ÉVOLUTION DU CONCEPT DE FRONT ROC ET COMBINAISON DE CLASSIFIEUR

YANNICK OUFELLA

2 SEPTEMBRE 2008

ENCADREMENT :

SÉBASTIEN ADAM	*sebastien.adam@univ-rouen.fr*
CLÉMENT CHATELAIN	*clement.chatelain@insa-rouen.fr*
LAURENT HEUTTE	*laurent.heutte@univ-rouen.fr*
YVES LECOURTIER	*yves.lecourtier@univ-rouen.fr*

UNIVERSITÉ DE ROUEN
UFR DES SCIENCES ET TECHNIQUES
MASTER GÉNIE INFORMATIQUE
MASTER GÉNIE ÉLECTRIQUE ET INFORMATIQUE INDUSTRIELLE

Remerciements

Je tiens à remercier Sébastien Adam et Clément Chatelain pour m'avoir conseillé et orienté sur le choix de mon stage, pour m'avoir encadré tout au long de mon travail et pour la pertinence de leurs remarques. Je remercie également Laurent Heutte et Yves Lecourtier de m'avoir fait l'honneur de participer à mon travail.

Merci également à tous les membres du laboratoire LITIS pour leurs enseignements et leurs présence au quotidien. Un grand merci à notre secrétaire de choc Laurence qui a toujours été là au cours de ces trois dernières années pour nous aider et nous soutenir.

Merci à tous.

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

Introduction générale

On dispose aujourd'hui, dans le domaine de la reconnaissance de formes, d'un grand nombre de classifieurs et de méthodes permettant de les combiner. Dès 1974, Kanal [Kanal, 1974] avait souligné pour des problèmes de classification :*"No single model exists for all pattern recognition problems and no single technique is applicable to all problems. Rather what we have is a good bag of tools and a bag of problems."*. Malgré les nombreux travaux dans le domaine, cela n'a pas permis de mettre en évidence la supériorité incontestable d'une méthode de classification et de combinaison sur une autre. Plutôt que de chercher à optimiser un seul classifieur en choisissant des sous-ensembles pour un problème donné, les chercheurs ont trouvé plus intéressant de combiner des méthodes de reconnaissance.

Depuis les années 90, la combinaison de classifieurs a donc été une des directions de recherche les plus soutenues dans le domaine de la reconnaissance de formes. Les méthodes de combinaison ont ainsi été appliquées avec succès dans des domaines aussi divers et variés que la reconnaissance de l'écrit, le vérification de signatures, l'identification de visages ou encore l'analyse d'images médicales. L'amélioration des performances des systèmes de reconnaissance est finalement le principal enjeu des recherches menées ces dernières années sur les systèmes de combinaison.

Évaluer les performances d'un système de classification est un enjeu de grande importance car ses performances peuvent être utilisées pour l'apprentissage, l'op-

timisation ou la comparaison des systèmes. Afin de présenter la méthodologie relative à l'évaluation des performances d'un classifieurs, nous proposons dans le chapitre 1 un panorama des méthodes existantes.

Le chapitre 2 est consacré à l'étude d'un système complet d'optimisation pour la sélection des hyperparamètres d'un classifieur. L'utilisation et l'optimisation d'un ensemble de classifieurs à la place d'un classifieur unique ouvre de nombreuses perspectives mais pose également de nombreux problèmes. La contribution minime proposée ici fait suite au travail présenté dans [Chatelain et al., 2008]. Elle cadre le système proposé dans un environnement d'évaluation statistique permettant la généralisation de l'approche.

Afin d'améliorer le système existant et de résoudre les problèmes, nous allons réduire l'ensemble des classifieurs issus de l'optimisation à un unique éléments. La contribution majeure du travail concerne la sélection et la combinaison des classifieurs. Ainsi, nous présentons dans le chapitre 3 les principales stratégies de combinaison (approche séquentielle, parallèle et hybride), les techniques de combinaisons couramment utilisées puis les résultats que nous obtenons avec ces méthodes sur notre ensemble de classifieurs optimisé.

Enfin, nous concluons sur les points importants du document, notamment sur l'approche expérimentale proposée et les résultats obtenus. Nous évoquons les perspectives et les voies de recherche qui semblent prometteuses pour améliorer l'approche proposée.

CHAPITRE 1

Évaluation des performances d'un classifieur

Évaluer les performances d'un système de classification est un enjeu de grande importance car ces performances peuvent être utilisées pour l'apprentissage en tant que tel ou pour optimiser les valeurs des hyperparamètres du classifieur. Pendant longtemps, le critère retenu pour évaluer ces performances a été le taux de bonne classification, c'est-à-dire le nombre d'éléments d'une base de test correctement classés. Le problème d'un tel critère est qu'il n'est pas adapté à des environnements mal définis. Dans de nombreuses situations, toutes les erreurs n'ont pas les mêmes conséquences. Certaines erreurs ont un coût plus important que d'autres, par exemple, pour les diagnostiques médicaux. Un mauvais diagnostic ou traitement peut, en effet, avoir différents coûts ou dangers selon le type d'erreur commise.
Dans cette partie, nous dressons un panorama des critères d'évaluation des systèmes de classification à deux classes puis de manière plus générale des systèmes multi-classes. Nous nous intéresserons dans un premier temps à l'évaluation des performances en deux classes à l'aide de mesures scalaires dans la section 1.1, puis multi-critères dans la section 1.2. Nous verrons notamment que la courbe ROC n'est pas applicable directement à des problèmes multi-classes et nous présenterons, dans la section 1.3, les adaptations à des problèmes multi-classes. Nous présentons ensuite plusieurs résultats dans la section 1.4.

1.1 Évaluation scalaire

Nous allons évoquer, dans cette partie, les méthodes et les métriques scalaires qui permettent d'évaluer et d'analyser les performances d'un système de classification. Parmi les méthodes les plus populaires, nous retrouvons le taux de bonne classification sans coûts et le taux de bonne classification avec coût.

1.1.1 Taux de bonne classification sans coûts

La première mesure à laquelle nous allons nous intéresser est le taux de bonne classification simplifié (tbc_s) ou sans coût. Il s'agit de l'indicateur le plus naturel et le plus évident permettant d'évaluer les performances d'un système de classification. Cette valeur, simple à calculer, correspond au nombre d'éléments correctement identifiés par le système. La définition du taux de bonne classification sans la prise en compte du rejet est :

$$tbc_s = \frac{\text{Nombre d'éléments correctement identifiés}}{\text{Nombre d'éléments total}} \tag{1.1.1}$$

On obtient le taux d'erreur par :

$$te_s = 1 - tbc_s \tag{1.1.2}$$

Lorsque le rejet d'une forme est possible, un troisième taux, dit "taux de rejet" (tr) est intégré. Il mesure le nombre d'éléments sur lesquels le système n'a pas pris de décision. Nous obtenons ainsi :

$$te_s = 1 - tbc_s - tr \tag{1.1.3}$$

Le problème rencontré avec le taux de bonne classification sans coût, est qu'il s'agit d'une mesure "faible", car elle ne tient pas compte de la distribution des classes et des coûts de classification. Prenons, par exemple, le cas d'un diagnostic médical, une base constituée de 5 personnes malades et de 995 personnes saines. Si nous décidons que tout le monde est sain, nous obtenons alors un taux de bonne reconnaissance $tbc_s = 99,5\%$. Au vu de ce résultat, le classifieur est donc très performant. Le problème est que de manière générale dans la prise de décision médicales, la répartition des classes n'est pas équilibrée et

généralement la classe la moins représentée est celle que l'on cherche à identifier. Ce critère ne suffit donc pas à l'évaluation pertinente des performances [Provost and Fawcett, 1997] [Provost and Fawcett, 1998]. Pour remédier à cela, nous allons incorporer un facteur qui pondèrera le score en prenant en compte la distribution des classes et les coûts associés aux décisions. Nous introduisons par ce biais le taux de bonne classification, tbc.

1.1.2 Taux de bonne classification avec coût

La seconde mesure que nous allons aborder est le taux de bonne classification avec coût. Il s'agit de l'évolution de la mesure précédente avec cette fois la prise en compte de la répartition des classes mais également des coûts de bonne et mauvaise classification. Plusieurs définitions existent en fonction des éléments qui sont pris en compte mais toutes utilisent la matrice de confusion, tableau 1.1, et la matrice des coûts classification, tableau 1.2.
Pour un système à C classes, nous définissons les indices i, $j \in \{1,\ldots,C\}$; ω_i représentant l'étiquette de la classe et N_{ω_i} le nombre d'éléments de la classe i présents dans la base. Afin de résumer au mieux les résultats de la classification, nous utilisons une matrice dite de confusion qui met en relation les décisions prises par le classifieur et les étiquettes des exemples. Associé à cette matrice, nous définissons la matrice des coûts qui fait correspondre à chaque élément de la matrice de confusion un coût. Nous introduisons d'une part, la notation $\varepsilon_{i,j}$ qui correspond au nombre d'éléments étiquetés de la classe i et identifiés comme des éléments de la classe j. Le terme de score est également utilisé pour définir cette valeur. D'autre part, nous définissons $Cost_{i,j}$ comme le coût associé à $\varepsilon_{i,j}$. En général la matrice de confusion est normalisée de manière à obtenir directement des taux de bonne classification lorsque $i = j$ et des taux d'erreur lorsque $i \neq j$.
Nous définissons à partir des matrices, 1.1 et 1.2, le taux de bonne classification et le taux d'erreur avec la prise en compte de la distribution des classes. On parle en général dans ce cas du taux de bonne classification par classe :

$$tbc = \sum_{i=1}^{C} \left[\frac{\varepsilon_{i,i}}{N_{\omega_i}} \right] * \sum_{i=1}^{C} N_{\omega_i} * \frac{1}{C} \tag{1.1.4}$$

		Décision				
		ω_1	ω_2	$\dots$	ω_C	
Étiquette	ω_1	$\varepsilon_{1,1}$	$\varepsilon_{1,2}$	$\dots$	$\varepsilon_{1,C}$	N_{ω_1}
	ω_2	$\varepsilon_{2,1}$	$\varepsilon_{2,2}$	$\dots$	$\varepsilon_{2,C}$	N_{ω_2}
	$\vdots$	$\vdots$		$\ddots$	$\vdots$	$\vdots$
	ω_C	$\varepsilon_{C,1}$	$\varepsilon_{C,2}$	$\dots$	$\varepsilon_{C,C}$	N_{ω_C}

TAB. 1.1: Matrice de confusion normalisée

		Décision			
		ω_1	ω_2	$\dots$	ω_C
Étiquette	ω_1	$Cost_{1,1}$	$Cost_{1,2}$	$\dots$	$Cost_{1,C}$
	ω_2	$Cost_{2,1}$	$Cost_{2,2}$	$\dots$	$Cost_{2,C}$
	$\vdots$	$\vdots$		$\ddots$	$\vdots$
	ω_C	$Cost_{C,1}$	$Cost_{C,2}$	$\dots$	$Cost_{C,C}$

TAB. 1.2: Matrice des coûts de classification

$$\begin{aligned} te &= \sum_{i=1}^{C} \left[\frac{(1-\varepsilon_{i,i})}{N_{\omega_i}} \right] * \sum_{i=1}^{C} N_{\omega_i} * \frac{1}{C} \\ &= 1 - tbc - tr \end{aligned} \tag{1.1.5}$$

Reprenons l'exemple présenté précédemment (1000 individus, 5 malades et 995 non malades) en intégrant, cette fois, la distribution des classes. Si nous décidons que tout le monde est sain, nous obtenons la matrice de confusion 1.3, à partir de laquelle nous calculons le taux de bonne reconnaissance :

		Décision	
		Non Malade	Malade
Étiquette	Non Malade	0,995	0
	Malade	0,005	0

TAB. 1.3: Matrice de confusion normalisée associée à l'exemple

$$tbc = \frac{O,995}{995} * (995+5) * \frac{1}{2} = 0,5 \Rightarrow tbc = 50\%$$

La prise en compte de la distribution des classes modifie radicalement notre perception des performances, passant de $99,5\%$ dans la section 1.1.1 à 50% maintenant. Pour aller plus loin, nous utilisons l'équation 1.1.4 en incorporant les coûts

de mauvaise classification définis dans la matrice 1.2 :

$$tbc = \sum_{i=1}^{C} \left[\frac{\sum_{j=1, j\neq i}^{C} \varepsilon_{i,j} Cost_{i,j}}{N_{\omega_i}} \right] * \sum_{i=1}^{C} N_{\omega_i} * \frac{1}{C} \tag{1.1.6}$$

Finalement, afin d'évaluer au mieux les classifieurs vis-à-vis des caractéristiques de la base et des contraintes du problème, nous incorporons les coûts de bonne classification :

$$tbc = \sum_{i=1}^{C} \left[\frac{(\varepsilon_{i,i} Cost_{i,i}) + \left(\sum_{j=1, j\neq i}^{C} \varepsilon_{i,j} Cost_{i,j} \right)}{N_{\omega_i}} \right] * \sum_{i=1}^{C} N_{\omega_i} * \frac{1}{C} \tag{1.1.7}$$

La pondération des taux de bonne et mauvaise classification permet de mieux évaluer les performances d'un système en intégrant les informations relatives aux bases de test. Le problème qui se pose alors est la connaissance des coûts de classification. En effet, dans la majeure partie des cas, les coûts sont inconnus ou difficiles à déterminer. Il est alors impossible d'évaluer correctement un classifieur et la nécessité d'une nouvelle mesure plus élaborée s'impose. En général, les classifieurs sont utilisés pour faire des prévisions afin d'aider à la prise de décision. Puisque des prédictions peuvent s'avérer être fausses, il est important de savoir quel est l'effet lorsqu'une erreur est commise. C'est pourquoi, les mesures que nous avons étudié précédemment ne se révèlent pas assez pertinentes pour évaluer la qualité d'un classifieur. L'idée proposée dans [Hanley and McNeil, 1982], [Zweig and Campbell, 1993], [Bradley, 1997], [Hand, 1997] et [Hand, 2001] est d'utiliser des courbes plutôt que des valeurs scalaires pour évaluer les performances des classifieurs. La littérature traitant du sujet met en évidence de deux types de courbes à savoir : la courbe Précision–Rappel et la courbe ROC toutes deux basées sur la matrice de confusion.

1.2 Évaluation multi-critères

L'existence réelle de grands biais, dans la distribution et la répartition des classes a été observée dans différents domaines par [Clearwater and Stern, 1991], [Fawcett and Provost, 1996], [Kubat et al., 1998] et [Saitta and Neri, 1998]. Par

exemple, dans la prise de décision médicale, les épidémies peuvent faire augmenter l'incidence d'une maladie avec le temps. Dans la détection de fraudes, la proportion des fraudes varie de manière significative de mois en mois et de zone en zone, [Fawcett and Povost, 1997]. Dans chacun des exemples à deux classes, la prédominance d'une classe peut changer radicalement sans pour autant altérer fondamentalement les caractéristiques de la classe. Si les proportions d'éléments positifs ou/et négatifs changent dans une base de test, nous voulons que le système d'évaluation des performances ne soit pas perturbé. Pour cela, nous allons comparer deux méthodes applicables aux classifieurs à deux classes, la courbe Précision-Rappel et la courbe ROC.

Les mesures que nous allons évoquer utilisent la matrice de confusion, tableau 1.4, qui permet la différenciation des erreurs selon chaque classe en vue d'évaluer un classifieur. Définissons maintenant plusieurs mesures de manière for-

	Décision Positifs	Décision Negatifs	
Étiquette Positifs	**Vrai Positifs, TP**	**Faux Négatifs, FN**	Pos [a]
Étiquette Négatifs	**Faux positifs, FP**	**Vrai Négatifs, TN**	Neg [b]
	PPos [c]	PNeg [d]	N

TAB. 1.4: Matrice de confusion

[a] Nombre d'éléments étiquetés positifs dans la base.
[b] Nombre d'éléments étiquetés négatifs dans la base.
[c] Nombre d'éléments classés positifs.
[d] Nombre d'éléments classés négatifs.

TAB. 1.5:

melle :

– Le taux de vrais positifs ("True positive rate"),

$$tpr = \frac{TP}{Pos} = \frac{TP}{TP+FN} \tag{1.2.8}$$

– Le taux de vrais négatifs ("True negative rate"),

$$tnr = \frac{TN}{Neg} = \frac{TN}{TN+FN} \tag{1.2.9}$$

– Le taux de faux positifs ("False positive rate"),

$$fpr = \frac{FP}{Neg} = \frac{FP}{FP+TN} \tag{1.2.10}$$

– Le taux de faux négatifs ("False negative rate"),

$$fnr = \frac{FN}{Pos} = \frac{FN}{FN+TN} \tag{1.2.11}$$

– Le taux de bonne classification ou l'exactitude (accuracy)

$$acc = tbc = Pos * tpr + Neg * (1 - fpr) \tag{1.2.12}$$

– La précision

$$prec = \frac{TP}{PPos} = \frac{TP}{TP+FP} \tag{1.2.13}$$

– Le rappel (recall)

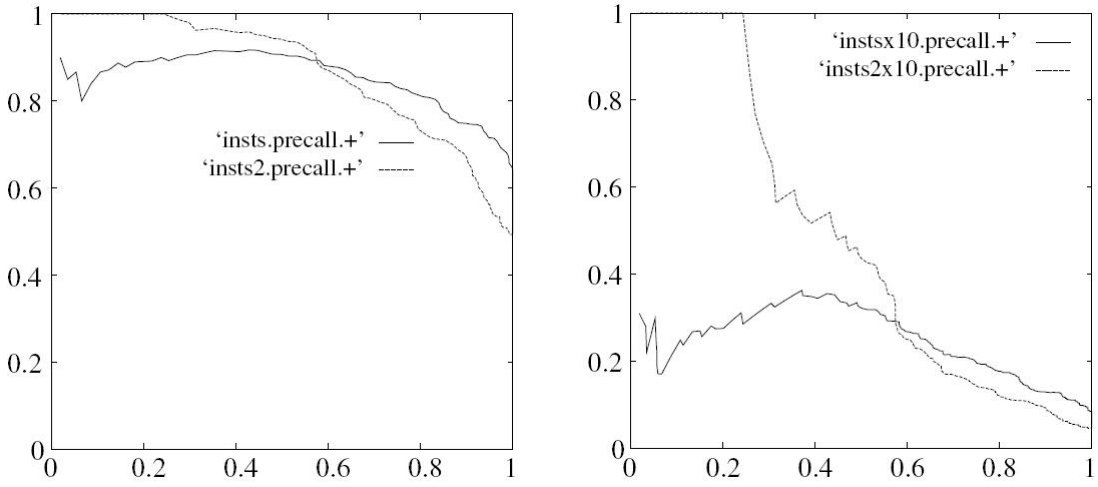
$$rec = tpr = \frac{TP}{Pos} = \frac{TP}{TP+FN} \tag{1.2.14}$$

Maintenant que nous avons caractérisé notre problème (estimation des taux de bonne et mauvaise classification, évaluation du type d'erreur, ...) via la matrice de confusion, nous allons représenter les performances des systèmes de classification à l'aide de la courbe Précision–Rappel puis de la courbe ROC.

1.2.1 Courbe Précision–Rappel

Nous allons maintenant traiter de la mesure "précision" et de la mesure "rappel". Ces mesures très utilisées en recherche documentaire (information retrieval) permettent d'évaluer les performances (la pertinence) du retour d'information vis-à-vis d'une requête, [Lewis, 1990], [Lewis, 1991]. La Précision et le Rappel sont généralement utilisés pour traiter des bases de documents statiques ; cependant, ils peuvent être utilisés dans des environnements dynamiques tels que la fouille de pages web, lorsque le nombre de documents non pertinents correspondant à une requête augmente à la vitesse de création de pages web sur

Internet.
Considérons la figure 1.1 qui présente deux classifieurs évalués par la courbe Précision-Rappel. Dans la figure 1.1a, la base de test est équilibrée dans la distribution des classes (1 :1). Pour la courbe 1.1b, les mêmes classifieurs sont utilisés, mais cette fois, le nombre d'éléments négatifs est dix fois plus important (1 :10). Nous constatons que les courbes Précision-Rappel, figures 1.1a et 1.1b, diffèrent sensiblement, ce qui indique une sensibilité de la représentation par rapport à la distribution. Il s'agit donc d'un problème important mettant en évidence la faiblesse de la courbe Précision–Rappel. De ce fait, nous devons faire appel à une autre représentation plus robuste vis-à-vis de la structure des bases de test. La courbe ROC, que nous allons maintenant présenter intervient dans ce sens.



(a) Courbes précision-Rappel, 1 :1

(b) Courbes précision-Rappel, 1 :10

FIG. 1.1: Courbes Précision-Rappel vis-à-vis de la distribution des classes, [Fawcett, 2006]

1.2.2 Courbe ROC

Un secteur très actif dans la reconnaissance de formes est la considération de la structure des classes et l'évaluation des performances dans des environnements mal définis ; c'est-à-dire avec des probabilités à priori non définies ou variantes [Provost and Fawcett, 2001], ou lorsque les coûts sont mal définis [Adams and Hand, 1999]. La méthode d'analyse performante qui a été développée pour ce domaine est le "Receiver Operator Characteristic" (ROC) [Metz, 1978],

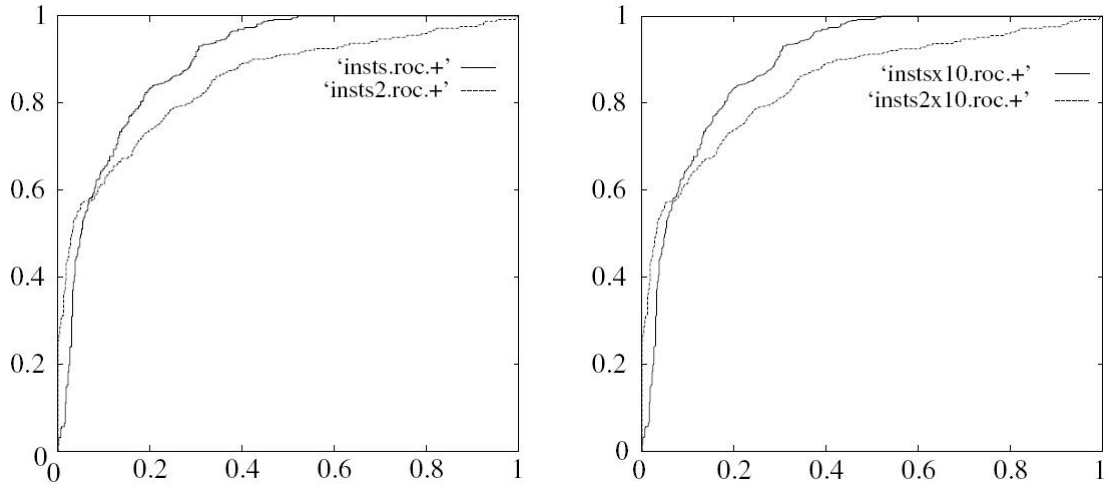
permettant à un classifieur d'être évalué vis-à-vis d'un ensemble de conditions possibles. L'analyse ROC, [Provost and Fawcett, 1997], [Swets et al., 2000], [Flach et al., 2003], s'est révélée très efficace pour donner une évaluation des classifieurs lorsque les coûts associés à la matrice de confusion ne sont pas connus au moment de la construction du classifieurs. La courbe ROC est une méthode de représentation graphique des performances d'un classifieur à deux classes. Depuis plusieurs années, son utilisation est devenue incontournable dans les méthodes d'évaluations, [Provost and Fawcett, 1997] et [Provost and Fawcett, 1998]. A l'origine, il s'agit d'une représentation issue du traitement du signal ayant pour objectif de déterminer un seuil de séparation entre le signal et le bruit, [Egan, 1975], [Swets et al., 2000]. L'analyse ROC a ensuite été étendue au domaine médical, [Swets, 1988], [Zou, 2002], puis au domaine d'apprentissage des systèmes (machine learning) avec [Spackman, 1989]. La courbe ROC permet de représenter les objectifs d'un système afin d'évaluer celui-ci. Encore faut-il savoir construire la courbe et quelle représentation utiliser. Les algorithmes 1 (p.90) et 2 (p.91) proposés respectivement par [Provost and Fawcett, 2004] et [Fawcett, 2006] donnent le principe de génération de la courbe ROC[1]. Dans la littérature, plusieurs approches basées sur la matrice de confusion, tableau 1.4, proposent différentes solutions quant au choix des axes de la courbe ROC.

- Soit le taux de vrais positifs (*true positives rate : tpr*) en ordonnée et le taux de faux positifs (*false positives rate : fpr*) en abscisse, [Hand, 2001], [Flach, 2004] [Fawcett, 2006],
- soit le taux de vrais négatifs *true negatives rate, tnr* en ordonnée et le *true positives rate : tpr* en abscisse, [Landgrebe and Duin, 2006], [Landgrebe and Duin, 200 [Landgrebe and Duin, 2008],
- ou alors le *false positives rate : fpr* en ordonnée et le *false negatives rate : fnr* en abscisse, [Ferri et al., 2003].

En général, les courbes ROC sont basées sur le taux de vrai positifs (*tpr*) et le taux de faux positifs (*fpr*). Il s'agit là de rapports qui ne dépendent donc pas de la distribution des classes. Cette méthode robuste permet de s'affranchir de la connaissance des coûts de classification et de la distribution des classes, [Flach, 2004], [Provost and Fawcett, 2004], [Flach, 2006] et [Fawcett, 2006].

[1] L'agorithme de [Provost and Fawcett, 2004] se base sur un classifieur de type rang alors que celui de [Fawcett, 2006] se base sur un classifieur de type mesure

Considérons la comparaison des figures 1.1 et 1.2 qui présentent deux classifieurs évalués par la courbe ROC et par la courbe Précision–Rappel. Dans les figures 1.1a et 1.2a, la base de test est équilibrée dans la distribution des classes(1 :1). Pour les courbes 1.1b et 1.2b, les mêmes classifieurs sont utilisés, mais cette fois, le nombre d'éléments négatifs est dix fois plus important (1 :10). Nous constatons que les courbes ROC, figures 1.2a et 1.2b, sont identiques alors que les courbes Précision–Rappel, figures 1.1a et 1.1b, diffèrent sensiblement. Si les proportions d'éléments positifs ou/et négatifs changent dans la base de test, la courbe ROC, elle, ne changera pas, [Fawcett, 2006]. Cette dernière est insensible aux modifications dans la distribution des classes.



(a) Courbes ROC, 1 :1 (b) Courbes ROC, 1 :10

FIG. 1.2: Courbes ROC vis-à-vis de la distribution des classes, [Fawcett, 2006]

Positionnons-nous maintenant, dans un contexte de comparaison des performances d'un classifieur, nous devons comparer deux courbes. La comparaison de deux courbes n'est pas une chose évidente. En effet, en fonction des zones de l'espace, les performances des classifieurs peuvent varier ; on peut avoir sur une zone une courbe qui est meilleure qu'une autre et sur une autre zone la seconde courbe qui est meilleure que la première. C'est pour cela que dans la littérature lorsque l'on souhaite comparer différents systèmes de classifications, il est pratique de réduire, que ce soit la courbe Précision–Rappel ou la courbe ROC, à des

valeurs scalaires. Dans le cas de la courbe Précision–Rappel, la valeur scalaire qui est classiquement extraite est la F-mesure et dans le cas de la courbe ROC la valeur scalaire communément extraite est l'aire sous la courbe ROC, AUC ("Area Under the ROC Curve"), [Hanley and McNeil, 1982] et [Bradley, 1997].

1.2.3 F–mesure

La F-mesure de [Van Rijsbergen, 1979] est un indicateur de synthèse communément utilisé pour évaluer les algorithmes de classification de données textuelles, à partir de la précision et du rappel. Elle est utilisée indifféremment pour les classifications et les catégorisations, [Nakache and Métais, 2005].

$$Mesure\ F = \frac{((1+\beta^2) * Precision * Rappel)}{((\beta^2 * Precision) + Rappel)}, avec\ \beta^2 = 1 \qquad (1.2.15)$$

La F-mesure correspond à une moyenne harmonique de la précision et du rappel. Le paramètre β permet de pondérer la précision ou le rappel et vaut généralement 1, [Nakache and Métais, 2005]. La mesure devient :

$$eq : 1.2.15 \Rightarrow \frac{(2 * Precision * Rappel)}{(Precision + Rappel} \qquad (1.2.16)$$

L'avantage de ce choix est que lorsque la précision est égale au rappel, on obtient : Précision = Rappel = F1-mesure. Ceci facilite la lecture et on recherche à maximiser la F-mesure en maximisant simultanément la précision et le rappel. Le problème que pose cette méthode est qu'elle ne permet pas la différenciation des erreurs et reste sensible à la distribution des classes car basée sur la précision et le rappel. Nous allons maintenant nous intéresser à l'aire sous la courbe ROC afin de comparer les deux mesures. Il est donc préférable d'utiliser la seconde mesure qui est à notre disposition à savoir : l'aire sous la courbe ROC.

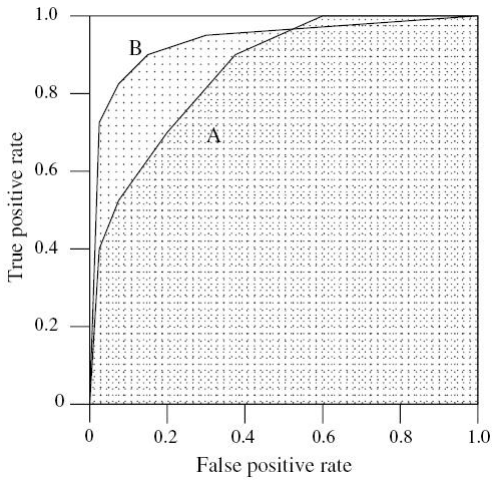
1.2.4 Aire sous la courbe ROC – AUC

Dans le cas le plus simple, un classifieur à deux classes forme une zone de 4 segments (un polygone au sens strict) à partir de la courbe ROC figure 1.3, avec le point donné par le classifieur, deux points triviaux (le classifieur qui prédit toujours la classe positif et le classifieur qui prédit toujours la classe négatif) et

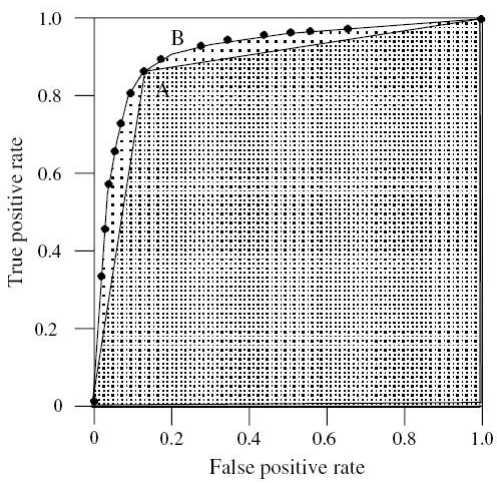
l'origine du repère. L'aire de cette zone est appelée "aire sous la courbe ROC ou AUC" ("Area Under the ROC Curve") et est devenue une meilleure alternative que l'exactitude (accuracy) ou l'erreur pour évaluer des classifieurs.
L'AUC d'un classifieur est équivalente à la probabilité qu'un classifieur donne un meilleur rang à un élément positif par rapport à un élément négatif, tous deux choisis aléatoirement dans la base, ce qui est équivalent au test de Wilcoxon, [Hanley and McNeil, 1982]. L'AUC est également très proche du coefficient de Gini, [Breiman et al., 1984], qui correspond à l'aire entre la courbe ROC et la diagonale de l'espace. Dans [Hand and Tills, 2001], la relation entre AUC et coefficient de Gini a été précisée pour donner

$$GINI + 1 = 2 * AUC \tag{1.2.17}$$

Soit l'exemple de la figure 1.3 présentant l'aire sous deux courbes ROC. Dans la figure 1.3a, le classifieur B dispose en moyenne, de l'aire la plus grande donc des meilleures performances. La figure 1.3b montre l'AUC pour un classifieur binaire, A, et un classifieur de mesure, B. Le classifieur A représente les performances de B lorsque B est utilisé avec un seuil fixé. Bien que les performances des deux soient égales à un point donné (seuil de A), A est tout de même moins performant pour les autres points.
La simplicité d'utilisation et les propriétés de l'AUC font de cette mesure un



(a) AUC pour deux courbes



(b) AUC pour un classifieur discret(A) et un classifieur probabiliste(B)

FIG. 1.3: Courbes ROC et AUC, [Fawcett, 2006]

élément central utilisé par la communauté scientifique comme critère de comparaison des performances des systèmes. Une première méthode de calcul proposée par [Hand, 2001], utilise le calcul intégrale sur tout l'espace définissant :

- Soit $\hat{P}(x)$ l'estimation de la probabilité qu'un objet ayant le vecteur de caractéristique x appartienne à la classe 0,
- soit $f(\hat{p}) = f(\hat{p}|0)$ la fonction de probabilité qu'un élément détecté de classe 0 appartienne à la classe 0 et $g(\hat{p}) = f(\hat{p}|1)$ la fonction de probabilité qu'un élément détecté de la classe 0 appartienne à la classe 1,
- soit $F(\hat{p}) = F(\hat{p}|0)$ et $G(\hat{p}) = G(\hat{p}|1)$ les fonctions de distribution cumulatives correspondantes.

Nous pouvons alors définir le calcul théorique de l'aire sous le courbe ROC.

$$AUC = \int G(u)dF(u) = \int G(u)f(u)du \tag{1.2.18}$$

Dans [Landgrebe and Duin, 2006] et [Landgrebe and Duin, 2007b], nous retrouvons la formulation de l'AUC par une intégrale à partir d'une représentation différente de l'espace ROC utilisant le *tpr* et *tnr*.

$$AUC = \int tnr * d(tpr) \tag{1.2.19}$$

Le calcul formel d'une intégrale étant difficile à réaliser, une approximation des calculs a été proposée dans [Hand, 2001] et [Flach, 2004].

$$\hat{AUC} = \frac{S_+ - \frac{Pos(Pos+1)}{2}}{Pos * Neg} \tag{1.2.20}$$

avec S_+ la somme des rangs des éléments postifs.

L'algorithme de calcul de l'aire sous la courbe, donnée en annexe page 92, est très proche de l'algorithme 2 pour la création de la courbe ROC. En effet, plutôt que de collecter les points ROC, l'algorithme cumule successivement les aires calculées pour de petits trapézoïdes. Les trapèzes sont utilisés plutôt que des rectangles, de manière à mieux approximer l'aire, comme nous pouvons le voir sur la figure 1.4, [Fawcett, 2006].

L'étude de la courbe ROC et de l'AUC ont été utilisées de manière intensive dans le domaine médical pour l'aide à la prise de décision [Hanley and McNeil, 1982],

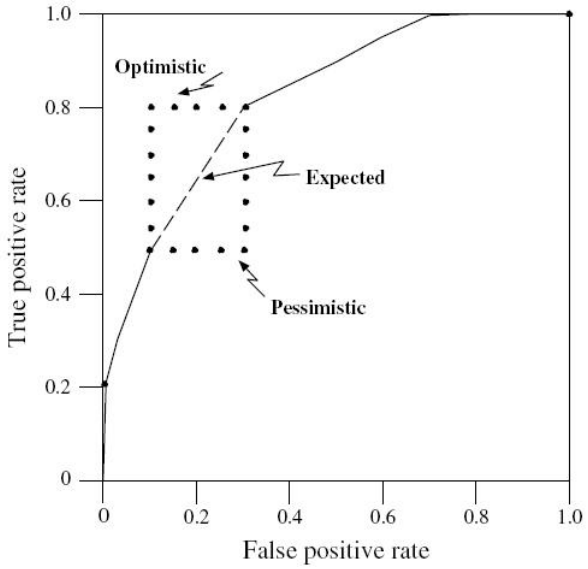


FIG. 1.4: Sélection de la méthode d'approximation pour le calcul de l'AUC, [Fawcett, 2006]

[Zweig and Campbell, 1993], dans le domaine de la découverte des connaissances, le data mining, la reconnaissance de formes [Adams and Hand, 1999] et la science en général [Swets et al., 2000]. Cependant, l'usage de la courbe ROC et de l'AUC ont seulement été démontré pour des problèmes à deux classes. Bien que l'analyse ROC puisse être étendue théoriquement à des problèmes multi-classes
[Srinivasan, 1999], les premières expérimentations excluent son utilisation pratique (complexité de calcul et compréhensibilité de la représentation). Néanmoins, malgré les difficultés, il est possible de produire une analyse ROC pour plus de deux classes. Les propriétés de la courbe ROC font de cette représentation un outil quasi indispensable pour évaluer, comparer et optimiser des systèmes de classification. L'idée va donc être de formuler une généralisation de la courbe ROC afin de déterminer les performances de classifieurs à plus de deux classes.

1.3 Généralisation de l'analyse ROC à des problèmes multi-classes

Jusqu'à présent, les discussions ont traité des problèmes à deux classes et nous avons constaté qu'une grande partie de la littérature sur l'analyse ROC se limite à ce cas. La courbe ROC est généralement utilisée dans la prise de décision médicale où les problèmes de diagnostic à deux classes sont courants (présence ou absence d'un facteur). Dans ces conditions, les deux axes de le courbe ROC représentent la différence entre l'erreur et la réussite d'un classifieur. L'évaluation des performances peut-être représentée en deux dimensions, ce qui est facile à visualiser. Avec l'extension aux systèmes multi-classes, la situation devient plus complexe, l'obstacle principal étant le nombre élevé de dimensions. En effet, pour un problème à C classes la matrice de confusion obtenue dispose de C^2 positions, contenant les C bonnes classifications (les éléments sur la diagonale principale) et $C(C-1)$ erreurs possibles (les éléments hors de la diagonale). Au lieu de gérer la différence entre les vrais et les faux positifs, nous devons gérer les C valeurs de bonne classification et les $C^2 - C$ erreurs possibles. Dans le cas spécifique de trois classes, nous obtenons une représentation à $3^2 - 3 = 6$ dimensions. L'adaptation de la courbe ROC à des systèmes multi-classes implique de définir une méthode globale pouvant supporter la représentation sous forme d'hyper-plans. La question de la représentation et de l'espace utilisable est encore aujourd'hui très ouverte et chacun tente de proposer une solution.

1.3.1 ROC multi-classes

Dans la littérature, nous retrouvons de nombreuses approches pour la généralisation de la courbe ROC en multi-classes. Commençons par distinguer deux grandes approches, une classe face à une autre et une classe face à toutes les autres, qui permettent de déterminer le nombre de dimensions de l'hyper-espace dans lequel sera dessiné la représentation ROC multi-classes, [Flach, 2004]. Ces approches vont déterminer la complexité et la faisabilité des calculs mais également les limites pour des systèmes à C classes.

$$\varepsilon_{i,j} = p(\omega_i) \int p(X|\omega_i) I_{ij}(X) dx \tag{1.3.21}$$

		Décision			
		ω_1	ω_2	...	ω_C
Étiquette	ω_1	$\varepsilon_{1,1}$	$\varepsilon_{1,2}$	...	$\varepsilon_{1,C}$
	ω_2	$\varepsilon_{2,1}$	$\varepsilon_{2,2}$	...	$\varepsilon_{2,C}$
	$\vdots$	$\vdots$		$\ddots$	$\vdots$
	ω_C	$\varepsilon_{C,1}$	$\varepsilon_{C,2}$	...	$\varepsilon_{C,C}$

TAB. 1.6: Matrice de confusion normalisée

Soit X, les observations devant être classées parmi les C classes, ω_1, ω_2, ..., ω_C. Chaque classe ω_i dispose d'une distribution conditionnelle $p(X|\omega_i)$ et d'une probabilité à priori $P(\omega_i)$.

Un contre un

La première approche, [Hand and Tills, 2001] et [Ferri et al., 2003], propose de manipuler une classe face à une autre et considère un coût de mauvaise classification différent pour chaque couple de classe. On cherche à affiner et à pondérer précisément le type d'erreur. Dans ces conditions, l'espace d'évaluation des performances est à $C(C-1)$ dimensions, ce qui revient à utiliser les éléments hors de la diagonale principale, les $\varepsilon_{i,j}$ avec $i \neq j$, de la matrice de confusion. Cette extension théorique pose néanmoins certains problèmes tels que la complexité des calculs et la représentation des performances. Par exemple pour trois classes, nous obtenons un espace à six dimensions difficilement représentable. Afin de réduire la complexité des calculs, une autre approche a été formulée consistant cette fois à étudier une classe face à toutes les autres. Nous allons maintenant détailler cette méthode.

Un contre tous

La seconde méthode, [Mossman, 1999], [Provost and Domingos, 2001], [Fawcett, 2006] et [Landgrebe and Duin, 2006], propose de manipuler les C classes en générant C courbes ROC, une pour chaque classe. Sur l'ensemble de toutes les classes, la $i^{\text{ème}}$ ($i \in \{1,\ldots,C\}$) courbe ROC correspond à l'évaluation des performances utilisant la classe c_i comme classe positive et toutes les autres classes comme

négatives, noté N_i.

$$N_i = \bigcup_{j \neq i} c_j \in E$$

avec $i, j \in \{1, \ldots, C\}$ et E l'ensemble de toutes les classes.
Le coût de mauvaise classification est, pour cette approche, fixe pour chaque classe car on ne cherche pas à différencier les erreurs. Dans ces conditions, l'espace d'évaluation des performances est à C dimensions, ce qui revient à n'utiliser que les éléments de la diagonale principale, les $\varepsilon_{i,i}$, de la matrice de confusion. Par exemple pour trois classes, nous obtenons un espace à trois dimensions facilement représentable. Dans le même esprit, la formulation proposée dans [Landgrebe and Duin, 2006] et [Landgrebe and Duin, 2007b] consiste à utiliser un cadre impliquant la pondération des sorties des classifieurs, ce qui est proche du seuil de classification utilisé dans le cas deux classes. La limitation de l'extension est la complexité des calculs qui est exponentielle avec l'augmentation du nombre de classes C, réduisant l'analyse à des problèmes ayant un faible C (3 à 6 classes).

A présent, comme nous l'avons fait dans la section 1.2.2, nous allons nous positionner dans un contexte de comparaison des performances de classifieurs. Nous devons pour cela comparer deux hyper-plans. Le problème est que selon les zones de l'espace les performances des classifieurs peuvent varier. On peut avoir sur une zone un hyper-plan qui est meilleur qu'un autre et sur une autre zone le second hyper-plan qui est meilleur que le premier. C'est pour cela que dans la littérature lorsque l'on souhaite comparer différents systèmes de classifications, on réduit les hyper-plans à des valeurs scalaires. Dans le cas général, la valeur scalaire qui est utilisée pour caractériser les performances ROC multi-classes est le volume sous l'hyper-surface ROC, "Volume Under the ROC hyper-Surface, VUS".

1.3.2 VUS

En deux classes, nous utilisions l'AUC comme critère d'évaluation des performances d'un classifieur. Maintenant que nous disposons d'un système à C classes, le critère d'évaluation que nous allons utiliser est le "volume sous l'hyper-surface ROC". Le VUS est l'adaptation des méthodes de calcul de l'AUC aux

problèmes multi-classes et nous retrouvons de la même manière que pour la courbe ROC multi-classes plusieurs approches.

Calcul d'un ensemble d'AUC

Les premières méthodes qui dérivent directement du calcul de l'AUC ont été proposées par [Hand and Tills, 2001] et [Provost and Domingos, 2001]. En effet, l'idée consiste à tracer un ensemble de courbes ROC, une classe contre une autre, à partir desquelles les AUC sont calculées. Avec l'équation 1.2.20, nous obtenons $C(C-1)$ aires et le critère d'évaluation des performances est alors :

- la moyenne des AUC pour [Hand and Tills, 2001],
- la somme des AUC pour [Provost and Domingos, 2001].

Dans le cas de deux classes, pour chaque paire de classes i et j, nous définissons $\hat{A}(i,j)$ comme étant la probabilité qu'un membre de la j dispose d'une probabilité à priori, plus faible, d'appartenir à la classe i, qu'un élément de la classe i. Dans ce cas, nous avons $\hat{A}(0|1) = \hat{A}(1|0)$. Cependant, dans le cas général (ayant plus de deux classes), nous avons $\hat{A}(i|j) \neq \hat{A}(j|i)$ ce qui nous oblige à adopter la notation suivante :

$$\hat{A}(i,j) = \frac{\left[\hat{A}(i|j) + \hat{A}(j|i)\right]}{2} \tag{1.3.22}$$

le calcul de la moyenne des AUC est alors :

$$M = \frac{2}{c(c-1)} \sum_{i<j} \hat{A}(i,j) \tag{1.3.23}$$

Le problème de cette généralisation est qu'elle ne correspond pas à l'extension théorique de l'AUC. En effet, aucun calcul de l'hyper-volume (VUS) n'est mis en oeuvre et cette méthode compromet l'indépendance vis-à-vis des probabilités à priori et des coûts de classification. En revanche, elle permet d'évaluer des systèmes ayant un nombre de classes très grand, ce qui ne sera pas toujours possible avec autres les solutions présentées ci-après.

Nous allons maintenant évoquer deux solutions qui étendent l'esprit de l'AUC aux systèmes multi-classes par le calcul formel du VUS puis dans la sélection d'une approximation.

Calcul formel et estimation - première approche

La première méthode du calcul du VUS que nous allons présenter est donnée dans [Ferri et al., 2003]. Nous nous basons ici sur la distinction de toutes les erreurs, évaluant une classe face à une autre, comme nous l'avons détaillé dans la partie 1.3.1 pour la construction de la courbe ROC multi-classes. Le calcul du volume sous l'hyper-surface ROC utilise un vecteur à $C(C-1)$ dimensions correspondant aux éléments de la matrice de confusion hors de la diagonale principale.
Pour des raisons de simplicité nous illustrons, par la suite, l'extension pour trois classes, les expressions pouvant être généralisées facilement. Dans ce contexte, nous considérons la matrice de coûts 1.7 pour un classifieur à trois classes. Nous

		Étiquette		
		a	b	c
Décision	a	h_a	x_1	x_2
	b	x_3	h_b	x_4
	c	x_4	x_6	h_c

TAB. 1.7:

obtenons alors le vecteur de dimension six composé de x_1, x_2, x_3, x_4, x_5 et x_6. Les valeurs h_a, h_b et h_c sont dépendantes et n'ont pas besoins d'être représentées, parce que :

$$h_a + x_3 + x_5 = 1 \ , \ h_b + x_1 + x_6 = 1 \ , \ h_c + x_2 + x_4 = 1$$

VUS maximum pour trois classes

Commençons par regarder le volume maximal représentant un classifieur parfait. Un point A est dans l'hypercube le plus grand si et seulement si :

$$x_3 + x_5 \leq 1 \ , \ x_1 + x_6 \leq 1 \ , \ x_2 + x_4 \leq 1$$

Il est facile d'obtenir le volume de l'espace déterminé par ces équations, simplement en utilisant la probabilité que six valeurs aléatoires sous une distribution uniforme $(0,1)$ suivent les conditions précédentes. Plus précisément :

$$VUS_3^{max} = P(U(0,1)+(0,1) \leq 1).P(U(0,1)+(0,1) \leq 1).P(U(0,1)+(0,1) \leq 1)$$

$$= [P(U(0,1) + (0,1) \leq 1)]^3$$

Il est facile de voir que la probabilité de la somme de deux valeurs aléatoires sous la distribution $U(1,0)$ est inférieure à 1 et exactement $\frac{1}{2}$, c'est-à-dire :

$$P(U(0,1)) + U(0,1) \leq 1) = \frac{1}{2}$$

en conséquence

$$VUS_3^{max} = \left(\frac{1}{2}\right)^3 = \frac{1}{8}$$

Nous considérons également le VUS maximal pour C classes. Il est facile de voir que le volume de l'espace pour C classes est :

$$VUS^{max} = \prod_C \left[P\left(\sum_{C-1} U(0,1) \leq 1 \right) \right] = \left[P\left(\sum_{C-1} \leq 1 \right) \right]^C$$

Cependant, la probabilité que la somme des $C-1$ valeurs aléatoires sous la distribution $U(0,1)$ soit inférieur à 1 est difficile à obtenir. En particulier, la fonction de densité de probabilité de la somme des n variables uniformes sur l'intervalle $[0,1]$

Calcul formel et estimation - seconde approche

Dans [Landgrebe and Duin, 2006] et [Landgrebe and Duin, 2007b], la démarche de calcul du VUS utilise uniquement les éléments de la diagonale principale (ce qui revient à faire du un contre tous), les $\varepsilon_{i,i}$ de la matrice de confusion, 1.6. Le VUS simplifié peut alors s'écrire :

$$VUS = \int \ldots \int \int x_{i_{1,1}} d\varepsilon_{2,2}\, d\varepsilon_{3,3} \ldots\, d\varepsilon_{C,C} \tag{1.3.24}$$

Ainsi la mesure considère l'hyper-volume à C dimension pour un problème à C classes. Cette mesure permet à un classifieur d'être évalué vis-à-vis de tous les points d'action correspondant aux valeurs sur la diagonale de la matrice de confusion. Si nous considérons ce fonctionnement, le VUS est similaire à l'AUC en cela qu'un classifieur fort aura un VUS élevé et un classifieur pauvre aura un score faible.

Dans un premier temps, considérons le cas trois classes. La dimension ROC simplifiée est de trois, entre les dimensions $\varepsilon_{1,1}, \varepsilon_{2,2}, \varepsilon_{3,3}$. Un classifieur aléatoire génère l'espace ROC présenté dans la figure 1.5. Un classifieur plus performant est présenté dans la figure 1.6, montrant comment le VUS augmente.

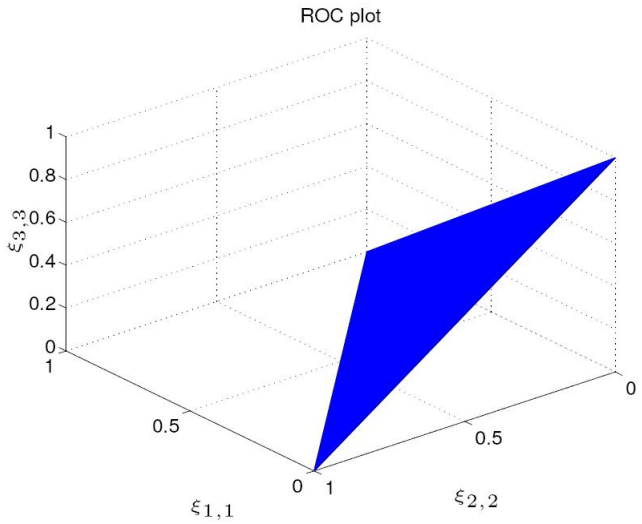


FIG. 1.5: Performance d'un classifieur aléatoire dans l'espace ROC trois classes

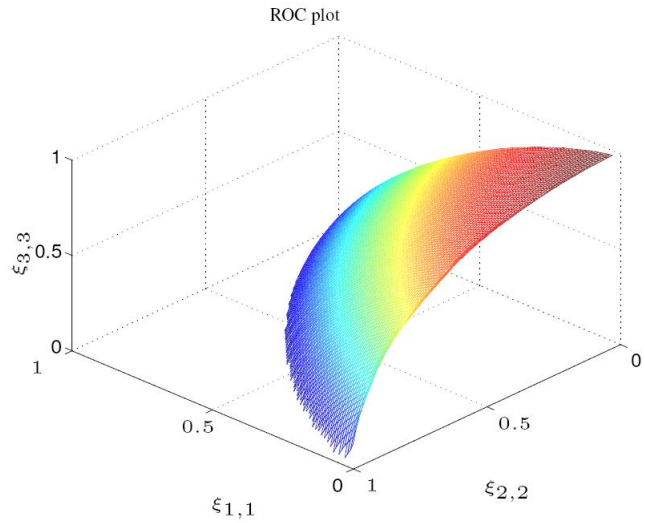


FIG. 1.6: Surface ROC pour un problème à trois classes

En effet, le VUS tend vers 1 lorsque la classification devient parfaite. Le volume occupé par un classifieur aléatoire peut être déterminé géométriquement par le calcul du volume sous la surface construite, ce qui correspond simplement à :

$\frac{1}{6}\varepsilon_{1,1}\varepsilon_{2,2}\varepsilon_{3,3} = \frac{1}{6}$. Ainsi l'évolution a modifiée la valeur minimale acceptable du VUS, de $\frac{1}{2}$ dans le cas deux classes, à $\frac{1}{6} = 0.16666$ dans le cas trois classes. Généraliser la limite pour C classes est géométriquement plus difficile. Une approche plus globale consiste à formaliser l'opérateur ROC comme un hyper-polyèdre, à l'instar de ce qui est formulé dans [Ferri et al., 2003]. Chaque sommet v_i de l'hyper-polyèdre peut être défini comme (notons que l'origine de l'espace est toujours inclue comme un sommet et qu'il y à C points par sommet) :

$$\begin{array}{cccccc} v_1 & 0 & 0 & 0 & \dots & 0 \\ v_2 & 1 & 0 & 0 & \dots & 0 \\ v_3 & 0 & 1 & 0 & \dots & 0 \\ v_4 & 0 & 0 & 1 & \dots & 0 \\ & \vdots & & & & \\ v_{C+1} & 0 & 0 & 0 & \dots & 1 \end{array} \tag{1.3.25}$$

Dans le cas pratique, l'estimation du VUS de manière générale nécessite une approche différente et une méthode appropriée consiste à utiliser la "numerical integration approach". Les points ROC sont prélevés de manière inégale puis recombinés via un rééchantillonnage linaire et une interpolation. La règle trapézoïdale est alors utilisée pour estimer le volume à C dimensions avec une fonction de r correspondant au nombre de pas utilisés par la surface ROC :

C	r	VUS estimé	VUS actuel
3	5	0.1667014	0.1666666
3	100	0.1666752	0.1666666
4	50	0.0417014	0.0416667
4	100	0.0416752	0.0416667
5	50	0.0083507	0.0083333
5	100	0.0083376	0.0083333
6	20	0.0014275	0.0013889
6	40	0.0013980	0.0013889

TAB. 1.8: Comparaison du VUS estimé et du VUS calculé

Ces résultats montrent que l'approche par intégration numérique ("numerical integration approach") fournit une bonne approximation du véritable VUS.

Expérimentation avec distribution des classes connues

Dans le but d'évaluer l'approche numérique du VUS et de vérifier les limites, plusieurs expériences sont conduites dans [Landgrebe and Duin, 2006] et [Landgrebe and Duin, 2007b]. L'idée réside en la création de classes gaussiennes avec des paramètres connus. Le premier ensemble de test est un problème gaussien à trois classes, ω_1, ω_2 et ω_3, dans lesquels les moyennes changent et les variances sont mises à l'unité. Les moyennes varient entre un problème séparable et un problème aléatoire. De la même manière la seconde expérience implique la variation de la moyenne de quatre classes gaussiennes. Les tableaux 1.9 et 1.10 montrent les résultats respectivement pour les cas à trois et quatre classes. Dans la figure 1.7, les distributions utilisées, dans la second et la quatrième expérience, en quatre classes, sont présentées, démontrant comment le chevauchement des classes augmente. Ces résultats vérifient que l'approche du

Moyennes	r	VUS estimé
$-0.05; 0.0; 0.05$	200	0.16876
$-0.3; 0.0; 0.3$	100	0.24140
$-0.5; 0.0; 0.5$	100	0.31428
$-1; 0.0; 1$	100	0.51214
$-1.5; 0.0; 1.5$	100	0.70597
$-4; 0.0; 4$	100	0.98582

TAB. 1.9: Résultats pour trois classes avec distribution connue, [Landgrebe and Duin, 2006]

Moyennes	r	VUS estimé
$-0.15; -0.05; 0.05; 0.15$	70	0.105688
$-0.75; -0.25; 0.25; 0.75$	50	0.07782
$-1; -0.33; 0.33; 1$	50	0.19972
$-1.5; -0.5; 0.5; 1.5$	50	0.33097
$-2.25; -0.75; 0.75; 2.25$	50	0.57990
$-3; -1; 1; 3$	50	0.75451

TAB. 1.10: Résultats pour quatre classes avec distribution connue, [Landgrebe and Duin, 2006]

VUS utilisée semble raisonnable. Nous constatons que si les problèmes varient du cas séparable vers le cas aléatoire, le VUS décroit en conséquence. Pour les cas ayant un fort chevauchement, les deux ensembles de tests montrent que le VUS approche de la limite basse prévue.

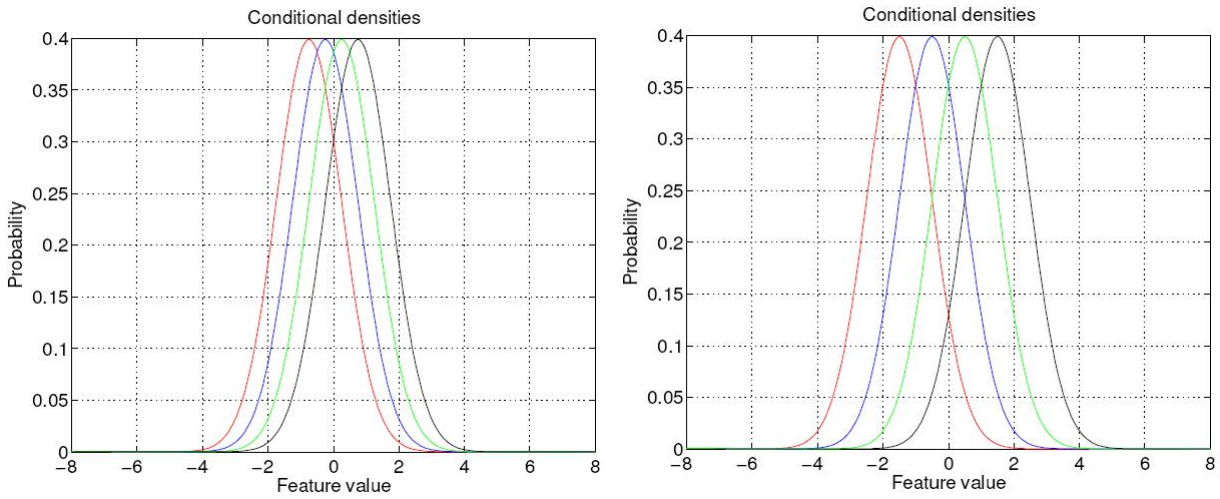


FIG. 1.7: Démonstration de la seconde et quatrième expérience dans la cas à quatre classes, [Landgrebe and Duin, 2006]

1.4 Application et résultats

Les résultats que nous présentons ici, démontrent l'intérêt du VUS dans des conditions réelles par comparaison de plusieurs classifieurs en compétition face à différents problèmes, [Landgrebe and Duin, 2006] et [Landgrebe and Duin, 2007b]. Le premier groupe de test considère les problèmes à trois classes, utilisant l'ensemble des bases suivantes : ***Banana*** est un ensemble à trois dimensions correspondant à une classe *Banana-Shaped*, [Duin et al., 2004], une classe gaussienne et une classe bimodale gaussienne, qui toutes se chevauchent, avec 5073 objets au total. ***Sign***, [Paclìck, 2004], est un ensemble correspondant à trois classes d'images de panneaux de signalisation routière, avec 381 objets au total. ***Sat***, [project, 2004], est un ensemble de 6435 valeurs multi-spectrales correspondant à une image satellite, avec 6 dimensions. les classes 1,3,5 et 6 ont été regroupées en une seule classe pour former un problème à trois classes avec les classes 2 et 4.
Le second groupe de test considère les problèmes à quatre classes, utilisant les ensembles suivant : ***Vehicle***, [Murphy and Aha, 1992], est une base composée de 846 objets représentant les silhouettes de quatre types véhicules. ***Digits***, est un ensemble composé de dix chiffres manuscrits, issus du "Dutch utility maps", [Murphy and Aha, 1992]. Dans cette base, les caractéristiques de Fourrier ont

été extraites des images. Il en résulte une représentation de 76 dimensions pour chaque chiffre. Les digits "3","6" et "9" ont été extraits et les autres regroupés pour former une unique classe.

Le protocole expérimental implique la rotation des données par une méthode de tirage aléatoire dans laquelle 80% des données sont utilisées pour l'apprentissage et le reste pour le test, et ce dix fois. Deux mesures des performances sont alors comparées, à savoir le taux d'équivalence d'erreur ("equal-error rate") et la mesure du VUS simplifée. Dans chaque test, plusieurs classifieurs sont comparés. Soit les abréviations suivantes : *sc* est l'utilisation de la variance des données. *pca* est l'analyse en composante principale. *fisher et nlfisher* sont la projection de Fisher et la projection de Fisher non linéaire. *nmc, ldc et qdc* sont respectivement les classifieurs "nearest-mean", "Bayes-linear" et "Bayes-quadratic". *mogc* est un mélange de classifieurs gaussiens. *knn3* est le classifieur des 3 plus proches voisins et *svc* est un classifieur à vecteur de support avec un noyau polynomial.

3-class	**Classifier**	**Error**	**VUS**
Banana	sc-pca1 nmc	0.329(0.004)	0.667(0.083)
	sc-mogc2,2,1	0.058(0.003)	0.990(0.002)
	sc-qdc	0.077(0.004)	0.970(0.006)
	sc-ldc	0.091(0.004)	0.964(0.007)
Sat	knn3	0.064(0.002)	0.911(0.020)
	ldc	0.111(0.001)	0.729(0.015)
	qdc	0.108(0.002)	0.862(0.012)
	mogc2,1,2	0.099(0.002)	0.866(0.012)
Sign	sc pca8 svc p2	0.115(0.018)	0.948(0.023)
	sc-pca10 mog2,2,2	0.075(0.003)	0.946(0.025)
	pca5 mog2,2,2	0.099(0.011)	0.954(0.019)
	pca5 qdc	0.179(0.020)	0.945(0.023)

FIG. 1.8: Résultats pour des problèmes à 3 classes [Landgrebe and Duin, 2006]

Les premiers résultats sont présentés pour les problèmes à trois classes, dans le tableau 1.8. La base *Banana* montre que le VUS tend vers des scores d'erreur équivalents, par exemple le classifieur *nmc* dispose d'une forte erreur et d'un VUS remarquablement bas par rapport aux autres classifieurs. Un résultat intéressant concerne la base *Sat*, comparons la seconde et la troisième approche.

4-class	Classifier	Error	VUS
Vehicle	fisher nmc	0.218(0.007)	0.512(0.037)
	ldc	0.219(0.006)	0.714(0.035)
	qdc	0.150(0.010)	0.834(0.036)
	sc-svc p2	0.164(0.007)	0.794(0.022)
	sc-svc p3	0.187(0.009)	0.727(0.039)
	nlfisher qdc	0.208(0.005)	0.724(0.041)
Digits	pca10 mog1,1,1,3	0.119(0.004)	0.985(0.008)
	pca15 mog1,1,1,3	0.114(0.003)	0.955(0.007)
	pca5 mog1,1,1,3	0.133(0.003)	0.956(0.008)
	pca10 qdc	0.127(0.004)	0.978(0.006)
	pca10 ldc	0.211(0.005)	0.704(0.041)
	nlfisher mogc1,1,1,3	0.158(0.003)	0.857(0.024)

FIG. 1.9: Résultats pour des problèmes à 4 classes [Landgrebe and Duin, 2006]

Dans ce cas, les deux classifieur ont le même taux d'erreur mais des scores de VUS sensiblement différents, montrant ainsi que le troisième modèle est un meilleur choix en moyenne. Pour la base *Sign*, les scores de VUS sont proches pour tous les classifieurs.
Ensuite, nous considérons les résultats en quatre classes, dans le tableau 1.9. Des observations peuvent également être faites. Par exemple, le premier et le second classifieur sont en concurrence vis-à-vis des taux d'erreur, mais les VUS sont significativement différents. Il apparait que le classifieur *linear* s'adapte mieux aux données que le modèle *fisher-nmc*. Finalement, dans le cas de digits le VUS tend à suivre le taux d'erreur. Nous pouvons voir que certains classifieurs ont de bonnes performance, le VUS approximant 1, alors que d'autres moins.

Les tests ont montrés, l'utilité du VUS approché dans le cadre multi-classes, montrant clairement des exemples où le VUS est nécessaire pour sélectionner le meilleur modèle de classifieurs lorsque les taux d'erreur sont équivalents.

Conclusion

Les courbes ROC sont des outils très utiles pour visualiser et évaluer les performances de classifieurs. Elles peuvent fournir une mesure des performances

plus riche que l'exactitude (accuracy) ou le taux d'erreur et ont des avantages par rapport aux autres mesures d'évaluation telles que la courbe Précision-Rappel et "lift curves". Cependant, comme pour toute métrique d'évaluation, son utilisation rigoureuse nécessite de connaître ses caractéristiques ainsi que ses limites. Les différents points que nous avons évoqués précédemment permettent de positionner les connaissances afin d'exploiter les possibilités de la courbe, de l'hyper-surface ROC à toute sorte de problèmes. Nous allons, dans ce qui va suivre, présenter une méthode d'optimisation multi-objectif pour la sélection de modèle SVM utilisant la courbe ROC.

CHAPITRE 2

Optimisation multi-objectif pour la sélection de modèle SVM

Le réglage des hyperparamètres d'un classifieur SVM est une étape cruciale afin d'établir un système de classification efficace. Généralement, au moins deux des paramètres doivent être soigneusement choisis : un paramètre relatif au noyau utilisé (γ dans le cas d'un noyau RBF par exemple), et le paramètre de régularisation (habituellement appelé C), qui permet d'intervenir sur le compromis entre l'erreur sur la base d'apprentissage et la complexité du modèle. La recherche de paramètres adaptés est appelée *sélection de modèle* dans la littérature, et ses résultats influent fortement sur les performances du classifieur.
Pendant longtemps, la sélection de modèle a été effectuée par une méthode de type "grid search", où une recherche systématique est mise en œuvre en discrétisant l'espace des paramètres à l'aide d'un pas fixe plus ou moins grand. Il a été montré que ces approches fonctionnaient mal et qu'elles étaient très gourmandes en temps de calcul [Hsu and Lin, 2002],
[Lavalle and Branicky, 2002].
Plus récemment, la sélection de modèle a été vue comme une tâche d'optimisation. Dans ce contexte, un algorithme d'optimisation est mis en œuvre afin de trouver l'ensemble d'hyperparamètres qui permettra d'obtenir les meilleures performances en classification. Parmi les algorithmes d'optimisation existants, la méthode de descente de gradient a été souvent employée pour la sélection de modèle SVM (voir [Chapelle et al., 2002], [Chung et al., 2003] par exemple). Cependant, il est bien connu que les méthodes à gradient imposent une dérivabilité du critère d'apprentissage et du noyau SVM par rapport aux paramètres

à optimiser, ce qui n'est pas toujours le cas. De plus, les performances des méthodes à descente de gradient dépendent fortement de l'initialisation et peuvent se stabiliser dans des extrema locaux.
Les algorithmes évolutionnaires ont également été employés pour la sélection de modèle SVM afin de surmonter les problèmes mentionnés ci-dessus. On peut citer par exemple les travaux décrits dans [Huang and Wang, 2006] ou dans [Wu et al., 2006] basés sur l'utilisation d'un algorithme génétique (AG), ou l'approche proposée par [Friedrichs and Igel, 2005] basée sur l'utilisation de stratégies évolutionnaires. Dans les deux cas, l'algorithme d'optimisation est employé dans le but de maximiser le taux de bonne classification du système.
Cependant, le fait d'utiliser un critère unique en tant qu'objectif pendant le processus d'optimisation constitue selon nous une limitation. En effet, comme nous l'avons montré dans le chapitre précédent, un critère unique ne suffit pas toujours à décrire les performances d'un système, en particulier dans le cas d'un problème comportant des effectifs de classes déséquilibrés ou des coûts de mauvaise classification asymétriques. Dans ces situations très fréquentes dans des problèmes réels, les probabilités a priori des classes et les coûts de mauvaise classification doivent idéalement être considérés pour évaluer les performances du classifieur. Or il est souvent difficile d'estimer ces coûts de mauvaise classification, par exemple quand le classifieur est inclus dans un système plus complexe. Dans le contexte d'un problème à deux classes sans connaissance des coûts, la courbe "Receiver Operating Characteristic" (ROC) introduite dans [Bradley, 1997] propose un meilleur critère d'évaluation des performances : elle représente le compromis entre le Faux Rejet (FR) et la Fausse Acceptation (FA), parfois aussi appelé compromis sensibilité/spécificité. Ainsi, pour l'optimisation d'un problème de classification à deux classes, deux critères doivent être minimisés à la place du critère unique et réducteur de bonne classification.
Nous considérons, ici, la sélection de modèle SVM comme un problème d'optimisation multi-objectif. L'algorithme d'optimisation évolutionnaire multi-objectif "Non dominated Sorting Genetic Algorithm II" (NSGAII, voir [Deb et al., 2000]) est appliqué pour optimiser les hyperparametres d'un SVM en utilisant FA et FR comme critères. Une telle stratégie permet d'obtenir en une seule génération un ensemble de classifieurs proposant chacun un compromis FA/FR optimal. Une fois cet ensemble de classifieurs entrainés, il sera possible de choisir le meilleur

du point de vue des contraintes de l'application, à l'aide d'une étape de validation sur une base dédiée.
La stratégie proposée est appliquée à un problème de discrimination chiffre/rejet qui s'inscrit dans un système d'extraction de champs numériques dans des documents manuscrits
[Chatelain et al., 2006]. Le terme rejet désigne ici tout ce qui n'est pas chiffre : lettre, mot ou fragment de mots, bruit, etc. Comme ce processus de discrimination chiffre/rejet est embarqué dans ce système plus complexe, les coûts de mauvaise classification ne peuvent pas être estimés a priori, et le meilleur compromis FA/FR du point de vue des performances globales du système (c'est-à-dire du point de vue du compromis rappel/précision en extraction des champs numériques) est inconnu. La stratégie proposée permet ainsi de surmonter ce problème en apprenant automatiquement plusieurs classifieurs proposant des compromis intéressants.

2.1 Position du problème

2.1.1 Les classifieurs SVM et leurs hyperparamètres pour la sélection de modèle

Comme décrit dans [Osuna et al., 1997], les problèmes de classification avec des coûts de mauvaise classification asymétriques et inconnus peuvent être pris en charge par les SVM en introduisant deux paramètres de pénalités différents C_- et C_+. Dans ce cas, étant donné un ensemble de m exemples d'apprentissage $x_i \in \Re^n$ appartenant à la classe y_i :

$$(x_1, y_1)...(x_m, y_m), x_i \in (R)^n, y_i \in -1, +1$$

la maximisation du lagrangien dual par rapport aux α_i devient :

$$Max_\alpha \left\{ \sum_{i=1}^{m} -\frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, y_j) \right\}$$

sous les contraintes :

$$\begin{cases} 0 \leq \alpha_i \leq C_+ & \text{pour } y_i = -1 \\ 0 \leq \alpha_i \leq C_- & \text{pour } y_i = +1 \\ \sum_{i=1}^{m} \alpha_i \alpha_j & \end{cases}$$

où les α_i représentent les multiplicateurs de Lagrange et $K(.)$ représente la fonction noyau. Dans le cas d'un noyau gaussien (RBF-SVM), $K(.)$ est défini par :

$$K(x_i, x_j) = exp\left(-\gamma x \left\|x_i - x_j\right\|^2\right)$$

Ainsi, dans le cas de coût de mauvaise classification asymétriques, trois paramètres doivent être déterminés pour réaliser un apprentissage optimal de SVM :

- Le paramètre du noyau, γ pour un RBF-SVM,
- les paramètres de pénalité introduits ci-dessus : C_- et C_+.

Dans la suite, une "ensemble d'hyperparamètres" désign donc un ensemble de valeurs données pour γ, C_-, C_+.

2.1.2 Critères pour la sélection de modèle SVM

Considérer la sélection de modèle comme un processus d'optimisation nécessite le choix d'un ou plusieurs critère(s) à optimiser. Comme indiqué précédemment, la courbe ROC d'un classifieur donné est un meilleur indicateur de performance que le simple taux de bonne classification.
Plusieurs approches ont été proposées dans la littérature pour obtenir la "meilleure courbe ROC possible", en réglant les paramètres intrinsèques d'un classifieur (en l'occurence la position et la valeur des α_i des vecteurs de support dans le cas des SVM). Ce type d'approche est généralement basé sur la réduction des deux critères FA et FR en un seul, tel que l'aire sous la courbe ROC (AUC : Area Under the ROC Curve) ou la F-mesure (FM). C'est le cas des travaux présentés dans [Rakotomamonjy, 2004], où un critère d'AUC est utilisé pour entrainer un classifieur SVM.
Dans ces travaux, les supports vecteurs et les α_i associés sont déterminés par minimisation du critère AUC. Cette approche ayant donné de bons résultats, nous les comparons avec les nôtres dans le chapitre 3.4, et référons à [Rakotomamonjy, 2004] pour les détails concernant le calcul de l'AUC et le processus d'optimisation de la méthode. Signalons que les approches reposant sur un critère AUC ont également été proposées dans [Ferri et al., 2002] et [Mozer et al., 2002] dans le cas d'autres classifieurs, et qu'une approche similaire basée sur la F-mesure est proposée dans [Musicant et al., 2003]. Dans tous ces travaux, le but est de réaliser un classifieur optimal au sens du critère de performance choisi (AUC ou FM).

Cependant, ces critères de performance ne sont que des indicateurs réducteurs de la courbe ROC. Ainsi, pour une valeur de FA donnée (respectivement FR), les classifieurs entrainés avec ce type de critère ne sont pas capables de produire le classifieur avec la valeur de FR optimale (respectivement FA). Ce qui signifie qu'un classifieur optimisant l'aire sous la courbe ROC ne garantit pas d'être le classifieur optimal pour une valeur donnée de FA (respectivement FR). Cette remarque est illustrée sur la figure 2.1. Plutôt que de rechercher un seul classifieur

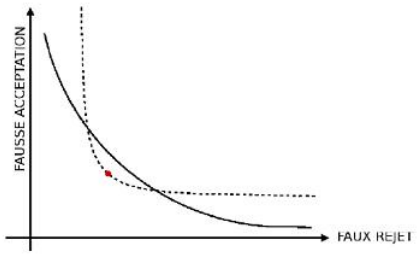


FIG. 2.1: La courbe en trait plein minimise l'aire sous la courbe ROC, mais en certains points la courbe en pointillés donne un meilleur compromis FA/FR

optimal en tous les points de la courbe, nous proposons de rechercher l'ensemble des classifieurs qui proposent les meilleurs points de fonctionnement, c'est à dire un ensemble d'ensembles d'hyperparamètres (γ, C_+, C_-). Ainsi, l'ensemble des points de fonctionnement optimaux de l'ensemble de classifieurs peut être vu comme un "front ROC" (voir figure 2.2). La méthode que nous proposons ici est

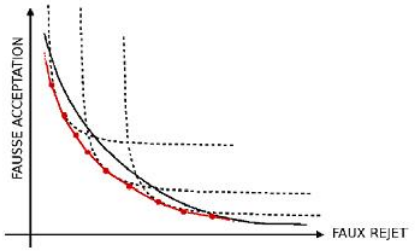


FIG. 2.2: Ensemble de compromis FA/FR optimaux obtenus par une population de classifieurs

basée sur l'optimisation des compromis FA/FR d'un ensemble de classifieurs a l'aide d'une véritable optimisation multicritères. Cela implique la mise en oeuvre d'un algorithme d'optimisation multiobjectif pour la recherche des ensembles d'hyperparamètres, chaque ensemble d'hyperparamètres optimisant un compromis FA/FR. La dimension de l'espace des objectifs étant supérieur a 1, le concept

de dominance employe dans le domaine de l'optimisation multiobjectif doit etre introduit pour comparer les performances de deux classifieurs.
Le concept de dominance a été propose par Vilfredo Pareto au 19eme siècle. On dit qu'un vecteur $\vec{u}$ (dans notre cas, un ensemble donne (C_+, C_-, γ)) domine un autre vecteur $\vec{v}$ si $\vec{u}$ n'est pas pire que $\vec{v}$ pour n'importe lequel des objectifs (FA et FR) et si $\vec{u}$ est meilleur que $\vec{v}$ pour au moins un objectif. La notation est la suivante : $\vec{u} \prec \vec{v}$. Plus formellement, un vecteur $\vec{u} = (u_1, u_2, ..., u_k)$ domine un vecteur $\vec{u} = (v_1, v_2, ..., v_k)$ si et seulement si :

$$\forall i \in 1, ..., k, u_i \leq v_i \wedge \exists j \in 1, ..., k : u_j < v_j$$

Étant donné le concept de dominance, l'objectif d'un algorithme d'optimisation multiobjectif est de chercher l'ensemble de Pareto, défini comme l'ensemble des solutions dans l'espace des paramètres engendrant des solutions non dominées dans l'espace des objectifs :

$$\text{Ensemble de Pareto } = \left\{ \vec{u} \in \vartheta / \neg \exists \ \vec{v} \in \vartheta, \vec{f(v)} \prec \vec{f(u)} \right\}$$

où ϑ désigne l'espace des paramètres ou les contraintes sont satisfaites, et $\vec{f}$ désigne le vecteur d'objectifs. Du point de vue de la sélection de modèle SVM, l'ensemble de Pareto correspond à la population d'ensembles d'hyperparamètres produisant tous les compromis FA/FR optimaux. Dans l'espace des objectifs, cet ensemble de compromis optimaux est appelé front de Pareto. Remarquons que dans le cadre de la sélection de modèles SVM, le front de Pareto pourrait être comparé à la courbe ROC qui décrirait le meilleur ensemble de compromis FA/FR. Dans notre cas, le front de Pareto correspond toutefois aux compromis FA/FR obtenus à l'aide d'un ensemble de classifieurs, alors que la courbe ROC est obtenue à l'aide d'un seul classifieur. Si la comparaison entre notre "front ROC" et une courbe ROC n'est pas théoriquement valide, elle permet toutefois de bien saisir le concept proposé dans cette article.
L'approche proposée cherche donc à approximer l'ensemble optimal de Pareto d'un classifieur SVM à deux classes à l'aide d'une optimisation multiobjectif évolutionnaire. Nous dressons maintenant un bref panorama des méthodes d'optimisation multiobjectif évolutionnaire, et décrivons l'algorithme choisi ainsi que son application à la sélection de modèles SVM.

2.2 Optimisation multiobjectif évolutionnaire

Nous recherchons l'ensemble de classifieurs SVM décrivant l'ensemble des compromis FA/FR optimaux. Les classifieurs sont paramétrés par les hyperparamètres (C_+, C_-, γ). Du point de vue de l'optimisation multiobjectif, cet ensemble peut être vu comme un ensemble de Pareto. L'ensemble des compromis FA/FR associés à ces classifieurs forme le front que nous recherchons. Les algorithmes évolutionnaires sont bien adaptés à la recherche de ce front car ils sont capables grâce à leur parallélisme implicite de dégager des solutions optimales de façon plus efficace qu'une méthode exhaustive.

2.2.1 Panorama des approches existantes

Depuis les premiers travaux de [Schaffer and Grefenstette, 1985] au milieu des années 80, un certain nombre d'approches d'optimisation multiobjectif évolutionnaire a été proposé : MOGA [Fonseca and Flemming, 1993], NSGA [Srinivas and Deb
NPGA [Horn et al., 1994],
SPEA [Zitzler and Thiele, 1999], NSGA II [Deb et al., 2000], PESA [Corne et al., 2000] ou encore SPEA2 [Zitzler et al., 2001]. Dans une étude comparative, [Khare et al., 2002] compare les performances des trois algorithmes les plus populaires : SPEA2, PESA et NSGA-II. Ces trois approches sont élitistes, c'est-à-dire que les meilleures solutions non dominées trouvées sont sauvegardées dans une archive afin d'assurer la préservation de bonnes solutions. Cette étude comparative a été menée sur différents problèmes, avec pour mesure de qualité les deux critères importants pour un algorithme multiobjectif : se rapprocher le plus possible du front de Pareto et obtenir une bonne dispersion des solutions sur ce front. Les résultats de cette étude (qui ont été confirmés dans [Zitzler et al., 2001] et [Bui et al., 2004]) montrent qu'aucun des algorithmes ne domine les autres au sens de Pareto. SPEA2 et NSGA-II offrent des performances similaires en terme de convergence et de diversité. Leur convergence est inférieure à celle de PESA mais la diversité des solutions est meilleure. L'étude montre également que NSGA-II est plus rapide que SPEA2.
Dans le contexte de la sélection de modèles SVM, le calcul des fonctions objectifs prend beaucoup de temps puisqu'il faut entrainer puis évaluer le classifieur pour chaque ensemble d'hyperparamètres. De plus, une bonne diversité des so-

lutions est nécessaire puisqu'on ne connait pas le point de fonctionnement sur le front de Pareto. Nous avons donc choisi l'algorithme NSGA-II. Nous donnons dans la partie suivante une description de cet algorithme.

2.2.2 NSGA-II

NSGA-II est une version modifiée de l'algorithme NSGA [Srinivas and Deb, 1994]. C'est une approche rapide, élitiste et sans paramètres qui manipule une population de solutions et utilise un mécanisme explicite de préservation de la diversité. Initialement, une population parent P_0 de N solutions (ou individus) est créée aléatoirement. Cette population est triée sur une base de non-dominance à l'aide d'un algorithme rapide. Ce tri associe un rang de dominance à chaque individu. Les individus non dominés ont un rang de 1 et constituent le front $\mathbb{F}_1$. Les autres fronts $\mathbb{F}_i$ sont ensuite définis récursivement en ignorant les solutions des fronts précédemment détectés. Ce tri est illustré sur la figure 2.3 (à gauche) dans le cas d'un problème à deux objectifs $(f1, f2)$, où pour une population de 16 individus, 3 fronts sont détectés. Les opérateurs de croisement, de recombinaison et de

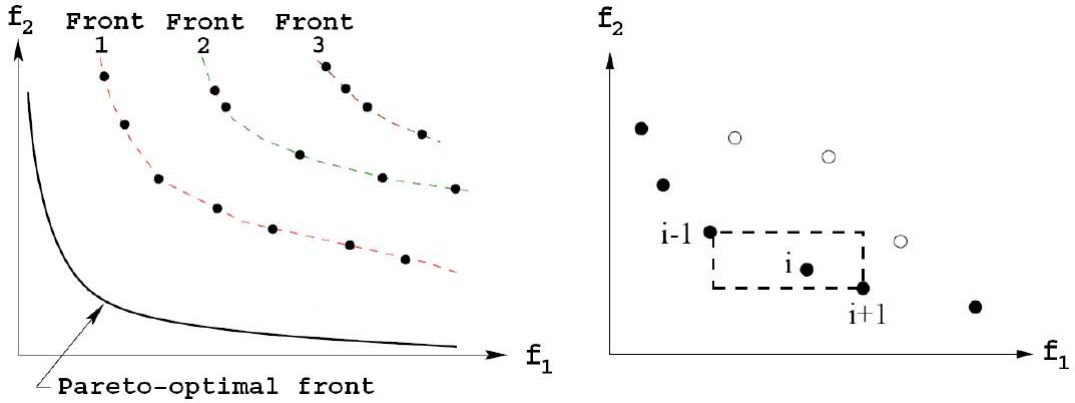


FIG. 2.3: Illustration du concept $\mathbb{F}_1$. Sur la figure de droite, les points noirs sont les vecteurs dominants, les points blancs sont dominés

mutation (voir [Goldberg, 1989] et [Deb et al., 2000] pour plus de détails) sont ensuite utilisés pour créer une population fille Q_0 de même taille que P_0. A l'issue de cette première étape, l'algorithme est itéré durant M générations. À chaque itération, t désigne le numéro de génération courante, $\mathbb{F}$ désigne le résultat de la procédure de tri, $\mathbb{F}_i$ désigne le i_{eme} front de $\mathbb{F}$, P_t et Q_t désignent respectivement

la population et la progéniture à la génération t, et R_t est une population temporaire.
Remarquons que chaque itération de l'algorithme débute avec une fusion des populations parent P_t et fille Q_t pour construire R_t. Cette population de $2N$ solutions est triée à l'aide de la procédure de tri de non-dominance pour construire la population P_{t+1}. Durant cette étape, un autre critère de tri est appliqué pour conserver l'effectif de P_{t+1} à une taille constante durant l'intégration des $\mathbb{F}_i$ successifs. Son but est de prendre en compte la contribution des solutions pour la diversité de la fonction objectif dans la population. Ce tri des individus de dominance équivalente est effectué selon une mesure de dispersion appelée *crowding distance* [Deb et al., 2000]. Cette mesure est basée sur le calcul de la distance moyenne aux deux points de part et d'autre de l'individu considéré selon les deux objectifs (voir figure 3 droite). Plus la surface (resp. volume pour 3 objectifs, hypervolume au delà de 3) autour de l'individu considéré est grande, plus la solution est bonne du point de vue de la diversité. Les solutions de R_t contribuant le plus à la diversité sont ainsi favorisées dans la construction de P_{t+1}. Cette étape est désignée dans l'algorithme 1 par : trier$(F_i \prec n)$, où $\prec n$ désigne une relation d'ordre partiel basée à la fois sur la dominance et sur la *crowding distance*. Selon cette relation, une solution i est meilleure qu'une solution j si $(i_{rank} < j_{rank})$ ou si $(i_{rank} = j_{rank})$ et $(i_{distance} > j_{distance})$. Grâce à cet algorithme, la population P_t converge nécessairement vers un ensemble de points du front de Pareto puisque les solutions non dominées sont préservées à travers les générations. De plus, le critère de dispersion (*crowding distance*) garantit une bonne diversité dans la population [Deb et al., 2000].

2.2.3 Application de NSGA-II à la sélection de modèle SVM

Dans cette section, nous présentons l'application de l'algorithme NSGA-II au problème de sélection demodèle SVM. Pour cela, deux points particuliers doivent être précisés :

– **Le codage des individus** : rappelons que trois paramètres sont impliqués dans l'apprentissage des classifieurs SVM avec des coûts de mauvaise classification déséquilibrés : C_+, C_- et γ. Ces trois paramètres constituent l'espace des paramètres de notre problème d'optimisation. Chaque individu de la population doit donc coder ces trois valeurs réelles. Nous avons choisi

un codage réel des paramètres afin d'être le plus précis possible.
– **La procédure d'évaluation** : chaque individu de la population correspond à un ensemble de trois hyperparamètres. Afin d'évaluer la qualité de cet individu, un apprentissage SVM classique piloté par l'ensemble d'hyperparamètres encodé est lancé. Ce classifieur SVM est ensuite évalué sur une base de validation à l'aide des critères FA et FR.

2.3 Application et résultats

Dans cette section, nous appliquons notre approche sur les bases de l'UCI afin de la valider et de comparer nos résultats à ceux présents dans la littérature. Jusqu'alors, nous avons présenter le cadre théorique permettant de répondre à notre problématique. Nous allons, maintenant, décrire la méthodologique que nous avons utilisée afin de parvenir aux résultats présentées dans la section 2.3.2.

2.3.1 Protocole expérimental

Le protocole expérimental est là pour faire le lien entre notre étude bibliographique et l'application sur des données réelles. Il va permettre de structurer les actions à mener et de reproduire nos expériences dans les mêmes conditions.

Estimer l'exactitude d'un classifieur induit par des algorithmes d'apprentissage supervisés est important non seulement pour prédire ces futures performances, mais également pour choisir un classifieur à partir d'un ensemble (sélection de modèle), ou pour combiner les classifieurs [Wolpert, 1992]. Pour estimer la précision finale d'un classifieur, nous souhaitons une méthode d'estimation ayant un faible biais et une faible variance. La validation croisée, ou "Cross-Validation", est une réponse statistique permettant de stabiliser les résultats de classification.
Nous avons deux séries de test à mener afin de répondre à nos problématiques. La première partie consiste en la sélection de modèles SVM, la seconde partie qui consiste en la sélection et la combinaison des classifieurs est détaillée dans le chapitre 3.
La figure 2.4 schématise l'ensemble du dispositif.

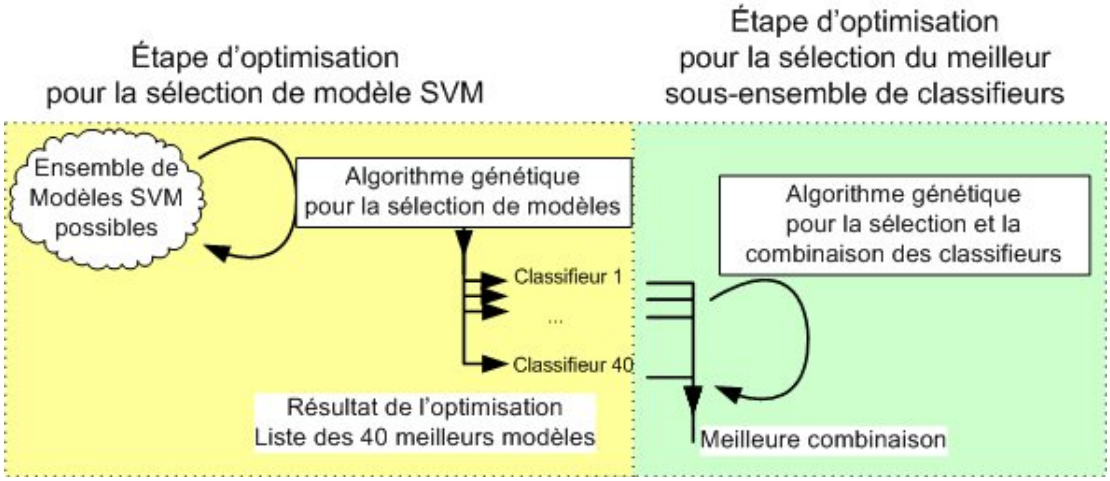


FIG. 2.4: Schéma de principe de l'ensemble du système

Pour répondre aux besoins de notre système, nous allons utiliser la cross-validation, c'est-à-dire, que la base d'éléments va être divisé en k sous-ensembles, "k-folds", tous équivalents en taille et par rapport à la distributions des classes. Nous allons ensuite regrouper les sous-ensembles de manière à former plusieurs nouvelles bases :

- une base d'apprentissage (base App) utilisée pour l'apprentissage des SVM,
- une base de validation (base Valid_SVM) utilisée par NSGA-II pour évaluer les SVM,
- une base de test (base Test) utilisée pour obtenir les performances sur le Front ROC (AUF) et les performances en combinaison,
- une base de validation (base Valid_Combi) pour optimiser la combinaison.

Par défaut, notre système doit donc diviser la base initiale en quatre folds au minimum. Le tableau 2.1 et la figure 2.5 présentent la manière dont sont créées les bases à partir des folds.

Le nombre de combinaison possibles issues de la cross validation dépend directement du nombre de division de la base initiale. Il s'agit d'un regroupement d'un ensemble de combinaison et d'arrangement des folds. Soit le nombre de sous-ensemble $k \in \mathbb{N}$ tel que $k \geq 4$ et $n_{app} \in \mathbb{N}$ le nombre de sous-ensemble utilisé pour la base d'apprentissage tel que $n_{app} = k - 3$.
Nous cherchons l'ensemble des combinaisons de n_{app} parmi k, $C_k^{n_{app}}$, et l'en-

base	# sous-ensemble utilisé
Apprentissage SVM	$k-3$
Validation des SVM	1
Test	1
Validation combinaison	1

TAB. 2.1: Répartition du nombre de sous-ensemble pour la division de la base initiale en k éléments

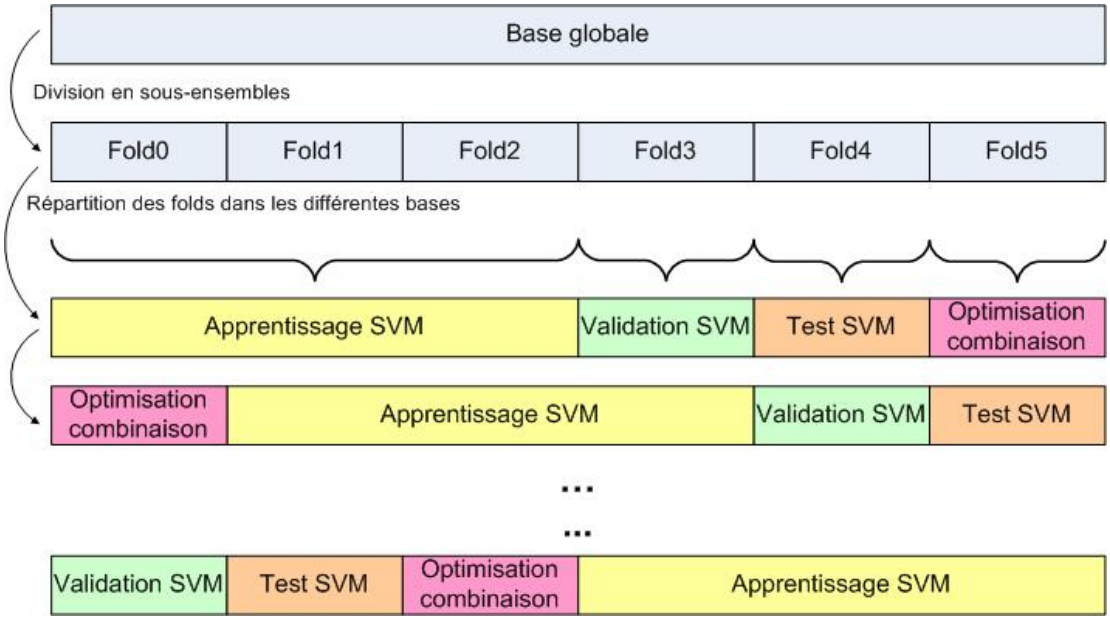


FIG. 2.5: Schéma de principe de la cross-validation pour un découpage de la base en 6 sous-ensembles

semble des arrangements de 3 parmi 3, A_3^3, afin de calculer le nombre de rotation de validation croisée, N_{CV}.

$$N_{CV} = C_k^{n_{app}} * A_3^3 \tag{2.3.1}$$

Prenons l'exemple d'un découpage de la base en 6 folds, figure 2.5, $k = 6$ et $n_{app} = 3$.

$$\begin{aligned} N_{CV} &= C_6^3 * A_3^3 \\ &= 20 * 6 = 120 \end{aligned} \tag{2.3.2}$$

Nous obtenons au final 120 manières d'organiser nos 6 folds afin de répondre aux besoins de notre système. Le nombre de sous-ensembles va donc définir la

faisabilité et la vitesse de traitement de notre système.

Nous venons de définir le socle de notre travail grâce à la validation croisée. Chaque ensemble de base généré va être utilisé par nos deux systèmes d'optimisation (la sélection de modèle SVM et la combinaison de classifieurs). Nous rappelons que la première étape consiste en la sélection de modèle de classifieurs SVM, l'idée est d'obtenir un ensemble de 40 classifieurs les plus performants entraînés sur la base d'apprentissage, optimisés par algorithme génétique sur la base de validation SVM puis évalués sur le base de test. La figure 2.6 schématise le fonctionnement de ce module.
Nous verrons, dans le chapitre 3, l'étape suivante qui consiste à sélectionner puis

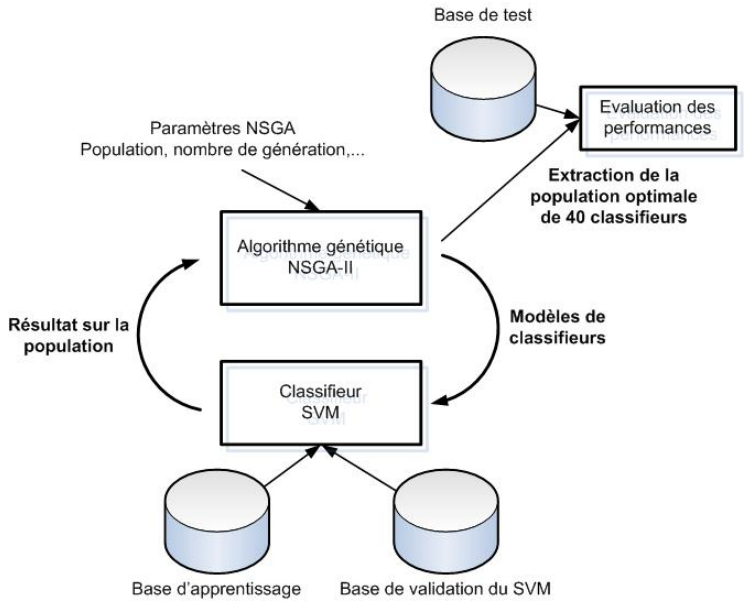


FIG. 2.6: Schéma d'optimisation pour la sélection de modèle SVM

à combiner les classifieurs.

Le choix du SVM a été motivé par son efficacité dans un contexte de classification à deux classes. Les plages de valeurs des hyperparamètres sont données dans la table 2.2. Une précision de 10^{-6} est utilisée pour ces paramètres (précision de la machine sur le type flottant). Les classifieurs SVM que nous utilisons sont issus de la librairie "LibSVM",[Chang and Lin, 2001]. En ce qui concerne les paramètres de NSGA-II, nous avons employé les valeurs classiques propo-

sées dans [Deb et al., 2000]. Parmi celles-là, notons que la taille de la population a été fixée à 40 afin d'obtenir suffisamment de points sur l'estimation du front de Pareto. Le nombre de générations a été fixé à 200 afin de pouvoir parcourir au maximum l'espace des hyperparamètres. A l'issue de l'optimisation par l'algorithme NSGA-II, on obtient donc une population de 40 classifieurs.

γ	C_-	C_+
$0-1$	$0-500$	$0-5000$

TAB. 2.2: Plage des valeurs pour γ, C_-, C_+

Notons que pour un ensemble donné d'hyperparamètres, les paramètres intrinsèques des classifieurs SVM (les positions et poids des vecteurs de support) sont déterminés à l'aide d'une optimisation mono-objectif adaptée à cette tâche. Ainsi, l'algorithme évolutionnaire se concentre sur le choix des hyperparamètres. Cette approche diffère donc des autres travaux mettant en œuvre des algorithmes évolutionnaires pour régler à la fois les paramètres intrinsèques et les hyperparamètres. Nous pouvons en particulier mentionner les travaux de [Kupinski and Anastasio, 1999], [Anastasio et al., 1998], [Fieldsend and Everson, 2004] et [Everson and Fieldsend, 2006]. Tous ces travaux sont limités à des classifieurs très simples (c'est-à-dire possédant un faible nombre de paramètres intrinsèques) à cause de l'impossibilité pour un algorithme évolutionnaire de traiter un nombre élevé de paramètres. Dans un contexte mono-objectif, une telle limitation a été contournée en développant des méthodes spécifiques telles que la maximisation du lagrangien pour les SVM ou la rétro-propagation du gradient pour les MLP. Dans un contexte multiobjectif, l'utilisation de la maximisation du lagrangien pour le réglage des paramètres intrinsèques couplée à l'algorithme évolutionnaire en charge des hyperparamètres en nombre plus réduit constitue ainsi une solution intéressante.

Nous allons dans la partie suivante présenter les résultats obtenus avec notre système sur différentes base de l'UCI Repository.

Problème	# exemples	# attributs	Distribution des classes	
			-1	1
australian	690	14	383	307
wdbc	569	30	212	357
breast cancer	658	9	434	224
ionosphère	351	34	126	225
heart	270	13	150	120
pima	768	8	500	268
sonar	208	60	111	97

TAB. 2.3: Description de problèmes de l'UCI à deux classes

2.3.2 Validation de l'approche sur les bases de l'UCI

La méthodologie présentée précédemment est démontrée sur un ensemble de données réelles issues de l'UCI. Elles nous permettent de tester notre approche et de comparer nos résultats aux travaux déjà effectués. Les bases que nous utilisons pour nos expérimentations sont exclusivement à deux classes et issues de différents domaines, traitement d'image, traitement de signal ou reconnaissance de caractère. Dans un premier temps, nous allons présenter les données utilisées puis nous enchaînerons sur la présentations des résultats et nous terminerons par les conclusions sur les expérimentations.

Les bases présentées ici sont toutes extraites de l'*UCI Machine Learning Repository* consultables à l'adresse suivante `http://archive.ics.uci.edu.ml/`. Le nombre d'exemple, de caractéristiques ainsi que la distribution des classes est donné dans le tableau 2.3.
Les bases que nous avons sélectionné sont les suivantes :

- **australian**, *Statlog Australian Credit Approval*, correspond à des caractéristiques pour une application de cartes bancaires.
- **wdbc**, *Wisconsin Diagnostic Breast Cancer* et **breast cancer**, *Wisconsin Breast Cancer Database*, représentent les caractéristiques extraites d'images de cancer du sein.
- **ionosphère**, données issues d'un radar composé d'un assemblage de 16 antennes.
- **heart**, *Statlog heart*, représente différentes caractéristiques de maladies car-

Problème	Distriburion des folds			
	0 à 4		5	
	-1	1	-1	1
australian	63	51	68	52
wdbc	35	59	37	62
breast cancer	72	37	74	39
ionosphère	21	37	21	2
heart	25	20	25	20
pima	83	44	85	48
sonar	18	16	21	17

TAB. 2.4: Description de la répartition des sous-ensembles pour la Cross-Validation

diaques.

– **pima**, *Pima Indians Diabetes*, correspond à une étude sur le diabète dans une population de femmes originaires de la tribu indienne d'Amérique du nord "Pima".
– **sonar**, *Connectionist Bench(Sonar, Mines vs. Rocks)*, contient les caractéristiques issues d'un sonar pour l'identification de pierres et d'objets métalliques.

L'idée est de comparer l'aire sous la courbe ROC obtenue par un classifieur unique avec l'aire sous notre front ROC. Nous insistons à nouveau sur le fait que cette comparaison n'est théoriquement pas correcte [1], mais qu'elle permet de visualiser ce qu'apporte notre approche par rapport aux approches classiques. Nous avons reporté dans le tableau 2.5 les meilleurs résultats parmi les travaux de [Boström, 2005], [Cortes and Mohri, 2004], [Ferri et al., 2002], [Rakotomamonjy, 2004], [Zhou and Jiang, 2004] et [Wu, 2005], que nous comparons à l'aire sous notre front ("AUF" pour Area Under the Front). Notons qu'il existe plusieurs représentations équivalentes de la courbe ROC que nous avons détaillé dans le chapitre 1.

Nous remarquons, à partir du tableau 2.5, que l'aire sous le front est proche voir nettement supérieure à l'aire sous la courbe, quel que soit le problème. Ces résultats montrent clairement que notre approche permet d'atteindre des points de fonctionnement localement beaucoup plus intéressants que les points d'une courbe globalement optimisée. Dans la mesure où dans la plupart des systèmes,

[1] En effet, il n'est pas correct de comparer un ensemble discret des meilleurs points de plusieurs courbes avec une courbe continue obtenue en faisant varier le seuil de décision d'un seul classifieur

Problème	AUC littérature	Référence	AUF
australian	90.25 ± 0.6	[Wu, 2005]	96.51 ± 0.01
wdbc	94.7 ± 4.6	[Ferri et al., 2002]	98.18 ± 0.01
breast cancer	99.13	[Boström, 2005]	98.83 ± 0.01
ionosphère	98.7 ± 3.3	[Rakotomamonjy, 2004]	97.06 ± 0.08
heart	92.6 ± 0.7	[Wu, 2005]	94.49 ± 0.03
pima	84.80 ± 6.5	[Cortes and Mohri, 2004]	94.37 ± 0.02
sonar	81.2 ± 5.1	[Zhou and Jiang, 2004]	86.06 ± 0.59

TAB. 2.5: Comparaison des AUC obtenues dans la littérature et des aires sous le front (AUF pour Area Under the Front).

un seul point de fonctionnement de la courbe est utilisé, notre approche se révèle particulièrement intéressante.

Conclusion

Dans cette partie, nous avons présenté une stratégie pour la sélection de modèle SVM pour des problèmes où les coûts de mauvaise classification sont déséquilibrés et inconnus. Pour cela, nous avons proposé une méthode d'apprentissage pour entrainer automatiquement une population de classifieurs proposant chacun des points de fonctionnement localement optimaux. L'approche est basée sur un algorithme évolutionnaire multiobjectif permettant d'optimiser les hyperparamètres des classifieurs. Le système produit ainsi un front ROC dans lequel il est possible de choisir le classifieur convenant le mieux aux contraintes de l'application visée.
Nous avons montré sur des bases de référence que cette approche fournissait des résultats intéressants. Soulignons que cette approche simple et générique peut être utilisée avec n'importe quel classifieur comportant des hyperparamètres (KPPV, réseaux de neurones, etc.). Concernant l'application aux SVM, d'autres paramètres (type de noyau, ...) et d'autres objectifs (nombre de vecteurs de support, temps de décision) peuvent également être intégrés dans le processus d'optimisation.

Le problème attenant au Front ROC est qu'il s'agit d'un ensemble discret

et nous le comparons aux ensembles continus, les courbes ROC. Comme nous l'avons déjà dit, la comparaison entre un ensemble discret et un ensemble continu n'est théoriquement pas correcte. Pour résoudre ce problème, l'idée que nous allons développer dans le chapitre 3 consiste à sélectionner et à combiner les classifieurs de notre ensemble afin de former un ensemble continu comparable aux éléments présents dans la littérature.

CHAPITRE 3

Combinaison de classifieurs

La combinaison de classifieurs est une excellente alternative à l'utilisation d'un unique classifieur et est devenue au fil du temps un domaine de recherche très riche. Les méthodes de sélection et de combinaison de classifieurs montrent leur intérêt ainsi que leurs performances dans de nombreuses applications, telles que la reconnaissance de formes ou de lieux, par rapport à l'utilisation d'un seul classifieur. Une multitude de recherches sont menées dans ce domaine avec par exemple [Grefenstette, 1992], [Ohkura and Ueda, 1995], [Kuncheva, 2002a], [Kuncheva, 2002b] et [Wang and Rhee, 2007] .
Prenons $D = D_1, D_2, ..., D_L$ un ensemble de classifieurs et $\Omega = \omega_1, ..., \omega_c$ un ensemble de classes. Chaque classifieur utilise un vecteur de caractéristique $x \in \mathbb{R}^n$ en entrée et lui assigne une classe parmi Ω, c'est-à-dire, $D_i : \mathbb{R}^n \mapsto \Omega$, où de manière équivalente, $D_i(x) \in \Omega, i = 1, ..., L$. Dans beaucoup de cas, la sortie du classifieur est un vecteur de dimension c fournissant la confiance envers chaque classe, c'est-à-dire,

$$D_i(x) = \left[d_{i,1}(x), ..., d_{i,c}\right]^T \tag{3.0.1}$$

Sans perdre la généralité de l'approche, nous pouvons réduire l'intervalle de $d_{i,j}$ à $[0,1]$, $i = 1, \ldots, L$, $j = 1, \ldots, c$ et appeler la sortie des classifieurs *"soft labels"* [Bezdek et al., 1999]. Ainsi, $d_{i,j}$ est donc le "support" fournit par le classifieur D_i concernant l'hypothèse que x appartienne à la classe ω_j (le plus souvent, il s'agit d'une estimation de la probabilité a postériori $P(\omega_i|x)$). Combiner des classifieurs revient à trouver la classe de x basée sur les sorties des L classifieurs $D_1(x), \ldots, D_L(x)$. Également, nous pouvons trouver un vecteur ayant c dimen-

sions, pour les classes, tel un "soft label" pour x, noté

$$D(x) = [\mu_1(x), \ldots, \mu_c(x)]^T \tag{3.0.2}$$

En fonction des besoins, il est également possible d'utiliser la règle du meilleur élément. Décider de l'appartenance de x à la classe ω_s si et seulement si

$$\mu_s(x) \geq \mu_t(x), \forall t = 1, \ldots, c \tag{3.0.3}$$

Deux stratégies sont évoquées dans la littérature sur la combinaison de classifieurs : "la sélection" et "la fusion" de classifieurs. La présomption dans la sélection de classifieurs est que chaque classifieur est performant sur une petite zone de l'espace des caractéristiques. Lorsque l'on souhaite émettre une décision sur un vecteur de caractéristique $x \in \mathbb{R}^n$, on regarde l'ensemble des réponses des classifieurs et celui donnant la plus grande proximité est sélectionné pour attribuer une classe au vecteur x. La fusion de classifieurs suppose que tous les classifieurs soient équivalent au niveau des performances sur l'ensemble de l'espace des caractéristiques et les décisions de tous les classifieurs sont prises en compte pour la décision finale. Il y a de nombreuses méthodes de combinaison à mi-chemin entre ces deux extrêmes, par exemple, quand les performances individuelles varient sur tout l'espace $\mathbb{R}^n$.

Nous allons, dans un premier temps, présenter les méthodes de combinaison de classifieurs qui vont nous permettre, dans un second temps, d'introduire les méthodes et les techniques de sélection de classifieurs, afin d'obtenir à partir de notre ensemble discret de classifieurs (qui forment le Front ROC) une solution pouvant être comparée aux résultats présentés dans la littérature (basés sur la courbe ROC).

3.1 Méthodes de combinaison de classifieurs

La multiplication des travaux sur la combinaison a entraîné la mise au point de nombreux schémas traitant les données de manières différentes [Heutte, 1994], [Moobed, 1996], [Rhaman and Fairhust, 1999]. Trois approches pour la combinaison de classifieurs peuvent être envisagées : parallèle, séquentielle et hybride. D'autres organisations avec bouclage ou avec interaction sont aussi possibles

[Vuurpijl and Schomaker, 1998]. Mais, malgré la diversité des schémas de combinaison, la détermination de la meilleure organisation reste un problème ouvert.

3.1.1 Approche séquentielle

La combinaison séquentielle, appelée également combinaison série, est organisée en niveaux successifs de décision permettant de réduire progressivement le nombre de classes possibles. Dans chaque niveau, il existe un seul classifieur qui prend en compte la réponse fournie par le classifieur placé en amont afin de traiter les rejets ou confirmer la décision obtenue sur la forme qui lui est présentée (figure 3.1). Une telle approche peut être vue comme un filtrage progressif des décisions dans la mesure où elle permet de diminuer au fur et à mesure l'ambiguïté sur la classe proposée. Cela permet généralement de diminuer le taux d'erreur globale de la chaîne de reconnaissance. Néanmoins, une combinaison de ce type demeure particulièrement sensible à l'ordre dans lequel sont placés les classifieurs. En effet, même s'ils ne nécessitent pas d'être les plus performants, les premiers classifieurs invoqués doivent être robustes, c'est-à-dire que la solution réelle de la forme à identifier doit apparaître dans les listes successives quelle que soit leur taille. En cas de mauvaise décision du premier classifieur, placé en amont de la série des classifieurs utilisés, l'erreur va se propager de façon irrévocable. Il faudra donc choisir judicieusement le premier classifieur afin d'éviter - autant que possible - l'apparition d'une telle situation. La combinaison séquentielle suppose donc une certaine connaissance a priori du comportement de chacun des classifieurs. Notons que dans cette approche, chaque classifieur est réglé en fonction du classifieur placé en amont de la chaîne. Une simple modification du premier classifieur peut provoquer un re-paramétrage (ré-apprentissage) des classifieurs suivants.

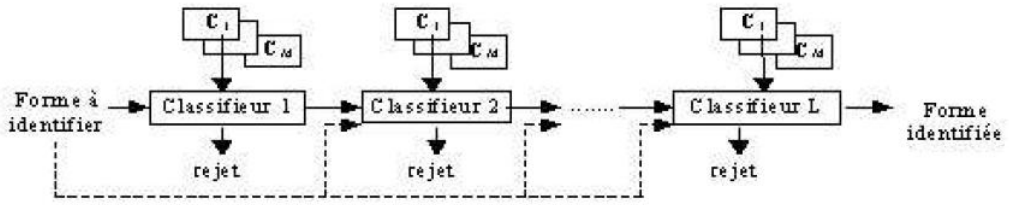


FIG. 3.1: Combinaison séquentielle de classifieurs

Le but ici n'est pas de décrire en détail les systèmes de combinaison séquentielle. Nous présentons maintenant deux exemples qui illustrent leur principe de fonctionnement. On pourra se référer à [Rhaman and Fairhust, 2003] pour un panorama assez complet de ces approches. L'approche proposée dans [Gader et al., 1991] est basée sur trois étages de décision. Les deux premiers étages mettent en oeuvre une comparaison directe du caractère à reconnaître avec l'ensemble des modèles, permettent de classer 70 à 80% des chiffres avec un taux d'erreur faible et sont capables de générer des décisions sur les classes d'appartenance des chiffres rejetés. Lorsque ces étages ne peuvent pas conclure, ils fournissent une liste d'hypothèses au dernier niveau de décision pour chercher le modèle dans une liste prédéfinie de modèles syntaxiques. Dans [Prevost et al., 2003] est présenté un système composé de deux étages pour améliorer la reconnaissance de caractères manuscrits. Le premier étage est un classifieur non supervisé qui fournit des scores à chacune des classes. Le second étage est un classifieur neuronal qui sépare les paires de classes les plus ambigues. Ce système séquentiel est basée sur l'idée que la classe correcte est systématiquement parmi les deux premières classes (celles ayant les probabilités les plus élevées) proposées par le premier classifieur. Les résultats expérimentaux montrent une amélioration de 30% par rapport à chacun des classifieurs utilisés pour une réponse de type classe dans un problème à 62 classes.

3.1.2 Approche parallèle

A la différence de l'approche séquentielle, l'approche parallèle laisse dans un premier temps les différents classifieurs opérer indépendamment les uns des autres puis fusionne leurs réponses respectives. Cette fusion est faite soit de manière démocratique, dans le sens où elle ne favorise aucun classifieur par rapport à un autre, soit au contraire dirigée et, dans ce cas, on attribue à la réponse de chaque classifieur un poids en fonction de ses performances. L'ordre d'exécution des classifieurs n'intervient pas dans cette approche. La figure 3.2 fournit une représentation de la combinaison parallèle de classifieurs.

L'inconvénient majeur de l'approche parallèle est qu'elle nécessite l'activation de tous les classifieurs du système qui doivent participer de manière concurrente et indépendante. Par contre, la décision finale est prise avec le maximum

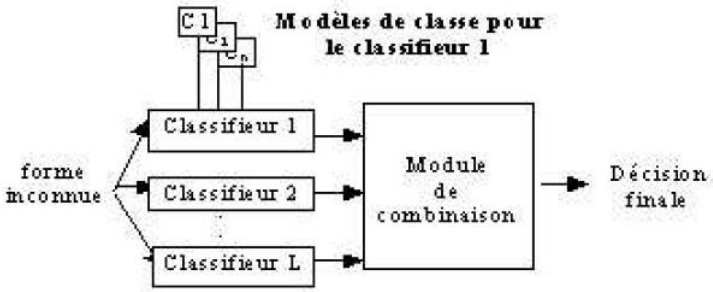


FIG. 3.2: Combinaison parallèle de classifieurs

de connaissances mises à disposition par chaque classifieur. Dès lors se posent les problèmes de précision des informations fournies par les classifieurs et de la confiance qu'on peut accorder à chacun d'eux. L'approche proposée dans [Huang et al., 1995] nécessite que chacun des classifieurs fournisse une confiance (probabilité ou distance) associée à chaque proposition ou classe. La décision finale est prise dans un réseau de neurones à partir de la combinaison des différents résultats fournis par les classifieurs. Pour améliorer la reconnaissance de mots, [Kim et al., 2000] proposent de combiner deux classifieurs, l'un de type HMM (Hidden Markov Model), l'autre de type MLP (Multi-Layer Perceptron). L'idée ici est que pour augmenter la complémentarité, les classifieurs doivent opérer avec des structures différentes. Les sorties du classifieur HMM sont normalisées avant la combinaison pour pouvoir les fusionner avec les sorties du MLP.

3.1.3 Approche hybride

L'approche hybride consiste à combiner à la fois des architectures séquentielles et parallèles afin de tirer pleinement avantage de chacun des classifieurs utilisés. La figure 3.3 présente un exemple de combinaison hybride dans laquelle on combine un classifieur en série avec deux classifieurs en parallèle. Ce type d'approche permet de générer de nombreux schémas de coopération qui peuvent rapidement devenir complexes à optimiser. Il illustre les deux aspects de la combinaison qui sont d'une part la réduction de l'ensemble des classes possibles et d'autres part la recherche d'un consensus entre les classifieurs afin d'aboutir à une décision unique.

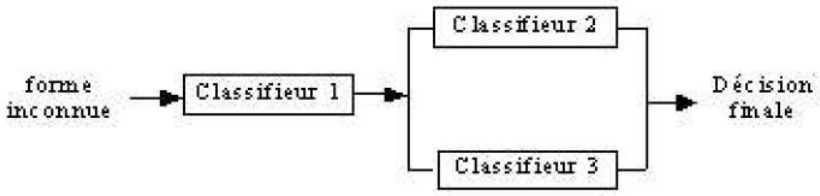


FIG. 3.3: Combinaison hybride de classifieurs

On peut citer dans ce cadre, les travaux de [Kim et al., 2000] qui propose un système de reconnaissance de mots cursifs anglais extraits des chèques bancaires. Ce système hybride est composé de deux étages. Dans le premier étage, deux classifieurs (PMC) utilisant des vecteurs de caractéristiques différents sont combinés par un autre classifieur de même type. La coopération de ce dernier avec un autre classifieur de type HMM est réalisée dans l'étage suivant par une règle de multiplication. Un autre exemple est celui présenté dans [Bellili et al., 2002]. Il décrit un système de reconnaissance de chiffres manuscrits par combinaison hybride de réseaux neuronaux de type MLP et de machines à vecteurs de support SVM. Cette méthode de combinaison consiste à introduire des classifieurs SVM spécialisés pour chaque paire de classes numériques (0 à 9) uniquement dans le voisinage des surfaces de séparation générées par le réseau MLP entre les exemples d'apprentissage de ces mêmes paires de classes. Cette architecture de combinaison est fondée sur la constatation que les deux premières solutions de la couche de sortie du MLP contiennent presque systématiquement la bonne classe de la forme à classifier et que certaines paires de classes constituent la majorité des confusions générées par le MLP. Les SVM sont introduits pour détecter la bonne classe parmi les deux meilleures hypothèses de classification fournies par le réseau. Ce choix se résume à un problème de classification à deux classes (binaire). Cependant, cette méthode peut sembler fastidieuse car elle nécessite un classifieur SVM pour chaque paire de classes. Une seconde originalité de cette méthode réside dans l'introduction de SVM uniquement pour les paires de classes qui constituent la majorité des confusions (erreurs) du réseau MLP. Certains auteurs ont proposé d'effectuer des combinaisons conditionnelles. Ainsi [Gosselin, 1997] propose de classer les classifieurs selon leur performance et de traiter une forme inconnue par le premier classifieur. Il propose d'accepter sa décision, si la forme n'est pas rejetée. Dans le cas contraire, la décision sera

prise suite à la combinaison du premier classifieur avec la sortie du deuxième classifieur. Le même raisonnement peut s'appliquer, jusqu'à ce que la forme soit classée ou que les sorties de tous les classifieurs soient combinées. Cette combinaison conditionnelle permet de réduire efficacement les temps d'exécution. L'inconvénient est la nécessité de fixer plusieurs seuils de rejet associés aux différents niveaux.

De nombreux travaux montrent que la combinaison de classifieurs (séquentielle, parallèle ou hybride) améliore nettement les performances du système de reconnaissance par rapport à chacun des classifieurs pris isolément. Cependant, parmi ces différentes architectures permettant de combiner un ensemble de classifieurs donnés, l'architecture parallèle est de loin celle qui a donné lieu aux travaux les plus importants. Sa simplicité de mise en oeuvre, sa capacité à exploiter les réponses des classifieurs à combiner en prenant en compte (ou non) le comportement de chacun des classifieurs et son efficacité prouvée dans de nombreux problèmes de classification expliquent son succès notamment sur l'approche séquentielle pour laquelle la connaissance du comportement de chaque classifieur est nécessaire a priori pour pouvoir obtenir un schéma de coopération efficace. L'intérêt porté par les chercheurs majoritairement à la combinaison parallèle de classifieurs est fondé pour plusieurs raisons :

- le concepteur peut ré-utiliser les développements de classifieurs effectués antérieurement, chacun pouvant avoir été développé dans un contexte différent et utiliser une représentation différente pour le même problème. Un exemple est l'identification de personnes par leur voix, leur visage ainsi que par leur signature.
- dans la combinaison, il est possible d'utiliser un grand nombre de caractéristiques mais en les distribuant sur des classifieurs différents.
- deux classifieurs différents peuvent présenter des performances globales équivalentes mais avoir leurs propres régions dans l'espace de caractéristiques où ils sont les plus performants.
- un classifieur est souvent sensible aux choix initiaux de ses paramètres (k et distance pour un k-ppv, nombre de couches et de neurones par couche pour un MLP, ...). Plutôt que de chercher la meilleure configuration de pa-

ramètres, la combinaison de l'ensemble peut tenir compte des avantages de ces classifieurs appris différemment.
– on peut avoir à notre disposition plusieurs bases d'apprentissage, chacune est collectée de manière différente ou construite dans des conditions différentes. L'apprentissage d'un même classifieur sur ces bases peut produire des résultats différents.

Ce sont ces avantages qui nous ont conduit à focaliser notre travail sur la combinaison parallèle. Pour aller plus loin, la section suivante est consacrée à la sélection des classifieurs, ce problème étant souvent évoqué comme une optimisation de l'espace de décisions des classifieurs.

Nous venons de présenter des méthodes qui permettent de combiner les classifieurs d'un ensemble. De nombreuses recherches ont montré que pour obtenir les meilleures performances en combinaison, il n'est pas forcément nécessaire d'utiliser tous les classifieurs disponibles. En effet, sélectionner les classifieurs ,selon un critère, avant de les combiner peut donner de meilleures performances globales pour le système. C'est pour cette raison que nous allons faire une étude bibliographique sur les méthodes de sélection de classifieurs dans la section suivante.

3.2 Méthodes de sélection de classifieurs

Dans ce qui va suivre, nous allons nous concentrer sur la sélection statique de classifieurs. Cela va nous permettre de faire une étude bibliographique sur les méthodes de sélection statiques de caractéristiques qui est plus riche que la sélection de statique classifieurs. En effet, la sélection de caractéristiques utilise des méthodes qui peuvent facilement être adaptées à la sélection de classifieurs. Pour faire de la sélection de caractéristique il faut d'abord choisir une méthode de sélection puis une technique de recherche. Dans la littérature plusieurs types d'approche ont été mises en place, qui sont le wrapper et le filter [Kohavi and John, 1997].

3.2.1 Wrapper

Il s'agit d'une approche qui teste différents sous-ensembles de jeux de caractéristiques et qui choisit le sous-ensemble donnant les meilleurs performances. C'est-à-dire que l'on va lancer plusieurs fois le système avec différentes combinaisons de caractéristiques pour conserver au final la meilleure solution [Kohavi and John, 1997]. Dans les méthodes de type wrapper le critère final utilisé pour trouver le bon sous-ensemble de caractéristique est la performance du sous-ensemble. C'est-à-dire qu'elle va sélectionner le sous-ensemble nous donnant les meilleures performances par rapport au type de combinaison utilisé. L'inconvénient majeur de cette méthode est son coût puisque l'on doit relancer plusieurs fois de suite l'algorithme d'apprentissage, étudier toutes les possibilités et retenir la meilleure. De plus ce système doit être relancé à chaque fois que l'on modifiera le type de combinaison. Dans la figure 3.4, on choisit le meilleur

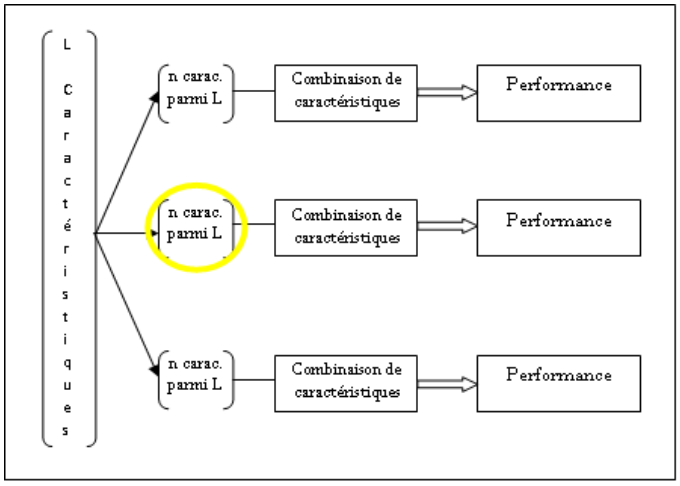


FIG. 3.4: Exemple pour la méthode Wrapper

sous-ensemble de n caractéristiques parmi les L disponibles, par rapport au taux de reconnaissance de chacun des sous-ensembles. Dans notre exemple, c'est le 2^{eme} sous-ensemble qui sera sélectionné.

3.2.2 Filter

Cette approche contrairement au wrapper est une approche moins gourmande en temps de calcul. Le principe est de sélectionner le sous-ensemble de caracté-

ristiques selon un certain critère, comme la corrélation entre les caractéristiques par exemple. On retient ensuite le sous-ensemble qui optimise ce critère pour toutes les formes à identifier [Kohavi and John, 1997]. Contrairement au wrapper, on ne choisit pas le sous-ensemble par rapport aux performances obtenues lors de la combinaison, il n'est donc pas nécessaire de tester plusieurs combinaisons pour connaitre la meilleure. La sélection ce fait à l'étape précédente et le choix du critère est indépendant du type de classifieur utilisé. Comme on peut le

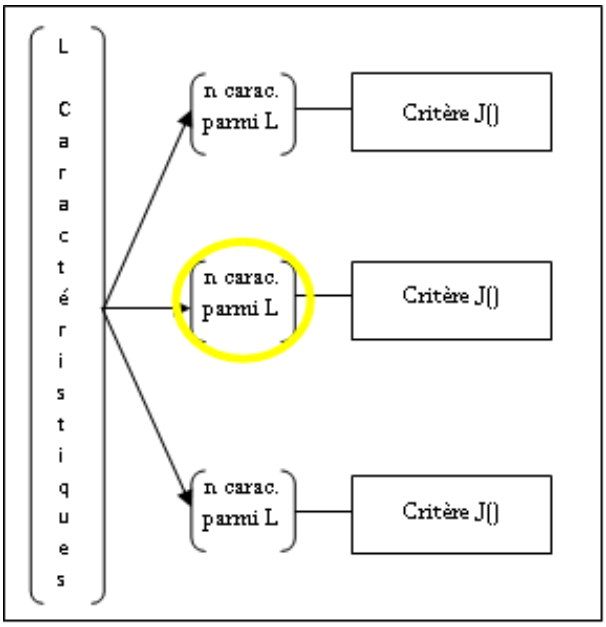


FIG. 3.5: Exemple pour la méthode Filter

voir sur la figure 3.5, nous avons choisi dans cet exemple le 2^{eme} ensemble de n caractéristiques, car celui-ci permet d'optimiser notre critère J(). Pour mettre en place la méthode du filter il faut au préalable avoir choisi un critère, pour cela plusieurs critères ont déjà été proposés dans la littérature.

Critère pour la méthode filter

Le choix du critère pour la méthode filter est très souvent rattaché à la variance S_B (inter-classe) et S_W (intra-classe). La variance appelée inter-classe S_B (between-class) est la variance qui permet de mesurer la distance entre la moyenne de chaque classe. Plus sa valeur est grande mieux c'est, [Webb, 2002].

Voici la formule correspondante :

$$S_B = \sum_{i=1}^{c} \frac{n_i}{n}(m_i - m)(m_i - m)^T \tag{3.2.4}$$

Avec n qui correspond au nombre de données et n_i au nombre de données de la classe i, et où m_i est obtenue à l'aide de la formule suivante :

$$m_i = \frac{1}{n_i} \sum_{j=1}^{n} Z_{ij} x_j \tag{3.2.5}$$

Où Z_{ij} vaut 1 si la caractéristique x_j appartient à la classe w_i. Et on obtient pour m, avec c correspondant au nombre de classe :

$$m = \sum_{i=1}^{c} \frac{n_i}{n} m_i \tag{3.2.6}$$

Or ce critère n'est pas suffisant pour faire une bonne sélection. Pour avoir un bon critère de sélection il faut que nous ayons une distance entre les classes (Sb) élevées et une valeur intra-classe faible (within-class, Sw). La variance appelée intra-class, S_W (within-class), est la varaiance qui permet de mesurer la distance entre chaque élément de la classe et sa moyenne. Plus sa valeur est petite, meilleur ce sera. Elle est donnée par la formule suivante :

$$S_W = \sum_{i=1}^{c} \frac{n_i}{n} \hat{\Sigma}_i \tag{3.2.7}$$

Avec :

$$\hat{\Sigma}_i = \frac{1}{n_i} \sum_{j=1}^{n} Z_{ij}(x_j - m_i)(x_j - m_i)^T \tag{3.2.8}$$

Qui correspond à la matrice de covariance de la classe w_i. Il existe plusieurs façons de trouver un critère avec ses deux données. La plus populaire est la suivante :

$$J_1 = Tr\{Sw + Sb\} = Tr\left\{\hat{\Sigma}\right\} \tag{3.2.9}$$

Dans cette formule on fait la trace (Tr) de la somme de Sw et Sb.
Avec :

$$\hat{\Sigma} = \frac{1}{n}\sum_{j=1}^{n}(x_j - m)(x_j - m)^T \tag{3.2.10}$$

Pour avoir de bons résultats, il faut que le résultat de l'équation J_1 soit de l'ordre de Sb. Ce critère est simple à calculer mais ne nous donnera pas forcément les meilleurs résultats. Il est possible aussi de combiner ses 2 valeurs autrement, nous permettant d'obtenir un critère plus intéressant pour notre sélection.
Voici différents critères qui peuvent être utilisés :

$$J_2 = Tr\left\{Sw^{-1}Sb\right\} \tag{3.2.11}$$

Pour avoir un bon critère, il faut que J2 soit maximisé.

$$J_3 = \frac{|\hat{\Sigma}|}{|Sw|} \tag{3.2.12}$$

Dans ce cas là aussi, il faut que J3 soit maximisé pour avoir de bonnes performances.

$$J_4 = \frac{Tr\{Sb\}}{Tr\{Sw\}} \tag{3.2.13}$$

Et pour finir dans ce cas là, il faut que J4 soit minimisé pour avoir de bonnes performances.

Nous venons de voir les deux principales méthodes qui organisent la sélection de classifieurs. Nous allons dans la section suivante parler des algorithmes qui permettent de mettre en place la sélection avec les avantages et les inconvénients de chacun.

3.3 Technique de recherche statiques

Le schéma de la figure 3.6 nous montre que les méthodes de sélection de caractéristiques sont divisées en 3 types.
Nous avons les méthodes complètes, c'est-à-dire optimale, qui sont divisées en deux groupes, les méthodes exhaustives et non exhaustives. Les méthodes

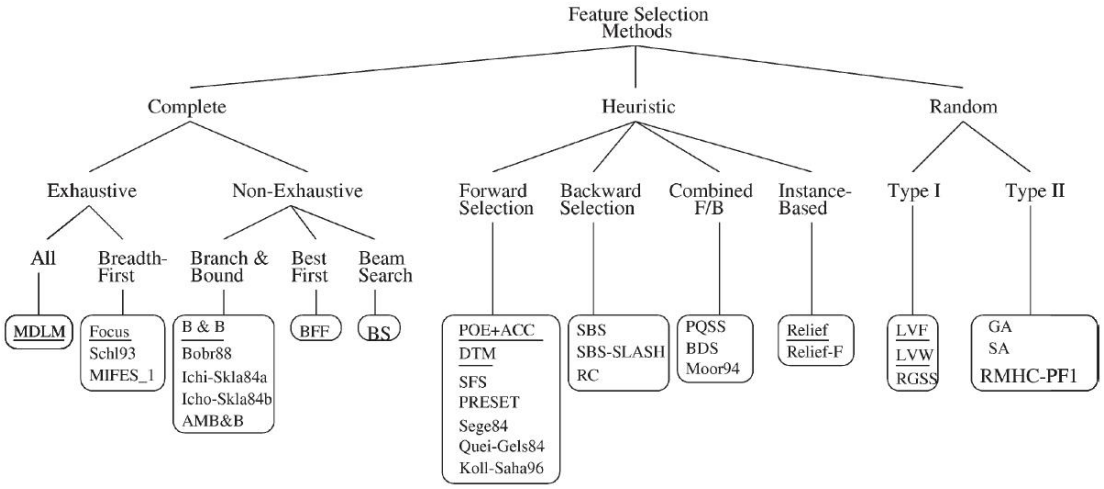


FIG. 3.6: Résumé des méthodes de sélection de caractéristiques.[Dash and Liu, 1997]

exhaustives correspondent aux méthodes qui énumèrent toutes les combinaisons possibles des différentes caractéristiques, pour ensuite choisir la meilleure combinaison. Avec ces méthodes on évalue les 2^N sous-ensembles, donc toutes les possibilités. Ensuite nous avons les méthodes non-exhaustives et optimales qui sont divisées en trois groupes. Pour commencer nous avons les méthodes "Branch and Bound" que nous expliquerons plus en détails un peu plus tard. Ensuite nous avons la méthode "Best-First" et "Beam-Search", qui est une amélioration de l'algorithme "Best-First".

Le deuxième groupe de méthodes correspond aux méthodes heuristiques, c'est-à-dire qu'il s'agit de méthodes nous donnant des résultats rapidement, mais qui ne sont pas forcément optimales. Il s'agit de méthodes approximatives. Dans cette partie nous trouvons le groupe "Forward Selection" qui correspond aux méthodes de recherche en avant, comme par exemple le SFS (Search Forward Selection) que nous expliquerons plus en détail dans la prochaine partie. Nous avons aussi les méthodes "Backward Selection" qui sont les méthodes de recherche en arrière comme le SBS (Search Backward selection), que nous expliquerons aussi par la suite. Ensuite dans les méthodes heuristiques nous avons aussi le groupe "Combined F/B", qui comme son nom l'indique, combine à la fois la recherche en avant et la recherche en arrière, comme par exemple le PTA(l,r) (Plus l Take Away r). Et enfin le dernier groupe "Instance Based" qui contient l'al-

gorithme "relief" dont le principe est de donner un score à chaque attribut basé sur sa capacité à bien classer ou non les plus proches voisins de chaque exemple [Kira and Rendell, 1992].

Enfin, il y a les méthodes dites "Random", qui sont divisées en deux groupes. Le premier groupe appelé type I correspond aux algorithmes qui crée de façon complètement aléatoire les sous-ensembles, comme par exemple les algorithmes LVW (Las Vegas Wrapper) et l'algorithme RGSS (Rapid Gravity Survey Systems) qui introduit de l'aléatoire dans les algorithmes SFS et SBS. Dans les algorithmes de type II, la notion d'aléatoire est aussi présente, mais biaisée, car chaque sous-ensemble est pondéré par son importance par rapport au problème, ce que l'on verra pour les algorithmes génétiques en particulier, dans la prochaine partie.

A partir de cette vue d'ensemble des techniques, nous allons faire une bibliographie de certaines de ces méthodes et les comparer entre elles afin de trouver les plus efficaces.

3.3.1 SBS (Sequential Backward Selection)

Le SBS est un processus de "recherche en arrière" des caractéristiques permettant d'obtenir le meilleur sous-ensemble. A l'initialisation nous avons un ensemble de caractéristiques. Celles-ci sont éliminées de manière itérative. A chaque itération, une caractéristique est éliminée de telle sorte que l'ensemble avec les caractéristiques restantes donne la performance la plus élevée [Hao et al., 2003]. Ce processus est répété jusqu'à ce que les performances cessent de converger ou si on désire arrêter le système au bout d'un certain nombre d'itération [Cardoso et al., 2000].

La figure 3.7, présente un ensemble de départ composé de 4 caractéristiques. A chaque itération on testes toute les combinaisons en supprimant une caractéristique différente dans chaque cas, puis on garde la combinaison nous donnant les meilleures performances. On obtient, alors, des meilleures performances lorsque l'on supprime C2, donc on conserve la combinaison C1, C3, C4. Et pour l'itération suivante nous obtenons de meilleures performances sans C4, donc le

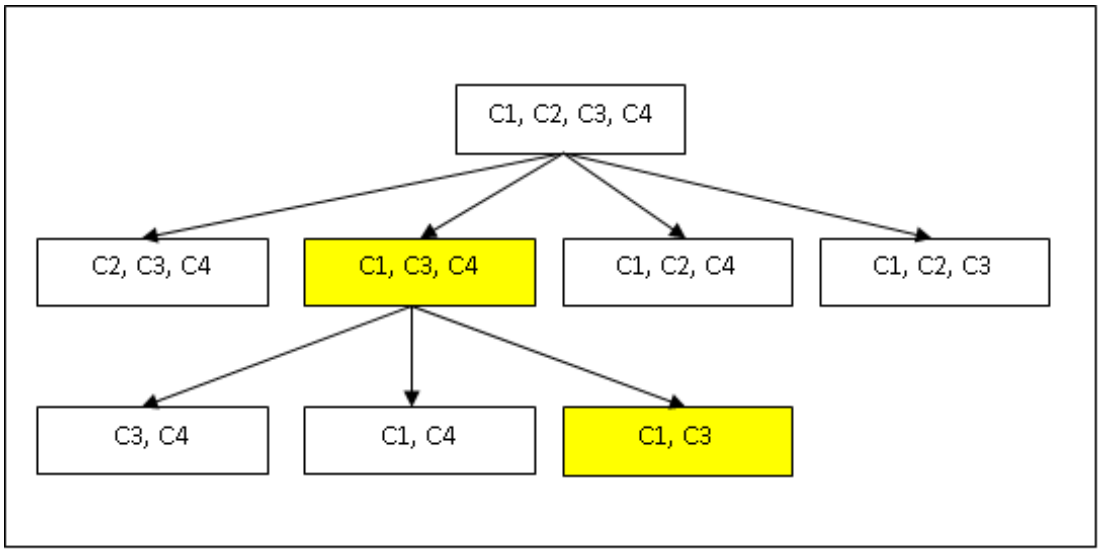


FIG. 3.7: Exemple pour le SBS

meilleur sous-ensemble est C1, C3.

3.3.2 SFS (Sequential Forward Selection)

Il s'agit d'un processus de "recherche en avant" des caractéristiques nous permettant d'obtenir le meilleur sous-ensemble. Dans cette méthode, le sous ensemble initial est vide. Les caractéristiques sont alors ajoutées une par une. A chaque étape, on ajoute à notre nouvel ensemble de caractéristiques, la caractéristique nous permettant d'obtenir les meilleures performances lors de la combinaison de celle-ci avec notre sous-ensemble final [Hao et al., 2003].
Comme pour le SBS la construction d'un nouvel ensemble de caractéristique s'arrêtera lorsque nous aurons obtenu un sous-ensemble final de taille définie. La caractéristique initiale est souvent choisie aléatoirement [Roli et al., 2001], la combinaison finale de nos caractéristiques peut alors être différente selon la caractéristique de départ. Il est aussi possible de choisir comme caractéristique de départ celle ayant les meilleures performances par rapport aux autres.

Dans l'exemple de la figure 3.8, on peut voir qu'au départ on part d'un ensemble vide et on sélectionne une première caractéristique (dans notre cas il s'agit de la caractéristique ayant les meilleures performances). Ensuite à chaque

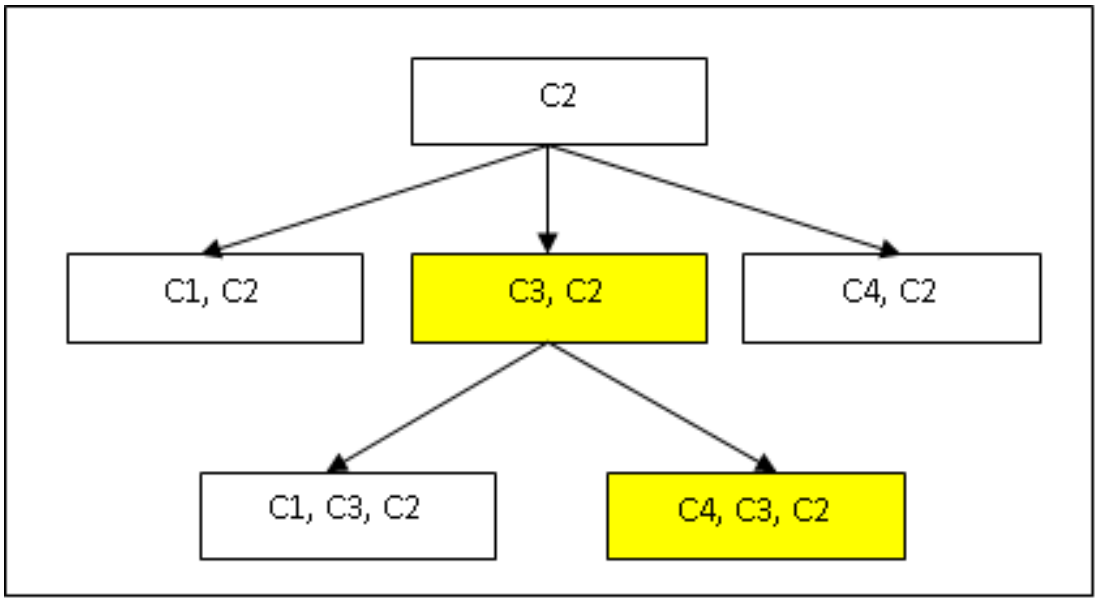


FIG. 3.8: Exemple pour le SFS

itération on ajoute une caractéristique au sous-ensemble, puis on conserve le sous-ensemble donnant les meilleurs résultats, dans notre cas C3, C2. Et pour l'itération suivante nous avons comme sous-ensemble donnant les meilleurs résultats C4, C3, C2.

3.3.3 GSFS(g) (generalized sequential forward selection) et GSBS(g) (generalized sequential backward selection)

Il s'agit des méthodes généralisées de SBS et SFS, c'est à dire, pour GSFS(g), au lieu de sélectionner à chaque itération la caractéristique permettant d'avoir les meilleures performances, on sélectionne à chaque itération, les g caractéristiques permettant d'obtenir les meilleures performances pour notre ensemble final. Cette méthode est plus optimale, car contrairement au SFS, les performances de l'ensemble final ne seront pas autant influencées par le choix de la caractéristique sélectionnée à l'itération précédente. En effet, à chaque itération on choisit un sous-ensemble de g caractéristiques. Cette méthode a un temps d'exécution très long [Hao et al., 2003].

La figure 3.9, nous fait remarquer qu'il s'agit du même principe que le SFS,

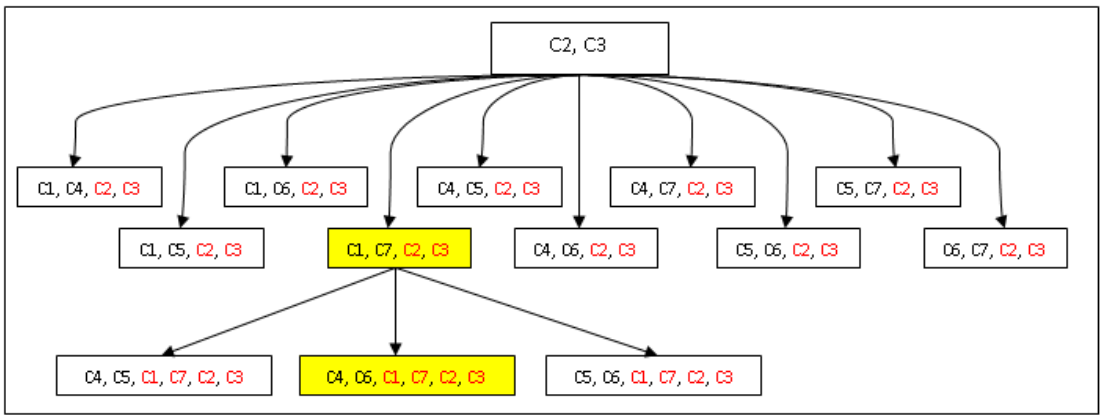


FIG. 3.9: Exemple pour le GSFS

sauf qu'au lieu de sélectionner à chaque itération la caractéristique nous permettant d'avoir les meilleures performances, on sélectionne g caractéristiques. On peut voir qu'à l'initialisation nous avons un ensemble vide et dans notre cas on décide de choisir le meilleur sous-ensemble de g caractéristiques (ici g=2), soit dans notre exemple C2 et C3. Ensuite nous sélectionnons le meilleur sous-ensemble contenant C2 et C3 ainsi que 2 autres caractéristiques : il s'agit dans notre exemple du sous-ensemble contenant C1, C6, C2 et C3. Puis de même pour la dernière itération, ce qui nous donne le sous-ensemble C4, C6, C1, C7, C2, C3.

3.3.4 PTA(l,r) (Plus l take away r)

Il s'agit d'un processus qui combine le SFS et le SBS. Il faut savoir qu'avec le SFS (ou SBS) on ajoutait (ou supprimait), la meilleure (ou moins bonne) caractéristique, ce qui influe les performances de notre sous-ensemble final, par rapport à la première caractéristique choisie [Pudil et al., 1994] La méthode du PTA(l,r) consiste, à chaque itération, à appliquer le processus du SFS l fois puis le processus du SBS r fois. Quand l>r, le sous-ensemble choisi augmente (forward), sinon il diminue (backward) [Zongker and Jain, 1996].

Dans l'exemple figure 3.10, on peut voir que l'on fait un SFS pendant 3 itérations (l=3) et un SBS sur le sous-ensemble obtenue du SFS (C1, C2, C3, C4) pendant 2 itérations (r=2). Cet exemple représente une seule itération, donc cette

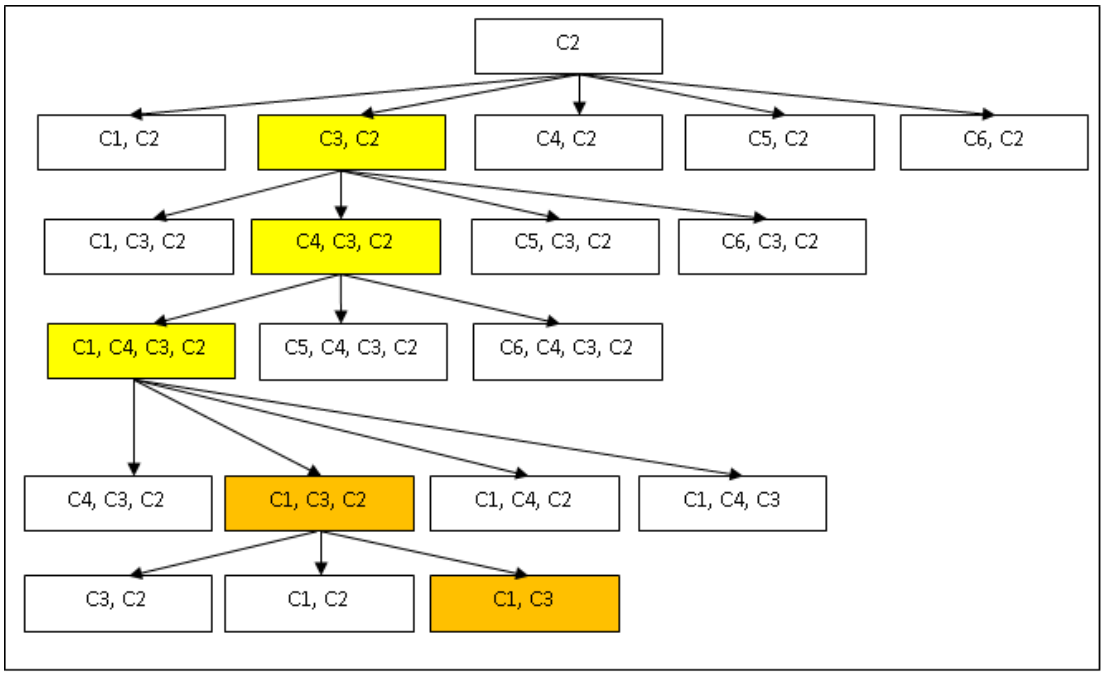


FIG. 3.10: Exemple pour le PTA(l,r)

opération est répétée jusqu'à ce que le critère d'arrêt soit atteint (même critère d'arrêt que le SFS et SBS).

3.3.5 GPTA(l,r) (generalized plus l take away r)

Il s'agit du processus généralisé du PTA(l,r), plus optimal que le PTA(l,r). C'est-à-dire que l'on utilise dans ce processus les méthodes GSFS et GSBS. A chaque itération on va directement ajouter le sous-ensemble de l caractéristiques nous donnant les meilleures performances, puis supprimer le sous-ensemble de r caractéristiques. Contrairement au PTA(l,r), où l'on ajoutait et supprimait séquentiellement, il s'agit aussi dans ce cas là d'une méthode plus optimale que le PTA(l,r) mais très couteuse en temps de calcul, donc très peu réalisable [Pudil et al., 1994], [Hao et al., 2003].

3.3.6 SFFS (Sequential Floating Forward Selection) et SFBS (Sequential Floating Backward Selection)

SFFS et SFBS combinent SFS et SBS de façon plus souple que PTA (l, r). Dans ces processus le nombre successif d'ajouts et de suppressions de caractéristiques n'est pas fixé. Dans SFFS, on ajoute une caractéristique au sous-ensemble choisi grâce au processus du SFS, puis on regarde si le fait de supprimer une caractéristique à l'aide du processus du SBS améliore les performances, si ce n'est pas le cas, aucune caractéristique n'est alors supprimer [Pudil et al., 1994], [Hao et al., 2003].

3.3.7 Branch and bound (B&B)

La méthode Branch and Bound (procédure par évaluation et séparation progressive) est un algorithme qui construit l'arborescence des sous-ensembles de caractéristiques en évaluant a priori les chances de trouver la solution optimale dans une branche particulière.

Cette méthode consiste à énumérer les solutions en utilisant certaines propriétés du problème. Elle permet d'éliminer a priori des solutions partielles qui ne mènent pas à la solution que l'on recherche, ce qui évite de faire une recherche exhaustive et donc permet d'obtenir des résultats en un temps raisonnable.
Pour appliquer la méthode de B&B, nous devons avoir ,[Rebaine, 2005] :

- Un moyen de calcul d'une borne inférieure d'une solution partielle
- Une stratégie de subdiviser l'espace de recherche pour créer des espaces de recherche de plus en plus petits.
- Un moyen de calcul d'une borne supérieure pour au moins une solution.

A la racine de notre arborescence nous avons notre ensemble de caractéristiques. Ensuite des procédures de calculs de bornes inférieures et supérieures sont appliquées à la racine. Lorsque les 2 bornes sont égales, cela veut dire que nous avons trouvé la meilleure solution. Sinon on divise l'ensemble des caractéristiques en plusieurs sous-ensembles. Lorsque l'on trouve une solution avec de très bons résultats, cela ne veut pas forcément dire que le résultat est optimal, car il est possible de trouver encore un meilleur résultat dans une autre branche. Par contre, ce résultat nous permettra de supprimer les branches ayant un moins bon

résultat. Car si la borne inférieure d'un nœud dépasse la valeur d'une solution déjà connue, alors on peut affirmer que la solution optimale globale n'est pas présente dans ce nœud. Le système s'arrête lorsque tous les nœuds sont explorés ou éliminés.

De plus cet algorithme est efficace car il choisit le meilleur sous-ensemble sans recherche exhaustive. Par exemple, on cherche le meilleur jeu de 12 caractéristiques dans un jeu de 24 caractéristiques. Avec cette méthode seulement 6000 sous-ensembles ont été évalués sur les 2704156 sous-ensembles qu'une recherche exhaustive exigerait d'évaluer [Narendra and Fukunaga, 1977]. La méthode B&B est équivalente à une méthode de recherche exhaustive quand le critère est monotone [Sklansky, 2000].

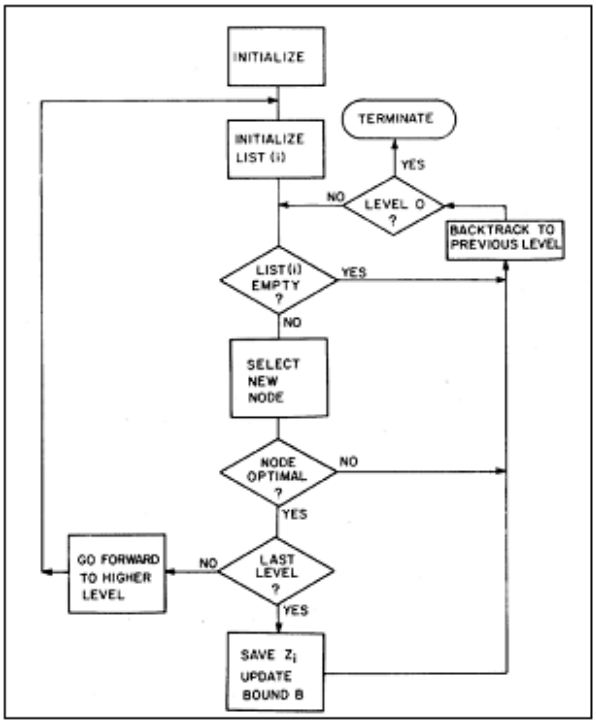


FIG. 3.11: Organigramme pour l'algorithme de Branch and Bound [Narendra and Fukunaga, 1977]

3.3.8 Algorithme génétique

Les algorithmes génétiques (AG), ont été développés dans les années 1970, comme une méthode d'optimisation efficace. Il existe un grand nombre de variétés d'AG, mais les principes de bases restent inchangés. Beaucoup d'études ont été faites sur les algorithmes génétiques pour la sélection de caractéristiques.

Dans un AG [Sklansky, 2000], on a une population de base qui est souvent composée de chaînes de caractères correspondant chacune à un chromosome. Souvent chaque chromosome est une chaine binaire de taille n. La population initiale est générée aléatoirement.
Les mécanismes d'un algorithme génétique de base sont assez simple [Goldberg, 1991]. Il s'agit de faire des copies de chaînes et des échanges de morceaux de chaînes. Il est composé de 3 étapes, qui conduisent généralement à de bons résultats : La reproduction, le crossover et la mutation. La reproduction correspond à la copie de chaque chaîne en fonction des valeurs de la fonction à optimiser. Ce qui correspond à donner un poids d'importance à chaque chaîne. Les chaînes ayant les meilleurs scores d'adaptation auront un poids plus important. Il s'agit donc d'un système de sélection appelé "roue de loterie biaisé". Cette roue est une roue classique sur laquelle chaque individu est représenté par une portion proportionnelle à son adaptation. Ensuite nous effectuons un tirage au sort homogène sur cette roue. Après la reproduction, le crossover est appliqué. Il correspond à un croisement des différentes chaînes tirées lors de la reproduction. Le crossover se fait en deux étapes. Pour commencer les nouveaux éléments produits par la reproduction sont appariés au hasard, puis chaque paire de chaînes subit un crossover. C'est-à-dire que l'on choisit une valeur k aléatoirement qui est comprise entre 1 et la longueur de la chaîne. Cette valeur correspond au nombre d'éléments de la chaîne qui seront échangés entre les paires de chaînes.
Pour finir la dernière étape est la mutation. Il s'agit d'une étape encore très floue en génétique, et qui joue un rôle secondaire dans les algorithmes génétiques [Goldberg, 1991]. Mais c'est une étape nécessaire car même si les étapes de reproduction et de crossover explorent et combinent efficacement les notions existantes, celles-ci peuvent parfois devenir trop zélées et perdre de la matière génétique potentiellement utile. Donc la mutation permet de modifier aléatoirement un élément de chaîne. C'est une modification qui n'apparait qu'occasionnelle-

ment. Ces étapes sont effectuées jusqu'à ce que l'on obtienne des résultats qui n'évoluent plus, [Rebaine, 2005]. La figure 3.12 présentes les différentes étapes.

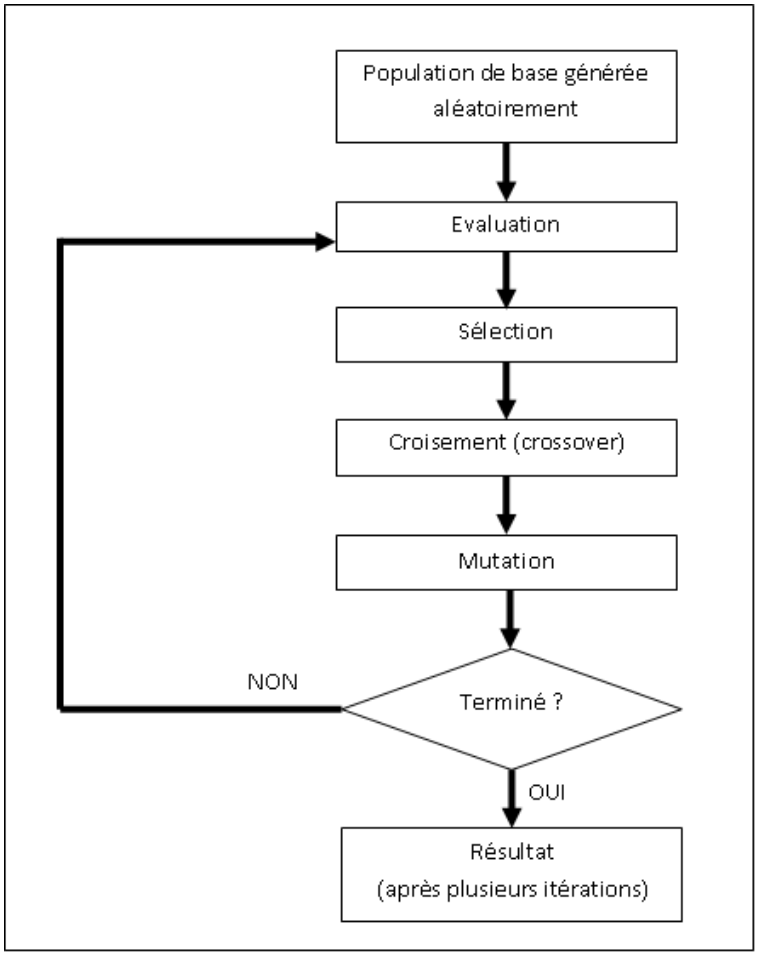


FIG. 3.12: Diagramme de l'agorithme génétique

3.3.9 Récapitulatif et comparaison des différentes méthodes

Dans le tableau 3.1, nous avons plusieurs types d'objectif. L'objectif de type A correspondant aux algorithmes qui cherchent le meilleur sous-ensemble pour un nombre de caractéristiques donné. Le type C correspond aux algorithmes qui cherchent un sous-ensemble avec une taille minimum et un taux d'erreur minimum. Et le type de recherche de type S correspond à une recherche séquentielle

Algorithme	Temps d'exécution	Type d'objectif	Type de recherche	Optimal
SFS, SBS	$\Theta(n^2)$	A	S	NO
GSFS(g), GSBS(g)	$\Theta(n^{g+1})$	A	S	NO
PTA(l,r)	$\Theta(n^2)$	A	S	NO
GPTA(l,r)	$\Theta(n^{max\{l+1,r+1\}})$	A	S	NO
SFFS, SBFS	$O(n^2)$	A	S	NO
Branch and Bound	$O(n^2)$	A	S	YES
GA	$\Theta(1)(\Theta(n))$	A et C	P	NO

TAB. 3.1: Récapitulatif des différentes méthodes [Sklansky, 2000]. Avec Θ correspondant à la complexité moyenne et O à la complexité maximale

et P à une recherche parallèle.

Dans la littérature Kudo et Sklansky [Sklansky, 2000] ont effectué plusieurs comparaisons entre ses différentes méthodes et ils en ont déduit que les méthodes SFFS et SBFS donnent de meilleurs résultats que les autres méthodes de sélection dans un temps raisonnable pour les problèmes ne contenants pas un trop grand nombre de données. Les algorithmes génétiques sont très bien adaptés pour les problèmes contenant beaucoup de données et nous donnent de meilleurs résultats que les autres méthodes. Même si les algorithmes évolutionnaires sont plus lent que le SFFS ou SBFS celui-ci nous donne de meilleur résultats sur les problèmes contenant beaucoup de données.

La présentation que nous venons de faire sur les méthodes qui permettent de sélectionner et de combinaison des classifieurs montre que la configuration d'un système de sélection dépend essentiellement de la façon dont on exploite les classifieurs. Vis-à-vis de notre objectif, nous allons orienter nos travaux vers un système de combinaison parallèle du fait des nombreux avantages et travaux déjà effectué dans la littérature. Concernant la sélection, nous avons choisi un système basé sur les algorithmes génétiques (technique de recherche statique) permettant ainsi d'explorer un grand nombre de possibilités sans être obligé de toutes les générer. La section suivante est consacrée à l'application de ces techniques à notre ensemble de classifieurs constituant le Front ROC.

3.4 Application et résultats

Dans cette section, nous présentons l'application de la combinaison de classifieurs au problème lié à la comparaison d'un ensemble discret, le Front ROC, avec des ensembles continus, les courbes ROC, évoqué dans le chapitre 2. Nous allons pour cela utiliser un algorithme génétique pour la sélection des classifieurs puis les combiner avec les méthodes de moyenne et de produit. Afin de valider notre approche de manière statistique, nous allons utiliser une méthode de cross-validation qui va venir en complément de l'algorithme génétique pour généraliser les résultats de notre approche.
Les expérimentations que nous présentons ici s'articulent en deux étapes qui sont la sélection de modèles SVM puis la combinaison des SVM ainsi obtenus. Avant toute chose, nous posons le cadre de ces expérimentations puis nous présenterons les résultats obtenus.

3.4.1 Protocole expérimental

Le protocole expérimental à ce niveau n'est que la continuité de ce que nous avons présenté dans la section 2.3. Pour rappel, nous avons deux séries de test à mener afin de répondre à nos problématiques. La première partie consiste en la sélection de modèles SVM, présenté dans le chapitre 2, la seconde partie consiste en la sélection et la combinaison des classifieurs issus de la première étape. La figure 3.13 schématise l'ensemble du dispositif.
Le travail dans cette partie, consiste à sélectionner puis à combiner les classifieurs obtenus en sortie du module d'optimisation pour la sélection de modèle SVM. Pour la sélection, nous utilisons un algorithme génétique avec un gène binaire de 40 bits où chaque élément du gène représente un classifieur. Nous utilisons une population de 40 individus et ce sur 200 générations. Pour la partie combinaison, nous utilisons deux méthodes en parallèle. Nous combinons les vecteurs de confiance des classifieurs par un opérateur produit d'une part, et par un opérateur moyenne d'autre part. L'optimisation est faite sur la combinaison du meilleur sous-ensemble par rapport au critère de l'AUC. Nous utilisons la base de validation combinaisons pour optimiser la sélection des classifieurs puis les performances de la meilleure population sont évaluées sur la base de test. Le principe de fonctionnement est présenté dans la figure 3.14.

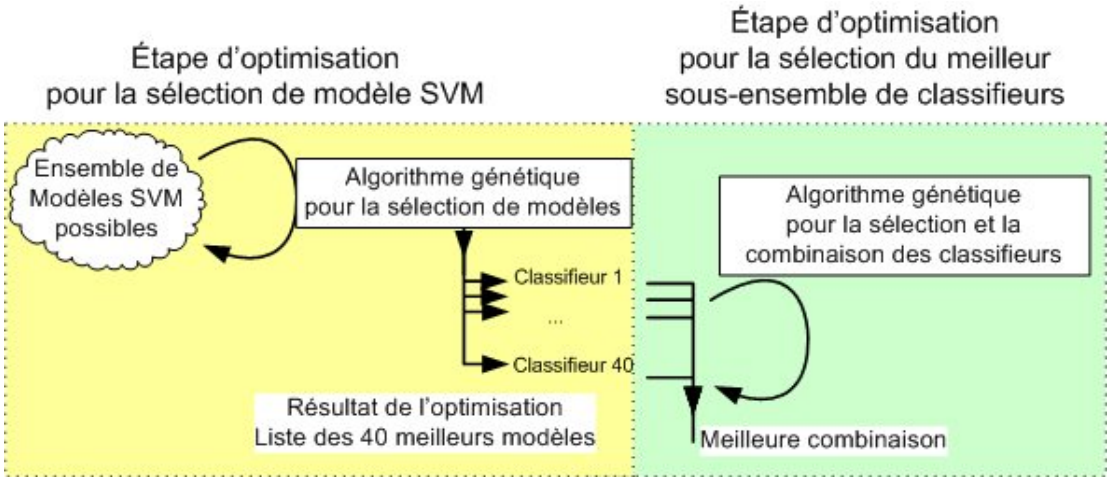


FIG. 3.13: Schéma de principe de l'ensemble du système

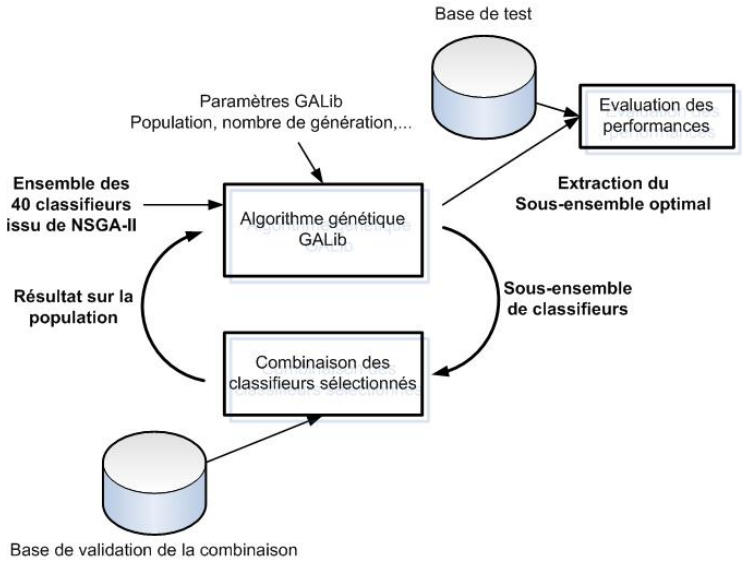


FIG. 3.14: Schéma d'optimisation pour la combinaison de classifieurs

Nous allons dans la partie suivante présenter les résultats obtenus avec notre système sur différentes base de l'UCI Repository. L'idée est non seulement de comparer les performances avant et après combinaison mais également de se comparer aux résultats de la littérature.

Problème	AUC sélection et combinaison		AUC combinaison des 40 classifieurs	
	Moyenne	Produit	Moyenne	Produit
australian	91.73 ± 0.02	91.58 ± 0.02	91.49 ± 0.03	70.35 ± 0.89
wdbc	97.65 ± 0.81	97.29 ± 0.81	97.5 ± 0.81	83.92 ± 2.89
breast cancer	98.97 ± 0.01	98.1 ± 0.04	98.95 ± 0.01	75.13 ± 3.95
ionosphère	96.58 ± 0.12	96.4 ± 0.12	96.45 ± 0.13	75.77 ± 1.47
heart	88.44 ± 0.15	87.54 ± 0.17	88.52 ± 0.15	72.65 ± 0.57
pima	79.09 ± 1.44	80.96 ± 0.16	81.18 ± 0.17	63.31 ± 0.2
sonar	67.21 ± 3.79	63.17 ± 2.72	67.59 ± 3.96	56.90 ± 0.87

TAB. 3.2: Comparaison des AUC obtenues en combinant les 40 classifieurs et celles obtenues par sélection, combinaison des classifieurs

3.4.2 Expérimentation

La méthodologie présentée précédemment est démontrée sur un ensemble de données réelles issues de l'UCI, nous permettant de tester notre approche et de comparer nos résultats aux travaux déjà effectués. Les bases que nous utilisons pour nos expérimentations sont exclusivement à deux classes et issues de différents domaines, traitement d'image, traitement de signal ou reconnaissance de caractère. Dans un premier temps, nous allons présenter les données utilisées puis nous enchaînerons sur la présentations des résultats et nous terminerons par les conclusions sur les expérimentations.

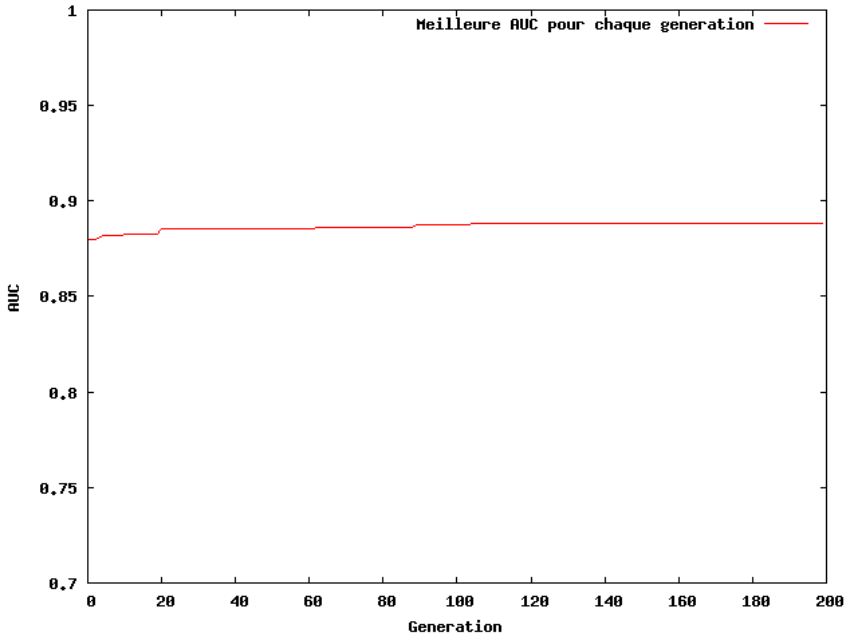
Les bases présentées ici sont toutes extraites de l'*UCI Machine Learning Repository* consultables à l'adresse suivante
`http://archive.ics.uci.edu.ml/`.
Le nombre d'exemple, de caractéristiques ainsi que la distribution des classes est donné dans le tableau 3.3. Les bases que nous utilisons sont identiques à celles de la section 2.3, **australian**, **wdbc**, **breast cancer**, **ionosphère**, **heart**, **pima** et **sonar**.

Avant d'étudier les performances globales en classification, nous allons regarder l'évolution des performances au cours du temps. La figure 3.15 présente deux courbes, une courbe correspondant à l'évolution des performances en fonction du nombre de génération et une autre représentant les performances en fonction du nombre de classifieurs utilisés pour la combinaison. D'autres courbes sont pré-

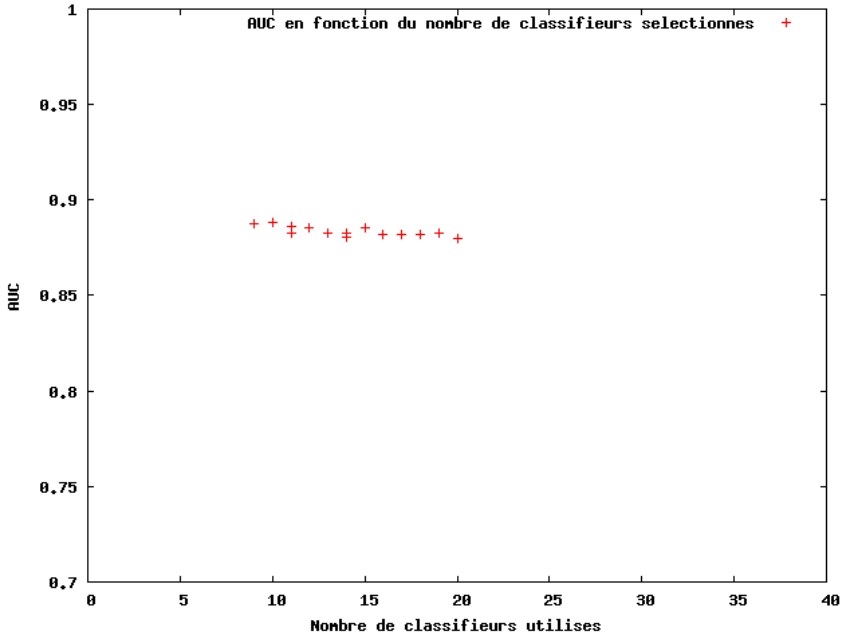
sentées en annexe B.
La courbe 3.15a est représentative de l'intérêt de l'algorithme évolutionnaire sur l'évolution des performances du système. En effet, l'AUC augmente de manière assez forte sur les 20 premières générations puis de manière plus faible. Nous observons au fur et à mesure des générations l'apparition de plateaux indiquant des maximums locaux. La force de l'algorithme évolutionnaire est de pouvoir sortir de maximums locaux afin de déterminer le sous-ensemble de classifieurs le plus performant globalement.
Le nombre de classifieurs sélectionnés à chaque générations est également une données importante. La figure 3.15b montre par génération, le nombre de classifieurs qui ont été sélectionnés pour donner la meilleure performance. Nous utilisons en général au maximum 20 classifieurs parmi notre ensemble de 40. Nous constatons également la dégradation des performances avec l'augmentation du nombre de classifieurs sélectionnées.

Les premiers résultats que nous allons commenter concernent la comparaison entre la combinaison "brute" des classifieurs, c'est-à-dire sans sélection, et la combinaison avec une sélection optimisée des classifieurs. Les résultats présentés dans le tableau 3.2 montrent la différence entre les deux opérateurs de combinaison. En effet, dans tous les cas, sauf un, l'utilisation de la moyenne comme opérateur de combinaison fournit des performances plus élevées. Ceci est principalement du au fait que dans un produit si une valeur est faible, elle pénalise de manière plus importante les résultats. L'écart entre les deux méthodes se réduit de manière importante lorsque nous sélectionnons les classifieurs. Nous constatons que l'algorithme génétique permet de réellement optimiser la combinaison par produit en éliminant les classifieurs peu performants.
Le second élément que nous pouvons remarquer est le peu de différence qui existe entre les performances avec et sans sélection pour l'opérateur moyenne. Il s'avère même, pour les bases heart, pima et sonar que les performances sont plus élevées sans la sélection.

Les résultats présentés dans le tableau 3.5, comparent les performances obtenues dans la littérature, celles obtenues par optimisation des modèles de classifieurs, le Front ROC et celles obtenues par combinaison des classifieurs.



(a) Évolution de l'AUC en fonction des générations



(b) AUC en fonction du nombre de classifieurs sélectionnés

FIG. 3.15: Base australian

Comme nous l'avons dit dans le chapitre 2, l'AUF améliore quasiment à chaque fois les performances par rapport à ce qui existe dans le littérature. Par contre, les performances de la combinaison des classifieurs du Front ROC donne des

résultats plus mitigés. Résultats qui sont meilleurs que ceux la littérature pour australian et wdbc, proches pour breat cancer et ionosphère, et même inférieurs pour heart, pima et sonar.

Problème	# exemples	# attributs	Distribution des classes	
			-1	1
australian	690	14	383	307
wdbc	569	30	212	357
breast cancer	658	9	434	224
ionosphère	351	34	126	225
heart	270	13	150	120
pima	768	8	500	268
sonar	208	60	111	97

TAB. 3.3: Description de problèmes de l'UCI à deux classes

Problème	Distriburion des folds			
	0 à 4		5	
	-1	1	-1	1
australian	63	51	68	52
wdbc	35	59	37	62
breast cancer	72	37	74	39
ionosphère	21	37	21	2
heart	25	20	25	20
pima	83	44	85	48
sonar	18	16	21	17

TAB. 3.4: Description de la répartition des sous-ensembles pour la Cross-Validation

Problème	AUC littérature	Référence	AUF	AUC combinaison	
				Moyenne	Produit
australian	90.25 ± 0.6	[Wu, 2005]	96.51 ± 0.01	91.73 ± 0.02	91.58 ± 0.02
wdbc	94.7 ± 4.6	[Ferri et al., 2002]	98.18 ± 0.01	97.65 ± 0.81	97.29 ± 0.81
breast cancer	99.13	[Boström, 2005]	98.83 ± 0.01	98.97 ± 0.01	98.1 ± 0.04
ionosphère	98.7 ± 3.3	[Rakotomamonjy, 2004]	97.06 ± 0.08	96.58 ± 0.12	96.4 ± 0.12
heart	92.6 ± 0.7	[Wu, 2005]	94.49 ± 0.03	88.44 ± 0.15	87.54 ± 0.17
pima	84.80 ± 6.5	[Cortes and Mohri, 2004]	94.37 ± 0.02	79.09 ± 1.44	80.96 ± 0.16
sonar	81.2 ± 5.1	[Zhou and Jiang, 2004]	86.06 ± 0.59	67.21 ± 3.79	63.17 ± 2.72

TAB. 3.5: Comparaison des AUC obtenues dans la littérature, des aires sous le front (AUF pour Area Under the Front) et des AUC obtenues par combinaison de classifieurs

Conclusion

Dans cette partie nous avons présenté un état de l'art des méthodes de combinaison puis des méthodes sélection des classifieurs. A partir de cette étude bibliographique, nous avons proposé un système d'optimisation permettant de sélectionner le sous-ensemble de classifieurs le plus performant. La genèse de ce système est lié au problème d'évaluation et de comparaison des performances d'un ensemble discret de classifieurs, formant le Front ROC, avec un ensemble continu, formant une courbe ROC. La combinaison a permis de résoudre ce point de théorie afin d'aboutir à un système optimisé d'extraction de classifieurs.

Nous avons montré sur des bases de référence que notre approche fournissait des résultats en demi-teinte vis-à-vis des performances obtenues avec le Front ROC. Nous obtenons d'excellents résultats sur certaines bases ce qui tend à prouver la valeur de cette méthode d'optimisation. Néanmoins, des travaux vont encore être nécessaire afin de valider complètement notre approche du problème.

Conclusion générale

Les travaux présentés dans ce rapport abordent les problèmes de sélection des hyperparamètres d'un classifieur et la combinaison d'un ensemble de classifieurs. L'objectif était d'apporter une amélioration au système présenté dans [Chatelain et al., 2008] par la sélection et la combinaison d'un sous-ensemble optimal. L'approche que nous avons proposée pour atteindre ce but repose essentiellement sur l'évaluation des performances et l'optimisation par algorithme évolutionnaire.

Dans ce contexte, la première partie du travail à concerné l'analyse des méthodes d'évaluation des performances sur des systèmes à deux classes mais également multiclasses. Nous nous sommes restreint, dans le cadre applicatif, aux problèmes à deux classes afin de réduire la complexité du système et de permettre une validation de l'approche.

Ainsi, la première contribution apportée dans le cadre de ce stage concerne la mise en place de la validation croisée de manière à cadrer statistiquement les résultats obtenus avec le système d'optimisation pour la sélection de modèle SVM.

La deuxième contribution concerne la mise en œuvre d'une méthode optimale de sélection de classifieur afin de les combiner. Cette méthode a consisté à utiliser les sorties du premier système d'optimisation (sélection de modèle de classifieurs) pour en sélectionner le sous-ensemble optimal. L'optimisation par algorithme évolutionnaire, nous a permis d'explorer l'espace des sous-ensembles

de classifieurs.

Pour conclure sur les perspectives, nous pensons que l'utilisation des algorithmes génétiques est une excellente solution permettant de répondre simplement et efficacement aux problèmes d'optimisation complexes. Concernant nos systèmes, les évolutions à court terme vont s'orienter dans deux directions. Premièrement, la généralisation de notre méthode à des problèmes multiclasses, rendant ainsi notre approche utilisable dans quasiment toutes les conditions. Deuxièmement, l'évolution des méthodes de combinaisons utilisées permettant ainsi peut-être d'accroître les performance du système.

ANNEXE A

Algorithmes

Algorithme 1 : Génération d'une courbe ROC à partir d'un ensemble d'exemples ordonnées, [Provost and Fawcett, 2004]

Données : E : Liste des couples $\langle I, p \rangle$ avec :
I : Étiquette de l'exemple.
p : Rang assigné à I par le cassifieur.
P,N : Nombre d'exemples respectivement positifs et négatifs présentdans E.
Sorties : R : Liste des points de la courbe ROC.

```
1  begin
2      T_count = 0 ;                                   /* Compteur de TP  */
3      F_count = 0 ;                                   /* Compteur de FP  */
4      plast = −∞ ;                                    /* Dernière valeur utilisée  */
5      R = ⟨⟩ ;                                        /* Liste des points ROC  */
6      classement de E dans l'ordre décroissant des valeurs ;
7      while (E ≠ 0) do
8          suppression du couple ⟨I, p⟩ de la tête de E;
9          if (p ≠ plast) then
10             ajout du point (F_count/N, T_count/P) à la fin de R ;
11             plast = p ;
12         end
13         if I est un exemple négatif then
14             T_count = T_count + 1 ;
15         else I est un exemple négatif
16             F_count = F_count + 1 ;
17         end
18     end
19     ajout du point (F_count/N, T_count/P) à la fin de R ;
20 end
```

Algorithme 2 : Méthode efficace pour la génération de la courbe ROC, [Fawcett, 2006]

Entrées : L, l'ensembles des exemples de test,
$f(i)$, probabilité donnée par le classifieur que l'exemple i soit positif,
P et N , le nombre d'exemples respectivement positifs et négatifs.
Sorties : R, la liste des points ROC ordonnés par rapport au fpr
Données : $P > 0$ et $N > 0$

begin
  $L_{range} \longleftarrow L$ rangé dans l'ordre décroissant des scores f ;
  $FP \longleftarrow TP \longleftarrow 0$;
  $R \longleftarrow \langle\rangle$;
  $f_{prev} \longleftarrow -\infty$;
  $i \longleftarrow 1$;
  while $i \leq |L_{range}|$ **do**
    if $f(i) \neq f_{prev}$ **then**
      mettre $(\frac{FP}{N}, \frac{TP}{N})$ dans R ;
      $f_{prev} \longleftarrow f(i)$;
    end
    if $L_{range}[]$ *est un exemple positif* **then**
      $TP \longleftarrow TP + 1$
    else I est un exemple négatif
      $FP \longleftarrow FP + 1$
    end
    $i \longleftarrow i + 1$
  end
end

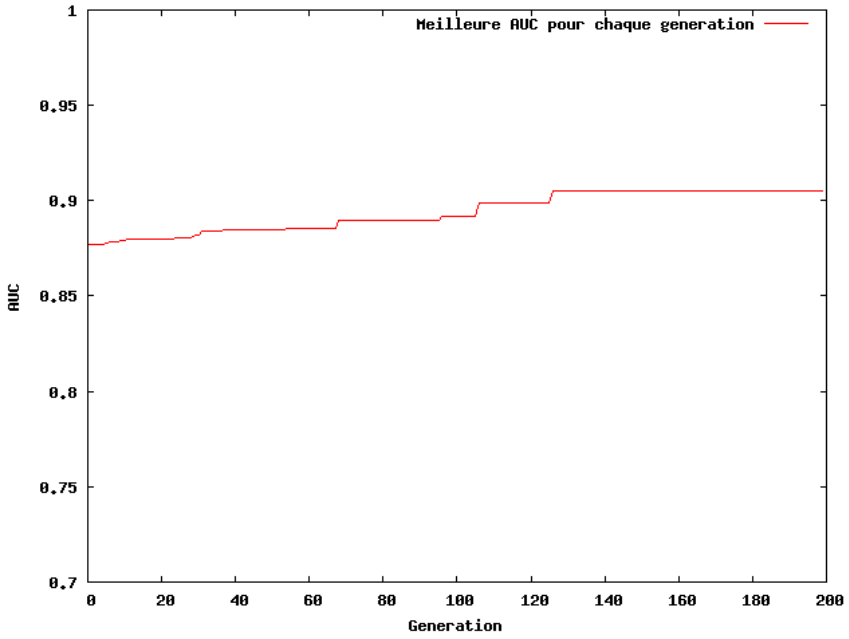
Algorithme 3 : Calcul de l'aire sous la courbe ROC, [Fawcett, 2006]

Entrées : L, l'ensembles des exemples de test,
$f(i)$, probabilité donnée par le classifieur que l'exemple i soit positif,
P et N , le nombre d'exemples respectivement positifs et négatifs.
Sorties : A, l'aire sous la courbe ROC
Données : $P > 0$ et $N > 0$

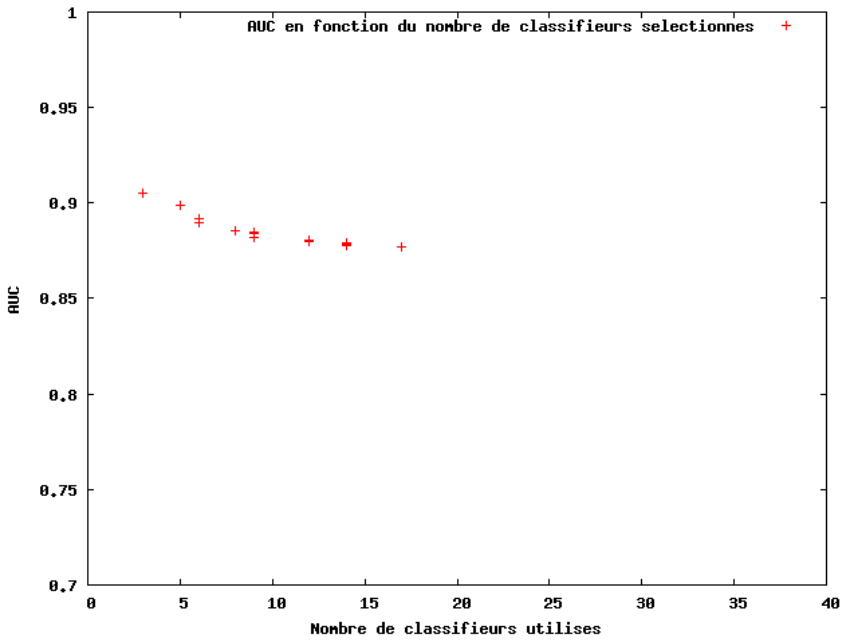
begin
  $L_{range} \longleftarrow L$ rangé dans l'ordre décroissant des scores f ;
  $FP \longleftarrow TP \longleftarrow 0$;
  $FP_{prev} \longleftarrow TP_{prev}$;
  $A \longleftarrow 0$;
  $f_{prev} \longleftarrow -\infty$;
  $i \longleftarrow 1$;
  while $i \leq L_{range}$ **do**
    if $f(i) \neq f_{prev}$ **then**
      $A \longleftarrow A + TRAPEZOID\_AREA(FP, FP_{prev}, TP, TP_{prev})$ $f_{prev} \longleftarrow f(i)$;
      $FP_{prev} \longleftarrow FP$;
      $TP_{prev} \longleftarrow TP$;
    end
    if $L_{range} est un exemple positif$ **then**
      $TP \longleftarrow TP + 1$
    else
      $FP \longleftarrow FP + 1$
    end
    $i \longleftarrow i + 1$;
  end
  $A \longleftarrow A + TRAPEZOID\_AREA(N, FP_{prev}, N, TP_{prev})$;
  $A \longleftarrow A/P * N$;
end
begin
  Fonction TRAPEZOID_AREA(X1,X2,Y1,Y2) $Base \longleftarrow |X1 - X2|$;
  $Height_{avg} \longleftarrow (Y1 + Y2)/2$;
  return $Base * Height_{avg}$;
end

ANNEXE B

Courbes associées à la combinaison de classifieurs

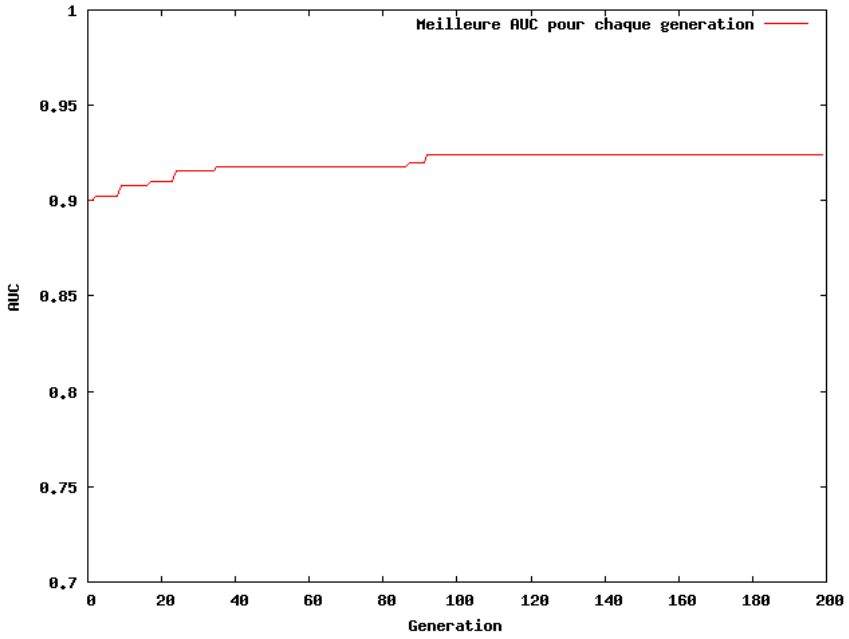


(a) Évolution de l'AUC en fonction des générations

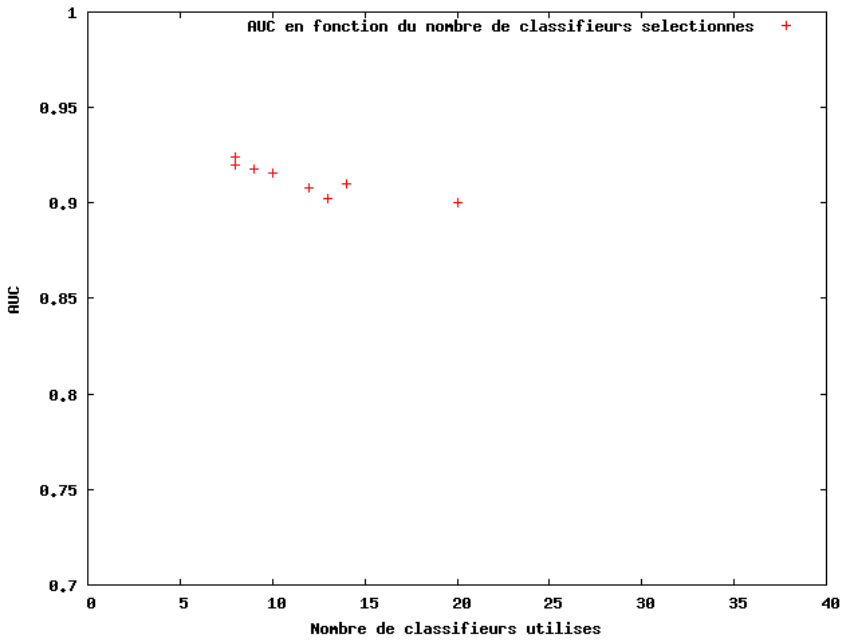


(b) AUC en fonction du nombre de classifieurs sélectionnés

FIG. B.1: Évaluation des performances sur la base pima



(a) Évolution de l'AUC en fonction des générations



(b) AUC en fonction du nombre de classifieurs sélectionnés

FIG. B.2: Évaluation des performances sur la base heart

RÉFÉRENCES

[Adams and Hand, 1999] Adams, N. and Hand, D. (1999). *Comparing classifiers when misallocation cost are uncertain*, volume 32, pages 1139–1147. Pattern Recognition.

[Alpaydin and Jordan, 1996] Alpaydin, E. and Jordan, M. (1996). Local linear perceptrons for classification. In *IEEE Trans. Neural Networks*, volume 7, pages 788–792.

[Anastasio et al., 1998] Anastasio, M., Kupinski, M., and Nishikawa (1998). Optimization and froc analysis of rule-based detection schemes using a multiobjective approach. In *IEEE Trans. Med. Imaging*, volume 17, pages 1089–1093.

[Bellili et al., 2002] Bellili, A., Gilloux, M., and Gallinari, P. (2002). Reconaissance de chiffres manuscrits par un système hybride mlp-svm. In *13ème Congrès Francophone ARIF-AFIA de Reconaissance des Formes et d'Intelligence Artificielle.*, volume 3, pages 761–769.

[Berro, 2001] Berro, A. (2001). Optimisation multiobjectif et stratégie d'évolution en environnement dynamique. Master's thesis, Université des Sciences Sociales Toulouse I.

[Bezdek et al., 1999] Bezdek, J., Keller, J., Krishnapuram, R., and Pal, N. (1999). *Fuzzy Models and Alorithms for Pattern Recognition an d Image Processing*. Norwell,MA :Kluwer.

[Boström, 2005] Boström, H. (2005). "maximizing the area under the roc curve using incremental reduced error pruning". *ROCML.*

[Bradley, 1997] Bradley, A. (1997). The use of the area under the roc curve in evaluation of machine learning algorithms. *Pattern Recognition*, 12 :1145–1159.

[Breiman et al., 1984] Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression trees*. Wadsworth International Group, Belmont, CA.

[Bui et al., 2004] Bui, L., Essam, D., Abbass, H., and Green, D. (2004). Performance analysis of multiobjective evolutionnary methods in noisy environnments. In *APS 2004*, pages 29–39.

[Cao et al., 1994] Cao, J., Ahmadi, M., and Shridhar, M. (1994). Handwritten numerals with multiple features and multistage classifiers. In *IEEE International Journal on Circuits and Systems*, volume 6 of *323–326*.

[Cardoso et al., 2000] Cardoso, J., Fidalgo, J., and Matos, M. (2000). Varaible selection for neural network classifier in the die casting industry. *Proceedings of the International Conference on Engineering Applications of Neural Networks, EANN*, pages 54–60.

[Chang and Lin, 2001] Chang, C. and Lin, C. (2001). Libsvm : a library support vector machines. Website. `http://www.csie.ntu.edu.tw/_cjlin/libsvm`.

[Chapelle et al., 2002] Chapelle, O., Vapnik, V., Bousquet, O., and Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. In *Machine Learning*, volume 46, pages 131–159.

[Chatelain, 2003] Chatelain, C. (2003). Les support vecteur machine (svm). -.

[Chatelain, 2004] Chatelain, C. (2004). Svm : Production des confiances et analyse de comportement. -.

[Chatelain, 2006] Chatelain, C. (2006). "extraction de séquences numériques dans des documents manuscrits quelconques". Master's thesis, Université de ROUEN.

[Chatelain et al., 2007] Chatelain, C., Adam, S., Lecourtier, Y., Heutte, L., and Paquet, T. (2007). "a multi-model selection framework for unknown misclassification costs problems". *ICDAR*. Brésil.

[Chatelain et al., 2008] Chatelain, C., Adam, S., Lecourtier, Y., Heutte, L., Paquet, T., and Oufella, Y. (2008). "optimisation multi-objectif pour la sélection de modèles svm". *RFIA - Amiens*.

[Chatelain et al., 2006] Chatelain, C., Heutte, L., and Paquet, T. (2006). *"Segmentation driven recognition applied to numerical field extraction from handwritten incoming mail documents"*, pages 564–575. Springer.

[Chung et al., 2003] Chung, K., Kao, W., Sun, C., and Lin, C. (2003). Radius margin bounds for support vector machines with rbf kernel. In *Neural comput*, volume 15, pages 2643–2681.

[Clearwater and Stern, 1991] Clearwater, S. and Stern, E. (1991). *A rule-learning program in high energy physics event classification*, volume 67, pages 159–182. Computer Physics Communications.

[Colette, 2006] Colette, Y. (2006). L'optimisation multiobjectif. Website. `http://ycollette.free.fr/spip/IMG/pdf/01_MultiObj.pdf`.

[Corne et al., 2000] Corne, D., Knowles, J., and Oates, M. (2000). The pareto enveloppe-based selection algorithm for multiobjective optimization. In *Parallel problem solving from nature*, pages 839–848.

[Cortes and Mohri, 2004] Cortes, C. and Mohri, M. (2004). "auc optimization vs. error rate minimization". -.

[Dasarathy and Sheela, 1978] Dasarathy, B. and Sheela, B. (1978). A composite classifier system design : Concepts and methodology. In *Proc. IEEE*, volume 67, pages 708–713.

[Dash and Liu, 1997] Dash, M. and Liu, H. (1997). Features selection for classification. *Intelligent Data Analysis*, 1 :131–156.

[Deb, 2000] Deb, K. (2000). *"Introduction to selection. Evolutionary computation 1 : advanced algorithms and operators"*, page 331. Bäck T., Fogel D.B., and Michalewicz Z. ; Institute of Physics Publishing, Bristol and Philadelphia.

[Deb et al., 2000] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2000). A fast elitist nondominated sorting genetic algorithm for multiobjective optimization : Nsga-ii. In *Parallel problem solving from nature*, pages 849–850.

[Deb et al., 2002] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). "a fast and elitist multiobjective genetic algorithm : Nsga-ii.". In *IEEE Transactions on Evolutionary Computation*, volume 6.2, pages 182–197.

[Dreisetl et al., 2000] Dreisetl, S., Ohno-Machado, S., and Binder, M. (2000). *Comparing trichotomous tests by three-way ROC analysis*, volume 20, pages 323–331. Medical Decision Making.

[Dridi, 2005] Dridi, L. (2005). "les algorithmes génétiques". Master's thesis, INRS-ETE.

[Duin, 2002] Duin, R. (2002). The combinig classifier : to train or not to train ? In *IEEE, 16th International Conference on Pattern Recognition (ICPR)*, volume II, pages 765–770.

[Duin et al., 2004] Duin, R., Juszcak, P., Paclìk, P., Pekalska, E., de Rider, D., and Tax, D. (2004). Prtools, a matlab toolbox for pattern reconition. Online.

[Egan, 1975] Egan, J. (1975). Signal detection theory and roc analysis. In *Cognition and Perception*. Academic Press.

[Everson and Fieldsend, 2006] Everson, R. and Fieldsend, J. (2006). *Multi-class ROC analysis from a multi-objective optimisation perspective*, volume 27 of *Pattern Recognition Letters*, pages 918–927. Elsevier Science inc.

[Fawcett, 2006] Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition Letters*, 27 :861–874.

[Fawcett and Povost, 1997] Fawcett, T. and Povost, F. (1997). *Adaptative Fraud Detection*, volume 1 of *Data Mining and Knowledge Discovery*, pages 291–316. Springer Netherlands.

[Fawcett and Provost, 1996] Fawcett, T. and Provost, F. (1996). Combining data mining and machine learning for efficient user profiling. In *Proc. second international conference on Knowledge Discovery and Data Mining*, pages 8–13, Menlo Park, CA. AAAI Press.

[Ferri et al., 2002] Ferri, C., Flach, P., and Hernandez-Orallo, J. (2002). "learning decision trees using the area under the roc curve". In *Proceedings of the 19th International Conference on Machine Learning*, pages 139–146.

[Ferri et al., 2003] Ferri, C., Hernàndez-Orallo, J., and Salido, M. (2003). *Volume under the ROC surface for multi-class problems*, volume 2837/2003 of *Lecture Notes on Computer Science*, pages 108–120. Springer Berlin /Heidelberg.

[Fieldsend and Everson, 2004] Fieldsend, J. and Everson, R. (2004). Roc optimisation of safety related systems. In *Proceedings of ROCAI 2004*, pages 37–44.

[Fieldsend and Everson, 2005a] Fieldsend, J. and Everson, R. (2005a). Formulation and comparison of multi-class roc surfaces. In *Proceeding of ROCML 2005, part of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 41–48.

[Fieldsend and Everson, 2005b] Fieldsend, J. and Everson, R. (2005b). Visualisation of multi-class roc surfaces. In *Proceeding of ROCML 2005, part of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 49–56.

[Flach, 2004] Flach, P. (2004). Tha many faces of roc analysis in machine learning. In *ICML 2004 tutorial*.

[Flach, 2006] Flach, P. (2006). Reinventing machine learning with roc analysis. In *SBIA/IBERAMIA 2006*.

[Flach et al., 2003] Flach, P., Blockeel, H., Ferri, C., Hernàndez-Orallo, J., and Struyf, J. (2003). Decision support for data mining ; introduction to roc analysis and its applications. In *Data mining and Decision support : Integration and Collaboration*. Kluwer Publishers.

[Fonseca and Flemming, 1993] Fonseca, C. and Flemming, P. (1993). genetic algorithm for multiobjective optimization : formulation, discussion and generalization. In *Proceeding of ICGA*, pages 416–423.

[Friedrichs and Igel, 2005] Friedrichs, F. and Igel, C. (2005). Evolutionnary tuning of multiple svm parameters. In *Neurocomputing*, volume 64, pages 107–117.

[Gader et al., 1991] Gader, L., Forester, B., Ganzberger, M., Gillies, M., Mitchell, B., Whalen, M., and Yocum, T. (1991). Recognition of handwritten digits using template and model matching. In *Pattern Recognition*, volume 24, pages 421–431.

[Giacinto and Roli, 2001] Giacinto, G. and Roli, F. (2001). Design of effective neural network ensembles for image classification processes. In *Image Vision and Computing Journal*, volume 22, pages 699–707.

[Goldberg, 1989] Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine learning*. Addison-Wesley Longman Publishing Co., Inc., Boston.

[Goldberg, 1991] Goldberg, D. (1991). *Algorithmes Génétiques*. Magnard.

[Gosselin, 1997] Gosselin, B. (1997). Cooperation of multilayer perceptron classifiers. In *8th Workshop on Circuits, Systems and Signal Processing*, pages 187–190.

[Grefenstette, 1992] Grefenstette, J. (1992). Genetic algorithms for changing environments. In *Problem solving from Nature 2*.

[Gunes, 2001] Gunes, V. (2001). *Reconnaissance des formes évolutives par combinaison, coopération et sélection de classifieurs*. PhD thesis, Université de la Rochelle.

[Hand, 1997] Hand, D. (1997). *Construction and assessment of classification rules*. Wiley.

[Hand, 2001] Hand, D. (2001). Measuring diagnostic accuracy of statistical prediction rules. *Statistica Neerlandica*, 53 :3–16.

[Hand and Tills, 2001] Hand, D. and Tills, R. (2001). A simple generalisation of the area under the roc curve for multiple classification problems. In *Machine Learning*, volume 45, pages 171–186.

[Hanley and McNeil, 1982] Hanley, J. and McNeil, B. (1982). The meaning and use of the area under a receiver operating characteristic (roc). *Rdiology*, 43 :29–36.

[Hao et al., 2003] Hao, H., Liu, C., and Sako, H. (2003). Comparison of genetic algorithm and sequential search method for classifier subset selection. *7th International Conference in Document Analysis and Recognition*, 2 :765–769.

[He and Frey, 2008] He, X. and Frey, E. (2008). *The meaning and use of the volume under a three-class ROC surface (VUS)*, volume 27, pages 1–1. IEEE transaction on Medical Imaging. Accepted for futur publication.

[Heutte, 1994] Heutte, L. (1994). *Reconnaissance de caractères manuscrits : application à la lecture automatique des chèques et des enveloppes postales.* PhD thesis, Université e Rouen.

[Ho et al., 1994] Ho, T., Hull, J., and Srihari, S. (1994). Decision combination in multiple classifier systems. In *IEEE Transactions on Pattern Analysis and Machine Learning*.

[Horn et al., 1994] Horn, J., Nafpliotis, N., and G.D.E (1994). Aniched pareto genetic algorithm for multiobjective optimization. In *Proceeding of IEEE-WCCC*, pages 82–87.

[Hsu and Lin, 2002] Hsu, C. and Lin, C. (2002). A simple decomposition method for support vector machine. In *Machine Learning*, volume 46, pages 219–314.

[Huang and Wang, 2006] Huang, C.-L. and Wang, C.-J. (2006). A ga-based feature selection and parameters optimization for support vector machine. In *Expert systems with application*, volume 31, pages 231–240.

[Huang et al., 1995] Huang, Y., Liu, K., and Suen, C. (1995). The combination of multiple classifiers bu neural network approach. In *International Journal of Pattern Recognition and Artificial Intelligence*, volume 9, pages 570–597.

[Jacobs et al., 1991] Jacobs, R., Jordan, M., Nowlan, S., and Hinton, G. (1991). adaptive mixture of local experts. In *Neural comput.*, volume 17, pages 90–93.

[Jain and Zongke, 1997] Jain, A. and Zongke, D. (1997). Feature selection : evaluation, application and small sample performance. In *IEEE Trans. Pattern Analysis and Machine Intelligence*, volume 19, pages 153–158.

[Kanal, 1974] Kanal, L. (1974). Patterns in pattern recognition. *IEEE Transactions on Information Theory*, 20 :674–722.

[Khare et al., 2002] Khare, V., Yao, X., and Deb, K. (2002). Performance scaling of multiobjective evolutionnary algorithm. Technical report, SCS, University of Birmingham.

[Kharroubi, 2002] Kharroubi, J. (2002). Etude de techniques de classement machines à vecteurs supports pour la vérification automatique du locuteur. Master's thesis, -.

[Kim et al., 2000] Kim, J., Kim, K., Nadal, C., and Suen, C. (2000). A methodology of combining hmm and mlp classifiers for curives word recognition. In *International Conference Document Analysis and Recognition, ICDAR*, pages 319–322.

[Kira and Rendell, 1992] Kira, K. and Rendell, L. (1992). practical approach to feaure selection. In *Proceedings of the 9th International Workshop on Machine Learning.*, pages 249–256.

[Kohavi, 1995] Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

[Kohavi and John, 1997] Kohavi, R. and John, G. (1997). Wrappers for features subset selection. *Artificial Intelligence*, 97 :273–324.

[Kubat et al., 1998] Kubat, M., Holte, R., and Matwin, S. (1998). *Machine learnong for the detection of oils spills in satellite radar images*, volume 30, pages 195–215. Machine Learning.

[Kuncheva, 2002a] Kuncheva, L. (2002a). Switching between selection and fusion in combining classifiers : An experiment. In *IEEE Transactions on systems, man, and cybernetics*, volume 32, pages 146–156.

[Kuncheva, 2002b] Kuncheva, L. (2002b). A theorical study on six classifier fusion strategies. In *IEEE S on PAMI*, volume 24.

[Kupinski and Anastasio, 1999] Kupinski, M. and Anastasio, M. (1999). Multiobjective genetic optimization of diagnostic classifiers with implications for generating receiver operating characteristic curves. In *IEEE Trans. Med. Imaging*, volume 8, pages 673–692.

[Landgrebe and Duin, 2006] Landgrebe, T. and Duin, R. (2006). A simplfied extension of the area under the roc to the multiclass domain. In *Proceedings of the Seventeenth Annual Symposium of the Pattern Recognition Association of South Africa, PRASA 2006*, pages 241–245.

[Landgrebe and Duin, 2007a] Landgrebe, T. and Duin, R. (2007a). Approximating the multiclass roc by pairwise analysis. In *Pattern Recognition Letters*, volume 28, pages 1747–1758.

[Landgrebe and Duin, 2007b] Landgrebe, T. and Duin, R. (2007b). A simplified volume under the roc hypersurface. In *SAIE Africa Research journal*, volume 98, pages 94–100.

[Landgrebe and Duin, 2008] Landgrebe, T. and Duin, R. (2008). Efficient multiclass roc approximation by decomposition via confusion matrix perturba-

tion analysis. *IEEE Transactions on Pattern Analysis and Machine Learning*, 30(5).

[Lane, 2000] Lane, T. (2000). Extensions of roc analysis to multi-class domains. In *ICML, Workshop on cost-sensitive learning.*

[Lavalle and Branicky, 2002] Lavalle, S. and Branicky, M. (2002). On the relationship between classical grid search and probabilistic roadmaps. In *International Journal of Robotics research*, volume 23, pages 673–692.

[Lee and Srihari, 1995] Lee, D.-S. and Srihari, S. (1995). A theory of classifier combination : the neural network approach. In *IEEE International Conference in Document Analysis and Recognition (ICDAR)*, pages 42–45.

[Lewis, 1990] Lewis, D. (1990). Representation quality in text classification : An introduction and experiment. In *Proc. workshop on speech and natural language*, pages 288–295, Hidden Valley, PA.

[Lewis, 1991] Lewis, D. (1991). Evaluating text categorization. In *Proc. workshop on speech and natural language*, pages 312–318.

[Metz, 1978] Metz, C. (1978). Basic principles of roc analysis. In *Seminars in Nuclear Medicine*, volume 3.

[Mohamadally and Fomani, 2006] Mohamadally, H. and Fomani, B. (2006). "svm : Machines à vecteurs de support ou séparateurs à vastes marges". *BD Web, ISTY3, Versailles St Quentin, France*.

[Moobed, 1996] Moobed, B. (1996). *Combinaison de classifieurs, une nouvelle approche*. PhD thesis, Université Paris sud, UFR Scintifique d'Orsay.

[Mossman, 1999] Mossman, D. (1999). Three way rocs. *Medical Dcision Making*, 19 :78–89.

[Mozer et al., 2002] Mozer, M., Dodier, R., Colagrosso, M., Guerra-Salcedo, C., and Wolniewiez, R. (2002). Prodding the roc curve : Constrained optimization of classifier performance. In *NIPS*, pages 1409–1415.

[Murphy and Aha, 1992] Murphy, P. and Aha, D. (1992). Uci repository of machine learning databases. `ftp://ftp.ics.uci.edu/pub/machine-learning-databases`.

[Musicant et al., 2003] Musicant, D., Kumar, V., and Ozgur, A. (2003). Optimizing f-measure with support vector machines. In *FLAIRS Conference*, pages 356–360.

[Nakache and Métais, 2005] Nakache, D. and Métais, E. (2005). Evaluation : nouvelle approche avec juges. In *XXIIIe congrès INFORSID*, page 15, Grenoble.

[Narendra and Fukunaga, 1977] Narendra, P. and Fukunaga, K. (1977). A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, 26 :917–922.

[Ohkura and Ueda, 1995] Ohkura, O. and Ueda, K. (1995). A genetic algorithm for nonstationnary function optimization problems. In *Trans. SICE*, volume 8, pages 269–276.

[Oliveira, 2008] Oliveira, E. (2008). Construction dynamique de forêts aléatoires. Master's thesis, Laboratoire LITIS, Université de Rouen.

[Osuna et al., 1997] Osuna, E., Freund, R., and Girosi, F. (1997). Support vector machines : Training and applications.

[Paclìck, 2004] Paclìck, P. (2004). *Building road sign classifiers*. PhD thesis, CTU Prague, Czech Republic.

[Prevost et al., 2003] Prevost, L., Michel-Sendis, C., Moises, A., Oudot, L., and Milgram, M. (2003). Combining model-based and discriminative classifiers : application to handwritten character recognition. In *7th International Conference on Document Analysis and Recognition*.

[project, 2004] project, E. (2004). European esprit 5516 project. Satimage dataset.

[Provost and Domingos, 2001] Provost, F. and Domingos, P. (2001). Well trained pets : Improving probability estimation trees. In *CeDER Working Paper IS-00-04*, New York University - NY 10012.

[Provost and Domingos, 2003] Provost, F. and Domingos, P. (2003). *Tree induction for probability-based ranking*, volume 52 of *Machine Learning*, pages 199–215. Springer Netherlands.

[Provost and Fawcett, 1997] Provost, F. and Fawcett, T. (1997). Analysis and visualisation of classifier performance : Comparison under imprecise class and cost distributions. In *Proc. third internat. Conf. on Knowledge Discovery and Data Mining, KDD-97*, pages 43–48, Menlo Park, CA. AAAI Press.

[Provost and Fawcett, 1998] Provost, F. and Fawcett, T. (1998). Robust classification systems for imprecise environments. In *Proc. AAAI-98*, pages 706–713, Menlo Park, CA. AAAI Press.

[Provost and Fawcett, 2001] Provost, F. and Fawcett, T. (2001). *Robust classification systems for imprecise environments*, volume 42, pages 203–231. Machine Learning.

[Provost and Fawcett, 2004] Provost, F. and Fawcett, T. (2004). *Robust Classification for Imprecise Environnements*, volume 42 of *Machine Learning*, pages 203–231. Springer Netherlands.

[Pudil et al., 1994] Pudil, P., Novovicova, J., and Kittler, J. (1994). Floating search methods in feature-selection. *Pattern Recognition Letter, PRL.*, 15 :1119–1125.

[Rakotomamonjy, 2004] Rakotomamonjy, A. (2004). Optimizing auc with support vector machine. In *European Conference on Artificial Intelligence Workshop on ROC Curve and AI*, pages 469–478.

[Rastrigin and Erenstein, 1981] Rastrigin, L. and Erenstein, R. (1981). *Method of Collective Recognition*. Russia : Energoizdat.

[Rebaine, 2005] Rebaine, D. (2005). Methode de branch and bound. Cours de conception et analyse des algorithmes. Université du Quebec à Chicoutimi. 2005.

[Rhaman and Fairhust, 1999] Rhaman, A. and Fairhust, M. (1999). A study of some multi-expert recognition strategies for industrial applications : issues of processing speed and implementability. In *Vision Interface*.

[Rhaman and Fairhust, 2003] Rhaman, A. and Fairhust, M. (2003). Multiple classifier decision combination strategies for character recognition : a review. In *Journal Document Analysis and Recognition, JDAR*, pages 166–194.

[Rogova, 1994] Rogova, G. (1994). Combining the results of several neural network classifiers. In *Neural Networks*, volume 7, pages 777–781.

[Roli et al., 2001] Roli, F., Giancinto, G., and Vernazza, G. (2001). Methods for designing multiple classifier systems. In -, pages 78–87. Springer-Verlagp.

[Sahiner et al., 2008] Sahiner, B., Chan, H., and Hadjiiski, L. (2008). *Performance analysis of three-class slassifiers : properties of a 3-D ROC surface*

and the normalized Volume Under the Surface for ideal observer, volume 27, pages 215–227. IEEE transaction on Medical Imaging.

[Saitta and Neri, 1998] Saitta, L. and Neri, F. (1998). *Learning in the "real world"*, volume 30, pages 133–163. Machine Learning.

[Schaffer and Grefenstette, 1985] Schaffer, J. and Grefenstette, J. (1985). Multiobjective learning via genetic algorithms. In *IJCAI*, pages 593–595.

[Shaomin, 2005] Shaomin, S. (2005). "a scored auc metric for classifier evaluation and selection". *ROCML*.

[Sharkley et al., 2000] Sharkley, A., Sharkley, N., Gerecke, U., and Chandroth, G. (2000). The "test and select" approach to ensemble combination. In *1st International Workshop, Multiple Classifier Systems (MCS), Lectures Notes in Computer Science*, volume 1857, pages 30–44.

[Sklansky, 2000] Sklansky, M. (2000). Comparison of algorithms that the select features for pattern classifiers. In *SSPR/SPR*.

[Spackman, 1989] Spackman, K. (1989). Signal detection theory : Valuable tools for evaluating inductive learning. In *Proc. sixth Internat. Workshop on machine learning*, pages 160–163, San Mateo, CA.

[Srinivas and Deb, 1994] Srinivas, N. and Deb, K. (1994). optimization using nondominated sorting in genetic algorithms. In *Evolutionnary computational*, pages 221–248.

[Srinivasan, 1999] Srinivasan, A. (1999). Note on the location of optimal classifiers in n-dimensional roc space. Technical Report PRG-TR-2-99, Oxford University Computing Laboratory.

[Swets, 1988] Swets, J. (1988). *Measuring the accuracy of diagnostic systems*, pages 1285–1293. Science 240.

[Swets et al., 2000] Swets, J., Dawes, R., and Monahan, J. (2000). *Better decisions through science*, volume 283, pages 82–87. Scientific American.

[Van Rijsbergen, 1979] Van Rijsbergen, K. (1979). *Information retrievel*. Butterworths, London, 2 edition. `www.dcs.gla.ac.uk/Keith/Preface.html`.

[Vuurpijl and Schomaker, 1998] Vuurpijl, L. and Schomaker, L. (1998). A framework for using multiple classifiers in a multiple agent architecture. In *3rd*

Europeen International Workshop on Handwriting Analysis and Recognition, pages 1–6.

[Wang and Rhee, 2007] Wang, X. and Rhee, P. K. (2007). Adaptive classifier selection system using context-driven genetic algorithm. In *Frontiers in the Convergence of Bioscience and Information Techologies*, pages 790–794.

[Webb, 2002] Webb, A. (2002). Statistical pattern recognition.

[Wolpert, 1992] Wolpert, D. (1992). Stacked generalization. In *Neural Networks*, pages 241–259.

[Woods et al., 1997] Woods, K., Kegelmeyer, W., and Bowyer, K. (1997). Combination of multiple classifiers using local accuracy estimates. In *IEEE Tansactions on Pattern Analysis and Machine Intelligence*, volume 19, pages 405–410.

[Wu et al., 2006] Wu, C.-H., Tzeng, G.-H., Goo, Y.-J., and Fang, W.-C. (2006). A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy. *Expert systems with application*, pages 231–240.

[Wu, 2005] Wu, S. (2005). auc metric for classifier evaluation and selection. In *ROCML 2005*.

[Zhou and Jiang, 2004] Zhou, Z.-H. and Jiang, Y. (2004). Nec4.5 : Neural ensemble based c4.5. In *IEEE Transaction on Knowledge and Data Engineering*.

[Zitzler et al., 2001] Zitzler, E., Laumanns, M., and Thiele, L. (2001). spea2 : Improving the strength pareto evolutionnary algorithm. Technical report, Swiss Federal Intitute of Technology.

[Zitzler and Thiele, 1999] Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionnary algorithms : A comparison case study and the strength pareto approach. In *Evolutionnary Computation*, pages 257–271.

[Zongker and Jain, 1996] Zongker, D. and Jain, A. (1996). Algorithms for feature selection : An evaluation. *Proceedings of the 13th Internaional Conference on Pattern Recognition, ICPR*, 2 :18–22.

[Zou, 2002] Zou, K. (2002). Receiver operating characteristics. On-line bibliography.

[Zweig and Campbell, 1993] Zweig, M. and Campbell, G. (1993). Receiver operating characteristic (roc) plots. *Clinical Chemistry*, 29 :561–577.