David et al. (2006, 2010) affirme qu'une famille de classificateurs $\mathcal{H}$ permet de s'adapter au domaine cible si elle :

1. permet de bien classifier le domaine source ; et
2. n'est pas capable de distinguer les exemples sources des exemples cibles.

Notons que cette observation apparaît de manière claire lorsque nous étudions la $\mathcal{H}$-divergence de Ben-David et al. (2006, 2010), qui est un concept distinct, mais semblable, à la $\mathcal{H}\Delta\mathcal{H}$-distance présentée au chapitre 6. Le lecteur est invité à se référer à la section E.2 (page 212) pour obtenir plus de détails.

Nous avons développé un algorithme qui «apprend» une nouvelle *représentation* des exemples appartenant aux échantillons sources et cibles pour laquelle la famille de classificateurs $\mathcal{H}$ possède les caractéristiques (1) et (2). Pour ce faire, nous intégrons à un algorithme d'apprentissage du type ***réseau de neurones*** un nouveau type de régularisateur. Ce dernier contraint la représentation interne du réseau de neurones de manière à ce qu'il soit très difficile d'y distinguer les exemples sources des exemples cibles.

Précisons que les réseaux de neurones sont des algorithmes qui siéent particulièrement bien à cette tâche, puisqu'ils sont renommés pour leur capacité à apprendre de nouvelles représentations des exemples favorisant la résolution d'un problème d'apprentissage. Notre algorithme d'apprentissage permet d'améliorer certains résultats empiriques constituant l'état de l'art en adaptation de domaine.

Chapitre 8

Conclusion et travaux futurs

Les contributions principales de cette thèse à la théorie PAC-bayésienne se catégorisent en trois cadres d'apprentissage distincts. Nous récapitulons ces contributions à la section 8.1. Nous discutons ensuite de travaux futurs que nous envisageons quant aux cadres de l'apprentissage transductif (section 8.2) et de l'adaptation de domaine (section 8.3). Nous élargissons finalement la discussion à d'autres cadres d'apprentissage (section 8.4).

8.1 Bref retour sur les contributions de la thèse

Nous avons élaboré, dans la partie II de la thèse, une conceptualisation originale de la théorie PAC-bayésienne pour le cadre d'apprentissage inductif, auquel s'applique initialement cette théorie statistique de l'apprentissage. Nous avons notamment fait ressortir l'importance de la notion d'espérance de désaccord entre les votants pour l'étude des votes de majorité, en particulier pour énoncer la $\mathcal{C}$-borne. Nous avons aussi montré que plusieurs résultats contenus dans la littérature PAC-bayésienne peuvent être retrouvés à l'aide d'une approche générale. C'est en prenant appui sur cette compréhension approfondie des fondements de la théorie PAC-bayésienne que nous avons pu, dans la partie III de la thèse, l'adapter à d'autres cadres d'apprentissage.

Au chapitre 5, nous avons présenté une analyse du cadre d'apprentissage transductif. Par une approche rigoureuse, nous avons développé une nouvelle borne sur le risque dans ce contexte. Notre résultat améliore substantiellement ceux présents dans la littérature, particulièrement lorsqu'on prend en considération l'espérance de désaccord par le biais de la $\mathcal{C}$-borne. Une caractéristique qui distingue le théorème PAC-bayésien général transductif de son pendant inductif est la possibilité d'utiliser facilement toute fonction convexe $\Delta : [0,1] \times [0,1] \to \mathbb{R}$ (nommée Δ-*fonction* dans la thèse) comme mesure de «distance» entre les risques.

Au chapitre 6, nous avons produit une première analyse PAC-bayésienne du cadre de l'adaptation de domaine. Notre analyse repose sur la définition d'une mesure de désaccord entre les

distributions source et cible qui dépend de la pondération des votants. Cette mesure a permis de démontrer un théorème PAC-Bayésien pour l'adaptation de domaine, menant ensuite à la conception d'un algorithme d'apprentissage. Grâce à l'astuce du noyau, notre algorithme produit un classificateur linaire dans d'espace d'entré ou dans un espace «augmenté». Des expérimentations sur des données réelles montrent que cet algorithme est compétitif avec les approches existantes.

Enfin, au chapitre 7, nous avons donné un aperçu des autres contributions apportées au domaine de l'apprentissage automatique au cours du doctorat, mais ne cadrant pas dans le corps de la thèse : des bornes PAC-bayésiennes sans terme «$\mathrm{KL}(Q\|P)$», une théorie PAC-Bayésienne pour votants comprimés (dépendant des exemples de l'échantillon d'entraînement), une étude empirique de l'efficacité de l'algorithme SCM, ainsi qu'un réseau de neurones apprenant une représentation des exemples propices à l'adaptation de domaine.

8.2 Travaux futurs dans le cadre d'apprentissage transductif

Nous avons maintes fois souligné la possibilité d'utiliser facilement toute Δ-fonction au sein du théorème PAC-bayésien dédié au cadre transductif. Pour l'instant, l'impact du choix de la Δ-fonction sur les bornes transductives a principalement été étudié sur la base d'observations empiriques. En effet, à la section 5.5.1 (page 108), nous avons observé l'effet de ce choix selon le contexte (en particulier, selon le ratio m/N d'exemples étiquetés contenus dans l'échantillon complet). Il serait pertinent d'approfondir cette étude pour cerner plus précisément les impacts du recours aux différentes Δ-fonctions, notamment par une analyse statistique.

Aussi, nous n'avons pas exploité le théorème PAC-bayésien comme source d'inspiration pour la conception d'algorithmes d'apprentissage œuvrant dans le cadre transductif, contrairement à ce qui a été fait dans Germain (2009) pour le cadre inductif et dans le chapitre 6 de cette thèse pour l'adaptation de domaine. Du point de vue de la conception d'algorithme, le choix de Δ-fonction se révèle aussi un aspect intéressant, car chaque Δ-fonction suggère son propre problème d'optimisation.

8.3 Travaux futurs dans le cadre de l'adaptation de domaine

Les bons résultats empiriques de notre algorithme d'adaptation de domaine *PBDA* nous incitent à poursuivre les expérimentations avec d'autres types de données réelles. Nous désirons notamment le mettre à profit pour résoudre des problèmes d'apprentissage impliquant des *données massives*[1], qui suscitent un grand intérêt chez les praticiens contemporains. En effet, les algorithmes d'apprentissage automatique gérant des données massives sont déjà utilisés

1. Le terme *données massives* (ou, en anglais, *Big Data*) désigne les sources de données dont la grande taille dépasse la capacité de traitement des systèmes informatiques traditionnels.

par les systèmes de recommandation en ligne ou les moteurs de recherche des grandes compagnies. Ces applications constituent souvent des problèmes d'adaptation de domaine, car la distribution génératrice des exemples varie naturellement en fonction du temps, du lieu et du contexte d'où proviennent les données. Qui plus est, les données massives contiennent typiquement beaucoup d'exemples non étiquetés, car les exemples sont recueillis de manière automatique par des systèmes informatiques, alors que leur étiquetage requiert l'intervention humaine.

En plus de posséder de bons résultats empiriques, l'algorithme PBDA repose sur des bases théoriques solides, ce qui bénéficierait à un domaine de recherche naissant comme celui de l'apprentissage sur les données massives. Cependant, le temps d'exécution de l'algorithme empêche son utilisation sur des ensembles d'entraînement de très grande taille. Cela s'explique principalement par le fait que PBDA minimise une fonction objectif par une descente de gradient «classique» : chaque itération du processus d'optimisation repose sur le calcul du gradient à l'aide de tous les exemples d'entraînement. Afin de traiter des données massives, il convient d'employer des méthodes d'optimisations stochastiques, qui utilisent seulement un petit sous-ensemble des exemples d'entraînement à chaque itération. Pour parvenir à modifier notre algorithme en ce sens, tout en conservant les garanties théoriques qui lui sont propres, nous envisageons de nous inspirer des travaux de Bordes et al. (2005); Nesterov (2012, 2014); Bach (2014). Une stratégie complémentaire à l'optimisation stochastique qui favoriserait l'accélération de l'algorithme PBDA serait de convexifier sa fonction objectif. Toutefois, cette dernière approche nous semble ardue, car la convexification devrait préserver le compromis particulier permettant à PBDA de s'«adapter» au domaine cible.

Cela étant dit, nous ne prétendons pas que l'analyse du cadre d'adaptation de domaine présentée dans cette thèse constitue la seule approche possible découlant de la théorie PAC-bayésienne. Au contraire, il s'agit d'une première tentative – qui s'est avérée fructueuse – et il convient d'explorer d'autres possibilités. Dans un premier temps, sans nous éloigner de la théorie présentée dans la thèse, nous planifions expérimenter un nouvel algorithme d'apprentissage inspiré du théorème 6.4 (page 120), mais qui incorpore le terme $\xi_Q(D_{\mathrm{S}}, D_{\mathrm{T}})$. Rappelons que nous avons ignoré ce terme lors de la conception de l'algorithme PBDA, sous l'hypothèse qu'il possède une valeur négligeable lorsque l'adaptation est possible. Pour ce faire, le recours à la borne supérieure $\sqrt{\chi^2(D_{\mathrm{T}} \| D_{\mathrm{S}})\, e_Q^{D_{\mathrm{S}}}} \geq \xi_Q(D_{\mathrm{S}}, D_{\mathrm{T}})$ de la proposition 6.5 (page 121) semble une avenue judicieuse, car la divergence chi-carrée entre les distributions source D_{S} et cible D_{T} peut être estimée à l'aide des échantillons d'entraînement S et T.

Finalement, nous prévoyons aussi élargir notre théorie et nos algorithmes pour englober des contextes d'adaptation de domaine légèrement différents de celui étudié dans cette thèse. Entre autres, il parait assez simple de généraliser notre théorie à la situation où l'algorithme d'apprentissage a accès à plusieurs domaines sources (des résultats préliminaires en ce sens sont présentés dans Germain et al., 2015a), ou encore à la situation où l'échantillon cible

contient quelques exemples étiquetés. De plus, nous pensons étendre nos résultats au cadre des *distributions glissantes* – où la distribution génératrice des données varie tranquillement avec le temps – qui est un problème connexe à celui de l'adaptation de domaine. Pour ce faire, nous croyons pouvoir nous inspirer des travaux de Mohri et Medina (2012).

8.4 Vers l'étude de nouveaux cadres d'apprentissage

Au terme de cette thèse, nous insistons à nouveau sur le fait que les avancées dans les cadres de l'apprentissage transductif et de l'adaptation de domaine décrites en ces pages reposent sur une approche originale de la théorie PAC-bayésienne. Nous croyons que cette approche favorise la compréhension des «concepts fondamentaux» de cette théorie statistique de l'apprentissage automatique et peut servir de point de départ à l'analyse de nombreux autres cadres d'apprentissage.

De fait, plusieurs travaux parus au cours des dernières années montrent que la théorie PAC-bayésienne s'applique dans une multitude d'autres contextes que ceux abordés ici. Parmi ceux-ci, mentionnons l'*estimation de densité* (Seldin et Tishby, 2009; Higgs et Shawe-Taylor, 2010), la *prédiction structurée* (McAllester, 2007; Giguère et al., 2013; London et al., 2014), le *co-groupement* (en anglais *co-clustering* ; Seldin et Tishby, 2009, 2010), les processus stochastiques de type *martingales* (Seldin et al., 2012), les problèmes de *machines à sous à plusieurs bras* (en anglais *multi-armed bandit* ; Seldin et al., 2011) et l'apprentissage par renforcement (Fard et Pineau, 2010; Fard et al., 2011). Nous espérons que l'éclairage que nous avons apporté à la théorie PAC-bayésienne inspirera d'autres développements s'inscrivant dans ces cadres d'apprentissage et aidera à entreprendre l'analyse de nouveaux types de problèmes.

Annexe A

Résultats mathématiques

Cette annexe est divisée comme suit :

1. La section A.1 présente des résultats qui se trouvent dans plusieurs ouvrages traitant des probabilités, ou encore qui découlent directement de résultats bien connus.
2. La section A.2 contient des résultats provenant de la littérature en apprentissage automatique.
3. La section A.3 contient résultats mathématiques «originaux» auxquels nous faisons référence dans la thèse, mais qui sont relégués en annexe afin d'alléger la lecture du document principal.

A.1 Résultats provenant de la littérature (probabilités)

Notons que l'***inégalité de Markov*** (lemme A.1) et l'***inégalité de Jensen*** (lemme A.2), énoncées ci-dessus, sont entre autres démontrées dans Ross (1994) et Mohri et al. (2012).

Lemme A.1 (Inégalité de Markov)**.** *Pour toute variable aléatoire non négative X dont l'espérance est $\mu \coloneqq \mathbf{E}(X)$ et pour toute valeur réelle $a > 0$, on a*

$$\Pr\left(X > a\right) \;<\; \frac{\mu}{a}\,.$$

Lemme A.2 (Inégalité de Jensen)**.** *Soit une fonction $g : \mathbb{R}^n \to \mathbb{R}$ d'une variable aléatoire X distribuée selon P. Si g est convexe dans le domaine où P est non nul, alors*

$$\mathbf{E}\, g(X) \;\geq\; g\left(\mathbf{E}\, X\right).$$

Théorème A.3 (Inégalité de Cantelli-Tchebychev)**.** *Pour toute variable aléatoire X dont l'espérance est $\mu \coloneqq \mathbf{E}(X)$ et la variance est $\sigma^2 \coloneqq \mathbf{Var}(X)$, et pour toute valeur réelle $a > 0$, on a*

$$\Pr\Big(X - \mu \geq a\Big) \;\leq\; \frac{\sigma^2}{\sigma^2 + a^2}\,.$$

Nous incluons ci-bas une démonstration de l'***inégalité de Cantelli-Tchebychev***, car cette dernière ne se retrouve pas fréquemment dans les manuels de probabilités. Nous empruntons la méthode de démonstration de Lacasse (2010).

Démonstration. Observons d'abord que $\Pr\Big(X-\mu \geq a\Big) \leq \Pr\Big(\Big[X-\mu+\frac{\sigma^2}{a}\Big]^2 \geq \Big[a+\frac{\sigma^2}{a}\Big]^2\Big)$. Appliquons maintenant l'inégalité de Markov (lemme A.1). Afin de borner cette probabilité. On obtient

$$\begin{aligned}
\Pr\left(\left[X-\mu+\frac{\sigma^2}{a}\right]^2 \geq \left[a+\frac{\sigma^2}{a}\right]^2\right) &\leq \frac{\mathbf{E}\left[X-\mu+\frac{\sigma^2}{a}\right]^2}{\left[a+\frac{\sigma^2}{a}\right]^2} \qquad \langle\ \textit{Inégalité de Markov}\ \rangle \\
&= \frac{\mathbf{E}\,(X-\mu)^2 + 2\left(\frac{\sigma^2}{a}\right)\mathbf{E}\,(X-\mu) + \left(\frac{\sigma^2}{a}\right)^2}{\left[a+\frac{\sigma^2}{a}\right]^2} \\
&= \frac{\sigma^2+\left(\frac{\sigma^2}{a}\right)^2}{\left[a+\frac{\sigma^2}{a}\right]^2} = \frac{\sigma^2\left(1+\frac{\sigma^2}{a^2}\right)}{(\sigma^2+a^2)\left(1+\frac{\sigma^2}{a^2}\right)} = \frac{\sigma^2}{\sigma^2+a^2}\,,
\end{aligned}$$

puisque $\mathbf{E}\,(X-\mu)^2 = \mathbf{Var}(X) = \sigma^2$ et $\mathbf{E}\,(X-\mu) = \mathbf{E}(X) - \mathbf{E}(X) = 0\,.$ □

Notons que dans la démonstration du théorème A.4 ci-dessous, Cover et Thomas (1991) considèrent que Q et P sont des distributions discrètes, mais leur démonstration se généralise directement aux distributions continues.

Théorème A.4 (Cover et Thomas, 1991, théorème 2.7.2)**.** *La divergence Kullback-Leibler* $\mathrm{KL}(Q\|P)$ *est convexe en la paire* (Q,P)*. Autrement dit, si* (Q_1,P_1) *et* (Q_2,P_2) *sont deux paires de distributions de probabilités, alors*

$$\mathrm{KL}\big(\,\lambda Q_1 + (1-\lambda)Q_2 \,\big\|\, \lambda P_1 + (1-\lambda)P_2\,\big) \;\leq\; \lambda\,\mathrm{KL}(Q_1\|P_1) + (1-\lambda)\,\mathrm{KL}(Q_2\|P_2)\,,$$

pour tout $\lambda \in [0,1]$ *.*

Corollaire A.5. *Les deux fonctions suivantes sont convexes :*

1. *La fonction* $\mathrm{kl}(q,p)$ *donnée par l'équation (4.6), c.-à-d. la divergence Kullback-Leibler entre deux distributions de Bernoulli ;*
2. *La fonction* $\mathrm{kl}_{\mathbf{2}}(q_1,q_2\,;\,p_1,p_2)$ *donnée par l'équation (4.37), c.-à-d. la divergence Kullback-Leibler entre deux distributions de variables aléatoires suivant une loi trinomiale.*

Démonstration. Ces résultats découlent directement du théorème A.4. □

Lemme A.6. *Soit deux entiers* a *et* b *tels que* $0 \leq a \leq b$*, alors*

$$\frac{1}{b+1} \;\leq\; \binom{b}{a}\left(\frac{a}{b}\right)^a\left(1-\frac{a}{b}\right)^{b-a} \;\leq\; 1\,.$$

Démonstration. L'expression $\binom{b}{a}\left(\frac{a}{b}\right)^a\left(1-\frac{a}{b}\right)^{b-a}$ correspond à la fonction de masse d'une loi binomiale de b épreuves avec probabilité de succès $\frac{a}{b}$, évalué au point a (l'événement le plus probable parmi les $b+1$ résultats possibles). □

Lemme A.7. *Soit deux entiers a et b tels que $0 < a < b$, alors*

$$\sqrt{\frac{b}{2\pi a(b-a)}}\, e^{-\frac{1}{12a}-\frac{1}{12(b-a)}} \;<\; \binom{b}{a}\left(\frac{a}{b}\right)^a\left(1-\frac{a}{b}\right)^{b-a} \;<\; \sqrt{\frac{b}{2\pi a(b-a)}}\, e^{\frac{1}{12b}}\,.$$

Démonstration. Le résultat est obtenu en réécrivant le coefficient binomial $\binom{b}{a} = \frac{b!}{a!(b-a)!}$, puis en bornant les expressions factorielles en utilisant l'approximation de Stirling, c'est-à-dire $\sqrt{2\pi n}\left(\frac{n}{e}\right)^n < n! < \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n}}$. □

Lemme A.8 (Inégalité de Cauchy-Schwarz)**.** *Soit X et Y des variables aléatoires. On a*

$$\left|\,\mathbf{E}(XY)\right|^2 \;\leq\; \mathbf{E}(X^2)\,\mathbf{E}(Y^2)\,.$$

A.2 Résultats provenant de la littérature (apprentissage automatique)

Le ***théorème du représentant*** (théorème A.9) affirme que le minimum f^* d'un problème d'optimisation respectant la forme de l'équation (A.1) s'exprime comme une combinaison linéaire de valeurs de la fonction de noyau $k(\cdot,\cdot)$ calculée sur les exemples de l'échantillon d'entraînement S (équation (A.2)).

Théorème A.9 (Schölkopf et al., 2001)**.** *Soit un espace d'entrée $\mathcal{X}$ non vide, un noyau semi-défini positif $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$, un échantillon d'entraînement $\{(x_1,y_1),\ldots,(x_m,y_m)\} \in (\mathcal{X}\times\mathbb{R})^m$, une fonction de perte arbitraire $c : (\mathcal{X}\times\mathbb{R}^2)^m \to \mathbb{R}\cup\{\infty\}$, une fonction de régularisation $g : \mathbb{R}\to\mathbb{R}$ strictement croissante sur $[0,\infty)$ et un ensemble de fonctions $\mathcal{F}$ tel que*

$$\mathcal{F} \;:=\; \left\{ f : \mathcal{X}\to\mathbb{R} \;\middle|\; f(\cdot) = \sum_{i=1}^{\infty} \beta_i k(z_i,\cdot),\ \beta_i\in\mathbb{R}, z_i\in\mathcal{X},\ \|f\| < \infty \right\},$$

*où $\|f\|$ est la norme de f dans l'*espace de Hilbert à noyau reproduisant *associée au noyau k (voir Schölkopf et al., 2001, équation (14)).*

Alors, tout $f^*\in\mathcal{F}$ minimisant la fonction de coût régularisée

$$c\left(\,(x_1,y_1,f(x_1)),\ldots,(x_m,y_m,f(x_m))\,\right) + g\left(\,\|f\|\,\right) \tag{A.1}$$

admet une représentation de la forme

$$f^*(\cdot) \;=\; \sum_{i=1}^{m} \alpha_i k(x_i,\cdot)\,. \tag{A.2}$$

Théorème A.10 (Lacasse, 2010, corollaire 3.2.2)**.** *Pour toute distribution de probabilité Q sur un ensemble de classificateurs et pour toute distribution D' sur $\mathcal{X}\times\mathcal{Y}$. Si $R_{D'}(G_Q) < 1/2$, alors*

$$R_{D'}(B_Q) \;\leq\; \frac{\underset{(x,y)\sim D'}{\mathbf{Var}}\left(W_Q(x,y)\right)}{\underset{(x,y)\sim D'}{\mathbf{Var}}\left(W_Q(x,y)\right) + \left(1/2 - R_{D'}(G_Q)\right)^2}\,,$$

où $W_Q(x,y)$ correspond au poids des classificateurs classifiant incorrectement l'exemple (x,y) dans le vote de majorité :

$$\begin{aligned} W_Q(x,y) \;&\stackrel{\text{def}}{=}\; \mathop{\mathbf{E}}_{h\sim Q} \mathtt{I}[h(x)\neq y] \qquad\qquad (A.3)\\ &=\; \mathop{\mathbf{E}}_{h\sim Q} \mathcal{L}_{01}\big(h(x),y\big) \;=\; \frac{1}{2}\left(1 - y\cdot \mathop{\mathbf{E}}_{h\sim Q} h(x)\right) \;=\; \frac{1}{2}\Big(1 - M_Q(x,y)\Big)\,. \end{aligned}$$

Lemme A.11 (Maurer, 2004)**.** *Soit un entier $n \geq 8$ et une collection de variables aléatoires i.i.d. $\mathbf{X} \coloneqq (X_1,\ldots,X_n)$ telles que $0 \leq X_i \leq 1$ et $\mu \coloneqq \mathbf{E}[X_i]$. Soit $M(\mathbf{X}) \coloneqq \frac{1}{n}\sum_{i=1}^n X_i$ la moyenne arithmétique des variables aléatoires. Alors*

$$\sqrt{n} \;\leq\; \mathbf{E}\, e^{n\,\mathrm{kl}(M(\mathbf{X}),\mu)} \;\leq\; 2\sqrt{n}\,.$$

Lemme A.12 (Maurer, 2004)**.** *Soit une variable aléatoire X dont la valeur appartient à $[0,1]$ et dont l'espérance est $\mu \coloneqq \mathbf{E}(X)$. Notons $\mathbf{X}$ le vecteur contenant n observations indépendantes de X. Considérons une variable aléatoire de Bernoulli X' (à valeur $\{0,1\}$) avec probabilité de succès μ (c.-à-d. $\Pr(X'=1)=\mu$). Notons $\mathbf{X}' \in \{0,1\}^n$ le vecteur contenant n observations indépendantes de X'.*

Soit une fonction convexe $g : [0,1]^n \to \mathbb{R}$, alors

$$\mathbf{E}\left[g(\mathbf{X})\right] \;\leq\; \mathbf{E}\left[g(\mathbf{X}')\right].$$

A.3 Résultats démontrés au cours de la thèse

Proposition A.13 (Concavité de l'équation (4.42))**.** *La fonction $F_c(e,d)$ est concave.*

Démonstration. Montrons que la matrice hessienne de $F_c(e,d)$ est une matrice négative semi-définie. Autrement dit, montrons que

$$\frac{\partial^2 F_c(e,d)}{\partial d^2} \leq 0\,; \quad \frac{\partial^2 F_c(e,d)}{\partial e^2} \leq 0\,; \quad \frac{\partial^2 F_c(e,d)}{\partial d^2}\frac{\partial^2 F_c(e,d)}{\partial e^2} - \left(\frac{\partial^2 F_c(e,d)}{\partial d\partial e}\right)^2 \geq 0\,.$$

En effet, on calcule

$$\begin{aligned} \frac{\partial^2 F_c(e,d)}{\partial d^2} &= \frac{2(1-4e)^2}{(2d-1)^3} \leq 0 \quad \forall e\in[0,1], d\in\left[0,\frac{1}{2}\right],\\ \frac{\partial^2 F_c(e,d)}{\partial e^2} &= \frac{8}{2d-1} \leq 0 \qquad\quad \forall e\in[0,1], d\in\left[0,\frac{1}{2}\right],\\ \frac{\partial^2 F_c(e,d)}{\partial d^2}\frac{\partial^2 F_c(e,d)}{\partial e^2} - \left(\frac{\partial^2 F_c(e,d)}{\partial d\partial e}\right)^2 &= \frac{2(1-4e)^2}{(2d-1)^3}\cdot\frac{8}{2d-1} - \left(\frac{4-16e}{(1-2d)^2}\right)^2 = 0\,. \end{aligned}$$

□

Lemme A.14 (Démonstration de l'équation (4.33))**.** *Considérons* $H \coloneqq \{(1,0),(0,1),(0,0)\}$ *et un entier* $n > 0$*. Tout point* $(\mathbf{x},\mathbf{y}) \in \big([0,1]\times[0,1]\big)^n$ *peut être écrit comme une combinaison convexe des points extrêmes* $\boldsymbol{\eta} \coloneqq (\eta_1,\eta_2,\ldots,\eta_n) \in H^n$ *:*

$$(\mathbf{x},\mathbf{y}) \;=\; \sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y})\cdot\boldsymbol{\eta}\,,$$

où

$$\rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y}) \;\stackrel{\text{def}}{=}\; \left(\prod_{i:\eta_i=(1,0)} x_i\right)\left(\prod_{i:\eta_i=(0,1)} y_i\right)\left(\prod_{i:\eta_i=(0,0)} 1{-}x_i{-}y_i\right).$$

Démonstration. Nous procédons par induction sur n, la taille du vecteur.

Démonstration pour $n = 1$ *:*

$$\begin{aligned}\sum_{\boldsymbol{\eta}\in H} \rho_{\boldsymbol{\eta}}((x_1,y_1))\cdot\boldsymbol{\eta} \;&=\; x_1\cdot((1,0)) + y_1\cdot((0,1)) + (1{-}x_1{-}y_1)\cdot((0,0))\\ &=\; ((x_1,y_1))\,.\end{aligned}$$

Démonstration pour $n > 1$ *:* Supposons que le résultat est vrai pour tout vecteur $(\mathbf{x},\mathbf{y})$ d'une taille n déterminée et montrons que cela implique

$$\sum_{(\boldsymbol{\eta},\eta_{n+1})\in H^{n+1}} \Big[\rho_{(\boldsymbol{\eta},\eta_{n+1})}\big((\mathbf{x},\mathbf{y}),(x_{n+1},y_{n+1})\big)\Big]\cdot(\boldsymbol{\eta},\eta_{n+1}) \;=\; \big((\mathbf{x},\mathbf{y}),\,(x_{n+1},y_{n+1})\big)\,,$$

où $(\mathbf{a},\,b)$ correspond au vecteur $\mathbf{a}$ augmenté d'un élément b.

On obtient

$$\begin{aligned}&\sum_{(\boldsymbol{\eta},\eta_{n+1})\in H^{n+1}} \Big[\rho_{(\boldsymbol{\eta},\eta_{n+1})}\big((\mathbf{x},\mathbf{y}),(x_{n+1},y_{n+1})\big)\Big]\cdot(\boldsymbol{\eta},\eta_{n+1})\\
&\quad= \sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y})\cdot x_{n+1}\cdot(\boldsymbol{\eta},(1,0)) + \sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y})\cdot y_{n+1}\cdot(\boldsymbol{\eta},(0,1))\\
&\qquad\qquad + \sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y})\cdot(1{-}x_{n+1}{-}y_{n+1})\cdot(\boldsymbol{\eta},(0,0))\\
&\quad= \left(\sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y})\cdot(x_{n+1}{+}y_{n+1}{+}1{-}x_{n+1}{-}y_{n+1})\cdot\boldsymbol{\eta},\; \sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y})\cdot(x_{n+1},y_{n+1})\right)\\
&\quad= \left(\sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y})\cdot\boldsymbol{\eta},\; \sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y})\cdot(x_{n+1},y_{n+1})\right)\\
&\quad= \Big((\mathbf{x},\mathbf{y}),\,(x_{n+1},y_{n+1})\Big)\,.\end{aligned}$$

Dans la dernière égalité, le terme $(\mathbf{x},\mathbf{y})$ du vecteur $\Big((\mathbf{x},\mathbf{y}),\,(x_{n+1},y_{n+1})\Big)$ est obtenu de l'hypothèse d'induction et le couple (x_{n+1},y_{n+1}) est une conséquence directement de l'égalité

suivante :

$$\sum_{\boldsymbol{\eta}\in H^n} \rho_{\boldsymbol{\eta}}(\mathbf{x},\mathbf{y}) \;=\; \prod_{i=1}^{n} \Big(x_i{+}y_i{+}1{-}x_i{-}y_i\Big) \;=\; 1\,.$$

□

Lemme A.15. *Soit $\beta \in [0,1]$, $q \in [0,1]$ et $p \in (0,1)$. On a*

$$\Delta_\beta(q,p) \;=\; \mathrm{kl}(q,p) + \frac{1-\beta}{\beta}\,\mathrm{kl}\Big(\frac{p-\beta\,q}{1-\beta}, p\Big)\,,$$

où $\mathrm{kl}(\cdot,\cdot)$ et $\Delta_\beta(\cdot,\cdot)$ sont définis respectivement par les équations (4.6) et (5.13).

Démonstration.

$$\begin{aligned}
&\Delta_\beta(q,p)\\
&\quad= q\ln\beta\tfrac{q}{p} + (\tfrac{p}{\beta}-q)\ln(1-\beta\tfrac{q}{p}) + (1-q)\ln\beta\tfrac{1-q}{1-p} + (\tfrac{1-p}{\beta}+q-1)\ln(1-\beta\tfrac{1-q}{1-p})\\
&\qquad - \ln\beta - (\tfrac{1}{\beta}-1)\ln(1-\beta)\\
&\quad= q\ln\tfrac{q}{p} + q\ln\beta + (\tfrac{p}{\beta}-q)\ln(1-\beta\tfrac{q}{p}) + (1-q)\ln\tfrac{1-q}{1-p} + (1-q)\ln\beta + (\tfrac{1-p}{\beta}+q-1)\ln(1-\beta\tfrac{1-q}{1-p})\\
&\qquad - \ln\beta - \ (\tfrac{1}{\beta}-1)\ln(1-\beta)\\
&\quad= q\ln\tfrac{q}{p} + (\tfrac{p}{\beta}-q)\ln(1-\beta\tfrac{q}{p}) + (1-q)\ln\tfrac{1-q}{1-p} + (\tfrac{1-p}{\beta}+q-1)\ln(1-\beta\tfrac{1-q}{1-p}) - (\tfrac{1}{\beta}-1)\ln(1-\beta)\\
&\quad= \mathrm{kl}(q,p) + (\tfrac{p}{\beta}-q)\ln(1-\beta\tfrac{q}{p}) + (\tfrac{1-p}{\beta}+q-1)\ln(1-\beta\tfrac{1-q}{1-p}) - (\tfrac{1}{\beta}-1)\ln(1-\beta)\\
&\quad= \mathrm{kl}(q,p) + (\tfrac{p}{\beta}-q)\ln(1-\beta\tfrac{q}{p}) + (\tfrac{1-p}{\beta}+q-1)\ln(1-\beta\tfrac{1-q}{1-p}) - \Big[(\tfrac{p}{\beta}-q) + (\tfrac{1-p}{\beta}+q-1)\Big]\ln(1-\beta)\\
&\quad= \mathrm{kl}(q,p) + (\tfrac{p}{\beta}-q)\Big[\ln(1-\beta\tfrac{q}{p}) - \ln(1-\beta)\Big] + (\tfrac{1-p}{\beta}+q-1)\Big[\ln(1-\beta\tfrac{1-q}{1-p}) - \ln(1-\beta)\Big]\\
&\quad= \mathrm{kl}(q,p) + (\tfrac{p}{\beta}-q)\ln\tfrac{1-\beta\frac{q}{p}}{1-\beta} + (\tfrac{1-p}{\beta}+q-1)\ln\tfrac{1-\beta\frac{1-q}{1-p}}{1-\beta}\\
&\quad= \mathrm{kl}(q,p) + \tfrac{1-\beta}{\beta}\left[\tfrac{p-\beta q}{1-\beta}\ln\tfrac{1-\beta\frac{q}{p}}{1-\beta} + \Big(1-\tfrac{p-\beta q}{1-\beta}\Big)\ln\tfrac{1-\beta\frac{1-q}{1-p}}{1-\beta}\right]\\
&\quad= \mathrm{kl}(q,p) + \tfrac{1-\beta}{\beta}\left[\tfrac{p-\beta q}{1-\beta}\ln\tfrac{\frac{p-\beta q}{1-\beta}}{p} + \Big(1-\tfrac{p-\beta q}{1-\beta}\Big)\ln\tfrac{1-\frac{p-\beta q}{1-\beta}}{1-p}\right]\\
&\quad= \mathrm{kl}(q,p) + \tfrac{1-\beta}{\beta}\,\mathrm{kl}\Big(\tfrac{p-\beta q}{1-\beta}, p\Big)\,.
\end{aligned}$$

□

Lemme A.16. *Soit des entiers positifs λ, m, N, K tels que $\lambda \le m \le N-\lambda$ et $0 \le K \le N$. On a*

$$F(k) \;\coloneqq\; \frac{\psi(k,K)\,\psi(m{-}k,N{-}K)}{\psi(m,N)} \;\le\; e^{\frac{1}{6\lambda}}\sqrt{2\pi m(1-\tfrac{m}{N})}\,,$$

pour $k = \max[0, K{+}m{-}N]$ ou $k = \min[m,K]$.

Démonstration. **(1)** Examinons d'abord le cas $k = \max[0, K{+}m{-}N]$.
Si $0 \ge K{+}m{-}N$, alors $F(0) = \frac{\psi(m,N-K)}{\psi(m,N)}$ croît selon K et la valeur maximum est atteinte en $K = N{-}m$. Alors

$$F(0) \;\le\; \frac{\psi(m,m)}{\psi(m,N)} \;=\; \frac{1}{\psi(m,N)}\,.$$

Si $0 \le K+m-N$, alors $F(K+m-N) = \frac{\psi(K+m-N,K)\,\psi(N-K,N-K)}{\psi(m,N)} = \frac{\psi(K+m-N,K)}{\psi(m,N)}$ décroît selon K et la valeur maximum est atteint en $K = N-m$. Alors

$$F(K+m-N) \;=\; F(0) \;\le\; \frac{1}{\psi(m,N)}\,.$$

(2) Examinons maintenant le cas $k = \min[m,K]$.
Si $m \le K$, alors $F(m) = \frac{\psi(m,K)}{\psi(m,N)}$ décroît selon K et la valeur maximale est atteint en $K = m$. Alors

$$F(m) \;\le\; \frac{\psi(m,m)}{\psi(m,N)} \;=\; \frac{1}{\psi(m,N)}\,.$$

Si $m \ge K$, alors $F(K) = \frac{\psi(K,K)\,\psi(m-K,N-K)}{\psi(m,N)} = \frac{\psi(m-K,N-K)}{\psi(m,N)}$ croît selon K et le maximum est atteint en $K = m$. Alors

$$F(K) \;=\; F(m) \;\le\; \frac{1}{\psi(m,N)}\,.$$

(3) Finalement, par le lemme A.7, on obtient

$$\begin{aligned}\frac{1}{\psi(m,N)} \;&\le\; \frac{1}{\sqrt{\frac{N}{2\pi m(N-m)}}\,e^{-\frac{1}{12m}-\frac{1}{12(N-m)}}}\\ &=\; \sqrt{2\pi\, m(1-\tfrac{m}{N})}\,e^{\frac{1}{12m}+\frac{1}{12(N-m)}}\\ &\le\; e^{\frac{1}{6\lambda}}\sqrt{2\pi m(1-\tfrac{m}{N})}\,.\end{aligned}$$

□

Lemme A.17. *Soit des entiers m, N, K tels que $0 \le m \le N$ et $0 \le K \le N$. On a*

$$\begin{aligned}\sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\Big(\frac{1}{k}+\frac{1}{K-k}\Big)\,\Big(\frac{1}{m-k}+\frac{1}{(N-K)-(m-k)}\Big)} \;&\le\; 2\sum_{k=1}^{m-1}\frac{1}{k} \qquad (A.4)\\ &\le\; 2\big[1+\ln(m-1)\big]\,,\end{aligned}$$

où

$$\mathcal{K}^*_{m,N,K} \;=\; \big\{\,\max[0,K+m-N]+1,\ldots,\min[m,K]-1\big\}\,,$$

et on a l'égalité à la ligne (A.4) lorsque $m = K = N-K$.

Démonstration. Premièrement, examinons le cas $m = K = N-K$.

$$\begin{aligned}&\sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\Big(\tfrac{1}{k}+\tfrac{1}{K-k}\Big)\,\Big(\tfrac{1}{m-k}+\tfrac{1}{(N-K)-(m-k)}\Big)}\\ &\quad=\; \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\Big(\tfrac{1}{k}+\tfrac{1}{m-k}\Big)\,\Big(\tfrac{1}{m-k}+\tfrac{1}{m-(m-k)}\Big)} \;=\; \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\Big(\tfrac{1}{k}+\tfrac{1}{m-k}\Big)\,\Big(\tfrac{1}{m-k}+\tfrac{1}{k}\Big)}\\ &\quad=\; \sum_{k\in\mathcal{K}^*_{m,N,K}} \Big(\tfrac{1}{k}+\tfrac{1}{m-k}\Big) \;=\; \sum_{k\in\mathcal{K}^*_{m,N,K}} \tfrac{1}{k} + \sum_{k\in\mathcal{K}^*_{m,N,K}} \tfrac{1}{m-k} \;=\; 2\sum_{k\in\mathcal{K}^*_{m,N,K}} \tfrac{1}{k} \;=\; 2\sum_{k=1}^{m-1}\tfrac{1}{k}\,.\end{aligned}$$

La dernière égalité s'explique par le fait que, lorsque $m = K = N - K$, l'ensemble $\mathcal{K}^*_{m,N,K}$ vaut $\{1, 2, \ldots, m-1\}$. Ainsi, les deux sommes sont équivalentes.

Nous distinguons maintenant quatre autres cas.

Cas 1 : $m \leq (N-K)$ et $m \leq K$.

$$\begin{aligned}
&\sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{K-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{(N-K)-(m-k)}\right)} \\
&\quad\leq \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{m-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{(N-K)-(m-k)}\right)} \\
&\quad\leq \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{m-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{m-(m-k)}\right)} \;=\; \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{m-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{k}\right)} \\
&\quad= \sum_{k\in\mathcal{K}^*_{m,N,K}} \left(\tfrac{1}{k}+\tfrac{1}{m-k}\right) \;=\; \sum_{k\in\mathcal{K}^*_{m,N,K}} \tfrac{1}{k} + \sum_{k\in\mathcal{K}^*_{m,N,K}} \tfrac{1}{m-k} \;=\; \sum_{k=1}^{m-1} \tfrac{1}{k} + \sum_{k=1}^{m-1} \tfrac{1}{m-k} \;=\; 2\sum_{k=1}^{m-1} \tfrac{1}{k}\,.
\end{aligned}$$

Cas 2 : $m \leq (N-K)$ et $m > K$.

$$\begin{aligned}
&\sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{K-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{(N-K)-(m-k)}\right)} \\
&\quad\leq \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{K-k}\right)\left(\tfrac{1}{K-k}+\tfrac{1}{(N-K)-(m-k)}\right)} \\
&\quad\leq \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{K-k}\right)\left(\tfrac{1}{K-k}+\tfrac{1}{m-(m-k)}\right)} \;=\; \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{K-k}\right)\left(\tfrac{1}{K-k}+\tfrac{1}{k}\right)} \\
&\quad= \sum_{k\in\mathcal{K}^*_{m,N,K}} \left(\tfrac{1}{k}+\tfrac{1}{K-k}\right) \;=\; \sum_{k=1}^{K-1} \left(\tfrac{1}{k}+\tfrac{1}{K-k}\right) \;=\; \sum_{k=1}^{K-1} \tfrac{1}{k} + \sum_{k=1}^{K-1} \tfrac{1}{K-k} \;=\; 2\sum_{k=1}^{K-1} \tfrac{1}{k} \;<\; 2\sum_{k=1}^{m-1} \tfrac{1}{k}\,.
\end{aligned}$$

Cas 3 : $m > (N-K)$ et $m \leq K$.

$$\begin{aligned}
&\sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{K-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{(N-K)-(m-k)}\right)} \\
&\qquad\leq \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{k}+\tfrac{1}{m-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{(N-K)-(m-k)}\right)} \\
&\qquad\leq \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\left(\tfrac{1}{(N-K)-(m-k)}+\tfrac{1}{m-k}\right)\left(\tfrac{1}{m-k}+\tfrac{1}{(N-K)-(m-k)}\right)} \\
&\qquad= \sum_{k\in\mathcal{K}^*_{m,N,K}} \left(\tfrac{1}{(N-K)-(m-k)}+\tfrac{1}{m-k}\right) \;=\; \sum_{k=m-N+K+1}^{m-1} \left(\tfrac{1}{(N-K)-(m-k)}+\tfrac{1}{m-k}\right) \\
&\qquad= 2\sum_{k=m-N+K+1}^{m-1} \tfrac{1}{m-k} \;<\; 2\sum_{k=1}^{m-1} \tfrac{1}{m-k} \;=\; 2\sum_{k=1}^{m-1} \tfrac{1}{k}\,.
\end{aligned}$$

Cas 4 : $m > (N-K)$ et $m > K$.

$$\begin{aligned}
&\sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\Big(\tfrac{1}{k}+\tfrac{1}{K-k}\Big)\,\Big(\tfrac{1}{m-k}+\tfrac{1}{(N-K)-(m-k)}\Big)} \\
&\qquad\le \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\Big(\tfrac{1}{k}+\tfrac{1}{K-k}\Big)\,\Big(\tfrac{1}{K-k}+\tfrac{1}{(N-K)-(m-k)}\Big)} \\
&\qquad\le \sum_{k\in\mathcal{K}^*_{m,N,K}} \sqrt{\Big(\tfrac{1}{(N-K)-(m-k)}+\tfrac{1}{K-k}\Big)\,\Big(\tfrac{1}{K-k}+\tfrac{1}{(N-K)-(m-k)}\Big)} \\
&\qquad= \sum_{k\in\mathcal{K}^*_{m,N,K}} \Big(\tfrac{1}{(N-K)-(m-k)}+\tfrac{1}{K-k}\Big) \;=\; \sum_{k=m-N+K+1}^{K-1} \Big(\tfrac{1}{(N-K)-(m-k)}+\tfrac{1}{K-k}\Big) \\
&\qquad= 2\sum_{k=m-N+K+1}^{K-1} \tfrac{1}{K-k} \;\le\; 2\sum_{k=1}^{K-1} \tfrac{1}{K-k} \;=\; 2\sum_{k=1}^{K-1} \tfrac{1}{k} \;\le\; 2\sum_{k=1}^{m-1} \tfrac{1}{k}\,.
\end{aligned}$$

Pour les quatre cas ci-haut, nous avons démontré que l'expression est plus petite ou égale à $2\sum_{k=1}^{m-1}\frac{1}{k}$. En utilisant la technique d'approximation par une intégrale, nous obtenons le résultat désiré :

$$2\sum_{k=1}^{m-1}\frac{1}{k} \;\le\; 2\left(1+\int_1^{m-1}\frac{1}{x}\,dx\right) \;=\; 2\Big(1+\ln(m-1)\Big)\,.$$

□

Annexe B

Détails des expérimentations

B.1 Implémentations disponibles en ligne

Calcul des bornes dans le cadre inductif. Le code source pour calculer les bornes inductives présentées au chapitre 4 a été rendu public à l'occasion de la publication de l'article Germain et al. (2015b) :

`http://graal.ift.ulaval.ca/majorityvote/`
`https://github.com/GRAAL-Research/majority-vote-bounds`

Calcul des bornes dans le cadre transductif. Le code source pour calculer les bornes inductives présentées au chapitre 5 a été rendu public à l'occasion de la publication de l'article Bégin et al. (2014) :

`http://graal.ift.ulaval.ca/aistats2014`
`https://github.com/GRAAL-Research/Transductive-PAC-Bayes`

Algorithme d'apprentissage pour l'adaptation de domaine (PBDA). Le code source de l'algorithme PBDA présenté au chapitre 6 a été rendu public à l'occasion de la publication de l'article Germain et al. (2013) :

`http://graal.ift.ulaval.ca/pbda/`
`https://github.com/pgermain/pbda`

B.2 Expérimentations avec l'algorithme AdaBoost

Les expérimentations empiriques avec l'algorithme d'apprentissage AdaBoost présentées dans différentes sections de la thèse ont été réalisées avec une implémentation de l'algorithme 2 (page 33) présenté en introduction. Les votants sont des *souches de décision* et la distribution *a priori* est uniforme.

Ensemble de votants. Comme défini à la section 2.4.3 (page 34), pour classifier un exemple

$$\mathbf{x} \;\coloneqq\; (a_1, a_2, \ldots, a_n) \in \mathbb{R}^n \,,$$

une souche de décision $h_{i,t,d}$ considère la valeur d'un attribut a_i. L'équation (2.29) détermine la sortie du classificateur en fonction de la valeur de seuil $t \in \mathbb{R}$ et la direction $d \in \{-1, +1\}$.

Étant donné un échantillon d'entraînement S, notons A_i (pour $i \in \{1, \ldots, n\}$) l'ensemble des valeurs du $i$ème attribut des exemples observés dans l'échantillon d'entraînement S. Puisque la théorie PAC-bayésienne requiert que l'ensemble des votants $\mathcal{H}$ soit indépendant de S, nous supposons que les valeurs de $\min[A_i]$ et $\max[A_i]$ sont connues préalablement à l'apprentissage. Autrement dit, nous connaissons les valeurs minimales et maximales de chaque attribut des exemples.

Notre ensemble de votants $\mathcal{H}$ est constitué de 20 votants par attributs ($|\mathcal{H}| = 20n$), dont les valeurs de seuils séparent l'intervalle $\Big[\min[A_i], \max[A_i]\Big]$ en 10 parties égales, avec deux souches de décision pour chaque valeur de seuil (une avec $d = -1$ et l'autre avec $d = 1$). Ainsi,

$$\mathcal{H} \;\coloneqq\; \bigcup_{i=1}^{n} \bigcup_{k=1}^{10} \Big\{ h_{i,t_{i;k},-1} \,,\, h_{i,t_{i;k},1} \Big\} \qquad \text{où} \qquad t_{i;k} \;\coloneqq\; \tfrac{k}{10} \min A_i + \tfrac{10-k}{10} \max A_i \,.$$

Distributions P et Q. Nous considérons toujours une distribution *a priori* P sur $\mathcal{H}$ uniforme (non informative). Autrement dit, pour tout $h_i \in \mathcal{H}$,

$$P(h_i) \;\coloneqq\; \frac{1}{|\mathcal{H}|} \,.$$

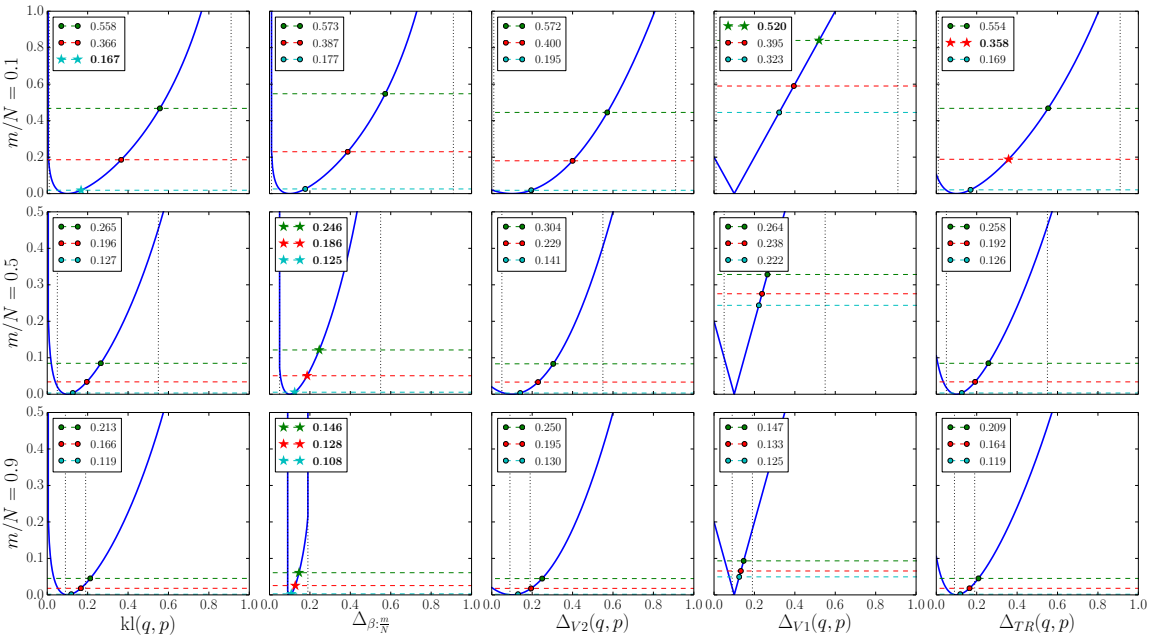
La distribution de poids *a posteriori* Q sur $\mathcal{H}$ est calculée à partir des poids $\boldsymbol{\alpha}$ retournés par AdaBoost, conformément à l'équation (2.28). Notons qu'AdaBoost peut choisir un même votant lors de plusieurs itérations distinctes, ce qui n'est pas pris en compte par l'équation (2.28). En pratique, lorsque cela advient, nous additionnons les poids dans la distribution Q.

L'expression de la divergence Kullback-Leibler (équation (4.1), page 58) entre les distributions P et Q devient alors :

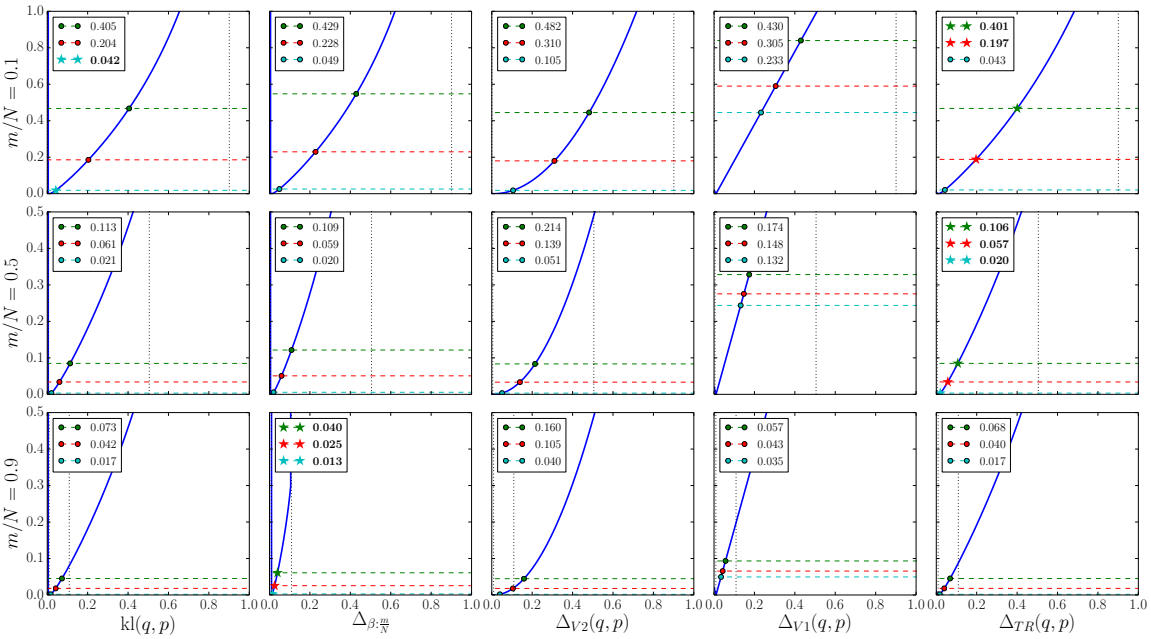
$$\begin{aligned}
\mathrm{KL}(Q\|P) &= \mathop{\mathbf{E}}_{h\sim Q} \ln \frac{Q(h)}{P(h)} \\
&= \sum_{i=1}^{|\mathcal{H}|} Q(h_i) \cdot \Big(\ln Q(h_i) - \ln |\mathcal{H}| \Big) \\
&= -\ln |\mathcal{H}| + \sum_{i=1}^{|\mathcal{H}|} Q(h_i) \ln Q(h_i) \,.
\end{aligned}$$

B.3 Résultats supplémentaires pour le cadre transductif

La figure B.1 (page 175) présente des résultats supplémentaires obtenus lors des expérimentations empiriques décrits à la section 5.5.1.



(a) Valeur $R_S(G_Q) \coloneqq 0.1$.



(b) Valeur $R_S(G_Q) \coloneqq 0.01$.

FIGURE B.1 – Étude du comportement des bornes transductives (résultats supplémentaires). Comme les résultats présentés par la figure 5.3 (page 110), tous les graphiques considèrent des valeurs fixes de $\mathrm{KL}(Q\|P) \coloneqq 5$ et $\delta \coloneqq 0.05$, mais on considère ici de plus petites valeurs de $R_S(G_Q)$. Les trois graphiques de chaque colonne partagent la même Δ-fonction, tandis que les cinq graphiques de chaque ligne partagent un même ratio $\frac{m}{N}$.

Annexe C

A PAC-Bayes Sample Compression Approach to Kernel Methods

Cet article a été publié dans le cadre de la conférence *International Conference on Machine Learning* (voir Germain et al., 2011). Son contenu fait aussi l'objet de la thèse de Shanian (2012).

Abstract. We propose a PAC-Bayes sample compression approach to kernel methods that can accommodate any bounded similarity function and show that the support vector machine (SVM) classifier is a particular case of a more general class of data-dependent classifiers known as majority votes of sample-compressed classifiers. We provide novel risk bounds for these majority votes and learning algorithms that minimize these bounds.

C.1 Introduction

Kernel methods such as the support vector machine (SVM) have provided state-of-the-art machine learning algorithms over the last decade. Despite their success, these methods are currently limited by the fact that the similarity function that they use must be a symmetric positive semi-definite kernel. However, for many applications, we would like to be able to use any similarity measure of input examples and not be limited by the fact that the chosen measure should be expressible as an inner product of feature vectors. We therefore propose here a PAC-Bayes sample-compression approach to kernel methods that can accommodate any bounded similarity function. Within the sample-compression framework (Floyd et Warmuth, 1995; Laviolette et Marchand, 2007) each sample-compressed classifier is partly represented by a subsequence of the training data, called the compression sequence. We show here that the SVM classifier is actually a particular case of a (weighted) majority vote of sample-compressed classifiers where the compression sequence of each classifier consists of at most a single training example. Inspired by the work of Germain et al. (2009b) on general loss bounds

for stochastic classifiers, we propose two different PAC-Bayes risk bounds for majority votes of sample-compressed classifiers which are valid for any similarity measure of input examples. Consequently, the proposed bounds also apply to the class of linear classifiers of similarity-based features that were studied by Chen et al. (2009b). Indeed, for the class of indefinite similarity measures, their risk bound is trivial (and useless) in the limit where each training example is used for a prototype. In contrast, the risk bounds presented here do not suffer from such a limitation.

For each proposed risk bound, we provide a learning algorithm that minimizes it. One of the PAC-Bayes risk bound depends on the KL divergence between the prior and the posterior over the set of sample-compressed classifiers and, consequently, the corresponding bound-minimizing learning algorithm is KL-regularized. The other PAC-Bayes risk bound has the unusual property of having no KL divergence when the posterior is *aligned* with the prior in some precise way defined below. Consequently, to minimize this risk upper bound, we only need to minimize the proposed empirical loss under the constraint that the posterior is kept aligned with the prior. When a positive semi-definite (PSD) kernel is used, our experiments indicate that the proposed algorithms are very competitive with the SVM. Good empirical results are also obtained when the proposed algorithms are used with a non-PSD kernel. Finally, the proposed algorithms are also competitive with the best similarity-based learning algorithms proposed by Chen et al. (2009b).

C.2 PAC-Bayesian Sample Compression

We consider binary classification problems where an example $z=(x,y)\in\mathcal{Z}$ is an input-output pair where $x\in\mathcal{X}$ and $y\in\mathcal{Y}=\{-1,+1\}$. We adopt the PAC setting where each example z is drawn according to a fixed, but unknown, probability distribution D on $\mathcal{Z}$. In the *sample compression setting*, learning algorithms have access to a data-dependent set of classifiers, that we refer to as *sc-classifiers*. Given a training sequence $S=\langle z_1,..,z_m\rangle$, each sc-classifier is described by a subsequence $S_{\mathbf{i}}$ of S called the *compression sequence*, and a *message* μ which represents the additional information needed to obtain a classifier from $S_{\mathbf{i}}$. The compression sequence $S_{\mathbf{i}}$ is defined by the following vector $\mathbf{i}$ of indices

$$\mathbf{i} \stackrel{\text{def}}{=} \langle i_1, i_2, \ldots, i_{|\mathbf{i}|}\rangle\,,$$

with $1\leq i_1<i_2<\ldots<i_{|\mathbf{i}|}\leq m$. The number of indices present in $\mathbf{i}$ is denoted by $|\mathbf{i}|$, and the vector of indices of a sc-classifier h by $\mathbf{i}_h$. The set of all the 2^m possible vectors of indices is denoted by $\mathcal{I}$. The fact that each sc-classifier is described by a compression sequence and a message implies that there exists a *reconstruction function* $\mathcal{R}$ that outputs a classifier

$$h^{\mu}_{S'} \stackrel{\text{def}}{=} \mathcal{R}(S',\mu) \tag{C.1}$$

when given an arbitrary compression sequence S' and a message μ chosen from the set $\mathcal{M}_{S'}$ of all messages that can be supplied with the compression sequence S'. $\mathcal{M}_{S'}$ must be defined

a priori (before observing the training data) for all possible sequences S'. The messages can be strings or values taken from a continuous set. In our case, $\mathcal{M}_{S'}$ will be continuous.

Given a training sequence S, $\mathcal{H}^S$ denotes the set of all sc-classifiers $\mathcal{R}(S_{\mathbf{i}}, \mu)$ such that $\mu \in \mathcal{M}_{S_{\mathbf{i}}}$ and $\mathbf{i} \in \mathcal{I}$. The perceptron learning rule and the SVM are examples where the final classifier can be reconstructed solely from a compression sequence. In contrast, the reconstruction function of the Set Covering Machine (Marchand et Shawe-Taylor, 2002) needs both a compression sequence and a message string.

The risk $R_D(h)$ (or generalization error) and the *empirical risk* $R_S(h)$ on S of a sc-classifier h are defined as

$$\begin{aligned} R_D(h) &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim D} I(h(x) \neq y) = \Pr_{(x,y)\sim D}(h(x) \neq y)\,, \\ R_S(h) &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim S} I(h(x) \neq y) = \frac{1}{m}\sum_{i=1}^{m} I(h(x_i) \neq y_i)\,, \end{aligned}$$

where $I(a) = 1$ if predicate a is true and 0 otherwise, and where $(x, y) \sim S$ means that (x, y) is drawn according to the uniform distribution on S.

Note that both R_D and R_S are defined only within the context of a training sequence S. In the non-sample compressed setting, $mR_S(h)$ is a binomial random variable of parameters $(m, R_D(h))$. In our setting this is no longer the case because the risk can then be biased by the elements of S that are in the compression sequence. However, if $a_h \stackrel{\text{def}}{=} \sum_{(x,y)\in S_{\mathbf{i}_h}} I(h(x \neq y)$, then $mR_S(h) - a_h$ is a binomial random variable with parameters $(m - |\mathbf{i}|, R_D(h))$. As mentioned in Laviolette et Marchand (2007), the empirical risk of a sc-classifier is usually computed on $S \setminus S_{\mathbf{i}}$. In order to obtain risk bounds having simpler statements, we decide not to follow this strategy and, therefore, deal with this bias directly in the proposed theory.

After observing the training sequence S, the task of the learner is to choose a *posterior* distribution Q over $\mathcal{H}^S$ such that the Q-weighted majority vote classifier B_Q will have the smallest possible risk. On any input example x, the output $B_Q(x)$ of the majority vote classifier (also called the *Bayes classifier*) is given by

$$B_Q(x) \stackrel{\text{def}}{=} \operatorname{sgn}\left[\mathop{\mathbf{E}}_{h\sim Q} h(x)\right], \tag{C.2}$$

where $\operatorname{sgn}(s) = +1$ if $s > 0$ and $\operatorname{sgn}(s) = -1$ otherwise.

Given a training sequence S, we denote by $Q_{\mathcal{I}}(\mathbf{i})$, the probability that a compression sequence $S_{\mathbf{i}}$ is chosen by Q, and by $Q_{S_{\mathbf{i}}}(\mu)$, the probability distribution of choosing μ given $S_{\mathbf{i}}$. Consequently,

$$Q_{\mathcal{I}}(\mathbf{i}) \stackrel{\text{def}}{=} \int_{\mu\in\mathcal{M}(S_{\mathbf{i}})} Q(h^{\mu}_{S_{\mathbf{i}}})\, d\mu \quad \text{and} \quad Q_{S_{\mathbf{i}}}(\mu) \stackrel{\text{def}}{=} Q(h^{\mu}_{S_{\mathbf{i}}} \mid S_{\mathbf{i}})\,.$$

Priors on sc-classifiers. In PAC-Bayes theory, risk bounds are obtained by comparing a *posterior* distribution Q on $\mathcal{H}^S$ to a *prior* defined before observing the training sequence S. Therefore, in standard PAC-Bayes bounds (McAllester, 2003a; Seeger, 2002), the prior is independent of S. In our setting, this seems problematic since sc-classifiers are defined upon S. To overcome this difficulty, we follow Laviolette et Marchand (2007) and define a *prior* $\mathcal{P}$ as a couple $\Big(P_{\mathcal{I}},\, (P_{S'})_{S'\in\mathcal{Z}^j,\, j\leq m}\Big)$, where $P_{\mathcal{I}}$ is a distribution on $\mathcal{I}$, and, for every possible compression sequence S', $P_{S'}$ is a distribution on $\mathcal{M}_{S'}$. Given a training sequence S, P denotes the distribution on $\mathcal{H}^S$ associate with the prior $\mathcal{P}$. Consequently,

$$P(h^{\mu}_{S_{\mathbf{i}}}) \;=\; P_{\mathcal{I}}(\mathbf{i})\, P_{S_{\mathbf{i}}}(\mu)\,. \tag{C.3}$$

Hence, although $\mathcal{P}$ (and thus $P_{\mathcal{I}}$) is defined without reference to any specific training sequence S, the distribution P on $\mathcal{H}^S$ refers to a specific realization of the prior $\mathcal{P}$ on the observed training sequence S. As a result, the risk bounds of this paper only depend on the observed training sequence and not on some prior distribution over all possible training sequences. The way it is accomplished is detailed in the proof of Claim 1 of Theorem C.5 in supplementary material (Section C.3.1).

Gibbs classifier. In the usual PAC-Bayes setting, a bound on $R_D(B_Q)$ is indirectly obtained by bounding the risk of an associated stochastic classifier known as the Gibbs classifier. To assign an output label to an input example x, the Gibbs classifier G_Q randomly chooses a classifier h according to Q and uses $h(x)$ for the assigned label. In the sample-compressed PAC-Bayes setting, given a training sequence S, G_Q randomly chooses $\mathbf{i}$ according to $Q_{\mathcal{I}}$, then chooses a message μ according to $Q_{S_{\mathbf{i}}}$, and then classifies x according to $h^{\mu}_{S_{\mathbf{i}}}(x)$. Given a distribution D and a training sequence S generated by D, the true risk $R_D(G_Q)$ and its empirical estimate $R_S(G_Q)$ on S are thus given by

$$\begin{aligned} R_D(G_Q) &= \mathop{\mathbf{E}}_{h^{\mu}_{S_{\mathbf{i}}}\sim Q}\; \mathop{\mathbf{E}}_{(x,y)\sim D} I(h^{\mu}_{S_{\mathbf{i}}}(x)\neq y)\,,\\ R_S(G_Q) &= \mathop{\mathbf{E}}_{h^{\mu}_{S_{\mathbf{i}}}\sim Q}\; \mathop{\mathbf{E}}_{(x,y)\sim S} I(h^{\mu}_{S_{\mathbf{i}}}(x)\neq y)\,. \end{aligned}$$

Note that whenever B_Q errs on (x,y), at least half of the classifiers, under measure Q err on (x,y). It follows that $R_D(B_Q)\leq 2R_D(G_Q)$. Hence, an upper-bound on $R(G_Q)$ also provides an upper bound on $R(B_Q)$ via this well-known factor-of-two rule. However, we focus in this paper on majority votes of sc-classifiers having a small compression size. In this setting, $\mathcal{H}^S$ consists mostly of "weak" classifiers having large risk $R(h)$. Then, $R_D(G_Q)$ is (almost) always large (near $1/2$) for any Q *even if the majority vote* B_Q *has very low risk.* Thus, the disparity between $R_D(B_Q)$ and $R_D(G_Q)$ is enormous. Consequently, trying to minimize an upper-bound on $R_D(G_Q)$ should not lead to a majority vote B_Q having low risk. In fact, our experiments of Section C.2.4 empirically confirm this to be the case. One way to obtain a more relevant bound on $R_D(B_Q)$ from the PAC-Bayes theory is to use a loss function for stochastic

classifiers which is distinct from the zero-one loss used for the deterministic classifiers. In order to obtain a tractable optimization problem, we propose to use a convex loss function of the margin of the Q-convex combination of sc-classifiers where the margin, on example (x, y), is defined as

$$M_Q(x,y) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{h^\mu_{S_\mathbf{i}} \sim Q} y h^\mu_{S_\mathbf{i}}(x)\,. \tag{C.4}$$

Note that $R_D(G_Q) = \frac{1}{2} - \frac{1}{2}\,\mathbf{E}_{(x,y)\sim D} M_Q(x,y)$ gives a relation between $R_D(G_Q)$ and $M_Q(x,y)$.

Similarly as in Germain et al. (2009b), we only consider losses that upper-bound the zero-one loss of B_Q. Hence, we consider functions $\zeta : [-1,1] \to \mathbb{R}$ of the form

$$\zeta(\alpha) = \sum_{k=0}^{\deg(\zeta)} a_k\, \alpha^k \;\; \text{such that } \zeta(\alpha) \geq I(-\alpha \leq 0)\,,$$

with $a_k \geq 0$. Then, we will provide PAC-Bayes bounds on the following expected loss

$$\zeta_D^Q \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim D} \zeta\left(-M_Q(x,y)\right), \tag{C.5}$$

based on its empirical (possibly biased) estimate $\zeta_S^Q \stackrel{\text{def}}{=} \mathbf{E}_{(x,y)\sim S}\, \zeta\left(-M_Q(x,y)\right)$. Such a ζ is called a *convex margin loss function* (or a *convex surrogate loss*). Since $\zeta(\alpha) \geq I(-\alpha \leq 0)$ we have

$$\zeta_D^Q \;\geq\; \mathop{\mathbf{E}}_{(x,y)\sim D} I(M_Q(x,y) \leq 0\,) \;\geq\; R_D(B_Q)\,.$$

Thus, ζ_D^Q is always an upper bound of $R_D(B_Q)$. In particular, the factor-of-two rule,

$$R_D(B_Q) \;\leq\; 2R_D(G_Q)\,,$$

simply corresponds to the case where $a_0 = a_1 = 1$, and $a_j = 0$ for all $j > 1$, as for these values, $\zeta_D^Q = 2R_D(G_Q)$.

The next theorem gives a bound on ζ_D^Q and, consequently, on $R_D(B_Q)$. It can be viewed as a generalization of Theorem 1.2.1 of Catoni (2007) to the sample compression setting and to general margin loss functions. (Proof provided in supplementary material, Section C.3.2.)

Theorem C.1. *For any D, any family $(\mathcal{H}^S)_{S\in D^m}$ of sets of sc-classifiers of size at most l, any prior $\mathcal{P}$, any $\delta \in (0,1]$, any $C_1 > 0$, and any margin loss function ζ such that $l \cdot \deg(\zeta) < m$, we have*

$$\Pr_{S\sim D^m}\left(\begin{array}{l} \forall\, Q \;\; on \;\; \mathcal{H}^S : \\ \zeta_D^Q \;\leq\; \zeta(1)[C'-1] + C' \cdot \left(\zeta_S^Q + \frac{2}{m\cdot C_1}[\zeta'(1)\cdot \mathrm{KL}(Q\|P) + \zeta(1)\cdot \ln\frac{1}{\delta}]\right) \end{array}\right) \geq 1-\delta\,,$$

where $\mathrm{KL}(\cdot\|\cdot)$ *is the Kullback-Leibler divergence, and where* $C' = \dfrac{C_1 \cdot \frac{m}{m - l\cdot\deg(\zeta)}}{1 - e^{-C_1 \cdot \frac{m - l\cdot\deg(\zeta)}{m}}}$.

The bound of Theorem C.1 holds for any constant C_1. Even if the bound can be made valid uniformly for k different values of C_1 by replacing δ with δ/k (thanks to the standard union bound argument), this is a disadvantage when one wants to make a bound minimization algorithm out of it because such an algorithm will have to tune this extra hyper-parameter. However, the next subsection shows that restricting Q to be an *aligned posterior* can provide a PAC-Bayes bound *independent* of the Kullback-Leibler divergence between the posterior and the prior. As a consequence, no constant C_1 will be present in the proposed bound. To our knowledge, this is new in PAC-Bayes theory.

C.2.1 The Case of Aligned Posteriors

To define the notion of aligned posteriors, we need to consider the boolean complement $-h^{\mu}_{S'}$ of any sc-classifier $h^{\mu}_{S'}$. Thus, we now consider that the message sets are of the form

$$\mathcal{M}_{S'} \;=\; \mathcal{M}^1_{S'} \times \{+,-\}\,,$$

and that we always have $h^{(\sigma,+)}_{S_{\mathbf{i}}} = -h^{(\sigma,-)}_{S_{\mathbf{i}}}\ \forall \sigma \in \mathcal{M}^1_{S_{\mathbf{i}}}$.

Definition C.2. Given a prior $\mathcal{P}$ and a training sequence S, a posterior Q is said to be *aligned* on P if $\;Q(h^{(\sigma,+)}_{S_{\mathbf{i}}}) + Q(h^{(\sigma,-)}_{S_{\mathbf{i}}}) = P(h^{(\sigma,+)}_{S_{\mathbf{i}}}) + P(h^{(\sigma,-)}_{S_{\mathbf{i}}})$ for all $(\mathbf{i},\sigma) \in \mathcal{I} \times \mathcal{M}^1_{S_{\mathbf{i}}}$.

Remark C.3. An aligned posterior is completely defined by the values of

$$w(\mathbf{i},\sigma) \;\stackrel{\text{def}}{=}\; Q(h^{(\sigma,+)}_{S_{\mathbf{i}}}) - Q(h^{(\sigma,-)}_{S_{\mathbf{i}}})\,, \tag{C.6}$$

under the constraints $|w(\mathbf{i},\sigma)| \leq \; P(h^{(\sigma,+)}_{S_{\mathbf{i}}}) + P(h^{(\sigma,-)}_{S_{\mathbf{i}}})$ for all $(\mathbf{i},\sigma) \in \mathcal{I} \times \mathcal{M}^1_{S_{\mathbf{i}}}$. Indeed, it immediately follows that Q can be recovered from $\mathcal{P}$ and w because

$$Q(h^{(\sigma,\pm)}_{S_{\mathbf{i}}}) \;=\; \frac{1}{2}\left(P(h^{(\sigma,+)}_{S_{\mathbf{i}}}) + P(h^{(\sigma,-)}_{S_{\mathbf{i}}}) \pm w(\mathbf{i},\sigma)\right). \tag{C.7}$$

Moreover, given any function $w : \mathcal{I} \times \mathcal{M}^1_{S_{\mathbf{i}}} \to \mathbb{R}$ satisfying the constraints following Equation (C.6) the function Q given by Equation (C.7) is a distribution aligned on P.

The next proposition follows directly from what precedes and points out that there is no loss of expressiveness for majority votes if we restrict ourselves to aligned posteriors.

Proposition C.4. *Let $\mathcal{P}$ be a prior, S a training sequence, and Q a distribution on $\mathcal{H}^S$ for which there exists $A > 0$ such that for all $\mathbf{i}$ and σ,*

$$A\,|Q(h^{(\sigma,+)}_{S_{\mathbf{i}}}) - Q(h^{(\sigma,-)}_{S_{\mathbf{i}}})| \leq P(h^{(\sigma,+)}_{S_{\mathbf{i}}}) + P(h^{(\sigma,-)}_{S_{\mathbf{i}}})\,.$$

Let Q' be a distribution aligned on P such that $w'(\mathbf{i},\sigma) = A\,(Q(h^{(\sigma,+)}_{S_{\mathbf{i}}}) - Q(h^{(\sigma,-)}_{S_{\mathbf{i}}}))$. Then Q' is Bayes-equivalent to Q (i.e., $B_{Q'}(x) = B_Q(x)\ \forall x \in \mathcal{X}$).

We now provide a PAC-Bayes bound for aligned posteriors which does not depend on how far is an aligned posterior from the prior.

Theorem C.5. *For any D, for any $m \geq 8$, for any family $(\mathcal{H}^S)_{S\in D^m}$ of sets of sc-classifiers of size at most l, for any prior $\mathcal{P}$, for any margin loss function ζ such that $l \cdot \deg(\zeta) < m$, and for any $\delta \in (0,1]$, we have*

$$\Pr_{S\sim D^m}\left(\begin{array}{l}\forall Q \text{ aligned on } P: \\ \zeta_D^Q \leq \zeta_S^Q + \frac{\zeta(1)}{\sqrt{\frac{1}{2}(m - l\deg\zeta)}}\sqrt{4\,l\deg\zeta + \ln\frac{2\sqrt{m}}{\delta}}\end{array}\right) \geq 1-\delta\,.$$

Proof. There are three main difficulties in this proof that prevents a straightforward reduction to a classical PAC-Bayes proof. The first difficulty is that we want to bound the general loss ζ_D^Q instead of the usual Gibbs' risk $R_D(G_Q)$. We overcome this first difficulty by defining a new family of "abstract" sc-classifiers whose Gibbs' risk is closely related to ζ_D^Q by Equation (C.9) below. The second difficulty comes from the fact that a part of the training data is used to construct the sc-classifiers and, consequently, the empirical risk ζ_S^Q provides a *biased* estimate of ζ_D^Q. This complicates a lot the evaluation of the expected value of $X_{\overline{P}}$ defined below, which is an essential step in all PAC-Bayes proofs. Claim 1 deals with this problem. Finally, in classical proofs, one can only relate $X_{\overline{P}}$, computed with distribution P, to $X_{\overline{Q}}$, computed with another distribution Q, at the cost of generating an extra term which is the Kullback-Leibler divergence between Q and P. Claim 2, below, shows that if Q is aligned on P, both random variables are the same. To our knowledge, this unexpected result is new in PAC-Bayes theory.

Let S be any training sequence and $d \stackrel{\text{def}}{=} \deg\zeta$. For each $k \in \{0,..,d\}$ and any k-tuple $(h_1,..,h_k)$, let us define $\overline{h}$ as an "abstract" sc-classifier $\overline{h_1..h_k}$ whose "abstract" true risk and empirical risk (resp. the cases where $U=D$ and $U=S$) are defined as

$$\overline{R}_U(\overline{h}) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim U}\frac{1}{2}\Big[1+\prod_{i=1}^{k}-yh_i(x)\Big]\,. \tag{C.8}$$

For the $k=0$ case, we have $\overline{R}_U(\overline{h})=\overline{R}_U(\overline{h_1..h_0})=1$.

For each S, let $\overline{\mathcal{H}^S}$ be the set of all such sc-classifiers. For each distribution P and Q on $\mathcal{H}^S$, denote by $\overline{P}$ and $\overline{Q}$, the following distributions on $\overline{\mathcal{H}^S}$:

$$\overline{P}(\overline{h}) \stackrel{\text{def}}{=} \frac{a_k}{\zeta(1)}\prod_{i=1}^{k}P(h_i) \quad \text{and} \quad \overline{Q}(\overline{h}) \stackrel{\text{def}}{=} \frac{a_k}{\zeta(1)}\prod_{i=1}^{k}Q(h_i)\,.$$

Since $\zeta(1)=\sum_{k=0}^{d}a_k$, both $\overline{P}$ and $\overline{Q}$ are probability distributions. Moreover, for $U=D$ and $U=S$, we have

$$\begin{aligned} R_U(G_{\overline{Q}}) &= \mathop{\mathbf{E}}_{\overline{h}\sim\overline{Q}}\overline{R}_U(\overline{h}) \\ &= \sum_{k=0}^{d}\frac{a_k}{\zeta(1)}\mathop{\mathbf{E}}_{h_1\sim Q}\cdots\mathop{\mathbf{E}}_{h_k\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim U}\frac{1}{2}\Big[1+\prod_{i=1}^{k}-yh_i(x)\Big] \end{aligned}$$

$$
\begin{aligned}
&= \sum_{k=0}^{d} \frac{a_k}{\zeta(1)} \mathop{\mathbf{E}}_{(x,y)\sim U} \frac{1}{2}\Big[1 + \prod_{i=1}^{k} \mathop{\mathbf{E}}_{h_i\sim Q} -yh_i(x)\Big] \\
&= \frac{1}{2}\Big[1 + \frac{1}{\zeta(1)} \mathop{\mathbf{E}}_{(x,y)\sim U} \sum_{k=0}^{d} a_k (\mathop{\mathbf{E}}_{h\sim Q} -yh(x))^k\Big] \\
&= \frac{1}{2}\Big[1 + \frac{1}{\zeta(1)}\ \zeta_U^Q\Big]\,. \qquad \text{(C.9)}
\end{aligned}
$$

Because the compression sequence size of each h_i is at most l, we have $|\mathbf{i}_{\overline{h}}| \leq l \cdot k$ for any $\overline{h} = \overline{h_1..h_k}$.

Similarly as McAllester (2003a), we consider the following Laplace transform:

$$
X_{\overline{P}} \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{\overline{h}\sim\overline{P}} e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2}. \qquad \text{(C.10)}
$$

In supplementary material (Section C.3.1), we prove the following:

Claim 1 : $\mathbf{E}_{S\sim D^m} X_{\overline{P}} \leq e^{4ld} \cdot 2\sqrt{m}\,.$

The fact that there is no KL-divergence in the bound is a consequence of the following claim.

Claim 2 : For any posterior Q aligned on P, we have

$$
X_{\overline{P}} = X_{\overline{Q}} \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{\overline{h}\sim\overline{Q}} e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2}.
$$

Proof of Claim 2. For each $k \in \{0,..,d\}$, define $\overline{\mathcal{H}^S_{(k)}}$ as the set of abstract classifiers $\overline{h}$ that are k-tuples $\overline{h_1..h_k}$. Now, for each $\overline{h} \in \overline{\mathcal{H}^S_{(k)}}$ and each $j = 0,..,2^k-1$, define $\overline{h}^{[j]} \stackrel{\text{def}}{=} \overline{h_1^{(s_1)}..h_k^{(s_k)}}$, where $s_1 s_2..s_k$ is the binary representation of the number j, and where $h^{(0)} = h$ and $h^{(1)} = -h$. For any $\overline{h} \in \overline{\mathcal{H}^S_{(k)}}$, let $\mathcal{G}(\overline{h})$ be the set of all $\overline{h}^{[j]}$ for the different choices of j. Note that, given any two $\overline{h}, \overline{h'} \in \overline{\mathcal{H}^S_{(k)}}$, both $\mathcal{G}(\overline{h})$ and $\mathcal{G}(\overline{h'})$ either coincide or are disjoint. They will coincide iff $\overline{h'} = \overline{h}^{[j]}$ for some j, and $\mathbf{i}_{\overline{h}^{[j]}} = \mathbf{i}_{\overline{h}}$. Moreover, if Q is aligned on P:

$$
\begin{aligned}
\sum_{j=0}^{2^k-1} \overline{P}(\overline{h}^{[j]}) &= \frac{a_k}{\zeta(1)} \sum_{\mathbf{s}\in\{0,1\}^k} \prod_{i=1}^{k} P\left(h_i^{(s_i)}\right) \\
&= \frac{a_k}{\zeta(1)} \prod_{i=1}^{k} \left[P\left(h_i^{(0)}\right) + P\left(h_i^{(1)}\right)\right] \\
&= \frac{a_k}{\zeta(1)} \prod_{i=1}^{k} \left[Q\left(h_i^{(0)}\right) + Q\left(h_i^{(1)}\right)\right] \\
&= \frac{a_k}{\zeta(1)} \sum_{\mathbf{s}\in\{0,1\}^k} \prod_{i=1}^{k} Q\left(h_i^{(s_i)}\right) = \sum_{j=0}^{2^k-1} \overline{Q}(\overline{h}^{[j]})\,. \qquad \text{(C.11)}
\end{aligned}
$$

Also, it follows directly from Equation (C.8) and the property $(q-p)^2 = ((1-q)-(1-p))^2$ that

$$
(\overline{R}_S(\overline{h}) - \overline{R}_D(\overline{h}))^2 = (\overline{R}_S(\overline{h}^{[j]}) - \overline{R}_D(\overline{h}^{[j]}))^2\,. \qquad \text{(C.12)}
$$

From Equations (C.11) and (C.12), we now have

$$\begin{aligned}
&\int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \overline{P}(\overline{h}) e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2} \\
&\quad = \frac{1}{2^k}\sum_{j=0}^{2^k-1}\int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \overline{P}(\overline{h}) e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2} \\
&\quad = \frac{1}{2^k}\sum_{j=0}^{2^k-1}\int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \overline{P}(\overline{h}^{[j]}) e^{(m-|\mathbf{i}_{\overline{h}^{[j]}}|)\cdot 2(\overline{R}_S(\overline{h}^{[j]})-\overline{R}_D(\overline{h}^{[j]}))^2} \\
&\quad = \frac{1}{2^k}\sum_{j=0}^{2^k-1}\int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \overline{P}(\overline{h}^{[j]}) e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2} \\
&\quad = \frac{1}{2^k}\int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \sum_{j=0}^{2^k-1}\overline{P}(\overline{h}^{[j]}) e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2} \\
&\quad = \frac{1}{2^k}\int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \sum_{j=0}^{2^k-1}\overline{Q}(\overline{h}^{[j]})\ e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2} \\
&\quad \vdots \\
&\quad = \int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \overline{Q}(\overline{h}) e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2}\,. \qquad \text{(C.13)}
\end{aligned}$$

Thus, the following proves Claim 2:

$$\begin{aligned}
&\mathop{\mathbf{E}}_{\overline{h}\sim\overline{P}} e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2} \\
&\quad = \sum_{k=0}^{\deg\zeta}\int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \overline{P}(\overline{h}) e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2} \\
&\quad = \sum_{k=0}^{\deg\zeta}\int_{\overline{h}\in\overline{\mathcal{H}^S_{(k)}}} \overline{Q}(\overline{h}) e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2} \\
&\quad = \mathop{\mathbf{E}}_{\overline{h}\sim\overline{Q}} e^{(m-|\mathbf{i}_{\overline{h}}|)\cdot 2(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2}\,.
\end{aligned}$$

Now, by Markov's inequality we have

$$\Pr_{S\sim D^m}\left(X_{\overline{P}} \le \frac{1}{\delta}\ \mathop{\mathbf{E}}_{S\sim D^m} X_{\overline{P}}\right) \ge 1-\delta\,.$$

Now, by applying the two claims and taking the logarithm on each side of the innermost inequality, we have

$$\Pr_{S\sim D^m}\left(\begin{array}{l} \forall\, Q \textit{ aligned on } P\ : \\ \ln X_{\overline{Q}}\ \le\ \ln\left[\frac{1}{\delta}\ e^{4ld}\cdot 2\sqrt{m}\right] \end{array}\right)\ \ge\ 1-\delta\,. \qquad \text{(C.14)}$$

By using Jensen's inequality on the concavity of $\ln(x)$ and on the convexity of $(q-p)^2$, and by using the fact that $m-|\mathbf{i}_{\overline{h}}|\ \ge\ m-l\cdot d$, we find

$$\begin{aligned}
\ln X_{\overline{Q}}\ &\ge\ \mathop{\mathbf{E}}_{\overline{h}\sim\overline{Q}}(m-|\mathbf{i}_{\overline{h}}|)\cdot 2\cdot(\overline{R}_S(\overline{h})-\overline{R}_D(\overline{h}))^2 \\
&\ge\ (m-ld)\cdot 2\cdot\left(\mathop{\mathbf{E}}_{\overline{h}\sim\overline{Q}}\overline{R}_S(\overline{h})-\mathop{\mathbf{E}}_{\overline{h}\sim\overline{Q}}\overline{R}_D(\overline{h})\right)^2.
\end{aligned}$$

The theorem then follows from Equations (C.9) and (C.14), and straightforward calculations. □

C.2.2 Majority Votes of sc-classifiers of Compression Size of at Most One

For the rest of the paper, we specialize ourselves to the case where each sc-classifier has a compression set size of at most one. In that case, each sample compression sequence $S_{\mathbf{i}}$ consists of at most a single training example and, consequently, each possible vector $\mathbf{i}$ has at most only one index (*i.e.*, $|\mathbf{i}| \leq 1$). When $|\mathbf{i}| = 1$ and its single index points to example (x_i, y_i) of S, we have $\mathbf{i} = \langle i \rangle$ and $S_{\mathbf{i}} = S_{\langle i \rangle} = (x_i, y_i)$. When $|\mathbf{i}| = 0$, then $\mathbf{i} = \langle\rangle$ and $S_{\mathbf{i}} = S_{\langle\rangle} = \emptyset$. In this latter case, the two sc-classifiers $h^{(\varepsilon,+)}_{S_{\langle\rangle}}$ and $h^{(\varepsilon,-)}_{S_{\langle\rangle}}$ are constant classifiers so that $h^{(\varepsilon,+)}_{S_{\langle\rangle}}(x) = +1$ and $h^{(\varepsilon,-)}_{S_{\langle\rangle}}(x) = -1$ for all $x \in \mathcal{X}$. Here ε denotes the empty message. Each sc-classifier $h^{(\sigma,s)}_{S_{\langle i \rangle}}$ (that we will define below) of compression size 1 uses a message $(\sigma, s) \in \mathcal{M}^1 \times \{-,+\}$ where $\mathcal{M}^1$ is a real interval having a length denoted by $|\mathcal{M}^1|$. Furthermore, we use a *uniform prior* over all the relevant parameters. More precisely, for all $i \in \{1, \ldots, m\}$, and $s \in \{-,+\}$, we have

$$P_{\mathcal{I}}(\langle\rangle) \;=\; P_{\mathcal{I}}(\langle i \rangle) \;=\; \frac{1}{m+1} \quad ; \quad P_{S_{\langle\rangle}}(\varepsilon, s) \;=\; \frac{1}{2} \quad ; \quad P_{S_{\langle i \rangle}}(\sigma, s) \;=\; \frac{1}{2|\mathcal{M}^1|} I(\sigma \in \mathcal{M}^1)\,.$$

Equation (C.3) implies that $P(h^{(\varepsilon,s)}_{S_{\langle\rangle}}) = P_{\mathcal{I}}(\langle\rangle) P_{S_{\langle\rangle}}(\varepsilon, s)$ and $P(h^{(\sigma,s)}_{S_{\langle i \rangle}}) = P_{\mathcal{I}}(\langle i \rangle) P_{S_{\langle i \rangle}}(\sigma, s)$. We do have

$$\sum_{s \in \{-,+\}} P(h^{(\varepsilon,s)}_{S_{\langle\rangle}}) + \sum_{i=1}^{m} \sum_{s \in \{-,+\}} \int_{\mathcal{M}^1} d\sigma P(h^{(\sigma,s)}_{S_{\langle i \rangle}}) \;=\; 1\,.$$

In the rest of the paper, we restrict ourselves to what we call *strongly aligned posteriors*, *i.e.*, aligned posteriors such that the function $w(\mathbf{i}, \sigma)$ is constant on its second argument. More precisely, for sc-classifiers of compression size zero, we use

$$\begin{aligned} Q(h^{(\varepsilon,s)}_{S_{\langle\rangle}}) \;&=\; \frac{1}{2}\left(P(h^{(\varepsilon,+)}_{S_{\langle\rangle}}) + P(h^{(\varepsilon,-)}_{S_{\langle\rangle}}) + s \cdot w(\langle\rangle, \varepsilon)\right) \\ &=\; \frac{1}{2}\left(\frac{1}{m+1} + s \cdot w_0\right), \end{aligned} \tag{C.15}$$

where $w_0 \stackrel{\text{def}}{=} w(\langle\rangle, \varepsilon)$ and must satisfy $|w_0| \leq \frac{1}{m+1}$.

For sc-classifiers of compression size one, we use

$$\begin{aligned} Q(h^{(\sigma,s)}_{S_{\langle i \rangle}}) \;&=\; \frac{1}{2}\left(P(h^{(\sigma,+)}_{S_{\langle i \rangle}}) + P(h^{(\sigma,-)}_{S_{\langle i \rangle}}) + s \cdot w(\langle i \rangle, \sigma)\right) \\ &=\; \frac{1}{2}\left(\frac{1}{m+1} + s \cdot w_i\right) \frac{1}{|\mathcal{M}^1|} I(\sigma \in \mathcal{M}^1)\,, \end{aligned} \tag{C.16}$$

where we have defined w_i by the equality $w(\langle i \rangle, \sigma) = w_i \frac{1}{|\mathcal{M}^1|} I(\sigma \in \mathcal{M}^1)$. Thus, we must satisfy $|w_i| \leq \frac{1}{m+1}$.

The specialization to strongly aligned posterior might seem too restrictive. However, this setting remains powerful enough to include kernels methods such as the SVM as a *special case*. Indeed, for any such strongly aligned posterior Q, the output of $B_Q(x)$ is

$$B_Q(x) \;=\; \operatorname{sgn}\left(w_0 + \sum_{i=1}^{m} w_i k(x_i, x)\right), \tag{C.17}$$

since $\mathbf{E}_{h\sim Q}\, h(x)$, in Equation (C.2), is given by

$$\begin{aligned}
&\sum_{s\in\{\text{-},+\}} Q(h^{(\varepsilon,s)}_{S_{\langle\rangle}}) h^{(\varepsilon,s)}_{S_{\langle\rangle}}(x) + \sum_{i=1}^{m} \sum_{s\in\{\text{-},+\}} \int_{\mathcal{M}^1} d\sigma Q(h^{(\sigma,s)}_{S_{\langle i\rangle}}) h^{(\sigma,s)}_{S_{\langle i\rangle}}(x) \\
&\qquad = \quad w_0 + \sum_{i=1}^{m} w_i \frac{1}{|\mathcal{M}^1|} \int_{\mathcal{M}^1} h^{(\sigma,+)}_{S_{\langle i\rangle}}(x) d\sigma \\
&\qquad = \quad w_0 + \sum_{i=1}^{m} w_i k(x_i, x)\,.
\end{aligned}$$

The last equality is obtained whenever $h^{(\sigma,+)}_{S_{\langle i\rangle}}$ satisfies $\int_{\mathcal{M}^1} h^{(\sigma,+)}_{S_{\langle i\rangle}}(x) d\sigma \;=\; |\mathcal{M}^1| k(x_i, x)$. Hence, we choose

$$h^{(\sigma,+)}_{S_{\langle i\rangle}}(x) \;=\; \operatorname{sgn}\left(\frac{1}{2}|\mathcal{M}^1| k(x_i, x) > \sigma\right) \quad \forall x \in \mathcal{X}\,, \tag{C.18}$$

for $\sigma \in \mathcal{M}^1 = [-1, +1]$ and $k(x', x) \leq 1\ \forall (x', x) \in \mathcal{X}^2$. This last condition implies that k must be bounded by 1. However, note that no other condition need to be satisfied for k. Indeed, k can be any normalized similarity measure; it does not even need to be symmetric.

If we compare the set of majority-vote classifiers described by Equation (C.17) to the set of SVM classifiers where the output $f_{\text{SVM}}(x)$ for any $x \in \mathcal{X}$ is given by

$$f_{\text{SVM}}(x) \;=\; \operatorname{sgn}\Big(\sum_{i=1}^{m} y_i \alpha_i k(x_i, x) + b\Big)\,,$$

we conclude that the latter set forms a strict subset of the former set. Indeed, even if for B_Q we must have $k(x', x) \leq 1\ \forall (x', x) \in \mathcal{X}^2$ and $|w_i| \leq \frac{1}{m+1}$ for all i, while no such restriction exists for f_{SVM}, we can always multiply b and each α_i by a positive constant such that $f_{\text{SVM}}(x) = B_Q(x)\ \forall x \in \mathcal{X}$. However, k in B_Q can be any similarity measure (possibly not symmetric in its two arguments), while k in f_{SVM} must be a positive semi-definite kernel. Note that Theorems C.1 and C.5 do apply to this larger class of majority votes of sc-classifiers of compression size of at most one.

Several generalizations from the above are possible. Indeed, for $Q(h^{(\sigma,s)}_{S_{\langle i\rangle}})$, we could consider distributions over σ that vary with i. This would effectively provide a mechanism for adapting the similarity measure to each training example. We could also use sc-classifiers having a compression size larger than one.

C.2.3 Bound Minimization Learning Algorithms

For the task of finding the posterior Q minimizing an upper bound on ζ_D^Q, note that Theorems C.1 and C.5 indicate that $l \cdot \deg(\zeta)$ should be small for the risk bound to be small. Hence, we consider sc-classifiers of compression sequence size of at most one and margin losses that are quadratic[1], *i.e.*, $\zeta(\alpha) = (1 + \frac{1}{q}\,\alpha)^2$. Hence, we have $\zeta(1) = (1 + q^{-1})^2$ and $\zeta'(1) = (2q+2)/q^2$. Algorithm *PBSC-A* finds a strongly aligned Q minimizing ζ_S^Q and, thus, the bound of Theorem C.5. Equations (C.15) and (C.16) show that any strongly aligned Q is determined by $\mathbf{w} \stackrel{\text{def}}{=} (w_0, w_1, \ldots, w_m)$. When $\mathcal{H}^S$ consists of sc-classifiers of compression sequence size of at most one, as defined in Section C.2.2, the margin of Q on (x, y) is $M_Q(x, y) = y\,[\,w_0 + \sum_{i=1}^m w_i k(x_i, x)\,]$.

Thus, *PBSC-A* minimizes

$$\begin{aligned}\zeta_S^Q &= \frac{1}{m}\sum_{j=1}^m \zeta\left(-M_Q(x_j, y_j)\right)^2 \\ &= \frac{1}{mq^2}\sum_{j=1}^m \left(q - y_j\left[w_0 + \sum_{i=1}^m w_i k(x_i, x_j)\right]\right)^2,\end{aligned}$$

subject to $|w_i| \leq \frac{1}{m+1}$ for $i = 1, \ldots, m$. Hence, *PBSC-A* solves a least-square problem under a ℓ_∞ norm constraint.

The objective function to minimize is convex in Q (with continuous first derivatives) and the domain of all strongly aligned posteriors is also convex. The set of solutions is therefore convex and coordinate descent methods, such as the one we have implemented (see Section C.3.3 for the details), are thus always guaranteed to converge to a solution.

Similarly, algorithm *PBSC-N* finds the posterior Q that minimizes the bound of theorem C.1, without restricting Q to be aligned to the prior. It considers the same set of sc-classifiers of size one defined previously. However, the non-aligned posterior Q is defined by

$$\mathbf{v} \stackrel{\text{def}}{=} (v_+, v_1, \ldots, v_{2m}, v_-)\,,$$

such that:

$$\begin{aligned} Q(h^{(\varepsilon,+)}_{S_{\langle\rangle}}) &= v_+\,, & Q(h^{(\sigma,+)}_{S_{\langle i\rangle}}) &= v_i \tfrac{1}{|\mathcal{M}^1|} I(\sigma \in \mathcal{M}^1)\,, \\ Q(h^{(\varepsilon,-)}_{S_{\langle\rangle}}) &= v_-\,, & Q(h^{(\sigma,-)}_{S_{\langle i\rangle}}) &= v_{m+i} \tfrac{1}{|\mathcal{M}^1|} I(\sigma \in \mathcal{M}^1)\,, \end{aligned}$$

where $i \in \{1, \ldots, m\}$. For Q to be a distribution, $\mathbf{v}$ must satisfy

$$v \geq 0 \text{ for all } v \in \mathbf{v} \quad \text{and} \quad \sum_{v \in \mathbf{v}} v = 1\,. \tag{C.19}$$

Note that the posterior Q minimizing the bound of Theorem C.1 is the one that minimizes $C \cdot \zeta_S^Q + \mathrm{KL}(Q\|P)$, subject to the constraints of Equation (C.19). The parameter $C > 0$

1. The sc-classifiers being weak, we have seen that linear margin losses are not suitable for model selection.

allow us to tune the influence of the regularizer. However, in Theorem C.1, C is a constant obtained from C_1, $\zeta'(1)$ and m.

To express ζ_S^Q in terms of $\mathbf{v}$, note that the margin $M_Q(x,y)$, on example (x,y), is given by

$$M_Q(x,y) \;=\; y\left[(v_+ - v_-) + \sum_{i=1}^{m}(v_i - v_{i+m})k(x_i,x)\right].$$

Consequently, we have

$$\zeta_S^Q \;=\; \frac{1}{mq^2}\sum_{j=1}^{m}\left(q - y_j\left[v_+ - v_- + \sum_{i=1}^{m}(v_i - v_{i+m})k(x_i,x_j)\right]\right)^2.$$

Moreover, the KL-divergence between Q and a uniform prior P is given by

$$\mathrm{KL}(Q\|P) \;=\; \mathop{\mathbf{E}}_{h\sim Q}\ln\frac{Q(h)}{P(h)} = \ln(2m+2) + \sum_{v\in\mathbf{v}} v\ln v\,.$$

The optimization problem for *PBSC-N* is thus a KL-regularized least-square problem. The objective function to minimize is convex in $\mathbf{v}$ (with continuous first derivatives) and $\mathbf{v}$ lies in a convex domain. A coordinate-pair descent algorithm that works iteratively exchanging weights between two components of $\mathbf{v}$, such as the one we have implemented (see Section C.3.3 for details), is thus assured to converge to a global optimum.

C.2.4 Empirical Results

We first compare both PBSC algorithms to the popular SVM, to the Regularized Least Squares Classifier (RLSC) [2] , and to an algorithm, denoted here as LINEAR, that just consists at minimizing ζ_S^Q for the linear margin loss function $\zeta(\alpha) = \alpha + 1$. The comparison with the latter is only to point out the need of a non-linear margin loss function. Note that *PBSC-N* has two hyper-parameters (C, q) to tune whereas *PBSC-A* needs only one (q). SVM and RLSC also need to tune only one hyper-parameter (C).

For all algorithms, we used the standard RBF kernel $k_{\mathrm{RBF}}(x,x') = \exp(-\gamma\|x-x'\|)$ and the sigmoid kernel $k_{\mathrm{SIG}}(x,x') = \tanh(s\,x\cdot x' + d)$. All hyper-parameters ($C$, q, γ, s, d) were determined by performing 10-fold cross validation on the training data. The experiments were performed on 22 data sets that, except for MNIST, were taken from the UCI repository. Each data set was randomly partitioned into a training set S of size $|S|$ and a testing set T of size $|T|$.

Table C.1 shows that, when using the RBF kernel, *PBSC-A* is very competitive with *PBSC-N*, SVM, and RLSC, and outperforms LINEAR. More precisely, *PBSC-A* has a better test risk than SVM on 11 datasets (the opposite occurs 6 times), and *PBSC-A* has a better test risk

2. RLSC also minimizes a quadratic loss function but with a regularizer different from the one used by PBSC.

Table C.1 – Empirical risk measured on the testing set T for the five different algorithms.

Dataset			RBF kernel					Sigmoid kernel	
Name	$\|T\|$	$\|S\|$	**SVM**	**RLSC**	**PBSC-A**	**PBSC-N**	**LINEAR**	**SVM**	**PBSC-A**
Adult	10000	1809	0.158	0.157	**0.156**	0.160	0.193	0.163	**0.157**
BreastC	340	343	**0.038**	**0.038**	0.044	**0.038**	0.144	**0.038**	**0.038**
Credit-A	300	353	0.190	0.160	**0.140**	0.173	0.200	0.190	**0.170**
Glass	107	107	**0.150**	**0.150**	**0.150**	0.168	0.187	**0.355**	0.411
Haberman	150	144	**0.267**	0.327	0.280	**0.267**	**0.267**	**0.273**	**0.273**
Heart	147	150	**0.197**	0.211	0.204	0.218	0.238	**0.184**	0.197
Ionosphere	175	176	0.057	**0.023**	0.040	0.040	0.326	0.126	**0.091**
Letter:AB	1055	500	**0.001**	0.002	**0.001**	**0.001**	0.038	0.009	**0.005**
Letter:DO	1058	500	0.014	0.015	**0.011**	0.012	0.069	**0.022**	0.028
Letter:OQ	1036	500	0.016	**0.011**	0.016	0.014	0.123	**0.018**	0.039
Liver	175	170	0.286	0.291	**0.280**	0.286	0.349	**0.400**	**0.400**
Mnist:0vs8	1916	500	**0.003**	0.009	0.004	0.004	0.031	0.007	**0.003**
Mnist:1vs7	1922	500	0.014	0.008	0.008	**0.007**	0.161	0.012	**0.007**
Mnist:1vs8	1936	500	0.011	**0.010**	**0.010**	0.011	0.292	**0.014**	0.015
Mnist:2vs3	1905	500	0.020	0.022	**0.019**	0.020	0.114	**0.025**	0.031
Mushroom	4062	4062	**0.000**	**0.000**	**0.000**	**0.000**	0.022	**0.000**	0.010
Ringnorm	3700	3700	0.015	0.017	**0.013**	**0.013**	0.103	**0.020**	0.035
sonar	104	104	0.154	0.250	**0.125**	0.192	0.490	0.250	**0.183**
Tic-tac-toe	479	479	**0.015**	0.019	0.019	0.052	0.365	**0.023**	0.159
Usvotes	200	235	0.075	**0.065**	**0.065**	**0.065**	0.140	0.070	**0.065**
Waveform	4000	4000	0.068	0.067	0.068	**0.066**	0.143	**0.067**	**0.067**
Wdbc	284	285	**0.042**	0.067	0.049	0.074	0.180	**0.366**	**0.366**

than RLSC on 12 datasets (the opposite occurs 4 times). Using the sign test methodology of Mendenhall (1983), this yields to a *p-value* of 16.6% for the hypothesis that *PBSC-A* and SVM algorithms are equivalent and a *p-value* of 3.8% for the hypothesis that *PBSC-A* and RLSC algorithms are equivalent. By using the common significance threshold of 5%, we conclude that *PBSC-A* is better than RLSC whereas no such conclusion holds for *PBSC-A* vs SVM. We also have a *p-value* of 10.5% when comparing *PBSC-A* and *PBSC-N*. Note that *PBSC-A* has a better practical value than *PBSC-N* as it requires one hyper-parameter less. Finally, we have a *p-value* of 0.0005% when comparing *PBSC-A* with LINEAR, supporting strongly the choice of the quadratic loss over the linear loss for algorithm design.

Unlike the RBF kernel, the sigmoid kernel is indefinite for certain parameter values. In this case, the standard SVM algorithm might not converge to a solution (like the popular SVM-Light implementation). In our experiments, we used the LIBSVM implementation of Chang et Lin (2011) because it returns a solution even if the kernel is indefinite. It turns out that *PBSC-A* and LIBSVM are competitive. Indeed, the sign test methodology gives a *p-value* of 50% for the hypothesis that both algorithms are equivalents.

To pursue the exploration with indefinite similarity measures, we ran *PBSC-A* on four binary data sets referenced by Chen et al. (2009b,a) and used the provided similarity measures. We followed the same experimental methodology as Chen et al. (2009b,a) and computed the mean and standard deviation of the empirical risk across 20 test/training standardized partitions. Table C.2 shows that *PBSC-A* is competitive with the Linear SVM using similarities as

Table C.2 – Mean and standard deviation (in parentheses) of the empirical risk across 20 partitions.

Dataset	Linear SVM		k-NN		PBSC-A	
Aural Sonar	**0.1425**	(0.694)	0.1825	(0.597)	0.1500	(0.827)
Voting	0.0534	(0.193)	0.0546	(0.174)	**0.0529**	(0.184)
Yeast-5-7	**0.2688**	(0.622)	0.3063	(0.580)	0.2975	(0.668)
Yeast-5-12	**0.1075**	(0.482)	0.1275	(0.439)	0.1088	(0.598)

features and is better than the k-Nearest Neighbor using similarities as a measure of distance. Note that Chen et al. (2009a) suggests an algorithm having generally better achievements on these data sets. But these results are obtained by substituting a "surrogate kernel function" to the real similarity function that one wants to use.

C.3 Supplementary Material

C.3.1 Proof of Claim 1 in the proof of Theorem C.5

Claim 1 : $\mathbf{E}_{S\sim D^m} X_{\overline{P}} \;\leq\; e^{4ld}\cdot 2\sqrt{m}\,.$

Proof. Let us first recall that

$$X_{\overline{P}} \;\stackrel{\text{def}}{=}\; \mathop{\mathbf{E}}_{\bar{h}\sim\overline{P}} e^{(m-|\mathbf{i}_{\bar{h}}|)\cdot 2(\overline{R}_S(\bar{h})-\overline{R}_D(\bar{h}))^2}.$$

Let us also introduce a new notation. Let $\widetilde{R}_S(\bar{h})$ be the *abstract empirical risk* computed on the examples of S that *are not* in the compression sequence of $\bar{h}$. More formally,

$$\widetilde{R}_S(\,\overline{h_1..h_k}\,) \;\stackrel{\text{def}}{=}\; \frac{1}{m-|\mathbf{i}_{\overline{h_1..h_k}}|}\sum_{j=1}^{m} I\left(\neg\veebar_{i=1}^{k}(h_i(x_j)\neq y_j)\right) I\left((x_j,y_j)\notin \mathbf{i}_{\overline{h_1..h_k}}\right),$$

where $\veebar$ denotes the exclusive or. Based on the definition of Equation (9) in the main paper, one can easily show that $\widetilde{R}_S(\bar{h}) = \overline{R}_U(\bar{h})$ with $U = S\backslash S_{\mathbf{i}_{\bar{h}}}$, where $\mathbf{i}_{\bar{h}}$ points to the examples belonging to the union of all compression sequences of the sc-classifiers in $\bar{h}$. Moreover, note that, contrarily to $\overline{R}_S(\bar{h})$ which may contain some bias, $\widetilde{R}_S(\bar{h})$ is an arithmetic mean of truly iid $(m-|\mathbf{i}_{\bar{h}}|)$ random variables. On another hand, these two random variables have values that are very close to each other. Indeed, since

$$0 \;\leq\; m\cdot\overline{R}_S(\bar{h}) - (m-|\mathbf{i}_{\bar{h}}|)\cdot\widetilde{R}_S(\bar{h}) \;\leq\; |\mathbf{i}_{\bar{h}}|\,,$$

we have

$$-|\mathbf{i}_{\bar{h}}| \;\leq\; -|\mathbf{i}_{\bar{h}}|\cdot\widetilde{R}_S(\bar{h}) \;\leq\; m\cdot\overline{R}_S(\bar{h}) - m\cdot\widetilde{R}_S(\bar{h}) \;\leq\; |\mathbf{i}_{\bar{h}}| - |\mathbf{i}_{\bar{h}}|\cdot\widetilde{R}_S(\bar{h}) \;\leq\; |\mathbf{i}_{\bar{h}}|\,.$$

Therefore,

$$\left|\overline{R}_S(\bar{h}) - \widetilde{R}_S(\bar{h})\right| \;\leq\; \frac{|\mathbf{i}_{\bar{h}}|}{m}\,. \tag{C.20}$$

Given a compression sequence $S_{\mathbf{i}}$, let $\mathbf{i}^c$ be the vector of indices of $\mathcal{I}$ that are not in the vector $\mathbf{i}$. We now have

$$\begin{aligned}\mathbf{E}_{S\sim D^m} X_{\overline{P}} &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{\bar{h}\sim\overline{P}} e^{(m-|\mathbf{i}_{\bar{h}}|)\cdot 2(\overline{R}_S(\bar{h})-\overline{R}_D(\bar{h}))^2} \\ &= \mathop{\mathbf{E}}_{\mathbf{i}\sim\overline{P}_{\mathcal{I}}}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}}\sim D^{|\mathbf{i}|}}\ \mathop{\mathbf{E}}_{\bar{\mu}\sim\overline{P}_{S_{\mathbf{i}}}}\ \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{m-|\mathbf{i}|}} e^{(m-|\mathbf{i}|)\cdot 2\,(\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))^2}\,,\end{aligned}$$

Hence, to prove Claim 1, it suffices to show that for all $\mathbf{i}\in\mathcal{I},\ S_{\mathbf{i}}\in(\mathcal{X}\times\mathcal{Y})^{|\mathbf{i}|},$ and $\mu\in\mathcal{M}_{S_{\mathbf{i}}}$, we have

$$\mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{m-|\mathbf{i}|}} e^{(m-|\mathbf{i}|)\cdot 2\,(\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))^2} \;\leq\; e^{4ld}\cdot 2\sqrt{m}\,. \tag{C.21}$$

Here is the sketch of the proof of Equation (C.21). Justification for Line (C.22) to (C.25) follows below.

$$\begin{aligned}&\mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{(m-|\mathbf{i}|)}} e^{(m-|\mathbf{i}|)\cdot 2\,(\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))^2} \\ &= \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{(m-|\mathbf{i}|)}} e^{(m-|\mathbf{i}|)\cdot 2\left(\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})+\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right)^2} \\ &\leq \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{(m-|\mathbf{i}|)}} e^{(m-|\mathbf{i}|)\cdot 2\left(\left[\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right]^2+2\left|\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right|\cdot\left|\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right|+\left[\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right]^2\right)} \\ &\leq \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{(m-|\mathbf{i}|)}} e^{(m-|\mathbf{i}|)\cdot 2\left(\left[\frac{|\mathbf{i}|}{m}\right]^2+2\frac{|\mathbf{i}|}{m}\cdot 1+\left[\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right]^2\right)} \qquad \text{(C.22)} \\ &\leq \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{(m-|\mathbf{i}|)}} e^{4ld\,+\,(m-|\mathbf{i}|)\cdot 2\left[\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right]^2} \qquad \text{(C.23)} \\ &\leq \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{(m-|\mathbf{i}|)}} e^{4ld\,+\,(m-|\mathbf{i}|)\cdot \mathrm{kl}\left(\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})\|\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right)} \qquad \text{(C.24)} \\ &= e^{4ld}\ \cdot \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{(m-|\mathbf{i}|)}} e^{(m-|\mathbf{i}|)\,\cdot\,\mathrm{kl}\left(\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})\|\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right)} \\ &\leq e^{4ld}\ \cdot 2\sqrt{m-|\mathbf{i}|} \qquad \text{(C.25)} \\ &\leq e^{4ld}\ \cdot 2\sqrt{m}\,.\end{aligned}$$

Line (C.22) follows from Equation (C.20) and the fact that the exponential function is increasing. For Line (C.23), since $|\mathbf{i}|\leq ld$, we simply have to show that $(m-|\mathbf{i}|)\cdot 2\left(\frac{|\mathbf{i}|}{m^2}+\frac{2}{m}\right)\leq 4\,,$ which follows from direct calculations. Line (C.24) follows directly from the property : $2(q-p)^2\leq \mathrm{kl}(q\parallel p)$, which can also be proved via straightforward calculations. Finally, for Line (C.25), first observe that $\widetilde{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})$ is an arithmetic mean of $(m-|\mathbf{i}|)$ iid random

variables. Thus, Line (C.25) is simply an application of Lemma A.11 (Maurer, 2004) with $M(\mathbf{X})$ replaced by $\widetilde{R}_S(h^{\bar{\mu}}_{S_\mathbf{i}})$, n replaced by $m-|\mathbf{i}|$, and μ replaced by $\overline{R}_D(h^{\bar{\mu}}_{S_\mathbf{i}})$. Thus, Claim 1 is proved. □

C.3.2 Proof of Theorem C.1: The PAC-Bayes bound with $\mathrm{KL}(Q\|P)$

Proof. Let S be any training sequence, $d \stackrel{\text{def}}{=} \deg\zeta$. Let $\mathcal{F}$ be a convex function to be defined later, and $\Delta(q,p) \stackrel{\text{def}}{=} \mathcal{F}(p) - C_1 \cdot q$.

For each $k \in \{0,..,d\}$ and any k-tupple $(h_1,..,h_k)$, let us define $\bar{h} = \overline{h_1..h_k}$ as an "abstract" sc-classifier whose "abstract" true and empirical risks are respectively defined as

$$\begin{aligned}
\overline{R}_D(\,\overline{h_1..h_k}\,) &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim D} I(\neg\underline{\vee}_{i=1}^k(h_i(x)\neq y))\,,\\
\overline{R}_S(\,\overline{h_1..h_k}\,) &\stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(x,y)\sim S} I(\neg\underline{\vee}_{i=1}^k(h_i(x)\neq y)) \;=\; \frac{1}{m}\sum_{j=1}^m I(\neg\underline{\vee}_{i=1}^k(h_i(x_j)\neq y_j))\,,
\end{aligned}$$

where $\underline{\vee}$ denotes the exclusive or.

Considering the above definitions, we also have

$$\overline{R}_D(\,\overline{h_1..h_k}\,) \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D} I(\neg\underline{\vee}_{i=1}^k(h_i(x)\neq y)) \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D}\frac{1}{2}\Big[1+\prod_{i=1}^k -yh_i(x)\Big]\,.$$

Since the compression sequence size of each h_i's is at most l, we have $|\mathbf{i}_{\bar{h}}| \leq l\cdot k$ for any $\bar{h} = \overline{h_1..h_k}$. Moreover, we have $\overline{R}_D(\overline{h_1..h_0}) = 1$ when $k=0$ because the exclusive or over an empty sequence outputs *false*. For each S, let $\overline{\mathcal{H}^S}$ be the set of all such sc-classifiers, and for each distribution Q on $\mathcal{H}^S$, denote by $\overline{Q}$ the following distribution on $\overline{\mathcal{H}^S}$:

$$\overline{Q}(\,\overline{h_1..h_k}\,) \;\stackrel{\text{def}}{=}\; \frac{a_k}{\zeta(1)}\,Q(h_1)\cdot\ldots\cdot Q(h_k) \;=\; \frac{a_k}{\zeta(1)}\prod_{i=1}^k Q(h_i)\,.$$

Since $\zeta(1) = \sum_{k=0}^d a_k$, $\overline{Q}$ is a probability distribution. We also denote by $\overline{P}$ the following distribution on $\overline{\mathcal{H}^S}$:

$$\overline{P}(\overline{h_1..h_k}) \;\stackrel{\text{def}}{=}\; \frac{a_k}{\zeta(1)}\cdot P(h1)\cdot\ldots\cdot P(h_k) \;=\; \frac{a_k}{\zeta(1)}\prod_{i=1}^k P(h_i)\,.$$

Moreover, for $U=D$ and $U=S$, we have

$$\begin{aligned}
R_U(G_{\overline{Q}}) &= \mathop{\mathbf{E}}_{\bar{h}\sim\overline{Q}}\overline{R}_U(\bar{h})\\
&= \sum_{k=0}^{\deg(\zeta)}\frac{a_k}{\zeta(1)}\mathop{\mathbf{E}}_{h_1\sim Q}\cdots\mathop{\mathbf{E}}_{h_k\sim Q}\mathop{\mathbf{E}}_{(x,y)\sim D}\frac{1}{2}\Big[1+\prod_{i=1}^k -yh_i(x)\Big]
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{k=0}^{\deg(\zeta)} \frac{a_k}{\zeta(1)} \mathop{\mathbf{E}}_{(x,y)\sim D} \frac{1}{2}\Big[1 + \prod_{i=1}^{k} \mathop{\mathbf{E}}_{h_i\sim Q} -yh_i(x)\Big] \\
&= \frac{1}{2}\Big[1 + \frac{1}{\zeta(1)} \mathop{\mathbf{E}}_{(x,y)\sim D} \sum_{k=0}^{\deg(\zeta)} a_k (\mathop{\mathbf{E}}_{h\sim Q} -yh(x))^k\Big] \\
&= \frac{1}{2}\Big[1 + \frac{1}{\zeta(1)} \mathop{\mathbf{E}}_{(x,y)\sim D} \sum_{k=0}^{\deg(\zeta)} a_k (-M_Q(x,y))^k\Big] \\
&= \frac{1}{2}\Big[1 + \frac{1}{\zeta(1)}\ \zeta_U^Q\Big]\,. \qquad \text{(C.26)}
\end{aligned}
$$

Now, let us consider the following Laplace transform

$$
X_{\overline{P}} \ \stackrel{\text{def}}{=}\ \mathbf{E}_{\bar{h}\sim\overline{P}}\, e^{(m-|\mathbf{i}_{\bar{h}}|)\Delta(\overline{R}_S(\bar{h}),\overline{R}_D(\bar{h}))}\,.
$$

It then follows from Markov's inequality that

$$
\Pr_{S\sim D^m}\left(X_{\overline{P}} \le \frac{1}{\delta}\ \mathop{\mathbf{E}}_{S\sim D^m} X_{\overline{P}}\right) \ge 1-\delta\,.
$$

By taking the logarithm on each side of the innermost inequality and by transforming the expectation over $\overline{P}$ into an expectation over $\overline{Q}$, we obtain

$$
\Pr_{S\sim D^m}\left(\forall Q\colon \ln\left[\mathop{\mathbf{E}}_{\bar{h}\sim\overline{Q}} \frac{\overline{P}(\bar{h})}{\overline{Q}(\bar{h})} e^{(m-|\mathbf{i}_{\bar{h}}|)\Delta(\overline{R}_S(\bar{h}),\overline{R}_D(\bar{h}))}\right] \le \ln\left[\frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{\bar{h}\sim\overline{P}} e^{(m-|\mathbf{i}_{\bar{h}}|)\Delta(\overline{R}_S(\bar{h}),\overline{R}_D(\bar{h}))}\right]\right) \ge 1-\delta\,. \qquad \text{(C.27)}
$$

Since $\zeta'(1)=\sum_{k=1}^{d} k\cdot a_k$, by straightforward calculation, one can show that

$$
\mathop{\mathbf{E}}_{\bar{h}\sim\overline{Q}} \ln\frac{\overline{P}(\bar{h})}{\overline{Q}(\bar{h})} \ =\ -\frac{\zeta'(1)}{\zeta(1)}\cdot \mathrm{KL}(Q\|P)\,.
$$

This, together with Jensen's inequality applied to the concave $\ln(x)$, gives

$$
\begin{aligned}
&\ln\left[\mathop{\mathbf{E}}_{\bar{h}\sim\overline{Q}} \frac{\overline{P}(\bar{h})}{\overline{Q}(\bar{h})}\ e^{(m-|\mathbf{i}_{\bar{h}}|)\Delta(\overline{R}_S(\bar{h}),\overline{R}_D(\bar{h}))}\right] \\
&\qquad \ge\ -\frac{\zeta'(1)}{\zeta(1)}\cdot \mathrm{KL}(Q\|P) + \mathop{\mathbf{E}}_{\bar{h}\sim\overline{Q}} (m-|\mathbf{i}_{\bar{h}}|)\,\Delta(\overline{R}_S(\bar{h}),\overline{R}_D(\bar{h}))\,. \qquad \text{(C.28)}
\end{aligned}
$$

Again from the Jensen's inequality applied to the convex function $\mathcal{F}$, together with Equation (C.26), and the fact that $m - l\cdot d \le (m-|\mathbf{i}_{\bar{h}}|) \le m$, we obtain

$$
\begin{aligned}
&\mathop{\mathbf{E}}_{\bar{h}\sim\overline{Q}} (m-|\mathbf{i}_{\bar{h}}|)\ \Delta(\overline{R}_S(\bar{h}),\overline{R}_D(\bar{h})) \\
&\qquad \ge\ (m-ld)\mathop{\mathbf{E}}_{\bar{h}\sim\overline{Q}} \mathcal{F}(\overline{R}_D(\bar{h})) - m\mathop{\mathbf{E}}_{\bar{h}\sim\overline{Q}} C_1\cdot\overline{R}_S(\bar{h})) \\
&\qquad \ge\ (m-ld)\,\mathcal{F}\left(\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\ \zeta_D^Q\right]\right) - m\,C_1\cdot\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\ \zeta_S^Q\right]. \qquad \text{(C.29)}
\end{aligned}
$$

Let us analyze the value of $\mathbf{E}_{S\sim D^m} X_{\overline{P}}$, appearing in the right-hand side of the innermost inequality of Equation (C.27) . First, let us define $\mathbf{i}^c$ as the vector of indices of $\mathcal{I}$ that are not in the vector $\mathbf{i}$. Thus, $|\mathbf{i}^c| = m - |\mathbf{i}|$. Now, note that

$$\mathop{\mathbf{E}}_{S\sim D^m} \mathop{\mathbf{E}}_{\bar{h}\sim\overline{P}} e^{(m-|\mathbf{i}_{\bar{h}}|)\,\Delta(\overline{R}_S(\bar{h}),\overline{R}_D(\bar{h}))} \;=\; \mathop{\mathbf{E}}_{\mathbf{i}\sim\overline{P}_{\mathcal{I}}} \mathop{\mathbf{E}}_{S_{\mathbf{i}}\sim D^{|\mathbf{i}|}} \mathop{\mathbf{E}}_{\bar{\mu}\sim\overline{P}_{S_{\mathbf{i}}}} \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{m-|\mathbf{i}|}} e^{|\mathbf{i}^c|\,\Delta(\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}}),\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))}\,.$$

Now, for each $h^{\bar{\mu}}_{S_{\mathbf{i}}} \in \overline{\mathcal{H}^S}$, define $a^{\mu}_{S_{\mathbf{i}}} \stackrel{\text{def}}{=} \sum_{(x,y)\in S_{\mathbf{i}}} I(h^{\bar{\mu}}_{S_{\mathbf{i}}}(x) \neq y)$. Observe that $m\cdot\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}}) - a^{\mu}_{S_{\mathbf{i}}}$ is the number of errors made by $h^{\bar{\mu}}_{S_{\mathbf{i}}}$ on $S_{\mathbf{i}^c}$. Since the later is iid and disjoint from the compression sequence of $h^{\bar{\mu}}_{S_{\mathbf{i}}}$, we have that $m\cdot\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}}) - a^{\mu}_{S_{\mathbf{i}}}$ is a random variable following a binomial law of parameters $(|\mathbf{i}^c|, \overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))$. Since, $(m-ld)\cdot\frac{k}{m} \leq |\mathbf{i}^c|\cdot\frac{k+a^{\mu}_{S_{\mathbf{i}}}}{m}$ for any $k\in\{0,..,|\mathbf{i}^c|\}$, we have

$$\begin{aligned}
&\mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{m-|\mathbf{i}|}} e^{|\mathbf{i}^c|\Delta(\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}}),\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))} \\
&\quad= \mathop{\mathbf{E}}_{S_{\mathbf{i}^c}\sim D^{m-|\mathbf{i}|}} e^{|\mathbf{i}^c|\mathcal{F}(\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))-C_1|\mathbf{i}^c|\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}})} \\
&\quad= e^{|\mathbf{i}^c|\mathcal{F}(\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))}\cdot\sum_{k=0}^{|\mathbf{i}^c|}\Pr_{S\sim D^m}\left(m\cdot\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}}) - a^{\mu}_{S_{\mathbf{i}}} = k\right) e^{-C_1|\mathbf{i}^c|\cdot\frac{k+a^{\mu}_{S_{\mathbf{i}}}}{m}} \\
&\quad\leq e^{|\mathbf{i}^c|\mathcal{F}(\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))}\cdot\sum_{k=0}^{|\mathbf{i}^c|}\Pr_{S\sim D^m}\left(m\cdot\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}}) - a^{\mu}_{S_{\mathbf{i}}} = k\right) e^{-C_1\cdot(m-ld)\cdot\frac{k}{m}} \\
&\quad= e^{|\mathbf{i}^c|\mathcal{F}(\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))}\cdot\sum_{k=0}^{|\mathbf{i}^c|}\binom{|\mathbf{i}^c|}{k}\left(\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right)^k\left(1-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\right)^{|\mathbf{i}^c|-k} e^{-C_1\cdot\frac{m-ld}{m}\cdot k} \\
&\quad= e^{|\mathbf{i}^c|\mathcal{F}(\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))}\left(1-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\left[1-e^{-C_1\cdot\frac{m-ld}{m}}\right]\right)^{|\mathbf{i}^c|}\,. \qquad \text{(C.30)}
\end{aligned}$$

The last equation being obtained from the Newton binomial $\sum_{i=0}^{k}\binom{m}{k}x^k y^{m-k} = (x+y)^k$.

Let us now define $\mathcal{F}$ such that $1 = e^{|\mathbf{i}^c|\mathcal{F}(\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))}\left(1-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}})\left[1-e^{-C_1\cdot\frac{m-ld}{m}}\right]\right)^{|\mathbf{i}^c|}$. Equivalently, let

$$\mathcal{F}(\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))) \;\stackrel{\text{def}}{=}\; -\ln\left(1-\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))\,\left[1-e^{-C_1\frac{m-ld}{m}}\right]\right). \qquad \text{(C.31)}$$

With this choice, we have $\mathbf{E}_{S\sim D^m}\mathbf{E}_{h^{\bar{\mu}}_{S_{\mathbf{i}}}\sim\overline{P}} e^{|\mathbf{i}^c|\Delta(\overline{R}_S(h^{\bar{\mu}}_{S_{\mathbf{i}}}),\overline{R}_D(h^{\bar{\mu}}_{S_{\mathbf{i}}}))} = 1\,.$

To finish the proof, let us combine Equations (C.28), (C.29), (C.30) and (C.31), in order to

rewrite the innermost inequality of Equation (C.27) as follows

$$(m-ld)\cdot\mathcal{F}\left(\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\,\zeta_D^Q\right]\right)-mC_1\cdot\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\,\zeta_S^Q\right]-\frac{\zeta'(1)}{\zeta(1)}\cdot\mathrm{KL}(Q\|P)\leq\ln\frac{1}{\delta}$$

$$\begin{aligned}\iff&(m-ld)\left\{-\ln\left(1-\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\,\zeta_D^Q\right]\left[1-e^{-C_1\frac{m-ld}{m}}\right]\right)\right\}\\&\qquad\leq mC_1\cdot\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\,\zeta_S^Q\right]+\frac{\zeta'(1)}{\zeta(1)}\cdot\mathrm{KL}(Q\|P)+\ln\frac{1}{\delta}\end{aligned}$$

$$\begin{aligned}\iff&\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\,\zeta_D^Q\right]\left[1-e^{-C_1\frac{m-ld}{m}}\right]\\&\qquad\leq 1-\exp\left\{-\left(\frac{1}{m-ld}\right)\left(mC_1\cdot\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\,\zeta_S^Q\right]+\frac{\zeta'(1)}{\zeta(1)}\cdot\mathrm{KL}(Q\|P)+\ln\frac{1}{\delta}\right)\right\}\\&\qquad\leq\left(\frac{1}{m-ld}\right)\left(mC_1\cdot\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\,\zeta_S^Q\right]+\frac{\zeta'(1)}{\zeta(1)}\cdot\mathrm{KL}(Q\|P)+\ln\frac{1}{\delta}\right),\end{aligned}$$

where the last transformation is an application of the inequality $1-e^{-x}\leq x$. We are now able to isolate ζ_D^Q to obtain

$$\begin{aligned}\zeta_D^Q&\leq\left(\frac{2\cdot\zeta(1)}{1-e^{-C_1\frac{m-ld}{m}}}\right)\left(\frac{1}{m-ld}\right)\left(mC_1\cdot\frac{1}{2}\left[1+\frac{1}{\zeta(1)}\,\zeta_S^Q\right]+\frac{\zeta'(1)}{\zeta(1)}\cdot\mathrm{KL}(Q\|P)+\ln\frac{1}{\delta}\right)-\zeta(1)\\&=\left(\frac{C_1\frac{m}{m-ld}}{1-e^{-C_1\frac{m-ld}{m}}}\right)\left(\zeta(1)+\zeta_S^Q+\frac{2}{mC_1}[\zeta'(1)\cdot\mathrm{KL}(Q\|P)+\zeta(1)\cdot\ln\frac{1}{\delta}]\right)-\zeta(1)\\&=\zeta(1)[C'-1]+C'\cdot\left(\zeta_S^Q+\frac{2}{mC_1}[\zeta'(1)\cdot\mathrm{KL}(Q\|P)+\zeta(1)\cdot\ln\frac{1}{\delta}]\right),\end{aligned}$$

where

$$C'=\frac{C_1\cdot\frac{m}{m-l\cdot\deg\zeta}}{1-e^{-C_1\cdot\frac{m-l\cdot\deg\zeta}{m}}}\,.$$

□

C.3.3 Details related to the PBSC algorithms

In this section, we present the theoretical development leading the two PBSC learning algorithms and the optimization procedures. Both algorithms build a majority vote of sc-classifier of compression sequence size of at most one as defined in Section C.2.2 of the main paper. The two algorithms that we present minimize a bound on the quadratic loss. Given a fixed parameter q, the loss an example having margin $(-\alpha)$ is given by:

$$\zeta(\alpha)\;=\;\left(1+\tfrac{1}{q}\,\alpha\right)^2.$$

The first algorithm, named *PBSC-A*, works with a strongly aligned posterior Q. The second algorithm, named *PBSC-N*, works with a non-aligned posterior Q.

PBSC-A: The aligned case

As seen in Section C.2.1 of the paper, the strongly aligned posterior Q is totally defined by a vector $\mathbf{w} \stackrel{\text{def}}{=} (w_0, w_1, \ldots, w_m)$. For Q to remain a valid distribution, each component of the vector $\mathbf{w}$ must remain in the interval $\left[-\frac{1}{m+1}, +\frac{1}{m+1}\right]$.

Theorem C.5 suggests to minimize the bound on ζ_D^Q given by the following expression:

$$\zeta_D^Q \;\leq\; \zeta_S^Q + \frac{\zeta(1)}{\sqrt{\frac{1}{2}(m - l\deg\zeta)}} \sqrt{4\,l\deg\zeta + \ln\frac{2\sqrt{m}}{\delta}}\,.$$

To do so, we only need to minimize the empirical risk ζ_S^Q because the last term of the right hand side is constant. The empirical risk, ζ_S^Q, is given by

$$\zeta_S^Q \;=\; \frac{1}{mq^2}\sum_{j=1}^{m}\left(q - y_j\left[w_0 + \sum_{i=1}^{m} w_i k(x_i, x_j)\right]\right)^2.$$

Let us define a matrix $\mathbf{G}$ of size $m+1 \times m$ as

$$G_{i,j} \;=\; \begin{cases} 1 & \text{for } i = 0\,, \\ k(x_i, x_j) & \text{for } 1 \leq i, j \leq m\,. \end{cases}$$

With this notation, the optimization problem of *PBSC-A* can be written as

$$\begin{aligned} \text{Minimize:} \quad & f_A(\mathbf{w}) \;= \sum_{j=1}^{m}\left(q - y_j \sum_{i=0}^{m} w_i G_{i,j}\right)^2 \\ \text{subject to:} \quad & |w_i| \leq \tfrac{1}{m+1} \text{ for } i = 0, 1, \ldots, m\,. \end{aligned}$$

We propose to solve this optimization problem by minimizing f_A coordinate-wise, similarly as it is done for AdaBoost (Schapire et Singer, 1999), with the difference that we will have to ensure that Q remains an aligned distribution at each step of the algorithm. Starting from the uniform distribution P (*i.e.*, $\mathbf{w} = \mathbf{0}$), the learning algorithm iteratively chooses (at random) $k \in \{0, \ldots, m\}$, and updates $w_k \leftarrow w_k + \delta$ (without updating the other weights) according to some optimally chosen value of δ. Let $\mathbf{w}_\delta$ be the new weight vector obtained with such an update. After an update, the objective function becomes:

$$f_A(\mathbf{w}_\delta) \;=\; \sum_{j=1}^{m}\left[q - y_j\left(\sum_{i=0}^{m} w_i G_{i,j} + \delta G_{k,j}\right)\right]^2.$$

The optimal value for δ is obtained when $\frac{df_A(\mathbf{w}_\delta)}{d\delta} = 0$, provided that $w_k + \delta \in [\frac{-1}{m+1}, \frac{1}{m+1}]$.

The derivative of f_A with respect to the δ is given by

$$\begin{aligned}
\frac{\partial f_A(\mathbf{w}_\delta)}{\partial \delta} &= \sum_{j=1}^{m} 2\left(q - y_j \sum_{i=0}^{m} w_i G_{i,j} - y_j \delta G_{k,j}\right)(-y_j G_{k,j}) \\
&= 2\sum_{j=1}^{m}\left[\delta G_{k,j}^2 + y_j G_{k,j}\left(y_j \sum_{i=0}^{m} w_i G_{i,j} - q\right)\right] \\
&= 2\left[\delta \sum_{j=1}^{m} G_{k,j}^2 + \sum_{j=1}^{m} G_{k,j}\left(\sum_{i=0}^{m} w_i G_{i,j} - q y_j\right)\right] \\
&= 2\left[\delta \sum_{j=1}^{m} G_{k,j}^2 + \sum_{j=1}^{m} G_{k,j} D_{\mathbf{w}}(j)\right], \qquad \text{(C.32)}
\end{aligned}$$

where $D_{\mathbf{w}}(j) \stackrel{\text{def}}{=} \sum_{i=0}^{m} w_i G_{i,j} - q y_j$.

Equation (C.32) implies that the optimal value for δ is given by

$$\delta \;=\; -\frac{\sum_{j=1}^{m} G_{k,j}\, D_{\mathbf{w}}(j)}{\sum_{j=1}^{m} G_{k,j}^2}\,. \qquad \text{(C.33)}$$

Algorithm 3 presents the complete optimization procedure that we have used.

Algorithm 3 : PBSC-A optimization procedure

1: **Initialize:** $w_i = 0 \;\; \forall i \in \{0,\ldots,m\}$ and $D_{\mathbf{w}}(j) \;=\; -\,q\,y_j \;\; \forall j \in \{1,\ldots,m\}$.
2: **repeat**
3: Choose at random $k \in \{0,..,m\}$.
4: Compute δ given by Equation (C.33) .
5: If $[w_k + \delta > \frac{1}{m+1}]$ then $\delta \leftarrow \frac{1}{m+1} - w_k$.
6: If $[w_k + \delta < \frac{-1}{m+1}]$ then $\delta \leftarrow \frac{-1}{m+1} - w_k$.
7: $w_k \leftarrow w_k + \delta$.
8: Update $D_{\mathbf{w}}(j) \;\leftarrow\; D_{\mathbf{w}}(j) + \delta\, G_{k,j} \;\; \forall j \in \{1,\ldots,m\}$.
9: **until** Convergence

PBSC-N: The non-aligned case

We consider the non-aligned scenario where the posterior Q is defined by a vector $\mathbf{v} \stackrel{\text{def}}{=} (v_+, v_1, \ldots, v_{2m}, v_-)$:

$$\begin{aligned}
Q(h^{(\varepsilon,+)}_{S_{\langle\rangle}}) &= v_+\,, \qquad Q(h^{(\sigma,+)}_{S_{\langle i\rangle}}) \;=\; v_i \tfrac{1}{|\mathcal{M}^1|} I(\sigma \in \mathcal{M}^1)\,, \\
Q(h^{(\varepsilon,-)}_{S_{\langle\rangle}}) &= v_-\,, \qquad Q(h^{(\sigma,-)}_{S_{\langle i\rangle}}) \;=\; v_{m+i} \tfrac{1}{|\mathcal{M}^1|} I(\sigma \in \mathcal{M}^1)\,,
\end{aligned}$$

under the constraints $v \geq 0$ for all $v \in \mathbf{v}$ and $\sum_{v\in\mathbf{v}} v = 1$.

Lets compute the Kullback-Leibler divergence $\mathrm{KL}(Q\|P)$ between this posterior Q and the uniform prior P. We find that

$$\begin{aligned}
\mathrm{KL}(Q\|P) &= \mathop{\mathbf{E}}_{h\sim Q} \ln \tfrac{Q(h)}{P(h)} \\
&= \sum_{i=1}^{m} \sum_{s\in\{-,+\}} \int_{\mathcal{M}^1} d\sigma Q(h_{S_{\langle i\rangle}}^{(\sigma,s)}) \ln \left[\frac{Q(h_{S_{\langle i\rangle}}^{(\sigma,s)})}{P(h_{S_{\langle i\rangle}}^{(\sigma,s)})}\right] + \sum_{s\in\{-,+\}} Q(h_{S_{\langle\rangle}}^{(\varepsilon,s)}) \ln \left[\frac{Q(h_{S_{\langle\rangle}}^{(\varepsilon,s)})}{P(h_{S_{\langle\rangle}}^{(\varepsilon,s)})}\right] \\
&= \sum_{i=1}^{2m} \int_{\mathcal{M}^1} d\sigma \frac{v_i}{|\mathcal{M}^1|} \ln \left[\frac{\frac{v_i}{|\mathcal{M}^1|}}{\frac{1}{2|\mathcal{M}^1|(m+1)}}\right] + v_+ \ln \frac{v_+}{\frac{1}{2(m+1)}} + v_- \ln \frac{v_-}{\frac{1}{2(m+1)}} \\
&= \sum_{v\in\mathbf{v}} v \ln \left[\frac{v}{\frac{1}{2(m+1)}}\right] \\
&= \ln(2m+2) + \sum_{v\in\mathbf{v}} v \ln[v]\,.
\end{aligned}$$

Moreover, we show in the main paper that the empirical risk ζ_S^Q is given by

$$\zeta_S^Q = \frac{1}{mq^2} \sum_{j=1}^{m} \left(q - y_j \left[v_+ - v_- + \sum_{i=1}^{m} (v_i - v_{i+m}) k(x_i, x_j) \right] \right)^2 .$$

Minimizing the bound of Theorem 1 amounts at finding $\mathbf{v}$ that minimizes

$$C \cdot \zeta_S^Q + \mathrm{KL}(Q\|P)\,,$$

for some constant $C > 0$. However, in Theorem C.1, we have $C = \frac{m \cdot C_1}{2\,\zeta'(1)}$.

Let $v_0 \stackrel{\text{def}}{=} v_+$ and $v_{2m+1} \stackrel{\text{def}}{=} v_-$. Let us define a matrix $\mathbf{G}$ of size $2m+2 \times m$ as

$$G_{i,j} = \begin{cases} 1 & \text{if } i = 0\,, \\ k(\mathbf{x}_i, \mathbf{x}_j) & \text{if } 1 \le i, j \le m\,, \\ -k(\mathbf{x}_{i-m}, \mathbf{x}_j) & \text{if } m+1 \le i \le 2m \text{ (and } 1 \le j \le m), \\ -1 & \text{if } i = 2m+1\,. \end{cases}$$

With this notation, the optimization problem for *PBSC-N* can be written as

$$\begin{aligned}
\text{Minimize:} \quad f_N(\mathbf{v}) &= \frac{C}{mq^2} \sum_{j=1}^{m} \left(q - y_j \sum_{i=0}^{2m+1} v_i G_{i,j} \right)^2 + \sum_{i=0}^{2m+1} v_i \ln v_i\,, \\
\text{subject to:} \quad & v_i \ge 0 \quad \text{for } i = 0, 1, \ldots, 2m+1\,, \\
& \sum_{i=0}^{2m+1} v_i = 1\,.
\end{aligned}$$

We propose to solve this optimization problem by minimizing f_N with a coordinate-pair descent algorithm that works iteratively by exchanging weights between two components of $\mathbf{v}$. Starting from the uniform distribution P (*i.e.*, $v_i = \frac{1}{2m+2}$ for $i = 0, 1, \ldots, 2m+1$), the

learning algorithm iteratively chooses (at random) $k, l \in \{0, \ldots, 2m+1\}$ (with $k \neq l$), and updates $v_k \leftarrow v_k + \delta$ and $v_l \leftarrow v_l - \delta$ (without updating the other weights) according to some optimally chosen value of δ. Let $\mathbf{v}_\delta$ be the new weight vector obtained with such an update. After an update, the objective function becomes

$$\begin{aligned} f_N(\mathbf{v}_\delta) &= \frac{C}{mq^2} \sum_{j=1}^{m} \left[q - y_j \left(\sum_{i=0}^{2m+1} v_i G_{i,j} + \delta G_{k,j} - \delta G_{l,j} \right) \right]^2 \\ &\quad + \sum_{i=0}^{2m+1} I(i \notin \{k,l\}) \cdot v_i \ln v_i + (v_k + \delta) \ln(v_k + \delta) + (v_l - \delta) \ln(v_l - \delta) \,. \end{aligned}$$

The optimal value for δ is obtained when $\frac{df_N(\mathbf{v}_\delta)}{d\delta} = 0$, provided that $v_k + \delta \in [0, v_k + v_l]$ and $v_l - \delta \in [0, v_k + v_l]$. The derivative of f_N with respect to the δ is given by

$$\begin{aligned} \frac{\partial f_N(\mathbf{v}_\delta)}{\partial \delta} &= \frac{C}{mq^2} \sum_{j=1}^{m} 2 \left(q - y_j \sum_{i=0}^{2m+1} v_i G_{i,j} - y_j \delta \left(G_{k,j} - G_{l,j} \right) \right) \left(-y_j \left(G_{k,j} - G_{l,j} \right) \right) + \ln \frac{v_k + \delta}{v_l - \delta} \\ &= \frac{2C}{mq^2} \sum_{j=1}^{m} \left[\delta \left(G_{k,j} - G_{l,j} \right)^2 + y_j \left(G_{k,j} - G_{l,j} \right) \left(y_j \sum_{i=0}^{2m+1} v_i G_{i,j} - q \right) \right] + \ln \frac{v_k + \delta}{v_l - \delta} \\ &= \frac{2C}{mq^2} \left[\delta \sum_{j=1}^{m} \left(G_{k,j} - G_{l,j} \right)^2 + \sum_{j=1}^{m} \left(G_{k,j} - G_{l,j} \right) \left(\sum_{i=0}^{2m+1} v_i G_{i,j} - q y_j \right) \right] + \ln \frac{v_k + \delta}{v_l - \delta} \\ &= \frac{2C}{mq^2} \left[\delta \sum_{j=1}^{m} \left(G_{k,j} - G_{l,j} \right)^2 + \sum_{j=1}^{m} \left(G_{k,j} - G_{l,j} \right) D_{\mathbf{v}}(j) \right] + \ln \frac{v_k + \delta}{v_l - \delta} \,, \end{aligned} \tag{C.34}$$

where $D_{\mathbf{v}}(j) = \sum_{i=0}^{2m+1} v_i G_{i,j} - q y_j \,.$

We find the optimal value for δ with the help of a root finding method. Algorithm 4 presents the complete optimization procedure that we have used.

Algorithm 4 : PBSC-N optimization procedure

1: **Initialize:** $v_i = \frac{1}{2m+2} \quad \forall i \in \{0, \ldots, m\}$ and $D_{\mathbf{v}}(j) = -q\, y_j \quad \forall j \in \{1, \ldots, m\}$.
2: **repeat**
3: Choose at random $k, l \in \{0, \ldots, 2m+1\}$ (with $k \neq l$).
4: Find δ given by the root of Equation (C.34).
5: If $\delta > v_l$ then $\delta \leftarrow v_l$.
6: If $\delta < -v_k$ then $\delta \leftarrow -v_k$.
7: $v_k \leftarrow v_k + \delta$ and $v_l \leftarrow v_l - \delta$.
8: Update $D_{\mathbf{v}}(j) \leftarrow D_{\mathbf{v}}(j) + \delta \left(G_{k,j} - G_{l,j} \right) \quad \forall j \in \{1, \ldots, m\}$.
9: **until** Convergence

Annexe D

A Pseudo-Boolean Set Covering Machine

Cet article a été publié dans le cadre de la conférence *Principles and Practice of Constraint Programming* (voir Germain et al., 2012a).

Abstract. The Set Covering Machine (SCM) is a machine learning algorithm that constructs a conjunction of Boolean functions. This algorithm is motivated by the minimization of a theoretical bound. However, finding the optimal conjunction according to this bound is a combinatorial problem. The SCM approximates the solution using a greedy approach. Even though SCM seems very efficient in practice, it is unknown how it compares to the optimal solution. To answer this question, we present a novel pseudo-Boolean optimization model that encodes the minimization problem. It is the first time a Constraint Programming approach addresses the combinatorial problem related to this machine learning algorithm. Using that model and recent pseudo-Boolean solvers, we empirically show that the greedy approach is surprisingly close to the optimal.

D.1 Introduction

Machine learning (Bishop, 2006) studies algorithms that "learn" to perform a task by observing examples. In the classification framework, a learning algorithm is executed on a training set which contains examples. Each example is characterized by a description and a label. A learning algorithm's goal is to generalize the information contained in the training set to build a classifier, *i.e.*, a function that takes as input an example description, and outputs a label prediction. A good learning algorithm produces classifiers of low risk, meaning a low probability of misclassifying a new example that was not used in the learning process.

Among all machine learning theories, *Sample Compression* (Floyd et Warmuth, 1995) studies

classifiers that can be expressed by a subset of the training set. This theory allow us to compute bounds on a classifier's risk based on two main quantities: the size of the *compression set* (the number of training examples needed to describe the classifier) and the *empirical risk* (the proportion of misclassified training examples). This suggests that a classifier should realize a tradeoff between its complexity, quantified here by the compression set size, and its accuracy on the training set.

Based on this approach, the *Set Covering Machine* (SCM) is a learning algorithm motivated by a sample compression risk bound (Marchand et Shawe-Taylor, 2002). However, instead of finding the optimal value of the bound, the SCM algorithm is a greedy approach that aims to quickly find a good solution near the optimal bound's value.

In this paper, we address the following question: "How far to the optimal is the solution returned by the SCM algorithm?". To answer this question, one needs to design a learning algorithm that directly minimizes the sample compression bound that inspired the SCM. This task is not a trivial one : unlike many popular machine learning algorithms that rely on the minimization of a convex function (as the famous Support Vector Machine (Cortes et Vapnik, 1995)), this optimization problem is based on a combinatorial function. Although Hussain et al. (2004) suggested a (convex) linear program version of the SCM, it remains a heuristic inspired by the bound. The present paper describes how to use Constraint Programming techniques to directly minimize the sample compression bound. More precisely, we design a pseudo-Boolean program that encodes the proper optimization problem, and finally show that the SCM is surprisingly accurate.

D.2 Problem Description

The Binary Classification problem in Machine Learning.

An *example* is a pair $(\mathbf{x}, y)$, where $\mathbf{x}$ is a *description* and y is a *label*. In this paper, we consider binary classification, where the description is a vector of n real-valued attributes (*i.e.*, $\mathbf{x} \in \mathbb{R}^n$) and the label is a Boolean value (*i.e.*, $y \in \{0, 1\}$). We say that a 0-labeled example is a *negative example* and a 1-labeled is a *positive example*.

A *dataset* contains several examples coming from the observation of the same phenomenon. We denote S the *training set* of m examples used to "learn" this phenomenon. As the examples are considered to be independently and identically distributed (iid) following a probability distribution D on $\mathbb{R}^n \times \{0, 1\}$, we have:

$$S \stackrel{\text{def}}{=} \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\} \sim D^m .$$

A *classifier* receives as input the description of an example and predicts a label. Thus, a classifier is a function $h : \mathbb{R}^n \to \{0, 1\}$. The *risk* $R(h)$ of a classifier is the probability of

misclassifying an example generated by the distribution D, and the *empirical risk* $R_S(h)$ of a classifier is the ratio of errors on its training set.

$$R(h) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{(\mathbf{x},y)\sim D} \mathtt{I}(h(\mathbf{x}) \neq y) \quad \text{and} \quad R_S(h) \stackrel{\text{def}}{=} \frac{1}{m} \sum_{(\mathbf{x},y)\in S} \mathtt{I}(h(\mathbf{x}) \neq y)\,,$$

where $\mathtt{I}$ is the indicator function: $\mathtt{I}(a) = 1$ if a is true and $\mathtt{I}(a) = 0$ otherwise.

A *learning algorithm* receives as input a training set and outputs a classifier. The challenge of a these algorithms is to generalize the information of the training set to produce a classifier of low risk. Since the data generating distribution D is unknown, a common practice to estimate the risk is to calculate the error ratio on a *testing set* containing examples that were not used in the training process.

Overview of the Sample Compression Theory.

The sample compression theory, first expressed by Floyd et Warmuth (1995), focuses on classifiers that can be expressed by a subset of the training set.

Consider a classifier obtained by executing a learning algorithm on the training set S containing m examples. The *compression set* $S_{\mathbf{i}}$ refers to examples of the training set that are needed to characterize the classifier.

$$S_{\mathbf{i}} \stackrel{\text{def}}{=} \{(\mathbf{x}_{i_1}, y_{i_1}), (\mathbf{x}_{i_2}, y_{i_2}), \ldots, (\mathbf{x}_{i_n}, y_{i_n})\} \subseteq S \quad \text{with} \;\; 1 \leq i_1 < i_2 < \ldots < i_n \leq m\,.$$

We sometimes use a *message string* μ that contains additional information [1]. The term *compressed classifier* refers to the classifier obtained solely with the compression set $S_{\mathbf{i}}$ and message string μ. Sample compression provides theoretical guarantees on a compressed classifier by upper-bounding its risk. Typically, those bounds suggest that a learning algorithm should favour classifiers of low empirical risk (accuracy) and that are expressed by a few training examples (sparsity). One can advocate for sparse classifiers because they are easy to understand by a human being.

The Set Covering Machine.

Suggested by Marchand et Shawe-Taylor (2002), the *Set Covering Machine* (SCM) is a learning algorithm directly motivated by the sample compression theory. It builds a conjunction or a disjunction of binary functions that rely on training set data. We focus here on the most studied case where each binary function is a *ball* $g_{i,j}$ characterized by two training examples, a center $(\mathbf{x}_i, y_i) \in S$ and a border $(\mathbf{x}_j, y_j) \in S$.

$$g_{i,j}(\mathbf{x}) \stackrel{\text{def}}{=} \begin{cases} y_i \text{ if } \|\mathbf{x}_i - \mathbf{x}\| < \|\mathbf{x}_i - \mathbf{x}_j\| \\ \neg y_i \text{ otherwise,} \end{cases} \tag{D.1}$$

1. See Marchand et Shawe-Taylor (2002) for further details about the *message* concept in sample compression theory.

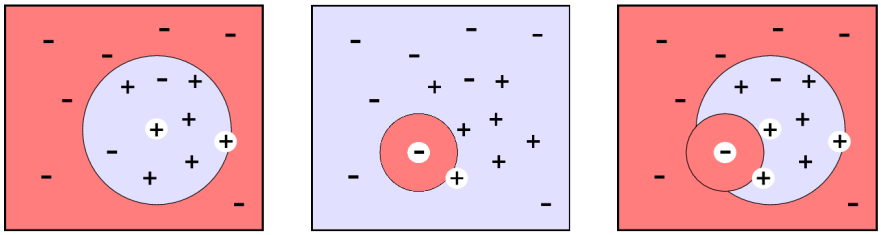


Figure D.1 – On a 2-dimensional dataset of 16 examples, from left to right: a positive ball, a negative ball, and the conjunction of both balls. Examples in the light blue region and the red region will be respectively classified positive and negative.

where $\|\cdot\|$ is the Euclidean norm. For simplicity, we omit the case $\|\mathbf{x}_i - \mathbf{x}\| = \|\mathbf{x}_i - \mathbf{x}_j\|$ and consider that a ball correctly classifies its center ($g_{i,j}(\mathbf{x}_i) = y_i$) and its border ($g_{i,j}(\mathbf{x}_j) = y_j$).

We denote $\mathcal{H}_S$ the set of all possible balls on a particular dataset S, and $\mathcal{B}$ the set of balls selected by the SCM algorithm among $\mathcal{H}_S$. Thus, the classification function related to a conjunction of balls is expressed by:

$$h_{\mathcal{B}}(\mathbf{x}) \stackrel{\text{def}}{=} \bigwedge_{g \in \mathcal{B}} g(\mathbf{x})\,. \tag{D.2}$$

As the disjunction case is very similar to the conjunction case, we simplify the following discussion by dealing only with the latter [2]. Figure D.1 illustrates an example of a classifier obtained by a conjunction of two balls.

The goal of the SCM algorithm is to choose balls among $\mathcal{H}_S$ to form the conjunction $h_{\mathcal{B}}$. By specializing the sample-compressed classifier's risk bound to the conjunction of balls, Marchand et Sokolova (2005) proposed to minimize the risk bound given by Theorem D.1 below. Note that the compression set $S_{\mathbf{i}}$ contains the examples needed to construct the balls of $h_{\mathcal{B}}$. Also, the message string μ identifies which examples of $S_{\mathbf{i}}$ are centers, and points out the border example associated with each center. In Theorem D.1, the variables n_p and n_b encode the length of the message string μ.

Theorem D.1. *(Marchand et Sokolova, 2005) For any data-generating distribution D for which we observe a dataset S of m examples, and for each $\delta \in (0, 1]$:*

$$\Pr_{S \sim D^m}\Big(\forall \mathcal{B} \subseteq \mathcal{H}_S \;:\; R(h_{\mathcal{B}}) \;\leq\; \varepsilon(\mathcal{B})\Big) \;\geq\; 1 - \delta\,,$$

where:

$$\varepsilon(\mathcal{B}) \stackrel{\text{def}}{=} 1 - \exp\left(\frac{-1}{m-(|S_{\mathbf{i}}|+k)} \ln\left[\binom{m}{|S_{\mathbf{i}}|+k} \cdot \binom{|S_{\mathbf{i}}|+k}{k} \cdot \binom{n_p}{n_b} \cdot \frac{1}{\zeta(n_b)\,\zeta(|S_{\mathbf{i}}|)\,\zeta(k)\,\delta}\right]\right), \tag{D.3}$$

and where k is the number of errors that $h_{\mathcal{B}}$ does on training set S, n_p is the number of positive examples in compression set $S_{\mathbf{i}}$, n_b is the number of different examples used as a border, and $\zeta(a) \stackrel{\text{def}}{=} \frac{6}{\pi^2}(a+1)^{-2}$.

2. The disjunction case equations can be recovered by applying De Morgan's law.

This theorem suggests minimizing the expression of $\varepsilon(\mathcal{B})$ in order to find a good balls conjunction. For fixed values of m and δ, this expression tends to decrease with decreasing values of $|S_{\mathbf{i}}|$ and k, whereas $n_b \leq n_p \leq |S_{\mathbf{i}}|$. Moreover, even if the expression of $\varepsilon(\mathcal{B})$ contains many terms, we notice that the quantity $\left[\binom{n_p}{n_b} \cdot \frac{1}{\zeta(n_b)\zeta(|S_{\mathbf{i}}|)\zeta(k)\delta}\right]$ is very small. If we neglect this term, it is easy to see that minimizing $\epsilon(\mathcal{B})$ boils down to find the minimum of Equation (D.4), which is the sum of the compression set size and the number of empirical errors.

$$\mathcal{F}(\mathcal{B}) \stackrel{\text{def}}{=} |S_{\mathbf{i}}| + k\,. \tag{D.4}$$

This consideration leads us to the SCM algorithm. We say that a ball belonging to a conjunction *covers an example* whenever it classifies it negatively. Note that a balls conjunction $h_{\mathcal{B}}$ negatively classifies an example $\mathbf{x}$ if and only if at least one ball of $\mathcal{B}$ covers $\mathbf{x}$. This implies that if one wants to add a new ball to an existing balls conjunction, he can only change the classification outcome on uncovered examples. A good strategy for choosing a ball to add to a conjunction is then to cover as few positive examples as possible to avoid misclassifying them. This observation underlies the heuristic of the SCM algorithm.

Given a training set S, the SCM algorithm (see Algorithm 5) is a greedy procedure for selecting a small subset $\mathcal{B}$ of all possible balls [3] so that a high number of negative examples of S are covered by at least one ball belonging to $\mathcal{B}$. At each step of the algorithm, the tradeoff between the number of covered negative examples and the number of covered positive examples is due to a heuristic (Line 6 of Algorithm 5) that depends on a penalty parameter $p \in [0, \infty)$. We initialize the algorithm with a selection of penalty values, allowing it to create a variety of balls conjunctions. The algorithm returns the best conjunction according to a *model selection function* of our choice.

Several model selection functions can be used along with the SCM algorithm. The function ε given by Equation (D.3) leads to excellent empirical results. In other words, by running the algorithm with a variety of penalty parameters, selecting from all generated balls conjunctions the one with the lowest bound value allow us to obtain a low risk classifier. This method as been shown by Marchand et Shawe-Taylor (2002) to be as good as cross-validation [4]. It is exceptional for a risk bound to have such property.

As we explain, the bound relies mainly on the sum $|S_{\mathbf{i}}| + k$, and our extensive experiments with the SCM confirms that the simple model selection function $\mathcal{F}$ given by Equation (D.4) gives equally good results. We are then interested to know if the SCM algorithm provides a good approximation of this function.

3. More precisely, the heuristic function (Line 6 of Algorithm 5) makes it possible to consider only balls whose borders are defined by positive examples (see Marchand et Shawe-Taylor (2002)).

4. Cross-validation is a widely used method for estimating reliability of a model, but substantially increases computational needs (see section 1.3 of Bishop (2006)).

Algorithm 5 SCM (*dataset S, penalties $\{p_1, \ldots, p_n\}$, selection function f*)

Consider all possible balls: $\mathcal{H}_S \leftarrow \{g_{i,j} \mid (\mathbf{x}_i, \cdot) \in S,\, (\mathbf{x}_j, 1) \in S,\, \mathbf{x}_i \neq \mathbf{x}_j\}$.
Initialize: $\mathcal{B}^* \leftarrow \emptyset$.
for $p \in \{p_1, p_2, \ldots, p_n\}$ **do**
  Initialize: $\mathcal{N} \leftarrow \{\mathbf{x} \mid (\mathbf{x}, 0) \in S\}$, $\mathcal{P} \leftarrow \{\mathbf{x} \mid (\mathbf{x}, 1) \in S\}$ and $\mathcal{B} \leftarrow \emptyset$.
  while $\mathcal{N} \neq \emptyset$ **do**
    Choose the best ball according to the following heuristic:

$$g \;\leftarrow\; \operatorname{argmax} g \in \mathcal{H}_S \,\{\; |\{\mathbf{x} \in \mathcal{N} \mid g(\mathbf{x}) = 0\}| \;-\; p \cdot |\{\mathbf{x} \in \mathcal{P} \mid g(\mathbf{x}) = 0\}| \;\}\,.$$

    Add this ball to current conjunction: $\mathcal{B} \leftarrow \mathcal{B} \cup \{g\}$.
    Clean covered examples: $\mathcal{N} \leftarrow \{\mathbf{x} \in \mathcal{N} \mid g(\mathbf{x}) = 1\}$, $\mathcal{P} \leftarrow \{\mathbf{x} \in \mathcal{P} \mid g(\mathbf{x}) = 1\}$.
    Retain the best conjunction : **if** $f(\mathcal{B}) < f(\mathcal{B}^*)$ **then** $\mathcal{B}^* \leftarrow \mathcal{B}$.
  end while
end for
return $\mathcal{B}^*$

To answer this question, next section presents a pseudo-Boolean optimization model that directly finds the set $\mathcal{B}$ that minimizes the function $\mathcal{F}$.

D.3 A Pseudo-Boolean Optimization Model

A pseudo-Boolean problem consists of linear inequality constraints with integer coefficients over binary variables. One can also have a linear objective function.

To solve our machine learning problem with a pseudo-Boolean solver, the principal challenge is to translate the original problem into this particular form. The main strategy to achieve this relies on the following observation:

Observation. As the classification function $h_{\mathcal{B}}$ is a conjunction (see Equation (D.2)), we observe that $h_{\mathcal{B}}$ misclassifies a positive example iff a negative ball covers it. Similarly, $h_{\mathcal{B}}$ misclassifies a negative example iff no ball covers it.

Equivalence rules. Let's first state two general rules that will be useful to express the problem with pseudo-Boolean constraints. For any positive integer $n \in \mathbb{N}^*$ and Boolean values $\alpha_1, \ldots, \alpha_n, \beta \in \{0, 1\}$, the conjunction and disjunction of the Boolean values α_i can be encoded with these linear inequalities:

$$\alpha_1 \wedge \ldots \wedge \alpha_n = \beta \quad \Leftrightarrow \quad n - 1 \;\geq\; \alpha_1 + \ldots + \alpha_n - n \cdot \beta \;\geq\; 0\,, \tag{D.5}$$

$$\alpha_1 \vee \ldots \vee \alpha_n = \beta \quad \Leftrightarrow \quad 0 \;\geq\; \alpha_1 + \ldots + \alpha_n - n \cdot \beta \;\geq\; 1 - n\,. \tag{D.6}$$

Program variables. Let $P \stackrel{\text{def}}{=} \{i \mid (\mathbf{x}_i, 1) \in S\}$ and $N \stackrel{\text{def}}{=} \{i \mid (\mathbf{x}_i, 0) \in S\}$ be two disjoint sets, containing indices of positive and negative examples respectively. We define m sets B_i,

each containing the indices of the borders that can be associated to center $\mathbf{x}_i$, and m sets C_j, each containing the indices of the centers that can be associated to border $\mathbf{x}_j$. As Marchand and Shawe-Taylor Marchand et Shawe-Taylor (2002), we only consider balls with positive borders. Thus, for $i, j \in \{1, \ldots, m\}$, we have:

$$B_i \stackrel{\text{def}}{=} \{j \mid j \in P,\, j \neq i\} \quad \text{and} \quad C_j \stackrel{\text{def}}{=} \{i \mid i \in P \cup N,\, j \in B_i\}\,.$$

In other words, B_k is the set of example indices that can be the border of a ball centered on x_k. Similarly, C_k is the set of example indices that can be the center of a ball whose border is x_k. Necessarily, we have $j \in B_k \iff k \in C_j$.

Given the above definitions of B_i and C_j, the solver have to determine the value of Boolean variables s_i, r_i and $b_{i,j}$ described below:

For every $i \in \{1, \ldots, m\}$:

— s_i is equal to 1 iff the example $\mathbf{x}_i$ belongs to the compression set.

— r_i is equal to 1 iff the $h_{\mathcal{B}}$ misclassifies the example $\mathbf{x}_i$.

— For every $j \in B_i$, $b_{i,j}$ is equal to 1 iff the example $\mathbf{x}_i$ is the center of a ball and $\mathbf{x}_j$ if the border of that same ball.

Objective function. The function to optimize (see Equation (D.4)) becomes:

$$\min \sum_{i=1}^{m} (r_i + s_i)\,. \tag{D.7}$$

Program constraints. If an example $\mathbf{x}_i$ is the center of a ball, we want exactly one example $\mathbf{x}_j$ to be its border. Also, if $\mathbf{x}_i$ is not the center of any ball, we don't want any example $\mathbf{x}_j$ to be its border. Those two conditions are encoded by:

$$\sum_{j \in B_i} b_{i,j} \;\leq\; 1 \quad \text{for } i \in \{1, \ldots, m\}\,. \tag{D.8}$$

An example belongs to the compression set iff it is a center or a border. We then have $s_k = \left[\bigvee_{i \in C_k} b_{i,k}\right] \vee \left[\bigvee_{j \in B_k} b_{k,j}\right]$. Equivalence rule (D.6) gives:

$$1 - |B_k \cup C_k| \;\leq\; -|B_k \cup C_k| \cdot s_k + \sum_{i \in C_k} b_{i,k} + \sum_{j \in B_k} b_{k,j} \;\leq\; 0 \quad \text{for } k \in \{1, \ldots, m\}\,. \tag{D.9}$$

We denote by $D_{i,j}$ the distance between examples $\mathbf{x}_i$ and $\mathbf{x}_j$. Therefore, D is a square matrix of size $m \times m$. For each example index $k \in \{1, \ldots, m\}$, let E_k be the set of all balls that cover (*i.e.*, negatively classify) the example $\mathbf{x}_k$:

$$E_k \stackrel{\text{def}}{=} \{b_{i,j} \mid i \in P, j \in B_i, D_{i,j} < D_{i,k}\} \cup \{b_{i,j} \mid i \in N, j \in B_i, D_{i,j} > D_{i,k}\}\,.$$

First, suppose that $\mathbf{x}_k$ is a positive example (thus, $k \in P$). Then, recall that the conjunction misclassifies the example $\mathbf{x}_k$ iff a ball covers it (see "observation" above). Therefore,

$r_k = \bigvee_{b_{i,j} \in E_k} b_{i,j}$. Using Equivalence Rule (D.6), we obtain:

$$1 - |E_k| \;\leq\; -|E_k| \cdot r_k + \sum_{b_{i,j} \in E_k} b_{i,j} \;\leq\; 0 \quad \text{for } k \in P\,. \tag{D.10}$$

Now, suppose that $\mathbf{x}_k$ is a negative example (thus, $k \in N$). Then, recall that the conjunction misclassifies $\mathbf{x}_k$ iff no ball covers it (see "observation" above). We have $r_k = \bigwedge_{b_{i,j} \in E_k} \neg\, b_{i,j}$. By using Equivalence Rule (D.5) and $\alpha = \neg\beta \Leftrightarrow \alpha = 1 - \beta$ (where $\alpha, \beta \in \{0, 1\}$), we obtain the following constraints:

$$\begin{aligned} & 0 \;\leq\; -|E_k| \cdot r_k + \sum_{b_{i,j} \in E_k} (1 - b_{i,j}) \;\leq\; |E_k| - 1 \\ \Leftrightarrow \quad & 1 \;\leq\; |E_k| \cdot r_k + \sum_{b_{i,j} \in E_k} b_{i,j} \;\leq\; |E_k| \quad \text{for } k \in N\,. \end{aligned} \tag{D.11}$$

D.4 Empirical Results on Natural Data

The optimization problem of minimizing Equation (D.7) under Constraints (D.8, D.9, D.10, D.11) gives a new learning algorithm that we call *PB-SCM*. To evaluate this new algorithm, we solve several learning problems using three well-known pseudo-Boolean solvers, PWBO (Martins et al., 2011), SCIP (Achterberg, 2004) and BSOLO (Manquinho et Marques-Silva, 2006), and compare the obtained results to the SCM (the greedy approach described by Algorithm 5).

We use the same seven datasets than Marchand et Shawe-Taylor (2002) and Marchand et Sokolova (2005), which are common benchmark datasets in the machine learning community. For each dataset, we repeat the following experimental procedure four times with training set sizes $m = |S|$ of 25, 50, 75 and 100 examples. First, we randomly split the dataset examples in a training set S of m examples and a testing set T containing all remaining examples [5]. Then, we execute the four learning algorithms (SCM algorithm and PB-SCM with three different solvers) on the same training set S, and compute the risk on the testing set T.

To obtain SCM results, the algorithm is executed with a set of 41 penalty values $\{10^{a/20} \mid a = 0, 1, \ldots, 40\}$ and the model selection function $\mathcal{F}$ given by Equation (D.4). The PB-SCM problem is solved with the three different solvers. For each solver, we fix the time limit to 3600 seconds and keep the solver's default values for other parameters. When a solver fails to converge in 3600 seconds, we consider the best solution so far. Using the solution of the SCM to provide an initial upper bound to the pseudo-Boolean solvers provided no speed-up.

Table D.1 shows the obtained results. Of course, except for T/O situations, the minimal value of the heuristic $\mathcal{F}$ is always obtained by solving the PB-SCM problem. However, it is surprising that the SCM often reaches the same minimum value. Moreover, the SCM

5. Training sets are smalls because of the extensive computational power needed by pseudo-Boolean solvers.

Table D.1 – Empirical results comparing the objective value $\mathcal{F}$ obtained by SCM and PB-SCM algorithms, the test risk of obtained classifiers and required running time ("T/O" means that the pseudo-Boolean solver reaches the time limit).

Dataset		**SCM**			**PB-SCM** (pwbo)			**PB-SCM** (scip)			**PB-SCM** (bsolo)		
name	**size**	$\mathcal{F}$	**risk**	**time**	$\mathcal{F}$	**risk**	**time**	$\mathcal{F}$	**risk**	**time**	$\mathcal{F}$	**risk**	**time**
breastw	25	**2**	0.046	0.04	**2**	0.081	0.03	**2**	0.064	0.71	**2**	0.046	0.05
	50	**2**	0.047	0.07	**2**	0.046	0.06	**2**	0.049	3.7	**2**	0.047	0.64
	75	**2**	0.044	0.12	**2**	0.041	0.16	**2**	0.044	7.4	**2**	0.044	3.7
	100	**2**	0.046	0.16	**2**	0.046	0.43	**2**	0.05	38	**2**	0.046	20
bupa	25	8	0.403	0.31	**7**	0.45	0.31	**7**	0.45	4.1	**7**	0.419	0.64
	50	14	0.431	1.32	**12**	0.495	589	**12**	0.495	47	**12**	0.464	989
	75	21	0.404	4.1	21	0.463	T/O	**19**	0.467	1763	24	0.419	T/O
	100	**27**	0.355	11	32	0.494	T/O	30	0.396	T/O	34	0.367	T/O
credit	25	**4**	0.202	0.11	**4**	0.202	0.08	**4**	0.202	2	**4**	0.202	0.22
	50	6	0.239	0.25	**5**	0.257	9.3	**5**	0.209	21	**5**	0.257	30.1
	75	9	0.216	0.61	**8**	0.266	1920	**8**	0.263	138	**8**	0.268	1862
	100	12	0.233	1.3	11	0.237	T/O	**10**	0.242	798	18	0.302	T/O
glass	25	**5**	0.333	0.11	**5**	0.261	0.03	**5**	0.297	12	**5**	0.261	0.2
	50	9	0.265	0.49	**8**	0.265	10.3	**8**	0.265	35	**8**	0.265	28
	75	16	0.307	1.5	**15**	0.273	T/O	**15**	0.227	736	**15**	0.227	T/O
	100	18	0.222	2.9	**17**	0.222	T/O	**17**	0.206	T/O	22	0.19	T/O
haberman	25	**5**	0.305	0.17	**5**	0.305	0.03	**5**	0.305	3.6	**5**	0.312	0.18
	50	**10**	0.246	0.94	**10**	0.332	34	**10**	0.332	30	**10**	0.246	65
	75	15	0.237	2.5	**14**	0.324	T/O	**14**	0.324	436	16	0.279	T/O
	100	21	0.278	4.5	**20**	0.289	T/O	**20**	0.33	T/O	23	0.289	T/O
pima	25	**8**	0.408	0.33	**8**	0.381	0.36	**8**	0.385	4	**8**	0.381	0.94
	50	15	0.312	0.9	**13**	0.306	2204	**13**	0.311	37	**13**	0.306	1985
	75	20	0.375	3.8	20	0.342	T/O	**19**	0.339	2641	24	0.336	T/O
	100	25	0.326	7.4	26	0.316	T/O	**23**	0.338	T/O	30	0.379	T/O
USvotes	25	**3**	0.112	0.07	**3**	0.11	0.011	**3**	0.107	0.21	**3**	0.12	0.08
	50	5	0.14	0.17	**4**	0.114	0.141	**4**	0.127	2.4	**4**	0.127	1.1
	75	5	0.119	0.28	**3**	0.131	0.183	**3**	0.131	54	**3**	0.131	33
	100	6	0.084	0.35	**4**	0.146	1.21	**4**	0.107	100	**4**	0.137	80

sometimes (quickly) finds the best value of $\mathcal{F}$ when the pseudo-Boolean programs time out, and there is no clear amelioration of the testing risk when PB-SCM finds a slightly better solution than SCM. We conclude that the greedy strategy of SCM is particularly effective.

D.5 Conclusion

We have presented a pseudo-Boolean model that encodes the core idea behind the combinatorial problem related to the Set Covering Machine. Extensive experiments have been done using three different pseudo-Boolean solvers. For the first time, empirical results show the effectiveness of the greedy approach of Marchand et Shawe-Taylor (2002) at building SCM of both small compression set and empirical risk. This is a very surprising result given the simplicity and the low complexity of the greedy algorithm.

Annexe E

Domain-Adversarial Neural Networks

Cet article a été présenté lors de l'atelier *Workshop on Transfer and Multi-Task Learning* (voir Ajakan et al., 2014), qui a eu lieu dans le cadre de la conférence *Neural Information Processing Systems*. Notons qu'une version allongée et bonifiée a été coécrite et fait l'objet d'un article soumis pour évaluation à la revue *Journal of Machine Learning Research* (voir Ganin et al., 2015).

Abstract. We introduce a new representation learning algorithm suited to the context of domain adaptation, in which data at training and test time come from similar but different distributions. Our algorithm is directly inspired by theory on domain adaptation suggesting that, for effective domain transfer to be achieved, predictions must be made based on a data representation that cannot discriminate between the training (source) and test (target) domains. We propose a training objective that implements this idea in the context of a neural network, whose hidden layer is trained to be predictive of the classification task, but uninformative as to the domain of the input. Our experiments on a sentiment analysis classification benchmark, where the target domain data available at training time is unlabeled, show that our neural network for domain adaption algorithm has better performance than either a standard neural network or an SVM, even if trained on input features extracted with the state-of-the-art marginalized stacked denoising autoencoders of Chen et al. (2012).

E.1 Introduction

The cost of generating labeled data for a new learning task is often an obstacle for applying machine learning methods. There is thus great incentive to develop ways of exploiting data from one problem that generalizes to another. Domain adaptation focuses on the situation where we have data generated from two different, but somehow similar, distributions. One

example is in the context of sentiment analysis in written reviews, where we might want to distinguish between the positive from the negative ones. While we might have labeled data for reviews of one type of products (*e.g.*, movies), we might want to be able to generalize to reviews of other products (*e.g.*, books). Domain adaptation tries to achieve such a transfer by exploiting an extra set of unlabeled training data for the new problem to which we wish to generalize (*e.g.*, unlabeled reviews of books).

One of the main approaches to achieve such a transfer is to learn a classifier *and* a representation which will favor the transfer. A large body of work exists on training both a classifier and a representation that are linear (Bruzzone et Marconcini, 2010; Germain et al., 2013; Cortes et Mohri, 2014). However, recent research has shown that non-linear neural networks can also be successful (Glorot et al., 2011). Specifically, a variant of the denoising autoencoder (Vincent et al., 2008), known as marginalized stacked denoising autoencoders (mSDA) (Chen et al., 2012), has demonstrated state-of-the-art performance on this problem. By learning a representation which is robust to input corruption noise, they have been able to learn a representation which is also more stable across changes of domain and can thus allow cross-domain transfer.

In this paper, we propose to control the stability of representation between domains explicitly into a neural network learning algorithm. This approach is motivated by theory on domain adaptation (Ben-David et al., 2006, 2010) that suggests that a good representation for cross-domain transfer is one for which an algorithm cannot learn to identify the domain of origin of the input observation. We show that this principle can be implemented into a neural network learning objective that includes a term where the network's hidden layer is working adversarially towards output connections predicting domain membership. The neural network is then simply trained by gradient descent on this objective. The success of this domain-adversarial neural network (DANN) is confirmed by extensive experiments on both toy and real world datasets. In particular, we show that DANN achieves better performances than a regular neural network and a SVM on a sentiment analysis classification benchmark. Moreover, we show that DANN can reach state-of-the-art performance by taking as input the representation learned by mSDA, confirming that minimizing domain discriminability explicitly improves over on only relying on a representation which is robust to noise.

E.2 Domain Adaptation

We consider binary classification tasks where $\mathcal{X} \subseteq \mathbb{R}^n$ is the input space and $\mathcal{Y}=\{0,1\}$ is the label set. Moreover, we have two different distributions over $\mathcal{X}\times\mathcal{Y}$, called the *source domain* D_{S} and the *target domain* D_{T}. A *domain adaptation* learning algorithm is then provided with a *labeled source sample* S drawn *i.i.d.* from D_{S}, and an *unlabeled target sample* T drawn

i.i.d. from $\mathcal{D}_{\mathrm{T}}^{\mathcal{X}}$, where $\mathcal{D}_{\mathrm{T}}^{\mathcal{X}}$ is the marginal distribution of D_{T} over $\mathcal{X}$.

$$S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m \sim (D_{\mathrm{S}})^m\,; \quad T = \{\mathbf{x}_i^t\}_{i=1}^{m'} \sim (\mathcal{D}_{\mathrm{T}}^{\mathcal{X}})^{m'}.$$

The goal of the learning algorithm is to build a classifier $\eta : \mathcal{X} \to \mathcal{Y}$ with a low *target risk*

$$R_{D_{\mathrm{T}}}(\eta) \stackrel{\mathrm{def}}{=} \Pr_{(\mathbf{x}^t, y^t) \sim D_{\mathrm{T}}} \Big(\eta(\mathbf{x}^t) \neq y^t\Big)\,,$$

while having no information about the labels of D_{T}.

E.2.1 Domain Divergence

To tackle the challenging domain adaptation task, many approaches bound the target error by the sum of the source error and a notion of distance between the source and the target distributions. These methods are intuitively justified by a simple assumption: the source risk is expected to be a good indicator of the target risk when both distributions are similar. Several notions of distance have been proposed for domain adaptation (Ben-David et al., 2006, 2010; Mansour et al., 2009a,b; Germain et al., 2013). In this paper, we focus on the $\mathcal{H}$-divergence used by Ben-David et al. (2006, 2010), and based on the earlier work of Kifer et al. (2004).

Definition E.1 (Ben-David et al., 2006, 2010; Kifer et al., 2004)**.** Given two domain distributions $\mathcal{D}_{\mathrm{S}}^{\mathcal{X}}$ and $\mathcal{D}_{\mathrm{T}}^{\mathcal{X}}$ over $\mathcal{X}$, and a hypothesis class $\mathcal{H}$, the *$\mathcal{H}$-divergence* between $\mathcal{D}_{\mathrm{S}}^{\mathcal{X}}$ and $\mathcal{D}_{\mathrm{T}}^{\mathcal{X}}$ is

$$d_{\mathcal{H}}(\mathcal{D}_{\mathrm{S}}^{\mathcal{X}}, \mathcal{D}_{\mathrm{T}}^{\mathcal{X}}) \stackrel{\mathrm{def}}{=} 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x}^s \sim \mathcal{D}_{\mathrm{S}}^{\mathcal{X}}} \left[\eta(\mathbf{x}^s) = 1\right] - \Pr_{\mathbf{x}^t \sim \mathcal{D}_{\mathrm{T}}^{\mathcal{X}}} \left[\eta(\mathbf{x}^t) = 1\right] \right|.$$

That is, the $\mathcal{H}$-divergence relies on the capacity of the hypothesis class $\mathcal{H}$ to distinguish between examples generated by $\mathcal{D}_{\mathrm{S}}^{\mathcal{X}}$ from examples generated by $\mathcal{D}_{\mathrm{T}}^{\mathcal{X}}$. Ben-David et al. (2006, 2010) proved that, for a symmetric hypothesis class $\mathcal{H}$, one can compute the *empirical $\mathcal{H}$-divergence* between two samples $S \sim (\mathcal{D}_{\mathrm{S}}^{\mathcal{X}})^m$ and $T \sim (\mathcal{D}_{\mathrm{T}}^{\mathcal{X}})^{m'}$ by computing

$$\hat{d}_{\mathcal{H}}(S, T) \stackrel{\mathrm{def}}{=} 2\left(1 - \min_{\eta \in \mathcal{H}} \left[\frac{1}{m} \sum_{i=1}^{m} I[\eta(\mathbf{x}_i^s) = 1] + \frac{1}{m'} \sum_{i=1}^{m'} I[\eta(\mathbf{x}_i^t) = 0]\right]\right), \tag{E.1}$$

where $I[a]$ is the indicator function which is 1 if predicate a is true, and 0 otherwise.

E.2.2 Proxy Distance

Ben-David et al. (2006, 2010) suggested that, even if it is generally hard to compute $\hat{d}_{\mathcal{H}}(S, T)$ exactly (*e.g.*, when $\mathcal{H}$ is the space of linear classifiers on $\mathcal{X}$), we can easily approximate it by running a learning algorithm on the problem of discriminating between source and target examples. To do so, we construct a new dataset

$$U \;=\; \{(\mathbf{x}_i^s, 1)\}_{i=1}^m \cup \{(\mathbf{x}_i^t, 0)\}_{i=1}^{m'}\,, \tag{E.2}$$

where the examples of the source sample are labeled 1 and the examples of the target sample are labeled 0. Then, the risk of the classifier trained on new dataset U approximates the "min" part of Equation (E.1). Thus, given a test error ϵ on the problem of discriminating between source and target examples, this *Proxy A-distance* (PAD) is given by

$$\hat{d}_A \;=\; 2(1-2\epsilon)\,. \tag{E.3}$$

In the experiments section of this paper, we compute the PAD value following the approach of Glorot et al. (2011); Chen et al. (2012), *i.e.*, we train a linear SVM on a subset of dataset U (Equation (E.2)), and we use the obtained classifier error on the other subset as the value of ϵ in Equation (E.3).

E.2.3 Generalization Bound on the Target Risk

The work of Ben-David et al. (2006, 2010) also showed that the $\mathcal{H}$-divergence $d_{\mathcal{H}}(\mathcal{D}_{\mathrm{S}}^{\mathcal{X}}, \mathcal{D}_{\mathrm{T}}^{\mathcal{X}})$ is upper bounded by its empirical estimate $\hat{d}_{\mathcal{H}}(S,T)$ plus a constant complexity term that depends on the *VC dimension* of $\mathcal{H}$ and the size of samples S and T. By combining this result with a similar bound on the source risk, the following theorem is obtained.

Theorem E.2 (Ben-David et al., 2006)**.** *Let $\mathcal{H}$ be a hypothesis class of VC dimension d. With probability $1-\delta$ over the choice of samples $S \sim (D_{\mathrm{S}})^m$ and $T \sim (\mathcal{D}_{\mathrm{T}}^{\mathcal{X}})^m$, for every $\eta \in \mathcal{H}$:*

$$R_{D_{\mathrm{T}}}(\eta) \;\leq\; R_S(\eta) + \sqrt{\frac{4}{m}\left(d\log\tfrac{2e\,m}{d} + \log\tfrac{4}{\delta}\right)} + \hat{d}_{\mathcal{H}}(S,T) + 4\sqrt{\frac{1}{m}\left(d\log\tfrac{2m}{d} + \log\tfrac{4}{\delta}\right)} + \beta\,,$$

with $\beta \geq \inf_{\eta^* \in \mathcal{H}} \left[R_{D_{\mathrm{S}}}(\eta^*) + R_{D_{\mathrm{T}}}(\eta^*)\right]$, and

$$R_S(\eta) \;=\; \frac{1}{m}\sum_{i=1}^{m} I\left[\eta(\mathbf{x}_i^s) \neq y_i^s\right]$$

is the empirical source risk.

The previous result tells us that $R_{D_{\mathrm{T}}}(\eta)$ can be low only when the β term is low, *i.e.*, only when there exists a classifier that can achieve a low risk on both distributions. It also tells us that, to find a classifier with a small $R_{D_{\mathrm{T}}}(\eta)$ in a given class of fixed VC dimension, the learning algorithm should minimize (in that class) a trade-off between the source risk $R_S(\eta)$ and the empirical $\mathcal{H}$-divergence $\hat{d}_{\mathcal{H}}(S,T)$. As pointed-out by Ben-David et al. (2006), a strategy to control the $\mathcal{H}$-divergence is to find a representation of the examples where both the source and the target domain are as indistinguishable as possible. Under such a representation, a hypothesis with a low source risk will, according to Theorem E.2, perform well on the target data. In this paper, we present an algorithm that directly exploits this idea.

E.3 A Domain-Adversarial Neural Network

The originality of our approach is to explicitly implement the idea exhibited by Theorem E.2 into a neural network classifier. That is, to learn a model that can generalize well from one domain to another, we ensure that the internal representation of the neural network contains no discriminative information about the origin of the input (source or target), while preserving a low risk on the source (labeled) examples.

E.3.1 Source Risk Minimization (Standard NN)

Let us consider the following standard neural network (NN) architecture with one hidden layer:

$$\begin{aligned} \mathbf{h}(\mathbf{x}) &= \mathrm{sigm}\big(\mathbf{b} + \mathbf{W}\mathbf{x}\big)\,, \\ \mathbf{f}(\mathbf{x}) &= \mathrm{softmax}\big(\mathbf{c} + \mathbf{V}\mathbf{h}(\mathbf{x})\big)\,, \end{aligned} \tag{E.4}$$

with

$$\mathrm{sigm}(\mathbf{a}) \stackrel{\text{def}}{=} \left[\frac{1}{1+\exp(-a_i)}\right]_{i=1}^{|\mathbf{a}|},$$

$$\mathrm{softmax}(\mathbf{a}) \stackrel{\text{def}}{=} \left[\frac{\exp(a_i)}{\sum_{j=1}^{|\mathbf{a}|}\exp(a_j)}\right]_{i=1}^{|\mathbf{a}|}.$$

Note that each component $f_y(\mathbf{x})$ of $\mathbf{f}(\mathbf{x})$ denotes the conditional probability that the neural network assigns $\mathbf{x}$ to class y. Given a training source sample $S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m$, the natural classification loss to use is the negative log-probability of the correct label:

$$\mathcal{L}\big(\mathbf{f}(\mathbf{x}), y\big) \stackrel{\text{def}}{=} \log\frac{1}{f_y(\mathbf{x})}\,.$$

This leads to the following learning problem on the source domain.

$$\min_{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c}} \left[\frac{1}{m}\sum_{i=1}^m \mathcal{L}\big(\mathbf{f}(\mathbf{x}_i^s), y_i^s\big)\right]. \tag{E.5}$$

We view the output of the hidden layer $\mathbf{h}(\cdot)$ (Equation (E.4)) as the internal representation of the neural network. Thus, we denote the source sample representations as

$$\mathbf{h}(S) \stackrel{\text{def}}{=} \big\{\mathbf{h}(\mathbf{x}_i^s)\big\}_{i=1}^m\,.$$

E.3.2 A Domain Adaptation Regularizer

Now, consider an unlabeled sample from the target domain $T = \{\mathbf{x}_i^t\}_{i=1}^{m'}$ and the corresponding representations $\mathbf{h}(T) = \{\mathbf{h}(\mathbf{x}_i^t)\}_{i=1}^{m'}$. Based on Equation (E.1), the empirical $\mathcal{H}$-divergence of a symmetric hypothesis class $\mathcal{H}$ between samples $\mathbf{h}(S)$ and $\mathbf{h}(T)$ is given by

$$\hat{d}_{\mathcal{H}}\big(\mathbf{h}(S), \mathbf{h}(T)\big) = 2\left(1 - \min_{\eta\in\mathcal{H}}\left[\frac{1}{m}\sum_{i=1}^m I\big[\eta(\mathbf{h}(\mathbf{x}_i^s)) = 1\big] + \frac{1}{m'}\sum_{i=1}^{m'} I\big[\eta(\mathbf{h}(\mathbf{x}_i^t)) = 0\big]\right]\right). \tag{E.6}$$

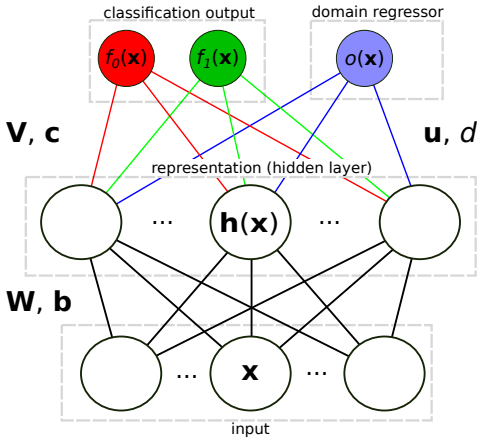


Figure E.1 – DANN architecture

Let us consider $\mathcal{H}$ as the class of hyperplanes in the representation space. Inspired by the Proxy A-distance (see Section E.2.2), we suggest estimating the "min" part of Equation (E.6) by a logistic regressor that model the probability that a given input (either $\mathbf{x}^s$ or $\mathbf{x}^t$) is from the source domain $\mathcal{D}_{\mathrm{S}}^{\mathcal{X}}$ (denoted $z=1$) or the target domain $\mathcal{D}_{\mathrm{T}}^{\mathcal{X}}$ (denoted $z=0$):

$$p(z=1 \mid \boldsymbol{\phi}) \;=\; o(\boldsymbol{\phi}) \;\stackrel{\text{def}}{=}\; \mathrm{sigm}(d + \mathbf{u}^\top \boldsymbol{\phi})\,, \tag{E.7}$$

where $\boldsymbol{\phi}$ is either $\mathbf{h}(\mathbf{x}^s)$ or $\mathbf{h}(\mathbf{x}^t)$. Hence, the function $o(\cdot)$ is a *domain regressor*.

This enables us to add a domain adaptation term to the objective of Equation (E.5), giving the following problem to solve:

$$\min_{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c}} \left[\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\mathbf{f}(\mathbf{x}_i^s), y_i^s) + \lambda \max_{\mathbf{u},d} \left(-\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}^d(o(\mathbf{x}_i^s), 1) - \frac{1}{m'} \sum_{i=1}^{m'} \mathcal{L}^d(o(\mathbf{x}_i^t), 0) \right) \right], \tag{E.8}$$

where the hyper-parameter $\lambda > 0$ weights the domain adaptation regularization term and

$$\mathcal{L}^d(o(\mathbf{x}), z) \;=\; -z \log\left(o(\mathbf{x})\right) - (1-z) \log\left(1-o(\mathbf{x})\right).$$

In line with Theorem E.2, this optimization problem implements a trade-off between the minimization of the source risk $R_S(\cdot)$ and the divergence $\hat{d}_{\mathcal{H}}(\cdot,\cdot)$. The hyper-parameter λ is then used to tune the trade-off between these two quantities during the learning process.

E.3.3 Learning Algorithm (DANN)

We see that Equation (E.8) involves a maximization operation. Hence, the neural network (parametrized by $\{\mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}\}$) and the domain regressor (parametrized by $\{\mathbf{u}, d\}$) are competing against each other, in an adversarial way, for that term. The obtained *domain adversarial neural network* (DANN) is illustrated by Figure E.1. In DANN, the hidden layer $\mathbf{h}(\cdot)$ maps an example (either source or target) into a representation in which the output layer $\mathbf{f}(\cdot)$ accurately classifies the source sample, while the domain regressor $o(\cdot)$ is unable to detect if an example belongs to the source sample or the target sample.

Algorithm 6 DANN

Input: samples $S = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^m$ and $T = \{\mathbf{x}_i^t\}_{i=1}^{m'}$,
hidden layer size l, adaptation parameter λ, learning rate α.
Output: neural network $\{\mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}\}$

$\mathbf{W}, \mathbf{V} \leftarrow$ random_init(l)
$\mathbf{b}, \mathbf{c}, \mathbf{u}, d \leftarrow 0$
while stopping criteria is not met **do**
  for i from 1 to m **do**
    `# Forward propagation`
    $\mathbf{h}(\mathbf{x}_i^s) \leftarrow \text{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}_i^s)$
    $\mathbf{f}(\mathbf{x}_i^s) \leftarrow \text{softmax}(\mathbf{c} + \mathbf{V}\mathbf{h}(\mathbf{x}_i^s))$
    `# Backpropagation`
    $\Delta_{\mathbf{c}} \leftarrow -(\mathbf{e}(y_i^s) - \mathbf{f}(\mathbf{x}_i^s))$
    $\Delta_{\mathbf{V}} \leftarrow \Delta_{\mathbf{c}}\, \mathbf{h}(\mathbf{x}_i^s)^\top$
    $\Delta_{\mathbf{b}} \leftarrow \left(\mathbf{V}^\top \Delta_{\mathbf{c}}\right) \odot \mathbf{h}(\mathbf{x}_i^s) \odot (1 - \mathbf{h}(\mathbf{x}_i^s))$
    $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{b}} \cdot (\mathbf{x}_i^s)^\top$
    `# Domain adaptation regularizer...`
    `# ...from current domain`
    $o(\mathbf{x}_i^s) \leftarrow \text{sigm}(d + \mathbf{u}^\top \mathbf{h}(\mathbf{x}_i^s))$
    $\Delta_d \leftarrow \lambda(1 - o(\mathbf{x}_i^s))$; $\Delta_{\mathbf{u}} \leftarrow \lambda(1 - o(\mathbf{x}_i^s))\mathbf{h}(\mathbf{x}_i^s)$
    $\text{tmp} \leftarrow \lambda(1 - o(\mathbf{x}_i^s))\mathbf{u} \odot \mathbf{h}(\mathbf{x}_i^s) \odot (1 - \mathbf{h}(\mathbf{x}_i^s))$
    $\Delta_{\mathbf{b}} \leftarrow \Delta_{\mathbf{b}} + \text{tmp}$; $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{W}} + \text{tmp} \cdot (\mathbf{x}_i^s)^\top$
    `# ...from other domain`
    $j \leftarrow$ uniform_integer$(1, \ldots, m')$
    $\mathbf{h}(\mathbf{x}_j^t) \leftarrow \text{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}_j^t)$
    $o(\mathbf{x}_j^t) \leftarrow \text{sigm}(d + \mathbf{u}^\top \mathbf{h}(\mathbf{x}_j^t))$
    $\Delta_d \leftarrow \Delta_d - \lambda o(\mathbf{x}_j^t)$; $\Delta_{\mathbf{u}} \leftarrow \Delta_{\mathbf{u}} - \lambda o(\mathbf{x}_j^t)\mathbf{h}(\mathbf{x}_j^t)$
    $\text{tmp} \leftarrow -\lambda o(\mathbf{x}_j^t)\mathbf{u} \odot \mathbf{h}(\mathbf{x}_j^t) \odot (1 - \mathbf{h}(\mathbf{x}_j^t))$
    $\Delta_{\mathbf{b}} \leftarrow \Delta_{\mathbf{b}} + \text{tmp}$; $\Delta_{\mathbf{W}} \leftarrow \Delta_{\mathbf{W}} + \text{tmp} \cdot (\mathbf{x}_j^t)^\top$
    `# Update neural network parameters`
    $\mathbf{W} \leftarrow \mathbf{W} - \alpha\Delta_{\mathbf{W}}$; $\mathbf{V} \leftarrow \mathbf{V} - \alpha\Delta_{\mathbf{V}}$
    $\mathbf{b} \leftarrow \mathbf{b} - \alpha\Delta_{\mathbf{b}}$; $\mathbf{c} \leftarrow \mathbf{c} - \alpha\Delta_{\mathbf{c}}$
    `# Update domain classifier parameters`
    $\mathbf{u} \leftarrow \mathbf{u} + \alpha\Delta_{\mathbf{u}}$; $d \leftarrow d + \alpha\Delta_d$
  end for
end while

To optimize Equation (E.8), one option would be to follow a hard-EM approach, where we would alternate between optimizing until convergence the adversarial parameters $\mathbf{u}, d$ and the other regular neural network parameters $\mathbf{W}, \mathbf{V}, \mathbf{b}, \mathbf{c}$. However, we've found that a simpler stochastic gradient descent (SGD) approach is sufficient and works well in practice. Here, an SGD approach consists in sampling a pair of source and target example $\mathbf{x}_i^s, \mathbf{x}_j^t$ and updating a gradient step update of all parameters of DANN. Crucially, while the update of the regular parameters follows as usual the opposite direction of the gradient, for the adversarial parameters $\mathbf{u}, d$ the step must follow the gradient's direction (since we maximize with respect to them, instead of minimizing). The algorithm is detailed in Algorithm 6. In the pseudocode, we use $\mathbf{e}(y)$ to represent a "one-hot" vector, consisting of all 0s except for a 1 at position y.

Also, $\odot$ is the element-wise product.

For each experiment described in this paper, we used *early stopping* as the stopping criteria: we split the source labeled sample to use 90% as the training set S and the remaining 10% as a validation set S_V. We stop the learning process when the risk on S_V is minimal.

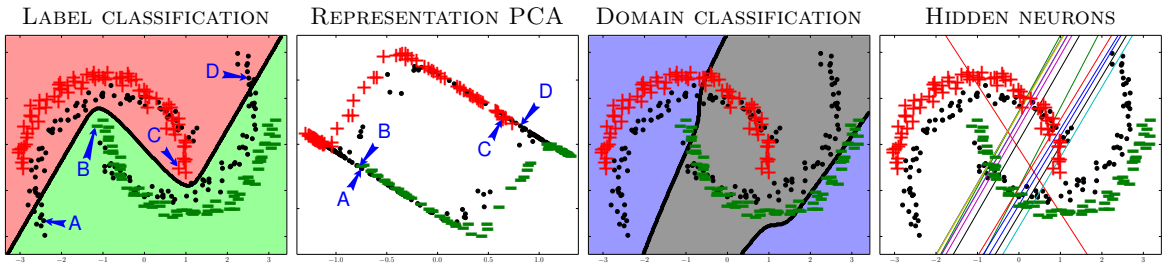
E.4 Related Work

As mentioned previously, the general approach of achieving domain adaptation by learning a new data representation has been explored under many facets. A large part of the literature however has focused mainly on linear hypothesis (see for instance Blitzer et al., 2006b; Bruzzone et Marconcini, 2010; Germain et al., 2013; Baktashmotlagh et al., 2013; Cortes et Mohri, 2014). More recently, non-linear representations have become increasingly studied, including neural network representations (Glorot et al., 2011; Li et al., 2014) and most notably the state-of-the-art mSDA (Chen et al., 2012). That literature has mostly focused on exploiting the principle of robust representations, based on the denoising autoencoder paradigm (Vincent et al., 2008). One of the contribution of this work is to show that domain discriminability is another principle that is complementary to robustness and can improve cross-domain adaptation.

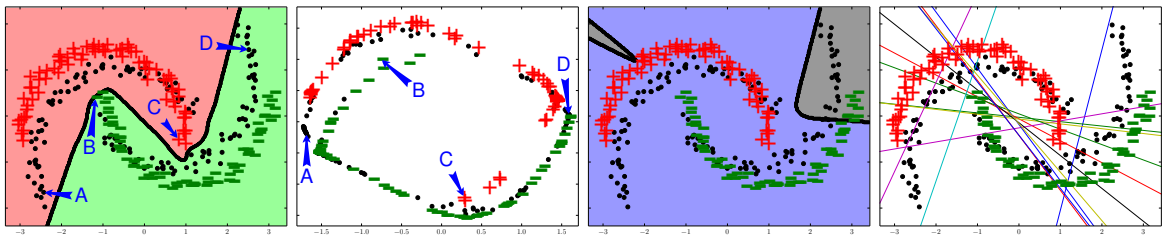
What distinguishes this work from most of the domain adaptation literature is DANN's inspiration from the theoretical work of Ben-David et al. (2006, 2010). Indeed, DANN directly optimizes the notion of $\mathcal{H}$-divergence. We do note the work of Huang et Yates (2012), in which HMM representations are learned for word tagging using a posterior regularizer that is also inspired by Ben-David et al.'s work. In addition to the tasks being different (word tagging versus sentiment classification), we would argue that DANN learning objective more closely optimizes the $\mathcal{H}$-divergence, with Huang et Yates (2012) relying on cruder approximations for efficiency reasons.

The idea of learning representations that are indiscriminate of some auxiliary label has also been explored in other contexts. For instance, Zemel et al. (2013) proposes the notion of fair representations, which are indiscriminate to whether an example belongs to some identified set of groups. The resulting algorithm is different from DANN and not directly derived from the $\mathcal{H}$-divergence.

Finally, we mention some related research on using an adversarial (minimax) formulation to learn a model, such as a classifier or a neural network, from data. There has been work on learning linear classifiers that are robust to changes in the input distribution, based on a minimax formulation (Bagnell, 2005; Liu et Ziebart, 2014). This work however assumes that a good feature representation of the input for a linear classifier is available and doesn't address the problem of learning it. We also note the work of Goodfellow et al. (2014), who



(a) Standard NN and a *non adverserial* domain regressor (Algorithm 6, without Lines 21 and 28)



(b) DANN (Algorithm 6)

Figure E.2 – The *inter-twinning moons* toy problem. We adopt the colors of Figure E.1 for classification output and domain regressor value.

propose generative adversarial networks to learn a good generative model of the true data distribution. This work shares with DANN the use of an adversarial objective, applying it instead to the unsupervised problem of generative modeling.

E.5 Experiments

E.5.1 Toy Problem

As a first experiment, we study the behavior of the proposed DANN algorithm on a variant of the *inter-twinning moons* 2D problem, where the target distribution is a rotation of the source distribution. For the source sample S, we generate a lower moon and an upper moon labeled 0 and 1 respectively, each of which containing 150 examples. The target sample T is obtained by generating a sample in the same way as S (without keeping the labels) and then by rotating each example by 35°. Thus, T contains 300 unlabeled examples. In Figure E.2, the examples from S are represented by "+"and "−", and the examples from T are represented by black dots.

We study the adaptation capability of DANN by comparing it to the standard NN. In our experiments, both algorithms share the same network architecture, with a hidden layer size of 15 neurons. We even train NN using the same procedure as DANN. That is, we keep updating the domain regressor component using target sample T (with a hyper-parameter $\lambda = 6$; the same value used for DANN), but we disable the *adversarial* back-propagation into the hidden

layer. To do so, we execute Algorithm 6 by omitting the lines numbered 21 and 28. In this way, we obtain a NN learning algorithm – based on the source risk minimization of Equation (E.5) – and simultaneously train the domain regressor of Equation (E.7) to discriminate between source and target domains. Using this toy experiment, we will first illustrate how DANN adapts its decision boundary compared to NN. Moreover, we will also illustrate how the representation given by the hidden layer is less adapted to the domain task with DANN than it is with NN. The results are illustrated in Figure E.2, where the graphs in part (a) relate to the standard NN, and the graphs in part (b) relate to DANN. By looking at the corresponding (a) and (b) graphs in each column, we compare NN and DANN from four different perspectives, described in detail below.

Label classification. The first column of Figure E.2 shows the decision boundaries of DANN and NN on the problem of predicting the labels of both source and the target examples. As expected, NN accurately classifies the two classes of the source sample S, but is *not fully adapted* to the target sample T. On the contrary, the decision boundary of DANN perfectly classifies examples from both source and target samples. DANN clearly adapts here to the target distribution.

Representation PCA. To analyze how the domain adaptation regularizer affects the representation $\mathbf{h}(\cdot)$ provided by the hidden layer, the second column of Figure E.2 presents a principal component analysis (PCA) on the set of all representations of source and target data points, *i.e.*, $\mathbf{h}(S) \cup \mathbf{h}(T)$. Thus, given the trained network (NN or DANN), every point from S and T is mapped into a 15-dimensional feature space through the hidden layer, and projected back into a two-dimensional plane defined by the first two principal components. In the DANN-PCA representation, we observe that target points are homogeneously spread out among the source points. In the NN-PCA representation, clusters of target points containing very few source points are clearly visible. Hence, the task of labeling the target points seems easier to perform on the DANN-PCA representation.

To push the analysis further, four crucial data points identified by A, B, C and D in the graphs of the first column (which correspond to the moon extremities in the original space) are represented again on the graphs of the second column. We observe that points A and B are very close to each other in the NN-PCA representation, while they clearly belong to different classes. The same happens to points C and D. Conversely, these four points are located at opposite corners in the DANN-PCA representation. Note also that the target point A (resp. D) – which is difficult to classify in the original space – is located in the "+"cluster (resp. "−"cluster) in the DANN-PCA representation. Therefore, the representation promoted by DANN is more suited for the domain adaptation task.

Domain classification. The third column of Figure E.2 shows the decision boundary on the domain classification problem, which is given by the domain regressor $o(\cdot)$ of Equation (E.7).

More precisely, $\mathbf{x}$ is classified as a source example when $o(\mathbf{x}) \geq 0.5$, and is classified as a domain example otherwise. Remember that, during the learning process of DANN, the $o(\cdot)$ regressor struggles to discriminate between source and target domains, while the hidden representation $\mathbf{h}(\cdot)$ is *adversarially* updated to prevent it to succeed. As explained above, we trained the domain regressor during the learning process of NN, but without allowing it to influence the learned representation $\mathbf{h}(\cdot)$.

On one hand, the DANN domain regressor utterly fails to discriminate the source and target distributions. On the other hand, the NN domain regressor shows a better (although imperfect) discriminant. This again corroborates that the DANN representation doesn't allow discriminating between domains.

Hidden neurons. In the plot of the last column of Figure E.2, the lines show the decision surfaces of the hidden layer neurons (defined by Equation (E.4)). In other words, each of the fifteen plot line corresponds to the points $\mathbf{x} \in \mathbb{R}^2$ for which the ith component of $\mathbf{h}(\mathbf{x})$ equals $\frac{1}{2}$, for $i \in \{1, \ldots, 15\}$.

We observe that the neurons of NN are grouped in three clusters, each one allowing to generate a straight line part of the curved decision boundary for the label classification problem. However, most of these neurons are also able to (roughly) capture the rotation angle of the domain classification problem. Hence, we observe that the adaptation regularizer of DANN prevents these kinds of neurons to be produced. It is indeed striking to see that the two predominant patterns in the NN neurons (*i.e.*, the two parallel lines crossing the plane from the lower left corner to the upper right corner) are absent among DANN neurons.

E.5.2 Sentiment Analysis Dataset

In this section, we compare the performance of our proposed DANN algorithm to a standard neural network with one hidden layer (NN) described by Equation (E.5), and a Support Vector Machine (SVM) with a linear kernel. To select the hyper-parameters of each of these algorithms, we use grid search and a very small validation set which consists in 100 labeled examples from the target domain. Finally, we select the classifiers having the lowest target validation risk.

We compare the algorithms on the *Amazon reviews* dataset, as pre-processed by Chen et al. (2012). This dataset includes four domains, each one composed of reviews of a specific kind of product (books, dvd disks, electronics, and kitchen appliances). Reviews are encoded in 5 000 dimensional feature vectors of unigrams and bigrams, and labels are binary: "0" if the product is ranked up to 3 stars, and "1" if the product is ranked 4 or 5 stars.

We perform twelve domain adaptation tasks. For example, "books → dvd" corresponds to the task for which books is the source domain and dvd disks the target one. All learning

Table E.1 – Error rates on the Amazon reviews dataset and Pairwise Poisson binomial test.

(a) Error rates on the Amazon reviews dataset

	Original data			**mSDA representation**		
source name → target name	DANN	NN	SVM	DANN	NN	SVM
books → dvd	0.201	**0.199**	0.206	0.176	**0.171**	0.175
books → electronics	**0.246**	0.251	0.256	**0.197**	0.228	0.244
books → kitchen	0.230	0.235	**0.229**	0.169	**0.166**	0.172
dvd → books	**0.247**	0.261	0.269	0.176	**0.173**	0.176
dvd → electronics	**0.247**	0.256	0.249	**0.181**	0.234	0.220
dvd → kitchen	**0.227**	**0.227**	0.233	**0.151**	0.153	0.178
electronics → books	**0.280**	0.281	0.290	0.237	0.241	**0.229**
electronics → dvd	**0.273**	0.277	0.278	**0.216**	0.228	0.261
electronics → kitchen	**0.148**	0.149	0.163	**0.118**	0.126	0.137
kitchen → books	**0.283**	0.288	0.325	**0.222**	0.226	0.234
kitchen → dvd	**0.261**	**0.261**	0.274	**0.208**	0.214	0.209
kitchen → electronics	0.161	0.161	**0.158**	0.141	**0.136**	0.138

(b) Pairwise Poisson binomial test

Original data			
	DANN	NN	SVM
DANN	0.50	**0.90**	**0.97**
NN	0.10	0.50	0.87
SVM	0.03	0.13	0.50

mSDA representations			
	DANN	NN	SVM
DANN	0.50	**0.82**	**0.88**
NN	0.18	0.50	0.90
SVM	0.12	0.10	0.50

algorithms are given 2 000 labeled source examples and 2 000 unlabeled target examples. Then, we evaluate them on separate target test sets (between 3 000 and 6 000 examples). Note that NN and SVM don't use the unlabeled target sample for learning. Here are more details about the procedure used for each learning algorithms.

DANN. The adaptation parameter λ is chosen among 9 values between 10^{-2} and 1 on a logarithmic scale. The hidden layer size l is either $1, 5, 12, 25, 50, 75, 100, 150$, or 200. Finally, the learning rate α is fixed at 10^{-3}.

NN. We use exactly the same hyper-parameters grid and training procedure as DANN above, except that we don't need an adaptation parameter. Note that one can train NN by using the DANN implementation (Algorithm 6) with $\lambda = 0$.

SVM. The hyper-parameter C of the SVM is chosen among 10 values between 10^{-5} and 1 on a logarithmic scale. This range of values is the same used by Chen et al. (2012) in their experiments.

The "Original data" part of Table E.1a shows the target test risk of all algorithms, and Table E.1b reports the probability that one algorithm is significantly better than another

according to the Poisson binomial test (Lacoste et al., 2012). We note that DANN has a significantly better performance than NN and SVM, with respective probabilities **0.90** and **0.97**. As the only difference between DANN and NN is the domain adaptation regularizer, we conclude that our approach successfully helps to find a representation suitable for the target domain.

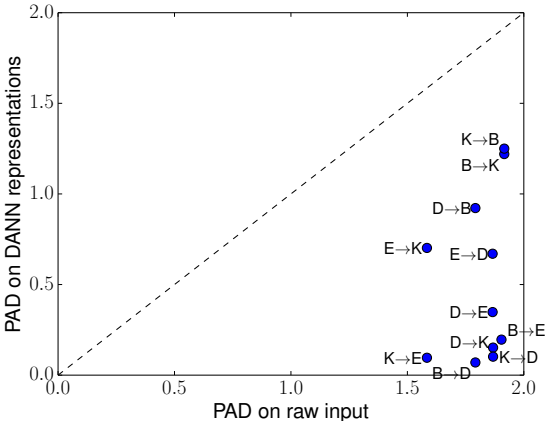
E.5.3 Combining DANN with Autoencoders

We now wonder whether our DANN algorithm can improve on the representation learned by the state-of-the-art *Marginalized Stacked Denoising Autoencoders* (mSDA) proposed by Chen et al. (2012). In brief, mSDA is an unsupervised algorithm that learns a new robust feature representation of the training samples. It takes the unlabeled parts of both source and target samples to learn a feature map from the input space $\mathcal{X}$ to a new representation space. As a *denoising autoencoder*, it finds a feature representation from which one can (approximately) reconstruct the original features of an example from its noisy counterpart. Chen et al. (2012) showed that using mSDA with a linear SVM classifier gives state-of-the-art performance on the *Amazon reviews* datasets. As an alternative to the SVM, we propose to apply our DANN algorithm on the same representations generated by mSDA (using representations of both source and target samples). Note that, even if mSDA and DANN are two representation learning approaches, they optimize different objectives, which can be complementary.

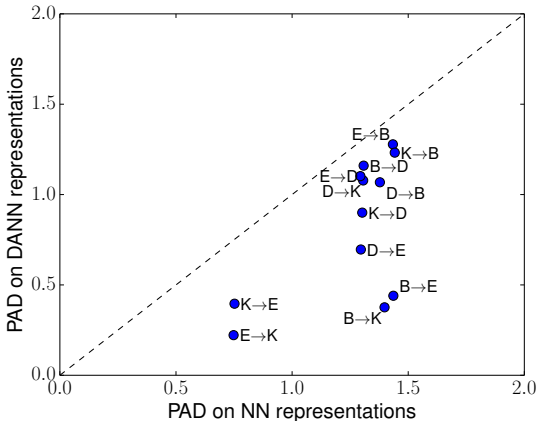
We perform this experiment on the same *amazon reviews* dataset described in the previous subsection. For each pair source-target, we generate the mSDA representations using a corruption probability of 50% and a number of layers of 5. We then execute the three learning algorithms (DANN, NN, and SVM) on these representations. More precisely, following the experimental procedure of Chen et al. (2012), we use the concatenation of the output of the 5 layers and the original input as the new representation. Thus, each example is now encoded in a vector of 30 000 dimensions. Note that we use the same grid search as in Subsection E.5.2, but with a learning rate α of 10^{-4} for both DANN and NN. The results of "mSDA representation" columns in Table E.1a confirm that combining mSDA and DANN is a sound approach. Indeed, the Poisson binomial test shows that DANN has a better performance than NN and SVM with probabilities **0.82** and **0.88** respectively, as reported in Table E.1b.

E.5.4 Proxy A-distance

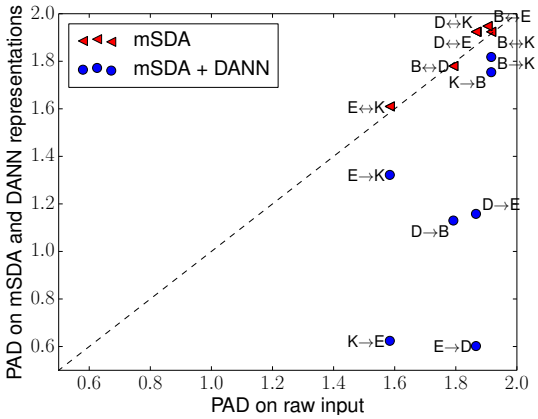
The theoretical foundation of DANN is the domain adaptation theory of Ben-David et al. (2006, 2010). We claimed that DANN finds a representation in which the source and the target example are hardly distinguishable. Our toy experiment of Section E.5.1 already points out some evidences, but we want to confirm it on real data. To do so, we compare the Proxy A-distance (PAD) on various representations of the *Amazon Reviews* dataset. These representations are obtained by running either NN, DANN, mSDA, or mSDA and DANN



(a) DANN on *Original data*.



(b) DANN & NN with 100 hidden neurons.



(c) DANN on *mSDA representations*.

Figure E.3 – Proxy A-distances (PAD). Note that the PAD values of mSDA representations are symmetric when swapping source and target samples. Also, some DANN results, when used on top of mSDA, have PAD values lower than 0.5 and doesn't appear on Figure (c).

combined. Recall that PAD, as described in Section E.2.2, is a metric estimating the similarity of the source and the target representations. More precisely, to obtain a PAD value, we use the following procedure: (1) we construct the dataset U of Equation (E.2) using both source and target representations of the training samples; (2) we randomly split U in two subsets of equal size; (3) we train linear SVMs on the first subset of U using a large range of C values; (4) we compute the error of all obtained classifiers on the second subset of U; and (5) we use the lowest error to compute the PAD value of Equation (E.3).

Firstly, Figure E.3a compares the PAD of DANN representations obtained in the experiments of Section E.5.2 (using the hyper-parameters values leading to the results of Table E.1) to the PAD computed on raw data. As expected, the PAD values are driven down by the DANN representations.

Secondly, Figure E.3b compares the PAD of DANN representations to the PAD of standard

NN representations. As the PAD is influenced by the hidden layer size (the discriminating power tends to increase with the dimension of the representation), we fix here the size to 100 neurons for both algorithms. We also fix the adaptation parameter of DANN to $\lambda \simeq 0.31$ as it was the value that has been selected most of the time during our preceding experiments on the *Amazon Reviews* dataset. Again, DANN is clearly leading to the lowest PAD values.

Lastly, Figure E.3c presents two sets of results related to Section E.5.3 experiments. On one hand, we reproduce the results of Chen et al. (2012), which noticed that the mSDA representations gave greater PAD values than those obtained with the original (raw) data. Although the mSDA approach clearly helps to adapt to the target task, it seems to contradict the theory of Ben-David et al.. On the other hand, we observe that, when running DANN on top of mSDA (using the hyper-parameters values leading to the results of Table E.1), the obtained representations have much lower PAD values. These observations might explain the improvements provided by DANN when combined with the mSDA procedure.

E.6 Conclusion and Future Work

In this paper, we have proposed a neural network algorithm, named DANN, that is strongly inspired by the domain adaptation theory of Ben-David et al. (2006, 2010). The main idea behind DANN is to encourage the network's hidden layer to learn a representation which is predictive of the source example labels, but uninformative about the domain of the input (source or target). Extensive experiments on the *inter-twinning moons* toy problem and *Amazon reviews* sentiment analysis dataset have shown the effectiveness of this strategy. Notably, we achieved state-of-the-art performances when combining DANN with the mSDA autoencoders of Chen et al. (2012), which turned out to be two complementary representation learning approaches.

We believe that our domain adaptation regularizer that we develop for the DANN algorithm can be incorporated into many other learning algorithms. Natural extensions of our work would be deeper network architectures, multi-source adaptation problems and other learning tasks beyond the basic binary classification setting. We also intend to meld the DANN approach with denoising autoencoders, to potentially improve on the two steps procedure of Section E.5.3.

Index

Bibliographie

Achterberg, T. 2004, *SCIP-a framework to integrate constraint and mixed integer programming*, Konrad-Zuse-Zentrum für Informationstechnik.

Ajakan, H., P. Germain, H. Larochelle, F. Laviolette et M. Marchand. 2014, «Domain-adversarial neural networks», *NIPS 2014 Workshop on Transfer and Multi-task learning : Theory Meets Practice*. URL http://arxiv.org/abs/1412.4446.

Ambroladze, A., E. Parrado-Hernández et J. Shawe-Taylor. 2006, «Tighter PAC-Bayes bounds», dans *NIPS*, p. 9–16.

Bach, F. 2014, «Adaptivity of averaged stochastic gradient descent to local strong convexity for logistic regression», *Journal of Machine Learning Research*, vol. 15, n° 1, p. 595–627.

Bache, K. et M. Lichman. 2013, «UCI machine learning repository», URL http://archive.ics.uci.edu/ml.

Bagnell, J. A. 2005, «Robust supervised learning.», dans *Proceedings of 20th National Conference on Artificial Intelligence*, AAAI Press / The MIT Press, p. 714–719.

Baktashmotlagh, M., M. Harandi, B. Lovell et M. Salzmann. 2013, «Unsupervised domain adaptation by domain invariant projection», dans *Proceedings of the 2013 IEEE International Conference on Computer Vision*, p. 769–776.

Banerjee, A. 2006, «On Bayesian bounds», dans *ICML*, p. 81–88.

Bégin, L., P. Germain, F. Laviolette et J.-F. Roy. 2014, «PAC-Bayesian theory for transductive learning», dans *AISTATS*, p. 105–113.

Ben-David, S., J. Blitzer, K. Crammer, A. Kulesza, F. Pereira et J. W. Vaughan. 2010, «A theory of learning from different domains», *Machine Learning*, vol. 79, n° 1-2, p. 151–175.

Ben-David, S., J. Blitzer, K. Crammer et F. Pereira. 2006, «Analysis of representations for domain adaptation», dans *NIPS*, p. 137–144.

Bishop, C. M. 2006, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Blitzer, J., R. T. McDonald et F. Pereira. 2006a, «Domain adaptation with structural correspondence learning», dans *EMNLP*, p. 120–128.

Blitzer, J., R. T. McDonald et F. Pereira. 2006b, «Domain adaptation with structural correspondence learning», dans *Conference on Empirical Methods in Natural Language Processing*, p. 120–128.

Bordes, A., S. Ertekin, J. Weston et L. Bottou. 2005, «Fast kernel classifiers with online and active learning», *Journal of Machine Learning Research*, vol. 6, p. 1579–1619.

Boser, B. E., I. Guyon et V. Vapnik. 1992, «A training algorithm for optimal margin classifiers», dans *COLT*, p. 144–152.

Breiman, L. 1996, «Bagging predictors», *Machine Learning*, vol. 24, n° 2, p. 123–140.

Breiman, L. 2001, «Random forests», *Machine Learning*, vol. 45, n° 1, p. 5–32.

Bruzzone, L. et M. Marconcini. 2010, «Domain adaptation problems : A DASVM classification technique and a circular validation strategy», *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, n° 5, p. 770–787.

Catoni, O. 2007, «PAC-Bayesian supervised classification : the thermodynamics of statistical learning», *arXiv preprint*. URL http://arxiv.org/abs/0712.0248.

Chang, C.-C. et C.-J. Lin. 2011, «LIBSVM : A library for support vector machines», *ACM Transactions on Intelligent Systems and Technology*, vol. 2, p. 27 :1–27 :27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chen, M., K. Q. Weinberger et J. Blitzer. 2011, «Co-training for domain adaptation», dans *NIPS*, p. 2456–2464.

Chen, M., Z. E. Xu, K. Q. Weinberger et F. Sha. 2012, «Marginalized denoising autoencoders for domain adaptation», dans *ICML*, p. 767–774.

Chen, Y., E. K. Garcia, M. R. Gupta, A. Rahimi et L. Cazzanti. 2009a, «Similarity-based classification : Concepts and algorithms», *Journal of Machine Learning Research*, vol. 10, doi :10.1145/1577069.1577096, p. 747–776.

Chen, Y., M. R. Gupta et B. Recht. 2009b, «Learning kernels from indefinite similarities», dans *ICML*, p. 145–152.

Cortes, C. et M. Mohri. 2014, «Domain adaptation and sample bias correction theory and algorithm for regression», *Theor. Comput. Sci.*, vol. 519, p. 103–126.

Cortes, C. et V. Vapnik. 1995, «Support-vector networks», *Machine Learning*, vol. 20, n° 3, p. 273–297.

Cover, T. M. et J. A. Thomas. 1991, *Elements of Information Theory*, Wiley.

Derbeko, P., R. El-Yaniv et R. Meir. 2004, «Explicit learning curves for transduction and application to clustering and compression algorithms», *J. Artif. Intell. Res. (JAIR)*, vol. 22, p. 117–142.

Fard, M. M. et J. Pineau. 2010, «PAC-Bayesian model selection for reinforcement learning», dans *NIPS*, p. 1624–1632.

Fard, M. M., J. Pineau et C. Szepesvári. 2011, «PAC-Bayesian policy evaluation for reinforcement learning», dans *UAI*, p. 195–202.

Floyd, S. et M. Warmuth. 1995, «Sample compression, learnability, and the vapnik-chervonenkis dimension», *Machine Learning*, vol. 21, p. 269–304.

Freund, Y. et R. E. Schapire. 1997, «A decision-theoretic generalization of on-line learning and an application to boosting», *Journal of Computer and System Sciences*, vol. 55, n° 1, p. 119–139.

Ganin, Y., E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand et V. S. Lempitsky. 2015, «Domain-adversarial training of neural networks», *ArXiv e-prints*. URL http://arxiv.org/abs/1505.07818, (en cours de révision pour publication au moment du dépôt de la thèse).

Germain, P. 2009, *Algorithmes d'apprentissage automatique inspirés de la théorie PAC-Bayes*, mémoire de maîtrise, Université Laval. URL http://www.theses.ulaval.ca/2009/26191/.

Germain, P., S. Giguère, J. Roy, B. Zirakiza, F. Laviolette et C. Quimper. 2012a, «A pseudo-Boolean set covering machine», dans *Principles and Practice of Constraint Programming - 18th International Conference, CP 2012*, p. 916–924.

Germain, P., A. Habrard, F. Laviolette et E. Morvant. 2012b, «PAC-Bayesian learning and domain adaptation», *NIPS 2012 Workshop on Multi-Trade-offs in Machine Learning*. URL http://arxiv.org/abs/1212.2340.

Germain, P., A. Habrard, F. Laviolette et E. Morvant. 2013, «A PAC-Bayesian approach for domain adaptation with specialization to linear classifiers», dans *ICML*, p. 738–746.

Germain, P., A. Habrard, F. Laviolette et E. Morvant. 2014, «An improvement to the domain adaptation bound in a PAC-bayesian context», *NIPS 2014 Workshop on Transfer and Multi-task learning : Theory Meets Practice*. URL https://hal.archives-ouvertes.fr/hal-01093565.

Germain, P., A. Habrard, F. Laviolette et E. Morvant. 2015a, «PAC-Bayesian theorems for domain adaptation with specialization to linear classifiers», *ArXiv e-prints*. URL http://arxiv.org/abs/1503.06944, (en cours de révision pour publication au moment du dépôt de la thèse).

Germain, P., A. Lacasse, F. Laviolette et M. Marchand. 2009a, «PAC-Bayesian learning of linear classifiers», dans *ICML*, p. 353–360.

Germain, P., A. Lacasse, F. Laviolette, M. Marchand et J.-F. Roy. 2015b, «Risk bounds for the majority vote : From a PAC-Bayesian analysis to a learning algorithm», *Journal of Machine Learning Research*, vol. 16, p. 787–860. URL http://jmlr.org/papers/v16/germain15a.html.

Germain, P., A. Lacasse, F. Laviolette, M. Marchand et S. Shanian. 2009b, «From PAC-Bayes bounds to KL regularization», dans *NIPS*, p. 603–610.

Germain, P., A. Lacoste, F. Laviolette, M. Marchand et S. Shanian. 2011, «A PAC-Bayes sample-compression approach to kernel methods», dans *ICML*, p. 297–304.

Giguère, S., F. Laviolette, M. Marchand et K. Sylla. 2013, «Risk bounds and learning algorithms for the regression approach to structured output prediction», dans *ICML (1)*, p. 107–114.

Glorot, X., A. Bordes et Y. Bengio. 2011, «Domain adaptation for large-scale sentiment classification : A deep learning approach», dans *ICML*, p. 513–520.

Goodfellow, I. J., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville et Y. Bengio. 2014, «Generative adversarial nets», dans *NIPS*, p. 2672–2680.

Higgs, M. et J. Shawe-Taylor. 2010, «A PAC-Bayes bound for tailored density estimation», dans *ALT*, p. 148–162.

Huang, F. et A. Yates. 2012, «Biased representation learning for domain adaptation», dans *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, p. 1313–1323.

Hussain, Z., S. Szedmak et J. Shawe-Taylor. 2004, «The linear programming set covering machine», URL http://eprints.pascal-network.org/archive/00001210/.

Joachims, T. 1999, «Transductive inference for text classification using support vector machines», dans *ICML*, p. 200–209.

Kifer, D., S. Ben-David et J. Gehrke. 2004, «Detecting change in data streams», dans *Very Large Data Bases*, p. 180–191.

Koltchinskii, V. et D. Panchenko. 2000, «Rademacher processes and bounding the risk of function learning», dans *High Dimensional Probability II*, Springer, p. 443–457.

Lacasse, A. 2010, *Bornes PAC-Bayes et algorithmes d'apprentissage*, thèse de doctorat, Université Laval. URL http://www.theses.ulaval.ca/2010/27635/.

Lacasse, A., F. Laviolette, M. Marchand, P. Germain et N. Usunier. 2006, «PAC-Bayes bounds for the risk of the majority vote and the variance of the Gibbs classifier», dans *NIPS*, p. 769–776.

Lacoste, A., F. Laviolette et M. Marchand. 2012, «Bayesian comparison of machine learning algorithms on single and multiple datasets», dans *AISTATS*, p. 665–675.

Langford, J. 2005, «Tutorial on practical prediction theory for classification», *Journal of Machine Learning Research*, vol. 6, p. 273–306.

Langford, J. et M. Seeger. 2001, «Bounds for averaging classifiers», cahier de recherche, Carnegie Mellon, Departement of Computer Science.

Langford, J. et J. Shawe-Taylor. 2002, «PAC-Bayes & margins», dans *NIPS*, p. 423–430.

Laviolette, F. et M. Marchand. 2005, «PAC-Bayes risk bounds for sample-compressed Gibbs classifiers», dans *ICML*, p. 481–488.

Laviolette, F. et M. Marchand. 2007, «PAC-Bayes risk bounds for stochastic averages and majority votes of sample-compressed classifiers», *Journal of Machine Learning Research*, vol. 8, p. 1461–1487.

Laviolette, F., M. Marchand et M. Shah. 2005, «A PAC-Bayes approach to the set covering machine», dans *NIPS*, p. 731–738.

Lever, G., F. Laviolette et J. Shawe-Taylor. 2013, «Tighter pac-bayes bounds through distribution-dependent priors», *Theor. Comput. Sci.*, vol. 473, p. 4–28.

Li, X. et J. Bilmes. 2007, «A Bayesian divergence prior for classifier adaptation», dans *AISTATS*, p. 275–282.

Li, Y., K. Swersky et R. Zemel. 2014, «Unsupervised domain adaptation by domain invariant projection», dans *NIPS 2014 Workshop on Transfer and Multitask Learning*.

Liu, A. et B. D. Ziebart. 2014, «Robust classification under sample selection bias», dans *NIPS*, p. 37–45.

London, B., B. Huang, B. Taskar et L. Getoor. 2014, «PAC-Bayesian collective stability», dans *AISTATS*, p. 585–594.

Manquinho, V. et J. Marques-Silva. 2006, «On using cutting planes in pseudo-boolean optimization», *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 2, p. 209–219.

Mansour, Y., M. Mohri et A. Rostamizadeh. 2009a, «Domain adaptation : Learning bounds and algorithms», dans *COLT*.

Mansour, Y., M. Mohri et A. Rostamizadeh. 2009b, «Multiple source adaptation and the rényi divergence», dans *UAI*, p. 367–374.

Marchand, M. et J. Shawe-Taylor. 2002, «The set covering machine», *Journal of Machine Learning Research*, vol. 3, p. 723–746.

Marchand, M. et M. Sokolova. 2005, «Learning with decision lists of data-dependent features», *Journal of Machine Learning Research*, vol. 6, p. 427–451.

Martins, R., V. Manquinho et I. Lynce. 2011, «Parallel search for boolean optimization», dans *RCRA International Workshop on Experimental Evaluation of Algorithms for solving problems with combinatorial explosion*, vol. 11, p. 26–59.

Maurer, A. 2004, «A note on the PAC-Bayesian theorem», *CoRR*, vol. cs.LG/0411099.

McAllester, D. 1999, «Some PAC-Bayesian theorems», *Machine Learning*, vol. 37, n° 3, p. 355–363.

McAllester, D. 2003a, «PAC-Bayesian stochastic model selection», *Machine Learning*, vol. 51, n° 1, p. 5–21.

McAllester, D. 2003b, «Simplified PAC-Bayesian margin bounds», dans *COLT*, p. 203–215.

McAllester, D. 2007, «Generalization bounds and consistency for structured labeling», dans *Predicting Structured Data*, édité par G. Bakır, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar et S. V. N. Vishwanathan, chap. 11, MIT Press, Cambridge, MA, p. 247–261.

McAllester, D. 2013, «A PAC-Bayesian tutorial with a dropout bound», *CoRR*, vol. abs/1307.2118.

Mendenhall, W. 1983, «Nonparametric statistics», *Introduction to Probability and Statistics*, vol. 604.

Mohri, M. et A. M. Medina. 2012, «New analysis and algorithm for learning with drifting distributions», dans *ALT*, p. 124–138.

Mohri, M., A. Rostamizadeh et A. Talwalkar. 2012, *Foundations of Machine Learning*, The MIT Press.

Nesterov, Y. 2012, «Efficiency of coordinate descent methods on huge-scale optimization problems», *SIAM Journal on Optimization*, vol. 22, n° 2, p. 341–362.

Nesterov, Y. 2014, «Subgradient methods for huge-scale optimization problems», *Math. Program.*, vol. 146, n° 1-2, p. 275–297.

Parrado-Hernández, E., A. Ambroladze, J. Shawe-Taylor et S. Sun. 2012, «PAC-bayes bounds with data dependent priors», *Journal of Machine Learning Research*, vol. 13, p. 3507–3531.

Platt, J. 1999, *Fast Training of Support Vector Machines Using Sequential Minimal Optimization*, chap. 12, MIT Press, p. 185–208. C. J. C. Burges, and A. Smola editors, Advances in Kernel Methods : Support Vector Learning.

Ross, S. M. 1994, *Initiation aux probabilités, Traduction de la quatrième édition américaine*, Presses polytechniques et universitaires romandes, ISBN 0125980620.

Roy, J., F. Laviolette et M. Marchand. 2011, «From PAC-Bayes bounds to quadratic programs for majority votes», dans *ICML*, p. 649–656.

Schapire, R. 2003, «The boosting approach to machine learning: An overview», dans *Nonlinear Estimation and Classification*, *Lecture Notes in Statistics*, vol. 171, Springer New York, p. 149–171.

Schapire, R. E. et Y. Singer. 1999, «Improved boosting using confidence-rated predictions», *Machine Learning*, vol. 37, n° 3, p. 297–336.

Schölkopf, B., R. Herbrich et A. J. Smola. 2001, «A generalized representer theorem», dans *COLT/EuroCOLT*, p. 416–426.

Seeger, M. 2002, «PAC-Bayesian generalization bounds for gaussian processes», *Journal of Machine Learning Research*, vol. 3, p. 233–269.

Seldin, Y., P. Auer, F. Laviolette, J. Shawe-Taylor et R. Ortner. 2011, «PAC-Bayesian analysis of contextual bandits», dans *NIPS*, p. 1683–1691.

Seldin, Y., F. Laviolette, N. Cesa-Bianchi, J. Shawe-Taylor et P. Auer. 2012, «PAC-Bayesian inequalities for martingales», dans *UAI*, p. 12.

Seldin, Y. et N. Tishby. 2009, «PAC-Bayesian generalization bound for density estimation with application to co-clustering», dans *AISTATS*, p. 472–479.

Seldin, Y. et N. Tishby. 2010, «PAC-Bayesian analysis of co-clustering and beyond», *Journal of Machine Learning Research*, vol. 11, p. 3595–3646.

Shalev-Shwartz, S. et S. Ben-David. 2014, *Understanding Machine Learning : From Theory to Algorithms*, Cambridge University Press, New York, NY, USA.

Shanian, S. 2012, *Sample Compressed PAC-Bayesian Bounds and Learning Algorithms*, thèse de doctorat, Université Laval. URL `http://www.theses.ulaval.ca/2012/29037/`.

Shawe-Taylor, J. et N. Cristianini. 2004, *Kernel Methods for Pattern Analysis*, Cambridge University Press.

Tolstikhin, I. O. et Y. Seldin. 2013, «PAC-Bayes-empirical-bernstein inequality», dans *NIPS*, p. 109–117.

Topsøe, F. 2000, «Some inequalities for information divergence and related measures of discrimination», *IEEE Transactions on Information Theory*, vol. 46, n° 4, p. 1602–1609.

Valiant, L. G. 1984, «A theory of the learnable», *Commununications of the ACM*, vol. 27, n° 11, p. 1134–1142.

Vapnik, V. 1998, *Statistical learning theory*, Wiley, I-XXIV, 1-736 p..

Vapnik, V. N. et A. Y. Chervonenkis. 1971, «On the uniform convergence of relative frequencies of events to their probabilities», *Theory of Probability & Its Applications*, vol. 16, n° 2, p. 264–280.

Vincent, P., H. Larochelle, Y. Bengio et P. Manzagol. 2008, «Extracting and composing robust features with denoising autoencoders», dans *ICML*, p. 1096–1103.

Younsi, M. 2012, «Proof of a combinatorial conjecture coming from the PAC-Bayesian machine learning theory», *ArXiv e-prints*. URL http://arxiv.org/abs/1209.0824v1.

Zemel, R. S., Y. Wu, K. Swersky, T. Pitassi et C. Dwork. 2013, «Learning fair representations», dans *ICML*, p. 325–333.

Zhang, C., L. Zhang et J. Ye. 2012, «Generalization bounds for domain adaptation», dans *NIPS*, p. 3320–3328.

Zhong, E., W. Fan, Q. Yang, O. Verscheure et J. Ren. 2010, «Cross validation framework to choose amongst models and datasets for transfer learning», dans *ECML/PKDD (3)*, p. 547–562.