Apprentissage automatique avec garanties de généralisation à l'aide de méthodes d'ensemble maximisant le désaccord

Thèse

Jean-Francis Roy

Doctorat en informatique
Philosophiæ doctor (Ph. D.)

Québec, Canada

Apprentissage automatique avec garanties de généralisation à l'aide de méthodes d'ensemble maximisant le désaccord

Thèse

Jean-Francis Roy

Sous la direction de :

François Laviolette, directeur de recherche
Mario Marchand, codirecteur de recherche

Résumé

Nous nous intéressons au domaine de l'apprentissage automatique, une branche de l'intelligence artificielle. Pour résoudre une tâche de classification, un algorithme d'apprentissage observe des données étiquetées et a comme objectif d'apprendre une fonction qui sera en mesure de classifier automatiquement les données qui lui seront présentées dans le futur. Plusieurs algorithmes classiques d'apprentissage cherchent à combiner des classificateurs simples en construisant avec ceux-ci un classificateur par vote de majorité.

Dans cette thèse, nous explorons l'utilisation d'une borne sur le risque du classificateur par vote de majorité, nommée la C-borne. Celle-ci est définie en fonction de deux quantités : la performance individuelle des votants, et la corrélation de leurs erreurs (leur désaccord). Nous explorons d'une part son utilisation dans des bornes de généralisation des classificateurs par vote de majorité. D'autre part, nous l'étendons de la classification binaire vers un cadre généralisé de votes de majorité. Nous nous en inspirons finalement pour développer de nouveaux algorithmes d'apprentissage automatique, qui offrent des performances comparables aux algorithmes de l'état de l'art, en retournant des votes de majorité qui maximisent le désaccord entre les votants, tout en contrôlant la performance individuelle de ceux-ci.

Les garanties de généralisation que nous développons dans cette thèse sont de la famille des bornes PAC-bayésiennes. Nous généralisons celles-ci en introduisant une borne générale, à partir de laquelle peuvent être retrouvées les bornes de la littérature. De cette même borne générale, nous introduisons des bornes de généralisation basées sur la C-borne. Nous simplifions également le processus de preuve des théorèmes PAC-bayésiens, nous permettant d'obtenir deux nouvelles familles de bornes. L'une est basée sur une différente notion de complexité, la divergence de Rényi plutôt que la divergence Kullback-Leibler classique, et l'autre est spécialisée au cadre de l'apprentissage transductif plutôt que l'apprentissage inductif.

Les deux algorithmes d'apprentissage que nous introduisons, MinCq et CqBoost, retournent un classificateur par vote de majorité maximisant le désaccord des votants. Un hyperparamètre permet de directement contrôler leur performance individuelle. Ces deux algorithmes étant construits pour minimiser une borne PAC-bayésienne, ils sont rigoureusement justifiés théoriquement. À l'aide d'une évaluation empirique, nous montrons que MinCq et CqBoost ont une performance comparable aux algorithmes classiques de l'état de l'art.

Abstract

We focus on machine learning, a branch of artificial intelligence. When solving a classification problem, a learning algorithm is provided labelled data and has the task of learning a function that will be able to automatically classify future, unseen data. Many classical learning algorithms are designed to combine simple classifiers by building a weighted majority vote classifier out of them.

In this thesis, we extend the usage of the C-bound, a bound on the risk of the majority vote classifier. This bound is defined using two quantities : the individual performance of the voters, and the correlation of their errors (their disagreement). First, we design majority vote generalization bounds based on the C-bound. Then, we extend this bound from binary classification to generalized majority votes. Finally, we develop new learning algorithms with state-of-the-art performance, by constructing majority votes that maximize the voters' disagreement, while controlling their individual performance.

The generalization guarantees that we develop in this thesis are in the family of PAC-Bayesian bounds. We generalize the PAC-Bayesian theory by introducing a general theorem, from which the classical bounds from the literature can be recovered. Using this same theorem, we introduce generalization bounds based on the C-bound. We also simplify the proof process of PAC-Bayesian theorems, easing the development of new families of bounds. We introduce two new families of PAC-Bayesian bounds. One is based on a different notion of complexity than usual bounds, the Rényi divergence, instead of the classical Kullback-Leibler divergence. The second family is specialized to transductive learning, instead of inductive learning.

The two learning algorithms that we introduce, MinCq and CqBoost, output a majority vote classifier that maximizes the disagreement between voters. An hyperparameter of the algorithms gives a direct control over the individual performance of the voters. These two algorithms being designed to minimize PAC-Bayesian generalization bounds on the risk of the majority vote classifier, they come with rigorous theoretical guarantees. By performing an empirical evaluation, we show that MinCq and CqBoost perform as well as classical state-of-the-art algorithms.

Table des matières

Liste des tableaux

Table des figures

Liste des algorithmes

Notation

a	Un scalaire (entier ou réel).
$\mathbf{a}$	Un vecteur (colonne).
$\mathbf{A}$	Une matrice.
$\mathbf{a}^\top$, $\mathbf{A}^\top$	La transposée d'un vecteur ou d'une matrice.
$\mathbf{A}^{-1}$	L'inverse d'une matrice.
$\mathbf{A}^\dagger$	La pseudo-inverse d'une matrice.
$\mathbf{0}_n$, $\mathbf{1}_n$	Un vecteur (colonne) de n zéros, un vecteur de n uns.
$\mathbf{0}_{m\times n}$, $\mathbf{1}_{m\times n}$	Une matrice de $m \times n$ zéros, une matrice de de $m \times n$ uns.
$\mathbf{I}_n$	La matrice identité, avec une diagonale de n éléments.
$\mathbf{A}_{i:}$	La i-ième ligne de la matrice $\mathbf{A}$.
$\mathbf{A}_{:j}$	La j-ième colonne de la matrice $\mathbf{A}$.
a_i	Le i-ième élément du vecteur a.
A_{ij}	L'élément en position (i, j) de la matrice $\mathbf{A}$.
$\preceq$, $\succeq$	Les opérateurs $\leq$ et $\geq$, appliqués élément par élément à deux vecteurs ou matrices.
$\lvert\mathbf{a}\rvert$	La cardinalité du vecteur $\mathbf{a}$.
$\lVert\mathbf{a}\rVert$	La norme Euclidienne du vecteur $\mathbf{a}$.
$\mathbf{f}(x)$	Une fonction retournant un vecteur.
$\{a_1, \ldots, a_n\}$	Un ensemble.
$\langle a_1, \ldots, a_n \rangle$	Une séquence.

Remerciements

Comme mieux vaut tard que jamais, me voici en train d'écrire les dernières lignes de ma thèse (ou les premières, dépendamment dans quel sens vous la lisez). À chaque fois que je donne une estimation du temps dont j'aurai besoin pour compléter quelque chose, ma blonde multiplie par 3. Dans le cas de cette thèse, je dois avouer que je n'ai pas calculé le facteur multiplicatif de mon « j'en avais environ pour 3 semaines mais le bébé est né un peu en avance donc ça m'a pris 10 mois ». Laissez-moi commencer par remercier mon garçon Olivier, de m'avoir laissé rédiger certains chapitres de cette thèse juste après sa naissance pendant qu'il dormait dans mes bras (dans un gros coussin, je n'ai quand même pas déposé mon laptop sur lui), et laissé terminer la rédaction pendant qu'il jouait avec son « tchou tchou ».

Je tiens à remercier François Laviolette mon directeur de recherche, pour tout ce qu'il m'a appris durant ma maîtrise et mon doctorat, que ce soit en recherche ou en enseignement. Notamment, il m'a montré la beauté que pouvaient avoir l'écriture scientifique et les démonstrations mathématiques. François est un grand pédagogue, a énormément d'intuition, et transmet ses idées plus rapidement que la vitesse à laquelle ses étudiants sont en mesure de les programmer. François dit souvent que son langage de programmation préféré est l'« étudiant gradué ». Je ne sais pas si ce langage est Turing-complet.

Je tiens également à remercier Mario Marchand, mon codirecteur de recherche, pour son focus et sa rigueur. On dit souvent de Mario qu'il ne « compile pas » lorsqu'il arrive à un point d'un article où quelque chose est mal défini (avouons que c'est vraiment inacceptable). Merci à François et Mario, pour leur confiance et leur soutien. En plus de me soutenir dans mes activités de recherche, ils m'ont laissé multiplier les expériences comme auxiliaire d'enseignement, comme enseignant, et pendant mes projets avec des entreprises. En d'autres mots, ils m'ont laissé apprendre ma distribution qui « maximisait le désaccord » !

Merci aux examinateurs de ma thèse, Philippe Giguère, Simon Lacoste-Julien et Pascal Tesson. Leurs questions, commentaires et suggestions m'ont permis d'améliorer la qualité du manuscrit.

Merci à mes collègues et amis du laboratoire GRAAL, avec qui j'ai partagé plusieurs années de ma vie, à travers les hauts et les bas de la recherche et les escapades en conférence. Je ne veux pas énumérer tout le monde de peur d'en oublier, j'ai quand même été présent pendant quelques générations : j'ai appris avec les anciens, et montré aux nouveaux. Mention spéciale à ces anciens (aujourd'hui de vieux sages), Pascal Germain, Sébastien Giguère, Alexandre Lacasse, Alexandre Lacoste et Francis Turgeon-Boutin, avec qui j'ai eu la chance de partager des idées, débattre sur divers sujets, ou partager un verre de scotch suite à la soumission ou l'acceptation d'un article. Mention encore plus spéciale à Pascal, avec qui j'ai partagé le plus d'aventures académiques en rédigeant le «*never-ending paper*» tout en buvant une IPA au *Laundromat Café* en Islande. Je n'ai pas pu m'empêcher de le copier en rédigeant moi aussi une thèse d'exactement 2^8 pages. Je tiens également à mentionner les affinités que j'ai eues à travailler avec Luc Bégin, qui a été la source de plusieurs idées derrière le dernier chapitre de cette thèse.

Je remercie mes amis proches, qui m'ont souvent permis de rester sain d'esprit, en me rappelant que mes blagues de minimums locaux, de probabilités non nulles et de distributions de probabilité n'étaient pas nécessairement drôles à l'extérieur du monde académique.

Bien évidemment, merci à mes parents Denis et Martine et à ma soeur Isabelle, qui m'ont toujours encouragé dans tout ce que j'ai entrepris, que ce soit une forteresse en légos, quand j'ai démonté (et brisé) mon ordinateur, mon départ vers la « grand ville » ou ce doctorat. Je n'aurais pas pu me rendre aussi loin sans eux. Merci à ma marraine Michelyne de m'avoir accueilli chez elle pendant mes premières années à Québec et de m'avoir encouragé à poursuivre mes études.

Et finalement, la dernière mais non la moindre, merci à ma conjointe Marie-Caroline, pour tout. Marie, je trouve difficilement les mots pour t'exprimer ma gratitude. Merci pour nos 10 belles années ensemble jusqu'à maintenant. Merci de penser toujours à tout, d'avoir des solutions à tous les problèmes. Merci d'écouter mes histoires interminables, que tu entends à plusieurs reprises car je les raconte à nouveau à chaque fois qu'on croise quelqu'un qui ne l'a pas entendue. Merci pour toutes les fois où tu as accepté de me laisser me casser la tête sur quelque chose jusqu'aux petites heures du matin, malgré que j'estimais à 10 minutes le temps que je voulais y passer. Merci pour nos discussions toujours intéressantes, et pour les moments où on met nos cerveaux à «*off*» en se tappant une saison complète d'une série. Merci d'être là pour moi et pour notre garçon, dans les beaux moments, mais aussi dans les moments plus difficiles. Merci pour ton infinie compréhension, pour ta générosité, pour ton sourire... Merci.

Chapitre 1

Introduction

L'*intelligence artificielle* est une discipline de plus en plus présente dans nos vies, souvent de manière indirecte dans des technologies qui en font un usage discret. Cependant, son utilisation et les applications qui en découlent sont de plus en plus médiatisées, de sorte que la population en général est maintenant mieux informée de son existence et de ses possibilités.

Malgré ce qu'on peut lire dans les médias, qui à notre avis ont souvent tendance à trop simplifier certains concepts complexes, l'intelligence artificielle n'est pas qu'une tentative de doter les machines de la pensée, en imitant la manière dont le cerveau humain fonctionne. Cette discipline est entre autres un mélange d'algorithmique, d'optimisation et de statistique, ayant pour but de résoudre des problèmes complexes, ou du moins qui sont complexes du point de vue d'un ordinateur. Ces problèmes sont fréquemment résolus très efficacement (ou naturellement) par les humains, mais sont très difficiles à résoudre pour une machine, qui elle n'est pas dotée de l'intelligence humaine.

Des exemples communs sont la reconnaissance de la parole, la reconnaissance de caractères manuscrits, la catégorisation de texte ou la conduite d'une automobile. Les êtres humains sont très efficaces à résoudre ces tâches. La résolution de celles-ci demande d'abord une période d'apprentissage, pendant laquelle la tâche est difficile. Cependant, avec le temps, leur résolution devient pratiquement instantanée. Par exemple, lorsque nous apprenons à conduire, la quantité d'information à analyser en « temps réel » est très grande : nous devons utiliser les contrôles disponibles dans la voiture pour la diriger et se rendre à destination, tout en portant attention à tout ce qui se passe autour de nous, comme la route, la signalisation routière, les autres véhicules environnants, les cyclistes et les piétons. L'apprentissage prend un certain temps et demande de la pratique régulière. Cependant, une fois la tâche bien apprise, nous la réalisons de manière quasi-automatique, sans y penser. Il n'est pas rare pour l'auteur de cette thèse, lors d'une distraction comme une discussion avec un passager, d'oublier temporairement la destination et d'être sur le « pilote automatique », c'est-à-dire de se rendre à sa destination la plus courante à l'époque : l'Université Laval.

L'intelligence artificielle est une discipline au sens très large : elle englobe toute tentative de rendre un ordinateur capable de résoudre un problème historiquement réservé à l'intelligence humaine. Pendant longtemps, les algorithmes d'intelligence artificielle n'étaient qu'une combinaison d'expressions conditionnelles « si ... alors ... », formant un *arbre de décision* imitant le raisonnement humain. Ces algorithmes étaient construits en consultant des experts d'un certain domaine, afin d'*encoder* leurs connaissances. L'intelligence artificielle englobe aujourd'hui davantage de disciplines, dont la plus en vogue est sans conteste l'*apprentissage automatique* («*machine learning*» en anglais, aussi connue sous le nom d'*apprentissage artificiel* ou d'*apprentissage machine*).

1.1 L'apprentissage automatique

L'apprentissage automatique est une science qui consiste à développer des *algorithmes d'apprentissage*, qui apprennent à résoudre une tâche. Dans le cas de l'*apprentissage supervisé*, le sujet de cette thèse, les modèles appris par ces algorithmes tentent de prédire automatiquement la solution à un problème, en apprenant à partir d'exemples solutionnés[1]. Nous dénotons $\mathcal{X}$ l'espace d'entrée, c'est-à-dire l'ensemble contenant les descriptions du problème à résoudre, et $\mathcal{Y}$ l'espace de sortie, c'est-à-dire l'ensemble contenant toutes les solutions possibles. Un exemple solutionné (ou étiqueté) est une paire $(x, y) \in \mathcal{X} \times \mathcal{Y}$, et l'algorithme d'apprentissage est à la recherche d'une fonction $h : \mathcal{X} \to \mathcal{Y}$ en mesure de bien étiqueter les nouveaux exemples observés. Typiquement, la description des exemples sera représentée par un vecteur de valeurs réelles. On aura donc $\mathcal{X} = \mathbb{R}^d$ pour une certaine dimensionnalité d. En fonction de la nature de l'espace de sortie $\mathcal{Y}$, nous retrouvons plusieurs catégories d'apprentissage.

1.1.1 La classification

La *classification* consiste à catégoriser des exemples parmi un ensemble de classes possibles. Le cadre de classification le plus étudié est la *classification binaire*, où un algorithme doit apprendre à distinguer parmi deux différentes classes, couramment appelées les classes *positive* et *négative*. On a donc $\mathcal{Y} = \{-1, 1\}$. Certains types de problèmes n'ont qu'une seule classe, qui peut être considérée comme la classe « normale ». On parle alors de *détection d'anomalie*, où l'on tente de détecter les exemples qui sortent de l'ordinaire. La classification *multi-classe* consiste simplement à la situation où chaque exemple appartient à une catégorie parmi un ensemble de $k > 2$ catégories. Dans cette situation, nous avons $\mathcal{Y} = \{1, 2, \dots, k\}$. Finalement, la classification *multi-étiquette* correspond à la situation où chaque exemple peut appartenir à zéro, une ou plusieurs catégories à la fois, parmi un ensemble de k catégories possibles. L'espace de sortie $\mathcal{Y}$ est donc un vecteur de k éléments $\{0, 1\}^k$, où chaque position correspond à une catégorie, et où on observe un 1 si l'exemple appartient à cette catégorie et 0 autrement.

1. Il existe d'autres familles d'algorithmes d'apprentissage automatique, comme l'*apprentissage non-supervisé* et l'*apprentissage par renforcement*.

Un exemple commun de ce type de problèmes est la classification de texte. En fonction d'un texte en entrée[2], on pourrait s'intéresser à la catégoriser de différentes manières. Par exemple, l'*analyse du sentiment* (ou *classification de polarité*) tente de déterminer si l'auteur d'un texte exprime un sentiment positif ou négatif (classification binaire). Un autre exemple classique est la *catégorisation de texte*, où le sujet de chaque document doit être déterminé parmi k sujets (classification multi-classe), ou bien chaque texte peut être étiqueté avec zéro, un ou plusieurs sujets à la fois (classification multi-étiquette). On peut également imaginer des situations où l'on collecte un ensemble de textes « normaux » et on tente de détecter ceux qui sortiront de l'ordinaire par la suite, par exemple pour lever automatiquement une alerte lorsqu'un courriel douteux est reçu (détection d'anomalie).

1.1.2 La régression et la prédiction structurée

Lorsque l'espace de sortie $\mathcal{Y}$ est une valeur réelle, on parle alors d'un problème de *régression*. On pourrait par exemple s'intéresser à prédire la pertinence d'un document en fonction de son contenu et d'une requête faire par un utilisateur sur le Web, tenter de prédire la valeur boursière des actions d'une compagnie en fonction de son historique et de textes de nouvelles dans les médias, ou prédire l'affinité de liaison entre une molécule et une protéine dans le cadre de la découverte de nouveaux médicaments.

Il existe également d'autres situations encore plus générales que la régression, où la valeur à prédire est arbitrairement complexe. On parle alors de *prédiction structurée*. Les structures à prédire peuvent prendre entre autres la forme d'un arbre, d'un graphe ou d'une séquence. Un exemple commun est la reconnaissance de la parole, où $\mathcal{Y}$ correspond à l'ensemble de toutes les séquences de mots possibles formant des phrases.

1.1.3 La généralisation

Un algorithme d'apprentissage est entraîné à l'aide de *données étiquetées*, c'est-à-dire de couples $(x, y) \in \mathcal{X} \times \mathcal{Y}$. L'algorithme retourne ensuite un *prédicteur* $h : \mathcal{X} \to \mathcal{Y}$, dont la tâche est de prédire de *nouveaux* exemples dont nous ne connaissons pas l'étiquette. Dans le cadre de la classification, ce prédicteur est nommé un classificateur.

Plus formellement, dans le cadre fréquentiste utilisé dans cette thèse, nous supposons qu'il existe une *distribution de probabilité* D sur $\mathcal{X} \times \mathcal{Y}$ qui *génère* les exemples que nous observons. Cette distribution D est inconnue, mais fixe. Nous supposons également que les exemples que nous observons sont indépendamment et identiquement distribués selon D, ce que nous dénotons par $(x, y) \sim D$, c'est-à-dire que la probabilité d'observer un exemple ne varie pas en fonction des exemples observés précédemment. Nous dénotons finalement par S l'ensemble

2. De nombreuses techniques existent pour transformer un texte en un vecteur de valeurs réelles, notamment *TF-IDF* (Salton et McGill, 1986 ; Dumais et al., 1998), *Word2Vec* (Mikolov et al., 2013) et *Doc2Vec* (Le et Mikolov, 2014).

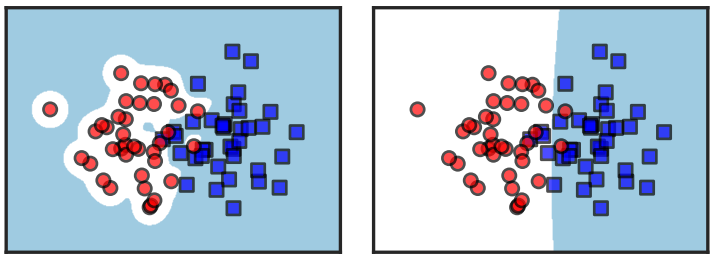

FIG. 1.1: Représentation graphique du principe de généralisation. Chaque exemple est représenté par un point dans un espace à deux dimensions. Un carré bleu correspond à un exemple positif, et un cercle rouge correspond à un exemple négatif. Le classificateur prédit qu'un exemple est positif si celui-ci se retrouve dans une zone bleutée, et prédit que l'exemple est négatif autrement. Le classificateur de gauche est plus *complexe* et ne fait *aucune erreur* sur les données observées jusqu'à présent. Le classificateur de droite est plus *simple*, mais fait *quelques erreurs* empiriques. Celui-ci devrait par contre mieux *généraliser* aux exemples qui seront observés dans le futur.

de données[3] de m exemples à la disposition de l'algorithme d'apprentissage. Nous avons donc $S = \langle (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \rangle \sim D^m$.

Comme un prédicteur appris par un algorithme d'apprentissage sera ultimement utilisé sur de nouvelles données qui ne font pas partie de l'ensemble d'entraînement S, celui-ci doit être évalué par rapport à sa performance sur la distribution inconnue D. C'est ce que nous appelons la *généralisation* : l'algorithme d'apprentissage doit être en mesure de *généraliser* sur les données qui n'ont pas encore été observées. La figure 1.1 montre un exemple d'ensemble de données, sur lequel deux classificateurs binaires ont été appris. À gauche, la fonction de classification ne fait aucune erreur mais est très complexe. À droite, le classificateur fait quelques erreurs, mais est plus simple et semble mieux généraliser aux données qui seront observées à l'avenir. On retrouvera généralement ce type de compromis pour un prédicteur : *erreur empirique* versus *complexité*. On dit d'un prédicteur qui s'est trop concentré sur un ensemble d'entraînement S et qui n'est pas en mesure de généraliser aux nouvelles données qu'il a fait du *surapprentissage* (ou de l'«*overfitting*»). La figure 1.2 résume finalement les deux phases d'un algorithme d'apprentissage : l'apprentissage et la prédiction.

Dans la plupart des chapitres de cette thèse, nous nous intéressons à la classification binaire. Cette tâche, bien que la plus simple parmi celles présentées plus haut, cache une multitude de défis intéressants à résoudre. Lorsque nous choisissons comme espace de sortie $\mathcal{Y} = \{-1, 1\}$,

3. Il serait plus rigoureux de parler d'une *séquence* de données plutôt que d'un ensemble, puisque les exemples peuvent être répétés et qu'ils sont généralement numérotés pour simplifier la notation. Cependant, il est courant dans la littérature d'utiliser tout de même le terme *ensemble*. Nous faisons cet abus de langage dans cette thèse, mais utilisons des crochets $\langle\ \rangle$ plutôt que des accolades $\{\ \}$ pour représenter la séquence.

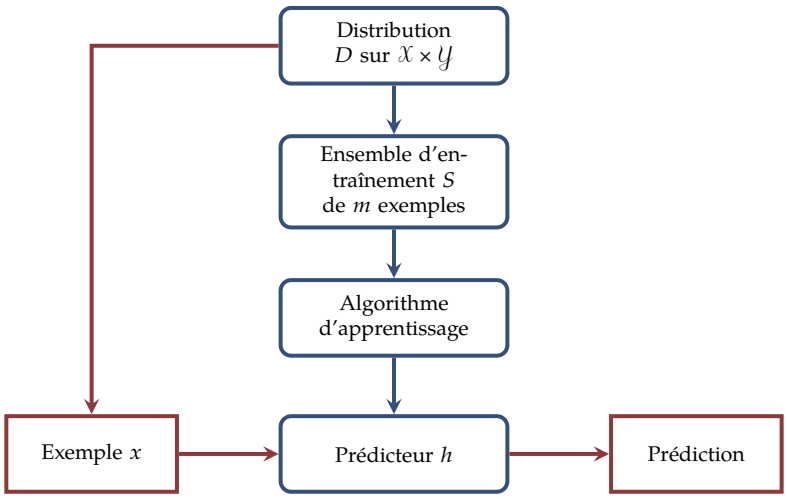

FIG. 1.2: Représentation des deux tâches d'un algorithme d'apprentissage. Le chemin déterminé par les flèches bleues représente le processus d'entraînement, et le chemin déterminé par les flèches rouges représente la tâche à résoudre une fois l'apprentissage terminé.

les équations et la notation nécessaires pour définir plusieurs concepts de base se voient grandement simplifiés. Pour cette raison, nous choisissons de poursuivre cette introduction en ne considérant que la classification binaire.

1.1.4 Le risque

En classification, la notion de « performance » d'un classificateur h se définit par un faible *risque*, c'est-à-dire la probabilité que celui-ci fasse une erreur sur un exemple. Le «*vrai*» *risque* d'un classificateur h est défini par

$$R_D(h) \triangleq \Pr_{(x,y)\sim D}(h(x) \neq y) = \mathop{\mathbf{E}}_{(x,y)\sim D} I(h(x) \neq y)\,,$$

où $I(a) = 1$ si le prédicat a est vrai, et 0 autrement. Le but ultime d'un algorithme d'apprentissage est de produire un classificateur h de faible risque. Cependant, le risque évalué sur la distribution D ne peut pas être calculé, comme cette dernière est inconnue. On ne peut calculer que la performance empirique, c'est-à-dire le *risque empirique*, défini par

$$R_S(h) \triangleq \frac{1}{m}\sum_{k=1}^{m} I\left(h(x_k) \neq y_k\right)\,.$$

Dans la prochaine section, nous présentons quelques familles de classificateurs parmi lesquelles un algorithme d'apprentissage peut choisir. Nous présentons également les *votes de majorité*, une famille de classificateurs combinant plusieurs classificateurs à l'aide d'une distribution de probabilité.

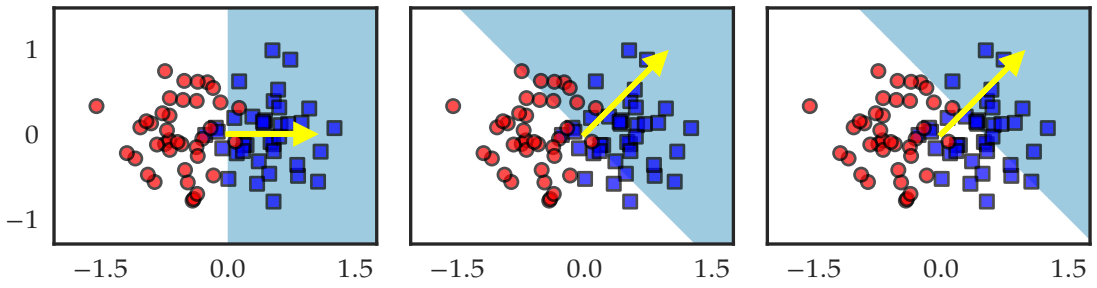

FIG. 1.3: Exemples de classificateurs linéaires, où le vecteur $\mathbf{w}$ est représenté par une flèche jaune. À gauche, $\mathbf{w} = \begin{bmatrix} 1 & 0 \end{bmatrix}^\top$ et $b = 0$. Au centre, $\mathbf{w} = \begin{bmatrix} 1 & 1 \end{bmatrix}^\top$ et $b = 0$. Finalement, la figure de droite montre, à l'aide du même vecteur $\mathbf{w}$, l'effet du biais b qui a ici une valeur de $-\frac{1}{2}$.

1.2 Les votes de majorité

Dans cette section, nous supposons que les exemples x sont représentés par un vecteur $\mathbf{x}$ à d dimensions, c'est-à-dire que $\mathcal{X} = \mathbb{R}^d$. Les exemples d'un ensemble d'entraînement sont donc numérotés par $\langle \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m \rangle$, alors que la notation $x_1, x_2, \ldots, x_d$ correspond ici aux composantes d'un vecteur $\mathbf{x}$. Les algorithmes d'apprentissage en classification binaire ont la tâche de trouver un classificateur $h : \mathbb{R}^d \to \{-1, 1\}$ de faible risque $R_D(h)$. Il existe de nombreuses familles de classificateurs binaires, dont nous présentons quelques exemples.

1.2.1 Quelques familles de classificateurs

Les *classificateurs linéaires* permettent de séparer un espace $\mathbb{R}^d$ en deux parties, à l'aide d'un hyperplan séparateur. Un classificateur linéaire classifie un exemple $\mathbf{x}$ en utilisant une fonction $h(\mathbf{x}) = \text{sgn}\left[\mathbf{w}^\top \mathbf{x} + b\right]$, où $\text{sgn}[a]$ retourne 1 si $a > 0$ et -1 autrement, $\mathbf{w}$ est un vecteur de $\mathbb{R}^d$, $\mathbf{w}^\top \mathbf{x}$ est le produit scalaire entre les vecteurs $\mathbf{w}$ et $\mathbf{x}$, et b est un scalaire généralement appelé le *biais*. Le vecteur $\mathbf{w}$ définit la direction du séparateur linéaire, alors que le biais b permet de déplacer l'hyperplan par une translation dans la direction du vecteur $\mathbf{w}$. La figure 1.3 montre trois exemples de classificateurs linéaires.

Les *arbres de décision* (ou «*decision trees*») sont définis par un ensemble de règles de décision formant un arbre, où chaque noeud interne est une condition déterminant dans quel sous-arbre l'exemple se situe, et chaque feuille correspond à une décision de classification. Les règles de décision des noeuds internes sont souvent définies par une valeur de seuil sur un attribut de l'exemple $\mathbf{x}$. La figure 1.4 montre un exemple d'un tel arbre de décision.

Les *souches de décision* (ou «*decision stumps*») sont simplement des arbres de décision de profondeur 1, c'est-à-dire qu'elles ne possèdent que trois noeuds : un noeud interne et deux feuilles. Lorsqu'une souche de décision est définie à l'aide d'une valeur de seuil sur un seul

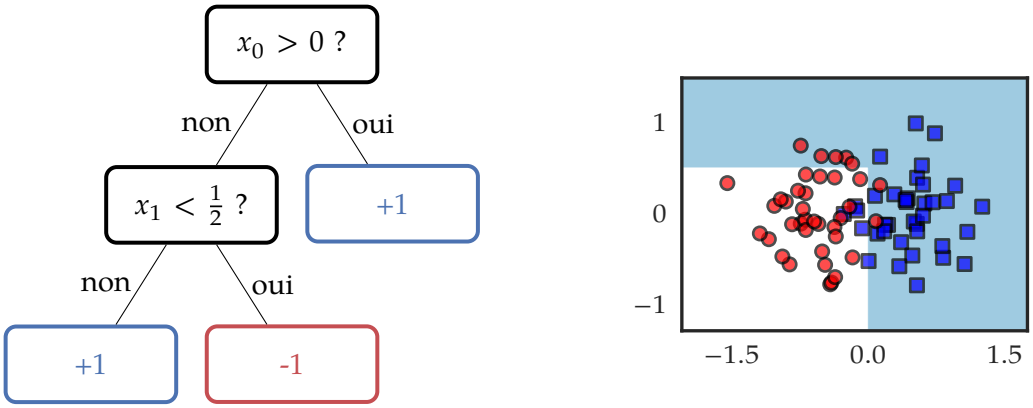

FIG. 1.4: Exemple d'un arbre de décision à deux noeuds internes et trois feuilles. La figure de gauche représente l'ensemble de règles de décision qui forment l'arbre, et la figure de droite représente la frontière de décision résultante sur un ensemble de données à deux dimensions.

attribut, elle permet d'exprimer un séparateur linéaire dont la direction correspond à l'une des composantes des vecteurs $\mathbf{x}$. La figure 1.5 en montre un exemple.

Nous présentons finalement une famille $\mathcal{F}$ de *votants à valeur réelle* (et non de classificateurs), qui au lieu de retourner directement -1 ou 1, retournent une valeur réelle. Il s'agit donc de régresseurs. Tout comme les classificateurs, les régresseurs pourront être utilisés pour fabriquer un classificateur plus complexe nommé le *vote de majorité*, qui sera défini plus loin. Nous considérons des fonctions de la forme $f_i(\mathbf{x}) = y_i k(\mathbf{x}_i, \mathbf{x})$, où $(\mathbf{x}_i, y_i)$ est un exemple de l'ensemble d'entraînement, et où $k(\mathbf{x}, \mathbf{x}')$ est une fonction *noyau*[4] calculant la *similarité* entre les exemples $\mathbf{x}$ et $\mathbf{x}'$. Plus les exemples $\mathbf{x}_i$ et $\mathbf{x}$ sont « similaires », plus la valeur de $k(\mathbf{x}_i, \mathbf{x})$ sera grande. Deux exemples classiques de noyaux sont le *noyau linéaire*, défini par $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$, et le *noyau « radial basis function » (RBF)*, défini par $k(\mathbf{x}, \mathbf{x}') = exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\right)$, où γ est un paramètre. Comme la valeur du noyau RBF est toujours positive, chaque votant $f_i(\mathbf{x}) = y_i k(\mathbf{x}_i, \mathbf{x})$ retourne une valeur de même signe que y_i. Un tel votant peut donc être vu comme une mesure de « confiance » qu'un exemple est de même signe que l'exemple $(\mathbf{x}_i, y_i)$. La figure 1.6 montre deux exemples de telles fonctions f_i, basées sur le noyau RBF, et définies sur deux exemples $(\mathbf{x}_i, y_i)$ différents.

1.2.2 Choix du meilleur classificateur

Les algorithmes d'apprentissage sont généralement définis de telle manière qu'ils choisissent le « meilleur » classificateur h parmi un ensemble $\mathcal{H}$ prédéfini de classificateurs. Par exemple, un algorithme pourrait choisir le meilleur arbre de décision, en minimisant un compromis entre le risque empirique $R_S(h)$ et une notion de complexité de l'arbre de décision.

4. Pour le besoin de cette introduction, nous n'entrons pas dans les détails de la définition d'un noyau. Un lecteur intéressé à ce vaste sujet peut se référer à CRISTIANINI et SHAWE-TAYLOR (2000b).

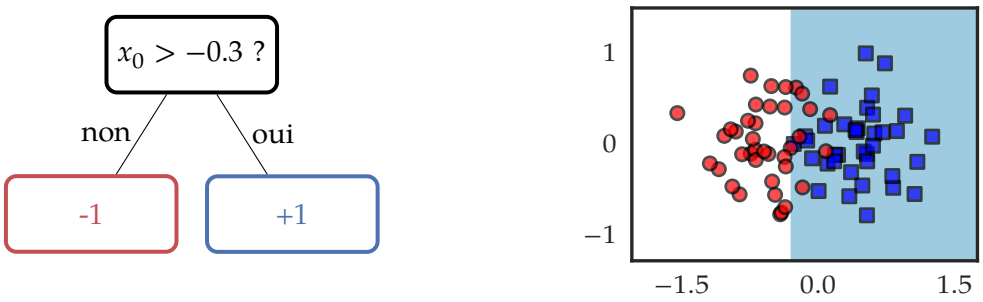

FIG. 1.5: Exemple d'une souche de décision. À gauche, nous présentons la règle de décision qui forme la souche, et à droite la frontière de décision correspondante.

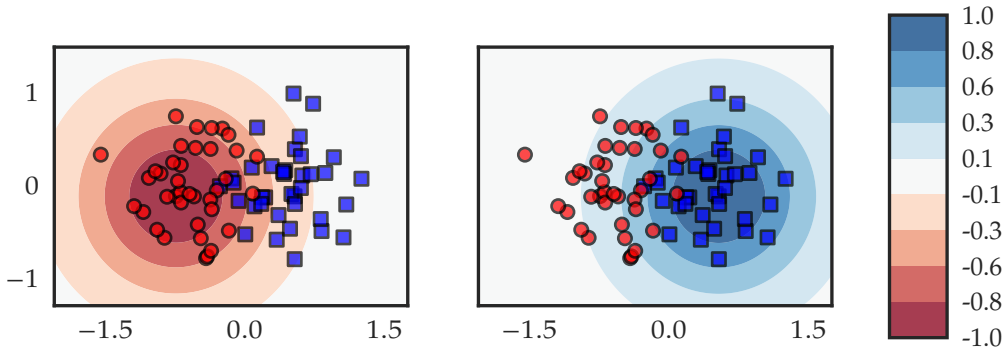

FIG. 1.6: Exemple de deux régresseurs $f_i(\mathbf{x}) = y_i\, k(\mathbf{x}_i, \mathbf{x})$ basés sur deux exemples $(\mathbf{x}_i, y_i)$ différents, où $k(\mathbf{x}, \mathbf{x}') = exp(-\|\mathbf{x} - \mathbf{x}'\|^2)$. Chaque fonction f_i classifie tous les exemples avec la même étiquette, avec une « confiance » qui décroît en fonction de la distance de $(\mathbf{x}_i, y_i)$.

Dans le cas d'une famille suffisamment variée $\mathcal{H}$ de classificateurs suffisamment performants, choisir un seul classificateur $h \in \mathcal{H}$ peut être une bonne stratégie d'apprentissage. Dans cette situation, le but d'un algorithme d'apprentissage est de trouver le classificateur h de plus faible « vrai » risque, c'est-à-dire $\text{argmin}_{h \in \mathcal{H}} R_D(h)$.

Cette stratégie n'est pas intéressante dans la situation où l'ensemble $\mathcal{H}$ est constitué de votants qui n'offrent pas une performance individuelle suffisante. Par exemple, les classificateurs linéaires et les souches de décision étant très simples, ceux-ci pourraient ne pas offrir une bonne performance sur des ensembles de données qui ne sont pas linéairement séparables. De manière similaire, s'ils ne sont considérés qu'individuellement, les votants à valeur réelle basés sur les noyaux RBF ne peuvent prédire qu'une seule classe. Cependant, nous verrons ci-bas qu'un ensemble de classificateurs simples peut être utilisé pour construire un classificateur ayant une bonne performance, en fabriquant un *vote de majorité*.

1.2.3 Combinaison de classificateurs

Les *votes de majorité*, (aussi appelés *votes de majorité pondérés* ou *classificateurs de Bayes*), sont une famille de classificateurs construits à l'aide d'un ensemble $\mathcal{H}$ de *votants* (qui sont des classificateurs ou des régresseurs), et d'une distribution de probabilité Q sur $\mathcal{H}$.

En classification binaire, un classificateur par vote de majorité B_Q classifie un exemple en calculant l'espérance sur tous les votants de $\mathcal{H}$, pondérée par la distribution Q, puis en prenant le signe du résultat :

$$B_Q(\mathbf{x}) \triangleq \operatorname{sgn}\left[\mathop{\mathbf{E}}_{h\sim Q} h(\mathbf{x})\right].$$

Lorsqu'une famille de classificateurs $\mathcal{H}$ (ou de régresseurs $\mathcal{F}$) contient des votants qui sont individuellement peu performants (généralement appelés des *votants faibles*) mais est suffisamment variée, il est possible de construire un vote de majorité qui lui, aura une bonne performance. Par exemple, la figure 1.7 présente six classificateurs, dont quatre souches de décision (h_1, h_2, h_4 et h_5) et deux classificateurs «*dummy*» (h_3 et h_6) qui ne font que classifier tous les exemples avec la même classe.

On remarque qu'aucun de ces classificateurs n'a une bonne performance sur l'ensemble de données : les souches de décision ont un risque de 33%, et les deux classificateurs «*dummy*» ont un risque de 50%. Cependant, tel que le montre la figure 1.8, un choix judicieux de distribution Q sur $\mathcal{H} = \{h_1, \ldots, h_6\}$ permet de créer un B_Q de risque nul.

Nous montrons finalement à la figure 1.9 un exemple de combinaison de régresseurs de la forme $f_i(\mathbf{x}) = y_i\, k(\mathbf{x}_i, \mathbf{x})$, où $k(\mathbf{x}, \mathbf{x}') = exp\left(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2\right)$, et où $\gamma = 1$. Si on considère que ces régresseurs classifient un exemple en fonction du signe de f_i, ceux-ci ne peuvent faire guère mieux que 50% de risque. Leur combinaison permet cependant de créer des frontières de décision plus intéressantes.

De nombreux algorithmes de la littérature construisent un vote de majorité à partir de votants de différente nature. Par exemple, les algorithmes de «*boosting*» (ou d'*amplification*) comme *AdaBoost* (Freund et Schapire, 1997) combinent généralement des souches de décision. Les *forêts aléatoires* («*random forests*») (Breiman, 2001), elles, construisent itérativement des arbres de décision plus complexes pour ensuite utiliser un vote démocratique, qui donne le même poids à tous les votants. La dernière couche d'un *réseau de neurones*[5] peut également être vue comme un vote de majorité, dont les votants sont implicitement définis par la structure du réseau. Finalement, les *algorithmes à noyaux* (ou «*kernel machines*») comme les *Support Vector Machines* (Cortes et Vapnik, 1995) peuvent également être vus comme des algorithmes retournant un vote de majorité, puisqu'ils retournent un classificateur formé d'une somme pondérée de fonctions noyaux.

5. Voir Goodfellow, Bengio et Courville (2016) pour les dernières avancées dans le domaine du «*deep learning*», le nom couramment donné aux réseaux de neurones profonds.

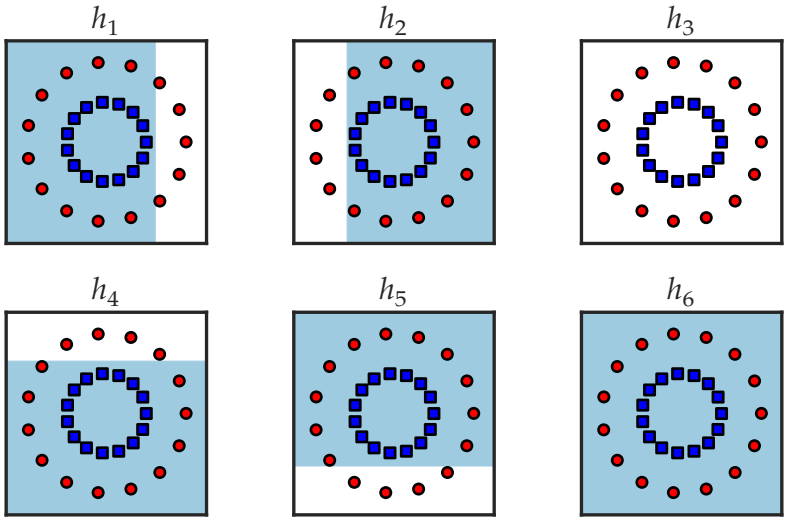

FIG. 1.7: Ensemble de six classificateurs avec une mauvaise performance individuelle, sur un ensemble de données à deux dimensions. Les classificateurs h_1, h_2, h_4 et h_5 ont un risque de 33%, et les classificateurs h_3 et h_6 ont un risque de 50%.

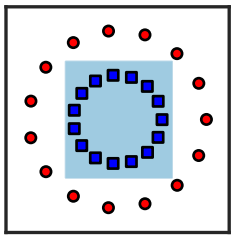

FIG. 1.8: Vote de majorité $B_Q(\mathbf{x}) = \operatorname{sgn}\left[\mathbf{E}_{h\sim Q}\, h(\mathbf{x})\right]$ de risque nul, construit à partir des classificateurs $\mathcal{H} = \{h_1, \ldots, h_6\}$ de la figure 1.7, avec $Q(h_1) = Q(h_2) = Q(h_4) = Q(h_5) = 0.16$, $Q(h_3) = 0.36$ et $Q(h_6) = 0$.

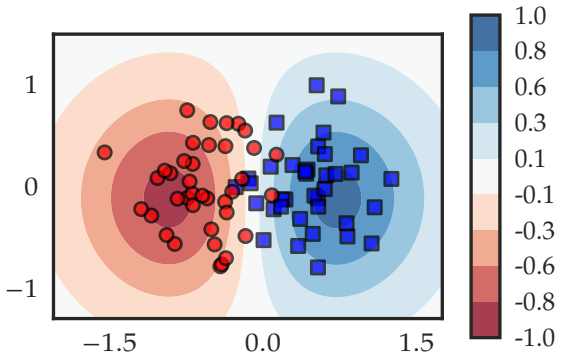

FIG. 1.9: Exemple d'un vote de majorité basé sur les deux régresseurs présentés à la figure 1.6. Plutôt que d'afficher la frontière de décision définie par $B_Q(\mathbf{x}) = \operatorname{sgn}\left[\mathbf{E}_{f\sim Q} f(\mathbf{x})\right]$, nous affichons plutôt la valeur de $\mathbf{E}_{f\sim Q} f(\mathbf{x})$ sans en prendre le signe, conservant ainsi la notion de « confiance » donnée par les régresseurs.

Étant donné un ensemble $\mathcal{H}$ de votants choisi a priori, la tâche d'un algorithme construisant un vote de majorité est de trouver une distribution Q sur $\mathcal{H}$ qui minimise le « vrai » risque du classificateur par vote de majorité, $R_D(B_Q)$. Rappelons cependant que la distribution D est inconnue, et que nous pouvons seulement calculer le risque empirique $R_S(B_Q)$ défini sur un ensemble d'entraînement S. Minimiser le risque empirique n'étant généralement pas une bonne stratégie d'apprentissage pour les raisons discutées plus haut, on voudra généralement minimiser une quantité qui dépend du risque empirique et de la « complexité » du classificateur. Dans la prochaine section, nous discutons des techniques et théories nous permettant d'obtenir une garantie qu'un algorithme apprendra un classificateur qui généralise bien sur la distribution inconnue D, et qui nous guideront dans le choix des quantités à optimiser pour construire un algorithme d'apprentissage.

1.3 Les garanties de généralisation[6]

Nous avons discuté plus haut du principe de surapprentissage, un phénomène qui arrive lorsqu'un algorithme d'apprentissage a un faible risque sur l'ensemble d'entraînement S de m exemples sur lequel il a été entraîné, mais ne généralise pas bien sur la distribution D qui génère les exemples (voir la figure 1.1 de la section 1.1.3).

Cependant, pour tout classificateur h (incluant le classificateur par vote de majorité B_Q), lorsque $m \to \infty$, $R_S(h)$ converge vers $R_D(h)$. Minimiser le risque empirique est donc une bonne stratégie lorsque m est très grand, car dans cette situation S est très représentatif de la distribution D. Malheureusement, en pratique, m n'est pas suffisamment grand et il faut donc se tourner vers d'autres avenues. La vitesse à laquelle $R_S(h)$ tend vers $R_D(h)$ dépend également du classificateur h. Lorsqu'un algorithme A choisit un classificateur $h \in \mathcal{H}$, nous désirons obtenir des garanties que $|R_D(h) - R_S(h)|$ est petit, *simultanément* pour tout $h \in \mathcal{H}$.[7] De la même manière, lorsqu'un algorithme A retourne une distribution Q sur $\mathcal{H}$ et construit donc un vote de majorité B_Q, nous désirons avoir des garanties que $|R_D(B_Q) - R_S(B_Q)|$ est petit, simultanément pour tout Q. Dans certaines situations où A retourne un classificateur de « complexité » limitée, il est possible d'obtenir ce type de garantie. Nous explorons dans cette thèse des théories d'apprentissage qui prédisent ce comportement de compromis entre le risque empirique et la complexité du classificateur retourné par un algorithme d'apprentissage, qui justifient la méthodologie appelée la *minimisation du risque structurel*, que nous introduisons plus loin en section 1.3.3. D'abord, présentons la situation où un algorithme d'apprentissage a retourné un classificateur h, avec lequel nous voulons estimer le vrai risque.

6. L'ordre de présentation et la notation de cette section ont été inspirés des notes de cours du cours IFT-7002 de Mario Marchand.

7. Obtenir des garanties *simultanément valides* pour tout $h \in \mathcal{H}$ est une condition *suffisante*, mais pas *nécessaire*: nous pourrions par exemple nous contenter de garanties de généralisation qui sont serrées seulement pour les classificateurs h de faible risque empirique. Dans l'approche *PAC-bayésienne* présentée dans cette thèse, nous serons en mesure d'obtenir des garanties valides simultanément pour tout h.

1.3.1 Estimation du vrai risque à l'aide d'un ensemble de test

Supposons que nous avons observé un certain nombre d'exemples étiquetés provenant d'une distribution inconnue D, et que nous désirons utiliser ces exemples à la fois pour entraîner un algorithme d'apprentissage pour obtenir un classificateur h, mais également estimer le *vrai* risque de h afin de s'assurer que nous n'avons pas fait de surapprentissage.

Une stratégie commune est de séparer les exemples observés en deux ensembles : un ensemble d'entraînement S de m exemples sur lequel l'algorithme sera entraîné afin d'obtenir un classificateur h, et un ensemble de test T de m' exemples, indépendant de S, qui sera utilisé exclusivement pour estimer le risque $R_D(h)$.

Le risque empirique $R_T(h)$ calculé sur l'ensemble de test T est alors un estimateur non biaisé de $R_D(h)$, puisque les exemples de T n'ont pas été utilisés pour entraîner h. Cependant, pour obtenir un bon estimateur, m' doit être suffisamment grand.

Une fois le risque $R_T(h)$ calculé, nous nous intéressons à obtenir un intervalle de confiance dans lequel $|R_D(h) - R_T(h)|$ se trouve avec grande probabilité. Lorsque nous nous intéressons à obtenir une borne supérieure sur le vrai risque, nous cherchons une fonction $B\Big(m', m'R_T(h), \delta\Big)$ permettant d'obtenir une borne supérieure de la forme

$$\underset{T\sim D^{m'}}{Pr}\left(R_D(h) \;\leq\; B\Big(m', m'R_T(h), \delta\Big)\right) \;\geq\; 1-\delta\,,$$

où $\delta \in (0, 1]$ est un paramètre de confiance. La borne sur l'ensemble test la plus serrée qui soit est obtenue par l'*inverse de la queue binomiale* (Langford, 2005), donnée par

$$\underset{T\sim D^{m'}}{Pr}\left(R_D(h) \;\leq\; \overline{\mathrm{Bin}}_+\Big(m', m'R_T(h), \delta\Big)\right) \;\geq\; 1-\delta\,,$$

où $\overline{\mathrm{Bin}}_+(m', k, \delta)$ est définie comme la plus grande valeur de risque r telle que la probabilité est au moins δ de faire au plus k erreur parmi m' exemples, c'est-à-dire

$$\overline{\mathrm{Bin}}_+(m', k, \delta) \;\triangleq\; max\left\{r : \mathrm{Bin}\Big(m', k, r\Big) \geq \delta\right\}, \tag{1.1}$$

et où $\mathrm{Bin}\Big(m', k, r\Big) \;\triangleq\; \sum_{i=0}^{k}\binom{m'}{i}(r)^i(1-r)^{m'-i}$ est la queue de la binomiale.

L'utilisation d'un ensemble de test nous permet donc d'obtenir une borne probabiliste nous garantissant qu'avec forte probabilité, le vrai risque ne sera pas trop loin du risque empirique calculé sur l'ensemble de test. Cependant, la quantité $B\Big(m', m'R_T(h), \delta\Big)$ augmente généralement lorsque m' diminue, et ainsi il faut s'assurer de conserver un nombre significatif d'exemples qui ne peuvent pas être utilisés lors de l'entraînement, ce qui est problématique dans la situation où nous n'avons pas accès à un grand nombre d'exemples. De plus, ces bornes ne sont valides que pour un seul h à la fois, ce qui implique qu'il est ensuite nécessaire de combiner plusieurs bornes en une seule à l'aide d'une méthode comme la *borne de l'union* (lemme A.1) pour obtenir des garanties de généralisation sur plusieurs h à la fois. Ceci a pour effet de dégrader les garanties en fonction du nombre de classificateurs h considérés.

1.3.2 Validation croisée et sélection d'hyperparamètres

La *validation croisée* est une technique permettant d'évaluer un algorithme d'apprentissage en évitant de mettre de côté un grand nombre m' d'exemples dans un ensemble de test. Cette technique est couramment utilisée dans les situations où le nombre d'exemples disponibles est limité, pour remplacer l'utilisation d'un ensemble de test T disjoint. L'algorithme 1 présente le pseudo-code de la validation croisée.

Algorithme 1 La validation croisée à k plis $CV(A, S, k)$

Entrée : Un algorithme A, un ensemble de données S et un nombre de *plis* $k \geq 2$
1 : Partitionner S en k ensembles disjoints $\{S_1, \dots, S_k\}$ de taille similaire
2 : **pour** $i \in \{1, \dots, k\}$ **faire**
3 : Apprendre un classificateur $h_i \leftarrow A(S \setminus S_i)$ en exécutant l'algorithme A sur $S \setminus S_i$
4 : Calculer le risque $R_i = R_{S_i}(h_i)$
5 : **fin pour**
Sortie : Le *risque de validation croisée* $R_{CV} = \frac{1}{k} \sum_{i=1}^{k} R_i$

La validation croisée est également utilisée en combinaison avec l'utilisation d'un ensemble de test, lorsqu'un algorithme A possède un ou plusieurs *hyperparamètres*, dont la valeur doit être choisie avant d'exécuter l'algorithme. Dans cette situation, les données sont séparées en deux ensembles S et T, puis la validation croisée est exécutée sur S afin de choisir les valeurs d'hyperparamètres. Une fois ces valeurs choisies, la totalité de l'ensemble S est réutilisée pour entraîner un classificateur final. L'algorithme 2 montre le pseudo-code de cette technique.

Algorithme 2 La sélection d'hyperparamètres par validation croisée

Entrée : Un algorithme A, un ensemble $\mathcal{P}$ de valeurs d'hyperparamètres à considérer, un ensemble d'entraînement S, un ensemble de test T et un nombre de plis $k \geq 2$
1 : **pour** $p \in \mathcal{P}$ **faire**
2 : Soit A_p l'algorithme A pour lequel les valeurs p d'hyperparamètres ont été fixées
3 : Obtenir le risque de validation croisée $R_{CV}(p) \leftarrow CV(A_p, S, k)$
4 : **fin pour**
5 : Choisir $p^\star = \operatorname{argmin}_p R_{CV}(p)$
6 : Entraîner $h \leftarrow A_{p^\star}(S)$
Sortie : Le risque de test $R_T(h)$

La validation croisée permet de maximiser l'utilisation des données, ce qui est intéressant dans la situation où le nombre d'exemples observés est petit. Cependant, contrairement à l'utilisation d'un ensemble de test dont les exemples n'ont pas été utilisés lors de l'entraînement, la validation croisée ne permet pas d'obtenir des garanties de généralisation rigoureuses. Dans la prochaine section, nous discutons des *bornes sur l'ensemble d'entraînement*, une famille de bornes donnant des garanties de généralisation simultanément valides pour tout classificateur que notre algorithme d'apprentissage puisse retourner, et qui ne nécessite pas de conserver un ensemble de test qui ne peut pas être utilisé pendant l'entraînement.

1.3.3 Bornes sur l'ensemble d'entraînement

Les bornes sur l'ensemble d'entraînement donnent des garanties de généralisation simultanément valides pour pour tout classificateur appartenant à une famille $\mathcal{H}$ de classificateurs. Ces bornes sont énoncées en fonction de ce qu'un algorithme d'apprentissage a pu accomplir sur un ensemble d'entraînement en termes de risque empirique et d'autres informations sur le classificateur appris, généralement reliées à sa complexité. Ces bornes nous permettent donc en quelque sorte d'obtenir des garanties de généralisation pour un algorithme, mais nous donnent également de l'information sur ce qu'un algorithme devrait optimiser comme quantité pour obtenir la meilleure garantie possible.

Nous cherchons donc une fonction $B\Big(m, m\,R_S(h), h, \mathcal{H}, \delta\Big)$ permettant d'obtenir une borne supérieure de la forme

$$\underset{S\sim D^m}{Pr}\Big(\forall h \in \mathcal{H}:\ \ R_D(h)\ \leq\ B\Big(m, m\,R_S(h), h, \mathcal{H}, \delta\Big)\Big)\ \geq\ 1-\delta\,.$$

L'une des premières bornes de ce type s'appelle le *rasoir d'Occam* (Blumer et al., 1990), en l'honneur du principe de parcimonie énoncé par Sir William of Occam, dont une formulation moderne est « les hypothèses suffisantes les plus simples sont les plus vraisemblables ». La borne du rasoir d'Occam dépend d'une distribution P sur $\mathcal{H}$ encodant notre connaissance a priori sur les classificateurs de $\mathcal{H}$, et est valide pour les ensemble $\mathcal{H}$ dénombrables. Cette borne expose un compromis entre le risque empirique $R_S(h)$ et un terme de complexité basé sur l'entropie d'information associé à h, donnée par $-\ln P(h)$.

En s'inspirant du rasoir d'Occam, une borne sur l'ensemble d'entraînement uniformément valide pour tout $h \in \mathcal{H}$ sur un ensemble $\mathcal{H}$ dénombrable peut également être obtenue à l'aide de l'inverse de la queue de la binomiale (Langford, 2005), avec $B\Big(m, m\,R_S(h), h, \mathcal{H}, \delta\Big) = \overline{\mathrm{Bin}}_+\Big(m, m\,R_S(h), \delta P(h)\Big)$, où la fonction $\overline{\mathrm{Bin}}_+$ est définie à l'équation (1.1).

Les deux bornes ci-haut nous indiquent les quantités importantes à prendre en considération pour qu'un algorithme d'apprentissage qui choisit un $h \in \mathcal{H}$ ait un faible risque $R_D(h)$. Cependant, celles-ci sont valides uniquement pour les ensembles $\mathcal{H}$ dénombrables comme les arbres de décision. Les familles $\mathcal{H}$ de classificateurs continus comme les séparateurs linéaires ne peuvent pas être considérés. Vapnik et Chervonenkis (1971) ont proposé des bornes valables pour les ensembles continus de classificateurs, en introduisant une quantité appelée la *dimension Vapnik-Chervonenkis d* d'un ensemble $\mathcal{H}$ de classificateurs, quantifiant sa complexité. Une telle borne sur l'ensemble d'entraînement est obtenue (voir Bousquet, Boucheron et Lugosi, 2004) avec $B\Big(m, m\,R_S(h), h, \mathcal{H}, \delta\Big) = R_S(h) + \sqrt{\frac{8}{m}\Big[d\ ln\Big(\frac{2m}{d}+1\Big) + ln\,\frac{2}{\delta}\Big]}$. Les bornes basées sur la dimension Vapnik-Chervonenkis sont généralement lâches, mais justifient le principe de la minimisation du risque structurel, en suggérant qu'un algorithme d'apprentissage devrait faire un compromis entre la minimisation du risque empirique et la complexité de la famille de classificateurs $\mathcal{H}$.

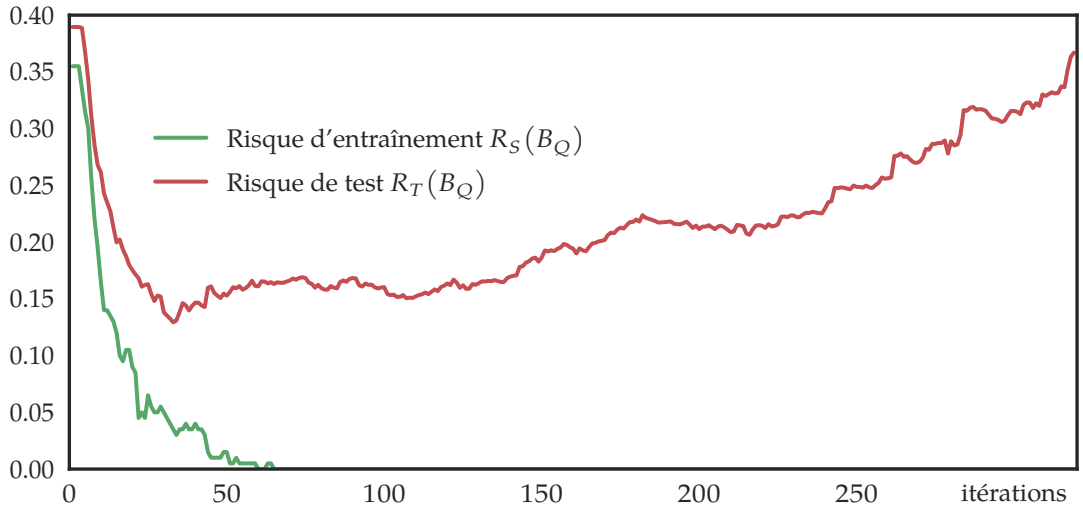

FIG. 1.10: Exemple d'effet de la complexité d'un classificateur sur la capacité d'un algorithme à bien généraliser. Les risques $R_S(B_Q)$ et $R_T(B_Q)$ sont calculés à chaque itération de l'algorithme CqBoost introduit au chapitre 5, qui produit un classificateur de plus en plus complexe en fonction du nombre d'itérations. On remarque qu'après 40 itérations, le risque $R_S(B_Q)$ continue de descendre jusqu'à l'atteinte d'un pallier à zéro, alors que le risque $R_T(B_Q)$ commence à augmenter. L'algorithme devra donc faire un compromis entre la minimisation du risque empirique et la complexité du classificateur produit.

1.3.4 Théorie PAC-bayésienne

La théorie PAC-bayésienne, initiée par McAllester (1999), nous permet d'obtenir des bornes sur l'ensemble d'entraînement pour le classificateur par vote de majorité B_Q, simultanément valides pour tout Q. Ces bornes sont généralement plus serrées que les bornes basées sur la dimension Vapnik-Chervonenkis, et permettent de faire ressortir un compromis entre le risque empirique et une notion de complexité, justifiant ainsi une minimisation du risque structurel. La figure 1.10 montre un exemple d'effet qu'a la complexité d'un classificateur par vote de majorité sur sa capacité à bien généraliser.

Les bornes PAC-bayésiennes portent ce nom car elles visent à établir des garanties *PAC*, pour *probablement approximativement correct*, pour des algorithmes issus de la méthodologie bayésienne.[8] Une extension fréquentiste de l'approche bayésienne consiste à considérer une distribution *a priori* P sur un ensemble $\mathcal{H}$ de votants représentant notre degré de confiance sur chaque votant h de $\mathcal{H}$, pour ensuite apprendre une distribution *a posteriori* Q sur $\mathcal{H}$ (et ainsi un classificateur par vote de majorité B_Q) à partir d'un ensemble de données S. Le choix de

8. Bien que la théorie PAC-bayésienne ait été introduite pour établir des garanties PAC aux algorithmes bayésiens (McAllester, 1999), cette méthode est plutôt utilisée comme une approche *fréquentiste*. À notre connaissance, le premier lien fort entre les approches bayésiennes et la théorie PAC-bayésienne a été introduit récemment par Germain, Bach et al. (2016).

cette distribution Q dépend de la distribution a priori P et de l'information apprise dans les données de S. La borne PAC-bayésienne suivante est inspirée de la borne originale de McAllester (1999). Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de classificateurs, toute distribution a priori P sur $\mathcal{H}$ et tout $\delta \in (0, 1]$, nous avons

$$\Pr_{S \sim D^m} \left(\forall Q \text{ sur } \mathcal{H} : \ R_D(B_Q) \ \leq \ 2 \left[R_S(G_Q) + \sqrt{\frac{1}{2m} \left[\mathrm{KL}(Q \,\|\, P) + ln \, \frac{2\sqrt{m}}{\delta} \right]} \right] \right) \ \geq \ 1 - \delta \,,$$

où $R_S(G_Q)$ est le *risque empirique de Gibbs*, défini par $\frac{1}{m} \sum_{k=1}^{m} \mathbf{E}_{h \sim Q} \, I(h(x_k) \neq y_k)$, et $\mathrm{KL}(Q \,\|\, P)$ est la *divergence Kullback-Leibler* entre les distributions Q et P, définie par $\mathbf{E}_{h \sim Q} \, ln \, \frac{Q(h)}{P(h)}$.

Il s'agit donc d'une borne de généralisation sur l'ensemble d'entraînement, en considérant que l'ensemble des classificateurs possibles est l'ensemble de tous les votes de majorité possibles sur $\mathcal{H}$. Cette borne nous apprend que pour choisir une distribution Q qui donnera un classificateur par vote de majorité B_Q de faible risque, un algorithme d'apprentissage peut minimiser une quantité qui dépend du risque empirique de Gibbs et de la divergence Kullback-Leibler entre les distributions Q et P, représentant la complexité du vote de majorité. Des algorithmes de la littérature minimisent directement l'expression de bornes PAC-bayésiennes (Germain, Lacasse, Laviolette et Marchand, 2009 ; Germain, Lacoste et al., 2011), et nous construisons également de tels algorithmes dans cette thèse.

Nous verrons dans cette thèse qu'il est possible de simplifier l'expression des bornes PAC-bayésiennes en introduisant une forme de restriction sur les distributions Q. Les bornes résultantes ne contiennent pas de terme de complexité $\mathrm{KL}(Q \ P)$ et nous permettent de développer des algorithmes d'apprentissage minimisant directement l'expression des bornes. Ces dernières ne dépendent que d'une quantité empirique nommée la C-borne, combinant deux informations reliées au vote de majorité. L'une correspond au risque empirique de Gibbs, et peut être interprétée comme une mesure de la confiance que le vote de majorité accorde aux votants individuels. L'expression de la borne suggère que cette confiance ne doit pas être trop grande. La seconde quantité correspond à une mesure de *désaccord* entre les votants, qui lui doit être le plus grand possible. Nous développerons deux algorithmes d'apprentissage, dont l'un minimise directement l'expression d'une telle borne de généralisation PAC-bayésienne, et l'autre s'en inspire et apporte une forme de régularisation basée sur la parcimonie.

1.4 Les différents cadres d'apprentissage

Jusqu'à maintenant dans cette introduction, nous avons fait la supposition que la tâche d'apprentissage est la suivante : il existe une distribution fixe mais inconnue D, de laquelle nous obtenons un échantillon S de manière i.i.d., et nous désirons apprendre une fonction qui aura une bonne performance sur les prochains exemples tirés selon D. Ce cadre d'apprentissage est le plus étudié en théorie de l'apprentissage statistique, et se nomme l'*apprentissage inductif*. La figure 1.2 en fait le résumé de manière visuelle.

Il existe cependant d'autres cadres d'apprentissage, dont certains rendent la tâche plus difficile. L'*adaptation de domaine* en est un exemple, où les données observées proviennent d'une distribution différente que celle sur laquelle nous désirons être en mesure de généraliser par la suite. Un autre exemple commun, plus près de la réalité pour plusieurs problèmes d'apprentissage, se nomme le «*distribution drift*» et correspond à la situation où la distribution qui génère les exemples change dans le temps. La détection de «*spam*», la prédiction de marché boursier et la prédiction de résultats politiques en fonction de contenu journalistique sont de bons exemples de problèmes dont la distribution évolue dans le temps.

Un autre cadre d'apprentissage, auquel nous nous intéressons dans le dernier chapitre de cette thèse, peut quand à lui être considéré comme une simplification du cadre inductif. Il s'agit du cadre *transductif*, qui correspond à la situation où nous ne nous intéressons qu'à prédire les étiquettes des exemples dont nous connaissons déjà la description. On considère donc que nous avons à notre disposition un ensemble fini Z nommé l'*échantillon complet*, dont un sous-ensemble S est étiqueté, et un sous-ensemble $U = Z \setminus S$ ne l'est pas. Ici, l'algorithme d'apprentissage ne cherche pas à apprendre un prédicteur en mesure de généraliser à une distribution inconnue, mais bien à étiqueter les exemples de l'ensemble U. La figure 1.11 présente l'apprentissage transductif de manière visuelle.

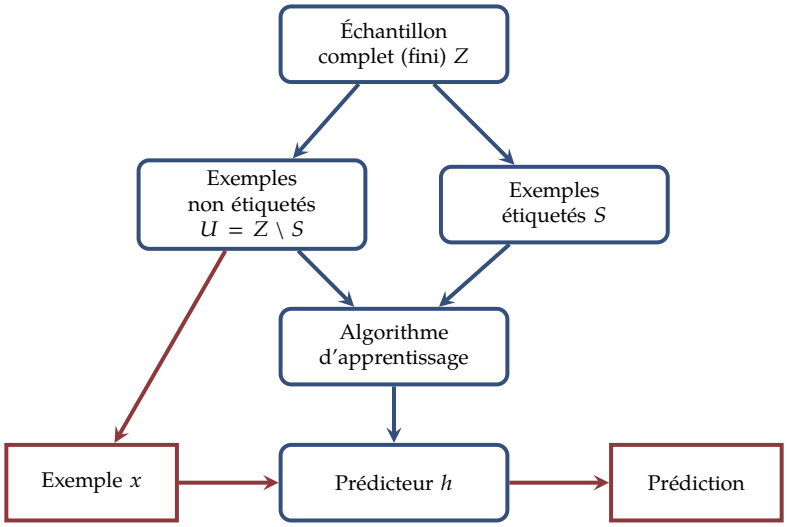

FIG. 1.11: Représentation du cadre de l'apprentissage transductif. Le chemin déterminé par les flèches bleues représente le processus d'apprentissage, et le chemin déterminé par les flèches rouges représente la tâche à résoudre une fois l'apprentissage terminé.

1.5 Contributions de la thèse

Cette thèse repose en partie sur des travaux antérieurs de certains de nos co-auteurs (LACASSE et al., 2007), où la C-borne a été publiée pour la première fois. À l'époque, des garanties PAC-

bayésiennes justifiant la minimisation de la C-borne empirique ont été développées, mais la création d'un tel algorithme d'apprentissage a été infructueuse. La plupart de nos contributions ont été une poursuite de ces idées de recherche. Nous avons entre autres

- Généralisé la C-borne aux votants à valeur réelle et redéfini les notions de base en fonction de la *marge* du vote de majorité, une quantité mieux étudiée dans la littérature relative aux méthodes d'ensemble (Laviolette, Marchand et Roy, 2011 ; Germain, Lacasse, Laviolette, Marchand et Roy, 2015) ;
- Unifié les théories PAC-bayésiennes en introduisant une borne très générale pouvant être spécialisée pour retomber sur celles de l'état de l'art (Germain, Lacasse, Laviolette, Marchand et Roy, 2015, collaboration) ;
- Introduit une restriction sur les votes de majorité menant à des bornes de généralisation ne dépendant que des deux premiers moments empiriques de la marge (Laviolette, Marchand et Roy, 2011 ; Germain, Lacasse, Laviolette, Marchand et Roy, 2015) ;
- Développé un algorithme d'apprentissage, sous la forme d'un programme quadratique, minimisant directement une borne de généralisation PAC-bayésienne. Cet algorithme travaille directement à maximiser le désaccord, tout en restreignant les votes de majorité à ne pas être trop confiants sur des votants individuels (Laviolette, Marchand et Roy, 2011 ; Germain, Lacasse, Laviolette, Marchand et Roy, 2015) ;
- Développé un algorithme itératif inspiré de la C-borne, de la théorie PAC-bayésienne et de l'optimisation de Lagrange, permettant de construire des votes de majorité parcimonieux (Laviolette, Marchand et Roy, 2014 ; Roy, Marchand et Laviolette, 2016) ;
- Généralisé la C-borne et les bornes PAC-bayésiennes associées aux votes de majorité à sortie arbitraire (Laviolette, Morvant, Ralaivola et Roy, 2014 ; Laviolette, Morvant, Ralaivola et Roy, 2017, collaboration) ;
- Simplifié les démonstrations de théorèmes PAC-bayésiens, permettant de les personnaliser plus facilement pour obtenir de nouvelles bornes (Bégin, Germain, Laviolette et Roy, 2016, collaboration) ;
- Étendu la théorie PAC-bayésienne basée sur la divergence de Rényi plutôt que la divergence Kullback-Leibler (Bégin, Germain, Laviolette et Roy, 2016, collaboration) ;
- Étendu de la théorie PAC-bayésienne au cadre de l'apprentissage transductif (Bégin, Germain, Laviolette et Roy, 2014, collaboration).

Nous avons également participé à l'élaboration de deux articles dont le contenu n'est pas inclus dans cette thèse. Dans Germain, Giguere, Roy, Zirakiza, Laviolette et Quimper (2012), nous étudions l'optimalité d'une heuristique sur laquelle est basée l'algorithme *Set Covering*

Machine (Marchand et Shawe-Taylor, 2001) à l'aide de l'optimisation pseudo-booléenne. Dans Fortier-Dubois, Laviolette, Marchand, Robitaille et Roy (2015), nous développons un algorithme d'apprentissage basé sur une borne de généralisation de *Rademacher*.

Nous rapportons ci-bas l'ensemble de nos publications dans des revues et conférences internationales avec comité de lecture, dans des « *workshops* » avec comité de lecture, associés à une conférence internationale, et un rapport technique non publié. Notons que nos publications listent les auteurs en ordre alphabétique, excepté Roy, Marchand et Laviolette (2016).

Bégin, Luc, Pascal Germain, François Laviolette et Jean-Francis Roy (2014). «PAC-Bayesian Theory for Transductive Learning». Dans : *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, p. 105–113.

Bégin, Luc, Pascal Germain, François Laviolette et Jean-Francis Roy (2016). «PAC-Bayesian Bounds Based on the Rényi Divergence». Dans : *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, p. 435–444.

Fortier-Dubois, Louis, François Laviolette, Mario Marchand, Louis-Emile Robitaille et Jean-Francis Roy (2015). *Efficient Learning of Ensembles with QuadBoost*. arXiv. url : http://arxiv.org/abs/1506.02535.

Germain, Pascal, Sébastien Giguere, Jean-Francis Roy, Brice Zirakiza, François Laviolette et Claude-Guy Quimper (2012). «A Pseudo-Boolean Set Covering Machine». Dans : *Principles and Practice of Constraint Programming*. Springer, p. 916–924.

Germain, Pascal, Alexandre Lacasse, François Laviolette, Mario Marchand et Jean-Francis Roy (2015). «Risk Bounds for the Majority Vote : From a PAC-Bayesian Analysis to a Learning Algorithm». Dans : *Journal of Machine Learning Research (JMLR)*, p. 787–860.

Laviolette, François, Mario Marchand et Jean-Francis Roy (2011). «From PAC-Bayes Bounds to Quadratic Programs for Majority Votes». Dans : *Proceedings of the 28th International Conference on Machine Learning (ICML)*, p. 649–656.

Laviolette, François, Mario Marchand et Jean-Francis Roy (2014). «CqBoost : A Column Generation Method for Minimizing the C-Bound». Dans : *NIPS Workshop on Optimization for Machine Learning*.

Laviolette, François, Emilie Morvant, Liva Ralaivola et Jean-Francis Roy (2014). «On Generalizing the C-Bound to the Multiclass and Multi-label Settings». Dans : *NIPS Workshop on Representation and Learning Methods for Complex Outputs*.

Laviolette, François, Emilie Morvant, Liva Ralaivola et Jean-Francis Roy (2017). «Risk Upper Bounds for General Ensemble Methods with an application to Multiclass Classification». Dans : *Neurocomputing* 219, p. 15–25.

Roy, Jean-Francis, Mario Marchand et François Laviolette (2016). «A Column Generation Bound Minimization Approach with PAC-Bayesian Generalization Guarantees». Dans : *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, p. 1241–1249.

Plusieurs de articles ont une nombre grandissant de citations, notamment notre article de JMLR qui semble avoir retenu l'attention dans la communauté de l'apprentissage PAC-bayésien. Cet article cumule aussi des citations farfelues et hors contexte, pour des raisons que nous ne nous expliquons toujours pas. Nos algorithmes MinCq et CqBoost ont également attiré l'attention en dehors de notre laboratoire de recherche. Ils ont entre autres été étendus à d'autres paradigmes que la classification binaire classique. Notamment, Bellet et al. (2014) ont adapté MinCq pour qu'il puisse prendre en considération des poids choisis a priori, Morvant, Habrard et Ayache (2014) l'ont utilisé dans le cadre de la *fusion de classificateurs*, et Morvant (2015) l'a appliqué à l'*adaptation de domaine*. Finalement, un algorithme nommé FusionCq (Goyal et al., 2017) étend CqBoost au cadre de l'apprentissage multi-vue.

1.6 Organisation de la thèse

Les chapitres 2 à 4 suivent le même ordre que Germain, Lacasse, Laviolette, Marchand et Roy (2015), qui est une réécriture plus pédagogique de plusieurs articles publiés entre 2007 et 2011, à partir de l'article original présentant la C-borne (Lacasse et al., 2007), jusqu'à l'algorithme MinCq (Laviolette, Marchand et Roy, 2011), l'algorithme de minimisation de la C-borne qui est central à cette thèse. Le chapitre 2 présente les votes de majorité, des notions de base jusqu'à l'introduction de la C-borne, de ses propriétés statistiques et empiriques. Le chapitre 3 présente l'unification de la théorie PAC-bayésienne, permettant de retrouver les bornes PAC-bayésiennes classiques et d'en créer de nouvelles tirant profit de la C-borne. Le chapitre 4 présente l'algorithme MinCq, qui minimise directement une borne PAC-bayésienne basée sur la C-borne.

Les chapitres 5, 6 et 7 construisent sur les chapitres précédents et peuvent être lus dans l'ordre ou dans le désordre. Le chapitre 5 présente CqBoost, un algorithme construisant itérativement un vote de majorité en utilisant la technique de la *génération de colonnes* et l'optimisation de Lagrange. CqBoost est inspiré de bornes PAC-bayésiennes mais ne les minimise pas directement. Le chapitre 6 généralise les votes de majorité, la C-borne et les bornes PAC-bayésiennes aux votants dont l'espace de sortie est arbitraire. Une étude empirique est présentée pour le cadre de la classification multi-classe. Finalement, le chapitre 7 présente un processus simplifié de preuve pour les théorèmes PAC-bayésiens, permettant de simplifier la personnalisation et la création de nouvelles borne PAC-bayésiennes. Nous proposons ensuite deux nouvelles familles de bornes PAC-bayésiennes : l'une dont la divergence Kullback-Leibler est remplacée par la *divergence de Rényi*, et l'autre spécialisée au cadre de l'apprentissage transductif. À partir de cette dernière, nous construisons l'algorithme TMinCq, une version de MinCq spécialisée au cadre transductif, et LapMinCq, une version de TMinCq introduisant un terme de régularisation basé sur la géométrie intrinsèque des données. La figure 1.12 présente le contenu de la thèse de manière visuelle, en divisant les concepts par catégorie et en montrant les liens entre eux.

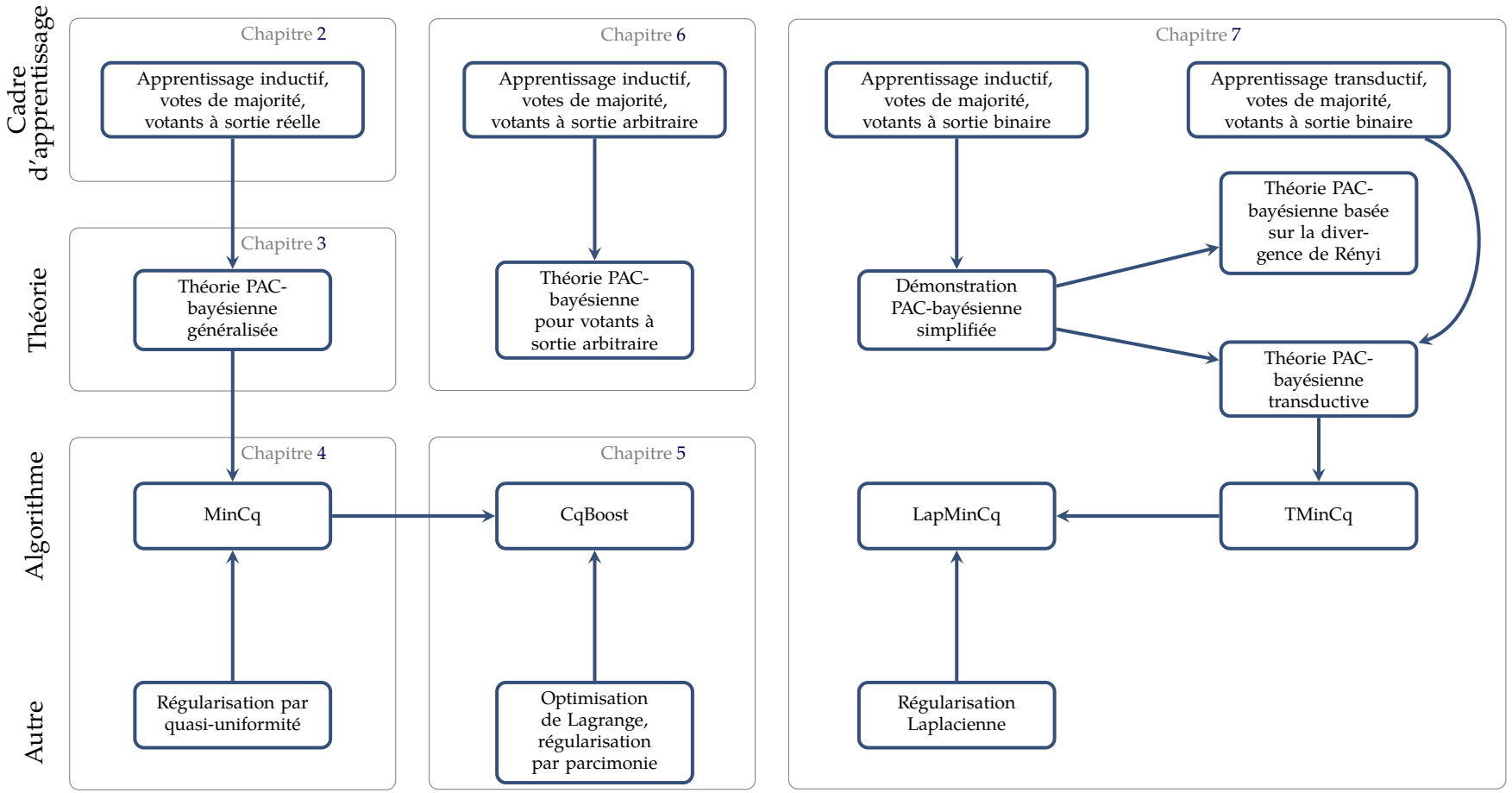

FIG. 1.12: Présentation visuelle de l'organisation de la thèse.

Chapitre 2

Votes de majorité en apprentissage automatique

Ce chapitre présente les *votes de majorité*, qui sont à la base de cette thèse. Nous y donnons les définitions de base et introduisons la C-borne et ses propriétés statistiques. Les algorithmes développés dans cette thèse reposent tous sur cette borne et ses extensions, ce qui lui donne un rôle central.

Les définitions et résultats présentés dans ce chapitre sont en partie dus à un précédent article de nos collaborateurs (Lacasse et al., 2007), dont nous avons révisé l'approche mathématique une première fois dans Laviolette, Marchand et Roy (2011), et de façon majeure dans Germain, Lacasse, Laviolette, Marchand et Roy (2015). Ce dernier article propose une revue complète de la notation et de l'ordre de présentation des divers concepts, qui se veut plus pédagogique qu'à l'origine. Nous avons participé à parts égales avec nos collaborateurs à cette réalisation.

2.1 Définitions de base et notation

Nous considérons les problèmes de classification où l'espace d'entrée $\mathcal{X}$ est un ensemble quelconque, et où l'espace de sortie est un ensemble discret désigné par $\mathcal{Y}$. Un exemple (x, y) est une paire entrée-sortie, où $x \in \mathcal{X}$ et $y \in \mathcal{Y}$.

Un *votant* est une fonction $\mathcal{X} \to \overline{\mathcal{Y}}$, où $\overline{\mathcal{Y}}$ est un ensemble relié à $\mathcal{Y}$. Dans cette thèse, sauf sous indication contraire, nous considérons les problèmes de classification binaire où $\mathcal{Y} = \{-1, 1\}$. L'espace de sortie $\overline{\mathcal{Y}}$ des votants est soit $\mathcal{Y}$ lui-même, ou son *enveloppe convexe* $[-1, +1]$. Dans cette thèse, nous utilisons la convention suivante : f désigne un votant à valeur réelle (c'est-à-dire, $\overline{\mathcal{Y}} = [-1, 1]$), et h désigne un votant à valeur binaire (c'est-à-dire, $\overline{\mathcal{Y}} = \{-1, 1\}$).

Nous considérons les algorithmes d'apprentissage qui construisent des *votes de majorité* ba-

sés sur un ensemble fini de votants. Cet ensemble sera désigné par $\mathcal{H}$ lorsqu'il s'agit d'un ensemble de votants à valeur binaire, et $\mathcal{F}$ lorsque les votants sont à valeur réelle. Lorsque nous considérons un votant tiré selon une distribution Q sur $\mathcal{H}$ ou $\mathcal{F}$, le symbole représentant le votant tiré permettra de déterminer s'il s'agit d'un votant à valeur binaire (h) ou à valeur réelle (f). Notons que l'utilisation d'un ensemble de votants à valeur réelle est plus générale que l'utilisation d'un votant à valeur binaire. Lorsque la théorie le permet, nous utiliserons donc des votants à valeur réelle.

Étant donné un exemple $x \in \mathcal{X}$, la sortie $B_Q(x)$ d'un classificateur par vote de majorité Q-pondéré B_Q (parfois appelé le *classificateur de Bayes*) est donnée par

$$B_Q(x) \triangleq \operatorname{sgn}\left[\mathop{\mathbf{E}}_{f\sim Q} f(x)\right], \tag{2.1}$$

où $\operatorname{sgn}(a) = 1$ si $a > 0$, $\operatorname{sgn}(a) = -1$ si $a < 0$, et $\operatorname{sgn}(0) = 0$.

Dans le cas d'une égalité dans le vote de majorité (c'est-à-dire, lorsque $\mathbf{E}_{f\sim Q} f(x) = 0$), nous considérons que le vote de majorité s'abstient ($B_Q(x) = 0$).

Nous adoptons le modèle *PAC* (*probablement approximativement correct*), où chaque exemple (x, y) est tiré *i.i.d.* (*indépendamment et identiquement distribué*) selon une distribution fixe mais inconnue D sur $\mathcal{X}\times\mathcal{Y}$. L'*ensemble d'entraînement* de m exemples est désigné par $S = \langle (x_1, y_1), \ldots, (x_m, y_m) \rangle \sim D^m$. Dans cette thèse, D' représente une distribution générique sur $\mathcal{X}\times\mathcal{Y}$. On utilise donc D' pour représenter la vraie distribution D qui est inconnue, ou la distribution uniforme sur un ensemble discret comme l'ensemble d'entraînement S.

Convention 2.1. *La notation $(x, y) \sim D'$ représente le tirage d'un élément selon une distribution. Pour simplifier la notation, lorsque l'opérande de droite de l'opérateur $\sim$ est un ensemble discret, nous considérons un tirage selon la distribution uniforme sur cet ensemble. Nous dénoterons donc $(x, y) \sim S$ le tirage d'un exemple selon la distribution uniforme sur l'ensemble discret S.*

Convention 2.2. *Tel que mentionné en introduction, il serait plus rigoureux de parler d'une séquence de données plutôt que d'un ensemble, puisque les exemples peuvent être répétés et qu'ils sont généralement ordonnés pour simplifier la notation. Dans cette thèse, nous utilisons tout de même le terme « ensemble de données », un abus de langage courant dans la littérature. Cependant, nous utilisons des crochets $\langle\ \rangle$ plutôt que des accolades $\{\ \}$ pour que la notation soit plus juste.*

2.1.1 Fonctions de perte

Afin de quantifier la précision d'un votant, nous utilisons une *fonction de perte* $\mathcal{L} : \overline{\mathcal{Y}}\times\mathcal{Y} \to [0, 1]$. La théorie PAC-bayésienne traditionnelle considère les votes de majorité de votants binaires de la forme $h : \mathcal{X} \to \{-1, 1\}$, et la *perte zéro-un* $\mathcal{L}_{01}(h(x), y) \triangleq I(h(x) \neq y)$, où $I(a) = 1$ si le prédicat a est vrai, et 0 autrement. L'extension de la perte zéro-un aux votants à valeur réelle (de la forme $f : \mathcal{X} \to [-1, 1]$) est donnée par la définition suivante.

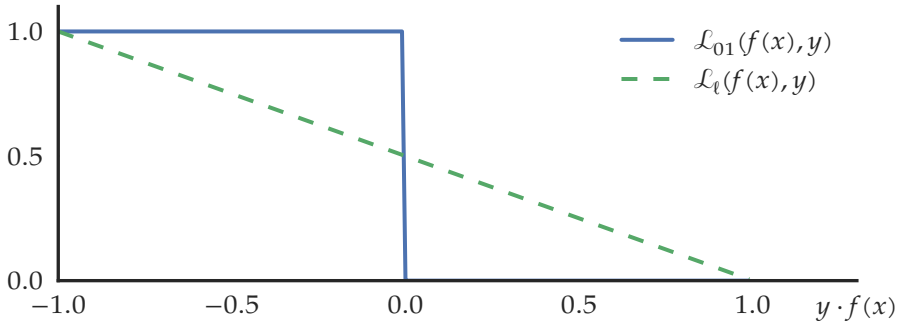

FIG. 2.1: La perte zéro-un $\mathcal{L}_{01}$ et la perte linéaire $\mathcal{L}_{\ell}$ en fonction de $y \cdot f(x)$.

Définition 2.3. Lorsque les votants sont des fonctions $f : \mathcal{X} \to [-1, 1]$, la perte zéro-un $\mathcal{L}_{01}$ est définie par

$$\mathcal{L}_{01}(f(x), y) \triangleq I(y \cdot f(x) \leq 0)\,.$$

Avec cette définition, lorsqu'un votant s'abstient (c'est-à-dire, lorsque $f(x)$ retourne 0), la perte sera de 1. Cette définition de la perte lorsqu'un votant s'abstient est la plus pessimiste, mais simplifie l'analyse des votes de majorité.

Dans cette thèse, nous considérons également la *perte linéaire*, définie comme suit.

Définition 2.4. Étant donné un votant $f : \mathcal{X} \to [-1, 1]$, la perte linéaire $\mathcal{L}_{\ell}$ est définie par

$$\mathcal{L}_{\ell}(f(x), y) \triangleq \frac{1}{2}\big(1 - y \cdot f(x)\big).$$

Notons que la perte linéaire est équivalente à la perte zéro-un lorsque l'espace de sortie des votants est binaire. Pour tout $(h(x), y) \in \{-1, 1\}^2$, nous avons effectivement

$$\mathcal{L}_{\ell}(h(x), y) \;=\; \mathcal{L}_{01}(h(x), y)\,, \tag{2.2}$$

comme $\mathcal{L}_{\ell}(h(x), y) = 1$ si $h(x) \neq y$, et $\mathcal{L}_{\ell}(h(x), y) = 0$ si $h(x) = y$.

Nous pouvons donc généraliser toutes les définitions impliquant des votants binaires aux votants à valeur réelle en s'inspirant de l'équation (2.2).

La figure 2.1 illustre la différence entre la perte zéro-un et la perte linéaire pour les votants à valeur réelle. Rappelons que lorsque $y \cdot f(x) = 0$, la perte est 1 (voir la définition 2.3).

2.1.2 Pertes espérées et risque du classificateur par vote de majorité

Étant donné une fonction de perte $\mathscr{L}$ et un votant f, la *perte espérée* $\mathbb{E}^{\mathscr{L}}_{D'}(f)$ de f par rapport à une distribution D' est définie par

$$\mathbb{E}^{\mathscr{L}}_{D'}(f) \triangleq \mathop{\mathbf{E}}_{(x,y)\sim D'} \mathscr{L}(f(x),y)\,. \tag{2.3}$$

En particulier, la *perte espérée empirique* sur un ensemble de données S est donnée par

$$\mathbb{E}^{\mathscr{L}}_{S}(f) = \frac{1}{m}\sum_{i=k}^{m} \mathscr{L}(f(x_k),y_k)\,.$$

Nous définissons alors le *risque* du vote de majorité $R_{D'}(B_Q)$ comme suit.

Définition 2.5. Pour toute distribution de probabilité Q sur un ensemble de votants, le *risque du classificateur par vote de majorité* $R_{D'}(B_Q)$, aussi appelé le *risque de Bayes*, est défini comme la perte zéro-un espérée du classificateur par vote de majorité B_Q, par rapport à D'. Alors,

$$R_{D'}(B_Q) \triangleq \mathbb{E}^{\mathscr{L}_{01}}_{D'}(B_Q) = \mathop{\mathbf{E}}_{(x,y)\sim D'} I\big(B_Q(x)\neq y\big) = \mathop{\mathbf{E}}_{(x,y)\sim D'} I\left(\mathop{\mathbf{E}}_{f\sim Q} y\cdot f(x)\leq 0\right).$$

Rappelons qu'étant donné la définition de B_Q de l'équation (2.1), le vote de majorité s'abstient dans le cas d'une égalité sur un exemple (x,y). La définition du risque de Bayes implique donc que le risque est de 1 dans ce cas, comme $R_{\langle(x,y)\rangle}(B_Q) = \mathscr{L}_{01}(0,y) = 1$. Notons qu'une égalité dans le vote de majorité est un événement rarement observé en pratique.

2.1.3 Le classificateur de Gibbs

La sortie du classificateur déterministe par vote de majorité B_Q est reliée de près à la sortie d'un classificateur stochastique appelé le *classificateur de Gibbs*. Pour classifier un exemple x, le classificateur de Gibbs G_Q choisit au hasard un votant f selon la distribution Q, et retourne $f(x)$. Notons la stochasticité du classificateur de Gibbs : celui-ci peut retourner différentes valeurs étant donné la même entrée x. Nous verrons plus loin comment le lien entre B_Q et G_Q est utilisé dans la théorie PAC-bayésienne.

Dans le cas des votants binaires, le risque de Gibbs correspond à la probabilité que G_Q se trompe sur un exemple tiré selon une distribution D'. Ainsi,

$$R_{D'}(G_Q) = \Pr_{\substack{(x,y)\sim D'\\ h\sim Q}}(h(x)\neq y) = \mathop{\mathbf{E}}_{h\sim Q}\mathbb{E}^{\mathscr{L}_{01}}_{D'}(h) = \mathop{\mathbf{E}}_{(x,y)\sim D'}\mathop{\mathbf{E}}_{h\sim Q} I(h(x)\neq y)\,. \tag{2.4}$$

Pour considérer les votants à valeur réelle, nous généralisons le risque de Gibbs comme suit.

Définition 2.6. Pour toute distribution de probabilité Q sur un ensemble de votants, le *risque de Gibbs* $R_{D'}(G_Q)$ est défini comme la perte linéaire espérée du classificateur G_Q, relativement à D'. On a donc

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

Il est bien connu dans la littérature PAC-bayésienne (Langford et Shawe-Taylor, 2003 ; McAllester, 2003b ; Germain, Lacasse, Laviolette et Marchand, 2009) que le risque de Bayes $R_{D'}(B_Q)$ est borné supérieurement par le double du risque de Gibbs $R_{D'}(G_Q)$. Ce fait s'étend également à notre définition plus générale du risque de Gibbs (la définition 2.6) prenant en considération les votants à valeur réelle.

Lemme 2.7. *Pour toute distribution de probabilité Q sur un ensemble de votants à valeur réelle $f : \mathcal{X} \to [-1, 1]$ et pour toute distribution D' sur $\mathcal{X}\times\{-1, 1\}$, nous avons*

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

Démonstration. Soit $(x, y) \in \mathcal{X}\times\{-1, 1\}$ un exemple quelconque. Démontrons d'abord la borne supérieure en ne considérant qu'un seul exemple, c'est-à-dire

$$R_{\langle(x,y)\rangle}(B_Q) \;\leq\; 2\, R_{\langle(x,y)\rangle}(G_Q)\;. \tag{2.5}$$

Notons que le risque $R_{\langle(x,y)\rangle}(B_Q)$ est soit 1, soit 0, dépendamment si B_Q se trompe sur l'exemple (x, y) ou non. Dans le cas où $R_{\langle(x,y)\rangle}(B_Q) = 0$, l'équation (2.5) est trivialement vraie. Si $R_{\langle(x,y)\rangle}(B_Q) = 1$, nous savons par la dernière égalité de la définition 2.5 que $\mathbf{E}_{f\sim Q}\, y \cdot f(x) \leq 0$. Alors, la définition 2.6 donne

$$2 \cdot R_{\langle(x,y)\rangle}(G_Q) \;=\; 2 \cdot \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{f\sim Q} y \cdot f(x)\right) \;\geq\; 1 = R_{\langle(x,y)\rangle}(B_Q)\;,$$

ce qui termine la première partie de cette démonstration.

Maintenant, en prenant l'espérance selon D' de chaque côté de l'équation (2.5), nous obtenons

$$R_{D'}(B_Q) \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D'} R_{\langle(x,y)\rangle}(B_Q) \;\leq\; \mathop{\mathbf{E}}_{(x,y)\sim D'} 2\, R_{\langle(x,y)\rangle}(G_Q) \;= 2\, R_{D'}(G_Q)\,,$$

comme voulu. □

Les bornes PAC-bayésiennes sur le vote de majorité sont généralement des bornes sur le risque de Gibbs, multiplié par un facteur deux. Même si une telle borne peut être serrée dans certaines situations, le facteur deux peut avoir un effet néfaste. Langford et Shawe-Taylor (2003) ont montré que sous certaines circonstances, ce facteur deux peut être réduit à un facteur de $(1 + \epsilon)$. Ceci dit, des distributions Q sur des votants donnant $R_{D'}(G_Q) \gg R_{D'}(B_Q)$ sont très communes. Le cas extrême arrive lorsque la perte linéaire sur chaque exemple est très près de $\frac{1}{2}$ (c'est-à-dire, pour chaque $(x, y) \sim D'$, $\mathbf{E}_{f\sim Q}\; y \cdot f(x) = \epsilon$). Dans une telle situation, le classificateur par vote de majorité ne fait aucune erreur car il a une marge positive sur chaque exemple ($R_{D'}(B_Q) = 0$), mais le classificateur de Gibbs associé se trompe près d'une fois sur deux ($R_{D'}(G_Q) = \frac{1}{2} - \frac{1}{2}\epsilon$). Dans une telle circonstance, la borne donnée par le lemme 2.7 est de $R_{D'}(B_Q) \leq 1-\epsilon$ et ne permet pas de bien représenter la performance du vote de majorité. Le problème vient du fait que le risque de Gibbs considère seulement

la sortie espérée d'une population de votants. En effet, le lemme 2.7 indique que le vote de majorité devrait être mauvais si la performance de chaque votant individuel est faible. Cette borne ne peut pas considérer le fait que la « communauté » de votants peut compenser pour l'erreur individuelle. En effet, si tous les votants pensent exactement la même chose, le risque du classificateur de Gibbs et le risque du classificateur par vote de majorité coïncident. Donc, si on veut un « effet de communauté », il faut qu'il y ait une certaine variété d'opinions dans le groupe de votants, un certain désaccord en quelque sorte. Cette idée nous amène à définir le concept de *probabilité de désaccord*. Nous verrons à la prochaine section qu'en tenant compte de cette notion, nous pouvons obtenir de meilleures garanties que celles données par le lemme 2.7.

Nous pouvons comparer la sortie de votants binaires en considérant la probabilité de désaccord entre eux :

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

où $D'_{\mathcal{X}}$ désigne la marginale en $\mathcal{X}$ de la distribution D'. La définition 2.8 étend cette notion de désaccord aux votants à valeur réelle.

Définition 2.8. Pour toute distribution Q sur un ensemble de votants, le *désaccord espéré* $d^{D'}_{Q}$ relatif à D' est défini comme

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

Remarque 2.9. La valeur de $d^{D'}_{Q}$ ne dépend pas des étiquettes y des exemples $(x,y)\sim D'$. Nous pouvons donc estimer le désaccord espéré en utilisant des données non étiquetées.

Notons que l'estimation du désaccord espéré en utilisant des données non-étiquetées est une idée qui a été exploitée dans la littérature. Notamment, Madani, Pennock et Flake (2005) l'utilisent pour borner le risque et la variance des erreurs de classificateurs entraînés sur des ensembles d'entraînement différents, dans le contexte de validation d'algorithmes et de sélection de modèle.

Dans la prochaine section, nous définissons une borne sur le risque du classificateur par vote de majorité qui dépend non seulement de la performance moyenne des votants, mais également de leur désaccord espéré.

2.2 La $\mathcal{C}$-borne: une borne sur le risque du classificateur par vote de majorité

Dans cette section, nous introduisons la $\mathcal{C}$-borne, une borne supérieure sur le risque du classificateur par vote de majorité (définition 2.5), qui est basée sur deux quantités : le risque de Gibbs (définition 2.6) et le désaccord espéré (définition 2.8). Nous commençons par étudier la marge d'un vote de majorité comme une variable aléatoire. À partir du premier moment de la marge, nous retrouvons facilement la borne donnée par deux fois le risque de Gibbs présentée au lemme 2.7. Nous suggérons donc d'étendre cette analyse au deuxième moment de la marge, pour obtenir la $\mathcal{C}$-borne. Finalement, nous analysons les propriétés statistiques de la $\mathcal{C}$-borne et faisons une étude empirique de son pouvoir de prédiction.

2.2.1 La marge du vote de majorité et ses moments

Les bornes sur le risque du vote de majorité proposées dans cette section sont le résultat de l'étude de la marge pondérée du vote de majorité en tant que variable aléatoire. Comme nous le verrons, cette notion est intimement liée aux notions de risque de Gibbs et de désaccord, vues à la section précédente.

Définition 2.10. Soit $M_Q^{D'}$ la variable aléatoire qui, étant donné un exemple (x, y) tiré selon D', retourne la *marge* du vote de majorité B_Q sur cet exemple, définie par

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

À partir des définitions 2.5 et 2.10, nous obtenons la propriété intéressante suivante : [1]

$$R_{D'}\left(B_Q\right) = \Pr_{(x,y)\sim D'}\left(M_Q(x,y) \leq 0\right). \tag{2.6}$$

La marge n'est pas seulement reliée au risque du classificateur par vote de majorité, mais également au risque de Gibbs. Considérons le premier moment $\mu_1\left(M_Q^{D'}\right)$ de la variable aléatoire $M_Q^{D'}$, défini par

$$\mu_1\left(M_Q^{D'}\right) \stackrel{\triangle}{=} \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\,. \tag{2.7}$$

1. Notons qu'avec un autre choix de définition pour la perte zéro-un (définition 2.3), l'égalité dans le vote de majorité (c'est-à-dire, lorsque $M_Q(x,y) = 0$) aurait été plus compliquée à gérer. Notamment, l'égalité aurait dû être relaxée à

$$\Pr_{(x,y)\sim D'}\left(M_Q(x,y) < 0\right) \leq R_{D'}\left(B_Q\right) \leq \Pr_{(x,y)\sim D'}\left(M_Q(x,y) \leq 0\right).$$

Il est important de noter que nous pouvons maintenant réécrire le risque de Gibbs (définition 2.6) comme une fonction de $\mu_1\left(M_Q^{D'}\right)$, comme

$$\begin{aligned} R_{D'}(G_Q) &= \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}^{\mathcal{L}_\ell}_{D'}(f) = \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{(x,y)\sim D'} \mathop{\mathbf{E}}_{f\sim Q} y\cdot f(x)\right) \\ &= \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\right) \\ &= \frac{1}{2}\left(1 - \mu_1\left(M_Q^{D'}\right)\right). \end{aligned} \tag{2.8}$$

De manière similaire, nous pouvons réécrire le désaccord espéré en fonction du second moment de la marge. Nous utilisons $\mu_2\left(M_Q^{D'}\right)$ pour désigner le second moment. Comme en classification binaire, $y \in \{-1, 1\}$, et donc $y^2 = 1$, le second moment de la marge ne dépend pas des étiquettes. En effet, nous avons

$$\begin{aligned} \mu_2\left(M_Q^{D'}\right) &\triangleq \mathop{\mathbf{E}}_{(x,y)\sim D'} \left[M_Q(x,y)\right]^2 \\ &= \mathop{\mathbf{E}}_{(x,y)\sim D'} y^2 \cdot \left[\mathop{\mathbf{E}}_{f\sim Q} f(x)\right]^2 \\ &= \mathop{\mathbf{E}}_{x\sim D'_X} \left[\mathop{\mathbf{E}}_{f\sim Q} f(x)\right]^2. \end{aligned} \tag{2.9}$$

Ainsi, à partir de la dernière inégalité et de la définition 2.8, le désaccord espéré peut être exprimé par

$$\begin{aligned} d_Q^{D'} &= \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{x\sim D'_X} \left[\mathop{\mathbf{E}}_{f\sim Q} f(x)\right]^2\right) \\ &= \frac{1}{2}\left(1 - \mu_2\left(M_Q^{D'}\right)\right). \end{aligned} \tag{2.10}$$

L'équation (2.10) montre que $0 \leq d_Q^{D'} \leq 1/2$, comme $0 \leq \mu_2\left(M_Q^{D'}\right) \leq 1$. Par contre, nous pouvons borner supérieurement le désaccord de manière plus serrée. Pour ce faire, définissons d'abord la variance de la marge comme suit :

$$\begin{aligned} \mathrm{Var}\left(M_Q^{D'}\right) &\triangleq \mathop{\mathbf{Var}}_{(x,y)\sim D'} \left(M_Q(x,y)\right) \\ &= \mu_2\left(M_Q^{D'}\right) - \left(\mu_1\left(M_Q^{D'}\right)\right)^2. \end{aligned} \tag{2.11}$$

Comme la variance ne peut pas être négative, il s'ensuit que

$$\mu_2\left(M_Q^{D'}\right) \geq \left(\mu_1\left(M_Q^{D'}\right)\right)^2, \tag{2.12}$$

ce qui implique que

$$1 - 2\cdot d_Q^{D'} \geq \left(1 - 2\cdot R_{D'}(G_Q)\right)^2. \tag{2.13}$$

Après quelques calculs simples, nous obtenons la borne de $d_Q^{D'}$ désirée, basée sur le risque de Gibbs :

$$d_Q^{D'} \;\leq\; 2 \cdot R_{D'}(G_Q) \cdot \big(1 - R_{D'}(G_Q)\big)\,. \tag{2.14}$$

Par tout ce qui précède, nous obtenons donc la proposition suivante.

Proposition 2.11. *Pour toute distribution Q sur un ensemble de votants, et toute distribution D' sur $\mathcal{X}\times\{-1,1\}$, nous avons*

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

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

Démonstration. L'équation (2.14) donne la première inégalité. Le reste de la proposition suit directement du fait que $f(x) = 2x(1-x)$ est une parabole dont le maximum unique est au point $\left(\frac{1}{2}, \frac{1}{2}\right)$. □

2.2.2 Redécouverte de la borne $R_{D'}(B_Q) \leq 2 \cdot R_{D'}(G_Q)$

Le facteur deux bien connu permettant de transformer une borne du risque de Gibbs $R_{D'}(G_Q)$ en une borne du risque $R_{D'}(B_Q)$ du vote de majorité est habituellement justifié par un argument similaire à celui donné au lemme 2.7. Par contre, tel que montré dans la preuve de la proposition 2.12 suivante, ce résultat peut également être obtenu en considérant que le risque du classificateur par vote de majorité est aussi la probabilité que la marge $M_Q^{D'}$ soit inférieure ou égale à zéro (équation (2.6)), et en appliquant simplement l'inégalité de Markov (lemme A.2), donnée en annexe A.

Proposition 2.12. *Pour toute distribution Q sur un ensemble de votants, et toute distribution D' sur $\mathcal{X}\times\{-1,1\}$, nous avons*

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

Démonstration. En commençant à partir de l'équation (2.6) et en utilisant l'inégalité de Markov (lemme A.2), nous avons

$$\begin{aligned}
R_{D'}(B_Q) &= \Pr_{(x,y)\sim D'}\left(M_Q(x,y) \leq 0\right) \\
&= \Pr_{(x,y)\sim D'}\left(1 - M_Q(x,y) \geq 1\right) \\
&\leq \mathop{\mathbf{E}}_{(x,y)\sim D'}\left(1 - M_Q(x,y)\right) && \text{(inégalité de Markov)} \\
&= 1 - \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y) \\
&= 1 - \mu_1\big(M_Q^{D'}\big) \\
&= 2 \cdot R_{D'}(G_Q)\,.
\end{aligned}$$

La dernière égalité est directement obtenue de l'équation (2.8). □

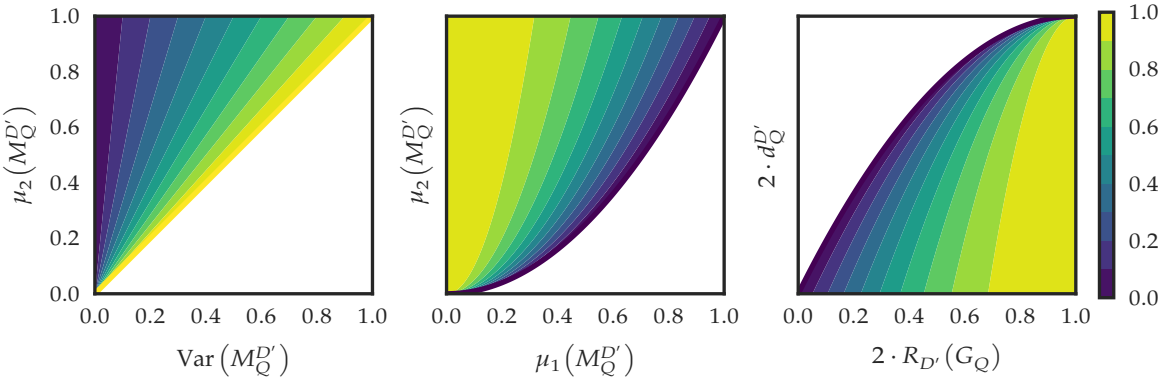

FIG. 2.2: Tracé du contour de chacune des trois formes de la C-borne, respectivement de gauche à droite. Le code de couleur donne la valeur de la C-borne, en fonction des deux variables qui la définissent. Notons que pour la troisième forme, nous affichons le graphique en multipliant les variables par deux, afin d'obtenir un graphique sur la même échelle que les deux autres formes.

Cette démonstration illustre que nous pouvons borner $R_{D'}(B_Q)$ supérieurement en considérant seulement le premier moment de la marge $\mu_1\left(M_Q^{D'}\right)$. Une fois ce fait réalisé, il devient naturel d'étendre ce résultat aux moments supérieurs. C'est ce que nous faisons dans la section 2.2.3, où nous utilisons l'inégalité de Cantelli (plutôt que l'inégalité de Markov), qui utilise non seulement le premier moment, mais aussi le second moment de la marge. Nous obtenons ainsi la C-borne du théorème 2.13.

2.2.3 La C-borne: une borne de $R_{D'}(B_Q)$

Nous présentons ici la borne sur laquelle sont basés la plupart des résultats de cette thèse. Nous appelons cette borne la C-borne. Nous l'exprimons sous trois formes différentes mais équivalentes. Deux d'entres elles ont été introduites dans LACASSE et al. (2007), et nous avons introduit celle dépendant des moments de la marge dans LAVIOLETTE, MARCHAND et ROY (2011). Chaque forme illustre une propriété différente de son comportement. La figure 2.2 illustre ces comportements.

Il est intéressant de constater que la preuve du théorème 2.13 ci-bas a le même point de départ que la preuve de la proposition 2.12, mais utilise l'inégalité de Cantelli (lemme A.5) plutôt que l'égalité de Markov (lemme A.2). Ainsi, le théorème 2.13 est basé sur la variance de la marge, en plus de la moyenne.

Théorème 2.13 (La C-borne de Lacasse et al. 2007)**.** *Pour toute distribution Q sur un ensemble de votants et pour toute distribution D' sur $\mathcal{X}\times\{-1,1\}$, si $\mu_1\big(M_Q^{D'}\big) > 0$ (ou de manière équivalente, si $R_{D'}\big(G_Q\big) < 1/2$), nous avons*

$$R_{D'}\big(B_Q\big) \;\leq\; C_Q^{D'}\,,$$

où

$$C_Q^{D'} \;\triangleq\; \underbrace{\frac{\mathrm{Var}\left(M_Q^{D'}\right)}{\mu_2\big(M_Q^{D'}\big)}}_{\textit{Première forme}} \;=\; \underbrace{1-\frac{\big(\mu_1\big(M_Q^{D'}\big)\big)^2}{\mu_2\big(M_Q^{D'}\big)}}_{\textit{Seconde forme}} \;=\; \underbrace{1-\frac{\big(1-2\cdot R_{D'}\big(G_Q\big)\big)^2}{1-2\cdot d_Q^{D'}}}_{\textit{Troisième forme}}\,.$$

Démonstration. En débutant par l'équation (2.6) et en utilisant l'inégalité de Cantelli (lemme A.5), avec $X = -M_Q(x,y)$, $\mu = \mathbf{E}_{(x,y)\sim D'}(-M_Q(x,y))$ et $a = \mathbf{E}_{(x,y)\sim D'} M_Q(x,y)$, nous obtenons

$$\begin{aligned}
R_{D'}\big(B_Q\big) \;&=\; \Pr_{(x,y)\sim D'}\Big(M_Q(x,y)\leq 0\Big)\\
&=\; \Pr_{(x,y)\sim D'}\left(-M_Q(x,y)+\mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\geq \mathop{\mathbf{E}}_{(x,y)\sim D'} M_Q(x,y)\right)\\
&\leq\; \frac{\mathbf{Var}_{(x,y)\sim D'}\left(M_Q(x,y)\right)}{\mathbf{Var}_{(x,y)\sim D'}\left(M_Q(x,y)\right)+\left(\mathbf{E}_{(x,y)\sim D'} M_Q(x,y)\right)^2} \qquad (2.15)\\
&=\; \frac{\mathrm{Var}\left(M_Q^{D'}\right)}{\mu_2\big(M_Q^{D'}\big)-\big(\mu_1\big(M_Q^{D'}\big)\big)^2+\big(\mu_1\big(M_Q^{D'}\big)\big)^2}\\
&=\; \frac{\mathrm{Var}\left(M_Q^{D'}\right)}{\mu_2\big(M_Q^{D'}\big)} \qquad (2.16)\\
&=\; \frac{\mu_2\big(M_Q^{D'}\big)-\big(\mu_1\big(M_Q^{D'}\big)\big)^2}{\mu_2\big(M_Q^{D'}\big)}\\
&=\; 1-\frac{\big(\mu_1\big(M_Q^{D'}\big)\big)^2}{\mu_2\big(M_Q^{D'}\big)} \qquad (2.17)\\
&=\; 1-\frac{\big(1-2\cdot R_{D'}\big(G_Q\big)\big)^2}{1-2\cdot d_Q^{D'}}\,. \qquad (2.18)
\end{aligned}$$

La ligne (2.15) correspond à l'application de l'inégalité de Cantelli. Les lignes (2.16) et (2.17) présentent respectivement la première et la seconde forme de $C_Q^{D'}$, et sont déduites des définitions de $\mu_1\big(M_Q^{D'}\big)$, $\mu_2\big(M_Q^{D'}\big)$, et $\mathrm{Var}\left(M_Q^{D'}\right)$ (voir les équations (2.7), (2.9) et (2.11)). La troisième forme de $C_Q^{D'}$ est obtenue à la ligne (2.18) en utilisant $\mu_1\big(M_Q^{D'}\big) = 1-2\cdot R_{D'}\big(G_Q\big)$ et $\mu_2\big(M_Q^{D'}\big) = 1-2\cdot d_Q^{D'}$, qui peuvent être directement dérivées des équations (2.8) et (2.10). □

La troisième forme de la C-borne montre que la borne décroît quand le risque de Gibbs $R_{D'}(G_Q)$ décroît, ou lorsque le désaccord $d_Q^{D'}$ augmente. Cette nouvelle borne suggère donc qu'un vote de majorité doit faire un compromis entre le risque de Gibbs et le désaccord, afin d'obtenir un risque de Bayes faible. Cette borne est donc plus informative que la borne habituelle de la proposition 2.12, qui ne met l'accent que sur la minimisation du risque de Gibbs.

La première forme de la C-borne met en évidence que sa valeur est toujours positive, comme la variance et le second moment de la marge sont positifs. La seconde forme met en évidence que la C-borne ne peut pas excéder la valeur un. Finalement, la borne est toujours définie (une division par zéro est impossible), car si $d_Q^{D'}$ était égal à $\frac{1}{2}$, alors $R_{D'}(G_Q)$ serait également égal à $\frac{1}{2}$ (proposition 2.11), ce qui est impossible par supposition.

2.3 Une analyse statistique du comportement de la C-borne

Cette section présente quelques propriétés de la C-borne. Nous discutons d'abord des conditions sous lesquelles la C-borne est optimale, dans le sens où si les seules informations connues d'un vote de majorité sont les deux premiers moments de la distribution des marges, il est possible que la valeur de la C-borne soit *exactement* le risque de Bayes (c'est-à-dire, $C_Q^{D'} = R_{D'}(B_Q)$). En d'autres mots, l'optimalité de la C-borne veut dire qu'il existe une variable aléatoire ayant les mêmes deux premiers moments que la distribution des marges, telle que l'inégalité de Cantelli du lemme A.5 est atteinte. En second lieu, nous montrons que la valeur de la C-borne peut être arbitrairement faible, particulièrement en présence de votants « non corrélés », même si le risque de Gibbs est grand. En d'autres mots, $C_Q^{D'} \ll R_{D'}(G_Q)$. Les résultats théoriques de cette section sont en majeure partie dus à nos coauteurs dans Germain, Lacasse, Laviolette, Marchand et Roy (2015), mais demeurent pertinents dans le cadre de cette thèse où la C-borne est centrale. Nous avons donc choisi de les inclure ici.

2.3.1 Conditions d'optimalité

Pour simplifier la notation, nous allons nous concentrer sur une variable aléatoire M représentant une distribution sur la marge, qui prend donc des valeurs dans $[-1, 1]$. Nous faisons abstraction des distributions sous-jacentes Q sur $\mathcal{H}$ ou $\mathcal{F}$, et D' sur $\mathcal{X}\times\{-1, 1\}$. Les deux premiers moments de M sont notés $\mu_1(M)$ et $\mu_2(M)$. Le classificateur par vote de majorité est dénoté B_M. Par l'équation (2.6), nous avons

$$R(B_M) \triangleq \Pr(M \leq 0). \tag{2.19}$$

De plus, si $\mu_1(M) > 0$, alors $R(B_M)$ est borné supérieurement par C_M, la C-borne donnée par la seconde forme du théorème 2.13,

$$C_M \triangleq 1 - \frac{(\mu_1(M))^2}{\mu_2(M)}. \tag{2.20}$$

La proposition suivante montre quand la C-borne peut être atteinte.

Proposition 2.14 (Optimalité de la C-borne)**.** *Soit M une variable aléatoire représentant la marge d'un vote de majorité. Si $\mu_1(M) > 0$, alors il existe une variable aléatoire $\widetilde{M}$ telle que*

$$\mu_1(\widetilde{M}) = \mu_1(M)\,, \quad \mu_2(\widetilde{M}) = \mu_2(M)\,, \quad \textit{et} \quad C_{\widetilde{M}} = C_M = R(B_{\widetilde{M}}) \tag{2.21}$$

si et seulement si

$$0 < \mu_2(M) \leq \mu_1(M)\,. \tag{2.22}$$

Démonstration. Premièrement, montrons que (2.22) implique (2.21).

Étant donné $0 < \mu_2(M) \leq \mu_1(M)$, considérons une distribution $\widetilde{M}$ concentrée en deux points, définie comme suit :

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

Cette distribution a les deux premiers moments requis, comme

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

Il s'ensuit directement de l'équation (2.20) que $C_{\widetilde{M}} = C_M$. De plus, par l'équation (2.19) et comme $\frac{\mu_2(M)}{\mu_1(M)} > 0$, nous obtenons comme désiré

$$R(B_{\widetilde{M}}) \;=\; \Pr(\widetilde{M} \leq 0) \;=\; C_M\,.$$

Maintenant, montrons que (2.21) implique (2.22). Considérons une distribution $\widetilde{M}$ telle que les égalités de la ligne (2.21) sont respectées. D'abord, comme $\mu_1(M) > 0$ par supposition, l'équation (2.12) implique que $\mu_2(M) > 0$, tel que voulu. La seconde inégalité de l'équation (2.22) est obtenue d'abord en utilisant la proposition 2.12 et l'équation (2.8). Nous obtenons

$$C_M \;=\; R(B_{\widetilde{M}}) \;\leq\; 1 - \mu_1(\widetilde{M}) \;=\; 1 - \mu_1(M)\,.$$

Par la définition de C_M et comme $\mu_1(M) > 0$ et $\mu_2(M) > 0$, nous avons

$$\begin{aligned} & 1 - \frac{(\,\mu_1(M)\,)^2}{\mu_2(M)} \;\leq\; 1 - \mu_1(M) \\ \Leftrightarrow \quad & \frac{(\,\mu_1(M)\,)^2}{\mu_2(M)} \;\geq\; \mu_1(M) \\ \Leftrightarrow \quad & (\,\mu_1(M)\,)^2 \;\geq\; \mu_1(M)\mu_2(M) \\ \Leftrightarrow \quad & \mu_1(M) \;\geq\; \mu_2(M)\,, \end{aligned}$$

comme voulu. □

Nous avons discuté dans la section 2.2.1 des multiples connexions entre les moments de la marge, le risque de Gibbs et le désaccord espéré d'un vote de majorité. La prochaine proposition exploite ces connexions pour dériver des expressions équivalentes à la ligne (2.22) de la proposition 2.14. Nous obtenons donc trois conditions nécessaires équivalentes sous lesquelles la C-borne est optimale.

Proposition 2.15. *Pour toute distribution Q sur un ensemble de votants et pour toute distribution D' sur $\mathcal{X}\times\{-1,1\}$, si $\mu_1\left(M_Q^{D'}\right) > 0$, ou de manière équivalente si $R_{D'}(G_Q) < 1/2$, alors les trois déclarations suivantes sont équivalentes :*

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

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

(iii) $C_Q^{D'} \leq 2\,R_{D'}(G_Q)$.

Démonstration. La véracité de $(i) \Leftrightarrow (ii)$ est une conséquence directe des équations (2.8) et (2.10). Pour démontrer $(ii) \Leftrightarrow (iii)$, exprimons $C_Q^{D'}$ dans sa troisième forme. Des calculs directs et la proposition 2.11 indiquant que $d_Q^{D'} \leq \frac{1}{2}$ nous permettent d'obtenir

$$
\begin{aligned}
& C_Q^{D'} = 1 - \frac{\left(1-2\,R_{D'}(G_Q)\right)^2}{1-2\,d_Q^{D'}} \leq 2R_{D'}(G_Q) \\
\Leftrightarrow \quad & \frac{\left(1-2\,R_{D'}(G_Q)\right)^2}{1-2\,d_Q^{D'}} \geq 1-2R_{D'}(G_Q) \\
\Leftrightarrow \quad & \left(1-2\,R_{D'}(G_Q)\right)^2 \geq \left(1-2R_{D'}(G_Q)\right)\left(1-2\,d_Q^{D'}\right) \\
\Leftrightarrow \quad & 1-2\,R_{D'}(G_Q) \geq 1-2\,d_Q^{D'} \\
\Leftrightarrow \quad & R_{D'}(G_Q) \leq d_Q^{D'} .
\end{aligned}
$$

□

Les propositions 2.14 et 2.15 illustrent un résultat intéressant : la C-borne est optimale si et seulement si sa valeur est plus petite que ou égale à deux fois le risque de Gibbs, la borne classique sur le risque du classificateur par vote de majorité (voir la proposition 2.12).

2.3.2 La C-borne peut être arbitrairement petite, même pour de grands risques de Gibbs.

Le prochain résultat montre que lorsque le nombre de votants tend vers l'infini (et que le poids de chaque votant tend vers zéro), la variance de M_Q tendra vers zéro, pourvu que la

moyenne de la covariance des sorties de chaque paire de votants distincts soit plus petite ou égale à 0. En particulier, la variance tendra toujours vers zéro si le risque des votants est non corrélé deux à deux. Pour quantifier la corrélation entre les votants, nous utiliserons le concept de covariance d'une paire de votants (f_1, f_2):

$$\begin{aligned}\mathrm{Cov}^{D'}(f_1, f_2) &\triangleq \mathop{\mathbf{Cov}}_{(x,y)\sim D'}\left(y \cdot f_1(x),\, y \cdot f_2(x)\right) \\ &= \mathop{\mathbf{E}}_{(x,y)\sim D'} f_1(x) f_2(x) - \left(\mathop{\mathbf{E}}_{(x,y)\sim D'} f_1(x)\right)\left(\mathop{\mathbf{E}}_{(x,y)\sim D'} f_2(x)\right).\end{aligned}$$

Notons que la covariance $\mathrm{Cov}^{D'}(f_1, f_2)$ est zéro lorsque f_1 et f_2 sont non corrélés.

Proposition 2.16. *Pour tout ensemble dénombrable de votants $\mathcal{F}$, toute distribution Q sur $\mathcal{F}$, et toute distribution D' sur $\mathcal{X}\times\{-1,1\}$, nous avons*

$$\mathrm{Var}\left(M_Q^{D'}\right) \leq \sum_{f\in\mathcal{F}} Q^2(f) + \sum_{f_1\in\mathcal{F}} \sum_{f_2\in\mathcal{F}\setminus\{f_1\}} Q(f_1)\, Q(f_2) \cdot \mathrm{Cov}^{D'}(f_1, f_2).$$

Démonstration. Par la définition de la marge (définition 2.10), nous réécrivons $M_Q(x, y)$ comme une somme de variables aléatoires :

$$\begin{aligned}&\mathop{\mathbf{Var}}_{(x,y)\sim D'}\left(M_Q(x,y)\right) \\ &\quad= \mathop{\mathbf{Var}}_{(x,y)\sim D'}\left(\sum_{f\in\mathcal{F}} Q(f)\cdot y\cdot f(x)\right) \\ &\quad= \sum_{f\in\mathcal{F}} Q^2(f) \mathop{\mathbf{Var}}_{(x,y)\sim D'}\left(y\cdot f(x)\right) + \sum_{f_1\in\mathcal{F}}\sum_{f_2\in\mathcal{F}\setminus\{f_1\}} Q(f_1)\, Q(f_2) \mathop{\mathbf{Cov}}_{(x,y)\sim D'}\left(y\cdot f_1(x), y\cdot f_2(x)\right).\end{aligned}$$

L'inégalité est une conséquence du fait que $\forall f \in \mathcal{F} : \mathbf{Var}_{(x,y)\sim D'}\left(y\cdot f(x)\right) \leq 1$. □

L'observation clé résultante est que $\sum_{f\in\mathcal{F}} Q^2(f)$ est généralement bien inférieure à 1. Considérons par exemple le cas où Q est uniforme sur $\mathcal{F}$, avec $|\mathcal{F}| = n$. Alors, $\sum_{f\in\mathcal{F}} Q^2(f) = 1/n$. De plus, si pour toute paire (f_1, f_2) de votants distincts de $\mathcal{F}$, $\mathrm{Cov}^{D'}(f_1, f_2) \leq 0$ alors $\mathrm{Var}\left(M_Q^{D'}\right) \leq 1/n$.

Dans un tel cas, $C_Q^{D'} \in O(1/n)$ tant que $1-2R_{D'}(G_Q)$ et $1-2d_Q^{D'}$ sont plus grands qu'une certaine constante positive indépendante de n. Alors, même quand $R_{D'}(G_Q)$ est grand, nous voyons que la C-borne peut s'approcher arbitrairement près de 0 quand nous augmentons le nombre de votants ayant une covariance paire à paire non positive. Plus précisément, nous avons

Corollaire 2.17. *Étant donné n votants non corrélés sous une distribution uniforme Q, si $R_{D'}(G_Q) < \frac{1}{2}$, alors nous avons*

$$R_{D'}(B_Q) \leq C_Q^{D'} \leq \frac{1}{n\cdot\left(1-2\,d_Q^{D'}\right)} \leq \frac{1}{n\cdot\left(1-2\,R_{D'}(G_Q)\right)^2}.$$

Démonstration. La première inégalité est la définition de la C-borne (théorème 2.13). La seconde inégalité est une conséquence de la proposition 2.16, considérant que dans le cas de la distribution uniforme sur des votants non corrélés, nous avons $\mathrm{Cov}^{D'}(f_1, f_2) = 0$, et alors $\mathrm{Var}\left(M_Q^{D'}\right) \leq 1/n$. En appliquant ceci à la première forme de la C-borne, on obtient

$$C_Q^{D'} = \frac{\mathrm{Var}\left(M_Q^{D'}\right)}{\mu_2\left(M_Q^{D'}\right)} = \frac{\mathrm{Var}\left(M_Q^{D'}\right)}{1-2\,d_Q^{D'}} \leq \frac{\frac{1}{n}}{1-2\,d_Q^{D'}} = \frac{1}{n\cdot\left(1-2\,d_Q^{D'}\right)}.$$

Finalement, la troisième inégalité du corollaire est une simple application de l'équation (2.13). □

2.4 Étude empirique du pouvoir de prédiction de la C-borne

Pour motiver davantage l'utilisation de la C-borne, nous examinons comment sa valeur empirique est reliée au risque du classificateur par vote de majorité en effectuant deux expérimentations. La première fait ressortir comment dans l'ensemble des classificateurs par votes de majorité étudiés, comment la C-borne surpasse la capacité individuelle des quantités du théorème 2.13 dans la tâche de prédire le risque du vote de majorité. La seconde expérimentation suggère que la C-borne est un excellent critère d'arrêt pour l'algorithme classique de la littérature AdaBoost (Freund et Schapire, 1997). Notons qu'une analyse empirique à cet endroit dans la thèse peut sembler non orthodoxe, mais nous croyons que les résultats ci-bas nous aideront à convaincre le lecteur que la création d'un algorithme d'apprentissage minimisant directement la C-borne est d'autant plus motivée.

2.4.1 Comparaison avec les autres indicateurs

Nous étudions comment $R_{D'}(G_Q)$, $\mathrm{Var}\left(M_Q^{D'}\right)$, $d_Q^{D'}$ et $C_Q^{D'}$ sont reliés à $R_{D'}(B_Q)$. Notons que ces quatre quantités apparaissent dans l'une ou l'autre des trois formes de la C-borne (théorème 2.13). Nous omettons une comparaison des deux premiers moments de la marge, comme il y a une relation linéaire entre $\mu_1\left(M_Q^{D'}\right)$ et $R_{D'}(G_Q)$, tout comme entre $\mu_2\left(M_Q^{D'}\right)$ et $d_Q^{D'}$.

Les résultats de la figure 2.3 sont obtenus en exécutant l'algorithme AdaBoost, avec comme votants des souches de décision, sur plusieurs ensembles de données de classification binaire provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (Lichman, 2013). Chaque ensemble de données est séparé en deux parties : un ensemble d'entraînement S et un ensemble de test T. L'ensemble S contient au moins la moitié des exemples et au plus 500, puis l'ensemble T contient le reste des exemples.[2] Nous exécutons AdaBoost sur l'ensemble S pendant 100 itérations, et calculons les quantités $R_T(G_Q)$, $\mathrm{Var}\left(M_Q^T\right)$, d_Q^T et C_Q^T sur l'ensemble de test T à chaque 5 itérations de boosting. Nous calculons donc ces quantités sur 20 votes de majorité différents pour chaque ensemble de données.

2. L'annexe B contient de l'information supplémentaire sur les ensembles de données utilisés dans cette thèse.

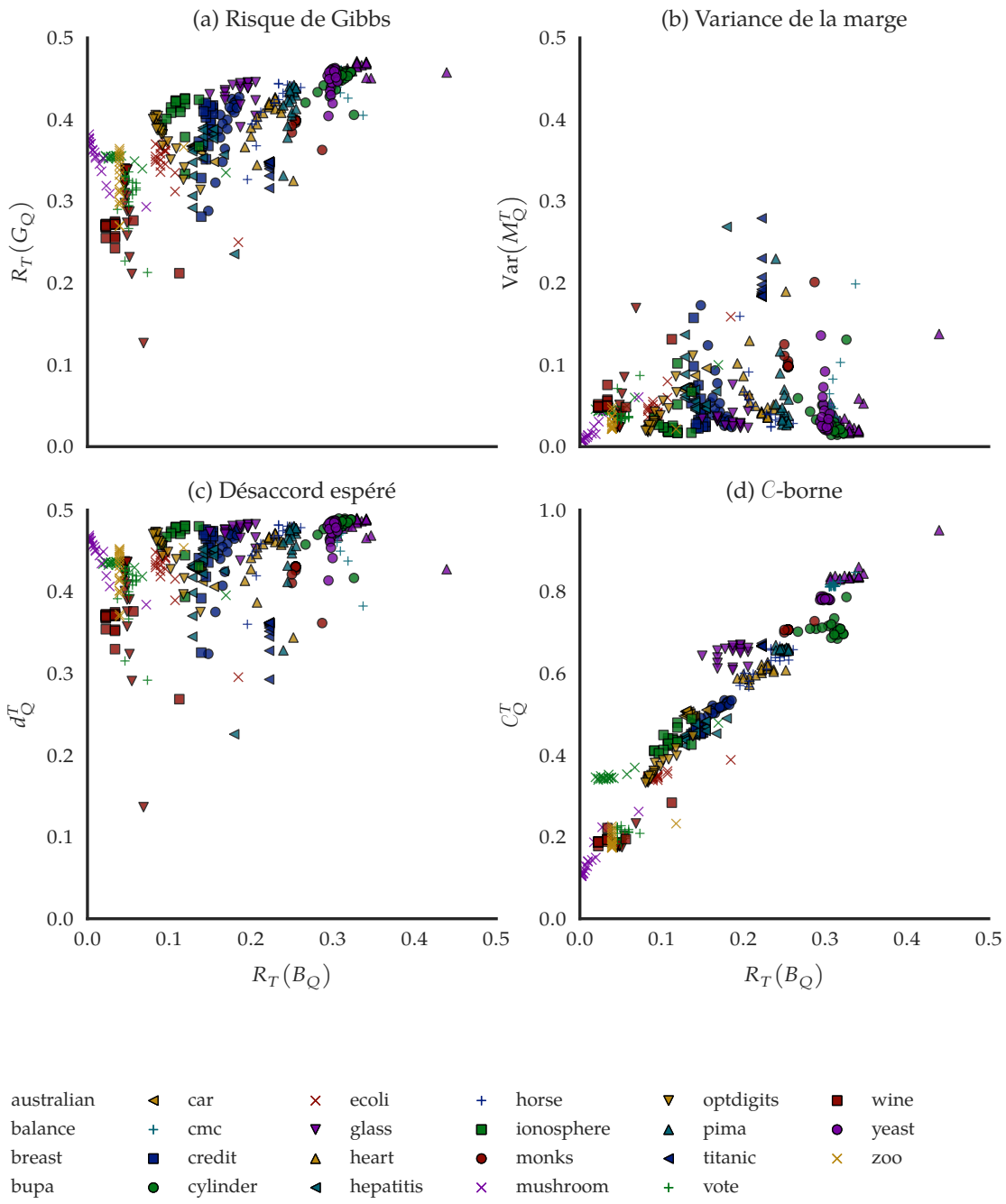

FIG. 2.3: Comparaison de $R_T(B_Q)$ par rapport à $R_T(G_Q)$, Var $\left(M_Q^T\right)$, d_Q^T et C_Q^T, respectivement. Pour chaque ensemble de données à qui nous avons associé un pictogramme différent, 20 classificateurs par vote de majorité différents sont considérés.

Dans la figure 2.3-(a), nous voyons que le risque de Bayes $R_T(B_Q)$ est presque toujours plus petit que le risque de Gibbs $R_T(G_Q)$. Il n'y a par contre pas de corrélation claire entre $R_T(B_Q)$ et $R_T(G_Q)$. Nous ne voyons également pas de corrélation claire entre $R_T(B_Q)$ et Var $\left(M_Q^T\right)$ ou entre $R_T(B_Q)$ et d_Q^T dans les figures 2.3-(b) et 2.3-(c) respectivement, sauf que généralement

Quantité	Corrélation de Pearson
Risque de Gibbs	0.7395
Variance de la marge	0.1973
Désaccord espéré	0.3626
C-borne	**0.9688**

TAB. 2.1: Corrélation de Pearson de chaque quantité de la figure 2.3 avec le risque du classificateur par vote de majorité. Toutes les valeurs sont calculées sur l'ensemble de test T.

$R_T(B_Q) > \text{Var}\left(M_Q^T\right)$ et $R_T(B_Q) < d_Q^T$. Par contre, la figure 2.3-(d) montre une forte corrélation entre C_Q^T et $R_T(B_Q)$. Il s'agit en effet d'une relation presque linéaire ! Nous fournissons également les valeurs de corrélation de Pearson (PEARSON, 1895) pour chacune de ces quantités, dans la table 2.1. On remarque que la C-borne semble être bien adaptée pour caractériser le comportement du risque de Bayes, alors que chaque quantité individuelle contenue dans la C-borne est insuffisante pour y arriver.

2.4.2 La C-borne comme critère d'arrêt pour AdaBoost

Nous évaluons maintenant la pertinence d'utiliser la valeur empirique de la C-borne en tant qu'outil de *sélection de modèle*, c'est-à-dire, comme un outil permettant de choisir de bonnes valeurs de paramètres pour un algorithme d'apprentissage. Plus spécifiquement, nous l'utilisons comme critère d'arrêt pour AdaBoost, et le comparons à d'autres critères d'arrêt.

Nous utilisons le même algorithme, les mêmes ensembles de données et les mêmes divisions entraînement/test qu'à l'expérimentation précédente. Nous exécutons AdaBoost sur l'ensemble S pendant 1000 itérations. À chaque itération, nous calculons la C-borne empirique C_Q^S (sur l'ensemble d'entraînement). Ensuite, nous choisissons le classificateur par vote de majorité (à une certaine itération) ayant la plus petite valeur de C_Q^S et calculons le risque de Bayes $R_T(B_Q)$ (sur l'ensemble de test). Nous comparons ce critère d'arrêt avec trois autres méthodes. Notons que pour toutes les méthodes de sélection de modèle, si plusieurs itérations différentes ont la même valeur pour la métrique utilisée (la même valeur de C-borne empirique, le même risque sur l'ensemble d'entraînement, etc.), le vote de majorité de l'itération la moins élevée est retenu.

La première méthode consiste à calculer le risque de Bayes empirique $R_S(B_Q)$ à chaque itération de boosting, pour ensuite choisir l'itération ayant le risque empirique le moins élevé. La seconde méthode consiste à appliquer de la validation croisée à 5 plis, pour choisir l'itération ayant le risque de validation croisée le plus faible. Finalement, la dernière méthode consiste à conserver 10% des exemples d'entraînement comme ensemble de validation, entraîner AdaBoost sur les 90% restants, et choisir l'itération ayant le risque minimal calculé sur l'ensemble de validation. Pour ces deux dernières méthodes, une fois le nombre d'itérations

Ensemble de données	Risque $R_T(B_Q)$ par critère d'arrêt *(et nombre d'itérations effectuées)*								
	C-borne C_Q^S		$R_S(B_Q)$		Ens. valid.		Valid. croisée		1000 itér.
australian	0.2087	*(25)*	0.2087	*(331)*	**0.1449**	*(1)*	0.2087	*(313)*	0.2464
balance	0.0351	*(43)*	0.0319	*(126)*	0.0383	*(56)*	**0.0288**	*(429)*	0.0319
breast	**0.0457**	*(72)*	0.0486	*(91)*	0.0857	*(1)*	0.0486	*(18)*	0.0571
bupa	0.3295	*(339)*	0.3295	*(160)*	0.3410	*(11)*	0.3410	*(27)*	**0.3121**
car	0.1376	*(14)*	**0.1303**	*(39)*	0.1547	*(3)*	0.1547	*(3)*	0.1336
cmc	0.3155	*(415)*	**0.3032**	*(8)*	0.3094	*(10)*	0.3104	*(20)*	0.3135
credit	0.1391	*(118)*	0.1478	*(127)*	**0.1333**	*(1)*	**0.1333**	*(1)*	0.1797
cylinder	0.3000	*(970)*	0.3000	*(981)*	0.3185	*(28)*	0.3185	*(64)*	**0.2963**
ecoli	**0.0833**	*(51)*	0.1012	*(296)*	0.0952	*(17)*	0.0952	*(28)*	0.1190
glass	**0.2056**	*(100)*	0.2150	*(198)*	0.2150	*(13)*	**0.2056**	*(11)*	0.2150
heart	0.2444	*(136)*	0.2519	*(141)*	0.2370	*(3)*	**0.1926**	*(16)*	0.2741
hepatitis	**0.1410**	*(62)*	0.1667	*(29)*	0.1795	*(1)*	0.1538	*(9)*	0.1667
horse	0.2391	*(185)*	0.2500	*(389)*	**0.2065**	*(1)*	0.2446	*(70)*	0.2500
ionosphere	0.1080	*(273)*	0.1193	*(54)*	0.3239	*(3)*	**0.0909**	*(37)*	0.1136
monks	0.2500	*(15)*	0.2454	*(8)*	**0.2315**	*(1)*	**0.2315**	*(1)*	0.2546
mushroom	0.0031	*(1000)*	0.0079	*(32)*	**0.0021**	*(129)*	**0.0021**	*(120)*	0.0031
optdigits	**0.0816**	*(734)*	0.0825	*(98)*	0.0954	*(40)*	0.0885	*(47)*	0.0831
pima	**0.2448**	*(282)*	0.2500	*(103)*	0.2552	*(1)*	0.2552	*(1)*	0.2552
titanic	**0.2222**	*(9)*	**0.2222**	*(1)*	**0.2222**	*(1)*	**0.2222**	*(1)*	**0.2222**
vote	0.0505	*(1)*	**0.0459**	*(30)*	0.0550	*(4)*	0.0734	*(7)*	0.0505
wine	0.0562	*(35)*	0.0899	*(7)*	**0.0337**	*(15)*	0.1124	*(5)*	0.0112
yeast	0.3059	*(852)*	0.3059	*(479)*	0.2937	*(12)*	**0.2907**	*(14)*	0.3079
zoo	0.1176	*(111)*	**0.0392**	*(9)*	0.0980	*(1)*	0.0980	*(1)*	0.1176

Comparaison statistique de l'utilisation de C_Q^S comme critère d'arrêt				
	vs $R_S(B_Q)$	vs Ens. valid.	vs Valid. croisée	vs 1000 itér.
Poisson binomial test	72%	58%	58%	81%
Sign test (valeur-p)	0.12	0.42	0.41	0.08

TAB. 2.2: Comparaison de plusieurs critères d'arrêt sur 1000 itérations de boosting de l'algorithme AdaBoost. Le «*Poisson binomial test*» donne la probabilité que C_Q^S soit un meilleur critère d'arrêt que chacune des autres approches. Le «*sign test*» donne un «*p-value*» représentant la probabilité que l'hypothèse nulle soit vraie, c'est-à-dire que d'utiliser C_Q^S comme critère d'arrêt a la même performance que chacune des autres approches.

choisi, AdaBoost est entraîné à nouveau sur la totalité de l'ensemble d'entraînement pour ce nombre d'itérations. Ces méthodes sont donc plus coûteuses en temps d'exécution.

La table 2.2 compare les risques de Bayes sur l'ensemble de test $R_T(B_Q)$ des votes de majorité sélectionnés par les différents critères d'arrêt. Nous calculons la probabilité que la C-borne

soit un meilleur critère d'arrêt que chacune des autres méthodes à l'aide de deux tests statistiques : le «*Poisson binomial test*» (Lacoste, Laviolette et Marchand, 2012)[3] et le «*sign test*» (Mendenhall, 1983). Ces deux tests suggèrent que l'utilisation de la $\mathcal{C}$-borne comme critère d'arrêt est au moins aussi performant que les méthodes de l'ensemble de validation et de la validation croisée, qui ont comme désavantage une augmentation du temps de calcul. Lorsque l'utilisation de la $\mathcal{C}$-borne est comparée deux à deux avec ces deux méthodes, celle-ci permet d'obtenir un risque meilleur ou égal 13 fois sur les 24 ensembles de données. Dans un second temps, nos expérimentations suggèrent que la sélection de modèle par la $\mathcal{C}$-borne est supérieure aux méthodes de la sélection par le meilleur risque empirique ou par un nombre fixe de 1000 itérations. En effet, bien que nos résultats ne soient pas statistiquement significatifs, nous observons une tendance plus forte avec un risque meilleur ou égal 18 fois sur 24 par rapport à ces deux méthodes. Nous pouvons donc conclure que la $\mathcal{C}$-borne empirique est un excellent critère d'arrêt pour AdaBoost.

2.5 Conclusion du chapitre

Nous avons présenté le concept de vote de majorité pondéré, un type de classificateur retourné par de nombreux algorithmes d'apprentissage automatique. Nous avons introduit le risque de ce classificateur, correspondant à la probabilité de faire une erreur sur un nouvel exemple à classifier. En utilisant la marge du vote de majorité, nous avons introduit deux bornes sur ce risque : une borne classique, n'utilisant que le premier moment de la marge, et la $\mathcal{C}$-borne, une borne prenant également en considération le second moment.

Nous avons étudié les propriétés statistiques de la $\mathcal{C}$-borne, avons évalué empiriquement son pouvoir de prédiction, et démontré que son utilisation est pertinente dans le contexte de sélection de modèle pour l'algorithme AdaBoost.

Jusqu'à maintenant, nous avons utilisé l'estimation empirique de la $\mathcal{C}$-borne, c'est-à-dire que nous avons évalué sa valeur sur un ensemble de données d'entraînement ou de test. Or, la quantité d'intérêt est sa « vraie » valeur, c'est-à-dire la valeur de la $\mathcal{C}$-borne sur la distribution génératrice des données D, qui est inconnue. Le prochain chapitre montrera comment obtenir des bornes de généralisation basées sur la $\mathcal{C}$-borne, dans le but d'obtenir de rigoureuses garanties de généralisation pour un algorithme d'apprentissage.

3. Le « Poisson binomial test » est un test statistique non paramétrique qui prend en considération les dépendances introduites lorsque plusieurs classificateurs sont évalués sur un même ensemble de test. Les expérimentations de Lacoste, Laviolette et Marchand (2012) suggèrent que cette approche est plus fiable que le « sign test ».

Chapitre 3

Théorie PAC-bayésienne unifiée et généralisée à la C-borne

Ce chapitre présente les travaux publiés dans Germain, Lacasse, Laviolette, Marchand et Roy (2015), qui ont été réalisés dans une perspective d'unification et de simplification de l'état de l'art (McAllester, 1999 ; Langford et Seeger, 2001 ; Seeger, 2002 ; Langford, 2005), de travaux de nos coauteurs (Lacasse et al., 2007 ; Germain, Lacasse, Laviolette et Marchand, 2009) et de nos propres travaux (Laviolette, Marchand et Roy, 2011). Notre contribution est à part égale avec nos coauteurs au niveau de l'unification des théories existantes, et notre contribution personnelle correspond à l'introduction de bornes PAC-bayésiennes ne dépendant pas de la divergence Kullback-Leibler (KL). Ces nouveaux résultats théoriques, présentés pour la première fois dans Laviolette, Marchand et Roy (2011), nous permettent de construire l'algorithme MinCq, présenté au chapitre 4. Par rapport à Laviolette, Marchand et Roy (2011), nous avons extrait une partie clé de la preuve du théorème PAC-bayésien : l'inégalité du changement de mesure. Cette extraction permet de grandement simplifier les démonstrations et les idées derrière les bornes PAC-bayésiennes sans KL. Notons également que la théorie PAC-bayésienne généralisée aux votants à valeur réelle a été introduite pour la première fois dans Laviolette, Marchand et Roy (2011).

Nous présentons une forme générale de la théorie PAC-bayésienne qui permet de borner la valeur de la C-borne C_Q^D en fonction de son estimation empirique C_Q^S. Nous dérivons ensuite des bornes sur le risque du vote de majorité $R_D(B_Q)$ basées sur les observations empiriques. Nous retrouvons premièrement la borne PAC-bayésienne classique (appelée ici la PAC-borne 3.8), qui borne le vrai risque de Gibbs en fonction de son homologue empirique. Nous présentons ensuite une borne PAC-bayésienne sur la vraie valeur de la C-borne en fonction de sa valeur empirique, la PAC-borne 3.13. Nous présentons la théorie nécessaire à la construction des PAC-bornes 3.23 et 3.30, la première ne dépendant pas de la divergence KL, et la seconde étant dans le cadre de la *compression d'échantillon*, nous permettant d'utiliser des

votants définis à partir des exemples de l'ensemble d'apprentissage.

Le premier théorème PAC-bayésien a été proposé par McAllester (1999). Étant donné un ensemble de votants $\mathcal{F}$, une distribution *a priori* P sur $\mathcal{F}$ choisie avant d'observer les données, et une distribution *a posteriori* Q sur $\mathcal{F}$ choisie après avoir observé un ensemble de données $S \sim D^m$ (Q est généralement choisi en exécutant un algorithme d'apprentissage sur S), les théorèmes PAC-bayésiens donnent des bornes serrées sur le risque du classificateur de Gibbs G_Q. Ces bornes sur $R_D(G_Q)$ dépendent généralement de deux quantités :

a) Le risque de Gibbs empirique $R_S(G_Q)$, qui est calculé sur les m exemples d'entraînement de S,
$$R_S(G_Q) \;=\; \frac{1}{m}\sum_{i=1}^{m} \mathop{\mathbf{E}}_{f\sim Q} \mathcal{L}_\ell(f(x_i), y_i)\,.$$

b) La *divergence Kullback-Leibler* entre les distributions Q et P, qui mesure à quel point la distribution Q choisie « diffère » de la distribution a priori P,
$$\mathrm{KL}(Q \,\|\, P) \;\triangleq\; \mathop{\mathbf{E}}_{f\sim Q} ln\, \frac{Q(f)}{P(f)}\,. \tag{3.1}$$

Notons que les bornes PAC-bayésiennes obtenues sont uniformément valides pour tout postérieur Q.

Convention 3.1. *Dans cette thèse, nous considérons que le support de Q est inclus dans le support de P, c'est-à-dire que si $P(h) = 0$, alors $Q(h) = 0$, bien que nous ne l'indiquons pas dans l'énoncé de chacun des théorèmes. Cette supposition permet permet à la définition de la divergence* $\mathrm{KL}(Q\,\|P)$ *d'être simplifiée, comme c'est le cas dans plusieurs articles de la littérature PAC-bayésienne. Notons qu'une définition plus générale permettrait de rendre les résultats de ce chapitre valides pour toutes distributions Q et P sans restriction. Nous omettons cette généralisation par soucis de simplicité.*

Ici, nous présentons un théorème PAC-bayésien très général (section 3.1), et le spécialisons pour obtenir une borne sur le risque de Gibbs $R_D(G_Q)$. Celle-ci est convertie en une borne sur le risque du classificateur par vote de majorité $R_D(B_Q)$ en utilisant le facteur 2 de la proposition 2.12 (section 3.2). Ensuite, nous définissons de nouvelles fonctions de perte qui dépendent d'une paire de votants (section 3.3). Ces nouvelles pertes nous permettent d'étendre la théorie PAC-bayésienne afin de borner directement $R_D(B_Q)$ par la C-borne (section 3.4).

3.1 Théorie PAC-bayésienne générale pour fonctions de perte à valeur réelle

Commençons par énoncer l'*inégalité du changement de mesure*, qui est l'une des étapes clés dans la plupart des démonstrations de théorèmes PAC-bayésiens. La preuve présentée ici est ins-

pirée de Seldin et Tishby (2010) et McAllester (2013), mais la même inégalité a été dérivée de l'inégalité de Fenchel dans Banerjee (2006), et de la formule variationnelle d'entropie relative de Donsker-Varadhan dans Seldin, Laviolette et al. (2012) et Tolstikhin et Seldin (2013).

Lemme 3.2 (Inégalité du changement de mesure)**.** *Pour tout ensemble mesurable $\mathcal{F}$, toute distribution P sur $\mathcal{F}$, toute distribution Q sur $\mathcal{F}$ dont le support est entièrement inclus dans P et toute fonction $\phi : \mathcal{F} \to \mathbb{R}$ mesurable sur P, nous avons*

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

Démonstration. Le résultat est obtenu par des calculs simples, en utilisant la définition de la divergence KL donnée par l'équation (3.1), et en utilisant l'inégalité de Jensen (lemme A.3) en annexe A) sur la fonction concave $ln(\cdot)$:

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

Notons que la dernière inégalité devient une égalité si Q et P partagent le même support. Nous présentons maintenant un théorème PAC-bayésien général bornant l'espérance de n'importe quelle fonction de perte à valeur réelle $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0,1]$. Ce théorème est légèrement plus général que le théorème PAC-bayésien général de Germain, Lacasse, Laviolette et Marchand (2009, théorème 2.1), qui est spécialisé à la perte linéaire espérée, et donne donc une borne sur le risque de Gibbs « généralisé » de la définition 2.6. Un résultat similaire est présenté dans Tolstikhin et Seldin (2013, Lemme 1).

Théorème 3.3 (Théorème PAC-bayésien général pour fonctions de perte à valeur réelle)**.** *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{F}$ 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 a priori P sur $\mathcal{F}$, pour tout $\delta \in (0,1]$ et pour toute fonction convexe $\Delta : [0,1]\times[0,1] \to \mathbb{R}$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

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

où $\mathrm{KL}(Q \,\|\, P)$ est la divergence Kullback-Leibler entre les distributions Q et P, définie à l'équation (3.1).

Démonstration. Nous présentons ici la preuve de ce théorème telle qu'elle a été publiée dans Germain, Lacasse, Laviolette, Marchand et Roy (2015). Commençons d'abord par noter que $\mathbf{E}_{f\sim P}\, e^{m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)}$ est une variable aléatoire non négative. Par l'inégalité de Markov (lemme A.2, en annexe A), nous avons avec probabilité d'au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que

$$\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)} \;\leq\; \frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)}.$$

En prenant le logarithme de chaque côté de l'inégalité, nous obtenons, toujours avec probabilité $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que

$$ln\left[\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)}\right] \leq ln\left[\frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)}\right].$$

Nous appliquons maintenant l'inégalité du changement de mesure (lemme 3.2) sur la partie gauche de l'inégalité, avec $\phi(f) = m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)$. Nous utilisons ensuite l'inégalité de Jensen (lemme A.3, en annexe A), en exploitant la convexité de Δ:

$$\begin{aligned}\forall\, Q \text{ sur } \mathcal{F}: \quad ln\left[\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)}\right] &\geq m\cdot\mathop{\mathbf{E}}_{f\sim Q}\Delta\left(\mathbb{E}_S^{\mathcal{L}}(f),\mathbb{E}_D^{\mathcal{L}}(f)\right) - \mathrm{KL}(Q\,\|\,P)\\ &\geq m\cdot\Delta\left(\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f),\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)\right) - \mathrm{KL}(Q\,\|\,P).\end{aligned}$$

Nous avons donc, avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,

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

Le résultat est ensuite obtenu par de simples calculs. □

Tel que montré dans Germain, Lacasse, Laviolette, Marchand et Roy (2015), le théorème PAC-bayésien général peut être utilisé pour retrouver plusieurs variantes communes de théorèmes PAC-bayésiens, simplement en choisissant une Δ-fonction appropriée. Parmi celles-ci, nous obtenons une borne similaire à celle proposée par Langford et Seeger (2001), Seeger (2002) et Langford (2005) en utilisant la divergence Kullback-Leibler entre deux distributions de Bernoulli de probabilité de succès p et q:

$$\Delta_{\mathrm{KL}}(q,p) \;\triangleq\; q\,ln\,\frac{q}{p} + (1-q)\,ln\,\frac{1-q}{1-p}. \tag{3.2}$$

Notons que $\Delta_{\mathrm{KL}}(q,p)$ est un raccourci de notation pour $\mathrm{KL}(Q\,\|\,P)$ de l'équation (3.1), avec $Q=(q,1-q)$ et $P=(p,1-p)$. Le corollaire A.8 (en annexe A) montre que $\Delta_{\mathrm{KL}}(q,p)$ est une fonction convexe.

Afin d'appliquer le théorème 3.3 avec $\Delta(q,p)=\Delta_{\mathrm{KL}}(q,p)$, nous avons besoin du lemme suivant.

Lemme 3.4. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout votant $f : \mathcal{X} \to \overline{\mathcal{Y}}$, pour toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}}\times\mathcal{Y} \to [0,1]$, et pour tout entier positif m, nous avons*

$$\mathop{\mathbf{E}}_{S\sim D^m} exp\left[m \cdot \Delta_{\mathrm{KL}}\Big(\mathbb{E}_S^{\mathcal{L}}(f),\, \mathbb{E}_D^{\mathcal{L}}(f)\Big)\right] \;\leq\; \xi(m)\,,$$

où

$$\xi(m) \;\triangleq\; \sum_{k=0}^{m}\binom{m}{k}\left(\frac{k}{m}\right)^k\left(1-\frac{k}{m}\right)^{m-k}. \tag{3.3}$$

De plus, $\sqrt{m} \leq \xi(m) \leq 2\sqrt{m}$.

Démonstration. Introduisons premièrement une variable aléatoire X_f suivant une distribution binomiale de m essais, avec une probabilité de succès $\mathbb{E}_D^{\mathcal{L}}(f)$. Alors, $X_f \sim B\left(m,\, \mathbb{E}_D^{\mathcal{L}}(f)\right)$.

Comme $e^{m\cdot\Delta_{\mathrm{KL}}\left(\cdot,\, \mathbb{E}_D^{\mathcal{L}}(f)\right)}$ est une fonction convexe, le lemme A.9 (dû à Maurer (2004) et fourni en annexe A), montre que

$$\mathop{\mathbf{E}}_{S\sim D^m} exp\left[m \cdot \Delta_{\mathrm{KL}}\Big(\mathbb{E}_S^{\mathcal{L}}(f),\, \mathbb{E}_D^{\mathcal{L}}(f)\Big)\right] \;\leq\; \mathop{\mathbf{E}}_{X_f\sim B\left(m,\mathbb{E}_D^{\mathcal{L}}(f)\right)} exp\left[m \cdot \Delta_{\mathrm{KL}}\Big(\tfrac{1}{m}X_f,\, \mathbb{E}_D^{\mathcal{L}}(f)\Big)\right].$$

Nous avons alors

$$\begin{aligned}
&\mathop{\mathbf{E}}_{X_f\sim B\left(m,\, \mathbb{E}_D^{\mathcal{L}}(f)\right)} e^{m\Delta_{\mathrm{KL}}\left(\frac{1}{m}X_f,\, \mathbb{E}_D^{\mathcal{L}}(f)\right)} \\
&\quad= \mathop{\mathbf{E}}_{X_f\sim B\left(m,\, \mathbb{E}_D^{\mathcal{L}}(f)\right)} \left(\frac{\frac{1}{m}X_f}{\mathbb{E}_D^{\mathcal{L}}(f)}\right)^{X_f}\left(\frac{1-\frac{1}{m}X_f}{1-\mathbb{E}_D^{\mathcal{L}}(f)}\right)^{m-X_f} \\
&\quad= \sum_{k=0}^{m}\Pr_{X_f\sim B\left(m,\, \mathbb{E}_D^{\mathcal{L}}(f)\right)}\Big(X_f = k\Big)\cdot\left(\frac{\frac{k}{m}}{\mathbb{E}_D^{\mathcal{L}}(f)}\right)^{k}\left(\frac{1-\frac{k}{m}}{1-\mathbb{E}_D^{\mathcal{L}}(f)}\right)^{m-k} \\
&\quad= \sum_{k=0}^{m}\binom{m}{k}\Big(\mathbb{E}_D^{\mathcal{L}}(f)\Big)^k\Big(1-\mathbb{E}_D^{\mathcal{L}}(f)\Big)^{m-k}\cdot\left(\frac{\frac{k}{m}}{\mathbb{E}_D^{\mathcal{L}}(f)}\right)^{k}\left(\frac{1-\frac{k}{m}}{1-\mathbb{E}_D^{\mathcal{L}}(f)}\right)^{m-k} \\
&\quad= \sum_{k=0}^{m}\binom{m}{k}\left(\frac{k}{m}\right)^k\left(1-\frac{k}{m}\right)^{m-k} \;=\; \xi(m)\,.
\end{aligned}$$

Maurer (2004) montre que $\xi(m) \leq 2\sqrt{m}$ pour $m \geq 8$, et $\xi(m) \geq \sqrt{m}$ pour $m \geq 2$. Le résultat est également facilement vérifiable manuellement pour $1 \leq m \leq 8$. □

Le théorème 3.5 ci-bas spécialise le théorème PAC-bayésien général au cas $\Delta(q,p) = \Delta_{\mathrm{KL}}(q,p)$, et reste applicable à toute fonction de perte à valeur réelle. Ce théorème peut être vu comme une étape intermédiaire à l'obtention du corollaire 3.6 de la prochaine section, qui utilise la perte linéaire afin de borner le risque de Gibbs. Par contre, le théorème 3.5 ci-bas est réutilisé dans la section 3.3 pour dériver des théorèmes PAC-bayésiens pour d'autres fonctions de perte.

Théorème 3.5. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{F}$ 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 a priori P sur $\mathcal{F}$ et pour tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

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

Démonstration. Le résultat est obtenu en démarrant du théorème 3.3 avec $\Delta(q,p) = \Delta_{\mathrm{KL}}(q,p)$. Le terme à droite de l'inégalité devient donc

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

Comme la distribution a priori P est indépendante de S, nous pouvons intervertir les deux espérances. Cette observation ainsi que le lemme 3.4, donnent

$$\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta_{\mathrm{KL}}\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)} \;=\; \mathop{\mathbf{E}}_{f\sim P}\mathop{\mathbf{E}}_{S\sim D^m} e^{m\cdot\Delta_{\mathrm{KL}}\left(\mathbb{E}_S^{\mathcal{L}}(f),\,\mathbb{E}_D^{\mathcal{L}}(f)\right)} \;\leq\; \mathop{\mathbf{E}}_{f\sim P}\xi(m) \;=\; \xi(m) \;\leq\; 2\sqrt{m}\,.$$

□

Notons que la borne du théorème 3.5 a été présentée dans Germain, Lacasse, Laviolette, Marchand et Roy (2015) sous une forme un peu plus serrée, en considérant un terme $\xi(m)$ plutôt que sa borne supérieure $2\sqrt{m}$. Cependant, comme dans cette thèse nous ne cherchons pas à obtenir les bornes les plus serrées possibles mais plutôt à justifier théoriquement nos nouveaux algorithmes d'apprentissage, nous choisissons de présenter cette borne ainsi que toutes celles qui en découlent en utilisant le terme $2\sqrt{m}$ pour plus de simplicité, d'autant plus que la perte est minime puisque $\sqrt{m} \leq \xi(m) \leq 2\sqrt{m}$ (Maurer, 2004).

3.2 Théorie PAC-bayésienne pour le risque de Gibbs

Cette section présente deux résultats PAC-bayésiens classiques qui bornent le risque de Gibbs. L'un de ceux-ci est utilisé pour exprimer une première borne sur le risque du classificateur par vote de majorité.

3.2.1 Théorèmes PAC-bayésiens pour le risque de Gibbs

Nous interprétons les deux résultats suivants comme de simples corollaires du théorème 3.5. En effet, à partir de la définition 2.6, la perte linéaire espérée du classificateur de Gibbs G_Q sur une distribution D' est $R_{D'}(G_Q)$. Ces deux corollaires sont très similaires à deux théorèmes PAC-bayésiens connus. Premièrement, le corollaire 3.6 est similaire au théorème PAC-bayésien de Langford et Seeger (2001), Seeger (2002) et Langford (2005), excepté que le terme

$m+1$ est remplacé par $2\sqrt{m}$. Comme $2\sqrt{m} \leq m+1$, notre résultat donne des bornes légèrement plus serrées. Similairement, le corollaire 3.7 fournit une légère amélioration de la borne PAC-bayésienne de McAllester (1999) et McAllester (2003a).

Corollaire 3.6. *(Langford et Seeger, 2001 ; Seeger, 2002 ; Langford, 2005) Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to [-1,1]$, pour toute distribution a priori P sur $\mathcal{F}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

$$\Delta_{\mathrm{KL}}\Big(R_S\left(G_Q\right), R_D\left(G_Q\right)\Big) \;\leq\; \frac{1}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta}\right].$$

Démonstration. Le résultat est directement obtenu du théorème 3.5 en utilisant la perte linéaire $\mathcal{L} = \mathcal{L}_\ell$ pour retrouver la définition du risque de Gibbs de la définition 2.6. □

Corollaire 3.7. *(McAllester, 1999 ; McAllester, 2003a) Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to [-1,1]$, pour toute distribution P sur $\mathcal{F}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

$$R_D\left(G_Q\right) \;\leq\; R_S\left(G_Q\right) + \sqrt{\frac{1}{2m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Démonstration. Le résultat est obtenu à partir du corollaire 3.6, en utilisant l'inégalité de Pinsker du lemme A.6, c'est-à-dire

$$2(q-p)^2 \;\leq\; \Delta_{\mathrm{KL}}(q,p)\,.$$

Nous avons alors avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,

$$2\cdot\Big(R_S\left(G_Q\right) - R_D\left(G_Q\right)\Big)^2 \;\leq\; \frac{1}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta}\right].$$

Le résultat est obtenu en isolant $R_D\big(G_Q\big)$ dans l'inégalité et en omettant la borne inférieure de $R_D\big(G_Q\big)$. Rappelons que la probabilité est « $\geq 1-\delta$ », alors si nous omettons un événement, la probabilité ne peut qu'augmenter (et continue donc d'être supérieure ou égale à $1-\delta$). □

3.2.2 Une première borne sur le risque du classificateur par vote de majorité

Nous énonçons ici une première borne PAC-bayésienne bornant le risque du classificateur par vote de majorité $R_D\big(B_Q\big)$, en utilisant d'une part le corollaire 3.7 et la borne supérieure du risque du classificateur par vote de majorité par deux fois le risque du classificateur de Gibbs (proposition 2.12). Notons que nous pouvons obtenir une borne légèrement plus serrée

en utilisant plutôt le corollaire 3.6 tel que nous l'avons fait dans Germain, Lacasse, Laviolette, Marchand et Roy (2015). Ceci dit, l'utilisation de ce corollaire rend plus complexe le calcul de la borne et nécessite l'introduction de notation supplémentaire. Nous présentons ici deux versions de cette borne : la première est en fonction du risque du classificateur de Gibbs et est plus commune dans la littérature PAC-bayésienne, alors que la seconde est définie en fonction du premier moment de la marge, qui lui est plus naturel dans le contexte de cette thèse.

PAC-borne 3.8. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to [-1,1]$, pour toute distribution a priori P sur $\mathcal{F}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

$$R_D(B_Q) \;\leq\; 2\cdot \bar{r} \;=\; 1 - \underline{\mu_1}\,,$$

où

$$\bar{r} \;\triangleq\; min\left(\frac{1}{2},\; R_S(G_Q) + \sqrt{\frac{1}{2m}\left[\mathrm{KL}(Q\,\|\,P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}\right),$$

$$\underline{\mu_1} \;\triangleq\; max\left(0,\; \mu_1\big(M_Q^S\big) - \sqrt{\frac{2}{m}\left[\mathrm{KL}(Q\,\|\,P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}\right).$$

Démonstration. Si $\bar{r} = \frac{1}{2}$ (ou respectivement $\underline{\mu_1} = 0$), la borne est trivialement valide, car $R_D(B_Q) \leq 1$. Sinon, la borne est une conséquence directe de la proposition 2.12 et du corollaire 3.7 dans le cas de l'inégalité, et l'égalité est une conséquence directe de l'équation (2.8). □

La figure 3.1 montre de manière plus visuelle l'application de cette borne. Celle-ci est une approche classique pour borner le risque du classificateur par vote de majorité, mais n'est pas serrée lorsque le risque de Gibbs est près de $\frac{1}{2}$ (lorsque le premier moment de la marge est près de 0). Tel qu'expliqué au chapitre 2, la C-borne est plus intéressante dans ces situations. La prochaine section propose une approche pour borner le « vrai » risque du classificateur par vote de majorité $R_D(B_Q)$ utilisant la C-borne empirique C_Q^S.

3.3 Erreur conjointe, succès conjoint et votants jumelés

Nous introduisons maintenant quelques notions nécessaires pour obtenir de nouveaux théorèmes PAC-bayésiens pour la C-borne dans la section 3.4.

3.3.1 L'erreur conjointe et le succès conjoint

Nous avons déjà défini le désaccord espéré $d_Q^{D'}$ d'une distribution Q de votants (définition 2.8). Dans le cas de votants binaires, le désaccord espéré correspond à

$$d_Q^{D'} \;=\; \mathop{\mathbf{E}}_{h_1\sim Q}\mathop{\mathbf{E}}_{h_2\sim Q}\Big(\mathop{\mathbf{E}}_{(x,y)\sim D'} I\big(h_1(x)\neq h_2(x)\big)\Big).$$

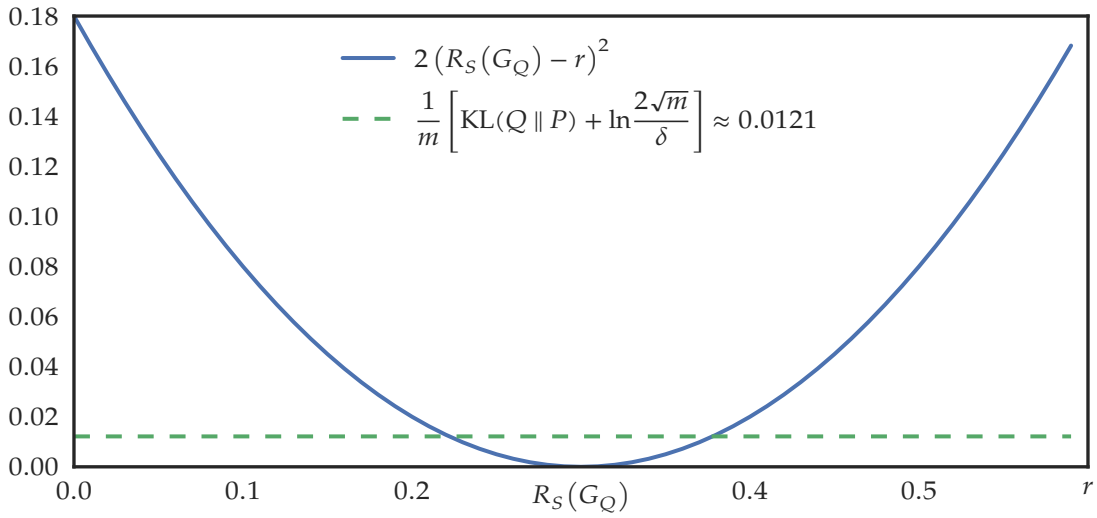

FIG. 3.1: Exemple d'application de la PAC-borne 3.8, où nous calculons également la borne inférieure associée, donnée dans la démonstration du corollaire 3.7. Nous supposons que $\mathrm{KL}(Q \,\|\, P) = 5$, $m = 1000$ et $\delta = 0.05$. Si nous observons un risque de Gibbs empirique $R_S(G_Q) = 0.30$, alors $R_D(G_Q) \in [0.222, 0.378]$ avec une confiance de 95%. Sur la figure, les intersections entre les deux courbes correspondent respectivement à la borne inférieure et supérieure. Avec ces valeurs, la PAC-borne 3.8 nous donne $R_D(B_Q) \lesssim 2 \cdot 0.378 = 0.756$.

Définissons maintenant deux notions reliées, soit l'erreur conjointe espérée $e_Q^{D'}$ et le succès conjoint espéré $s_Q^{D'}$. Dans le cas de votants binaires, ces deux concepts sont naturellement exprimés par

$$e_Q^{D'} = \mathop{\mathbf{E}}_{h_1 \sim Q} \mathop{\mathbf{E}}_{h_2 \sim Q} \left(\mathop{\mathbf{E}}_{(x,y)\sim D'} I\big(h_1(x) \neq y\big) I\big(h_2(x) \neq y\big) \right),$$
$$s_Q^{D'} = \mathop{\mathbf{E}}_{h_1 \sim Q} \mathop{\mathbf{E}}_{h_2 \sim Q} \left(\mathop{\mathbf{E}}_{(x,y)\sim D'} I\big(h_1(x) = y\big) I\big(h_2(x) = y\big) \right).$$

Étendons maintenant ces équations au cas des votants à valeur réelle.

Définition 3.9. Pour toute distribution de probabilité Q sur un ensemble de votants, nous définissons l'*erreur conjointe espérée* $e_Q^{D'}$ relative à D' et le *succès conjoint espéré* $s_Q^{D'}$ relatif à D' par

$$e_Q^{D'} \triangleq \mathop{\mathbf{E}}_{f_1 \sim Q} \mathop{\mathbf{E}}_{f_2 \sim Q} \left(\mathop{\mathbf{E}}_{(x,y)\sim D'} \mathcal{L}_\ell(f_1(x), y) \cdot \mathcal{L}_\ell(f_2(x), y) \right),$$
$$s_Q^{D'} \triangleq \mathop{\mathbf{E}}_{f_1 \sim Q} \mathop{\mathbf{E}}_{f_2 \sim Q} \left(\mathop{\mathbf{E}}_{(x,y)\sim D'} \big[1 - \mathcal{L}_\ell(f_1(x), y)\big] \cdot \big[1 - \mathcal{L}_\ell(f_2(x), y)\big] \right).$$

À partir des définitions de la perte linéaire (définition 2.4) et de la marge (définition 2.10),

nous voyons facilement que

$$e_Q^{D'} \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D'}\left(\frac{1-M_Q(x,y)}{2}\right)^2 \;=\; \frac{1}{4}\Big(1-2\cdot\mu_1\big(M_Q^{D'}\big)+\mu_2\big(M_Q^{D'}\big)\Big)\,,$$
$$s_Q^{D'} \;=\; \mathop{\mathbf{E}}_{(x,y)\sim D'}\left(\frac{1+M_Q(x,y)}{2}\right)^2 \;=\; \frac{1}{4}\Big(1+2\cdot\mu_1\big(M_Q^{D'}\big)+\mu_2\big(M_Q^{D'}\big)\Big)\,.$$

En considérant la définition du désaccord espéré de l'équation (2.10), c'est-à-dire que $d_Q^{D'} = \frac{1}{2}\left(1-\mu_2\big(M_Q^{D'}\big)\right)$, nous pouvons conclure que $e_Q^{D'}$, $s_Q^{D'}$ et $d_Q^{D'}$ somment toujours à 1 :[1]

$$e_Q^{D'} \;+\; s_Q^{D'} \;+\; d_Q^{D'} \;=\; 1\,.$$

Nous pouvons maintenant réécrire le premier moment de la marge et le risque de Gibbs comme

$$\begin{aligned}\mu_1\big(M_Q^{D'}\big) \;&=\; s_Q^{D'} - e_Q^{D'} &&=\; 1-\left(2e_Q^{D'}+d_Q^{D'}\right),\\ R_{D'}(G_Q) \;&=\; \tfrac{1}{2}\left(1-s_Q^{D'}+e_Q^{D'}\right) \;=\; \tfrac{1}{2}\left(2e_Q^{D'}+d_Q^{D'}\right).\end{aligned} \tag{3.4}$$

Alors, la troisième forme de la C-borne du théorème 2.13 peut être réécrite comme

$$C_Q^{D'} \;=\; 1-\frac{\left(1-\left(2e_Q^{D'}+d_Q^{D'}\right)\right)^2}{1-2d_Q^{D'}}\,. \tag{3.5}$$

3.3.2 Votants jumelés et leurs fonctions de perte

La prochaine généralisation du théorème PAC-bayésien nous permettra de borner séparément d_Q^D, e_Q^D ou s_Q^D, et donc de borner C_Q^D. Pour démontrer ce résultat, nous avons d'abord besoin d'un nouveau type de votants, que nous appelons les *votants jumelés*.

Définition 3.10. Étant donné deux votants $f_i : \mathcal{X} \to [-1,1]$ et $f_j : \mathcal{X} \to [-1,1]$, le *votant jumelé* $f_{ij} : \mathcal{X} \to [-1,1]^2$ retourne le couple[2]

$$f_{ij}(x) \;\triangleq\; \big\langle f_i(x), f_j(x)\big\rangle\,.$$

Étant donné un ensemble de votants $\mathcal{F}$ pondéré par une distribution Q sur $\mathcal{F}$, nous définissons un ensemble de votants jumelés $\mathcal{F}^2$ pondéré par une distribution Q^2 comme

$$\mathcal{F}^2 \;\triangleq\; \left\{f_{ij} \,:\, f_i, f_j \in \mathcal{F}\right\}, \quad \text{et} \quad Q^2(f_{ij}) \;\triangleq\; Q(f_i)\cdot Q(f_j)\,. \tag{3.6}$$

Notons que cette définition s'applique également aux ensembles $\mathcal{H}$ de votants binaires $h : \mathcal{X} \to \{-1,1\}$.

1. Ce fait est plutôt intuitif dans le cas des votants binaires, car étant donné un exemple (x,y) et deux votants binaires h_1 et h_2, nous avons que soit les deux votants classifient mal l'exemple ($h_1(x) = h_2(x) \neq y$), soit les deux votants le classifient bien ($h_1(x) = h_2(x) = y$), ou bien les deux votants sont en désaccord ($h_1(x) \neq h_2(x)$).

2. Notons que la notation utilisée ici peut porter à confusion, puisque $\langle\cdot,\cdot\rangle$ est également utilisé dans la littérature pour désigner un produit scalaire. Cependant, dans cette thèse nous utilisons cette notation pour représenter une séquence. La notion de votant jumelé est d'ailleurs généralisées à des séquences de votants de taille arbitraire en annexe A.2.

Nous présentons maintenant trois fonctions de perte pour les votants jumelés. Nous rappelons qu'une fonction de perte a la forme $\overline{\mathcal{Y}}\times\mathcal{Y} \to [0,1]$, où $\overline{\mathcal{Y}}$ est l'espace de sortie des votants. Comme un votant jumelé retourne un couple, ces nouvelles fonctions de perte font correspondre $[-1,1]^2 \times \{-1,1\}$ à $[0,1]$. Ainsi,

$$\begin{aligned}
\mathcal{L}_e(f_{ij}(x),\, y) &\triangleq \mathcal{L}_\ell(f_i(x), y) \cdot \mathcal{L}_\ell(f_j(x), y)\,, \\
\mathcal{L}_s(f_{ij}(x),\, y) &\triangleq \Big[1 - \mathcal{L}_\ell(f_i(x), y)\Big] \cdot \Big[1 - \mathcal{L}_\ell(f_j(x), y)\Big]\,, \\
\mathcal{L}_d(f_{ij}(x),\, y) &\triangleq \mathcal{L}_\ell(f_i(x) \cdot f_j(x)\,,\, 1)\,.
\end{aligned} \tag{3.7}$$

L'observation clé pour comprendre les prochains théorèmes est que les pertes espérées associées aux votants jumelés de l'ensemble $\mathcal{F}^2$, définis à l'équation (3.6), permettent de retrouver les valeurs de $e_Q^{D'}$, $s_Q^{D'}$ et $d_Q^{D'}$. En effet, il s'ensuit directement de l'équation (2.3), et des définitions 2.8 et 3.9, que

$$e_Q^{D'} = \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathbb{E}_{D'}^{\mathcal{L}_e}\big(f_{ij}\big)\,; \quad s_Q^{D'} = \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathbb{E}_{D'}^{\mathcal{L}_s}\big(f_{ij}\big)\,; \quad d_Q^{D'} = \mathop{\mathbf{E}}_{f_{ij}\sim Q^2} \mathbb{E}_{D'}^{\mathcal{L}_d}\big(f_{ij}\big)\,. \tag{3.8}$$

3.4 Théorie PAC-bayésienne pour fonctions de perte associées aux votants jumelés

Tel qu'expliqué à la section 3.2, les théorèmes PAC-bayésiens classiques tels que le corollaire 3.6 et 3.7 fournissent une borne supérieure de $R_D(G_Q)$ valide uniformément pour toute distribution a posteriori Q. Une borne de $R_D(B_Q)$ est typiquement obtenue en multipliant les bornes classiques par un facteur 2, comme dans la PAC-borne 3.8.

Dans cette section, nous présentons une première borne de $R_D(B_Q)$ s'appuyant sur la C-borne du théorème 2.13, présentée pour la première fois dans Lacasse et al. (2007) puis revue et généralisée dans Germain, Lacasse, Laviolette, Marchand et Roy (2015). Une borne uniforme sur C_Q^D est obtenue en utilisant la troisième forme de la C-borne, via une borne sur le risque de Gibbs $R_D(B_Q)$ et une borne sur le désaccord espéré d_Q^D. De manière équivalente, cette borne pourra être réécrite en fonction des deux premiers moments de la marge $\mu_1\big(M_Q^D\big)$ et $\mu_2\big(M_Q^D\big)$. Pour obtenir une borne sur le désaccord d_Q^D, nous utilisons la notion de votants jumelés présentée à la section précédente. Ceci nous permet d'exprimer une nouvelle borne PAC-bayésienne sur le risque du classificateur par vote de majorité.

3.4.1 Théorèmes PAC-bayésiens pour e_Q^D, s_Q^D ou d_Q^D.

Le théorème PAC-bayésien suivant permet de borner le désaccord espéré d_Q^D, le succès conjoint espéré s_Q^D ou l'erreur conjointe espérée e_Q^D d'un vote de majorité (voir les définitions 2.8 et 3.9).

Théorème 3.11. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to [-1,1]$, pour toute distribution a priori P sur $\mathcal{F}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

$$\Delta_{\mathrm{KL}}\left(\alpha_Q^S, \alpha_Q^D\right) \;\leq\; \frac{1}{m}\left[2{\cdot}\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta}\right],$$

où $\alpha_Q^{D'}$ est soit $e_Q^{D'}$, $s_Q^{D'}$ ou $d_Q^{D'}$.

Démonstration. Le théorème 3.11 est déduit du théorème 3.5. Nous présentons ici la preuve pour $\alpha_Q^{D'} = d_Q^{D'}$. Les deux autres cas sont très similaires.

Considérons l'ensemble de votants jumelés $\mathcal{F}^2$ et la distribution a posteriori Q^2 de l'équation (3.6). Considérons également la distribution a priori P^2 sur $\mathcal{F}^2$ définie par $P^2(f_{ij}) \triangleq P(f_i)\cdot P(f_j)$. Alors, nous avons

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

Finalement, de l'équation (3.8) nous obtenons $\mathbf{E}_{f_{ij}\sim Q^2}\mathbb{E}_D^{\mathcal{L}_d}\left(f_{ij}\right) = d_Q^D$ et $\mathbf{E}_{f_{ij}\sim Q^2}\mathbb{E}_S^{\mathcal{L}_d}\left(f_{ij}\right) = d_Q^S$. Le résultat est obtenu en appliquant le théorème 3.5, avec $\mathcal{L} = \mathcal{L}_d$, $\mathcal{F} = \mathcal{F}^2$, $P = P^2$ et $Q = Q^2$. □

Nous présentons maintenant un corollaire qui, tout comme le corollaire 3.7 dans la situation où nous bornons le risque du classificateur de Gibbs, donne une borne plus simple à interpréter que le théorème 3.11 et qui peut être calculée de manière analytique. Comme ultimement nous utiliserons ce corollaire pour borner la vraie valeur de la $\mathcal{C}$-borne, nous avons besoin cette fois-ci d'une borne inférieure de $\alpha_Q^{D'}$.

Corollaire 3.12. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to [-1,1]$, pour toute distribution P sur $\mathcal{F}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

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

où $\alpha_Q^{D'}$ est soit $e_Q^{D'}$, $s_Q^{D'}$ ou $d_Q^{D'}$.

Démonstration. Le résultat est obtenu à partir du théorème 3.11, en utilisant l'inégalité de Pinsker du lemme A.6. Nous avons alors, avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$, que pour toute distribution a posteriori Q sur $\mathcal{F}$,

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

Le résultat est obtenu en isolant α_Q^D dans l'inégalité. □

3.4.2 Une nouvelle borne pour le risque du classificateur par vote de majorité

Basé sur le fait que le corollaire 3.12 donne une borne inférieure sur le désaccord espéré d_Q^D, nous dérivons maintenant la PAC-borne 3.13, une borne PAC-bayésienne pour la $\mathcal{C}$-borne, et ainsi donc une borne sur le risque du classificateur par vote de majorité qui devrait être plus précise que la PAC-borne 3.8 selon notre analyse du chapitre 2. Nous donnons également une forme équivalente, dépendant des deux premiers moments de la marge $\mu_1\left(M_Q^S\right)$ et $\mu_2\left(M_Q^D\right)$.

PAC-borne 3.13. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X}\to[-1,1]$, pour toute distribution a priori P sur $\mathcal{F}$ et tout $\delta\in(0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

$$R_D\left(B_Q\right) \;\leq\; 1 - \frac{\left(1-2\cdot\overline{r}\right)^2}{1-2\cdot\underline{d}} \;=\; 1 - \frac{\left(\underline{\mu_1}\right)^2}{\overline{\mu_2}},$$

où

$$\begin{aligned}
\overline{r} &\triangleq min\left(\tfrac{1}{2},\, R_S(G_Q) + \sqrt{\frac{1}{2m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right),\\
\underline{d} &\triangleq max\left(0,\, d_Q^S - \sqrt{\frac{1}{2m}\left[2\cdot\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right),\\
\underline{\mu_1} &\triangleq max\left(0,\, \mu_1\left(M_Q^S\right) - \sqrt{\frac{2}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right),\\
\overline{\mu_2} &\triangleq min\left(1,\, \mu_2\left(M_Q^S\right) + \sqrt{\frac{2}{m}\left[2\cdot\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right).
\end{aligned}$$

Démonstration. Par la proposition 2.11, nous savons que $d_Q^S \leq \frac{1}{2}$ (et par conséquent, $\mu_2\left(M_Q^S\right) \geq 0$ par l'équation (2.10)). Ceci, ainsi que le fait que m est fini implique que $\underline{d} < \frac{1}{2}$ (et donc que $\overline{\mu_2} > 0$), et donc que le dénominateur de la fraction dans la PAC-borne 3.13 est toujours strictement positif.

Nécessairement, $\overline{r} \leq \frac{1}{2}$ (respectivement, $\underline{\mu_1} \geq 0$). Considérons les deux cas suivants.

Cas 1 : $\bar{r} = \frac{1}{2}$ (et $\underline{\mu_1} = 0$). Alors, la borne de $R_D\left(B_Q\right)$ est 1, ce qui est trivialement vrai.

Cas 2 : $\bar{r} < \frac{1}{2}$ (et $\underline{\mu_1} > 0$). Alors, nous pouvons appliquer le théorème 2.13 pour obtenir une borne supérieure de $R_D(B_Q)$. La borne désirée est obtenue en remplaçant d_Q^D par sa borne inférieure $\underline{d}$, et $R_D(G_Q)$ par sa borne supérieure $\bar{r}$, ou respectivement $\mu_2\left(M_Q^D\right)$ par sa borne supérieure $\overline{\mu_2}$ et $\mu_1\left(M_Q^D\right)$ par sa borne inférieure $\underline{\mu_1}$. Les deux bornes peuvent donc être déduites en appliquant convenablement le corollaire 3.7 (en remplaçant δ par $\delta/2$) et le corollaire 3.12 (en remplaçant α_Q^S par d_Q^S, α_Q^D par d_Q^D et δ par $\delta/2$). □

La PAC-borne 3.13 devrait être plus serrée que la PAC-borne 3.8 dans les situations où les votants sont faibles individuellement, comme elle prend en considération non seulement le risque du classificateur de Gibbs, mais également le désaccord. Par contre, elle a un inconvénient supplémentaire : elle dégrade rapidement si les bornes sur le numérateur et le dénominateur ne sont pas serrées. Notons par contre que dans le cadre de l'apprentissage transductif, nous pouvons obtenir des bornes plus serrées car les étiquettes des exemples n'affectent pas la valeur de $d_Q^{D'}$ (voir la définition 2.8). Dans ce contexte, nous avons accès à l'ensemble de toutes les entrées $x \in \mathcal{X}$, et pouvons donc calculer le « vrai désaccord » sans avoir à le borner inférieurement. La section 7.3.2 du chapitre 7 s'attaque à la théorie PAC-bayésienne adaptée à l'apprentissage transductif. Notons finalement que la PAC-borne 3.13 a été présentée sous une forme un peu plus serrée dans Germain, Lacasse, Laviolette, Marchand et Roy (2015), en fonction du risque de Gibbs et du désaccord espéré. La PAC-borne telle que nous la présentons dans cette thèse, en fonction des deux premiers moments de la marge, a été publiée pour la première fois dans Roy, Marchand et Laviolette (2016).

Évaluons maintenant les PAC-bornes 3.8 et 3.13 sur de vraies données afin d'étudier leur comportement.

3.4.3 Comparaison empirique des PAC-bornes 3.8 et 3.13

Nous proposons maintenant une comparaison empirique entre les deux PAC-bornes présentées jusqu'ici. Les résultats numériques de la figure 3.2 sont obtenus en utilisant AdaBoost (Freund et Schapire, 1997) avec des souches de décision comme votants, sur l'ensemble de données *mushroom* provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (Lichman, 2013). Cet ensemble contient 8124 exemples et est séparé en deux moitiés : un ensemble d'entraînement S et un ensemble de test T. Pour chaque itération de boosting nous fournissant une distribution Q, nous calculons la valeur des PAC-bornes 3.8 et 3.13 en utilisant une distribution P uniforme sur l'ensemble des votants.

Nous pouvons voir que la PAC-borne 3.13 est en général légèrement plus serrée que la PAC-borne 3.8. Par contre, nous voyons à la figure 3.2 qu'après 8 itérations de boosting, les deux bornes se dégradent. Au niveau de la PAC-borne 3.8, ce comportement est naturel puisque

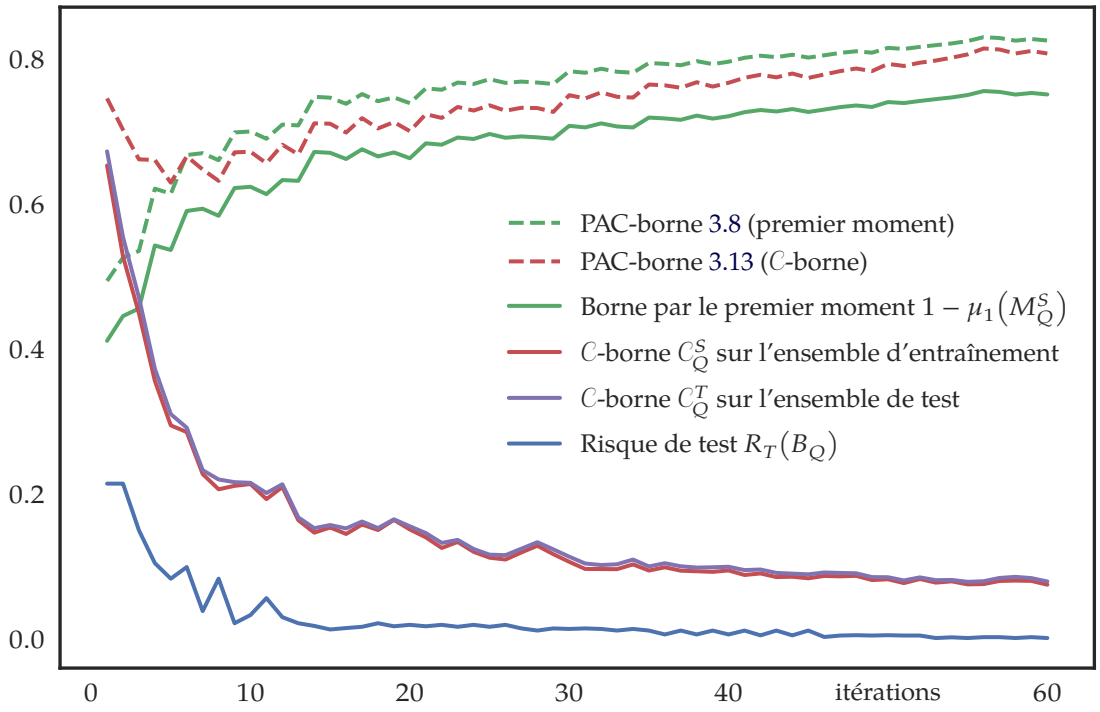

FIG. 3.2: Comparaison des bornes sur $R_D(B_Q)$ pendant 60 itérations de boosting.

l'estimation empirique de la quantité bornée, $1-\mu_1(M_Q^S)$, croît également. Tel que mentionné à la section 2.4, AdaBoost retourne des votes de majorité dont le risque du classificateur de Gibbs est très près de $1/2$, ou de manière équivalente, où le premier moment de la marge est très faible.

Le comportement de la PAC-borne 3.13 est par contre plus surprenant et décevant, puisque tout comme le risque sur l'ensemble de test, l'estimation empirique de la C-borne C_Q^S décroît avec le nombre d'itérations, alors que la PAC-borne associée croît. Ceci est dû au fait que le dénominateur de C_Q^S tend vers 0, c'est-à-dire que le second moment de la marge $\mu_2(M_Q^S)$ est près de 0 (voir la première ou la seconde forme du théorème 2.13). Dans ce contexte, comme le premier moment de la marge $\mu_1(M_Q^S)$ a une petite valeur également, tout relâchement dans la borne sur $\mu_1(M_Q^D)$ a un effet multiplicatif sur la PAC-borne 3.13. Notons également que l'algorithme AdaBoost n'a pas comme objectif de minimiser les bornes PAC-bayésiennes présentées dans cette thèse.

La PAC-borne 3.13 nécessite deux approximations pour borner supérieurement C_Q^D: l'une sur $R_D(G_Q)$ et l'autre sur d_Q^D. Dans GERMAIN, LACASSE, LAVIOLETTE, MARCHAND et ROY (2015), nous avons développé un nouveau théorème PAC-bayésien basé sur une variable aléatoire trivalente plutôt que bivalente (de Bernoulli). La borne résultante est légèrement plus serrée que la PAC-borne 3.13, mais présente exactement le même comportement sur des données réelles.

Dans le cadre de cette thèse, nous utilisons la théorie PAC-bayésienne justifier des algorithmes d'apprentissage, et ne sommes pas à la recherche de la borne la plus serrée possible. Comme la borne en question provient originellement de Lacasse et al. (2007) et est principalement due à nos coauteurs, nous ne l'introduirons pas ici.

Dans la prochaine section, nous présentons une nouvelle restriction sur la distribution Q qui permet d'obtenir une nouvelle famille de bornes PAC-bayésiennes ne dépendant pas de la divergence KL. Nous verrons que ces bornes nous permettront de faciliter le développement d'un algorithme d'apprentissage minimisant une borne PAC-bayésienne basée sur la C-borne.

3.5 Bornes PAC-bayésiennes sans régularisation KL

Avoir des théorèmes PAC-bayésiens qui bornent la différence entre C_Q^S et C_Q^D ouvre la porte aux algorithmes de minimisation structurelle de la C-borne. Comme la plupart des résultats en théorie PAC-bayésienne, la borne sur C_Q^D dépend de son estimation empirique, ainsi que la divergence Kullback-Leibler entre la distribution a posteriori Q et la distribution définie a priori P. Dans cette section, nous présentons une extension théorique de notre approche PAC-bayésienne, qui est nécessaire pour le développement de l'algorithme de minimisation de C_Q^D, présenté au chapitre 4.

Les prochains théorèmes introduisent des bornes PAC-bayésiennes qui ont la propriété surprenante de ne pas avoir de terme KL. Cette nouvelle approche a été développée étant donné que nos essais de construire un algorithme minimisant les PAC-bornes présentées dans les sections précédentes n'ont précédemment pas porté fruit, en ce sens que les algorithmes minimiseurs de bornes ainsi obtenus n'avaient pas une performance s'approchant de l'état de l'art sur la plupart des jeux de données « benchmark ».

Ces bornes sans terme KL lié à la C-borne ont été présentées pour la première fois dans Laviolette, Marchand et Roy (2011). Une autre version de borne sans KL a été présentée dans Germain, Lacoste et al. (2011), mais dans un contexte de borne basée sur l'approche par compression d'échantillon (Floyd et Warmuth, 1995). Nous étudions ce paradigme plus en détails dans la section 3.6. Ces résultats ont été repris et améliorés dans Germain, Lacasse, Laviolette, Marchand et Roy (2015), où les démonstrations ont simplifiées par l'extraction de l'inégalité du changement de mesure, l'endroit dans la démonstration où le terme KL apparaît dans les bornes PAC-bayésiennes classiques. Les bornes présentées ici sont également plus générales que celles présentées dans Laviolette, Marchand et Roy (2011) : elles considèrent les distribution *alignées*, un concept défini plus bas, plutôt que seulement les distributions *quasi-uniformes*, un cas particulier des distributions alignées.

La littérature contient quelques essais de développer des bornes PAC-bayésiennes qui ne dé-

pendent pas de la divergence KL, voir par exemples les «*localized priors*» de Catoni (2007) ou les «*distribution-dependant priors*» de Lever, Laviolette et Shawe-Taylor (2013). L'idée usuelle est de borner la divergence KL avec une certaine inégalité de concentration. Ici, le terme KL disparaît simplement de la borne, pourvu que nous restreignons la distribution a posteriori à être *alignée* sur la distribution a priori. Le fait que ces nouvelles bornes PAC-bayésiennes ne contiennent pas de divergence KL indique que cette restriction aux distributions a posteriori alignées implique une action de régularisation « naturelle ». Notons également que nous présentons au chapitre 7 des bornes PAC-bayésiennes basées sur d'autres divergences que la divergence KL. Ces travaux ont été réalisés plus tard dans Bégin, Germain, Laviolette et Roy (2016).

3.5.1 Ensembles de votants symétriques et distribution alignées

Dans cette section, nous supposons que l'ensemble (possiblement infini) $\mathcal{F}$ de votants est *symétrique*[3].

Définition 3.14. Un ensemble de votants $\mathcal{F}$ est dit *symétrique* s'il existe une bijection $c : \mathcal{F} \to \mathcal{F}$ telle que pour tout $f \in \mathcal{F}$,

$$c(f) \;=\; -f\,.$$

De plus, nous disons qu'une distribution Q sur un ensemble symétrique $\mathcal{F}$ est *alignée* sur une distribution a priori P si

$$Q(f) + Q(c(f)) \;=\; P(f) + P(c(f))\,, \quad \forall f \in \mathcal{F}\,.$$

Lorsque P est la distribution uniforme et Q est alignée sur P, nous appelons Q une distribution *quasi-uniforme*. Notons que la distribution uniforme est elle-même quasi-uniforme.

Dans le cas où $\mathcal{F}$ est fini, nous considérons des ensembles symétriques $\mathcal{F}$ de $2n$ votants $f : \mathcal{X} \to \overline{\mathcal{Y}}$. Dans ce contexte, pour tout $x \in \mathcal{X}$ et tout $i \in \{1, \ldots, n\}$, nous avons $f_{i+n}(x) = -f_i(x)$. De plus, les distributions Q quasi-uniformes finies sont telles que pour tout $i \in \{1, \ldots, n\}$,

$$Q(f_i) + Q(f_{i+n}) \;=\; \frac{1}{n}\,. \tag{3.9}$$

L'équation (3.9) montre que lorsqu'une distribution Q est restreinte à être quasi-uniforme, la somme des poids donnée à une paire de votants complémentaires est égale à $\frac{1}{n}$. Comme Q

3. Cette notion a été introduite en anglais par « auto-complemented » dans Laviolette, Marchand et Roy (2011), puis renommée « self-complemented » dans Germain, Lacasse, Laviolette, Marchand et Roy (2015). Ce terme indique simplement que l'ensemble des votants est fermé (ou clos) sous l'opération de complémentarité. C'est par exemple le terme utilisé par Demiriz, Bennett et Shawe-Taylor (2002). Un autre terme pour représenter la même idée est « classe d'hypothèse symétrique », introduit dans Daniely et al. (2011). Comme c'est ce terme qui est utilisé dans la littérature récente, nous choisissons de l'adopter pour cette thèse.

est une distribution, le poids de tout votant est borné inférieurement par 0 et borné supérieurement par $\frac{1}{n}$. Cette restriction fait donc apparaître une régularisation L_∞ (*L-infini*). Notons que dans ce contexte, la valeur maximale de $\mathrm{KL}(Q \parallel P)$ est atteinte lorsque tous les votants ont un poids de 0 ou $\frac{1}{n}$. En effet, une distribution quasi-uniforme Q est telle que $\mathrm{KL}(Q \parallel P) \leq n\left(\frac{1}{n}\right) ln\left(\frac{1}{n}/\frac{1}{2n}\right) = ln\, 2$. Conséquemment, la valeur du terme KL est nécessairement petit et ne joue qu'un petit rôle dans les bornes PAC-bayésiennes calculées sur des distributions quasi-uniformes. Les distributions quasi-uniformes ont été introduites dans Germain, Lacasse, Marchand et al. (2009). Cet article montre que si on se limite aux distributions Q quasi-uniformes, nous obtenons une forme de régularisation donnée par le terme KL, qui se situe entre une régularisation L_1 et une régularisation L_2. Le terme KL ne disparaît pas des bornes PAC-bayésiennes, mais cet article montre quand même qu'il y a un intérêt à considérer les distributions quasi-uniformes en théorie PAC-bayésienne.

Les prochains théorèmes et corollaires sont des spécialisations qui permettent d'améliorer légèrement ces bornes PAC-bayésiennes, en faisant disparaître complètement le terme KL.

3.5.2 Théorèmes PAC-bayésiens sans KL pour le risque de Gibbs

Spécialisons premièrement le théorème 3.3 aux distribution alignées et à la perte linéaire $\mathcal{L}_\ell$. Nous avons premièrement besoin d'une nouvelle inégalité du changement de mesure, comme c'est à ce niveau de la preuve du théorème 3.3 que le terme KL apparaît.

Lemme 3.15 (Inégalité du changement de mesure pour distribution alignées)**.**
Pour tout ensemble symétrique $\mathcal{F}$, toute distribution P sur $\mathcal{F}$, toute distribution Q alignée sur P et toute fonction $\phi : \mathcal{F} \to \mathbb{R}$ mesurable sur P telle que $\phi(f) = \phi(c(f))$ pour tout $f \in \mathcal{F}$, nous avons

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

Démonstration. Premièrement, notons qu'il est possible de transformer l'espérance sur Q par une espérance sur P, en utilisant le fait que $\phi(f) = \phi(c(f))$ pour tout $f \in \mathcal{F}$, et le fait que Q est alignée sur P.

$$\begin{aligned}
2 \cdot \mathop{\mathbf{E}}_{f\sim Q} \phi(f) &= \int_{\mathcal{F}} Q(f)\, \phi(f)\, df + \int_{\mathcal{F}} Q(c(f))\, \phi(c(f))\, df \\
&= \int_{\mathcal{F}} Q(f)\, \phi(f)\, df + \int_{\mathcal{F}} Q(c(f))\, \phi(f)\, df \\
&= \int_{\mathcal{F}} \Big(Q(f) + Q(c(f))\Big)\, \phi(f)\, df \\
&= \int_{\mathcal{F}} \Big(P(f) + P(c(f))\Big)\, \phi(f)\, df \\
&= \int_{\mathcal{F}} P(f)\, \phi(f)\, df + \int_{\mathcal{F}} P(c(f))\, \phi(f)\, df \\
&= \int_{\mathcal{F}} P(f)\, \phi(f)\, df + \int_{\mathcal{F}} P(c(f))\, \phi(c(f))\, df \\
&= 2 \cdot \mathop{\mathbf{E}}_{f\sim P} \phi(f)\,.
\end{aligned}$$

Le résultat est obtenu en changeant l'espérance sur Q par une espérance sur P, pour ensuite appliquer l'inégalité de Jensen (Lemme A.3, en annexe A).

$$\mathop{\mathbf{E}}_{f\sim Q}\phi(f) \;=\; \mathop{\mathbf{E}}_{f\sim P}\phi(f) \;=\; \mathop{\mathbf{E}}_{f\sim P} ln\, e^{\phi(f)} \;\leq\; ln\left(\mathop{\mathbf{E}}_{f\sim P} e^{\phi(f)}\right).$$

□

Théorème 3.16 (Théorème PAC-bayésien pour distribution alignées)**.** *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, tout ensemble symétrique $\mathcal{F}$ de votants $\mathcal{X}\to[-1,1]$, toute distribution a priori P sur $\mathcal{F}$, toute fonction convexe $\Delta:[0,1]\times[0,1]\to\mathbb{R}$ pour laquelle $\Delta(q,p)=\Delta(1-q,1-p)$ et pour tout $\delta\in(0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$\Delta\left(R_S(G_Q), R_D(G_Q)\right) \;\leq\; \frac{1}{m}\left[ln\left(\frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}_\ell}(f),\,\mathbb{E}_D^{\mathcal{L}_\ell}(f)\right)}\right)\right].$$

Démonstration. La démonstration suit exactement les mêmes étapes que la démonstration du théorème 3.3, en utilisant la perte linéaire $\mathcal{L}=\mathcal{L}_\ell$ et en remplaçant l'inégalité du changement de mesure du lemme 3.2 par celle du lemme 3.15, avec $\phi(f)=m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}_\ell}(f),\,\mathbb{E}_D^{\mathcal{L}_\ell}(f)\right)$. Notons que cette fonction ϕ possède la propriété désirée, comme

$$\Delta\left(\mathbb{E}_S^{\mathcal{L}_\ell}(f),\,\mathbb{E}_D^{\mathcal{L}_\ell}(f)\right)=\Delta\left(1-\mathbb{E}_S^{\mathcal{L}_\ell}(c(f)),\,1-\mathbb{E}_D^{\mathcal{L}_\ell}(c(f))\right)=\Delta\left(\mathbb{E}_S^{\mathcal{L}_\ell}(c(f)),\,\mathbb{E}_D^{\mathcal{L}_\ell}(c(f))\right).$$

Les autres étapes de la démonstration sont exactement les mêmes que la démonstration du théorème 3.3. □

L'annexe A.2 présente une version plus générale des deux résultats ci-haut, laissée en annexe car la notation nécessaire à cette généralisation est beaucoup plus lourde.

Plusieurs Δ-fonctions ont la propriété que $\Delta(q,p)=\Delta(1-q,1-p)$. Notamment, la divergence Kullback-Leibler Δ_{KL} entre deux distributions de Bernoulli de probabilité de succès p et q, la distance quadratique $\Delta_{V^2}\triangleq 2(q-p)^2$ et la distance de variation $\Delta_V\triangleq 2|q-p|$. La démonstration de cette propriété pour Δ_{KL} est faite plus bas.

Spécialisons maintenant le théorème 3.16 au cas où $\Delta(q,p)=\Delta_{\mathrm{KL}}(q,p)$, défini à l'équation (3.2). En faisant ainsi, nous retrouvons le théorème PAC-bayésien classique (théorème 3.5), mais cette fois-ci en considérant les distributions a posteriori qui sont alignées. Cette version n'a donc pas de terme KL.

Corollaire 3.17. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, toute distribution a priori P sur un ensemble symétrique $\mathcal{F}$ de votants $\mathcal{X}\to[-1,1]$ et tout $\delta\in(0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$\Delta_{\mathrm{KL}}\left(R_S(G_Q),\,R_D(G_Q)\right) \;\leq\; \frac{1}{m}\left[ln\,\frac{2\sqrt{m}}{\delta}\right].$$

Démonstration. Ce résultat s'ensuit du théorème 3.16, en choisissant $\Delta(q,p) = \Delta_{\mathrm{KL}}(q,p)$. Ce théorème nécessite que la fonction Δ choisie soit telle que $\Delta(q,p) = \Delta(1-q,1-p)$. C'est le cas pour la fonction Δ_{KL}, car

$$\begin{aligned}
\Delta_{\mathrm{KL}}(1-q,1-p) &= (1-q)\,ln\,\frac{1-q}{1-p} + (1-(1-q))\,ln\,\frac{1-(1-q)}{1-(1-p)} \\
&= (1-q)\,ln\,\frac{1-q}{1-p} + q\,ln\,\frac{q}{p} \\
&= q\,ln\,\frac{q}{p} + (1-q)\,ln\,\frac{1-q}{1-p} \\
&= \Delta_{\mathrm{KL}}(q,p)\,. \qquad (3.10)
\end{aligned}$$

Le reste de la démonstration est basé sur le lemme 3.4. □

Le corollaire suivant est très similaire à la borne PAC-bayésienne originale de McAllester (2003a), mais sans le terme KL.

Corollaire 3.18. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, tout ensemble symétrique $\mathcal{F}$ de votants $\mathcal{X} \to [-1,1]$, toute distribution a priori P sur $\mathcal{F}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$R_D(G_Q) \;\leq\; R_S(G_Q) + \sqrt{\frac{1}{2m}\left[ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Démonstration. Le résultat est dérivé du corollaire 3.17, en utilisant l'inégalité de Pinsker du lemme A.6, c'est-à-dire

$$2(q-p)^2 \;\leq\; \Delta_{\mathrm{KL}}(q,p)\,,$$

et en isolant $R_D(G_Q)$ pour obtenir l'inégalité. □

Contrairement au théorème 3.3, le théorème 3.16 ne peut pas être utilisé directement avec des votants jumelés comme nous avons pu le faire dans la démonstration du théorème 3.11, car une distribution a posteriori qui est le résultat d'un produit de deux distributions alignées n'est pas nécessairement alignée elle-même. Nous devons donc nous assurer que nous pouvons faire disparaître le terme KL dans ce cas également, ce qui est le propos de la prochaine section.

3.5.3 Théorèmes PAC-bayésiens sans KL pour le désaccord espéré d_Q^D

Le théorème suivant est similaire au théorème 3.16 pour les distribution a posteriori alignées, mais est spécialisé aux votants jumelés. Plutôt que d'utiliser la fonction de perte linéaire, nous utilisons la fonction de perte $\mathcal{L}_d$ de l'équation (3.7), qui peut être vue comme une perte

linéaire appliquée aux votants jumelés. Encore une fois, les deux prochains résultats peuvent être vus comme un cas particulier des deux théorèmes de l'annexe A.2.

Dans cette section, nous utiliserons la notation suivante. Étant donné $f_{ij} = \langle f_i, f_j \rangle$ tel que défini à la définition 3.10, les votants $f_{i^c j}$, f_{ij^c} et $f_{i^c j^c}$ sont définis comme

$$\begin{aligned} f_{i^c j}(x) &\triangleq \langle c(f_i)(x), f_j(x) \rangle , \\ f_{ij^c}(x) &\triangleq \langle f_i(x), c(f_j)(x) \rangle , \\ f_{i^c j^c}(x) &\triangleq \langle c(f_i)(x), c(f_j)(x) \rangle . \end{aligned}$$

Rappelons que par l'équation (3.6) nous avons $\mathcal{F}^2 \triangleq \{ f_{ij} : f_i, f_j \in \mathcal{F} \}$ et $Q^2(f_{ij}) \triangleq Q(f_i) \cdot Q(f_j)$. Similairement, définissons $P^2(f_{ij}) \triangleq P(f_i) \cdot P(f_j)$. En utilisant cette notation, généralisons premièrement l'inégalité du changement de mesure du lemme 3.15 aux votants jumelés.

Lemme 3.19 (Inégalité du changement de mesure pour votants jumelés et distributions alignées)**.** *Pour tout ensemble symétrique $\mathcal{F}$, toute distribution P sur $\mathcal{F}$, toute distribution Q alignée sur P et toute fonction $\phi : \mathcal{F}^2 \to \mathbb{R}$ mesurable sur P telle que $\phi(f_{ij}) = \phi(f_{i^c j}) = \phi(f_{ij^c}) = \phi(f_{i^c j^c})$ pour tout $f_{ij} \in \mathcal{F}^2$, nous avons*

$$\mathop{\mathbf{E}}_{f_{ij} \sim Q^2} \phi(f_{ij}) \;\leq\; ln \left(\mathop{\mathbf{E}}_{f_{ij} \sim P^2} e^{\phi(f_{ij})} \right) .$$

Démonstration. Premièrement, notons qu'il est possible de transformer l'espérance sur Q^2 par une espérance sur P^2 en utilisant le fait que $\phi(f_{ij}) = \phi(f_{i^c j}) = \phi(f_{ij^c}) = \phi(f_{i^c j^c})$ pour tout $f_{ij} \in \mathcal{F}^2$, et le fait que Q est alignée sur P. Plus spécifiquement, nous avons

$$\begin{aligned} & 4 \cdot \mathop{\mathbf{E}}_{f_{ij} \sim Q^2} \phi(f_{ij}) \\ &\quad = \int_{\mathcal{F}^2} Q^2(f_{ij})\, \phi(f_{ij})\, df_{ij} + \int_{\mathcal{F}^2} Q^2(f_{i^c j})\, \phi(f_{i^c j})\, df_{ij} + \int_{\mathcal{F}^2} Q^2(f_{ij^c})\, \phi(f_{ij^c})\, df_{ij} + \int_{\mathcal{F}^2} Q^2(f_{i^c j^c})\, \phi(f_{i^c j^c})\, df_{ij} \\ &\quad = \int_{\mathcal{F}^2} Q^2(f_{ij})\, \phi(f_{ij})\, df_{ij} + \int_{\mathcal{F}^2} Q^2(f_{i^c j})\, \phi(f_{ij})\, df_{ij} + \int_{\mathcal{F}^2} Q^2(f_{ij^c})\, \phi(f_{ij})\, df_{ij} + \int_{\mathcal{F}^2} Q^2(f_{i^c j^c})\, \phi(f_{ij})\, df_{ij} \\ &\quad = \int_{\mathcal{F}^2} \big(Q^2(f_{ij}) + Q^2(f_{i^c j}) + Q^2(f_{ij^c}) + Q^2(f_{i^c j^c}) \big) \phi(f_{ij})\, df_{ij} \\ &\quad = \int_{\mathcal{F}^2} \big(P^2(f_{ij}) + P^2(f_{i^c j}) + P^2(f_{ij^c}) + P^2(f_{i^c j^c}) \big) \phi(f_{ij})\, df_{ij} \\ &\quad \;\vdots \\ &\quad = 4 \cdot \mathop{\mathbf{E}}_{f_{ij} \sim P^2} \phi(f_{ij}) . \end{aligned}$$

Le résultat est obtenu en changeant l'espérance sur Q^2 par une espérance sur P^2, et en appliquant l'inégalité de Jensen (lemme A.3 de l'annexe A.1),

$$\mathop{\mathbf{E}}_{f_{ij} \sim Q^2} \phi(f_{ij}) \;=\; \mathop{\mathbf{E}}_{f_{ij} \sim P^2} \phi(f_{ij}) \;=\; \mathop{\mathbf{E}}_{f_{ij} \sim P^2} ln\, e^{\phi(f_{ij})} \;\leq\; ln \left(\mathop{\mathbf{E}}_{f_{ij} \sim P^2} e^{\phi(f_{ij})} \right) .$$

□

Théorème 3.20 (Théorème PAC-bayésien pour votants jumelés et distributions alignées)**.** *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, tout ensemble symétrique $\mathcal{F}$ de votants $\mathcal{X}\to[-1,1]$, toute distribution P sur $\mathcal{F}$, toute fonction convexe $\Delta:[0,1]\times[0,1]\to\mathbb{R}$ pour laquelle $\Delta(q,p)=\Delta(1-q,1-p)$ et tout $\delta\in(0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$\Delta\Big(\,d_Q^S,\,d_Q^D\,\Big)\;\leq\;\frac{1}{m}\left[ln\left(\frac{1}{\delta}\;\mathop{\mathbf{E}}_{S\sim D^m}\;\mathop{\mathbf{E}}_{f_{ij}\sim P^2}e^{\,m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}_d}(f_{ij}),\,\mathbb{E}_D^{\mathcal{L}_d}(f_{ij})\right)}\right)\right],$$

où f_{ij} est défini à la définition 3.10, et où $P^2(f_{ij})\triangleq P(f_i)\cdot P(f_j)$.

Démonstration. Le théorème 3.20 est déduit du théorème 3.16, en utilisant l'inégalité du changement de mesure du lemme 3.19 plutôt que celle du lemme 3.15, avec

$$\phi(f_{ij})\;=\;m\cdot\Delta\left(\mathbb{E}_S^{\mathcal{L}_d}(f_{ij}),\,\mathbb{E}_D^{\mathcal{L}_d}(f_{ij})\right).$$

Comme la fonction de perte $\mathcal{L}_d$ est telle que

$$\mathbb{E}_{D'}^{\mathcal{L}_d}(f_{i^cj^c})\;=\;\mathbb{E}_{D'}^{\mathcal{L}_d}(f_{ij}),\quad\text{et}\quad\mathbb{E}_{D'}^{\mathcal{L}_d}(f_{i^cj})\;=\;\mathbb{E}_{D'}^{\mathcal{L}_d}(f_{ij^c})\;=\;1-\mathbb{E}_{D'}^{\mathcal{L}_d}(f_{ij}),$$

nous avons que $\phi(f_{ij})$ a la propriété requise pour appliquer le lemme 3.19. □

Spécialisons maintenant le théorème 3.20 à $\Delta(q,p)=\Delta_{\mathrm{KL}}(q,p)$.

Corollaire 3.21. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, tout ensemble symétrique $\mathcal{F}$ de votants $\mathcal{X}\to[-1,1]$, toute distribution a priori P sur $\mathcal{F}$ et tout $\delta\in(0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$\Delta_{\mathrm{KL}}\Big(d_Q^S,\,d_Q^D\Big)\;\leq\;\frac{1}{m}\left[ln\,\frac{2\sqrt{m}}{\delta}\right].$$

Démonstration. Le résultat est obtenu directement du théorème 3.20, en choisissant $\Delta(q,p)=\Delta_{\mathrm{KL}}(q,p)$. Le reste de la preuve est basé sur le lemme 3.4. □

Similairement au corollaire 3.18, nous pouvons facilement dériver le résultat suivant.

Corollaire 3.22. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, tout ensemble symétrique $\mathcal{F}$ de votants $\mathcal{X}\to[-1,1]$, toute distribution a priori P sur $\mathcal{F}$ et tout $\delta\in(0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$d_Q^D\;\geq\;d_Q^S-\sqrt{\frac{1}{2m}\left[ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Démonstration. Le résultat est dérivé du corollaire 3.21, en utilisant l'inégalité de Pinsker (lemme A.6) et en isolant d_Q^D pour obtenir l'inégalité. □

3.5.4 Une borne sur le risque du classificateur par vote de majorité sans KL

Finalement, nous utilisons les résultats précédents pour borner C_Q^D (et donc borner $R_D(B_Q)$) pour les distribution a posteriori Q alignées, donnant lieu à la PAC-borne 3.23. À part le fait que cette borne n'a pas de terme KL, celle-ci est similaire à la PAC-borne 3.13, comme elle borne séparément le risque de Gibbs et le désaccord espéré.

PAC-borne 3.23. *Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour tout ensemble symétrique $\mathcal{F}$ de votants $\mathcal{X}\to[-1,1]$, pour toute distribution a priori P sur $\mathcal{F}$ et tout $\delta\in(0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$R_D(B_Q) \;\leq\; 1-\frac{\left(1-2\cdot\overline{r}\right)^2}{1-2\cdot\underline{d}} \;=\; 1-\frac{\left(\underline{\mu_1}\right)^2}{\overline{\mu_2}},$$

où

$$\overline{r} \triangleq min\left(\tfrac{1}{2},\, R_S(G_Q)+\sqrt{\frac{1}{2m}\left[ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right), \qquad \underline{d} \triangleq max\left(0,\, d_Q^S-\sqrt{\frac{1}{2m}\left[ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right),$$

$$\underline{\mu_1} \triangleq max\left(0,\, \mu_1\big(M_Q^S\big)-\sqrt{\frac{2}{m}\left[ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right), \qquad \overline{\mu_2} \triangleq min\left(1,\, \mu_2\big(M_Q^S\big)+\sqrt{\frac{2}{m}\left[ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right).$$

Démonstration. L'inégalité est une conséquence directe du théorème 2.13 et des corollaires 3.18 et 3.22. L'égalité entre les deux formes de la borne est une application directe des équations (2.8) et (2.10). □

Cette nouvelle borne PAC-bayésienne nous donne un point de départ pour la conception de MinCq, l'algorithme d'apprentissage introduit au chapitre 4 qui consiste essentiellement à trouver le vote de majorité B_Q qui minimise ladite borne. La prochaine section présente une dernière PAC-borne qui, elle, permet l'utilisation d'un ensemble de votants basés sur des fonctions noyaux, dont la définition dépend d'une partie de l'ensemble d'entraînement S. La théorie PAC-bayésienne requiert normalement que la distribution a priori P sur l'ensemble des votants soit définie avant d'avoir observé l'ensemble d'entraînement S. Pour lever cette restriction, nous utilisons la théorie de la *compression d'échantillon*.

3.6 Compression d'échantillon : utilisation de votants construits à partir d'exemples

Les théorèmes PAC-bayésiens des sections précédentes ne sont pas valides lorsque l'ensemble $\mathcal{F}$ contient des votants étant définis à partir des exemples de l'ensemble d'entraînement, par exemple lorsque les fonctions utilisées sont de la forme $\pm k(x_i,\cdot)$ pour un certain noyau

$k : \mathcal{X} \times \mathcal{X} \to [-1, 1]$, fonctions largement utilisés dans les *méthodes à noyaux* comme les *Support Vector Machines* (*SVMs*) (Cortes et Vapnik, 1995). Le problème vient du fait que chaque votant est défini à partir d'un exemple (x_i, y_i) de l'ensemble d'entraînement.

Il existe principalement deux méthodes connues pour résoudre ce problème dans le cadre d'une approche PAC-bayésienne. La première méthode, introduite par Langford et Shawe-Taylor (2003), considère un ensemble substitut $\mathcal{F}^k$ de votants contenant tous les classificateurs linéaires possibles induits par le noyau k. Cet espace est aussi connu comme un «*Reproducible Kernel Hilbert Space*» (*RKHS*).[4] Les auteurs utilisent ensuite le *théorème du représentant* pour montrer que la fonction de classification admet une représentation sous la forme d'une combinaison linéaire des exemples de l'ensemble d'entraînement, comme c'est le cas pour le SVM. Pour éviter le *fléau de la dimensionnalité* et permettre le calcul analytique de la divergence Kullback-Leibler, ils proposent de restreindre les choix de distribution a priori et a posteriori sur $\mathcal{F}^k$ aux fonctions gaussiennes isotropiques centrées sur un vecteur représentant un certain classificateur linéaire. En se basant sur cette idée, Germain, Lacasse, Laviolette et Marchand (2009) ont suggéré un algorithme d'apprentissage pour classificateurs linéaires consistant à minimiser une borne PAC-bayésienne similaire à celle proposée dans Langford et Shawe-Taylor (2003).

La seconde méthode, présentée très brièvement dans cette section, est basée sur la compression d'échantillon de Floyd et Warmuth (1995), qui a été adaptée à la théorie PAC-bayésienne par Laviolette et Marchand (2005) et Laviolette et Marchand (2007), permettant de traiter directement le cas où les votants sont construits à partir d'exemples de l'ensemble d'entraînement, sans avoir à impliquer les notions de RKHS ou le théorème du représentant. Contrairement à la première approche décrite ci-haut, l'approche de compression d'échantillon permet d'utiliser n'importe quelle mesure de similarité pour construire des votants, et pas seulement des noyaux.

Cette section contient une adaptation des résultats présentés dans Germain, Lacasse, Laviolette, Marchand et Roy (2015). La PAC-borne résultante est une justification théorique permettant à l'algorithme MinCq du chapitre 4 d'utiliser des votants définis à partir des exemples d'entraînement. La section correspondante dans Germain, Lacasse, Laviolette, Marchand et Roy (2015) est totalement due à nos coauteurs, mais nous présentons ici une version encore plus simplifiée nous permettant de finaliser la justification théorique de MinCq. Dans Germain, Lacasse, Laviolette, Marchand et Roy (2015), les *votants comprimés* sont de taille *au plus* λ. Ici, nous ne considérons que les votants de taille *exactement* λ. Dans cette section, nous ajoutons également une contribution supplémentaire : une PAC-borne avec un terme KL, sur laquelle sera basée l'algorithme CqBoost du chapitre 5. Notons que nous ne fournissons pas les preuves des quatre théorèmes PAC-bayésiens sur lesquels dépendent les

4. Pour plus d'informations sur les RKHS, voir Cristianini et Shawe-Taylor (2000b) et Schölkopf, Herbrich et Smola (2001).

deux PAC-bornes. Ces preuves sont plus techniques et requièrent l'introduction de notation et de lemmes supplémentaires, dont nous ne sommes pas l'auteur. Le lecteur intéressé à consulter les éléments nécessaire pour construire ces preuves peut se référer à Germain, Lacasse, Laviolette, Marchand et Roy (2015, section 7).

Cadre général de compression d'échantillon

Dans le cadre de la *compression d'échantillon*, les algorithmes d'apprentissage ont accès à un ensemble de votants dépendants des données, que nous appelons des *votants comprimés*. Étant donné une séquence d'entraînement[5] $S = \langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$, chaque votant comprimé est décrit par une séquence $S_{\mathbf{i}}$ d'éléments de S, appelée la *séquence de compression*, et un *message* σ représentant l'information supplémentaire nécessaire pour obtenir un votant à partir de $S_{\mathbf{i}}$. Si $\mathbf{i} = \langle i_1, i_2, .., i_k \rangle$, alors $S_{\mathbf{i}} \triangleq \left\langle \left(x_{i_1}, y_{i_1}\right), \left(x_{i_2}, y_{i_2}\right), \ldots, \left(x_{i_k}, y_{i_k}\right) \right\rangle$. Dans cette thèse, les répétitions sont permises dans $S_{\mathbf{i}}$, et k, le nombre d'indices présents dans $\mathbf{i}$ (en comptant les répétitions), est dénoté par $|\mathbf{i}|$.

Le fait que chaque votant comprimé soit décrit par une séquence de compression et un message implique qu'il existe une *fonction de reconstruction* $\mathcal{R}(S_{\mathbf{i}}, \sigma)$ qui retourne un votant lorsqu'on l'applique à une séquence de compression arbitraire $S_{\mathbf{i}}$ et un message σ. Le message σ est choisi à partir de l'ensemble $\Sigma_{S_{\mathbf{i}}}$ de tous les messages pouvant être fournis avec la séquence de compression $S_{\mathbf{i}}$. Dans le cadre PAC-bayésien, $\Sigma_{S_{\mathbf{i}}}$ doit être défini a priori (c'est-à-dire, avant d'avoir observé les données d'entraînement) pour toutes les séquences $S_{\mathbf{i}}$ possibles, et peut être un ensemble discret ou continu. Le cadre de la compression d'échantillon généralise le cadre classique, ce dernier correspondant au cas où $|\mathbf{i}| = 0$, les votants étant définis seulement avec les messages (sans utiliser d'exemples). Plusieurs algorithmes classiques de la littérature peuvent être analysés avec ce cadre général de compression d'échantillon. Notamment, l'algorithme Support Vector Machines (Cortes et Vapnik, 1995) peut être vu comme un vote de majorité de votants comprimés avec une taille de compression de 1 (Germain, Lacoste et al., 2011), l'algorithme des k plus proches voisins peut être vu comme le cas dégénéré où tout l'ensemble d'entraînement S appartient à l'ensemble de compression, et l'algorithme *Set Covering Machine* (Marchand et Shawe-Taylor, 2001) est théoriquement justifié par une borne basée sur la compression d'échantillon.

Cadre simplifié de compression d'échantillon

Pour les besoins de cette thèse, nous considérons un cadre simplifié où chaque votant comprimé a une séquence de compression d'*exactement* λ exemples (possiblement avec répétitions), et une chaîne de messages de λ bits que nous représentons par une séquence de «−1» et «+1».

5. La littérature de la théorie de la compression d'échantillon est plus rigoureuse et parlent bel et bien d'une *séquence* de données plutôt que d'un ensemble. Nous cessons donc temporairement notre abus de langage présent dans le reste de la thèse (convention 2.2).

Plutôt que d'être définie sur un ensemble de votants comprimés, la distribution de poids Q est définie sur $\mathcal{I}_\lambda \times \Sigma_\lambda$, où

$$\mathcal{I}_\lambda \triangleq \left\{ \langle i_1, i_2, \ldots, i_\lambda \rangle \,:\, i_j \in \{1, \ldots, m\} \; \forall j \in \{1, \ldots, \lambda\} \right\} \quad \text{et} \quad \Sigma_\lambda \triangleq \left\{-1, 1\right\}^\lambda. \tag{3.11}$$

En d'autres mots, $Q(\mathbf{i}, \boldsymbol{\sigma})$ correspond au poids associé au votant comprimé $\mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma})$, c'est-à-dire, le votant comprimé de séquence de compression $\mathbf{i} = \left\langle i_1, i_2, \ldots, i_{|\mathbf{i}|} \right\rangle \in \mathcal{I}_\lambda$ et de message $\boldsymbol{\sigma} = \langle \sigma_1, \sigma_2, \ldots, \sigma_\lambda \rangle \in \Sigma_\lambda$. En particulier, une distribution a priori (ou a posteriori) sur l'ensemble de tous les votants comprimés est maintenant simplement une distribution a priori sur l'ensemble $\mathcal{I}_\lambda \times \Sigma_\lambda$. Alors, une telle distribution peut bel et bien être définie a priori, avant d'observer les données de S.[6] L'ensemble des votants comprimés est donc seulement défini lorsque la séquence d'entraînement S est donnée, et correspond à

$$\mathcal{F}^{\mathcal{R}}_{S,\lambda} \triangleq \left\{ \mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma}) \,:\, \mathbf{i} \in \mathcal{I}_\lambda, \, \boldsymbol{\sigma} \in \Sigma_\lambda \right\}.$$

Finalement, étant donné une séquence d'entraînement S et une fonction de reconstruction $\mathcal{R}$, pour une définition Q sur $\mathcal{I}_\lambda \times \Sigma_\lambda$, nous définissons le classificateur par vote de majorité par

$$B_{Q,S} \triangleq \operatorname{sgn}\left[\mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim Q} \mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma}) \right].$$

Nous définissons le risque de du classificateur par vote de majorité $R_{D'}\left(B_{Q,S}\right)$ et le risque du classificateur de Gibbs $R_{D'}\left(G_{Q,S}\right)$ d'une distribution Q sur $\mathcal{I}_\lambda \times \Sigma_\lambda$ relativement à une distribution D' comme suit :

$$\begin{aligned} R_{D'}\left(B_{Q,S}\right) &\triangleq \mathbb{E}^{\mathcal{L}_{01}}_{D'}\left(B_{Q,S}\right), \\ R_{D'}\left(G_{Q,S}\right) &\triangleq \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim Q} \mathbb{E}^{\mathcal{L}_\ell}_{D'}\left(\mathcal{R}(S_\mathbf{i}, \boldsymbol{\sigma}) \right). \end{aligned}$$

Tous les résultats PAC-bayésiens présentés jusqu'à maintenant peuvent être généralisés au cadre de la compression d'échantillon. Pour cette thèse, nous nous contenterons d'énoncer les théorèmes nécessaires à la justification des algorithmes MinCq et CqBoost, présentés respectivement aux chapitres 4 et 5. Nous présentons premièrement deux bornes PAC-bayésiennes avec terme KL, l'une pour borner le risque de Gibbs, et l'autre pour borner le désaccord espéré. Nous présentons ensuite deux bornes PAC-bayésiennes pour distributions alignées, cette fois-ci sans terme KL.

Bornes PAC-bayésiennes avec et sans KL pour la compression d'échantillon

Généralisons premièrement la notion d'ensemble symétrique de votants (définition 3.14) au cadre de la compression d'échantillon.

6. Laviolette et Marchand (2007) décrit un cadre plus général, où pour chaque $S \in (\mathcal{X} \times \mathcal{Y})^m$, une distribution a priori est définie sur $\mathcal{I}_\lambda \times \Sigma_{S_\mathbf{i}}$. Alors, les messages peuvent dépendre de la séquence de compression $S_\mathbf{i}$.

Définition 3.24. Une fonction de reconstruction $\mathscr{R}$ est *symétrique* si pour toute séquence d'entraînement $S \in (\mathscr{X} \times \mathscr{Y})^m$ et tout $(\mathbf{i}, \boldsymbol{\sigma}) \in \mathscr{I}_\lambda \times \Sigma_\lambda$, nous avons

$$-\mathscr{R}(S_{\mathbf{i}}, \boldsymbol{\sigma}) \quad = \quad \mathscr{R}(S_{\mathbf{i}}, -\boldsymbol{\sigma}) \,,$$

où, si $\boldsymbol{\sigma} = \langle \sigma_1, .., \sigma_\lambda \rangle$, alors $-\boldsymbol{\sigma} = \langle -\sigma_1, .., -\sigma_\lambda \rangle$.

Les théorèmes suivants sont une généralisation des corollaires 3.7 et 3.18 pour la compression d'échantillon.

Théorème 3.25. *Soit $\mathscr{R}$ une fonction de reconstruction qui retourne des votants comprimés de taille λ (où $\lambda < m$). Pour toute distribution D sur $\mathscr{X} \times \{-1, 1\}$, pour toute distribution a priori P sur $\mathscr{I}_\lambda \times \Sigma_\lambda$ et tout $\delta \in (0, 1]$, nous avons avec probabilité au moins $1 - \delta$ sur le choix d'une séquence de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q,*

$$R_D(G_{Q,S}) \;\leq\; R_S(G_{Q,S}) + \sqrt{\frac{1}{2(m-\lambda)} \left[\mathrm{KL}(Q \,\|\, P) + 4\lambda + ln\, \tfrac{2\sqrt{m-\lambda}}{\delta} \right]}\,.$$

Démonstration. Voir Germain, Lacasse, Laviolette, Marchand et Roy (2015, théorème 39) pour une démonstration d'une version légèrement différente de ce théorème. Le même processus de preuve peut être appliqué ici. □

Théorème 3.26. *Soit $\mathscr{R}$ une fonction de reconstruction symétrique qui retourne des votants comprimés de taille λ (où $\lambda < m$). Pour toute distribution D sur $\mathscr{X} \times \{-1, 1\}$, pour toute distribution a priori P sur $\mathscr{I}_\lambda \times \Sigma_\lambda$ et tout $\delta \in (0, 1]$, nous avons avec probabilité au moins $1 - \delta$ sur le choix d'une séquence de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$R_D(G_{Q,S}) \;\leq\; R_S(G_{Q,S}) + \sqrt{\frac{1}{2(m-\lambda)} \left[4\lambda + ln\, \tfrac{2\sqrt{m-\lambda}}{\delta} \right]}\,.$$

Démonstration. Voir Germain, Lacasse, Laviolette, Marchand et Roy (2015, théorème 41) pour une démonstration d'une version légèrement différente de ce théorème. Le même processus de preuve peut être appliqué ici. □

Dans ces théorèmes, le terme $m - \lambda$ vient du fait que nous devons enlever les éléments de l'ensemble de compression pour le calcul de la borne, sinon les données ne peuvent pas être considérées *i.i.d.* Le terme 4λ vient du fait qu'ainsi, il y a un biais entre $R_S(G_{Q,S})$ et $R_{S \setminus S_{\mathbf{i}}}(G_{Q,S})$ qu'il faut borner. L'idée est la même dans les théorèmes qui suivent.

Définissons maintenant le désaccord espéré adapté au cadre de la compression d'échantillon. Étant donné une séquence d'entraînement S et une fonction de reconstruction $\mathscr{R}$, nous défi-

nissons le désaccord espéré d'une distribution Q sur $\mathcal{J}_\lambda \times \Sigma_\lambda$ par rapport à D' comme

$$\begin{aligned} d^{D'}_{Q,S} &\triangleq \mathop{\mathbf{E}}_{x\sim D'_\mathcal{X}} \mathop{\mathbf{E}}_{(\mathbf{i},\boldsymbol{\sigma})\sim Q} \mathop{\mathbf{E}}_{(\mathbf{i}',\boldsymbol{\sigma}')\sim Q} \mathcal{L}_\ell\left(\, \mathcal{R}(S_\mathbf{i},\boldsymbol{\sigma})(x), \mathcal{R}(S_{\mathbf{i}'},\boldsymbol{\sigma}')(x)\,\right) \\ &= \mathop{\mathbf{E}}_{(\mathbf{i},\mathbf{i}',\boldsymbol{\sigma},\boldsymbol{\sigma}')\sim Q^2} \mathbb{E}^{\mathcal{L}_d}_{D'}\Big(\overline{\mathcal{R}}(S_{\mathbf{i},\mathbf{i}'},\boldsymbol{\sigma},\boldsymbol{\sigma}')\Big)\,, \end{aligned}$$

où

$$\begin{aligned} Q^2(\mathbf{i},\mathbf{i}',\boldsymbol{\sigma},\boldsymbol{\sigma}') &\triangleq Q(\mathbf{i},\boldsymbol{\sigma})\cdot Q(\mathbf{i}',\boldsymbol{\sigma}')\,, \\ \overline{\mathcal{R}}(S_{\mathbf{i},\mathbf{i}'},\boldsymbol{\sigma},\boldsymbol{\sigma}')(x) &\triangleq \left\langle\, \mathcal{R}(S_\mathbf{i},\boldsymbol{\sigma})(x),\ \mathcal{R}(S_{\mathbf{i}'},\boldsymbol{\sigma}')(x)\,\right\rangle\,. \end{aligned}$$

Alors, $\overline{\mathcal{R}}$ est une nouvelle fonction de reconstruction qui retourne un *votant comprimé jumelé*, une version des votants jumelés (définition 3.10) en compression d'échantillon. Nous adaptons maintenant les corollaires 3.12 (appliqué avec $\alpha^{D'}_Q = d^{D'}_Q$) et 3.22 aux votants comprimés, et obtenons donc les théorèmes PAC-bayésiens suivants pour le désaccord espéré.

Théorème 3.27. *Soit $\mathcal{R}$ une fonction de reconstruction, qui retourne des votants comprimés de taille λ (où $\lambda < \lfloor \frac{m}{2} \rfloor$). Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour toute distribution P sur $\mathcal{J}_\lambda \times \Sigma_\lambda$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'une séquence de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q,*

$$d^D_{Q,S} \ \geq\ d^S_{Q,S} - \sqrt{\frac{1}{2(m-2\lambda)}\left[2\cdot \mathrm{KL}(Q\,\|\,P) + 8\lambda + ln\,\tfrac{2\sqrt{m-2\lambda}}{\delta}\right]}\,.$$

Démonstration. Voir Germain, Lacasse, Laviolette, Marchand et Roy (2015, théorème 42) pour une démonstration d'une version légèrement différente de ce théorème. Le même processus de preuve peut être appliqué ici. □

Théorème 3.28. *Soit $\mathcal{R}$ une fonction de reconstruction symétrique, qui retourne des votants comprimés de taille λ (où $\lambda < \lfloor \frac{m}{2} \rfloor$). Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour toute distribution P sur $\mathcal{J}_\lambda \times \Sigma_\lambda$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'une séquence de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$d^D_{Q,S} \ \geq\ d^S_{Q,S} - \sqrt{\frac{1}{2(m-2\lambda)}\left[8\lambda + ln\,\tfrac{2\sqrt{m-2\lambda}}{\delta}\right]}\,.$$

Démonstration. Voir Germain, Lacasse, Laviolette, Marchand et Roy (2015, théorème 42) pour une démonstration d'une version légèrement différente de ce théorème. Le même processus de preuve peut être appliqué ici. □

Nous avons maintenant les éléments nécessaires pour construire deux PAC-bornes, l'une simultanément valide pour toute distribution Q, dans laquelle intervient une divergence Kullback-Leibler, et l'une restreinte aux distributions Q alignées, cette fois-ci sans KL.

Deux bornes sur le risque du vote de majorité en compression d'échantillon

Exploitons maintenant les théorèmes 3.25, 3.26, 3.27 et 3.28, ainsi que la $\mathcal{C}$-borne du théorème 2.13, pour obtenir deux bornes sur le risque du vote de majorité utilisant des noyaux comme votants. Étant donné une fonction de similarité (possiblement un noyau) $k : \mathcal{X} \times \mathcal{X} \to [-1, 1]$ et une séquence d'entraînement de m éléments, nous considérons un vote de majorité de votants comprimés dont la taille de compression est toujours égale à 1, donnés par la fonction de reconstruction

$$\mathcal{R}_k(S_{\mathbf{i}}, \langle\sigma\rangle)(x) \;\triangleq\; \sigma \cdot k(x_i, x)\,,$$

où $\mathbf{i} \in \mathcal{I}_1 = \{\langle 1\rangle, \langle 2\rangle, \ldots, \langle m\rangle\}$, $\langle\sigma\rangle \in \Sigma_1$ (alors, $\sigma \in \{-1, 1\}$), et i est l'index contenu dans $\mathbf{i}$. Ici, les éléments des ensembles $\mathcal{I}_1$ et Σ_1 sont obtenus des équations (3.11), avec $\lambda = 1$. Notons que $\mathcal{R}_k$ est symétrique (définition 3.24) car $-\mathcal{R}_k(S_{\mathbf{i}}, \langle\sigma\rangle) = \mathcal{R}_k(S_{\mathbf{i}}, \langle-\sigma\rangle)$ pour tout $(\mathbf{i}, \sigma)$.

Une fois la séquence d'entraînement $S \sim D^m$ observée, la fonction de reconstruction (symétrique) $\mathcal{R}_k$ nous donne l'ensemble de $2m$ votants comprimés suivants :

$$\mathcal{F}_{S,1}^{\mathcal{R}_k} \;\triangleq\; \Big\{k(x_1, \cdot), k(x_2, \cdot), \ldots, k(x_m, \cdot), -k(x_1, \cdot), -k(x_2, \cdot), \ldots, -k(x_m, \cdot)\Big\}\,.$$

Notons que $\mathcal{F}_{S,1}^{\mathcal{R}_k}$ est un ensemble symétrique de votants comprimés, et la marge du vote de majorité donné par la distribution Q sur $\mathcal{F}_{S,1}^{\mathcal{R}_k}$ est

$$M_{Q,S}(x, y) \;\triangleq\; y\left(\sum_{i=1}^{m}\Big[Q(\,k(x_i, \cdot)\,) - Q(\,-k(x_i, \cdot)\,)\Big]\,k(x_i, x)\right).$$

Conséquemment, les premier et second moments empiriques de cette marge sont

$$\mu_1\big(M_{Q,S}^S\big) \;=\; \frac{1}{m}\sum_{i=1}^{m} M_{Q,S}(x_i, y_i), \quad \text{et} \quad \mu_2\big(M_{Q,S}^S\big) \;=\; \frac{1}{m}\sum_{i=1}^{m}\Big[M_{Q,S}(x_i, y_i)\Big]^2.$$

Le risque empirique de Gibbs et le désaccord espéré empirique associés sont donnés par

$$R_S\big(G_{Q,S}\big) \;=\; \frac{1}{2}\Big(1 - \mu_1\big(M_{Q,S}^S\big)\Big), \quad \text{et} \quad d_{Q,S}^S \;=\; \frac{1}{2}\Big(1 - \mu_2\big(M_{Q,S}^S\big)\Big)\,. \tag{3.12}$$

Énonçons finalement les deux bornes suivantes sur le risque du classificateur par vote de majorité $R_D\big(B_{Q,S}\big)$ de votants noyaux, dans un premier temps pour toute distribution Q, et dans un second temps pour les distributions Q alignées.

PAC-borne 3.29. *Soit $k : \mathcal{X} \times \mathcal{X} \to [-1, 1]$. Pour toute distribution D sur $\mathcal{X}\times\{-1, 1\}$, pour toute distribution a priori P sur $\mathcal{F}_{S,1}^{\mathcal{R}_k}$ et tout $\delta \in (0, 1]$, nous avons avec probabilité au moins $1 - \delta$ sur le choix d'une séquence de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q,*

$$R_D\big(B_{Q,S}\big) \;\leq\; 1 - \frac{(\,1 - 2\cdot\overline{r}\,)^2}{1 - 2\cdot\underline{d}} \;=\; 1 - \frac{\Big(\,\underline{\mu_1}\,\Big)^2}{\overline{\mu_2}}\,,$$

où

$$\begin{aligned}
\overline{r} &\triangleq min\left(\tfrac{1}{2},\, R_S(G_{Q,S}) + \sqrt{\tfrac{1}{2(m-1)}\left[\mathrm{KL}(Q \,\|\, P) + 4 + ln\, \tfrac{2\sqrt{m-1}}{\delta/2}\right]}\right),\\
\underline{d} &\triangleq max\left(0,\, d^S_{Q,S} - \sqrt{\tfrac{1}{2(m-2)}\left[2\cdot\mathrm{KL}(Q \,\|\, P) + 8 + ln\, \tfrac{2\sqrt{m-2}}{\delta/2}\right]}\right),\\
\underline{\mu_1} &\triangleq max\left(0,\, \mu_1\big(M^S_{Q,S}\big) - \sqrt{\tfrac{2}{m-1}\left[\mathrm{KL}(Q \,\|\, P) + 4 + ln\, \tfrac{2\sqrt{m-1}}{\delta/2}\right]}\right),\\
\overline{\mu_2} &\triangleq min\left(1,\, \mu_2\big(M^S_{Q,S}\big) + \sqrt{\tfrac{2}{m-2}\left[2\cdot\mathrm{KL}(Q \,\|\, P) + 8 + ln\, \tfrac{2\sqrt{m-2}}{\delta/2}\right]}\right).
\end{aligned}$$

Démonstration. La preuve est pratiquement identique à celle de la PAC-borne 3.13, à l'exception qu'elle dépend des bornes PAC-bayésiennes pour compression d'échantillon. En effet, l'inégalité est une conséquence du théorème 2.13, ainsi que des théorèmes 3.25 et 3.27. L'égalité $1 - \frac{(1-2\cdot\overline{r})^2}{1-2\cdot\underline{d}} = 1 - \frac{(\underline{\mu_1})^2}{\overline{\mu_2}}$ est une application directe de l'équation (3.12). □

PAC-borne 3.30. *Soit $k : \mathcal{X}\times\mathcal{X} \to [-1,1]$. Pour toute distribution D sur $\mathcal{X}\times\{-1,1\}$, pour toute distribution a priori P sur $\mathcal{F}^{\mathcal{R}_k}_{S,1}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'une séquence de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$R_D(B_{Q,S}) \;\leq\; 1 - \frac{(\,1-2\cdot\overline{r}\,)^2}{1-2\cdot\underline{d}} \;=\; 1 - \frac{\left(\,\underline{\mu_1}\,\right)^2}{\overline{\mu_2}}\,,$$

où

$$\begin{aligned}
\overline{r} &\triangleq min\left(\tfrac{1}{2},\, R_S(G_{Q,S}) + \sqrt{\tfrac{1}{2(m-1)}\left[4 + ln\, \tfrac{2\sqrt{m-1}}{\delta/2}\right]}\right),\\
\underline{d} &\triangleq max\left(0,\, d^S_{Q,S} - \sqrt{\tfrac{1}{2(m-2)}\left[8 + ln\, \tfrac{2\sqrt{m-2}}{\delta/2}\right]}\right),\\
\underline{\mu_1} &\triangleq max\left(0,\, \mu_1\big(M^S_{Q,S}\big) - \sqrt{\tfrac{2}{m-1}\left[4 + ln\, \tfrac{2\sqrt{m-1}}{\delta/2}\right]}\right),\\
\overline{\mu_2} &\triangleq min\left(1,\, \mu_2\big(M^S_{Q,S}\big) + \sqrt{\tfrac{2}{m-2}\left[8 + ln\, \tfrac{2\sqrt{m-2}}{\delta/2}\right]}\right).
\end{aligned}$$

Démonstration. La preuve est pratiquement identique à celle de la PAC-borne 3.23, à l'exception qu'elle dépend des bornes PAC-bayésiennes pour compression d'échantillon. En effet, l'inégalité est une conséquence du théorème 2.13, ainsi que des théorèmes 3.26 et 3.28. L'égalité $1 - \frac{(1-2\cdot\overline{r})^2}{1-2\cdot\underline{d}} = 1 - \frac{(\underline{\mu_1})^2}{\overline{\mu_2}}$ est une application directe de l'équation (3.12). □

Les PAC-bornes basées sur la compression d'échantillon présentées dans cette section nous permettront de justifier théoriquement les algorithmes MinCq (chapitre 4) et CqBoost (chapitre 5), dans les situations où les votants utilisés sont des fonctions noyaux définis sur les exemples de l'ensemble d'entraînement.

3.7 Conclusion du chapitre

Nous avons présenté la théorie PAC-bayésienne, une théorie nous permettant d'obtenir des bornes de généralisation pour les votes de majorité. Nous avons introduit une unification et une simplification de la théorie, nous permettant de retrouver les bornes de l'état de l'art, mais également d'en développer de nouvelles.

Nous avons introduit quatre bornes de généralisation basées sur la C-borne, qui sont plus précises que les bornes de l'état de l'art dans la situation où un vote de majorité est basé sur des votants peu performants individuellement. Deux d'entre elles ne considèrent pas de restriction sur les distributions a posteriori possibles, et contiennent dans leur expression la divergence Kullback-Leibler (KL) entre la distribution a posteriori Q et la distribution a priori P, alors que les deux autres ajoutent la restriction que Q doit être alignée sur P, ce qui a pour effet de faire disparaître le terme KL. Deux d'entres elles considèrent que les votants sont définis a priori, avant d'avoir vu l'ensemble d'entraînement S, alors que les deux autres considèrent que les votants peuvent être une fonction noyau, où chaque fonction peut être basé sur un exemple de l'ensemble d'entraînement. Plus précisément,

- la PAC-borne 3.13 justifie la minimisation de la C-borne empirique si la valeur du terme KL est contrôlée ;
- la PAC-borne 3.29 est similaire, mais justifie également les algorithmes retournant un vote de majorité de fonctions noyaux ;
- la PAC-borne 3.23 justifie la minimisation de la C-borne empirique lorsque l'algorithme est restreint aux distributions alignées ;
- la PAC-borne 3.30 est similaire, mais justifie également les algorithmes retournant un vote de majorité de fonctions noyaux.

Nous avons maintenant tous les outils théoriques nécessaires pour développer des algorithmes d'apprentissage minimisant la C-borne, dont les différentes PAC-bornes fourniront les garanties de généralisation. Dans les deux prochains chapitres, nous nous attaquons à la tâche de développer de tels algorithmes.

Chapitre 4

MinCq : apprendre en minimisant la C-borne

Dans ce chapitre, nous introduisons un nouvel algorithme d'apprentissage nommé MinCq, qui étant donné un ensemble symétrique de votants, retourne un vote de majorité pondéré minimisant la C-borne tout en bénéficiant des garanties de généralisation fournies par les PAC-bornes 3.23 et 3.30. MinCq peut être simplement exprimé comme un programme quadratique sur une matrice semi-définie positive et a une performance comparable à l'état de l'art.

MinCq est l'algorithme central de cette thèse. Il a été présenté pour la première fois dans Laviolette, Marchand et Roy (2011). Une version plus complète, mieux justifiée (avec la compression d'échantillon) et avec plus d'expérimentation a été présentée dans Germain, Lacasse, Laviolette, Marchand et Roy (2015). Dans cette thèse, la présentation de l'algorithme diffère quelque peu de ces deux articles : nous introduisons maintenant une matrice de vote $\mathbf{F}$ permettant de simplifier la notation. Cette nouvelle notation a été présentée pour la première fois dans Roy, Marchand et Laviolette (2016). Nous donnons également les informations nécessaires pour résoudre les programmes quadratiques présentés avec un solveur standard tel que CVXOPT (Dahl et Vandenberghe, 2007).

Comme c'est le cas pour les algorithmes de boosting (Freund et Schapire, 1997), MinCq est conçu pour construire un vote de majorité pondéré à partir de votants qui ont une mauvaise performance individuelle, souvent appelés «*weak learners*» dans la littérature. Conséquemment, la décision d'un tel vote de majorité est basée sur une faible majorité (c'est-à-dire, avec un risque de Gibbs très près de $1/2$). Rappelons que dans les situations où le risque de Gibbs est grand (c'est-à-dire, lorsque le premier moment de la marge est près de 0), la C-borne peut tout de même avoir une petite valeur si les votants du vote de majorité sont maximalement décorrélés.

Malheureusement, minimiser la valeur empirique de la C-borne a tendance à surapprendre les données d'entraînement. Pour éviter ce problème, MinCq utilise une distribution Q de votants qui est contrainte à être quasi-uniforme (voir l'équation (3.9)), et pour qui le premier moment de la marge est forcée à ne pas être trop près de 0. Plus précisément, la valeur $\mu_1\left(M_Q^S\right)$ est contrainte à être plus grande qu'une certaine constante positive μ. Ce μ devient donc un hyperparamètre de l'algorithme, choisi par la méthode de validation croisée, tout comme le paramètre C de l'algorithme SVM. Cette nouvelle stratégie d'apprentissage est justifiée par la PAC-borne 3.23, dédiée aux distributions quasi-uniformes[1] et la PAC-borne 3.30, permettant en plus l'utilisation de votants définis à partir de noyaux. MinCq peut être vu comme un algorithme qui cherche simplement le vote de majorité de marge au moins μ minimisant la PAC-borne 3.23 (ou la PAC-borne 3.30 lors de l'utilisation de noyaux).

MinCq est également justifié par deux propriétés importantes des votes de majorité quasi-uniformes. Premièrement, comme le montrera le théorème 4.3, il n'y a pas de perte de généralité lorsque nous nous restreignons aux distributions quasi-uniformes. De plus, comme le montrera le théorème 4.4, pour toute marge $\mu > 0$ et toute distribution quasi-uniforme Q telle que $\mu_1\left(M_Q^S\right) \geq \mu$, il existe une autre distribution quasi-uniforme Q' dont la marge est exactement μ et dont le vote de majorité est exactement le même. Ce vote de majorité a donc la même valeur de C-borne.

Pour minimiser la C-borne, l'algorithme d'apprentissage doit substantiellement minimiser le second moment de la distribution des marges $\mu_2\left(M_Q^S\right)$ (et ainsi maximiser le désaccord espéré), tout en conservant son premier moment $\mu_1\left(M_Q^S\right)$ au dessus du seuil μ. Plusieurs algorithmes d'apprentissage exploitent cette stratégie de différentes manières. Par exemple, la variance de la distribution des marges est contrôlée dans Breiman (2001) pour produire des *forêts aléatoires* (*random forests*), par Dredze, Kulesza et Crammer (2010) dans le cadre du *transfer learning*, et par Shen et Li (2010) dans leur algorithme MDBoost. Ainsi, minimiser la variance de la marge est une idée bien connue et utilisée. Nous proposons ici une nouvelle justification théorique justifiant indirectement ce type d'algorithme, et proposons l'algorithme MinCq qui lui, minimise directement la C-borne et est directement justifié par la théorie.

4.1 De la C-borne à l'algorithme d'apprentissage MinCq

Nous considérons les algorithmes d'apprentissage qui construisent des votes de majorité basés sur un espace symétrique (fini) d'hypothèses $\mathcal{F} = \{f_1, \ldots, f_{2n}\}$ de votants à valeur réelle. Nous rappelons que ces votants peuvent être des classificateurs tels que des *souches de décision*, ou bien des fonctions à valeur réelle telles que des *noyaux k* évalués sur les exemples de S, tels que $f_i(\cdot) = y_i\, k(x_i, \cdot)$.

1. La PAC-borne 3.23 est plus générale et est dédiée aux distribution a posteriori Q alignées sur une distribution a priori P. Par contre, dans ce chapitre, nous considérons toujours la distribution a priori P uniforme, rendant la distribution Q quasi-uniforme.

Nous considérons la seconde forme de la C-borne, qui dépend des deux premiers moments de la marge du classificateur par vote de majorité (voir le théorème 2.13) :

$$C_Q^{D'} \;=\; 1 - \frac{\left(\mu_1\left(M_Q^{D'}\right)\right)^2}{\mu_2\left(M_Q^{D'}\right)}\,.$$

Nos premiers essais de minimiser la C-borne nous ont confrontés à deux problèmes.

Problème 4.1. *La minimisation de la C-borne empirique sans régularisation a tendance à surapprendre les données d'entraînement.*

Problème 4.2. *La plupart du temps, les distributions Q qui minimisent la C-borne C_Q^S sont telles que $\mu_1\left(M_Q^S\right)$ et $\mu_2\left(M_Q^S\right)$ sont très près de 0. Comme $C_Q^S = 1 - \left(\mu_1\left(M_Q^S\right)\right)^2 / \mu_2\left(M_Q^S\right)$, il y a une instabilité numérique de forme $0/0$. Comme $\left(\mu_1\left(M_Q^D\right)\right)^2 / \mu_2\left(M_Q^D\right)$ peut seulement être estimé par $\left(\mu_1\left(M_Q^S\right)\right)^2 / \mu_2\left(M_Q^S\right)$, ce problème amplifie le problème 4.1.*

Dans ce qui suit, nous montrons que de se restreindre aux distributions quasi-uniformes est une solution naturelle au problème 4.1. D'entrée de jeu, nous avons montré dans la section 3.5.1 qu'avec ces distributions, il est possible de borner supérieurement le risque du classificateur par vote de majorité sans avoir besoin de terme de régularisation KL. Alors, selon cette théorie PAC-bayésienne, ces distributions fournissent une forme de régularisation « intégrée ».

Le prochain théorème montre que cette restriction de distribution Q ne réduit pas l'ensemble des votes de majorité possibles.

Théorème 4.3. *Pour toute distribution Q sur $\mathcal{F}$, il existe une distribution quasi-uniforme Q' sur $\mathcal{F}$ qui induit exactement le même vote de majorité que Q, ainsi que la même C-borne empirique et réelle.*

$$B_{Q'} = B_Q\,, \quad C_{Q'}^S = C_Q^S \quad \textit{et} \quad C_{Q'}^D = C_Q^D\,.$$

Démonstration. Soit Q une distribution sur $\mathcal{F} = \{f_1, \ldots, f_{2n}\}$, soit $M \triangleq \max\limits_{i \in \{1,\ldots,n\}} \left|Q(f_{i+n}) - Q(f_i)\right|$, et soit Q' une distribution sur $\mathcal{F}$ définie par

$$Q'(f_i) \;\triangleq\; \frac{1}{2n} + \frac{Q(f_i) \;-\; Q(f_{i+n})}{2nM}\,,$$

où les index de f sont définis *modulo* $2n$ (c'est-à-dire, $f_{(i+n)+n} \;= f_i$). Alors, il est facile de montrer que Q' est une distribution quasi-uniforme. De plus, pour tout exemple $x \in \mathcal{X}$,

nous avons

$$\begin{aligned}\mathop{\mathbf{E}}_{f\sim Q'} f(x) &\triangleq \sum_{i=1}^{2n} Q'(f_i) f_i(x) \;=\; \sum_{i=1}^{n} \Big(Q'(f_i) - Q'(f_{i+n})\Big) f_i(x) \\ &= \sum_{i=1}^{n} \frac{2Q(f_i) - 2Q(f_{i+n})}{2nM} f_i(x) \;=\; \frac{1}{nM} \sum_{i=1}^{2n} Q(f_i) f_i(x) \\ &= \frac{1}{nM} \mathop{\mathbf{E}}_{f\sim Q} f(x)\,.\end{aligned}$$

Comme $nM > 0$, nous avons que $B_{Q'}(x) = B_Q(x)$ pour tout $x \in \mathcal{X}$. Nous avons également que $M_{Q'}(x,y) = \frac{1}{nM} M_Q(x,y)$, ce qui implique que $\Big(\mu_1\big(M^{D'}_{Q'}\big)\Big)^2 = \Big(\frac{1}{nM}\mu_1\big(M^{D'}_{Q}\big)\Big)^2$ et $\mu_2\big(M^{D'}_{Q'}\big) = \Big(\frac{1}{nM}\Big)^2 \mu_2\big(M^{D'}_{Q}\big)$ pour $D' = D$ et $D' = S$.

Le théorème résulte alors de la définition de la C-borne. □

Le théorème 4.3 montre une propriété intéressante de la C-borne : différentes distributions Q donnant un vote de majorité équivalent donneront une même valeur de C-borne (réelle et empirique). Comme la C-borne est une borne du risque du vote de majorité, cette propriété est tout à fait appropriée.

De plus, la PAC-borne 3.23 combinée au théorème 4.3 indiquent que de se restreindre aux distributions quasi-uniformes est une solution naturelle au problème du surapprentissage (voir le problème 4.1). Malheureusement, le problème 4.2 persiste car parmi toutes les distributions a posteriori Q qui minimisent la C-borne, il existe toujours une distribution Q avec une marge empirique aussi proche de 0 que nous le désirons.

Théorème 4.4. *Pour tout $\mu \in (0,1]$ et pour toute distribution quasi-uniforme Q sur $\mathcal{F}$ ayant une marge empirique $\mu_1\big(M^S_Q\big) \geq \mu$, il existe une distribution quasi-uniforme Q' sur $\mathcal{F}$ ayant une marge empirique égale à μ, telle que Q et Q' induisent le même vote de majorité, et ont la même valeur de C-borne, empirique et réelle. C'est-à-dire,*

$$\mu_1\big(M^S_{Q'}\big) = \mu\,,\;\; B_{Q'} = B_Q\,,\;\; C^S_{Q'} = C^S_Q \;\;\textit{et}\;\; C^D_{Q'} = C^D_Q\,.$$

Démonstration. Soit Q une distribution quasi-uniforme sur $\mathcal{F} = \{f_1, \ldots, f_{2n}\}$ telle que $\mu_1\big(M^S_Q\big) \geq \mu$. Nous définissons Q' comme

$$Q'(f_i) \;\triangleq\; \frac{\mu}{\mu_1\big(M^S_Q\big)} \cdot Q(f_i) \;+\; \left(1 - \frac{\mu}{\mu_1\big(M^S_Q\big)}\right) \cdot 1/2n\,, \quad i \in \{1, \ldots, 2n\}\,.$$

Clairement, Q' est une distribution quasi-uniforme, comme elle est une combinaison convexe d'une distribution quasi-uniforme et de la distribution uniforme. Alors, de manière similaire à la démonstration du théorème 4.3, on peut facilement montrer que $\mathbf{E}_{f\sim Q'} f(x) = \frac{\mu}{\mu_1\left(M^S_Q\right)} \mathbf{E}_{f\sim Q} f(x)$, ce qui implique le résultat. □

Les bornes d'ensemble d'entraînement, c'est-à-dire les bornes qui sont uniformément valides pour tout vote de majorité Q, sont connues pour se dégrader lorsque la capacité de classification augmente (voir la section 1.3.3 pour une explication des bornes sur l'ensemble d'entraînement). Tel que montré par le théorème 4.4 pour les votes de majorité, cette capacité augmente lorsque μ s'approche de 0, puisque le nombre de votes de majorité possibles augmente. Ainsi, la PAC-borne 3.23, qui borne la C-borne réelle C_Q^D à partir de la C-borne empirique C_Q^S, va se dégrader pour les petites valeurs de μ. Cette dégradation est d'autant plus importante que C_Q^S aura une forme instable de la forme $0/0$.

Tel qu'expliqué plus haut, une manière de résoudre l'instabilité identifiée au problème 4.2 est de se restreindre aux distributions quasi-uniformes dont la marge empirique est plus grande ou égale à un certain seuil μ. Grâce au théorème 4.4, il est équivalent de se restreindre aux distributions quasi-uniformes ayant une marge empirique *exactement égale à* μ. Des théorèmes 2.13 et 4.4, il s'ensuit que *minimiser la C-borne, sous la contrainte que* $\mu_1\left(M_Q^S\right) \geq \mu$, *est équivalent à minimiser* $\mu_2\left(M_Q^S\right)$ *sous la contrainte que* $\mu_1\left(M_Q^S\right) = \mu$. De cette observation, et du fait que de minimiser la PAC-borne 3.23 est équivalent à trouver la distribution Q quasi-uniforme minimisant la C-borne empirique C_Q^S, nous pouvons maintenant définir l'algorithme MinCq.

Dans cette thèse, μ représente une telle restriction sur la marge. De plus, nous disons qu'une valeur μ est D'*-réalisable*[2] s'il existe une distribution quasi-uniforme Q telle que $\mu_1\left(M_Q^{D'}\right) = \mu$. L'algorithme proposé, appelé MinCq, résout le problème suivant.

Définition 4.5 (MinCq)**.** Étant donné un ensemble symétrique $\mathcal{F}$ de votants, un ensemble d'entraînement S et $\mu > 0$ une valeur S-réalisable. Parmi toutes les distributions quasi-uniformes Q de marge empirique $\mu_1\left(M_Q^S\right)$ égale à μ, MinCq consiste à trouver celle qui minimise $\mu_2\left(M_Q^S\right)$.

Ce problème peut être directement traduit en un programme quadratique à $2n$ variables. Également, en considérant le fait que Q est une distribution quasi-uniforme sur un ensemble symétrique de votants, on peut également traduire MinCq en un programme quadratique à n variables. La prochaine section montre comment passer de la définition de l'algorithme à ces deux versions du programme quadratique.

2. Nous savons déjà que pour toute distribution quasi-uniforme Q, il existe une distribution équivalente de marge aussi près de 0 que désiré. Pour déterminer si une valeur μ est D'-réalisable, nous n'avons donc qu'à vérifier si la valeur μ est plus petite ou égale à la plus grande marge pouvant être réalisée sur la distribution D'. La valeur de cette marge maximale est obtenue en considérant une distribution Q donnant un poids de $\frac{1}{n}$ au votant f dont la valeur $\mathbf{E}_{(x,y)\sim D'}\, y \cdot f(x)$ est la plus grande, aucun poids au votant complémentaire de f, et un poids uniforme aux autres votants.

4.2 Le programme quadratique MinCq

Dans cette section, nous construisons des *programmes quadratiques* (problèmes d'optimisation dont la fonction objectif est quadratique et les contraintes sont linéaires) résolvant le problème d'optimisation de la définition 4.5.

4.2.1 Notation et définitions

Définissons d'abord la matrice de vote **F** et le vecteur d'étiquettes **y** qui seront utilisées définir les programmes quadratiques.

Définition 4.6 (La matrice de vote)**.** Soit $S = \langle (x_1, y_1), (x_2, y_2), \dots, (x_m, y_m) \rangle$ un ensemble d'entraînement et $\mathcal{F} = \{f_1, f_2, \dots, f_{2n}\}$ un ensemble symétrique de votants. La *matrice de vote* **F** est donnée par

$$\mathbf{F} \triangleq \begin{bmatrix} f_1(x_1) & f_2(x_1) & \dots & f_{2n}(x_1) \\ f_1(x_2) & f_2(x_2) & \dots & f_{2n}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(x_m) & f_2(x_m) & \dots & f_{2n}(x_m) \end{bmatrix}.$$

Dans le cas où nous considérons un ensemble symétriques de votants binaires $\mathcal{H} = \{h_1, h_2, \dots, h_{2n}\}$, nous appellerons la matrice **H** associée la *matrice de classification*.

Définition 4.7 (Le vecteur d'étiquettes)**.** Étant donné $S = \langle (x_1, y_1), (x_2, y_2), \dots, (x_m, y_m) \rangle$ un ensemble d'entraînement, le *vecteur d'étiquettes* **y** est donnée par

$$\mathbf{y} \triangleq \begin{bmatrix} y_1 & y_2 & \dots & y_m \end{bmatrix}^\top,$$

où chaque élément y_k est soit -1 ou 1. Notons que nous avons donc que $\text{diag}(\mathbf{y})^\top \text{diag}(\mathbf{y}) = \mathbf{I}_m$.

4.2.2 Programme quadratique à $2n$ variables

Construisons premièrement le programme quadratique à $2n$ variables étant facilement déduit de la définition du problème d'optimisation de MinCq (définition 4.5).

Considérant les définitions précédentes et étant donné une valeur μ S-réalisable, MinCq est résolu par le problème d'optimisation décrit par l'algorithme 3. Notons que la formulation ci-bas est différente de celle présentée dans Laviolette, Marchand et Roy (2011) et Germain, Lacasse, Laviolette, Marchand et Roy (2015) : nous avons simplifié la notation en introduisant l'utilisation de la matrice de vote **F** de la définition 4.6. Cette nouvelle notation est celle qui a été développée dans Roy, Marchand et Laviolette (2016).

Explication de l'algorithme 3 Il s'ensuit des définitions des deux premiers moments de la marge $\mu_1\left(M_Q^S\right)$ et $\mu_2\left(M_Q^S\right)$ (voir les équations (2.7) et (2.9)) que la fonction à minimiser

Algorithme 3 Programme quadratique MinCq à $2n$ variables

$$
\begin{aligned}
\text{Résoudre} \quad & \underset{\mathbf{q}}{\operatorname{argmin}} \quad \frac{1}{m}\,\mathbf{q}^\top \mathbf{F}^\top \mathbf{F}\,\mathbf{q}\,, \\
\text{sous contraintes} \quad & \frac{1}{m}\mathbf{y}^\top \mathbf{F}\,\mathbf{q} \;=\; \mu\,, \\
& \begin{bmatrix}\mathbf{I}_n & \mathbf{I}_n\end{bmatrix}\mathbf{q} \;=\; \frac{1}{n}\mathbf{1}_n\,, \\
& \mathbf{q} \;\geq\; \mathbf{0}_{2n}\,,
\end{aligned}
$$

où $\mathbf{q}$ est un vecteur de $2n$ variables représentant les poids associés aux votants, $\mathbf{F}$ est la matrice de vote définie à la définition 4.6, et $\mathbf{y}$ est le vecteur d'étiquettes défini à la définition 4.7.

Ce programme quadratique peut être réécrit de manière standard, sous la forme proposée par Boyd et Vandenberghe (2004, section 4, équation 4.34), avec $P = \frac{2}{m}\mathbf{F}^\top\mathbf{F}$, $q = \mathbf{0}_{2n}$, $r = 0$, $G = -\mathbf{I}_{2n}$, $h = \mathbf{0}_{2n}$, $A = \begin{bmatrix}\frac{1}{m}\mathbf{y}^\top\mathbf{F} \\ \mathbf{I}_n \quad \mathbf{I}_n\end{bmatrix}$, et $b = \begin{bmatrix}\mu \\ \frac{1}{n}\mathbf{1}_n\end{bmatrix}$. Cette forme peut être utilisée pour implémenter MinCq dans un solveur tel que CVXOPT (Dahl et Vandenberghe, 2007).

correspond à $\mu_2\big(M_Q^S\big)$, et que la première contrainte d'égalité correspond à fixer le premier moment de la marge à la valeur μ :

$$
\begin{aligned}
\mu_2\big(M_Q^S\big) \;&=\; \frac{1}{m}\sum_{k=1}^{m}\left(\sum_{i=1}^{2n} q_i f_i(x_k)\right)\left(\sum_{j=1}^{2n} q_j f_j(x_k)\right) \\
&=\; \frac{1}{m}\sum_{i=1}^{2n}\sum_{j=1}^{2n} q_i q_j \sum_{k=1}^{m} f_i(x_k) f_j(x_k) \;=\; \frac{1}{m}\mathbf{q}^\top\mathbf{F}^\top\mathbf{F}\,\mathbf{q}\,,
\end{aligned}
$$

$$
\mu_1\big(M_Q^S\big) \;=\; \frac{1}{m}\sum_{k=1}^{m} y_k \sum_{i=1}^{2n} q_i f_i(x_k) \;=\; \frac{1}{m}\sum_{k=1}^{m} y_k \mathbf{F}_{k:}\,\mathbf{q} \;=\; \frac{1}{m}\mathbf{y}^\top\mathbf{F}\,\mathbf{q} \;=\; \mu\,.
$$

Les autres contraintes du programme quadratique correspondent aux contraintes nécessaires pour obtenir une distribution quasi-uniforme. En effet, nous avons

$$
\begin{bmatrix}\mathbf{I}_n & \mathbf{I}_n\end{bmatrix}\mathbf{q} \;=\; \frac{1}{n}\mathbf{1}_n \qquad\Leftrightarrow\qquad q_i + q_{i+n} \;=\; \frac{1}{n} \quad \text{pour tout } i \in [1,n]\,,
$$

et

$$
\mathbf{q} \;\geq\; \mathbf{0}_{2n} \qquad\Leftrightarrow\qquad q_i \;\geq\; 0 \quad \text{pour tout } i \in [1,2n]\,.
$$

Pour démontrer que l'algorithme 3 est un programme quadratique, il suffit de montrer que $\mathbf{F}^\top\mathbf{F}$ est une matrice semi-définie positive, ce qui est nécessairement le cas pour toute matrice $\mathbf{F}$. En effet, une matrice $\mathbf{P}$ de $n\times n$ éléments est semi-définie positive si et seulement si pour tout vecteur $\mathbf{q} \in \mathbb{R}^n$, $\mathbf{q}^\top\mathbf{P}\mathbf{q} \geq 0$. Or, si $\mathbf{P}$ peut être écrite sous la forme $\mathbf{P} = \mathbf{F}^\top\mathbf{F}$ comme dans le cas qui nous intéresse, nous avons que $\mathbf{q}^\top\mathbf{P}\mathbf{q} \;=\; \mathbf{q}^\top\mathbf{F}^\top\mathbf{F}\mathbf{q} \;=\; \left\|\mathbf{q}^\top\mathbf{F}\right\|^2$ et cette valeur est nécessairement positive ou nulle puisqu'il s'agit d'un carré.

Finalement, le vote de majorité Q-pondéré retourné par l'algorithme 3 est donné par

$$B_Q(x) \;=\; \operatorname{sgn}\left[\sum_{i=1}^{2n} q_i f_i(x)\right].$$

4.2.3 Programme quadratique à n variables

Réduisons maintenant le programme quadratique 3 à $2n$ variables en un programme quadratique à n variables. Cette simplification est rendue possible grâce au fait que l'ensemble de votants est symétrique, et que la distribution des poids est contrainte à être quasi-uniforme. Dans cette section, nous réutilisons la matrice $\mathbf{F}$ de $m \times 2n$ éléments et le vecteur $\mathbf{q}$ de $2n$ éléments définis à la section précédente, mais nous dénotons $\hat{\mathbf{F}}$ la matrice composée des n premières colonnes de la matrice $\mathbf{F}$, et $\hat{\mathbf{q}}$ le vecteur composé des n premiers éléments du vecteur $\mathbf{q}$.

Premièrement, nous allons montrer que la fonction objectif de MinCq peut être réécrite en fonction des n premières variables seulement. Soit $\mathbf{P} = \mathbf{F}^\top \mathbf{F}$ et $\hat{\mathbf{P}} = \hat{\mathbf{F}}^\top \hat{\mathbf{F}}$. Comme $\mathbf{P}$ est symétrique et que $q_{i+n} = \frac{1}{n} - q_i$ pour tout $i \in [1, n]$, nous avons

$$\begin{aligned}
\mu_2\big(M_Q^S\big) \;&=\; \sum_{i=1}^{2n}\sum_{j=1}^{2n} q_i q_j P_{ij} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} q_i q_j P_{ij} + \sum_{i=n+1}^{2n}\sum_{j=1}^{n} q_i q_j P_{ij} + \sum_{i=1}^{n}\sum_{j=n+1}^{2n} q_i q_j P_{ij} + \sum_{i=n+1}^{2n}\sum_{j=n+1}^{2n} q_i q_j P_{ij} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} q_i q_j P_{ij} - \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{1}{n} - q_i\right) q_j P_{ij} - \sum_{i=1}^{n}\sum_{j=1}^{n} q_i \left(\frac{1}{n} - q_j\right) P_{ij} \\
&\quad + \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{1}{n} - q_i\right)\left(\frac{1}{n} - q_j\right) P_{ij} \\
&= 4\sum_{i=1}^{n}\sum_{j=1}^{n} q_i q_j P_{ij} - \frac{2}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} q_j P_{ij} - \frac{2}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} q_i P_{ij} + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} P_{ij} \\
&= 4\sum_{i=1}^{n}\sum_{j=1}^{n} q_i q_j P_{ij} - \frac{4}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} q_i P_{ij} + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} P_{ij}\,.
\end{aligned}$$

Le dernier terme de cette équation étant une constante par rapport à $\mathbf{q}$, minimiser la dernière ligne de l'équation précédente revient à minimiser à un facteur 4 près l'équation

$$\sum_{i=1}^{n}\sum_{j=1}^{n} q_i q_j P_{ij} - \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} q_i P_{ij} \;=\; \hat{\mathbf{q}}^\top \hat{\mathbf{P}} \hat{\mathbf{q}} - \frac{1}{n}\mathbf{1}_n^\top \hat{\mathbf{P}} \hat{\mathbf{q}} \;=\; \frac{1}{m}\hat{\mathbf{q}}^\top \hat{\mathbf{F}}^\top \hat{\mathbf{F}} \hat{\mathbf{q}} - \frac{1}{mn}\mathbf{1}_n^\top \hat{\mathbf{F}}^\top \hat{\mathbf{F}} \hat{\mathbf{q}}\,.$$

Représentons maintenant le premier moment de la marge, encore une fois en n'utilisant que les n premiers votants. Soit $\mathbf{p} = \frac{1}{m}\mathbf{F}^\top \mathbf{y}$ et $\hat{\mathbf{p}} = \frac{1}{m}\hat{\mathbf{F}}^\top \mathbf{y}$. Notons que $p_{i+n} = -p_i$ pour tout

$i \in [1, n]$. Nous avons

$$\begin{aligned}
\mu_1\left(M_Q^S\right) &= \sum_{i=1}^{2n} q_i\, p_i \\
&= \sum_{i=1}^{n} q_i\, p_i + \sum_{i=1}^{n} \left(\frac{1}{n} - q_i\right) p_{i+n} \\
&= \sum_{i=1}^{n} q_i\, p_i - \sum_{i=1}^{n} \left(\frac{1}{n} - q_i\right) p_i \\
&= 2\sum_{i=1}^{n} q_i\, p_i - \frac{1}{n}\sum_{i=1}^{n} p_i \\
&= 2\hat{\mathbf{p}}^\top \hat{\mathbf{q}} - \frac{1}{n}\hat{\mathbf{p}}^\top \mathbf{1}_n \\
&= \frac{2}{m}\mathbf{y}^\top \hat{\mathbf{F}}\, \hat{\mathbf{q}} - \frac{1}{mn}\mathbf{y}^\top \hat{\mathbf{F}}\, \mathbf{1}_n\,.
\end{aligned}$$

Finalement, notons que cette formulation du problème ne nécessite pas de contrainte d'égalité forçant la somme des poids de deux votants complémentaires à être égale à $\frac{1}{n}$, étant donné que les poids des compléments ne font pas partie du problème d'optimisation. Cependant, il faut forcer les poids des n premiers votants à être positifs et au plus $\frac{1}{n}$, afin que les poids complémentaires donnés par la contrainte de quasi-uniformité soient également positifs. Une fois les poids des votants complémentaires calculés, la somme de tous les poids sera nécessairement égale à 1, et nous obtiendrons une distribution Q sur les $2n$ votants tel que voulu.

Étant donné tout ce qui précède, nous obtenons le programme quadratique suivant.

Algorithme 4 Programme quadratique MinCq à n variables

Résoudre	$\underset{\hat{\mathbf{q}}}{argmin}$	$\frac{1}{m}\hat{\mathbf{q}}^\top \hat{\mathbf{F}}^\top \hat{\mathbf{F}}\hat{\mathbf{q}} - \frac{1}{mn}\mathbf{1}_n^\top \hat{\mathbf{F}}^\top \hat{\mathbf{F}}\hat{\mathbf{q}}$
sous contraintes		$\frac{2}{m}\mathbf{y}^\top \hat{\mathbf{F}}\, \hat{\mathbf{q}} \;=\; \mu + \frac{1}{mn}\mathbf{y}^\top \hat{\mathbf{F}}\, \mathbf{1}_n\,,$
		$\mathbf{0}_n \preceq\; \mathbf{q} \;\preceq\; \frac{1}{n}\mathbf{1}_n\,,$

où $\hat{\mathbf{q}}$ est un vecteur de n variables représentant les poids des n premiers votants, et $\hat{\mathbf{F}}$ correspond aux n premières colonnes de la matrice de vote associée à l'ensemble d'entraînement S et l'ensemble symétrique de votants $\mathcal{F}$.

Ce programme quadratique peut être réécrit de manière standard, sous la forme proposée par Boyd et Vandenberghe (2004, section 4, équation 4.34), avec $P = \frac{2}{m}\hat{\mathbf{F}}^\top \hat{\mathbf{F}}$, $q = -\frac{1}{mn}\hat{\mathbf{F}}^\top \hat{\mathbf{F}}\mathbf{1}_n$, $r = 0$, $G = \begin{bmatrix} -\mathbf{I}_n \\ \mathbf{I}_n \end{bmatrix}$, $h = \begin{bmatrix} \mathbf{0}_n \\ \frac{1}{n}\mathbf{1}_n \end{bmatrix}$, $A = \frac{2}{m}\mathbf{y}^\top \hat{\mathbf{F}}$, et $b = \mu + \frac{1}{mn}\mathbf{y}^\top \hat{\mathbf{F}}\, \mathbf{1}_n$.

Finalement, le vote de majorité Q-pondéré retourné par l'algorithme 4 est

$$B_Q(x) \;=\; \text{sgn}\left[\sum_{i=1}^{n}\left(2\,q_i - \frac{1}{n}\right)f_i(x)\right].$$

4.3 Expérimentations

Cette section présente des expérimentations qui ont été menées sur plusieurs ensembles de données et dans divers contextes, pour comparer MinCq aux algorithmes états de l'art SVM et AdaBoost. MinCq y est comparé, en fonction du type de votants utilisé, à l'algorithme le plus pertinent. Les ensembles de données utilisés dans ces expérimentations sont décrits davantage en annexe B. Des informations supplémentaires reliées aux résultats des différents algorithmes sur les différents ensembles de données peuvent également y être retrouvés.

4.3.1 Comparaison de MinCq aux algorithmes de l'état de l'art

Nous comparons maintenant MinCq (algorithme 4) aux algorithmes de l'état de l'art dans trois différents contextes : la *reconnaissance de caractères manuscrits*, les *tâches de classification binaire classiques* et *l'analyse du sentiment de critiques sur Amazon*. Un *contexte* (Lacoste, Laviolette et Marchand, 2012) représente une distribution sur différentes tâches qu'un algorithme d'apprentissage peut avoir à résoudre, et un échantillon d'un contexte est une collection d'ensembles de données.

Pour chaque contexte, chaque ensemble de données est séparé aléatoirement en deux parties : un ensemble d'entraînement S d'au plus 500 exemples[3] et un ensemble de test T contenant le reste des exemples. Lorsque des valeurs d'hyperparamètres doivent être choisies pour un algorithme, la validation croisée à 5 plis est exécutée sur l'ensemble d'entraînement S. Les valeurs d'hyperparamètres minimisant le risque moyen de validation croisée sont choisies. En utilisant ces valeurs, l'algorithme est entraîné sur la totalité de l'ensemble d'entraînement S, et sa performance est ensuite évaluée sur l'ensemble de test T (voir l'algorithme 2). De l'information supplémentaire sur les contextes et les ensembles de données peut être retrouvée en annexe B.

Pour les deux premiers contextes, nous comparons quatre algorithmes : MinCq utilisant des souches de décision comme votants (dénoté MinCq^S), MinCq utilisant des *noyaux RBF (radial basis functions)* $k(x,x') \;=\; exp\left(-\gamma\|x-x'\|^2\right)$ comme votants (dénoté MinCq$^K_\gamma$), AdaBoost (Freund et Schapire, 1997) utilisant des souches de décision (dénoté AdaBoost^S), et les *Support Vector Machines* (*SVMs*) (Cortes et Vapnik, 1995) avec des noyaux RBF, dénoté SVM$^K_\gamma$. Pour le dernier contexte, nous comparons MinCq en utilisant des noyaux linéaires $k(x,x') = x \cdot x'$ comme votants (dénoté MinCq$^K_\ell$), et SVM utilisant ce même noyau, dénoté SVM$^K_\ell$.

3. Le choix de sélectionner un maximum de 500 exemples a été fait pour accélérer l'entraînement des algorithmes. Les résultats empiriques présentés dans Laviolette, Marchand et Roy (2011) sont similaires et aucune limite de nombre d'exemple d'entraînement n'a été appliquée.

Pour les trois variantes de MinCq, le programme quadratique est résolu en utilisant CVXOPT (Dahl et Vandenberghe, 2007), un solveur de problèmes d'optimisation convexe. Des expérimentations ont également été faites avec le solveur *SLSQP* disponible dans la librairie *Scipy* (Jones, Oliphant, Peterson et al., 2001), offrant des performances similaire mais dont le temps d'exécution est plus lent en pratique. CVXOPT a donc été retenu pour les expérimentations.

AdaBoost^S Pour AdaBoost^S, nous utilisons des souches de décision comme votants. Pour chaque attribut, 10 souches de décision (et leur complément) sont générées, pour un total de 20 souches de décision par attribut. Le nombre d'itérations de boosting sont choisies parmi 15 valeurs entre 10^2 et 10^6.

MinCq^S Pour MinCq^S, nous utilisons les 10 mêmes souches de décision par attribut que pour AdaBoost^S. Notons que nous n'avons pas à considérer les compléments, car MinCq considère automatiquement les ensembles symétriques de votants. L'hyperparamètre μ de MinCq est choisi parmi 15 valeurs entre 10^{-2} et $10^{-0.5}$ sur une échelle logarithmique.

SVM$^K_\gamma$ L'hyperparamètre γ du noyau RBF et l'hyperparamètre C du SVM sont choisis parmi 15 valeurs entre 10^{-4} et 10^1 pour γ, et parmi 15 valeurs entre 10^{-4} et 10^4 pour C, tous deux sur une échelle logarithmique.

MinCq$^K_\gamma$ Pour MinCq$^K_\gamma$, nous considérons 15 valeurs de μ parmi 10^{-4} et 10^{-2} sur une échelle logarithmique, et les 15 mêmes valeurs de γ que SVM$^K_\gamma$ pour le noyau RBF.

Dans les expérimentations avec les algorithmes ci-haut, chaque ensemble de données est normalisé en utilisant une tangente hyperbolique[4]. Pour chaque exemples x, chaque attribut $x_1, x_2, \ldots, x_n$ est normalisé avec $x'_i = tanh\left[\frac{x_i - \overline{x_i}}{\sigma_i}\right]$, où $\overline{x_i}$ et σ_i sont respectivement la moyenne et l'écart-type du $i^{\text{ème}}$ attribut, calculé sur l'ensemble d'entraînement S. Comme les algorithmes ci-bas sont exécutés sur des ensembles de données déjà normalisées à l'aide d'une pondération *TF-IDF* (Salton et McGill, 1986 ; Dumais et al., 1998), aucune normalisation supplémentaire n'a été effectuée.

SVM$^K_\ell$ Lorsque nous utilisons le noyau linéaire, le paramètre C du SVM est choisi parmi 15 valeurs entre 10^{-4} et 10^4, sur une échelle logarithmique. Pour toutes les expérimentations avec le SVM, l'implémentation de Pedregosa et al. (2011) a été utilisée.

MinCq$^K_\ell$ Pour MinCq$^K_\ell$, nous considérons 15 valeurs de μ entre 10^{-4} et 10^{-2} sur une échelle logarithmique.

4. Normaliser les attributs est une étape de prétraitement commune avant d'exécuter un algorithme d'apprentissage. L'utilisation de la tangente hyperbolique comme fonction de normalisation est un choix traditionnel provenant d'articles et expérimentations précédents dans notre laboratoire de recherche, notamment dans Germain, Lacasse, Laviolette, Marchand et Roy (2015).

Contexte de reconnaissance de caractères manuscrits

Un premier contexte d'intérêt pour comparer MinCq avec d'autres algorithmes d'apprentissage est la reconnaissance de caractères manuscrits. Pour cette tâche, nous utilisons l'ensemble de données *MNIST database of handwritten digits* (Lecun et Cortes, 2009), un ensemble de données classique sur lequel se comparent de nombreux algorithmes d'apprentissage. Cet ensemble contenant 10 classes, nous séparons d'abord l'ensemble de données original en 45 tâches de classification binaire, où l'union de tous les ensembles de données binaires retrouve l'ensemble de données original, et où l'intersection entre toute paire d'ensembles de données donne l'ensemble vide. Ainsi, tout exemple de l'ensemble de données original se retrouve dans un et un seul ensemble de données binaires, évitant ainsi d'obtenir une corrélation entre les différents ensembles. Pour chaque ensemble résultant, nous choisissons aléatoirement un ensemble d'entraînement S de 500 exemples, et l'ensemble de test T contient tous les exemples restants. La table 4.1 montre les risques sur l'ensemble de test pour chaque ensemble de données binaire et chaque algorithme, ainsi que deux tests statistiques comparant MinCq aux autres algorithmes. Les tests statistiques utilisés sont le «*Poisson binomial test*» (Lacoste, Laviolette et Marchand, 2012) et le «*sign test*» (Mendenhall, 1983), dont une brève explication est fournie dans la description de la table. Ces deux méthodes suggèrent que sur ce contexte, MinCq$_\gamma^K$ a une performance similaire à SVM$_\gamma^K$, mais que MinCq^S surpasse AdaBoost^S. La figure 4.1 montre à nouveau les risques de test mais de manière plus visuelle, cette fois-ci avec un nuage de points.

Contexte des tâches classiques de classification binaire

Ce second contexte d'intérêt est plus général : il regroupe de multiples ensembles de données de classification binaire, provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (Lichman, 2013). Ces ensembles de données sont couramment utilisés pour évaluer la performance d'algorithmes d'apprentissage, et permettent de vérifier la performance d'un algorithme d'apprentissage sur plusieurs tâches diverses. Pour chaque ensemble de données, la moitié des exemples (jusqu'à un maximum de 500) est choisie aléatoirement et utilisée comme ensemble d'apprentissage S. L'ensemble de test T contient le reste des exemples. La table 4.2 montre les risques de test résultants dans ce contexte, pour chaque algorithme. La figure 4.2 présente les mêmes résultats de manière plus visuelle.

La table 4.2 montre également le résultat d'une comparaison statistique sur ce contexte, en utilisant le « Poisson binomial test » et le « sign test ». Dans ce contexte, les deux tests statistiques ne montrent aucune différence statistique de performance ni entre MinCq$_\gamma^K$ et SVM$_\gamma^K$, ni entre MinCq^S et AdaBoost^S, impliquant que ces paires d'algorithmes ont une performance similaire dans un contexte plus général.

Ensemble de données	MinCq^K_γ	SVM^K_γ	MinCq^S	AdaBoost^S
MNIST 0-vs-1	0.0035	**0.0018**	**0.0035**	0.0070
MNIST 0-vs-2	0.0057	**0.0048**	0.0201	**0.0182**
MNIST 0-vs-3	**0.0047**	0.0057	**0.0142**	**0.0142**
MNIST 0-vs-4	**0.0049**	0.0078	0.0098	**0.0078**
MNIST 0-vs-5	**0.0072**	0.0093	**0.0155**	0.0207
MNIST 0-vs-6	0.0252	**0.0107**	**0.0194**	**0.0194**
MNIST 0-vs-7	**0.0065**	0.0074	0.0139	**0.0084**
MNIST 0-vs-8	**0.0107**	0.0146	0.0224	**0.0205**
MNIST 0-vs-9	**0.0067**	0.0096	0.0183	**0.0115**
MNIST 1-vs-2	0.0226	**0.0087**	**0.0243**	**0.0243**
MNIST 1-vs-3	0.0222	**0.0120**	**0.0137**	**0.0137**
MNIST 1-vs-4	**0.0044**	**0.0044**	**0.0062**	0.0097
MNIST 1-vs-5	**0.0093**	0.0121	**0.0204**	0.0260
MNIST 1-vs-6	**0.0053**	0.0070	**0.0061**	0.0079
MNIST 1-vs-7	**0.0076**	0.0084	**0.0126**	0.0185
MNIST 1-vs-8	**0.0053**	0.0062	0.0282	**0.0265**
MNIST 1-vs-9	**0.0035**	0.0061	**0.0026**	0.0061
MNIST 2-vs-3	0.0206	**0.0159**	**0.0374**	0.0458
MNIST 2-vs-4	**0.0019**	0.0068	**0.0145**	0.0280
MNIST 2-vs-5	0.0123	**0.0102**	0.0389	**0.0287**
MNIST 2-vs-6	0.0154	**0.0115**	**0.0375**	**0.0375**
MNIST 2-vs-7	0.0258	**0.0129**	0.0322	**0.0258**
MNIST 2-vs-8	**0.0126**	0.0145	**0.0406**	0.0522
MNIST 2-vs-9	0.0086	**0.0076**	**0.0162**	0.0190
MNIST 3-vs-4	0.0095	**0.0086**	**0.0133**	0.0190
MNIST 3-vs-5	**0.0302**	0.0312	0.0784	**0.0754**
MNIST 3-vs-6	**0.0028**	0.0047	**0.0151**	0.0189
MNIST 3-vs-7	0.0154	**0.0145**	**0.0172**	0.0236
MNIST 3-vs-8	0.0295	**0.0257**	**0.0428**	0.0485
MNIST 3-vs-9	**0.0094**	0.0113	**0.0178**	0.0310
MNIST 4-vs-5	**0.0073**	0.0094	**0.0188**	0.0229
MNIST 4-vs-6	**0.0059**	**0.0059**	**0.0137**	0.0225
MNIST 4-vs-7	0.0131	**0.0112**	**0.0262**	0.0299
MNIST 4-vs-8	**0.0069**	0.0079	**0.0177**	0.0187
MNIST 4-vs-9	0.0242	**0.0233**	**0.0562**	0.0620
MNIST 5-vs-6	**0.0114**	0.0176	**0.0290**	0.0342
MNIST 5-vs-7	**0.0049**	0.0079	**0.0138**	0.0188
MNIST 5-vs-8	**0.0198**	0.0208	**0.0521**	0.0552
MNIST 5-vs-9	**0.0103**	0.0144	**0.0195**	0.0318
MNIST 6-vs-7	0.0019	**0.0009**	**0.0028**	0.0065
MNIST 6-vs-8	**0.0137**	0.0166	**0.0186**	0.0254
MNIST 6-vs-9	**0.0000**	0.0029	**0.0058**	0.0068
MNIST 7-vs-8	**0.0093**	**0.0093**	**0.0140**	0.0196
MNIST 7-vs-9	0.0517	**0.0332**	**0.0683**	0.0720
MNIST 8-vs-9	0.0262	**0.0252**	**0.0330**	0.0436

Comparaison statistique		
	MinCq^K_γ vs SVM^K_γ	MinCq^S vs AdaBoost^S
Poisson binomial test	66%	100%
Sign test (valeur-p)	0.22	0.00

TAB. 4.1: Risque sur les ensembles de test pour chaque algorithme et chaque ensemble de classification binaire tiré de MNIST, où *MNIST x-vs-y* indique que les algorithmes doivent distinguer le chiffre x du chiffre y. Pour chaque paire d'algorithmes utilisant les mêmes votants, le meilleur risque est indiqué en gras. La table donne également le résultat de tests statistiques. Le « Poisson binomial test » donne la probabilité que MinCq ait une meilleure performance que l'autre algorithme dans ce contexte. Le « sign test » donne une valeur-p représentant la probabilité que MinCq et l'autre algorithme ont la même performance.

Ensemble de données	MinCq^K_γ	SVM^K_γ	MinCq^S	AdaBoost^S
australian	0.1391	**0.1333**	**0.1449**	0.1942
balance	0.0543	**0.0319**	0.0288	**0.0256**
breast	0.0401	**0.0372**	**0.0372**	0.0430
bupa	**0.2543**	0.2775	0.4509	**0.3006**
car	0.0676	**0.0301**	0.1409	**0.1336**
cmc	**0.3032**	0.3207	**0.3052**	0.3176
credit	**0.1217**	0.1246	**0.1275**	0.1681
cylinder	**0.2185**	0.2704	0.2889	**0.2852**
ecoli	**0.0714**	0.0893	**0.0774**	0.0833
glass	0.2243	**0.2056**	0.2710	**0.2056**
heart	0.1556	**0.1481**	**0.1704**	0.2148
hepatitis	**0.1558**	0.2078	**0.1688**	0.1948
horse	0.1902	**0.1685**	0.2228	**0.1957**
ionosphere	0.0971	**0.0686**	0.1314	**0.1200**
monks	0.2454	**0.2130**	**0.2361**	0.2546
mushroom	0.0102	**0.0059**	**0.0042**	0.0043
optdigits	0.0343	**0.0340**	0.0849	**0.0834**
pima	**0.2448**	0.2552	**0.2422**	0.2708
titanic	0.2222	**0.2146**	**0.2116**	0.2222
vote	**0.0461**	0.0507	**0.0507**	**0.0507**
wine	**0.0449**	0.0562	0.0562	**0.0449**
yeast	0.2876	**0.2815**	0.3028	**0.2988**
zoo	**0.0392**	0.0588	**0.0392**	0.1176

Comparaison statistique		
	MinCq^K_γ vs SVM^K_γ	MinCq^S vs AdaBoost^S
Poisson binomial test	35%	74%
Sign test (valeur-p)	0.80	0.42

TAB. 4.2: Risque sur les ensembles test pour chaque algorithme dans le contexte des tâches de classification binaire classiques. Voir la table 4.1 pour une explication des tests statistiques.

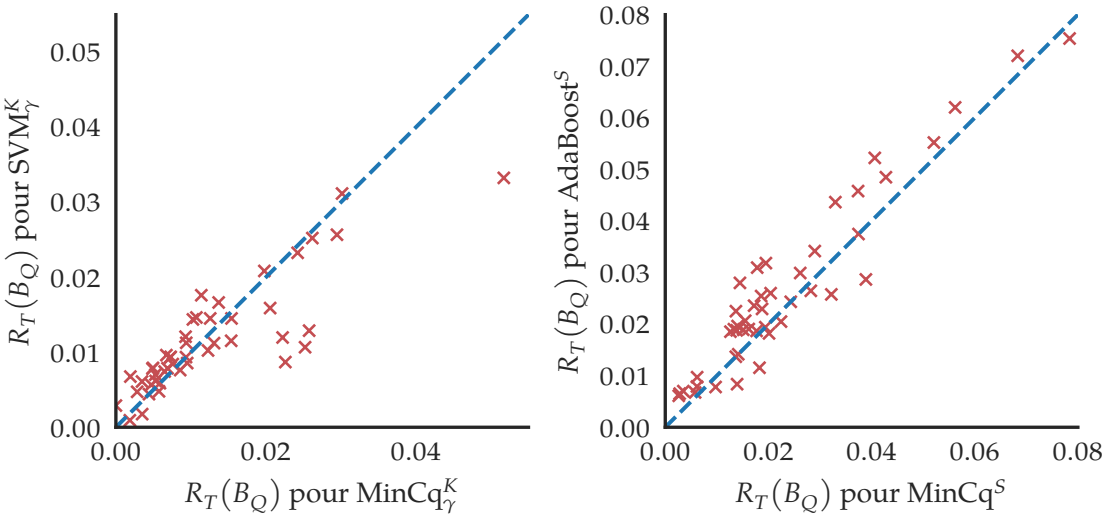

FIG. 4.1: Comparaison des risques sur l'ensemble de test pour chaque algorithme et chaque ensemble de classification binaire tiré de MNIST, représenté de manière plus visuelle. La figure de gauche montre une comparaison entre MinCq_γ^K (axe des x) et SVM_γ^K (axe des y). La figure de droite compare MinCq^S (axe des x) et AdaBoost^S (axe des y). Pour chacun des nuages de points, un point représente une paire de risques pour un ensemble de données binaires particulier. Un point au dessus de la ligne diagonale indique une meilleure performance de MinCq.

Contexte de l'analyse du sentiment de critiques sur Amazon

Ce contexte contient 4 ensembles de données d'analyse du sentiment, représentant des types de produits : *books* (livres), *DVDs*, *electronics* (électronique) et *kitchen appliances* (appareils de cuisine). La tâche est d'apprendre sur les critiques en langage naturel des usagers d'*Amazon.com*, et d'en prédire la *polarité*, qui est soit négative (3 étoiles ou moins) ou positive (4 ou 5 étoiles). Les ensembles de données proviennent de BLITZER, DREDZE et PEREIRA (2007), où les critiques ont déjà été converties en ensembles d'*unigrammes* et de *bigrammes* de termes, avec pour chacun le nombre d'occurrences.

Pour chaque ensemble de données, un ensemble d'entraînement de 1000 critiques positives et 1000 critiques négatives est fourni. Les critiques restantes forment l'ensemble de test. L'espace des attributs original de ces ensembles de données est entre 90 000 et 200 000 dimensions. Par contre, comme la plupart des unigrammes et bigrammes ne sont pas significatifs et pour diminuer la dimensionnalité, nous ne considérons que les unigrammes et bigrammes apparaissant au moins 10 fois dans l'ensemble d'entraînement (comme dans CHEN, WEINBERGER et BLITZER (2011)), réduisant ainsi le nombre de dimensions à entre 3500 et 6500. Également comme dans CHEN, WEINBERGER et BLITZER (2011), nous appliquons une pondération *TF-IDF* (SALTON et MCGILL, 1986 ; DUMAIS et al., 1998), une pratique commune en traitement

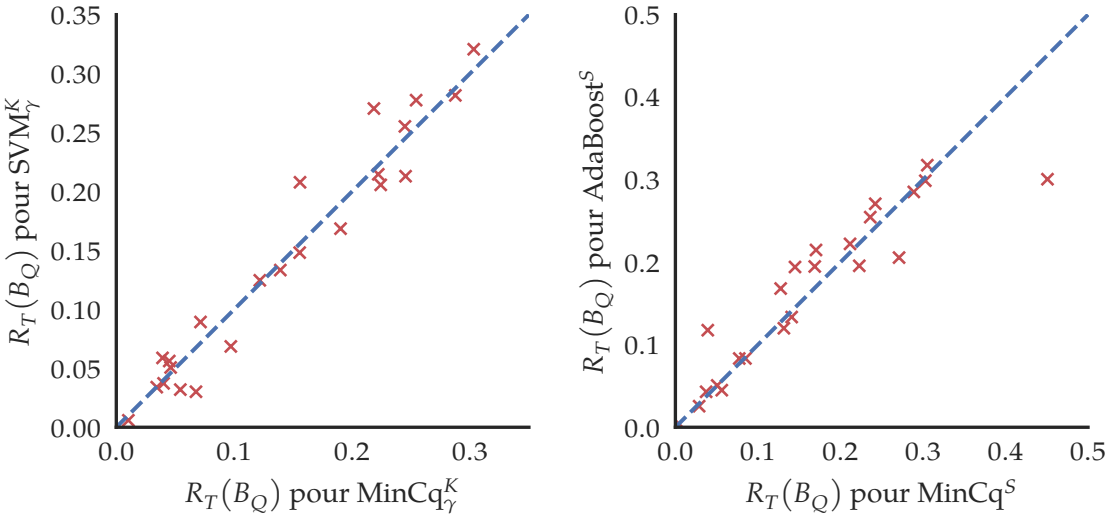

FIG. 4.2: Comparaison des risques sur l'ensemble de test pour chaque algorithme et chaque ensemble de données dans le contexte des tâches classiques de classification binaire, représenté de manière plus visuelle. La figure de gauche montre une comparaison entre MinCq_γ^K (axe des x) et SVM_γ^K (axe des y). La figure de droite compare MinCq^S (axe des x) et AdaBoost^S (axe des y). Pour chacun des nuages de points, un point représente une paire de risques pour un ensemble de données binaires particulier. Un point au dessus de la ligne diagonale indique une meilleure performance de MinCq.

des langues naturelles. La table 4.3 montre les risques de test résultants pour chaque algorithme. La figure 4.3 présente les mêmes résultats de manière plus visuelle.

La table 4.3 montre également une comparaison statistique des algorithmes dans ce contexte, encore une fois à l'aide du « Poisson binomial test » et du « sign test ». MinCq_ℓ^K a un léger avantage sur SVM_ℓ^K, comme l'algorithme gagne sur chaque ensemble de données. Les deux tests statistiques montrent également un léger avantage à MinCq par rapport à SVM.

Ces expérimentations montrent que de minimiser la C-borne, et ainsi favoriser les votes de majorité pour qui les votants sont maximalement décorrélés, est une approche intéressante. MinCq est très compétitif avec AdaBoost et le SVM sur des tâches classiques de classification binaire, ainsi qu'en analyse du sentiment de critiques sur Amazon. MinCq montre également un gain de performance par rapport aux algorithmes de l'état de l'art dans le contexte de la reconnaissance de caractères manuscrits[5]. Ceci implique que pour certaines tâches, la minimisation de la C-borne offre une performance de l'état de l'art.

Ceci dit, pour toutes les expérimentations ci-haut, nous observons que la valeur empirique

5. Notons cependant que dans ce contexte, les réseaux de neurones profonds sont particulièrement efficaces, bien que nous ne comparons pas à ces derniers dans cette thèse.

Ensemble de données	MinCq_ℓ^K	SVM_ℓ^K
Books	**0.1566**	0.1601
DVD	**0.1623**	0.1670
Electronics	**0.1320**	0.1347
Kitchen	**0.1156**	0.1194

Comparaison statistique	
	MinCq_ℓ^K vs SVM_ℓ^K
Poisson binomial test	88%
Sign test (valeur-p)	0.06

TAB. 4.3: Risque sur les ensembles test pour chaque algorithme dans le contexte de l'analyse du sentiment de critiques sur Amazon. Voir la table 4.1 pour une explication des tests statistiques.

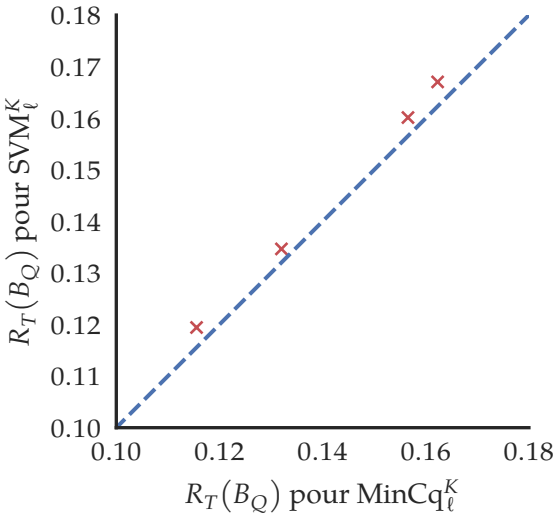

FIG. 4.3: Comparaison des risques sur l'ensemble de test pour chaque algorithme et chaque ensemble de données d'analyse du sentiment de critiques sur Amazon, représenté de manière plus visuelle. La figure montre une comparaison entre MinCq_ℓ^K (axe des x) et SVM_ℓ^K (axe des y). Sur ce nuage de points, un point représente une paire de risques pour un ensemble de données particulier. Un point au dessus de la ligne diagonale indique une meilleure performance de MinCq.

des différentes PAC-bornes est triviale (près de 1). Nous rappelons que, inspiré des PAC-bornes 3.23 et 3.30, l'algorithme MinCq apprend les poids d'un vote de majorité en minimisant le second moment de la marge, tout en fixant le premier moment à une certaine valeur μ. Dans nos expérimentations, la valeur de μ choisie par validation croisée est toujours très proche de 0 (typiquement, $\mu = 10^{-4}$). Ceci implique que $C_Q^S = 1 - \mu^2/\mu_2\left(M_Q^S\right)$ est très proche d'une forme $1 - \frac{0}{0}$, menant à la dégradation des bornes PAC-bayésiennes pour C_Q^D. Notons que pour toutes les expérimentations précédentes, les votants considérés étaient *faibles*. Ceux-ci ne permettant pas au vote de majorité d'avoir une grande marge, il est naturel que de la validation croisée ait choisi de très petites valeurs de μ. Dans la prochaine section, nous explorons l'utilisation de votants plus *forts* dans le but de comparer le comportement des bornes PAC-bayésiennes pour C_Q^D.

4.3.2 Expérimentations avec des votants forts

Dans la prochaine expérimentation, nous montrons comment obtenir de meilleures bornes en utilisant des votants plus *forts*, c'est-à-dire des votants avec une meilleure performance individuelle. Pour y arriver, plutôt que de considérer des souches de décision, nous considérons des arbres de décision.[6] Nous utilisons 100 arbres de décision générés avec l'implémentation de *Scikit-Learn* (Pedregosa et al., 2011). Nous choisissons une profondeur de 10 et un nombre d'attributs par noeud de 1. En utilisant ces votants forts, il est possible de considérer des valeurs de μ beaucoup plus élevées.[7]

La figure 4.4 montre la valeur empirique de la C-borne ainsi que les valeurs de bornes PAC-bayésiennes correspondantes, pour plusieurs valeurs de μ, sur l'ensemble de données *Mushroom*. Des 8124 exemples, 500 ont été réservés pour construire les arbres de décisions, 4062 ont été utilisés pour l'ensemble d'entraînement, et le reste des exemples ont été utilisés pour l'ensemble de test. La figure 4.4 montre que les bornes PAC-bayésiennes deviennent plus serrées lorsque μ augmente. Notons par contre que la C-borne empirique augmente de 0.001 à 0.016. Le risque sur l'ensemble de test (absent sur la figure) montre un comportement légèrement similaire à la C-borne empirique : il est nul pour la plupart des valeurs de μ, mais augmente légèrement pour les plus grandes valeurs (tout en restant en dessous de 0.001).

Nous obtenons donc des bornes serrées pour de grandes valeurs de μ: la PAC-borne 3.13 atteint une valeur de près de 0.2. Nous remarquons également que la PAC-borne 3.23 donne des valeurs qui ne sont que très légèrement plus faibles que la PAC-borne 3.13. Ce comportement est est dû au fait que MinCq retourne un vote de majorité quasi-uniforme dont la valeur de $\mathrm{KL}(Q \,\|\, P)$ est très faible. Il semble que les bornes PAC-bayésiennes ne sont pas assez serrées pour guider précisément la sélection d'une valeur de μ, ce qui nous suggère de continuer

6. Une souche de décision est simplement un arbre de décision de profondeur 1.

7. Notons que les arbres de décision ont été entraînés en utilisant un ensemble d'exemples disjoint de celui utilisé pour l'entraînement. De ce fait, toutes les PAC-bornes calculées sont valides même si elles ne sont pas construites pour permettre l'utilisation de *votants comprimés*.

l'utilisation de la validation croisée pour sélectionner une bonne valeur pour μ. Notons finalement que dans Germain, Lacasse, Laviolette, Marchand et Roy (2015), nous présentons ce graphique pour des bornes supplémentaires qui sont encore plus serrées et atteignent des valeurs sous 0.1 pour cette même expérimentation. Le comportement de ces bornes est par contre le même que pour la borne 3.13.

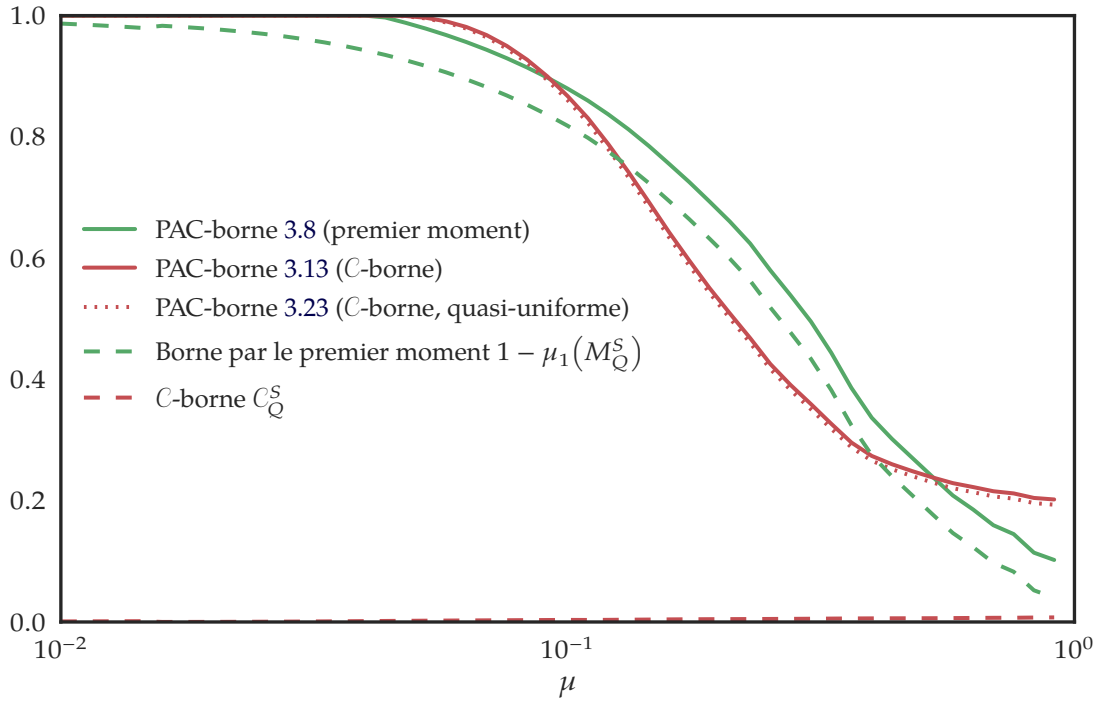

Fig. 4.4: Valeurs empiriques de la C-borne et des bornes PAC-bayésiennes correspondantes, les PAC-bornes 3.8, 3.13, et 3.23, sur les votes de majorité produits par MinCq entraîné sur des arbres de décision, pour plusieurs valeurs de μ.

4.4 Conclusion du chapitre

Nous avons introduit un algorithme d'apprentissage, nommé MinCq, qui

- peut être écrit sous la forme d'un programme quadratique ;
- trouve un vote de majorité minimisant la C-borne empirique, en minimisant le second moment de la marge, tout en contrôlant la valeur du premier moment à l'aide d'un hyperparamètre μ ;
- a été appliqué à des souches de décision, à des fonctions noyaux et à des arbres de décision ;

- est restreint aux distributions Q quasi-uniformes, et ainsi obtient une garantie de généralisation via les PAC-bornes 3.23 et 3.30 ;
- est très compétitif avec les algorithmes de l'état de l'art.

MinCq a par contre certains désavantages : d'une part, il retourne une distribution Q qui est *dense*, dans le sens où un poids non nul est attribué à la plupart des votants, et il ne peut considérer que les ensembles finis de classificateurs. Étant donné m le nombre d'exemples d'entraînement et n le nombre de votants considérés, la complexité algorithmique de MinCq est en $O\left(m \times n^2\right)$ pour la construction de la matrice $\mathbf{F}^\top\mathbf{F}$, puis en $O\left(n^3\right)$ pour la résolution du programme quadratique. MinCq n'est donc pas adapté lorsque le nombre de votants à considérer est très grand, voire infini.

Dans le prochain chapitre, nous introduisons CqBoost, un algorithme d'apprentissage itératif également inspiré de la C-borne et des bornes de généralisation PAC-bayésiennes, mais mieux adapté à la situation où le nombre de votants est très grand.

Chapitre 5

CqBoost : un algorithme itératif minimisant la $\mathcal{C}$-borne

L'algorithme MinCq présenté au chapitre 4 est très simple, compétitif avec l'état de l'art, et justifié théoriquement par des bornes de généralisation. Par contre, il présente quelques désavantages : il ne peut considérer qu'un nombre fini de votants, et comme sa complexité algorithmique est cubique en nombre de votants considérés, il est limité à cet égard. De plus, MinCq retourne un vote de majorité *dense* sur l'ensemble des votants, c'est-à-dire qu'il affecte un poids non nul à la plupart d'entre eux.

Dans ce chapitre, nous faisons un premier pas vers un algorithme mieux adapté au *traitement des données massives*. Nous introduisons CqBoost, un algorithme d'apprentissage inspiré par la minimisation d'une borne de généralisation, qui cette fois-ci est itératif. À chaque itération, CqBoost choisit un votant à ajouter au vote de majorité. L'ensemble de votants considérés peut alors être très grand, voire infini. CqBoost a également la propriété intéressante de retourner un vote de majorité *parcimonieux* sur l'ensemble des votants considérés : seulement un petit sous-ensemble de ceux-ci auront un poids non nul. Cette propriété permet d'obtenir un classificateur qui sera plus rapide d'exécution, et si le problème le permet, pourrait même être plus facile à comprendre par un être humain. CqBoost a été publié dans Roy, Marchand et Laviolette (2016), où pour la première fois dans nos publications nous avons dérogé de l'ordre alphabétique. Notons que les domaines plus théoriques conservent généralement la tendance provenant des mathématiques de publier en ordre alphabétique. En apprentissage automatique, de plus en plus d'auteurs énumèrent les auteurs en ordre de contribution. Nous suivons maintenant la tendance pour lever l'ambiguïté sur les contributions de chacun.

Nous verrons que contrairement à MinCq qui ajoute des contraintes sur la famille de votes de majorité possibles et en tire des garanties de généralisation, CqBoost ne minimise pas directement une borne sur le risque. Les restrictions appliquées à MinCq sont levées, ce qui a comme désavantage de permettre à CqBoost de faire du surapprentissage. Le nombre d'ité-

rations sera donc un paramètre important à considérer, tout comme d'autres algorithmes itératifs comme AdaBoost. Nous fournirons également des explications de ce phénomène basées sur les PAC-bornes 3.13 et 3.29.

Nous introduisons d'abord des notions d'optimisation et de dualité. Ces notions sont nécessaires à la compréhension des étapes préliminaires à la construction de CqBoost. Nous présentons ensuite la *génération de colonnes*, une technique d'optimisation sur laquelle est basée CqBoost. Nous construisons ensuite l'algorithme, explorons les similarités avec d'autres algorithmes basés sur la génération de colonnes, puis effectuons des expérimentations empiriques. Ces expérimentations montrent que CqBoost performe aussi bien que les algorithmes de l'état de l'art, et retourne des solutions beaucoup plus parcimonieuses que MinCq.

5.1 Optimisation et dualité

Lorsqu'on a à résoudre un problème d'optimisation sous contraintes, il est commun d'étudier ce problème sous diverses formes afin d'en comprendre les propriétés. Le problème original est appelé le *problème primal*, et plusieurs techniques permettent d'obtenir ce qu'on appelle un *problème dual*, un problème d'optimisation relié au problème primal, dont les variables originales deviennent des contraintes, et les contraintes originales deviennent des variables.

Notons que nous ne ferons pas dans cette thèse une revue complète ni une explication en détails de la théorie de l'optimisation. En effet, ce sujet est très vaste, et nous ne présentons que les parties qui nous seront utiles pour la suite. Voir Boyd et Vandenberghe (2004) pour une revue plus complète.

5.1.1 Définitions de base

Énonçons d'abord un problème d'optimisation très général.

$$\begin{array}{lll} \text{Résoudre} & \min\limits_{\mathbf{w}} & f_0(\mathbf{w}) \\ \text{sous contraintes} & f_i(\mathbf{w}) & \leq 0, \quad i = 1, \dots, m \\ & h_i(\mathbf{w}) & = 0, \quad i = 1, \dots, p\,, \end{array} \tag{5.1}$$

où $\mathbf{w}$ est un vecteur de n éléments.

La fonction f_0 est nommée la *fonction objectif*. Un vecteur $\mathbf{w}$ optimal, c'est-à-dire un vecteur $\mathbf{w}$ qui est une solution au problème, sera dénoté $\mathbf{w}^*$. Notons que pour l'instant, aucune hypothèse sur la convexité de ce problème n'est nécessaire.

Nous allons construire la formulation duale du problème en utilisant la méthode de *dualité de Lagrange*. L'idée générale de la méthode est d'augmenter la fonction objectif du problème

primal (original) en y incluant une somme pondérée des contraintes. On obtient ainsi le *Lagrangien* suivant :

$$\Lambda(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \triangleq f_0(\mathbf{w}) + \sum_{i=1}^{m} \alpha_i f_i(\mathbf{w}) + \sum_{j=1}^{p} \beta_j h_i(\mathbf{w}) , \tag{5.2}$$

où $\boldsymbol{\alpha}$ et $\boldsymbol{\beta}$ sont les vecteurs de *multiplicateurs de Lagrange*, associés respectivement aux contraintes d'inégalité et d'égalité. On appelle ces variables les *variables duales*, alors que $\mathbf{w}$ est le vecteur de *variables primales*.

Définissons maintenant le *Lagrangien dual*, qui est le minimum du Lagrangien en fonction des variables primales $\mathbf{w}$. Il est donc une fonction des variables duales.

$$\Lambda^D(\boldsymbol{\alpha}, \boldsymbol{\beta}) \triangleq \inf_{\mathbf{w}} \Lambda(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) .$$

Notons que Λ^D est concave, même si Λ n'est pas convexe (Boyd et Vandenberghe, 2004, section 5.1.2). De plus, pour tout $\boldsymbol{\alpha} \succeq \mathbf{0}_m$ et pour tout $\boldsymbol{\beta}$, $\Lambda^D(\boldsymbol{\alpha}, \boldsymbol{\beta})$ nous donne une borne inférieure de la valeur optimale du problème primal.

Le problème dual associé est le suivant :

$$\begin{array}{lll} \text{Résoudre} & \max\limits_{\boldsymbol{\alpha},\, \boldsymbol{\beta}} & \Lambda^D(\boldsymbol{\alpha}, \boldsymbol{\beta}) \\ \text{sous contraintes} & & \boldsymbol{\alpha} \succeq \mathbf{0}_m . \end{array} \tag{5.3}$$

Sous certaines conditions, la solution du problème primal est égale à la solution du problème dual (la borne inférieure de la valeur optimale du problème primal est atteinte). Si le problème primal est *convexe*, alors généralement la dualité forte est vérifiée.[1]

Notons que comme la valeur optimale $\mathbf{w}^*$ est un minimiseur selon $\mathbf{w}$ du Lagrangien (5.2), alors il existe des vecteurs $\boldsymbol{\alpha}$ et $\boldsymbol{\beta}$ tels que le gradient du Lagrangien selon $\mathbf{w}$, évalué au point $\mathbf{w}^*$, est égal au vecteur $\mathbf{0}_n$:

$$\nabla f_0(\mathbf{w}^*) + \sum_{i=1}^{m} \alpha_i \nabla f_i(\mathbf{w}^*) + \sum_{j=1}^{p} \beta_j \nabla h_i(\mathbf{w}^*) = \mathbf{0}_n .$$

Cette observation est à la base des conditions *Karush-Kuhn-Tucker (KKT)* qui donnent des conditions nécessaires (et suffisantes si le problème primal (5.1) est convexe) pour qu'un point $\mathbf{w}^*$ et une paire de points $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ soient optimaux dans les problèmes primal (5.1) et dual (5.3). Ces conditions sont les suivantes :

1. Le problème primal doit également respecter des conditions de régularité, comme par exemple la *condition de Slater* (Boyd et Vandenberghe, 2004), qui est respectée point $\mathbf{w}$ dans le domaine tel que $f_1(\mathbf{w}) < 0, \dots, f_m(\mathbf{w}) < 0$ et $h_1(\mathbf{w}) = 0, \dots, h_m(\mathbf{w}) = 0$. Nous n'entrerons pas dans plus de détails ici.

$$
\begin{aligned}
&\textit{Faisabilité primale} \\
&\qquad f_i(\mathbf{w}^*) \;\leq\; 0, \qquad i = 1, \dots, m \\
&\qquad g_i(\mathbf{w}^*) \;=\; 0, \qquad i = 1, \dots, p \\
&\textit{Faisabilité duale} \\
&\qquad \alpha_i^* \;\geq\; 0, \qquad i = 1, \dots, m \\
&\textit{Écart de complémentarité} \\
&\qquad \alpha_i^* f_i(\mathbf{w}^*) \;=\; 0, \qquad i = 1, \dots, m \\
&\textit{Stationnarité} \\
&\nabla f_0(\mathbf{w}^*) + \sum_{i=1}^{m} \alpha_i^* \nabla f_i(\mathbf{w}^*) + \sum_{j=1}^{p} \beta_j^* \nabla h_i(\mathbf{w}^*) \;=\; \mathbf{0}_n \,.
\end{aligned}
\tag{5.4}
$$

Cette caractérisation d'un point optimal $\mathbf{w}^*$ du problème primal peut permettre de simplifier le Lagrangien dual, et ainsi d'obtenir un problème dual intéressant. La prochaine section utilise ces notions pour construire le problème dual d'un programme quadratique général.

5.1.2 Le dual d'un programme quadratique

Le prochain théorème donne la forme duale standard d'un programme quadratique général (avec contraintes d'égalité et d'inégalité). Ce résultat, probablement déjà connu, est une généralisation de nombreux exemples d'application de la méthode de Lagrange aux programmes quadratiques avec contraintes d'inégalité (Boyd et Vandenberghe, 2004 ; Cristianini et Shawe-Taylor, 2000a). Notons une légère erreur détectée dans Cristianini et Shawe-Taylor (2000a, exemple 5.24) : la matrice $\mathbf{Q}$ de la fonction objectif du programme quadratique dual doit être inversée. Comme à notre connaissance cette forme générale comportant les deux types de contraintes n'existe pas dans la littérature, nous énonçons ce résultat comme un théorème.

Théorème 5.1. *Soit le programme quadratique suivant à n variables, m contraintes d'inégalité et p contraintes d'égalité :*

$$
\begin{aligned}
&\textit{Résoudre} \qquad \underset{\mathbf{w}}{\operatorname{argmin}} \quad \frac{1}{2}\, \mathbf{w}^\top \mathbf{M}\, \mathbf{w} \;+\; \mathbf{c}^\top \mathbf{q} \\
&\textit{sous contraintes} \qquad\quad \mathbf{A}\, \mathbf{w} \;=\; \mathbf{b}\,, \\
&\qquad\qquad\qquad\qquad\quad \mathbf{E}\, \mathbf{w} \;\preceq\; \mathbf{d}\,,
\end{aligned}
\tag{5.5}
$$

où $\mathbf{w}$ *et* $\mathbf{c}$ *sont des vecteurs (colonnes) de n éléments,* $\mathbf{M}$ *est une matrice symétrique semi-définie positive de taille* $n \times n$*,* $\mathbf{A}$ *est une matrice de taille* $p \times n$*,* $\mathbf{b}$ *est un vecteur de p éléments,* $\mathbf{E}$ *est une matrice de taille* $m \times n$ *et* $\mathbf{d}$ *est un vecteur de m éléments.*

La représentation duale, qui est également un programme quadratique, est la suivante :

$$\begin{aligned} &\textit{Résoudre} && \underset{\boldsymbol{\alpha},\boldsymbol{\beta}}{\operatorname{argmin}} \ \frac{1}{2}\begin{bmatrix}\boldsymbol{\alpha}^\top & \boldsymbol{\beta}^\top\end{bmatrix}\mathbf{F}\,\mathbf{M}^\dagger\mathbf{F}^\top\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix} + \left(\mathbf{F}\mathbf{M}^\dagger\mathbf{c}+\mathbf{g}\right)^\top\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix} + \frac{1}{2}\mathbf{c}^\top\mathbf{M}^\dagger\mathbf{c} \\ &\textit{sous contraintes} && \boldsymbol{\alpha} \ \succeq \ \mathbf{0}_m\,, \end{aligned}$$

où $\boldsymbol{\alpha}$ est un vecteur de m variables, $\boldsymbol{\beta}$ un vecteur de p variables, $\mathbf{F} = \begin{bmatrix}\mathbf{E}\\ \mathbf{A}\end{bmatrix}$ et $\mathbf{g} = \begin{bmatrix}\mathbf{d}\\ \mathbf{b}\end{bmatrix}$.

Démonstration. Spécialisons premièrement le problème primal de l'équation (5.1) à notre problème. Nous avons

$$\begin{aligned} f_0(\mathbf{w}) &= \frac{1}{2}\,\mathbf{w}^\top\,\mathbf{M}\,\mathbf{w} \ + \ \mathbf{c}^\top\,\mathbf{w}\,, \\ f_i(\mathbf{w}) &= (\mathbf{E}\,\mathbf{w}-\mathbf{d})_{i:}\,, \qquad \text{pour } i = 1\ldots m\,, \\ &\text{et} \\ h_i(\mathbf{w}) &= (\mathbf{A}\,\mathbf{w}-\mathbf{b})_{i:}\,, \qquad \text{pour } i = 1\ldots p\,. \end{aligned}$$

À partir de ces équations, le Lagrangien du programme quadratique (5.5) est

$$\begin{aligned} \Lambda(\mathbf{w},\boldsymbol{\alpha},\boldsymbol{\beta}) &= f_0(\mathbf{w}) + \sum_{i=1}^{m}\alpha_i f_i(\mathbf{w}) + \sum_{j=1}^{p}\beta_j h_i(\mathbf{w}) \\ &= \frac{1}{2}\,\mathbf{w}^\top\,\mathbf{M}\,\mathbf{w} + \mathbf{c}^\top\,\mathbf{w} + \sum_{i=1}^{m}\alpha_i\,(\mathbf{E}\,\mathbf{w}-\mathbf{d})_i + \sum_{j=1}^{p}\beta_j\,(\mathbf{A}\,\mathbf{w}-\mathbf{b})_j \\ &= \frac{1}{2}\,\mathbf{w}^\top\,\mathbf{M}\,\mathbf{w} + \mathbf{c}^\top\,\mathbf{w} + \boldsymbol{\alpha}^\top\,(\mathbf{E}\,\mathbf{w}-\mathbf{d}) + \boldsymbol{\beta}^\top\,(\mathbf{A}\,\mathbf{w}-\mathbf{b}) \\ &= \frac{1}{2}\,\mathbf{w}^\top\,\mathbf{M}\,\mathbf{w} + \mathbf{c}^\top\,\mathbf{w} + \boldsymbol{\alpha}^\top\mathbf{E}\,\mathbf{w} - \boldsymbol{\alpha}^\top\mathbf{d} + \boldsymbol{\beta}^\top\mathbf{A}\,\mathbf{w} - \boldsymbol{\beta}^\top\mathbf{b} \\ &= \left(\frac{1}{2}\,\mathbf{w}^\top\,\mathbf{M} + \mathbf{c}^\top + \boldsymbol{\alpha}^\top\mathbf{E} + \boldsymbol{\beta}^\top\mathbf{A}\right)\mathbf{w} - \boldsymbol{\alpha}^\top\mathbf{d} - \boldsymbol{\beta}^\top\mathbf{b} \qquad (5.6) \\ &= \left(\frac{1}{2}\,\mathbf{M}\,\mathbf{w} + \mathbf{c} + \mathbf{E}^\top\boldsymbol{\alpha} + \mathbf{A}^\top\boldsymbol{\beta}\right)^\top\mathbf{w} - \mathbf{d}^\top\boldsymbol{\alpha} - \mathbf{b}^\top\boldsymbol{\beta}\,. \qquad (5.7) \end{aligned}$$

Les gradients des fonctions f_0, f_i pour $i = 1,\ldots,m$ et h_i pour $i = 1,\ldots,p$ évalués au point optimal $\mathbf{w}^*$ sont les suivants :

$$\begin{aligned} \nabla f_0(\mathbf{w}^*) &= \mathbf{M}\,\mathbf{w}^* + \mathbf{c}\,, \\ \nabla f_i(\mathbf{w}^*) &= (\mathbf{E}_{i:})^\top\,, \qquad \text{pour } i = 1,\ldots,m\,, \\ &\text{et} \\ \nabla h_i(\mathbf{w}^*) &= (\mathbf{A}_{i:})^\top\,, \qquad \text{pour } i = 1,\ldots,p\,, \end{aligned}$$

où $\mathbf{E}_{i:}$ et $\mathbf{A}_{i:}$ sont respectivement la i-ième ligne de la matrice $\mathbf{E}$ (de $m{\times}n$ éléments) et la i-ième ligne de la matrice $\mathbf{A}$ (de $p{\times}n$ éléments). En prendre la transposée nous donne un vecteur colonne de n éléments, dans les deux cas.

On a donc comme gradient du Lagrangien au point optimal $\mathbf{w}^*$ le vecteur suivant :

$$\begin{aligned}\nabla f_0(\mathbf{w}^*) + \sum_{i=1}^{m} \alpha_i \nabla f_i(\mathbf{w}^*) + \sum_{j=1}^{p} \beta_j \nabla h_i(\mathbf{w}^*) &= \mathbf{M}\,\mathbf{w}^* + \mathbf{c} + \sum_{i=1}^{m} \alpha_i \left(\mathbf{E}_i\right)^\top + \sum_{j=1}^{p} \beta_j \left(\mathbf{A}_i\right)^\top \\ &= \mathbf{M}\,\mathbf{w}^* + \mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta}\,.\end{aligned}$$

Le vecteur résultant étant égal au vecteur $\mathbf{0}_n$ à l'optimum (voir la contrainte de stationnarité des conditions KKT de l'équation (5.4)), on obtient la caractérisation suivante de la solution primale $\mathbf{w}^*$ en fonction des vecteurs de variables duales $\boldsymbol{\alpha}$ et $\boldsymbol{\beta}$:

$$\begin{aligned} & & \mathbf{0}_n &= \mathbf{M}\,\mathbf{w}^* + \mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta} \\ \Leftrightarrow & & & \\ & & \mathbf{M}\,\mathbf{w}^* &= -\mathbf{c} - \mathbf{E}^\top \boldsymbol{\alpha} - \mathbf{A}^\top \boldsymbol{\beta} \\ \Leftrightarrow & & & \\ & & \mathbf{w}^* &= \mathbf{M}^\dagger \left(-\mathbf{c} - \mathbf{E}^\top \boldsymbol{\alpha} - \mathbf{A}^\top \boldsymbol{\beta}\right) \end{aligned} \tag{5.8}$$

En substituant l'équation (5.8) dans l'équation (5.7), nous obtenons le Lagrangien dual :

$$\begin{aligned}\Lambda^D(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \inf_{\mathbf{w}} \Lambda(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\ &= \Lambda(\mathbf{w}^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\ &= \left(\frac{1}{2}\mathbf{M}\,\mathbf{w}^* + \mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta}\right)^\top \mathbf{w}^* - \mathbf{d}^\top \boldsymbol{\alpha} - \mathbf{b}^\top \boldsymbol{\beta} \\ &= \left(\frac{1}{2}\mathbf{M}\mathbf{M}^\dagger \left(-\mathbf{c} - \mathbf{E}^\top \boldsymbol{\alpha} - \mathbf{A}^\top \boldsymbol{\beta}\right) + \mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta}\right)^\top \mathbf{w}^* - \mathbf{d}^\top \boldsymbol{\alpha} - \mathbf{b}^\top \boldsymbol{\beta} \\ &= \left(\frac{1}{2}\left(-\mathbf{c} - \mathbf{E}^\top \boldsymbol{\alpha} - \mathbf{A}^\top \boldsymbol{\beta}\right) + \mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta}\right)^\top \mathbf{w}^* - \mathbf{d}^\top \boldsymbol{\alpha} - \mathbf{b}^\top \boldsymbol{\beta} \\ &= \left(\frac{1}{2}\left(\mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta}\right)\right)^\top \mathbf{w}^* - \mathbf{d}^\top \boldsymbol{\alpha} - \mathbf{b}^\top \boldsymbol{\beta} \\ &= \left(\frac{1}{2}\left(\mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta}\right)\right)^\top \mathbf{M}^\dagger \left(-\mathbf{c} - \mathbf{E}^\top \boldsymbol{\alpha} - \mathbf{A}^\top \boldsymbol{\beta}\right) - \mathbf{d}^\top \boldsymbol{\alpha} - \mathbf{b}^\top \boldsymbol{\beta} \\ &= -\frac{1}{2}\left(\mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta}\right)^\top \mathbf{M}^\dagger \left(\mathbf{c} + \mathbf{E}^\top \boldsymbol{\alpha} + \mathbf{A}^\top \boldsymbol{\beta}\right) - \mathbf{d}^\top \boldsymbol{\alpha} - \mathbf{b}^\top \boldsymbol{\beta} \\ &= -\frac{1}{2}\left(\boldsymbol{\alpha}^\top \mathbf{E}\,\mathbf{M}^\dagger\,\mathbf{E}^\top \boldsymbol{\alpha} + \boldsymbol{\beta}^\top \mathbf{A}\,\mathbf{M}^\dagger\,\mathbf{A}^\top \boldsymbol{\beta} + 2\,\boldsymbol{\alpha}^\top \mathbf{E}\,\mathbf{M}^\dagger\,\mathbf{A}^\top \boldsymbol{\beta}\right. \\ &\qquad \left. + \mathbf{c}^\top \mathbf{M}^\dagger\,\mathbf{c} + 2\,\boldsymbol{\alpha}^\top \mathbf{E}\,\mathbf{M}^\dagger \mathbf{c} + 2\,\boldsymbol{\beta}^\top \mathbf{A}\,\mathbf{M}^\dagger\,\mathbf{c}\right) - \mathbf{d}^\top \boldsymbol{\alpha} - \mathbf{b}^\top \boldsymbol{\beta} \\ &= -\frac{1}{2}\begin{bmatrix}\boldsymbol{\alpha}^\top & \boldsymbol{\beta}^\top\end{bmatrix} \mathbf{F}\,\mathbf{M}^\dagger \mathbf{F}^\top \begin{bmatrix}\boldsymbol{\alpha} \\ \boldsymbol{\beta}\end{bmatrix} - \left(\mathbf{F}\mathbf{M}^\dagger \mathbf{c} + \mathbf{g}\right)^\top \begin{bmatrix}\boldsymbol{\alpha} \\ \boldsymbol{\beta}\end{bmatrix} - \frac{1}{2}\mathbf{c}^\top \mathbf{M}^\dagger \mathbf{c}\,,\end{aligned}$$

où la dernière égalité est obtenue en combinant certaines matrices et vecteurs ensemble :

$$\mathbf{F} = \begin{bmatrix}\mathbf{E} \\ \mathbf{A}\end{bmatrix} \quad \text{et} \quad \mathbf{g} = \begin{bmatrix}\mathbf{d} \\ \mathbf{b}\end{bmatrix}.$$

Le résultat final est obtenu en construisant le problème dual à partir du lagrangien dual (voir équation (5.3)), et en remplaçant le problème de maximisation par un problème de minimisation. □

À partir de cette formulation duale d'un programme quadratique général, nous pouvons obtenir un problème dual pour MinCq. Ceci dit, celui-ci ne permet pas de déduire de nouvelle information sur MinCq, et n'a pas permis de construire un nouvel algorithme basé sur la génération de colonnes. Parfois, pour obtenir un problème dual intéressant, il est nécessaire de transformer la fonction objectif primale en introduisant de nouvelles variables et contraintes d'égalité (Boyd et Vandenberghe, 2004, section 5.7.1). La prochaine section utilise cette idée, ainsi que le retrait de certaines contraintes du problème d'optimisation original de MinCq, dans le but de construire CqBoost.

5.2 Génération de colonnes

La *génération de colonnes* (ou «*column generation*») est une méthode originellement utilisée pour résoudre des programmes linéaires, par exemple par le solveur commercial CPLEX (IBM Corp, 2010). Dans Demiriz, Bennett et Shawe-Taylor (2002), les auteurs utilisent cette technique pour construire un algorithme de boosting basé sur la programmation linéaire, nommé LPBoost.

L'idée générale de la génération de colonnes est de restreindre un problème d'optimisation primal en ne considérant qu'un sous-ensemble des classificateurs de base (*decision stumps*, noyaux basés sur les exemples de l'ensemble d'entraînement, etc.), qui sont les *colonnes* de la matrice de classification associée au problème. Ce problème est appelé le problème primal restreint (ou «*restricted master problem*»).

Généralement, la formulation duale d'un problème d'optimisation a comme propriété que les variables du problème primal sont associées à des contraintes dans le problème dual. Les variables (colonnes) qui ne sont pas considérées dans le problème primal restreint correspondent à des contraintes qui sont ignorées dans le problème dual. Résoudre le problème primal restreint correspond donc à résoudre une *relaxation du problème dual*.

La génération de colonnes ajoute itérativement des colonnes à considérer, en utilisant le problème original comme un *oracle* qui choisit les colonnes à ajouter. Par exemple dans LPBoost (Demiriz, Bennett et Shawe-Taylor, 2002), les variables duales représentent le coût des erreurs de classification. Ces coûts sont utilisés pour choisir la prochaine colonne à ajouter au problème primal restreint. En ajoutant itérativement des colonnes à considérer, l'algorithme converge vers l'optimum du problème d'optimisation original, *possiblement avant d'avoir généré toutes les colonnes*.

Notons que dans le cas plus simple où les colonnes ont déjà été générées a priori, un algorithme basé sur la génération de colonnes sélectionnera itérativement la meilleure colonne à ajouter au problème d'optimisation. Ces algorithmes peuvent par contre être utilisés avec un nombre *infini* de colonnes, par exemple en considérant un ensemble infini de classifica-

teurs. Dans ce cas, l'algorithme doit, à chaque itération, *générer* une nouvelle colonne qui viole les contraintes du problème dual. L'algorithme s'arrête lorsqu'aucune colonne ne viole les contraintes du problème dual (ou lorsqu'il devient impossible de générer une telle colonne), jusqu'à une certaine précision ϵ. La valeur de ϵ peut être vue comme un paramètre qui aura un effet de critère d'arrêt de l'algorithme.

Bi, T. Zhang et Bennett (2004) ont étendu la génération de colonnes aux programmes quadratiques, en présentant un algorithme de boosting nommé CG-Boost, dont le problème d'optimisation est très similaire à celui du SVM (Cortes et Vapnik, 1995). Shen et Li (2010) ont également utilisé la génération de colonnes pour construire l'algorithme *MDBoost* (pour *Margin Distribution Boost*), dont le problème d'optimisation est relié de près à MinCq. Finalement, Shen, Li et Hengel (2013) utilisent la génération de colonnes pour construire une famille d'algorithmes de boosting permettant de retrouver plusieurs algorithmes existants, en choisissant la bonne fonction de perte et le bon régularisateur. Leur cadre général porte également le nom de CGBoost (cette fois-ci sans trait d'union). Notons que Shen, Li et Hengel (2013) ne citent pas correctement les travaux de Bi, T. Zhang et Bennett (2004). Il est seulement indiqué qu'il est simple d'appliquer les méthodes de généralisation de colonnes aux mixtures de noyaux tel que dans Bi, T. Zhang et Bennett (2004), alors que c'est dans cet article qu'une importante généralisation de la génération de colonnes de la programmation linéaire vers la programmation quadratique a été réalisée.

5.3 Conception de CqBoost

Lors de la conception de CqBoost, l'une des étapes clé a été la définition du problème par l'introduction de la matrice de vote $\mathbf{F}$ de la définition 4.6. Les versions précédentes de l'algorithme présentées dans Laviolette, Marchand et Roy (2011) et Germain, Lacasse, Laviolette, Marchand et Roy (2015) nécessitaient l'introduction de notation supplémentaire, et rendaient beaucoup plus difficile la construction du dual de Lagrange.

Rappelons d'abord le problème d'optimisation (à $2n$ variables) de MinCq (algorithme 3) :

$$\begin{aligned} \text{Résoudre :} \quad & \underset{\mathbf{q}}{\operatorname{argmin}} \ \frac{1}{m}\mathbf{q}^\top \mathbf{F}^\top \mathbf{F}\,\mathbf{q} \\ \text{sous contraintes :} \quad & \frac{1}{m}\mathbf{y}^\top \mathbf{F}\,\mathbf{q} \;=\; \mu\,, \quad \begin{bmatrix}\mathbf{I}_n & \mathbf{I}_n\end{bmatrix}\mathbf{q} \;=\; \frac{1}{n}\mathbf{1}_n\,, \quad \mathbf{q} \;\geq\; \mathbf{0}_{2n}\,, \end{aligned}$$

où $\mathbf{q}$ est un vecteur de $2n$ variables représentant les poids associés aux votants, $\mathbf{F}$ est la matrice de vote définie à la définition 4.6, et $\mathbf{y}$ est le vecteur d'étiquettes défini à la définition 4.7.

Cette formulation du problème a l'avantage d'être simple à résoudre comme il s'agit d'un programme quadratique, mais a comme désavantage de considérer toute la matrice $\mathbf{F}$ de $m \times 2n$ éléments. Le calcul de $\mathbf{F}^\top\mathbf{F}$ se fait en $O(m \times n^2)$, puis la résolution du programme

quadratique se fait en $O(n^3)$. Conséquemment, MinCq n'est pas adapté lorsque le nombre de votants considérés est grand.

Relaxation des contraintes

Dans l'optique de créer un algorithme itératif qui, à chaque itération, choisira un nouveau votant à ajouter au vote de majorité, la contrainte de quasi-uniformité $\begin{bmatrix}\mathbf{I}_n & \mathbf{I}_n\end{bmatrix}\mathbf{q} = \frac{1}{n}\mathbf{1}_n$ est problématique. En effet, celle-ci implique que la plupart des votants considérés auront un poids non nul, puisque la somme des poids d'un votant et de son complément doit être d'exactement $\frac{1}{n}$. Or, nous désirons construire CqBoost de telle manière que celui-ci débute avec un poids de zéro pour chaque votant, et à chaque itération sélectionne un nouveau votant à qui il attribuera un poids. Malgré le fait que la solution au problème d'optimisation de MinCq pourrait en théorie attribuer un poids égal de $\frac{1}{2n}$ à plusieurs paires de votants complémentaires (annulant ainsi leur vote, ce qui serait équivalent à leur donner un poids nul), nous ne l'observons pas en pratique. En effet, pour arriver à la simplification du problème où le premier moment de la marge peut être contraint à être *exactement* égal à μ, MinCq a besoin de faire varier les poids de la plupart des votants. Pour un algorithme itératif qui ne sélectionne qu'un nouveau votant à la fois, obtenir une marge exactement égale à μ est généralement impossible jusqu'à ce qu'une assez grande quantité de votants soit sélectionnée.

La première étape de conception consiste donc à retirer cette contrainte de quasi-uniformité, mais d'ajouter comme contrainte que la somme des poids doit égaler 1, assurant ainsi que Q est bel et bien une distribution. Nous relaxons également la contrainte sur le premier moment de la marge, en la contraignant plutôt à être *plus grande ou égale* à μ. De cette manière, nous évitons tout de même le problème d'instabilité numérique $0/0$ de la C-borne (voir le problème 4.2 du chapitre 4), sans pour autant restreindre les votes de majorités possibles. Le problème d'optimisation relaxé[2] est donc le suivant :

$$\begin{aligned} \text{Résoudre :} \quad & \underset{\mathbf{q}}{\operatorname{argmin}} \ \frac{1}{m}\mathbf{q}^\top \mathbf{F}^\top \mathbf{F}\,\mathbf{q} \\ \text{sous contraintes :} \quad & \frac{1}{m}\mathbf{y}^\top \mathbf{F}\,\mathbf{q} \geq \mu\,, \quad \mathbf{q} \geq \mathbf{0}_{2n}\,, \quad {\mathbf{1}_{2n}}^\top \mathbf{q} = 1\,. \end{aligned} \tag{5.9}$$

Cette version relaxée du problème d'optimisation correspond à trouver la distribution Q, dont le premier moment empirique de la marge $\mu_1\left(M_Q^S\right)$ est au moins μ, qui minimise le second moment empirique de la marge $\mu_2\left(M_Q^S\right)$. Il s'agit donc simplement d'une minimisation de la C-borne empirique C_Q^S. Notons que pour obtenir une justification théorique de cet algorithme, la théorie PAC-bayésienne *avec* KL doit être considérée, ce dont nous discutons à la section 5.4.

2. Dans Roy, Marchand et Laviolette (2016), la contrainte d'inégalité sur le premier moment de la marge est dans le mauvais sens, malgré que le texte d'explication soit valide. Cependant, le code utilisé pour les expérimentations publiées dans cet article était correct : il n'y a donc pas d'erreur dans les résultats, seulement dans l'énoncé du problème.

Introduction d'une variable représentant la marge

Malgré la relaxation des contraintes et l'introduction de la matrice de vote $\mathbf{F}$, nos premières tentatives de créer un dual de Lagrange intéressant n'ont pas porté fruit. Pour y arriver, nous reprenons une idée introduite par Shen et Li (2010), et introduisons un vecteur γ de m variables, où chaque $\gamma_k = y_k \sum_{i=1}^{2n} q_i F_{ki}$, la marge du vote de majorité sur l'exemple (x_k, y_k). De manière équivalente, $\gamma = \text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}$. Le problème d'optimisation résultant est le suivant :

$$\begin{aligned} &\text{Résoudre :} \quad \underset{\mathbf{q},\gamma}{\text{argmin}} \;\; \frac{1}{m}\gamma^\top\gamma \\ &\text{sous contraintes :} \quad \gamma \;=\; \text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}\,, \quad \frac{1}{m}\mathbf{1}_m{}^\top\gamma \;\geq\; \mu\,, \quad \mathbf{q} \;\geq\; \mathbf{0}_{2n}\,, \quad \mathbf{1}_{2n}{}^\top\mathbf{q} \;=\; 1\,. \end{aligned} \tag{5.10}$$

Notons que la fonction objectif de l'équation (5.10) est équivalente à la fonction objectif de l'équation (5.9), comme $\text{diag}(\mathbf{y})^\top\text{diag}(\mathbf{y}) = \mathbf{I}_m$. Nous obtenons en effet que

$$\gamma^\top\gamma \;=\; (\text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q})^\top\,\text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q} \;=\; \mathbf{q}^\top\mathbf{F}^\top\text{diag}(\mathbf{y})^\top\text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q} \;=\; \mathbf{q}^\top\mathbf{F}^\top\mathbf{F}\,\mathbf{q}\,.$$

En additionnant une somme pondérée des contraintes à la fonction objectif, nous obtenons le Lagrangien suivant :

$$\begin{aligned} \Lambda(\mathbf{q},\gamma,\boldsymbol{\alpha},\beta,\boldsymbol{\xi},\nu) \;\triangleq\; &\frac{1}{m}\gamma^\top\gamma + \boldsymbol{\alpha}^\top(\gamma - \text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}) - \beta\left(\frac{1}{m}\mathbf{1}_m{}^\top\gamma - \mu\right) \\ &- \boldsymbol{\xi}^\top\mathbf{q} + \nu\left(\mathbf{1}_{2n}{}^\top\mathbf{q} - 1\right), \end{aligned} \tag{5.11}$$

où $\boldsymbol{\alpha}$, β, $\boldsymbol{\xi}$ et ν sont les multiplicateurs de Lagrange. Les multiplicateurs β et $\boldsymbol{\xi}$ doivent être non négatifs car ils sont reliés à des contraintes d'inégalité. Le dual de Lagrange est obtenu en trouvant les vecteurs $\mathbf{q}^\star$ et $\gamma^\star$ minimisant le Lagrangien. La condition de stationnarité des conditions KKT (voir l'équation (5.4)) indiquent que cette solution est atteinte quand les dérivées partielles en fonction des vecteurs $\mathbf{q}$ et γ sont nulles.

Commençons par donner la dérivée partielle en fonction de $\mathbf{q}^\star$. Nous avons

$$\begin{aligned} &\frac{\partial}{\partial\mathbf{q}^\star}\Lambda(\mathbf{q}^\star,\gamma^\star,\boldsymbol{\alpha},\beta,\boldsymbol{\xi},\nu) \\ &\quad= \frac{\partial}{\partial\mathbf{q}^\star}\left[\frac{1}{m}\gamma^\star{}^\top\gamma^\star + \boldsymbol{\alpha}^\top(\gamma^\star - \text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}^\star) - \beta\left(\frac{1}{m}\mathbf{1}_m{}^\top\gamma^\star - \mu\right) - \boldsymbol{\xi}^\top\mathbf{q}^\star + \nu\left(\mathbf{1}_{2n}{}^\top\mathbf{q}^\star - 1\right)\right] \\ &\quad= \frac{\partial}{\partial\mathbf{q}^\star}\left[\boldsymbol{\alpha}^\top(\gamma^\star - \text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}^\star) - \boldsymbol{\xi}^\top\mathbf{q}^\star + \nu\,\mathbf{1}_{2n}{}^\top\mathbf{q}^\star - \nu\right] \\ &\quad= \frac{\partial}{\partial\mathbf{q}^\star}\left[\boldsymbol{\alpha}^\top\gamma^\star - \frac{1}{m}\boldsymbol{\alpha}^\top\text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}^\star - \boldsymbol{\xi}^\top\mathbf{q}^\star + \nu\,\mathbf{1}_{2n}{}^\top\mathbf{q}^\star\right] \\ &\quad= \frac{\partial}{\partial\mathbf{q}^\star}\left[-\boldsymbol{\alpha}^\top\text{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}^\star - \boldsymbol{\xi}^\top\mathbf{q}^\star + \nu\,\mathbf{1}_{2n}{}^\top\mathbf{q}^\star\right] \\ &\quad= -\mathbf{F}^\top\text{diag}(\mathbf{y})\,\boldsymbol{\alpha} - \boldsymbol{\xi} + \nu\,\mathbf{1}_{2n}\,. \end{aligned}$$

La condition que cette dérivée partielle doit être nulle implique donc la contrainte suivante :

$$\mathbf{F}^\top \mathrm{diag}(\mathbf{y})\,\boldsymbol{\alpha} \;=\; \nu\,\mathbf{1}_{2n} - \boldsymbol{\xi}\,. \tag{5.12}$$

La dérivée partielle en fonction de $\gamma^\star$ nous donne

$$\begin{aligned}
&\frac{\partial}{\partial \gamma^\star}\Lambda(\mathbf{q}^\star, \gamma^\star, \boldsymbol{\alpha}, \beta, \boldsymbol{\xi}, \nu) \\
&\quad= \frac{\partial}{\partial \gamma^\star}\left[\frac{1}{m}{\gamma^\star}^\top\gamma^\star + \boldsymbol{\alpha}^\top(\gamma^\star - \mathrm{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}^\star) - \beta\left(\frac{1}{m}\mathbf{1}^\top\gamma^\star - \mu\right) - \boldsymbol{\xi}^\top\mathbf{q}^\star + \nu\left({\mathbf{1}_{2n}}^\top\mathbf{q}^\star - 1\right)\right] \\
&\quad= \frac{\partial}{\partial \gamma^\star}\left[\frac{1}{m}{\gamma^\star}^\top\gamma^\star + \boldsymbol{\alpha}^\top\gamma^\star - \boldsymbol{\alpha}^\top\mathrm{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}^\star - \frac{\beta}{m}{\mathbf{1}_m}^\top\gamma^\star + \beta\mu - \boldsymbol{\xi}^\top\mathbf{q}^\star + \nu\,{\mathbf{1}_{2n}}^\top\mathbf{q}^\star - \nu\right] \\
&\quad= \frac{\partial}{\partial \gamma^\star}\left[\frac{1}{m}{\gamma^\star}^\top\gamma^\star + \boldsymbol{\alpha}^\top\gamma^\star - \frac{\beta}{m}{\mathbf{1}_m}^\top\gamma^\star\right] \\
&\quad= \frac{2}{m}\gamma^\star + \boldsymbol{\alpha} - \frac{\beta}{m}\mathbf{1}_m\,.
\end{aligned}$$

La condition que cette dérivée partielle doit être nulle implique donc la contrainte suivante :

$$\gamma^\star \;=\; -\frac{m}{2}\boldsymbol{\alpha} + \frac{\beta}{2}\mathbf{1}_m\,. \tag{5.13}$$

Les contraintes des équations (5.12) et (5.13) nous donnent une caractérisation de la valeur optimale de $\mathbf{q}$ et γ. Cette information peut donc être transférée dans le Lagrangien de l'équation (5.11), nous donnant le Lagrangien dual suivant :

$$\begin{aligned}
&\Lambda^D(\boldsymbol{\alpha}, \beta, \boldsymbol{\xi}, \nu) \\
&\quad= \inf_{\mathbf{q},\gamma}\Lambda(\mathbf{q}, \gamma, \boldsymbol{\alpha}, \beta, \boldsymbol{\xi}, \nu) \\
&\quad= \Lambda(\mathbf{q}^\star, \gamma^\star, \boldsymbol{\alpha}, \beta, \boldsymbol{\xi}, \nu) \\
&\quad= \frac{1}{m}{\gamma^\star}^\top\gamma^\star + \boldsymbol{\alpha}^\top(\gamma^\star - \mathrm{diag}(\mathbf{y})\mathbf{F}\,\mathbf{q}^\star) - \beta\left(\frac{1}{m}{\mathbf{1}_m}^\top\gamma^\star - \mu\right) - \boldsymbol{\xi}^\top\mathbf{q}^\star + \nu\left({\mathbf{1}_{2n}}^\top\mathbf{q}^\star - 1\right) \\
&\quad= \frac{1}{m}{\gamma^\star}^\top\gamma^\star + \boldsymbol{\alpha}^\top\gamma^\star - \left(\mathbf{F}^\top\mathrm{diag}(\mathbf{y})\,\boldsymbol{\alpha}\right)^\top\mathbf{q}^\star - \frac{\beta}{m}{\mathbf{1}_m}^\top\gamma^\star + \beta\mu - \left(\boldsymbol{\xi} + \nu\,\mathbf{1}_{2n}\right)^\top\mathbf{q}^\star - \nu \\
&\quad= \frac{1}{m}{\gamma^\star}^\top\gamma^\star + \boldsymbol{\alpha}^\top\gamma^\star + \left(\boldsymbol{\xi} + \nu\,\mathbf{1}_{2n}\right)^\top\mathbf{q}^\star - \frac{\beta}{m}{\mathbf{1}_m}^\top\gamma^\star + \beta\mu - \left(\boldsymbol{\xi} + \nu\,\mathbf{1}_{2n}\right)^\top\mathbf{q}^\star - \nu \qquad (5.14) \\
&\quad= \frac{1}{m}{\gamma^\star}^\top\gamma^\star + \boldsymbol{\alpha}^\top\gamma^\star - \frac{\beta}{m}{\mathbf{1}_m}^\top\gamma^\star + \beta\mu - \nu \\
&\quad= \left(\frac{1}{m}\gamma^\star + \boldsymbol{\alpha} - \frac{\beta}{m}\mathbf{1}_m\right)^\top\gamma^\star + \beta\mu - \nu \\
&\quad= \left(\frac{1}{m}\left(-\frac{m}{2}\boldsymbol{\alpha} + \frac{\beta}{2}\mathbf{1}_m\right) + \boldsymbol{\alpha} - \frac{\beta}{m}\mathbf{1}_m\right)^\top\left(-\frac{m}{2}\boldsymbol{\alpha} + \frac{\beta}{2}\mathbf{1}_m\right) + \beta\mu - \nu \qquad (5.15) \\
&\quad= \left(-\frac{1}{2}\boldsymbol{\alpha} + \frac{\beta}{2m}\mathbf{1}_m + \boldsymbol{\alpha} - \frac{\beta}{m}\mathbf{1}_m\right)^\top\left(-\frac{m}{2}\boldsymbol{\alpha} + \frac{\beta}{2}\mathbf{1}_m\right) + \beta\mu - \nu \\
&\quad= \left(\frac{1}{2}\boldsymbol{\alpha} - \frac{\beta}{2m}\mathbf{1}_m\right)^\top\left(-\frac{m}{2}\boldsymbol{\alpha} + \frac{\beta}{2}\mathbf{1}_m\right) + \beta\mu - \nu
\end{aligned}$$

$$
\begin{aligned}
&= -\frac{m}{4}\boldsymbol{\alpha}^\top\boldsymbol{\alpha} + \frac{\beta}{4}\boldsymbol{\alpha}^\top\mathbf{1}_m + \frac{\beta}{4}\mathbf{1}_m{}^\top\boldsymbol{\alpha} - \frac{\beta^2}{4m}\mathbf{1}_m{}^\top\mathbf{1}_m + \beta\mu - \nu \\
&= -\frac{m}{4}\boldsymbol{\alpha}^\top\boldsymbol{\alpha} + \frac{\beta}{2}\mathbf{1}_m{}^\top\boldsymbol{\alpha} - \frac{\beta^2}{4} + \beta\mu - \nu\,, \qquad (5.16)
\end{aligned}
$$

où la ligne (5.14) est obtenue en substituant la contrainte de l'équation (5.12) et la ligne (5.15) est obtenue en substituant $\gamma^\star$ par sa valeur optimale donnée à l'équation (5.13).

Le problème d'optimisation dual est obtenu en maximisant le Lagrangien dual de l'équation (5.16), sous contrainte (5.12), et sous contrainte que les multiplicateurs de Lagrange β et $\boldsymbol{\xi}$ sont non négatifs. Nous obtenons donc

$$
\begin{aligned}
\text{Résoudre :} \quad & \underset{\boldsymbol{\alpha},\beta,\boldsymbol{\xi},\nu}{\operatorname{argmax}} \; -\frac{m}{4}\boldsymbol{\alpha}^\top\boldsymbol{\alpha} + \frac{\beta}{2}\mathbf{1}_m{}^\top\boldsymbol{\alpha} - \frac{\beta^2}{4} + \beta\mu - \nu \\
\text{sous contraintes :} \quad & \mathbf{F}^\top \operatorname{diag}(\mathbf{y})\,\boldsymbol{\alpha} \;=\; \nu\,\mathbf{1}_{2n} - \boldsymbol{\xi}\,, \quad \beta \;\geq\; 0\,, \quad \boldsymbol{\xi} \;\geq\; \mathbf{0}_{2n}\,.
\end{aligned}
$$

Nous remarquons que le vecteur $\boldsymbol{\xi}$ est non négatif, et est absent de la fonction objectif. Son seul rôle est donc d'affecter la contrainte d'égalité sur le vecteur $\boldsymbol{\alpha}$. Il s'agit donc d'un vecteur de *variables d'écart*, qui peuvent être retirées du problème d'optimisation en transformant la contrainte d'égalité en contrainte d'inégalité. Avec ce changement et en considération la négation de la fonction objectif, nous obtenons la réécriture suivante du problème dual :[3]

$$
\begin{aligned}
\text{Résoudre :} \quad & \underset{\boldsymbol{\alpha},\beta,\nu}{\operatorname{argmin}} \; \frac{m}{4}\boldsymbol{\alpha}^\top\boldsymbol{\alpha} - \frac{\beta}{2}\mathbf{1}_m{}^\top\boldsymbol{\alpha} + \frac{\beta^2}{4} - \beta\mu + \nu \\
\text{sous contraintes :} \quad & \mathbf{F}^\top \operatorname{diag}(\mathbf{y})\,\boldsymbol{\alpha} \;\preceq\; \nu\,\mathbf{1}_{2n}\,, \quad \beta \;\geq\; 0\,.
\end{aligned} \qquad (5.17)
$$

Cette formulation duale cherche une pondération $\boldsymbol{\alpha}$ sur les exemples, dont le compromis entre le premier et second moment est contrôlé par une variable β. Le problème est contraint par un terme $\mathbf{F}^\top \operatorname{diag}(\mathbf{y})\,\boldsymbol{\alpha}$, qui peut être vu comme un « score » donné à chaque votant, et correspond à la somme pondérée par $\boldsymbol{\alpha}$ des exemples correctement classifiés, moins la somme pondérée des exemples classifiés incorrectement (incluant la « confiance » de chaque votant lorsque ceux-ci sont des fonctions à valeur réelle). Cette valeur est donc positive pour un votant si celui-ci classifie bien la majorité des exemples, pondérés par $\boldsymbol{\alpha}$. Cette mesure est appelée le «*edge*» d'un votant (Demiriz, Bennett et Shawe-Taylor, 2002 ; Shen, Li et Hengel, 2013), et peut être utilisée pour *guider le choix du prochain votant à ajouter au vote de majorité*. Le « edge » se trouve à être le dual de la marge (Shen, Li et Hengel, 2013), et est également le critère de sélection de colonne qui apparaît dans plusieurs algorithmes basés sur la génération de colonnes, tels que LPBoost (Demiriz, Bennett et Shawe-Taylor, 2002), CG-Boost (Bi, T. Zhang et Bennett, 2004) et MDBoost (Shen et Li, 2010).

Nous avons maintenant tous les éléments nécessaires pour énoncer l'algorithme CqBoost, qui à chaque itération résout le programme quadratique primal de l'équation (5.9), et sélectionne

3. Suite à la correction de l'erreur du sens de la contrainte d'inégalité sur le premier moment de la marge présente dans Roy, Marchand et Laviolette (2016), la fonction objectif du problème dual est légèrement différente, et correspond à inverser le signe du multiplicateur β (seulement dans la fonction objectif, et non dans la contrainte).

une nouvelle colonne en choisissant celle qui viole le plus la contrainte du problème dual de l'équation (5.17).

L'algorithme 5 montre le pseudo-code de CqBoost. Pour simplifier la notation et faciliter la comparaison avec les autres algorithmes basés sur la génération de colonnes, nous restreignons l'algorithme au cas où l'ensemble $\mathcal{H}$ de votants est fini, et où la matrice de classification $\mathbf{F}$ associée est déjà construite. Les algorithmes de l'état de l'art basés sur la génération de colonnes font également cette simplification (Demiriz, Bennett et Shawe-Taylor, 2002 ; Bi, T. Zhang et Bennett, 2004 ; Shen et Li, 2010). Notons qu'aucun de ces algorithmes (incluant CqBoost) ne sont limités ni aux ensembles de votants finis, ni à la situation où les colonnes ont été générées a priori. Nous verrons dans la section 5.4 que la théorie supportant CqBoost est également valide pour les ensembles infinis de votants. Considérer de tels ensembles ouvre par contre la voie à plusieurs questions de recherche que nous n'avons pas étudiées dans cette thèse.

Algorithme 5 CqBoost

Entrée : T, le nombre d'itérations maximal

$\mathbf{q} \leftarrow \mathbf{0}_{2n}$
$\boldsymbol{\alpha} \leftarrow \frac{1}{m}\mathbf{1}_m$
$\nu \leftarrow -\infty$
$\tilde{\mathbf{F}} \leftarrow$ une matrice vide
pour $t \leftarrow 1, \dots, T$ **faire**
  $i \leftarrow \operatorname{argmax}_i \sum_{k=1}^{m} \alpha_k y_k F_{ki}$ ▷ Choisir la colonne avec le plus grand « edge »
  si $\sum_{k=1}^{m} \alpha_k y_k F_{ki} \leq \nu + \epsilon$ **alors** ▷ Si la contrainte n'est pas violée à ϵ près
    Quitter la boucle
  fin si
  Mettre à jour $\tilde{\mathbf{F}}$ en ajoutant la i-ième colonne de $\mathbf{F}$
  $\mathbf{q}, \boldsymbol{\alpha}, \nu \leftarrow$ solution de (5.10) et (5.17) en considérant $\tilde{\mathbf{F}}$ (de taille $m \times t$)
fin pour

Sortie : $\mathbf{q}$

L'algorithme 5 est initialisé dans le cas dégénéré où aucune colonne n'a été choisie. Le vecteur $\mathbf{q}$ est initialisé à $\mathbf{0}_{2n}$.[4] Le vecteur $\boldsymbol{\alpha}$ est initialisé à sa valeur optimale dans cette situation dégénérée, c'est à dire

$$\frac{\partial}{\partial \boldsymbol{\alpha}} \left[\frac{m}{4} \boldsymbol{\alpha}^\top \boldsymbol{\alpha} - \frac{\beta}{2} \mathbf{1}_m{}^\top \boldsymbol{\alpha} + \frac{\beta^2}{4} - \beta\mu + \nu \right] = \mathbf{0}_m \quad \Leftrightarrow \quad \boldsymbol{\alpha} = \frac{\beta}{m} \mathbf{1}_m ,$$

ou de manière équivalente comme β est positif, $\boldsymbol{\alpha}$ est initialisé à une distribution uniforme sur tous les exemples. Finalement, la valeur de ν est initialisée à $-\infty$, puisque celle-ci n'est pas

4. Notons qu'à chaque itération, la sortie de l'algorithme exécuté sur le problème primal restreint ne retournera des poids que sur les colonnes générées jusqu'à présent. La contrainte que $\mathbf{q}$ soit une distribution ne s'applique que sur l'ensemble de votants correspondants à ces colonnes. Nous devons donc par la suite convertir ce vecteur en une distribution sur l'ensemble original de votants. Nous avons donc un léger abus de notation dans le pseudo-code de CqBoost.

bornée et qu'une valeur infiniment petite rend impossible à l'algorithme de quitter la boucle avant d'avoir choisi une première colonne. À chaque itération, CqBoost construit une matrice $\tilde{\mathbf{F}}^\top \tilde{\mathbf{F}}$ en $O(m \times t^2)$, et résout un programme quadratique en $O(t^3)$, où t est seulement le nombre de colonnes générées jusqu'à maintenant. La seule étape de CqBoost dépendant du nombre de votants considérés n est lors du choix du prochain votant à considérer. Notons finalement que nous considérons deux manières de contrôler le nombre d'itérations de CqBoost : soit en manipulant la précision ϵ, soit en contrôlant directement le nombre d'itérations maximal T.

Tel que voulu, nous avons construit CqBoost, un algorithme itératif minimisant la C-borne empirique. À la limite, la solution de CqBoost converge vers celle de la variante de MinCq de l'équation (5.9) obtenue en enlevant la contrainte de quasi-uniformité. Cependant, le retrait de cette contrainte retire également les garanties de généralisation qu'avait MinCq via les PAC-bornes 3.23 et 3.30. Dans la prochaine section, nous discutons de l'impact du choix du nombre d'itérations sur le pouvoir généralisation de l'algorithme.

5.4 Garanties de généralisation

L'algorithme MinCq présenté au chapitre 4 est restreint aux distributions quasi-uniformes, ce qui lui procure des garanties de généralisation via les PAC-bornes 3.23 et 3.30. Nous avons vu dans ce chapitre que la quasi-uniformité induit une régularisation de type L_∞ sur le vecteur de poids $\mathbf{q}$, comme chacun des poids q_i est borné supérieurement par $\frac{1}{n}$. Le retrait de cette contrainte dans CqBoost retire également la garantie de généralisation associée.

Dans cette section, nous allons montrer que cet algorithme est tout de même justifié par les PAC-bornes 3.13 et 3.29 du chapitre 3, qui ne restreignent pas la distribution Q à être alignée sur la distribution a priori P, lorsque la valeur du terme $\mathrm{KL}(Q \,\|\, P)$ est contrôlée.

La PAC-borne 3.13 nous dit que pour toute distribution D sur $\mathcal{X} \times \{-1, 1\}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to [-1, 1]$, pour toute distribution a priori P sur $\mathcal{F}$ et tout $\delta \in (0, 1]$, nous avons avec probabilité au moins $1 - \delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,

$$R_D\left(B_Q\right) \;\leq\; 1 - \frac{\left(max\left(0,\, \mu_1\left(M_Q^S\right) - \sqrt{\frac{2}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right)\right)^2}{min\left(1,\, \mu_2\left(M_Q^S\right) + \sqrt{\frac{2}{m}\left[2{\cdot}\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right)},$$

où $\mathrm{KL}(Q \,\|P)$ est la divergence Kullback-Leibler entre les distributions Q et P, définie par $\mathrm{KL}(Q \,\|\, P) \;\triangleq\; \mathbf{E}_{f \sim Q}\, ln\, \frac{Q(f)}{P(f)}$. La PAC-borne 3.29 est similaire, mais donne des garanties de généralisation dans le cas où les votants sont des fonctions noyaux définies à l'aide d'un exemple de l'ensemble d'entraînement.

Cette borne de généralisation nous indique que pour que CqBoost soit en mesure de bien généraliser, il doit :

1. trouver une distribution Q qui a une petite valeur de C-borne empirique C_Q^S ;
2. trouver une distribution Q dont la valeur de $\mathrm{KL}(Q \,\|\, P)$ n'est pas trop grande.

Le premier point est exactement ce que CqBoost cherche à faire. Le second point, lui, n'est pas directement contrôlé par l'algorithme. Énonçons d'abord une borne supérieure sur la valeur de $\mathrm{KL}(Q \,\|\, P)$ dans le cas où $\mathcal{F}$ est un ensemble symétrique fini de $2n$ votants. Considérons également la distribution *a priori* P uniforme, c'est-à-dire une distribution où chaque votant (original et son complément) obtient un poids de $\frac{1}{2n}$. Étant donné que $\mathcal{F}$ est symétrique, cette distribution est équivalente à avoir un poids de 0 sur tous les votants. Nous avons donc :

$$\mathrm{KL}(Q \,\|\, P) \;=\; \mathop{\mathbf{E}}_{f\sim Q} ln\, \frac{Q(f)}{P(f)} \;=\; \mathop{\mathbf{E}}_{f\sim Q} ln\, \frac{Q(f)}{1/2n} \;=\; \mathop{\mathbf{E}}_{f\sim Q} ln\,(2nQ(f)) \;\leq\; \mathop{\mathbf{E}}_{f\sim Q} ln\, 2n \;=\; ln\, 2n\,.$$

Dans le cas où $n \ll m$, trouver la distribution Q minimisant la C-borne empirique C_Q^S correspond à minimiser les PAC-bornes 3.13 ou 3.29 (dépendamment du choix de votants). Cependant, dans les situations où le nombre de votants considérés est très grand, un contrôle supplémentaire sera nécessaire. Nous verrons à la section 5.4.2 que le nombre maximal d'itérations T a un tel effet de contrôle.

Mais d'abord, nous introduisons dans la prochaine section deux possibilités d'ensembles $\mathcal{F}$ continus, pour lesquels les PAC-bornes 3.13 et 3.29 sont valides, fournissant ainsi à CqBoost la possibilité d'être exécuté sur des ensembles infinis de votants.

5.4.1 Généralisation aux ensembles infinis de votants

L'un des avantages d'un algorithme itératif tel que CqBoost est qu'il est possible de considérer des ensembles infinis $\mathcal{F}$ de votants. À chaque itération, il s'agit simplement de *générer* un nouveau votant qui viole la contrainte duale, plutôt que d'en *choisir* un parmi une liste prédéfinie. Les autres algorithmes basés sur la génération de colonne permettent également de considérer une infinité de votants.

Nous devons par contre valider que la théorie PAC-bayésienne sur laquelle est basée CqBoost permet une telle généralisation. Pour y arriver, nous devons généraliser trois quantités : la marge $M_Q(x, y)$, ce à quoi correspond le poids $Q(f_i)$ sur un votant $f_i \in \mathcal{F}$, et la divergence Kullback-Leibler $\mathrm{KL}(Q \,\|\, P)$.

Quand $\mathcal{F}$ correspond au cas simple d'un ensemble fini de votants, un poids $Q(f_i)$ dénote simplement la masse de probabilité de la distribution Q sur le votant $f_i \in \mathcal{F}$. Nous considérons ici deux cas d'ensembles continus $\mathcal{F}$, en nous limitant à une gaussienne centrée en 0 et de

variance 1 pour la distribution a priori P, et à des mixtures de gaussiennes de variance 1 pour la distribution a posteriori Q.[5]

- Le premier cas est l'ensemble des fonctions linéaires $x \mapsto \left\langle \mathbf{w}, \boldsymbol{\phi}(x) \right\rangle \in \mathbb{R}$, où $\boldsymbol{\phi}$ est une fonction qui transforme un exemple x vers un espace de caractéristiques de grande dimension, $\mathbf{w}$ est un vecteur de poids arbitraire dans l'espace des vecteurs de caractéristiques, et $\langle , \rangle$ correspond au produit scalaire dans cet espace.
- Le second cas est lorsque $\mathcal{F}$ correspond à l'ensemble des classificateurs linaires $x \mapsto \operatorname{sgn}\left(\left\langle \mathbf{w}, \boldsymbol{\phi}(x) \right\rangle\right) \in \{-1, +1\}$.

Dans ces deux situations, la fonction $\boldsymbol{\phi}$ est fixe, et $\mathcal{F}$ correspond à l'ensemble des vecteurs de poids dans l'espace des caractéristiques. À chaque itération t, l'algorithme d'apprentissage est à la recherche d'une distribution Q qui consiste en une mixture de t gaussiennes isotropes. Alors, pour tout $\mathbf{v} \in \mathcal{F}$,

$$Q(\mathbf{v}) \;=\; \sum_{i=1}^{t} q_i \left(\frac{1}{\sqrt{2\pi}} \right)^N exp\left(-\frac{1}{2} \|\mathbf{v} - \mathbf{w}_i\|^2 \right), \tag{5.18}$$

où q_i correspond au poids associé à la gaussienne centrée sur $\mathbf{w}_i$, N correspond à la dimension[6] de $\mathbf{v}$, et $\|\cdot\|$ dénote la norme Euclidienne. Pour considérer de tels ensembles infinis dans l'algorithme 5, l'étape 4 est remplacée par la génération d'un vecteur $\mathbf{w}_i \in \mathcal{F}$ ayant une grande valeur de « edge », violant la contrainte du problème d'optimisation dual de l'équation (5.17). Une fois $\mathbf{w}_t$ choisi, les poids $q_t, q_{t-1}, \ldots, q_1$ seront trouvés de la même manière que lorsque nous considérons un ensemble fini de votants.

La marge $M_Q(x, y)$ d'une mixture de gaussiennes Q sur l'ensemble des fonctions linéaires est donnée par

$$M_Q(x, y) \;=\; \int_{\mathcal{F}} Q(\mathbf{v})\, y \left\langle \mathbf{v}, \boldsymbol{\phi}(x) \right\rangle d\mathbf{v} \;=\; \sum_{i=1}^{t} q_i\, y \left\langle \mathbf{w}_i, \boldsymbol{\phi}(x) \right\rangle,$$

et dans le cas des classificateurs linéaires, nous avons

$$M_Q(x, y) \;=\; \int_{\mathcal{F}} Q(\mathbf{v}) \operatorname{sgn}\left(y \left\langle \mathbf{v}, \boldsymbol{\phi}(x) \right\rangle \right) d\mathbf{v} \;=\; \sum_{i=1}^{t} q_i\, F\left(y \left\langle \mathbf{w}_i, \boldsymbol{\phi}(x) \right\rangle \right),$$

où $F : \mathbb{R} \to [-1, +1]$ est une fonction monotone croissante de type sigmoïde, obtenue par la cumulative de gaussiennes suivante :

$$F\left(y \left\langle \mathbf{w}_i, \boldsymbol{\phi}(x) \right\rangle \right) \;\triangleq\; -1 + \sqrt{\frac{2}{\pi}} \int_{-\infty}^{y \langle \mathbf{w}_i, \boldsymbol{\phi}(x) \rangle} e^{-\frac{1}{2} z^2}\, dz\,.$$

5. Le choix des gaussiennes pour les ensembles infinis de classificateurs est commun dans la littérature PAC-bayésienne, comme ce choix permet d'obtenir des expressions analytiques pour le vote de majorité, la marge et $\mathrm{KL}(Q \,\|P)$. Voir Langford et Shawe-Taylor (2003) et Germain, Lacasse, Laviolette et Marchand (2009) pour plus de détails.

6. Pour considérer des espaces de caractéristiques de dimension infinie, chaque gaussienne peut être remplacée par un processus gaussien.

Étant donné un vecteur $\mathbf{w}$, désignons par $G_{\mathbf{w}}$ la gaussienne isotrope de variance unitaire centrée sur $\mathbf{w}$. Considérons la gaussienne $G_{\mathbf{0}}$ comme distribution a priori P. Avec ces définitions, lorsque la distribution Q est donnée par l'équation 5.18, nous avons

$$\begin{aligned}
\mathrm{KL}(Q \parallel P) &= \int_{\mathcal{F}} \sum_{i=1}^{t} q_i G_{\mathbf{w}_i}(\mathbf{v}) \, ln \, \frac{\sum_{i=1}^{t} q_i G_{\mathbf{w}_i}(\mathbf{v})}{G_{\mathbf{0}}(\mathbf{v})} d\mathbf{v} \\
&\leq \int_{\mathcal{F}} \sum_{i=1}^{t} q_i G_{\mathbf{w}_i}(\mathbf{v}) \, ln \, \frac{G_{\mathbf{w}_i}(\mathbf{v})}{G_{\mathbf{0}}(\mathbf{v})} d\mathbf{v} \\
&= \frac{1}{2} \sum_{i=1}^{t} q_i \|\mathbf{w}_i\|^2 ,
\end{aligned}$$

où l'inégalité est une application de l'inégalité de Jensen (lemme A.3) en annexe A sur la fonction convexe $x \, ln \, x$, et où la dernière égalité est un résultat connu pour les distributions gaussiennes. Comme Q est une distribution et ainsi les q_i somment à un, nous avons que $\mathrm{KL}(Q \parallel P) \ll m$ si la norme Euclidienne de chaque votant $\mathbf{w}_i$ est bornée supérieurement par une constante beaucoup plus petite que m. C'est le cas par exemple pour l'ensemble des fonctions linéaires $x \mapsto \left\langle \mathbf{w}, \boldsymbol{\phi}(x) \right\rangle$ où $\mathbf{w} = \boldsymbol{\phi}(x')$ pour un point arbitraire $x' \in \mathcal{X}$, et où $\|\mathbf{w}\|^2 = \left\langle \boldsymbol{\phi}(x'), \boldsymbol{\phi}(x') \right\rangle \triangleq k(x', x') = 1$ pour un noyau normalisé.

Cette généralisation à deux familles d'ensembles continus $\mathcal{F}$ de votants, et le fait que les bornes de généralisation 3.13 et 3.29 sont valides avec ces ensembles, ouvrent la voie à l'utilisation de CqBoost sur des ensembles de votants infinis. Cette avenue est intéressante à explorer, et est laissée comme travaux futurs. Dans la prochaine section, nous présentons des résultats empiriques illustrant un lien entre la valeur de la C-borne empirique, la valeur de $\mathrm{KL}(Q \parallel P)$ et le nombre d'itérations effectuées par l'algorithme CqBoost.

5.4.2 Contrôle des garanties de généralisation par le nombre d'itérations

À chaque itération, CqBoost choisit et ajoute un nouveau votant à intégrer au vote de majorité. Plus le nombre de votants choisis est *élevé*, plus la fonction de classification induite par le vote de majorité a la possibilité d'être *complexe*.

Les bornes de généralisation classiques, telles que les bornes *Vapnik-Chervonenkis* (*VC*) (Blumer et al., 1989), sont connues pour se dégrader lorsque la capacité de classification augmente. Ce type de borne prédirait donc que le nombre d'itérations ne devrait pas être trop élevé. L'argument du *rasoir d'Occam* (Blumer et al., 1990), disant que « les hypothèses suffisantes les plus simples sont les plus vraisemblables », nous disent également qu'à deux votes de majorité avec une performance similaire, le plus simple (celui composé de moins de votants) devrait être le « meilleur ».

Dans les bornes PAC-bayésiennes, la complexité du modèle est capturé par la divergence Kullback-Leibler entre les distributions Q et P. Explorons empiriquement la valeur de $\mathrm{KL}(Q \parallel P)$ en fonction du nombre d'itérations.

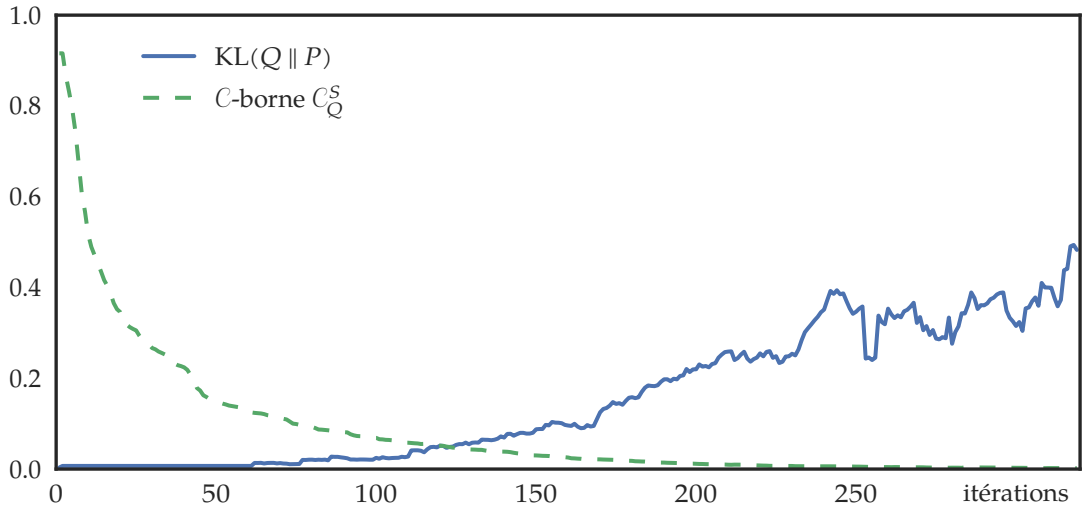

FIG. 5.1: Valeurs de C_Q^S et $\mathrm{KL}(Q \,\|\, P)$ en fonction du nombre d'itérations de CqBoost.

Pour y arriver, nous considérons une distribution P uniforme, c'est-à-dire que chaque votant (et son complément) de l'ensemble symétrique $\mathcal{F}$ a un poids de $\frac{1}{2n}$. Cette distribution est équivalente à ne donner aucun poids aux votants, comme nous initialisons CqBoost. À chaque itération de CqBoost, nous calculons la valeur de la C-borne empirique C_Q^S, et la valeur de $\mathrm{KL}(Q \,\|\, P)$, où nous convertissons la distribution Q retournée par CqBoost, qui n'est définie que sur les t votants choisis (à chaque itération t) en une distribution équivalente sur tous les votants de $\mathcal{F}$. Cette distribution équivalente normalise le poids des votants choisis (et leur complément), tout en laissant les votants non considérés à leur poids uniforme $\frac{1}{2n}$. La figure 5.1 montre les valeurs pour la C-borne empirique C_Q^S et $\mathrm{KL}(Q \,\|\, P)$ à chaque itération, lors de l'exécution de CqBoost sur l'ensemble de données *Ringnorm*, provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (LICHMAN, 2013). Parmi les 7400 exemples de cet ensembles de données, 200 exemples ont été sélectionnés aléatoirement pour former un ensemble d'entraînement S sur lequel CqBoost a été exécuté, et le reste des exemples constituent l'ensemble de test T. L'ensemble $\mathcal{F}$ considéré est un ensemble de 10 souches de décision par attribut, et leur complément.

Cette figure montre que plus le nombre de votants choisis est grand, plus la C-borne empirique C_Q^S est minimisée. Nous constatons également que la valeur de $\mathrm{KL}(Q \,\|\, P)$ a tendance à augmenter en fonction du nombre d'itérations, comme la distribution s'éloigne de plus en plus de la distribution a priori P, qui elle est uniforme. Ce comportement indique que selon les PAC-bornes 3.13 et 3.29, le compromis entre la minimisation de la C-borne empirique et la divergence Kullback-Leibler peut être contrôlé efficacement en choisissant le nombre d'itérations maximal, c'est-à-dire en faisant du «*early-stopping*». La figure 5.2 montre le risque sur l'ensemble d'entraînement S, le risque sur l'ensemble test T et les deux C-bornes associées.

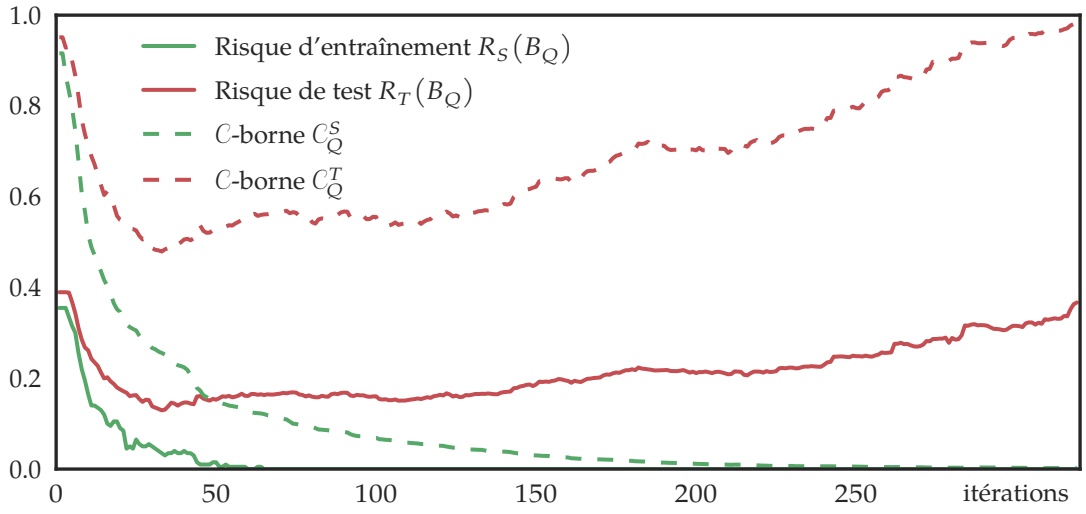

FIG. 5.2: Valeurs de $R_S(B_Q)$, $R_T(B_Q)$, C_Q^S et C_Q^T en fonction du nombre d'itérations de CqBoost.

Cette figure montre que tel que nous l'avons prévu, CqBoost est bel et bien en mesure de faire du surapprentissage de l'ensemble d'entraînement S. On remarque d'abord que le comportement de la C-borne sur les ensembles d'entraînement et de test (le réel estimateur du risque du vote de majorité $R_D(B_Q)$) est très similaire aux valeurs de risque sur ces mêmes ensembles, ce qui confirme dans un premier temps que la C-borne, tant dans sa version empirique C_Q^S que de test C_Q^T, est une bonne fonction substitut pour le risque. À partir d'un certain nombre d'itérations, le risque sur l'ensemble test T se met à augmenter, montrant le phénomène de surapprentissage. Par contre, tel que suggéré par la figure 5.1, ce comportement est également prédit par la théorie. En effet, comme la valeur du KL augmente avec le nombre d'itérations, les garanties de généralisation peuvent se dégrader, dépendamment du compromis entre la minimisation de la C-borne empirique et la valeur du KL. Le choix du nombre d'itérations sera important, celui-ci pouvant être contrôlé via un nombre maximal ou la précision ϵ définis dans l'algorithme 5. Notons que le même phénomène est observé dans d'autres algorithmes de boosting, tel qu'AdaBoost (FREUND et SCHAPIRE, 1997). Notons finalement que ce phénomène de surapprentissage n'est pas observé sur tous les ensembles de données, notamment quand le nombre d'attributs est petit.

5.5 Relations avec d'autres approches de génération de colonnes

Dans cette section, nous discutons des similarités et différences entre CqBoost et trois algorithmes basés sur la génération de colonnes : LPBoost (DEMIRIZ, BENNETT et SHAWE-TAYLOR, 2002), CG-Boost (BI, T. ZHANG et BENNETT, 2004) et MDBoost (SHEN et LI, 2010).

LPBoost est un algorithme de génération de colonnes basé sur la programmation linéaire,

avec une régularisation de type L_1 sur le vecteur de poids $\mathbf{q}$. Les problèmes d'optimisation primal et dual sont :

$$\begin{aligned}
\textbf{Primal :}\quad & \text{Résoudre :} && \underset{\mathbf{q},\boldsymbol{\xi}}{\operatorname{argmin}}\ \mathbf{1}^\top \mathbf{q} + C\,\mathbf{1}^\top \boldsymbol{\xi} \\
& \text{sous contraintes :} && \operatorname{diag}(\mathbf{y})\mathbf{H}\,\mathbf{q} \preceq \mathbf{1} - \boldsymbol{\xi}, \\
& && \mathbf{q} \succeq \mathbf{0}, \quad \boldsymbol{\xi} \succeq \mathbf{0}, \\
\textbf{Dual :}\quad & \text{Résoudre :} && \underset{\boldsymbol{\alpha}}{\operatorname{argmax}}\ \mathbf{1}^\top \boldsymbol{\alpha} \\
& \text{sous contraintes :} && \mathbf{H}^\top \operatorname{diag}(\mathbf{y})\,\boldsymbol{\alpha} \preceq \mathbf{1}, \\
& && \mathbf{0} \preceq \boldsymbol{\alpha} \preceq C\,\mathbf{1},
\end{aligned}$$

où $\mathbf{q}$ est un vecteur de poids sur les votants, $\boldsymbol{\xi}$ est un vecteur de m variables d'écart reliées à la marge sur chaque exemple, où $C > 0$ est un paramètre de régularisation, et $\boldsymbol{\alpha}$ est un vecteur de poids sur les m exemples.

Nous observons d'abord que le dual de LPBoost est similaire au dual de CqBoost (équation (5.17)). Dans les deux cas, le vecteur $\boldsymbol{\alpha}$ représente des poids sur les exemples, et la contrainte d'inégalité guidant le choix du prochain votant dans l'algorithme de génération de colonnes est très similaire. Notons également que LPBoost considère une régularisation de type L_1 sur $\mathbf{q}$, et ainsi pénalisera explicitement les solutions denses.

CG-Boost est une famille d'algorithmes de génération de colonnes qui généralise LPBoost aux fonctions objectif avec régularisation de type L_2. La différence principale avec LPBoost est que le vecteur de poids $\mathbf{q}$ est régularisé avec un norme L_2, ce qui transforme le programme linéaire original en un programme quadratique. Notons que la fonction objectif primale de CqBoost est également quadratique. Les problèmes d'optimisation primal et dual de CG-Boost sont comme suit :

$$\begin{aligned}
\textbf{Primal :}\quad & \text{Résoudre :} && \underset{\mathbf{q},\boldsymbol{\xi}}{\operatorname{argmin}}\ \frac{1}{2}\mathbf{q}^\top \mathbf{q} + C\,\mathbf{1}^\top \boldsymbol{\xi} \\
& \text{sous contraintes :} && \operatorname{diag}(\mathbf{y})\mathbf{H}\,\mathbf{q} + \boldsymbol{\xi} \preceq \mathbf{1}, \\
& && \mathbf{q} \succeq \mathbf{0}, \quad \boldsymbol{\xi} \succeq \mathbf{0}, \\
\textbf{Dual :}\quad & \text{Résoudre :} && \underset{\boldsymbol{\alpha}}{\mathit{argmax}}\ \underset{\mathbf{q}}{\operatorname{argmin}}\ \mathbf{1}^\top \boldsymbol{\alpha} - \frac{1}{2}\mathbf{q}^\top \mathbf{q} \\
& \text{sous contraintes :} && \mathbf{H}^\top \operatorname{diag}(\mathbf{y})\,\boldsymbol{\alpha} \preceq \mathbf{q}, \\
& && \mathbf{0} \preceq \boldsymbol{\alpha} \preceq C\,\mathbf{1},
\end{aligned}$$

où $\mathbf{q}$, $\boldsymbol{\xi}$ et $\boldsymbol{\alpha}$ ont la même signification que dans LPBoost, mais où les variables primales $\mathbf{q}$ ne disparaissent pas dans le problème dual. Néanmoins, nous remarquons qu'à nouveau, la contrainte d'inégalité du problème dual est très similaire à celle de LPBoost et de CqBoost,

induisant ainsi la même heuristique pour sélectionner la prochaine colonne à chaque itération.

MDBoost débute avec les mêmes buts en tête que CqBoost : son problème d'optimisation travaille directement sur la distribution des marges. Sa fonction objectif minimise la variance de la marge, en maximisant son premier moment. Un hyperparamètre D contrôle le compromis entre ces deux quantités. Afin de mieux illustrer la similarité avec la fonction objectif de CqBoost, commençons par réécrire la fonction objectif de MDBoost en fonction des deux premiers moments de la marge. Démarrons de Shen et Li (2010, équation (4)), avec notre notation, comme la notation originale de MDBoost rend la comparaison avec les autres approches plus difficile.

$$\begin{aligned} \text{Résoudre} \quad & \underset{\mathbf{q},\gamma}{\operatorname{argmin}} \quad \frac{D}{2(m-1)} \sum_{k>l} (\gamma_k - \gamma_l)^2 - \sum_{k=1}^{m} \gamma_k \\ \text{sous contraintes} \quad & \mathbf{1}^\top \mathbf{q} \;=\; 1\,, \\ & \boldsymbol{\gamma} \;=\; \operatorname{diag}(\mathbf{y})\, \mathbf{H}\, \mathbf{q}\,, \\ & \mathbf{q} \;\geq\; \mathbf{0}\,. \end{aligned}$$

La fonction objectif peut être réécrite de la manière suivante :

$$\begin{aligned} & \frac{D}{2(m-1)} \sum_{k=1}^{m} \sum_{l>m} (\gamma_l - \gamma_k)^2 - \sum_{k=1}^{m} \gamma_k \\ &= \frac{D}{4(m-1)} \sum_{k=1}^{m} \sum_{l\neq k} (\gamma_l - \gamma_k)^2 - \sum_{k=1}^{m} \gamma_k \\ &= \frac{D}{4(m-1)} \sum_{k=1}^{m} \sum_{l=1}^{m} (\gamma_l - \gamma_k)^2 - \sum_{k=1}^{m} \gamma_k \\ &= \frac{D}{4(m-1)} \left(2m \sum_{k=1}^{m} \gamma_k^2 - 2 \left(\sum_{k=1}^{m} \gamma_k \right)^2 \right) - \sum_{k=1}^{m} \gamma_k \\ &= \frac{D}{4(m-1)} \left(2m^2\, \mu_2\big(M_Q^S\big) - 2 \left(m\, \mu_1\big(M_Q^S\big) \right)^2 \right) - m\, \mu_1\big(M_Q^S\big) \\ &= \frac{m^2 D}{2(m-1)} \left(\mu_2\big(M_Q^S\big) - \left(\mu_1\big(M_Q^S\big) \right)^2 \right) - m\, \mu_1\big(M_Q^S\big)\,. \end{aligned}$$

En choisissant $D' = \frac{m\,D}{2(m-1)}$ et en divisant par m, la fonction objectif de MDBoost correspond à minimiser

$$D' \left(\mu_2\big(M_Q^S\big) - \left(\mu_1\big(M_Q^S\big) \right)^2 \right) - \mu_1\big(M_Q^S\big)\,.$$

Notons que cette stratégie de minimisation est également indirectement justifiée par la C-borne, que les auteurs ne semblent pas connaître. Le compromis entre les deux premiers moments de la marge est cependant différent de CqBoost, pour qui la fonction objectif est directement inspirée d'une borne supérieure du risque du vote de majorité. En comparant

les deux fonctions objectif, nous constatons que MDBoost aura tendance à choisir des distributions Q avec un plus grand premier moment de la marge, alors que CqBoost fournit un contrôle direct sur ce premier moment. Les problèmes d'optimisation primal et dual de MDBoost sont comme suit :

$$\begin{aligned}
\textbf{Primal :}\quad & \text{Résoudre :} && \underset{\mathbf{q},\gamma}{\operatorname{argmin}} \ \frac{D}{2}\,\gamma^\top \mathbf{A}\gamma \ - \ \mathbf{1}^\top\gamma \\
& \text{sous contraintes :} && \gamma \ = \ \operatorname{diag}(\mathbf{y})\mathbf{H}\,\mathbf{q}\,, \\
& && \mathbf{q} \ \geq \ \mathbf{0}, \quad \mathbf{1}^\top\mathbf{q} \ = \ 1\,, \\
\\
\textbf{Dual :}\quad & \text{Résoudre :} && \underset{\boldsymbol{\alpha},\nu}{\operatorname{argmin}} \ \frac{1}{2}\,(\boldsymbol{\alpha}-\mathbf{1})^\top \mathbf{A}^\dagger\,(\boldsymbol{\alpha}-\mathbf{1}) + D\nu \\
& \text{sous contraintes :} && \mathbf{H}^\top \operatorname{diag}(\mathbf{y})\,\boldsymbol{\alpha} \ \preceq \ \nu\mathbf{1}\,,
\end{aligned}$$

où $\mathbf{A}$ est une matrice de $m \times m$ éléments telle que $A_{ii} = 1$ pour $i \in \{1,\dots,m\}$, $A_{ij} = \frac{-1}{m-1}$ for $i \neq j$, et où $\mathbf{A}^\dagger$ est la pseudo-inverse de $\mathbf{A}$. Encore une fois, nous remarquons que la stratégie pour choisir le prochain votant dans l'algorithme de génération de colonnes sera le même que tous les algorithmes mentionnés ci-haut. Notons également que pour simplifier la comparaison avec CqBoost, la version de MDBoost mentionnée dans cette thèse est celle définie en fonction de la marge normalisée (dont les éléments du vecteur $\mathbf{q}$ somment a 1 et est donc une distribution), alors que Shen et Li (2010) proposent également une version où la marge n'est pas normalisée.

Finalement, notons que Shen, Li et Hengel (2013) ont développé un cadre général de génération de colonnes nommé CGBoost (à ne pas confondre avec CG-Boost de Bi, T. Zhang et Bennett (2004)), à partir duquel il est possible de retrouver plusieurs variantes d'algorithmes de boosting, en choisissant une fonction de perte et une fonction de régularisation. Il est possible que l'on puisse retrouver le problème d'optimisation de CqBoost à partir de ce cadre général, bien que nous n'y soyons pas arrivé. Ceci étant dit, comme nous l'avons montré, CqBoost est motivé par une borne de généralisation PAC-bayésienne, ce qui en fait une contribution en soit. Il n'est pas connu pour l'instant si la forme générale proposée par Shen, Li et Hengel (2013) peut être motivée par une borne de généralisation.

Dans la prochaine section, nous évaluons empiriquement la performance de CqBoost, en terme de risque et de parcimonie, en le comparant à MDBoost, LPBoost, CG-Boost, AdaBoost, MinCq et au SVM.

5.6 Expérimentations

Nous comparons maintenant la performance de CqBoost au niveau du risque et de la parcimonie, premièrement contre les algorithmes MDBoost, LPBoost et CG-Boost, mais également contre AdaBoost (Freund et Schapire, 1997), SVM (Cortes et Vapnik, 1995) et MinCq. Nous

considérons dans un premier temps des souches de décision comme votants. Pour chaque attribut, 10 souches de décision et leur inverse sont générés.

Nous exécutons tous les algorithmes sur des tâches de classification binaire classiques provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (Lichman, 2013). Tout comme dans les expérimentations du chapitre 4, les données sont normalisées en utilisant une tangente hyperbolique, et la moitié des exemples ont été choisis au hasard pour former un ensemble d'entraînement S jusqu'à un maximum de 500 exemples. Les exemples restants forment un ensemble test T permettant d'évaluer la performance des algorithmes.

La valeur des hyperparamètres est choisie par validation croisée à 5 plis, sur l'ensemble d'entraînement, parmi 15 valeurs sur une échelle logarithmique. La valeur de l'hyperparamètre μ de CqBoost et MinCq est choisi parmi des valeurs entre 10^{-2} et $10^{-0.5}$. La valeur de l'hyperparamètre D de MDBoost est choisi parmi des valeurs entre 10^{0} et 10^{2}. La valeur de l'hyperparamètre C de CG-Boost et LPBoost est choisi parmi des valeurs entre 10^{-3} et 10^{3}. Le nombre d'itérations d'AdaBoost est choisi parmi des valeurs entre 10^{2} et 10^{6}. Finalement, le critère de précision ϵ déterminant le nombre d'itérations de tous les algorithmes basés sur la génération de colonne est fixé à 10^{-6}. Il est donc possible que CqBoost ait fait du surapprentissage sur certains ensembles de données, mais nous n'avons pas ajouté d'hyperparamètre supplémentaire afin de rendre la comparaison plus juste. Tous les programmes linéaires et quadratiques sont résolus à l'aide du solveur CVXOPT (Dahl et Vandenberghe, 2007).

La table 5.1 montre, pour chaque algorithme, le risque sur l'ensemble test et le nombre de poids non nuls dans le vote de majorité retourné par l'algorithme. En termes de risque, nous observons que CqBoost est très compétitif avec tous les autres algorithmes basés sur la génération de colonnes. En effet, il gagne (ou égalise) 13 fois sur un total de 23 ensembles de données. L'algorithme arrivant second est MDBoost, avec 10 victoires, suivi de près par CG-Boost avec 9 victoires. Finalement, LPBoost ne remporte que 7 fois. Lorsque nous comparons tous les algorithmes entre eux, CqBoost est celui qui a le plus grand nombre de victoires. Le « sign test » indique que CqBoost offre une performance significativement supérieure à LP-Boost et MinCq. Ce résultat suggère que l'utilisation de la parcimonie comme régularisateur est supérieur à l'utilisation de la quasi-uniformité dans ce contexte. Les tests statistiques suggèrent également que CqBoost et MDBoost offrent une performance très similaire, ce qui est un résultat attendu puisque les deux algorithmes minimisent une quantité qui est très reliée. En termes de parcimonie, nous observons que CqBoost atteint son but de retourner des votes de majorité beaucoup plus parcimonieux que MinCq. Nous obtenons en prime la surprise que celui-ci offre une performance significativement supérieure. Également, nous observons que LPBoost est l'algorithme retournant les distributions les plus parcimonieuses. Sa performance en termes de risque est cependant moins bonne que CqBoost. Ces expérimentations suggèrent que CqBoost offre un excellent compromis entre parcimonie et précision.

Ensemble de données	CqBoost^S	MDBoost^S	LPBoost^S	CG-Boost^S	AdaBoost^S	MinCq^S
australian	0.1507 *(40)*	**0.1420*** ***(38*)***	0.1710 *(39)*	0.2000 *(140)*	0.1942 *(90)*	0.1449 *(280)*
balance	**0.0256*** *(17)*	**0.0256*** *(24)*	0.0288 ***(16*)***	0.0288 *(40)*	0.0256* *(16*)*	0.0288 *(80)*
breast	0.0372 *(46)*	0.0372 *(44)*	0.0372 ***(20*)***	**0.0344*** *(90)*	0.0430 *(48)*	0.0372 *(180)*
bupa	**0.2890*** ***(31*)***	**0.2890*** *(35)*	0.3121 *(39)*	0.3237 *(60)*	0.3006 *(71)*	0.4509 *(60)*
car	0.1401 ***(16*)***	0.1547 *(22)*	**0.1376** ***(16*)***	**0.1376** *(66)*	0.1336* *(17)*	0.1409 *(120)*
cmc	0.3001 *(21)*	**0.2991*** *(29)*	0.3114 ***(18*)***	0.3032 *(90)*	0.3176 *(34)*	0.3052 *(180)*
credit	**0.1246*** *(29)*	0.1275 *(31)*	0.1333 ***(1*)***	0.1304 *(150)*	0.1681 *(87)*	0.1275 *(300)*
cylinder	0.3296 ***(18*)***	0.3111 *(25)*	0.2889 *(52)*	**0.2704*** *(330)*	0.2852 *(165)*	0.2889 *(660)*
ecoli	**0.0536*** *(27)*	0.0595 ***(22*)***	0.0952 *(25)*	0.0774 *(71)*	0.0833 *(43)*	0.0774 *(140)*
glass	0.2150 *(45)*	**0.2056*** *(55)*	0.3084 ***(43*)***	0.2243 *(90)*	0.2056* *(57)*	0.2710 *(180)*
heart	**0.1852** *(26)*	0.2000 ***(14*)***	0.1926 ***(14*)***	0.2000 *(130)*	0.2148 *(59)*	0.1704* *(260)*
hepatitis	**0.1429*** *(38)*	0.2078 ***(9*)***	0.1558 *(17)*	0.1948 *(190)*	0.1948 *(56)*	0.1688 *(380)*
horse	**0.1739*** *(43)*	**0.1739*** *(57)*	0.2065 ***(6*)***	0.1902 *(261)*	0.1957 *(101)*	0.2228 *(520)*
ionosphere	0.1314 *(90)*	0.1143 *(103)*	0.1314 ***(70*)***	**0.1086*** *(330)*	0.1200 *(162)*	0.1314 *(660)*
monks	**0.2315*** *(10)*	**0.2315*** ***(9*)***	**0.2315*** *(11)*	**0.2315*** *(109)*	0.2546 *(15)*	0.2361 *(120)*
mushroom	0.0042 *(54)*	0.0195 ***(27*)***	**0.0031*** *(37)*	0.0100 *(207)*	0.0043 *(50)*	0.0042 *(420)*
optdigits	0.0843 *(174)*	0.0837 *(151)*	0.0825 ***(88*)***	**0.0800*** *(590)*	0.0834 *(272)*	0.0849 *(1180)*
pima	**0.2370*** *(26)*	0.2448 ***(24*)***	0.2786 *(50)*	0.2500 *(80)*	0.2708 *(90)*	0.2422 *(160)*
titanic	**0.2222** *(6)*	0.2328 *(29)*	**0.2222** ***(5*)***	**0.2222** *(30)*	0.2222 *(7)*	0.2116* *(60)*
vote	**0.0507*** *(33)*	**0.0507*** *(13)*	**0.0507*** ***(1*)***	**0.0507*** *(155)*	0.0507* *(23)*	0.0507* *(320)*
wine	**0.0674** *(59)*	**0.0674** *(49)*	**0.0674** ***(27*)***	0.0787 *(130)*	0.0449* *(57)*	0.0562 *(260)*
yeast	0.2907 *(19)*	0.2907 ***(15*)***	0.2988 *(27)*	**0.2886*** *(80)*	0.2988 *(72)*	0.3028 *(160)*
zoo	**0.0392*** *(20)*	**0.0392*** *(54)*	**0.0392*** ***(10*)***	0.1373 *(134)*	0.1176 *(10*)*	0.0392* *(320)*

Comparaison statistique de CqBoost^S versus les autres algorithmes

Poisson binomial test	68%	83%	78%	90%	73%
Sign test (valeur-p)	0.40	0.04	0.06	0.06	0.05

TAB. 5.1: Comparaison du risque $R_T(B_Q)$ et de parcimonie (nombre de votants au poids non nul) entre CqBoost, MDBoost, LPBoost, CG-Boost, AdaBoost et MinCq, en utilisant des souches de décision comme votants. Une valeur en gras indique que l'algorithme a obtenu la meilleure performance parmi les algorithmes basés sur la génération de colonnes. Une étoile indique que la performance est la meilleure parmi tous les algorithmes confondus. La table présente également le résultat de tests statistiques.

Ensemble de données	CqBoost$^K_\gamma$	MDBoost$^K_\gamma$	LPBoost$^K_\gamma$	CG-Boost$^K_\gamma$	AdaBoost$^K_\gamma$	MinCq$^K_\gamma$	SVM$^K_\gamma$
australian	0.1478 ***(26)****	0.1449 *(36)*	0.1449 *(73)*	**0.1362** *(345)*	0.1565 *(93)*	0.1391 *(690)*	0.1333* *(218)*
balance	0.0543 *(24)*	0.0383 *(89)*	**0.0288*** ***(23)****	0.0319 *(313)*	0.0319 *(39)*	0.0575 *(624)*	0.0351 *(37)*
breast	**0.0401*** *(28)*	**0.0401*** *(49)*	**0.0401*** ***(6)****	**0.0401*** *(350)*	0.0401* *(53)*	0.0401* *(700)*	0.0401* *(51)*
bupa	**0.2775** ***(30)****	**0.2775** *(31)*	0.2948 *(38)*	0.2832 *(174)*	0.2832 *(50)*	0.2948 *(344)*	0.2717* *(110)*
car	0.1002 ***(30)****	0.0537 *(166)*	**0.0342*** *(87)*	0.1971 *(504)*	0.2679 *(74)*	0.3013 *(1000)*	0.0342* *(97)*
cmc	0.3217 *(32)*	**0.3124** ***(29)****	0.3227 *(30)*	0.3217 *(501)*	0.3124 *(66)*	0.3135 *(1000)*	0.3063* *(323)*
credit	**0.1304*** ***(24)****	**0.1304*** *(131)*	0.1391 *(73)*	0.1333 *(345)*	0.1449 *(84)*	0.1362 *(690)*	0.1304* *(118)*
cylinder	0.3370 *(18)*	**0.2704** *(193)*	0.3593 ***(17)****	0.3630 *(270)*	0.3000 *(87)*	0.2926 *(540)*	0.2667* *(152)*
ecoli	**0.0595*** *(25)*	0.0655 *(48)*	0.1012 ***(12)****	0.1131 *(169)*	0.0952 *(46)*	0.0893 *(336)*	0.1012 *(42)*
glass	0.2056 ***(38)***	**0.1869*** *(44)*	0.2150 ***(38)***	0.2897 *(110)*	0.2336 *(37)**	0.2617 *(214)*	0.1869* *(64)*
heart	**0.1481*** *(24)*	**0.1481*** *(27)*	0.1630 ***(14)****	0.1704 *(135)*	0.1481* *(44)*	0.1556 *(270)*	0.1556 *(87)*
hepatitis	0.1688 *(19)*	**0.1558*** *(70)*	0.1818 ***(18)****	0.1948 *(78)*	0.1818 *(51)*	0.2078 *(156)*	0.1818 *(33)*
horse	0.1739 ***(25)****	0.1630 *(28)*	**0.1359*** *(33)*	0.1957 *(184)*	0.1793 *(82)*	0.1848 *(368)*	0.2011 *(85)*
ionosphere	0.1371 ***(26)****	0.1543 *(70)*	**0.0971*** *(45)*	0.1200 *(176)*	0.1257 *(45)*	0.1257 *(352)*	0.0971* *(43)*
monks	**0.2361** ***(21)****	0.2454 *(63)*	0.2454 *(50)*	0.3287 *(216)*	0.2870 *(47)*	0.3426 *(432)*	0.2083* *(96)*
mushroom	0.0677 ***(34)****	**0.0123** *(127)*	0.0261 *(53)*	0.0498 *(502)*	0.0413 *(82)*	0.0900 *(1000)*	0.0081* *(49)*
optdigits	0.0882 ***(27)****	**0.0659*** *(158)*	0.0882 *(77)*	0.0978 *(500)*	0.0870 *(58)*	0.1372 *(1000)*	0.0960 *(77)*
pima	0.2630 *(29)*	0.2578 *(38)*	**0.2474*** ***(18)****	0.2500 *(384)*	0.2526 *(65)*	0.2604 *(768)*	0.2604 *(254)*
titanic	**0.2199*** ***(15)****	**0.2199*** *(24)*	0.2269 *(61)*	0.2222 *(500)*	0.2199* *(19)*	0.2199* *(1000)*	0.2269 *(234)*
vote	**0.0507*** ***(33)****	0.0553 *(107)*	0.0553 *(37)*	0.0553 *(218)*	0.0553 *(48)*	0.0599 *(436)*	0.0507* *(54)*
wine	**0.0337*** *(30)*	**0.0337*** *(29)*	0.0449 ***(16)****	0.0449 *(89)*	0.0449 *(21)*	0.0337* *(178)*	0.0562 *(30)*
yeast	**0.2744*** ***(40)****	0.2774 *(65)*	0.2876 *(88)*	0.2785 *(502)*	0.2815 *(80)*	0.3232 *(1000)*	0.2785 *(337)*
zoo	0.0392 *(22)*	0.0588 *(27)*	**0.0000*** ***(18)***	0.0980 *(50)*	0.0000* *(23)*	0.0392 *(100)*	0.1373 *(12)**

Comparaison statistique de CqBoost$^K_\gamma$ versus les autres algorithmes

Poisson binomial test	21%	53%	88%	57%	88%	38%
Sign test (valeur-p)	0.93	0.19	0.01	0.25	0.03	0.75

TAB. 5.2: Comparaison du risque $R_T(B_Q)$ et de parcimonie (nombre de votants au poids non nul) entre CqBoost, MDBoost, LPBoost, CG-Boost, AdaBoost, MinCq et SVM, en utilisant des noyaux RBF comme votants. Une valeur en gras indique que l'algorithme a obtenu la meilleure performance parmi les algorithmes basés sur la génération de colonnes. Une étoile indique que la performance est la meilleure parmi tous les algorithmes confondus. La table présente également le résultat de tests statistiques.

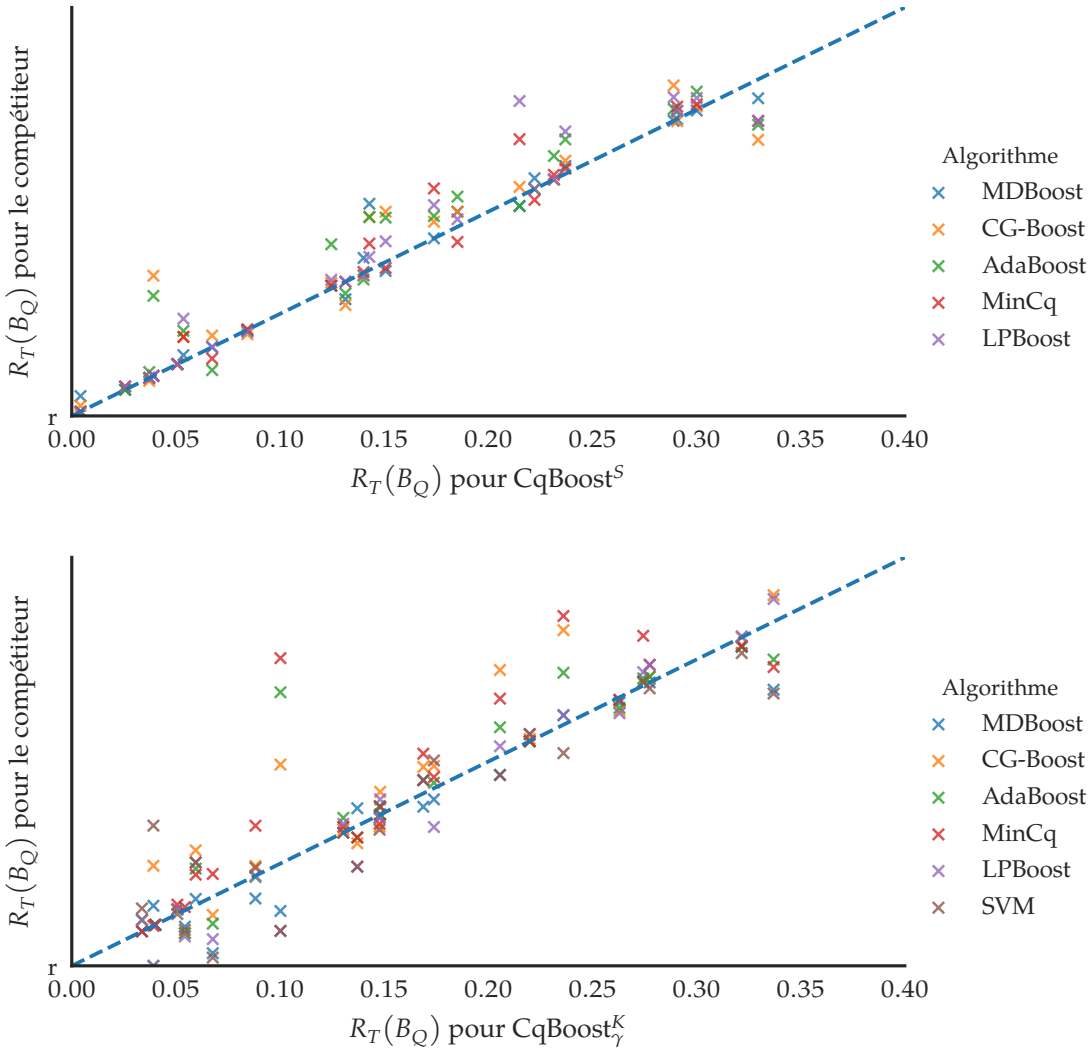

FIG. 5.3: Présentation visuelle du risque sur l'ensemble de test de CqBoost^S et CqBoost$^K_\gamma$ par rapport aux autres algorithmes. Un point au dessus de la ligne diagonale indique une meilleure performance de CqBoost.

La table 5.2 montre des résultats en utilisant des noyaux RBF comme votants. Pour chaque exemple (x, y), nous considérons $f(\cdot) = \pm k(x, \cdot)$, où $k(x, x') = exp\left(-\|x - x'\|^2 \Big/ 2\sigma^2\right)$, et où σ est le paramètre de « largeur » du noyau. Ce paramètre est fixé à la distance moyenne au carré entre les paires d'exemples de l'ensemble d'entraînement. Toutes les valeurs d'hyperparamètres ont été choisis par validation croisée à 5 plis, sur l'ensemble d'entraînement, parmi 15 valeurs sur une échelle logarithmique. La valeur de μ pour CqBoost et MinCq est choisie entre 10^{-4} et 10^{-2}. La valeur de D pour MDBoost est choisie entre 10^2 et 10^6. La valeur de l'hyperparamètre C de LPBoost et CG-Boost est choisie entre 10^{-3} et 10^3, et celui du SVM entre 10^{-4} et 10^4. Le nombre d'itérations d'AdaBoost est choisi entre 10^3 et 10^7. Le paramètre de précision ϵ de tous les algorithmes basés sur la génération de colonnes a été fixé à 10^{-8}.

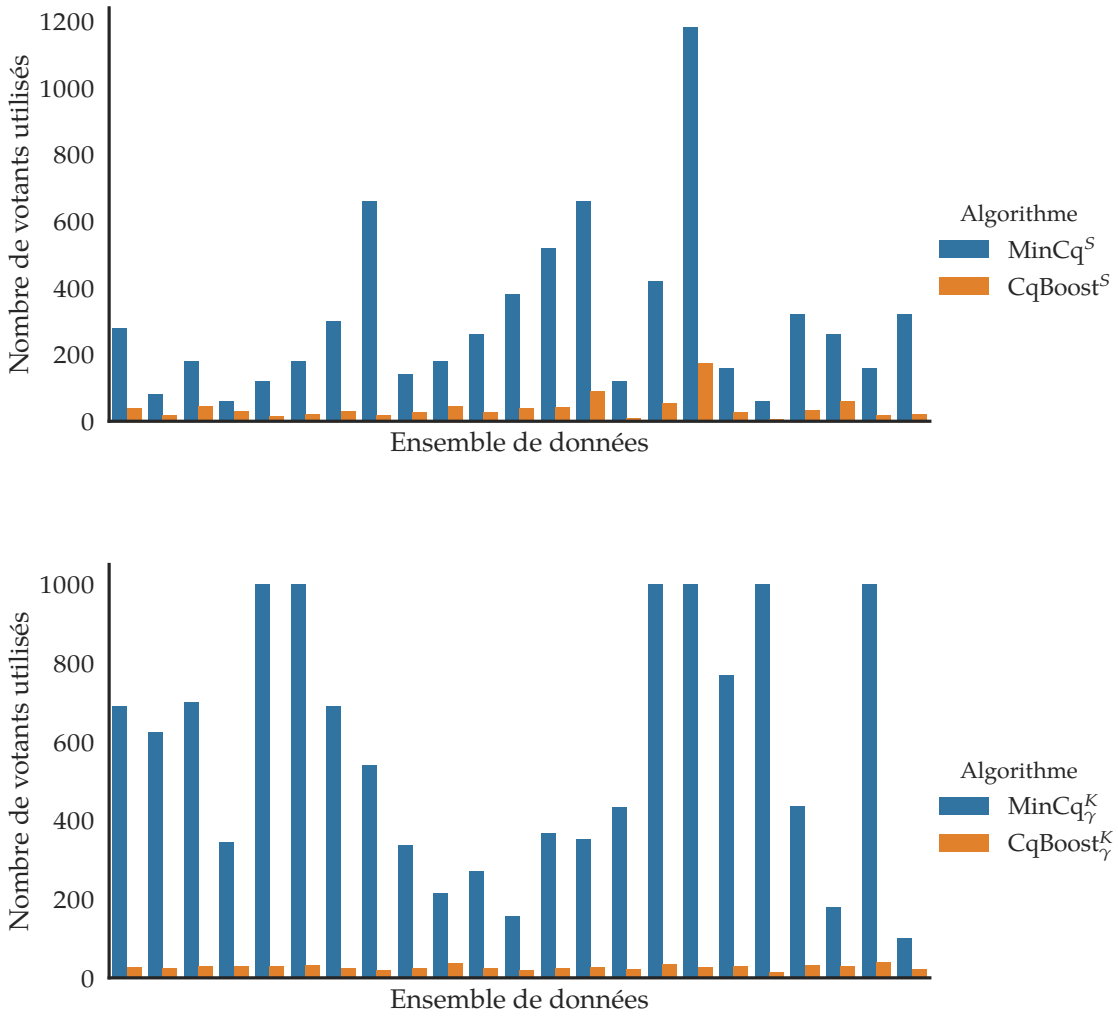

FIG. 5.4: Présentation visuelle du nombre de votants utilisés par CqBoost^S et CqBoost$^K_\gamma$ par rapport à MinCq^S et MinCq$^K_\gamma$. Chaque paire de bandes correspond à un ensemble de données.

Dans cette expérimentation, nous observons que CqBoost, MDBoost et LPBoost ont une performance similaire. Nous remarquons également que MDBoost a une performance légèrement supérieure à CqBoost, mais ce résultat n'est pas statistiquement significatif selon le « sign test ». CqBoost présente une performance supérieure à CG-Boost et MinCq, de manière statistiquement significative. En termes de parcimonie, nous observons que CqBoost atteint encore son but en retournant des votes de majorité significativement plus parcimonieux que MinCq. De manière surprenante encore une fois, celui-ci est également significativement plus performant que MinCq en terme de risque. En utilisant des noyaux RBF comme votants, contrairement aux résultats avec les souches de décision, CqBoost produit des solutions plus parcimonieuses que LPBoost, même si ce dernier pénalise explicitement les solutions denses.

Les figures 5.3 et 5.4 résument une partie des tables de résultats de manière plus visuelle.

Les temps d'exécution de chaque algorithme sur les différents ensembles de données sont rapportés en annexe B. Nous remarquons que malgré que CqBoost soit conçu avec le but de pouvoir considérer un plus grand nombre de votants que MinCq, les temps d'exécution sont plus lents que MinCq sur les ensembles de données considérés dans nos expérimentations. L'analyse de la complexité de CqBoost nous suggère par contre que l'avantage de CqBoost au niveau du temps d'exécution sera de plus en plus observable en fonction de l'augmentation du nombre de votants considérés, s'il existe une solution où le nombre de votants nécessaires est significativement plus petit que le nombre total de votants. Notons que la même observation peut être faite en comparant CG-Boost et SVM : le SVM offre un temps d'exécution de loin inférieur à CG-Boost sur les ensembles de données considérés dans cette thèse. Nous laissons comme travaux futurs l'exécution d'expérimentations à plus grande échelle.

5.7 Conclusion du chapitre

Nous avons proposé un algorithme d'apprentissage itératif, nommé CqBoost, qui

- trouve un vote de majorité parcimonieux minimisant la C-borne empirique, en minimisant le second moment de la marge, tout en contrôlant la valeur du premier moment à l'aide d'un hyperparamètre μ;
- contrairement à MinCq, ne minimise pas directement une borne de généralisation PAC-bayésienne, mais offre un compromis entre la minimisation de la C-borne empirique et la garantie de généralisation via le nombre d'itérations ;
- est donc indirectement motivé par les PAC-bornes 3.13 et 3.30 ;
- peut être appliqué à des ensembles infinis de votants, dont deux exemples ont été fournis ;
- est très compétitif avec l'état de l'art, que ce soit en termes de performance ou de parcimonie.

Comme travaux futurs, il serait intéressant de mener des expérimentations à plus grande échelle, notamment en considérant des ensembles infinis de votants et ainsi ne pas construire la matrice de vote à l'avance mais plutôt générer un bon votant à chaque itération. Il serait également intéressant d'explorer des techniques pour choisir le nombre d'itérations, par exemple en conservant un ensemble de validation sur lequel le risque et la C-borne seront calculés à chaque itération. Finalement, il serait intéressant de vérifier si l'ajout d'une contrainte supplémentaire de régularisation L_∞, telle que la quasi-uniformité apporte dans le cas de MinCq, serait une manière de contrôler le surapprentissage.

Chapitre 6

Votes de majorités à sortie arbitraire et classification multi-classe

Il est bien connu qu'apprendre des prédicteurs en mesure de produire une sortie plus riche que la classification binaire, par exemple la classification multi-classe ou multi-étiquette, et pour qui des garanties théoriques existent est encore aujourd'hui un domaine de recherche actif. D'un point de vue pratique, plusieurs relaxations de problèmes complexes ont été mises au point. Une approche commune consiste à décomposer le problème d'apprentissage complexe en plusieurs problèmes plus simples (Allwein, Schapire et Singer, 2001 ; Tsoumakas et Vlahavas, 2007 ; Read et al., 2011 ; Mroueh et al., 2012 ; Y. Zhang et Schneider, 2012). Par exemple, c'est l'idée principale utilisée par la technique des *Error-Correcting Output Codes* (*ECOC*) (Dietterich et Bakiri, 1995), qui rendent possible la réduction de problèmes multi-classe ou multi-étiquette à plusieurs problèmes de classification binaire.

Dans ce chapitre, nous nous concentrons toujours sur les fonctions de prédiction qui prennent la forme d'un vote de majorité pondéré. Nous généralisons certains éléments théoriques présentés aux chapitres 2 et 3 aux votes de majorité constitués de votants dont l'espace sortie est arbitraire, contrairement aux chapitres précédents qui se concentrent sur la classification binaire. Notamment, nous étudions une notion de marge généralisée, à partir de laquelle nous pouvons retrouver une C-borne générale. Cette C-borne nous permet d'obtenir une borne de généralisation PAC-bayésienne prenant en considération les deux premiers moments de la marge, contrairement à d'autres bornes de généralisation multi-classe de l'état de l'art qui ne considèrent que le premier moment (Morvant, Koço et Ralaivola, 2012 ; Kuznetsov, Mohri et Syed, 2014), et qui ne seront donc pas serrées dans les situations où les votants ne performent pas bien individuellement. Nous proposons également une relaxation de la notion de marge générale, nommée la ω-marge, qui nous permet d'obtenir une C-borne plus simple à calculer, nommée la ω-C-borne. Celle-ci ouvre la voie à la conception d'algorithmes d'apprentissage pour la prédiction à sortie arbitraire, avec garanties théoriques.

Nous fournissons deux spécialisations de ce cadre général, premièrement à la classification multi-classe, puis à la classification multi-étiquette. Dans le cadre de la classification multi-classe, nous montrons que le risque du classificateur par vote de majorité est corrélé avec la C-borne multi-classe.

Une partie de ces résultats théoriques ont été publiés dans un article de «*workshop*» (Laviolette, Morvant, Ralaivola et Roy, 2014), puis une version plus complète a été publiée dans un article de revue (Laviolette, Morvant, Ralaivola et Roy, 2017). Nous présentons dans cette thèse une revue de ces extensions théoriques, avec une notation unifiée avec le reste de la thèse et des bornes PAC-bayésiennes plus générales.

Le reste de ce chapitre est divisé comme suit. La section 6.1 présente notre cadre général de votes de majorité constitués de votants à sortie arbitraire. Les sections 6.2 et 6.3 spécialisent ce cadre général à la classification multi-classe et à la classification multi-étiquette, puis nous concluons à la section 6.4.

6.1 Un cadre général pour les votes de majorité de votants à sortie arbitraire

Dans cette section, nous proposons un cadre général pour les votes de majorité Q-pondérés. Nous présentons une définition générale pour la marge, et proposons une C-borne construite pour les votes de majorités combinant plusieurs votants à sortie arbitraire. Rappelons que ces votants doivent être générés *a priori*, et ainsi sont traités comme des « boîtes noires ». Nous discutons également de la manière d'estimer cette borne à partir d'un ensemble de données S de m exemples, tirés *i.i.d.* d'une distribution inconnue D. Pour y arriver, nous dérivons une borne supérieure PAC-bayésienne sur le risque vote de majorité Q-pondéré $R_D(B_Q)$, considérant l'estimation empirique de la C-borne sur l'ensemble S. Notons que plusieurs notions comme le vote de majorité B_Q, la marge M_Q et la C-borne C_Q^D, sont redéfinis de manière plus générale dans ce chapitre sans pour autant introduire de nouvelle notation pour y référer.

6.1.1 Une C-borne générale pour la prédiction à sortie arbitraire

Étant donné un espace d'entrée $\mathcal{X}$ et un espace de sortie *fini* $\mathcal{Y}$, nous supposons qu'il existe une fonction $\mathbf{Y} : \mathcal{Y} \to \mathbf{H}_\mathcal{Y}$ qui transforme un élément de $\mathcal{Y}$ vers un espace de caractéristiques $\mathbf{H}_\mathcal{Y}$, où $\mathbf{H}_\mathcal{Y}$ est un espace vectoriel tel qu'un espace de Hilbert. Pour simplifier la notation et les équations dans le reste de ce chapitre, nous supposons que tous les vecteurs dans $\mathbf{Y}(\mathcal{Y})$ sont de norme unitaire. La plupart des résultats suivants sont également valides sans cette supposition, mais dans ce cas la notation doit prendre une forme plus complexe que celle que nous proposons dans ce chapitre.

Soit $Im\,\mathcal{Y}$ l'image de $\mathcal{Y}$ sous $\mathbf{Y}(\cdot)$, et $\overline{Im\,\mathcal{Y}}$ ($\subseteq \mathbf{H}_\mathcal{Y}$) son enveloppe convexe. Nous considérons un

ensemble (possiblement infini) de votants $\mathcal{H} \subseteq \left\{\mathbf{h} : \mathcal{X} \to \overline{Im\, \mathcal{Y}}\right\}$. Notons que nous utilisons un symbole en gras pour distinguer les votants et les fonctions qui retournent un vecteur de ceux et celles qui retournent une valeur réelle ou entière.

Notons que le fait de supposer l'existence d'une fonction $\mathbf{Y} : \mathcal{Y} \to \mathbf{H}_{\mathcal{Y}}$ est fréquent dans les méthodes à noyaux (Cortes, Mohri et Weston, 2007 ; Brouard, d'Alché-Buc et Szafranski, 2011 ; Giguere et al., 2014). En effet, une telle fonction projetant les éléments de l'espace de sortie dans un espace de caractéristiques existe toujours si on considère un *noyau de sortie* $k_{\mathcal{Y}} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. De plus, considérer que les vecteurs dans $\mathbf{Y}(\mathcal{Y})$ sont de norme unitaire est équivalent à supposer que le noyau de sortie $k_{\mathcal{Y}}(\cdot, \cdot)$ est normalisé.

Il est intéressant de constater que l'*astuce du noyau* s'applique ici, c'est-à-dire, il est possible de ne considérer que la forme duale, en ne se référant qu'au noyau $k_{\mathcal{Y}}(\cdot, \cdot)$ et jamais explicitement à l'espace de caractéristiques $\mathbf{Y}(\cdot)$, qui peut être très complexe. Finalement, notons qu'une grande variété de noyaux existe dans la littérature. Par exemple, lorsque l'espace de sortie est une chaîne, des noyaux classiques sont le *blended spectrum kernel*, le *N-gram kernel*, le *weighted degree kernel*, etc. Lorsque l'espace de sortie est un graphe, il existe des noyaux comme le *Tanimoto kernel*. Voir Gärtner (2003) pour plus d'informations sur les noyaux existants pour la prédiction structurée.

Pour toute distribution de probabilité Q sur $\mathcal{H}$, nous définissons le vote de majorité B_Q par

$$\begin{aligned}
\forall x \in \mathcal{X}, \quad B_Q(x) &\triangleq \operatorname*{argmin}_{c \in \mathcal{Y}} \left\| \mathbf{Y}(c) - \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} \mathbf{h}(x) \right\|^2 \\
&= \operatorname*{argmin}_{c \in \mathcal{Y}} \left(\left\| \mathbf{Y}(c) \right\|^2 + \left\| \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} \mathbf{h}(x) \right\|^2 - 2 \left\langle \mathbf{Y}(c), \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} \mathbf{h}(x) \right\rangle \right) \\
&= \operatorname*{argmin}_{c \in \mathcal{Y}} \left(1 + \left\| \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} \mathbf{h}(x) \right\|^2 - 2 \left\langle \mathbf{Y}(c), \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} \mathbf{h}(x) \right\rangle \right) \\
&= \operatorname*{argmax}_{c \in \mathcal{Y}} \left\langle \mathbf{Y}(c) \,,\, \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} \mathbf{h}(x) \right\rangle , \qquad (6.1)
\end{aligned}$$

où la troisième égalité est une simplification qui est rendue possible grâce à la supposition que les vecteurs dans $\mathbf{Y}(\mathcal{Y})$ sont de norme unitaire. Notons que nous verrons à la section suivante que dans le cas binaire, nous retrouvons le vote de majorité défini aux chapitres précédents.

Comme dans le cas de la classification binaire, l'objectif est de trouver une distribution Q qui minimise le « vrai risque » $R_D(B_Q)$, donné par

$$R_D(B_Q) = \mathop{\mathbf{E}}_{(x,y) \sim D} I\left(B_Q(x) \neq y\right) .$$

En s'inspirant de la définition de la marge de Breiman (2001), nous proposons la définition suivante pour la marge généralisée, qui mesure la « confiance » de prédiction comme la dis-

tance entre le poids de classification accordé à la bonne prédiction, moins le poids de classification le plus grand accordé à l'une des mauvaises prédictions.

Définition 6.1 (La marge généralisée)**.** Pour tout exemple $(x, y) \in \mathcal{X}\times\mathcal{Y}$ et toute distribution Q sur $\mathcal{H}$, nous définissons la *marge généralisée* $M_Q(x, y)$ du vote de majorité Q-pondéré par

$$M_Q(x,y) \;\triangleq\; \left\langle \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \mathbf{h}(x)\,,\, \mathbf{Y}(y) \right\rangle - \max_{\substack{c\in\mathcal{Y}\\ c\neq y}} \left\langle \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \mathbf{h}(x)\,,\, \mathbf{Y}(c) \right\rangle . \tag{6.2}$$

Les deux premiers moments de la marge selon une distribution D' sont dénotés $\mu_1\big(M_Q^{D'}\big)$ et $\mu_2\big(M_Q^{D'}\big)$, respectivement.

Avec cette définition en main, il est évident que le vote de majorité B_Q fait une erreur sur un exemple (x, y) si et seulement si la marge réalisée sur (x, y) est négative. Nous avons donc, comme dans le cadre de la classification binaire,

$$R_D(B_Q) \;=\; \Pr_{(x,y)\sim D} \big(M_Q(x,y) \leq 0\big) . \tag{6.3}$$

En utilisant exactement la même preuve que pour la C-borne binaire du chapitre 2, nous obtenons la C-borne généralisée suivante.

Théorème 6.2 (C-borne généralisée)**.** *Pour toute distribution Q sur un ensemble $\mathcal{H}$ de votants $\mathbf{h} : \mathcal{X} \to \overline{Im\ \mathcal{Y}}$ et toute distribution D' sur $\mathcal{X}\times\mathcal{Y}$, si $\mu_1\big(M_Q^{D'}\big) > 0$, alors*

$$R_{D'}(B_Q) \;\leq\; 1 - \frac{\big(\mu_1\big(M_Q^{D'}\big)\big)^2}{\mu_2\big(M_Q^{D'}\big)} .$$

Tous les résultats de ce chapitre tiennent pour les deux cas extrêmes : le cas où les votants sont peu performants individuellement, comme nous constatons souvent dans les méthodes d'ensemble, et le cas où les votants sont plus expressifs et performent très bien individuellement. Un exemple typique du premier cas est rencontré lorsque nous utilisons une fonction noyau $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ qui induit un ensemble de votants $\mathcal{H} = \Big\{ k(x, \cdot)\, \mathbf{Y}(y) \mid (x, y) \in S \Big\}$. Cette situation arrive également lorsqu'on considère un ensemble de fonctions de prédiction apprises avec différents hyperparamètres. Ces votants faibles peuvent également être des souches de décision, ou même des arbres de décision de différente profondeur. Le second cas arrive notamment lorsque la tâche est de combiner plusieurs votants obtenus à partir d'autres algorithmes d'apprentissage. C'est par exemple le cas en *apprentissage multi-vue*, lorsqu'on combine plusieurs modèles appris à partir de descriptions différentes des données. Notons que la C-borne binaire a déjà montré sa pertinence dans une telle situation (Morvant, Habrard et Ayache, 2014).

6.1.2 Redécouverte de la classification binaire à partir du cadre général

À partir du cadre général de la section 6.1, plusieurs choix de fonctions $\mathbf{Y}(\cdot)$ permettent de retrouver la classification binaire classique présentée au chapitre 2.

Le choix le plus intuitif serait de considérer $\mathbf{Y} : \{-1,+1\} \to \mathbb{R}$, avec $\mathbf{Y}(+1) = 1$ et $\mathbf{Y}(-1) = -1$, mais ce choix particulier ne donnerait pas une marge dont les valeurs sont dans $[-1,1]$. Pour retrouver directement le cadre binaire du chapitre 2, nous devons utiliser l'espace de caractéristiques proposé ci-bas.

Considérons $\mathbf{Y} : \{-1,+1\} \to \mathbb{R}^2$, avec $\mathbf{Y}(+1) = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)^\top$ et $\mathbf{Y}(-1) = \left(-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)^\top$. Dans ce cas, chaque votant $\mathbf{h}$ retourne un vecteur de $\mathbb{R}^2$, dont la première coordonnée h_1 est un élément de $\left[-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right]$, et la seconde coordonnées est toujours $\frac{1}{\sqrt{2}}$. Nous avons donc

$$\begin{aligned}
B_Q(x) &= \operatorname*{argmax}_{c\in\{-1,+1\}} \left\langle \mathbf{Y}(c)\,,\, \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \mathbf{h}(x) \right\rangle \\
&= \operatorname*{argmax}_{c\in\{-1,+1\}} \left\langle \left(\tfrac{1}{\sqrt{2}}\, c,\, \tfrac{1}{\sqrt{2}}\right)^\top,\, \left(\tfrac{1}{\sqrt{2}} \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \sqrt{2}\, h_1(x)\,,\, \tfrac{1}{\sqrt{2}}\right)^\top \right\rangle \\
&= \operatorname*{argmax}_{c\in\{-1,+1\}} \left[\tfrac{1}{2}\, c \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \sqrt{2}\, h_1(x) \;+\; \tfrac{1}{2} \right] \\
&= \operatorname*{argmax}_{c\in\{-1,+1\}} \left[c \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \sqrt{2}\, h_1(x) \right] \\
&= \operatorname{sgn}\left[\mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \sqrt{2}\, h_1(x) \right] \\
&= \operatorname{sgn}\left[\mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h(x) \right],
\end{aligned}$$

et

$$\begin{aligned}
M_Q(x,y) &= \left\langle \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \mathbf{h}(x)\,,\, \mathbf{Y}(y) \right\rangle - \max_{\substack{c\in\{-1,+1\} \\ c\neq y}} \left\langle \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \mathbf{h}(x)\,,\, \mathbf{Y}(c) \right\rangle \\
&= \left\langle \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \mathbf{h}(x)\,,\, \mathbf{Y}(y) \right\rangle - \left\langle \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \mathbf{h}(x)\,,\, \mathbf{Y}(-y) \right\rangle \\
&= \left(\tfrac{1}{2}\, y \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \sqrt{2}\, h_1(x) \;+\; \tfrac{1}{2} \right) - \left(-\tfrac{1}{2}\, y \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \sqrt{2}\, h_1(x) \;+\; \tfrac{1}{2} \right) \\
&= y \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \sqrt{2}\, h_1(x) \\
&= y \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h(x)\,,
\end{aligned}$$

où pour tout $\mathbf{h} \in \mathcal{H}$, $h(x) \triangleq \sqrt{2} h_1(x)$. Comme $\mathbf{E}_{\mathbf{h}\sim Q}\, h(x)$ représente la marge « binaire » dont la valeur est dans $[-1,1]$ et $y \in \{-1,1\}$ est une étiquette binaire, nous retrouvons les quantités usuelles du chapitre 2.

6.1.3 Un théorème PAC-bayésien pour estimer la $\mathcal{C}$-borne généralisée

Dans cette section, nous discutons brièvement de l'estimation du « vrai » risque du classificateur par vote de majorité, en utilisant la $\mathcal{C}$-borne généralisée empirique $\mathcal{C}_Q^S$ du théorème 6.2. Pour y arriver, nous généralisons la PAC-borne 3.13 à nos définitions généralisées du vote de majorité et de la marge.

La démonstration de cette PAC-borne dépend de deux bornes PAC-bayésiennes. Une première borne inférieurement le « vrai » premier moment de la marge généralisée en fonction de son estimation empirique, et la seconde borne supérieurement le « vrai » second moment de la marge généralisée en fonction de son estimation empirique. Pour arriver à démontrer ces deux bornes, nous avons d'abord besoin d'un théorème PAC-bayésien général permettant de borner des fonctions de pertes qui ne prennent pas leur valeur dans $[0,1]$, mais bien dans $[0,B]$, où $B > 0$ est une borne sur la valeur de sortie de la fonction de perte. Nous introduisons un tel théorème général ainsi que deux corollaires dans la section A.3 de l'annexe A. Le corollaire A.14 de cette section s'énonce comme suit.

Corollaire 6.3. *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to \overline{\mathcal{Y}}$, pour tout $B > 0$, pour toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0,B]$, pour toute distribution a priori P sur $\mathcal{F}$ et pour tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

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

et

$$\mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_D^{\mathcal{L}}(f) \;\geq\; \mathop{\mathbf{E}}_{f\sim Q} \mathbb{E}_S^{\mathcal{L}}(f) - B\sqrt{\frac{1}{2m}\left[\mathrm{KL}(Q\,\|\,P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Nous remarquons que l'introduction d'une borne supérieure B pour la fonction de perte implique simplement l'introduction d'un facteur B devant la racine carrée de la borne inférieure et de la borne supérieure.

Nous avons maintenant les outils nécessaires pour borner les deux premiers moments de la marge généralisée $M_Q(x,y)$. Comme pour tout $x \in \mathcal{X}$ nous avons $\mathbf{h}(x) \in \overline{Im\,\mathcal{Y}}$, et pour tout $c \in \mathcal{Y}$ nous avons $\|\mathbf{Y}(c)\| = 1$, alors $\left\langle \mathbf{E}_{\mathbf{h}\sim Q}\,\mathbf{h}(x), \mathbf{Y}(c)\right\rangle$ a toujours une valeur dans $[-1,1]$. Il s'ensuit de l'équation (6.2) que $B = 2$ est toujours une borne supérieure de $\left|M_Q(x,y)\right|$. Le premier moment de la marge $\mu_1\left(M_Q^{D'}\right)$ prend donc des valeurs dans $[-B,B]$, et le second moment $\mu_2\left(M_Q^{D'}\right)$ des valeurs entre $[-B^2,B^2]$. Nous devons finalement définir une quantité supplémentaire : nous dénotons $M_{\mathbf{h}}(x,y)$ la fonction telle que $\mathbf{E}_{\mathbf{h}\sim Q}\,M_{\mathbf{h}}(x,y) = M_Q(x,y)$. Notons que la valeur de $M_{\mathbf{h}}(x,y)$ est dans $[-B,B]$.

Théorème 6.4. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to \overline{Im\,\mathcal{Y}}$, pour toute distribution a priori P sur $\mathcal{H}$, pour tout $B > 0$ tel que $M_{\mathbf{h}}(x,y) \in [-B,B]$ pour tout*

$\mathbf{h} \in \mathcal{H}$ *et tout* $(x, y) \in \mathcal{X} \times \mathcal{Y}$*, et pour tout* $\delta \in (0, 1]$*, nous avons avec probabilité au moins* $1 - \delta$ *sur le choix d'un ensemble de données de* m *exemples* $S \sim D^m$ *que pour toute distribution a posteriori* Q *sur* $\mathcal{H}$*,*

$$\mu_1\big(M_Q^D\big) \;\geq\; \mu_1\big(M_Q^S\big) - B\sqrt{\frac{2}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Démonstration. Considérons d'abord la fonction de perte

$$\mathcal{L}_{\mu_1}\big(\mathbf{h}(x), y\big) \;\triangleq\; \frac{1}{2}\big(B + M_{\mathbf{h}}(x, y)\big)\,.$$

Comme $M_{\mathbf{h}}(x, y) \in [-B, B]$, nous avons que $\mathcal{L}_{\mu_1}\big(\mathbf{h}(x), y\big) \in [0, B]$, tel que voulu. Nous remarquons également que $\mathbf{E}_{\mathbf{h}\sim Q}\,\mathbb{E}_{D'}^{\mathcal{L}_{\mu_1}} = \frac{1}{2}\left(B - \mu_1\big(M_Q^{D'}\big)\right)$.

En utilisant cette fonction de perte, la borne supérieure du corollaire A.14 appliquée avec $\mathcal{H} = \mathcal{H}$ nous indique que pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to \overline{Im\,\mathcal{Y}}$, pour toute distribution a priori P sur $\mathcal{H}$ et pour tout $\delta \in (0, 1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,

$$\frac{1}{2}\left(B - \mu_1\big(M_Q^D\big)\right) \;\leq\; \frac{1}{2}\left(B - \mu_1\big(M_Q^S\big)\right) + B\sqrt{\frac{1}{2m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Le résultat est obtenu en multipliant par deux et en soustrayant B de chaque côté de l'inégalité, en déplaçant le facteur 2 devant la racine carrée à l'intérieur de celle-ci, puis en multipliant par -1 de chaque côté, renversant ainsi l'inégalité. □

Afin d'obtenir une borne sur le second moment de la marge $\mu_2\big(M_Q^D\big)$, nous devons introduire des votants jumelés, tel que nous l'avons fait à la définition 3.10 du chapitre 3.

Définition 6.5. Étant donné deux votants $\mathbf{h}_i : \mathcal{X} \to \overline{Im\,\mathcal{Y}}$ et $\mathbf{h}_j : \mathcal{X} \to \overline{Im\,\mathcal{Y}}$, le *votant jumelé* $\mathbf{h}_{ij} : \mathcal{X} \to \overline{Im\,\mathcal{Y}}^2$ retourne le couple

$$\mathbf{h}_{ij}(x) \;\triangleq\; \big\langle \mathbf{h}_i(x), \mathbf{h}_j(x) \big\rangle\,.$$

Étant donné un ensemble $\mathcal{H}$ de votants et une distribution Q sur $\mathcal{H}$, nous définissons un ensemble de votants jumelés $\mathcal{H}^2$ et une distribution Q^2 par

$$\mathcal{H}^2 \;\triangleq\; \big\{\mathbf{h}_{ij} \,:\, \mathbf{h}_i, \mathbf{h}_j \in \mathcal{H}\big\}\,, \quad \text{et} \quad Q^2(\mathbf{h}_{ij}) \;\triangleq\; Q(\mathbf{h}_i) \cdot Q(\mathbf{h}_j)\,.$$

Nous avons maintenant les définitions nécessaires pour énoncer et démontrer le prochain théorème.

Théorème 6.6. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to \overline{Im\,\mathcal{Y}}$, pour toute distribution a priori P sur $\mathcal{H}$, pour tout $B > 0$ tel que $M_{\mathbf{h}}(x,y) \in [-B,B]$ pour tout $\mathbf{h} \in \mathcal{H}$ et tout $(x,y) \in \mathcal{X}\times\mathcal{Y}$, et pour tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$\mu_2\big(M_Q^D\big) \;\le\; \mu_2\big(M_Q^S\big) + B^2\sqrt{\frac{2}{m}\left[2\cdot \mathrm{KL}(Q\,\|\,P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Démonstration. Considérons l'ensemble de votants jumelés $\mathcal{H}^2$ et la distribution a posteriori Q^2 définis ci-haut. Nous considérons également la fonction de perte suivante, définie sur les votants jumelés :

$$\mathcal{L}_{\mu_2}\big(\mathbf{h}_{ij}(x), y\big) \;\triangleq\; \frac{1}{2}\big(B^2 + M_{\mathbf{h}_i}(x,y)\cdot M_{\mathbf{h}_j}(x,y)\big)\,.$$

Comme $M_{\mathbf{h}_i}(x,y)\cdot M_{\mathbf{h}_j}(x,y) \in \big[-B^2, B^2\big]$, nous avons que $\mathcal{L}_{\mu_2}\big(\mathbf{h}_{ij}(x), y\big) \in \big[0, B^2\big]$. Nous remarquons également que $\mathbf{E}_{\mathbf{h}_{ij}\sim Q^2}\,\mathbb{E}_{D'}^{\mathcal{L}_{\mu_2}} = \frac{1}{2}\left(B^2 - \mu_2\big(M_Q^{D'}\big)\right)$.

En utilisant cette fonction de perte, la borne inférieure du corollaire A.14, appliqué avec $\mathcal{H} = \mathcal{H}$ et $B = B^2$, nous indique que pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to \overline{Im\,\mathcal{Y}}$, pour toute distribution a priori P sur $\mathcal{H}$ et pour tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,

$$\frac{1}{2}\left(B^2 - \mu_2\big(M_Q^D\big)\right) \;\ge\; \frac{1}{2}\left(B^2 - \mu_2\big(M_Q^S\big)\right) - B^2\sqrt{\frac{1}{2m}\left[2\cdot \mathrm{KL}(Q\,\|\,P) + ln\,\frac{2\sqrt{m}}{\delta}\right]},$$

où le facteur 2 devant le terme KL apparaît car $\mathrm{KL}\left(Q^2\,\|\,P^2\right) = 2\cdot\mathrm{KL}(Q\,\|\,P)$. Le résultat est obtenu en multipliant par deux et en soustrayant B^2 de chaque côté de l'inégalité, en déplaçant le facteur 2 devant la racine carrée à l'intérieur de celle-ci, en multipliant par -1 de chaque côté, renversant ainsi l'inégalité. □

Avec ces deux bornes PAC-bayésiennes en main, il ne reste plus qu'à énoncer une PAC-borne sur le risque du classificateur par vote de majorité (de votants à sortie arbitraire), ci-bas.

PAC-borne 6.7. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de votants $\mathcal{X} \to \overline{Im\,\mathcal{Y}}$, pour toute distribution a priori P sur $\mathcal{H}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$R_D\left(B_Q\right) \;\le\; 1 - \frac{\left(\underline{\mu_1}\right)^2}{\overline{\mu_2}}\,,$$

où

$$\underline{\mu_1} \triangleq max\left(0,\, \mu_1\big(M_Q^S\big) - B\sqrt{\frac{2}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right),$$
$$\overline{\mu_2} \triangleq min\left(1,\, \mu_2\big(M_Q^S\big) + B^2\sqrt{\frac{2}{m}\left[2{\cdot}\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta/2}\right]}\right),$$

et où $B \in (0, 2]$ *borne supérieurement la valeur absolue de la marge* $\left|M_Q(x, y)\right|$ *pour tout* (x, y).

Démonstration. Le résultat est une application directe des théorèmes 6.4 et 6.6, et de la définition de la $\mathcal{C}$-borne généralisée du théorème 6.2. □

La PAC-borne 6.7 montre que l'utilisation de la marge généralisée introduit un facteur multiplicatif B devant la racine dans l'expression de la borne liée au premier moment, et un facteur multiplicatif B^2 dans la borne reliée au second moment. Ce facteur B borne supérieurement la marge, et sera dans $(0, 2]$ dans le cas où les vecteurs dans $\mathbf{Y}(\mathcal{Y})$ sont de norme unitaire, tel que supposé dans ce chapitre. Ce théorème PAC-bayésien n'est par contre pas limité à la marge généralisée, il peut également être utilisé pour borner des *substituts* de celles-ci, tel qu'illustré ci-bas.

6.1.4 Un substitut pour la marge

La notion générale de marge peut être complexe à utiliser en pratique, étant donné qu'elle dépend d'un terme « *max* ». Nous définissons ci-bas un substitut pour cette marge, en remplaçant le second terme de l'équation 6.2 par un paramètre de seuil ω.

Définition 6.8 (La ω-marge)**.** Pour tout exemple $(x, y) \in \mathcal{X}\times\mathcal{Y}$ et toute distribution Q sur $\mathcal{H}$, nous définissons la *ω-marge* $M_{Q,\omega}(x, y)$ du vote de majorité Q-pondéré par

$$M_{Q,\omega}(x, y) \triangleq \left\langle \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} \mathbf{h}(x)\,,\, \mathbf{Y}(y) \right\rangle - \omega\,. \tag{6.4}$$

Les deux premiers moments de la marge selon une distribution D' sont dénotés $\mu_1\big(M_{Q,\omega}^{D'}\big)$ et $\mu_2\big(M_{Q,\omega}^{D'}\big)$, respectivement.

Trivialement, la ω-marge borne supérieurement la marge généralisée lorsque $\omega = -1$. De plus, comme pour tout $\mathbf{Y}(c) \in Im\ \mathcal{Y}$ nous avons $\|\mathbf{Y}(c)\| = 1$ et pour tout $x \in \mathcal{X}$, $\mathbf{E}_{\mathbf{h}\sim Q}\,\mathbf{h}(x) \in \overline{Im\ \mathcal{Y}}$, alors la ω-marge donne toujours une borne inférieure de la marge généralisée lorsque $\omega = 1$. Nous verrons que dans le cadre de la classification multi-classe, c'est également le cas pour $\omega = \frac{1}{2}$, et dans le cas de la classification multi-étiquette pour $\omega = \frac{k-1}{k}$, où k est le nombre de classes. Lorsque la ω-marge est une borne inférieure pour la marge, nous pouvons l'utiliser pour définir une $\mathcal{C}$-borne de la manière suivante.

Corollaire 6.9 (La ω-$\mathcal{C}$-borne). *Pour toute distribution Q sur un ensemble $\mathscr{H}$ de votants $\mathbf{h} : \mathcal{X} \to \overline{Im\ \mathcal{Y}}$, toute distribution D' sur $\mathcal{X}\times\mathcal{Y}$ et tout ω tel que $M_{Q,\omega}(x,y) \leq M_Q(x,y)$ pour tout $(x,y) \sim D'$, si $\mu_1\big(M^{D'}_{Q,\omega}\big) > 0$, alors*

$$R_{D'}(B_Q) \ \leq \ C^{D'}_{Q,\omega} \ \triangleq \ 1 - \frac{\big(\mu_1\big(M^{D'}_{Q,\omega}\big)\big)^2}{\mu_2\big(M^{D'}_{Q,\omega}\big)}\,.$$

Démonstration. Le résultat est une application de la $\mathcal{C}$-borne générale du théorème 6.2, en remplaçant la marge généralisée de la définition 6.1 par sa borne inférieure donnée par la ω-marge de la définition 6.8. □

Notons que même pour les valeurs de ω pour qui $C^{D'}_{Q,\omega}$ ne donne pas une borne supérieure de $R_{D'}(B_Q)$, la valeur de $C^{D'}_{Q,\omega}$ reste tout de même intéressante, puisqu'elle conserve le même comportement que $R_{D'}(B_Q)$ simultanément pour plusieurs valeurs de Q. Nous montrerons expérimentalement ce comportement à la fin de la section 6.2.

Dans les prochaines sections, nous réutilisons ces résultats théoriques en les spécialisant à la classification multi-classe et à la prédiction multi-étiquette.

6.2 Spécialisation à la classification multi-classe

En apprentissage multi-classe, nous considérons les problèmes de classification où l'espace d'entrée $\mathcal{X}$ est un ensemble quelconque, et où l'espace de sortie est un ensemble discret $\mathcal{Y} = \{1, \ldots, k\}$, où k est le nombre de classes. Pour retrouver la classification multi-classe à partir de notre cadre général de la section 6.1, nous définissons une fonction $\mathbf{Y}(\cdot)$ telle que l'image de $\mathcal{Y}$ est $Im\ \mathcal{Y} = \{0,1\}^k$. Plus précisément, l'image d'une étiquette c est le vecteur canonique à k dimensions $(0, \ldots, 1, \ldots, 0)^\top$, où la seule entrée non nulle est un 1 à la c-ième position. L'ensemble $\mathscr{H}$ est un ensemble de votants multi-classe $\mathbf{h} : \mathcal{X} \to \overline{Im\ \mathcal{Y}}$, où l'enveloppe convexe $\overline{Im\ \mathcal{Y}}$ correspond ici à l'ensemble de tous les vecteurs à k dimensions dont la somme des composantes est 1. En classification multi-classe, comme le seul élément du vecteur $\mathbf{Y}(c)$ étant différent de zéro (et égal à 1) et le c-ième, les définitions du classificateur par vote de majorité, de la marge et de la ω-marge peuvent être respectivement réécrits comme

$$\begin{aligned}
B_Q(x) \ &= \ \operatorname*{argmax}_{c\in\mathcal{Y}} \ \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_c(x)\,,\\
M_Q(x,y) \ &= \ \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x) - \max_{c\in\mathcal{Y},\, c\neq y} \ \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_c(x)\,,\\
M_{Q,\omega}(x,y) \ &= \ \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x) - \omega\,,
\end{aligned}$$

où $h_c(x)$ est la c-ième coordonnée de $\mathbf{h}(x)$. Avec cette définition de la marge multi-classe, nous retrouvons celle proposée par Breiman (2001) en introduisant les *forêts aléatoires* («*random forests*»).

Les définitions en classification binaire sont généralement simplifiées par le fait que les deux classes sont « encodées » par -1 et $+1$. Par exemple, le classificateur par vote de majorité Q-pondéré B_Q défini à l'équation (2.1) considère le *signe* de l'espérance Q-pondérée des sorties des votants. Cette définition n'est possible que si les votants retournent une valeur entre -1 et 1, et prendre le signe du résultat correspond à *choisir la classe majoritaire*.

Le prochain théorème montre une relation entre le risque de $B_Q(\cdot)$ et la ω-marge du vote de majorité multi-classe.

Théorème 6.10. *Soit $k \geq 2$ le nombre de classes. Pour toute distribution D' sur $\mathcal{X}\times\mathcal{Y}$ et pour toute distribution Q sur un ensemble $\mathcal{H}$ de votants multi-classe, nous avons*

$$\Pr_{(x,y)\sim D'}\left(M_{Q,\frac{1}{k}}(x,y) \leq 0\right) \;\leq\; R_{D'}(B_Q) \;\leq\; \Pr_{(x,y)\sim D'}\left(M_{Q,\frac{1}{2}}(x,y) \leq 0\right).$$

Démonstration. Démontrons d'abord l'inégalité de gauche. Nous avons

$$\begin{aligned}
R_{D'}(B_Q) &= \Pr_{(x,y)\sim D'}\left(M_Q(x,y) \leq 0\right)\\
&= \Pr_{(x,y)\sim D'}\left(\mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x) \;\leq\; \max_{c\in\mathcal{Y},c\neq y} \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_c(x)\right)\\
&\geq \Pr_{(x,y)\sim D'}\left(\mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x) \;\leq\; \mathop{\mathbf{E}}_{c\in\mathcal{Y},c\neq y} \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_c(x)\right)\\
&= \Pr_{(x,y)\sim D'}\left(\mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x) \;\leq\; \frac{1}{k-1}\sum_{c=1,c\neq y}^{k} \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_c(x)\right)\\
&= \Pr_{(x,y)\sim D'}\left(\mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x) \;\leq\; \frac{1}{k-1}\left[1 - \mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x)\right]\right)\\
&= \Pr_{(x,y)\sim D'}\left(\frac{k}{k-1}\mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x) \;\leq\; \frac{1}{k-1}\right)\\
&= \Pr_{(x,y)\sim D'}\left(\mathop{\mathbf{E}}_{\mathbf{h}\sim Q} h_y(x) - \frac{1}{k} \;\leq\; 0\right)\\
&= \Pr_{(x,y)\sim D'}\left(M_{Q,\frac{1}{k}}(x,y) \;\leq\; 0\right).
\end{aligned}$$

L'inégalité de droite est vérifiée en observant simplement que nécessairement, si le poids $\mathbf{E}_{\mathbf{h}\sim Q} h_y(x)$ accordé à la bonne classe est plus grand que $\frac{1}{2}$, alors $B_Q(x,y)$ classifie correctement cet exemple. □

Conséquemment, la marge des points entre les frontières reliées à la ω-marge entre $\omega = \frac{1}{2}$ et $\omega = \frac{1}{k}$ peut être positive ou négative, en fonction de la valeur de ω. La figure 6.1 montre un exemple visuel de la marge multi-classe et la ω-marge, avec $\omega = \frac{1}{k}$ et $\omega = \frac{1}{2}$.

Le théorème 6.10 nous permet d'obtenir la ω-C-borne suivante.

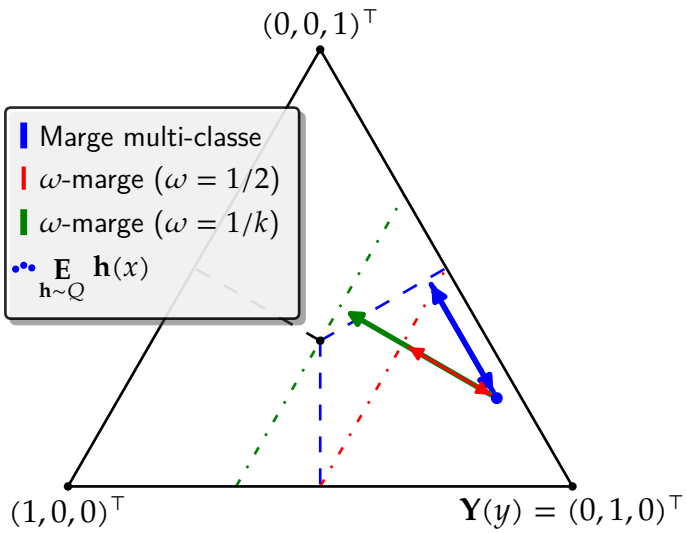

FIG. 6.1: Représentation de la marge multi-classe sur un exemple $(x, y) \in \mathcal{X}\times\mathcal{Y}$, dans le système de coordonnées barycentrique défini par $\overline{Im\, \mathcal{Y}}$ lorsque $\mathcal{Y} = \{1,2,3\}$ et la classe réelle y est 2. Nous avons $\mathbf{Y}(1) = (1,0,0)^\top$, $\mathbf{Y}(2) = (0,1,0)^\top$, et $\mathbf{Y}(3) = (0,0,1)^\top$. Chaque ligne pointillée représente la frontière à partir de laquelle l'une des marges devient négative. Un vote de majorité classifie correctement un exemple si la sortie du vote se retrouve du même côté de l'hyperplan relié à la marge multi-classe que le point $\mathbf{Y}(y)$.

Corollaire 6.11 (La ω-C-borne multi-classe)**.** *Pour toute distribution Q sur un ensemble de votants multi-classe $\mathcal{H}$, et pour toute distribution D' sur $\mathcal{X}\times\mathcal{Y}$, si $\mu_1\left(M^{D'}_{Q,\frac{1}{2}}\right) > 0$, alors*

$$R_{D'}(B_Q) \;\leq\; C^{D'}_{Q,\frac{1}{2}} \;=\; 1 - \frac{\left(\mu_1\left(M^{D'}_{Q,\frac{1}{2}}\right)\right)^2}{\mu_2\left(M^{D'}_{Q,\frac{1}{2}}\right)}\,.$$

Démonstration. Le résultat est une conséquence du corollaire 6.9 et du théorème 6.10. □

Le fait que la marge multi-classe se trouve dans la région entre les ω-marges, avec $\omega = \frac{1}{2}$ et $\omega = \frac{1}{k}$ laisse croire qu'il y a possiblement une certaine valeur ω fournissant un bon estimateur de la vraie marge. Si c'est le cas, nous pouvons considérer l'utilisation de $C^{D'}_{Q,\omega}$ pour cette valeur ω comme substitut au risque du classificateur par vote de majorité, malgré que $C^{D'}_{Q,\omega}$ ne soit pas une borne supérieure de $R_{D'}(B_Q)$ pour les valeurs $\omega < \frac{1}{2}$. Comme la ω-marge est linéaire, contrairement à la marge multi-classe, cette propriété ouvrirait la voix au développement d'une généralisation multi-classe des algorithmes MinCq et CqBoost présentés aux chapitres 4 et 5.

Évaluons maintenant le pouvoir prédictif de la ω-C-borne dans le contexte de la classification multi-classe. Nous avons vu dans le chapitre 2 que la C-borne binaire est un excellent substitut

pour le risque du classificateur par vote de majorité de votants binaires. En effet, les valeurs de $C_Q^{D'}$ et $R_{D'}(B_Q)$, tel que le montre la figure 2.3 de la section 2.4. Nous étendons ici cette analyse dans le cadre de la classification multi-classe, lorsque le nombre de classes est petit.

Dans cette expérimentation, nous générons des votes de majorité pondérés en exécutant une implémentation multi-classe de l'algorithme AdaBoost (Freund et Schapire, 1997)—nommée *AdaBoost-SAMME*[1] (Zhu et al., 2009)—sur des ensembles de données multi-classe provenant du dépôt de données UCI (Lichman, 2013). Nous séparons chaque ensemble de données en deux moitiés : un ensemble d'entraînement S, et un ensemble de test T. Nous entraînons l'algorithme sur S, en utilisant comme votants des arbres de décision. Pour chaque ensemble de données, nous considérons 100, 250, 500 et 1000 arbres de décision, de profondeur 2, 3, 4 et 5, obtenant ainsi un total de 16 votes de majorité par ensemble de données. Nous exécutons cette expérimentation pour 10 séparations entraînement/test. Pour trois de ces séparations, la figure 6.2 montre les valeurs des différentes quantités reliées au risque du classificateur par vote de majorité, et l'effet de la valeur de ω pour la ω-C-borne. Nous montrons finalement dans la table 6.1 la corrélation de Pearson (Pearson, 1895) entre chacune de ces quantités et le risque du classificateur par vote de majorité, moyennée pour les 10 séparations entraînement/test.

Quantité	Corrélation de Pearson
Borne classique $1 - \mu_1\left(M_Q^T\right)$	0.6709
C-borne multi-classe C_Q^T	0.8757
ω-C-borne $C_{Q,\omega}^{Ts}$ avec $\omega = 1/2$	0.5535
ω-C-borne $C_{Q,\omega}^{Ts}$ avec $\omega = 1/3 + 1/(3k)$	0.8811
ω-C-borne $C_{Q,\omega}^{Ts}$ avec $\omega = 1/6 + 2/(3k)$	**0.8950**
ω-C-borne $C_{Q,\omega}^{Ts}$ avec $\omega = 1/k$	0.8627

Tab. 6.1: Corrélations de Pearson des différentes valeurs avec le risque du classificateur par vote de majorité. Toutes les valeurs ont été calculées sur l'ensemble de test T.

Nous remarquons dans la figure 6.2 et la table 6.1 que pour certaines valeurs de ω, la ω-C-borne est hautement corrélée avec le risque du classificateur par vote de majorité. Malheureusement, la seule valeur ω qui fournisse une borne supérieure est $\omega = \frac{1}{2}$, et cette valeur ne montre pas le même pouvoir de prédiction. Cependant, nos résultats empiriques montrent qu'un choix judicieux de valeur ω peut améliorer la corrélation entre la ω-C-borne et le risque.

Ces expérimentations confirment l'utilité des C-bornes multi-classes basées sur la notion de marge comme substitut pour le risque du classificateur par vote de majorité, comme la C-borne binaire l'était en classification binaire. Ceci ouvre la porte au développement d'un algorithme similaire à MinCq ou CqBoost, qui cette fois-ci prend en considération la ω-C-borne. Un tel algorithme ne demanderait pas l'ajout d'un hyperparamètre supplémentaire, comme

1. Nous utilisons l'implémentation fournie dans la librairie *Scikit-Learn* (Pedregosa et al., 2011).

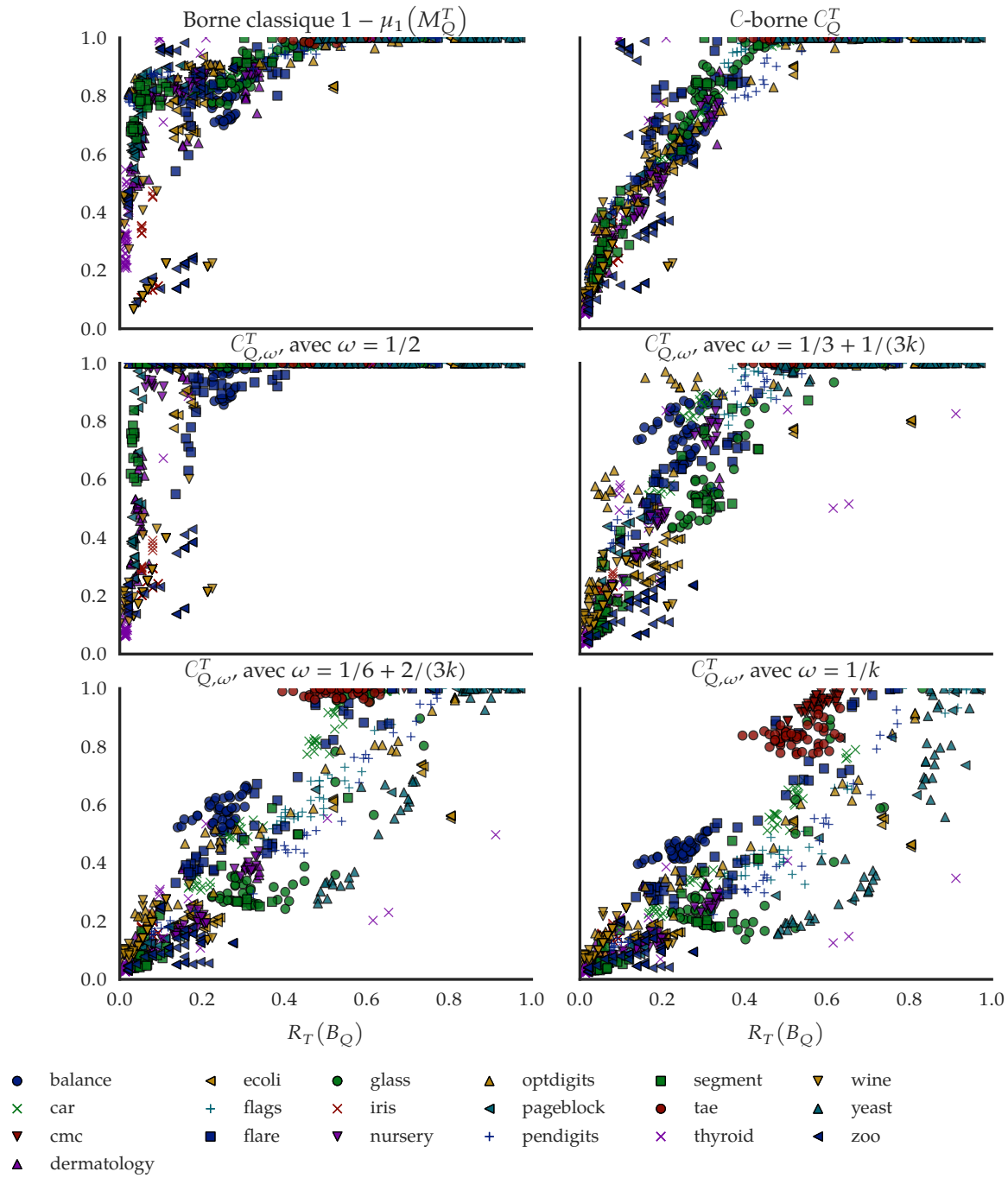

FIG. 6.2: Comparaison du risque du classificateur par vote de majorité avec la borne supérieure classique de deux fois le risque de Gibbs (ou $1 - \mu_1\left(M_Q^T\right)$), la C-borne multi-classe, et la ω-C-borne pour plusieurs valeurs de ω entre $\frac{1}{2}$ et $\frac{1}{k}$. Toutes les valeurs sont calculées sur l'ensemble de test T.

la valeur de ω est additionnée à la valeur de μ dans la contrainte sur le premier moment de la marge, et est constante dans la fonction objectif. Malheureusement, nos efforts à ce niveau ont été vains, comme nous n'avons pas été en mesure de battre les résultats obtenus par la méthode qui consiste simplement à diviser les problèmes multi-classe en plusieurs problèmes de classification binaire (Allwein, Schapire et Singer, 2001).

Dans la prochaine section, nous explorons la spécialisation du cadre général de la section 6.1 au cadre de la classification multi-étiquette.

6.3 Spécialisation à la classification multi-étiquette

Dans cette section, nous explorons la spécialisation du cadre général de la section 6.1 au cadre de la classification multi-étiquette. En classification multi-étiquette, l'espace des k classes possibles est $\{1, \dots, k\}$, mais nous considérons cette fois-ci un espace de sortie *multi-étiquette* $\mathcal{Y} = \{0,1\}^k$, qui contient des vecteurs $\mathbf{y} = (y_1, \dots, y_k)^\top$. Étant donné un exemple $(x, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$, le vecteur de sortie $\mathbf{y}$ est défini comme suit :

$$\forall j \in \{1, \dots, k\}, \quad y_j = \begin{cases} 1 \text{ si } x \text{ est étiqueté avec } j \\ 0 \text{ autrement.} \end{cases}$$

Nous définissons maintenant la fonction $\mathbf{Y}(\cdot)$ de telle sorte que $\|\mathbf{Y}(\mathbf{c})\| = 1$, c'est-à-dire que l'image de $\mathcal{Y}$ est $Im\, \mathcal{Y} = \left\{-\frac{1}{k}, \frac{1}{k}\right\}^k$, et

$$\forall j \in \{1, \dots, k\}, \quad Y_j(\mathbf{y}) = \begin{cases} \frac{1}{\sqrt{k}} \text{ si } y_j = 1 \ (x \text{ est étiqueté avec } j) \\ \frac{-1}{\sqrt{k}} \text{ autrement,} \end{cases}$$

où $Y_j(\mathbf{y})$ est la j-ième coordonnée de $\mathbf{Y}(\mathbf{y})$. Étant donné cette définition, nous avons bel et bien que pour tout $\mathbf{c} \in \mathcal{Y}$, $\|\mathbf{Y}(\mathbf{c})\| = 1$. L'ensemble $\mathcal{H}$ est construit avec des *votants multi-étiquette* $\mathbf{h} : \mathcal{X} \to \overline{Im\, \mathcal{Y}}$. Étant donné les définitions qui précèdent, nous pouvons maintenant réécrire le vote de majorité de votants multi-étiquette, la marge multi-étiquette et la ω-marge multi-étiquette comme suit :

$$\begin{aligned} B_Q(x) &= \operatorname*{argmax}_{\mathbf{c} \in \mathcal{Y}} \left[\sum_{j=1}^{k} \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} h_j(x) Y_j(\mathbf{c}) \right], \\ M_Q(x, \mathbf{y}) &= \sum_{j=1}^{k} \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} h_j(x) Y_j(\mathbf{y}) - \max_{\mathbf{c} \in \mathcal{Y}, \mathbf{c} \neq \mathbf{y}} \left[\sum_{j=1}^{k} \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} h_j(x) Y_j(\mathbf{c}) \right], \\ M_{Q,\omega}(x, \mathbf{y}) &= \sum_{j=1}^{k} \mathop{\mathbf{E}}_{\mathbf{h} \sim Q} h_j(x) Y_j(\mathbf{y}) - \omega, \end{aligned}$$

où $h_j(x)$ est la j-ième coordonnée de $\mathbf{h}(x)$.

Le prochain théorème met en relation le risque de $B_Q(\cdot)$ et la ω-marge du vote de majorité multi-étiquette.

Théorème 6.12. *Soit $k \geq 2$ le nombre de classes possibles. Pour toute distribution D' sur $\mathcal{X} \times \mathcal{Y}$ et toute distribution Q sur un ensemble de votants multi-étiquette $\mathcal{H}$, nous avons*

$$R_{D'}(B_Q) \;\leq\; \Pr_{(x,\mathbf{y})\sim D'}\left(M_{Q,\frac{k-1}{k}}(x,\mathbf{y}) \leq 0\right).$$

Démonstration. Nous devons montrer que

$$\Pr_{(x,\mathbf{y})\sim D'}\left(M_Q(x,\mathbf{y}) \leq 0\right) \leq \Pr_{(x,\mathbf{y})\sim D'}\left(M_{Q,\frac{k-1}{k}}(x,\mathbf{y}) \leq 0\right).$$

Pour y arriver, nous allons montrer que si la ω-marge est positive, alors nécessairement la marge multi-étiquette est également positive. En d'autres termes, pour tout exemple $(x, \mathbf{y})$,

$$M_{Q,\frac{k-1}{k}}(x,\mathbf{y}) > 0 \implies M_Q(x,\mathbf{y}) > 0\,.$$

L'enveloppe convexe $\overline{Im\,\mathcal{Y}}$ est un hypercube dont les sommets sont les vecteurs $\mathbf{Y}(\mathbf{c})$, pour $\mathbf{c} \in \mathcal{Y}$. Étant donné un sommet $\mathbf{Y}(\mathbf{y})$, nous dénotons $H_\mathbf{y}$ l'hyperplan passant par tous les points $\mathbf{Y}^{(j)}(\mathbf{y})$, où $\mathbf{Y}^{(j)}(\mathbf{y})$ est le point dans l'hypercube qui a exactement les mêmes coordonnées que $\mathbf{Y}(\mathbf{y})$, sauf la j-ième composante, qui elle est 0.

Considérons maintenant la région $R_\mathbf{y}$ de l'hypercube $\overline{Im\,\mathcal{Y}}$ contenant tous les points qui sont du même côté que $\mathbf{Y}(\mathbf{y})$ par rapport à l'hyperplan $H_\mathbf{y}$. Supposons pour l'instant que ces points sont exactement ceux tels que $M_{Q,\frac{k-1}{k}}(x,\mathbf{y}) > 0$. Alors clairement, pour tout $k \geq 2$, le point $\mathbf{Y}(\mathbf{y})$ est strictement plus près du point $\mathbf{E}_{\mathbf{h}\sim Q}\,\mathbf{h}(x)$ que tout autre $\mathbf{Y}(\mathbf{c})$ si le vecteur $\mathbf{E}_{\mathbf{h}\sim Q}\,\mathbf{h}(x)$ se trouve dans la région $R_\mathbf{y}$. Ceci implique que la marge $M_Q(x,\mathbf{y})$ est strictement positive, tel que voulu. La figure 6.3 montre un exemple avec $k = 2$ et $\mathbf{Y}(\mathbf{y}) = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)^\top$, où $H_\mathbf{y}$ est représenté par une ligne pointillée rouge, et $R_\mathbf{y}$ est la région délimitée par le coin supérieur droit et l'hyperplan $H_\mathbf{y}$.

Pour terminer la preuve, nous devons donc démontrer que la région $R_\mathbf{y}$ est bel et bien la région contenant les points tels que $M_{Q,\frac{k-1}{k}}(x,\mathbf{y}) > 0$. De manière équivalente, nous devons montrer que l'intersection de $H_\mathbf{y}$ et de l'hypercube $\overline{Im\,\mathcal{Y}}$ est exactement l'ensemble des points tels que $M_{Q,\frac{k-1}{k}}(x,\mathbf{y}) = 0$, c'est-à-dire, les vecteurs $\mathbf{E}_{\mathbf{h}\sim Q}\,\mathbf{h}(x)$ tels que $\left\langle \mathbf{E}_{\mathbf{h}\sim Q}\,\mathbf{h}(x), \mathbf{Y}(\mathbf{y})\right\rangle - \frac{k-1}{k} = 0$. L'algèbre linéaire de base nous indique que les points P qui sont sur l'hyperplan $H_\mathbf{y}$ doivent satisfaire l'équation $\langle (P - P_0), N\rangle = 0$, où N est la normale de l'hyperplan, et P_0 est n'importe quel point de P. Il est trivial de voir que $\mathbf{Y}(\mathbf{y})$ est la normale de $H_\mathbf{y}$, et nous pouvons prendre le point $P_0 = \mathbf{Y}^{(1)}(\mathbf{y})$. Ainsi, l'équation à satisfaire devient $\left\langle \left(P - \mathbf{Y}^{(1)}(\mathbf{y})\right), \mathbf{Y}(\mathbf{y})\right\rangle = 0$.

Comme toutes les coordonnées de $\mathbf{Y}(\mathbf{y})$ sont soit $\frac{1}{\sqrt{k}}$ ou $-\frac{1}{\sqrt{k}}$, et comme toutes les coordonnées de $\mathbf{Y}^{(1)}(\mathbf{y})$ sont les mêmes que celles de $\mathbf{Y}(\mathbf{y})$ sauf la première qui a une valeur de 0, nous avons alors que $\left\langle \mathbf{Y}^{(1)}(\mathbf{y}), \mathbf{Y}(\mathbf{y})\right\rangle = \frac{k-1}{k}$. Le résultat s'ensuit de

$$\left\langle \left(P - \mathbf{Y}^{(1)}(\mathbf{y})\right), \mathbf{Y}(\mathbf{y})\right\rangle \;=\; \langle P, \mathbf{Y}(\mathbf{y})\rangle - \left\langle \mathbf{Y}^{(1)}(\mathbf{y}), \mathbf{Y}(\mathbf{y})\right\rangle \;=\; \langle P\,,\, \mathbf{Y}(\mathbf{y})\rangle - \frac{k-1}{k}\,. \qquad \square$$

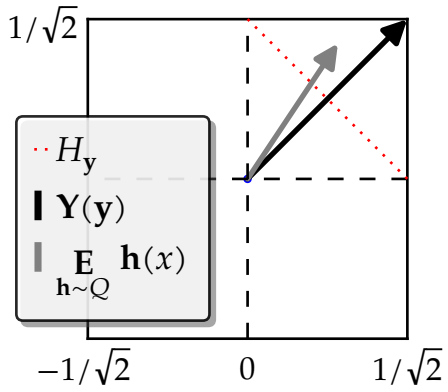

FIG. 6.3: Représentation graphique de la ω-marge multi-étiquette, avec $\omega = \frac{k-1}{k}$, et du vote de majorité appliqué à l'exemple $(x, \mathbf{y})$ lorsque $k = 2$ et $\mathbf{y} = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)^\top$. Les sommets de l'hypercube correspondent aux différentes possibilités de multi-étiquettes, c'est-à-dire $\mathbf{Y}(\mathcal{Y}) = \left\{\left(\frac{-1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right)^\top, \left(\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right)^\top, \left(\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)^\top, \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)^\top\right\}$. Chaque ligne noire pointillée correspond à la frontière de décision de la marge multi-étiquette, c'est-à-dire qu'un vote de majorité classifie correctement un exemple si le vecteur résultant du vote est du même côté des hyperplans que le sommet correspondant à la bonne étiquette. La ligne pointillée rouge correspond à la frontière de décision de la ω-marge multi-étiquette avec $\omega = \frac{k-1}{k}$, c'est-à-dire que celle-ci est positive si le vecteur résultant du vote de majorité se retrouve du même côté que le sommet correspondant à la bonne étiquette.

Le théorème 6.12 nous permet d'obtenir la ω-C-borne suivante.

Corollaire 6.13 (La ω-C-borne multi-étiquette)**.** *Pour toute distribution Q sur un ensemble de votants multi-étiquette $\mathcal{H}$, et pour toute distribution D' sur $\mathcal{X}\times\mathcal{Y}$, si $\mu_1\left(M^{D'}_{Q,\frac{k-1}{k}}\right) > 0$, alors*

$$R_{D'}(B_Q) \;\leq\; C^{D'}_{Q,\frac{k-1}{k}} \;=\; 1 - \frac{\left(\mu_1\left(M^{D'}_{Q,\frac{k-1}{k}}\right)\right)^2}{\mu_2\left(M^{D'}_{Q,\frac{k-1}{k}}\right)}\,.$$

Démonstration. Le résultat est une conséquence du corollaire 6.9 et du théorème 6.12. □

Cette spécialisation des notions de marge et de la ω-C-borne au cadre de la classification multi-étiquette peut sembler moins pertinente que la spécialisation au cas multi-classe, car pour que la marge soit positive, le vote de majorité ne doit faire aucune erreur sur aucune des classes. Par exemple, si nous avons un problème à 100 classes (et ainsi les vecteurs $\mathbf{y}$ ont 100 composantes), une erreur sur une seule des étiquettes provoquera une marge négative. Ceci dit, les idées présentées dans cette section restent intéressantes à investiguer, car malgré que la marge puisse être souvent négative, étudier ses deux premiers moments statistiques

via la C-borne ou la ω-C-borne pourrait tout de même s'avérer intéressant. Nous n'avons par contre malheureusement aucun résultat expérimental dans le cadre multi-étiquette pour le démontrer, que nous laissons comme travaux futurs.

6.4 Conclusion du chapitre

Dans ce chapitre, nous avons étendu le concept de votes de majorité en reprenant les définitions de base et en les énonçant de manière à ce que les votants puissent prédire un élément de sortie arbitrairement complexe. Nous avons également étendu le concept de marge, à partir de laquelle nous obtenons une C-borne ainsi qu'une borne de généralisation PAC-bayésienne basée sur celle-ci. Cette C-borne est par contre complexe à manipuler puisqu'elle dépend de termes « max ». Pour surmonter cette difficulté, nous avons proposé une relaxation de la marge généralisée, nommée la ω-marge, qui remplace le terme dépendant d'un « max » par un paramètre, puis avons proposé une C-borne associée, nommée la ω-C-borne.

Nous avons spécialisé ce cadre général à deux cas : la classification multi-classe, pour laquelle nous avons montré que la ω-C-borne est très corrélée avec le risque du classificateur par vote de majorité, et la classification multi-étiquette.

Comme travaux futurs, il serait intéressant d'explorer l'utilisation de ces résultats théoriques pour le développement d'un algorithme d'apprentissage multi-classe ou multi-étiquette similaire à MinCq ou CqBoost. À ce jour, nos tentatives ont été infructueuses, mais il n'est pas exclus que nous n'ayons pas suffisamment exploré de possibilités, par exemple l'utilisation de votants plus forts que des arbres de décision de faible profondeur. Nous croyons que le développement d'un tel algorithme serait pertinent dans les situations où le nombre de classes est peu élevé.

Chapitre 7

Théorie PAC-bayésienne personnalisable et apprentissage transductif

Dans ce chapitre, nous proposons un processus simplifié de preuve pour les bornes de généralisation PAC-bayésiennes, nous permettant de diviser la démonstration en une succession de quatre inégalités. Cette nouvelle preuve simplifie la « personnalisation » des théorèmes PAC-bayésiens. Nous présentons deux personnalisations. Une première est une nouvelle famille de bornes basées sur la *divergence de Rényi* entre les distributions a priori et à posteriori, alors que la divergence normalement utilisée dans cette théorie est celle de Kullback-Leibler. Nous évaluons empiriquement les valeurs de ces bornes en calculant la perte associée à chaque inégalité de la preuve simplifiée en les comparant aux bornes classiques basées sur la divergence Kullback-Leibler. Nous présentons également une nouvelle famille de bornes PAC-bayésiennes adaptées à l'*apprentissage transductif*. Nous proposons finalement une extension de l'algorithme MinCq (et conséquemment CqBoost) pour l'apprentissage transductif.

Les notions théoriques présentées dans ce chapitre sont une unification de deux articles où nous partageons le crédit avec nos collaborateurs. Dans Bégin, Germain, Laviolette et Roy (2014), nous proposons l'extension de la théorie PAC-bayésienne au cadre de l'apprentissage transductif. Dans Bégin, Germain, Laviolette et Roy (2016), nous proposons le processus de preuve simplifié[1] permettant d'obtenir de nouvelles familles de bornes PAC-bayésiennes, dont celles basées sur la divergence de Rényi plutôt que sur la divergence Kullback-Leibler. Les nouvelles démonstrations développées dans ce dernier article nous permettant de simplifier le développement des bornes PAC-bayésiennes transductives, nous avons décidé dans ce chapitre de renverser l'ordre chronologique et d'entremêler ces deux articles.

1. Cette nouvelle approche de démonstration a été introduite pour la première fois dans la thèse de notre collaborateur Pascal Germain (Germain, 2015). Les résultats qui en découlent sont un travail de collaboration avec ce dernier et nos autres coauteurs.

Certains éléments de ce chapitre peuvent être vus comme des simplifications d'autres chapitres de cette thèse. Notamment, le processus de preuve simplifié aurait pu être utilisé au chapitre 3. Cependant, l'ordre dans lequel nous présentons les éléments de cette thèse permet de conserver la chronologie globale et ainsi de mieux distinguer nos différentes contributions. Le chapitre 3 contribue à unifier les différentes bornes PAC-bayésiennes de l'état de l'art et étend les résultats en utilisant la C-borne. Le présent chapitre simplifie encore plus les démonstrations, permettant le développement de bornes de généralisation complètement différentes de celles généralement obtenues par la théorie PAC-bayésienne. Notons que dans ce chapitre, nous nous limitons aux bornes PAC-bayésiennes basées sur le risque du classificateur de Gibbs (et non sur la C-borne), puis aux ensembles de votants $\mathcal{H}$ qui sont des classificateurs, et non des votants à valeur réelle. La plupart des résultats présentés peuvent par contre être étendus en utilisant les techniques présentées au chapitre 3.

Ce chapitre est organisé comme suit. La section 7.1 pose un regard neuf sur la démonstration des théorèmes PAC-bayésiens, permettant leur simplification et facilitant la création de nouveaux théorèmes. Puis, la section 7.2 présente une nouvelle famille de bornes PAC-bayésiennes basées sur la divergence de Rényi plutôt que sur la divergence Kullback-Leibler traditionnelle. La section 7.3 présente une spécialisation de la théorie PAC-bayésienne au cadre de l'apprentissage transductif. Par la suite, la section 7.4 présente une spécialisation de l'algorithme MinCq au cadre transductif. Nous concluons finalement à la section 7.5.

7.1 Un regard neuf sur les preuves PAC-bayésiennes

Les bornes PAC-bayésiennes classiques bornent indirectement le risque du classificateur par vote de majorité B_Q en bornant le risque du classificateur stochastique de Gibbs G_Q. Étant donné un ensemble de votants $\mathcal{H}$ et une distribution a priori P sur $\mathcal{H}$, les théorèmes PAC-bayésiens présentés au chapitre 3 bornent le « vrai » risque du classificateur de Gibbs, simultanément pour toutes les distributions a posteriori Q. Celles-ci utilisent deux ingrédients principaux : une fonction convexe $\Delta : [0,1]^2 \to \mathbb{R}$ qui relie le risque empirique au vrai risque, et un terme de complexité dépendant de la divergence Kullback-Leibler (KL) entre Q et P. Lorsqu'aucune supposition supplémentaire n'est faite sur la distribution Q, la plupart des bornes PAC-bayésiennes dépendent de la divergence KL (Catoni, 2007 ; Parrado-Hernández et al., 2012 ; Lever, Laviolette et Shawe-Taylor, 2013 ; Germain, Lacasse, Laviolette, Marchand et Roy, 2015).[2] Rappelons maintenant les éléments et quantités de base, sur lesquelles sont basés notre preuve simplifiée.

2. Il y a comme exceptions notables les bornes présentées au chapitre 3 ne contenant aucune divergence mais restreignant la distribution Q à être alignée sur P, et une borne PAC-bayésienne dépendant de la *divergence chi carré* développée par Honorio et Jaakkola (2014), que nous retrouvons plus loin dans ce chapitre.

7.1.1 Rappel des notions de base

Nous considérons les problèmes de classification binaire, avec un espace d'entrée $\mathcal{X}$ arbitraire et un espace de sortie $\mathcal{Y} = \{-1, 1\}$. Un exemple est une paire $(x, y) \in \mathcal{X} \times \mathcal{Y}$, où x est la description y est l'étiquette.

Nous considérons dans un premier temps le cadre de l'apprentissage *inductif*, où chaque exemple (x, y) est tiré indépendamment et identiquement distribué (*i.i.d.*) d'une distribution fixe mais inconnue D sur $\mathcal{X} \times \mathcal{Y}$. L'ensemble $S = \langle (x_1, y_1), (x_2, y_2), \dots, (x_m, y_m) \rangle \sim D^m$ est l'ensemble d'entraînement. La tâche de l'algorithme d'apprentissage est, étant donné l'ensemble d'entraînement S, d'apprendre un classificateur $h : \mathcal{X} \to \mathcal{Y}$ qui sera en mesure de classifier de nouveaux exemples tirés selon la distribution D. Comme dans le reste de cette thèse, le risque d'un classificateur h sur une distribution D' est défini par la probabilité que celui-ci fasse une erreur sur un exemple tiré selon D',

$$R_{D'}(h) \triangleq \mathop{\mathbf{E}}_{(x,y)\sim D'} I\Big[h(x) \neq y\Big],$$

où $I(a) = 1$ si le prédicat a est vrai, et 0 autrement.

Lorsque nous calculons le risque sur un ensemble discret S' au lieu d'une distribution de probabilité, nous considérons la distribution uniforme sur S', obtenant ainsi la perte zéro-un moyenne :

$$R_{S'}(h) = \frac{1}{|S'|} \sum_{(x,y)\in S'} I\Big[h(x) \neq y\Big].$$

Nous considérons un ensemble $\mathcal{H}$ de votants $h : \mathcal{X} \to \mathcal{Y}$ qui sont des classificateurs, c'est-à-dire qui retournent $+1$ ou -1 et non une valeur réelle. Nous considérons une distribution a priori P sur $\mathcal{H}$ qui encode les connaissances a priori sur le problème à résoudre, et une distribution a posteriori Q sur $\mathcal{H}$, obtenue en exécutant un algorithme d'apprentissage sur l'ensemble d'entraînement S. Tel que discuté au chapitre 3, la théorie PAC-bayésienne prend en considération le classificateur stochastique de Gibbs G_Q, qui étant donné une distribution Q sur $\mathcal{H}$, classifie un exemple x en tirant un classificateur h au hasard selon Q, puis en retournant $h(x)$. Le risque du classificateur de Gibbs est donné par

$$R_{D'}(G_Q) \triangleq \mathop{\mathbf{E}}_{(x,y)\sim D'} \mathop{\mathbf{E}}_{h\sim Q} I\Big[h(x) \neq y\Big].$$

Les bornes PAC-bayésiennes classiques donnent des garanties sur le risque de généralisation $R_D(G_Q)$, qui correspond à la probabilité que G_Q fasse une erreur sur un exemple tiré selon D.[3] Ces bornes dépendent de deux ingrédients : le risque de Gibbs empirique $R_S(G_Q)$, et la divergence Kullback-Leibler entre les distributions Q et P, dont nous rappelons la définition à la table 7.1.

3. Dans le chapitre 2, nous définissons également des bornes sur les deux premiers moments de la marge $M_Q(x, y)$, permettant d'obtenir des bornes PAC-bayésiennes plus précises que celles n'utilisant que le risque du classificateur de Gibbs. Les résultats présentés dans ce chapitre peuvent être étendus de la même manière.

Divergence	Notation	Espérance sur P	Espérance sur Q
Kullback-Leibler	$\mathrm{KL}(Q \,\|\, P)$	$\mathop{\mathbf{E}}_{f\sim P} \frac{Q(f)}{P(f)} \ln \frac{Q(f)}{P(f)}$	$\mathop{\mathbf{E}}_{f\sim Q} \ln \frac{Q(f)}{P(f)}$
Rényi	$D_\alpha(Q \,\|\, P)$	$\frac{1}{\alpha-1} \ln \left[\mathop{\mathbf{E}}_{f\sim P} \left(\frac{Q(f)}{P(f)} \right)^{\alpha} \right]$	$\frac{1}{\alpha-1} \ln \left[\mathop{\mathbf{E}}_{f\sim Q} \left(\frac{Q(f)}{P(f)} \right)^{\alpha-1} \right]$
chi carré	$\chi^2(Q \,\|\, P)$	$\mathop{\mathbf{E}}_{f\sim P} \left[\left(\frac{Q(f)}{P(f)} \right)^2 - 1 \right]$	$\mathop{\mathbf{E}}_{f\sim Q} \left[\frac{Q(f)}{P(f)} - 1 \right]$

TAB. 7.1: Différentes divergences entre les distributions Q et P, apparaissant dans les bornes PAC-bayésiennes.

Rappelons que dans cette thèse, nous considérons que le support de Q est inclus dans le support de P, c'est-à-dire que si $P(h) = 0$, alors $Q(h) = 0$ (voir la convention 3.1). Cette supposition permet aux définitions des différentes divergences définies à la table 7.1 de ne pas être infinies ou non définies pour certaines distributions Q. Des définitions plus générales permettraient de rendre les résultats de ce chapitre valides pour toutes distributions Q et P sans restriction. Nous omettons ces généralisations par soucis de simplicité.

Nous rappelons maintenant l'*inégalité du changement de mesure* basée sur la divergence Kullback-Leibler, définie au lemme 3.2. Pour tout ensemble $\mathcal{H}$, toute distribution P sur $\mathcal{H}$, toute distribution Q sur $\mathcal{H}$ dont le support est inclus dans le support de P et toute fonction $\phi : \mathcal{H} \to \mathbb{R}$, mesurable sur P, nous avons

$$\mathop{\mathbf{E}}_{h\sim Q} \phi(h) \;\leq\; \mathrm{KL}(Q \,\|\, P) + \ln \left(\mathop{\mathbf{E}}_{h\sim P} e^{\phi(h)} \right) . \tag{7.1}$$

7.1.2 Une preuve personnalisable

Le prochain théorème est similaire aux théorèmes généraux du chapitre 2, avec les exceptions suivantes :

- Deux occurences de la variable m sont remplacés par une variable $m' > 0$, qui mathématiquement peut être égale à n'importe quelle valeur, mais dont le valeur est généralement fixée à m. La valeur m' peut être modifiée pour tenter d'obtenir des bornes plus serrées ;
- La borne n'est exprimée que pour le risque du classificateur de Gibbs, permettant ainsi de simplifier davantage l'expression de la borne ;
- La borne n'est exprimée que pour les votants qui sont des classificateurs, comme l'extension au paradigme de l'apprentissage transductif plus loin dans ce chapitre n'a pas été généralisée aux votants à valeur réelle.

Les étapes de la preuve sont similaires aux idées classiques et celles présentées au chapitre 3. Cependant, l'approche est nouvelle. En particulier, la démonstration débute directement avec la quantité à borner (et non avec une variable aléatoire se retrouvant dans le terme de droite de la borne), puis est divisée en quatre inégalités successives, présentées de manière schématique à la figure 7.1. Comme nous le verrons aux sections 7.2 et 7.3, cette approche simplifiera grandement la personnalisation de la preuve, nous permettant d'obtenir des théorèmes PAC-bayésiens basés sur la divergence de Rényi, et d'autres spécialisés au cadre de l'apprentissage transductif plutôt qu'inductif. Cette démonstration permet aussi de faire ressortir les différentes approximations permettant d'obtenir la borne de généralisation, avec lesquelles nous expérimenterons empiriquement à la section 7.2.3.

Théorème 7.1. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de classificateurs $h : \mathcal{X} \to \{-1, 1\}$, pour toute distribution a priori P sur $\mathcal{H}$, pour tout $\delta \in (0, 1]$, pour tout $m' > 0$ et pour toute fonction convexe $\Delta : [0, 1]\times[0, 1] \to \mathbb{R}$, nous avons avec probabilité au moins $1 - \delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$\Delta\Big(R_S(G_Q), R_D(G_Q)\Big) \;\leq\; \frac{1}{m'}\left[\mathrm{KL}(Q \,\|\, P) + ln\, \frac{\mathcal{I}^{\mathrm{K}}_{\Delta}(m, m')}{\delta}\right],$$

où

$$\mathcal{I}^{\mathrm{K}}_{\Delta}(m, m') \;\triangleq\; \sup_{r\in[0,1]} \left[\sum_{k=0}^{m} \mathrm{Bin}^m_k(r)\; e^{m'\,\Delta\left(\frac{k}{m}, r\right)}\right], \tag{7.2}$$

et où $\mathrm{Bin}^m_k(r)$ est la fonction de masse de la distribution binomiale :

$$\mathrm{Bin}^m_k(r) \;\triangleq\; \binom{m}{k} (r)^k (1 - r)^{m-k}.$$

Démonstration. La démonstration suivante suit exactement les inégalités présentées de manière schématique à la figure 7.1. Nous démarrons avec la quantité à borner, c'est-à-dire $\Delta\Big(R_S(G_Q),\, R_D(G_Q)\Big)$. Nous appliquons d'abord l'inégalité de Jensen (lemme A.3) à la fonction convexe $\Delta(\cdot, \cdot)$, et l'inégalité du changement de mesure basée sur la divergence Kullback-Leibler de l'équation (7.1), avec $\phi(h) = m'\,\Delta\Big(R_S(h), R_D(h)\Big)$. Nous avons donc, pour toute distribution Q sur $\mathcal{H}$,

$$\begin{aligned}
m'\,\Delta\Big(R_S(G_Q), R_D(G_Q)\Big) \;&=\; m'\,\Delta\Big(\mathop{\mathbf{E}}_{h\sim Q} R_S(h), \mathop{\mathbf{E}}_{h\sim Q} R_D(h)\Big) \\
&\leq\; \mathop{\mathbf{E}}_{h\sim Q} m'\,\Delta\Big(R_S(h), R_D(h)\Big) \\
&\leq\; \mathrm{KL}(Q \,\|\, P) + ln\Big(\underbrace{\mathop{\mathbf{E}}_{h\sim P} e^{m'\,\Delta\left(R_S(h), R_D(h)\right)}}_{X_P(S)}\Big).
\end{aligned}$$

Considérons maintenant la variable aléatoire $X_P(S)$ définie ci-haut, et appliquons l'inégalité de Markov (lemme A.2), pour obtenir

$$\Pr_{S\sim D^m}\left(X_P(S) \;\leq\; \frac{1}{\delta}\,\mathop{\mathbf{E}}_{S'\sim D^m} X_P(S')\right) \;\geq\; 1-\delta\,.$$

Ceci implique qu'avec probabilité au moins $1-\delta$ sur le choix d'un ensemble $S \sim D^m$, nous avons, simultanément pour tout Q sur $\mathcal{H}$, que

$$m'\,\Delta\Big(R_S\big(G_Q\big),\,R_D\big(G_Q\big)\Big) \;\leq\; \mathrm{KL}(Q\,\|\,P) + ln\,\frac{\mathop{\mathbf{E}}_{S'\sim D^m} X_P(S')}{\delta}\,. \tag{7.3}$$

Finalement, bornons supérieurement le terme $\mathbf{E}_{S'\sim D^m} X_P(S')$, premièrement en échangeant les deux espérances, puis en utilisant le fait que le nombre d'erreurs $m\,R_{S'}(h)$ suit une distribution binomiale[4], avec paramètres m et $R_D(h)$:

$$\begin{aligned}
\mathop{\mathbf{E}}_{S'\sim D^m} X_P(S') \;&=\; \mathop{\mathbf{E}}_{S'\sim D^m}\mathop{\mathbf{E}}_{h\sim P}\; e^{m'\,\Delta\left(R_{S'}(h),\,R_D(h)\right)}\\
&=\; \mathop{\mathbf{E}}_{h\sim P}\mathop{\mathbf{E}}_{S'\sim D^m}\; e^{m'\,\Delta\left(R_{S'}(h),\,R_D(h)\right)}\\
&=\; \mathop{\mathbf{E}}_{h\sim P}\sum_{k=0}^{m}\Pr_{S'\sim D^m}\left(R_{S'}(h)=\frac{k}{m}\right)\,e^{m'\,\Delta\left(\frac{k}{m},\,R_D(h)\right)}\\
&=\; \mathop{\mathbf{E}}_{h\sim P}\sum_{k=0}^{m}\mathrm{Bin}_k^m\Big(R_D(h)\Big)\,e^{m'\,\Delta\left(\frac{k}{m},\,R_D(h)\right)}\\
&\leq\; \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\mathrm{Bin}_k^m(r)\,e^{m'\,\Delta\left(\frac{k}{m},\,r\right)}\right]\\
&=\; \mathcal{I}^{\mathrm{K}}_{\Delta}(m,m')\,.
\end{aligned}$$

Le résultat final est obtenu en remplaçant $\mathbf{E}_{S'\sim D^m} X_P(S')$ par sa borne supérieure $\mathcal{I}^{\mathrm{K}}_{\Delta}(m,m')$ dans l'équation (7.3). □

Notons que la plupart des théorèmes PAC-bayésiens de cette thèse et de la littérature utilisent $m' = m$. Dans ce cas particulier, nous utilisons le raccourci de notation $\mathcal{I}^{\mathrm{K}}_{\Delta}(m) \triangleq \mathcal{I}^{\mathrm{K}}_{\Delta}(m,m)$.

7.1.3 Choix de la Δ-fonction

Tel que discuté au chapitre 3, une borne telle que celle énoncée par le théorème 7.1 est un outil générique permettant de dériver diverses bornes PAC-bayésiennes, comme Δ peut être n'importe quelle fonction convexe. Par contre, nous devons calculer (ou borner supérieurement) la valeur de $\mathcal{I}^{\mathrm{K}}_{\Delta}(m,m')$ pour obtenir une borne calculable.

4. Notons que les preuves du chapitre 3 sont généralisées aux votants à valeur réelle à cette étape en appliquant le lemme de Maurer (lemme A.9).

		$\Delta\Big(\mathop{\mathbf{E}}_{h\sim Q} R_S(h),\ \mathop{\mathbf{E}}_{h\sim Q} R_D(h)\Big)$
Inégalité de Jensen	$\leq$	$\mathop{\mathbf{E}}_{h\sim Q} \Delta\big(R_S(h),\ R_D(h)\big)$
Changement de mesure	$\leq$	$\dfrac{1}{m'}\Big[\mathrm{KL}(Q\,\|\,P) + ln\Big(\mathop{\mathbf{E}}_{h\sim P} e^{m'\Delta\left(R_S(h),R_D(h)\right)}\Big)\Big]$
Inégalité de Markov	$\underset{1-\delta}{\leq}$	$\dfrac{1}{m'}\Big[\mathrm{KL}(Q\,\|\,P) + ln\Big(\dfrac{1}{\delta}\mathop{\mathbf{E}}_{S'\sim D^m}\mathop{\mathbf{E}}_{h\sim P} e^{m'\Delta\left(R_{S'}(h),R_D(h)\right)}\Big)\Big]$
Échange des espérances	$=$	$\dfrac{1}{m'}\Big[\mathrm{KL}(Q\,\|\,P) + ln\Big(\dfrac{1}{\delta}\mathop{\mathbf{E}}_{h\sim P}\mathop{\mathbf{E}}_{S'\sim D^m} e^{m'\Delta\left(R_{S'}(h),R_D(h)\right)}\Big)\Big]$
Loi binomiale	$=$	$\dfrac{1}{m'}\Big[\mathrm{KL}(Q\,\|\,P) + ln\Big(\dfrac{1}{\delta}\mathop{\mathbf{E}}_{h\sim P}\sum_{k=0}^{m}\mathrm{Bin}_k^m(R_D(h)) e^{m'\Delta\left(\frac{k}{m},R_D(h)\right)}\Big)\Big]$
Supremum sur le risque	$\leq$	$\dfrac{1}{m'}\Big[\mathrm{KL}(Q\,\|\,P) + ln\Big(\dfrac{1}{\delta}\sup_{r\in[0,1]}\Big\{\sum_{k=0}^{m}\mathrm{Bin}_k^m(r)\, e^{m'\Delta\left(\frac{k}{m},r\right)}\Big\}\Big)\Big]$

FIG. 7.1: Idée de la preuve revisitée de la borne PAC-bayésienne du théorème 7.1. Le symbole $\underset{1-\delta}{\leq}$ indique que l'inégalité est valide avec probabilité au moins $1-\delta$.

Un choix commun est $\Delta = \Delta_{\mathrm{KL}}$, la divergence Kullback-Leibler entre deux distribution de Bernoulli avec probabilité de succès p et q, définie dans la table 7.2. Nous pouvons réécrire $\Delta_{\mathrm{KL}}(q,p) = H(q,p) - H(q)$, en utilisant les définitions usuelles de l'*entropie* et l'*entropie croisée*,

$$H(q) \triangleq -q\,ln\,q - (1-q)\,ln(1-q)\,, \qquad H(q,p) \triangleq -q\,ln\,p - (1-q)\,ln(1-p)\,. \tag{7.4}$$

Il est ici facile de voir que le terme r disparaît dans chaque terme de la somme contenue dans $\mathcal{T}^{\mathrm{K}}_{\Delta_{\mathrm{KL}}}(m)$, donnant la simplification suivante :

$$\begin{aligned}
\mathcal{T}^{\mathrm{K}}_{\Delta_{\mathrm{KL}}}(m) &= \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\mathrm{Bin}_k^m(r)\, e^{m\,\Delta_{\mathrm{KL}}\left(\frac{k}{m},r\right)}\right] \\
&= \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\binom{m}{k}(r)^k(1-r)^{m-k}\, e^{m\left(H\left(\frac{k}{m},r\right)-H\left(\frac{k}{m}\right)\right)}\right] \\
&= \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\binom{m}{k}(r)^k(1-r)^{m-k}\, e^{m\left(-\frac{k}{m}\,ln\,r-\left(1-\frac{k}{m}\right)ln(1-r)\right)}\, e^{-m\,H\left(\frac{k}{m}\right)}\right] \\
&= \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\binom{m}{k}(r)^k(1-r)^{m-k}\,\frac{1}{(r)^k}\frac{1}{(1-r)^{(m-k)}}\, e^{-m\,H\left(\frac{k}{m}\right)}\right] \\
&= \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\binom{m}{k}\, e^{-m\,H\left(\frac{k}{m}\right)}\right] \\
&= \sum_{k=0}^{m}\binom{m}{k}\left(\frac{k}{m}\right)^k\left(1-\frac{k}{m}\right)^{(m-k)} = \sum_{k=0}^{m}\eta(k,m) = \xi(m)\,,
\end{aligned} \tag{7.5}$$

où $\eta(a,b) \triangleq \binom{b}{a}\left(\frac{a}{b}\right)^a\left(1-\frac{a}{b}\right)^{b-a}$, et où $\xi(m)$ est défini à l'équation (3.3).

Fonction	Notation	Expression
Entropie	$H(q)$	$-q\,ln\,q-(1-q)\,ln(1-q)$
Entropie croisée	$H(q,p)$	$-q\,ln\,p-(1-q)\,ln(1-p)$
Divergence Kullback-Leibler	$\Delta_{\mathrm{KL}}(q,p)$	$H(q,p)-H(q)$
Divergence transductive	$\Delta^{\star}_{\beta}(q,p)$	$\frac{1}{\beta}\left(H(\beta)-pH\left(\beta\frac{q}{p}\right)-(1-p)H\left(\beta\frac{1-q}{1-p}\right)\right)$
Distance quadratique	$\Delta_{V^2}(q,p)$	$2(q-p)^2$
Distance de variation	$\Delta_{V}(q,p)$	$2\lvert q-p\rvert$
Distance linéaire	$\Delta_{\ell}(q,p)$	$p-q$
Discrimination triangulaire	$\Delta_{T}(q,p)$	$\frac{(q-p)^2}{q+p}+\frac{(q-p)^2}{2-q-p}$
Divergence de Catoni	$\Delta_{c}(q,p)$	$ln\,\frac{e^{-c\,q}}{1-p\,(1-e^{-c})}$

TAB. 7.2: Différentes Δ-fonctions pouvant être utilisées dans les bornes PAC-bayésiennes.

Il est également possible d'éviter d'avoir à calculer la somme de l'équation (7.5) en bornant supérieurement la valeur de $\mathcal{I}_{\Delta_{\mathrm{KL}}}(m)$ par une expression plus simple, en bornant la fonction $\eta(\cdot,\cdot)$ par l'un des lemmes 7.2 ou 7.3, ci-bas.

Lemme 7.2. *Étant donné deux entiers a et b tels que $0 \leq a \leq b$,*

$$\frac{1}{b+1} \;\leq\; \eta(a,b) \;\leq\; 1\,.$$

Démonstration. Le résultat est déduit de l'observation que $\eta(a,b)$ correspond à la fonction de masse d'une épreuve de Bernoulli de b essais avec probabilité de succès $\frac{a}{b}$, évaluée au point a, l'événement le plus probable parmi les $b+1$ résultats possibles. □

Du lemme 7.2, nous obtenons $\mathcal{I}^{\mathrm{K}}_{\Delta_{\mathrm{KL}}}(m) \leq m+1$ de manière triviale. Par contre, MAURER (2004) propose une borne plus serrée, soit $\mathcal{I}^{\mathrm{K}}_{\Delta_{\mathrm{KL}}}(m) \leq 2\sqrt{m}$. Le lemme 7.3 présente l'une des étapes clés de la démonstration de ce résultat. Nous réutilisons ce lemme pour obtenir une nouvelle garantie dans le cadre de l'apprentissage transductif à la section 7.3.

Lemme 7.3. *Étant donné deux entiers a et b tels que $0<a<b$,*

$$\sqrt{\tfrac{b}{2\pi a(b-a)}}e^{-\frac{1}{12a}-\frac{1}{12(b-a)}} \;<\; \eta(a,b) \;<\; \sqrt{\tfrac{b}{2\pi a(b-a)}}e^{\frac{1}{12b}}.$$

Démonstration. Le résultat est obtenu avec des calculs directs, en utilisant les bornes de Stirling pour la factorielle, c'est-à-dire $\sqrt{2\pi n}\left(\frac{n}{e}\right)^n < n! < \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n}}$. □

Nous concluons cette section en énonçant deux bornes PAC-bayésiennes utilisant les résultats précédents. La borne (a) est la même que le corollaire 3.6, et la borne (b) est la même que le corollaire 3.7.

Corollaire 7.4. *Pour toute distribution D, tout ensemble $\mathcal{H}$ de classificateurs, toute distribution a priori P sur $\mathcal{H}$ et tout $\delta \in (0,1]$, avec probabilité au moins $1-\delta$ sur le choix de $S \sim D^m$, nous avons*

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

$$a)\ \Delta_{\mathrm{KL}}(R_S(G_Q), R_D(G_Q)) \leq \frac{1}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta}\right],$$

$$b)\ R_D(G_Q) \leq R_S(G_Q) + \sqrt{\frac{1}{2m}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Démonstration. La borne (a) est obtenue du théorème 7.1, avec $\Delta(q,p) = \Delta_{\mathrm{KL}}(q,p)$, et $\mathcal{I}^{\mathrm{K}}_{\Delta_{\mathrm{KL}}}(m) \leq 2\sqrt{m}$ (Maurer, 2004). La borne (b) est obtenue de la borne (a), en utilisant l'inégalité de Pinsker : $\Delta_{\mathrm{KL}}(q,p) \geq 2(q-p)^2$ du lemme A.6. □

D'autres choix de Δ-fonctions permettent d'obtenir différentes bornes PAC-bayésiennes de la littérature. Par exemple, utiliser $\Delta_c = ln\,\frac{e^{-c\,q}}{1-p\,(1-e^{-c})}$ permet de retrouver la borne de Catoni (2007). Nous pouvons également retrouver des bornes similaires à celles de Pentina et Lampert (2015) et Alquier, Ridgway et Chopin (2015), en considérant la fonction linéaire $\Delta_\ell(q,p) = p - q$. Nous verrons également à la section 7.3 comment créer une divergence spécialisée au cadre de l'apprentissage transductif (Vapnik, 1998).

Dans la prochaine section, nous profitons de la démonstration simplifiée du théorème 7.1 et introduisons une nouvelle inégalité de changement de mesure permettant d'obtenir une borne PAC-bayésienne basée sur la divergence de Rényi.

7.2 Bornes PAC-bayésiennes basées sur la divergence de Rényi

Introduisons d'abord la *divergence de Rényi* (Rényi, 1961), sur laquelle sera basée une nouvelle inégalité de changement de mesure et une nouvelle famille de bornes PAC-bayésiennes.

Définition 7.5 (Divergence de Rényi)**.** Pour tout $\alpha > 1$, la divergence de Rényi entre les distributions Q et P est donnée par

$$D_\alpha(Q \,\|\, P) \triangleq \frac{1}{\alpha - 1}\, ln\left[\mathop{\mathbf{E}}_{h\sim P}\left(\frac{Q(h)}{P(h)}\right)^\alpha\right],$$

où $D_\alpha(Q \,\|\, P) = \mathrm{KL}(Q \,\|\, P)$ quand α tend vers 1.

Notons que la valeur de $D_\alpha(Q \,\|\, P)$ est toujours plus grande ou égale à la valeur de $\mathrm{KL}(Q \,\|\, P)$. De plus, étant donné une distribution a priori uniforme $U_{\mathcal{H}}$ sur $\mathcal{H}$ et une distribution a posteriori $U_{\mathcal{H}'}$ qui elle est uniforme sur un sous-ensemble $\mathcal{H}' \subseteq \mathcal{H}$, la divergence KL et la divergence de Rényi sont égales pour toute valeur de α. En particulier, quand $\mathcal{H}$ est un ensemble discret, nous avons $U_{\mathcal{H}}(h)=\frac{1}{|\mathcal{H}|}$ pour tout $h \in \mathcal{H}$, et $U_{\mathcal{H}'}(h)=\frac{1}{|\mathcal{H}'|}$ pour tout $h \in \mathcal{H}'$ et $U_{\mathcal{H}'}(h)=0$ autrement. Ainsi, $\forall \alpha \in (1, \infty)$:

$$D_\alpha(U_{\mathcal{H}'} \,\|\, U_{\mathcal{H}}) \;=\; \mathrm{KL}(U_{\mathcal{H}'} \,\|\, U_{\mathcal{H}}) \;=\; -\ln\left(\frac{|\mathcal{H}'|}{|\mathcal{H}|}\right).$$

Cette situation correspond au cas où la distribution $U_{\mathcal{H}'}$ décrit un vote de majorité démocratique, tels que ceux obtenus en exécutant un algorithme d'apprentissage tel que le *Bagging* (Breiman, 1996) ou les forêts aléatoires (Breiman, 2001).

7.2.1 Une nouvelle inégalité de changement de mesure

Nous présentons maintenant une nouvelle inégalité de changement de mesure, qui au lieu d'être basée sur la divergence Kullback-Leibler telle que l'inégalité du lemme 3.2, est basée sur la divergence de Rényi définie à la définition 7.5.

Théorème 7.6 (Changement de mesure basé sur Rényi)**.** *Pour tout ensemble $\mathcal{H}$ de votants, pour toute distribution P sur $\mathcal{H}$, toute distribution Q sur $\mathcal{H}$ dont le support est inclus dans le support de P, pour tout $\alpha > 1$, et pour toute fonction $\phi : \mathcal{H} \to \mathbb{R}$ mesurable sur P, nous avons*

$$\frac{\alpha}{\alpha-1} \ln \mathop{\mathbf{E}}_{h\sim Q} \phi(h) \;\leq\; D_\alpha(Q \,\|\, P) + \ln\left(\mathop{\mathbf{E}}_{h\sim P} \phi(h)^{\frac{\alpha}{\alpha-1}}\right).$$

Démonstration. Nous transformons d'abord l'espérance sur Q par une espérance sur P, et appliquons l'inégalité de Hölder (annexe A, lemme A.4), avec $r = \alpha$ et $s = \frac{\alpha}{\alpha-1}$. Plus précisément, nous avons

$$\begin{aligned}
\frac{\alpha}{\alpha-1} \ln \mathop{\mathbf{E}}_{h\sim Q} \phi(h) \;&=\; \frac{\alpha}{\alpha-1} \ln \mathop{\mathbf{E}}_{h\sim P}\left[\frac{Q(h)}{P(h)}\,\phi(h)\right] \\
&\leq\; \frac{\alpha}{\alpha-1} \ln\left(\left[\mathop{\mathbf{E}}_{h\sim P}\left(\frac{Q(h)}{P(h)}\right)^{\alpha}\right]^{\frac{1}{\alpha}} \left[\mathop{\mathbf{E}}_{h\sim P} \phi(h)^{\frac{\alpha}{\alpha-1}}\right]^{\frac{\alpha-1}{\alpha}}\right) \\
&=\; \frac{1}{\alpha-1} \ln\left[\mathop{\mathbf{E}}_{h\sim P}\left(\frac{Q(h)}{P(h)}\right)^{\alpha}\right] + \ln\left[\mathop{\mathbf{E}}_{h\sim P} \phi(h)^{\frac{\alpha}{\alpha-1}}\right] \\
&=\; D_\alpha(Q \,\|\, P) + \ln\left[\mathop{\mathbf{E}}_{h\sim P} \phi(h)^{\frac{\alpha}{\alpha-1}}\right].
\end{aligned}$$

Notons que l'inégalité de Hölder est valide lorsque $\frac{1}{r} + \frac{1}{s} = 1$, ce qui est le cas ici. □

Le théorème 7.6, avec $\phi(h)$ remplacé par $e^{(\alpha-1)\phi(h)}$, a été présenté dans Atar et Merhav (2015, équation (8)) en tant que « *risk-sensitive functional comparison bounds* »[5] (voir également Atar,

5. Atar et Merhav (2015) utilisent une autre définition de la divergence de Rényi, qui diffère par un facteur α.

Chowdhary et Dupuis (2015, corollaire 2.4)). La preuve présentée dans cette thèse est cependant beaucoup plus simple. Notons également que la fonction ϕ dans Atar et Merhav (2015) doit être bornée, alors que cette limitation n'est pas nécessaire ici. Cependant, le théorème 7.6 n'offre pas d'information intéressante dans les situation où ϕ n'est pas bornée, comme la quantité à droite de l'inégalité est infinie.

Nous observons que d'appliquer l'inégalité de Jensen sur la fonction concave $ln(\cdot)$ de la partie de gauche de l'inégalité du théorème 7.6 (avec $\phi(h)$ remplacé par $e^{\frac{\alpha-1}{\alpha}\phi(h)}$) permet d'obtenir l'inégalité de changement de mesure suivante, également basée sur la divergence de Rényi, mais qui est moins serrée :

$$\mathop{\mathbf{E}}_{h\sim Q}\phi(h) \;\leq\; D_\alpha(Q\,\|\,P) + ln\left(\mathop{\mathbf{E}}_{h\sim P} e^{\phi(h)}\right).$$

Cette inégalité a la même forme que l'inégalité de changement de mesure basée sur la divergence KL (lemme 3.2), en remplaçant $\mathrm{KL}(Q\,\|\,P)$ par $D_\alpha(Q\,\|\,P)$. De nouvelles bornes PAC-bayésiennes pourraient être dérivées à partir de cette inégalité, mais celles-ci seraient nécessairement moins serrées que les bornes traditionnelles, puisque la valeur de la divergence KL est toujours plus petite ou égale à la valeur de la divergence de Rényi, pour tout $\alpha > 1$. Pour cette raison, les bornes développées plus bas sont plutôt basées sur le théorème 7.6.

7.2.2 Bornes basées sur la divergence de Rényi

Nous présentons maintenant notre nouvelle famille de bornes PAC-bayésiennes basées sur la divergence de Rényi. La preuve associée à cette borne suit les étapes de la démonstration « personnalisable » présentée à la section 7.1, et contient les mêmes inégalités à l'exception du changement de mesure. Plutôt que d'utiliser le changement de mesure basé sur la divergence Kullback-Leibler, nous utilisons l'inégalité de changement de mesure basée sur la divergence de Rényi du théorème 7.6. La figure 7.2 montre les grandes lignes de la preuve, de manière schématique.

Théorème 7.7. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de classificateurs $h : \mathcal{X} \to \mathcal{Y}$, pour toute distribution P sur $\mathcal{H}$, pour tout $\delta \in (0,1]$, pour tout $\alpha > 1$ et pour toute fonction convexe $\Delta : [0,1]\times[0,1] \to \mathbb{R}$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$ln\;\Delta\Big(R_S\big(G_Q\big),\, R_D\big(G_Q\big)\Big) \;\leq\; \frac{1}{\alpha'}\left[D_\alpha(Q\,\|\,P) + ln\,\frac{\mathcal{I}^{\mathrm{R}}_\Delta(m,\alpha')}{\delta}\right],$$

où $\alpha' = \frac{\alpha}{\alpha-1}$, et

$$\mathcal{I}^{\mathrm{R}}_\Delta(m,\alpha') \;\triangleq\; \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\mathrm{Bin}^m_k(r)\,\Delta\left(\tfrac{k}{m},\,r\right)^{\alpha'}\right]. \tag{7.6}$$

		$ln\,\Delta\Big(\mathop{\mathbf{E}}_{h\sim Q} R_S(h),\ \mathop{\mathbf{E}}_{h\sim Q} R_D(h)\Big)$
Inégalité de Jensen	$\leq$	$ln\Big(\mathop{\mathbf{E}}_{h\sim Q} \Delta\big(R_S(h),\ R_D(h)\big)\Big)$
Changement de mesure	$\leq$	$\frac{1}{\alpha'}\left[D_\alpha(Q\,\|\,P) + ln\Big(\mathop{\mathbf{E}}_{h\sim P} \Delta\big(R_S(h),\ R_D(h)\big)^{\alpha'}\Big)\right]$
Inégalité de Markov	$\underset{1-\delta}{\leq}$	$\frac{1}{\alpha'}\left[D_\alpha(Q\,\|\,P) + ln\Big(\frac{1}{\delta}\mathop{\mathbf{E}}_{S'\sim D^m}\mathop{\mathbf{E}}_{h\sim P} \Delta\big(R_{S'}(h),\ R_D(h)\big)^{\alpha'}\Big)\right]$
Échange des espérances	$=$	$\frac{1}{\alpha'}\left[D_\alpha(Q\,\|\,P) + ln\Big(\frac{1}{\delta}\mathop{\mathbf{E}}_{h\sim P}\mathop{\mathbf{E}}_{S'\sim D^m} \Delta\big(R_{S'}(h),\ R_D(h)\big)^{\alpha'}\Big)\right]$
Loi binomiale	$=$	$\frac{1}{\alpha'}\left[D_\alpha(Q\,\|\,P) + ln\Big(\frac{1}{\delta}\mathop{\mathbf{E}}_{h\sim P}\sum_{k=0}^{m} \mathrm{Bin}_k^m(R_D(h))\,\Delta\big(\frac{k}{m}, R_D(h)\big)^{\alpha'}\Big)\right]$
Supremum sur le risque	$\leq$	$\frac{1}{\alpha'}\left[D_\alpha(Q\,\|\,P) + ln\Big(\frac{1}{\delta}\sup_{r\in[0,1]}\Big\{\sum_{k=0}^{m} \mathrm{Bin}_k^m(r)\,\Delta\big(\frac{k}{m},\ r\big)^{\alpha'}\Big\}\Big)\right]$

FIG. 7.2: Idée de la preuve de la borne PAC-bayésienne du théorème 7.7. Le symbole $\underset{1-\delta}{\leq}$ indique que l'inégalité est valide avec probabilité au moins $1-\delta$. L'étape indiquée en gras est celle qui introduit la différence avec la borne du théorème 7.1, présentée à la figure 7.1.

Démonstration. Nous démarrons avec la quantité à borner, c'est-à-dire $\Delta\big(R_S(G_Q), R_D(G_Q)\big)$. Nous appliquons d'abord l'inégalité de Jensen (lemme A.3) à la fonction convexe $\Delta(\cdot,\cdot)$, et l'inégalité du changement de mesure basé sur la divergence de Rényi (théorème 7.6), avec $\phi(h) = \Delta\big(R_S(h),\ R_D(h)\big)$. Nous avons donc, pour toute distribution Q sur $\mathcal{H}$,

$$\begin{aligned}
\alpha'\ ln\,\Delta\big(R_S(G_Q),\ R_D(G_Q)\big) \ &=\ \alpha'\ ln\,\Delta\left(\mathop{\mathbf{E}}_{h\sim Q} R_S(h),\ \mathop{\mathbf{E}}_{h\sim Q} R_D(h)\right) \\
&\leq\ \alpha'\ ln \mathop{\mathbf{E}}_{h\sim Q} \Delta\big(R_S(h),\ R_D(h)\big) \\
&\leq\ D_\alpha(Q\,\|\,P) + ln\Big(\underbrace{\mathop{\mathbf{E}}_{h\sim P} \Delta(R_S(h),\ R_D(h))^{\alpha'}}_{X_P(S)}\Big).
\end{aligned}$$

Considérons la variable aléatoire $X_P(S)$ définie ci-haut, et appliquons l'inégalité de Markov (lemme A.2). Nous avons donc, avec probabilité au moins $1-\delta$ sur le choix de $S\sim D^m$, que pour toute distribution Q sur $\mathcal{H}$,

$$\alpha'\,ln\,\Delta\big(R_S(G_Q),\ R_D(G_Q)\big)\ \leq\ D_\alpha(Q\,\|\,P) + ln\,\frac{\mathop{\mathbf{E}}_{S'\sim D^m} X_P(S')}{\delta}.$$

Le résultat final est obtenu en remplaçant $\mathbf{E}_{S'\sim D^m} X_P(S')$ par une borne supérieure en appliquant les mêmes étapes que dans la preuve du théorème 7.1,

$$\begin{aligned}\mathop{\mathbf{E}}_{S'\sim D^m} X_P(S') &= \mathop{\mathbf{E}}_{S'\sim D^m}\mathop{\mathbf{E}}_{h\sim P}\Delta\Big(R_{S'}(h),\, R_D(h)\Big)^{\alpha'} \\ &\leq \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\mathrm{Bin}_k^m(r)\Delta\left(\frac{k}{m},\, r\right)^{\alpha'}\right] \\ &= \mathcal{I}_\Delta^{\mathrm{R}}(m,\alpha')\,. \end{aligned}$$ □

En comparant les bornes des théorèmes 7.1 et 7.7, nous observons que les deux peuvent être paramétrées, en utilisant m' pour les bornes basées sur la divergence KL, et en utilisant α pour celles basées sur la divergence de Rényi. Dans ces dernières, la valeur de α a également un impact sur la valeur de la divergence. Nous remarquons également que la Δ-fonction apparaît en tant qu'exposant dans le théorème 7.1, alors qu'elle est la base d'un exposant dans le théorème 7.7. Comme les valeurs pourraient être plus petites dans cette dernière borne, ce résultat ouvre la voie à l'exploration d'alternatives pour les dernières étapes de la démonstration. Nous discutons d'une idée d'alternative à la section 7.2.4.

Le théorème 7.7 est énoncé comme une borne supérieure sur le logarithme de la Δ-fonction choisie, pour simplifier la comparaison avec le théorème 7.1. En effet, sous cette forme, les parties de droite des deux bornes sont très similaires. Pour borner la Δ-fonction directement, nous pouvons simplement appliquer une fonction exponentielle de chaque côté de de l'inégalité du théorème 7.7. De cette manière, nous obtenons que pour tout Q sur $\mathcal{H}$,

$$\Delta\Big(R_S(G_Q),\, R_D(G_Q)\Big) \;\leq\; \left[\mathop{\mathbf{E}}_{h\sim P}\left(\frac{Q(h)}{P(h)}\right)^{\alpha}\right]^{\frac{1}{\alpha}}\left[\frac{\mathcal{I}_\Delta^{\mathrm{R}}(m,\alpha')}{\delta}\right]^{\frac{1}{\alpha'}}. \tag{7.7}$$

En choisissant $\alpha = 2$ (et ainsi $\alpha' = 2$) dans l'équation (7.7), nous obtenons un cas spécial intéressant qui cette fois-ci est basé sur la *divergence chi carré* $\chi^2(Q\,\|\,P) \triangleq \mathop{\mathbf{E}}_{h\sim P}\left[\left(\frac{Q(h)}{P(h)}\right)^2 - 1\right]$. En fait, nous obtenons le corollaire suivant.

Corollaire 7.8. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de classificateurs $h : \mathcal{X}\to\mathcal{Y}$, pour toute distribution P sur $\mathcal{H}$, pour tout $\delta\in(0,1]$ et pour toute fonction convexe $\Delta : [0,1]\times[0,1]\to\mathbb{R}$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S\sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$\Delta\Big(R_S(G_Q),\, R_D(G_Q)\Big) \;\leq\; \sqrt{\left(\chi^2\left(Q\,\|\,P\right)+1\right)\left[\frac{\mathcal{I}_\Delta^{\mathrm{R}}(m,2)}{\delta}\right]},$$

où

$$\mathcal{I}_\Delta^{\mathrm{R}}(m,2) \;=\; \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\mathrm{Bin}_k^m(r)\,\Delta\left(\tfrac{k}{m},\, r\right)^2\right],$$

et où $\chi^2\left(Q\,\|\,P\right)$ est la divergence chi carré, *définie à la table 7.1.*

Démonstration. Nous débutons avec le théorème 7.7, où nous appliquons la fonction exponentielle de chaque côté de l'inégalité, obtenant ainsi l'inégalité de l'équation (7.7). En choisissant $\alpha = 2$ et en observant que $\alpha' = \frac{\alpha}{\alpha-1} = 2$, nous obtenons le résultat :

$$\begin{aligned}
\left[\mathop{\mathbf{E}}_{h\sim P}\left(\frac{Q(h)}{P(h)}\right)^{\alpha}\right]^{\frac{1}{\alpha}}\left[\frac{\mathcal{I}^{\mathrm{R}}_{\Delta}(m,\alpha')}{\delta}\right]^{\frac{1}{\alpha'}} &= \left[\mathop{\mathbf{E}}_{h\sim P}\left(\frac{Q(h)}{P(h)}\right)^{2}\right]^{\frac{1}{2}}\left[\frac{\mathcal{I}^{\mathrm{R}}_{\Delta}(m,2)}{\delta}\right]^{\frac{1}{2}} \\
&= \left[\mathop{\mathbf{E}}_{h\sim Q}\frac{P(h)}{Q(h)}\left(\frac{Q(h)}{P(h)}\right)^{2}\right]^{\frac{1}{2}}\left[\frac{\mathcal{I}^{\mathrm{R}}_{\Delta}(m,2)}{\delta}\right]^{\frac{1}{2}} \\
&= \left[\mathop{\mathbf{E}}_{h\sim Q}\left(\frac{Q(h)}{P(h)}\right)\right]^{\frac{1}{2}}\left[\frac{\mathcal{I}^{\mathrm{R}}_{\Delta}(m,2)}{\delta}\right]^{\frac{1}{2}} \\
&= \left[\mathop{\mathbf{E}}_{h\sim Q}\left(\frac{Q(h)}{P(h)}-1\right)+1\right]^{\frac{1}{2}}\left[\frac{\mathcal{I}^{\mathrm{R}}_{\Delta}(m,2)}{\delta}\right]^{\frac{1}{2}} \\
&= \left[\chi^2\left(Q\,\|\,P\right)+1\right]^{\frac{1}{2}}\left[\frac{\mathcal{I}^{\mathrm{R}}_{\Delta}(m,2)}{\delta}\right]^{\frac{1}{2}} \\
&= \sqrt{\left(\chi^2\left(Q\,\|\,P\right)+1\right)\frac{\mathcal{I}^{\mathrm{R}}_{\Delta}(m,2)}{\delta}}\,. \qquad \square
\end{aligned}$$

En utilisant ce corollaire avec la Δ-fonction linéaire $\Delta_\ell(q,p) = p - q$, nous obtenons le corollaire 7.9 ci-bas, qui est très similaire à celui de Honorio et Jaakkola (2014, lemme 7). Ce résultat ne peut pas être directement comparé au nôtre, puisqu'il s'applique à une famille paramétrée de classificateurs linéaires dans un cadre différent de celui étudié ici. Cependant, nous remarquons que le corollaire 7.9 a un terme de complexité plus petit, dû au facteur $\frac{1}{4}$ dans la racine carrée.

Corollaire 7.9. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de classificateurs $h : \mathcal{X} \to \mathcal{Y}$, pour toute distribution P sur $\mathcal{H}$ et pour tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$R_D(G_Q) \;\leq\; R_S(G_Q) + \sqrt{\frac{\chi^2\left(Q\,\|\,P\right)+1}{4m\delta}}\,.$$

Démonstration. Nous débutons la démonstration à partir du corollaire 7.8, avec $\Delta = \Delta_\ell$. Nous avons donc pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{H}$ de classificateurs $h : \mathcal{X} \to \mathcal{Y}$, pour toute distribution P sur $\mathcal{H}$ et pour tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{H}$,

$$R_D(G_Q) - R_S(G_Q) \;\leq\; \sqrt{\left(\chi^2\left(Q\,\|\,P\right)+1\right)\left[\frac{\mathcal{I}^{\mathrm{R}}_{\Delta_\ell}(m,2)}{\delta}\right]}\,.$$

Le résultat est obtenu en évaluant la valeur de $\mathcal{I}^{\mathrm{R}}_{\Delta_\ell}(m,2)$. Nous avons dans ce cas

$$\begin{aligned}
\mathcal{I}^{\mathrm{R}}_{\Delta_\ell}(m,2) &= \sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\mathrm{Bin}^m_k(r)\ \left(r-\tfrac{k}{m}\right)^2\right] \\
&= \frac{1}{m^2}\sup_{r\in[0,1]}\left[\sum_{k=0}^{m}\mathrm{Bin}^m_k(r)\ (k-rm)^2\right] \\
&= \frac{1}{m^2}\sup_{r\in[0,1]}\left[mr(1-r)\right] \qquad (7.8)\\
&= \frac{1}{m^2}\left[\frac{m}{2}\left(1-\frac{1}{2}\right)\right] \qquad (7.9)\\
&= \frac{1}{4m}\,,
\end{aligned}$$

ce qui démontre le résultat. L'égalité de l'équation (7.8) vient du fait que la première expression est la variance d'une distribution binomiale avec paramètres m et r, qui est connue pour être égale à $mr(1-r)$. L'égalité de l'équation (7.9) est trouvée en maximisant $mr(1-r)$ en fonction de r, en trouvant la valeur de r telle que la dérivée de l'expression est nulle. □

7.2.3 Étude empirique

Dans les expérimentations suivantes, nous comparons la précision des bornes PAC-bayésiennes basées sur la divergence de Rényi avec celles basées sur la divergence KL. De plus, nous étudions l'effet de chaque inégalité utilisée dans la borne (voir les figures 7.1 et 7.2). Pour y arriver, nous avons besoin de connaître et calculer chaque quantité intervenant à chaque étape de la preuve, incluant la distribution génératrice des données D, qui est normalement inconnue. Nous avons donc choisi de considérer une distribution synthétique. Nous considérons que chaque exemple généré par D est un tirage aléatoire parmi les 8124 exemples de l'ensemble de données *mushroom*, provenant du dépôt de données UCI (Lichman, 2013). L'ensemble d'entraînement $S \sim D^m$ contient donc m exemples, tirés *avec remise* avec *probabilité uniforme* parmi l'ensemble de données complet. À partir de cet ensemble de données m, nous apprenons un vote de majorité en utilisant l'algorithme AdaBoost (Freund et Schapire, 1997). Nous expérimentons avec trois sortes de votants différents.

Souches de décision Pour chacun des 22 attributs de l'ensemble de données mushroom, nous construisons 10 souches de décision avec des seuils de décision distribués également entre la valeur minimale et la valeur maximale de l'attribut. Pour chaque votant obtenu, nous considérons également la souche de décision qui prédit la décision inverse. Nous obtenons donc 440 votants très faibles.

Arbres (faibles) de décision Nous générons 500 arbres de décision en utilisant la librairie *Scikit-Learn* (Pedregosa et al., 2011). Chaque arbre est appris en utilisant 100 exemples

choisis aléatoirement parmi les exemples de l'ensemble de données.[6] Pour obtenir des arbres de précision individuelle assez faibles, nous fixons le paramètre *depth* à 3 et le paramètre *max_features* à 2.

Arbres (forts) de décision Nous générons 500 arbres de décision en utilisant la même procédure que pour les arbres de décision faibles, mais cette fois-ci avec les paramètres *depth* = 6 et *max_features* = 5.

Dans nos expérimentations, nous choisissons une distribution a priori P uniforme sur l'ensemble de votants utilisé. Nous utilisons deux Δ-fonctions: La divergence Kullback-Leibler entre deux distributions de Bernoulli Δ_{KL}, et la distance quadratique Δ_{V^2}. Rappelons que ces deux fonctions permettent de retrouver les bornes du corollaire 7.4 lorsque le changement de mesure basé sur la divergence KL est utilisé, et lorsque $m' = m$. Pour nos expérimentations, fixons les valeurs de $m' = m$ pour les bornes basées sur le KL, et $\alpha = 1.1$ pour les bornes basées sur la divergence de Rényi, puisque ces valeurs sont celles qui fournissent des valeurs de bornes près de l'optimal peu importe les valeurs des autres quantités qui interviennent dans celles-ci. Nous présentons les résultats obtenus pour ces choix de valeurs, et notons que nous ne présentons aucun résultat expérimental utilisant la distance linéaire Δ_ℓ et $\alpha = 2$ (donnant le corollaire 7.9), puisque les bornes résultantes sont significativement moins serrées.

Les quatre étapes montrées dans la figure 7.3 correspondent aux quatre inégalités de la démonstration des preuves PAC-bayésiennes (voir les figures 7.1 et 7.2). Par exemple, les valeurs montrées à l'étape *Inégalité de Jensen*, pour l'expérimentation avec la divergence KL et la Δ-fonction Δ_{KL}, sont calculées en trouvant la valeur de $r \geq R_S(G_Q)$ telle que

$$\Delta_{\mathrm{KL}}\left(R_S(G_Q),\, r\right) \;=\; \sum_{h \in \mathcal{H}} Q(h) \Delta_{\mathrm{KL}}\left(R_S(h),\, R_D(h)\right) \,.$$

Les valeurs à l'étape *Changement de mesure* sont calculées en trouvant la valeur de r telle que

$$m\, \Delta_{\mathrm{KL}}\left(R_S(G_Q), r\right) \;=\; \mathrm{KL}(Q \,\|\, P) + ln \sum_{h \in \mathcal{H}} P(h) e^{\left(m \Delta_{\mathrm{KL}}(R_S(h),\, R_D(h))\right)} \,.$$

Les deux dernières étapes sont calculées en utilisant la même méthode. Notons que la dernière inégalité est un supremum sur une valeur continue r, et doit dont être approximée lorsque le choix de Δ-fonction ne fournit pas une expression analytique. Comme nous expérimentations montrent que l'argument du supremum est « *smooth* » et n'a qu'un ou deux maximums locaux, nous avons utilisé une méthode classique de recherche des zéros d'une fonction : la *méthode de Brent* (Brent, 1973).

6. Notons que les bornes telles que présentées dans ce chapitre ne sont valides que lorsque les votants ne dépendent pas des exemples de l'ensemble d'entraînement. Comme notre but ici est d'étudier le comportement des bornes en utilisant des votants de précision différente, les arbres de décision simulent la situation où nous avons de la connaissance a priori sur la distribution des données. Notons qu'il existe des familles de bornes dont les votants peuvent être définis à partir d'exemples de l'ensemble d'entraînement, comme les bornes basées sur la compression d'échantillon présentées à la section 3.6.

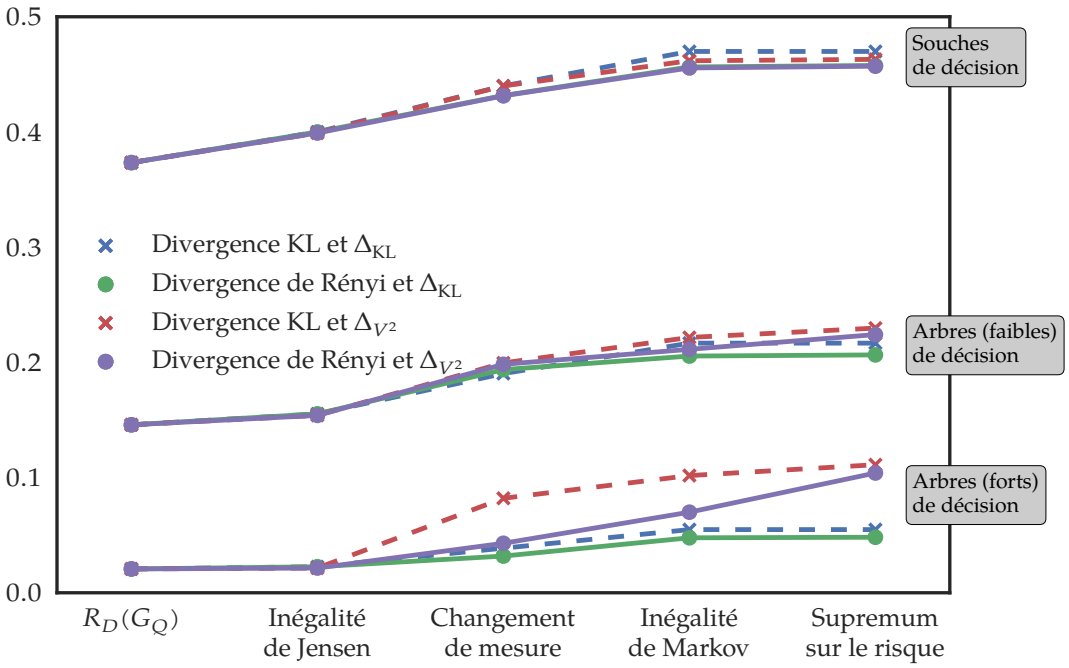

FIG. 7.3: Valeurs de bornes pour chaque sorte de votants : les souches de décision, et les arbres de décision faibles et forts. Les lignes pointillées correspondent aux bornes traditionnelles, considérant la divergence Kullback-Leibler. Les lignes pleines correspondes aux bornes considérant la divergence de Rényi. La valeur de la dernière étape donne la borne finale. Le risque du vote de majorité dans ces expérimentations est de 0.01 en utilisant les souches de décision, 0.001 en utilisant les arbres faibles de décision, et 0.002 en utilisant les arbres forts de décision.

En utilisant les arbres faibles de décision et les valeurs d'inégalités de la figure 7.3, la figure 7.4 met en relation la valeur de chaque Δ-fonction (en fonction du risque empirique de Gibbs) avec la partie de droite de chaque inégalité du processus de preuve. Cette figure offre une vue différente de la même expérimentation, et aide à la compréhension de l'impact du choix d'une Δ-fonction.

Nous observons qu'étant donné un vote de majorité et une Δ-fonction donnée, les bornes finales obtenues avec l'approche de Rényi donnent des bornes qui sont *légèrement* plus serrées que les bornes traditionnelles.[7] Dans certaines situations, nous observons que l'étape du changement de mesure est significativement plus serrée en utilisant les bornes basées sur la divergence de Rényi. Cependant, cet avantage est perdu dans les étapes subséquentes, lorsque nous appliquons l'inégalité de Markov et le supremum sur le risque.

7. Notons que cette observation n'est pas seulement vraie pour notre choix de valeurs de m' et α. En effet, nous avons observé empiriquement que les bornes basées sur la divergence de Rényi, avec la *meilleure* valeur de α, sont toujours plus serrées que les bornes basées sur la divergence KL, avec le meilleur choix de valeur pour m'.

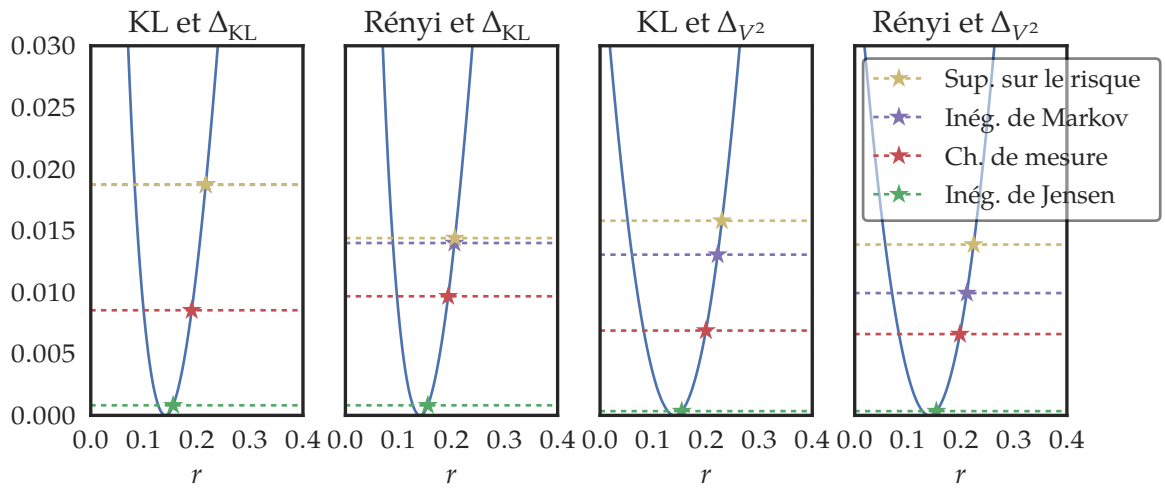

FIG. 7.4: Représentation alternative des quantités obtenues avec les arbres de décision faibles. La courbe bleue correspond à la fonction $\Delta(R_D(G_Q), r)$. Chaque ligne horizontale pointillée correspond à la valeur donnée par la partie de droite de la borne, après chaque inégalité. Sur chacune de ces lignes, la position de l'étoile donne la valeur de l'inégalité (sur l'axe des x). Notons que sur la figure la plus à gauche, l'inégalité liée au supremum sur le risque est en fait une égalité, puisque la borne basée sur la divergence KL, avec la Δ-fonction Δ_{KL}, fournit une expression analytique exacte pour le supremum.

7.2.4 Discussion

Dans la section 7.1, nous avons présenté une méthodologie de démonstration permettant de faire ressortir quatre inégalités principales constituant le fondement de toute borne PAC-bayésienne. La section 7.2 montre comment obtenir une nouvelle famille de bornes PAC-bayésiennes en remplaçant l'une de ces inégalités, l'inégalité du changement de mesure, par une nouvelle inégalité basée sur la divergence de Rényi. Nous avons également observé expérimentalement que cette inégalité est parfois plus serrée que l'inégalité originale, mais que l'avantage obtenu à cette étape est amoindri par les étapes subséquentes. Nous obtenons ainsi des borne qui ne sont que légèrement plus serrées.

Cependant, nous pensons que cette nouvelle méthodologie de preuve pourra motiver de nouvelles interventions sur les autres inégalités de la preuve. En particulier, nous avons vu que l'inégalité de Markov est également une étape où il y a une perte de précision dans la borne. Une piste de solution est de remplacer l'inégalité de Markov par l'inégalité de Cantelli, qui prendrait en considération la variance de la variable aléatoire étudiée. Cette variance est souvent très grande dans les bornes classiques, puisque la variable aléatoire dépend de l'exponentielle de la Δ-fonction (c'est-à-dire, $e^{m'\Delta(\cdot,\cdot)}$). Ce fait rend l'utilisation de l'inégalité de Cantelli peu appropriée dans le cas des bornes classiques basées sur la divergence KL, alors qu'elle pourrait mener à des améliorations avec notre nouvelle famille de bornes basées sur la divergence de Rényi. En effet, notre nouvelle famille de bornes devrait être moins sen-

sible à ce phénomène, puisque la Δ-fonction apparaît plutôt comme base de l'exponentielle (c'est-à-dire, $\Delta(\cdot,\cdot)^{\alpha'}$).

Finalement, notons que la plupart des bornes présentées ici ne sont pas explicites, sauf le corollaire 7.9 qui donne des bornes avec des valeurs empiriques décevantes. Celles-ci peuvent donc être moins attrayantes pour la conception de nouveaux algorithmes d'apprentissage, car il faut d'abord trouver une Δ-fonction pour laquelle la valeur de $\mathcal{J}^{\mathrm{R}}_{\Delta}(m,\alpha')$ de l'équation (7.6) est bornée par une expression analytique. De telles nouvelles bornes analytiques ouvriraient la voix à la conception d'algorithmes d'apprentissage avec une nouvelle forme de régularisation basée sur la divergence de Rényi, alors que la plupart des algorithmes issus des bornes PAC-bayésiennes sont régularisés par la divergence KL (Germain, Lacasse, Laviolette et Marchand, 2009 ; Parrado-Hernández et al., 2012 ; Pentina et Lampert, 2015 ; Alquier, Ridgway et Chopin, 2015).

Dans la prochaine section, nous explorons un cadre d'apprentissage différent : l'apprentissage transductif. Nous montrons comment adapter la preuve personnalisable de la section 7.1 pour cette situation, et montrons comment construire une Δ-fonction adaptée à cette nouvelle famille de bornes, obtenant ainsi de nouvelles bornes explicites.

7.3 Théorie PAC-bayésienne transductive

En classification, la plupart des algorithmes sont conçus pour le cadre de l'apprentissage inductif, qui correspond à l'expérimentation suivante : un algorithme reçoit un échantillon fini d'exemples (l'ensemble d'entraînement), généré de manière indépendamment et identiquement distribuée (*i.i.d.*) d'une distribution inconnue D. On demande ensuite à l'algorithme de produire un classificateur ayant une faible probabilité de classifier incorrectement un exemple tiré de la distribution D (c'est-à-dire, un faible *risque de généralisation*). La figure 7.5 présente cette situation de manière schématique.

Certaines tâches ne peuvent pas être modélisées par ce cadre d'apprentissage. Le cadre *i.i.d.* implique par exemple qu'il n'y a aucune corrélation entre les exemples d'entraînement, ce qui est une hypothèse forte. Par exemple, considérons une expérimentation où l'on recueille un ensemble fini Z d'exemples (possiblement non *i.i.d.*), où l'on demande ensuite à un expert d'étiqueter un sous-ensemble S d'exemples tirés de Z (sans remise), où nous exécutons ensuite un algorithme d'apprentissage sur Z (où le sous-ensemble S est étiqueté, et $Z \setminus S$ ne l'est pas), pour ensuite demander au classificateur obtenu d'étiqueter les exemples restants $Z \setminus S$. La figure 7.6 présente cette situation de manière schématique. Le cadre transductif introduit par Vapnik (1998) propose un paradigme pour lequel on peut obtenir des garanties de généralisation dans une telle situation non *i.i.d*.

Dans cette section, nous présentons d'abord plus formellement les différences entre l'ap-

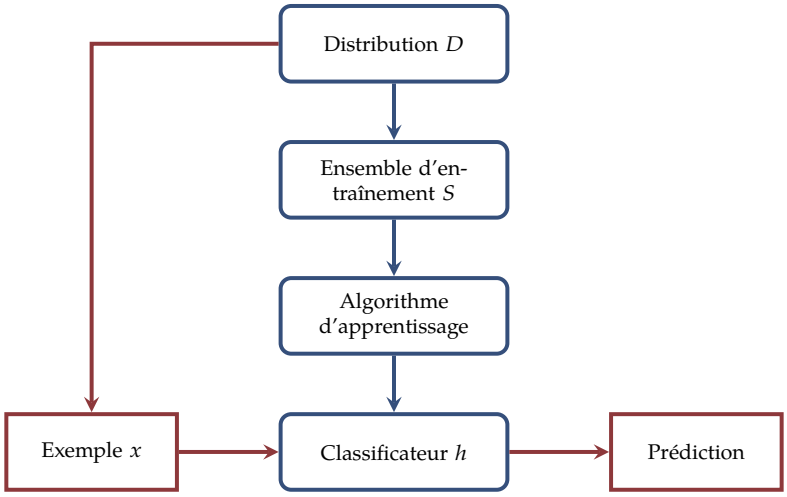

FIG. 7.5: Représentation du cadre de l'apprentissage inductif. Le chemin déterminé par les flèches bleues représente le processus d'apprentissage, et le chemin déterminé par les flèches rouges représente la tâche à résoudre une fois l'apprentissage terminé.

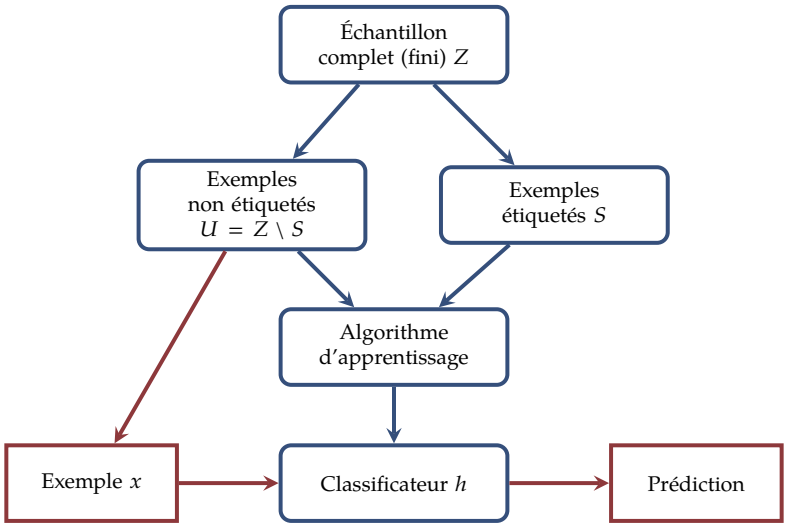

FIG. 7.6: Représentation du cadre de l'apprentissage transductif. Le chemin déterminé par les flèches bleues représente le processus d'apprentissage, et le chemin déterminé par les flèches rouges représente la tâche à résoudre une fois l'apprentissage terminé.

prentissage inductif et l'apprentissage transductif. Nous présentons ensuite une extension du théorème PAC-bayésien inductif, adapté au cadre de l'apprentissage transductif en adaptant la preuve personnalisable développée à la section 7.1. Nous analysons finalement le comportement de cette borne en considérant différentes Δ-fonctions, puis la comparons avec l'état de l'art sur des données réelles.

7.3.1 Comparaison entre l'apprentissage inductif et transductif

En apprentissage inductif, nous considérons que chaque exemple (x, y) est tiré *i.i.d.* à partir d'une distribution fixe mais inconnue D sur $\mathcal{X}\times\mathcal{Y}$. De cet ensemble D est tiré un ensemble d'entraînement de m exemples, de manière indépendamment et identiquement distribuée (*i.i.d.*). La tâche est d'apprendre un classificateur h qui étant donné S, sera en mesure de classifier correctement de nouveaux exemples tirés selon la distribution inconnue D.

En apprentissage transductif, nous considérons un ensemble Z de z exemples, $Z = \langle (x_1, y_1), (x_2, y_2), \ldots, (x_z, y_z)\rangle$, souvent appelé l'*échantillon complet*, qui contient *tous* les exemples d'intérêt pour la tâche à résoudre.[8] Nous obtenons un ensemble d'entraînement S en *tirant m exemples de Z sans remise*. Les exemples de Z restants forment un ensemble U de $z-m$ exemples. Dans le cadre transductif, un algorithme d'apprentissage reçoit un ensemble d'entraînement S et $U_{\mathcal{X}} = \langle x_{m+1}, x_{m+2}, \ldots, x_z\rangle$, l'ensemble des exemples non étiquetés de U. La tâche de l'algorithme d'apprentissage transductif est d'apprendre un classificateur $h : Z_{\mathcal{X}} \to \mathcal{Y}$ qui classifie correctement les exemples non étiquetés de l'ensemble U.[9] La tâche est donc de minimiser le risque du classificateur sur l'ensemble U.

Nous définissons les risques $R_Z(h)$ et $R_S(h)$ de manière habituelle lorsque nous considérons des ensembles finis (voir la convention 2.1). Notons que nous pouvons retrouver $R_Z(h)$ à partir de $R_S(h)$ et $R_U(h)$, comme

$$R_Z(h) \;=\; \frac{1}{z}\Big(mR_S(h) + (z-m)R_U(h)\Big)\,. \tag{7.10}$$

Le cadre de l'apprentissage transductif est intimement relié au cadre de l'apprentissage *semi-supervisé*, qui est décrit par la situation suivante. À partir d'une distribution D inconnue, un ensemble d'entraînement S de m exemples est tiré *i.i.d.*, et un ensemble d'exemples *non étiquetés*, potentiellement très grand, est également tiré *i.i.d.* selon D. Dans les deux cas, une quantité potentiellement grande d'exemples non étiquetés est disponible à l'algorithme d'apprentissage, et celui-ci pourra en tirer de l'information lors de l'entraînement.

En apprentissage inductif et transductif, le but est de trouver le classificateur avec le risque le plus bas possible sur une distribution ou un ensemble qui n'est pas complètement connu : la distribution D en apprentissage inductif, et l'ensemble U en apprentissage transductif, pour lequel nous ne connaissons pas les étiquettes. Dans la prochaine section, nous développons des théorèmes PAC-bayésiens adaptés au cadre transductif, nous permettant de borner supérieurement ces risques à partir du risque empirique et d'un terme de complexité.

8. Dans le «*Setting 1*» de l'apprentissage transductif de Vapnik (1998), l'échantillon complet est tiré *i.i.d.* à partir d'une distribution inconnue sur $\mathcal{X}\times\mathcal{Y}$. Cette supposition n'est pas requise ici.

9. Dans Vapnik (1998), le classificateur transductif est défini en utilisant $U_{\mathcal{X}}$ comme espace d'entrée, c'est-à-dire $h : U_{\mathcal{X}} \to \mathcal{Y}$. Par contre, les bornes PAC-bayésiennes ont besoin que le classificateur soit également applicable sur les exemples de S, comme celles-ci nécessitent le calcul de $R_S(h)$.

7.3.2 Théorèmes PAC-bayésiens transductifs avec et sans KL

Dans cette section, nous proposons une analyse PAC-bayésienne adaptée à l'apprentissage transductif, en proposant une famille de nouvelles bornes. Une partie de ces résultats a été présentée dans Bégin, Germain, Laviolette et Roy (2014), que nous reprenons dans cette thèse en considérant le processus de preuve personnalisable présenté à la section 7.1 et dans Bégin, Germain, Laviolette et Roy (2016). Nous ajoutons également de nouveaux résultats basés sur nos contributions du chapitre 3, puis nous comparons le comportement de ces bornes. Notons qu'une borne transductive a été développée dans Laviolette, Marchand et Roy (2011), mais celle-ci est une borne asymptotique est n'est donc pas utilisable en pratique.

Le but final de cette section est d'obtenir une PAC-borne transductive sur le risque du classificateur par vote de majorité. Dans le chapitre 3, nous avons développé deux PAC-bornes : l'une utilisant deux fois le risque de Gibbs, et l'une utilisant plutôt la C-borne. La C-borne dépend non seulement du risque de Gibbs (ou du premier moment de la marge), mais aussi du désaccord espéré (ou du second moment de la marge), dont le calcul ne nécessite pas les étiquettes des exemples (voir la remarque 2.9). Dans le cadre transductif, *le désaccord espéré n'a donc pas à être borné*, car on peut calculer sa vraie valeur.

Nous nous concentrons donc uniquement sur la théorie PAC-bayésienne pour le risque de Gibbs. Nous présentons les théorèmes sous la même forme que dans Bégin, Germain, Laviolette et Roy (2014), mais les démonstrations associées sont basées sur la méthodologie de démonstration personnalisable présentée à la section 7.1.

La théorie PAC-bayésienne a d'abord été étendue au cadre transductif par Derbeko, El-Yaniv et Meir (2004). Nous proposons ici des bornes plus serrées, qui ne souffrent pas du principal problème de la borne de Derbeko, El-Yaniv et Meir (2004) : sa valeur diverge vers l'infini comme le nombre d'exemples non étiquetés grandit. Nous proposons un théorème PAC-bayésien transductif général, permettant de dériver diverses bornes en choisissant une fonction convexe Δ. Un avantage supplémentaire du cadre transductif est que comparativement au cadre inductif, notre théorème PAC-bayésien transductif n'a pas de limitation au niveau du choix de la fonction Δ, puisque le supremum qui doit être estimé dans le cas inductif est remplacé par un *max* dans le cas transductif et peut donc être calculé de manière exacte. Nous proposons finalement une borne prenant en considération les exemples non étiquetés via le désaccord espéré, alors que les bornes transductives usuelles ne considèrent que le nombre d'exemples non étiquetés.

Un théorème PAC-bayésien transductif général

Dans le cadre inductif, une supposition importante utilisée pour dériver les garanties PAC-bayésiennes est que les m exemples de l'ensemble d'entraînement S sont tirés *i.i.d.* de la distribution D. Dans la preuve du théorème 7.1, nous utilisons cette observation pour exprimer

la probabilité de classifier incorrectement k exemples parmi m comme une distribution binomiale. Cette supposition ne tient pas dans ici. En effet, en apprentissage transductif, l'ensemble des exemples étiquetés S est un sous-ensemble d'un ensemble fini Z. Alors, le nombre d'erreurs observées dans S suit une *distribution hypergéométrique*, comme S contient m tirages *sans remise* de Z. Cette idée est exploitée dans la démonstration du théorème 7.10, ci bas.

Théorème 7.10. *Pour tout ensemble Z de z exemples, pour tout ensemble $\mathcal{H}$ de classificateurs $h : \mathcal{X} \to \mathcal{Y}$, pour toute distribution a priori P sur $\mathcal{H}$, pour tout $\delta \in (0,1]$, pour tout $m' > 0$ et pour toute fonction convexe $\Delta : [0,1]\times[0,1] \to \mathbb{R}$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un sous-ensemble S de m exemples tiré sans remise parmi Z, que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$\Delta\Big(R_S(G_Q), R_Z(G_Q)\Big) \;\leq\; \frac{1}{m'}\left[\mathrm{KL}(Q \,\|\, P) + ln\, \frac{\mathfrak{J}_\Delta^{\mathrm{K}}(m, m', z)}{\delta}\right],$$

où

$$\mathfrak{J}_\Delta^{\mathrm{K}}(m, m', z) \;\triangleq\; \max_{\hat{k}\in\{0,\ldots,z\}}\left[\sum_{k\in\mathcal{K}_{mz\hat{k}}} \mathrm{Hyp}_k^{m,z}(\hat{k})\; e^{m'\Delta\left(\frac{k}{m}, \frac{\hat{k}}{z}\right)}\right], \tag{7.11}$$

où $\mathrm{Hyp}_k^{m,z}(\hat{k})$ *est la fonction de masse de la distribution hypergéométrique :*

$$\mathrm{Hyp}_k^{m,z}(\hat{k}) \;\triangleq\; \frac{\binom{\hat{k}}{k}\binom{z-\hat{k}}{m-k}}{\binom{z}{m}},$$

et $\mathcal{K}_{mz\hat{k}} \;\triangleq\; \Big\{max[0,\, \hat{k}+m-z], \ldots, min[m,\, \hat{k}]\Big\}$.

Démonstration. Dénotons $[Z]^m$ la distribution uniforme sur tous les sous-ensembles de Z de taille m. Nous démarrons avec la quantité à borner, c'est-à-dire $\Delta\Big(R_S(G_Q),\, R_Z(G_Q)\Big)$. Nous appliquons d'abord l'inégalité de Jensen (lemme A.3) à la fonction convexe $\Delta(\cdot,\cdot)$, et l'inégalité du changement de mesure basée sur la divergence Kullback-Leibler de l'équation (7.1), avec $\phi(h) = m'\,\Delta\Big(R_S(h), R_Z(h)\Big)$. Nous avons donc, pour toute distribution Q sur $\mathcal{H}$,

$$\begin{aligned}
m'\,\Delta\Big(R_S(G_Q), R_Z(G_Q)\Big) \;&=\; m'\,\Delta\Big(\mathop{\mathbf{E}}_{h\sim Q} R_S(h), \mathop{\mathbf{E}}_{h\sim Q} R_Z(h)\Big) \\
&\leq\; \mathop{\mathbf{E}}_{h\sim Q} m'\,\Delta\Big(R_S(h), R_Z(h)\Big) \\
&\leq\; \mathrm{KL}(Q \,\|\, P) + ln\Big(\underbrace{\mathop{\mathbf{E}}_{h\sim P} e^{m'\,\Delta\left(R_S(h), R_Z(h)\right)}}_{X_P(S)}\Big).
\end{aligned}$$

Considérons maintenant la variable aléatoire $X_P(S)$ définie ci-haut, et appliquons l'inégalité de Markov (lemme A.2), pour obtenir

$$\Pr_{S\sim D^m}\left(X_P(S) \;\leq\; \frac{1}{\delta}\mathop{\mathbf{E}}_{S'\sim[Z]^m} X_P(S')\right) \;\geq\; 1-\delta\,.$$

Ceci implique qu'avec probabilité au moins $1-\delta$ sur le choix d'un ensemble $S \sim [Z]^m$, nous avons, simultanément pour tout Q sur $\mathcal{H}$, que

$$m' \, \Delta\Big(R_S(G_Q), \, R_Z(G_Q)\Big) \; \leq \; \mathrm{KL}(Q \,\|\, P) + ln \, \frac{\underset{S' \sim [Z]^m}{\mathbf{E}} X_P(S')}{\delta} \, . \tag{7.12}$$

Finalement, bornons supérieurement le terme $\mathbf{E}_{S' \sim [Z]^m} X_P(S')$, premièrement en échangeant les deux espérances, puis en utilisant le fait que le nombre d'erreurs $m \, R_{S'}(h)$ suit une distribution hypergéométrique de m tirages parmi une population de taille z contenant $zR_Z(h)$ succès. Nous avons donc

$$\begin{aligned}
\underset{S' \sim [Z]^m}{\mathbf{E}} X_P(S') \; &= \; \underset{S' \sim [Z]^m}{\mathbf{E}} \underset{h \sim P}{\mathbf{E}} \; e^{\, m' \, \Delta\Big(R_{S'}(h), R_Z(h)\Big)} \\
&= \; \underset{h \sim P}{\mathbf{E}} \underset{S' \sim [Z]^m}{\mathbf{E}} \; e^{\, m' \, \Delta\Big(R_{S'}(h), R_Z(h)\Big)} \\
&= \; \underset{h \sim P}{\mathbf{E}} \sum_{k=max(0, \, zR_Z(h)+m-z)}^{min(m, \, zR_Z(h))} \underset{S' \sim [Z]^m}{Pr} \left(R_{S'}(h) = \frac{k}{m} \right) \; e^{\, m' \, \Delta\Big(\frac{k}{m}, R_Z(h)\Big)} \\
&= \; \underset{h \sim P}{\mathbf{E}} \sum_{k=max(0, \, zR_Z(h)+m-z)}^{min(m, \, zR_Z(h))} \mathrm{Hyp}_k^{m,z}\Big(zR_Z(h)\Big) \; e^{\, m' \, \Delta\Big(\frac{k}{m}, R_Z(h)\Big)} \\
&\leq \; \underset{\hat{k} \in \{0, \ldots, z\}}{max} \left[\sum_{k=max(0, \, \hat{k}+m-z)}^{min(m, \, \hat{k})} \mathrm{Hyp}_k^{m,z}(\hat{k}) \; e^{\, m' \, \Delta\Big(\frac{k}{m}, \frac{\hat{k}}{z}\Big)} \right] \\
&= \; \mathcal{J}_\Delta^{\mathrm{K}}(m, m', z) \, .
\end{aligned}$$

Le résultat final est obtenu en remplaçant $\mathbf{E}_{S' \sim [Z]^m} X_P(S')$ par sa borne supérieure $\mathcal{J}_\Delta^{\mathrm{K}}(m, m', z)$ dans l'équation (7.12). □

La figure 7.7 montre les grandes lignes de la preuve, de manière schématique et plus simple à comparer avec les preuves schématisées aux figures 7.1 et 7.2.

Maintenant, en suivant la démarche de la section 3.5 du chapitre 3, nous dérivons une borne transductive générale sans régularisateur KL en considérant des distributions Q alignées sur la distribution a priori P. En utilisant les mêmes définitions qu'à la section 3.5, cette fois-ci en considérant le cadre transductif et les ensembles de classificateurs $\mathcal{H}$ plutôt que les ensembles de votants $\mathcal{F}$, nous obtenons la borne suivante.

Théorème 7.11. *Pour tout ensemble Z de z exemples, pour tout ensemble symétrique $\mathcal{H}$ de classificateurs $h : \mathcal{X} \to \mathcal{Y}$, pour toute distribution a priori P sur $\mathcal{H}$, pour tout $\delta \in (0, 1]$, pour tout $m' > 0$ et pour toute fonction convexe $\Delta : [0,1] \times [0,1] \to \mathbb{R}$ pour laquelle $\Delta(q,p) = \Delta(1-q, 1-p)$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un sous-ensemble S de m exemples tiré sans remise parmi Z, que pour toute distribution a posteriori Q alignée sur P,*

$$\Delta\Big(R_S(G_Q), \, R_Z(G_Q)\Big) \; \leq \; \frac{1}{m'} \left[ln \frac{\mathcal{J}_\Delta^{\mathrm{K}}(m, m', z)}{\delta} \right] ,$$

		$\Delta\Big(\mathop{\mathbf{E}}_{h\sim Q} R_S(h),\ \mathop{\mathbf{E}}_{h\sim Q} R_Z(h)\Big)$
Inégalité de Jensen	$\leq$	$\mathop{\mathbf{E}}_{h\sim Q} \Delta\big(R_S(h),\ R_Z(h)\big)$
Changement de mesure	$\leq$	$\dfrac{1}{m'}\Big[\mathrm{KL}(Q \,\|\, P) + ln\Big(\mathop{\mathbf{E}}_{h\sim P} e^{m'\Delta\left(R_S(h),\, R_Z(h)\right)}\Big)\Big]$
Inégalité de Markov	$\underset{1-\delta}{\leq}$	$\dfrac{1}{m'}\left[\mathrm{KL}(Q \,\|\, P) + ln\left(\dfrac{1}{\delta} \mathop{\mathbf{E}}_{S'\sim[Z]^m} \mathop{\mathbf{E}}_{h\sim P} e^{m'\Delta\left(R_{S'}(h),\, R_Z(h)\right)}\right)\right]$
Échange des espérances	$=$	$\dfrac{1}{m'}\left[\mathrm{KL}(Q \,\|\, P) + ln\left(\dfrac{1}{\delta} \mathop{\mathbf{E}}_{h\sim P} \mathop{\mathbf{E}}_{S'\sim[Z]^m} e^{m'\Delta\left(R_{S'}(h),\, R_Z(h)\right)}\right)\right]$
Loi hypergéométrique	$=$	$\dfrac{1}{m'}\left[\mathrm{KL}(Q \,\|\, P) + ln\left(\dfrac{1}{\delta} \mathop{\mathbf{E}}_{h\sim P} \sum_{k\in\mathcal{K}_{m z \hat{k}}} \mathrm{Hyp}_k^{m,z}(zR_Z(h)) e^{m'\Delta\left(\frac{k}{m},\, R_Z(h)\right)}\right)\right]$
Maximum sur le risque	$\leq$	$\dfrac{1}{m'}\left[\mathrm{KL}(Q \,\|\, P) + ln\left(\dfrac{1}{\delta} \max_{\hat{k}\in\{0,\ldots,z\}} \left\{ \sum_{k\in\mathcal{K}_{m z \hat{k}}} \mathrm{Hyp}_k^{m,z}(\hat{k})\, e^{m'\Delta\left(\frac{k}{m},\, \frac{\hat{k}}{z}\right)} \right\}\right)\right]$

FIG. 7.7: Idée de la preuve de la borne transductive du théorème 7.10. Le symbole $\underset{1-\delta}{\leq}$ indique que l'inégalité est valide avec probabilité au moins $1-\delta$. L'étape indiquée en gras est celle qui introduit la différence avec la borne du théorème 7.1, présentée à la figure 7.1.

où $\mathcal{I}_\Delta^{\mathrm{K}}(m, m', z)$ *est défini à l'équation 7.11.*

Démonstration. La démonstration suit exactement les mêmes étapes que la démonstration du théorème 7.10, en remplaçant l'utilisation de l'inégalité du changement de mesure du lemme 3.2 (ou de l'équation (7.1)) par celle du lemme 3.15, en spécialisant le lemme aux classificateurs $h \in \mathcal{H}$, et avec $\phi(h) = m' \cdot \Delta\big(R_S(h),\, R_Z(h)\big)$. Les autres étapes de la démonstration restent exactement les mêmes. □

Il y a une correspondance entre les expressions $\mathcal{I}_\Delta^{\mathrm{K}}(m, m')$ du théorème 7.1 (le cas inductif) et l'expression $\mathcal{I}_\Delta^{\mathrm{K}}(m, m', z)$ des théorèmes 7.10 et 7.11 (le cas transductif). Notons d'abord que comme nous avons fait dans le cas inductif, nous simplifierons la notation avec $\mathcal{I}_\Delta^{\mathrm{K}}(m, z)$ lorsque $m' = m$. Dans le cadre inductif, pour calculer la valeur de $\mathcal{I}_\Delta^{\mathrm{K}}(m)$, il nous faut trouver le supremum d'une expression qui expérimentalement ne semble avoir qu'un ou deux maximums locaux, mais dont nous n'avons pas la preuve. Pour cette raison, l'utilisation d'une Δ-fonction quelconque pourrait causer problème et le calcul des bornes pourrait être erroné. Dans le cadre transductif par contre, la valeur $\mathcal{I}_\Delta^{\mathrm{K}}(m, z)$ est donnée par la valeur maximale d'une somme sur une variable discrète $\hat{k} \in \{0, 1, 2, \ldots, z\}$. Cette valeur peut être calculée directement pour toute Δ-fonction, à condition que m et z ne soient pas déraisonnablement grands. Ceci ouvre la porte à l'utilisation de plusieurs Δ-fonctions.

Par exemple, un choix naturel de Δ-fonction est la divergence $\Delta_{\mathrm{KL}}(q,p)$, comme celle-ci permet d'obtenir l'une des bornes les plus serrées dans le cadre inductif. Par les équations (3.2) et (7.11), we nous obtenons

$$\mathcal{J}^{\mathrm{K}}_{\Delta_{\mathrm{KL}}}(m,z) \;=\; \max_{\hat{k}\in\{0,\ldots,z\}}\left[\sum_{k\in\mathcal{K}_{mz\hat{k}}}\frac{\binom{\hat{k}}{k}\binom{z-\hat{k}}{m-k}}{\binom{z}{m}}\left(\frac{k/m}{\hat{k}/z}\right)^{k}\left(\frac{1-k/m}{1-\hat{k}/z}\right)^{m-k}\right].$$

Malheureusement, l'expression obtenue ne se simplifie pas comme dans le cas inductif (voir l'équation (7.5)). Dans la prochaine section, nous concevons dans la prochaine section une nouvelle Δ-fonction adaptée au cadre transductif.

7.3.3 Une Δ-fonction pour le cadre transductif

Dans le cadre inductif, nous pouvons exprimer $\mathcal{J}^{\mathrm{K}}_{\Delta}(m)$ par une somme de termes $\eta(k,m)$, définis à l'équation (7.5), pouvant être bornés en utilisant les lemmes 7.2 ou 7.3. Pour retrouver le même phénomène dans le cadre transductif, nous suggérons d'utiliser la Δ-fonction suivante, qui paire chacun des trois coefficients binomiaux de $\mathcal{J}^{\mathrm{K}}_{\Delta}(m,z)$ avec un terme d'entropie approprié, défini à l'équation (7.4).

$$\Delta^{\star}_{\beta}(q,p) \;\triangleq\; \frac{H(\beta) - pH\left(\beta\frac{q}{p}\right) - (1-p)H\left(\beta\frac{1-q}{1-p}\right)}{\beta}. \tag{7.13}$$

Le paramètre β sera typiquement fixé à $\frac{m}{z}$. Tel que montré par le lemme A.15 de l'annexe A.4, l'équation (7.13) peut être réécrite comme

$$\Delta^{\star}_{\beta}(q,p) \;=\; \Delta_{\mathrm{KL}}(q,p) + \tfrac{1-\beta}{\beta}\,\Delta_{\mathrm{KL}}\left(\tfrac{p-\beta q}{1-\beta},p\right). \tag{7.14}$$

L'équation précédente illustre que lorsque $z\to\infty$ et m est fini, $\Delta^{\star}_{m/z}(q,p)$ converge vers $\Delta_{\mathrm{KL}}(q,p)$. Nous retrouvons donc la divergence KL utilisée dans le cadre inductif lorsque la cardinalité de l'échantillon complet est infinie.

Il est intéressant de noter que la formulation de $\Delta^{\star}_{\beta}(q,p)$ de l'équation (7.14) apparaît dans la preuve du théorème transductif de Derbeko, El-Yaniv et Meir (2004). Par contre, tel que nous discutons dans la section 7.3.5, nous sommes en mesure avec notre approche de dériver des bornes plus serrées en utilisant cette même Δ-fonction. En effet, lorsque nous introduisons l'équation (7.13) dans l'équation (7.11), avec $\beta=\frac{m}{z}$, nous avons

$$\mathcal{J}^{\mathrm{K}}_{\Delta^{\star}_{m/z}}(m,z) \;=\; \max_{\hat{k}\in\{0,\ldots,z\}}\left[\sum_{k\in\mathcal{K}_{mz\hat{k}}}\frac{\eta(k,\hat{k})\,\eta(m-k,\,z-\hat{k})}{\eta(m,z)}\right].$$

Par le lemme 7.2, nous obtenons trivialement que

$$\mathcal{J}^{\mathrm{K}}_{\Delta^{\star}_{m/z}}(m,z) \;\leq\; \max_{\hat{k}\in\{0,\ldots,z\}}\sum_{k=0}^{m} z+1 \;=\; (m+1)(z+1). \tag{7.15}$$

Par contre, cette borne supérieure sur $\mathcal{T}^{\mathrm{K}}_{\Delta^\star_{m/z}}(m,z)$ est loin d'être serrée, tel que démontré par le théorème suivant.

Théorème 7.12. *Soit m et z deux entiers quelconques tels que $20 \leq m \leq z-20$, nous avons*

$$\mathcal{T}_{\Delta^\star_{m/z}}(m,z) \;\leq\; t(m,z) \;\triangleq\; 3\,ln(m)\sqrt{m\left(1-\frac{m}{z}\right)}. \tag{7.16}$$

Démonstration. Étant donné des valeurs m, z et $\hat{k}$ fixes, soit $k^- = max[0, \hat{k}+m-z]$, $k^+ = min[m,\hat{k}]$, $\mathcal{K}^*_{mz\hat{k}} = \mathcal{K}_{mz\hat{k}} \setminus \{k^-, k^+\}$, et

$$F(k) = \frac{\eta(k,\hat{k})\,\eta(m-k, z-\hat{k})}{\eta(m,z)}.$$

Le lemme A.16 de l'annexe A montre que

$$F(k^-) + F(k^+) \;\leq\; 2\,e^{\frac{1}{6\times 20}}\sqrt{2\pi\, m\left(1-\frac{m}{z}\right)}.$$

De plus, en utilisant le lemme 7.3 et des manipulations algébriques, nous obtenons pour tout $k \in \mathcal{K}^*_{mz\hat{k}}$,

$$F(k) \;<\; \frac{\gamma}{\sqrt{2\pi}}\sqrt{m\left(1-\frac{m}{z}\right)\left(\frac{1}{k}+\frac{1}{\hat{k}-k}\right)\left(\frac{1}{m-k}+\frac{1}{z-\hat{k}-m+k}\right)},$$

où $\gamma = e^{\frac{1}{12}\left[\frac{1}{k}+\frac{1}{z-\hat{k}}+\frac{1}{m}+\frac{1}{z-m}\right]} \leq e^{\frac{1}{12}\left[2+\frac{2}{20}\right]}$, comme $m \geq 20$ et $z-m \geq 20$.

De plus, le lemme A.17 montre que $\displaystyle\sum_{k\in\mathcal{K}^*_{mz\hat{k}}}\sqrt{\left(\frac{1}{k}+\frac{1}{\hat{k}-k}\right)\left(\frac{1}{m-k}+\frac{1}{z-\hat{k}-m+k}\right)} \;\leq\; 2[1+ln(m)]$.

$$\text{Alors,}\quad \sum_{k\in\mathcal{K}_{mz\hat{k}}} F(k) \;\leq\; C(m)\sqrt{m\left(1-\frac{m}{z}\right)}, \tag{7.17}$$

où $C(m) = 2\,e^{\frac{1}{6\times 20}}\sqrt{2\pi} + \frac{\gamma}{\sqrt{2\pi}}2[1+ln(m)]$. Pour $m \geq 20$, nous avons $C(m) \leq 3\,ln(m)$. Comme l'équation (7.17) est indépendante de $\hat{k}$, le résultat est démontré. □

Du théorème 7.12, nous concluons que $\mathcal{T}^{\mathrm{K}}_{\Delta_{\mathrm{KL}}}(m,z) \leq t(m,z)$. En effet, les équations (7.11) et (7.14) donnent $\Delta_{\mathrm{KL}}(q,p) \leq \Delta^\star_\beta(q,p)$, et alors $\mathcal{T}^{\mathrm{K}}_{\Delta_{\mathrm{KL}}}(m,z) \leq \mathcal{T}^{\mathrm{K}}_{\Delta^\star_\beta}(m,z)$. Par contre, il ne faut pas conclure de cette inégalité que les bornes obtenues par le théorème 7.10 sont plus serrées en utilisant Δ_{KL} plutôt que $\Delta^\star_\beta$ comme Δ-fonction, puisque le choix de Δ impacte les deux côtés de l'inégalité du théorème 7.10.

7.3.4 Nouvelles bornes PAC-bayésiennes transductives explicites

Les prochains résultats présentent deux bornes transductives dérivées des théorèmes 7.10 et 7.12. La borne (a) est la plus serrée, alors que la borne (b) a une forme explicite, c'est-à-dire qu'elle borne directement la quantité d'intérêt par une inégalité.

Corollaire 7.13. *Pour tout ensemble Z de $z \geq 42$ exemples*[10]*, pour tout ensemble $\mathcal{H}$ de classificateurs, pour toute distribution a priori P sur $\mathcal{H}$ et pour tout $\delta \in (0, 1]$, avec probabilité au moins $1 - \delta$ sur le choix d'un ensemble S de m exemples parmi Z (tel que $20 \leq m \leq z-20$), nous avons*

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

$$a)\ \Delta^{\star}_{m/z}\Big(R_S(G_Q), R_Z(G_Q)\Big) \ \leq\ \frac{1}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\frac{t(m,z)}{\delta}\right],$$

$$b)\ R_Z(G_Q) \ \leq\ R_S(G_Q) + \sqrt{\frac{1-\frac{m}{z}}{2m}\left[\mathrm{KL}(Q \,\|\, P) + ln\frac{t(m,z)}{\delta}\right]},$$

où $t(m, N)$ est défini à l'équation (7.16).

Démonstration. La borne (a) est obtenue du théorème 7.10, avec $\Delta(q, p) = \Delta^{\star}_{m/z}(q, p)$, et du théorème 7.12. De la borne (a), en utilisant l'équation (7.14) et l'inégalité de Pinsker ($\Delta_{\mathrm{KL}}(q, p) \geq 2(q-p)^2$) deux fois, nous avons

$$\begin{aligned}
&\Delta^{\star}_{m/z}\Big(R_S(G_Q), R_Z(G_Q)\Big) \\
&\quad\geq\ 2\Big(R_S(G_Q) - R_Z(G_Q)\Big)^2 + 2\left(\frac{z}{m} - 1\right)\left(\frac{R_Z(G_Q) - \frac{m}{z}R_S(G_Q)}{1-\frac{m}{z}} - R_Z(G_Q)\right)^2 \qquad (7.18)\\
&\quad=\ \frac{2\Big(R_S(G_Q) - R_Z(G_Q)\Big)^2}{1-\frac{m}{z}},
\end{aligned}$$

et la borne (b) est obtenue en isolant $R_Z(G_Q)$ dans l'expression

$$2\Big(R_S(G_Q) - R_Z(G_Q)\Big)^2 \ \leq\ \left(1 - \frac{m}{z}\right)\frac{1}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\frac{t(m,z)}{\delta}\right].$$

□

Les deux bornes présentées par le corollaire 7.13 sont analogues aux deux bornes inductives du corollaire 7.4. De plus, ces bornes transductives convergent vers leur analogue inductif lorsque $z \to \infty$ (à un petit facteur près, comme le terme $ln(t(m, z))$ tend vers $ln(3\, ln(m)\sqrt{m})$ au lieu de $ln(2\sqrt{m})$, mais l'effet est petit étant donné que cette expression est divisée par m). Plus précisément, la borne (a) peut être vue comme une généralisation transductive de la borne inductive de Seeger (2002). En effet, tel que montré à l'équation (7.14),

$$\Delta^{\star}_{m/z}\Big(R_S(G_Q), R_Z(G_Q)\Big) \ \simeq\ \Delta_{\mathrm{KL}}\Big(R_S(G_Q), R_Z(G_Q)\Big)$$

lorsque $z \gg m$ (c'est-à-dire, lorsque le ratio $\frac{m}{z}$ tend vers 0). Également, la borne (b) généralise la borne PAC-bayésienne de McAllester (2003a), comme le facteur multiplicatif $\frac{1-\frac{m}{z}}{2m}$ est réduit à $\frac{1}{2m}$ lorsque $z \gg m$.

10. Ce théorème est également valide pour $z \geq 40$. Ceci dit, le nombre 42 nous semble plus approprié, car il est la réponse à la grande question sur la vie, l'univers et le reste (Adams, 1979).

Nous dérivons maintenant deux bornes PAC-bayésiennes explicites sans divergence KL, en utilisant le théorème 7.11. Pour pouvoir utiliser celui-ci avec une certaine Δ-fonction, celle-ci doit satisfaire $\Delta(q,p) = \Delta(1-q,1-p)$. Le lemme suivant montre que $\Delta^\star_\beta$ a bel et bien cette propriété.

Lemme 7.14. *Soit $\Delta^\star_\beta$, la Δ-fonction définie à l'équation (7.13). Cette Δ-fonction a la propriété suivante :*

$$\Delta^\star_\beta(q,p) = \Delta^\star_\beta(1-q,1-p)\,.$$

Démonstration. Rappelons premièrement que $\Delta_{\text{KL}}(1-q,1-p) = \Delta_{\text{KL}}(q,p)$ (voir équation (3.10)). Utilisons la réécriture de $\Delta^\star_\beta$ de l'équation (7.14). Nous avons

$$\begin{aligned}
\Delta^\star_\beta(1-q,1-p) &= \Delta_{\text{KL}}(1-q,1-p) + \tfrac{1-\beta}{\beta}\Delta_{\text{KL}}\Big(\tfrac{(1-p)-\beta(1-q)}{1-\beta}, 1-p\Big) \\
&= \Delta_{\text{KL}}(q,p) + \tfrac{1-\beta}{\beta}\Delta_{\text{KL}}\Big(\tfrac{1-p-\beta+\beta q}{1-\beta}, 1-p\Big) \\
&= \Delta_{\text{KL}}(q,p) + \tfrac{1-\beta}{\beta}\Delta_{\text{KL}}\Big(\tfrac{(1-\beta)-(p-\beta q)}{1-\beta}, 1-p\Big) \\
&= \Delta_{\text{KL}}(q,p) + \tfrac{1-\beta}{\beta}\Delta_{\text{KL}}\Big(1-\tfrac{(p-\beta q)}{1-\beta}, 1-p\Big) \\
&= \Delta_{\text{KL}}(q,p) + \tfrac{1-\beta}{\beta}\Delta_{\text{KL}}\Big(\tfrac{(p-\beta q)}{1-\beta}, p\Big) \\
&= \Delta^\star_\beta(q,p)\,,
\end{aligned}$$

tel que voulu. □

Le prochain résultat présente deux bornes transductives sans divergence KL, analogues aux deux bornes transductives du corollaire 7.13. Encore une fois, la borne (a) est plus serrée, alors que la borne (b) est plus explicite.

Corollaire 7.15. *Pour tout ensemble Z de $z \geq 42$ exemples, pour tout ensemble symétrique $\mathcal{H}$ de classificateurs, pour toute distribution a priori P sur $\mathcal{H}$ et pour tout $\delta \in (0,1]$, avec probabilité au moins $1-\delta$ sur le choix d'un ensemble S de m exemples parmi Z (tel que $20 \leq m \leq z-20$), nous avons*

$$\begin{aligned}
&\forall Q \text{ aligné sur } P\text{:} \\
&\quad a)\, \Delta^\star_{m/z}\Big(R_S(G_Q),\, R_Z(G_Q)\Big) \;\leq\; \frac{1}{m}\left[ln\frac{t(m,z)}{\delta}\right], \\
&\quad b)\, R_Z(G_Q) \;\leq\; R_S(G_Q) + \sqrt{\frac{1-\frac{m}{z}}{2m}\left[ln\frac{t(m,z)}{\delta}\right]},
\end{aligned}$$

où $t(m,N)$ est défini à l'équation (7.16).

Démonstration. La démonstration suit les mêmes étapes que la démonstration du corollaire 7.13, en remplaçant l'utilisation du théorème 7.10 par le théorème 7.11, et en utilisant le lemme 7.14. □

7.3.5 Relation avec l'état de l'art

Les bornes transductives présentées dans ce chapitre peuvent être vues comme une amélioration des travaux de Derbeko, El-Yaniv et Meir (2004). Premièrement, notons que nous avons relevé une erreur dans la preuve du théorème 18 de Derbeko, El-Yaniv et Meir (2004) rendant leur borne invalide. À la fin de la démonstration, pour obtenir une borne explicite, les auteurs bornent la divergence $\Delta^{\star}_{m/z}\big(R_S(G_Q), R_Z(G_Q)\big)$, en appliquant l'inégalité $\Delta_{\mathrm{KL}}(q,p) \geq \frac{(q-p)^2}{2p}$ deux fois (voir l'équation (17) de Derbeko, El-Yaniv et Meir (2004)). Par contre, tel qu'indiqué dans McAllester (2003b), cette inégalité ne tient que lorsque $q < p$, et ainsi ne peut pas être appliquée sur le terme $\Delta_{\mathrm{KL}}\big(\frac{R_Z(G_Q)-\frac{m}{z}R_S(G_Q)}{1-m/z}, R_Z(G_Q)\big)$ quand $R_S(G_Q) < R_Z(G_Q)$, car nous avons nécessairement $\frac{R_Z(G_Q)-\frac{m}{z}R_S(G_Q)}{1-m/z} \geq R_Z(G_Q)$. Cette erreur peut être corrigée en utilisant plutôt l'inégalité de Pinsker, comme nous l'avons fait à l'équation (7.18). Nous présentons une version corrigée ainsi qu'une preuve détaillée en annexe A, au théorème A.18. Le théorème corrigé indique qu'une borne supérieure sur $R_Z(G_Q)$ est donnée par

$$R_S(G_Q) + \sqrt{\frac{1-\frac{m}{z}}{2(m-1)}\left[\mathrm{KL}(Q \,\|\, P) + ln\,\frac{m}{\delta} + 7\,ln(z+1)\right]}. \tag{7.19}$$

La différence majeure entre la borne (b) du corollaire 7.13 et l'expression de l'équation (7.19) est que le terme de complexité $ln(m) + 7\,ln(z+1)$ de celle-ci remplace le terme $ln(t(m,z))$.

Notre résultat mène donc à des bornes beaucoup plus serrées. En effet, nous obtenons déjà une borne plus serrée en bornant lâchement $ln(t(m,z))$ par $ln((m+1)(z+1))$, en utilisant l'équation (7.15) plutôt qu'en utilisant l'inégalité bien plus serrée de l'équation (7.16) du théorème 7.12. Le problème majeur de l'expression de Derbeko, El-Yaniv et Meir (2004) est que la valeur de la borne (autant dans la version originale que dans la version corrigée) diverge vers l'infini lorsque z grandit, sauf si le nombre d'exemples étiquetés m tend lui aussi vers l'infini. Ce comportement pour une borne transductive est bien sûr à éviter. Tel que discuté dans la section 7.3.4, nos deux bornes présentées au corollaire 7.13 convergent vers leur analogue inductif lorsque le ratio $\frac{m}{z}$ tend vers 0.

7.3.6 Bornes transductives sur le risque du classificateur par vote de majorité

Utilisons maintenant les bornes (b) des corollaires 7.13 et 7.15 pour borner le risque du classificateur par vote de majorité dans le cadre transductif.

Nous énonçons d'abord deux PAC-bornes n'utilisant que le risque du classificateur de Gibbs ou le premier moment de la marge, qui sont les homologues transductives de la PAC-borne 3.8, dont la première est simultanément valide pour toute distribution a posteriori Q, et la seconde est spécialisée aux distributions Q alignées sur la distribution a priori P.

PAC-borne 7.16. *Pour tout ensemble Z de $z \geq 42$ exemples, pour tout ensemble $\mathcal{H}$ de classificateurs $\mathcal{X} \to \{-1,1\}$, pour toute distribution a priori P sur $\mathcal{H}$ et tout $\delta \in (0,1]$, nous avons avec probabilité*

au moins $1-\delta$ *sur le choix d'un ensemble* S *de* m *exemples parmi* Z *(tel que* $20 \leq m \leq z-20$*), que pour toute distribution a posteriori* Q *sur* $\mathcal{H}$*,*

$$R_Z(B_Q) \;\leq\; 2\cdot\bar{r} \;=\; 1-\underline{\mu_1}\,,$$

où

$$\bar{r} \;\triangleq\; min\left(\tfrac{1}{2},\, R_S(G_Q)+\sqrt{\frac{1-\frac{m}{z}}{2m}\left[\mathrm{KL}(Q\,\|\,P)+ln\frac{t(m,z)}{\delta}\right]}\right),$$
$$\underline{\mu_1} \;\triangleq\; max\left(0,\, \mu_1\big(M_Q^S\big)-\sqrt{\frac{2\left(1-\frac{m}{z}\right)}{m}\left[\mathrm{KL}(Q\,\|\,P)+ln\frac{t(m,z)}{\delta}\right]}\right),$$

et où $t(m,z)$ *est défini à l'équation* (7.16)*.*

Démonstration. L'inégalité est une conséquence directe de la proposition 2.12 et du corollaire 7.13-(b). L'égalité est une application directe de l'équation (2.8). □

PAC-borne 7.17. *Pour tout ensemble* Z *de* $z \geq 42$ *exemples, pour tout ensemble symétrique* $\mathcal{H}$ *de classificateurs* $\mathcal{X}\to\{-1,1\}$*, pour toute distribution a priori* P *sur* $\mathcal{H}$ *et tout* $\delta\in(0,1]$*, nous avons avec probabilité au moins* $1-\delta$ *sur le choix d'un ensemble* S *de* m *exemples parmi* Z *(tel que* $20\leq m\leq z-20$*), que pour toute distribution a posteriori* Q *alignée sur* P*,*

$$R_Z(B_Q) \;\leq\; 2\cdot\bar{r} \;=\; 1-\underline{\mu_1}\,,$$

où

$$\bar{r} \;\triangleq\; min\left(\tfrac{1}{2},\, R_S(G_Q)+\sqrt{\frac{1-\frac{m}{z}}{2m}\left[ln\frac{t(m,z)}{\delta}\right]}\right),$$
$$\underline{\mu_1} \;\triangleq\; max\left(0,\, \mu_1\big(M_Q^S\big)-\sqrt{\frac{2\left(1-\frac{m}{z}\right)}{m}\left[ln\frac{t(m,z)}{\delta}\right]}\right),$$

et où $t(m,z)$ *est défini à l'équation* (7.16)*.*

Démonstration. L'inégalité est une conséquence directe de la proposition 2.12 et du corollaire 7.15-(b). L'égalité est une application directe de l'équation (2.8). □

Utilisons maintenant la C-borne du théorème 2.13 pour borner le risque du classificateur par vote de majorité dans le cadre transductif. Dans ce cadre, la C-borne est calculée sur l'échantillon complet Z, par

$$C_Q^Z \;=\; 1-\frac{\big(1-2\cdot R_Z(G_Q)\big)^2}{1-2\cdot d_Q^Z}\,. \tag{7.20}$$

Ici, comme nous travaillons avec des classificateurs binaires et comme le désaccord espéré (définition 2.8) et le deuxième moment de la marge sont calculés sur l'ensemble complet Z, nous avons

$$d_Q^Z \;=\; \frac{1}{2}\left(1 - \mathop{\mathbf{E}}_{x\sim Z_{\mathcal{X}}}\left[\mathop{\mathbf{E}}_{h\sim Q} h(x)\right]^2\right) \qquad \text{et} \qquad \mu_2\big(M_Q^Z\big) \;=\; \mathop{\mathbf{E}}_{x\sim Z_{\mathcal{X}}}\left[\mathop{\mathbf{E}}_{h\sim Q} h(x)\right]^2 ,$$

où $Z_{\mathcal{X}}$ est la marginale en $\mathcal{X}$ de la distribution uniforme sur Z. L'ensemble $Z_{\mathcal{X}}$ étant disponible à l'algorithme d'apprentissage dans le cadre transductif, il est possible de calculer le « vrai » désaccord (et le « vrai » deuxième moment de la marge) dans ce contexte, éliminant ainsi le besoin de borner le désaccord espéré. Une borne supérieure sur $R_Z\big(G_Q\big)$ peut être directement convertie en une borne supérieure sur C_Q^Z, et ainsi sur $R_Z\big(B_Q\big)$. En utilisant le désaccord espéré sur l'échantillon complet, ces bornes extraient plus d'information à propos du problème d'apprentissage. La C-borne devient un outil performant pour dériver des bornes transductives sur le risque du classificateur par vote de majorité.

Nous dérivons maintenant deux PAC-bornes transductives. La première est valide simultanément pour toute distribution a posteriori Q et découle du corollaire 7.13. La seconde est restreinte aux distributions Q alignées sur P, ne comporte pas de terme $\mathrm{KL}(Q \,\|\, P)$ et découle du corollaire 7.15.

PAC-borne 7.18. *Pour tout ensemble Z de $z \geq 42$ exemples, pour tout ensemble $\mathcal{H}$ de classificateurs $\mathcal{X} \to \{-1, 1\}$, pour toute distribution a priori P sur $\mathcal{H}$ et tout $\delta \in (0, 1]$, nous avons avec probabilité au moins $1 - \delta$ sur le choix d'un ensemble S de m exemples parmi Z (tel que $20 \leq m \leq z-20$), que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$R_Z\big(B_Q\big) \;\leq\; 1 - \frac{\big(1 - 2\cdot\bar{r}\,\big)^2}{1 - 2\cdot d_Q^Z} \;=\; 1 - \frac{\Big(\underline{\mu_1}\Big)^2}{\mu_2\big(M_Q^Z\big)} ,$$

où

$$\bar{r} \;\triangleq\; min\left(\tfrac{1}{2},\, R_S\big(G_Q\big) + \sqrt{\frac{1 - \frac{m}{z}}{2m}\left[\mathrm{KL}(Q \,\|\, P) + ln\frac{t(m,z)}{\delta}\right]}\right) ,$$

$$\underline{\mu_1} \;\triangleq\; max\left(0,\, \mu_1\big(M_Q^S\big) - \sqrt{\frac{2\left(1 - \frac{m}{z}\right)}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\frac{t(m,z)}{\delta}\right]}\right) ,$$

et où $t(m, z)$ est défini à l'équation (7.16).

Démonstration. L'inégalité est une conséquence directe du théorème 2.13 et du corollaire 7.13-(b). L'égalité est une application directe des équations (2.8) et (2.10). □

PAC-borne 7.19. *Pour tout ensemble Z de $z \geq 42$ exemples, pour tout ensemble symétrique $\mathcal{H}$ de classificateurs $\mathcal{X} \to \{-1,1\}$, pour toute distribution a priori P sur $\mathcal{H}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble S de m exemples parmi Z (tel que $20 \leq m \leq z-20$), que pour toute distribution a posteriori Q alignée sur P,*

$$R_Z(B_Q) \;\leq\; 1 - \frac{\left(1-2\cdot\bar{r}\right)^2}{1-2\cdot d_Q^Z} \;=\; 1 - \frac{\left(\underline{\mu_1}\right)^2}{\mu_2\left(M_Q^Z\right)},$$

où

$$\bar{r} \triangleq min\left(\tfrac{1}{2},\, R_S(G_Q) + \sqrt{\frac{1-\frac{m}{z}}{2m}\left[ln\frac{t(m,z)}{\delta}\right]}\right),$$

$$\underline{\mu_1} \triangleq max\left(0,\, \mu_1\left(M_Q^S\right) - \sqrt{\frac{2\left(1-\frac{m}{z}\right)}{m}\left[ln\frac{t(m,z)}{\delta}\right]}\right),$$

et où $t(m,z)$ est défini à l'équation (7.16).

Démonstration. L'inégalité est une conséquence directe du théorème 2.13 et du corollaire 7.15-(b). L'égalité est une application directe des équations (2.8) et (2.10). □

Nous avons présenté un théorème PAC-bayésien général qui permet d'utiliser n'importe quelle fonction convexe Δ du risque de l'ensemble d'entraînement et du risque de l'échantillon complet. Chaque choix de Δ-fonction mène à une nouvelle borne transductive, et toutes ces bornes peuvent être calculées à un certain prix computationnel lorsque z est grand. À partir d'une nouvelle Δ-fonction appropriée, la $\Delta_\beta^\star$-fonction de l'équation (7.13), nous avons dérivé une borne explicite plus serrée que les bornes transductives PAC-bayésiennes de l'état de l'art. Nous avons également énoncé de nouvelles PAC-bornes transductives, dont celles basées sur la C-borne utilisent le fait que le désaccord espéré de l'échantillon complet peut être calculé sans les étiquettes. Comme nous l'avons fait au chapitre 3, nous avons également proposé une nouvelle borne PAC-bayésienne sur la C-borne ne dépendant pas de la divergence KL. Cette PAC-borne transductive ouvre la porte à une version de MinCq (et conséquemment, une version de CqBoost) adaptée à l'apprentissage transductif. Notons que nous n'avons pas généralisé les résultats de ce chapitre au cadre de la compression d'échantillon, ce qui permettrait entre autres l'utilisation comme votants des fonctions définies sur les exemples de l'ensemble d'entraînement. Ceci dit, cette généralisation est directe, en suivant la même méthodologie qu'à la section 3.6. Notons par contre que les bornes de ce chapitre ne sont valides que pour les ensembles de votants qui sont des classificateurs binaires, c'est-à-dire qui retournent -1 ou $+1$. La généralisation aux votants à valeur réelle est ici plus complexe, puisque nous n'avons pas d'équivalent du lemme A.9 pour la distribution hypergéométrique apparaissant dans le cadre transductif.

Dans la prochaine section, nous faisons une étude empirique comparant les valeurs de nos bornes transductives avec les valeurs des bornes de l'état de l'art et les bornes inductives.

7.3.7 Étude empirique

Nous menons ici trois expérimentations. D'abord, nous illustrons brièvement l'effet du changement de paradigme du cadre inductif vers le cadre transductif sur les valeurs des bornes, afin de valider que celles-ci sont bel et bien en mesure de tirer profit des connaissances supplémentaires sur le problème à résoudre. Nous explorons ensuite quel est l'impact du choix d'une Δ-fonction sur la valeur de la borne du théorème 7.10. Finalement, nous montrons les valeurs de bornes sur des ensembles de données naturelles.

Comparaison des bornes inductives et transductives

Dans cette première expérimentation, nous comparons les bornes inductives et transductives en nous basant sur l'expérimentation de la section 3.4.3, afin d'illustrer le gain potentiel que peuvent avoir les bornes transductives en considérant les informations supplémentaires à disposition.

Comme le cadre inductif et le cadre transductif sont des paradigmes différents et ainsi les bornes ne peuvent pas être directement comparées, nous devons considérer la même supposition (fausse en pratique) qu'à l'étude empirique de la section 7.2.3 : nous considérons une distribution D synthétique pour le cadre inductif, qui ici correspond à l'échantillon complet Z du cadre inductif. Nous considérons que chaque exemple de l'ensemble d'entraînement S est tiré aléatoirement *sans remise* de cet ensemble de données. Les bornes inductives et transductives sont alors calculées sur ces ensembles.[11]

La figure 7.8 présente des résultats numériques obtenus en exécutant l'algorithme AdaBoost (Freund et Schapire, 1997) avec des souches de décision comme votants, sur l'ensemble de données *mushroom* provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (Lichman, 2013). La distribution D dans le cadre inductif et l'échantillon complet Z dans le cadre transductif correspondent aux 8124 exemples de l'ensemble de données. Parmi ces exemples, un ensemble d'entraînement S de 4062 exemples est tiré sans remise. Pour chaque itération de boosting nous fournissant une distribution Q, nous calculons les valeurs des PAC-bornes 3.8, 3.13, 7.16 et 7.18, c'est-à-dire les deux bornes inductives également présentées à la figure 3.2 et leur équivalent transductif. La distribution P utilisée dans le calcul des bornes est une distribution uniforme sur l'ensemble des votants.

11. Notons que dans ce contexte, les bornes inductives qui sont calculées sont valides même si le tirage a été fait sans remise. En effet, lorsqu'on considère un ensemble de données fini, l'espérance de toute fonction convexe appliquée au tirage sans remise (cadre transductif) est borné supérieurement par l'espérance de la même fonction convexe appliquée au tirage avec remise (cadre inductif) (Hoeffding, 1963 ; Tolstikhin, Blanchard et Kloft, 2014 ; Bardenet et Maillard, 2015). Cette propriété pour les tirages dans un ensemble fini rend valide les bornes inductives avec un tirage sans remise.

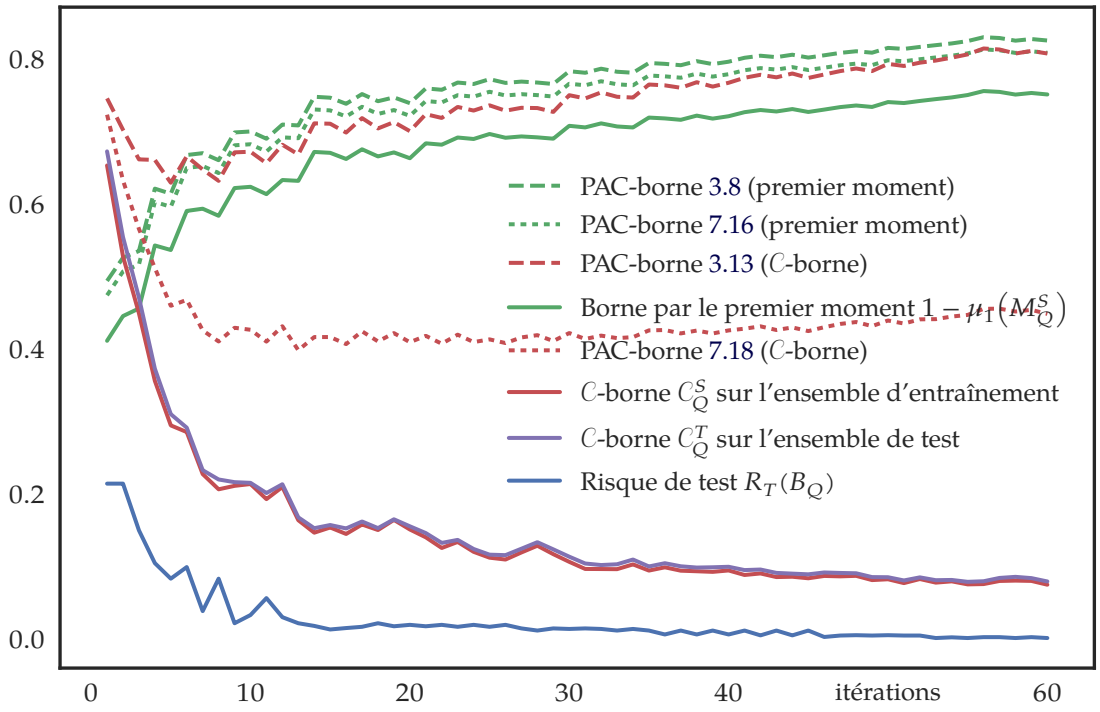

FIG. 7.8: Comparaison des bornes inductives sur $R_D(B_Q)$ et transductives sur $R_Z(B_Q)$ pendant 60 itérations de boosting.

Nous pouvons voir que la PAC-borne transductive 7.16 n'utilisant que le premier moment de la marge offre un léger gain par rapport à la PAC-borne inductive 3.8. Rappelons que la PAC-borne transductive utilise une Δ-fonction différente spécialisée au cadre transductif, et prend en considération le nombre d'exemples z de l'échantillon complet. Nous observons par contre un très large gain lors de l'utilisation de la PAC-borne transductive 7.18 par rapport à la PAC-borne inductive 3.13 utilisant la C-borne. Ce gain est dû à la possibilité d'utiliser tous les exemples pour calculer le « vrai » second moment de la marge, alors que dans le cadre inductif il faut l'estimer en le bornant supérieurement par une borne PAC-bayésienne. L'utilisation de la C-borne dans le cadre transductif est donc d'autant plus intéressante.

Utilisation de différentes Δ-fonctions

Un aspect intéressant du théorème 7.10 est la possibilité d'utiliser toute Δ-fonction pour calculer une borne PAC-bayésienne transductive. Le choix d'une Δ-fonction mène à des bornes plus ou moins serrées. En plus du choix de la Δ-fonction, la valeur de ces bornes dépendent du risque de Gibbs sur l'ensemble d'entraînement $R_S(G_Q)$, de la valeur de la divergence entre les distributions a posteriori et a priori $\mathrm{KL}(Q \,\|\, P)$, des tailles m et z de l'ensemble d'entraînement et de l'échantillon complet, puis de la valeur de confiance δ.

Ici, nous expérimentons cinq Δ-fonctions. Les deux premières ont été présentées plus haut. Il s'agit de la divergence KL entre deux distributions de Bernoulli de l'équation (3.2), et la $\Delta^{\star}_{\beta}$-fonction de l'équation (7.13). Nous avons expérimenté avec plusieurs autres candidats de Δ-fonctions, mais présentons ici celles qui ont le comportement le plus intéressant. Il s'agit de la *distance de variation* Δ_V, de la *distance quadratique* Δ_{V^2}, et de la *discrimination triangulaire* Δ_T, d'abord présentées dans la table récapitulative 7.2, et définies par

$$\Delta_V(q,p) \triangleq 2\,|q-p|\,, \qquad \Delta_{V^2}(q,p) \triangleq 2\,(q-p)^2\,, \qquad \Delta_T(q,p) \triangleq \frac{(q-p)^2}{q+p} + \frac{(q-p)^2}{2-q-p}\,.$$

Notons que ces trois divergences sont bien connues dans la littérature (par exemple, voir Topsøe (2000)). Notons également que la distance quadratique mène à la borne PAC-bayésienne de McAllester (2003a), par l'inégalité de Pinsker $\Delta_{\mathrm{KL}}(q,p) \geq \Delta_{V^2}(q,p)$.

La figure 7.9 compare les Δ-fonctions mentionnées ci-haut (Δ_{KL}, $\Delta^{\star}_{m/z}$, Δ_{V^2}, Δ_V et Δ_T), avec un risque de Gibbs empirique fixé à $R_S(G_Q) = 0.2$, une divergence $\mathrm{KL}(Q \,\|\, P)$ fixée à 5, et un niveau de confiance δ fixé à 0.05. Nous faisons ensuite varier la taille z de l'échantillon complet et le ratio $\frac{m}{z}$. Plus précisément, nous considérons 9 paires possibles (z,m), où $z \in \{200, 500, 5000\}$ et $m \in \left\{\frac{1}{10}z, \frac{1}{2}z, \frac{9}{10}z\right\}$. Pour chaque paramètre, une borne supérieure du risque de Gibbs sur l'échantillon complet $R_Z(G_Q)$ est calculée à l'aide du théorème 7.10, en trouvant la valeur de $r \geq R_S(G_Q)$ telle que

$$\Delta\Big(R_S(G_Q), r\Big) = \frac{1}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\frac{\overset{\frown}{\mathcal{I}}^{\mathrm{K}}_{\Delta}(m,z)}{\delta}\right], \tag{7.21}$$

où la valeur exacte de $\overset{\frown}{\mathcal{I}}^{\mathrm{K}}_{\Delta}(m,z)$ est calculée par l'équation (7.11) pour toute Δ-fonction. La figure 7.9 illustre que lorsque la Δ-fonction est choisie, la borne sur le risque dépend d'un compromis entre le taux de croissance de $\Delta\Big(R_S(G_Q), r\Big)$ et la valeur de la partie de droite de l'équation (7.21), qui elle est influencée par l'amplitude de $\overset{\frown}{\mathcal{I}}^{\mathrm{K}}_{\Delta}(m,z)$. Les figures 7.10 et 7.11 montrent des résultats similaires à la figure 7.9, mais cette fois-ci avec des risques $R_S(G_Q) = 0.1$ et $R_S(G_Q) = 0.01$.

Nous constatons que la divergence Δ_{KL}, donnant généralement les meilleures bornes PAC-bayésiennes connues dans le cadre inductif, est souvent moins serrée que les autres Δ-fonctions dans le cadre transductif. Les bornes données par Δ_{KL}, Δ_{V^2} et Δ_T sont similaires sur les figures 7.9, 7.10 et 7.11. La distance de variation Δ_V donne les bornes les plus serrées sur de petits ensembles de données, mais perd son avantage lorsque z grandit. La fonction $\Delta^{\star}_{m/z}$ montre de très bonnes performances lorsque le ratio m/z grandit. Ce phénomène est relié au fait que $\Delta^{\star}_{m/z}$ est la seule Δ-fonction qui s'ajuste à ce ratio. Ainsi, sa valeur est toujours dans l'intervalle des risques réalisables. En effet, nous pouvons déduire de l'équation (7.10), pourvu que $0 \leq R_U(G_Q) \leq 1$, nous avons :

$$\frac{m}{z}R_S(G_Q) \;\leq\; R_Z(G_Q) \;\leq\; \frac{m}{z}R_S(G_Q) + \frac{z-m}{z}. \tag{7.22}$$

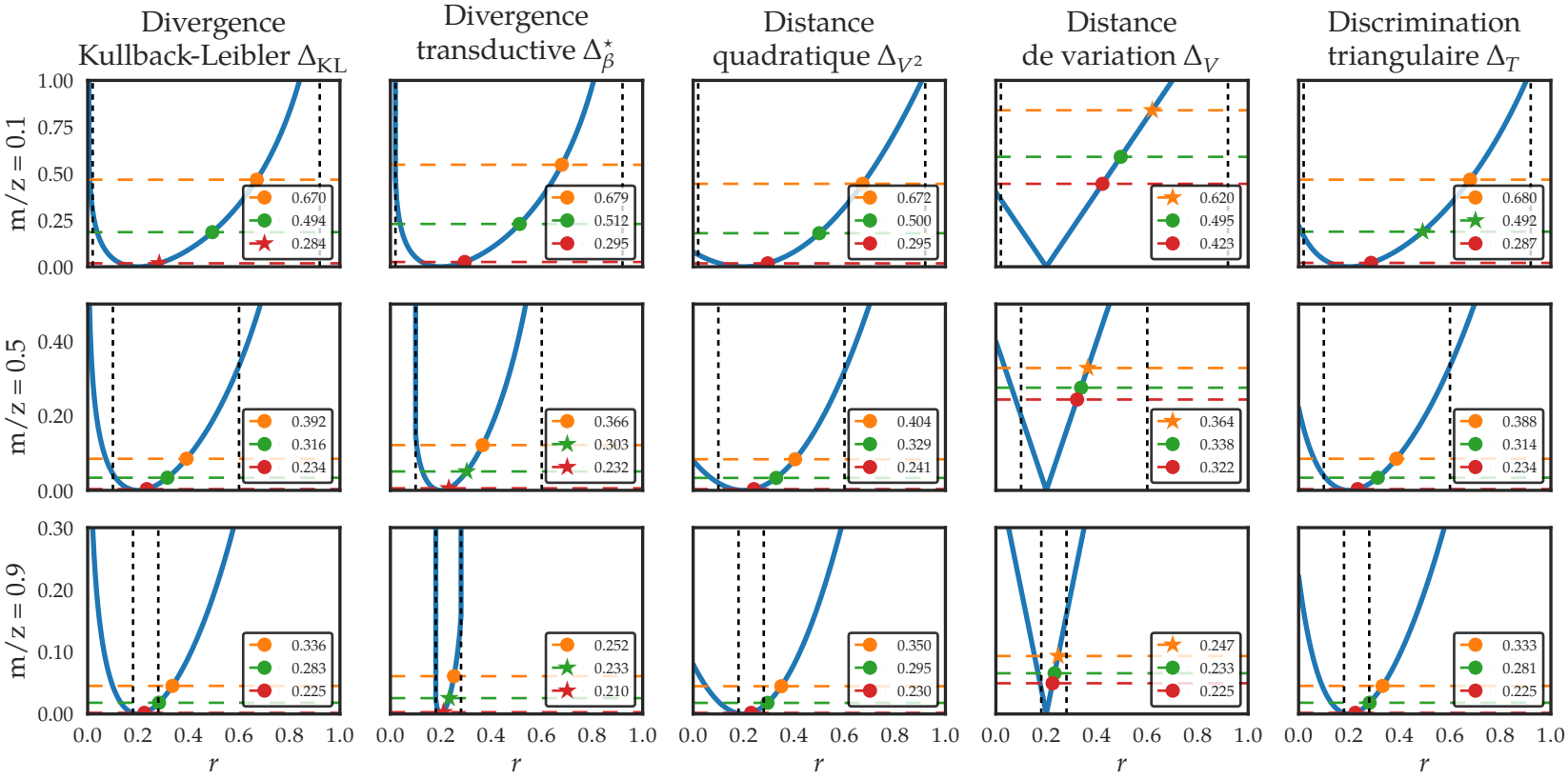

FIG. 7.9: Étude du comportement des bornes obtenues par le théorème 7.10. Tous les graphiques considèrent $R_S(G_Q) = 0.2$, $\mathrm{KL}(Q \,\|\, P) = 5$ et $\delta = 0.05$. Chaque colonne partage une même Δ-fonction. Chaque ligne partage un même ratio $\frac{m}{z}$. Un graphique peut être interprété comme suit. Les deux lignes verticales montrent la valeur minimale et la valeur maximale de $R_Z(G_Q)$ (voir l'équation (7.22)). La courbe bleue correspond à la fonction $\Delta(0.2, r)$. Chaque ligne pointillée horizontale correspond à la valeur donnée par $\frac{1}{m}\Big[\mathrm{KL}(Q \,\|\, P) + ln\left(\overline{\jmath}_{\Delta}^{\mathrm{K}}(m,z)/\delta\right)\Big]$ pour trois valeurs de z: $z = 200$ (ligne verte), $z = 500$ (ligne rouge), et $z = 5000$ (ligne violette). Pour chacune de ces lignes, la position du point montre la valeur de la borne (c'est-à-dire, le r résolvant l'équation (7.21)). Finalement, la valeur de la borne est rapportée dans la légende. Une étoile remplace le point si la borne est la plus serrée parmi toutes les Δ-fonctions.

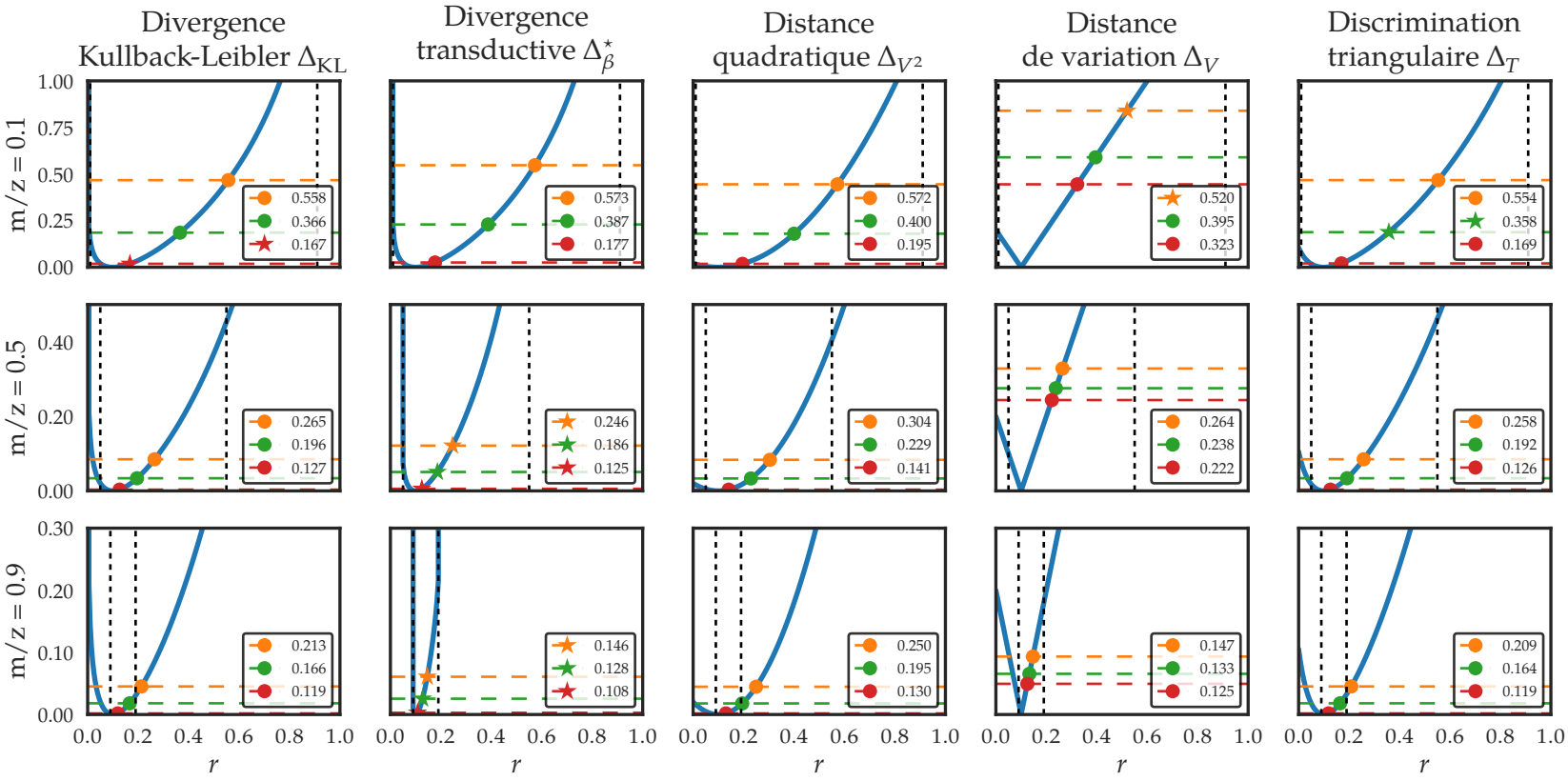

FIG. 7.10: Étude du comportement des bornes obtenues par le théorème 7.10, avec $R_S(G_Q) = 0.1$, $\mathrm{KL}(Q \,\|\, P) = 5$ et $\delta = 0.05$.

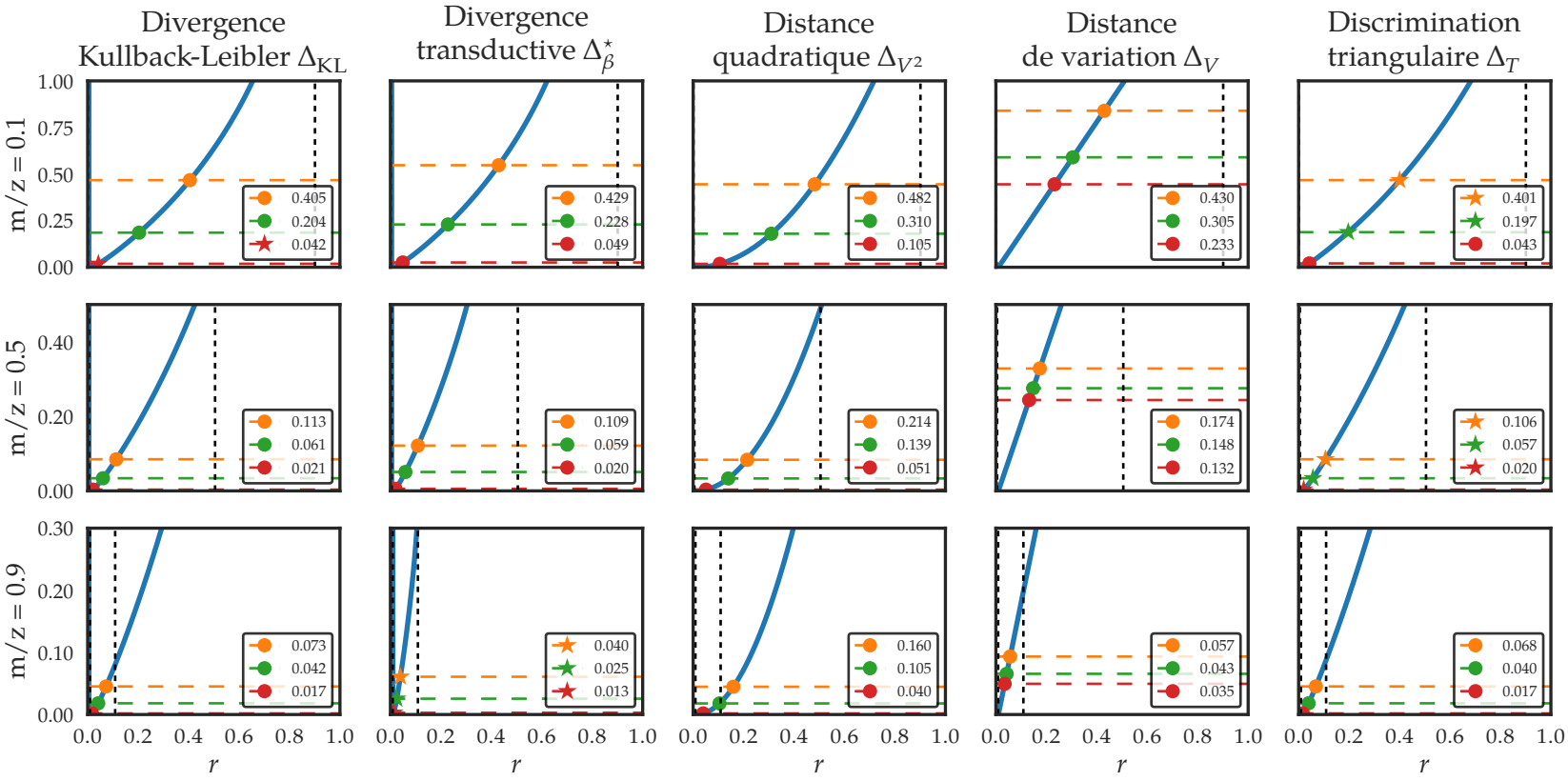

FIG. 7.11: Étude du comportement des bornes obtenues par le théorème 7.10, avec $R_S(G_Q) = 0.01$, $\mathrm{KL}(Q \parallel P) = 5$ et $\delta = 0.05$.

Valeurs de bornes sur données naturelles

Nous comparons maintenant les valeurs de bornes sur des ensembles de données raisonnablement grands, provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (Lichman, 2013). Pour chaque ensemble de données de z exemples, nous tirons aléatoirement (sans remise) un ensemble de données de m exemples, avec des ratios m/z de 0.1 et 0.5. Pour obtenir une distribution a posteriori Q sur un ensemble de classificateurs, nous exécutons l'algorithme AdaBoost en utilisant des souches de décision, pendant 200 itérations.

Dans la table 7.3, pour chaque ensemble de données et chaque ratio m/z, nous calculons le risque de Gibbs sur l'échantillon complet Z et l'ensemble d'entraînement S. Nous calculons les bornes explicites du corollaire 7.13-(b) et de l'équation (7.19) (nommée Derbeko dans la table). Nous calculons également les bornes du théorème 7.10, en utilisant $\Delta = \Delta_{\mathrm{KL}}$ et $\Delta = \Delta^{\star}_{m/z}$, comme ce sont les deux choix les plus intéressants dans ce cadre (voir la discussion de la section 7.3.7). Finalement, nous calculons le risque du classificateur par vote de majorité sur S et Z, la valeur de la PAC-borne 7.16 n'utilisant que le risque du classificateur de Gibbs, et la valeur de la PAC-borne 7.18, utilisant également le désaccord espéré. Toutes les bornes ont été calculées avec un taux de confiance $\delta = 0.05$.

La table 7.3 confirme que la borne explicite du corollaire 7.13-(b) est plus serrée que la borne de Derbeko, El-Yaniv et Meir (2004). Elle corrobore également les observations de la figure 7.9: la borne du théorème 7.10 avec $\Delta = \Delta_{\mathrm{KL}}$ performe bien lorsque m/z est petit, alors que $\Delta = \Delta^{\star}_{m/z}$ performe bien lorsque m/z est grand. Finalement, nous voyons que lorsque le risque de Gibbs est plus petit que 0.5, la borne sur le risque du classificateur par vote de majorité utilisant la C-borne transductive est beaucoup plus serrée que celle utilisant le risque de Gibbs.

7.4 Spécialisation de MinCq à l'apprentissage transductif

Dans cette section, nous nous basons sur la théorie PAC-bayésienne transductive présentée cihaut pour construire une extension de l'algorithme MinCq spécialisée au cadre de l'apprentissage transductif. Cette extension, nommée *TMinCq*, minimise simplement la PAC-borne 7.19, qui dépend d'une borne PAC-bayésienne sur le premier moment de la marge, mais dont le second moment peut être calculé de manière exacte sur l'ensemble Z.

7.4.1 TMinCq : minimiser la borne PAC-bayésienne transductive

La PAC-borne transductive 7.19 et les discussions du chapitre 4 nous suggèrent la stratégie d'apprentissage suivante : étant donné un ensemble symétrique $\mathcal{H}$ de classificateurs binaires, trouver une distribution Q quasi-uniforme qui minimise le « vrai » second moment de la marge (calculé sur l'ensemble Z), tout en contraignant le premier moment de la marge empirique à être égal à une valeur μ, un hyperparamètre de l'algorithme.

Info. sur les données			Classif. de Gibbs						Classif. par vote de majorité			
Nom	z	m/z	$R_S(G_Q)$	$R_Z(G_Q)$	C.7.13-b	Derbeko	T.7.10-Δ_{KL}	T.7.10-$\Delta^{\star}_{m/z}$	$R_S(B_Q)$	$R_Z(B_Q)$	T.7.16	T.7.18
car	1728	0.1	0.193	0.194	**0.555**	0.793	**0.527**	0.546	0.105	0.159	1.092	-
car	1728	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
letter-ab	1555	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
mushroom	8124	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**
nursery	12959	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	-
optdigits	3823	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
pageblock	5473	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
pendigits	7494	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	-
segment	2310	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	-
spambase	4601	0.5	0.225	0.226	**0.488**	0.539	0.489	**0.486**	0.137	0.143	0.973	**0.961**

TAB. 7.3: Comparaison des bornes transductives sur le risque du classificateur de Gibbs et sur le classificateur par vote de majorité.

Définition 7.20 (TMinCq)**.** Étant donné un ensemble symétrique $\mathcal{H}$ de classificateurs, un échantillon complet $Z = \langle (x_1, y_1), \ldots, (x_z, y_z) \rangle$ de plus de 42 exemples à partir duquel un ensemble d'entraînement S de $m > 20$ exemples étiquetés a été tiré (sans remise), et soit $\mu > 0$ une valeur S-réalisable. Parmi toutes les distributions quasi-uniformes Q de marge empirique $\mu_1\big(M_Q^S\big)$ égale à μ, TMinCq consiste à trouver celle qui minimise $\mu_2\big(M_Q^Z\big)$.

Afin de construire le programme quadratique qui résout ce problème, nous devons d'abord définir une matrice de classification, homologue à la matrice de la définition 4.6, mais qui est définie sur tous les exemples de l'ensemble Z.

Définition 7.21 (La matrice de classification transductive)**.** Soit $Z = \langle (x_1, y_1), \ldots, (x_z, y_z) \rangle$ un échantillon complet et $\mathcal{H} = \{h_1, h_2, \ldots, h_{2n}\}$ un ensemble symétrique de classificateurs. La *matrice de classification transductive* $\mathbf{T}$ est donnée par

$$\mathbf{T} \triangleq \begin{bmatrix} h_1(x_1) & h_2(x_1) & \ldots & h_{2n}(x_1) \\ h_1(x_2) & h_2(x_2) & \ldots & h_{2n}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ h_1(x_z) & h_2(x_z) & \ldots & h_{2n}(x_z) \end{bmatrix}.$$

Nous dénotons $\widehat{\mathbf{T}}$ la matrice contenant les n premières colonnes de $\mathbf{T}$. Notons également que la matrice $\mathbf{H}$ définie à la définition 4.6 correspond aux m premières lignes de $\mathbf{T}$, et nous dénotons $\widehat{\mathbf{H}}$ la matrice contenant les n premières colonnes de $\mathbf{H}$. Nous réutilisons tel quel le vecteur d'étiquettes $\mathbf{y}$ de la définition 4.7, qui étant donné un ensemble d'entraînement $S = \langle (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \rangle$, est donné par

$$\mathbf{y} \triangleq \begin{bmatrix} y_1 & y_2 & \ldots & y_m \end{bmatrix}^\top.$$

Construisons premièrement un programme quadratique à $2n$ variables étant facilement déduit de la définition du problème d'optimisation de TMinCq. Considérant les définitions précédentes et étant donné une valeur μ S-réalisable, TMinCq est résolu par le problème d'optimisation décrit dans l'algorithme 6.

Finalement, le vote de majorité Q-pondéré retourné par l'algorithme 6 est donné par

$$B_Q(x) = \operatorname{sgn}\left[\sum_{i=1}^{2n} q_i h_i(x)\right].$$

L'algorithme 6 est très similaire à l'algorithme 3 (MinCq), avec comme différences que le second moment de la marge est calculé sur l'échantillon complet Z, et que l'algorithme n'est justifié théoriquement que pour les ensembles de classificateurs binaires. Tout comme nous avons pu le faire avec MinCq au chapitre 4 et en utilisant les mêmes astuces, nous pouvons réduire le programme quadratique 6 à $2n$ variables en un programme quadratique à n variables.

Algorithme 6 Programme quadratique TMinCq à $2n$ variables

$$
\begin{array}{lll}
\text{Résoudre} & \underset{\mathbf{q}}{\textit{argmin}} & \dfrac{1}{z}\,\mathbf{q}^\top\,\mathbf{T}^\top\mathbf{T}\,\mathbf{q}\,, \\
\text{sous contraintes} & & \dfrac{1}{m}\mathbf{y}^\top\mathbf{H}\,\mathbf{q} \;=\; \mu\,, \\
& & \begin{bmatrix}\mathbf{I}_n & \mathbf{I}_n\end{bmatrix}\mathbf{q} \;=\; \dfrac{1}{n}\mathbf{1}_n\,, \\
& & \mathbf{q} \;\geq\; \mathbf{0}_{2n}\,,
\end{array}
$$

où $\mathbf{q}$ est un vecteur de $2n$ variables représentant les poids associés aux votants, $\mathbf{T}$ est la matrice transductive de classification définie à la définition 7.21, $\mathbf{H}$ est la matrice de classification (inductive) définie à la définition 4.6, et $\mathbf{y}$ est le vecteur d'étiquettes défini à la définition 4.7.

Ce programme quadratique peut être réécrit de manière standard, sous la forme proposée par Boyd et Vandenberghe (2004, section 4, équation 4.34), avec $P = \frac{2}{z}\mathbf{T}^\top\mathbf{T}$, $q = \mathbf{0}_{2n}$, $r = 0$, $G = -\mathbf{I}_{2n}$, $h = \mathbf{0}_{2n}$, $A = \begin{bmatrix}\frac{1}{m}\mathbf{y}^\top\mathbf{H} \\ \mathbf{I}_n \quad \mathbf{I}_n\end{bmatrix}$, et $b = \begin{bmatrix}\mu \\ \frac{1}{n}\mathbf{1}_n\end{bmatrix}$.

Algorithme 7 Programme quadratique TMinCq à n variables

$$
\begin{array}{lll}
\text{Résoudre} & \underset{\hat{\mathbf{q}}}{\textit{argmin}} & \dfrac{1}{z}\hat{\mathbf{q}}^\top\hat{\mathbf{T}}^\top\hat{\mathbf{T}}\hat{\mathbf{q}} - \dfrac{1}{zn}\mathbf{1}_n^\top\hat{\mathbf{T}}^\top\hat{\mathbf{T}}\hat{\mathbf{q}} \\
\text{sous contraintes} & & \dfrac{2}{m}\mathbf{y}^\top\hat{\mathbf{H}}\,\hat{\mathbf{q}} \;=\; \mu + \dfrac{1}{mn}\mathbf{y}^\top\hat{\mathbf{H}}\,\mathbf{1}_n\,, \\
& & \mathbf{0}_n \preceq\; \mathbf{q} \;\preceq\; \dfrac{1}{n}\mathbf{1}_n\,,
\end{array}
$$

où $\hat{\mathbf{q}}$ est un vecteur de n variables représentant les poids des n premiers votants, $\hat{\mathbf{T}}$ est défini à la définition 7.21, $\hat{\mathbf{H}}$ correspond aux n premières colonnes de la matrice de classification (inductive) définie à la définition 4.6, et $\mathbf{y}$ est le vecteur d'étiquettes défini à la définition 4.7.

Ce programme quadratique peut être réécrit de manière standard, sous la forme proposée par Boyd et Vandenberghe (2004, section 4, équation 4.34), avec $P = \frac{2}{z}\hat{\mathbf{T}}^\top\hat{\mathbf{T}}$, $q = -\frac{1}{zn}\hat{\mathbf{T}}^\top\hat{\mathbf{T}}\mathbf{1}_n$, $r = 0$, $G = \begin{bmatrix}-\mathbf{I}_n \\ \mathbf{I}_n\end{bmatrix}$, $h = \begin{bmatrix}\mathbf{0}_n \\ \frac{1}{n}\mathbf{1}_n\end{bmatrix}$, $A = \frac{2}{m}\mathbf{y}^\top\hat{\mathbf{H}}$, et $b = \mu + \frac{1}{mn}\mathbf{y}^\top\hat{\mathbf{H}}\,\mathbf{1}_n$.

Nous obtenons le programme quadratique défini à l'algorithme 7. De nouveau, l'algorithme 7 est très similaire à l'algorithme 4 (MinCq), avec comme différences que le second moment de la marge est calculé sur l'échantillon complet Z, et que l'algorithme n'est justifié théoriquement que pour les ensembles de classificateurs binaires. Le vote de majorité Q-pondéré retourné par cet algorithme est

$$B_Q(x) \;=\; \operatorname{sgn}\left[\sum_{i=1}^{n}\left(2\,q_i - \frac{1}{n}\right)h_i(x)\right].$$

Dans la prochaine section, nous présentons des résultats d'expérimentations sur des ensembles de données dédiées à la comparaison d'algorithmes transductifs ou semi-supervisés.

7.4.2 Expérimentations

Nous évaluons la performance de TMinCq sur des ensembles de données tirés de Chapelle, Schölkopf et Zien (2006). Les éditeurs de ce livre dédié à l'apprentissage semi-supervisé ont fourni les mêmes ensembles de données aux auteurs des différents chapitres. Les résultats des algorithmes ont ensuite été comparés dans un chapitre dédié aux expérimentations.

Le cadre d'expérimentation de Chapelle, Schölkopf et Zien (2006) propose 8 ensembles de données[12], certains artificiels et d'autres naturels, pour comparer des algorithmes d'apprentissage dans le cadre transductif ou semi-supervisé.

Pour chaque ensemble, 12 ensembles d'entraînement de 10 exemples et 12 ensembles d'entraînement de 100 exemples sont fournis. Le petit nombre d'exemples étiquetés rend difficile la tâche de sélectionner les valeurs d'hyperparamètres. Plusieurs auteurs ont eu à fixer les valeurs de certains hyperparamètres à des valeurs choisies car elles « performent bien expérimentalement ». Autrement dit, plusieurs d'entre eux ont simplement optimisé les valeurs de certains hyperparamètres sur l'ensemble de test, ce qui donne des informations sur la meilleure performance possible de l'algorithme mais ne correspond pas nécessairement à sa performance en pratique, lorsque la tâche est de classifier des exemples dont nous n'avons pas l'étiquette.

Nous avons évalué l'algorithme TMinCq sur 6 ensembles de données parmi ceux proposés dans le cadre d'expérimentation décrit ci-haut. Pour chaque ensemble de données et chacun des 12 ensembles d'entraînement de 100 exemples fournis[13], la technique de validation croisée à 5 plis a été utilisée pour choisir les valeurs d'hyperparamètres. Même si notre théorie PAC-bayésienne ne justifie que les ensembles de votants qui sont des classificateurs binaires, nous avons tout de même évalué empiriquement MinCq (qui ne considère que les exemples étiquetés) et TMinCq en utilisant les deux mêmes familles de votants que dans nos expérimentations des chapitres précédents. MinCq^S et TMinCq^S utilisent comme votants 10 souches de décision par attribut, et les 10 souches de décision inverses. MinCq$^K_\gamma$ et TMinCq$^K_\gamma$ utilisent comme votants des noyaux RBF $k(x, x') = exp\left(-\gamma||x - x'||^2\right)$, dont la valeur de γ a été choisie par validation croisée parmi des valeurs entre 10^{-4} et 10^1 sur une échelle logarithmique. La valeur de l'hyperparamètre μ a été choisi parmi des valeurs entre 10^{-2} et $10^{-0.5}$ pour MinCq^S et TMinCq^S, et entre 10^{-4} et 10^{-2} pour MinCq$^K_\gamma$ et TMinCq$^K_\gamma$, également sur une échelle logarithmique. La table 7.4 est une reproduction des résultats présentés dans Chapelle, Schölkopf et Zien (2006), où les résultats de nos algorithmes ont été ajoutés.

Nous remarquons d'abord que même si la théorie ne supporte pas l'utilisation de votants à valeur réelle, TMinCq$^K_\gamma$ offre de meilleurs performances en pratique que TMinCq^S. La per-

12. Les ensembles de données de Chapelle, Schölkopf et Zien (2006) peuvent être téléchargés à `http://olivier.chapelle.cc/ssl-book/benchmarks.html`.

13. Les ensembles étiquetés de seulement 10 exemples n'ont pas été considérés car la sélection des hyperparamètres est complexe, et TMinCq n'est théoriquement justifié par la théorie que pour $m > 20$ exemples.

	g241c	g241d	Digit1	USPS	COIL	BCI
1-NN	40.28	37.49	6.12	7.64	23.27	44.83
SVM	23.11	24.64	5.53	9.75	22.93	34.31
21.2.8 MVU + 1-NN	44.05	43.21	3.99	6.09	32.27	47.42
21.2.8 LEM + 1-NN	42.14	39.43	2.52	6.09	36.49	48.64
21.2.4 QC + CMN	22.05	28.20	3.15	6.36	10.03	46.22
21.2.6 Discrete Reg.	43.65	41.65	2.77	4.68	9.61	47.67
21.2.1 TSVM	18.46	22.42	6.15	9.77	25.80	33.25
21.2.1 SGT	17.41	9.11	2.61	6.80	–	45.03
21.2.10 Cluster-Kernel	13.49	4.95	3.79	9.68	21.99	35.17
21.2.3 Data-Dep. Reg.	20.31	32.82	2.44	5.10	11.46	47.47
21.2.11 LDS	18.04	23.74	3.46	4.96	13.72	43.97
21.2.5 Laplacian RLS	24.36	26.46	2.92	4.68	11.92	31.36
21.2.7 CHM (normed)	24.82	25.67	3.79	7.65	–	36.03
MinCq^S	28.18	29.61	11.13	12.71	17.01	42.22
TMinCq^S	35.55	36.59	18.03	16.23	20.46	39.69
MinCq$^K_\gamma$	24.19	28.59	7.36	8.23	14.48	39.61
TMinCq$^K_\gamma$	24.04	26.89	7.16	7.44	18.27	41.20

TAB. 7.4: Comparaison de MinCq et TMinCq avec les algorithmes de l'état de l'art dans le contexte de l'apprentissage transductif, sur des ensembles de données provenant du cadre expérimental de CHAPELLE, SCHÖLKOPF et ZIEN (2006). La description des différents ensembles de données et des algorithmes évalués peut y être retrouvée. Chaque algorithme est entraîné sur un ensemble S de 100 exemples étiquetés et évalué sur l'ensemble U des exemples non étiquetés. Ce processus est répété 12 fois, comme 12 ensembles d'entraînement S sont fournis. La valeur indiquée dans chaque cellule est la moyenne des risques sur les différents ensembles U.

formance de TMinCq^S est décevante : le nombre de fois où cet algorithme performe mieux qu'un autre algorithme parmi ceux montrés à la table 7.4 est 4, 4, 0, 0, 6 et 8 sur un total de 13 algorithmes, respectivement pour chaque ensemble de données de gauche à droite. Par ailleurs, TMinCq^S performe moins bien que MinCq^S, qui lui ne considère pas les exemples non-étiquetés. TMinCq$^K_\gamma$ offre une performance un peu plus intéressante, avec un nombre d'algorithmes surpassés de 6, 6, 0, 5, 6 et 8. Ici, la performance de la version transductive de l'algorithme surpasse celle de l'algorithme inductif sur la plupart des ensembles de données.

Nous croyons que la mauvaise performance de TMinCq est due au fait que celui-ci n'est pas directement inspiré des propriétés que peuvent avoir la distribution des exemples : les exemples non étiquetés sont seulement utilisés pour calculer le second moment de la marge de manière exacte. La plupart des algorithmes transductifs et semi-supervisés sont construits en fonction de certaines suppositions qui sont souvent faites sur les données afin qu'un algorithme puisse tirer profit des exemples non étiquetés. Nous discutons de ces suppositions dans la prochaine section.

7.4.3 L'apprentissage transductif en pratique

Une question importante qui se pose est la suivante : est-ce que l'utilisation des données non étiquetées permet d'améliorer la précision de la prédiction des algorithmes supervisés standards ? En général on ne sait pas, mais avec certaines suppositions supplémentaires sur la structure du problème, nous constatons une amélioration empirique (Chapelle, Schölkopf et Zien, 2006). Pour que l'information supplémentaire donnée par les exemples non étiquetés soit intéressante, la *distribution des exemples* doit être utile à la tâche de classification. Si ce n'est pas le cas, l'utilisation de ces exemples n'améliorera pas la précision du classificateur résultant, et pourrait même dégrader les performances. Chapelle, Schölkopf et Zien (2006) indiquent que pour que l'apprentissage transductif fonctionne, au moins une des suppositions suivantes doit tenir, et l'algorithme utilisé doit être en mesure de tenir compte de celle-ci.

En pratique, il peut être difficile de déterminer si un ensemble de données respecte l'une ou plusieurs de ces suppositions. C'est pourquoi le cadre expérimental de Chapelle, Schölkopf et Zien (2006) a été conçu : les différents ensembles de données utilisés pour comparer les diverses algorithmes d'apprentissage sont analysés de manière à déterminer laquelle ou lesquelles de ces suppositions sont vérifiées.

Supposition de continuité semi-supervisée («*semi-supervised smoothness assumption*»)

Dans le cas supervisé, nous sommes intéressés à savoir si la sortie (l'étiquette dans le cas de la classification) varie de manière lisse en fonction de la distance entre deux exemples. Dans le contexte semi-supervisé, nous devons également prendre en considération la *densité* $p(x)$ *des entrées*. Plus il y a d'exemples dans une certaine région de l'espace d'observation, plus la probabilité d'y observer un nouvel exemple tiré de cette même distribution est haute. Cette supposition implique donc que les étiquettes varient de manière plus lisse dans les régions de haute densité que dans les régions de basse densité. De manière équivalente, si deux points sont liés par un chemin de haute densité, alors leurs sorties sont probablement proches.

Définition 7.22 (Supposition de continuité semi-supervisée)**.** Si deux points x_1 et x_2 appartenant à une région de haute densité sont proches, alors leurs sorties correspondantes y_1 et y_2 devraient l'être.

Supposition de regroupement («*cluster assumption*»)

Si nous prenons en considération que les points de chaque classe ont tendance à former un regroupement, les données non étiquetées aident à trouver les frontières de chaque regroupement. L'une des premières formes d'apprentissage semi-supervisé est d'exécuter un algorithme de «*clustering*» et d'appliquer ensuite une classe à chaque regroupement.

Définition 7.23 (Supposition de regroupement)**.** Si deux points sont dans le même regroupement, ils ont probablement la même classe.

Notons que cette supposition n'implique pas que chaque classe ne forme qu'un groupement. Elle énonce simplement que des exemples avec une étiquette différente ne devraient pas se retrouver dans un même groupement. De manière équivalente, nous pouvons formuler cette supposition comme suit :

Définition 7.24 (Séparation de basse densité)**.** La frontière de décision devrait se trouver dans une région de basse densité.

De nombreux algorithmes implémentent cette supposition, notamment *Transductive Boosting* (Taira et Haruno, 2001), une adaptation transductive de l'algorithme AdaBoost (Freund et Schapire, 1997), et *Transductive SVM* (Joachims, 2006), une adaptation de l'algorithme SVM (Cortes et Vapnik, 1995). Dans Amini, Laviolette et Usunier (2008), les auteurs utilisent également cette supposition en étiquetant itérativement les données non étiquetées ayant une marge supérieure à un certain seuil, déterminé par la minimisation d'une borne transductive sur le risque du classificateur par vote de majorité.

Supposition de variété («*manifold assumption*»)

Cette supposition indique que les données de haute dimensionnalité doivent se trouver sur une variété (ou «*manifold*») de basse dimensionnalité. Une variété est un espace construit à partir d'autres espaces plus simples. Par exemple, un cylindre dans un espace en trois dimensions peut être obtenu en repliant une bande plane (en deux dimensions) sur elle-même.

Lorsque les données sont dans un espace de grande dimensionnalité, nous devons vivre avec le *fléau de la dimensionnalité*. Non seulement la complexité des algorithmes croît très rapidement en fonction du nombre de dimensions, mais l'expressivité de la notion de distance euclidienne entre les exemples décroît. En effet, la distance entre chaque paire d'exemples tend à devenir de plus en plus semblable. Par contre, si la supposition de variété est vérifiée, il devient possible à l'algorithme d'apprentissage d'opérer dans l'espace correspondant à cette variété, et donc d'éviter le fléau de la dimensionnalité.

L'adaptation du SVM nommée *Laplacian SVM* (ou *LapSVM*) de Belkin, Niyogi et Sindhwani (2006) implémente directement ces deux dernières suppositions en ajoutant un terme de régularisation oeuvrant dans la géométrie intrinsèque des données.

Définition 7.25 (Supposition de variété)**.** Les données de haute dimensionnalité doivent se trouver sur une variété de basse dimensionnalité.

Afin de terminer ce chapitre sur une note plus positive que les résultats décevants de l'algorithme TMinCq, nous construisons dans la prochaine section un algorithme intégrant un

terme de régularisation basé sur la géométrie intrinsèque des données, inspiré de LapSVM. Cet algorithme, nommé LapMinCq, offre des résultats intéressants sur un ensemble de données « jouet » et ouvre la porte à une avenue de recherche intéressante.

7.4.4 LapMinCq : régularisation par le Laplacien du graphe

Il est souvent soutenu en apprentissage transductif que, si la distribution qui génère les données a un support sur une *variété* (*manifold*) dans l'espace ambiant, une métrique sur cette variété tend à être plus appropriée pour l'apprentissage (Belkin, Matveeva et Niyogi, 2004 ; Sindhwani, Niyogi et Belkin, 2005 ; Belkin, Niyogi et Sindhwani, 2006 ; Niyogi, 2013). En effet, si la structure des données est un facteur clé dans l'apprenabilité d'une tâche (autrement dit, si la supposition de variété définie à la définition 7.25 est respectée), c'est la géométrie intrinsèque des données qui capturera cette structure pertinente le plus précisément. Une approximation de la géométrie intrinsèque de la distribution qui génère les données devrait considérer une relation entre les points de l'ensemble d'entraînement. Ainsi, aucune quantité reliée à la géométrie intrinsèque associée à un exemple en particulier ne peut être indépendante du reste de l'échantillon. Comme les deux premiers moments de la marge $\mu_1\left(M_Q^{D'}\right)$ et $\mu_2\left(M_Q^{D'}\right)$ sont des moyennes de valeurs qui, individuellement, ne dépendent que d'un exemple, il y a très peu d'information sur cette géométrie qui est contenue dans la C-borne. Ainsi, dans la perspective d'améliorer TMinCq en le modifiant pour utiliser cette géométrie, nous devons considérer une Q-moyenne de fonctions qui dépendent individuellement de plus d'un exemple. En d'autres mots, les U-statistiques d'ordre au moins 2 doivent être considérées (voir Lee (1990) pour plus de détails sur les U-statistiques).

Afin d'utiliser la géométrie intrinsèque des données, plusieurs algorithmes comme *LapSVM* (pour *Laplacian Support Vector Machines*) et *LapRLS* (pour *Laplacian Regularized Least Squares*) (Belkin, Niyogi et Sindhwani, 2006) introduisent un terme de régularisation basé sur une fonction de «*smoothness*». Dans notre situation, une telle fonction devrait pénaliser les votes de majorité dont la valeur réelle (avant d'en prendre le signe) n'est pas *smooth* en fonction de la distance entre les exemples. Soit $\mathcal{H}$ un ensemble de classificateurs et Q une distribution sur $\mathcal{H}$. Dénotons H_Q la Q-moyenne des fonctions de $\mathcal{H}$, c'est-à-dire,

$$H_Q(x) \triangleq \mathop{\mathbf{E}}_{h\sim Q} h(x)\,.$$

Notons que l'on retrouve la notion de marge sur un exemple avec $M_Q(x,y) = yH_Q(x)$ et le vote de majorité avec $B_Q(x) = \operatorname{sgn}\left[H_Q(x)\right]$. Étant donné un échantillon complet Z de z exemples et en s'inspirant la fonction utilisée pour LapSVM (Belkin, Niyogi et Sindhwani, 2006), mais en l'appliquant la la valeur $H_Q(x)$, nous obtenons la fonction de smoothness suivante :

$$U_{Q^2}^Z \triangleq \frac{1}{2\,z^2}\sum_{k=1}^{z}\sum_{l=1}^{z} W_{kl}\left(H_Q(x_k) - H_Q(x_l)\right)^2\,, \tag{7.23}$$

où W_{kl} est un élément d'une matrice symétrique $\mathbf{W}$ mesurant la similarité entre les exemples, par exemple

$$W_{kl} = \begin{cases} e^{-\|x_k - x_l\|} & \text{si } x_k, x_l \text{ sont une paire de } k \text{ plus proches voisins} \\ 0 & \text{autrement}, \end{cases}$$

pour une norme $\|\cdot\|$ sur $\mathcal{X}$. Étant donné une fonction H_Q, il est facile de constater que si $\mathcal{U}^Z_{Q^2}$ a une petite valeur, pour tous x et x' proches (dans le sens de la mesure de similarité W), $H_Q(x)$ et $H_Q(x')$ devraient avoir des valeurs réelles proches. Une distribution Q qui donne une petite valeur de $\mathcal{U}^Z_{Q^2}$ impliquera donc que H_Q est « smooth » par rapport à la mesure de similarité W. Il est intéressant de souligner que

$$\begin{aligned} \mathcal{U}^Z_{Q^2} &= \frac{1}{2z^2} \begin{bmatrix} H_Q(x_1) & \dots & H_Q(x_z) \end{bmatrix} \mathbf{L} \begin{bmatrix} H_Q(x_1) \\ \vdots \\ H_Q(x_z) \end{bmatrix} \\ &= \frac{1}{z^2} \mathbf{q}^\top \mathbf{T}^\top \mathbf{L}\, \mathbf{T}\, \mathbf{q}, \end{aligned} \tag{7.24}$$

où $\mathbf{T}$ est défini à la définition 7.21, où $\mathbf{L} = \mathbf{D} - \mathbf{W}$ est le *Laplacien du graphe* (voir Chung (1997)) dont les sommets sont les exemples et les poids sur les arêtes sont contrôlés par $\mathbf{W}$, et où $\mathbf{D}$ est une matrice diagonale de $z \times z$ éléments, telle que $D_{kk} = \sum_l W_{kl}$. Le Laplacien du graphe est une quantité couramment utilisée dans la littérature, notamment dans les algorithmes LapSVM et LapRLS mentionnés plus haut (Belkin, Niyogi et Sindhwani, 2006) qui utilisent un terme de régularisation similaire avec les mêmes motivations, et dans les algorithmes de «*spectral clustering*» et de «*normalized cut*» (voir Von Luxburg (2007) pour un turoriel sur ces sujets). Ce type de régularisation est également appelé *régularisation Laplacienne* (Weinberger et al., 2006 ; Ando et T. Zhang, 2007). Notons finalement que l'égalité entre les équations (7.23) et (7.24) est rapportée dans Belkin, Niyogi et Sindhwani (2006) avec un facteur 2 de différence, ce qui est une erreur de leur part. L'égalité entre les équations (7.23) et (7.24) est démontrée au lemme A.19 en annexe A.

La prochaine extension transductive de MinCq, que nous nommons *LapMinCq*, consiste à ajouter à TMinCq un terme de régularisation basé sur l'Équation (7.23) pour favoriser les distributions quasi-uniformes Q qui admettent des fonctions H_Q plus « smooth ». Plus précisément, la définition 7.26 présente l'algorithme résultant.

Définition 7.26 (LapMinCq)**.** Étant donné un ensemble symétrique $\mathcal{H}$ de classificateurs, un échantillon complet $Z = \langle (x_1, y_1), \dots, (x_z, y_z) \rangle$ de plus de 42 exemples à partir duquel un ensemble d'entraînement S de $m > 20$ exemples étiquetés a été tiré (sans remise), et soit $\mu > 0$ une valeur S-réalisable. Parmi toutes les distributions quasi-uniformes Q de marge empirique $\mu_1\big(M^S_Q\big)$ égale à μ, LapMinCq consiste à trouver celle qui minimise $\mu_2\big(M^Z_Q\big) + \zeta\, \mathcal{U}^Z_{Q^2}$, où $\zeta \geq 0$ est un hyperparamètre contrôlant le compromis entre le second moment de la marge et le régularisateur intervenant dans la géométrie intrinsèque des données.

Tout comme les autres versions de MinCq, l'algorithme LapMinCq peut être transformé en un programme quadratique à $2n$ ou n variables, pouvant ensuite être résolu par un solveur. Ce programme quadratique est très similaire à TMinCq (algorithme 6), puisque seule la fonction objectif est modifiée. En effet, plutôt que de minimiser seulement $\mu_2\left(M_Q^Z\right)$, nous devons minimiser $\mu_2\left(M_Q^Z\right)+\zeta\, \mathcal{U}_{Q^2}^Z$. Sous forme matricielle, en considérant la matrice de classification transductive $\mathbf{T}$ de la définition 7.21 et le Laplacien $\mathbf{L}$, minimiser cette expression correspond à trouver le vecteur $\mathbf{q}$ qui minimise

$$\frac{1}{z}\mathbf{q}^\top\mathbf{T}^\top\mathbf{T}\,\mathbf{q}+\frac{\zeta}{z^2}\mathbf{q}^\top\mathbf{T}^\top\mathbf{L}\,\mathbf{T}\,\mathbf{q}\,,$$

ou de manière équivalente,

$$\frac{1}{z}\mathbf{q}^\top\mathbf{T}^\top\left(\mathbf{I}_z+\frac{\zeta}{z}\mathbf{L}\right)\mathbf{T}\,\mathbf{q}\,.$$

Les programmes quadratiques résultants sont les suivants.

Algorithme 8 Programme quadratique LapMinCq à $2n$ variables

$$\begin{aligned}
\text{Résoudre}\qquad & \underset{\mathbf{q}}{argmin}\quad \frac{1}{z}\,\mathbf{q}^\top\,\mathbf{T}^\top\left(\mathbf{I}_z+\frac{\zeta}{z}\mathbf{L}\right)\mathbf{T}\,\mathbf{q}\,,\\
\text{sous contraintes}\qquad & \frac{1}{m}\mathbf{y}^\top\mathbf{H}\,\mathbf{q}\;=\;\mu\,,\\
& \begin{bmatrix}\mathbf{I}_n & \mathbf{I}_n\end{bmatrix}\mathbf{q}\;=\;\frac{1}{n}\mathbf{1}_n\,,\\
& \mathbf{q}\;\geq\;\mathbf{0}_{2n}\,,
\end{aligned}$$

où $\mathbf{q}$ est un vecteur de $2n$ variables représentant les poids associés aux votants, $\mathbf{T}$ est la matrice transductive de classification définie à la définition 7.21, $\mathbf{H}$ est la matrice de classification (inductive) définie à la définition 4.6, $\mathbf{y}$ est le vecteur d'étiquettes défini à la définition 4.7, et $\mathbf{L}$ est le Laplacien du graphe. La forme standard de ce programme quadratique est la même que pour l'algorithme 6, avec $P=\frac{2}{m}\mathbf{T}^\top\left(\mathbf{I}_z+\frac{\zeta}{z}\mathbf{L}\right)\mathbf{T}$.

Algorithme 9 Programme quadratique LapMinCq à n variables

$$\begin{aligned}
\text{Résoudre}\qquad & \underset{\hat{\mathbf{q}}}{argmin}\quad \frac{1}{z}\hat{\mathbf{q}}^\top\hat{\mathbf{T}}^\top\left(\mathbf{I}_z+\frac{\zeta}{z}\mathbf{L}\right)\hat{\mathbf{T}}\hat{\mathbf{q}}-\frac{1}{zn}\mathbf{1}_n^\top\hat{\mathbf{T}}^\top\left(\mathbf{I}_z+\frac{\zeta}{z}\mathbf{L}\right)\hat{\mathbf{T}}\hat{\mathbf{q}}\\
\text{sous contraintes}\qquad & \frac{2}{m}\mathbf{y}^\top\hat{\mathbf{H}}\,\hat{\mathbf{q}}\;=\;\mu+\frac{1}{mn}\mathbf{y}^\top\hat{\mathbf{H}}\,\mathbf{1}_n\,,\\
& \mathbf{0}_n\preceq\;\mathbf{q}\;\preceq\;\frac{1}{n}\mathbf{1}_n\,,
\end{aligned}$$

où $\hat{\mathbf{q}}$ est un vecteur de n variables représentant les poids des n premiers votants, $\hat{\mathbf{T}}$ est défini à la définition 7.21, $\hat{\mathbf{H}}$ correspond aux n premières colonnes de la matrice de classification (inductive) définie à la définition 4.6, $\mathbf{y}$ est le vecteur d'étiquettes défini à la définition 4.7, et $\mathbf{L}$ est le Laplacien du graphe. La forme standard de ce programme quadratique est la même que pour l'algorithme 7, avec $P=\frac{2}{m}\hat{\mathbf{T}}^\top\left(\mathbf{I}_z+\frac{\zeta}{z}\mathbf{L}\right)\hat{\mathbf{T}}$ et $q=-\frac{1}{zn}\hat{\mathbf{T}}^\top\left(\mathbf{I}_z+\frac{\zeta}{z}\mathbf{L}\right)\hat{\mathbf{T}}\mathbf{1}_n$.

Le vote de majorité Q-pondéré retourné par l'algorithme 8 est donné par

$$B_Q(x) \;=\; \operatorname{sgn}\left[\sum_{i=1}^{2n} q_i h_i(x)\right],$$

et celui de l'algorithme 9 par

$$B_Q(x) \;=\; \operatorname{sgn}\left[\sum_{i=1}^{n} \left(2\, q_i - \frac{1}{n}\right) h_i(x)\right].$$

Afin de démontrer que les algorithmes 8 et 9 sont bien des programmes quadratiques, nous devons montrer que les matrices $\mathbf{T}^\top \left(\mathbf{I}_z + \frac{\zeta}{z}\mathbf{L}\right)\mathbf{T}$ et $\hat{\mathbf{T}}^\top \left(\mathbf{I}_z + \frac{\zeta}{z}\mathbf{L}\right)\hat{\mathbf{T}}$ sont semi-définies positives. Nous savons par les propriétés de la matrice identité $\mathbf{I}_z$ que celle-ci est définie positive (et donc semi-définie positive), et par les propriétés du Laplacien $\mathbf{L}$ que celui-ci est également semi-défini positif. Comme la somme de deux matrices semi-définies positives et la multiplication d'une matrice semi-définie positive par une valeur réelle positive ont comme résultat une matrice ayant également cette propriété, comme $\zeta \geq 0$ et $z > 0$, nous déduisons que la matrice $\mathbf{I}_z + \frac{\zeta}{z}\mathbf{L}$ est semi-définie positive. Cette propriété ainsi que le fait que les entrées de cette matrice sont des valeurs réelles implique qu'il existe une matrice $\mathbf{A}$ telle que $\mathbf{I}_z + \frac{\zeta}{z}\mathbf{L} = \mathbf{A}\,\mathbf{A}^\top$ est la seule et unique *décomposition de Cholesky* de $\mathbf{I}_z + \frac{\zeta}{z}\mathbf{L}$. Nos deux matrices peuvent donc être réécrites sous la forme $\mathbf{T}^\top\mathbf{A}\mathbf{A}^\top\mathbf{T}$ et $\hat{\mathbf{T}}^\top\mathbf{A}\mathbf{A}^\top\hat{\mathbf{T}}$. Maintenant, nous savons qu'une matrice $\mathbf{P}$ de $z \times z$ éléments est semi-définie positive si et seulement si pour tout vecteur $\mathbf{q} \in \mathbb{R}^n$, $\mathbf{q}^\top\mathbf{P}\mathbf{q} \geq 0$. Or, dans le cas qui nous intéresse, $\mathbf{q}^\top\mathbf{P}\mathbf{q}$ peut être écrit sous la forme $\mathbf{q}^\top\mathbf{T}^\top\mathbf{A}\mathbf{A}^\top\mathbf{T}\mathbf{q} \;=\; \left\|\mathbf{q}^\top\mathbf{T}^\top\mathbf{A}\right\|^2$ ou sous la forme $\mathbf{q}^\top\hat{\mathbf{T}}^\top\mathbf{A}\mathbf{A}^\top\hat{\mathbf{T}}\mathbf{q} \;=\; \left\|\mathbf{q}^\top\hat{\mathbf{T}}^\top\mathbf{A}\right\|^2$, et ces valeurs sont nécessairement positives ou nulles puisqu'il s'agit de carrés.

Il est finalement intéressant de noter que contrairement aux algorithmes MinCq et TMinCq, LapMinCq n'est pas seulement basé sur les U-statistiques d'ordre 1, il implique également les U-statistiques d'ordre 2. En effet, la quantité $\mathcal{U}^Z_{Q^2}$ dépend de moyennages de valeurs qui dépendent individuellement de paires de points.

7.4.5 Expérimentations sur un ensemble de données artificiel

Afin de comparer le comportement de MinCq, TMinCq et LapMinCq dans le cadre transductif, nous expérimentons à l'aide d'un ensemble artificiel de données classique nommé *les deux lunes*[14]. Il s'agit d'un problème de classification binaire en deux dimensions, où les exemples de chaque classe forment un regroupement en forme de lune. Belkin, Niyogi et Sindhwani (2006) utilisent cet ensemble de données pour comparer les algorithmes SVM (Cortes et Vapnik, 1995), *Transductive SVM* (Joachims, 2006) et leur algorithme *Laplacian SVM*, en considérant le cas extrême où il n'y a qu'un seul exemple étiqueté par classe. Il s'agit donc ici

14. Cet ensemble de données est également connu sous le nom des *deux croissants*, surtout lorsque les auteurs viennent de la France.

d'évaluer empiriquement si un algorithme transductif est en mesure d'utiliser efficacement les données non étiquetées. Notons par contre que même s'il n'y a qu'un seul exemple étiqueté par classe, cet ensemble de données respecte les trois suppositions permettant à l'apprentissage transductif de fonctionner en pratique : la supposition de continuité semi-supervisée (définition 7.22), la supposition de regroupement (définition 7.23) et la supposition de variété (définition 7.25).

D'abord, la figure 7.12 présente une comparaison entre MinCq (algorithme 4) dans le cadre inductif (qui ignore complètement les exemples non étiquetés), TMinCq (algorithme 7), qui calcule le second moment de la marge en utilisant les données étiquetées et non étiquetées, et LapMinCq (algorithme 9), qui offre un compromis entre le second moment de la marge et le terme de régularisation oeuvrant dans la géométrie intrinsèque des données. Nous exécutons ces trois algorithmes en utilisant deux types de votants : des souches de décision, et des noyaux RBF centrés sur les exemples[15], c'est-à-dire $f(\cdot) = y\,k(x,\cdot)$ pour chaque exemple (x,y), et deux votants $f(\cdot) = \pm k(x,\cdot)$ pour chaque exemple non étiqueté x, où $k(x,x') = exp\left(-\|x-x'\|^2/2\sigma^2\right)$, et où σ est le paramètre de « largeur » du noyau, est fixé à 10^{-1}.

On remarque que comme MinCq ne tient pas compte des données non étiquetées, la frontière de décision du vote de majorité résultant ne fait que séparer les deux exemples étiquetés. Nous voyons également que TMinCq arrive à tenir compte des données non étiquetées et produit un classificateur avec une meilleure performance. Ceci dit, il est facile de voir sur cet ensemble de données que la géométrie des données est peu considérée : seul le désaccord entre les votants (le second moment de la marge) tient compte des données non étiquetées. Finalement, on remarque une impressionnante performance de LapMinCq en utilisant des noyaux RBF comme votants : les deux lunes sont presque parfaitement séparées. LapMinCq est par contre incapable d'offrir une bonne performance en utilisant des souches de décision. Notre hypothèse est que la fonction de « smoothness » définie à l'équation (7.23) n'est pas un bon terme de régularisation lorsque les votants sont des classificateurs binaires.

La figure 7.13 présente l'effet de l'hyperparamètre ζ de l'algorithme LapMinCq, contrôlant le compromis entre la minimisation du second moment de la marge et la régularisation dans la géométrie intrinsèque des données. La valeur de ζ est variée entre 10^0 et 10^9. On remarque encore une fois que les souches de décision ne sont pas appropriées pour LapMinCq : l'augmentation de la valeur de ζ ne fait que dégrader les résultats. Les résultats sont beaucoup plus intéressants lors de l'utilisation de noyaux RBF comme votants. En effet, on remarque qu'il est possible de trouver un bon compromis entre la minimisation du second moment de la marge et la régularisation basée sur le Laplacien du graphe, mais qu'aucune de ces quantités

15. Notons que les algorithmes TMinCq et LapMinCq ont été définis en ne considérant que les votants qui sont des classificateurs binaires, car la théorie PAC-bayésienne transductive n'est jusqu'à maintenant valide que pour ce cas. Nous décidons tout de même de présenter des résultats pour des fonctions à valeur réelle car celles-ci offrent de meilleurs résultats empiriques.

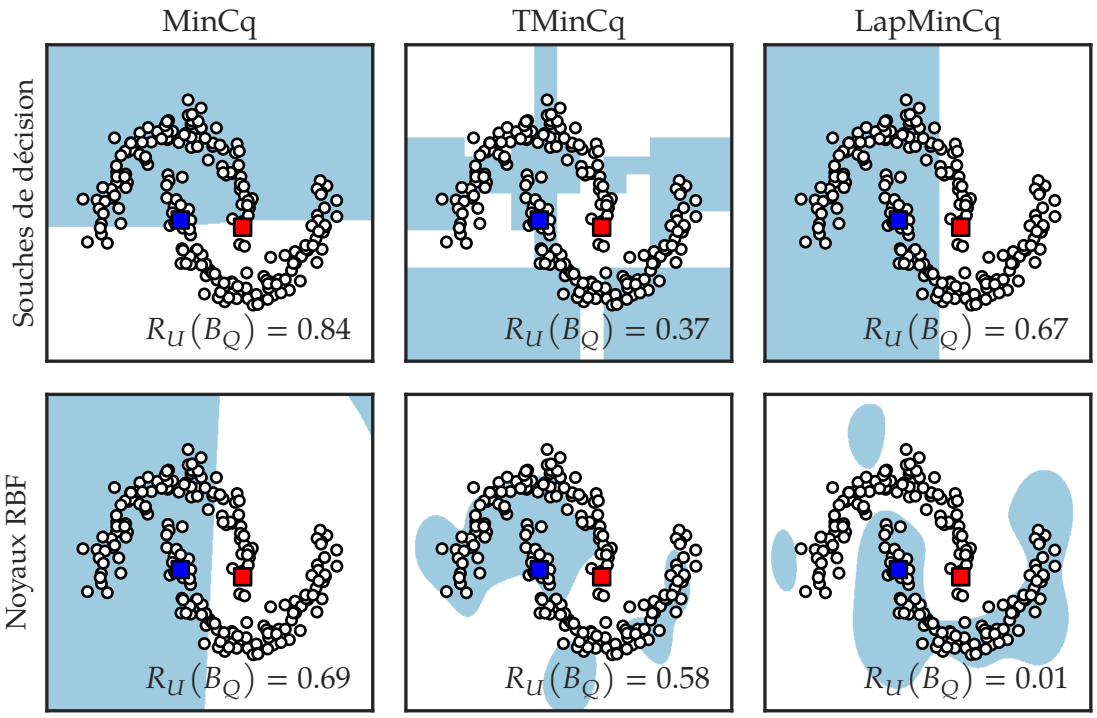

FIG. 7.12: Comparaison de MinCq, TMinCq et LapMinCq sur l'ensemble de données des deux lunes. Pour chacun des trois algorithmes, la valeur du paramètre μ est fixée à 10^{-4}, et le paramètre σ du noyau RBF est fixé à 10^{-1}. Le paramètre ζ de LapMinCq est fixé à 10^{-6}.

individuelles ne réussit à donner d'aussi bons résultats.

Ces résultats encourageants sur un ensemble de données synthétiques ouvrent la voie à une exploration plus exhaustive de ces algorithmes sur divers ensembles de données. Nous laissons cette exploration comme travaux futurs.

7.5 Conclusion du chapitre

Dans ce chapitre, nous avons proposé un processus simplifié de preuve pour les bornes PAC-bayésiennes. Cette nouvelle démonstration peut être facilement personnalisée, et ainsi elle permet d'obtenir de nouveau théorèmes PAC-bayésiens.

En personnalisant la preuve, nous avons créé une nouvelle famille de bornes PAC-bayésiennes basées sur la divergence de Rényi plutôt que sur la traditionnelle divergence Kullback-Leibler. Un paramètre α offre un compromis entre la valeur de la divergence et la valeur des autres termes de la borne. Lorsque $\alpha = 2$, nous obtenons une borne basée sur la divergence chi carré. Nous avons comparé empiriquement les valeurs de bornes obtenues avec les bornes classiques, et observé que les bornes sont légèrement plus serrées en pratique. Également, nous avons observé que la perte de précision associée aux différentes inégalités composant la

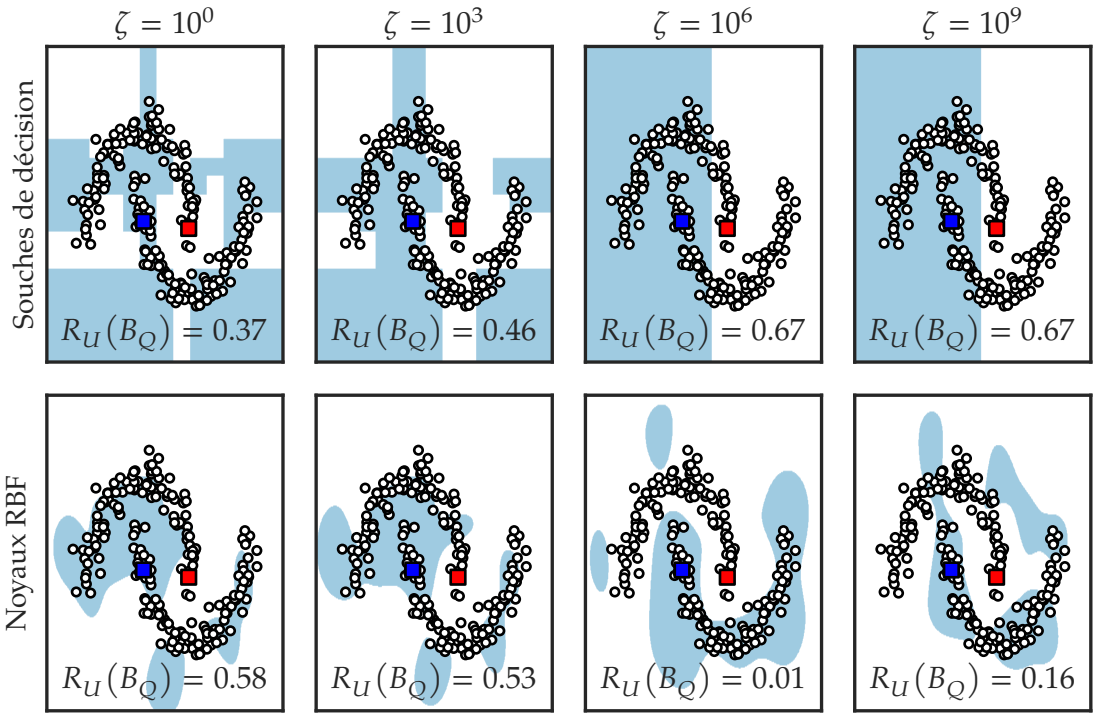

FIG. 7.13: Exploration de l'effet de l'hyperparamètre ζ de l'algorithme LapMinCq, en utilisant comme votants des souches de décision et des noyaux RBF.

preuve sont différentes : l'inégalité du changement de mesure est plus serré, mais cet avantage est perdu dans les étapes subséquentes (inégalité de Markov supremum sur le risque). Cette nouvelle famille de bornes ouvre donc la voie à l'exploration de différentes améliorations de ces étapes, afin d'obtenir des bornes encore plus serrées.

À l'aide d'une personnalisation différente à une autre étape, nous avons obtenu une nouvelle famille de bornes PAC-bayésiennes spécialisées à l'apprentissage transductif plutôt qu'inductif. Ces bornes ne nécessitent pas d'étape de supremum sur le risque (continu), mais plutôt le calcul d'un maximum sur le nombre d'erreurs (discret). Ceci ouvre la voie à l'utilisation de Δ-fonctions pour lesquelles nous n'avons pas d'expression analytique pour calculer ce supremum. Nous avons montré empiriquement ce que différentes Δ-fonctions ont comme impact sur la valeur de la borne, et avons comparé notre borne PAC-bayésienne transductive aux bornes inductives et à une borne PAC-bayésienne transductive de la littérature.

Comme le cadre de l'apprentissage transductif nous permet de calculer la valeur exacte du second moment de la marge, nous pouvons transformer les bornes sur le premier moment de la marge en une borne sur le risque du classificateur par vote de majorité. De celle-ci, nous avons déduit une version transductive de l'algorithme MinCq, qui minimise le « vrai » second moment de la marge, et dont seule une estimation empirique de son premier mo-

ment est nécessaire. Nous avons comparé cet algorithme, nommé TMinCq, aux algorithmes de l'état de l'art dans un cadre expérimental spécialisé pour l'apprentissage transductif et semi-supervisé. Malheureusement, les résultats obtenus avec cet algorithme sont plutôt décevants et demanderaient selon nous davantage de développement sur comment bien choisir les valeurs d'hyperparamètres pour que cette approche atteigne son plein potentiel.

En se basant sur la *supposition de variété* utilisée par plusieurs algorithmes de la littérature dont LapSVM, nous avons construit une version de TMinCq ajoutant un terme de régularisation oeuvrant dans la géométrie intrinsèque des données. Cet algorithme semble très prometteur mais n'a pas une justification théorique aussi rigoureuse que MinCq ou TMinCq, et le nombre de paramètres à optimiser rend son analyse empirique plus complexe. LapMinCq est très prometteur, et nous laissons son exploration comme travaux futurs.

Conclusion

Dans cette thèse, nous avons étudié les votes de majorité, la théorie PAC-bayésienne et la construction d'algorithmes d'apprentissage rigoureusement justifiés théoriquement.

Votes de majorité et C-borne Au chapitre 2, nous avons présenté les concepts de base des classificateurs par vote de majorité, une famille de classificateurs combinant des votants généralement peu performants individuellement en un vote de majorité pondéré de meilleure performance. Nous avons présenté la notion de marge d'un classificateur par vote de majorité, dont le premier moment statistique encode une notion de confiance sur la performance individuelle des votants, et dont le second moment statistique encode une notion de désaccord entre les paires de votants.

En utilisant ces deux quantités, nous avons redéfini la C-borne, une borne sur le risque du classificateur par vote de majorité précédemment introduite sous une forme différente. Cette borne suggère que pour construire un vote de majorité de faible risque, le désaccord espéré entre les paires de votants devrait être maximisé, alors que la performance individuelle des votants ne devrait pas être trop grande. Nous avons également étudié le pouvoir prédictif de la C-borne, en évaluant la corrélation entre sa valeur et la valeur du risque du classificateur par vote de majorité, et en utilisant la C-borne comme critère se sélection de modèle pour l'algorithme AdaBoost.

Au chapitre 6, nous avons généralisé les concepts de vote de majorité, de marge de de C-borne aux votes de majorité constitués de votants à sortie arbitrairement complexe. La notion générale de marge étant plus difficile à manipuler en raison d'un terme « max », nous avons également introduit une relaxation de la marge généralisée, nommée la ω-marge. Ce cadre général a ensuite été spécialisé aux cadres de la classification multi-classe et multi-étiquette.

Théorie PAC-bayésienne En collaboration avec d'autres auteurs étudiants, nous avons unifié différents théorèmes PAC-bayésiens de la littérature en introduisant au chapitre 3 un théorème général pouvant être spécialisé pour retrouver ces bornes ou en créer de nouvelles. En introduisant de nouvelles fonctions de perte et en les combinant à ce théorème, nous avons introduit de nouvelles bornes de généralisation PAC-bayésiennes basées sur la C-borne. Celles-

ci prennent donc en considération non seulement la performance individuelle des votants, mais également leur désaccord.

Nous avons développé une nouvelle restriction sur les votes de majorité, permettant d'obtenir des garanties de généralisation PAC-bayésiennes ne dépendant pas de la divergence Kullback-Leibler, ouvrant la voie à la création d'algorithmes d'apprentissage minimisant directement les bornes introduites.

Au chapitre 7, également en collaboration, nous avons introduit un nouveau processus simplifié de démonstration pour les bornes PAC-bayésiennes. Ce processus de preuve est personnalisable, permettant de créer de nouveaux théorèmes PAC-bayésiens plus facilement. Nous avons ainsi créé deux nouvelles familles de bornes. La première dépend de la divergence de Rényi plutôt que de la divergence Kullback-Leibler classique. La seconde est spécialisée au cadre de l'apprentissage transductif, plutôt qu'à l'apprentissage inductif.

Développement d'algorithmes d'apprentissage Au chapitre 4, nous avons introduit un nouvel algorithme d'apprentissage nommé MinCq, qui est rigoureusement justifié théoriquement comme il minimise directement une borne de généralisation PAC-bayésienne sur le risque du classificateur par vote de majorité.

MinCq est un programme quadratique simple, qui construit un vote de majorité maximisant le désaccord entre les votants, tout en restreignant celui-ci à ne pas être trop confiant envers les votants qui ont une bonne performance individuelle. Empiriquement, MinCq est très compétitif avec les algorithmes de l'état de l'art.

Au chapitre 5, nous avons utilisé la théorie de l'optimisation de Lagrange pour créer un algorithme itératif nommé CqBoost, qui ne minimise pas directement une borne de généralisation PAC-bayésienne, mais en est fortement inspiré. Cet algorithme retourne des votes de majorité beaucoup plus parcimonieux que MinCq et n'a pas à résoudre un programme quadratique qui croît en fonction du nombre de votants considérés. Il est donc mieux adapté que MinCq à la situation ou le nombre de votants est très grand, voire infini. CqBoost se compare avantageusement aux algorithmes de l'état de l'art utilisant également l'optimisation de Lagrange.

Au chapitre 7, nous avons utilisé notre théorie PAC-bayésienne transductive comme base pour construire TMinCq, une version transductive de MinCq. Cet algorithme minimise le « vrai » désaccord et non une estimation empirique. Cependant, les résultats empiriques de TMinCq ne sont pas aussi intéressants que nous espérions.

Finalement, en introduisant dans la fonction objectif de TMinCq un terme de régularisation qui oeuvre dans la géométrie intrinsèque des données, nous avons défini l'algorithme LapMinCq. Les résultats empiriques préliminaires très encourageants ouvrent la voie aux travaux futurs dans cette direction.

Annexe A

Résultats mathématiques auxiliaires

Cette annexe contient des énoncés de théorèmes, corollaires et lemmes de la littérature qui sont ajoutés pour en faciliter la consultation pendant la lecture de la thèse, deux théorèmes généralisant des résultats de la thèse mais dont l'énoncé et la démonstration nécessitent l'introduction d'une notation plus complexe, et des démonstrations de théorèmes qui sont trop laborieuses pour être incluses dans le corps de la thèse sans nuire à sa fluidité.

A.1 Théorèmes, corollaires et lemmes de la littérature

Lemme A.1 (Borne de l'union (inégalité de Boole))**.** *Pour tout ensemble dénombrable d'événements* $\{A_1, A_2, \dots, A_n\}$*, nous avons*

$$\Pr\left(\bigcup_{i=1}^{n} A_i\right) \leq \sum_{i=1}^{n} A_i\,.$$

Lemme A.2 (Inégalité de Markov)**.** *Pour toute variable aléatoire* X *telle que* $\mathbf{E}(X) = \mu$*, et pour tout* $a > 0$*, nous avons*

$$\Pr\left(|X| \geq a\right) \leq \frac{\mu}{a}\,.$$

Lemme A.3 (Inégalité de Jensen)**.** *Pour toute variable aléatoire* X *et pour toute fonction convexe* f*, nous avons*

$$f(\mathbf{E}[X]) \leq \mathbf{E}[f(X)]\,.$$

Lemme A.4 (Inégalité de Hölder)**.** *Soit* X *et* Y *deux variables aléatoires, et soit* $r > 0$ *et* $s > 0$ *tels que* $\frac{1}{r} + \frac{1}{s} = 1$*. Alors,*

$$\mathbf{E}\left[|XY|\right] \leq \left(\mathbf{E}\left[|X|^r\right]\right)^{\frac{1}{r}} \left(\mathbf{E}\left[|Y|^s\right]\right)^{\frac{1}{s}}.$$

Lemme A.5 (Inégalité de Cantelli)**.** *Pour toute variable aléatoire* X *telle que* $\mathbf{E}(X) = \mu$ *et* $\mathbf{Var}(X) = \sigma^2$*, et pour tout* $a > 0$*, nous avons*

$$\Pr\left(X - \mu \geq a\right) \leq \frac{\sigma^2}{\sigma^2 + a^2}\,.$$

Démonstration. Observons premièrement que $\Pr\left(X-\mu \geq a\right) \leq \Pr\left(\left[X-\mu+\frac{\sigma^2}{a}\right]^2 \geq \left[a+\frac{\sigma^2}{a}\right]^2\right)$. Appliquons maintenant l'inégalité de Markov (lemme A.2) pour borner cette probabilité. Nous obtenons

$$\begin{aligned}
\Pr\left(\left[X-\mu+\frac{\sigma^2}{a}\right]^2 \geq \left[a+\frac{\sigma^2}{a}\right]^2\right) &\leq \frac{\mathbf{E}\left[X-\mu+\frac{\sigma^2}{a}\right]^2}{\left[a+\frac{\sigma^2}{a}\right]^2} && \text{(Inégalité de Markov)} \\
&= \frac{\mathbf{E}\left(X-\mu\right)^2 + 2\left(\frac{\sigma^2}{a}\right)\mathbf{E}\left(X-\mu\right) + \left(\frac{\sigma^2}{a}\right)^2}{\left[a+\frac{\sigma^2}{a}\right]^2} \\
&= \frac{\sigma^2+\left(\frac{\sigma^2}{a}\right)^2}{\left[a+\frac{\sigma^2}{a}\right]^2} = \frac{\sigma^2\left(1+\frac{\sigma^2}{a^2}\right)}{\left(\sigma^2+a^2\right)\left(1+\frac{\sigma^2}{a^2}\right)} = \frac{\sigma^2}{\sigma^2+a^2},
\end{aligned}$$

comme $\mathbf{E}\left(X-\mu\right)^2 = \mathbf{Var}(X) = \sigma^2$ et $\mathbf{E}\left(X-\mu\right) = \mathbf{E}(X) - \mathbf{E}(X) = 0$. □

Lemme A.6 (Inégalité de Pinsker)**.** *Pour toutes valeurs p et q, nous avons*

$$2(q-p)^2 \leq \Delta_{\mathrm{KL}}(q,p)\,.$$

Notons que la preuve du théorème A.7 (ci-bas) par Cover et Thomas (1991) considère que les distributions de probabilité Q et P sont discrètes, mais leur argument est directement généralisable aux distributions continues.

Théorème A.7. *(Cover et Thomas, 1991, théorème 2.7.2) La divergence Kullback-Leibler $\mathrm{KL}(Q \,\|\, P)$ est convexe par rapport à la paire (Q,P), c'est-à-dire, si (Q_1,P_1) et (Q_2,P_2) sont deux paires de distributions de probabilité, alors*

$$\mathrm{KL}(\,\lambda Q_1 + (1-\lambda)Q_2 \,\|\, \lambda P_1 + (1-\lambda)P_2\,) \leq \lambda\,\mathrm{KL}(Q_1 \,\|\, P_1) + (1-\lambda)\,\mathrm{KL}(Q_2 \,\|\, P_2)\,,$$

pour tout $\lambda \in [0,1]$.

Corollaire A.8. *La fonction $\Delta_{\mathrm{KL}}(q,p)$ de l'équation (3.2), c'est-à-dire, la divergence Kullback-Leibler entre deux distributions de Bernoulli, est convexe.*

Démonstration. Ce corollaire est une conséquence directe du théorème A.7. □

Lemme A.9. *(Maurer, 2004) Soit X une variable aléatoire quelconque de valeur dans $[0,1]$ et de moyenne $\mu = \mathbf{E}(X)$. Notons par $\mathbf{x}$ le vecteur contenant les résultats de n réalisations indépendantes de X. Considérons maintenant une variable aléatoire de Bernoulli X' (de valeur dans $\{0,1\}$) avec probabilité de succès μ (c'est-à-dire, $\Pr(X'=1) = \mu$). Notons par $\mathbf{x}' \in \{0,1\}^n$ le vecteur contenant les résultats de n réalisations indépendantes de X'.*

Si la fonction $f : [0,1]^n \to \mathbb{R}$ est convexe, alors

$$\mathbf{E}\left[f(\mathbf{x})\right] \leq \mathbf{E}\left[f(\mathbf{x}')\right].$$

A.2 Un théorème PAC-bayésien général pour uplets de votants et distributions alignées

Cette annexe présente une inégalité du changement de mesure qui généralise à la fois le lemme 3.15 et le lemme 3.19. Nous présentons également un théorème PAC-bayésien généralisant les théorèmes 3.16 et 3.20. Comme ces généralisations demandent une notation et des idées plus complexes, nous les fournissons en annexe plutôt que dans le texte principal de la thèse. Les versions plus simples, pouvant être déduites de cette annexe, sont présentées dans la thèse avec leur preuve associée.

Soit $\mathcal{F}$ un ensemble symétrique dénombrable de votants à valeur réelle. Rappelons que dans le cas le plus général, $\mathcal{F}$ est symétrique s'il existe une bijection $c : \mathcal{F} \to \mathcal{F}$ telle que $c(f) = -f$ pour tout $f \in \mathcal{F}$. De plus, pour toute distribution Q alignée sur une distribution a priori P sur $\mathcal{F}$, et pour tout $f \in \mathcal{F}$, nous avons

$$Q(f) + Q(c(f)) \;=\; P(f) + P(c(f))\,.$$

Nous devons d'abord définir une nouvelle notation. Soit $\mathbf{k}$ une séquence de longueur $|\mathbf{k}|$ où chaque élément $k_i \in \{1, \ldots, |\mathcal{F}|\}$ est l'index d'un votant. Soit $f_{\mathbf{k}} : \mathcal{X} \to \mathcal{Y}^{|\mathbf{k}|}$ une fonction retournant un $|\mathbf{k}|$-uplet[1] défini par

$$f_{\mathbf{k}}(x) \;\triangleq\; \left\langle f_{k_1}(x), \ldots, f_{k_{|\mathbf{k}|}}(x) \right\rangle .$$

Rappelons que $P^{|\mathbf{k}|}$ et $Q^{|\mathbf{k}|}$ sont les produits cartésiens des distributions de probabilité P et Q. Alors, la, probabilité de tirer un certain $f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}$ est donnée par

$$Q^{|\mathbf{k}|}(f_{\mathbf{k}}) \;\triangleq\; Q(f_{k_1}) \cdot Q(f_{k_2}) \cdot \ldots \cdot Q(f_{k_{|\mathbf{k}|}}) = \prod_{i=1}^{|\mathbf{k}|} Q(f_{k_i})\,.$$

Finalement, pour chaque $f_{\mathbf{k}}$ et chaque $j \in \{0, \ldots, 2^{|\mathbf{k}|}-1\}$, définissons

$$f_{\mathbf{k}}^{[j]}(x) \;\triangleq\; \left\langle f_{k_1}^{(s_1^j)}(x), \ldots, f_{k_{|\mathbf{k}|}}^{(s_{|\mathbf{k}|}^j)}(x) \right\rangle ,$$

où $s_1^j s_2^j \ldots s_{|\mathbf{k}|}^j$ est la représentation binaire du nombre j, et où $f^{(0)} = f$ et $f^{(1)} = c(f)$. Notons que $f_{\mathbf{k}}^{[0]} = f_{\mathbf{k}}$.

Voici maintenant une nouvelle inégalité du changement de mesure sans KL, nécessaire pour démontrer le prochain théorème PAC-bayésien.

1. Un *uplet* est une collection ordonnée de valeurs dont le nombre d'éléments est inconnu. Lorsque le nombre d'éléments est connu, on préfixe le terme avec ce nombre (par exemple un $|\mathbf{k}|$-uplet). Le terme anglophone *tuple* est également souvent utilisé en français.

Théorème A.10 (Inégalité du changement de mesure pour uplets de votants et distributions alignées)**.** *Pour tout ensemble symétrique de votants $\mathcal{F}$, toute distribution P sur $\mathcal{F}$, toute distribution Q alignée sur P, toute séquence $\mathbf{k}$ d'indices de votants d'au moins un élément et toute fonction $\phi : \mathcal{F}^{|\mathbf{k}|} \to \mathbb{R}$ mesurable sur P pour laquelle $\phi\left(f_{\mathbf{k}}^{[j]}\right) = \phi\left(f_{\mathbf{k}}^{[j']}\right)$ pour tous $j, j' \in \{0, \dots, 2^{|\mathbf{k}|}-1\}$, nous avons*

$$\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \phi(f_{\mathbf{k}}) \;\leq\; ln\left(\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{\phi(f_{\mathbf{k}})}\right).$$

Démonstration. Premièrement, notons qu'il est possible de transformer l'espérance sur $Q^{|\mathbf{k}|}$ par une espérance sur $P^{|\mathbf{k}|}$ en utilisant le fait que $\phi\left(f_{\mathbf{k}}^{[j]}\right) = \phi\left(f_{\mathbf{k}}^{[j']}\right)$ pour tous $j, j' \in \{0, \dots, 2^{|\mathbf{k}|}-1\}$ et le fait que la distribution Q est alignée sur P. En effet, nous avons

$$\begin{aligned}
&2^{|\mathbf{k}|} \cdot \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \phi(f_{\mathbf{k}}) \\
&= \int_{\mathcal{F}^{|\mathbf{k}|}} Q^{|\mathbf{k}|}\left(f_{\mathbf{k}}^{[0]}\right) \phi\left(f_{\mathbf{k}}^{[0]}\right) df_{\mathbf{k}} + \dots + \int_{\mathcal{F}^{|\mathbf{k}|}} Q^{|\mathbf{k}|}\left(f_{\mathbf{k}}^{[2^{|\mathbf{k}|}-1]}\right) \phi\left(f_{\mathbf{k}}^{[2^{|\mathbf{k}|}-1]}\right) df_{\mathbf{k}} \\
&= \int_{\mathcal{F}^{|\mathbf{k}|}} Q^{|\mathbf{k}|}\left(f_{\mathbf{k}}^{[0]}\right) \phi\left(f_{\mathbf{k}}\right) df_{\mathbf{k}} + \dots + \int_{\mathcal{F}^{|\mathbf{k}|}} Q^{|\mathbf{k}|}\left(f_{\mathbf{k}}^{[2^{|\mathbf{k}|}-1]}\right) \phi(f_{\mathbf{k}}) df_{\mathbf{k}} \\
&= \int_{\mathcal{F}^{|\mathbf{k}|}} \sum_{j=0}^{2^{|\mathbf{k}|}-1} \left(Q^{|\mathbf{k}|}(f_{\mathbf{k}}^{[j]})\right) \phi(f_{\mathbf{k}}) df_{\mathbf{k}} \\
&= \int_{\mathcal{F}^{|\mathbf{k}|}} \sum_{j=0}^{2^{|\mathbf{k}|}-1} \left(\prod_{i=1}^{k} \left[Q\left(f_{k_i}^{(s_i^j)}\right)\right]\right) \phi(f_{\mathbf{k}}) df_{\mathbf{k}} \qquad \text{(A.1)} \\
&= \int_{\mathcal{F}^{|\mathbf{k}|}} \prod_{i=1}^{k} \left[Q\left(f_{k_i}^{(0)}\right) + Q\left(f_{k_i}^{(1)}\right)\right] \phi(f_{\mathbf{k}}) df_{\mathbf{k}} \qquad \text{(A.2)} \\
&= \int_{\mathcal{F}^{|\mathbf{k}|}} \prod_{i=1}^{k} \left[Q(f_{k_i}) + Q(c(f_{k_i}))\right] \phi(f_{\mathbf{k}}) df_{\mathbf{k}} \\
&= \int_{\mathcal{F}^{|\mathbf{k}|}} \prod_{i=1}^{k} \left[P(f_{k_i}) + P(c(f_{k_i}))\right] \phi(f_{\mathbf{k}}) df_{\mathbf{k}} \\
&\;\;\vdots \\
&= 2^{|\mathbf{k}|} \cdot \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} \phi(f_{\mathbf{k}}),
\end{aligned}$$

où l'égalité entre les lignes (A.1) et (A.2) est plus évidente en constatant que le développement des termes du produit de la ligne (A.2) permet d'obtenir la ligne (A.1).

Le résultat est obtenu en transformant l'espérance sur $Q^{|\mathbf{k}|}$ par une espérance sur $P^{|\mathbf{k}|}$, et en appliquant l'inégalité de Jensen (lemme A.3).

$$\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \phi(f_{\mathbf{k}}) \;=\; \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} \phi(f_{\mathbf{k}}) \;=\; \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} ln\, e^{\phi(f_{\mathbf{k}})} \;\leq\; ln\left(\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{\phi(f_{\mathbf{k}})}\right).$$

□

Théorème A.11 (Théorème PAC-bayésien général pour uplets de votants et distributions alignées)**.** *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$ avec $\mathcal{Y} = \{-1, 1\}$, tout ensemble symétrique $\mathcal{F}$ de votants $\mathcal{X} \to \overline{\mathcal{Y}}$, toute distribution a priori P sur $\mathcal{F}$, toute séquence $\mathbf{k}$ d'indices de votants d'au moins un élément, toute fonction convexe $\Delta : [0,1] \times [0,1] \to \mathbb{R}$, toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}}^{|\mathbf{k}|} \times \mathcal{Y}^{|\mathbf{k}|} \to [0,1]$ pour laquelle $\Delta\left(\mathbb{E}_S^{\mathcal{L}}\left(f_{\mathbf{k}}^{[j]}\right), \mathbb{E}_D^{\mathcal{L}}\left(f_{\mathbf{k}}^{[j]}\right)\right) = \Delta\left(\mathbb{E}_S^{\mathcal{L}}\left(f_{\mathbf{k}}^{[j']}\right), \mathbb{E}_D^{\mathcal{L}}\left(f_{\mathbf{k}}^{[j']}\right)\right)$, pour tous $j, j' \in \{0, \ldots, 2^{|\mathbf{k}|}-1\}$ et tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q alignée sur P,*

$$\Delta\left(\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}), \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right) \;\leq\; \frac{1}{m}\left[ln\left(\frac{1}{\delta} \mathop{\mathbf{E}}_{S \sim D^m} \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)}\right)\right].$$

Démonstration. Cette démonstration suit la plupart des étapes de celle du théorème 3.3. Nous avons que $\mathbf{E}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)}$ est une variable aléatoire non négative. Par l'inégalité de Markov, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$, que

$$\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)} \;\leq\; \frac{1}{\delta} \mathop{\mathbf{E}}_{S \sim D^m} \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)}.$$

Ensuite, en prenant le logarithme de chaque côté de l'inégalité, nous obtenons, toujours avec probabilité $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$, que

$$ln\left[\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)}\right] \;\leq\; ln\left[\frac{1}{\delta} \mathop{\mathbf{E}}_{S \sim D^m} \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)}\right].$$

Maintenant, plutôt que d'utiliser l'inégalité du changement de mesure du lemme 3.2, nous utilisons celle du théorème A.10 sur la partie de gauche de l'inégalité, avec $\phi(f_{\mathbf{k}}) = m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)$. Nous utilisons ensuite l'inégalité de Jensen (lemme A.3), en exploitant la convexité de Δ.

$$\begin{aligned}\forall\, Q \text{ alignée sur } P :\; ln\left[\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)}\right] &\geq m \cdot \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right) \\ &\geq m \cdot \Delta\left(\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}), \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right).\end{aligned}$$

Nous avons alors, avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q alignée sur P, que

$$m \cdot \Delta\left(\mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}), \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim Q^{|\mathbf{k}|}} \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right) \;\leq\; ln\left[\frac{1}{\delta} \mathop{\mathbf{E}}_{S \sim D^m} \mathop{\mathbf{E}}_{f_{\mathbf{k}} \sim P^{|\mathbf{k}|}} e^{m \cdot \Delta\left(\mathbb{E}_S^{\mathcal{L}}(f_{\mathbf{k}}),\, \mathbb{E}_D^{\mathcal{L}}(f_{\mathbf{k}})\right)}\right].$$

Le résultat final est obtenu en divisant par m de chaque côté de l'inégalité. □

A.3 Un théorème PAC-bayésien général pour fonctions de pertes à valeur bornée

Dans cette section, nous généralisons d'abord quelques théorèmes et corollaires du chapitre 3 au cas où les fonctions de perte $\mathcal{L}$ retournent des valeurs dans $[0, B]$ pour une certaine valeur $B > 0$, plutôt que dans $[0, 1]$. Nous utilisons ces résultats au chapitre 6 pour obtenir deux bornes PAC-bayésiennes sur les deux premiers moments de la marge généralisée..

Énonçons et démontrons d'abord un théorème PAC-bayésien général pour ces fonctions de perte B-bornées, qui est une généralisation directe du théorème 3.3.

Théorème A.12 (Théorème PAC-bayésien général pour fonctions de perte à valeur réelle bornée par B)**.** *Pour toute distribution D sur $\mathcal{X} \times \mathcal{Y}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to \overline{\mathcal{Y}}$, pour tout $B > 0$, pour toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0, B]$, pour toute distribution a priori P sur $\mathcal{F}$, pour tout $\delta \in (0, 1]$, et pour toute fonction convexe $\Delta : [0, 1] \times [0, 1] \to \mathbb{R}$, nous avons avec probabilité au moins $1 - \delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

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

où $\mathrm{KL}(Q \,\|\, P)$ est la divergence Kullback-Leibler entre les distributions Q et P, définie à l'équation (3.1).

Démonstration. Cette démonstration est extrêmement similaire à la démonstration du théorème 3.3, à l'exception que nous la généralisons pour permettre l'utilisation de fonctions de perte dont la valeur retournée appartient à $[0, B]$. Commençons d'abord par noter que $\mathbf{E}_{f\sim P} e^{m\cdot\Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)}$ est une variable aléatoire non négative. Par l'inégalité de Markov (lemme A.2), nous avons avec probabilité d'au moins $1 - \delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que

$$\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)} \;\leq\; \frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)}.$$

En prenant le logarithme de chaque côté de l'inégalité, nous obtenons, toujours avec probabilité $1 - \delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que

$$ln\left[\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)}\right] \leq ln\left[\frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)}\right].$$

Nous appliquons maintenant l'inégalité du changement de mesure (lemme 3.2) sur la partie gauche de l'inégalité, avec $\phi(f) = m \cdot \Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)$. Nous utilisons ensuite l'inégalité de Jensen (lemme A.3, en annexe A), en exploitant la convexité de Δ :

$$\begin{aligned}\forall\, Q \text{ sur } \mathcal{F}: \quad ln\left[\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)}\right] &\geq m \cdot \mathop{\mathbf{E}}_{f\sim Q}\Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right) - \mathrm{KL}(Q \,\|\, P)\\ &\geq m\cdot\Delta\left(\frac{1}{B}\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)\right) - \mathrm{KL}(Q \,\|\, P).\end{aligned}$$

Nous avons donc, avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,

$$m \cdot \Delta\left(\frac{1}{B}\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f)\right) - \mathrm{KL}(Q \,\|\, P) \;\leq\; ln\left[\frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)}\right].$$

Le résultat est ensuite obtenu par de simples calculs. □

Nous remarquons que la valeur B se retrouve à deux endroits dans la borne, à l'intérieur de la Δ-fonction qui elle prend toujours des arguments dans $[0,1]$. Nous énonçons maintenant un corollaire similaire au théorème 3.5, qui est une application du théorème précédent, avec $\Delta = \Delta_{\mathrm{KL}}$.

Corollaire A.13. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to \overline{\mathcal{Y}}$, pour tout $B > 0$, pour toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0,B]$, pour toute distribution a priori P sur $\mathcal{F}$ et pour tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

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

Démonstration. Le résultat est obtenu en démarrant du théorème A.12 avec $\Delta(q,p) = \Delta_{\mathrm{KL}}(q,p)$. Le terme à droite de l'inégalité devient donc

$$\frac{1}{m}\left[\mathrm{KL}(Q \,\|\, P) + ln\left(\frac{1}{\delta}\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta_{\mathrm{KL}}\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)}\right)\right].$$

Comme la distribution a priori P est indépendante de S, nous pouvons intervertir les deux espérances. Cette observation ainsi que le lemme 3.4, donnent

$$\mathop{\mathbf{E}}_{S\sim D^m}\mathop{\mathbf{E}}_{f\sim P} e^{m\cdot\Delta_{\mathrm{KL}}\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)} \;=\; \mathop{\mathbf{E}}_{f\sim P}\mathop{\mathbf{E}}_{S\sim D^m} e^{m\cdot\Delta_{\mathrm{KL}}\left(\frac{1}{B}\mathbb{E}_S^{\mathcal{L}}(f), \frac{1}{B}\mathbb{E}_D^{\mathcal{L}}(f)\right)} \;\leq\; \mathop{\mathbf{E}}_{f\sim P} 2\sqrt{m} \;=\; 2\sqrt{m}\,.$$

Notons que le lemme 3.4 est applicable ici, en considérant comme fonction de perte $\mathcal{L}'(\cdot,\cdot) = \frac{1}{B}\mathcal{L}(\cdot,\cdot)$, qui nous ramène à une perte (et une perte espérée) à valeur dans $[0,1]$. □

Le prochain corollaire n'a pas d'équivalent dans le chapitre 3, mais permet de simplifier le passage entre le corollaire A.13 et les bornes sur les deux premiers moments de la marge généralisée du chapitre 6.

Corollaire A.14. *Pour toute distribution D sur $\mathcal{X}\times\mathcal{Y}$, pour tout ensemble $\mathcal{F}$ de votants $\mathcal{X} \to \overline{\mathcal{Y}}$, pour tout $B > 0$, pour toute fonction de perte $\mathcal{L} : \overline{\mathcal{Y}} \times \mathcal{Y} \to [0,B]$, pour toute distribution a priori P sur $\mathcal{F}$ et pour tout $\delta \in (0,1]$, nous avons avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$ que pour toute distribution a posteriori Q sur $\mathcal{F}$,*

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

et

$$\mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_D^{\mathcal{L}}(f) \;\geq\; \mathop{\mathbf{E}}_{f\sim Q}\mathbb{E}_S^{\mathcal{L}}(f) - B\sqrt{\frac{1}{2m}\left[\mathrm{KL}(Q\,\|\,P) + ln\,\frac{2\sqrt{m}}{\delta}\right]}.$$

Démonstration. Le résultat est obtenu en démarrant du théorème A.13, où nous appliquons l'inégalité de Pinsker (lemme A.6), c'est-à-dire

$$2(q-p)^2 \;\leq\; \Delta_{\mathrm{KL}}(q,p)\,.$$

Nous avons donc, avec probabilité au moins $1-\delta$ sur le choix d'un ensemble de données de m exemples $S \sim D^m$, que pour toute distribution a priori Q sur $\mathcal{F}$,

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

Le résultat est obtenu en divisant par deux chaque côté de l'inégalité, puis en prenant la racine carrée des deux côtés, nous donnant ainsi une borne inférieure et une borne supérieure. □

A.4 Démonstrations supplémentaires

Lemme A.15. *Soit $\beta \in [0,1]$, $q \in [0,1]$ et $p \in (0,1)$. Nous avons*

$$\Delta^{\star}_{\beta}(q,p) \;=\; \Delta_{\mathrm{KL}}(q,p) + \frac{1-\beta}{\beta}\,\Delta_{\mathrm{KL}}\left(\frac{p-\beta\,q}{1-\beta},p\right),$$

où $\Delta_{\mathrm{KL}}(\cdot,\cdot)$ et $\Delta^{\star}_{\beta}(\cdot,\cdot)$ sont définis aux équations (3.2) et (7.13), respectivement.

Démonstration.

$$\begin{aligned}
\Delta^{\star}_{\beta}(q,p) \;&=\; q\,ln\,\beta\frac{q}{p} + \left(\frac{p}{\beta}-q\right)ln\left(1-\beta\frac{q}{p}\right) + (1-q)\,ln\,\beta\frac{1-q}{1-p} \\
&\quad + \left(\frac{1-p}{\beta}+q-1\right)ln\left(1-\beta\frac{1-q}{1-p}\right) - ln\,\beta - \left(\frac{1}{\beta}-1\right)ln(1-\beta) \\
&=\; q\,ln\,\frac{q}{p} + q\,ln\,\beta + \left(\frac{p}{\beta}-q\right)ln\left(1-\beta\frac{q}{p}\right) + (1-q)\,ln\,\frac{1-q}{1-p} + (1-q)\,ln\,\beta \\
&\quad + \left(\frac{1-p}{\beta}+q-1\right)ln\left(1-\beta\frac{1-q}{1-p}\right) - ln\,\beta - \left(\frac{1}{\beta}-1\right)ln(1-\beta) \\
&=\; q\,ln\,\frac{q}{p} + \left(\frac{p}{\beta}-q\right)ln\left(1-\beta\frac{q}{p}\right) + (1-q)\,ln\,\frac{1-q}{1-p} + \left(\frac{1-p}{\beta}+q-1\right)ln\left(1-\beta\frac{1-q}{1-p}\right) \\
&\quad - \left(\frac{1}{\beta}-1\right)ln(1-\beta) \\
&=\; \Delta_{\mathrm{KL}}(q,p) + \left(\frac{p}{\beta}-q\right)ln\left(1-\beta\frac{q}{p}\right) + \left(\frac{1-p}{\beta}+q-1\right)ln\left(1-\beta\frac{1-q}{1-p}\right) \\
&\quad - \left(\frac{1}{\beta}-1\right)ln(1-\beta) \\
&=\; \Delta_{\mathrm{KL}}(q,p) + \left(\frac{p}{\beta}-q\right)ln\left(1-\beta\frac{q}{p}\right) + \left(\frac{1-p}{\beta}+q-1\right)ln\left(1-\beta\frac{1-q}{1-p}\right) \\
&\quad - \left[\left(\frac{p}{\beta}-q\right)+\left(\frac{1-p}{\beta}+q-1\right)\right]ln(1-\beta) \\
&=\; \Delta_{\mathrm{KL}}(q,p) + \left(\frac{p}{\beta}-q\right)\left[ln\left(1-\beta\frac{q}{p}\right) - ln(1-\beta)\right] \\
&\quad + \left(\frac{1-p}{\beta}+q-1\right)\left[ln\left(1-\beta\frac{1-q}{1-p}\right) - ln(1-\beta)\right] \\
&=\; \Delta_{\mathrm{KL}}(q,p) + \left(\frac{p}{\beta}-q\right)ln\,\frac{1-\beta\frac{q}{p}}{1-\beta} + \left(\frac{1-p}{\beta}+q-1\right)ln\,\frac{1-\beta\frac{1-q}{1-p}}{1-\beta} \\
&=\; \Delta_{\mathrm{KL}}(q,p) + \frac{1-\beta}{\beta}\left[\frac{p-\beta q}{1-\beta}\,ln\,\frac{1-\beta\frac{q}{p}}{1-\beta} + \left(1-\frac{p-\beta q}{1-\beta}\right)ln\,\frac{1-\beta\frac{1-q}{1-p}}{1-\beta}\right] \\
&=\; \Delta_{\mathrm{KL}}(q,p) + \frac{1-\beta}{\beta}\left[\frac{p-\beta q}{1-\beta}\,ln\,\frac{\frac{p-\beta q}{1-\beta}}{p} + \left(1-\frac{p-\beta q}{1-\beta}\right)ln\,\frac{1-\frac{p-\beta q}{1-\beta}}{1-p}\right] \\
&=\; \Delta_{\mathrm{KL}}(q,p) + \frac{1-\beta}{\beta}\,\Delta_{\mathrm{KL}}\left(\frac{p-\beta\,q}{1-\beta},p\right).
\end{aligned}$$

□

Lemme A.16. *Soit m, z, $\hat{k}$ et λ des entiers tels que $\lambda \leq m \leq z-\lambda$ et $0 \leq \hat{k} \leq z$. Nous avons*

$$F(k) \;=\; \frac{\eta(k,\hat{k})\,\eta(m-k,z-\hat{k})}{\eta(m,z)} \;\leq\; e^{\frac{1}{6\lambda}}\sqrt{2\pi m\left(1-\frac{m}{z}\right)},$$

pour $k = max\left[0,\hat{k}+m-z\right]$ et $k = min\left[m,\hat{k}\right]$.

Démonstration. Premièrement, étudions le cas où $k = max\left[0,\hat{k}+m-z\right]$.

Si $0 \geq \hat{k}+m-z$, alors $F(0) = \frac{\eta(m,z-\hat{k})}{\eta(m,z)}$ croît par rapport à $\hat{k}$, et le maximum est atteint à $\hat{k} = z-m$. Nous avons

$$F(0) \;\leq\; \frac{\eta(m,m)}{\eta(m,z)} \;=\; \frac{1}{\eta(m,z)}\,.$$

Si $0 \leq \hat{k}+m-z$, alors $F(\hat{k}+m-z) = \frac{\eta(\hat{k}+m-z,\hat{k})\,\eta(z-\hat{k},z-\hat{k})}{\eta(m,z)} = \frac{\eta(\hat{k}+m-z,\hat{k})}{\eta(m,z)}$ décroît en fonction de $\hat{k}$, et le maximum est atteint à $\hat{k} = z-m$. Alors

$$F(\hat{k}+m-\hat{k}) \;=\; F(0) \;\leq\; \frac{1}{\eta(m,z)}\,.$$

Maintenant, étudions le cas où $k = min\left[m,\hat{k}\right]$.

Si $m \leq \hat{k}$, alors $F(m) = \frac{\eta(m,\hat{k})}{\eta(m,z)}$ décroît en fonction de $\hat{k}$, et le maximum est atteint à $\hat{k} = m$. Nous avons

$$F(m) \;\leq\; \frac{\eta(m,m)}{\eta(m,z)} \;=\; \frac{1}{\eta(m,z)}\,.$$

Si $m \geq \hat{k}$, alors $F(\hat{k}) = \frac{\eta(\hat{k},\hat{k})\,\eta(m-\hat{k},z-\hat{k})}{\eta(m,z)} = \frac{\eta(m-\hat{k},z-\hat{k})}{\eta(m,z)}$ croît en fonction de $\hat{k}$, et le maximum est atteint à $\hat{k} = m$. Alors

$$F(\hat{k}) \;=\; F(m) \;\leq\; \frac{1}{\eta(m,z)}\,.$$

Finalement, par le lemme 7.3, nous obtenons

$$\frac{1}{\eta(m,z)} \;\leq\; \frac{1}{\sqrt{\frac{z}{2\pi m(z-m)}}e^{-\frac{1}{12m}-\frac{1}{12(z-m)}}} \;=\; \sqrt{2\pi\, m\left(1-\tfrac{m}{z}\right)}e^{\frac{1}{12m}+\frac{1}{12(z-m)}} \;\leq\; e^{\frac{1}{6\lambda}}\sqrt{2\pi m\left(1-\frac{m}{z}\right)}.$$

□

Lemme A.17. *Soit m, z et $\hat{k}$ trois entiers tels que $0 \leq m \leq z$ et $0 \leq \hat{k} \leq z$. Nous avons*

$$\sum_{k\in\mathcal{K}^*_{mz\hat{k}}}\sqrt{\left(\frac{1}{k}+\frac{1}{\hat{k}-k}\right)\left(\frac{1}{m-k}+\frac{1}{(z-\hat{k})-(m-k)}\right)} \;\leq\; 2\sum_{k=1}^{m-1}\frac{1}{k} \tag{A.3}$$

$$\leq\; 2(1+ln(m-1))\,,$$

où

$$\mathcal{K}^*_{mz\hat{k}} \;=\; \left\{max[0,\hat{k}+m-z]+1,\ldots,min[m,\hat{k}]-1\right\}.$$

Nous avons l'égalité à la ligne (A.3) lorsque $m = \hat{k} = z-\hat{k}$.

Démonstration. Examinons d'abord le cas où $m = \hat{k} = z - \hat{k}$.

$$\sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{m-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)}$$

$$= \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{m-k}\right)\left(\frac{1}{m-k} + \frac{1}{m-(m-k)}\right)} = \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{m-k}\right)\left(\frac{1}{m-k} + \frac{1}{k}\right)}$$

$$= \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \left(\frac{1}{k} + \frac{1}{m-k}\right) = \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \frac{1}{k} + \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \frac{1}{m-k} = 2 \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \frac{1}{k} = 2 \sum_{k=1}^{m-1} \frac{1}{k}.$$

L'avant-dernière égalité vient du fait que lorsque $m = \hat{k} = z - \hat{k}$, l'ensemble $\mathcal{K}^*_{mz\hat{k}}$ est égal à $\{1, 2, \ldots, m-1\}$. Les deux sommes sont alors équivalentes.

Examinons maintenant tous les autres cas. Nous distinguons 4 cas distincts, où chaque démonstration consiste à utiliser l'inégalité correspondante pour faire croître la valeur de l'expression.

Cas 1 : $m \leq (z - \hat{k})$ et $m \leq \hat{k}$.

$$\sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{m-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)}$$

$$\leq \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{m-k}\right)\left(\frac{1}{m-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)}$$

$$\leq \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{m-k}\right)\left(\frac{1}{m-k} + \frac{1}{m-(m-k)}\right)} = \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{m-k}\right)\left(\frac{1}{m-k} + \frac{1}{k}\right)}$$

$$= \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \left(\frac{1}{k} + \frac{1}{m-k}\right) = \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \frac{1}{k} + \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \frac{1}{m-k} = \sum_{k=1}^{m-1} \frac{1}{k} + \sum_{k=1}^{m-1} \frac{1}{m-k} = 2 \sum_{k=1}^{m-1} \frac{1}{k}.$$

Cas 2 : $m \leq (z - \hat{k})$ et $m > \hat{k}$.

$$\sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{m-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)}$$

$$\leq \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{\hat{k}-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)}$$

$$\leq \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{\hat{k}-k} + \frac{1}{m-(m-k)}\right)} = \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{\hat{k}-k} + \frac{1}{k}\right)}$$

$$= \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right) = \sum_{k=1}^{\hat{k}-1} \left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right) = \sum_{k=1}^{\hat{k}-1} \frac{1}{k} + \sum_{k=1}^{\hat{k}-1} \frac{1}{\hat{k}-k} = 2 \sum_{k=1}^{\hat{k}-1} \frac{1}{k} < 2 \sum_{k=1}^{m-1} \frac{1}{k}.$$

Cas 3 : $m > (z - \hat{k})$ et $m \leq \hat{k}$.

$$\begin{aligned}
&\sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{m-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)} \\
&\quad \leq \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{m-k}\right)\left(\frac{1}{m-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)} \\
&\quad \leq \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{(z-\hat{k})-(m-k)} + \frac{1}{m-k}\right)\left(\frac{1}{m-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)} \\
&\quad = \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \left(\frac{1}{(z-\hat{k})-(m-k)} + \frac{1}{m-k}\right) = \sum_{k=m-z+\hat{k}+1}^{m-1} \left(\frac{1}{(z-\hat{k})-(m-k)} + \frac{1}{m-k}\right) \\
&\quad = 2 \sum_{k=m-z+\hat{k}+1}^{m-1} \frac{1}{m-k} < 2 \sum_{k=1}^{m-1} \frac{1}{m-k} = 2 \sum_{k=1}^{m-1} \frac{1}{k}.
\end{aligned}$$

Cas 4 : $m > (z - \hat{k})$ et $m > \hat{k}$.

$$\begin{aligned}
&\sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{m-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)} \\
&\quad \leq \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{k} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{\hat{k}-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)} \\
&\quad \leq \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \sqrt{\left(\frac{1}{(z-\hat{k})-(m-k)} + \frac{1}{\hat{k}-k}\right)\left(\frac{1}{\hat{k}-k} + \frac{1}{(z-\hat{k})-(m-k)}\right)} \\
&\quad = \sum_{k \in \mathcal{K}^*_{mz\hat{k}}} \left(\frac{1}{(z-\hat{k})-(m-k)} + \frac{1}{\hat{k}-k}\right) = \sum_{k=m-z+\hat{k}+1}^{\hat{k}-1} \left(\frac{1}{(z-\hat{k})-(m-k)} + \frac{1}{\hat{k}-k}\right) \\
&\quad = 2 \sum_{k=m-z+\hat{k}+1}^{\hat{k}-1} \frac{1}{\hat{k}-k} \leq 2 \sum_{k=1}^{\hat{k}-1} \frac{1}{\hat{k}-k} = 2 \sum_{k=1}^{\hat{k}-1} \frac{1}{k} \leq 2 \sum_{k=1}^{m-1} \frac{1}{k}.
\end{aligned}$$

Pour chaque cas, nous avons montré que l'expression est plus petite ou égale à $2 \sum_{k=1}^{m-1} \frac{1}{k}$. En utilisant la méthode d'approximation par une intégrale, nous obtenons tel que voulu

$$2 \sum_{k=1}^{m-1} \frac{1}{k} \leq 2\left(1 + \int_1^{m-1} \frac{1}{x}\, dx\right) = 2\big(1 + ln(m-1)\big).$$

□

Théorème A.18 (Version corrigée du théorème 18 de Derbeko, El-Yaniv et Meir (2004))**.** *Pour tout ensemble Z de z exemples, pour tout ensemble $\mathcal{H}$ de classificateurs, pour toute distribution a priori P sur $\mathcal{H}$ et pour tout $\delta \in (0, 1]$, nous avons avec probabilité au moins $1 - \delta$ sur le choix d'un ensemble de données de m exemples tirés sans remise parmi Z que pour toute distribution a posteriori Q sur $\mathcal{H}$,*

$$R_Z(G_Q) \leq R_S(G_Q) + \sqrt{\frac{1 - \frac{m}{z}}{2(m-1)}\left[\mathrm{KL}(Q \,\|\, P) + ln\, \frac{m}{\delta} + 7\, ln(z+1)\right]}.$$

Démonstration. Utilisons un raccourci de notation $R_S = R_S(G_Q)$ et $R_Z = R_Z(G_Q)$. Nous démarrons de l'équation (17) de Derbeko, El-Yaniv et Meir (2004) :

$$\Delta_{\mathrm{KL}}(R_S, R_Z) + \frac{1-\frac{m}{z}}{\frac{m}{z}} \Delta_{\mathrm{KL}}\left(\frac{R_Z - \frac{m}{z}R_S}{1-\frac{m}{z}}, R_Z\right) - \frac{7}{m} ln(z+1) \;\leq\; \frac{\mathrm{KL}(Q \,\|\, P) + ln\,\frac{m}{\delta}}{m-1}\,.$$

En appliquant l'inégalité de Pinsker (lemme A.6) deux fois, nous obtenons

$$\begin{aligned}\Delta_{\mathrm{KL}}(R_S, R_Z) + \frac{1-\frac{m}{z}}{\frac{m}{z}} \Delta_{\mathrm{KL}}\left(\frac{R_Z - \frac{m}{z}R_S}{1-\frac{m}{z}}, R_Z\right) &\geq 2(R_S - R_Z)^2 + 2\left(\tfrac{z}{m} - 1\right)\left(\tfrac{R_Z - \frac{m}{z}R_S}{1-\frac{m}{z}} - R_Z\right)^2 \\ &= \frac{2(R_S - R_Z)^2}{1-\frac{m}{z}}.\end{aligned}$$

Le résultat est obtenu en isolant R_Z dans

$$\frac{2(R_S - R_Z)^2}{1-\frac{m}{z}} - \frac{7}{m} ln(z+1) \;\leq\; \frac{\mathrm{KL}(Q \,\|\, P) + ln\,\frac{m}{\delta}}{m-1}\,.$$

□

Notons que dans Derbeko, El-Yaniv et Meir (2004), le résultat est énoncé comme une borne sur $R_U(G_Q)$, c'est-à-dire, une borne sur le risque par rapport aux exemples non étiquetés. Comme

$$R_Z(G_Q) \;=\; \frac{1}{z}\big(mR_S(G_Q) + (z-m)R_U(G_Q)\big),$$

le théorème A.18 ci-haut peut être converti directement d'une borne sur $R_Z(G_Q)$ vers une borne sur $R_U(G_Q)$. Nous avons alors

$$\frac{1}{z}\big(mR_S(G_Q) + (z-m)R_U(G_Q)\big) \;\leq\; R_S(G_Q) + \sqrt{\frac{1-\frac{m}{z}}{2(m-1)}\Big[\mathrm{KL}(Q \,\|\, P) + ln\,\tfrac{m}{\delta} + 7\,ln(z+1)\Big]},$$

et donc

$$R_U(G_Q) \;\leq\; R_S(G_Q) + \sqrt{\frac{1}{2(m-1)\left(1-\frac{m}{z}\right)}\Big[\mathrm{KL}(Q \,\|\, P) + ln\,\tfrac{m}{\delta} + 7\,ln(z+1)\Big]}.$$

Lemme A.19. *Soit Z un ensemble de z exemples $\langle (x_1, y_1), (x_2, y_2), \dots, (x_z, y_z) \rangle$, $f : Z \to \mathbb{R}$ une fonction, $\mathbf{W}$ une matrice symétrique de taille $z \times z$ mesurant la similarité entre chaque paire d'exemples de Z, $\mathbf{D}$ une matrice diagonale de $z \times z$ éléments où $D_{kk} = \sum_l W_{kl}$, et $\mathbf{L} = \mathbf{D} - \mathbf{W}$ le Laplacien du graphe. Nous avons*

$$\frac{1}{2z^2} \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} \left(f(x_k) - f(x_l)\right)^2 = \frac{1}{z^2} \begin{bmatrix} f(x_1) & \dots & f(x_z) \end{bmatrix} \mathbf{L} \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_z) \end{bmatrix}.$$

Démonstration. Le résultat est obtenu par une succession d'opérations arithmétiques simples, fournies ici comme l'égalité rapportée dans Belkin, Niyogi et Sindhwani (2006) est erronée. En effet, nous avons

$$\begin{aligned}
&\frac{1}{z^2} \begin{bmatrix} f(x_1) & \dots & f(x_z) \end{bmatrix} \mathbf{L} \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_z) \end{bmatrix} \\
&= \frac{1}{z^2} \begin{bmatrix} f(x_1) & \dots & f(x_z) \end{bmatrix} (\mathbf{D} - \mathbf{W}) \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_z) \end{bmatrix} \\
&= \frac{1}{z^2} \begin{bmatrix} f(x_1) & \dots & f(x_z) \end{bmatrix} \mathbf{D} \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_z) \end{bmatrix} - \frac{1}{z^2} \begin{bmatrix} f(x_1) & \dots & f(x_z) \end{bmatrix} \mathbf{W} \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_z) \end{bmatrix} \\
&= \frac{1}{z^2} \begin{bmatrix} f(x_1) & \dots & f(x_z) \end{bmatrix} \mathbf{D} \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_z) \end{bmatrix} - \frac{1}{z^2} \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} f(x_k) f(x_l) \\
&= \frac{1}{z^2} \left(\begin{bmatrix} f(x_1) & \dots & f(x_z) \end{bmatrix} \begin{bmatrix} \sum_{l=1}^{z} W_{1l} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sum_{l=1}^{z} W_{zl} \end{bmatrix} \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_z) \end{bmatrix} - \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} f(x_k) f(x_l) \right) \\
&= \frac{1}{z^2} \left(\begin{bmatrix} \sum_{l=1}^{z} W_{1l} f(x_1) & \dots & \sum_{l=1}^{z} W_{zl} f(x_z) \end{bmatrix} \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_z) \end{bmatrix} - \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} f(x_k) f(x_l) \right) \\
&= \frac{1}{z^2} \left(\sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} \left(f(x_k)\right)^2 - \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} f(x_k) f(x_l) \right) \\
&= \frac{1}{2z^2} \left(2 \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} \left(f(x_k)\right)^2 - 2 \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} f(x_k) f(x_l) \right) \\
&= \frac{1}{2z^2} \left(\sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} \left(f(x_k)\right)^2 + \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} \left(f(x_l)\right)^2 - 2 \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} f(x_k) f(x_l) \right) \\
&= \frac{1}{2z^2} \sum_{k=1}^{z} \sum_{l=1}^{z} W_{kl} \left(f(x_k) - f(x_l)\right)^2 .
\end{aligned}$$

□

Annexe B

Informations complémentaires sur les expérimentations

Cette annexe contient de l'information supplémentaire sur les ensembles de données utilisés et les résultats de l'exécution des algorithmes comparés dans cette thèse.

B.1 Ensembles de données

Les ensembles de données suivants sont divisés en différents *contextes*. Le plus général est une collection d'ensembles de données divers, permettant de comparer des algorithmes en fonction de leur capacité à résoudre des tâches diverses. D'autres contextes sont plus spécifiques et permettent de comparer différents algorithmes sur une tâche particulière.

Notons que le texte de présentation des divers contextes est parfois présent dans le texte principal de la thèse, mais est rapporté ici pour rendre cette annexe plus indépendante du texte principal.

B.1.1 Contexte des tâches classiques de classification binaire

Ce contexte est très général : il regroupe de multiples ensembles de données de classification binaire, provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (Lichman, 2013). Ces ensembles de données sont couramment utilisés pour évaluer la performance d'algorithmes d'apprentissage, et permettent de vérifier la performance d'un algorithme d'apprentissage sur plusieurs tâches diverses. Pour chaque ensemble de données, la moitié des exemples (jusqu'à un maximum de 500) est choisie aléatoirement et utilisée comme ensemble d'apprentissage S. L'ensemble de test T contient le reste des exemples. La table B.1 présente ces ensembles de données. Ceux-ci sont utilisés dans les expérimentations des sections 2.4, 4.3 et 5.6.

Nom	$\|S\|$	$\|T\|$	d	Nom	$\|S\|$	$\|T\|$	d
australian	345	345	14	horse	184	184	26
balance	312	313	4	ionosphere	175	176	34
breast	349	350	9	monks	216	216	6
bupa	172	173	6	mushroom	500	7624	22
car	500	1228	6	optdigits	500	3323	64
cmc	500	973	9	pima	384	384	8
credit	345	345	15	titanic	500	1701	3
cylinder	270	270	35	vote	217	218	16
ecoli	168	168	7	wine	89	89	13
glass	107	107	9	yeast	500	984	8
heart	135	135	13	zoo	50	51	16
hepatitis	77	78	19				

TAB. B.1: Ensembles de données du contexte de tâches classiques de classification binaire. L'ensemble S correspond à l'ensemble d'entraînement, l'ensemble T à l'ensemble de test, et d correspond au nombre de dimensions des données.

B.1.2 Contexte de reconnaissance de caractères manuscrits

Ce contexte permet de comparer divers algorithmes dans la tâche de la reconnaissance des caractères manuscrits. Pour cette tâche, nous utilisons l'ensemble de données *MNIST database of handwritten digits* (LECUN et CORTES, 2009), un ensemble de données classique sur lequel se comparent de nombreux algorithmes d'apprentissage. Cet ensemble contenant 10 classes, nous séparons d'abord l'ensemble de données original en 45 tâches de classification binaire, où l'union de tous les ensembles de données binaires retrouve l'ensemble de données original, et où l'intersection entre toute paire d'ensembles de données donne l'ensemble vide. Ainsi, tout exemple de l'ensemble de données original se retrouve dans un et un seul ensemble de données binaires, évitant ainsi d'obtenir une corrélation entre les différents ensembles. Pour chaque ensemble résultant, nous choisissons aléatoirement un ensemble d'entraînement S de 500 exemples, et l'ensemble de test T contient tous les exemples restants. La table B.2 montre le détail des différents ensembles de données produits. Ceux-ci sont utilisés dans les expérimentations de la section 4.3.

B.1.3 Contexte de l'analyse du sentiment de critiques sur Amazon

Ce contexte contient 4 ensembles de données d'analyse du sentiment, représentant des types de produits : *books* (livres), *DVDs*, *electronics* (électronique) et *kitchen appliances* (appareils de cuisine). La tâche est d'apprendre sur les critiques en langage naturel des usagers d'*Amazon.com*, et d'en prédire la *polarité*, qui est soit négative (3 étoiles ou moins) ou positive (4 ou 5 étoiles). Les ensembles de données proviennent de BLITZER, DREDZE et PEREIRA (2007), où les critiques ont déjà été converties en ensembles d'*unigrammes* et de *bigrammes* de termes, avec pour chacun le nombre d'occurrences.

Nom	$\|S\|$	$\|T\|$	d	Nom	$\|S\|$	$\|T\|$	d
MNIST 0-vs-1	500	1142	784	MNIST 2-vs-9	500	1050	784
MNIST 0-vs-2	500	1044	784	MNIST 3-vs-4	500	1052	784
MNIST 0-vs-3	500	1060	784	MNIST 3-vs-5	500	995	784
MNIST 0-vs-4	500	1025	784	MNIST 3-vs-6	500	1058	784
MNIST 0-vs-5	500	968	784	MNIST 3-vs-7	500	1103	784
MNIST 0-vs-6	500	1031	784	MNIST 3-vs-8	500	1052	784
MNIST 0-vs-7	500	1077	784	MNIST 3-vs-9	500	1066	784
MNIST 0-vs-8	500	1025	784	MNIST 4-vs-5	500	959	784
MNIST 0-vs-9	500	1040	784	MNIST 4-vs-6	500	1022	784
MNIST 1-vs-2	500	1151	784	MNIST 4-vs-7	500	1070	784
MNIST 1-vs-3	500	1170	784	MNIST 4-vs-8	500	1017	784
MNIST 1-vs-4	500	1133	784	MNIST 4-vs-9	500	1032	784
MNIST 1-vs-5	500	1077	784	MNIST 5-vs-6	500	966	784
MNIST 1-vs-6	500	1139	784	MNIST 5-vs-7	500	1011	784
MNIST 1-vs-7	500	1187	784	MNIST 5-vs-8	500	960	784
MNIST 1-vs-8	500	1134	784	MNIST 5-vs-9	500	974	784
MNIST 1-vs-9	500	1148	784	MNIST 6-vs-7	500	1074	784
MNIST 2-vs-3	500	1070	784	MNIST 6-vs-8	500	1022	784
MNIST 2-vs-4	500	1036	784	MNIST 6-vs-9	500	1037	784
MNIST 2-vs-5	500	977	784	MNIST 7-vs-8	500	1070	784
MNIST 2-vs-6	500	1041	784	MNIST 7-vs-9	500	1083	784
MNIST 2-vs-7	500	1087	784	MNIST 8-vs-9	500	1031	784
MNIST 2-vs-8	500	1034	784				

Tab. B.2: Ensembles de données du contexte de la reconnaissance de caractères manuscrits. Chaque ensemble de classification binaire est tiré de MNIST, où *MNIST x-vs-y* indique que les algorithmes doivent distinguer le chiffre x du chiffre y. L'ensemble S correspond à l'ensemble d'entraînement, l'ensemble T à l'ensemble de test, et d correspond au nombre de dimensions des données.

Pour chaque ensemble de données, un ensemble d'entraînement de 1000 critiques positives et 1000 critiques négatives est fourni. Les critiques restantes forment l'ensemble de test. L'espace des attributs original de ces ensembles de données est entre 90 000 et 200 000 dimensions. Par contre, comme la plupart des unigrammes et bigrammes ne sont pas significatifs et pour diminuer la dimensionnalité, nous ne considérons que les unigrammes et bigrammes apparaissant au moins 10 fois dans l'ensemble d'entraînement (comme dans Chen, Weinberger et Blitzer (2011)), réduisant ainsi le nombre de dimensions à entre 3500 et 6500. Également comme dans Chen, Weinberger et Blitzer (2011), nous appliquons une pondération *TF-IDF* (Salton et McGill, 1986 ; Dumais et al., 1998), une pratique commune en traitement des langues naturelles. La table B.3 montre le détail des différents ensembles de données. Ceux-ci sont utilisés dans les expérimentations de la section 4.3.

Nom	$\|S\|$	$\|T\|$	d
books	2000	4465	6340
dvd	2000	3586	6075
electronics	2000	5681	4326
kitchen	2000	5945	3766

TAB. B.3: Ensembles de données du contexte de l'analyse du sentiment. L'ensemble S correspond à l'ensemble d'entraînement, l'ensemble T à l'ensemble de test, et d correspond au nombre de dimensions des données.

Name	$\|S\|$	$\|T\|$	d	k
balance	312	313	4	3
car	864	864	6	4
cmc	736	737	9	3
dermatology	183	183	34	6
ecoli	168	168	7	8
flags	97	97	26	8
flare	161	162	10	3
glass	107	107	9	6
iris	75	75	4	3
nursery	6479	6480	8	5
optdigits	1911	1912	64	10
pageblock	2736	2737	10	5
pendigits	3747	3747	16	10
segment	1155	1155	19	7
tae	75	76	5	3
thyroid	1400	1400	27	4
wine	89	89	13	3
yeast	742	742	8	10
zoo	50	51	16	7

TAB. B.4: Ensembles de données du contexte de l'apprentissage multi-classe. L'ensemble S correspond à l'ensemble d'entraînement, l'ensemble T à l'ensemble de test, d correspond au nombre de dimensions des données, et k correspond au nombre de classes.

B.1.4 Contexte de l'apprentissage multi-classe

Tout comme le contexte de tâches classiques de classification binaire, ce contexte est très général. Il regroupe de multiples ensembles de données de classification multi-classe, provenant du dépôt d'ensembles de données d'apprentissage automatique UCI (LICHMAN, 2013). Pour chaque ensemble de données, la moitié des exemples est choisie aléatoirement et utilisée comme ensemble d'apprentissage S. L'ensemble de test T contient le reste des exemples. La table B.4 présente ces ensembles de données. Ceux-ci sont utilisés dans les expérimentations de la section 6.2.

Name	$\|Z\|$	d
car	1728	6
letter-ab	1555	16
mushroom	8124	22
nursery	12959	8
optdigits	3823	64
pageblock	5473	10
pendigits	7494	16
segment	2310	19
spambase	4601	57

TAB. B.5: Ensembles de données utilités pour le calcul de bornes en apprentissage transductif. L'ensemble Z correspond à l'échantillon complet, c'est-à-dire toutes les données. Plusieurs tailles d'ensemble d'entraînement S sont considérées dans les expérimentations. Le valeur d correspond au nombre de dimensions des données.

Nom	$\|S\|$	$\|U\|$	d
bci-100	100	300	117
coil2-100	100	1400	241
digit1-100	100	1400	241
g241c-100	100	1400	241
g241n-100	100	1400	241
usps-100	100	1400	241

TAB. B.6: Ensembles de données du contexte de l'apprentissage transductif. L'ensemble S correspond à l'ensemble d'entraînement, l'ensemble U à l'ensemble des données non étiquetées, et d correspond au nombre de dimensions des données.

B.1.5 Contexte de l'apprentissage transductif

Nous présentons ici deux contextes utilisés dans les expérimentations de la section 7.3. Dans un premier temps, les expérimentations de la section 7.3.7 sont basées sur des ensembles de données avec un grand nombre d'exemples, dont un sous-ensemble est étiqueté. Nous présentons ici le nombre total d'exemples $|Z|$.

Dans un second temps, nous utilisons le cadre d'expérimentation proposé par CHAPELLE, SCHÖLKOPF et ZIEN (2006) pour la comparaison d'algorithmes semi-supervisés ou transductifs. Pour chaque ensemble, 12 ensembles d'entraînement de 10 exemples et 12 ensembles d'entraînement de 100 exemples sont fournis. Les auteurs comparent donc les résultats des différents algorithmes en moyennant le risque sur 12 ensembles d'exemples non étiquetés, en entraînant les algorithmes sur les 10 ou 100 exemples étiquetés. La table B.6 présente ces ensembles de données, que nous utilisons à la section 7.4.2.

B.2 Résultats individuels des algorithmes

Cette section présente les résultats individuels des algorithmes sur chaque ensemble de données sur lequel celui-ci a été exécuté. Nous rapportons le risque sur l'ensemble d'entraînement, le risque de validation croisée, le risque sur l'ensemble de test et les valeurs d'hyperparamètres choisis. Pour chaque algorithme, les hyperparamètres choisis sont ceux qui minimisent le risque de validation croisée parmi une grille de 15 valeurs par hyperparamètre, sur une échelle logarithmique.

Nous présentons dans l'ordre les résultats dans le contexte de l'analyse du sentiment utilisé à la section 4.3, dans le contexte de la reconnaissance de caractères manuscrits utilisé à la section 4.3, et finalement dans le contexte de la résolution de tâches classiques d'apprentissage automatique utilisé aux sections 4.3 et 5.6.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	μ
books	0.0240	0.1655	0.1566	204.44	0.000720
dvd	0.0790	0.1725	0.1623	239.30	0.000100
electronics	0.0055	0.1260	0.1320	135.31	0.000139
kitchen	0.0080	0.1180	0.1156	135.99	0.000193

TAB. B.7: Résultats de MinCq avec noyaux linéaires comme votants, avec μ choisi parmi 15 valeurs entre 10^{-4} et 10^{-2}, dans le contexte de l'analyse du sentiment.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	C
books	0.0225	0.1635	0.1601	245.91	1.000000
dvd	0.0165	0.1775	0.1670	247.12	1.000000
electronics	0.0155	0.1430	0.1347	100.80	1.000000
kitchen	0.0160	0.1115	0.1194	106.19	1.000000

TAB. B.8: Résultats de SVM avec noyau linéaire, avec C choisi parmi 15 valeurs entre 10^{-4} et 10^4, dans le contexte de l'analyse du sentiment.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	γ	μ
MNIST 0-vs-1	0.0000	0.0000	0.0035	4.19	0.372759	0.000373
MNIST 0-vs-2	0.0000	0.0060	0.0057	4.88	0.002683	0.000720
MNIST 0-vs-3	0.0000	0.0020	0.0047	4.41	0.006105	0.000720
MNIST 0-vs-4	0.0000	0.0000	0.0049	4.05	0.006105	0.001000
MNIST 0-vs-5	0.0000	0.0000	0.0072	10.85	0.848343	0.002683
MNIST 0-vs-6	0.0040	0.0050	0.0252	29.39	0.848343	0.010000
MNIST 0-vs-7	0.0000	0.0020	0.0065	3.04	0.006105	0.000518
MNIST 0-vs-8	0.0000	0.0080	0.0107	5.06	0.006105	0.001931
MNIST 0-vs-9	0.0000	0.0080	0.0067	4.41	0.848343	0.000373
MNIST 1-vs-2	0.0000	0.0000	0.0226	23.29	0.372759	0.007197
MNIST 1-vs-3	0.0000	0.0000	0.0222	10.20	0.071969	0.007197
MNIST 1-vs-4	0.0000	0.0080	0.0044	5.14	0.006105	0.000720
MNIST 1-vs-5	0.0000	0.0020	0.0093	3.36	0.031623	0.000373
MNIST 1-vs-6	0.0000	0.0000	0.0053	3.03	0.002683	0.000373
MNIST 1-vs-7	0.0000	0.0080	0.0076	3.85	0.013895	0.000720
MNIST 1-vs-8	0.0020	0.0080	0.0053	4.49	0.006105	0.000720
MNIST 1-vs-9	0.0000	0.0060	0.0035	4.87	0.031623	0.000518
MNIST 2-vs-3	0.0000	0.0140	0.0206	4.65	0.372759	0.000373
MNIST 2-vs-4	0.0000	0.0100	0.0019	3.75	0.372759	0.000373
MNIST 2-vs-5	0.0000	0.0060	0.0123	4.96	0.013895	0.000518
MNIST 2-vs-6	0.0000	0.0100	0.0154	4.36	0.006105	0.000720
MNIST 2-vs-7	0.0000	0.0150	0.0258	11.83	0.071969	0.010000
MNIST 2-vs-8	0.0000	0.0120	0.0126	4.03	0.013895	0.000518
MNIST 2-vs-9	0.0000	0.0080	0.0086	4.11	0.006105	0.000373
MNIST 3-vs-4	0.0000	0.0000	0.0095	12.40	0.071969	0.010000
MNIST 3-vs-5	0.0000	0.0180	0.0302	2.80	0.006105	0.000518
MNIST 3-vs-6	0.0000	0.0000	0.0028	12.80	0.163789	0.002683
MNIST 3-vs-7	0.0000	0.0100	0.0154	25.13	0.372759	0.010000
MNIST 3-vs-8	0.0000	0.0200	0.0295	4.08	0.006105	0.000139
MNIST 3-vs-9	0.0000	0.0000	0.0094	13.03	0.071969	0.010000
MNIST 4-vs-5	0.0000	0.0000	0.0073	10.32	0.031623	0.010000
MNIST 4-vs-6	0.0000	0.0040	0.0059	3.55	0.006105	0.000720
MNIST 4-vs-7	0.0000	0.0080	0.0131	5.02	0.006105	0.000720
MNIST 4-vs-8	0.0000	0.0040	0.0069	4.10	0.006105	0.000518
MNIST 4-vs-9	0.0000	0.0240	0.0242	4.70	0.013895	0.001389
MNIST 5-vs-6	0.0000	0.0120	0.0114	3.49	0.163789	0.000373
MNIST 5-vs-7	0.0000	0.0040	0.0049	4.11	4.393971	0.000373
MNIST 5-vs-8	0.0000	0.0200	0.0198	5.41	0.013895	0.000518
MNIST 5-vs-9	0.0000	0.0140	0.0103	4.22	0.006105	0.000373
MNIST 6-vs-7	0.0000	0.0000	0.0019	4.07	0.031623	0.000518
MNIST 6-vs-8	0.0000	0.0040	0.0137	5.16	0.848343	0.000373
MNIST 6-vs-9	0.0000	0.0020	0.0000	4.31	0.013895	0.001000
MNIST 7-vs-8	0.0000	0.0080	0.0093	3.36	0.002683	0.000373
MNIST 7-vs-9	0.0120	0.0160	0.0517	3.95	0.006105	0.007197
MNIST 8-vs-9	0.0000	0.0180	0.0262	3.76	0.002683	0.000373

TAB. B.9: Résultats de MinCq avec noyaux RBF comme votants, avec μ choisi parmi 15 valeurs entre 10^{-4} et 10^{-2} et γ choisi parmi 15 valeurs entre 10^{-4} et 10^{1}, dans le contexte de la reconnaissance de caractère manuscrits.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	C	γ
MNIST 0-vs-1	0.0000	0.0020	0.0018	1.44	51.794747	0.001179
MNIST 0-vs-2	0.0000	0.0060	0.0048	2.60	51.794747	0.013895
MNIST 0-vs-3	0.0000	0.0020	0.0057	2.90	193.069773	0.006105
MNIST 0-vs-4	0.0000	0.0000	0.0078	3.30	51.794747	0.001179
MNIST 0-vs-5	0.0000	0.0080	0.0093	3.00	193.069773	0.002683
MNIST 0-vs-6	0.0000	0.0120	0.0107	2.05	51.794747	0.006105
MNIST 0-vs-7	0.0000	0.0020	0.0074	3.21	51.794747	0.002683
MNIST 0-vs-8	0.0000	0.0100	0.0146	2.90	51.794747	0.013895
MNIST 0-vs-9	0.0020	0.0100	0.0096	2.99	1.000000	0.006105
MNIST 1-vs-2	0.0000	0.0060	0.0087	2.87	193.069773	0.013895
MNIST 1-vs-3	0.0000	0.0020	0.0120	1.80	193.069773	0.001179
MNIST 1-vs-4	0.0020	0.0080	0.0044	3.13	13.894955	0.000518
MNIST 1-vs-5	0.0000	0.0020	0.0121	3.37	3.727594	0.002683
MNIST 1-vs-6	0.0000	0.0000	0.0070	3.25	13.894955	0.000100
MNIST 1-vs-7	0.0000	0.0080	0.0084	3.23	51.794747	0.006105
MNIST 1-vs-8	0.0000	0.0140	0.0062	3.08	193.069773	0.013895
MNIST 1-vs-9	0.0000	0.0100	0.0061	3.68	51.794747	0.013895
MNIST 2-vs-3	0.0000	0.0220	0.0159	3.03	51.794747	0.013895
MNIST 2-vs-4	0.0000	0.0140	0.0068	2.74	51.794747	0.013895
MNIST 2-vs-5	0.0000	0.0060	0.0102	3.52	51.794747	0.013895
MNIST 2-vs-6	0.0000	0.0080	0.0115	3.46	193.069773	0.013895
MNIST 2-vs-7	0.0000	0.0140	0.0129	2.84	51.794747	0.013895
MNIST 2-vs-8	0.0000	0.0120	0.0145	3.43	193.069773	0.006105
MNIST 2-vs-9	0.0000	0.0140	0.0076	2.85	51.794747	0.006105
MNIST 3-vs-4	0.0000	0.0000	0.0086	3.85	51.794747	0.013895
MNIST 3-vs-5	0.0000	0.0200	0.0312	3.17	51.794747	0.013895
MNIST 3-vs-6	0.0000	0.0060	0.0047	3.37	51.794747	0.006105
MNIST 3-vs-7	0.0000	0.0080	0.0145	2.80	193.069773	0.002683
MNIST 3-vs-8	0.0000	0.0200	0.0257	2.96	193.069773	0.013895
MNIST 3-vs-9	0.0000	0.0180	0.0113	3.66	3.727594	0.006105
MNIST 4-vs-5	0.0000	0.0120	0.0094	3.72	51.794747	0.013895
MNIST 4-vs-6	0.0000	0.0040	0.0059	3.26	51.794747	0.013895
MNIST 4-vs-7	0.0000	0.0080	0.0112	2.78	193.069773	0.006105
MNIST 4-vs-8	0.0000	0.0040	0.0079	3.12	51.794747	0.013895
MNIST 4-vs-9	0.0000	0.0240	0.0233	3.50	193.069773	0.006105
MNIST 5-vs-6	0.0000	0.0100	0.0176	1.57	193.069773	0.006105
MNIST 5-vs-7	0.0000	0.0080	0.0079	2.38	51.794747	0.006105
MNIST 5-vs-8	0.0000	0.0220	0.0208	2.84	51.794747	0.013895
MNIST 5-vs-9	0.0000	0.0200	0.0144	3.76	1.000000	0.006105
MNIST 6-vs-7	0.0000	0.0000	0.0009	2.48	51.794747	0.006105
MNIST 6-vs-8	0.0000	0.0040	0.0166	3.41	193.069773	0.001179
MNIST 6-vs-9	0.0000	0.0020	0.0029	2.45	51.794747	0.002683
MNIST 7-vs-8	0.0000	0.0100	0.0093	2.82	193.069773	0.006105
MNIST 7-vs-9	0.0000	0.0180	0.0332	3.11	51.794747	0.013895
MNIST 8-vs-9	0.0000	0.0180	0.0252	3.39	193.069773	0.002683

TAB. B.10: Résultats de SVM avec noyau RBF, avec C choisi parmi 15 valeurs entre 10^{-4} et 10^4 et γ choisi parmi 15 valeurs entre 10^{-4} et 10^1, dans le contexte de la reconnaissance de caractère manuscrits.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	μ
MNIST 0-vs-1	0.0000	0.0000	0.0035	404.72	0.043940
MNIST 0-vs-2	0.0000	0.0080	0.0201	278.54	0.012798
MNIST 0-vs-3	0.0020	0.0060	0.0142	808.37	0.092106
MNIST 0-vs-4	0.0000	0.0020	0.0098	1096.69	0.150859
MNIST 0-vs-5	0.0040	0.0140	0.0155	805.64	0.092106
MNIST 0-vs-6	0.0000	0.0080	0.0194	664.57	0.092106
MNIST 0-vs-7	0.0000	0.0040	0.0139	817.84	0.056234
MNIST 0-vs-8	0.0040	0.0140	0.0224	628.74	0.092106
MNIST 0-vs-9	0.0100	0.0100	0.0183	1212.96	0.193070
MNIST 1-vs-2	0.0020	0.0260	0.0243	754.59	0.071969
MNIST 1-vs-3	0.0000	0.0040	0.0137	647.14	0.071969
MNIST 1-vs-4	0.0000	0.0080	0.0062	724.05	0.092106
MNIST 1-vs-5	0.0000	0.0120	0.0204	648.01	0.092106
MNIST 1-vs-6	0.0000	0.0020	0.0061	682.84	0.092106
MNIST 1-vs-7	0.0060	0.0140	0.0126	952.92	0.092106
MNIST 1-vs-8	0.0060	0.0460	0.0282	573.15	0.071969
MNIST 1-vs-9	0.0000	0.0080	0.0026	722.27	0.071969
MNIST 2-vs-3	0.0040	0.0340	0.0374	1029.41	0.071969
MNIST 2-vs-4	0.0140	0.0280	0.0145	1175.59	0.117877
MNIST 2-vs-5	0.0020	0.0240	0.0389	1077.07	0.092106
MNIST 2-vs-6	0.0000	0.0180	0.0375	927.30	0.071969
MNIST 2-vs-7	0.0000	0.0180	0.0322	917.30	0.056234
MNIST 2-vs-8	0.0180	0.0340	0.0406	1236.04	0.092106
MNIST 2-vs-9	0.0000	0.0260	0.0162	885.23	0.071969
MNIST 3-vs-4	0.0000	0.0080	0.0133	870.70	0.071969
MNIST 3-vs-5	0.0000	0.0500	0.0784	862.68	0.043940
MNIST 3-vs-6	0.0060	0.0140	0.0151	1322.36	0.150859
MNIST 3-vs-7	0.0000	0.0160	0.0172	497.39	0.012798
MNIST 3-vs-8	0.0100	0.0420	0.0428	863.30	0.071969
MNIST 3-vs-9	0.0000	0.0260	0.0178	763.08	0.056234
MNIST 4-vs-5	0.0060	0.0240	0.0188	647.22	0.092106
MNIST 4-vs-6	0.0000	0.0080	0.0137	776.41	0.034333
MNIST 4-vs-7	0.0020	0.0180	0.0262	1010.80	0.092106
MNIST 4-vs-8	0.0000	0.0060	0.0177	291.96	0.010000
MNIST 4-vs-9	0.0080	0.0560	0.0562	959.23	0.056234
MNIST 5-vs-6	0.0000	0.0200	0.0290	785.79	0.056234
MNIST 5-vs-7	0.0040	0.0080	0.0138	1028.77	0.092106
MNIST 5-vs-8	0.0120	0.0720	0.0521	963.55	0.071969
MNIST 5-vs-9	0.0020	0.0320	0.0195	772.66	0.071969
MNIST 6-vs-7	0.0000	0.0000	0.0028	645.30	0.026827
MNIST 6-vs-8	0.0020	0.0080	0.0186	1326.42	0.150859
MNIST 6-vs-9	0.0000	0.0020	0.0058	657.82	0.034333
MNIST 7-vs-8	0.0000	0.0120	0.0140	870.39	0.071969
MNIST 7-vs-9	0.0100	0.0440	0.0683	877.27	0.071969
MNIST 8-vs-9	0.0000	0.0200	0.0330	721.43	0.056234

TAB. B.11: Résultats de MinCq avec souches de décision comme votants, avec μ choisi parmi 15 valeurs entre 10^{-2} et $10^{-0.5}$, dans le contexte de la reconnaissance de caractère manuscrits.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	t
MNIST 0-vs-1	0.0000	0.0040	0.0070	27508.66	10000
MNIST 0-vs-2	0.0000	0.0140	0.0182	57075.46	10000
MNIST 0-vs-3	0.0000	0.0120	0.0142	56404.67	372
MNIST 0-vs-4	0.0000	0.0020	0.0078	55665.06	193
MNIST 0-vs-5	0.0000	0.0080	0.0207	183409.94	19306
MNIST 0-vs-6	0.0000	0.0160	0.0194	29696.81	10000
MNIST 0-vs-7	0.0000	0.0080	0.0084	56650.47	10000
MNIST 0-vs-8	0.0000	0.0180	0.0205	110371.88	19306
MNIST 0-vs-9	0.0000	0.0120	0.0115	53755.14	37275
MNIST 1-vs-2	0.0000	0.0180	0.0243	185645.73	193
MNIST 1-vs-3	0.0000	0.0100	0.0137	51376.57	71968
MNIST 1-vs-4	0.0000	0.0100	0.0097	54159.83	37275
MNIST 1-vs-5	0.0000	0.0040	0.0260	129533.80	1389
MNIST 1-vs-6	0.0000	0.0000	0.0079	36056.42	372
MNIST 1-vs-7	0.0000	0.0100	0.0185	79492.27	10000
MNIST 1-vs-8	0.0000	0.0420	0.0265	169014.98	193
MNIST 1-vs-9	0.0000	0.0120	0.0061	56252.50	10000
MNIST 2-vs-3	0.0000	0.0480	0.0458	193997.24	19306
MNIST 2-vs-4	0.0000	0.0440	0.0280	189088.08	100
MNIST 2-vs-5	0.0000	0.0360	0.0287	200753.05	100
MNIST 2-vs-6	0.0000	0.0220	0.0375	205582.44	71968
MNIST 2-vs-7	0.0000	0.0360	0.0258	201024.59	10000
MNIST 2-vs-8	0.0000	0.0460	0.0522	204357.85	372
MNIST 2-vs-9	0.0000	0.0340	0.0190	208598.03	37275
MNIST 3-vs-4	0.0000	0.0100	0.0190	201668.42	19306
MNIST 3-vs-5	0.0000	0.0700	0.0754	202653.83	193
MNIST 3-vs-6	0.0000	0.0200	0.0189	209692.83	100
MNIST 3-vs-7	0.0000	0.0120	0.0236	172507.34	19306
MNIST 3-vs-8	0.0000	0.0500	0.0485	180298.22	19306
MNIST 3-vs-9	0.0000	0.0300	0.0310	205681.99	71968
MNIST 4-vs-5	0.0000	0.0240	0.0229	180697.47	1389
MNIST 4-vs-6	0.0000	0.0180	0.0225	162841.56	100
MNIST 4-vs-7	0.0000	0.0220	0.0299	204803.03	100
MNIST 4-vs-8	0.0000	0.0160	0.0187	182887.64	193
MNIST 4-vs-9	0.0000	0.0720	0.0620	168014.20	719
MNIST 5-vs-6	0.0000	0.0260	0.0342	204589.81	2682
MNIST 5-vs-7	0.0000	0.0200	0.0188	199319.92	1389
MNIST 5-vs-8	0.0000	0.0460	0.0552	186404.76	719
MNIST 5-vs-9	0.0000	0.0460	0.0318	177608.33	100
MNIST 6-vs-7	0.0000	0.0020	0.0065	59598.62	19306
MNIST 6-vs-8	0.0000	0.0120	0.0254	59614.07	37275
MNIST 6-vs-9	0.0000	0.0040	0.0068	43118.38	10000
MNIST 7-vs-8	0.0000	0.0200	0.0196	207445.04	719
MNIST 7-vs-9	0.0000	0.0480	0.0720	169512.75	719
MNIST 8-vs-9	0.0000	0.0360	0.0436	177432.72	71968

TAB. B.12: Résultats de AdaBoost avec souches de décision comme votants, avec un nombre d'itérations t choisi parmi 15 valeurs entre 10^2 et 10^6, dans le contexte de la reconnaissance de caractère manuscrits.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	γ	μ
australian	0.1217	0.1246	0.1391	0.90	0.006105	0.007197
balance	0.0160	0.0640	0.0543	0.65	0.848343	0.001000
breast	0.0229	0.0286	0.0401	0.83	0.031623	0.003728
bupa	0.2035	0.2676	0.2543	0.62	0.163789	0.002683
car	0.0060	0.0660	0.0676	1.30	0.372759	0.001389
cmc	0.2680	0.2860	0.3032	1.70	0.163789	0.010000
credit	0.0725	0.1159	0.1217	0.77	0.372759	0.007197
cylinder	0.0000	0.2037	0.2185	0.52	0.163789	0.000100
ecoli	0.0179	0.0237	0.0714	0.30	0.163789	0.000720
glass	0.0561	0.1597	0.2243	0.19	0.372759	0.001389
heart	0.1333	0.1481	0.1556	0.24	0.000518	0.000373
hepatitis	0.0000	0.1675	0.1558	0.14	0.071969	0.001931
horse	0.1250	0.1309	0.1902	0.35	0.000518	0.000139
ionosphere	0.0114	0.1022	0.0971	0.32	0.071969	0.000193
monks	0.1019	0.1944	0.2454	0.39	0.163789	0.000100
mushroom	0.0020	0.0080	0.0102	1.21	0.071969	0.000518
optdigits	0.0000	0.0220	0.0343	1.18	1.930698	0.000373
pima	0.1927	0.2161	0.2448	0.85	0.031623	0.000268
titanic	0.2300	0.2180	0.2222	1.22	0.002683	0.001000
vote	0.0138	0.0368	0.0461	0.41	0.031623	0.000720
wine	0.0000	0.0111	0.0449	0.26	0.848343	0.001000
yeast	0.2560	0.2800	0.2876	1.37	0.163789	0.001931
zoo	0.0000	0.0000	0.0392	0.09	0.031623	0.001000

TAB. B.13: Résultats de MinCq avec noyaux RBF comme votants, avec μ choisi parmi 15 valeurs entre 10^{-4} et 10^{-2} et γ choisi parmi 15 valeurs entre 10^{-4} et 10^{1}, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	γ	C
australian	0.1391	0.1246	0.1333	0.19	0.001179	3.727594
balance	0.0000	0.0480	0.0319	0.17	0.163789	2682.695795
breast	0.0229	0.0229	0.0372	0.20	0.372759	0.071969
bupa	0.1919	0.2271	0.2775	0.10	0.006105	2682.695795
car	0.0100	0.0300	0.0301	0.40	0.013895	10000.000000
cmc	0.2840	0.2960	0.3207	0.27	0.000228	2682.695795
credit	0.0899	0.1217	0.1246	0.18	0.031623	13.894955
cylinder	0.0000	0.2074	0.2704	0.16	0.031623	719.685673
ecoli	0.0179	0.0237	0.0893	0.08	0.031623	3.727594
glass	0.1589	0.1792	0.2056	0.05	0.372759	1.000000
heart	0.1481	0.1556	0.1481	0.07	0.071969	0.268270
hepatitis	0.0897	0.1675	0.2078	0.04	0.031623	3.727594
horse	0.1304	0.1362	0.1685	0.10	0.000518	13.894955
ionosphere	0.0341	0.0565	0.0686	0.09	0.163789	1.000000
monks	0.1065	0.1574	0.2130	0.13	0.013895	719.685673
mushroom	0.0000	0.0060	0.0059	0.29	0.031623	719.685673
optdigits	0.0000	0.0260	0.0340	0.34	0.031623	193.069773
pima	0.1745	0.2134	0.2552	0.20	0.163789	3.727594
titanic	0.2180	0.2180	0.2146	0.18	0.031623	2682.695795
vote	0.0367	0.0368	0.0507	0.12	0.002683	13.894955
wine	0.0112	0.0111	0.0562	0.04	0.006105	51.794747
yeast	0.2580	0.2740	0.2815	0.30	0.006105	719.685673
zoo	0.0000	0.0000	0.0588	0.03	0.031623	193.069773

TAB. B.14: Résultats de SVM avec noyau RBF, avec C choisi parmi 15 valeurs entre 10^{-4} et 10^4 et γ choisi parmi 15 valeurs entre 10^{-4} et 10^1, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	μ
australian	0.0899	0.1159	0.1449	0.20	0.056234
balance	0.0288	0.0673	0.0288	0.12	0.071969
breast	0.0229	0.0314	0.0372	0.22	0.316228
bupa	0.3837	0.2353	0.4509	0.27	0.193070
car	0.1020	0.1180	0.1409	0.10	0.092106
cmc	0.2760	0.3020	0.3052	0.13	0.020962
credit	0.1188	0.1449	0.1275	0.21	0.071969
cylinder	0.0444	0.2556	0.2889	0.62	0.016379
ecoli	0.0357	0.0654	0.0774	0.24	0.150859
glass	0.0374	0.2996	0.2710	0.12	0.043940
heart	0.0963	0.1630	0.1704	0.17	0.092106
hepatitis	0.1154	0.2183	0.1688	0.26	0.247091
horse	0.0870	0.1797	0.2228	0.49	0.071969
ionosphere	0.0000	0.1248	0.1314	0.61	0.043940
monks	0.2546	0.3195	0.2361	0.04	0.012798
mushroom	0.0000	0.0080	0.0042	0.42	0.010000
optdigits	0.0140	0.0680	0.0849	1.69	0.056234
pima	0.1797	0.2109	0.2422	0.29	0.092106
titanic	0.2300	0.2140	0.2116	0.08	0.247091
vote	0.0367	0.0368	0.0507	0.12	0.026827
wine	0.0000	0.0111	0.0562	0.17	0.150859
yeast	0.2820	0.2980	0.3028	0.12	0.056234
zoo	0.0000	0.0200	0.0392	0.20	0.043940

TAB. B.15: Résultats de MinCq avec souches de décision comme votants, avec μ choisi parmi 15 valeurs entre 10^{-2} et $10^{-0.5}$, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	t
australian	0.0957	0.1478	0.1942	241.78	193
balance	0.0032	0.0193	0.0256	134.03	19306
breast	0.0086	0.0543	0.0430	1797.26	100
bupa	0.1570	0.2792	0.3006	113.65	100
car	0.1060	0.1060	0.1336	186.08	100
cmc	0.2720	0.3000	0.3176	232.92	19306
credit	0.1159	0.1536	0.1681	255.66	100
cylinder	0.0000	0.2519	0.2852	2318.75	1389
ecoli	0.0060	0.0891	0.0833	685.25	100
glass	0.0374	0.2446	0.2056	427.75	100
heart	0.0444	0.1630	0.2148	900.88	100
hepatitis	0.0000	0.1933	0.1948	762.85	100
horse	0.0380	0.1955	0.1957	1909.01	100
ionosphere	0.0000	0.1195	0.1200	2666.59	193
monks	0.2685	0.3285	0.2546	119.91	10000
mushroom	0.0000	0.0060	0.0043	5522.31	193
optdigits	0.0000	0.0740	0.0834	14579.93	372
pima	0.1823	0.2342	0.2708	185.54	100
titanic	0.2300	0.2280	0.2222	127.63	10000
vote	0.0000	0.0414	0.0507	2012.60	193
wine	0.0000	0.0346	0.0449	115.74	100
yeast	0.2520	0.3260	0.2988	218.70	100
zoo	0.0000	0.0200	0.1176	134.66	10000

TAB. B.16: Résultats de AdaBoost avec souches de décision comme votants, avec un nombre d'itérations t choisi parmi 15 valeurs entre 10^2 et 10^6, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	μ
australian	0.0899	0.1159	0.1449	0.20	0.056234
balance	0.0288	0.0673	0.0288	0.12	0.071969
breast	0.0229	0.0314	0.0372	0.22	0.316228
bupa	0.3837	0.2353	0.4509	0.27	0.193070
car	0.1020	0.1180	0.1409	0.10	0.092106
cmc	0.2760	0.3020	0.3052	0.13	0.020962
credit	0.1188	0.1449	0.1275	0.21	0.071969
cylinder	0.0444	0.2556	0.2889	0.62	0.016379
ecoli	0.0357	0.0654	0.0774	0.24	0.150859
glass	0.0374	0.2996	0.2710	0.12	0.043940
heart	0.0963	0.1630	0.1704	0.17	0.092106
hepatitis	0.1154	0.2183	0.1688	0.26	0.247091
horse	0.0870	0.1797	0.2228	0.49	0.071969
ionosphere	0.0000	0.1248	0.1314	0.61	0.043940
monks	0.2546	0.3195	0.2361	0.04	0.012798
mushroom	0.0000	0.0080	0.0042	0.42	0.010000
optdigits	0.0140	0.0680	0.0849	1.69	0.056234
pima	0.1797	0.2109	0.2422	0.29	0.092106
titanic	0.2300	0.2140	0.2116	0.08	0.247091
vote	0.0367	0.0368	0.0507	0.12	0.026827
wine	0.0000	0.0111	0.0562	0.17	0.150859
yeast	0.2820	0.2980	0.3028	0.12	0.056234
zoo	0.0000	0.0200	0.0392	0.20	0.043940

TAB. B.17: Résultats de MinCq avec souches de décision comme votants, avec μ choisi parmi 15 valeurs entre 10^{-2} et $10^{-0.5}$, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	t
australian	0.0957	0.1478	0.1942	241.78	193
balance	0.0032	0.0193	0.0256	134.03	19306
breast	0.0086	0.0543	0.0430	1797.26	100
bupa	0.1570	0.2792	0.3006	113.65	100
car	0.1060	0.1060	0.1336	186.08	100
cmc	0.2720	0.3000	0.3176	232.92	19306
credit	0.1159	0.1536	0.1681	255.66	100
cylinder	0.0000	0.2519	0.2852	2318.75	1389
ecoli	0.0060	0.0891	0.0833	685.25	100
glass	0.0374	0.2446	0.2056	427.75	100
heart	0.0444	0.1630	0.2148	900.88	100
hepatitis	0.0000	0.1933	0.1948	762.85	100
horse	0.0380	0.1955	0.1957	1909.01	100
ionosphere	0.0000	0.1195	0.1200	2666.59	193
monks	0.2685	0.3285	0.2546	119.91	10000
mushroom	0.0000	0.0060	0.0043	5522.31	193
optdigits	0.0000	0.0740	0.0834	14579.93	372
pima	0.1823	0.2342	0.2708	185.54	100
titanic	0.2300	0.2280	0.2222	127.63	10000
vote	0.0000	0.0414	0.0507	2012.60	193
wine	0.0000	0.0346	0.0449	115.74	100
yeast	0.2520	0.3260	0.2988	218.70	100
zoo	0.0000	0.0200	0.1176	134.66	10000

TAB. B.18: Résultats de AdaBoost avec souches de décision comme votants, avec un nombre d'itérations t choisi parmi 15 valeurs entre 10^2 et 10^6, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	μ
australian	0.0899	0.1130	0.1507	10.75	0.247091
balance	0.0256	0.0737	0.0256	2.86	0.056234
breast	0.0200	0.0486	0.0372	10.69	0.316228
bupa	0.1279	0.2561	0.2890	1.56	0.117877
car	0.1100	0.1240	0.1401	10.35	0.056234
cmc	0.2720	0.3000	0.3001	10.66	0.150859
credit	0.1275	0.1449	0.1246	6.52	0.316228
cylinder	0.2259	0.2481	0.3296	2.68	0.247091
ecoli	0.0179	0.0952	0.0536	1.36	0.316228
glass	0.0280	0.2528	0.2150	1.35	0.092106
heart	0.1037	0.2000	0.1852	1.16	0.316228
hepatitis	0.0513	0.1808	0.1429	0.86	0.247091
horse	0.0707	0.1631	0.1739	2.95	0.247091
ionosphere	0.0000	0.1081	0.1314	7.92	0.150859
monks	0.2685	0.2873	0.2315	0.76	0.316228
mushroom	0.0000	0.0060	0.0042	49.84	0.092106
optdigits	0.0100	0.0620	0.0843	140.24	0.150859
pima	0.1745	0.2160	0.2370	8.52	0.316228
titanic	0.2300	0.2300	0.2222	3.56	0.056234
vote	0.0367	0.0368	0.0507	4.12	0.056234
wine	0.0000	0.0458	0.0674	1.80	0.316228
yeast	0.2780	0.2940	0.2907	11.95	0.193070
zoo	0.0000	0.0200	0.0392	0.34	0.247091

TAB. B.19: Résultats de CqBoost avec souches de décision comme votants, avec μ choisi parmi 15 valeurs entre 10^{-2} et $10^{-0.5}$, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	C
australian	0.0696	0.1275	0.2000	90.09	0.138950
balance	0.0032	0.0129	0.0288	11.95	19.306977
breast	0.0229	0.0286	0.0344	42.15	0.002683
bupa	0.0814	0.2620	0.3237	5.41	1.000000
car	0.1040	0.1180	0.1376	118.96	138.949549
cmc	0.2740	0.3080	0.3032	109.78	0.007197
credit	0.1536	0.1507	0.1304	87.47	0.019307
cylinder	0.0519	0.2593	0.2704	228.90	0.051795
ecoli	0.0238	0.0594	0.0774	6.61	0.138950
glass	0.0654	0.2719	0.2243	5.46	0.051795
heart	0.0296	0.1630	0.2000	12.98	0.138950
hepatitis	0.1795	0.2058	0.1948	13.17	0.001000
horse	0.1304	0.1634	0.1902	86.02	0.002683
ionosphere	0.0000	0.1249	0.1086	114.46	7.196857
monks	0.2685	0.2687	0.2315	26.40	7.196857
mushroom	0.0000	0.0080	0.0100	505.31	19.306977
optdigits	0.0300	0.0620	0.0800	2118.57	0.002683
pima	0.1771	0.2109	0.2500	47.20	0.019307
titanic	0.2300	0.2300	0.2222	27.58	1.000000
vote	0.0367	0.0368	0.0507	43.16	0.007197
wine	0.0000	0.0111	0.0787	7.19	1.000000
yeast	0.2620	0.2940	0.2886	85.54	0.007197
zoo	0.0000	0.0400	0.1373	5.85	2.682696

TAB. B.20: Résultats de CG-Boost avec souches de décision comme votants, avec C choisi parmi 15 valeurs entre 10^{-3} et 10^{3}, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	D
australian	0.1072	0.1188	0.1420	10.68	5.179475
balance	0.0256	0.0705	0.0256	2.84	51.794747
breast	0.0171	0.0314	0.0372	8.35	13.894955
bupa	0.1279	0.2677	0.2890	1.94	7.196857
car	0.1200	0.1220	0.1547	6.90	2.682696
cmc	0.2660	0.3040	0.2991	25.29	2.682696
credit	0.1275	0.1449	0.1275	7.39	5.179475
cylinder	0.2259	0.2444	0.3111	5.87	1.930698
ecoli	0.0179	0.0954	0.0595	1.14	10.000000
glass	0.0000	0.2524	0.2056	1.49	37.275937
heart	0.1259	0.1630	0.2000	0.52	1.930698
hepatitis	0.0897	0.1167	0.2078	0.34	2.682696
horse	0.0761	0.1739	0.1739	3.76	10.000000
ionosphere	0.0000	0.1024	0.1143	10.84	51.794747
monks	0.2685	0.2687	0.2315	0.83	1.000000
mushroom	0.0140	0.0140	0.0195	22.08	26.826958
optdigits	0.0100	0.0600	0.0837	123.14	26.826958
pima	0.1771	0.2240	0.2448	7.73	1.930698
titanic	0.2400	0.2300	0.2328	11.04	10.000000
vote	0.0367	0.0368	0.0507	1.38	10.000000
wine	0.0000	0.0464	0.0674	2.07	71.968567
yeast	0.2760	0.2900	0.2907	8.48	1.389495
zoo	0.0000	0.0200	0.0392	0.77	51.794747

TAB. B.21: Résultats de MDBoost avec souches de décision comme votants, avec D choisi parmi 15 valeurs entre 10^0 et 10^2, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	C
australian	0.0870	0.1275	0.1710	18.30	0.372759
balance	0.0032	0.0064	0.0288	2.90	19.306977
breast	0.0200	0.0343	0.0372	7.49	0.138950
bupa	0.1105	0.2679	0.3121	2.39	0.372759
car	0.1040	0.1180	0.1376	10.42	138.949549
cmc	0.2800	0.3080	0.3114	14.21	0.138950
credit	0.1565	0.1565	0.1333	0.39	0.007197
cylinder	0.1333	0.2593	0.2889	13.50	0.138950
ecoli	0.0179	0.0952	0.0952	1.41	1.000000
glass	0.0280	0.2519	0.3084	1.52	1.000000
heart	0.1407	0.1778	0.1926	0.60	0.138950
hepatitis	0.1026	0.1302	0.1558	0.44	0.372759
horse	0.1630	0.1635	0.2065	0.64	0.051795
ionosphere	0.0000	0.1252	0.1314	10.50	51.794747
monks	0.2685	0.2687	0.2315	0.70	1.000000
mushroom	0.0000	0.0100	0.0031	37.87	51.794747
optdigits	0.0180	0.0680	0.0825	163.17	0.138950
pima	0.1719	0.2161	0.2786	26.99	0.372759
titanic	0.2300	0.2300	0.2222	2.25	1.000000
vote	0.0367	0.0368	0.0507	0.22	0.019307
wine	0.0000	0.0686	0.0674	0.72	7.196857
yeast	0.2920	0.2980	0.2988	20.82	0.138950
zoo	0.0000	0.0000	0.0392	0.24	138.949549

TAB. B.22: Résultats de LPBoost avec souches de décision comme votants, avec C choisi parmi 15 valeurs entre 10^{-3} et 10^{3}, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	μ
australian	0.1275	0.1304	0.1391	0.79	0.000268
balance	0.0577	0.0862	0.0575	0.72	0.001931
breast	0.0314	0.0314	0.0401	0.74	0.000518
bupa	0.2267	0.2676	0.2948	0.54	0.001931
car	0.2860	0.3000	0.3013	1.34	0.001931
cmc	0.2780	0.2920	0.3135	1.36	0.000139
credit	0.1507	0.1536	0.1362	0.44	0.001389
cylinder	0.2556	0.2741	0.2926	0.55	0.000373
ecoli	0.0298	0.0296	0.0893	0.31	0.002683
glass	0.2336	0.2892	0.2617	0.19	0.000193
heart	0.1111	0.1556	0.1556	0.32	0.003728
hepatitis	0.1795	0.1800	0.2078	0.14	0.010000
horse	0.1250	0.1363	0.1848	0.36	0.001389
ionosphere	0.1364	0.1648	0.1257	0.19	0.000100
monks	0.3056	0.3288	0.3426	0.49	0.000268
mushroom	0.0840	0.0900	0.0900	1.92	0.000268
optdigits	0.0960	0.1040	0.1372	1.32	0.000100
pima	0.1979	0.2186	0.2604	1.03	0.007197
titanic	0.2220	0.2240	0.2199	0.79	0.000193
vote	0.0367	0.0413	0.0599	0.43	0.000268
wine	0.0112	0.0111	0.0337	0.15	0.000373
yeast	0.3160	0.3280	0.3232	1.17	0.000100
zoo	0.0200	0.0200	0.0392	0.36	0.000373

TAB. B.23: Résultats de MinCq avec noyaux RBF comme votants, avec μ choisi parmi 15 valeurs entre 10^{-5} et 10^{-2} et où γ est fixé en fonction des données, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	C
australian	0.1391	0.1246	0.1333	0.22	1.000000
balance	0.0160	0.0385	0.0351	0.18	10000.000000
breast	0.0314	0.0343	0.0401	0.20	3.727594
bupa	0.1802	0.2445	0.2717	0.09	51.794747
car	0.0140	0.0260	0.0342	0.34	2682.695795
cmc	0.2420	0.3040	0.3063	0.58	719.685673
credit	0.0841	0.1246	0.1304	0.21	719.685673
cylinder	0.0333	0.2481	0.2667	0.23	10000.000000
ecoli	0.0238	0.0237	0.1012	0.09	13.894955
glass	0.1308	0.2052	0.1869	0.06	719.685673
heart	0.1481	0.1778	0.1556	0.26	3.727594
hepatitis	0.1026	0.1925	0.1818	0.04	51.794747
horse	0.1467	0.1688	0.2011	0.10	51.794747
ionosphere	0.0114	0.0968	0.0971	0.09	2682.695795
monks	0.1065	0.1574	0.2083	0.34	719.685673
mushroom	0.0000	0.0080	0.0081	0.35	2682.695795
optdigits	0.0060	0.0540	0.0960	0.35	2682.695795
pima	0.2005	0.2162	0.2604	0.21	1.000000
titanic	0.2240	0.2260	0.2269	0.68	13.894955
vote	0.0367	0.0368	0.0507	0.12	13.894955
wine	0.0112	0.0111	0.0562	0.05	13.894955
yeast	0.2560	0.2780	0.2785	0.27	51.794747
zoo	0.0000	0.0200	0.1373	0.03	719.685673

TAB. B.24: Résultats de SVM avec noyau RBF, avec C choisi parmi 15 valeurs entre 10^{-4} et 10^4 et où γ est fixé en fonction des données, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	μ
australian	0.1391	0.1391	0.1478	5.95	0.010000
balance	0.0481	0.0799	0.0543	4.46	0.000268
breast	0.0257	0.0286	0.0401	5.57	0.005179
bupa	0.1977	0.2503	0.2775	1.45	0.000720
car	0.0920	0.1240	0.1002	17.78	0.000100
cmc	0.2520	0.2840	0.3217	24.79	0.000139
credit	0.1217	0.1362	0.1304	8.24	0.000100
cylinder	0.2667	0.2889	0.3370	2.65	0.001931
ecoli	0.0238	0.0237	0.0595	1.38	0.000193
glass	0.1495	0.2160	0.2056	0.85	0.000268
heart	0.1259	0.1630	0.1481	0.91	0.007197
hepatitis	0.1410	0.2058	0.1688	0.40	0.007197
horse	0.1304	0.1523	0.1739	1.15	0.002683
ionosphere	0.1250	0.1421	0.1371	1.27	0.001000
monks	0.1481	0.2037	0.2361	1.88	0.000100
mushroom	0.0580	0.0640	0.0677	19.25	0.000373
optdigits	0.0440	0.0620	0.0882	13.81	0.000268
pima	0.1927	0.2213	0.2630	7.94	0.002683
titanic	0.2220	0.2200	0.2199	9.75	0.005179
vote	0.0367	0.0368	0.0507	2.70	0.000518
wine	0.0112	0.0111	0.0337	0.75	0.003728
yeast	0.2520	0.2780	0.2744	20.86	0.000193
zoo	0.0000	0.0200	0.0392	0.24	0.000518

TAB. B.25: Résultats de CqBoost avec noyaux RBF comme votants, avec μ choisi parmi 15 valeurs entre 10^{-5} et 10^{-2} et où γ est fixé en fonction des données, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	C
australian	0.1391	0.1333	0.1362	342.86	2.682696
balance	0.0385	0.0576	0.0319	311.27	1000.000000
breast	0.0257	0.0314	0.0401	410.16	19.306977
bupa	0.2035	0.2506	0.2832	38.82	372.759372
car	0.2000	0.2500	0.1971	1372.90	1000.000000
cmc	0.2820	0.2940	0.3217	1501.91	138.949549
credit	0.1565	0.1507	0.1333	359.50	2.682696
cylinder	0.2815	0.2815	0.3630	133.18	51.794747
ecoli	0.0476	0.0414	0.1131	27.07	19.306977
glass	0.2150	0.3273	0.2897	6.41	1000.000000
heart	0.1259	0.1481	0.1704	10.08	138.949549
hepatitis	0.1154	0.1542	0.1948	2.99	372.759372
horse	0.1359	0.1580	0.1957	37.84	19.306977
ionosphere	0.1364	0.1706	0.1200	36.84	372.759372
monks	0.2917	0.3057	0.3287	70.89	1000.000000
mushroom	0.0400	0.0500	0.0498	1680.08	1000.000000
optdigits	0.0600	0.0780	0.0978	1589.92	1000.000000
pima	0.2161	0.2212	0.2500	524.05	1.000000
titanic	0.2300	0.2300	0.2222	1417.88	0.051795
vote	0.0367	0.0552	0.0553	74.85	138.949549
wine	0.0112	0.0111	0.0449	4.55	19.306977
yeast	0.2520	0.2760	0.2785	1251.63	1000.000000
zoo	0.1000	0.1000	0.0980	1.00	51.794747

TAB. B.26: Résultats de CG-Boost avec noyaux RBF comme votants, avec C choisi parmi 15 valeurs entre 10^{-3} et 10^3 et où γ est fixé en fonction des données, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	D
australian	0.1188	0.1362	0.1449	5.71	193.069773
balance	0.0192	0.0544	0.0383	27.65	517947.467923
breast	0.0229	0.0286	0.0401	9.03	1389.495494
bupa	0.2035	0.2561	0.2775	1.91	719.685673
car	0.0180	0.0620	0.0537	212.50	1000000.000000
cmc	0.2760	0.2900	0.3124	22.43	193.069773
credit	0.0725	0.1333	0.1304	44.78	138949.549437
cylinder	0.0444	0.2556	0.2704	40.51	1000000.000000
ecoli	0.0179	0.0176	0.0655	3.04	19306.977289
glass	0.1495	0.2061	0.1869	1.19	2682.695795
heart	0.1259	0.1630	0.1481	1.06	193.069773
hepatitis	0.0000	0.1925	0.1558	1.58	268269.579528
horse	0.1141	0.1632	0.1630	2.19	372.759372
ionosphere	0.0682	0.1421	0.1543	4.73	10000.000000
monks	0.1065	0.1990	0.2454	6.51	71968.567300
mushroom	0.0040	0.0120	0.0123	154.84	1000000.000000
optdigits	0.0200	0.0540	0.0659	179.53	517947.467923
pima	0.1927	0.2239	0.2578	10.62	372.759372
titanic	0.2220	0.2200	0.2199	11.17	100.000000
vote	0.0000	0.0321	0.0553	15.52	1000000.000000
wine	0.0112	0.0111	0.0337	0.71	1389.495494
yeast	0.2540	0.2800	0.2774	43.82	1389.495494
zoo	0.0000	0.0200	0.0588	0.36	71968.567300

TAB. B.27: Résultats de MDBoost avec noyaux RBF comme votants, avec D choisi parmi 15 valeurs entre 10^2 et 10^6 et où γ est fixé en fonction des données, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Ensemble de données	$R_S(B_Q)$	$R_{CV}(B_Q)$	$R_T(B_Q)$	Temps d'exécution (secondes)	C
australian	0.1072	0.1536	0.1449	73.50	1000.000000
balance	0.0417	0.0481	0.0288	13.44	138.949549
breast	0.0314	0.0314	0.0401	3.88	2.682696
bupa	0.1919	0.2561	0.2948	4.59	138.949549
car	0.0120	0.0160	0.0342	195.59	1000.000000
cmc	0.2840	0.2960	0.3227	41.74	51.794747
credit	0.0870	0.1391	0.1391	70.67	1000.000000
cylinder	0.2815	0.2778	0.3593	5.63	7.196857
ecoli	0.0595	0.0472	0.1012	1.47	7.196857
glass	0.1308	0.2442	0.2150	2.53	1000.000000
heart	0.1333	0.1704	0.1630	1.18	7.196857
hepatitis	0.1026	0.1675	0.1818	0.71	51.794747
horse	0.1141	0.1473	0.1359	3.82	19.306977
ionosphere	0.0511	0.1081	0.0971	6.33	372.759372
monks	0.1250	0.2128	0.2454	12.09	372.759372
mushroom	0.0100	0.0240	0.0261	126.44	1000.000000
optdigits	0.0280	0.0640	0.0882	135.68	372.759372
pima	0.2161	0.2291	0.2474	12.85	2.682696
titanic	0.2240	0.2260	0.2269	49.55	19.306977
vote	0.0000	0.0505	0.0553	9.37	372.759372
wine	0.0112	0.0229	0.0449	0.75	51.794747
yeast	0.2420	0.2820	0.2876	128.39	1000.000000
zoo	0.0000	0.0400	0.0000	0.39	372.759372

TAB. B.28: Résultats de LPboost avec noyaux RBF comme votants, avec C choisi parmi 15 valeurs entre 10^{-3} et 10^3 et où γ est fixé en fonction des données, dans le contexte de la résolution de tâches classiques d'apprentissage automatique.

Bibliographie

Adams, Douglas (1979). *The Hitchhiker's Guide to the Galaxy*. Pan Books.

Allwein, Erin L., Robert E. Schapire et Yoram Singer (2001). «Reducing Multiclass to Binary : A Unifying Approach for Margin Classifiers». Dans : *Journal of Machine Learning Research (JMLR)* 1, p. 113–141.

Alquier, Pierre, James Ridgway et Nicolas Chopin (2015). «On the Properties of Variational Approximations of Gibbs Posteriors». Dans : *ArXiv e-prints* 1506.04091.

Amini, Massih-Reza, François Laviolette et Nicolas Usunier (2008). «A Transductive Bound for the Voted Classifier with an Application to Semi-supervised Learning». Dans : *Advances in Neural Information Processing Systems 21 (NIPS)*, p. 65–72.

Ando, Rie Kubota et Tong Zhang (2007). «Learning on Graph with Laplacian Regularization». Dans : *Advances in Neural Information Processing Systems 19 (NIPS)*. MIT ; 1998.

Atar, Rami, Kenny Chowdhary et Paul Dupuis (2015). «Robust Bounds on Risk-sensitive Functionals via Renyi Divergence». Dans : *SIAM/ASA Journal on Uncertainty Quantification* 3.1, p. 18–33.

Atar, Rami et Neri Merhav (2015). «Information-theoretic Applications of the Logarithmic Probability Comparison Bound». Dans : *IEEE International Symposium on Information Theory (ISIT)*.

Banerjee, Arindam (2006). «On Bayesian Bounds». Dans : *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, p. 81–88.

Bardenet, Rémi et Odalric-Ambrym Maillard (2015). «Concentration Inequalities for Sampling Without Replacement». Dans : *Bernoulli* 21.3, p. 1361–1385.

Bégin, Luc, Pascal Germain, François Laviolette et Jean-Francis Roy (2014). «PAC-Bayesian Theory for Transductive Learning». Dans : *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, p. 105–113.

Bégin, Luc, Pascal Germain, François Laviolette et Jean-Francis Roy (2016). «PAC-Bayesian Bounds Based on the Rényi Divergence». Dans : *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, p. 435–444.

Belkin, Mikhail, Irina Matveeva et Partha Niyogi (2004). «Regularization and Semi-supervised Learning on Large Graphs». Dans : *Proceedings of the 17th Annual Conference on Learning Theory (COLT)*.

Belkin, Mikhail, Partha Niyogi et Vikas Sindhwani (2006). «Manifold Regularization : A Geometric Framework for Learning from Labeled and Unlabeled Examples». Dans : *Journal of Machine Learning Research (JMLR)* 7, p. 2399–2434.

Bellet, Aurélien, Amaury Habrard, Emilie Morvant et Marc Sebban (2014). «Learning a Priori Constrained Weighted Majority Votes». Dans : *Machine Learning* 97.1-2, p. 129–154.

Bi, Jinbo, Tong Zhang et Kristin P. Bennett (2004). «Column-generation Boosting Methods for Mixture of Kernels». Dans : *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*, p. 521.

Blitzer, John, Mark Dredze et Fernando Pereira (2007). «Biographies, Bollywood, Boomboxes and Blenders : Domain Adaptation for Sentiment Classification». Dans : *Annual Meeting-Association For Computational Linguistics*. T. 45. 1, p. 440.

Blumer, Anselm, Andrzej Ehrenfeucht, David Haussler et Manfred K Warmuth (1989). «Learnability and the Vapnik-Chervonenkis Dimension». Dans : *Journal of the ACM (JACM)* 36.4, p. 929–965.

Blumer, Anselm, Andrzej Ehrenfeucht, David Haussler et Manfred K Warmuth (1990). «Occam's Razor». Dans : *Readings in machine learning*, p. 201–204.

Bousquet, Olivier, Stéphane Boucheron et Gábor Lugosi (2004). «Introduction to Statistical Learning Theory». Dans : *Advanced Lectures on Machine Learning*. Springer, p. 169–207.

Boyd, Stephen et Lieven Vandenberghe (2004). *Convex Optimization*. New York, NY, USA : Cambridge University Press. isbn : 0521833787.

Breiman, Leo (1996). «Bagging Predictors». Dans : *Machine Learning* 24.2, p. 123–140.

Breiman, Leo (2001). «Random Forests». Dans : *Machine Learning* 45.1, p. 5–32.

Brent, Richard P (1973). *Algorithms for Minimization Without Derivatives*. Courier Corporation.

Brouard, Céline, Florence d'Alché-Buc et Marie Szafranski (2011). «Semi-supervised Penalized Output Kernel Regression for Link Prediction». Dans : *Proceedings of the 28th International Conference on Machine Learning (ICML)*, p. 593–600.

Catoni, Olivier (2007). *PAC-Bayesian Supervised Classification : the Thermodynamics of Statistical Learning*. Monograph series of the Institute of Mathematical Statistics.

Chapelle, Olivier, Bernhard Schölkopf et Alexander Zien, éds. (2006). *Semi-Supervised Learning*. Cambridge, MA : MIT Press. url : http://www.kyb.tuebingen.mpg.de/ssl-book.

Chen, Minmin, Kilian Q. Weinberger et John Blitzer (2011). «Co-Training for Domain Adaptation». Dans : *Advances in Neural Information Processing Systems 24 (NIPS)*, p. 2456–2464.

Chung, F. R. K. (1997). *Spectral Graph Theory*. Providence, Rhode Island : RI : Amer. Math. Soc.

Cortes, Corinna, Mehryar Mohri et Jason Weston (2007). «A General Regression Framework for Learning String-to-string Mappings». Dans : *Predicting Structured Data*, p. 143–168.

Cortes, Corinna et Vladimir Vapnik (1995). «Support-Vector Networks». Dans : *Machine Learning* 20.3, p. 273–297.

Cover, Thomas M. et Joy A. Thomas (1991). «Elements of Information Theory». Dans : Wiley. Chap. 12.

Cristianini, Nello et John Shawe-Taylor (2000a). *An Introduction to Support Vector Machines : And Other Kernel-based Learning Methods*. New York, NY, USA : Cambridge University Press. isbn : 0-521-78019-5.

Cristianini, Nello et John Shawe-Taylor (2000b). *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge, U.K. : Cambridge University Press.

Dahl, Joachim et Lieven Vandenberghe (2007). *CVXOPT*. url : http://mloss.org/software/view/34/.

Daniely, Amit, Sivan Sabato, Shai Ben-David et Shai Shalev-Shwartz (2011). «Multiclass Learnability and the ERM Principle». Dans : *COLT*, p. 207–232.

Demiriz, Ayhan, Kristin P. Bennett et John Shawe-Taylor (2002). «Linear Programming Boosting via Column Generation». Dans : *Machine Learning* 46.1-3, p. 225–254.

Derbeko, Philip, Ran El-Yaniv et Ron Meir (2004). «Explicit Learning Curves for Transduction and Application to Clustering and Compression Algorithms». Dans : *Journal of Artifial Intelligence Research (JAIR)* 22, p. 117–142.

Dietterich, Thomas G. et Ghulum Bakiri (1995). «Solving Multiclass Learning Problems via Error-Correcting Output Codes». Dans : *Journal of Artificial Intelligence Research* 2.263, p. 286.

Dredze, Mark, Alex Kulesza et Koby Crammer (2010). «Multi-Domain Learning by Confidence-weighted Parameter Combination». Dans : *Machine Learning* 79.1-2, p. 123–149.

Dumais, Susan, John Platt, David Heckerman et Mehran Sahami (1998). «Inductive Learning Algorithms and Representations for Text Categorization». Dans : *Proceedings of the seventh international conference on Information and knowledge management*. ACM, p. 148–155.

Floyd, Sally et Manfred Warmuth (1995). «Sample Compression, Learnability, and the Vapnik-Chervonenkis Dimension"». Dans : *Machine Learning* 21.3, p. 269–304.

Fortier-Dubois, Louis, François Laviolette, Mario Marchand, Louis-Emile Robitaille et Jean-Francis Roy (2015). *Efficient Learning of Ensembles with QuadBoost*. arXiv. url : http://arxiv.org/abs/1506.02535.

Freund, Yoav et Robert E. Schapire (1997). «A Decision-theoretic Generalization of On-line Learning and an Application to Boosting». Dans : *Journal of Computer and System Sciences* 55, p. 119–139.

Gärtner, Thomas (2003). «A Survey of Kernels for Structured Data». Dans : *ACM SIGKDD Explorations Newsletter* 5.1, p. 49–58.

Germain, Pascal (2015). «Généralisations de la théorie PAC-bayésienne pour l'apprentissage inductif, l'apprentissage transductif et l'adaptation de domaine». Thèse de doct. Université Laval.

Germain, Pascal, Francis Bach, Alexandre Lacoste et Simon Lacoste-Julien (2016). «PAC-Bayesian Theory Meets Bayesian Inference». Dans : *Advances in Neural Information Processing Systems 29 (NIPS)*. Sous la dir. de D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon et R. Garnett. Curran Associates, Inc., p. 1876–1884.

Germain, Pascal, Sébastien Giguere, Jean-Francis Roy, Brice Zirakiza, François Laviolette et Claude-Guy Quimper (2012). «A Pseudo-Boolean Set Covering Machine». Dans : *Principles and Practice of Constraint Programming*. Springer, p. 916–924.

Germain, Pascal, Alexandre Lacasse, François Laviolette et Mario Marchand (2009). «PAC-Bayesian Learning of Linear Classifiers». Dans : *Proceedings of the 26th International Conference on Machine Learning (ICML)*. Sous la dir. de Léon Bottou et Michael Littman. Montreal : Omnipress, p. 353–360.

Germain, Pascal, Alexandre Lacasse, François Laviolette, Mario Marchand et Jean-Francis Roy (2015). «Risk Bounds for the Majority Vote : From a PAC-Bayesian Analysis to a Learning Algorithm». Dans : *Journal of Machine Learning Research (JMLR)*, p. 787–860.

Germain, Pascal, Alexandre Lacasse, Mario Marchand, Sara Shanian et François Laviolette (2009). «From PAC-Bayes Bounds to KL Regularization». Dans : *Advances in Neural Information Processing Systems*, p. 603–610.

Germain, Pascal, Alexandre Lacoste, François Laviolette, Mario Marchand et Sara Shanian (2011). «A PAC-Bayes Sample-compression Approach to Kernel Methods». Dans : *Proceedings of the 28th International Conference on Machine Learning (ICML)*, p. 297–304.

Giguere, Sébastien, François Laviolette, Mario Marchand et Amélie Rolland (2014). «PAC-Bayesian Risk Bounds and Learning Algorithms for the Regression Approach to Structured Output Prediction». Dans : *Advanced Structured Prediction*, p. 239.

Goodfellow, Ian, Yoshua Bengio et Aaron Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Goyal, Anil, Emilie Morvant, Pascal Germain et Massih-Reza Amini (2017). «PAC-Bayesian Analysis for a Two-Step Hierarchical Multiview Learning Approach». Dans : *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, p. 205–221.

Hoeffding, Wassily (1963). «Probability Inequalities for Sums of Bounded Random Variables». Dans : *Journal of the American statistical association* 58.301, p. 13–30.

Honorio, Jean et Tommi Jaakkola (2014). «Tight Bounds for the Expected Risk of Linear Classifiers and PAC-Bayes Finite-Sample Guarantees». Dans : *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, p. 384–392.

IBM Corp (2010). *IBM ILOG CPLEX Optimizer*. url : http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

Joachims, Thorsten (2006). «Transductive Support Vector Machines». Dans : *Semi-Supervised Learning*. Sous la dir. d'O. Chapelle, B. Schölkopf et A. Zien. MIT Press, p. 105–118.

Jones, Eric, Travis Oliphant, Pearu Peterson et al. (2001). *SciPy: Open source scientific tools for Python*. url : http://www.scipy.org/.

Kuznetsov, Vitaly, Mehryar Mohri et Umar Syed (2014). «Multi-Class Deep Boosting». Dans : *Advances in Neural Information Processing Systems 27 (NIPS)*, p. 2501–2509.

Lacasse, Alexandre, François Laviolette, Mario Marchand, Pascal Germain et Nicolas Usunier (2007). «PAC-Bayes Bounds for the Risk of the Majority Vote and the Variance of the

Gibbs Classifier». Dans : *Advances in Neural Information Processing Systems 19 (NIPS)*, p. 769–776.

Lacoste, Alexandre, François Laviolette et Mario Marchand (2012). «Bayesian Comparison of Machine Learning Algorithms on Single and Multiple Datasets». Dans : *AISTATS*, p. 665–675.

Langford, John (2005). «Tutorial on Practical Prediction Theory for Classification». Dans : *Journal of Machine Learning Research (JMLR)* 6, p. 273–306.

Langford, John et Matthias Seeger (2001). *Bounds for Averaging Classifiers*. Rapp. tech. Carnegie Mellon, Departement of Computer Science.

Langford, John et John Shawe-Taylor (2003). «PAC-Bayes & Margins». Dans : *Advances in Neural Information Processing Systems 15 (NIPS)*, p. 423–430.

Laviolette, François et Mario Marchand (2005). «PAC-Bayes Risk Bounds for Sample-Compressed Gibbs Classifiers». Dans : *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, p. 481–488.

Laviolette, François et Mario Marchand (2007). «PAC-Bayes Risk Bounds for Stochastic Averages and Majority Votes of Sample-Compressed Classifiers». Dans : *Journal of Machine Learning Research (JMLR)* 8, p. 1461–1487.

Laviolette, François, Mario Marchand et Jean-Francis Roy (2011). «From PAC-Bayes Bounds to Quadratic Programs for Majority Votes». Dans : *Proceedings of the 28th International Conference on Machine Learning (ICML)*, p. 649–656.

Laviolette, François, Mario Marchand et Jean-Francis Roy (2014). «CqBoost : A Column Generation Method for Minimizing the C-Bound». Dans : *NIPS Workshop on Optimization for Machine Learning*.

Laviolette, François, Emilie Morvant, Liva Ralaivola et Jean-Francis Roy (2014). «On Generalizing the C-Bound to the Multiclass and Multi-label Settings». Dans : *NIPS Workshop on Representation and Learning Methods for Complex Outputs*.

Laviolette, François, Emilie Morvant, Liva Ralaivola et Jean-Francis Roy (2017). «Risk Upper Bounds for General Ensemble Methods with an application to Multiclass Classification». Dans : *Neurocomputing* 219, p. 15–25.

Le, Quoc V et Tomas Mikolov (2014). «Distributed Representations of Sentences and Documents». Dans : *Proceedings of the 31st International Conference on Machine Learning (ICML)*, p. 1188–1196.

Lecun, Yann et Corinna Cortes (2009). «The MNIST Database of Handwritten Digits». Dans : url : http://yann.lecun.com/exdb/mnist/.

Lee, Alan J. (1990). *U-Statistics : Theory and Practice*. New York, NY : Marcel Dekker.

Lever, Guy, François Laviolette et John Shawe-Taylor (2013). «Tighter PAC-Bayes Bounds Through Distribution-dependent Priors». Dans : *Theoretical Computer Science* 473, p. 4–28.

Lichman, Moshe (2013). *UCI Machine Learning Repository*. url : http://archive.ics.uci.edu/ml.

Madani, Omid, David M. Pennock et Gary W. Flake (2005). «Co-Validation : Using Model Disagreement on Unlabeled Data to Validate Classification Algorithms». Dans : *Advances in Neural Information Processing Systems 17 (NIPS)*. Sous la dir. de L. K. Saul, Y. Weiss et L. Bottou. MIT Press, p. 873–880. url : http://papers.nips.cc/paper/2603-co-validation-using-model-disagreement-on-unlabeled-data-to-validate-classification-algorithms.pdf.

Marchand, Mario et John Shawe-Taylor (2001). «Learning with the Set Covering Machine». Dans : *Proceedings of the 18th International Conference on Machine Learning (ICML)*, p. 345–352.

Maurer, Andreas (2004). «A Note on the PAC-Bayesian Theorem». Dans : *The Computing Research Repository (CoRR)* cs.LG/0411099.

McAllester, David (1999). «Some PAC-Bayesian Theorems». Dans : *Machine Learning* 37.3, p. 355–363.

McAllester, David (2003a). «PAC-Bayesian Stochastic Model selection». Dans : *Machine Learning* 51.1, p. 5–21.

McAllester, David (2003b). «Simplified PAC-Bayesian Margin Bounds». Dans : *Conference on Learning Theory (COLT)*, p. 203–215.

McAllester, David (2013). «A PAC-Bayesian Tutorial with a Dropout Bound». Dans : *The Computing Research Repository (CoRR)* abs/1307.2118.

Mendenhall, William (1983). «Nonparametric statistics». Dans : *Introduction to Probability and Statistics* 604.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado et Jeff Dean (2013). «Distributed Representations of Words and Phrases and their Compositionality». Dans : *Advances in Neural Information Processing Systems 26 (NIPS)*, p. 3111–3119.

Morvant, Emilie (2015). «Domain Adaptation of Weighted Majority Votes via Perturbed Variation-Based Self-Labeling». Dans : *Pattern Recognition Letters* 51, p. 37–43.

Morvant, Emilie, Amaury Habrard et Stéphane Ayache (2014). «Majority Vote of Diverse Classifiers for Late Fusion». Dans : *Structural, Syntactic, and Statistical Pattern Recognition*. Springer, p. 153–162.

Morvant, Emilie, Sokol Koço et Liva Ralaivola (2012). «PAC-Bayesian Generalization Bound on Confusion Matrix for Multi-Class Classification». Dans : *International Conference on Machine Learning*.

Mroueh, Youssef, Tomaso Poggio, Lorenzo Rosasco et Jean-Jacques Slotine (2012). «Multiclass Learning with Simplex Coding». Dans : *Advances in Neural Information Processing Systems 25 (NIPS)*, p. 2789–2797.

Niyogi, Partha (2013). «Manifold Regularization and Semi-supervised Learning : Some Theoretical Analyses». Dans : *Journal of Machine Learning Research* 14.1, p. 1229–1250.

Parrado-Hernández, Emilio, Amiran Ambroladze, John Shawe-Taylor et Shiliang Sun (2012). «PAC-Bayes Bounds with Data Dependent Priors». Dans : *Journal of Machine Learning Research (JMLR)* 13, p. 3507–3531.

Pearson, Karl (1895). «Note on Regression and Inheritance in the Case of Two Parents». Dans : *Proceedings of the Royal Society of London* 58, p. 240–242.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot et E. Duchesnay (2011). «Scikit-Learn : Machine Learning in Python ». Dans : *Journal of Machine Learning Research (JMLR)* 12, p. 2825–2830.

Pentina, Anastasia et Christoph H. Lampert (2015). «Lifelong Learning with Non-i.i.d. Tasks». Dans : *Advances in Neural Information Processing Systems 28 (NIPS)*.

Read, Jesse, Bernhard Pfahringer, Geoff Holmes et Eibe Frank (2011). «Classifier Chains for Multi-label Classification». Dans : *Machine learning* 85.3, p. 333–359.

Rényi, Alfréd (1961). «On Measures of Entropy and Information». Dans : *Fourth Berkeley symposium on mathematical statistics and probability*. T. 1, p. 547–561.

Roy, Jean-Francis, Mario Marchand et François Laviolette (2016). «A Column Generation Bound Minimization Approach with PAC-Bayesian Generalization Guarantees». Dans : *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, p. 1241–1249.

Salton, Gerard et Michael J McGill (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc.

Schölkopf, Bernhard, Ralf Herbrich et Alex J. Smola (2001). «A Generalized Representer Theorem». Dans : *International Conference on Computational Learning Theory*, p. 416–426.

Seeger, Matthias (2002). «PAC-Bayesian Generalization Bounds For Gaussian Processes». Dans : *Journal of Machine Learning Research (JMLR)* 3, p. 233–269.

Seldin, Yevgeny, François Laviolette, Nicolò Cesa-Bianchi, John Shawe-Taylor et Peter Auer (2012). «PAC-Bayesian Inequalities for Martingales». Dans : *IEEE Transactions on Information Theory* 58.12, p. 7086–7093.

Seldin, Yevgeny et Naftali Tishby (2010). «PAC-Bayesian Analysis of Co-clustering and Beyond». Dans : *Journal of Machine Learning Research (JMLR)* 11, p. 3595–3646.

Shen, Chunhua et Hanxi Li (2010). «Boosting Through Optimization of Margin Distributions». Dans : *IEEE Transactions on Neural Networks* 21.4, p. 659–666. eprint : 0904.2037.

Shen, Chunhua, Hanxi Li et Anton van den Hengel (2013). «Fully Corrective Boosting with Arbitrary Loss and Regularization». Dans : *Neural networks : the official journal of the International Neural Network Society* 48, p. 44–58. issn : 1879-2782.

Sindhwani, Vikas, Partha Niyogi et Mikhail Belkin (2005). «Beyond the Point Cloud : from Transductive to Semi-supervised Learning». Dans : *Proceedings of the 22nd international conference on Machine learning*. ACM, p. 824–831.

Taira, Hirotoshi et Masahiko Haruno (2001). «Text Categorization Using Transductive Boosting». Dans : *EMCL '01 : Proceedings of the 12th European Conference on Machine Learning*. London, UK : Springer-Verlag, p. 454–465. isbn : 3-540-42536-5.

Tolstikhin, Ilya O., Gilles Blanchard et Marius Kloft (2014). «Localized Complexities for Transductive Learning». Dans : *Proceedings of the 27th Annual Conference on Learning Theory (COLT)*, p. 857–884.

Tolstikhin, Ilya O. et Yevgeny Seldin (2013). «PAC-Bayes-Empirical-Bernstein Inequality». Dans : *Advances in Neural Information Processing Systems 26 (NIPS)*, p. 109–117.

Topsøe, Flemming (2000). «Some Inequalities for Information Divergence and Related Measures of Discrimination». Dans : *IEEE Transactions on Information Theory* 46.4, p. 1602–1609.

Tsoumakas, Grigorios et Ioannis Vlahavas (2007). «Random k-labelsets : An Ensemble Method for Multilabel Classification». Dans : *European Conference on Machine Learning*, p. 406–417.

Vapnik, Vladimir (1998). *Statistical Learning Theory*. Wiley, p. I–XXIV, 1–736. isbn : 978-0-471-03003-4.

Vapnik, Vladimir et Alexey Chervonenkis (1971). «On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities». Dans : *Theory of Probability and its Applications* 16.2, p. 264.

Von Luxburg, Ulrike (2007). «A Tutorial on Spectral Clustering». Dans : *Statistics and computing* 17.4, p. 395–416.

Weinberger, Kilian Q., Fei Sha, Qihui Zhu et Lawrence K. Saul (2006). «Graph Laplacian Regularization for Large-scale Semidefinite Programming». Dans : *Advances in Neural Information Processing Systems 18 (NIPS)*, p. 1489–1496.

Zhang, Yi et Jeff Schneider (2012). «Maximum Margin Output Coding». Dans : *Proceedings of the 29th International Conference on Machine Learning (ICML)*, p. 1575–1582.

Zhu, Ji, Hui Zou, Saharon Rosset et Trevor Hastie (2009). «Multi-class AdaBoost». Dans : *Statistics and its Interface* 2.3, p. 349–360.