# Apprentissage de Fonctions de Classification et d'Ordonnacement avec des Données Partiellement Etiquetées

Massih-Reza Amini

Université Pierre et Marie Curie - Paris 6

# Habilitation à Diriger des Recherches

Spécialité : Informatique

# Apprentissage de Fonctions de Classification et d'Ordonnacement avec des Données Partiellement Étiquetées

Masssih-Reza Amini

**Résumé**

Avec le développement des technologies d'information on assiste depuis quelques années à une nouvelle impulsion pour la conception de nouveaux cadres d'apprentissage automatique.

C'est le cas par exemple du paradigme *semi-supervisé* qui a vu le jour vers la fin des années 90 dans la communauté apprentissage. Les premiers travaux dans ce cadre ont été motivés par le développement du *web* qui a entraîné une production massive de données textuelles très hétérogènes. Ces masses de données sont généralement livrées sous forme brute, sans étiquetage *a priori* et pour les exploiter on était alors réduit à utiliser des techniques non-supervisées. Ces approches bien que totalement génériques ne permettent cependant qu'une analyse limitée des informations de contenu et ne répondent pas ainsi aux demandes de nombreuses tâches de Recherche d'Information (RI). L'idée pragmatique développée pour l'apprentissage *semi-supervisé* était née de la question ; "comment réduire l'effort d'étiquetage et utiliser simultanément une petite quantité de données étiquetée avec la masse de données non-étiquetées pour apprendre ?"

Un autre exemple de l'émergence de nouveaux cadres d'apprentissage concerne le développement de méthodes automatiques pour la recherche et *l'ordonnancement* d'entités d'information sur des corpus de grandes tailles. Récemment beaucoup de travaux se sont intéressés à la formulation des différentes formes de la tâche d'ordonnancement. Ces travaux ont proposé des algorithmes et développé des cadres théoriques pour la prédiction d'ordres totaux ou partiels sur les exemples. La Recherche d'Information est une fois encore le domaine par excellence où les algorithmes d'apprentissage de fonctions d'ordonnancement jouent un rôle prépondérant. Dans notre étude nous nous sommes intéressés à deux cadres d'ordonnancement d'instances et d'alternatives. Dans le premier cas il s'agit d'ordonner les exemples (où *instances*) d'une collection donnée de façon à ce que les exemples jugés pertinents soient ordonnés au–dessus des exemples non–pertinents et dans le second cas nous cherchons à ordonner les *alternatives* d'une collection donnée par rapport à chaque exemple d'entrée.

Ce mémoire présente mes travaux de recherche depuis ma thèse soutenue en 2001 suivant les deux axes *apprentissage semi-supervisé* et *apprentissage de fonctions d'ordonnancement* évoqués plus haut. J'ai commencé à m'intéresser à la problématique d'apprentisage semi-supervisé pour la classification à la fin de ma thèse jusqu'à fin 2003. En 2004 et 2005 j'ai abordé la problématique d'apprentissage supervisé de fonctions d'ordonnancement avec comme application phare le résumé automatique de textes. En 2006 je me suis intéressé à l'apprentissage actif de fonctions d'ordonnancement et nous avons été parmi les premiers à proposer un cadre théorique pour l'apprentissage actif de fonctions d'ordonnancement d'alternatives.

# Table des matières

# Table des figures

# Liste des tableaux

# Notations

| | |
|---|---|
| $\mathcal{X}$ | L'espace vectoriel des exemples |
| $\mathcal{L}$ | L'espace des étiquettes |
| $\mathcal{A}$ | L'ensemble des altérnatifs |
| $\Pi_{\mathcal{A}}$ | L'ensemble des permutations sur $\mathcal{A}$ |
| $\mathcal{Z}$ | L'ensemble $\mathcal{X} \times \mathcal{L}$ |
| $\mathcal{D}$ | Une distribution sur $\mathcal{Z}$ |
| $\mathcal{D}_{\mathcal{X}}$ | Une distribution marginale sur $\mathcal{X}$ |
| $Z_l \in \mathcal{Z}^n$ | L'ensemble d'apprentissage composé d'exemples étiquetés |
| $X_u \in \mathcal{X}^m$ | L'ensemble d'apprentissage composé d'exemples non–étiquetés |
| $X_u^{(k)}$ | Un sous–ensemble de $X_u$ de taille $k$ |
| $Z_{\mathcal{U}}$ | Un sous–ensemble de $X_u$ dont les étiquettes des exemples ont été activées |
| $n$ | La taille de $Z_l$ |
| $m$ | La taille de $X_u$ |
| $x \in \mathcal{X}$ | Une observation |
| $x' \in \mathcal{X}$ | Une observation non–étiquetée |
| $y \in \mathcal{L}$ | Une étiquette |
| $t$ | Un vecteur indicateur de classe |
| $\tilde{t}, \tilde{y}$ | Un vecteur indicateur et une étiquette de classe estimés par un classifieur |
| $\hat{t}, \hat{y}$ | Un vecteur indicateur et une étiquette de classe estimés par un modèle d'erreur |
| $\alpha$ | La probabilité de *mauvaise* classification |
| $\pi_k$ | La probabilité de la classe $k$ |
| $f_k$ | La probabilité conditionnelle de la classe $k$ |
| $P$ | Une partition en classes de $X_u$ |
| $\mathcal{C}_{pap}$ | Un classifieur dont les sorties prédisent des probabilités à posteriori de classe |
| $\mathcal{R}$ | Un algorithme d'apprentissage de fonctions d'ordonnacement |
| $d_c$ | La divergence associée à la fonction de coût $c$ |
| $\bar{g}$ | Une fonction d'ordonnancement de $\mathcal{X}$ vers $\Pi_{\mathcal{A}}$ |
| $g$ | La fonction de score de $\mathcal{X}$ vers $\mathbb{R}$ associée à $\bar{g}$ |
| $\mathcal{G}$ | Une classe de fonctions |
| $\mathcal{Q}$ | Une classe de fonctions de coût |
| $\mathcal{R}(\mathcal{Q})$ | La compléxité de Rademacher de la classe $\mathcal{Q}$ |
| $\hat{\mathcal{R}}(\mathcal{Q})$ | La complexité de Rademacher conditionnelle de la classe $\mathcal{Q}$ |
| $\sigma, \nu$ | Deux variables de Rademacher |

# Première partie

# Classification semi-supervisé

# 1 APPRENTISSAGE AVEC UN SUPERVISEUR IMPARFAIT

Notre motivation principale dans cette partie était de développer des algorithmes semi–supervisé efficaces qui à l'origine étaient conçues suivant un schéma génératif. L'hypothèse de base de ces approches est que la frontière entre les classes passe par des régions de basses densités. Elles trouvent cette frontière en modélisant les données avec un modèle de mélange de densités et tentent de maximiser la vraisemblance jointe des données étiquetées et non–étiquetées en utilisant l'algorithme `EM` (Dempster et al., 1977). Les inconvénients majeures de telles approches sont que les hypothèses distributionelles sur les données sont rarement vérifiées en pratique et l'estimation des paramètres peut parfois s'avérer difficile voir impossible dans certains cas (e.g. inversion de matrice de covariances) (McLachlan, 1992).

L'approche que nous avons préconisée est une approche discriminante et elle découle de l'idée développée en apprentissage supervisé qui est que pour faire de la classification il est plus efficace de chercher directement la frontière de séparation sans faire d'hypothèses distributionnelles sur les données (McLachlan, 1992).

Notre point de départ est le travail de McLachlan qui a étendu le critère vraisemblance classifiante (`CML` - Classification Maximum Likelihood) à l'apprentissage semi–supervisé de modèles génératifs (McLachlan, 1992). Nous proposons une nouvelle extension de ce critère au cas discriminant et proposons une nouvelle version de l'algorithme `CEM` pour optimiser ce critère en présence de données étiquetées et non–étiquetées (Section 1.3). Ce nouveau cadre englobe les deux cas particuliers du clustering et de l'apprentissage supervisé de classifieurs discriminants. En effet, avec des données non–étiquetées seules, notre algorithme se réduit à une version discriminante du `CEM` pour le clustering où on estime des probabilités à posteriori de clusters avec des classifieurs discriminants au lieu des densités conditionnelles comme c'est usuellement le cas. Et, avec des données étiquetées seules, notre algorithme se réduit à trouver les paramètres d'un classifieur en maximisant l'entropie croisée entre les classes estimées et les vraies classes. La maximisation de la version discriminante du critère `CML` en présence de données partiellement

étiquetées revient alors à maximiser simultanément l'entropie croisée des données étiquetées et la vraisemblance classifiante des données non–étiquetées. Dans une deuxième étape nous nous sommes intéressés à améliorer l'étiquetage des données non–étiquetées estimées par notre algorithme semi–supervisé. Nous avons proposé pour cela un modèle correcteur d'étiquettes décrit en section 1.4. Nous avons évalué empiriquement la contribution de ces idées en effectuant une série de tests sur différentes bases de données et avons comparé notre approche avec ceux de l'état de l'art (Section 1.5).

## 1.1 Notations

On note par $\mathcal{X} \subset \mathbb{R}^d$ et $\mathcal{L} = \{1, ..., c\}$, respectivement les espaces de représentation des exemples et d'étiquette de classes. Nous supposons que chaque exemple appartient à une et une seule classe et que la base d'apprentissage est composée de $n$ exemples étiquetées $Z_l = ((x_i, y_i))_{i=1}^n$ et de $m$ exemples non–étiquetés $X_u = (x'_j)_{j=1}^{n+m}$. Le but ici est de construire un classifier $h : \mathcal{X} \to \mathcal{L}$ sur la base de ces $n + m$ exemples.

Pour chaque exemple étiqueté $x \in Z_l$, on pose $y_i$ et $t_i = \{t_{ki}\}_{k \in \mathcal{L}}$ respectivement l'étiquette et le vecteur indicateur de classe associé à $x_i$.

$$\forall x_i \in Z_l, \forall k \in \mathcal{L}, y_i = k \Leftrightarrow t_{ki} = 1 \text{ et } \forall h \neq k, t_{hi} = 0$$

Au cours de l'apprentissage, les algorithmes semi–supervisé estiment des étiquettes pour les exemples non–étiquetés. On pose $\tilde{y}$ et $\tilde{t}$ respectivement l'étiquette et le vecteur indicateur de classe de l'exemple non–étiquetée $x'$ estimés par ces algorithmes.

La probabilité conditionnelle de la classe $k$ d'une observation $x'$ est notée par $f_k(x')$, $f_k(x', \theta_k)$ désigne le modèle paramétrique de cette probabilité. Pour un modèle génératif, $\Theta$ désigne le vecteur de tous ces paramètres et $\theta_k$ correspond aux paramètres de la $k^{ième}$ classe. Avec cette approche, chaque exemple est supposé être modélisé par un mélange de $c$ groupes, avec des proportions $\pi_1, ..., \pi_c$ vérifiant les conditions :

$$\sum_{k=1}^{c} \pi_k = 1 \text{ et } \forall k, \pi_k \geq 0$$

Avec l'approche discriminante, on tente d'estimer directement les probabilités à posteriori de classe. $G_k(x_i, \beta_k) = p(y = k \mid x, \beta_k)$ désigne la probabilité à posteriori de la classe $k$, $\beta_k$ correspond aux paramètres spécifiques de la classe $k$ et $B$ désigne l'ensemble des paramètres du classifieur.

Toutes les méthodes que nous proposons, partitionnent itérativement les exemples non–étiquetés de l'ensemble $X_u$ en $c$ classes. $P$ désigne un tel partitionnement et $P^{(j)}$ correspond à la partition trouvée à la $j^{ième}$ itération et $P_k^{(j)}$ est la $k^{ième}$ composante de $P^{(j)}$.

## 1.2 Apprentissage semi-supervisé : modèle génératif

L'algorithme `CEM` est une technique générale de clustering qui repose sur l'hypothèse de génération des données par un modèle de mélange de densités :

$$p(x', \Theta) = \sum_{k=1}^{c} \pi_k f_k(x', \theta_k) \tag{1.1}$$

Cet algorithme et toutes ces variantes ont toujours été utilisés dans ce cadre génératif. Dans ce qui suit nous allons d'abord présenter l'algorithme `CEM` non–supervisé de base et ensuite son extension à l'apprentissage semi–supervisé.

### 1.2.1 Algorithme CEM non–supervisé

Symons distinguent deux approches principales pour le clustering : l'approche du maximum de vraisemblance (`ML` - Maximum Likelihood) et celle du maximum de vraisemblance classifiante (`CML` - Classification Maximum Likelihood). Les méthodes à base de maximum de vraisemblance optimise le log-vraisemblance des données (equation 1.2) en modélisant les densités du mélange, le clustering se fait ensuite avec les estimées de ces densités et la règle de Bayes.

$$L_{ML}(\Theta) = \sum_{x' \in X_u} \log \sum_{k=1}^{c} \pi_k f_k(x', \theta_k) \tag{1.2}$$

Le maximum de vraisemblance classifiante optimise directement la classification des données en différents groupes, dans ce cas chaque exemple est supposé appartenir à une composante du mélange. Avec les deux approches, on suppose que les données sont générées par un mélange de densité (équation 1.1). La différence essentielle entre `ML` et `CML` est que dans le cas `CML`, les vecteurs indicateurs de classe $\tilde{t}$ des données non–étiquetées font parties des paramètres du modèle et elles sont estimées en même temps que $\Theta$.

Le critère `CML` est la log-vraisemblance complète des données et il s'écrit :

$$L_{CML}(P, \Theta) = \sum_{x' \in X_u} \sum_{k=1}^{c} \tilde{t}_{ki} \log p(x'_i, \tilde{y} = k, \Theta) \tag{1.3}$$

L'optimisation de CML se fait par le biais de l'algorithme CEM. Avec ce dernier on estime itérativement les groupes $P_k$, les proportions des classes $\pi_k$ et les paramètres $\theta_k$ des fonctions de densité modélisant les données(Celeux and Govaert, 1992).

En comparaison avec l'algorithme EM, CEM contient une étape additionnelle de classification Étape-C (Algorithme 1), où chaque exemple $x'$ est affecté à une et une seule composante du mélange (entre les étapes *Estimation* et *Maximisation* de l'algorithme EM).

### 1.2.2 Algorithme CEM semi–supervisé Génératif

McLachlan a étendu le critère CML et l'algorithme CEM au cas où les deux ensembles d'exemples étiquetés et non–étiquetés sont utilisés pour l'apprentissage (McLachlan, 1992). Dans ce cas, les vecteurs d'indicateur de classe pour les exemples étiquetés sont connus et restent inchangés pendant l'apprentissage tandis que ceux des exemples non–étiquetés sont estimés comme dans le cas non–supervisé. La log-vraisemblance complète des données (équation 1.3) s'écrit :

$$L_{ssCML}(P,\Theta) = \sum_{i=1}^{n}\sum_{k=1}^{c} t_{ki}\log p(x_i, y=k, \Theta) + \sum_{i=n+1}^{n+m}\sum_{k=1}^{c} \tilde{t}_{ki}\log p(x'_i, \tilde{y}=k, \Theta) \quad (1.4)$$

Le premier terme de cette sommation porte sur les données étiquetées et le second terme sur les exemples non–étiquetés. L'algorithme CEM peut facilement être adapté au cas semi–supervisé en maximisant l'équation (1.4) au lieu de (1.3). L'algorithme (1) décrit les deux versions non–supervisé et semi–supervisé du CEM. Dans le cas non–supervisé, la partition initiale $P^{(0)}$ est choisie aléatoirement et les fonctions de densités $f_k(.,\theta_k^{(0)})$ sont estimées sur cette partition. Dans le cas semi–supervisé, les composantes de densités initiales $f_k(.,\theta_k^{(0)})$ sont estimées en utilisant les données étiquetées. De plus, les vecteurs indicateurs de classes $\tilde{t}_{ki}$ des données non–étiquetées sont estimées à l'étape de classification (Étape-C) tandis qu'elles restent fixes à leur valeur connue pour les données étiquetées.

Pour les données non–étiquetées les probabilités à posteriori de classe sont estimées (Étape-E) et une décision est prise sur les classes (Étape-C). Dans les deux cas, l'algorithme converge vers un minimum local de $L_{CML}(P^{(j+1)}, \Theta^{(j)})$ dans sa version non–supervisé et de $L_c(^{(j+1)}, \Theta^{(j)})$ dans sa version semi–supervisé.

À titre de comparaison, la version semi–supervisé du critère Maximum de vraisemblance (1.2) s'écrit :

$$L_{ssML}(\Theta) = \sum_{i=1}^{n}\log p(x_i, y_i, \Theta) + \sum_{i=n+1}^{n+m}\log\left(\sum_{k=1}^{c}\pi_k f_k(x'_i, \theta_k)\right) \quad (1.5)$$

**Algorithm 1**: L'algorithme CEM génératif non–supervisé et semi–supervisé

**Entrée** : Un ensemble d'exemples non–étiqueté $X_u$, et d'exemples étiquetés $Z_l$ dans le cas semi–supervisé

**Initialisation**:

- ❒ Choisir une partition aléatoire sur les exemples $P^{(0)}$ dans le cas non–supervisé. Dans le cas semi–supervisé, les fonctions de densité de classes $f_k(., \theta_k^{(0)})$ sont estimées sur les données étiquetées et $P^{(0)}$ est définie en utilisant ces estimées.
- ❒ Initialiser aléatoirement $\Theta^{(0)}$
- ❒ $j \leftarrow 0$

**répéter**

- ❒ `Étape-E` : Estimer les probabilités à posteriori de classe que chaque exemple $x' \in X_u$ appartient $P_k^{(j)}$.

$$\forall x'_i \in X_u, \forall k \in \mathcal{L}, \mathbb{E}[\tilde{t}_{ki} \mid x'_i; P^{(j)}, \Theta^{(j)}] = \frac{\pi_k^{(j)} f_k(x'_i, \theta_k^{(j)})}{p(x', \Theta^{(j)})}$$

- ❒ `Étape-C` : Affecter chaque exemple $x'_i \in X_u$ au groupe $P_k^{(j+1)}$ suivant $\mathbb{E}[\tilde{t} \mid x']$, soit $P^{(j+1)}$ cette nouvelle partition.
- ❒ `Étape-M` : Estimer les nouveaux paramètres $\Theta^{(j+1)}$ qui maximise $L_{CML}(P^{(j+1)}, \Theta^{(j)})$ dans le cas non–supervisé et $L_{ssCML}(P^{(j+1)}, \Theta^{(j)})$ dans le cas semi–supervisé.
- ❒ $j \leftarrow j + 1$

**jusqu'à** *la convergence* ;

**Sortie** : Les classes des exemples $x' \in X_u$

La première partie de cette somme est sur les données étiquetées, dans ce cas chaque exemple de la base $Z_l$ provient d'une composante connue du mélange. La deuxième partie est sur les données non–étiquetées et dans ce cas les exemples de la base $X_u$ sont supposés être générés par le mélange (1.1).

(Nigam et al., 2000) ont présenté un des tous premiers algorithmes semi-supervisé pour apprendre les paramètres du modèle génératif de Naïve Bayes pour la classification de textes. Ils ont d'abord proposé à optimiser la log-vrasimeblance des données partiellement étiquetées (1.5) avec un algorithme type `EM` (ne comportant pas l'étape intermédiaire de classification de l'algorithme 1). En notant que l'estimation des paramètres avec une somme de $log$ de somme

dans (1.5) devient vite irréalisable, ils ont alors suggéré d'estimer les paramètres du modèle génératif avec un critère à base de la vraisemblance complète des données où la partie concernant les données non–étiquetées de (1.4) est pondérée par un facteur réel $\delta$ entre 0 et 1. Il est enfin à noter que (McLachlan, 1992) (page $40 - 43$) donne une analyse complète de l'algorithme `EM` optimisant le critère (1.4) dans le cas où le facteur $\delta$ vaut 1.

L'avantage majeur de l'algorithme `CEM` par rapport à l'algorithme `EM` est qu'il est très aisé d'en dériver une version discriminante pour apprendre un classifieur avec des données partiellement étiquetées. Nous montrons cette extension dans ce qui suit.

## 1.3 Apprentissage semi–supervisé : modèle discriminant

Avec une approche générative, le calcul des probabilités à posteriori de classe $p(\tilde{y} = k \mid x', \Theta)$ se fait d'une manière indirecte en passant par des estimations conditionnelles de densités. Il est connu que ce procédé aboutit à des estimés peu fiable en grande dimension où lorsque l'on dispose de très peu d'exemples étiquetés, ce qui constitue exactement le cas intéressant en apprentissage semi–supervisé.

### 1.3.1 Adaptation de la vraisemblance classifiante au cas discriminant

Une façon naturelle d'apprendre à classer serait d'utiliser un modèle discriminant pour calculer directement les probabilités à posteriori. L'algorithme semi–supervisé discriminant que nous montrons dans la suite fait l'hypothèse explicite que la sortie du classifieur discriminant estime des probabilités à posteriori de classes. Avec la règle de Bayes on peut réécrire le critère (1.4) de façon à mettre en évidence les probabilités à posteriori :

$$L_c(P,B) = \sum_{i=1}^{n}\sum_{k=1}^{c} t_{ki} \log p(y=k \mid x_i, B) + \sum_{i=1}^{n} p(x_i) + \sum_{i=n+1}^{n+m}\sum_{k=1}^{c} \tilde{t}_{ki} \log p(\tilde{y}=k \mid x'_i, B) + \sum_{i=n+1}^{n+m} p(x'_i)$$

Comme aucune hypothèse distributionelle sur la génération des données n'est faite dans le cas discirminant, l'optimisation de ce critère est équivalente à l'optimisation du critère suivant (McLachlan, 1992) :

$$L'_c(P,B) = \sum_{i=1}^{n}\sum_{k=1}^{c} t_{ki} \log p(y=k \mid x_i, B) + \sum_{i=n+1}^{n+m}\sum_{k=1}^{c} \tilde{t}_{ki} \log p(\tilde{y}=k \mid x'_i, B) \qquad (1.6)$$

### 1.3.2 Algorithme CEM semi–supervisé Discriminant

Le critère (1.14) est optimisé par l'algorithme CEM semi–supervisé discriminant (algorithme 2). Un classifieur $\mathcal{C}_{pap}$ est d'abord entraîné sur l'ensemble des exemples étiquetés $Z_l$. Les sorties de ce classifieur sont utilisées pour estimer les probabilités à posteriori de classe pour les données non–étiquetées. Chaque exemple non–étiqueté $x'_i$ est ainsi affecté à la classe dont la sortie correspondante est la plus grande. Les variables binaires indicatrices de classes de $x'_i$ sont définies suivant cette affectation (`Étape-C` ). Avec les données étiquetées initiales et l'ensemble étiqueté obtenu sur les exemples non–étiquetés un nouveau classifieur est entraîné pour maximiser $L'_c$ (`Étape-M` ). Les nouveaux paramètres ainsi obtenus sont utilisés pour obtenir de nouvelles estimations des probabilités à posteriori de classe et donc une nouvelle partition des données non–étiquetées. Il est à noter que comme dans le cas précédent, les étiquettes de classe des données étiquetées sont gardées fixes tout au long de l'apprentissage. l'`Étape-E` est triviale dans le cas discriminant comme les estimées des probabilités à posteriori sont données par les sorties du classifieur et de ce fait cette étape n'apparaît pas explicitement dans l'algorithme (2). Cet algorithme itère alors les deux étapes `C` et `M` jusqu'à la convergence de $L'_c(P, B)$ Vittaut et al. (2002).

On remarque que la première sommation dans l'équation (1.14) correspond à l'entropie croisée entre les vraies classes et les estimées des probabilités à posteriori. Ce dernier est un critère supervisé classique d'apprentissage de modèles discriminants. La seconde sommation dans (1.14) correspond au critère `CML` présenté précédemment.

Pour les deux ensembles étiquetés et non–étiquetés, l'algorithme 2 maximise simultanément l'entropie croisée des données étiquetés et le critère `CML` des données non–étiquetées. La convergence de cet algorithme à un minima local de $L'_c$ est garantie et elle est prouvée dans la section (1.4.3). Nous allons maintenant montrer qu'il est possible d'améliorer les deux algorithmes (1) et (2) en y incorporant un modèle corrigeant les étiquettes prédites par le classifieur pour les données non–étiquetées au cours des itérations des algorithmes `CEM` .

## 1.4 Modélisation de l'erreur d'étiquetage

L'idée de ce modèle correcteur d'étiquettes se comprend en considérant un procédé d'apprentissage *idéal* où les vraies étiquettes des données non–étiquetées pourraient être trouvées. En comparaison avec ce procédé, les algorithmes semi–supervisé introduites précédemment estiment à chaque itération des étiquettes de classes erronées pour certaines données non–étiquetées. Ces erreurs d'étiquetage sont inhérentes à n'importe quel algorithme semi–supervisé.

**Algorithm 2**: L'algorithme CEM semi–supervisé discriminant

**Entrée** :

- ❒ Un ensemble d'exemples étiquetés $Z_l$, Un ensemble d'exemples non–étiquetés $X_u$
- ❒ Un classifieur $\mathcal{C}_{pap}$ dont les sorties prédisent des probabilités à posteriori de classe

**Initialisation**:

- ❒ Entraîner $\mathcal{C}_{pap}$ sur $Z_l$, soit $B^{(0)}$ les paramètres obtenus
- ❒ $j \leftarrow 0$

**répéter**

- ❒ `Étape-C` : Affecter chaque exemple $x'_i \in X_u$ au groupe $P_k^{(j+1)}$ suivant la sortie de $\mathcal{C}_{pap}$

$$\forall x'_i \in X_u, \forall k \in \mathcal{L}, \tilde{t}_{ki}^{(j+1)} = \begin{cases} 1 & \text{si } p(\tilde{y}^{(j)} = k \mid x'_i, B) = \max\limits_{h \in \mathcal{L}} p(\tilde{y}^{(j)} = h \mid x'_i, B), \\ 0 & \text{sinon.} \end{cases}$$

  Soit $P^{(j+1)}$ la nouvelle partition obtenue sur les données non–étiquetées

- ❒ `Étape-M` : Estimer les nouveaux paramètres $B^{(j+1)}$ qui maximise $L'_c(P^{(j+1)}, B^{(j)})$
- ❒ $j \leftarrow j + 1$

**jusqu'à** *la convergence de* $L'_c$ ;

**Sortie** : Les classes des exemples $x' \in X_u$

Nous faisons l'hypothèse que ces erreurs d'étiquetage correspondent à un processus stochastique et nous proposons de le modéliser avec un modèle d'erreur (ou correcteur d'étiquettes) dont les paramètres seront appris en même temps que les paramètres du système semi–supervisé. Ce modèle d'erreur sera appliqué seulement aux étiquettes estimées des données non–étiquetées. Dans notre cadre, nous supposons de plus que les étiquettes des données étiquetées sont correctes et donc elles ne seront pas changées au cours de l'apprentissage.

Dans ce qui suit nous allons introduire de nouvelles notations en plus de celles introduites à la section 2. Nous montrons ensuite comment les paramètres de ce modèle d'erreur pourraient être estimés avec les deux versions génératives et discriminantes de l'algorithme `CEM` .

Nous posons $\hat{y}$ et $\hat{t}$ l' étiquette et le vecteur indicateur de classe d'un exemple non–étiqueté estimé par le modèle d'erreur, tandis que $\tilde{y}$ et $\tilde{t}$ correspondent aux estimées d'un classifieur avant l'application du modèle d'erreur. Nous notons les probabilités de mauvaise classification par :

$$\forall (k, h) \in \mathcal{L}^2, \alpha_{kh} = p(\hat{y} = k \mid \tilde{y} = h) \tag{1.7}$$

Ces probabilités sont sujettes aux contraintes :

$$\forall h, \sum_{k} \alpha_{kh} = 1 \tag{1.8}$$

La probabilité jointe d'un exemple et son étiquette corrigée s'écrit :

$$p(x'_i, \hat{y} = k) = \sum_{h=1}^{c} p(x'_i \mid \tilde{y} = h, \hat{y} = k) \times p(\hat{y} = k, \tilde{y} = h) \tag{1.9}$$

Nous supposons ici que :

$$p(x'_i \mid \tilde{y} = h, \hat{y} = k) = p(x'_i \mid \tilde{y} = h) \tag{1.10}$$

Sous l'hypothèse (1.10) et la définition (1.7), la probabilité jointe (1.9) s'écrit :

$$p(x'_i, \hat{y} = k) = \sum_{h=1}^{c} \alpha_{kh} \times p(\tilde{y} = h) \times p(x'_i \mid \tilde{y} = h) = \sum_{h=1}^{c} \alpha_{kh} \pi_h f_h(x'_i, \theta_h) \tag{1.11}$$

### 1.4.1 Mise à jour des paramètres du modèle génératif et du modèle d'erreur

Le modèle d'erreur tente de corriger les étiquettes imparfaites $\tilde{t}$ estimées pour les données non–étiquetées. Après l'application de ce modèle, la log-vraisemblance complète est alors calculée en fonction de l'ensemble des exemples étiquetés $Z_l$ et l'ensemble des exemples non–étiquetés $X_u$ avec leur étiquettes corrigée $\hat{t}_i, i \in \{n+1, ..., n+m\}$. Dans ce cas, la log-vraisemblance complète des données s'écrit :

$$L_c(P, \Theta, \Lambda) = \sum_{i=1}^{n} \sum_{k=1}^{c} t_{ki} \log p(x_i, y = k) + \sum_{i=n+1}^{n+m} \sum_{k=1}^{c} \hat{t}_{ki} \log p(x'_i, \hat{y} = k) \tag{1.12}$$

Lorsque l'on introduit les fonctions de densité de probabilité $f_k$ et le modèle d'erreur (1.7) dans le critère d'apprentissage `CML` (1.12), en utilisant (1.11) ce dernier s'écrit :

$$L_c(P, \Theta, \Lambda) = \sum_{i=1}^{n} \sum_{k=1}^{c} t_{ki} \log(\pi_k f_k(x_i, \theta_k)) + \sum_{i=n+1}^{n+m} \sum_{k=1}^{c} \left[ \hat{t}_{ki} \log \left( \sum_{h=1}^{c} \alpha_{kh} \pi_h f_h(x'_i, \theta_h) \right) \right] \tag{1.13}$$

On note par $P^{(j)}$ la partition courante des données et par $\Theta^{(j)}, \Lambda^{(j)}$ les paramètres respectifs du modèle génératif et du modèle d'erreur. Comme pour l'algorithme (1), on adopte ici une approche itérative pour la maximisation de (1.13). Les paramètres $\Theta$ sont d'abord initialisés sur les données étiquetées $Z_l$. On itère ensuite les trois étapes `E`, `C` et `M` jusqu'à la convergence de $L_c$.

**Algorithm 3**: CEM génératif semi–supervisé avec modélisation d'erreur d'étiquetage

**Entrée** :

- ❒ Un ensemble d'exemples non–étiqueté $X_u$
- ❒ Un ensemble d'exemples étiquetés $Z_l$

**Initialisation**:

- ❒ Estimer les fonctions de densité de classes $f_k(., \theta_k^{(0)})$ sur les données étiquetées et $P^{(0)}$ est définie en utilisant ces estimées.
- ❒ Initialiser aléatoirement $\Theta^{(0)}$
- ❒ Initialiser aléatoirement $\alpha_{kh}^{(0)}$ entre 0 et 1.
- ❒ $j \leftarrow 0$

**répéter**

- ❒ `Étape-E` : Estimer les probabilités jointes de classe de chaque exemple $x' \in X_u$ et de son étiquette.

$$\forall x'_i \in X_u, \forall k \in \mathcal{L}, p(x'_i, \hat{y}^{(j)} = k) = \sum_h \alpha_{kh}^{(j)} \pi_h^{(j)} f_h(x'_i, \theta_h^{(j)})$$

- ❒ `Étape-C` : Affecter chaque exemple $x'_i \in X_u$ au groupe $P_k^{(j+1)}$ suivant $p(x'_i, \hat{y}^{(j)} = k)$ :

$$\forall x'_i \in X_u, \hat{y}_i^{(j+1)} = \underset{k}{\text{argmax}} p(x'_i, \hat{y}^{(j)} = k)$$

  Soit $P^{(j+1)}$ cette nouvelle partition

- ❒ `Étape-M` : Estimer les nouveaux paramètres $(\Theta^{(j+1)}, \Lambda^{(j+1)})$ qui maximisent $L_c$
  - $\Theta^{(j+1)} = \underset{\Theta^{(j)}}{\text{argmax}} L_c(P^{(j+1)}, \Theta^{(j)}, \Lambda^{(j)})$
  - $\Lambda^{(j+1)} = \underset{\Lambda^{(j)}}{\text{argmax}} L_c(P^{(j+1)}, \Theta^{(j+1)}, \Lambda^{(j)})$
- ❒ $j \leftarrow j + 1$

**jusqu'à** *la convergence* $L_c(P, \Theta, \Lambda)$ ;

**Sortie** : Les classes des exemples $x' \in X_u$

À chaque itération, dans les étapes `E` et `C`, le modèle d'erreur modifie les affectations de classe des données non–étiquetées. Tous les paramètres du modèle génératif et du modèle d'erreur

sont modifiés à l'étape M. Les nouvelles valeurs de $\alpha$ dépendent de ses anciennes valeurs et des estimés des densités conditionnelles courantes. Nous donnerons une preuve de convergence de l'algorithme à la section 1.4.3.

Les formules de ré-estimation de $\alpha$ ainsi que les détails de cet algorithme pour les deux cas particuliers (modèle Naïve-Bayes et Gaussien) que nous avons utilisés dans nos expériences sont donnés à l'annexe A.

### 1.4.2 Mise à jour des paramètres du modèle discriminant et du modèle d'erreur

Nous allons maintenant montrer comment intégrer le modèle d'erreur dans l'algorithme (2). En suivant le raisonnement décrit dans la section 1.3, la log-vraisemblance des données complètes dans le cas discriminant avec les étiquettes corrigées $\hat{t}$ pour les données non–étiquetées s'écrit :

$$L'_c(P, B, \Lambda) = \sum_{i=1}^{n}\sum_{k=1}^{c} t_{ki} \log p(y = k \mid x_i, B) + \sum_{i=n+1}^{n+m}\sum_{k=1}^{c} \hat{t}_{ki} \log p(\hat{y} = k \mid x'_i, B) \quad (1.14)$$

Avec la règle de Bayes et l'équation (1.11) on a :

$$p(x'_i, \hat{y} = k) = p(x_i) \times \sum_{h=1}^{c} \{\alpha_{kh}\, p(\tilde{y} = h \mid x'_i)\} \quad (1.15)$$

Avec cette équation, le critère (1.14) devient :

$$L'_c(P, B, \Lambda) = \sum_{i=1}^{n}\sum_{k=1}^{c} t_{ki} \log p(y = k \mid x_i, B) + \sum_{i=n+1}^{n+m}\sum_{k=1}^{c} \left[\hat{t}_{ki} \log \left(\sum_{h=1}^{c} \alpha_{kh} p(\tilde{y} = h \mid x'_i, B)\right)\right] \quad (1.16)$$

Pour maximiser le critère (1.16), les paramètres $B$ d'un classifieur $\mathcal{C}_{pap}$ sont d'abord initialisés en entraînant le classifieur sur les données étiquetées $Z_l$. On itère ensuite deux étapes jusqu'à la convergence de $L'_c$ (Algorithme 4). Dans la première étape, le classifieur est considéré comme un superviseur imparfait pour les données non–étiquetées. Pour étiqueter un exemple non–étiqueté $x'$, les sorties du classifieur sont pondérés en utilisant les probabilités de mauvaises classification $\alpha_{kh}$. À l'étape M, les paramètres du modèle d'erreur et ceux du classifieur sont mises à jour en utilisant les données étiquetées et les étiquettes imparfaites obtenues à l'étape précédente.

À cette étape on cherche des paramètres $B$ du classifieur et $\Lambda$ du modèle d'erreur qui maximisent $L'_c(P^{(j+1)}, B^{(j)}, \Lambda^{(j)})$. Comme pour l'algorithme 2, l'étape E se découle directement des sorties du classifieur et elle n'apparaît pas explicitement dans l'algorithme 4.

Cette algorithme converge itérativement vers un maximum local de (1.16) comme on va le démontrer dans la section suivante. Les formules de ré-estimations pour le cas du classifieur logistique que nous avons utilisées dans nos expériences sont données à l'annexe A.

**Algorithm 4**: CEM semi–supervisé discriminant avec modélisation d'erreurs d'étiquetage

**Entrée** :

- ❒ Un ensemble d'exemples étiquetés $Z_l$, Un ensemble d'exemples non–étiquetés $X_u$
- ❒ Un classifieur $\mathcal{C}_{pap}$ dont les sorties prédisent des probabilités à posteriori de classe

**Initialisation**:

- ❒ Entraîner $\mathcal{C}_{pap}$ sur $Z_l$, soient $B^{(0)}$ les paramètres obtenus
- ❒ $j \leftarrow 0$

**répéter**

- ❒ `Étape-C` : Estimer les probabilités à posteriori imparfaites en utilisant les sorties du classifieur. Affecter chaque exemple $x'_i \in X_u$ au groupe $P_k^{(j+1)}$ suivant la sortie du modèle d'erreur

$$\forall x'_i \in X_u, \hat{y}_i^{(j+1)} = \underset{k}{\operatorname{argmax}} \sum_{h=1}^{c} \alpha_{kh}^{(j)} p(\tilde{y}^{(j)} = h \mid x'_i)$$

  Soit $P^{(j+1)}$ la nouvelle partition obtenue sur les données non–étiquetées.

- ❒ `Étape-M` : Estimer les nouveaux paramètres $B^{(j+1)},\ \Lambda^{(j+1)}$ qui maximisent $L'_c(P^{(j+1)}, B^{(j)}, \Lambda^{(j)})$
  - $B^{(j+1)} = \underset{B^{(j)}}{\operatorname{argmax}} L'_c(P^{(j+1)}, B^{(j)}, \Lambda^{(j)})$
  - Trouver les paramètres $\Lambda^{(j+1)}$ qui maximisent $L'_c(P^{(j+1)}, B^{(j+1)}, \Lambda^{(j)})$ sous les contraintes $\forall k, h, \alpha_{kh}^{(j+1)} \in [0, 1]$ et $\forall h, \sum_k \alpha_{kh}^{(j+1)} = 1$.
- ❒ $j \leftarrow j + 1$

**jusqu'à** *la convergence de* $L'_c$ ;

**Sortie** : Les classes des exemples $x' \in X_u$

### 1.4.3 Convergence

Les algorithmes semi–supervisé $1 - 4$, convergent vers un optimum local de leur fonction objective. Nous allons montrer la preuve de convergence pour l'algorithme 4, la même preuve s'applique aux autres algorithmes (3, 2, 1).

**Lemme 1.** *Le critère* `CML` *étendu,* $L'_c$*, croît à chaque séquence* $(P^{(j)}, B^{(j)}, \Lambda^{(j)})$ *de l'algorithme 4 et la séquence* $L'_c(P^{(j)}, B^{(j)}, \Lambda^{(j)})$ *converge vers un point stationnaire.*

*Preuve* : Nous allons d'abord montrer que $L'_c(P^{(j)}, B^{(j)}, \Lambda^{(j)})$ croît.

- Avec l'équivalence $\forall x'_i \in X_u, \forall k' \neq k, \hat{y}_i^{(j+1)} = k \Leftrightarrow p(\hat{y}^{(j+1)} = k \mid x'_i) \geq p(\hat{y}^{(j+1)} = k' \mid x'_i)$ (`Étape-C` ), nous avons

$$L'_c(P^{(j+1)}, B^{(j)}, \Lambda^{(j)}) \geq L'_c(P^{(j)}, B^{(j)}, \Lambda^{(j)})$$

- Et, comme $(P^{(j+1)}, B^{(j+1)}, \Lambda^{(j)})$ et $(P^{(j+1)}, B^{(j+1)}, \Lambda^{(j+1)})$ maximisent itérativement $L'_c$ (`Étape-M` ), on a

$$L'_c(P^{(j+1)}, B^{(j+1)}, \Lambda^{(j+1)}) \geq L'_c(P^{(j+1)}, B^{(j)}, \Lambda^{(j)})$$

Finalement, il n'y a qu'un nombre fini de partitions d'exemples en $c$ groupes, la séquence croissante $L'_c(P^{(j)}, B^{(j)}, \Lambda^{(j)})$ prend alors un nombre fini de valeurs et converge ainsi vers un point stationnaire. Ce point constitue l'optimum local de la fonction objective $L'_c$.

## 1.5 Résultats expérimentaux

Nous allons analyser dans cette partie les résultats expérimentaux des algorithmes semi–supervisé que nous avons proposés sur différents jeux de tests.

### 1.5.1 Les collections et les mesures d'évaluation

Dans nos expériences nous avons utilisé les collections `Email spam` et `Mushroom` de la base de données UCI [1] (C.L. Blake and C.J. Merz, 1998), les données `7sectors` du projet WebKB de CMU [2] et aussi la collection `Computation and Language (Cmp_lg)` de TIPSTER SUMMAC [3] pour le résumé de textes. Le tableau 1.1 récapitule les caractéristiques de ces différentes bases.

La base `7sectors` est constituée de $4477$ documents *html* partitionnés suivant un schéma hiérarchique. Nous avons étiqueté chaque document de cette base selon l'étiquette de classe du premier noeud de la hiérarchie le contenant. Il y a au total sept classes ($c = 7$) de base correspondant à différents secteurs d'activité industriels. Nous avons testé nos algorithmes en considérant $c(c-1)/2$ problèmes de classification binaires d'une classe contre les autres. Avec cette collection, les documents sont filtrés en enlevant les balises *html* et aussi les mots d'un

1. ftp ://ftp.ics.uci.edu/pub/machine-learning-databases/
2. http ://www.cs.cmu.edu/afs/cs/project/theo-20/www/data/bootstrappingIE/7sectors.tar.gz
3. http ://www.itl.nist.gov/iaui/894.02/related_projects/tipster_summac

anti-dictionnaire. Les termes occurrents dans moins de trois documents sont aussi enlevés. Nous avons ensuite lemmatisé les mots restant avec le lemmatiseur de Porter et avons appliqué la technique de sélection de variables de (Mladenic and Grobelnik, 1998) pour élaguer le vocabulaire à 3000 termes les plus informatifs selon cette technique. Chaque document de la base est alors représenté dans l'espace vectoriel constitué par la fréquence de ses termes.

La collection `Cmp_lg` est constituée de 183 articles scientifiques. Dans les compétitions Summac (SUMMAC, 1998), cette collection est utilisée pour faire du résumé automatique de textes. Le but est d'extraire les phrases les plus informatives dans chaque document pour constituer un résumé de ce document. Dans cette partie, nous considérons cette tâche comme une tâche de classification où les phrases d'un document doivent être classées suivant deux classes *pertinent*, *non–pertinent* par rapport à son résumé. Les résultats que nous avons obtenus ici sur la tâche de résumé sont antérieurs à notre étude d'apprentissage de fonctions d'ordonnancement. Nous allons voir dans le chapitre suivant que cette tâche est plutôt une tâche d'ordonnancement qu'une tâche de classification.

TABLE 1.1 – Apprentissage semi–supervisé : caractéristiques des collections.

| **Collections** | | taille | dimension | proportion des classes (%) |
|---|---|---|---|---|
| `Email spam` | | 4601 | 57 | 39.4 - 60.6 |
| `Mushroom` | | 8124 (−2480) | 22 | 38.2 - 61.8 |
| `Cmp_lg` | | 28985 | 5 | 10 - 90 |
| `7sectors` | basic | 949 | 3000 pour chaque | 21.2 |
| | energy | 355 | | 7.9 |
| | financial | 964 | | 21.5 |
| | health | 400 | | 8.9 |
| | transportation | 511 | | 11.4 |
| | technology | 998 | | 22.3 |
| | utilities | 300 | | 6.8 |

La collection `Cmp_lg` est constituée de 28985 phrases, la représentation des phrases est une version continue de celle proposé par (Kupiec et al., 1995). Cette représentation a été utilisée avec succès dans (Amini and Gallinari, 2002). Chaque phrase $s$ de longueur $l(s)$ est caractérisée par un vecteur de dimension 5, $s = (s_1, s_2, s_3, s_4, s_5)$, où $s_1 = \frac{l(s)}{\sum_{s'} l(s')}$ représente la longueur normalisée de $s$, $s_2$ est le nombre normalisé de mots clés contenus dans $s$ (comme *in conclusion*, *this article*, etc. - répertorié comme important pour le résumé (Edmundson, 1969)), $s_3$ représente le nombre normalisé d'acronymes (comme `U.S.A.`, `N.A.S.A.`, `I.B.M`, etc.) contenus dans $s$, $s_4$ est un indicateur de position de la phrase $s$ dans le document ; cette caractéristique prend

ses valeurs dans (début, milieu, fin) et finalement $s_5$ est le nombre normalisé de mots communs entre $s$ et la requête constituée des mots du titre et les mots les plus fréquents du document.

`Email spam` et `Mushroom` sont des bases de données classiques de la collection UCI. La base `Email spam` contient $4601$ e-méls caractérisés suivant $57$ prédicateurs quantitatifs. Ce problème constitue à classer les e-mél suivant les deux catégories *spam* et *non–spam*.

La base `Mushroom` est composée de $8124$ exemples correspondant à $23$ espèces de champignons. Chaque observation est identifiée comme comestible ou poisoneux et caractérisée avec $22$ attributs qualitatifs. Dans nos expériences, nous avons supprimé de cette base $2480$ exemples avec des attributs manquants.

Les deux bases de la collection UCI ont approximativement la même proportion des exemples dans les deux classes (tableau 1.1). Pour ces deux bases nous avons utilisé le pourcentage de bonne classification (PBC) comme mesure d'évaluation :

$$\text{PBC} = \frac{\#\text{d'exemples bien classés dans la base test}}{\#\text{d'exemples total dans la base test}}$$

Pour le résumé automatique, nous avons adopté l'évaluation SUMMAC en utilisant un taux de compression de $10\%$ pour chaque document. Nous avons ainsi formé les résumés des documents de la base test en sélectionnant les $10\%$ de phrases les mieux scorés par la sortie d'un classifieur. Pour l'évaluation nous avons comparé ces phrases avec les phrases de résumé désirées de chaque document. Ces phrases ont été générées sur la base des résumés synthétiques de chaque article en utilisant une méthode d'alignement décrite dans (Marcu, 1999). Dans ce cas, la mesure PBC n'a pas trop de sens comme il y a $9$ fois plus de phrases non–pertinentes que de phrases pertinentes. L'évaluation ici consiste alors à calculer la précision moyenne (PM) des systèmes définie comme :

$$\text{PM} = \frac{\#\text{de phrases extraites par le système et qui sont dans les résumés désirés}}{\#\text{total de phrases extraites par le système}}$$

Pour la tâche de la classification de textes (la base `7sectors`) nous avons utilisé comme mesure d'évaluation le point sur la courbe précision-rappel pour lequel la précision et le rappel sont égaux (BP - Break even Point).

### 1.5.2 Les expériences

Avec les algorithmes semi–supervisé génératifs, nous avons utilisé un classifieur de Naïve–Bayes et un classifieur gaussien linéaire respectivement sur les données discrètes (`7sectors`) et continues (`Mushroom`). Pour le cas discriminant, nous avons utilisé un classifieur logistique. Nous avons aussi mené des tests avec des classifieurs plus complexes que le classifieur

logistique mais ces expériences n'ont pas été concluentes. Nous avons comparé les différents algorithmes (le classifieur de base, les algorithmes CEM génératifs et discriminants avec ou sans le modèle d'erreur) sur les différentes collections décrites plus–haut. Les classifieurs de base (Naïve–Bayes, Gaussien et logistique) sont entraînés en supervisé total sur le même sous–ensemble d'exemples étiquetées que celui employé dans les algorithmes semi–supervisés. Nous avons aussi comparé l'algorithme semi–supervisé EM (Nigam et al., 2000) avec nos algorithmes génératifs et le transductive SVM (Joachims, 1999) avec nos algorithmes discriminants.

### 1.5.3 Les résultats

***Classifieurs génératifs***

Nous allons commencer à présenter les résultats obtenus avec le classifieur Naïve–Bayes et les trois algorithmes semi–supervisé : EM semi–supervisé (Nigam et al., 2000), CEM génératif semi–supervisé (Algorithme 1) et CEM génératif semi–supervisé avec modélisation de l'erreur d'étiquetage (Algorithme 3). Les performances de ces algorithmes sont comparées en cross-validant $5$ fois les bases Mushroom et 7sectors. À chaque évaluation, $25\%$ des exemples sont utilisés en test et pour l'apprentissage on a gardé une proportion fixe d'exemples étiquetés, non–étiquetés ($5-95\%$ pour Mushroom et $1-99\%$ pour 7sectors). Les résultats de ces expériences sont montrés dans les tableaux 1.2 et 1.3.

TABLE 1.2 – PBC du classifieur supervisé NB et les trois algorithmes semi–supervisé génératifs sur la base Mushroom .

| Algorithmes | NB | EM semi–sup. | CEM gen. semi–sup. | CEM gen. semi–sup. imp. |
|---|---|---|---|---|
| PBC ($\%$) | $48.5 \pm 5$ | $85.8 \pm 2$ | $86.1 \pm 3$ | $90.7 \pm 2$ |

L'utilisation de données non–étiquetées améliore considérablement les performances des algorithmes semi-supervisé par rapport au classifieur Naïve-Bayes de base ($+36.4\%$ pour Mushroom et $+8.9\%$ pour 7sectors). La modélisation de l'erreur permet une amélioration supplémentaire en comparaison avec l'algorithme semi–supervisé CEM ($+4.6\%$ pour Mushroom et $+3.6\%$ en moyenne pour 7sectors). Dans les expériences sur 7sectors nous avons utilisé un très faible pourcentage de données étiquetés. Le gain en performance semble être plus important sur les bases plus petites comme eneregy ou utilities. Pour la base Mushroom, le gain en performance est de plus de $40\%$ en comparaison avec le classifieur Naïve–Bayes de base.

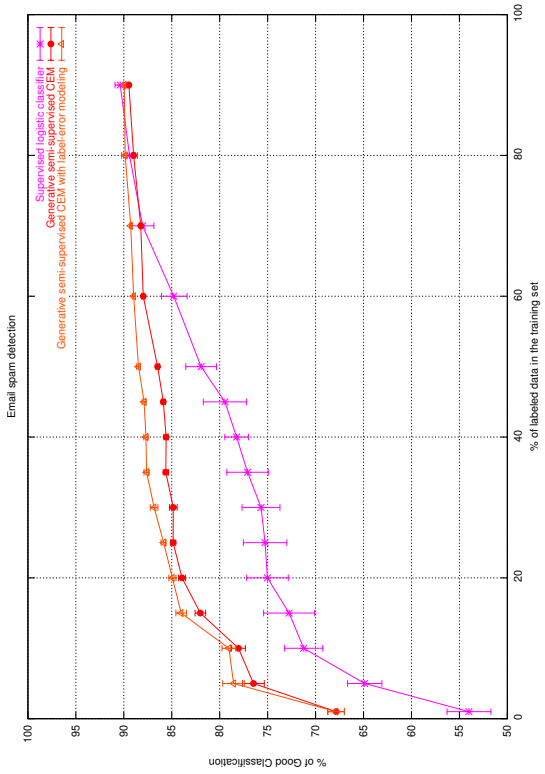
Nous avons examiné l'impact de la variation de la proportion d'exemples étiquetés, non–étiquetés de la base d'apprentissage sur les performances des algorithmes. Les figures 1.1 et 1.2

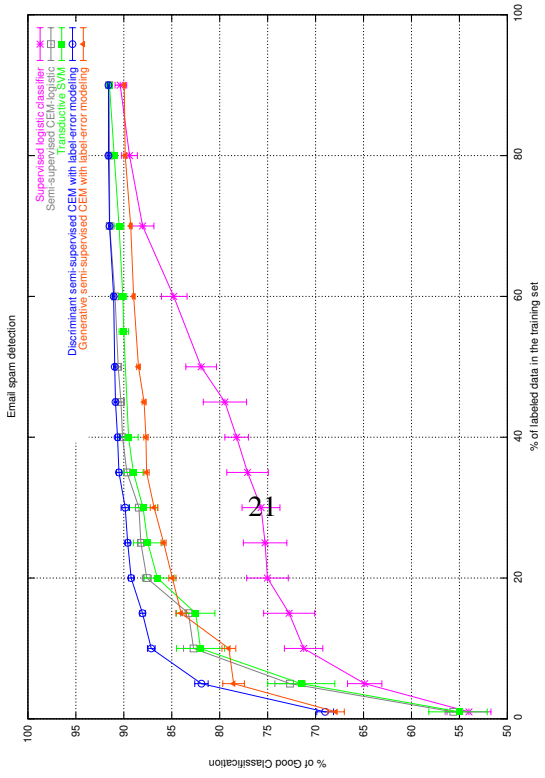
TABLE 1.3 – Break even point du classifieur supervisé NB et les trois algorithmes semi–supervisé génératifs sur la base `7sectors` .

| **Algorithmes** | | energy | financ. | health | techn. | transp. | utilities |
|---|---|---|---|---|---|---|---|
| basic | NB sup. | $57.1 \pm 3.1$ | $78.4 \pm 4.3$ | $83.8 \pm 2.5$ | $77 \pm 2.8$ | $79.2 \pm 3.4$ | $56.1 \pm 2.1$ |
| | `EM` semi-sup. | $64.9 \pm 1.8$ | $87.1 \pm 1.9$ | $85.7 \pm 1.5$ | $84.5 \pm 1.4$ | $81.6 \pm 1.8$ | $60.8 \pm 1.8$ |
| | `CEM` semi-sup. | $65.2 \pm 2.3$ | $87.3 \pm 2.8$ | $85.9 \pm 1.9$ | $84.7 \pm 1.8$ | $82.9 \pm 1.9$ | $59.9 \pm 1.7$ |
| | `CEM` semi-sup. imper. | **68.8±1.2** | **93.4±.7** | **87.6±.9** | **87.5±1.2** | **86.3±1.4** | **62.6±.8** |
| energy | NB sup. | - | $73.2 \pm 2.7$ | $65.5 \pm 2.4$ | $53.3 \pm 2.5$ | $66.2 \pm 2.1$ | $73 \pm 2.4$ |
| | `EM` semi-sup. | - | $80.1 \pm 1.5$ | $80.7 \pm 1.3$ | $60.5 \pm 1.4$ | $70.6 \pm 1.1$ | $81.9 \pm 1.5$ |
| | `CEM` semi-sup. | - | $82.1 \pm 1.4$ | $80.8 \pm 1.2$ | $60.9 \pm 1.6$ | $74.1 \pm 1.3$ | $82.8 \pm 1.2$ |
| | `CEM` semi-sup. imper. | - | **85.9±1.3** | **82.1±.9** | **66.7±1.2** | **79.7±1.5** | **86.2±.8** |
| financ. | NB sup. | - | - | 67±2.1 | 88.7±1.8 | 78.3±2.3 | 53.6±3.2 |
| | `EM` semi-sup. | - | - | $73.7 \pm 1.1$ | $91.1 \pm 1.1$ | $87.6 \pm 1.4$ | $56.4 \pm 1.3$ |
| | `CEM` semi-sup. | - | - | $74.9 \pm 1.5$ | $90.9 \pm .7$ | $87.8 \pm .9$ | $57.9 \pm 1.6$ |
| | `CEM` semi-sup.imper. | - | - | **82±.8** | **92.8±.6** | **91.9±.5** | **62.9±1.3** |
| health | NB sup. | - | - | - | $85 \pm 1.3$ | $79 \pm 1.5$ | $70.9 \pm 1.8$ |
| | `EM` semi-sup. | - | - | - | $86.5 \pm 1.8$ | $85.6 \pm 1.8$ | $79.4 \pm 1.2$ |
| | `CEM` semi-sup. | - | - | - | $87.9 \pm 1.1$ | $85.1 \pm 1.4$ | $78.9 \pm 1.5$ |
| | `CEM` semi-sup. imper. | - | - | - | **88.6±.9** | **89.8±1.2** | **81.6±1.3** |
| techn. | NB sup. | - | - | - | - | $78.9 \pm 2.1$ | $80.4 \pm 1.5$ |
| | `EM` semi-sup. | - | - | - | - | $86.6 \pm 1.5$ | $85.4 \pm 1.3$ |
| | `CEM` semi-sup. | - | - | - | - | $87.1 \pm 1.4$ | $84.5 \pm 1.6$ |
| | `CEM` semi-sup. imper. | - | - | - | - | **90.5±.9** | **87.3±1.1** |
| transp. | NB sup. | - | - | - | - | - | $51.2 \pm 3.2$ |
| | `EM` semi-sup. | - | - | - | - | - | $69.4 \pm 1.8$ |
| | `CEM` semi-sup. | - | - | - | - | - | $69.9 \pm 1.7$ |
| | `CEM` semi-sup. imper. | - | - | - | - | - | **72.3±1.3** |

montrent respectivement les performances obtenues sur les bases `Email spam` et `Cmp_lg` en fonction de la proportion d'exemples étiquetés de la base d'apprentissage. Sur l'axe des abscisses, $5\%$ signifie que $5\%$ des exemples de la base d'apprentissage ont été utilisés comme exemples étiquetés pour entraîner les classifieurs en plus des $95\%$ autres, d'exemples non–étiquetés de la base d'apprentissage dans le cas semi–supervisé. Chaque expérience a été menée en sélectionnant aléatoirement 20 fois les bases apprentissage-test.

Sur l'axe des ordonnées chaque point représente la performance moyenne d'un algorithme


Email spam detection
% of Good Classification
% of labeled data in the training set
Supervised logistic classifier
Generative semi-supervised CEM
Generative semi-supervised CEM with label-error modeling


Email spam detection
% of Good Classification
% of labeled data in the training set
Supervised logistic classifier
Semi-supervised CEM Naive
Transductive SVM
Discriminant semi-supervised CEM with label-error modeling
Generative semi-supervised CEM with label-error modeling

obtenue sur les 20 bases apprentissage-test et les bars d'erreurs correspondent à la déviation standard des performances estimées. Les caractéristiques des exemples sont continues pour les deux collections `Email spam` et `Cmp_lg`. Dans ces cas, avec les algorithmes génératifs les exemples sont supposés être distribués selon une distribution normale. Les courbes de performances (figure 2.1 haut et figure 2.2) des algorithmes semi–supervisé génératifs (`CEM` semi-supervisé génératif et `CEM` semi-supervisé génératif avec modélisation de l'erreur) confirment les conclusions obtenues dans le cas précédent où la proportion des exemples étiquetés, non–étiquetés de la base apprentissage était gardée fixe.

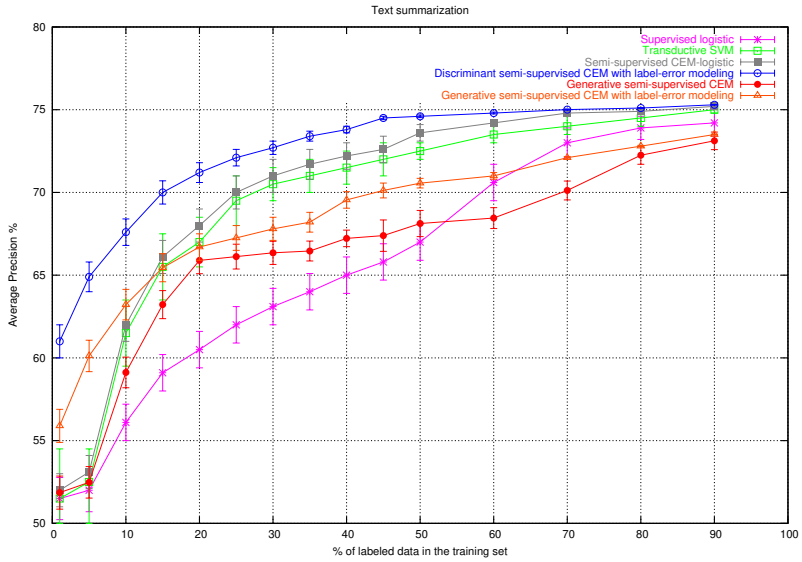



FIGURE 1.2 – Courbes de performances pour la base `Cmp_lg` pour le classifieur logistique supervisé (étoile), `CEM` discriminant semi–supervisé (carré plein), transductive SVM (carré vide), `CEM` discriminant semi–supervisé avec modélisation de l'erreur (cercle vide), `CEM` génératif semi–supervisé (cercle plein) et `CEM` génératif semi–supervisé avec modélisation de l'erreur (triangle vide). Chaque point représente la performance moyenne sur 20 bases apprentissage/test choisies aléatoirement. Les bars d'erreurs correspondent à la déviation standard des performances estimées.

On remarque que les algorithmes `CEM` semi–supervisé génératif et `EM` semi–supervisé ont des comportements similaires et qu'avec le modèle d'erreur les performances sont améliorées

d'une manière significative pour toutes les proportions d'exemples étiquetés, non–étiquetés de la base d'apprentissage.

***Classifieurs discriminants***

Nous avons mené les mêmes tests avec les trois algorithmes discriminants : le classifieur logistic supervisé et les deux algorithmes semi–supervisé discriminants (`CEM` discriminant semi–supervisé (Algorithme 2) et `CEM` discriminant semi–supervisé avec modélisation de l'erreur (Algorithme 4)). Ces résultats sont donnés respectivement en bas de la figure 1.1 et de la figure 1.2 pour les bases `Email spam` et `Cmp_lg`. En comparant ces trois algorithmes, on peut faire le même constat que dans le cas génératif ; l'utilisation de données non–étiquetés avec les données étiquetées permet d'améliorer considérablement les performances du classifieur de base pour n'importe quelle proportion de données d'apprentissage étiquetées, non–étiquetées. La modélisation de l'erreur permet aussi une nette amélioration pour les deux bases de données et avec n'importe quelle proportion d'exemples étiquetés, non–étiquetés.

Par exemple, si l'on considère la tâche du résumé de textes (Figure 1.2), en utilisant seulement $5\%$ de phrases étiquetées, l'algorithme `CEM` discriminant semi–supervisé avec modélisation de l'erreur améliore de $12\%$ la performance du classifieur logisitque de base entraîné sur $5\%$ d'exemples étiquetés de la base d'apprentissage. Avec une proportion d'exemples étiquetés, non–étiquetés de $40 - 60\%$, cet algorithme atteint la même performance en test que le classifieur logistique de base entraîné sur la totalité des exemples de la base d'apprentissage plus leur étiquette.

Avec $5\%$ d'exemples étiquetées le modèle d'erreur améliore les résultats de l'algorithme `CEM` semi–supervisé discriminant d'à peu près $10\%$. Cette amélioration est moindre lorsque la proportion d'exemples étiquetés est plus grande mais reste assez conséquente (approximativement de $5\%$ en précision moyenne pour $10\%$ d'exemples étiquetés (Figure 1.2)). L'algorithme `CEM` discriminant semi–supervisé avec modélisation de l'erreur fournit ainsi une amélioration importante en performance comparé aux deux algorithmes discrimiants de base et `CEM` semi–supervisé, spécialement lorsqu'il y a très peu d'exemples étiquetés.

***Discriminant vs. génératif***

Une seconde observation sur l'ensemble des bases et n'importe quelles proportions d'exemples étiquetés, non–étiquetés de la base d'apprentissage est que l'apprentissage discriminant est plus performant que l'apprentissage génératif. En bas de la figure 2.1 et la figure 1.2, la courbe de performance du meilleur modèle génératif (`CEM` génératif avec modélisation de l'erreur) est au-dessous des courbes de performance des algorithmes semi–supervisé discriminant avec ou

sans modélisation de l'erreur.

***EM vs. CEM***

Dans le cas génératif, les algorithmes semi–supervisé `EM` et `CEM` ont des comportements similaires. On peut voir cette ressemblance avec par exemple les résultats du tableau 1.2 obtenus sur la base `Mushroom` . L'avantage majeur de l'algorithme `CEM` par rapport à `EM` est qu'il est possible de l'étendre facilement au cas discriminant. De plus, comme l'algorithme `CEM` discriminant semi–supervisé (Algorithme 2) optimise simultanément le critère d'entropie croisée des données étiquetées et la vraisemblance classifiante des données non–étiquetées (section 1.3.2), il fournit un lien entre la classification et le clustering.

***Pourquoi un classifieur logistique pour l'apprentissage semi–supervisé ?***

Tous les algorithmes introduits ici peuvent être instanciés avec n'importe quels estimateurs de fonctions ou de classifieurs discriminants. Dans le cas discriminant, nous avons mené des tests avec différents classifieurs autres que le classifieur logistique. Ces tests ont révélé que la régression linéaire a un comportement similaire par rapport au classifieur logistique et que des classifieurs plus complexes ne sont pas plus performants par rapport à ce dernier. Dans nos expériences nous avons constaté que pour l'apprentissage semi-supervisé, des classifieurs à plus faible variance ont de bonnes performances en test. Une des explications pour cela est qu'à cause du faible nombre d'exemples étiquetés, les classifieurs complexes ne sont pas capables de trouver la bonne frontière non–linéaire pour séparer les classes. De plus, dans les problèmes réels, il semble que la frontière de décision entre peu d'exemples est approximativement linéaire et que les données non–étiquetées n'apportent pas d'information supplémentaire pour apprendre une bonne frontière non–linéaire.

Nous avons aussi comparé nos algorithmes `CEM` semi–supervisé discriminants avec le SVM transductive de (Joachims, 1999). Dans nos expériences, nous avons trouvé des courbes de performances assez similaires entre le SVM transductive et le `CEM` semi–supervisé discriminant sans modélisation de l'erreur (Figure 2.1 (bas) et Figure 2.2). On a néanmoins constaté une plus forte variance dans les performances avec le SVM transductive qu'avec notre algorithme `CEM` semi–supervisé discriminant et ceci plus particulièrement lorsqu'il y très peu d'exemples étiquetés. De plus, en comparaison avec l'algorithme `CEM` semi–supervisé discriminant avec modélisation de l'erreur, le modèle SVM transductive a donnée des performances plus basses.

*En somme, sur toutes les collections et avec les deux méthodes génératives et discriminantes, l'apprentissage semi–supervisé permet un gain notable dans les performances comparé à l'apprentissage supervisé en utilisant le même nombre d'exemples étiquetés. L'apprentissage*

*du classifieur logistique est plus performant que l'apprentissage des classifieurs génératifs du Naïve-Bayes ou du mélange de Gaussiens. L'apprentissage semi–supervisé avec le modèle d'erreur améliore encore les performances et nous avons fait ce constat sur toutes les expériences que nous avons menées.*

Tous nos résultats ici sont algorithmiques et expérimentaux. Il reste encore beaucoup à faire en particulier à développer un cadre théorique pour l'apprentissage semi–supervisé qui pourra aider à comprendre ces observations.

## 1.6 Conclusion

Nous avons proposé une nouvelle famille d'algorithmes discriminant pour l'apprentissage semi–supervisé. Ces méthodes ont été introduites en utilisant le formalisme du maximum de vraisemblance classifiante et l'algorithme `CEM` . Cette extension au cas classique génératif en utilisant ce formalisme montre que ce dernier fournit un cadre très général pour décrire les deux méthodes d'apprentissage semi–supervisé génératives et discriminantes. Nous avons aussi intégré un modèle d'erreur pour corrigé les étiquettes estimées par ces algorithmes pour les données non–étiquetées.

Ces modèles ont été testés sur différentes collections de données et la combinaison de l'algorithme semi–supervisé discriminant avec le modèle d'erreur s'est montrée être particulièrement performante. Ce qui est frappant dans ces résultats est que les différents algorithmes ont montré des performances identiques sur les différentes collections. Il est à noter que tous les algorithmes que nous avons décrits ici sont facilement implémentables. Il reste néanmoins beaucoup de questions ouvertes en particulier celle de "quand l'apprentissage semi–supervisé peut-être bénéfique à un problème donné ?"

## Deuxième partie

# Apprentissage de Fonctions d'Ordonnancement

# 2 ORDONNANCEMENT SUPERVISÉ

La problématique d'apprentissage de fonctions d'ordonnancement telle que nous l'allons étudier ici est issue des travaux de Cohen et al. (1998). Ses travaux ont été principalement motivés par les applications en RI où il s'agit d'ordonner partiellement les éléments de faÃğon à ce que les exemples pertinents pour une requête donnée soient triés au-dessus des exemples non–pertinents. (Cohen et al., 1998) ont formulé ce problème comme l'apprentissage d'une relation de préférences dont le but est de trouver une relation binaire sur les paires d'observations d'un ensemble d'apprentissage donné en identifiant l'observation qui doit obtenir un score plus élevé que l'autre [1]. Ce travail a mis en évidence qu'apprendre à ordonner les observations revient à apprendre un classifieur binaire sur les paires d'observations. D'autres travaux se sont aussi intéressés à l'apprentissage de fonctions d'ordonnancement en utilisant des algorithmes de classification binaire comme base de développement de leurs algorithmes (Weston and Watkins, 1999; Crammer and Signer, 2003; Dekel et al., 2004). D'un point de vue théorique, l'apprentissage de fonctions d'ordonnanement dans le cadre de la classifiction ou de régression habituel transgresse l'hypothèse fondamentale en apprentissage supervisé qui est que les observations sont échantillonnées idenpendaments selon une distribution fixe et inconnue. En effet les paires cruciales constituées à partir de variables aléatoires indépendantes ne sont plus indépendantes entre elles.

Dans notre étude d'apprentissage supervisé de fonctions d'ordonnancement, nous avons établit un nouveau cadre de classification binaire dans lequel les exemples d'apprentissage sont supposés être interdépendants et non plus indépendants. Ce cadre unifié nous a permis de définir certains cas d'ordonnancement d'instances et d'alternatives comme des cas particuliers de la classification binaire d'exemples interdépendants. L'idée centrale de ce cadre repose que l'existence d'une fonction de transformation déterministe qui à partir d'un ensemble d'appren-

1. On appelera ces paires des *paires cruciales* dans la suite.

tissage constitué d'exemples indépendants construit un ensemble d'exemples interdépendants. Avec cette fonction de transformation et les résultats récents en statistique sur des sommes de variables aléatoires partiellement dépendantes de Janson (2004), nous avons obtenu de nouvelles bornes de généralisation pour des fonctions d'ordonnancement. Cette extension a l'avantage de montrer le lien théorique qui existe entre la classification bianire et l'ordonnancement et permet entre autres de retrouver les bornes existantes en classification dans le cas où les données sont supposées être i.i.d. J'ai réalisé ce travail en collaboration avec Nicolas Usunier qui a soutenu sa thèse au mois de décembre 2006 (Usunier, 2006).

Dans ce chapitre, nous décrivons les résultats théoriques que nous avons obtenus dans notre étude d'apprentissage supervisé de fonctions d'ordonnancement. Les définitions formelles d'ordonnancement d'instances et d'alternatives sont données en section 2.1. La section 2.2 montre la relation entre certains cas d'ordonnancement et la classification binaire. Sur la base des résultats de Janson, nous présentons en section 2.3 une borne de test pour des classifieurs entraÃőnés avec des données interdépendants. En section 2.4 nous dérivons une borne de généralisation en utilisant les résultats de Janson et en étendant la compléxité de Rademacher au cas de données interdépendants. La conclusion de ce chapitre est donnée en section 2.5.

## 2.1 Deux tÃćches d'ordonnancement en RI

Nous nous sommes intéressés dans notre étude à deux types d'ordonnancement que l'on rencontre habituellement en RI qui sont l'ordonnancement d'alternatives et d'instances.

### 2.1.1 Ordonnancement d'alternatives

Ce type d'ordonnancement est le plus répandu en RI et il englobe des tÃćches comme la recherche documentaire ou le résumé automatique de textes. Il s'agit ici d'ordoner les éléments (communément appelés *alternatives*) d'une collection donnée par rapport à chaque observation en entrée de telle faÃğon que l'ordre prédit reflète le critère de pertinence pour chacune des observations. Par exemple en recherche documentaire une observation est une requête et le but est d'ordonner les documents (alternatives) d'une collection donnée de faÃğon à ce que les documents pertinents soient ordonnés au-dessus des documents non-pertinents. Formellement, pour chaque observation $x \in \mathcal{X}$, notons $\mathcal{A}_x = \{1, ..., |\mathcal{A}_x|\}$ l'ensemble de ses alternatives candidates que l'on déterminera par rapprot à leur indice. En recherche documentaire cela revient à déterminer un sous-ensemble de la collection initiale des documents en rapport avec une requête donnée. Dans le cadre supervisé nous supposons de plus qu'à chaque observation est associée un

vecteur désiré [2] $y \in \mathcal{L}$. Le vecteur de sortie $y = (y^1, ..., y^{|\mathcal{A}_x|}) \subset \mathbb{R}^{|\mathcal{A}_x|}$ définit ainsi l'ordre que l'on cherche à prédire sur les alternatives dans $\mathcal{A}_x$. La fonction de score $f$ qui doit prédire cet ordre prend en entrée une observation $x$ et un indice $i$ dans $\mathcal{A}_x$ et renvoie un score réel reflétant la *similarité* entre une observationun et alternatif i.e. $f : \mathcal{X} \times \mathcal{A}_x \to \mathbb{R}$.

Nous supposons par ailleurs que les couples $(x, y) \in \mathcal{X} \times \mathcal{L}$ sont générés indépendamment suivant une distribution de probabilité $\mathcal{D}$.

Pour une fonction de coÃżt $L_o : \mathbb{R}^{|\mathcal{A}|} \times \mathbb{R}^{|\mathcal{A}|} \to \mathbb{R}_+$ donnée, le risque empirique de la fonction $f$ sur un ensemble d'apprentissage $Z_l = \{(x_i, y_i)\}_{i \in \{1,..,n\}}$ de taille $n$ (supposé échantilonné i.i.d suivant $\mathcal{D}$) est défini comme la moyenne d'erreur au sens de $L_o$ de $f$ sur $Z_l$ :

$$\hat{R}^n_{OA}(f, Z_l) = \frac{1}{n} \sum_{i=1}^{n} L_o \left( f(x_i, k)_{k=1}^{|\mathcal{A}_{x_i}|}, y_i \right) \tag{2.1}$$

Et, l'erreur de généralisation est définie comme l'espérence suivant $\mathcal{D}$ de $L_{OA}$ :

$$R_{OA}(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}} L_o \left( f(x, k)_{k=1}^{|\mathcal{A}_x|}, y \right) \tag{2.2}$$

En RI, plusieurs fonctions de coÃżt existent mais il est souvent impossible de dériver des algorithmes qui estiment les paramètres d'une fonction de score en optimisant les risques empiriques associés à ces coÃżts. Ce problème est essentiellement dÃż à la non dérivabilité de ces fonctions de coÃżts ou à l'impossibilité de les majorer par des fonctions dérivables. Dans la partie algorithmique de notre étude présentée au chapitre **??**, nous avons utilisé le coÃżt mesurant le nombre moyen d'alternatifs mal-ordonnés par $f$ qui pour une observation $x$ donnée est défini par :

$$L_o \left( f(x, k)_{k=1}^{|\mathcal{A}_x|}, y \right) = \frac{1}{\sum_{i,j} [[y^i > y^j]]} \sum_{i,j : y^i > y^j} [[f(x, i) \leq f(x, j)]] \tag{2.3}$$

où $[[\pi]] = 1$ si le prédicat $\pi$ est vrai et 0 sinon. Ce coÃżt est largement utilisé dans l'état de l'art pour concevoir de nouveaux algorithmes d'ordonnancement car il présente l'avantage de pouvoir être majoré par une fonction exponentielle qui elle est facilement dérivable et admet un minimum global. Un cas particulier de ce coÃżt ainsi que son majorant exponentiel utilisés en ordonnancement bipartite sont présentés dans (Freund et al., 2004). Nous nous sommes intéressés à l'étude des fonctions d'ordonnancement par le biais de cet article qui constitue une référence en la matière dans l'état-de-l'art.

2. Par souci d'homogénéité nous avons gardé la même notation qu'au chapitre précédent pour désigner les valeurs de sortie d'une observation.

### 2.1.2 Ordonnancement d'instances

En ordonnancement d'instances ce sont les observations que l'on cherche à ordonner entre elles. Ce cas de figure correspond par exemple au filtrage d'information où il s'agit de maintenir une liste ordonnée de documents traitants d'un sujet spécifique ; les nouveaux documents qui arrivent sont automatiquement insérés dans la liste ordonnée. Dans ce cas, à chaque observation $x \in \mathcal{X}$ est associée une valeur réelle de sortie $y \in \mathcal{L} \subset \mathbb{R}$ et les couples d'exemples $(x, y)$ sont supposés être générés i.i.d suivant une distribution $\mathcal{D}$ donnée. La fonction de score $f : \mathcal{X} \to \mathbb{R}$ que l'on cherche à apprendre associe ainsi une sortie réelle à chaque observation en entrée. Pour un ensemble d'apprentissage $Z_l = \{(x_i, y_i)\}_{i \in \{1,..,n\}}$ échantillonés i.i.d suivant $\mathcal{D}$ et constitué de $n$ couples (observation, sortie réelle), on définit le coÃżt d'ordonnancement comme précédemment par une fonction $L_o : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$ et le risque empirique de $f$ sur $Z_l$ est défini par

$$\hat{R}^n_{OI}(f, Z_l) = L_o(f(X), Y)$$

Où $f(X) = (f(x_1), ...f(x_n))$ avec $X$ l'ensemble des observations de la base d'apprentissage et $Y$ l'ensemble des sorties associées. Comme dans le cas d'ordonnancement d'alternatives l'erreur de généralisation de la fonction $f$ est définie comme l'espérence suivant $\mathcal{D}$ de $\hat{R}^n_{OI}$.

#### *Ordonnancement bipartite*

L'ordonnancement bipartite est un cas particulier d'ordonnancement d'instances où les sorties associées aux observations sont des valeurs discrètes dans $\{-1, 1\}$. L'exemple du filtrage d'information que nous avons cité plus haut est plus précisement issu de ce cadre. Sur un ensemble d'apprentissage $Z_l$ de taille $n$, en notant $k$ le nombre d'exemples négatifs (d'étiquette $-1$) et $p$ le nombre d'exemples positifs (d'étiquette $+1$), l'erreur empirique d'une fonction de score $f$ sur $Z_l$ est alors :

$$\hat{R}^n_{OI}(f, Z_l) = \frac{1}{kp} \sum_{i:y_i=+1} \sum_{j:y_j=-1} [[f(x_i) \leq f(x_j)]]$$

Ce cadre a été le plus étudié dans la littérature aussi bien d'un point de vue pratique que théorique (Agarwal et al., 2005; Freund et al., 2004). Une propriété intéressante de ce cadre est qu'il existe un lien direct entre le risque empirique $\hat{R}^n_{OI}(f, Z_l)$ d'une fonction de score $f$ sur un ensemble d'apprentissage $Z_l$ et l'aire sous la courbe ROC (AUC) de cette fonction sur $Z_l$ (Cortes and Mohri, 2004) :

$$\begin{aligned} AUC(f, Z_l) &= \frac{1}{kp} \sum_{i:y_i=+1} \sum_{j:y_j=-1} \left( [[f(x_i) > f(x_j)]] + \frac{1}{2}[[f(x_i) = f(x_j)]] \right) \\ &\geq 1 - \hat{R}^n_{OI}(f, Z_l) \end{aligned}$$

## 2.2 Relation entre classification binaire et quelques cas d'ordonnancement

Nous avons montré que dans certains cas, la minimisation du risque empirique d'ordonnancement peut se faire avec un algorithme de classification binaire appliqué à un ensemble transformé de l'ensemble de départ. Dans cette section, nous allons exhiber cette fonction de transformation pour le cas d'ordonnancement d'instances. Dans le cas d'ordonnancement d'alternatives il n'existe par contre pas une telle fonction de transformation d'une manière générale.

### 2.2.1 Fonction de Transformation dans le cas d'ordonnancement d'instances

La fonction de transformation dans le cas d'ordonnancement d'instances construit d'une manière déterministe des paires cruciales sur la base d'un ensemble de départ. Ainsi pour un ensemble d'apprentissage $Z_l$ de taille $n$ l'ensemble transformé est de la forme :

$$\begin{aligned} T(Z_l) &= \left\{((x_i, x_j), 1) \,|(i,j) \in \{1,..,n\}^2 \text{ et } y_i > y_j\right\} \\ &= ((\xi_1, d_1), ..., (\xi_M, d_M)) \end{aligned}$$

où $M = \sum_{i,j}[[y_i > y_j]]$ est le nombre total des paires cruciales que l'on peut créer sur $Z_l$. En associant un classifieur $c_f : \mathcal{X} \times \mathcal{X} \to \{-1, +1\}$ sur les paires d'exemples définit à partir d'une fonction de score par :

$$c_f(x, x') = c_f(\xi) = sgn(f(x) - f(x'))$$

Où $sgn$ est la fonction de signe dans $\{-1, +1\}$, on peut voir que l'erreur empirique d'ordonnancement d'instances de $f$ sur $Z_l$, $\hat{R}^n_{OI}(f, Z_l)$ est égale à l'erreur du classifieur associé $c_p$ sur $T(Z_l)$ :

$$\hat{R}^n_{OI}(f, Z_l) = \frac{1}{\sum_{i,j}[[y_i > y_j]]} \sum_{i,j:y_i>y_j} [[f(x_i) \leq f(x_j)]] = \frac{1}{M} \sum_{k=1}^{M} [[d_k c_f(\xi_k) \leq 0]]$$

### 2.2.2 Fonction de Transformation dans le cas d'ordonnancement d'alternatives

Pour le cas d'ordonnancement d'alternatives, il n'existe pas, d'une manière générale, une fonction de transformation qui permet d'assimiler cette tÃćche comme une classifcation de paires. Ceci est dÃż au fait qu'en général le nombre d'alternatives varie d'une observation à l'autre. Ce problème s'illustre bien avec la fonction de coÃżt $L_o$ (2.3), dans ce cas avec un nombre de paires cruciales variable il n'est plus possible de factoriser le quotient de ce nombre

dans (2.1) et donc de l'assimiler comme une erreur de classification. Il existe néanmoins des cas particulier d'ordonnancement d'alternatives, comme la classification multi-tÃćches, où le nombre d'alternatives est fixe et où il est possible d'exprimer l'erreur empirique d'ordonnancement comme une erreur de classification de paires.

## 2.3 Borne de test

Les études théoriques que nous avons menées se basent exclusivement sur l'existence d'une fonction de transformation $T$ d'un ensemble d'exemples indépendants, permettant l'assimilation de l'erreur empirique d'ordonnancement à l'erreur de classification de paires. Dans la suite nous nous concentrons sur les problèmes d'ordonnancement pour lesquels il existe une telle fonction et nous ferons l'analyse dans le cadre de la classification d'exemples interdependants sur la base de cette fonction de transformation.

À partir d'un ensemble $T(S) = (\xi_i, d_i)_{i\in\{1,\dots,M\}}$ obtenu sur la base d'un ensemble étiqueté $S$, posons le classifieur de paires $c_f : \mathcal{X} \times \mathcal{X} \to \{-1, +1\}$ associé à une fonction de score $f : \mathcal{X} \to \mathbb{R}$ et considérons l'erreur empirique de $c_f$ sur $T(S)$ :

$$\hat{R}^T_M(c_f, T(S)) = \frac{1}{M}\sum_{i=1}^{M} L(c_f(\xi_i), d_i) \tag{2.4}$$

Où, $L : -1, 1^2 \to [0, 1]$ est une fonction de coÃżt donnée. Nous définissons l'erreur de généralisation de $c_f$ comme dans le cas de classification qui est la probabilité d'erreur de classification de $c_f$ sur une paire cruciale :

$$R^T(c_f) = \mathbb{E}_{T(S)}\hat{R}^T_M(c_f, T(S)) \tag{2.5}$$

Le risque (2.4) est une somme de variables aléatoires interdépendants et les résultats classiques en statistique, basés sur l'hypothèse d'indépendance des variables aléatoires, ne permettent pas de la majorer avec (2.5) pour en dériver une borne de test.

### 2.3.1 Théorème de Janson

La borne de test que nous avons obtenue ici est une application du théorème de Janson (2004) qui étend le théorème de Hoeffding (1963) au cas de variables aléatoires interdépendantes. Le résultat de ce théroème fait apparaÃőtre une mesure d'interdépendance entre les variables qui est définie par rapport au nombre de sous-ensembles de variables intépendantes

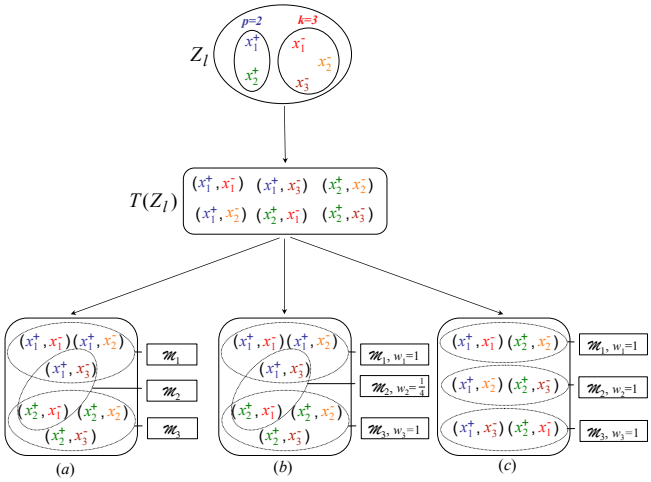



FIGURE 2.1 – Exemple de trois recouvrements d'un ensemble transformé $T(S)$ d'exemples interdépendants correspondant à un problème d'ordonnancemet bipartite. En (a) les trois sous-ensembles $\mathfrak{M}_1$, $\mathfrak{M}_2$ et $\mathfrak{M}_3$ forment un recouvrement de $T(S)$. En (b) les ensembles $\{\mathfrak{M}_j, w_j\}_{j\in\{1,2,3\}}$ forment un recouvrement fractionnaire de $T(S)$ et en (c) les ensembles $\{\mathfrak{M}_j, w_j\}_{j\in\{1,2,3\}}$ forment un recouvrement propre exact de $T(S)$.

construits sur cette base. Ce degré est d'autant plus important qu'il y a de sous-ensembles de variables indépendantes et la probabilité de déviation entre l'erreur empirique (2.4) sur une base de test particulière et l'erreur de généralisation (2.5) est d'autant plus faible que ce degré est petit.

Afin de mieux comprendre le théorème de Janson nous illustrons sur un exemple simple d'ordonnancement bipartite (figure 2.1) les définitions données dans (Janson, 2004). Pour simplifier la présentation, nous assimilerons dans cet exemple l'ensemble transformé $T(S)$ à l'ensemble des indices $\mathcal{M}$ des exemples interdépendants. Sur cet exemple, l'ensemble $S$ contient $p = 2$ exemples pertinents et $k = 3$ exemples non-pertinents, l'ensemble $T(S)$ contient toutes les paires cruciales ($k \times p = 6$ au total), chacune formée par un exemple pertinent et un exemple non-pertinent. La figure 2.1, (a) montre un *recouvrement* de $T(S)$ par une famille $\mathfrak{M}_1$, $\mathfrak{M}_2$ et $\mathfrak{M}_3$ de sous-ensembles, i.e. $\cup_j \mathfrak{M}_j = T(S)$. La figure 2.1, (b) montre une famille $\{(\mathfrak{M}_j, w_j)\}$ avec $\mathfrak{M}_j \subset T(S)$ et $w_j \in [0, 1]$ qui forme un *recouvrement fractionnaire* de $T(S)$ car $\forall i \in T(S), \sum_j w_j[[i \in \mathfrak{M}_j]] \geq 1$ et finalement la figure 2.1, (c) montre un *recouvrement*

*fractionnaire propre exact* de $T(S)$ constitué de sous-ensembles $\mathfrak{M}_j$ contenant des variables indépendantes et $\forall i \in T(S), \sum_j w_j[[i \in \mathfrak{M}_j]] = 1$.

Le nombre chromatique $\chi(\mathcal{M})$ est le plus petit entier tel qu'il existe un recouvrement propore $\{\mathfrak{M}_j\}_j$ de $\mathcal{M}$ et le nombre chromatique pur, $\chi^*(\mathcal{M})$, est le minimum de $\sum_j w_j$ sur l'ensemble des recouvrements fractionnaires propres de $\mathcal{M}$. Comme un recouvrement propre est aussi un recouvrement fractionnaire telle que $\forall j, w_j = 1$, cela implique que $\chi^*(\mathcal{M}) \leq \chi(\mathcal{M})$. Dans notre exemple nous avons $\chi^*(\mathcal{M}) = \chi(\mathcal{M}) = 3$.

Le théorème de Janson s'enonce :

**Théorème 2** (Théorème de Janson (2004))**.** *Soient $Y_1, ..., Y_M$, $M$ variables aléatoires tel que $\forall i \in \mathcal{M} = \{1, ..., M\}, \exists (a_i, b_i) \in \mathbb{R}^2, Y_i \in [a_i, b_i]$ et, soit $\chi^*(\mathcal{M})$ le nombre chromatique pur associé à l'ensemble $\mathcal{M}$, on a alors :*

$$\forall \epsilon > 0, \ \ \mathbb{P}\left(\mathbb{E}\left(\sum_{i=1}^{M} Y_i\right) - \sum_{i=1}^{M} Y_i > \epsilon\right) \leq \exp\left(-\frac{2\epsilon^2}{\chi^*(\mathcal{M})\sum_{i=1}^{M}(b_i - a_i)^2}\right)$$

Nous pouvons voir que $\chi^*(\mathcal{M}) = \chi(\mathcal{M}) = 1$ signifie qu'il existe un recouvrement unique de $\mathcal{M}$ composé de variables aléatoires indépendantes, ceci implique que le théorème de Janson est une extension directe du théorème de Heoffding comme pour une valeur particulière de $\chi^*(\mathcal{M})$, impliquant les hypohtèses du théorème de Heoffding, on obtient bien ce dernier théorème.

### 2.3.2 Application à la classification de données interdépendantes

La borne de test générique que nous avons obtenue dans le cadre de notre étude de classification de données interdépendantes est une application du théorème de Janson sur la somme de variables aléatoires $Y_i = \frac{1}{M}[[d_i c_f(\xi_i) \leq 0]]$ de l'équation (2.4). Dans ce cas nous avons $\forall i, Y_i \in [0, \frac{1}{M}]$ et la borne de test sur une base $S$ s'écrit :

$$\forall \delta > 0, \mathbb{P}\left(R^T(c_f) \leq \hat{R}_M^T(c_f, T(S)) + \sqrt{\frac{\chi^*(T)\ln(\frac{1}{\delta})}{2M}}\right) \geq 1 - \delta \tag{2.6}$$

Dans notre étude de fonctions d'ordonnancement l'ensemble des données interdépendantes est construit sur la base d'une fonction de transformation $T$, nous pouvons alors lier le degré d'interdépendance des données $\chi^*(T)$ à cette fonction de transformtion.

## 2.4 Borne de généralisation

Comme dans le cas de la classification, la borne de test (2.6) présente des restrictions. Son intérpretation est que pour un classifieur $c_f$ donné il existe un ensemble transformé $T(S)$ sur lequel l'inégalité $R^T(c_f) - \hat{R}^T_M(c_f, T(S)) \leq \sqrt{\frac{\chi^*(T)\ln(\frac{1}{\delta})}{2M}}$ tient avec une probabilité d'au moins $1-\delta$. Ces ensembles $T(S)$ peuvent être différents pour différents classifieurs, autrement dit, il n'y a qu'un certain nombre de classifieurs satisfaisants cette inégalité.

La borne de généralisation que l'on cherche à avoir doit être vraie pour n'importe quel classifeur d'un ensemble de fonctions donné. L'idée est de considérer les déviations uniformes entre l'erreur de généralisation (2.5) et l'erreur empirique (2.4) d'un classifieur quelconque sur un ensemble d'apprentissage quelconque.

Pour dériver notre borne de généralisation nous avons étendu la théorie de Rademacher étudiée en classification de données indépendantes (Bartlett and Mendelson, 2003; Taylor and Cristianini, 2004) au cas de classification avec des données interdépendantes. Dans le cas classique de la classification de données i.i.d, l'intérêt majeur de ce cadre par rapport à celui de Vapnik, est qu'il est possbile d'obtenir une borne dépendante des données permettant ainsi d'avoir des estimations plus fines et facilement calculables de l'erreur de généralisation.

### 2.4.1 Étapes d'otention d'une borne de généralisation : Rappel

Dans cette section nous donnons les grandes lignes d'obtention d'une borne de généralisation dans le cas de la classification de données i.i.d. utilisant la théorie développée autour de la compléxité de Rademacher. Dans les sections 2.4.3 et 2.4.4 nous étenderons les outils statistiques utilisés dans cette théorie qui avec le théorème de Janson permettent de prendre en compte des données interdépendentes dans la borne.

Soit $\mathcal{G}$ une classe de fonctions à valeur discrète [3] dans $\mathcal{L} = \{-1, 1\}$ entraÃőnées sur des données échantillonnées i.i.d suivant une distribution $\mathcal{D}$. Pour une fonction de coÃżt donnée $L : \mathcal{L} \times \mathcal{L} \to \mathbb{R}_+$, et un classifieur $c : \mathcal{X} \to \mathcal{L}$, posons $R_n(c, Z_l) = \frac{1}{n}\sum_{i=1}^{n} L(c(x_i), y_i)$, le risque empirique du classifieur $c$ sur une base d'apprentissage $Z_l$ de taille $n$ et $R(c) = \mathbb{E}_{(x,y)\sim\mathcal{D}} L(c(x), y)$, l'erreur de généralisation de $c$.

La compléxité de Rademacher d'une classe de fonction est une mesure de capacité estimant l'aptitude de cette classe à s'ajuster aux données bruitées. Cette aptitude est mesurée grÃ́ce aux variables aléatoires indépendantes de *Rademacher*, $\sigma_1, ..., \sigma_n$ prenant des valeurs dans $\{-1, 1\}$ et telles ques $\forall i, \mathbb{P}(\sigma_i = -1) = \mathbb{P}(\sigma_i = 1) = 1/2$.

Sur un ensemble d'apprentissage $Z_l$ de taille $n$ la compléxité de Rademacher empirique

3. Ceci n'est pas une restriction, si l'ensemble $\mathcal{G}$ considéré est un ensemble de fonctions à valeur réelles, nous prendrons alors l'ensemble de classifieurs associé $\{x \mapsto sgn(c(x)) \mid c \in \mathcal{G}\}$ où $sgn$ est la fonction de signe à valeur dans $\{-1, 1\}$.

mesure ainsi la meilleure correlation entre la classe de fonction et des étiquettes aléatoires :

$$\hat{\mathcal{R}}_n(\mathcal{G}, Z_l) = \mathbb{E}_\sigma \sup_{c \in \mathcal{G}} \frac{2}{n} \sum_{i=1}^{n} \sigma_i c(x_i)$$

Et, la complexité de Rademacher de $\mathcal{G}$ est l'espérence sur toutes les bases d'apprentissage de taille $n$ de $\hat{\mathcal{R}}_n(\mathcal{G})$ :

$$\mathcal{R}(\mathcal{G}) = \mathbb{E}_{Z_l} \hat{\mathcal{R}}_n(\mathcal{G}, Z_l) = \mathbb{E}_{Z_l \sigma} \sup_{c \in \mathcal{G}} \frac{2}{n} \sum_{i=1}^{n} \sigma_i c(x_i)$$

La borne de généralisation faisant intervenir la compléxité Rademacher d'une classe de fonction $\mathcal{G}$ est donnée par le théorème suivant (Bartlett and Mendelson (2003); Taylor and Cristianini (2004) - chapitre 4) :

**Théorème 3.** *Pour toute $\delta \in ]0, 1]$, on a avec une probabilité au moins égale à $1 - \delta$,*

$$\forall c \in \mathcal{G}, \forall Z_l, R(c) \leq R_n(c, Z_l) + \mathcal{R}(L \circ \mathcal{G}) + \sqrt{\frac{\log \frac{2}{\delta}}{2n}}$$

*et aussi avec une probabilité au moins égale à $1 - \delta$*

$$\forall c \in \mathcal{G}, \forall Z_l, R(c) \leq R_n(c, Z_l) + \hat{\mathcal{R}}_n(L \circ \mathcal{G}, Z_l) + 3\sqrt{\frac{\log \frac{2}{\delta}}{2n}}$$

Où $L \circ \mathcal{G} = \{(x, y) \mapsto L(c(x), y)\}$. L'intérêt majeur de ce théorème apparaÃőt dans la deuxième borne qui fait intervenir la compléxité de Rademacher empirique de la classe $L \circ \mathcal{G}$. Il est en effet possible d'estimer facilement la borne pour des fonctions d'une certaine classe en calculant cette complexité sur une base d'apprentissage quelconque.

Il y a trois étapes majeures qui permettent d'obtenir ces bornes. Nous allons présenter les grandes lignes de ces étapes sans insister sur les détails techniques, ceci afin de bien illustrer les changements que nous avons dÃż apporter pour obtenir la nouvelle borne de généralisation de classifieurs entraÃőnés avec des données interdépendantes.

***Étape 1 : borner le supremum sur $\mathcal{G}$ de*** $R(c) - R_n(c, Z_l)$

Pour que la borne de généralisation soit vraie d'une manière uniforme pour toutes les fonctions $c$ d'une classe $\mathcal{G}$ donnée et toutes bases d'apprentissage $Z_l$, on majore $\sup_{c \in \mathcal{G}}[R(c) - R_n(c, Z_l)]$ en remarquant que

$$\forall c \in \mathcal{G}, \forall Z_l, R(c) - R_n(c, Z_l) \leq \sup_{c \in \mathcal{G}}[R(c) - R_n(c, Z_l)]$$

La majoration du suprémum se fait par un outil puissant développé pour des processus empiriques par (McDiarmid, 1989) :

**Théorème 4** (Théorème des différences bornées (McDiarmid, 1989))**.** *Soient $X_1, ..., X_n$, $n$ variables aléatoires indépendantes à valeur dans $\mathcal{X}$. Soit $\Phi : \mathcal{X}^n \rightarrow \mathbb{R}$ telle que : $\forall i \in \{1, ..., n\}, \exists c_i \in \mathbb{R}$ tel que $\forall (x_1, ..., x_n) \in \mathcal{X}^n, \forall x' \in \mathcal{X}$ l'inégalité suivate est vraie :*

$$|\Phi(x_1, .., x_{i-1}, x_i, x_{i+1}, .., x_n) - \Phi(x_1, .., x_{i-1}, x', x_{i+1}, .., x_n)| \leq c_i$$

*Alors,*

$$\forall \epsilon > 0, \mathbb{P}(\Phi(x_1, ..., x_n) - \mathbb{E}\Phi > \epsilon) \leq e^{\frac{-2\epsilon^2}{\sum_{i=1}^{n} c_i^2}}$$

En considérant la fonction suivante :

$$\Phi : Z_l \mapsto \sup_{c \in \mathcal{G}} [R(c) - R_n(c, Z_l)]$$

On peut remarquer que pour deux échantillons $Z_l$ et $Z_l^i$ de taille $n$ contenant exactement les mêmes exemples à part le couple $(x_i, y_i) \in Z_l$ à la place duquel $Z_l^i$ contient le couple $(x', y')$ échantillonné suivant la même distribution $\mathcal{D}$, la différence $|\Phi(Z_l) - \Phi(Z_l^i)|$ est bornée par $c_i = 1/n$ puisque la fonction de coÃżt $L$ (intervenant dans $R$ et $R_n$) est à valeur dans $[0, 1]$. On se place alors dans le cas d'application du théorème de McDiarmid pour la fonction $\Phi$ considérée et $c_i = 1/n, \forall i$ et on a le résultat :

$$\forall \epsilon > 0, \mathbb{P}\left(\sup_{c \in \mathcal{G}} [R(c) - R_n(c, Z_l)] - \mathbb{E}_{Z_l} \sup_{c \in \mathcal{G}} [R(c) - R_n(c, Z_l)] > \epsilon\right) \leq e^{-2n\epsilon^2}$$

***Étape 2 : borner*** $\mathbb{E}_{Z_l} \sup_{c \in \mathcal{G}} [R(c) - R_n(c, Z_l)]$ ***par la compléxité de Rademacher*** $\mathcal{R}(L \circ \mathcal{G})$

Cette étape est une étape de *symmétrisation* puisqu'elle consiste à introduire dans $\mathbb{E}_{Z_l} \sup_{c \in \mathcal{G}} [R(c) - R_n(c, Z_l)]$ un second ensemble $Z_l^{'}$ lui aussi échantillonné suivant $\mathcal{D}^n$ et jouant un rÃŵle symétrique par rapport à $Z_l$. Cette étape est la plus technique dans l'obtention de la borne de généralisation et elle permet de trouver le premier point du théorème 3.

La majoration de $\mathbb{E}_{Z_l} \sup_{c \in \mathcal{G}} [R(c) - R_n(c, Z_l)]$ commence en remarquant que $R(c) = \mathbb{E}_{Z_l^{'}} R_n(c, Z_l^{'})$ et $R(n, Z_l) = \mathbb{E}_{Z_l^{'}} R(n, Z_l)$. On a alors

$$\begin{aligned} \mathbb{E}_{Z_l} \sup_{c \in \mathcal{G}} (R(c) - R_n(c, Z_l)) &= \mathbb{E}_{Z_l} \sup_{c \in \mathcal{G}} [\mathbb{E}_{Z_l^{'}} (R_n(c, Z_l^{'}) - R_n(c, Z_l))] \\ &\leq \mathbb{E}_{Z_l} \mathbb{E}_{Z_l^{'}} \sup_{c \in \mathcal{G}} [R(c, Z_l^{'}) - R_n(c, Z_l)] \end{aligned}$$

L'inégalité précédente est due au fait que le supremum d'une espérence est inférieur à l'espérence du suprémum. Le deuxième point dans cette étape consiste à introduire les variables de Rademacher dans le supremum :

$$\mathbb{E}_{Z_l}\mathbb{E}_{Z_l'}\sup_{c\in\mathcal{G}}\left[\frac{1}{n}\sum_{i=1}^{n}\sigma_i(L(c(x_i'),y_i')-L(c(x_i),y_i))\right]$$

Pour $i$ fixé, cette introduction à l'effet suivant : $\sigma_i = 1$ ne change rien mais $\sigma_i = -1$ revient à intervertir les deux exemples $(x_i', y_i')$ et $(x_i, y_i)$. Ainsi lorsque l'on prend les espérences sur $Z_l$ et $Z_l^{'}$, l'introdcution ne change rien. Ceci étant vrai pour tout $i$, on a, en prenant l'espérence sur $\sigma = (\sigma_1, ..., \sigma_n)$ :

$$\mathbb{E}_{Z_l}\mathbb{E}_{Z_l^{'}}\sup_{c\in\mathcal{G}}[R(c,Z_l^{'})-R_n(c,Z_l)] = \mathbb{E}_{Z_l}\mathbb{E}_{Z_l^{'}}\mathbb{E}_{\sigma}\sup_{c\in\mathcal{G}}\left[\frac{1}{n}\sum_{i=1}^{n}\sigma_i(L(c(x_i'),y_i')-L(c(x_i),y_i))\right]$$

En appliquant l'inégalité triangulaire sur $\sup = ||.||_\infty$ il vient

$$\begin{aligned}&\mathbb{E}_{Z_l}\mathbb{E}_{Z_l'}\mathbb{E}_{\sigma}\sup_{c\in\mathcal{G}}\left[\frac{1}{n}\sum_{i=1}^{n}\sigma_i(L(c(x_i'),y_i')-L(c(x_i),y_i))\right]\\ &\qquad\leq \mathbb{E}_{Z_l}\mathbb{E}_{Z_l'}\mathbb{E}_{\sigma}\sup_{c\in\mathcal{G}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i L(c(x_i'),y_i') + \mathbb{E}_{Z_l}\mathbb{E}_{Z_l'}\mathbb{E}_{\sigma}\sup_{c\in\mathcal{G}}\frac{1}{n}\sum_{i=1}^{n}(-\sigma_i)L(c(x_i'),y_i')\end{aligned}$$

Finalement comme $\sigma_i$ et $-\sigma_i$ ont la même distribution nous avons

$$\mathbb{E}_{Z_l}\mathbb{E}_{Z_l^{'}}\sup_{c\in\mathcal{G}}[R(c,Z_l^{'})-R_n(c,Z_l)] \leq \underbrace{2\mathbb{E}_{Z_l}\mathbb{E}_{\sigma}\sup_{c\in\mathcal{G}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i L(c(x_i),y_i)}_{\mathcal{R}(L\circ\mathcal{G})}$$

En récapitulant les résultats obtenus jusqu'à ce stade, nous avons :

1. $\forall c\in\mathcal{G}, \forall Z_l, R(c)-R_n(c,Z_l)\leq \sup_{c\in\mathcal{G}}[R(c)-R_n(c,Z_l)]$
2. $\forall\epsilon>0, \mathbb{P}\left(\sup_{c\in\mathcal{G}}[R(c)-R_n(c,Z_l)]-\mathbb{E}_{Z_l}\sup_{c\in\mathcal{G}}[R(c)-R_n(c,Z_l)]>\epsilon\right)\leq e^{-2n\epsilon^2}$
3. $\mathbb{E}_{Z_l}\sup_{c\in\mathcal{G}}(R(c)-R_n(c,Z_l))\leq\mathcal{R}(L\circ\mathcal{G})$

Et donc le premier point du théorème 3 s'obtient en résolvant pour $\epsilon$, l'équation $e^{-2n\epsilon^2}=\delta$.

***Étape 3 : borner*** $\mathcal{R}(L\circ\mathcal{G})$ ***en fonction de*** $\hat{\mathcal{R}}_n(L\circ\mathcal{G}, Z_l)$

Cette étape se réalise en appliquant à nouveau le théorème de McDiarmid à la fonction $\Phi : Z_l \mapsto \hat{\mathcal{R}}_n(L\circ\mathcal{G}, Z_l)$ avec $c_i = 2/n$ (Taylor and Cristianini, 2004). Nous obtenons dans ce cas :

$$\forall\epsilon>0, \mathbb{P}(\mathcal{R}(L\circ\mathcal{G})>\hat{\mathcal{R}}_n(L\circ\mathcal{G},Z_l)+\epsilon)\leq e^{-m\epsilon^2/2}$$

Cette borne donne avec le résultat précédent le troisième point du théorème 3.

### 2.4.2 Extension du cadre aux données interdépendantes

Nous avons vu que l'outil statistique permettant d'atteindre les résultats du théorème précédent est le théormème de McDiarmid. Ce théorème n'est pas applicable dans l'état à notre cadre d'étude ici puisqu'il suppose que les variables d'entrées soient des variables aléatoires indépendantes. En effet, en considérant un ensemble d'apprentissage $Z_l$ constitués de variables aléatoires indépendantes et l'ensemble transformaé $T(Z_l)$ nous souhaitons obtenir une borne sur $\sup_{c_f \in \mathcal{G}}[R^T(c_f) - \hat{R}^T_M(c_f, T(Z_l)]$, or le théorème de Mcdiarmid ne peut pas être utilisé sur la fonction :

$$T(Z_l) \mapsto \sup_{c_f \in \mathcal{G}}[R^T(c_f) - \hat{R}^T_M(c_f, T(Z_l))] \tag{2.7}$$

car cette fonction n'est pas décomposable en une somme de fonctions prenant en comptes des exemples transformés indépendants.

Pour effectuer l'étape 1 (section 2.4.1) nous avons dÃż étendre le théorème de McDiarmid en utilisant l'approche de décomposition de Janson de faÃğon à pouvoir prendre en compte des recouvrements propres exacts d'un ensemble transformé qui eux contiennent des variables aléatoires indépendantes. Ce résultat a été démontré dans l'article (Usunier et al., 2006), nous avons ainsi pu l'appliquer sur une borne supérieure de (2.7) qui elle est une combinaison linéaire de fonctions prenant en entrée des variables aléatoires indépendantes. L'extension du théorème de McDiarmid ainsi que la majoration de (2.7) sont données à la section 2.4.3. L'équivalent de l'étape 2 (section 2.4.1) est écrit à la section 2.4.4 et il débouchera sur une nouvelle mesure de complexité pour des classifieurs entraÃőnés avec des données interdépendantes appelée *Complexité de Rademacher fractionnaire*.

### 2.4.3 Extension du théorème de McDiarmid

Nous allons commencer la présentation par l'introduction d'une fonction $\Phi$ qui majore $T(Z_l) \mapsto \sup_{c_f \in \mathcal{G}}[R^T(c_f) - \hat{R}^T_M(c_f, T(Z_l))]$ et ensuite nous présenterons l'extension du théorème de Mcdiarmid qui elle s'applique à la fonction $\Phi$.

***Borner*** $\sup_{c_f \in \mathcal{G}}[R^T(c_f) - \hat{R}^T_M(c_f, T(Z_l))]$

Soit $Z_l$ une base d'apprentissage constituée d'exemples aléatoires indépendants et $\{\mathfrak{M}_j, w_j\}_{j \in \{1,\ldots,\Upsilon\}}$ un recouvrement propre exact (les ensembles $\mathfrak{M}_j$ sont indépendants et $\sum_{j=1}^{\Upsilon} w_j[[i \in \mathfrak{M}_j]] = 1$)

de l'ensemble transformé $T(Z_l)$. Dans ce cas toute somme de la forme $\sum_{i=1}^{M} t_i$ peut s'écrire :

$$\sum_{i=1}^{M} t_i = \sum_{i=1}^{M} \sum_{j=1}^{\Upsilon} w_j [[i \in \mathfrak{M}_j]] t_i = \sum_{j=1}^{\Upsilon} w_j \sum_{i \in \mathfrak{M}_j} t_i \tag{2.8}$$

En ré-écrivant maintenant $R^T(c_f) - R_M^T(c_f, T(Z_l))$ comme

$$R^T(c_f) - R_M^T(c_f, T(Z_l)) = \frac{1}{M} \sum_{i=1}^{M} \left( \mathbb{E}_{T(\tilde{Z}_l)} L(c_f(\tilde{\xi}_i), \tilde{d}_i) - L(c_f(\xi_i), d_i) \right)$$

Nous pouvons appliquer le résultat précédent (2.8) pour obtenir :

$$R^T(c_f) - R_M^T(c_f, T(Z_l)) = \frac{1}{M} \sum_{j=1}^{\Upsilon} w_j \left( \mathbb{E}_{T(\tilde{Z}_l)} \sum_{i \in \mathfrak{M}_j} L(c_f(\tilde{\xi}_i), \tilde{d}_i) - \sum_{i \in \mathfrak{M}_j} L(c_f(\xi_i), d_i) \right)$$

En prenant le sumpremum sur l'ensemble des fonctions et en remarquant que le suprémum d'une somme est inférieur à la somme des supremums nous trouvons :

$$\sup_{c_f \in \mathcal{G}} [R^T(c_f) - \hat{R}_M^T(c_f, T(Z_l))] \leq \sum_{j=1}^{\Upsilon} \frac{w_j}{M} \sup_{c_f \in \mathcal{G}} \left[ \mathbb{E}_{T(\tilde{Z}_l)} \sum_{i \in \mathfrak{M}_j} L(c_f(\tilde{\xi}_i), \tilde{d}_i) - \sum_{i \in \mathfrak{M}_j} L(c_f(\xi_i), d_i) \right]$$

Cette majoration consiste la première étape d'obtention de la borne de généralisation. Nous remarquons que le deuxième terme de l'inégalité précédente s'écrit comme une somme pondérée de variables indépendantes de la forme $\sum_{j=1}^{\Upsilon} w_j \phi_j((\xi_1, d_1), ..., (\xi_{|\mathfrak{M}_j|}, d_{|\mathfrak{M}_j|}))$ avec :

$$\phi_j : (\xi_i, d_i)_{i=1}^{|\mathfrak{M}_j|} \mapsto \frac{1}{M} \sup_{c_f \in \mathcal{G}} \mathbb{E}_{T(\tilde{Z}_l)} \sum_{i \in \mathfrak{M}_j} L(c_f(\tilde{\xi}_i), \tilde{d}_i) - \sum_{i \in \mathfrak{M}_j} L(c_f(\xi_i), d_i)$$

En effet, chacune des $\phi_j$ n'est fonction que de variables indépendantes : les seuls indices $i$ des variables $\xi_i$ dont $\phi_j$ va dépendre sont ceux appartenant à $\mathfrak{M}_j$

***Extension du théorème de McDiarmid***

Nous allons maintenant énoncer l'extension du théorème de McDiarmid qui considère des fonctions $\Phi$ partant d'un ensemble d'exemples interdépendants et pouvant s'écrire comme une combinaison de fonctions prennant en entrée des données indépendantes.

**Théorème 5** (Extension du théorème de McDiarmid aux données intérdépendantes)**.** *Soient $X_1, ..., X_n$, $n$ variables aléatoires indépendantes à valeur dans $\mathcal{X}$. Soient $T : \mathcal{X}^n \to \mathcal{T}^M$, $\{\mathfrak{M}_j, w_j\}_{j=1}^{\Upsilon}$ et $\Phi : \mathcal{T}^M \to \mathbb{R}$ telles que :*

1. *$\{\mathfrak{M}_j, w_j\}_{j=1}^{\Upsilon}$ est un recouvrement propre exact de $\{1, ..., M\}$ pour des varibles aléatoires $\{t_i\}_{i=1}^{M}$ avec $\mathfrak{M}_j = \{\mu_1^j, ..., \mu_{|\mathfrak{M}_j|}^j\}$,*
2. *$\sum_{j=1}^{\Upsilon} w_j = \chi^*(T)$,*
3. *Il existe $\Upsilon$ fonctions $\phi_1, ..., \phi_\Upsilon$ telles que :*
   - *$\forall j \in \{1, ..., \Upsilon\}, \phi_j : \mathcal{T}^{|\mathfrak{M}_j|} \to \mathbb{R}$,*
   - *$\forall t = (t_1, ..., t_M) \in \mathcal{T}^M, \Phi(t) = \sum_{j=1}^{\Upsilon} w_j \phi_j(\mu_1^j, ..., \mu_{|\mathfrak{M}_j|}^j)$,*
   - *$\exists (c_1, ..., c_M) \in \mathbb{R}^M$ tels que :*

$$\forall j \in \{1, ..., \Upsilon\}, \forall i \in \{1, ..., |\mathfrak{M}_j|\}, \forall t = (t_1, ..., t_{|\mathfrak{M}_j|}) \in \mathcal{T}^{|\mathfrak{M}_j|}, \forall t' \in \mathcal{T}$$
$$|\phi_j(t_1, .., t_{|\mathfrak{M}_j|}) - \phi_j(t_1, .., t_{i-1}, t', t_{i+1}, .., t_{|\mathfrak{M}_j|})| \leq c_{\mathfrak{M}_j}^i$$

*Alors*

$$\forall \epsilon > 0, \mathbb{P}\left(\mathbb{E}_{Z_l}(\phi \circ T) - (\phi \circ T)(Z_l)\right) \leq \exp\left(\frac{-2\epsilon^2}{\chi^*(T)\sum_{i=1}^{M} c_i^2}\right)$$

La preuve du théorème est donnée en Annexe B. l'interprétation des hypothèses est la suivante : la fonction $T$ crée un ensemble de variables aléatoires interdépendantes $t_i$ à partir d'une base d'apprentissage $Z_l$ formée de variables indépendantes. Avec la décomposition introduite dans (Janson, 2004) on décompose l'ensemble d'apprentissage transformé $T(Z_l)$ en sous-ensembles $\{\mathfrak{M}_j = \{\mu_1^j, ..., \mu_{|\mathfrak{M}_j|}^j\}\}$ de variables indépendantes. Si maintenant on peut trouver une fonction $\Phi$ sur l'ensemble transformé qui peut s'écrire comme une somme pondérée de fonctions $\phi_j$ prenant leur valeur sur chaque sous-ensemble indépendant $\mathfrak{M}_j$ et tel que chaque $\phi_j$ admet des différences bornées sur chacune de ses variables d'entrées alors on peut borner la fonction $S \mapsto (\phi \circ T)(S) - \mathbb{E}_S(\phi \circ T)$. Ceci est bien le cas de la borne supérieure de $\sup_{c_f \in \mathcal{G}}[R^T(c_f) - \hat{R}_M^T(c_f, T(Z_l))]$ qui s'écrit comme $\sum_{j=1}^{\Upsilon} w_j \phi_j((\xi_1, d_1), ..., (\xi_{|\mathfrak{M}_j|}, d_{|\mathfrak{M}_j|}))$. Les coefficients du théorème sont alors égaux à $c_i = \frac{1}{M}$. On a dans ce cas, pour tout $\delta \in ]0, 1]$ et avec une probabilité au moins $1 - \delta$

$$\sup_{c_f \in \mathcal{G}} \left(R^T(c_f) - R_M^T(c_f, T(Z_l)\right) \leq$$
$$\mathbb{E}_{T(Z_l)}\left(\sum_{j=1}^{\Upsilon} \frac{w_j}{M} \sup_{c_f \in \mathcal{G}}\left[\mathbb{E}_{T(\tilde{Z}_l)} \sum_{i \in \mathfrak{M}_j} L(c_f(\tilde{\xi}_i), \tilde{d}_i) - \sum_{i \in \mathfrak{M}_j} L(c_f(\xi_i), d_i)\right]\right) + \sqrt{\frac{\chi^*(T)\ln(1/\delta)}{2M}} \quad (2.9)$$

### 2.4.4 Compléxité de Rademacher fractionnaire

Nous allons étendre la notion de compléxité de Rademacher présentée à la section 2.4.1 au cas de données interdépendantes. Cette nouvelle compléxité sera obtenue en appliquant la méthode de symmétrisation à l'expréssion :

$$\mathbb{E}_{T(Z_l)}\left(\sum_{j=1}^{\Upsilon}\frac{w_j}{M}\sup_{c_f\in\mathcal{G}}\left[\mathbb{E}_{T(\tilde{Z}_l)}\sum_{i\in\mathfrak{M}_j}L(c_f(\tilde{\xi}_i),\tilde{d}_i)-\sum_{i\in\mathfrak{M}_j}L(c_f(\xi_i),d_i)\right]\right) \tag{2.10}$$

La première majoration ici est en sortant le suprémum de l'espérance :

$$\begin{aligned}&\mathbb{E}_{T(Z_l)}\left(\sum_{j=1}^{\Upsilon}\frac{w_j}{M}\sup_{c_f\in\mathcal{G}}\left[\mathbb{E}_{T(\tilde{Z}_l)}\sum_{i\in\mathfrak{M}_j}L(c_f(\tilde{\xi}_i),\tilde{d}_i)-\sum_{i\in\mathfrak{M}_j}L(c_f(\xi_i),d_i)\right]\right)\\&\qquad\leq\mathbb{E}_{T(Z_l),T(\tilde{Z}_l)}\sum_{j=1}^{\Upsilon}\frac{w_j}{M}\sup_{c_f\in\mathcal{G}}\left[\sum_{i\in\mathfrak{M}_j}\Big(L(c_f(\tilde{\xi}_i),\tilde{d}_i)-L(c_f(\xi_i),d_i)\Big)\right]\end{aligned}$$

En considérant $\sigma=(\sigma_1,...,\sigma_M)$, une réalisation de $M$ variables de Rademacher indépendantes, nous avons pour chaque somme sur $\mathfrak{M}_j$

$$\begin{aligned}&\mathbb{E}_{T(Z_l),T(\tilde{Z}_l)}\sup_{c_f\in\mathcal{G}}\left[\sum_{i\in\mathfrak{M}_j}\sigma_i\Big(L(c_f(\tilde{\xi}_i),\tilde{d}_i)-L(c_f(\xi_i),d_i)\Big)\right]\\&\qquad=\mathbb{E}_{T(Z_l),T(\tilde{Z}_l)}\sup_{c_f\in\mathcal{G}}\left[\sum_{i\in\mathfrak{M}_j}\Big(L(c_f(\tilde{\xi}_i),\tilde{d}_i)-L(c_f(\xi_i),d_i)\Big)\right]\end{aligned}$$

L'introduction de $\sigma_i$ pour chacune de ces sommes ne change rien et le terme $-\sigma_i$ correspond à une échange des exemples $(\tilde{\xi}_i,\tilde{d}_i)$ et $(\xi_i,d_i)$. Dans chacune des sommes, $(\xi_i,d_i)$ sont indépendants donc un cet échange, s'il y a lieu, est sans effet sur les autres termes de la soomes. De plus, lorsque l'on prend l'espérence sur $T(Z_l)$ et $T(\tilde{Z}_l)$ la valeur de $\sigma_i$ devient sans effet sur le terme considéré car $(\tilde{\xi}_i,\tilde{d}_i)$ et $(\xi_i,d_i)$ ont la même distribution. Comme les $\sigma_i$ ont la même distribution de probabilité nous avons comme dans le cas de la classification binaire :

$$\begin{aligned}&\mathbb{E}_{T(Z_l),T(\tilde{Z}_l)}\sup_{c_f\in\mathcal{G}}\left[\sum_{i\in\mathfrak{M}_j}\sigma_i\Big(L(c_f(\tilde{\xi}_i),\tilde{d}_i)-L(c_f(\xi_i),d_i)\Big)\right]\\&\qquad=\mathbb{E}_{T(Z_l),T(\tilde{Z}_l)}\mathbb{E}_{\sigma}\sup_{c_f\in\mathcal{G}}\left[\sum_{i\in\mathfrak{M}_j}\sigma_i\Big(L(c_f(\tilde{\xi}_i),\tilde{d}_i)-L(c_f(\xi_i),d_i)\Big)\right]\end{aligned}$$

En utilisant l'inégalité triangulaire sur le suprémum dans la seconde inégalité, on peut majorer (2.10) :

$$\mathbb{E}_{T(Z_l)}\left(\sum_{j=1}^{\Upsilon}\frac{w_j}{M}\sup_{c_f\in\mathcal{G}}\left[\mathbb{E}_{T(\tilde{Z}_l)}\sum_{i\in\mathfrak{M}_j}L(c_f(\tilde{\xi}_i),\tilde{d}_i)-\sum_{i\in\mathfrak{M}_j}L(c_f(\xi_i),d_i)\right]\right) \tag{2.11}$$
$$\leq \mathbb{E}_{T(Z_l)}\frac{2}{M}\mathbb{E}_\sigma\sum_{j=1}^{\Upsilon}w_j\sup_{c_f\in\mathcal{G}}\sum_{i\in\mathfrak{M}_j}\sigma_i L(c_f(\xi_i),d_i)$$

Nous définissons alors la compléxité de *Rademacher fractionnaire empirique* de $\mathcal{G}$ par :

$$\mathcal{R}_M^T(\mathcal{G},T(Z_l))=\frac{2}{M}\mathbb{E}_\sigma\sum_{j=1}^{\Upsilon}w_j\sup_{c_f\in\mathcal{G}}\sum_{i\in\mathfrak{M}_j}\sigma_i c_f(\xi_i)$$

Et la compléxité de *Rademacher fractionnaire* est $\mathcal{R}^T(\mathcal{G})=\mathbb{E}_{T(Z_l)}\mathcal{R}_M^T(\mathcal{G},T(Z_l))$. Nous remarquons que la compléxité de Rademacher fractionnaire empirique est une somme pondérée de compléxités de Rademacher sur des sous-ensembles indépendants de l'ensemble transformé. Le terme fractionnaire vient de celui du recouvrement fractionnaire trouvé sur les exemples transformés Janson (2004).

Avec la définition précédente et d'après les équations (2.9) et (2.11) nous avons $\forall\delta\in]0,1]$ l'inégalité suivante qui est vraie avec une probabilité au moins égale à $1-\delta$ :

$$\forall Z_l,\forall c_f\in\mathcal{G},R^T(c_f)\leq R_M^T(c_f,T(Z_l))+\mathcal{R}^T(L\circ\mathcal{G})+\sqrt{\frac{\chi^*(T)\ln(1/\delta)}{2M}}$$

La fontion $T(Z_l)\mapsto\mathcal{R}_M^T(\mathcal{G},T(Z_l))$ vérifie elle aussi les conditions du théorème 5 avec $\forall i,c_i=\frac{2}{M}$ nous obtenons le résultat final suivant qui est une borne dépendante des données.

**Théorème 6.** *Soit $T:Z_l\to(\mathcal{T}\times\{-1,+1\})^M$ une fonction de transformation, prenant en entrée des bases d'apprentissage $Z_l$ formées par $m$ variables aléatoires indépendantes à valeurs dans $\mathcal{X}$. Soit $\mathcal{G}$ une classe de fonctions de $\mathcal{T}$ vers $\{-1,+1\}$. On a alors pour tout $\delta\in]0,1]$ :*

$$\mathbb{P}\left(\forall c_f\in\mathcal{G},R^T(c_f)\leq R_M^T(c_f,T(Z_l))+\mathcal{R}_M^T(L\circ\mathcal{G})+3\sqrt{\frac{\chi^*(T)\ln(1/\delta)}{2M}}\right)\geq 1-\delta$$

### 2.4.5 Estimation des bornes pour quelques exemples d'application

Pour mieux voir l'application du théorème 6 au cas particulier d'ordonnancement bipartite présenté précedemment. Nous allons d'abord majorer la complexité Rademacher fractionnaire empirique pour une classe de fonctions à noyaux à norme bornée dont le suprémum peut être calculé sur un ensemble d'apprentissage et ensuite nous donnerons la borne de généralisation pour le cas d'ordonnancement bipartite.

***Borner*** $\mathcal{R}_M^T(\mathcal{G}_B)$ ***pour*** $\mathcal{G}_B = \{\xi \mapsto \langle w, \xi\rangle \mid \langle w, w\rangle \leq B^2\}$

La majoration découle de la propriété équivalente de la compléxité de Rademacher (Taylor and Cristianini, 2004) page 100, ainsi que de la propriété (2.8) des récouvrements propres exacts. Pour la clarté de la présentation nous allons donner ce calcul. Remarqons d'abord que pour une fonction $c_f \in \mathcal{G}$, nous avons d'après l'inégalité de Cauchy Schwartz $\forall \xi \in \mathcal{T}, c_f(\xi) \leq B\|\xi\|$. En utilisant la forme bilinéaire du produit scalaire et le résultat précédent nous pouvons alors écrire :

$$\mathcal{R}_M^T(\mathcal{G}_B) = \sum_{j=1}^{\Upsilon} \frac{w_j}{M} \mathbb{E}_\sigma \frac{2}{M_j} \sup_{c_f \in \mathcal{G}} \sum_{i \in \mathfrak{M}_j} \sigma_i c_f(\xi_i) \leq 2B \sum_{j=1}^{\Upsilon} \frac{w_j}{M} \mathbb{E}_\sigma(\| \sum_{i \in \mathfrak{M}_j} \sigma_i \xi_i \|)$$

En écrivant $\|x\| = \sqrt{\langle x, x\rangle}$ et en utilisant l'inégalité de Jensen avec la fonction concave $t \mapsto \sqrt{t}$, on obtient :

$$\mathcal{R}_M^T(\mathcal{G}_B) \leq 2B \sum_{j=1}^{\Upsilon} \frac{w_j}{M} \sqrt{\mathbb{E}_\sigma \left[ \sum_{k,l \in \mathfrak{M}_j} \sigma_k \sigma_l K(\xi_k, \xi_l) \right]}$$

Pour $k \neq l$, il y a quatre possibilités de valeurs pour des variables de Rademacher $\sigma_k$, $\sigma_l$ prenant des valeurs $-1$ et $+1$ avec des probabilités égales à $\frac{1}{2}$. L'effet de combinaison dans $\sigma_k \sigma_l K(\xi_k, \xi_l)$ s'annule alors et l'inégalité précédente devient

$$\mathcal{R}_M^T(\mathcal{G}_B) \leq 2B \sum_{j=1}^{\Upsilon} \frac{w_j}{M} \sqrt{\left[ \sum_{k \in \mathfrak{M}_j} K(\xi_k, \xi_k) \right]} = \frac{2B\chi^*(T)}{M} \sum_{j=1}^{\Upsilon} \frac{w_j}{\chi^*(T)} \sqrt{\left[ \sum_{k \in \mathfrak{M}_j} K(\xi_k, \xi_k) \right]}$$

en remarquant que $\sum_{j=1}^{\Upsilon} w_j = M$ et utilisant à nouveau l'inégalité de Jensen avec la fonction concave $t \mapsto \sqrt{t}$ et d'après (2.8) on obtient alors

$$\mathcal{R}_M^T(\mathcal{G}_B) \leq 2B \frac{\sqrt{\chi^*(T)}}{M} \sqrt{\sum_{i=1}^{M} K(\xi_i, \xi_i)}$$

***Ordonnancement bipartite***

Rappelons que nous voulons dans ce cas ordonner les $p$ exemples positifs d'un ensemble d'apprentissage $Z_l = ((x_i, y_i)_{i=1}^n)$ au-dessus de ses $k$ exemples négatifs (on suppose $p \leq k$). La transformation $T : \mathcal{X}^n \to \mathcal{T}^M$ organise les $M = pk$ exemples de $T(Z_l)$ en formant des

paires qui contiennent un élement positif avec un élement négatif de la faÃğon suivante :

$$\begin{array}{rcl} \mathfrak{M}_1 & : & \left(x_{\pi(1)}, x_{\nu(1)}\right), ..., \left(x_{\pi(p)}, x_{\nu(p)}\right) \\ \mathfrak{M}_1 & : & \left(x_{\pi(1)}, x_{\nu(1)}\right), ..., \left(x_{\pi(p)}, x_{\nu(p+1)}\right) \\ & & ... \\ \mathfrak{M}_k & : & \left(x_{\pi(1)}, x_{\nu(k)}\right), ..., \left(x_{\pi(p)}, x_{\nu(p-1)}\right) \end{array}$$

Où $\pi(i)$ (resp. $\nu(j)$) représente l'indice du $i$-ième exemple positif (resp. $j$-ième exemple négatif) de $T(Z_l)$. Cette décomposition correspond à un recouvrement propre exact de $T(Z_l)$ illustré dans la figure 2.1 (c) avec $\chi^*(T) = \max(k, p)$. En considérant la fonction de coÃżt $\Delta(y, c) = \min(1, \max(1 - yc, 0))$ qui est 1-Lipshitzienne et $c \in \mathcal{G}_B$. Nous avons d'après le théorème 6

$$\forall \delta \in ]0, 1], \mathbb{P}\left(\forall c_f \in \mathcal{G}_B, R^T(c_f) \leq R^T_M(c_f, T(Z_l)) + \mathcal{R}^T_M(\Delta \circ \mathcal{G}_B) + 3\sqrt{\frac{\chi^*(T)\ln(1/\delta)}{2M}}\right) \geq 1-\delta$$

De plus nous pouvons démontrer d'après les propriétés de la compléxité de Rademacher que $\mathcal{R}^T_M(\Delta \circ \mathcal{G}_B) \leq \mathcal{R}^T_M(\mathcal{G}_B)$ (Taylor and Cristianini (2004), p.101) ce qui d'après le calcul précédent et le théorème 6 donne une borne de généralisation de fonctions d'ordonnancement bipartite calculable sur les données d'apprentissage :

$$\mathbb{P}\left(\begin{array}{ll} \forall c_f \in \mathcal{G}_B, & \\ R^T(c_f) \leq & R^T_M(c_f, T(Z_l)) + \\ & \frac{2B\sqrt{\max(k,p)}}{kp}\sqrt{\sum_{i=1}^{p}\sum_{j=1}^{k}\|x_{\pi(i)} - x_{\nu(j)}\|^2_K} + 3\sqrt{\frac{\ln(1/\delta)}{2\min(k,p)}} \end{array}\right) \geq 1 - \delta$$

***Classification binaire***

Le cadre de la classification binaire est un cas particulier de notre cadre et il correspond à la fonction de transformation $T$ identité ($\chi^*(T) = 1$). Dans ce cas en prenant la fonction de coÃżt $h : t \mapsto \max(1 - t, 0)$ nous retrouvons la borne de généralisation de classifieurs entraÃőnés sur des données i.i.d (Taylor and Cristianini (2004), page 102) [4] :

$$\mathbb{P}\left(\forall f \in \mathcal{G}_B, \mathbb{E}[[yf(x) \leq 0]] \leq \frac{1}{n}\sum_{i=1}^{n} h(y_i f(x_i)) + \frac{2B}{n}\sqrt{\sum_{i=1}^{n} K(x_i, x_i)} + 3\sqrt{\frac{\ln(1/\delta)}{2n}}\right) \geq 1-\delta$$

4. Dans le cas où on définit la compléxité de Rademacher avec des valeurs absolues, comme dans Taylor and Cristianini (2004), le terme $\frac{2B}{n}$ devient $\frac{4B}{n}$.

## 2.5 Conclusion

Nous avons proposé ici un formalisme qui est un généralisation des bornes de généralisation utilisant la compléxité de Rademacher. Nous obtenons ainsi les mêmes résultats qur la cas particulier de la classification d'exemples i.i.d et avec notre cadre nous sommes capables d'inférer de nouvelles bornes pour des problèmes qui peuvent être définis comme la classification de données interdépendantes avec une fonction de transformation déterministe donnée.

# 3 APPRENTISSAGE ACTIVE POUR L'ORDONNANCEMENT

Dans ce chapitre nous allons présenter une stratégie d'apprentissage actif de fonctions d'ordonnancement d'alternatives. La motivation principale qui nous a conduit à l'élaboration de cette stratégie est que pour trouver une fonction d'ordonnancement efficace il est nécessaire d'avoir une base d'apprentissage qui demande souvent l'étiquetage manuel des alternatives sur plusieurs exemples. Comme pour le cas d'apprentissage de fonctions de classification, notre étude ici vise à réduire cet effort d'étiquetage qui pour les tâches comme la plupart des applications de RI (que nous considérons dans la suite de ce chapitre) peut-être qualifié d'irréaliate.

Différentes stratégies d'apprentissage actif ont été proposées dans le cadre de la classification. Une méthode phare est l'échantillonnage sélective qui consiste à sélectionner un ou plusieurs exemples d'un ensemble non-étiqueté et à intérroger un oracle pour obtenir leur étiquette. Ces nouveaux exemples sont ensuite ajoutés à l'ensemble d'entraînement pour apprendre un nouveau classifieur. Ces stratégies actives ont été développées autour de deux idées centrales. $(a)$ La réduction de l'espace des versions (Cohen et al., 1996), qui dans le cas des fonctions linéaires discriminantes, consiste à sélectionner l'exemple non-étiqueté ayant la plus faible marge avec la frontière de décision en cours (Tong and Koller, 2002), et $(b)$ la sélection d'exemples non-étiquetés réduisants une approximation de l'erreur de généralisation (Chapelle, 2005).

Les motivations théoriques qui ont servi au développment de ces techniques ne peuvent malheureusement pas être étendues au cadre de l'ordonnacement. Il n'y a en effet aucune équivalence à la notion de l'espace des versions dans ce cas et les approximations de l'erreur de généralisation étaient jusqu'à maintenant inexistantes. La notion de marge peut néanmoins être étendue dans le cas où on cherche à prédire un ordre total sur les exemples avec une fonction à valeur réelle. Brinker (2004) a ainsi montré que la sélection des alternatives avec la plus petite différence de scores peut être assimilée à une notion de marge qu'il a qualifié de *marge étendue* et qui s'est avérée être une approche heuristique efficace en pratique. Dans les cas de prédictions d'ordre partiels qui nous intéressent ici, cette heurisitque devient néanmoins inopérante.

En effet les scores de deux alternatives pertinentes ou non-pertinentes peuvent être très proches et pour une requête donnée, la marge étendue peut être nulle indépendamment du fait que les alternatives pertinentes doivent avoir un score plus élevé que les alternatives non-pertinentes.

L'approche d'apprentissage actif que nous avons développée est une stratégie d'échnatillonnage sélective basée sur une nouvelle borne de généralisation pour des fonctions d'ordonnancement d'alternatives utilisant des données non–étiquetées. Notre stratégie est issue d'un résultat similaire proposé par (Kääriäinen, 2005) dans le cadre de la classification où l'erreur de généralisation d'un classifieur est bornée par celle d'un autre classifieur et un deuxième terme qui fait intervenir le désaccord entre ces deux classifieurs sur les étiquettes des exemples non-étiquetés. L'extension de ce résultat a nécessité la définition d'une notion de *divergence* entre deux fonctions d'ordonnancement.

J'ai réalisé la partie théorique de ce travail avec Nicolas Usunier et Vinh Truong. Nous avons ensuite développé la stratégie sélective en collaboration avec François Laviolette, professeur à l'université Laval à Québec et son thésard Alexandre Lacasse.

Le plan de ce chapitre est comme suit : à la section 3.1 nous allons donner les définitions qui vont nous servir à établir notre cadre et présenterons une borne de test de fonctions d'ordonnancement sur la base des données étiquetées et non-étiquetées. Nous allons montrer à la section 3.2 l'extension de cette borne vers une borne de généralisation et présenterons notre stratégie d'échantillonnage sélective sur la base de ce résultat. À la section 3.4 nous présenterons les résultats que nous avons obtenus avec cette stratégie pour la tâche de résumé de texte.

## 3.1 Borne de test avec des données non-étiquetées

Nous rappelons que l'ensemble des alternatives $\mathcal{A}$ est désigné par $\{1, ..., A\}$ et les espaces d'entrée et de sortie sont respectivement notés par $\mathcal{X}$ et $\mathcal{L}$. En ordonnant la sortie d'une fonction de score $f : \mathcal{X} \to \mathbb{R}$, on peut définir une fonction de rang associée $\bar{f} : \mathcal{X} \to \sigma_A$, Où $\sigma_A$ est l'ensemble des permutations sur $\{1, ..., A\}$. Ainsi pour un exemple $x \in \mathcal{X}$, une alternative $i \in \mathcal{A}$ est préférée sur une alternative $j \in \mathcal{A}$ ssi $\bar{f}(x)(i) < \bar{f}(x)(j)$. Nous supposons que l'ensemble d'apprentissage est formé d'une base *étiquetée* $Z_l = ((x_i, \ell_i))_{i=1}^n \in \mathcal{Z}^n$ et d'une base *non-étiquetée* $X_u = (x'_j)_{j=n+1}^{n+m} \in \mathcal{X}^m$, où $\mathcal{Z}$ représente l'ensemble $\mathcal{X} \times \mathcal{L}$. Ainsi chaque paire $(x, l) \in Z_l$ est échantillonnée i.i.d suivant une distribution fixe mais inconnue $\mathcal{D}$ et on note la distribution marginale sur $\mathcal{X}$ par $D_{\mathcal{X}}$.

Nous rappelons de plus que pour chaque exemple $x$, il y a juste un sous-ensemble $\mathcal{A}_x$ de $\mathcal{A}$ qui est considéré, et que $\mathcal{A}_x$ est connu même si l'étiquette de $x$ est inconnue. De plus, l'ensemble des étiquettes possibles pour $x$, noté $\mathcal{L}_x$, ne contient que les relations de préférences sur $\mathcal{A}_x$.

Ainsi, lorsque les étiquettes sont induites par des jugements de pertinence binaires, n'importe que étiquette de $\mathcal{L}_x$ peut être représentée par deux ensembles d'indices $Y_x^+$ et $Y_x^-$ d'alternatives pertinentes et non-pertinentes de $\mathcal{A}_x$.

Ces notations permettent de formuler naturellement des fonctions de coût en RI. Par exemples, la précision@$k$ qui comptabilise la proportion d'alternatives pertinentes parmi les $k$ premiers rangs peut-être défini comme :

$$c_{p@k}(\bar{f}(x), \ell) = \frac{1}{k} \sum_{i \in Y_x^+} [[\bar{f}(x)(i) \leq k]] \tag{3.1}$$

Un autre exemple est le nombre moyen d'alternatives non-pertinentes ordonnées au-dessus d'alternatives pertinentes par $\bar{f}$ (déja présenté au chapitre **??**) :

$$c_{Rloss}(\bar{f}(x), \ell) = \frac{1}{|Y_x^+||Y_x^-|} \sum_{j \in Y_x^-} \sum_{i \in Y_x^+} [[\bar{f}(x)(j) < \bar{f}(x)(i)]] \tag{3.2}$$

Finalement, nous désignons par $\hat{\epsilon}_Z(\bar{f}) = \frac{1}{n} \sum_{i=1}^{n} c(\bar{f}(x_i), \ell_i)$ le risque empirique de la fonction $\bar{f}$ et par $\epsilon(\bar{f}) = \mathbb{E}_{(x,\ell)\sim\mathcal{D}} c(\bar{f}(x), \ell)$ son vrai risque.

### 3.1.1 Fonctions d'ordonnancements de Gibbs

Nous définissons une fonction de rang aléatoire comme une variable aléatoire à valeur dans $\sigma_A$ tel que pour chaque exemple $x$, une fonction de rang aléatoire $\bar{f}_\theta$ est choisie suivant une distribution de probabilité $\Theta$ sur un ensemble fini de fonctions de rang $\{\bar{f}_1, ..., \bar{f}_K\}$. Si $\Theta$ est la distribution uniforme on note la fonction de rang aléatoire correspondante par $\bar{f}_K$.

L'erreur de généralisation que nous proposons dans la suite est basée sur une fonction de divergence $d_c : \sigma_A \times \sigma_A \to [0, 1]$ associée à la fonction de risque $c$ et qui mesure pour chaque exemple $x \in \mathcal{X}$ le désaccord entre deux fonctions de rang $\bar{f}$ et $\bar{f}'$ sur l'ordonnancement des alternatives associées à $x$. Nous définissons $d_c$ par

$$d_c(\bar{f}(x), \bar{f}'(x)) = \max_{\ell \in \mathcal{L}_x} \left[ c(\bar{f}(x), \ell) - c(\bar{f}'(x), \ell) \right]$$

$d_c$ est clairement une mesure de divergence bornée par 1. Nous avons en effet $1 \geq d_c(y, y') \geq 0$ pour tout $y$, $y'$ et $d_c(y, y') = 0$ ssi $y = y'$. De plus, nous avons :

$$\forall (x, \ell) \in \mathcal{Z},\ c(\bar{f}(x), \ell) \leq c(\bar{f}'(x), \ell) + d_c(\bar{f}(x), \bar{f}'(x))$$

Pour deux fonctions de rangs aléatoires $\bar{f}_\Theta$ et ${\bar{f}'}_\Lambda$ la notion du risque peut être étendue en posant $c(\bar{f}_\Theta(x), \ell) = \mathbb{E}_{\theta\sim\Theta} c(\bar{f}_\theta(x), \ell)$ et $d_c(\bar{f}_\Lambda(x), {\bar{f}'}_\Theta(x)) = \mathbb{E}_{\lambda\sim\Lambda,\theta\sim\Theta} d_c(\bar{f}_\lambda(x), {\bar{f}'}_\theta(x))$.

De ces définitions il vient :

$$\forall (x,\ell) \in \mathcal{Z},\ c(\bar{f}_\Lambda(x),\ell) \leq c(\bar{f'}_\Theta(x),\ell) + d_c(\bar{f}_\Lambda(x),\bar{f'}_\Theta(x)) \tag{3.3}$$

L'équation (3.3) montre un lien entre les valeurs d'une fonction de risque $c$ sur $\bar{f}_\Lambda$ et $\bar{f}_\Theta$ et indique ainsi un lien possible entre le risque de ces deux fonctions de rangs. Dans le cas stochastique, le vrai risque et le risque empirique d'une fonction de rang aléatoire $\bar{f}_\Theta$ sont définis par :

$$\hat{\epsilon}_Z(\bar{f}_\Theta) = \frac{1}{n}\sum_{i=1}^{n} c(\bar{f}_\Theta(x_i),\ell_i) = \mathbb{E}_{\theta\sim\Theta}\frac{1}{n}\sum_{i=1}^{n} c(\bar{f}_\theta(x_i),\ell_i)$$
$$\epsilon(\bar{f}_\Theta) = \mathbb{E}_{(x,\ell)\sim\mathcal{D}} c(\bar{f}_\Theta(x),\ell) = \mathbb{E}_{\theta\sim\Theta}\mathbb{E}_{(x,\ell)\sim\mathcal{D}} c(\bar{f}_\theta(x),\ell)$$

### 3.1.2 Divergence entre fonctions d'ordonnancements

Nous remarquons que si $Z$ est échantillonné i.i.d. suivant $\mathcal{D}$, alors $\hat{\epsilon}_Z(\bar{f}_\Theta)$ est un estimateur non-biaisé de $\epsilon(\bar{f}_\Theta)$. De même si $X_u$ est échantillonné i.i.d. suivant $\mathcal{D}_\mathcal{X}$, la moyenne de $d_c(\bar{f}_\Lambda(x'),\bar{f'}_\Theta(x'))$ pour $x' \in X_u$ est un estimateur non-biaisé de $\mathbb{E}_{x'\sim\mathcal{D}_\mathcal{X}} d_c(\bar{f}_\Lambda(x'),\bar{f'}_\Theta(x'))$. Le théorème suivant est alors une extension de celui présenté de (Kääriäinen, 2005) proposé dans le cadre de la classification.

**Théorème 7.** *Soient $\bar{f}_\Lambda$ et $\bar{f'}_\Theta$ deux fonctions de rangs, nous avons alors :*

$$\epsilon(\bar{f}_\Lambda) \leq \epsilon(\bar{f'}_\Theta) + \mathbb{E}_{x'\sim\mathcal{D}_\mathcal{X}} d_c(\bar{f}_\Lambda(x'),\bar{f'}_\Theta(x'))$$

*Preuve* : Ce résultat découle directement de l'inégalité 3.3 en prenant l'espérence sur $(x,\ell) \sim \mathcal{D}$.

S'il existe ainsi une borne sur l'erreur de $\bar{f'}_\Theta$ il est alors possible d'obtenir une borne supérieure sur l'erreur de $\bar{f}_\Lambda$ en utilisant les données non–étiquetées. Nous allons voir comment on peut obtenir une telle borne avec une fonction de rang $\bar{f'}_\Theta$ estimée sur des ensembles cross-validatés d'un ensemble d'apprentissage étiquetée de départ. Ces ensembles sont définis comme :

**Définition 8** (Ensembles de cross-validation)**.** *Soit un ensemble étiqueté $Z$ échantillonné i.i.d. suivant la distribution $\mathcal{D}$, un ensemble de cross-validation (CV) de taille $K$ est n'importe quel partitionnement de $Z$ en $K$ sous-ensembles disjoints $Z_1,...,Z_K$ de taille égale*[1]*. De plus, pour tout ensemble $Z_1,...,Z_K$, nous posons $i = 1,...,K$, $Z_i^{app} = \bigcup_{j\neq i} Z_j$ et $Z_i^{test} = Z_i$.*

1. Les résulats demeurent valides si la taille des sous-ensembles cross-validés ne divise pas $|Z|$, Mais afin de simlifier la présentation, nous nous restreignons au cas où la taille de ces sous-ensembles divise $|Z|$.

Ainsi, étant donné un algorihme d'apprentissage de fonctions de rang $\mathcal{R}$, et un ensemble CV de taille $K$, $\{Z_1, ..., Z_K\}$ sur $Z$, une *fonction aléatoire de rang obtenue par cross-validation* est une fonction aléatoire définie avec une distribution de probabilité uniforme sur l'ensemble $\{\mathcal{R}(Z_1^{app}), \ldots, \mathcal{R}(Z_K^{app})\}$. Nous posons $\bar{f}_K^{cv}$, la fonction aléatoire ainsi obtenue, et notons par $\bar{f}_j$ la fonction $\mathcal{R}(Z_j^{app})$. Les résultats suivants montrent comment une borne sur le risque d'une telle fonction peut être estimée, et comment le théorème 7 peut être appliqué en pratique. Ces résultats sont basés sur la version suivante de la borne de Hoeffding :

**Théorème 9** (Borne de Hoeffding)**.** *Soient $X_1, ..., X_n$, $n$ copies d'une variable aléatoire $X$ à valeur dans $[0, 1]$, pour tout $\delta > 0$ nous avons :*

$$\mathbb{P}\Big(\mathbb{E}X \leq \tfrac{1}{n}\sum_{i=1}^{n} X_i + \sqrt{\tfrac{\ln(1/\delta)}{2n}}\Big) > 1 - \delta$$

En combinant le théorème 7 avec la borne de Hoeffding nous obtenons la borne suivante sur le risque de $\bar{f}_K^{cv}$

**Lemme 10.** *Soit $Z$ un échantillon de taille $n$ obtenu i.i.d. suivant la distribution $\mathcal{D}$ et soit $\{Z_1, ..., Z_K\}$ un ensemble CV de taille $K$ tel que $K$ divise $n$. Nous avons alors une borne sur l'erreur de $\bar{f}_K^{cv}$ avec une probabiltié au moins égale à $1 - \delta/2$ :*

$$\epsilon(\bar{f}_K^{cv}) \leq \tfrac{1}{K}\sum_{j=1}^{K} \hat{\epsilon}_{Z_j}(\bar{f}_j^{cv}) + \sqrt{\tfrac{K}{2n}\ln\tfrac{2K}{\delta}}$$

*Preuve* : Pour tout $j \in \{1, \ldots, K\}$, la borne de Hoeffding donne

$$\mathbb{P}\left(\epsilon(\bar{f}_j^{cv}) > \hat{\epsilon}_{Z_j}(\bar{f}_j^{cv}) + \sqrt{\frac{K}{2n}\ln\frac{2K}{\delta}}\right) \leq \frac{\delta}{2K}$$

Le résultat de la lemme est alors une application de la borne de l'union $\mathbb{P}(\cup A_i) \leq \sum \mathbb{P}(A_i)$ sur les $K$ inégalités précédentes.

La lemme suivante borne $\mathbb{E}d_c(\bar{f}(x'), \bar{f}_K^{cv}(x'))$ avec sa valeur empirique estimé sur un ensemble non-étiqueté.

**Lemme 11.** *Soit $X_u$ un ensemble non-étiqueté obtenu indépendamment de l'ensemble étiqueté $Z$, et soit $\bar{f}$ une fonction de rang apprise indépendamment du sous-ensemble $X_u^{(k)} = \{x'_{j_1}, ..., x'_{j_k}\}$ de $X_u$ de taille $k$. Nous avons alors :*

$$\mathbb{P}\left(\mathop{\mathbb{E}}_{x'\sim\mathcal{D}_\mathcal{X}} d_c(\bar{f}(x'), \bar{f}_K^{cv}(x')) \leq \frac{1}{k}\sum_{l=1}^{k} d_c\left(\bar{f}(x'_{j_l}), \bar{f}_K^{cv}(x'_{j_l})\right) + \sqrt{\frac{\ln(\frac{2}{\delta})}{2k}}\right) \quad > \quad 1 \; - \; \frac{\delta}{2}$$

*Preuve* : Comme $d_c$ est une variable aléatoire à valeur dans $[0,1]$ et comme les exemples $x' \in X_u^{(k)}$, sont échantillonné i.i.d., le résultat du théorème est obtenu grâce à au théorème 9, [avec $\delta := \delta/2$, $n := k$, $X := d_c(\bar{f}(x'), \bar{f}_K^{cv}(x'))$, $X_i := d_c(\bar{f}(x'_{j_i})\bar{f}_K^{cv}(x'_{j_i}))$ ].

Le théroème 7, avec les deux lemmes précedentes donnent une borne supérieure sur le risque de n'importe quelle fonction de rang. Cette borne peut être estimée précisement avec les données étiquetées et un sous-ensemble de données non-étiquetées, pourvue que $n/K$ et la taille de ce dernier soient assez grands, et que la fonction de divergence puisse se calculer facilement. La proposition suivante montre que ceci est le cas pour par exemple la fonction de risque $c := c_{Rloss}$ que nous avons considérée dans nos expériences.

**Proposition 12.** *Soient $\bar{f}$ et $\bar{f}'$ deux fonctions de rang, et $x$ un exemple non-étiqueté. Nous nous plaçons dans le cas où les jugements de pertinence sur les alternatives sont des jugements binaires. Nous avons alors avec la fonction de risque de l'équation 3.2 :*

$$d_{c_{Rloss}}(\bar{f}(x), \bar{f}'(x)) \leq \max_{p,q:p+q=A_x} \frac{1}{pq} \sum_{k=1}^{p} \delta(\bar{f}(x), \bar{f}'(x))_k$$

*Où $\delta(\bar{f}(x), \bar{f}'(x))$ est une liste de taille $A_x$ contenant toutes les valeurs décroissantes de $\bar{f}(x)(i) - \bar{f}'(x)(i)$ pour $1 \leq i \leq A_x$.*

*Preuve* : supposons que la vraie étiquette de $x$ soit $Y_x = (Y_1, ..., Y_{A_x})$. Nous notons par $rg(i) = \bar{f}(x)(i) - 1$, et pout chaque $i \in Y_x^+$, $rg_+(i) = \sum_{j \in Y_x^+} [[\bar{f}(x)(i) > \bar{f}(x)(j)]]$ (le nombre d'alternatives pertinentes ordonnées avant $i$).

Nous avons alors, avec la définition de $c_{Rloss}$ (3.2) :

$$c_{Rloss}(\bar{f}(x), \ell) = \frac{1}{|Y_x^+||Y_x^-|} \sum_{i \in Y_x^+} (rg(i) - rg_+(i))$$

Comme $rg_+$ et $rg'_+$ considère seulement des alternatives pertinents, nous avons $\sum_{i \in Y_x^+} rg_+(x) = \sum_{i \in Y_x^+} rg'_+(i)$, et ainsi

$$c_{Rloss}(\bar{f}(x), \ell) - c_{Rloss}(\bar{f}'(x), \ell) \leq \frac{1}{|Y_x^+||Y_x^-|} \sum_{i=1}^{|Y_x^+|} \delta(\bar{f}(x), \bar{f}'(x))_i$$

Le résultat s'obtient en prenant la valeur maximale du terme de droite de l'équation précédente sur toutes les valeurs possibles de ces nombres.

Pour un exemple $x$, la compléxité de $d_{c_{Rloss}}$ est $O(|\mathcal{A}_x| ln |\mathcal{A}_x|)$, car le calcul le plus coûteux ici est l'ordonnancement d'une liste de taille $|\mathcal{A}_x|$.

## 3.2 Borne Uniforme du risque pour l'apprentissage actif

La borne de test 7, comme celle étudiée en classifiction au chapitre 2 section 2.4, présente des restrictions et ne peut être utilisée dans l'état pour une stratégie d'apprentissage actif. Nous avons en effet besoin que la borne soit valide d'une manière uniforme pour toutes fonctions de rangs $\bar{f}$, et pour toutes les séquences possibles de requêtes. Pour rendre cette borne uniforme, l'outil que nous avons utilisé ici est le schéma de compression d'exemples proposé par (Floyd and Warmuth, 1995) dans le cadre de l'apprentissage supervisé.

Suivant ce schéma, étant donné une base d'apprentissage étiquetée $S$ de taille $n$, n'importe quel classifieur retourné par l'algorithme d'apprentissage est décrit par un ensemble de compression. Cet ensemble est un sous-ensemble de l'ensemble d'apprentissage $S$ et il peut être décrit par un vecteur d'indices $\vec{i} = (i_1, i_2, \ldots, i_k)$ avec $i_j \in \{1, \ldots, n\}$ $\forall j$ et $i_1 < i_2 < \ldots < i_k$. Ceci implique qu'il existe une fonction de *reconstruction*, associée à l'algorithme, qui prend en entrée un ensemble d'apprentissage et un vecteur d'indices et qui donne en sortie un classifieur. Le perceptron et les MVS en sont un exemple. Floyd and Warmuth (1995) ont proposé de borner l'erreur de test d'une fonction de reconstruction sur la base de ce vecteur d'indices $\vec{i}$.

Récemment, (Laviolette et al., 2005) ont étendu cette borne vers une borne qui reste valide simultanément pour tout classifieurs qui peut être reconstruits s'il existe une distribution a priori sur l'ensemble de tout les vecteurs d'indices possibles.

Ainsi comme nous considérons un algorithme d'apprentissage de fonction de rang déterministe $\mathcal{R}$, l'ensemble de toutes les fonctions de rangs qui peuvent être issu de $\mathcal{R}$ ne dépend que de l'ensemble de tout les exemples de $X_u$ pour qui une étiquette a été demandée durant l'exécution de l'algorithme. De plus, si nous faisons l'hypohtèse suivante :

**Hypothèse 13.** *Il existe une fonction déterministe $\phi : \mathcal{X} \to \mathcal{L}$ tel que pour tout $(x, \ell)$ échantilloné suivant $\mathcal{D}$, nous avons $\ell = \phi(x)$.*

L'ensemble de toutes les fonctions de rangs que $\mathcal{R}$ peut sortir ne dépendra que de l'ensemble $Z$ ainsi que de l'ensemble de tout les exemples activés. Ainsi comme pour le schéma de compression, nous avons une fonction de reconstruction associée à $\mathcal{R}$. Nous pouvons alors appliquer les mêmes outils que ceux développés pour le schéma de compression d'exemples et déduire des bornes de généralisation pour toutes fonctions de rangs qui peuvent être reconstruites. La section suivante formalise cette idée.

## 3.3 Stratégie de selection pour l'apprentissage actif de fonctions d'ordonnacement

En partant de l'ensemble d'exemples non-étiquetés $X_u$, la minimisation de l'erreur de généralisation de $\bar{f}$ peut être faite en considérant un sous-ensemble $X_u^{(k)}$ constitué de $k$ éléments de $X_u$ pour lequel la valeur de $d_c(\bar{f}(x'), \bar{f}_K^{cv}(x'))$ est maximale (Algorithm 5). Nous pouvons alors demander les étiquettes de $x' \in X_u^{(k)}$ et apprendre $\bar{f}$ sur $Z_l \cup Z_l k$, où $Z_l^k$ l'ensemble des exemples étiquetés jusqu'à lors, avec les exemples $x' \in X_u$ plus leur étiquette associée.

**Algorithm 5**: Une stratégie active pour l'ordonnancement

**Entrée** :

- ❒ Un ensemble d'exemple étiquetés $Z_l$ et un ensemble d'exemples non-étiquetés $X_u$,
- ❒ $k$ le nombre d'exemples à activer à chaque itération,
- ❒ $T$ le nombre maximum d'iterations.

**Initialisation**:

- ❒ $\forall j \in \{1, ..., K\}$ Apprendre $\bar{f}_j^{cv}$ sur $Z_j$, Poser $Z_l^k \leftarrow \emptyset$ et $t \leftarrow 1$.

**répéter**

- ❒ Apprendre $\bar{f}$ sur $Z_l \cup Z_l k$,
- ❒ Sélectionner un sous-ensemble $X_u^{(k)} \subset X_u | \forall x' \in X_u k$ la valeur $d_c(\bar{f}(x'), \bar{f}_K^{cv}(x'))$ est maximale,
- ❒ Demander les éiquettes de $x'$ for $x' \in X_u k$,
- ❒ Supprimer $X_u k$ de $X_u$ et réaffecter $Z_l k$, $Z_l^k \leftarrow Z_l^k \cup X_u k$, $t \leftarrow t + 1$

**jusqu'à** *Convergence de* $\sum_{x' \in X_u} d_c(\bar{f}(x'), \bar{f}_K^{cv}(x')) \vee t > T$ ;

**Sortie** : $\bar{f}$

Dans la suite nous supposons que $Z = \{(x_1, \ell_1) \dots (x_n, \ell_n)\}$ et $X_u = \{x'_{n+i}\}_{i \in \{1,m\}}$. De plus, nous supposons que chaque requête de l'algorithme d'apprentissage actif correspond à l'activation d'exactement $k$ exemples non-étiquetés (pour un paramètres $k$ fixé à l'avance). Le nombre total d'exemples activés sera alors un multiple de $k$. L'ensemble de compression de n'importe quelle fonction de rang associé à l'algorithme $\mathcal{R}$ est l'union de l'ensemble $Z$ et d'un ensemble de taille $k \cdot t$ de $X_u$ et l'ensemble des exemples étiquetés est toujours l'ensemble

de compression car l'algorithme considère toujours $Z$. Nous pouvons maintenant définir une distribution a priori de l'ensemble de toutes les sorties de $\mathcal{R}$ en posant une distribution a priori $P_{\mathbb{N}}$ sur $\mathbb{N}$ avec, pour tout $t$ qui a un poids dans $P_{\mathbb{N}}$, une probabilité a priori $P_t$ sur l'ensemble de tout les vecteurs d'indices possibles de la forme $\vec{i} = (1, 2, \ldots, n, i_1, i_2, \ldots, i_{kt})$ avec $i_j \in \{n+1, \ldots, n+m\}\ \forall j$ et $i_1 < i_2 < \ldots < i_{kt}$.

La pluapart du temps, la distribution a priori $P_{\mathbb{N}}$ aura tous ces poids dans l'ensemble $\{1, 2, \ldots T\}$ pour un paramètre $T$ fixé à l'avance. De plus, comme les exemples de $X_u$ sont supposés être échantillonnés i.i.d., nous avons choisi $P_t$ comme étant une distribution uniforme sous la contrainte que les $n$ premieres indices (correspondant aux données étiquetées) sont toujours connues, ce qui revient à poser $P_t(\vec{i}, t) = \binom{m}{kt}^{-1}$ pour tout $(\vec{i}, t)$. Nous désignons par $\mathcal{R}_{(\vec{i},t)}$ la fonction de rang correspondante. Sous ces hypothèses, nous avons le résultat suivant.

**Théorème 14.** *Soit $\mathcal{R}$ un algorithme d'apprentissage actif de fonctions de rang tel que ses ensembles d'exemples où il faut prédire leur étiquette sont tous de taille $k$. Soient $P_{\mathbb{N}}$ et $\{P_t\}_{t\in\mathbb{N}}$ les probabilités a priori définies précédemment et $\bar{f}_K^{cv}$ une fonction de rang stochastique (définie par exemple en cross validation sur un ensemble étiqueté). Nous avons $\forall t \in \mathbb{N}$ et $\forall \vec{i} = (1, 2, \ldots, n, i_1, i_2, \ldots, i_{kt})$ :*

$$\mathbb{P}\left( \mathop{\mathbb{E}}_{x'\sim\mathcal{D}_{\mathcal{X}}} d_c\left(\mathcal{R}_{(\vec{i},t)}(x'),\ \bar{f}_K^{cv}(x')\right) \leq \hat{\epsilon}_{Z\cup X_u kt} + \sqrt{\frac{\ln\binom{m}{kt} - \ln P_{\mathbb{N}}(t) + \ln(2/\delta)}{2(m-kt)}} \right) > 1 - \frac{\delta}{2}$$

*Où*

$\hat{\epsilon}_{Z\cup X_u kt} \stackrel{\text{def}}{=} \frac{1}{m-kt} \sum_{x'\in X_u\setminus X_u kt} d_c\left(\mathcal{R}_{(\vec{i},t)}(x'),\ \bar{f}_K^{cv}(x')\right)$.

*Preuve* : Ce résultat s'obtient d'une manière similaire à la preuve du lemme 10, pour tout $(\vec{i}, t)$, en utilisant l'inégalité de Hoeffding [avec $\delta := \frac{\delta . P_{\mathbb{N}}(t)\cdot P_t(\vec{i},t)}{2}$] et en appliquant la borne de l'union.

Le theorème 14, avec celui de 7 et le lemme 10 nous donnent une borne de généralisation (avec une confiance d'au moinds égale à $1-\delta$) pour n'importe quelle fonction de rang apprise suivant ce procédé d'apprentissage actif. Ce résultat montre de plus que n'importe quel algorithme $\mathcal{R}$ convergera vers une fonction de rang pas pire que la fonction de rang obtenue en cross-validation $\bar{f}_K^{cv}$ ceci même si $\mathcal{R}$ construit seulement une fonction de rang déterministe. De plus, il est clair par la définition de la divergence $d_c$ que, pour n'importe quelle fonction déja construite $\bar{f}$, l'étiquette correspondante à n'importe quel exemple non-étiqueté pour lequelle la fonction $d_c$ est maximale donnera lieu à une des trois situations suivantes : (1)–la valeur du risque de $\bar{f}_K^{cv}$ est bonne et celle de $\bar{f}$ ne l'est pas,(2)–la valeur du risque de $\bar{f}_K^{cv}$ est mauvaise est celle de $\bar{f}$ est bonne , ou (3)–les deux valeurs sont mauvaises. Lorsque l'on est confrontée

aux situations (1) ou (3), $\bar{f}$ n'est pas performante. D'un point de vu d'apprentissage actif, ces situations sont celles qui vont dans le sens d'amélioration de $\bar{f}$. De plus, si $\bar{f}_K^{cv}$ a une petite valeur de risque les situations (1) et (3) ont beaucoup de chance de se produire. Cette idée est centrale dans la conception de notre stratégie d'apprentissage actif.

## 3.4 Résultats expérimentaux

Dans nos expériences, nous avons comparé notre stratégie sélective avec une strategie aléatoire, où les exemples sont choisis aléatoirement dans l'ensemble non-étiqueté, et l'heuristique de la marge étendue proposé par (Brinker, 2004) et qu'on a adapté au cas d'ordres partiels. L'algorithme d'apprentissage supervisé de fonction de rang est l'algorithme **??** présenté au chapitre 2. Nous avons utilisé la mesure de divergence $d_{c_{Rloss}}$ introduite à la section 3.1 pour activer les exemples de l'ensemble $X_u$. L' objet de nos expériences est de voir si la stratégie d'apprentissage actif que nous avons développée est efficace pour la tâche du résumé de texte Amini (2000); Amini and Usunier (2007). Les performances des systèmes sont moyennées sur 10 bases apprentissage/non-étiquetée/test obtenues aléatoirement à partir de l'ensemble de départ. Nous rappelons que les requête qu'on active ici sont les documents pour lesquels la liste des phrases apparaissant dans leur résumé est demandée.

### 3.4.1 Résumé de textes

Les expériences que nous avons menées ici sont sur la base `WIPO` et sur la même collection que celle utilisée dans la section **??** du chapitre **??**. Nous rappelons qu'au total il y a $854$ documents dans cette collection et le partitionnement étiqueté/non-étiqueté/test sont respectivement $60/394/400$ et $30/394/400$ dans chaque expérience. Nous voulions voir ici l'évolution des performances en fonction du nombre de documents activés et la mesure de performance que nous avons préconisée pour cela est la précision moyenne (présentée à la section 1.5 du chapitre 1) en fixant le taux de compression à $10\%$.

La figure 3.1, montre les performances des algorithmes d'apprentissage actif suivant les strategies aléatoires, marge étendue et notre approche pour différents nombres de fonctions de rang aléatoires et différentes tailles de la base d'apprentissage de départ (30 et 60 documents) et les mêmes tailles des bases non-étiquetées et tests. Dans les deux expériences notre stratégie d'apprentissage actif montre un net avantage sur les deux autres stratégies actives. La performance assez faible de la stratégie à base de la marge étendue peut s'expliquer par le fait qu'une bonne fonction de score doit être capable d'ordonner les phrases pertinentes au-desussus des phrases

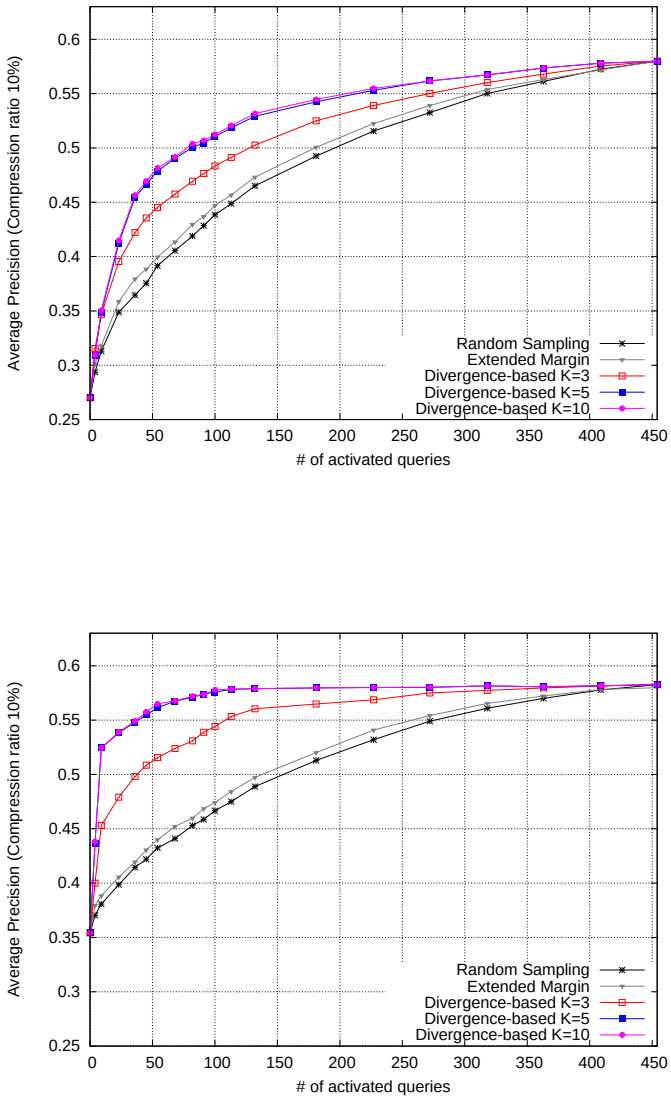



FIGURE 3.1 – La précision moyenne à $10\%$ de compression en fonction du nombre d'exemples activés pour les strategies aléatoires, marge étendue et notre approche. Les résultats sont moyennés sur $10$ ensembles étiqueté/non-étiqueté/test obtenus aléatoirement de l'ensemble de départ. Pour le même nombre de documents dans l'ensemble non-étiqueté $(394)$ et test $(400)$. Les performances sont obtenues pour $30$ (haut) et $60$ (bas) documents dans la base d'apprentissage étiquetée de départ.

non-pertinentes mais on ne doit pas attendre à ce que cette fonction de score soit confiante sur les rangs relatifs de deux phrases pertinentes ou non-pertinentes.

Nous remarquons que lorsque les fonctions de rang aléatoires (FRA) sont suffisament entraînées (figure 3.1 bas) en activant $50$ documents avec $5$ ou $10$ FRA, la performance de la fonction de rang finale est approximativement la même que si cette dernière avait été entraînée sur la totalité des exemples étiquetés et non-étiquetés plus leur étiquettes.

Le taux de convergence de la performance des fonctions de rang déterministe est néanmoins plus faible avec un petit nombre de FRA. Ceci est dû au fait que le partitionnement de la base d'apprentissage en différents ensembles cross-validés (sur lesquels chaque FRA est entraîné) est plus grand avec un grand nombre de FRA. Notre stratégie apparaît donc être assez efficace lorsque la taille de la base d'apprentissage est raisonable avec un nombre de FRA pas trop faible.

## 3.5 Conclusion

Nous avons proposé une stratégie d'apprentissage actif pour apprendre les fonctions de rang. L'analyse théorique et la définition de la notion de désaccord entre deux fonctions de rang ont conduit à ce résultat. Nous avons obtenu de bons résultats empiriques sur l'application de résumé automatique dans laquelle l'effort d'étiquetage est particulièrement coûteux. Cette stratégie d'apprentissage actif est à notre connaissance la première qui peut s'appliquer au cas où on cherche à prédire un ordre partiel sur les exemple. De plus les résultats empiriques ont été encourageants et ont mis la lumière sur certains aspects qu'une stratégie d'apprentissage actif de ces fonctions doit respecter.

La faiblesse majeure de notre éthude théorique est que nous considérons des fonctions de rang aléatoires construites sur des ensembles cross-validés. Dans ce cas l'erreur de généralisation de la fonction de rang apprise par la stratégie active est majorée par celle de la fonction de cross-validation. Notre analyse présente néanmoins plusieurs avantages : $(1)$ notre approche tend à minimiser une nouvelle borne sur l'erreur de généralisation d'une fonction de rang. $(2)$ la borne reste valide durant la phase d'apprentissage. $(3)$ Notre approche à base de la compression d'exemples donne un cadre général permettant d'apprendre d'autres algorithmes d'apprentissge actif. Finalement, les résultats empiriques obtenus suggèrent que l'approche active à base de désaccord entre fonctions de rangs est efficace.

# Bibliographie


Agarwal, S., Graepel, T., Herbrich, R., Har-Peled, S., and Roth, D. (2005). Generalization bounds for the area under the roc curve. *Journal of Machine Learning Research*, 6 :393–425.

Amini, M.-R. (2000). Interactive learning for text summarization. In *Proceedings of the PKDD/MLTIA Workshop on Machine Learning and Textual Information Access*, Lyon - France.

Amini, M.-R. and Gallinari, P. (2002). The use of unlabeled data to improve supervised learning for text summarization. In *Proceedings of the* $25^{th}$ *ACM SIGIR Conference*, pages 105–112.

Amini, M.-R. and Usunier, N. (2007). A contextual query expansion approach by term clustering for robust text summarization. In *Proceedings of the 7th Document Understanding Conference*, Rocherster - USA.

Bartlett, P. and Mendelson, S. (2003). Rademacher and gaussian complexities. *Journal of Machine Learning Research*, 3 :463–482.

Brinker, K. (2004). Active learning of label ranking functions. In *Proceedings of of* $21^{st}$ *International Conference on Machine learning*, pages 129–136.

Celeux, G. and Govaert, G. (1992). A classification em algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, 14(3) :315–332.

Chapelle, O. (2005). Active learning for parzen window classifier. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 49–56.

C.L. Blake and C.J. Merz (1998). *UCI Repository of machine learning databases*. http ://www.ics.uci.edu/∼mlearn/MLRepository.html, University of California, Irvine, Dept. of Information and Computer Sciences.

Cohen, D., Ghahramani, Z., and Jordan, M. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4 :129–145.

Cohen, W. W., Schapire, R. E., and Singer, Y. (1998). Learning to order things. In *Advances in neural information processing systems 10*.

Cortes, C. and Mohri, M. (2004). Auc optimization vs. error rate minimization. In *Advances in neural information processing systems 16*.

Crammer, K. and Signer, Y. (2003). A family of additive online algorithms for category ranking. *Journal of Machine Learning Research*, 3(6) :1025–1058.

Dekel, O., Manning, C., and Singer, Y. (2004). Log-linear models for label ranking. In *Advances in Neural Information Processing Systems 16*.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society*, 39(1) :1–38.

Edmundson, H. (1969). New methods in automatic extracting. *Journal of the ACM*, 16(2) :264–285.

Floyd, S. and Warmuth, M. (1995). Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning*, 21(3) :269–304.

Freund, Y., Iyer, R., Schapire, R., and Singer, Y. (2004). An efficient boosting algorihm for combining preferences. *Journal of Machine Learning Research*, 4 :933–969.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Society*, 58 :13–30.

Janson, S. (2004). Large déviations for sums of partly dependent random variables. *Random Structures and Algorithms*, 24(3) :234–245.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proceedings of the* $16^{th}$ *International Conference on Machine Learning*, pages 200–209.

Kääriäinen, M. (2005). Generalization error bounds using unlabeled data. In *Proceedings of the* $18^{th}$ *Annual Conference on Learning Theory*, pages 127–142.

Kupiec, J., Pedersen, J., and Chen, F. (1995). A trainable document summarizer. In *Proceedings of the* $18^{th}$ *ACM SIGIR Conference*, pages 68–73.

Laviolette, F., Marchand, M., and Shah, M. (2005). Margin-sparsity trade-off for the set covering machine. *Proceedings of the* $16^{th}$ *European Conference on Mahcine Learning (ECML)*, pages 206–217.

Marcu, D. (1999). The automatic construction of large-scale corpora for summarization research. In *Proceedings of the* $22^{nd}$ *ACM SIGIR Conference*, pages 137–144.

McDiarmid, C. (1989). On the method of bounded differences. *Surveys in combinatorics*, 141 :148–188.

McLachlan, G. J. (1992). *Discriminant Analysis and Statistical Pattern Recognition*. John Wiley & Sons, New York.

Mladenic, D. and Grobelnik, M. (1998). Feature selection for classification based on text hierarchy. Technical report, Working Notes of Learning from Text and the Web, Conference Automated Learning and Discovery, Carnegie Mellon University, Pittsburgh.

Nigam, K., McCallum, A., Thurn, S., and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39(2/3) :127–163.

SUMMAC (1998). *Text Summarization Evaluation Conference*. http ://www-nlpir.nist.gov/related_projects/tipster_summac/, SUMMAC-TIPSTER.

Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge Press University, New York, USA.

Tong, S. and Koller, D. (2002). Support vector machine active learning with applications to text classification. *Journal of Machine learning Research*, 2 :45–66.

Usunier, N. (2006). *Apprentissage de fonctions d'ordonnancement : une étude théorique de la réduction à la classification et deux applications à la Recherche d'Information.* PhD thesis, Université Pierre et Marie Curie, Paris 6.

Usunier, N., Amini, M., and Gallinari, P. (2006). Generalization error bounds for classifiers trained with interdependent data. In *Advances in Neural Information Processing Systems 18*, pages 1369–1376.

Vittaut, J.-N., Amini, M.-R., and Gallinari, P. (2002). Learning classification with both labeled and unlabeled data. In *Proceedings of the 13th European Conference on Machine Learning (ECML'02) - Helsinki, Finland*, pages 468–476.

Weston, J. and Watkins, C. (1999). Support vector machines for multi-class pattern recognition. In *Proceedings of the Seventh European Symposium On Artificial Neural Networks*, pages 219–224.