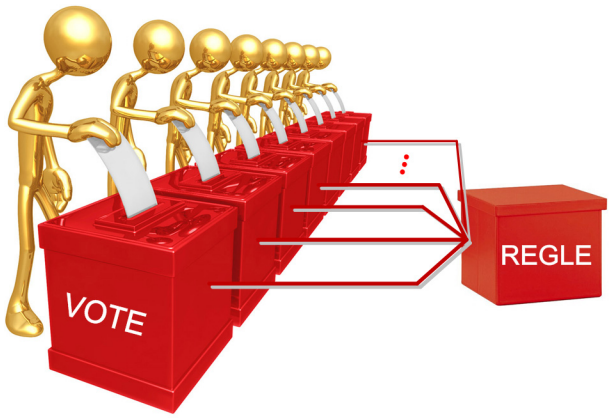
Modélisation probabiliste de classifieurs d'ensemble pour des problèmes à deux classes

Yuan Dong

Thèse
de doctorat
de l'UTT

Yuan DONG

Modélisation probabiliste de classifieurs d'ensemble pour des problèmes à deux classes



Spécialité :
Optimisation et Sûreté des Systèmes

2013TROY0013 **Année 2013**

utt
université de technologie
Troyes

THESE

pour l'obtention du grade de

DOCTEUR de l'UNIVERSITE DE TECHNOLOGIE DE TROYES

Spécialité : OPTIMISATION ET SURETE DES SYSTEMES

présentée et soutenue par

Yuan DONG

le 8 juillet 2013

Modélisation probabiliste de classifieurs d'ensemble pour des problèmes à deux classes

JURY

M. L. HEUTTE	PROFESSEUR DES UNIVERSITES	Président
M. P. BEAUSEROY	PROFESSEUR DES UNIVERSITES	Directeur de thèse
Mme S. LELANDAIS-BONADE	PROFESSEUR DES UNIVERSITES	Rapporteur
M. F. MORAIN-NICOLIER	PROFESSEUR DES UNIVERSITES	Rapporteur
M. A. SMOLARZ	MAITRE DE CONFERENCES - HDR	Directeur de thèse

THÈSE

présentée pour obtenir le grade de

Docteur de l'Université de Technologie de Troyes

Spécialité : Optimisation et Sûreté des Systèmes

présentée et soutenue publiquement par

Yuan DONG

le 08 Juillet 2013

Titre :

Modélisation probabiliste de classifieurs d'ensemble pour des problèmes à deux classes

préparée au sein du laboratoire ICD - équipe LM2S

COMPOSITION DU JURY

M.	Pierre BEAUSEROY	Co-Directeur de thèse	Professeur des Universités
M.	Laurent HEUTTE	Examinateur	Professeur des Universités
Mme.	Sylvie LELANDAIS-BONADÉ	Rapporteur	Professeur des Universités
M.	Frédéric MORAIN-NICOLIER	Rapporteur	Professeur des Universités
M.	André SMOLARZ	Co-Directeur de thèse	Maître de Conférences-HDR

Remerciements

Je tiens à exprimer ma gratitude à toutes les personnes qui m'ont aidée et encouragée pendant mes études de doctorat.

Je tiens dans un premier temps à exprimer ma profonde reconnaissance à mes directeurs de thèse, Messieurs Pierre BEAUSEROY et André SMOLARZ qui m'ont accompagnée tout au long de ces trois ans et demi. Je ne saurais assez les remercier pour leur disponibilité, leur soutien indéfectible, leur enthousiasme scientifique et leurs encouragements qui m'ont permis de mener à bien mes travaux, surtout dans les moments les plus difficiles. Je remercie particulièrement Monsieur SMOLARZ qui a investi beaucoup de temps et d'efforts dans ce manuscrit. Merci beaucoup à vous.

J'exprime toute ma gratitude aux membres du jury qui me font l'honneur de participer à l'examen de cette thèse : Madame Sylvie LELANDAIS-BONADÉ, Professeur des Universités à l'Université d'ÉVRY et Monsieur Frédéric MORAIN-NICOLIER, Professeur des Universités à l'Université Reims Champagne Ardennes en leur qualité de rapporteurs, ainsi que Monsieur Laurent HEUTTE, Professeur des Universités à l'Université de Rouen en sa qualité d'examinateur.

Tout mon travail s'est déroulé au sein du laboratoire LM2S de l'Université de Technologie de Troyes. Je souhaite également remercier tous les membres de l'équipe pour leur acceuil, leurs sympathie ainsi que leurs idées constructives. Un grand merci aux secrétaires du pôle ROSAS : Mesdames Marie-José ROUSSELET et Véronique BANSE ainsi qu'aux secrétaires de l'école doctorale : Mesdames Isabelle LECLERCQ, Pascale DENIS et Thérèse KAZARIAN.

Je remercie Madame Ling GONG, administrateur du programme CSC-UT/INSA et Madame Yaping QIANG, secrétaire de l'ambassade de Chine en France, pour leur supervision, leurs conseils et leurs aides chaleureuses. Je remercie également le China Scholarship Council pour avoir contribué au financement de cette thèse.

Je tiens à remercier mes amis avec lesquels j'ai vécu des moments agréables durant mes années de présence à Troyes : Peng Li, Xiaoyue MA, Wenhua ZHAO, Wenjin ZHU, Fei ZHU, Tian WANG, Lei QIN et Guoliang ZHU. Je remercie également mes collègues de bureau Xiyan He, Imane MAATOUK, Hai Canh VU et les autres doctorants du LM2S, pour leurs aides lorsque le besoin s'en faisait sentir, mais aussi leur bonne humeur.

Enfin, je ne saurais comment remercier mes parents, mon frère et mon fiancé : les quatre personnes les plus importantes de toute ma vie qui m'ont toujours soutenu, m'ont donné confiance et m'ont encouragé et sans qui rien n'aurait été possible.

Yuan DONG

Résumé

L'objectif de cette thèse est d'améliorer ou de préserver les performances d'un système décisionnel quand l'environnement peut impacter certains attributs de l'espace de représentation à un instant donné ou en fonction de la position géographique de l'observation. S'inspirant des méthodes d'ensemble, notre approche a consisté à prendre les décisions dans des sous-espaces de représentation résultant de projections de l'espace initial, espérant ainsi travailler dans des sous-espaces non impactés. La décision finale est alors prise par fusion des décisions individuelles. Dans ce contexte, trois méthodes de classification (one-class SVM, Kernel PCA et Kernel ECA) ont été testées en segmentation d'images texturées qui constitue un support applicatif parfaitement adéquat en raison des ruptures de modèle de texture aux frontières entre deux régions. Ensuite, nous avons proposé une nouvelle règle de fusion reposant sur un test du rapport de vraisemblance pour un ensemble de classifieurs indépendants. Par rapport au vote majoritaire, cette règle de fusion a montré de meilleures performances face à l'altération de l'espace de représentation. Enfin, nous avons établi un modèle conjoint pour l'ensemble des variables décisionnelles de Bernoulli corrélées associées aux décisions des classifieurs individuels. Cette modélisation doit permettre de lier les performances des classifieurs individuels à la performance de la règle de décision globale et d'étudier et de maîtriser l'impact des changements de l'espace initial sur la performance globale.

Mots clés : apprentissage automatique, classification, fusion multicapteurs, reconnaissance des formes (informatique).

Abstract

The objective of this thesis is to improve or maintain the performance of a decision-making system when the environment can impact some attributes of the feature space at a given time or depending on the geographical location of the observation. Inspired by ensemble methods, our approach has been to make decisions in representation subspaces resulting of projections of the initial space, expecting that most of the subspaces are not impacted. The final decision is then made by fusing the individual decisions. In this context, three classification methods (one-class SVM, Kernel PCA and Kernel ECA) were tested on a textured images segmentation problem which is a perfectly adequate application support because of texture pattern changes at the border between two regions. Then, we proposed a new fusion rule based on a likelihood ratio test for a set of independent classifiers. Compared to the majority vote, this fusion rule showed better performance against the alteration of the performance space. Finally, we modeled the decision system using a joint model for all decisions based on the assumption that decisions of individual classifiers follow a correlated Bernoulli law. This model is intended to link the performance of individual classifiers to the performance of the overall decision rule and to investigate and control the impact of changes in the original space on the overall performance.

Key words : machine learning, classification, multisensor data fusion, pattern recognition.

Table des matières

Liste des tableaux

Liste des figures

Chapitre I

Introduction générale

I.1 Problématique de la thèse

La maîtrise du fonctionnement et la surveillance des systèmes complexes constituent un enjeu majeur dont un des objectifs est de garantir la sûreté de fonctionnement de ces systèmes. Dans le cas des systèmes complexes, un des problèmes récurrents est la difficulté d'établir un modèle analytique de fonctionnement. En l'absence de ce type de modèle, la détermination d'une règle de surveillance peut s'appuyer sur l'expertise ou sur les données provenant du système. Dans de nombreuses applications, les approches à partir de données ont apporté des résultats pertinents. Le principe de ces méthodes consiste à exploiter les données disponibles pour estimer ou apprendre une fonction permettant d'inférer une catégorie à partir d'un ensemble de mesures. Parmi les problèmes d'apprentissage possibles, l'apprentissage supervisé est le plus étudié. Il s'agit d'exploiter un jeu de données initial constitué de couples mesures-label pour estimer une fonction de prédiction du label à partir des mesures. Cette fonction ou règle de décision doit ensuite être exploitée avec de nouvelles mesures. Les mesures sont généralement issues de divers capteurs (caméras, radar, sonar...) qui forment l'espace de représentation complet du problème. Les labels représentent les différents états du système ou plus généralement des catégories cibles. Les labels des données initiales sont fournis par des experts.

Jusqu'à une période récente, les données étaient relativement peu nombreuses et souvent collectées périodiquement de façon centralisée. La baisse des coûts des capteurs et les nouveaux modes de transmission de l'information (transmission sans fil...) ouvrent la voie à un accroissement significatif du nombre de capteurs et de la quantité d'information disponible sur les systèmes. Cette évolution conduit à des modes de traitements nouveaux comme par exemple le traitement distribué des données de surveillance. Mais ces capteurs bas coûts posent des problèmes spécifiques en terme de fiabilité, en terme d'accessibilité (les liaisons peuvent être interrompues...), de périodes d'échantillonnage et d'exploitation de la bande passante disponible.

Les mesures issues de ces capteurs constituent l'espace de représentation du système de surveillance. Dans ce contexte, les pannes de capteurs, leur remplacement, la perturbation des transmissions sont autant de phénomènes qui peuvent venir modifier les mesures. Ces perturbations se traduisent par un changement brutal des lois associées à la réponse des capteurs concernés. Il en résulte une perception incomplète ou déformée du système qui ne correspond plus à ce qui a été appris. La distribution des données observées ne correspond plus à la distribution des données utilisées lors

de l'apprentissage. Ces dernières sont souvent recueillies en vue de mettre au point le système de diagnostic et donc en prenant toutes les précautions nécessaires pour éviter ces dysfonctionnements indésirables.

Une stratégie possible consiste, en l'absence de connaissance plus précise, à affecter à ces données perturbées un label de rejet. Toutefois, dans de nombreux cas, le sous-ensemble des mesures non perturbées pourrait suffire à prendre une décision. Le rejet est donc une solution peu satisfaisante quand le système subit des perturbations presque en permanence. L'option consistant à ignorer les perturbations conduit à appliquer une règle de décision sur des données inappropriées. Les performances du système de décision peuvent alors se dégrader très sensiblement. Une dernière option consiste à essayer de développer des classifieurs ayant une bonne aptitude à maintenir leurs performances en présence de perturbations. Cette option fait l'objet du travail de cette thèse.

I.2 Objectif et contributions

De façon plus générale, précisément en raison de la complexité et de la variabilité de l'environnement dans les applications pratiques, il est nécessaire de concevoir un système décisionnel qui est par nature robuste vis à vis des changements ou des perturbations. Les objectifs de cette thèse sont de proposer des nouvelles méthodes de classification et donc de construire la règle de décision performante, dans le but de préserver ou d'améliorer les performances globales d'un système décisionnel dans l'environnement non-stationnaire.

Les travaux que nous avons menés portent sur les situations pour lesquelles des changements ou des perturbations de l'environnement n'affectent qu'une partie seulement des capteurs ou des mesures (le cas où tous les capteurs sont affectés étant considéré comme une situation extrême). Dans une telle situation, nous considérons que tout se passe comme si l'environnement n'agissait que sur une sélection d'attributs au sein de l'espace de représentation et il nous a donc semblé opportun d'adopter une méthode de classification reposant sur un ensemble de classifieurs distincts opérant dans des sous-espaces d'attributs dont on espère que certains seront non altérés par l'environnement.

Les contributions principales de cette thèse sont les suivantes :

1. une évaluation de l'efficacité d'une méthode de sélection d'un ensemble d'espaces de représentation pour l'élaboration d'un ensemble de classifieurs est réalisée. Elle montre l'efficacité de travailler sur des sous-ensemble de représentation pour la classification d'ensemble. Elle montre également que les méthodes d'ensembles permettent d'améliorer la robustesse des décisions face aux changements ou perturbations.
2. une évaluation des performances de la méthode d'ensemble avec différentes méthodes de classification individuelles et différentes façons de définir les espaces de représentation est menée. Elle montre la sensibilité du résultat final à l'ensemble des composants du classifieur d'ensemble et de ses paramètres. Elle met clairement

en évidence la difficulté d'optimiser les choix de méthodes et de paramètres lors de la conception d'un classifieur d'ensemble.

3. une modélisation probabiliste du classifieur d'ensemble est proposée. Elle utilise comme point de départ un modèle conjoint pour un ensemble de variables décisionnelles de Bernoulli corrélées. Ce modèle permet de lier les performances individuelles des classifieurs à la performance de la règle de décision globale. Cet aspect est très intéressant dans le cas des méthodes d'ensemble. Plus précisément, si l'on change l'ensemble des classifieurs sur lesquels repose le classifieur global en retirant certains classifieurs, la connaissance du modèle conjoint complet permet rapidement d'avoir une estimation des performances de la nouvelle règle d'ensemble.

I.3 Structure du mémoire

A la suite de cette introduction, le chapitre II présente un état de l'art sur les méthodes de classification non paramétriques. Les travaux effectués dans cette thèse ne font pas référence aux méthodes dites paramétriques et elles n'ont donc pas été prises en considération dans cette revue bibliographique. Nous avons distingué les méthodes individuelles (arbres de décision, réseaux de neurones, méthode des k plus proches voisins, SVM...) des méthodes d'ensemble (Bagging, Boosting et Random subspace method) que nous avons retenu dans nos travaux.

Le chapitre III consiste dans un premier temps à définir un ensemble d'espaces de représentation selon le principe de la méthode RSM (random subspace method). Sur la base de trois méthodes de classification (one-class SVM, analyse en composante principale à noyau et analyse en composante d'entropie à noyau), nous avons mis au point un classifieur dans chaque sous-espace de représentation et défini ainsi un ensemble de classifieurs. Comme dans beaucoup de méthodes d'ensemble, la décision finale est ensuite prise par combinaison des décisions des classifieurs, et nous avons opté pour la règle de fusion du vote majoritaire. Une comparaison entre les trois méthodes est présentée, montrant l'intérêt des méthodes d'ensemble en présence de perturbations.

En utilisant la même stratégie de construction d'un ensemble de classifieurs que celle présentée dans le chapitre III, le chapitre IV présente le développement d'une nouvelle règle de fusion pour des classifieurs indépendants. Cette règle repose sur un test du rapport de vraisemblance s'inspirant du critère de *Bayes* et du critère de *Neyman-Pearson*. Par rapport au vote majoritaire, cette règle de fusion montre une bonne performance quand l'espace complet de représentation est perturbé. Ensuite dans le chapitre IV nous proposons une modélisation probabiliste conjointe des sorties des classifieurs individuels afin de prendre en compte les dépendances entre ces classifieurs. Nous nous sommes pour cela appuyés sur le modèle de Bahadur-Lazarsfeld et nous avons testé également une variante reposant sur le maximum d'entropie de la distribution conjointe. Cela revient à établir un modèle conjoint pour un ensemble de variables de Bernoulli corrélées. Ce modèle permet de mieux mettre en relation les performances individuelles des classifieurs et la performance de la règle de décision globale. Il doit permettre également d'ajuster aisément la règle globale en cas de perte de capteurs.

Enfin, le chapitre V est consacré aux extensions des travaux du chapitre IV. D'une part, afin de pallier l'inconvénient du modèle standard du maximum d'entropie en termes de complexité de calcul, nous proposons quelques pistes de simplifications. D'autre part, nous proposons quelques réflexions et pistes sur la manière de prendre en compte le cas où des capteurs sont en panne à un moment donné.

Chapitre II

État de l'art sur les méthodes de classification

II.1 Introduction

En apprentissage automatique on peut distinguer deux grandes familles : l'apprentissage supervisé et l'apprentissage non-supervisé.

L'apprentissage supervisé est réalisé à partir d'une base de données contenant un ensemble des mesures ou d'attributs ainsi que les valeurs cibles encore nommées labels. Le but est de trouver un modèle qui décrit au mieux la relation entre les attributs et les labels afin de prédire le label d'une nouvelle données issue du système à surveiller. La régression (Oltean et Dioşan - 2009) et la classification sont deux types de tâches qui se situent dans le cadre de l'apprentissage supervisé.

L'apprentissage non-supervisé se distingue de l'apprentissage supervisé par le fait qu'il n'y a pas de label a priori sur les données. Il s'agit donc de partitionner un ensemble de données hétérogènes en plusieurs sous-ensembles de manière à ce que les données considérées comme les plus similaires soient associées au sein d'un ensemble homogène et qu'au contraire les données considérées comme différentes se retrouvent dans un autre ensemble distinct. L'objectif est de réaliser une extraction de connaissance organisée à partir des données. L'apprentissage non-supervisé est aussi habituellement appelé *clustering* (Jain et al. - 1999; Bezdek et al. - 1994; Li et al. - 2003; De Falco et al. - 2004; Liu et al. - 2005).

C'est sur le problème de classification qu'a porté notre travail. Dans de nombreux domaines comme les affaires, les sciences, l'industrie et la médecine, beaucoup de problèmes peuvent être vus comme des problèmes de classification. De plus, l'arrivée sur le marché d'ordinateurs de plus en plus performants a rendu possible les approches de classification dans une grande variété d'applications telles que la prédiction de faillite, l'inspection des produits, le diagnostic et le pronostic dans le domaine médical, la reconnaissance des formes (reconnaissance de la parole, de l'écriture manuscrite, reconnaissance du visage...), les applications liées à internet (catégorisation de textes, fouille de données)...

Dans le cas de la classification supervisée (Bishop - 2006), chaque exemple est décrit par un nombre m de mesures rassemblées dans un vecteur d'attributs $\boldsymbol{x} \in \mathcal{X}^m \subseteq \mathbb{R}^m$ ainsi que par un label $y \in \mathcal{Y} = \{1, 2, \ldots, K\}$ qui lui est associé. Le processus d'apprentissage a pour but d'établir une transformation h de l'espace d'attributs vers

l'espace de labels.

$$\mathcal{X}^m \xrightarrow{h} \mathcal{Y} = \{1, 2, \ldots, K\}$$

C'est cette transformation h qui est souvent appelée classifieur et dont le rôle est ensuite de permettre d'associer un label à tout nouvel exemple pour lequel on ne dispose pas d'information a priori. La construction d'un classifieur est une étape cruciale pour le problème de classification et pour cette raison il faut très souvent considérer d'autres étapes préliminaires de prétraitement afin de garantir la performance globale de classification. La structure générale d'un tel processus de classification est illustrée à la figure II.1.

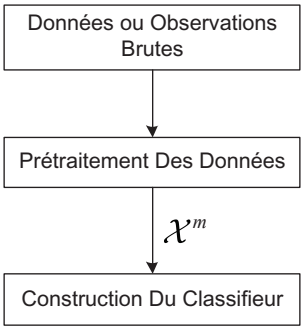


Figure II.1 – Structure d'un système de classification

Le recueil et le prétraitement des données constituent des étapes préparatoires indispensables à la classification. En effet les données brutes peuvent être bruitées ou incomplètes et il est alors nécessaire de procéder à quelques traitements adaptés (filtrage, égalisation d'histogramme, réduction de la dispersion, amélioration du rapport signal à bruit...) pour ne pas pénaliser les performances du système d'apprentissage. Par ailleurs, la définition et le choix des attributs représentant les données reposent la plupart du temps sur des traitements préliminaires des données (transformations, échantillonage, codage, calculs...) (Zhang et al. - 2003).

Face à une masse importante de données brutes, la définition et le choix des attributs les plus informatifs peuvent être effectués par un expert lorsque cela est possible. Dans le cas où l'on ne dispose pas d'expertise, la méthode la plus simple consiste souvent en une recherche la plus exhausive possible en espérant ainsi sélectionner les exemples et les attributs les plus informatifs et pertinents. Toutefois, une telle recherche n'est faisable que dans le cas de faibles quantités de données. Les diverses approches existantes seront détaillées dans la section suivante.

A l'issue du processus de prétraitement, on peut procéder à la construction d'un classifieur qui va opérer dans l'espace complet de représentation $\mathcal{X}^m$ ou dans des sous-espaces de $\mathcal{X}^m$ comme nous le verrons dans la suite. Lors de la mise au point d'un classifieur, le choix d'une méthode d'apprentissage approprié est une étape cruciale sur le plan des performances du système de classification.

Aujourd'hui, une grande variété de méthodes d'apprentissage est proposée dans la littérature. Ces méthodes peuvent se distinguer selon différents groupes en fonction des critères utilisés ou du contexte. Par exemple :

- Approches supervisées ou non-supervisées selon que l'on dispose ou pas des labels des données d'apprentissage (Chaovalit et Zhou - 2005) ;
- Approches paramétriques ou non-paramétriques, selon que les distributions des données conditionnellement aux classes sont connues ou pas (Sahoo et Kumar - 2012; Sampat et al. - 2005) ;
- Dans le cas non-paramétrique, on peut chercher à estimer les distributions des données conditionnellement aux classes ou bien on peut chercher à déterminer des frontières de décision séprant les classes (Liu et al. - 2011)

Malgré la profusion dans la littérature de méthodes de conception de classifieurs sur la base d'un ensemble d'apprentissage, la plupart d'entre eux reposent sur un schéma général en trois points essentiels (Fayyad et al. - 1996),

1. Le choix d'une structure de classifieur (arbres de décision, discriminateur linéaire...).
2. Le critère d'évaluation des performances du classifieur. Le critère le plus classique est le taux de bonne décision. D'autres facteurs peuvent également être pris en considération comme par exemple, le temps d'exécution, la stabilité, l'interprétabilité ou des contraintes de performance (Grall-Maes et Beauseroy - 2009).
3. La sélection du modèle. La construction d'un classifieur s'apparente à un processus de recherche dans un espace de classifieurs. Une fois choisis une forme de représentation et un critère de choix, la construction d'un classifieur devient un problème d'optimisation où l'algorithme de recherche est utilisé pour trouver un ensemble de paramètres qui optimise ce critère.

Ce chapitre est principalement dédié aux problèmes du prétraitement des données et aux méthodes de construction de classifieurs. La section II.2 est consacrée à la détermination d'espaces de représentation efficaces et, plus particulièrement à deux techniques, l'extraction d'attributs et la sélection d'attributs. La section II.3 détaille les quatre familles d'approches présentées dans la littérature pour la construction de classifieurs. Les méthodes d'ensemble sont brièvement présentées dans la section II.4 et une dernière section détaillera quelques critères pour l'évaluation de classifieurs.

II.2 Prétraitement des données

Comme nous l'avons déjà mentionné, le prétraitement des données constitue une étape préliminaire à la classification. En général, il comporte schématiquement deux points : la sélection d'un échantillon d'exemples pertinents et la définition d'un espace de représentation. La sélection d'exemples à partir d'un grand nombre de données est un processus d'optimisation tentant de minimiser la taille de l'échantillon en préservant les performances de classification (Liu et Metoda - 2001). Il existe beaucoup de techniques pour ce problème, dont la plus connue est *l'échantillonage aléatoire* (Reinartz -

2002) qui sélectionne au hasard un sous-ensemble d'exemples et *l'échantillonage stratifié* qui est applicable lorsque la distribution des classes n'est pas uniforme. D'autres techniques pour traiter des données déséquilibrées peuvent être trouvées dans Japkowicz et Stephen (2002).

Chaque exemple sélectionné est décrit par un nombre d'attributs relativement exhaustifs qui définissent un espace de représentation initial $\mathcal{X}^m$. Pour tout problème d'apprentissage de règle de décision et pour certains problèmes de caractérisation, l'espace de représentation définit la perception disponible du système à surveiller. Dans de nombreux cas on peut être confronté à un espace de représentation de très grande dimension, voire de dimension supérieure à la taille de l'échantillon. On parle alors du fléau de la dimension (en anglais *curse of dimensionality*) (Devijver et Kittler - 1982) et il est donc souvent utile, voire nécessaire de limiter le nombre d'attributs utilisés dans un classifieur pour acquérir un modèle robuste avec une bonne prédiction et moins de calculs. L'objectif de la sélection d'un espace de représentation est alors de déterminer un nouvel espace de représentation $\mathcal{X}^d$ de dimension $d \leq m$ en cherchant à préserver le potentiel discriminant des attributs initiaux (Liu et Motoda - 1998).

Pour résoudre ce problème de réduction de dimension, on distingue en général deux types d'approches (Liu et Motoda - 1998; Torkkola - 2003)

- l'extraction d'attributs qui réduit la dimension par une transformation (linéaire ou non-linéaire) du vecteur d'attributs de départ ;
- la sélection d'attributs qui consiste à sélectionner un nombre $d \leq m$ d'attributs dans l'ensemble $\mathcal{X}^m$ de départ.

Dans les deux cas, il s'agit de conserver au mieux l'information discriminante et de supprimer le bruit, les attributs inutiles, redondants...

Bien que, dans les travaux présentés ici, nous n'ayons eu recours qu'à la sélection d'attributs, nous passons brièvement en revue les principales techniques d'extraction d'attributs.

II.2.1 Extraction d'attributs

L'extraction d'attributs consiste à déterminer un ensemble restreint de mesures discriminantes par une transformation T sur les observations : $\boldsymbol{z} = T(\boldsymbol{x})$, $\boldsymbol{z} \in \mathcal{X}^d$ et $\boldsymbol{x} \in \mathcal{X}^m$, qui préserve au mieux la séparation des classes de l'espace initial. L'extraction d'attributs est aussi nommé la génération d'attributs, la construction d'attributs ou la transformation d'attributs dans la littérature. On peut avoir recours à une telle approche lorsque la perte d'interprétation des attributs initiaux ainsi engendrée n'est pas cruciale. Toutefois, dans le cas où il est essentiel de conserver certains attributs initiaux ou si le nombre d'attributs non pertinents dépasse le nombre d'attributs pertinents, on préférera opter pour une approche de sélection d'attributs au sein de l'ensemble initial (Torkkola - 2003).

Le but de l'extraction d'attributs est de trouver une transformation optimale T^* qui extrait un nouveau vecteur d'attributs de dimension la plus faible possible et satisfaisant un critère de performance choisi à l'avance, comme par exemple, la probabilité d'erreur (Beauseroy - 2007). Un critère optimal possible pour la classification est le

risque de Bayes appliqué dans l'espace transformé, ce qui peut être s'écrire,

$$E[Z] = \int_{\boldsymbol{z}} p(\boldsymbol{z})(1 - \max_i(p(y_i|\boldsymbol{z})))d\boldsymbol{z} \tag{II.1}$$

où $\boldsymbol{z}$ représente le vecteur d'attributs d'un exemple dans l'espace transformé et y_i est le label de classe associée. Ce critère exige de connaître la distribution a posteriori des classes qui est difficile à obtenir en pratique avec uniquement un ensemble d'apprentissage. Cependant, il est possible d'utiliser le taux d'erreur d'approximation basé sur la borne de Bhattacharyya (*Bhattacharyya bound*) ou basé sur un critère de divergence inter-classes (Devijver et Kittler - 1982; Guorong et al. - 1996; Saon et Padmanabhan - 2001).

Il existe une très grande diversité de méthodes d'extraction d'attributs qui peuvent entrer dans le cadre très général de la *projection pursuit* définit par (Friedman et Tukey - 1974). Parmi celles-ci, deux grandes familles peuvent être différenciées, les méthodes linéaires et les méthodes non-linéaires. Cette très brève présentation introduit quelques éléments concernant chacune de ces approches.

II.2.1.1 Méthodes linéaires

Les méthodes linéaires consistent à déterminer une matrice de projection $\boldsymbol{W}$ qui permet de transformer l'observation de départ $\boldsymbol{x}$ vers un nouveau vecteur d'attributs $\boldsymbol{z}$ par projection $\boldsymbol{z} = \boldsymbol{W}^t\boldsymbol{x}$

Une méthode connue est l'analyse en composante principale (*Principal Component Analysis*, PCA)(Devijver et Kittler - 1982; Fukunaga - 1990). La matrice de projection $\boldsymbol{W}$ est formée par les vecteurs propres correspondant aux plus grandes valeurs propres de la matrice de covariance pour les données de toutes les classes. PCA recherche à représenter un ensemble des données le plus fidèlement possible dans un sous-espace en minimisant l'erreur quadratique moyenne entre les données projetées et les données originales, sans tenir compte de la discrimination entre les classes.

L'analyse factorielle discriminante ou l'analyse discriminante linéaire (*Linear Discriminant Analysis*, LDA) produit une transformation qui est optimale dans certains cas pour la discrimination (Fisher - 1936; Fukunaga - 1990). LDA trouve les vecteurs propres de $(\boldsymbol{S}_w^{-1}\boldsymbol{S}_b)$, où $\boldsymbol{S}_w$ est la matrices de variance-covariance intra-classes et $\boldsymbol{S}_b$ est la matrice de variance-covariance inter-classes. La matrice $\boldsymbol{S}_w^{-1}$ capte la compacité de chaque classe, et $\boldsymbol{S}_b$ représente la dispersion des moyennes des classes. Les vecteurs propres correspondant aux plus grandes valeurs propres de $(\boldsymbol{S}_w^{-1}\boldsymbol{S}_b)$ forment les colonnes de la matrice de projection $\boldsymbol{W}$. LDA n'utilise que les statistiques du second-ordre (i.e. les variances-covariances), elle est donc optimale dans le cas gaussien. Par ailleurs, le rang maximum de $\boldsymbol{S}_b$ est de $K-1$, où K est le nombre de classes différentes. LDA ne peut donc pas produire plus de $K-1$ attributs. Bien que des extensions aient été proposées pour traiter le dernier problème (Okada et Tomita - 1985), celui de la sous-optimalité dans le cas gaussien demeure.

On peut noter également quelques approches reposant sur l'analyse en composante indépendante (*Independent Component Analysis*, ICA), (Girolami et al. - 1998; Yang et Moody - 2000).

II.2.1.2 Méthodes non-linéaires

L'introduction de fonctions noyaux permet de se ramener à des méthodes linéaires dans le cas non-linéaire. Les fonctions noyaux, sous réserve qu'elles soient définies positives, définissent implicitement un produit scalaire dans un espace de Hilbert de grande dimension (Aronszajn - 1950). Les transformations sont donc recherchées dans cet espace de grande dimension. Cette opération peut être réalisée sans expression définitive de la fonction qui permet de déployer les données dans l'espace de grande dimension puisque seuls les produits scalaires sont nécessaires pour déterminer la solution. Par exemple, les extensions des approches de l'analyse en composantes principales (*Kernel Principal Component Analysis*) (Schölkopf et al. - 1998b, 2001; Mika et al. - 1999b), de l'anaylse factorielle discriminante (Mika et al. - 1999a, 2000, 2001; Yang et al. - 2005) et de la détermination du discriminateur optimal au sens d'un critère de contraste ont été développés (Richard et al. - 2001; Lengellé - 2002).

Torkkola (2003) présente une nouvelle approche d'extraction d'attributs en maximisant le critère d'information mutuelle $MI(\boldsymbol{z}_i, y_i)$ pour les vecteurs d'attributs transformés $\boldsymbol{z}_i$ (linéaires ou non-linéaires) et les labels y_i par la méthode du gradient. Une mesure de divergence quadratique qui permet de faire une estimation non-paramétrique est employée.

Dans le cas où chaque attribut est associé à un capteur, la perte ou l'altération de certains capteurs opérée par l'environnement, s'apparente davantage à une sélection d'attributs.

II.2.2 Sélection d'attributs

La sélection d'attributs consiste à sélectionner un sous-ensemble d'attributs de l'ensemble de départ sans transformation. L'objectif est d'identifier et d'exclure le plus possible d'attributs non pertinents et redondants (Yu et Liu - 2004). En général, les attributs sont caractérisés comme :

- Pertinent : Les attributs pertinents influent sensiblement sur les sorties du classifieur.
- Non pertinent : Les attributs non pertinents n'ont aucune influence sur les sorties du classifieur.
- Redondant : Une redondance existe chaque fois qu'un attribut peut se substituer à un autre.

La sélection de d attributs parmi m en effectuant une recherche exhaustive requiert l'examen de C_m^d sous-ensembles d'attributs. De nombreuse approches basées sur des algorithmes de recherche heuristique ou aléatoire ont été proposées en tentant de réduire la complexité de calcul. Ces approches ont besoin d'un critère d'arrêt afin d'éviter une recherche exhaustive. Le schéma dans la figure II.2 expose un cadre traditionnel de la sélection d'attributs, qui comporte trois étapes principales suivantes (Liu et Motoda - 1998; Yu et Liu - 2004; Dash et Liu - 1997)

1. Le processus de génération a pour but de proposer un sous-ensemble d'attributs candidat selon une certaine stratégie de recherche (Langley - 1994; Siedlecki et

Sklansky - 1988). Le processus de génération peut se dérouler selon 3 principes :

i) méthode asendante : on part de 0 attribut et on inclut successivement des attributs,

ii) méthode descendante : on part de l'ensemble complet des attributs et on les exclut successivement,

iii) on tire au hasard un sous-ensemble d'attributs.

2. Chaque sous-ensemble candidat est évalué au moyen d'un critère de qualité. La valeur du critère est comparée à celle obtenue avec le candidat précédent. Si elle est meilleure que l'ancienne, le sous-ensemble précédent va être remplacé par le nouveau.
3. Enfin, on doit décider de la condition d'arrêt de la recherche. Par exemple, on peut cesser d'inclure ou d'exclure des attributs si aucun attribut alternatif n'améliore la performance de la classification, on peut continuer à réviser l'ensemble des attributs aussi longtemps que la performance ne se dégrade pas ou bien on peut continuer de générer des ensembles de candidats jusqu'au bout de l'espace de recherche et puis sélectionner le meilleur.

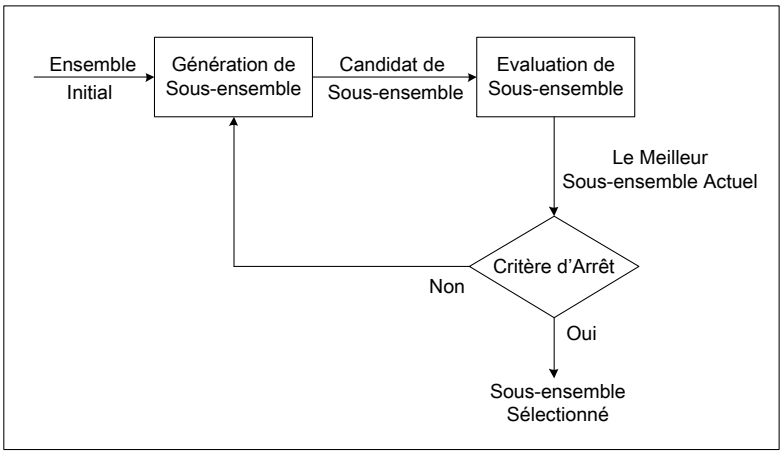


Figure II.2 – Le schéma d'un cadre traditionnel de la sélection d'attributs

Un sous-ensemble optimal dépend toujours de la fonction d'évaluation utilisée. Généralement, une fonction d'évaluation cherche à mesurer la capacité discriminante d'un attribut ou d'un sous-ensemble. On peut distinguer cinq types de fonctions d'évaluation (Dash et Liu - 1997) : la mesure de distance, la mesure d'information ou d'incertitude, la mesure de dépendance, la mesure de cohérence et la mesure de taux d'erreur de classification.

Blum et Langley (Blum et Langley - 1997) classent un grand nombre d'approches de sélection d'attributs en trois grands types de méthodes selon la dépendance à l'algorithme d'apprentissage : *filtre* qui filtrent les attributs avant l'algorithme d'apprentissage, *wrapper* qui utilsent l'algorithme d'apprentissage comme fonction d'évaluation et les méthodes embarquées. Nous allons brièvement passer en revue ces méthodes dans les sections suivantes.

II.2.2.1 Méthodes de *filtre*

Une première approche possible consiste à sélectionner les attributs avant la phase d'apprentissage et donc indépendamment de celle-ci. John et al. (1994) ont nommé ces méthodes, méthodes de filtre parcequ'elles consistent à filtrer les attributs non pertinents avant l'algorithme d'apprentissage. Le système le plus simple de filtre est d'évaluer chaque attribut individuellement en fonction de sa corrélation avec la fonction d'évaluation utilisée (par exemple, en utilisant une mesure d'information mutuelle), puis de sélectionner les d attributs qui ont les scores plus élevés. Le meilleur choix de d peut alors être déterminé par une approche *holdout*. Cette méthode est ordinairement appliquée aux tâches de catégorisation de texte (Lewis - 1992b,a).

FOCUS (Almuallim et Dietterich - 1991) et *Relief* (Kira et Rendell - 1992a,b) sont deux approches bien connues dans les méthodes de filtre. *FOCUS*, initialement définie pour le cas de données Booléennes non bruitées, consiste à examiner tous les sous-ensembles d'attributs pour en sélectionner un de dimension minimale qui soit suffisant pour discriminer les classes. Almuallim et Dietterich (1991) ont appliqué *FOCUS* à l'algorithme d'apprentissage d'un arbre de décision et comparé sa performance avec celle d'un arbre de décision traditionnel (p. ex. ID3). Les résultats expérimentaux ont démontré que *FOCUS* est plus performante que ID3 seul quand on prend en compte un nombre croissant d'attributs non pertinents.

Relief est une autre approche qui consiste à assigner un poids de pertinence à chaque attribut qui indique la pertinence entre l'attribut et la cible (par exemple un label de classe). Des exemples de l'ensemble d'apprentissage sont échantillonnés au hasard puis cette valeur de pertinence est mise à jour sur la base de la différence entre l'exemple sélectionné et les deux exemples les plus proches de la même classe et de la classe opposée (nommées « near-hit » et « near-miss »). D'après les expérimentations de Kira et Rendell (1992a,b) où *FOCUS* et *Relief* sont appliquées à ID3, *Relief* est considérablement plus rapide que la recherche exhaustive (*FOCUS*) et plus performante que la recherche heuristique. De plus, elle est plus efficace pour le cas bruité et l'interaction d'attributs. Toutefois, il faut noter que *Relief* devient inopérant lorsqu'il y a des attributs redondants.

Il existe d'autres exemples de méthodes de filtre appliquées à d'autres méthodes de classification. Par exemple, Cardie (1993) et Kubat et al. (1993) utilisent l'arbre de décision comme méthode de filtre pour sélectionner des attributs dans l'algorithme des plus proches voisins et aussi pour le classifieur bayésien naïf. Singh et Provan (1996) emploient certaines mesures basées sur la théorie de l'information pour filtrer des attributs afin de construire un réseau bayésien et Koller et Sahami (1996) utilisent une mesure d'entropie croisée dans la construction d'un classifieur bayésien naïf et d'un arbre de décision.

Un des point faibles des méthodes de filtre est que les sous-ensembles d'attributs sélectionnés peuvent avoir un score de pertinence élevé, même quand ils sont inappropriés pour la méthode de classification utilisé. Comme l'ont montré Kohavi et John (1997), l'ensemble d'attributs idéal dépend de la méthode utilisé par la suite, ce qui constitue un argument de poids pour justifier l'inclusion de l'algorithme d'apprentissage dans le processus de sélection d'attributs et qui conduit au développement des

méthodes de *wrappers* ou des méthodes embarquées.

II.2.2.2 Méthodes de type *wrapper*

Les méthodes de type *wrapper* réalisent la sélection d'attributs en utilisant directement l'algorithme d'apprentissage pour mesurer les performances de différents sous-ensembles d'attributs et sélectionner le meilleur, sans intégrer la connaissance sur la structure spécifique de la fonction de classification ou de régression. L'algorithme d'apprentissage est donc considéré comme une boîte noire fournissant un classifieur en fonction des exemples donnés et la sélection s'effectue autour de cette boîte noire. Certains chercheurs étudient l'intérêt d'une telle approche avec en interne différentes méthodes, par exemple une méthode de type arbre de décision (Doak - 1992; John et al. - 1994; Caruana et Freitag - 1994), des plus proches voisins (Langley et Sage - 1994) ou support vector machines (Yu et Cho - 2003). Plusieurs parcours de l'espace des sous-ensembles d'attributs sont étudiés. Ils ont observé expérimentalement que, dans les problèmes réels et pratiques, les méthodes de type *wrapper* peuvent améliorer considérablement la performance du système décisionnel.

L'inconvénient majeur des méthodes de type *wrapper* par rapport aux méthodes de *filtre* est le coût en temps de calcul, qui résulte de la réponse de l'algorithme d'apprentissage dont les paramètres doivent aussi être optimisés pour chaque sous-ensemble d'attributs candidat. La métrique la plus couramment utilisée pour savoir si un sous-ensemble d'attributs permet d'obtenir un bon classifieur est d'effectuer une validation croisée de cette méthode sur ce sous-ensemble d'attributs (John et al. - 1994). La validation croisée $n\text{-}fold$ (Breiman et al. - 1984; Weiss et Kulikowski - 1991), par exemple, divise un ensemble d'apprentissage en n partitions de tailles à peu près égales. L'algorithme d'apprentissage est ensuite exécuté n fois, en utilisant à chaque fois $n-1$ partitions comme ensemble d'apprentissage et la partition restante comme ensemble de test. La précision estimée finale résulte de la moyenne sur les n essais. Cela implique de relancer n fois l'algorithme pour chaque sous-ensemble d'attributs testé et la complexité de calcul en est augmentée d'autant. Cet inconvénient a conduit certains chercheurs à développer des techniques ingénieuses pour accélérer le processus d'évaluation.

II.2.2.3 Méthodes embarquées

Contrairement aux méthodes de type *wrapper*, les méthodes dites embarquées (*embedded methods*) opèrent la sélection d'attributs en même temps que la mise au point du classifeur (Blum et Langley - 1997). Dans ce cas, la structure de l'algorithme d'apprentissage joue un rôle crucial dans le principe des méthodes utilisées puisqu'elles utilisent un critère d'évaluation spécifiquement associé aux paramètres de l'algorithme d'apprentissage considéré. Par exemple, Weston et al. (2001) mesurent l'importance d'un attribut en utilisant une borne supérieure de la probabilité d'erreur qui est seulement valide pour les SVM.

En outre, un exemple particulier de méthodes embarquées, souvent nommées méthodes d'attributs pondérés (en anglais *feature-weighting methods*) dans la littérature, consiste à appliquer une fonction de pondération aux attributs en assignant un de-

gré de pertinence perçue de chaque attribut. La méthode d'attributs pondérés la plus connue est la règle du perceptron (Minsky et Papert - 1969), qui ajoute ou ôte des poids à l'unité d'intégration linéaire en réponse aux erreurs commises sur l'ensemble d'apprentissage. Par ailleurs, la méthode de *least-mean squares* (Widrow et Hoff - 1960) appliqué aux unités linéaires, qui est généralisé par les réseaux de neurones multicouches, entraîne aussi des modifications des poids afin de réduire l'erreur sur l'ensemble d'apprentissage.

II.3 Construction des classifieurs

Compte tenu de leur importance dans les méthodes d'ensemble qui font l'objet du travail de cette thèse, les arbres de décision sont présentés en détail dans cette partie bien que nous ne les ayons pas utilisés dans le cadre de nos travaux.

A l'issue de la détermination des exemples utilisés et de la sélection d'un sous-espace de représentation, le choix d'un algorithme d'apprentissage pour construire un classifieur est une étape cruciale. Ces dernières années, de nombreux algorithmes d'apprentissage ont été proposés et étudiés dans la littérature basés sur des théories différentes telles que la théorie de la décision de Bayes, la théorie de Vapnik-Chervonenkis, etc. Kotsiantis et al. (2006) proposent de les classer en quatre familles : les algorithmes logiques, les algorithmes basés sur le perceptron, les algorithmes statistiques et les algorithmes SVM (Support Vecteur Machine). Nous allons présenter quelques exemples pour chaque famille.

II.3.1 Algorithmes logiques

Les algorithmes logiques d'apprentissage construisent un système de décision par le raisonnement inductif à partir des exemples. Leur intérêt particulier est d'être compréhensibles et interprétables. Dans ce sens nous nous concentrons sur deux types des méthodes : les arbres de décision et les algorithmes à base de règles.

II.3.1.1 Arbres de décision

L'arbre de décision est un classifieur décrit par une structure arborescente, qui a été largement employé pour représenter des modèles de classification en raison de sa capacité à décomposer un processus complexe de décision en plusieurs sous-décisions plus simples, fournissant ainsi une solution qui est souvent plus facile à interpréter. Les arbres de décision présentent certains avantages par rapport aux autres algorithmes d'apprentissage, tels que le faible coût de calcul lors de la construction du modèle et la capacité à traiter les attributs redondants. Par ailleurs, le modèle construit présente généralement une bonne capacité de généralisation (Han et Kamber - 2001; Tan et al. - 2005).

La construction d'un arbre de décision consiste en général à élaborer une structure d'arbre qui comporte quatre formes de base :

- Structure ascendante (*bottom-up*) (Landeweerd et al. - 1983),
- Structure descendante (*top-down*) (Quinlan - 1986, 1993; Breiman et al. - 1984),
- Structure de type *hybride*, qui utilise une procédure de *bottom-up* pour diriger et aider une procédure de *top-down* (Kim et Landgrebe - 1990).
- Structure de *tree growing-pruning*, qui permet de faire croître un arbre jusque' à sa taille maximale et puis de lélaguer sélectivement.

La plupart des algorithmes reposant sur des arbres de décision sont bâtis sur une structure *top-down*, comme par exemple ID3 (Quinlan - 1986), C4.5 (Quinlan - 1993) et CART (Breiman et al. - 1984). Nous nous intéresserons particulièrement à cette structure dans la suite.

Le critère de segmentation dans la structure *top-down* est important, il permet de sélectionner un ensemble d'attributs à chaque noeud interne permettant de segmenter l'espace des exemples en deux ou plusieurs sous-espaces conformément aux valeurs d'attributs sélectionnés. Dans la pratique en raison du temps de calcul, le nombre d'attributs utilisés à chaque noeud interne est généralement inférieur au nombre d'attributs disponibles (You et Fu - 1976).

II.3.1.1.1 Critères de segmentation

Comme nous l'avons évoqué précédemment, les critères de segmentation se distinguent selon deux types : univariés (i.e. un seul attribut sélectionné) et multivariés (i.e. un ensemble d'attributs sélectionnés). Les critères univariés consistent à trouver le meilleur attribut associé à un noeud interne, permettant de segmenter l'espace d'exemples situé à ce noeud. Ben-Bassat (1982) distingue trois catégories de critères univariés : les critères issus de la théorie de l'information, les critères issus de la mesure de distance et les critères issus de mesures de dépendance. Ces catégories sont parfois arbitraires et non distinctes. Certaines mesures appartenant aux différentes catégories peuvent être considérées comme des approximations l'une de l'autre. Nous présentons maintenant quelques-uns de ces critères.

- **Gain d'information.** Il s'agit d'un critère de segmentation utilisant une mesure d'entropie (Quinlan - 1986). Soit $\mathcal{A}_n^{(t)} = \{ (\boldsymbol{x}_1 \ , \ y_1), \ (\boldsymbol{x}_2 \ , \ y_2), \ \ldots, (\boldsymbol{x}_n \ , \ y_n) \}$ un ensemble d'apprentissage de n exemples distribués dans K classes à un noeud t et $p_k^{(t)}$ la proportion d'exemples dans la classe k. Chaque exemple est décrit par un vecteur d'attributs $\boldsymbol{X} = [X_1 \ , \ X_2 \ , \ \ldots \ , \ X_d]^t$. Si l'attribut X_i, sélectionné au noeud t, peut prendre $|X_i|$ valeurs, alors l'ensemble d'apprentissage $\mathcal{A}_n^{(t)}$ sera segmenté en $\{\mathcal{A}_{n,1}^{(t)}, \ldots, \mathcal{A}_{n,s}^{(t)}, \ldots, \mathcal{A}_{n,|X_i|}^{(t)}\}$, où $\mathcal{A}_{n,s}^{(t)}$ est l'ensemble des exemples de $\mathcal{A}_n^{(t)}$ pour lesquels l'attribut X_i prend la valeur $x_{i,s}$. La mesure d'entropie est alors définie par,

$$E(\mathcal{A}_n^t) = \sum_{k=1}^{K} -p_k^{(t)} \log_2(p_k^{(t)}) \tag{II.2}$$

Le gain d'information est alors exprimé comme,

$$GI(X_i, \mathcal{A}_n^t) = E(\mathcal{A}_n^t) - \sum_{s=1}^{|X_i|} \frac{|\mathcal{A}_{n,s}^t|}{|\mathcal{A}_n|} \cdot E(\mathcal{A}_{n,s}^t) \tag{II.3}$$

Tous les attributs candidats sont examinés et nous choisissons celui qui maximise le gain $GI(X_i, \mathcal{A}_n^t)$.

- **Ratio de gain.** Cette mesure est proposé par Quinlan (1993) qui normalise le gain d'information de la manière suivante,

$$RG(X_i, \mathcal{A}_n^t) = \frac{GI(X_i, \mathcal{A}_n^t)}{IV(X_i, \mathcal{A}_n^t)} \tag{II.4}$$

avec

$$IV(X_i, \mathcal{A}_n^t) = \sum_{s=1}^{|X_i|} -\frac{\left|\mathcal{A}_{n,s}\right|}{|\mathcal{A}_n^t|} \log_2 \frac{\left|\mathcal{A}_{n,s}\right|}{|\mathcal{A}_n^t|} \tag{II.5}$$

Les expérimentations (Quinlan - 1988) ont déjà montré que le critère du ratio de gain construit un arbre de décision relativement plus petit lorsque tous les attributs sont binaires. De plus, ce critère donne un arbre de décision ayant une plus grande capacité prédictive dans le cas où beaucoup d'attributs ayant le plus de valeurs possibles sont des attributs redondants.

- **Indice de Gini.** C'est un critère mesurant la divergence entre les distributions de probabilité des labels de classes dans l'ensemble d'apprentissage $\mathcal{A}_n^t$ au noeud t (Breiman et al. - 1984; Gelfand et al. - 1991). L'indice de Gini est estimé par,

$$Gini(\mathcal{A}_n^t) = 1 - \sum_{k=1}^{K} \left(p_k^{(t)} \right)^2 \tag{II.6}$$

Par conséquent, le critère d'évaluation pour la sélection d'attribut X_i est donné par,

$$GiniGain(X_i, \mathcal{A}_n^t) = Gini(\mathcal{A}_n^t) - \sum_{s=1}^{|X_i|} \frac{\left|\mathcal{A}_{n,s}^t\right|}{|\mathcal{A}_n|} \cdot Gini(\mathcal{A}_{n,s}^t) \tag{II.7}$$

Pour les critères multivariés, plusieurs attributs participent à un test de segmentation à un noeud interne. Trouver le meilleur critère multivarié est évidemment plus compliqué que de trouver le meilleure critère univarié. En outre, bien que ce type des critères puisse améliorer considérablement la performance de l'arbre de décision, ils sont beaucoup moins populaires que les critères univariés. La plupart des critères multivariés reposent sur une combinaison linéaire des attributs disponibles. Trouver la meilleure combinaison linéaire peut être effectuée par la recherche gloutonne (Breiman et al. - 1984; Murthy et al. - 1994), l'optimisation linéaire (Duda et Hart - 1973; Bennett et Mangasarian - 1994), l'analyse discriminante linéaire (Duda et Hart - 1973; Friedman - 1977; Sklansky et Wassel - 1981; Lin et Fu - 1983; Loh et Vanichsetakul - 1988; John - 1996) ou encore (Utgoff - 1989; Lubinsky - 1994; Sethi et Yoo - 1994).

II.3.1.1.2 Critères d'arrêt

La phase de croissance d'un arbre continue jusqu'à ce qu'un critère d'arrêt soit validé. Les conditions suivantes constituent des règles d'arrêt assez communes :

- Tous les exemples dans l'ensemble d'apprentissage à un noeud interne appartiennent à une seule classe ;

- La profondeur maximale de l'arbre a été atteinte ;
- Le nombre de cas dans un noeud terminal est inférieur au nombre minimum de cas définis pour un noeud parent ;
- Le résultat du meilleur critère de segmentation n'est plus supérieure à un seuil fixé.

II.3.1.1.3 Méthodes d'élagage

Il est souvent souhaitable de construire un arbre de décision moins complexe, car la complexité peut influer sérieusement sur la capacité de généralisation de l'arbre (Breiman et al. - 1984). La complexité d'un arbre est habituellement mesurée par l'une des métriques suivantes : le nombre total de noeuds, le nombre total de feuilles, la profondeur de l'arbre et le nombre d'attributs utilisés. Elle dépend explicitement du critère d'arrêt utilisé et de la méthode d'élagage. Un critère d'arrêt serré engendrera un petit (*underfitting*) arbre, mais un critère d'arrêt lâche engendrera un grand arbre et une situation de sur-apprentissage. Les méthodes d'élagage proposées initialement par Breiman et al. (1984) ont été développées pour résoudre ce dilemme. Conformément à cette méthodologie, un critère d'arrêt lâche est d'abord utilisé, puis cet arbre est élagué en enlevant les branches qui ne contribuent pas à la précision de généralisation. De nombreuse études ont illustré que l'utilisation des méthodes d'élagage permettent d'améliorer la performance de généralisation d'un arbre de décision, en particulier dans les situations bruités.

La méthode d'élagage proposée par Breiman et al. (1984), *cost complexity pruning* (également appelé *weakest link pruning* ou *error complexity pruning*) se déroule en deux étapes. Dans la première étape, une séquence d'arbres de plus en plus petits est construite sur la base des données d'apprentissage. Dans la deuxième étape, l'un de ces arbres est choisi comme l'arbre élagué, en fonction de l'erreur de classification sur un ensemble d'élagage. L'ensemble d'élagage est une portion de l'ensemble d'apprentissage, qui est réservée exclusivement à l'élagage. L'utilisation d'un ensemble séparé d'élagage est une pratique assez courante.

Beaucoup de chercheurs ont étudié l'efficacité des méthodes d'élagage existantes (Cohen - 1993; Esposito et al. - 1995; Mingers - 1989) et aucune méthode d'élagage n'apparaît comme absolument meilleure que les autres. Le choix d'une méthode d'élagage dépend donc de facteurs tels que la taille de l'ensemble d'apprentissage et la disponibilité des données supplémentaires pour l'élagage.

II.3.1.1.4 Avantages et inconvénients

Les arbres de décision sont intéressants comme outil de classification pour les raisons suivantes,

- Les arbres de décision sont explicites et ils peuvent être convertis en une combinaison d'un ensemble de règles. Cette représentation est ainsi considérée comme compréhensible ;
- Les arbres de décision sont capables de traiter des attributs nominaux et numériques ;

- Les arbres de décision sont capables de traiter des données contenant des erreurs et des valeurs manquantes.
- Les arbres de décision sont considérés comme une méthode non paramétrique, donc qui ne nécessitent aucune hypothèse sur la distribution des données et sur la structure du classifieur.

En revanche, les arbres de décision ont quelques inconvénients tels que :

- La situation de chevauchement (si au moins une classe est en commun pour deux noeuds internes) peut faire que le nombre de noeuds terminaux soit beaucoup plus grand que le nombre de classes actuelles et augmenter donc le temps de recherche et l'espace de mémoire exigé, en particulier pour le cas contenant un grand nombre de classes ;
- Des erreurs peuvent s'accumuler de niveau en niveau dans un grand arbre. Wu et al. (1975) indiquent que la précision et l'efficacité ne peuvent pas être optimisées simultanément. Une borne de l'efficacité doit être satisfaite pour une précision donnée ;
- Enfin, il est difficle de construire un arbre de décision optimal (Hancock et al. - 1996; Hyafil et Rivest - 1976; Naumov - 1991). La performance d'un arbre dépend fortement de la façon dont cet arbre est construit.

II.3.1.2 Méthodes à base de règles

Les méthodes à base de règles utilisent une stratégie *separate-and-conquer* qui peut remonter au système d'apprentissage (Michalski - 1969). Ils peuvent aussi être nommés algorithmes de *covering* (Pagallo et Haussler - 1990). Tous ces algorithmes partagent la même structure : un algorithme de *separate-and-conquer* recherche une règle qui explique une partie des exemples de l'ensemble d'apprentissage, exclut ces exemples, puis traite les exemples restants en apprenant davantage de règles et ainsi de suite jusqu'à ce qu'il ne reste plus aucun exemple. Cela assure que chaque exemple dans l'ensemble d'apprentissage initial est couvert par une règle au moins. En bref, l'objectif des algorithmes à base de règles est de construire un ensemble de règles qui sont compatibles avec les données d'apprentissage.

Les algorithmes d'arbre de décision présentés précédement en utilisant une stratégie *divide-and-conquer* (Quinlan - 1986) peuvent aussi facilement se transformer en un ensemble de règles en générant une règle pour chaque passage de la racine à une feuille.

La différence entre les deux est que les arbres de décision évaluent la qualité moyenne d'un nombre d'ensembles disjoints (un ensemble correspond à une valeur de l'attribut testé), tandis que les méthodes à base de règles évaluent la qualité d'un ensemble d'exemples couverts par une règle candidate.

II.3.2 Algorithmes du Perceptron

II.3.2.1 Perceptron (linéaire)

Un autre type de méthode d'apprentissage bien connu repose sur le principe de l'algorithme du perceptron. Le perceptron est un classifieur linéaire proposé initialement par Rosenblatt (1958, 1962), qui peut être brièvement décrit comme suit :

Considérons un exemple $\boldsymbol{x} = [x_1\ ,\ x_2\ ,\ \ldots\ ,\ x_d]^t$ et $\boldsymbol{w} = [w_1\ ,\ w_2\ ,\ \ldots\ ,\ w_d]^t$ un vecteur de poids associés aux attributs, aussi nommé vecteur de prédiction (généralement les valeurs des poids sont prises sur l'intervalle $[-1, 1]$). Le perceptron calcule la somme pondérée des attributs $\sum_{i=1}^{d} w_i x_i$ (sortie du perceptron) qui est ensuite comparée à un seuil θ pour obtenir l'estimation du label $\hat{y}$ de $\boldsymbol{x}$, par exemple au moyen de la fonction signe comme suit

$$Z = \boldsymbol{w} \cdot \boldsymbol{x} - \theta = \sum_{i=1}^{d} w_i x_i - \theta \tag{II.8}$$

avec

$$\hat{y} = sign(Z) = \begin{cases} 0 & \text{si} \quad Z < 0 \\ 1 & \text{sinon} \end{cases} \tag{II.9}$$

En général, le moyen le plus commun pour la construction des classifieurs en utilisant l'algorithme du perceptron est d'exécuter l'algorithme à plusieurs reprises sur un ensemble d'apprentissage jusqu'à ce qu'un vecteur optimal de prédiction permettant à tous les exemples de l'ensemble d'apprentissage d'être correctement classés soit trouvé. Par exemple, on peut prendre comme vecteur initial de prédiction un vecteur nul $\boldsymbol{w} = \boldsymbol{0}$. Le label $\hat{y}$ d'un exemple $\boldsymbol{x}$ est obtenu d'après les équations II.8 et II.9 et si cette prédiction $\hat{y}$ diffère du label vrai y, le vecteur de prédiction est mis à jour selon la formule $\boldsymbol{w} = \boldsymbol{w} + y\boldsymbol{x}$, sinon $\boldsymbol{w}$ n'est pas modifié. Le processus se répète sur tout l'ensemble d'apprentissage pour trouver le vecteur optimal de prédiction. Ce vecteur optimal de prédiction est ensuite utilisé pour prédire les labels inconnus de nouveaux exemples.

WINNOW (Littlestone et Warmuth - 1994) est l'une de ces techniques pour construire un classifieur linéaire basée sur le principe du perceptron. Les résultats expérimentaux (Blum - 1997) montrent que WINNOW peut s'adapter rapidement aux changements d'espaces de représentation (Cf. point N° 2 page 7). Un nombre d'algorithmes similaires à WINNOW ont été développés, tels que ceux proposés par Auer et Warmuth (1998).

Freund et Schapire (1999) ont proposé une nouvelle méthode linéaire en employant l'algorithme du perceptron en ligne, nommé le vote-perceptron, qui est une variante de la méthode de *leave-one-out* proposée par Helmbold et Warmuth (1995). Dans l'algorithme du vote-perceptron, davantage d'informations sont mémorisées au cours de la phase d'apprentissage et ces informations sont ensuite utilisées pour obtenir de meilleures prédictions sur les données de test. Les informations que nous conservons constituent une liste de tous les vecteurs de prédiction générés au cours de l'apprentissage. Pour chaque vecteur de prédiction dans cette liste, le nombre d'exemples mal classés dans l'ensemble d'apprentissage est compté et ce compte sert de poids à ce

vecteur de prédiction. Finalement pour un nouvel exemple à prédire, toutes les prédictions binaires sont d'abord calculées en utilisant tous les vecteurs de prédiction conservés et puis la décision finale est obtenue par la combinaison de ces prédictions binaires par la majorité pondérée. Les poids utilisés ici sont les effectifs d'exemples mal classés associés aux vecteurs de prédiction. Intuitivement cela correspond au fait que les bons vecteurs de prédiction tendent à survivre longtemps et ont ainsi un plus grands poids dans le processus de la majorité. Par ailleurs, suite aux travaux de Aizerman et al. (1964), les auteurs intègrent les fonctions noyaux et l'algorithme du perceptron de sorte que l'algorithme proposé puisse opérer efficacement dans les espaces de grande dimension.

En résumé, les méthodes linéaires basés sur le perceptron ont l'avantage d'offrir une faible complexité de calcul dans les cas où peu d'attributs sont pertinents. Toutefois, notons que les algorithmes du type perceptron sont des méthodes de classification binaire. Pour le problème multi-classes, il faut donc se ramener à un ensemble de problèmes de classification binaire.

II.3.2.2 Réseaux de neurones (non-linéaire)

Le perceptron présenté plus haut est en fait un réseau de neurones monocouche, qui peut uniquement classer les données linéairement séparables. Autrement dit, dans le cas où l'on peut correctement séparer l'espace des données avec une droite ou avec un hyperplan, on dit que les données sont linéairement séparables et le perceptron va trouver une bonne solution. Dans le cas contraire le perceptron devient inopérant et les réseaux de neurones artificiels multicouche constituent une alternative pour résoudre ce problème.

Un réseau de neurones multicouche se compose d'une série de couches d'unités ou neurones auxquels sont attachés des poids. Chaque neurone effectue une tâche relativement simple : recevoir les informations externes ou les sorties des neurones de la couche précédente et les utiliser pour calculer sa propre sortie (nommée activation) qui se propage aux neurones connectés de la couche suivante. Une deuxième tâche consiste à ajuster les poids. On distingue généralement trois types de couches dans un réseau de neurones :

- la couche d'entré constituée des neurones qui reçoivent les informations de l'extérieur du réseau et renvoient les résultats intermédiaires aux neurones de la couche suivante ;
- la couche de sortie qui calcule les résultats finaux et fournit la réponse du classifieur ;
- les couches cachées qui se situent entre la couche d'entrée et la couche de sortie.

Le système du réseau de neurones est naturellement parallèle car plusieurs neurones peuvent effectuer leurs opérations en même temps. Au cours de l'opération, les neurones peuvent être mis à jour de façon synchrone ou de façon asynchrone. Dans la mise à jour synchrone, tous les neurones renouvellent simultanément leurs activations. Dans le cas asynchrone, chaque neurone a une probabilité de renouvellement qui est généralement fixée à l'avance et la mise à jour s'effectue sur un seul neurone à la fois. Dans certains

cas, le dernier modèle présente plusieurs avantages.

Nous supposons que chaque neurone fournit une contribution aux neurones qui lui sont connectés. Le bilan des contributions s_k au neurone k est donné par la somme pondérée des sorties de tous les neurones qui lui sont connectés complété par un seuil θ_k, nommé aussi coefficient de biais (Cf. équation II.10).

$$s_k = \sum_j w_{jk} \cdot y_j + \theta_k \tag{II.10}$$

Dans l'équation II.10 w_{jk} est le poids qui détermine l'effet du neurone j sur le neurone k et y_j représente l'état d'activation (la sortie) du neurone j connecté au neurone k. Une contribution positive est considérée comme une excitation et une contribution négative comme une inhibition. Dans certains cas, des règles plus complexes pour combiner les contributions peuvent être utilisées, telles que celle proposée par exemple par Feldman et Ballard (1982)

$$s_k = \sum_j w_{jk} \prod_m y_{jm} + \theta_k \tag{II.11}$$

II.3.2.2.1 Fonctions d'activation

Après avoir calculé s_k, une fonction d'activation g_k est nécessaire pour déterminer l'état d'activation $\hat{y}_k$ du neurone k donné par la relation

$$\hat{y}_k = g_k(s_k) \tag{II.12}$$

Bien que beaucoup de fonctions g puissent théoriquement être utilisées comme fonction d'activation, les fonctions sigmoïdes sont les plus populaires. Elles sont mesurables et possèdent la propriété,

$$g(s) \rightarrow \begin{cases} 0 & \text{si} \quad s \rightarrow -\infty \\ 1 & \text{si} \quad s \rightarrow \infty \end{cases} \tag{II.13}$$

Les fonctions suivantes sont des choix admissibles utilisées le plus fréquemment :

- Fonction seuil

$$g(s) = sign(s) \tag{II.14}$$

- Fonction logistique

$$g(s) = \frac{1}{1 + e^{-s}} \tag{II.15}$$

- Fonction tangente hyperbolique

$$g(s) = \tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}} = \frac{e^{2s} - 1}{e^{2s} + 1} \tag{II.16}$$

- Fonction arc tangente

$$g(s) = \frac{2}{\pi} \arctan(\frac{\pi s}{2}) \tag{II.17}$$

On peut montrer qu'il existe une relation linéaire entre la fonction logistique et la fonction tangente hyperbolique. Par conséquent les prédictions produites par la fonction logistique devraient être proches de celles produites par la fonction tangente hyperbolique.

II.3.2.2.2 Topologies des réseaux de neurones

Comme nous l'avons vu, la façon de lier un neurone à un autre est de prendre la sortie du premier, d'affecter un poids à sa valeur et de la relier à l'entrée d'un autre neurone. En général, on distingue deux topologies de réseaux de neurones :

- Les réseaux de neurones feed-forward, dans lesquels les neurones ne sont connectés que dans le sens de l'entrée vers la sortie et sans aucune rétroaction (Cf. figure II.3). Ce type est relativement simple et est couramment utilisé dans beaucoup de domaines.

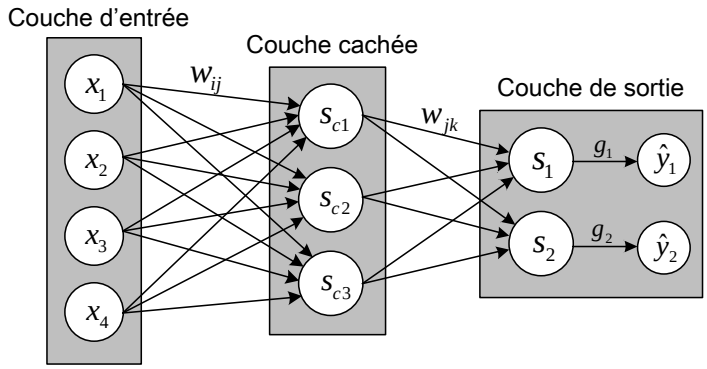


Figure II.3 – Un réseau de neurones feed-forward

- Les réseaux de neurones récurrents, qui contiennent des connexions de rétroaction. Les neurones de sortie peuvent par exemple voir leur sortie utilisée comme entrée d'un neurone de la couche précédente ou de la même couche, comme le montre la figure II.4. Contrairement aux réseaux feed-forward, les propriétés dynamiques du réseau sont importantes. Dans certains cas, les valeurs d'activation des neurones subissent un processus de relaxation de sorte que le réseau évolue vers un état stable dans lequel les activations ne changent plus. Dans autres applications, les changements des valeurs d'activation des neurones de sortie sont significatifs de telle sorte que le comportement dynamique constitue la sortie du réseau (Pearlmutter - 1990; Anderson - 1977; Kohonen - 1977; Hopfield - 1982).

En général, la détermination d'une taille appropriée de la couche cachée est une difficulté. Une sous-estimation du nombre de neurones peut mener à une mauvaise approximation et à une mauvaise performance de la généralisation, tandis que les neurones excessifs peuvent engendrer du sur-apprentissage et rendre la recherche de l'optimum global plus difficile. Beaucoup d'études concernent ce problème, telles que la détermination du nombre minimum de neurones et du nombre d'exemples nécessaires pour construire un réseau de neurones feed-forward approprié (Camargo et Yoneyama - 2001; Kon et Plaskota - 2000).

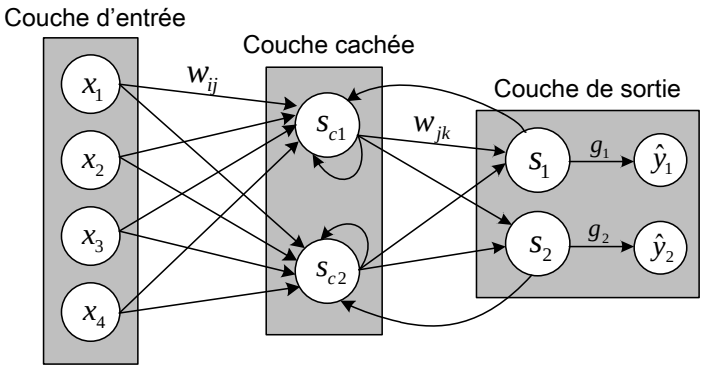


Figure II.4 – Un réseau de neurones récurrent

II.3.2.2.3 Ajustage des poids

En résumant, un réseaux de neurones dépend de trois aspects fondamentaux :
i) l'architecture du réseau (feed-forward ou récurrent) ;
ii) les fonctions d'activation des neurones ;
iii) la détermination des poids des connexions

Une fois les deux premiers aspects fixés, la performance du réseau de neurones est alors définie par les valeurs des poids. Habituellement, les poids du réseau sont initialement choisis aléatoirement, puis les exemples de l'ensemble d'apprentissage sont présentés au réseau. Les attributs d'un exemple sont présentés sur les neurones d'entrée et la sortie finale du réseau est comparée avec la sortie attendue de ce exemple. Ensuite, tous les poids du réseau sont ajustés légèrement dans le sens qui amène la valeur de sortie du réseau au plus proche de celle attendue. De nombreuses méthodes permettant de corriger les poids vis à vis d'un ensemble d'apprentissage donnés ont été étudiés (Neocleous et Schizas - 2002) et la plus connue et largement utilisée est celle de rétro-propagation du gradient. La règle générale pour la mise à jour les poids est donnée par la formule :

$$\Delta w_{ji} = \eta \delta_j \hat{y}_i$$

- η est un nombre positif (nommé le taux d'apprentissage), qui détermine la taille d'un pas dans la recherche descendante du gradient.
- $\hat{y}_i$ est la sortie calculée du neurone i ;
- pour les neurones de sortie, $\delta_j = \hat{y}_j(1 - \hat{y}_j)(y_j - \hat{y}_j)$ où y_j est la sortie attendue du neurone j
- pour les neurones cachés, $\delta_j = \hat{y}_j(1 - \hat{y}_j) \sum_k \delta_k w_{kj}$

Les réseaux de neurones feed-forward sont généralement construits par l'algorithme de rétro-propagation du gradient original ou par certaines de ses variantes. Hormis leur grande lenteur, un des problèmes majeurs réside dans la convergence vers des un minimum local. L'une des approches pour accélérer le processus d'apprentissage est

d'estimer les poids initiaux au lieu de les tirer au hasard (Yam et Chow - 2001).

II.3.2.2.4 Méthodes existantes

Différentes approches ont été mises en œuvre pour estimer les poids des réseaux de neurones et pour définir leur architecture : algorithmes génétiques (Siddique et Tokhi - 2001; Yen et Lu - 2000), méthodes bayésiennes (Vivarelli et Williams - 2001). D'autres techniques ont été proposées récemment qui tentent d'améliorer les algorithmes d'apprentissage d'un réseau de neurones en modifiant l'architecture du réseau lors du processus d'apprentissage. Ces techniques incluent l'élagage des neurones ou des poids inutiles (Castellano et al. - 1997), et les algorithmes constructifs où les neurones supplémentaires sont ajoutés selon les besoins (Parekh et al. - 2000).

Les réseaux à fonction de base radiale (*radial basis function*, RBF) ont été largement appliqués aux domaines de la sciences et de l'ingénierie (Robert et Lakhmi - 2001). Un réseau à RBF est un réseau ayant trois couches, dans lequel chaque neurone caché a une fonction d'activation radiale et chaque neurone de sortie calcule une somme pondérée des sorties des neurones cachés. Son processus d'apprentissage est généralement divisé en deux étapes. Tout d'abord, les centres et les rayons des RBF de la couche cachée sont déterminées par des méthodes de *clustering*. Deuxièmement, les poids reliant la couche cachée à la couche de sortie sont déterminés par la méthode de la décomposition en valeurs singulières (*Singular Value Decomposition*, SVD) ou la méthode des moindres carrés (*Least Mean Squared*, LMS). Un choix approprié du nombre des fonctions de base radiales (i.e. le nombre des neurones cachés), qui contrôle la complexité et la généralisation des réseaux RBF, demeure un problème critique. Les réseaux RBF avec peu de neurones cachés ne peuvent pas s'adapter suffisamment aux données d'apprentissage en raison de la flexibilité limitée. Au contraire, ceux ayant beaucoup de neurones cachés amènent une mauvaise généralisation car ils sont trop flexibles et sensibles au bruit dans les données d'apprentissage.

Beaucoup d'études ont comparé les performances entre les réseaux de neurones et les classifieurs conventionnels (Curram et Mingers - 1994; Huang et Lippmann - 1987; Michie et al. - 1994), qui illustrent que les réseaux de neurones présentent l'avantage d'être une méthode auto-adaptative dans laquel le système peut s'adapter aux données sans aucune forme explicite de la distribution pour le modèle sous-jacent. On note aussi que les réseaux de neurones peuvent approcher toutes les fonctions avec une précision arbitraire (Cybenko - 1989; Hornik - 1991; Hornik et al. - 1989). Étant donné que tous les processus de classification consistent à chercher une relation entre les attributs des exemples et leurs labels et à présenter cette relation sous la forme de fonctions, l'identification exacte de la fonction sous-jacente est importante. Les réseaux de neurones ont également la propriété d'être des modèles non-linéaires permettant plus de flexibilité pour modéliser les relations complexes. Enfin, les réseaux de neurones peuvent estimer les probabilités a posteriori, qui fournissent la base pour établir la règle de classification et pour effectuer l'analyse statistique des données (Richard et Lippmann - 1991). Toutefois les inconvénients majeurs des réseaux de neurones résident dans leur lenteur, leur convergence vers des minima locaux, une architecture difficile à paramétrer et aussi le manque de capacité à interpréter leurs sorties. Sur ce dernier

point, de nombreux chercheurs se sont déjà consacré à améliorer la compréhensibilité des réseaux de neurones, où la solution la plus attrayante est d'extraire les règles symboliques (Setiono et Leow - 2000; Zhou - 2004; Roy - 2000).

II.3.3 Méthodes statistiques

À l'inverse des réseaux de neurones, les méthodes statistiques sont caractérisés par l'exploitation d'un modèle explicite de probabilité conditionnellement à chaque classe. Dans cette catégorie, nous pouvons trouver par exemple les réseaux bayésiens et les *instance-based learning algorithms*.

II.3.3.1 Réseaux bayésiens

Un réseau bayésien est en informatique et en statistique un modèle graphique probabiliste représentant un ensemble de variables aléatoires (ou un ensemble d'attributs) sous la forme d'un graphe orienté acyclique (Pearl - 1988). En considérant un ensemble fini d'attributs discrets $\boldsymbol{X} = [X_1\ ,\ X_2\ ,\ \ldots\ ,\ X_d]^t$, le réseau bayésien consiste à définir un graphe orienté acyclique qui représente une distribution de probabilité conjointe sur cet ensemble d'attributs $\boldsymbol{X}$, formellement par une description $B = \langle\, G\,,\, \Theta\,\rangle$. B représente un réseau bayésien. Le premier composant G est un graphe orienté acyclique dont les noeuds correspondent aux attributs $X_1,\ X_2,\ \ldots, X_d$, et dont les liens représentent les relations causales directes entre les attributs tandis que l'absence de liens dans le graphe implique une indépendance conditionnelle. Plus précisément, chaque attribut X_i est indépendant de ses non-descendants pour les états de ses parents donnés dans le graphe G. Le second composant Θ représente l'ensemble des paramètres de la distribution de probabilité conjointe. Ces paramètres probabilistes sont décrits par un ensemble de tableaux (en anglais *conditional probability tables* ou CPTs) dont chacun énumère la probabilité locale d'un attribut qui prend chacune des valeurs possibles pour toutes les combinaisons des valeurs de ses parents, noté $\theta_{x_i \mid pa(x_i)} = P_B\left[x_i \mid pa(x_i)\right]$ où x_i est une valeur de X_i, $pa(X_i)$ dénote un ensemble de parents de X_i et $pa(x_i)$ correspond aux valeurs de $pa(X_i)$. Par ailleurs, l'hypothèse d'indépendance conditionnelle permet de réduire le nombre de paramètres. Finalement un réseau bayésien B détermine une distribution unique de probabilité conjointe sur $\boldsymbol{X}$ par les tableaux de probabilité locale,

$$P_B\left[x_1,\ x_2,\ \ldots, x_d\right] = \prod_{i=1}^{d} P_B\left[x_i \mid pa(x_i)\right] = \prod_{i=1}^{d} \theta_{x_i \mid pa(x_i)} \tag{II.18}$$

Les réseaux bayésiens naïfs (Duda et Hart - 1973; Langley et al. - 1992) sont un réseau bayésien très simple consistant en un graphe orienté acyclique avec un seul parent représentant le noeud non-observé (i.e. une variable de label de classe) et plusieurs enfants correspondant aux noeuds observés (i.e. les variables des attributs), comme le montre la figure II.5. Ils apprennent, à partir des données d'apprentissage, la probabilité conditionnelle de chaque attribut X_i pour un label de classe y donné. La classification est ensuite effectuée en appliquant la règle de Bayes pour calculer la probabilité a posteriori de Y pour une réalisation $\boldsymbol{x} = [x_1\ ,\ x_2\ ,\ \ldots\ ,\ x_d]^t$ du vecteur d'attributs $\boldsymbol{X}$.

Le label de classe $\hat{y}$ est ensuite prédit avec la plus grande probabilité a posteriori (Cf. équation II.19)

$$P\left[Y = y \mid \boldsymbol{X} = \boldsymbol{x}\right] = \alpha \cdot P\left[Y = y\right] \cdot \prod_{i=1}^{d} P\left[X_i = x_i \mid Y = y\right] \tag{II.19}$$

où α est une constante de normalisation. C'est en fait la définition de Bayes naïf trouvée communément dans la littérature (Langley et al. - 1992). Il faut noter que ce modèle est basé sur une forte hypothèse d'indépendance : tous les attributs X_i sont indépendants entre eux conditionnellement à une classe donnée, c'est-à-dire,

$$P\left[X_i = x_i \mid X_j = x_j, Y = y\right] = P\left[X_i = x_i \mid Y = y\right] \quad \forall\, y \text{ tel que } \quad P\left[Y = y\right] > 0$$

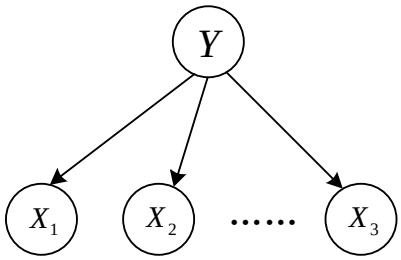


Figure II.5 – La structure d'un réseau bayésien naïf

La construction d'un réseau bayésien consiste à trouver un réseau B qui s'adapte le mieux à un ensemble d'apprentissage. La clé du problème réside dans le moyen de choisir le meilleur réseau. L'approche commune est d'introduire une fonction de score qui évalue chaque réseau par rapport aux données d'apprentissage, et puis de rechercher le meilleur réseau selon les valeurs de cette fonction. En général, ce problème d'optimisation n'est pas réalisable (Chickering - 1995), mais il existe, pour certains réseaux avec un nombre restreint de classes, quelques algorithmes efficaces (Chow et Liu - 1968; Pearl - 1988; Cormen et al. - 1990).

Un problème des classifieurs basés sur les réseaux bayésiens est qu'ils ne conviennent pas aux données représentées dans des espaces de grande dimension pour des raisons de temps de calcul et d'espace mémoire (Cheng et al. - 2002). Par ailleurs, les réseaux bayésiens ne peuvent pas traiter les attributs continus. Par conséquent, les attributs continus doivent être discrétisés avant d'entrer dans le processus de la construction du réseau dans la plupart des cas (Yang et Webb - 2003).

II.3.3.2 Méthodes "Instance-based learning"

Les algorithmes d'apprentissage *instance-based* (Mitchell - 1997) (aussi nommés « lazy-learning-algorithms » car le processus de généralisation est retardé jusqu'à ce

qu'une requête soit effectuée) requièrent moins de temps de calcul lors de la phase d'apprentissage que les algorithmes d'apprentissage « eager-learning-algorithms », tels que les arbres de décision, les réseaux de neurones et les réseaux bayésiens, mais plus de temps de calcul au cours du processus de classification. L'algorithme des plus proches voisins est le plus connu des algorithmes d'apprentissage *instance-based*. Il effectue la classification sur la base des exemples d'apprentissage les plus proches dans l'espace d'attributs (Cover et Hart - 1967).

Le principe de la méthode des k plus proches voisins (en abrégé kNN) est basé sur le fait que les exemples se situent généralement à proximité des autres exemples ayant les propriétés similaires. Pour un nouvel exemple $\boldsymbol{x}_0$ à classer, le processus de la prédiction de son label $\hat{y}_0$ se fait selon les étapes suivantes,

1. estimer les similarités entre $\boldsymbol{x}_0$ et chaque exemple $\boldsymbol{x}_i$ dans l'ensemble d'apprentissage $\mathcal{A}_n$ par une fonction de similarité $s_i = s(\boldsymbol{x}_0, \boldsymbol{x}_i)$ $(i = 1, \ldots, n)$.
2. créer la liste décroissante $s_{(1)} \geq s_{(2)} \geq \ldots \geq s_{(n)}$ des similarités. Si $s_j = s_{(k)}$, cela signifie que $\boldsymbol{x}_j$ est le $k^{\text{ième}}$ exemple d'apprentissage le plus proche de $\boldsymbol{x}_0$. Par conséquent, le nombre de k plus proches exemples de $\boldsymbol{x}_0$ peut alors être défini comme un ensemble des voisins dont les similarités sont au minimum $s_{(k)}$, noté $N(\boldsymbol{x}_0, k) = \left\{ \boldsymbol{x}_i \mid s_i \geq s_{(k)} \right\}$.
3. estimer la probabilité de chaque classe (Cf. équation II.20),

$$\widehat{P}\left[Y = c\right] = \frac{\sum_{\boldsymbol{x}_i \in N(\boldsymbol{x}_0, k)} \mathbb{1}_{y_i = c}(\boldsymbol{x}_i)}{|N(\boldsymbol{x}_0, k)|} \qquad \forall\, c \;\in\; \{1, 2, \ldots, K\} \tag{II.20}$$

   $|N(\boldsymbol{x}_0, k)|$ représente l'effectif de l'ensemble $N(\boldsymbol{x}_0, k)$, qui est exactement égale à k en général mais il est possible de dépasser k dans certains cas.
4. choisir le label y_0^* de $\boldsymbol{x}_0$ qui maximise la probabilité estimée (Cf. équation II.20). Pour obtenir des résultats plus précis, plusieurs algorithmes utilisent une stratégie de pondérations qui modifient les mesures de similarité et l'influence de chaque voisin dans $N(\boldsymbol{x}_0, k)$ (Wettschereck et al. - 1997).

En bref, trois aspects doivent être préalablement définis pour mettre en œuvre l'algorithme de kNN : la mesure de similarité $s(\boldsymbol{x}_i, \boldsymbol{x}_j)$, le nombre de voisins k et le seuil de décision θ. Le choix du nombre k est certainement le plus important : pour un k trop grand, certains des voisins dans $N(\boldsymbol{x}_0, k)$ ne seront plus similaires à l'observation en question, ce qui introduit un biais dans la prédiction. Au contraire, un trop petit nombre k de voisins engendre une prédiction instable à cause d'un manque d'informations. Ce problème est particulièrement évident lors de la présence de bruit (Wettschereck et al. - 1997). Le choix du nombre optimal k consiste à mettre en balance ce compromis (Hastie et al. - 2001). Une méthode classique pour sélectionner cette valeur k est d'utiliser la validation croisée sur l'ensemble d'apprentissage (Celisse et Mary-Huard - 2011).

Un avantage des algorithmes d'apprentissage *instance-based* permettant de se distinguer des arbres de décision et de certains réseaux de neurones est sa stabilité (Breiman - 1996a). Un algorithme d'apprentissage est considéré '*instable*' lorsque des petits changements dans l'ensemble d'apprentissage ou l'ensemble de test peuvent causer des grandes variations pour les résultats de classification.

Son inconvénient majeur est de requérir plus de temps de calcul au cours du processus de classification, comme nons l'avons déjà mentionné. Le temps de classer un exemple dépend essentiellement du nombre d'exemples d'apprentissage conservés et du nombre d'attributs utilisés pour décrire chaque exemple. Par conséquent, les méthods de filtrage d'exemples et les méthodes de sélection d'attributs (Yu et Liu - 2004) ont été proposés pour réduire le nombre d'exemples conservés et pour déterminer les attributs efficaces. Brighton et Mellish (2002) montrent que leur algorithme ICF et l'algorithme RT3 de Wilson et Martinez (2000) accomplissent une réduction maximum d'exemples en conservent la capacité prédictive. Les résultats obtenus par ces algorithmes sont assez impressionnants : une moyenne de 80% d'exemples sont exclus sans dégrader considérablement les performances de classification. Par ailleurs, le choix d'une fonction de similarité plus appropriée pour un ensemble des données spécifique est aussi important pour améliorer la performance de classification.

II.3.4 Méthodes de *Support Vector Machines*

Les *support vector machines* (SVM) sont un nouveau type de méthodes d'apprentissage pour la classification binaire, motivé par les résultats de la théorie de l'apprentissage statistique (Vapnik - 1995, 1998). Les méthodes de SVM aspirent à la minimisation du risque structurel (en anglais '*structural risk minimization*') qui donne un compromis entre la complexité de l'espace des fonctions de décision (exprimée par la dimension de VC) et la qualité de l'adaptation aux données d'apprentissage (i.e. l'erreur empirique) (Vapnik et Kotz - 1982). Les SVM ont montré de bonnes performances dans de nombreux domaines d'applications comme par exemple, le classement de textes, la reconnaissance de formes, le diagnostic médical, etc. Ils sont maintenant reconnus comme l'un des outils standards pour l'apprentissage.

L'idée des algorithms de SVM est de partager l'espace en deux parties à l'aide d'un hyperplan qui maximise la distance minimale des observations à ce plan (i.e. la marge). Les observations qui sont situées les plus proches de l'hyperplan séparateur (sur la marge), sont appelées les « vecteurs supports ».

On considère un problème de classification à deux classes (ω_0 et ω_1) et un ensemble $\mathcal{A}_n = \{ (\boldsymbol{x}_i, y_i) \in \mathcal{X}^m \times \mathcal{Y},\ i = 1, \ldots, n \}$ de n exemples et on pose $y_i = -1$ si $\boldsymbol{x}_i \in \omega_0$ et $y_i = +1$ si $\boldsymbol{x}_i \in \omega_1$. L'équation d'un hyperplan séparateur est définie à une constante multiplicative près par :

$$f(\boldsymbol{x}) = \langle\ \boldsymbol{w},\ \boldsymbol{x}\ \rangle + b = 0 \tag{II.21}$$

où $|b| \,/\, \| \boldsymbol{w} \|$ est la distance perpendiculaire de l'hyperplan à l'origine et $\| \boldsymbol{w} \|$ est la norme euclidienne de $\boldsymbol{w}$. La classe d'une nouvelle observation $\boldsymbol{x}$ non apprise sera donnée en fonction du signe de l'expression du membre de gauche de l'équation (II.21).

II.3.4.1 SVM linéaires

Si les classes ω_0 et ω_1 sont linéairement séparables, l'hyperplan séparateur vérifie alors les conditions suivantes :

$$\begin{cases} \langle\, \boldsymbol{w}\,,\, \boldsymbol{x}_i \,\rangle + b \geq +1 & \text{pour } \; y_i = +1 \\ \langle\, \boldsymbol{w}\,,\, \boldsymbol{x}_i \,\rangle + b \leq -1 & \text{pour } \; y_i = -1 \end{cases} \tag{II.22}$$

ce qui peut se résumer par une description unifiée :

$$y_i(\langle\, \boldsymbol{w}\,,\, \boldsymbol{x}_i \,\rangle + b) - 1 \geq 0 \quad \forall\, (\boldsymbol{x}_i, y_i) \;\in\; \mathcal{A}_n \tag{II.23}$$

Deux hyperplans supplémentaires H_1 et H_2 sont alors déterminés par les points qui donnent l'égalité dans l'équation (II.22), comme le montre la figure II.6. Les points encerclés qui se situent sur les hyperplans H_1 et H_2 sont les « vecteurs supports » et la marge vaut donc $2/\parallel \boldsymbol{w} \parallel$. Maximiser la marge revient à minimiser le carré de la norme $\parallel \boldsymbol{w} \parallel^2$ du vecteur $\boldsymbol{w}$ sous les contraintes (II.23). Le problème d'optimisation peut être alors formulé comme suit :

$$\begin{cases} \min & \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 \\ \text{s.c.} & y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 \quad \forall\, i = 1, \ldots, n \end{cases} \tag{II.24}$$

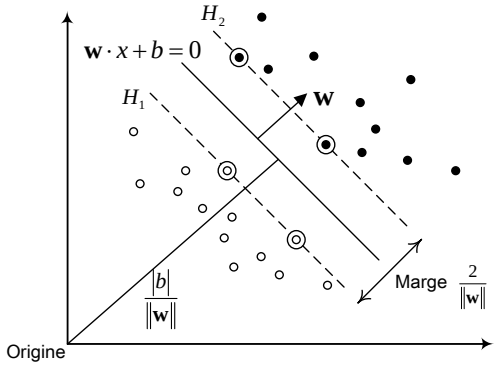


Figure II.6 – Principe des SVM. Les *vecteurs support* sont encerclés

Malheureusement, de nombreux ensembles de données ne sont pas toujours parfaitement linéairement séparables (il existe toujours quelques points qui ne peuvent pas être correctement classés). On peut alors résoudre ce problème en utilisant une marge souple qui tolère quelques erreurs de classification (Veropoulos et al. - 1999; Cortes et Vapnik - 1995). Des variables ressort ξ_i sont introduites pour permettre de relâcher les contraintes (II.23). Le problème d'optimisation (II.24) devient alors,

$$\begin{cases} \min & \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C \sum_{i=1}^{n} \xi_i \\ \text{s.c.} & y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i,\; \xi_i \geq 0,\; C > 0,\; \forall\, i = 1, \ldots, n \end{cases} \tag{II.25}$$

C est une constante qui permet de contrôler le compromis entre le nombre d'erreurs de classification et la largeur de la marge. Elle est choisie par l'utilisateur à l'avance, en général au moyen d'une recherche exhaustive dans l'espace des paramètres en utilisant par exemple la validation croisée sur l'ensemble d'apprentissage. Le terme $C\sum_{i=1}^{n}\xi_i$ a pour effet de pénaliser les échantillons mal classés.

Le problème d'optimisation (II.25) peut être résolu par la technique classique de programmation quadratique. Les multiplicateurs de Lagrange sont d'abord introduits, le lagrangien est donné par,

$$\min_{\boldsymbol{w},b,\xi} L_P = \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\left[\, y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) - 1 + \xi_i \,\right] - \sum_{i=1}^{n}\beta_i\xi_i \qquad \text{(II.26)}$$

avec $\alpha_i, \beta_i \geq 0$. En annulant les dérivées partielles du lagrangien par rapport à $\boldsymbol{w}$ et b, selon les conditions de Kuhn-Tucker (Fletcher - 1987), on obtient :

$$\begin{cases} \boldsymbol{w} = \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i \\ \sum_{i=1}^{n}\alpha_i y_i = 0 \end{cases} \qquad \text{(II.27)}$$

En réinjectant ces valeurs dans l'équation (II.26), on obtient la formulation duale,

$$\begin{cases} \min\limits_{\alpha} & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j(\boldsymbol{x}_i\cdot\boldsymbol{x}_j) - \sum_{k=1}^{n}\alpha_k \\ \text{s.c.} & \sum_{i=1}^{n}\alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \end{cases} \qquad \text{(II.28)}$$

Les observations $\boldsymbol{x}_i$ dont le lagrangien est non nul, sont dénommées vecteurs support. Il s'agit des observations situées dans la marge ou mal classés.

II.3.4.2 SVM non-linéaires

Les classifieurs linéaires ont des capacités de classification limitées. Dans la plupart des applications les données sont rarement séparables linéairement. Une solution consiste alors à projeter les données dans un espace de très grande dimension, éventuellement infinie, nommé l'espace de Hilbert $\mathcal{H}$ au moyen d'une transformation non-linéaire ϕ :

$$\mathbb{R}^d \xrightarrow{\ \phi\ } \mathcal{H}$$

Dans ce nouvel espace, il est alors probable qu'il existe un hyperplan séparant les données. D'après l'équation (II.28), il est par ailleurs facile de voir que le classifieur SVM ne dépend que des produits scalaires dans l'espace $\mathcal{H}$ (i.e. $\langle\, \phi(\boldsymbol{x}_i)\,,\, \phi(\boldsymbol{x}_j)\,\rangle$). Heureusement, il existe une classe de fonctions qui satisfont les conditions de Mercer (Vapnik - 1995; Courant et Hilbert - 1954), nommées fonctions noyaux K, qui permettent de calculer les produits scalaires de l'image dans l'espace original sans prendre en considération la forme explicite de la transformation non-linéaire,

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle\, \phi(\boldsymbol{x}_i)\,,\, \phi(\boldsymbol{x}_j)\,\rangle$$

Par conséquent, l'équation (II.28) peut être re-écrite avec la fonction noyau,

$$\begin{cases} \min\limits_{\alpha} & \frac{1}{2}\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{n}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i,\boldsymbol{x}_j) - \sum\limits_{k=1}^{n}\alpha_k \\ \text{s.c.} & \sum\limits_{i=1}^{n}\alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \end{cases} \tag{II.29}$$

Les multiplicateurs de Lagrange α_i^* sont déterminés par la résolution de l'équation (II.29) et une fonction de classification mettant en œuvre l'hyperplan optimal dans l'espace de Hilbert est alors donnée par,

$$f(\boldsymbol{x}) = sign\left(\sum_{i=1}^{n}\alpha_i^* y_i K(\boldsymbol{x},\boldsymbol{x}_i) + b\right) \tag{II.30}$$

Le classifieur n'est donc défini qu'à partir des $\boldsymbol{x}_i$ qui sont situés dans la marge ou mal classés, les vecteurs support. En fait, le nombre de vecteurs support est souvent petit. Pour cette raison, les méthodes de SVM sont bien appropriés aux applications où le nombre d'attributs est grand par rapport au nombre d'exemples d'apprentissage, par exemple la catégorisation de textes (Joachims - 1998). Quelques fonctions noyaux K les plus largement utilisées dans les SVM sont,

- les noyaux polynomiaux de degré p : $K(\boldsymbol{x},\boldsymbol{x'}) = (\langle\, \boldsymbol{x}\,,\, \boldsymbol{x'}\,\rangle + 1)^p$
- les noyaux Gaussiens : $K(\boldsymbol{x},\boldsymbol{x'}) = \exp\left(-\dfrac{\|\boldsymbol{x}-\boldsymbol{x'}\|^2}{2\sigma^2}\right)$

La plus grande limitation des méthodes de SVM réside dans le choix d'une fonction noyau, car c'est elle qui définit un espace induit où la classification est effectuée. Certains travaux sur la limitation du noyau ont été réalisés en utilisant les connaissances préalables (Schölkopf et al. - 1998a; Burges - 1999), mais le meilleur choix du noyau pour un problème spécifique est toujours un sujet de recherche. Par ailleurs, concernant la détermination des paramètres du noyau, on fixe un intervalle de valeurs potentielles puis on utilise la validation croisée sur l'ensemble d'apprentissage. Ce processus peut être considéré de façon analogue au choix du nombre de noeuds cachés dans un réseau de neurones. Pour cette raison, une autre limitation des SVM est la faible vitesse d'apprentissage. Finalement, les méthodes de SVM classiques sont des méthodes de classification binaire, le cas multi-classes doit être décomposé en un ensemble de problèmes à deux classes ou traité directement par des méthodes multi-classes (Lee et al. - 2004; Weston et Watkins - 1998). Ces dernières années, beaucoup de variantes des algorithmes de SVM classiques ont été proposées, telles que ν-SVM (Schölkopf et al. - 2000; Chang et Lin - 2002), SVDD (Tax et Duin - 2004) et one-class SVM (Schölkopf et al. - 2001).

II.4 Méthodes d'ensemble

Les algorithmes d'apprentissage que nous avons présentés précédement, consistent à construire un classifieur unique. Toutefois, de nombreuses études empiriques confirment que la performance des méthodes d'ensemble qui consistent à combiner plusieurs

classifieurs simples et locaux est généralement meilleure que celle d'un seul classifieur élémentaire de base (Freund et Schapire - 1996; Bauer et Kohavi - 1999; Dietterich - 2000). On observe souvent que la performance en généralisation (sur des nouveaux exemples) d'un système reposant sur un ensemble de classifieurs continue à augmenter lorsqu'on augmente le nombre de classifieurs, et cela, même lorsque la performance en apprentissage semble avoir atteint le maximum (Kleinberg - 2000). En général, la conception d'un système de classification à base d'ensemble de classifieurs comporte schématiquement deux étapes principales : la définition et la mise au point des classifieurs et le choix de la méthode d'intégration ou de fusion de ces classifieurs.

II.4.1 Production d'un ensemble de classifieurs

La première étape est donc de produire un ensemble de classifieurs élémentaires performants, c'est-à-dire qui présentent de meilleurs taux de bonne prédiction sur un nouveau exemple doit être au moins supérieur à celui de la conjecture aléatoire (50% pour un problème à deux classes). Dans cette étape on cherchera aussi à obtenir une bonne diversité entre les classifieurs afin que les décisions fournies par les classifieurs soient peu corrélées qui doivent être évaluées sur différentes parties de l'ensemble d'apprentissage. La construction d'un ensemble de classifieurs est généralement réalisée par les techniques suivantes :

i) soit en utilisant des algorithmes d'apprentissage différents,

ii) soit en utilisant le même algorithme mais avec des paramètres ou initialisations différents,

iii) soit en utilisant des sous-ensembles d'apprentissage différents avec le même algorithme. Parmi toutes les approches, on va présenter les plus connues, le bagging, le boosting et la sélection aléatoire de sous-espaces de représentation (en anglais '*random subspace method*' ou RSM).

II.4.1.1 Bagging

Le bagging est une méthode d'ensemble qui introduit la technique du rééchantillonnage dans la production d'un ensemble de classifieurs (Breiman - 1996a). A partir d'un ensemble d'apprentissage original $\mathcal{A}_n$, cette méthode consiste à générer plusieurs ensembles de données indépendants de taille n par bootstrap (tirage aléatoire avec remise dans $\mathcal{A}_n$ de n couples $(\boldsymbol{x}_i, y_i)$ de données étiquetées). Pour chaque échantillon ainsi généré on construit un classifieur en utilisant le même algorithme d'apprentissage. La décision finale est obtenue par vote majoritaire.

Breiman (1996a) signale qu'il est important d'avoir un algorithme d'apprentissage instable (i.e. sensible aux changements des données d'apprentissage) et qu'il s'agit d'une condition préalable pour que le bagging fonctionne efficacement. Par exemple, les algorithmes d'apprentissage tels que les arbres de décision sont instables. Les études de Breiman (1996b, 1997) et Dietterich (2000) montrent que le bagging fonctionne en réduisant la variance de l'erreur de classification en laissant le biais inchangé. Le biais et la variance de l'erreur sont deux critères utiles pour étudier le comportement

d'un algorithme d'apprentissage et sont donc souvent employés pour la conception d'ensembles de classifieurs (Valentini et Dietterich - 2002, 2004; Buciu et al. - 2006).

II.4.1.2 Boosting

Le boosting est une méthode itérative d'ajout de classifieurs faibles dans un ensemble (Freund - 1995, 2001; Meir et Rätsch - 2003), qui consiste à faire varier la base d'apprentissage par des pondérations sucessives des mêmes exemples afin de se focaliser sur les exemples « difficiles » (i.e. les exemples mal classés par plusieurs classifieurs déjà intégrés). Le classifieur fourni dans chaque cycle est également pondéré par la qualité de sa classification. La décision finale est donc obtenue par la somme pondérée des sorties des classifieurs élémentaires de l'ensemble. Cette technique de la production d'un ensemble peut s'utiliser avec un algorithme d'apprentissage quelconque. Nous pouvons obtenir de très bons classifieurs simplement en assemblant un grand nombre de classifieurs faibles. Par contre, la difficulté existante est d'incorporer des connaissances a priori et d'effectuer le meilleur choix d'un algorithme d'apprentissage faible.

AdaBoost est une version classique de l'approche boosting (Freund et Schapire - 1997), qui exige moins d'instabilité que le bagging parce que cette méthode permet d'engendrer de beaucoup plus grands changements dans l'ensemble d'apprentissage. Plusieurs études comparatives sur AdaBoost et bagging (Breiman - 1996a; Schapire et al. - 1998; Bauer et Kohavi - 1999) ont montré que, en général, le bagging tend à diminuer la variance sans influer excessivement sur le biais, tandis que AdaBoost réduit à la fois le biais et la variance. Il semble donc que AdaBoost soit plus efficace que le bagging, mais le bagging reste le plus efficace au sens de la réduction de la variance.

II.4.1.3 RSM

Une autre approche efficace pour la construction d'un ensemble de classifieurs élémentaires est d'utiliser différents sous-espacess de représentation. Ho (1995, 1998a,b) montre que la sélection aléatoire de sous-espaces de représentation est une approche prometteuse, nommé *random subspace method* (RSM). Cette approche consiste à construire un ensemble de règles sur la base d'une méthode de classification et d'un choix aléatoire d'espaces de représentation. Supposons que la dimension de l'espace de représentation initial soit m. La méthode RSM consiste à tirer au sort d attributs parmi m afin d'obtenir un sous-espace de représentation de dimension d ($d < m$). Une variante de l'ensemble d'apprentissage peut donc être constituée pour ce sous-espace de représentation afin d'y construire un classifieur. On a donc autant de classifieurs que de sous-espaces de représentation de dimension d et on fusionne ensuite ces classifieurs pour la décision finale. Les résultats expérimentaux (Ho - 1998a) illustrent que l'utilisation d'un grand nombre de classifeurs reposant sur des espaces de petite dimension peut être préférable à l'apprentissage d'un classifieur unique dans l'espace initial.

L'intérêt de RSM est que la sélection d'espaces de représentation est implicitement embarquée dans le processus d'apprentissage d'un ensemble. En effet, l'objectif des algorithmes traditionnels de sélection d'espaces de représentation est de trouver le meilleur espace de représentation pour l'algorithme d'apprentissage considéré, tandis

qu'ici, il s'agit de sélectionner un ensemble de plusieurs espaces de représentation pour constituer un système de décision reposant sur plusieurs classifieurs.

II.4.2 Intégration d'un ensemble de classifieurs

L'intégration des sorties des classifieurs élémentaires ainsi générés est un autre aspect important intervenant dans les performances du classifieur final. Le vote est la technique d'intégration la plus couramment utilisée où la prédiction finale est obtenue en comptabilisant les sorties (votes) des classificateurs élémentaires (Roli et al. - 2001). La forme la plus simple est le vote majoritaire, dans lequel chaque classifieur apporte un seul vote pour une classe particulière et la prédiction finale se fait en faveur de la classe qui a reçu une majorité de votes (Hall et al. - 2000). Une variante réside dans le vote majoritaire pondéré, dans lequel l'influence de chaque classifieur sur la prédiction finale est pondérée par sa performance estimée sur un ensemble de validation. La prédiction finale est alors de choisir la classe ayant la plus grande valeur de la somme pondéré des votes. Ces techniques apparaissent comme simples et efficaces d'après plusieurs études (Hansen et Salamon - 1990; Krogh et Vedelsby - 1995; Tumer et Ghosh - 1996; Bauer et Kohavi - 1999; Opitz et Maclin - 1999).

Par ailleurs, une autre idée prometteuse proposée par Kotsiantis et Pintelas (2004) est de profiter des avantages de la fusion des classifieurs et la sélection dynamique. Les algorithmes qui sont initialement utilisés pour construire l'ensemble sont testés sur un sous-ensemble d'apprentissage de petite dimension, afin de trouver les classifieurs ayant la « pire qualité » de classification d'un point de vue statistique. Ces « pires classifieurs » sont exclus du vote final dans la partie test.

Hormis la technique du vote, celle du *stacking* (Wolpert - 1992) se propose d'améliorer l'efficacité en exécutant plusieurs processus d'apprentissage et en combinant les résultats de chaque cycle. La différence principale entre le vote et le *stacking* est que le dernier combine les classifieurs élémentaires d'une manière non-linéaire. La tâche de combinaison, appelée post-apprenant, est d'intégrer les classifieurs élémentaires indépendants dans un seul classifieur supérieur (nommé post-classifieur) par le réapprentissage sur un post-ensemble d'apprentissage dont les valeurs d'attributs sont les prédictions des classifieurs élémentaires sur un ensemble de validation et les labels réels des classes étant les cibles. Ting et Witten (1999) décrivent une situation étendue où les attributs du post-ensemble d'apprentissage sont les probabilitées des classes produites par les classifieurs élémentaires au lieu des prédictions de classes. Une telle technique d'intégration est utilisé par Breiman (1996c) et LeBlanc et Tibshirani (1996).

II.5 Évaluation des classifieurs

Comme nous venons de le voir, le choix d'un algorithme d'apprentissage est une étape cruciale et il est donc nécessaire de disposer de critères pour évaluer les performances des algorithmes d'apprentissage et du système de décision qui en découle. Dans le cas d'un système de classification binaire, les critères les plus largement utilisés sont ceux classiquement définis dans le cadre des tests statistiques de deux hypothèses H_0 et

H_1. Il s'agit des probabilités ou taux d'erreur (ou de risque) de première (Cf. équation II.31) et de seconde espèce (Cf. équation II.32) qui sont illustrées dans le tableau II.1. Dans le contexte de la classification binaire, l'hypothèse H_0 sera par exemple celle de l'appartenance d'un individu à une classe et H_1 sera celle de l'appartenance à l'autre classe. Supposons qu'une classe est décrite par une région notée ω_0 dans l'espace de représentation $\mathcal{X}^m$, que l'autre classe est représentée par une région notée ω_1 et que l'on dispose d'un classifieur $h : \mathcal{X}^m \xrightarrow{h} \mathcal{Y} = \{0, 1\}$ tel que $h(\boldsymbol{x}) = 0$ correspond à la décision de classer $\boldsymbol{x}$ dans la classe de H_0, on peut alors écrire les caractéristiques du classifieur (ou le taux global de bonne décision) selon les formules qui suivent :

$$\alpha = P[D_1 \mid H_0] = P[h(\boldsymbol{X}) = 1 \mid \boldsymbol{X} \in \omega_0] \quad \boldsymbol{X} \in \mathcal{X}^m \tag{II.31}$$

$$\beta = P[D_0 \mid H_1] = P[h(\boldsymbol{X}) = 0 \mid \boldsymbol{X} \in \omega_1] \tag{II.32}$$

$$\text{d'où} \quad P_e = \alpha P[H_0] + \beta P[H_1] \tag{II.33}$$

Il existe au moins trois techniques pour estimer l'erreur globale d'un classifieur ou ses composantes α et β. Une première technique consiste à utiliser deux tiers de l'ensemble d'apprentissage pour apprendre le classifieur et l'autre tiers pour estimer la performance. Une deuxième technique est celle de la validation croisée, dans laquel l'ensemble d'apprentissage est divisé en plusieurs sous-ensembles de taille égale et mutuellement exclusifs. L'un des sous-ensembles est utilisé comme ensemble de test et l'union des autres comme ensemble d'apprentissage. Les critères de performance choisis sont alors estimés par le taux moyen d'erreur de chaque sous-ensemble de test. Une dernière technique est celle du *leave-one-out* qui est une version particulière de la validation croisée, dans laquelle tous les sous-ensembles de test se composent d'un seul exemple. Ce type de validation réclame bien sûr davantage de calculs, mais elle s'avère particulièrement utile lorsqu'on souhaite une estimation précise du taux d'erreur du classifieur.

Vérité / Décision	H_0	H_1
D_1	risque ou erreur de première espèce α	$1 - \beta$
D_0	$1 - \alpha$	risque ou erreur de seconde espèce β

Tableau II.1 – Critères de performances d'un système de décision binaire

Des études récentes ont montré que l'erreur d'un classifieur peut être généralement décomposée en deux termes : le biais et la variance, ce qui peut fournir plus informations importantes sur la performance de prédiction du classifieur (Bauer et Kohavi - 1999). Le biais mesure la contribution à l'erreur de la tendance centrale (*central tendency*) du classifieur appris sur des données différentes. La variance est une mesure de la contribution à l'erreur de déviations par rapport à la tendance centrale. Le biais et la variance sont évaluées sur la base d'une distribution de l'ensembles d'apprentissage,

comme une distribution contenant tous ensembles d'apprentissage possibles de taille spécifique pour un domaine particulier. En général, les algorithmes d'apprentissage avec un grand biais génèrent des modèles simples et très contraints qui sont assez insensibles aux fluctuations des données, de sorte que la variance est faible. Au contraire, les algorithmes avec une grande variance peuvent construire les modèles arbitrairement complexes qui s'adaptent plus facilement aux variations des données. Le piège évident de ce type des algorithmes d'apprentissage est le problème de surapprentissage, telles que les arbres de décision, les réseaux de neurones et les SVM.

D'autres critères de performances peuvent être pris en considération, comme le temps d'apprentissage, l'espace de stockage et la robustesse au bruit. Bien que le temps d'apprentissage varie selon la nature de l'application et l'ensemble de données, les spécialistes s'accordent sur une compartimentation en général pour les algorithmes d'apprentissage exitants. Par exemple, les algorithmes d'apprentissage paresseux, comme kNN, n'ont pas besoin de grands temps d'apprentissage parce que les exemples d'apprentissage sont simplement stockés. Les réseaux bayésiens naïfs sont appris très rapidement car ils n'exigent qu'une seule passe sur les données pour estimer la loi de probabilité des attributs. Toutefois, en considérant l'aspect de espace de stockage, les algorithmes classiques des kNN nécessitent beaucoup d'espace pour l'apprentissage et l'espace d'exécution est au moins aussi grand que l'espace d'apprentissage, tandis que les réseaux bayésiens naïfs nécessitent peu d'espace pendant la phase d'apprentissage et de classification pour stocker les probabilités a priori et conditionnelles. En outre, l'interprétabilité des algorithmes d'apprentissage est aussi un indicateur de performance. Les algorithmes logiques tels que les arbres de décision et les algorithmes des règles sont considérés comme très faciles à interpréter, alors que les réseaux de neurones et les SVM ont une interprétabilité notoirement faible. kNN est également considéré comme ayant une interprétabilité faible car la collection non-structurée d'exemples d'apprentissage n'est pas lisible.

La difficulté principale posée par les méthodes d'ensemble réside dans la recherche d'un modèle optimal. La procédure la plus commune consiste, dans un premier temps à optimiser individuellement chaque classifieur de base dont la structure est souvent choisie empiriquement, puis à optimiser la fonction de fusion. Il y a fort à parier que cette procédure est sous-optimale. De plus, elle rend le réglage fin du système décisionnel très complexe (optimisation au sens de Neyman-Pearson par exemple). Cette difficulté est notamment due à l'absence de résultats théoriques permettant de lier les performances individuelles des classifieurs à la performance du classifieur global.

II.6 Conclusion

Dans ce chapitre, nous avons présenté les principales méthodes de sélection et d'extraction d'attributs ainsi que les méthodes d'apprentissage existantes, à l'exception toutefois des méthodes paramétriques qui ne sont pas appropriées dans le cadre de nos travaux (nous ne faisons pas d'hypothèse sur le modèle de distribution des données à traiter). Cet état de l'art nous a permis d'avoir une vision claire du domaine de

recherche et des méthodes proposées et constitue la base du travail que nous avons réalisé. Toutefois, aucun des découpages proposés ici n'est parfaitement net et l'important est de rechercher les outils appropriés à notre problème. Comme nous l'avons présenté dans la partie d'introduction de ce mémoire (Cf. section I page 1), nous avons choisi dans cette thèse de nous intéresser au problème de la surveillance et du diagnostic d'un système complexe par la classification de mesures issues d'un ensemble de capteurs situés dans un environnement non-stationnaire dont une partie des capteurs peuvent en subir les effets. Dans ce contexte il nous a donc semblé opportun d'adopter une méthode de classification reposant sur un ensemble de classifieurs distincts opérant dans des sous-espaces de représentation dont on espère que certains seront non altérés par l'environnement. Par ailleurs, nous avons également souligné dans l'introduction que l'environnement n'affectant qu'une partie seulement des capteurs, il opère en quelque sorte une sélection d'attributs au sein de l'espace de représentation. Il nous a donc semblé naturel d'adopter une méthode de réduction de la dimension de l'espace de représentation initial par sélection d'attributs.

Chaque famille de méthodes de sélection d'espaces de représentation a ses propres avantages et faiblesses. En général, les méthodes de filtre sont rapides puiqu'elles n'interagissent pas avec l'algorithme d'apprentissage utilisé. Par contre, les méthodes de type *wrapper* peuvent avoir une meilleure performance que les méthodes de filtre. Toutefois, elles nécessitent une durée d'exécution beaucoup plus longue puisqu'elles impliquent de relancer plusieurs fois l'algorithme d'apprentissage afin d'évaluer les sous-ensembles d'attributs à chaque itération. Les méthodes embarquées sont réputées plus rapides que les méthodes de type *wrapper* parce que la sélection d'attributs et l'optimisation de l'apprenant peuvent être réalisées en même temps. Elles peuvent aussi donner de meilleures performances que les méthodes de filtre, mais avec un risque de surapprentissage.

On peut maintenant compléter la figure II.1 en précisant le choix possible d'une méthode de classification au sein de deux grandes familles : les méthodes individuelles et les méthodes d'ensemble (Cf. figure II.7). La section II.3 a présenté les méthodes individuelles selon quatre types : les méthodes logiques, celles reposant sur l'algorithme du perceptron, les méthodes statistiques et les SVM. Les méthodes logiques contenant principalement les arbres de décision et les méthodes à base de règles permettent généralement une meilleure compréhensibilité et interprétabilité mais elles sont très sensibles à l'ensemble d'apprentissage, aux attributs non pertinents et au bruit. De plus, la plupart d'entre elles, comme ID3 et C4.5, exigent que les attributs utilisés soient binaires. Par contre, les réseaux de neurones et les SVM peuvent traiter efficacement les données décrites par les attributs multi-dimensionnels et continus mais ils ne sont pas simples sur le plan de la compréhensibilité et de l'interprétabilité. Plus particulièrement, les méthodes de SVM ont résolu le problème des données non-séparables au moyen de la projection des données dans un espace induit de dimension supérieure et la complexité de calcul est réduite grâce à l'introduction de la fonction noyau. kNN est un algorithme stable mais requiert beaucoup de temps de calcul lors de la phase de classification.

Si nous nous concentrons uniquement sur la meilleure précision de la classification, il pourrait être difficile de trouver un seul classifieur qui fonctionne aussi bien que les

méthodes d'ensemble. Malgré des avantages évidents, les méthodes d'ensemble ont au moins trois faiblesses. Premièrement l'accroissement du stockage est une conséquence directe car tous les classifieurs dans l'ensemble doivent être mémorisés après l'appentissage. La taille du stockage exigé dépend de la taille de chaque classifieur lui-même et de la taille de l'ensemble (le nombre de classifieurs dans l'ensemble). La deuxième faiblesse est l'accroissement du calcul parce que tous les classifieurs dans l'ensemble sont requis pour traiter un exemple à classer. La dernière est la compréhensibilité de plus en plus faible. Avec l'augmentation du nombre de classifieurs dans la règle de décision, il est plus difficile de percevoir le processus conduisant à une décision.

En conclusion, il n'existe pas de méthode optimale d'apprentissage dans l'absolu, car on a vu qu'un algorithme d'apprentissage dépend beaucoup des données et de l'application. Le choix d'une approche ou la mise au point d'une méthode de classification doit donc prendre en compte les situations particulières que l'on doit traiter et c'est ce à quoi nous nous sommes attachés.

Notre objectif dans cette thèse est donc de concevoir et de construire une règle de décision robuste face à des perturbations ou changements brutaux de l'environnement. A l'issue du passage en revue dans ce chapitre des méthodes de classification existantes, il nous semble que les méthodes d'ensemble constituent certainement un choix adéquat pour atteindre cet objectif. Le chapitre III présente ainsi une classification robuste via la définition d'un ensemble de sous-espaces de représentation efficaces. Dans le chapitre IV une telle méthode d'ensemble est re-développée par l'étude et la modélisation d'une fonction de fusion de décisions afin de mieux contrôler la performance du système décisionnel global.

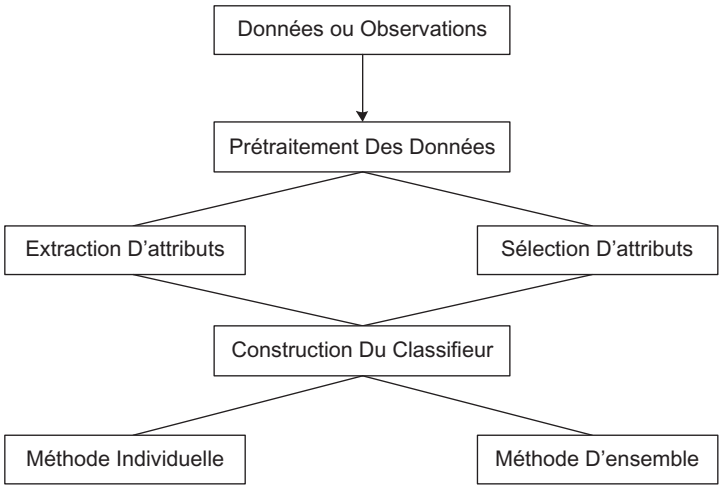


Figure II.7 – Structure d'un système de classification

Chapitre III

Classification robuste via la définition d'un ensemble d'espaces de représentation

III.1 Introduction

Dans un environnement non-stationnaire, les performances du système décisionnel peuvent se dégrader sensiblement si des changements ou des perturbations transitoires surviennent sur une partie des mesures du système. Ces changements peuvent être dûs à l'apparition de bruit, de dysfonctionnement ou de panne de certains capteurs etc. La règle de décision qui a été apprise n'est plus adaptée aux données nouvelles à classer qui se trouvent alors décrites par des attributs perturbés ne correspandant plus à l'espace initial de représentation dans lequel s'est fait l'apprentissage.

Dans une telle situation, il peut être intéressant de considérer des sous-espaces de représentation obtenus par projection à partir de l'espace initial, ce qui revient ainsi à sélectionner des attributs parmi tous ceux disponibles initialement. De cette manière, on peut raisonnablement espérer obtenir des espaces de représentation de dimension inférieure à celle de l'espace de départ, mais reposant en revanche sur des attributs non altérés. Une fois sélectionné un ensemble d'espaces de représentation, on peut alors mettre au point un classifieur dans chaque sous-espace (voir par exemple la section II.4.1.3) et définir ainsi un ensemble de classifieurs. La décision finale peut ensuite être prise en fusionnant les sorties de tous ces classifieurs. Une telle approche s'inscrit dans le cadre de ce que l'on nomme les méthodes d'ensemble. Elles s'organisent schématiquement en deux phases principales : la phase d'apprentissage comportant la construction d'un ensemble de classifieurs et la phase de prédiction effectuant la combinaison ou fusion des sorties des classifieurs élémentaires afin d'obtenir la décision finale (Cf. figure III.1). Comparées à une classification unique les méthodes d'ensemble permettent de réduire la variance des erreurs de classification des différents classifieurs (ce qui permet de stabilser la décision finale en présence de changements) et de diminuer la possibilité de choisir un mauvais classifieur (Dietterich - 2000). La performance globale du système décisionnel peut, de cette manière, être préservée ou améliorée.

Les deux méthodes d'ensemble les plus populaires sont le bagging (Cf. section II.4.1.1) et le boosting (Cf. section II.4.1.2) qui reposent sur deux exploitations différentes de l'ensemble d'apprentissage. Une autre solution consiste à utiliser une approche par sélection d'attributs pour former des sous-espaces de représentation dif-

férents afin d'obtenir différents ensembles d'apprentissage par projection de l'ensemble d'apprentissage initial dans ces sous-espaces. C'est cette dernière approche que nous avons retenue, pour les raisons que nous avons évoquées au début de cette section et notre démarche a été de générer un ensemble de classifieurs mis au point sur des projections $\mathcal{A}_n^1,\ \mathcal{A}_n^2,\ \ldots, \mathcal{A}_n^L$ de l'ensemble d'apprentissage initial $\mathcal{A}_n$ dans les sous-espaces de représentation sélectionnés. L'objectif de la phase d'apprentissage est alors de construire un ensemble $h_1,\ h_2,\ \ldots, h_L$ de L classifieurs, performants et divers.

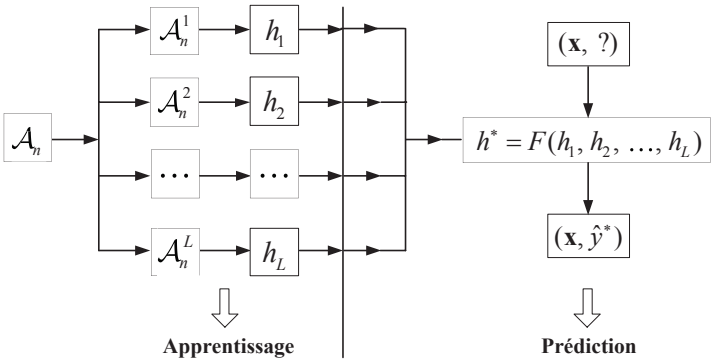


Figure III.1 – Le schéma de principe des méthodes d'ensemble

Notre choix d'opter pour un ensemble de sous-espaces de représentation n'a pas pour objectif principal de réduire la dimension de l'espace de représentation initial, mais surtout de garantir la robustesse des décisions face à des changements locaux de l'information délivrée par le système sous surveillance.

Dans ce chapitre nous considérons donc un problème de classification de données représentées par un vecteur dans un espace d'attributs pouvant être altéré par des anomalies causées par la présence de bruit ou la panne de capteurs. Dans une telle situation, on peut admettre que l'apprentissage des règles de décision permettant la classification a été réalisé sur des données représentant une classe unique dite « normale ». En effet, dans la plupart des situations réelles, il est très souvent difficile voire impossible d'obtenir des exemples en nombre suffisant pour la ou les classes que l'on pourrait qualifier d'« anormale ». L'idée principale des méthodes de classification à une classe consiste à ajuster le modèle de classifieur sur la base d'un ensemble d'apprentissage constitué de données de la classe dite normale. Un nouvel exemple est alors classé en comparant son score, obtenu par le modèle appris, à un seuil de décision. L'altération de l'espace des attributs engendre alors des données d'une ou plusieurs classes nouvelles que nous désignerons par la suite sous l'appellation unique de « classe de rejet ». Dans ce contexte, il nous a semblé naturel d'adopter une approche du type « classification à une classe » (en anglais *one-class classification*). Toutefois, malgré la dénomination « classification à une classe », dans la pratique on est amené à considérer les deux hypothèses ou classes suivantes :

- « classe normale » c'est la classe cible, la seule pour laquelle on dispose de données d'apprentissage (d'où l'appellation) et qui sont nommées données positives,
- « classe de rejet » c'est une (ou plusieurs) classe nouvelle se différenciant de la classe normale et qui est engendrée par l'altération de tout ou partie des at-

tributs représentant les données et pour laquelle on ne dispose pas de données d'apprentissage.

Dans cette thèse nous avons exploité et adapté trois méthodes. Tout d'abord une méthode inspirée des SVM (Support Vector Machine), baptisée *One-class SVM* proposée par Schölkopf et al. (2001). Une seconde méthode reposant sur l'ACP (Analyse en composantes Principales) avec une variante utilisant un noyau (KPCA) afin de s'affranchir des aspects non linéaires dans la distribution des données, (Schölkopf et al. - 1998b). Enfin, une méthode plus récente proposée par Jenssen (2010) qui utilise une transformation des données fondée sur l'entropie de Rényi et baptisée *Kernel Entropy Component Analysis* (KECA). Pour ces deux dernières méthodes (KPCA et KECA), nous avons utilisé l'erreur de reconstruction comme mesure de classification d'après Diamantaras et Kung (1996) et Hoffmann (2007).

Les approches de la définition des sous-espaces de représentation dans le contexte de l'apprentissage d'ensemble sont présentées dans la section III.2. La section III.3 introduit brièvement les trois méthodes d'apprentissage testées. La section suivante présente les détails de l'approche que nous avons proposée et enfin une dernière partie illustre et analyse les résultats expérimentaux que nous avons obtenus sur des images de textures qui constituent un support applicatif tout à fait appropriée pour simuler des situations d'environnement non-stationnaire.

III.2 Définition des sous-espaces de représentation pour un ensemble

Traditionnellement la sélection d'attributs (Cf. section II.2) se focalise sur la recherche d'un sous-espace de représentation pertinent qui sera utilisé pour la construction d'un seul classifieur. Dans le cas des méthodes d'ensemble, nous cherchons à définir plusieurs sous-espaces de représentation afin de construire un ensemble de classifieurs. Une condition importante de bonnes performances d'un ensemble de classifieurs repose sur la différence entre les divers classifieurs. Il est évident que la combinaison de plusieurs classifieurs identiques ne produit aucun gain. D'après l'étude de Krogh et Vedelsby (1995), l'erreur globale d'un ensemble repose généralement sur deux aspects : l'erreur de généralisation moyenne de chaque classifieur individuel et la diversité entre les classifieurs. En d'autres termes on peut dire qu'un ensemble idéal se compose de plusieurs classifieurs performants aussi divers que possible. De nombreuse études ont empiriquement prouvé qu'un tel ensemble de classifieurs peut donner de meilleures performances globales (Opitz et Shavlik - 1996; Breiman - 1996a; Freund et Schapire - 1996). En résumé, il s'agit de trouver un bon compromis entre la diversité et la capacité prédictive individuelle de chaque classifieur.

III.2.1 Diversité

La diversité joue donc un rôle important dans la construction d'un ensemble de classifieurs. On trouve une littérature abondante sur le sujet avec beaucoup de travaux

présentant des mesures de la diversité. Kuncheva et Whitaker (2003) ont étudié différentes approches statistiques de mesure de la diversité selon que les mesures se font sur des paires de classifieurs ou pas. On trouvera également quelques discussions sur cet aspect dans la thèse de He (2009).

III.2.1.1 Mesures par paires

Soit donc un ensemble de classifieurs $\{h_1,\ h_2,\ \ldots, h_L\}$ et un ensemble d'apprentissage $\mathcal{A}_n = \{\,(\boldsymbol{x}_1\ ,\ y_1),\ (\boldsymbol{x}_2\ ,\ y_2),\ \ldots, (\boldsymbol{x}_n\ ,\ y_n)\,\}$. Dans le cas des mesures par paires, on définit alors un vecteur de sortie $\boldsymbol{s}_\ell = \left[s_1^\ell\ ,\ s_2^\ell\ ,\ \ldots\ ,\ s_n^\ell\right]^t$ pour chaque classifieur individuel h_ℓ $(\ell = 1, \ldots, L)$, dont les éléments représentent la différence, pour chaque $\boldsymbol{x}_i \in \mathcal{A}_n$, entre la prédiction de h_ℓ et le label vrai de $\boldsymbol{x}_i$,

$$s_i^\ell = \begin{cases} 1 & \text{si } h_\ell(\boldsymbol{x}_i) = y_i \\ 0 & \text{sinon} \end{cases} \quad , (i = 1, \ldots, n) \tag{III.1}$$

La relation entre une paire h_ℓ et h_k peut alors être décrite par un tableau de contingence (Cf. tableau III.1) donnant la distribution conjointe (en fréquences absolues) des décisions entre h_ℓ et h_k au sein de $\mathcal{A}_n$.

h_ℓ \ h_k	$s_i^k = 1$	$s_i^k = 0$
$s_i^\ell = 1$	n_{11}	n_{10}
$s_i^\ell = 0$	n_{01}	n_{00}

Tableau III.1 – Distribution des décisions entre deux classifieurs

Sur la base de ce tableau, on peut définir plusieurs mesures de diversité entre deux classifieurs. Skalak (1996) et Ho (1998b) ont proposé une statistique reposant sur les différences de décisions. Giacinto et Roli (2001) définissent une mesure dite de double faute. Toutes ces mesures conduisent au calcul d'une mesure moyenne sur l'ensemble des paires possibles.

III.2.1.2 Autres mesures

Kuncheva et Whitaker (2003) a résumé six types de mesures n'opérant pas sur des paires de classifieurs. Mesure d'entropie, de variance (Kohavi et Wolpert - 1996), d'unanimité entre les classifieurs (Fleiss - 1981; Dietterich - 2000), de difficulté (Hansen et Salamon - 1990), de diversité généralisée (Partridge et Krzanowski - 1997) et enfin de coïncidence de diversité d'échec (*coincident failure diversity*) (Partridge et Krzanowski - 1997) qui est une modification de la mesure de diversité généralisée. Un peu plus tard, Cunningham et Carney (2000) ont proposé d'utiliser une entropie conditionnelle associée à la diversité d'un ensemble comme une fonction de perte (i.e. une mesure d'erreur).

Pour conclure, notons toutefois que la notion de diversité, bien que très utilisée, est très mal définie.

III.2.2 Choix des sous-espaces de représentation via la sélection d'attributs

Comme nous l'avons dit en introduction, nous avons choisi l'approche par sélection de sous-espaces de représentation pour construire un ensemble de classifieurs. Différentes techniques ont été proposées dans la littérature.

La sélection aléatoire de sous-espaces de représentation (*Random Subspace Method*, RSM) (Cf. section II.4.1.3) procède par tirage aléatoire de d attributs parmi les m ($d < m$) attributs initiaux pour engendrer un sous-espace de représentation dans lequel est mis au point un classifieur. Ho (1998b) a montré que la règle de décision finale obtenue par vote majoritaire sur un ensemble de classifieurs définis sur un grand nombre d'espaces de petite dimension pouvait s'avérer plus performante qu'un classifieur unnique dans l'espace complet initial.

Une autre méthode plus sophistiquée pour déterminer plusieurs sous-ensemble d'attributs est proposé par Guerra-Salcedo et Whitley (1999), où un algorithme génétique (*genetic algorithm*, GA) est employé afin d'explorer l'espace de tous les sous-ensembles d'attributs possibles. Leurs expérimentations comparent trois méthodes différentes de construction d'un ensemble : le bagging et Adaboost en utilisant l'espace complet, les sous-espaces construits par RSM et les sous-espaces construits par GA respectivement. Les résultats expérimentaux montrent que l'ensemble de classifieurs reposant sur des sous-espaces construits par GA a la meilleure performance, suivi par RSM. La sélection génétique d'attributs (*Genetic Ensemble Feature Selection*, GEFS) proposée par Opitz (1999) a également utilisé un algorithme génétique pour rechercher des sous-ensembles d'attributs. GEFS part de la population initiale de classifieurs, où chacun dépend d'une combinaison dynamique d'attributs. Des opérateurs de croisement et de mutation sont ensuite utilisés pour rechercher des nouveaux sous-ensembles d'attributs afin de construire des nouveaux classifieurs candidats. Chaque nouveau classifieur candidat est évalué par une mesure qu'il intitule « fitness » et qui prend en compte la capacité de généralisation et la diversité.

$$\text{Fitness}_\ell = \text{Performance}_\ell + \lambda \text{Diversité}_\ell$$

où λ contrôle le compromis entre la performance et la diversité des classifieurs. La population des classifieurs est ensuite réduite aux L classifieurs ayant les meilleurs « fitness » scores et le vote majoritaire est appliqué pour déterminer la décision finale de l'ensemble. GEFS a donné de meilleurs résultats en généralisation avec des réseaux de neurones que le bagging et AdaBoost. Cette approche, comme AdaBoost, va dans le sens qui a soutenu nos travaux, à savoir que l'optimum global de l'ensemble ne résulte pas de l'optimum individuel des classifieurs. Toutefois de nombreuses expérimenations comparatives dans la littérature, montrent que la méthode RSM est plus performante et offre l'avantage d'une bonne interprétabilité. Par conséquent, nous avons opté pour cette apprcohe dans cette thèse.

III.3 Algorithmes d'apprentissage testés

Comme nous l'avons indiqué dans l'introduction de ce chapitre (III.1), nous nous plaçons dans le contexte de la classification à une classe pour laquelle nous avons testé trois méthodes que nous présentons dans cette section.

III.3.1 One-class SVM

Le one-class SVM (Schölkopf et al. - 2001) est une méthode inspirée des SVM, qui repose sur la formulation suivante : *considérant un ensemble de données tirées d'une loi de probabilité P, le problème est d'estimer un sous-ensemble S de l'espace de représentation des données tel que la probabilité qu'un point test tiré de la loi P appartienne à $\bar{S}$ (complément de S) soit bornée par un seuil spécifié a priori.*

Soit donc $\mathcal{X}^m \subseteq \mathbb{R}^m$ l'espace de représentation complet de dimension m et $\mathcal{A}_n = \{ (\boldsymbol{x}_i, y_i) \mid i = 1, \ldots, n \}$ un ensemble d'apprentissage de taille n avec $\boldsymbol{x}_i \in \mathcal{X}^m$ et y_i label de la classe de $\boldsymbol{x}_i$ ($y_i = 0$ ou 1). Plus précisément, l'objectif de la méthode de one-class SVM est d'estimer une fonction f qui soit positive sur un sous-ensemble $S \subset \mathcal{X}^m$ contenant une certaine proportion des données de la classe apprise et négative sur son complément $\bar{S}$. Soit ϕ une transformation non-linéaire telle que le produit scalaire de l'image des observations $\boldsymbol{x}$ par ϕ (Cf. équation III.2) puisse être calculé par un noyau simple K (noyau semi-défini positif sur $\mathbb{R}^m$ vérifiant les conditions de Mercer (Boser et al. - 1992; Vapnik - 1995),

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle\, \phi(\boldsymbol{x}_i)\,,\, \phi(\boldsymbol{x}_j)\, \rangle \tag{III.2}$$

comme par exemple un noyau gaussien,

$$K_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}} \tag{III.3}$$

La stratégie de cette méthode est de projeter d'abord les données dans un espace induit par le noyau K utilisé puis d'en séparer une proportion $1 - \nu$ de l'origine à l'aide de l'hyperplan le plus éloigné de l'origine. Normalement un tel hyperplan est appelé l'hyperplan de marge maximale. Dans l'espace initial, cet hyperplan séparateur correspond à l'enveloppe d'une région de petit volume contenant cette proportion des données d'apprentissage $\mathcal{A}_n$. Pour cela il faut déterminer le vecteur normal $\boldsymbol{w}$ et un seuil ρ en résolvant le problème suivant,

$$\begin{cases} \min\limits_{\boldsymbol{w}, \boldsymbol{\xi}, \rho} & \frac{1}{2} \| \boldsymbol{w} \|^2 + \frac{1}{\nu n} \sum\limits_{i=1}^{n} \xi_i - \rho \\ \text{sous les contraintes} & \langle\, \boldsymbol{w}, \phi(\boldsymbol{x}_i) \,\rangle \geq \rho - \xi_i, \quad \xi_i \geq 0 \end{cases} \tag{III.4}$$

Des variables ressorts ξ_i, associées à chaque exemple, sont introduites pour pouvoir relâcher les contraintes. Cela permet que certains exemples d'apprentissage puissent être hors du domaine S. Par conséquent, l'algorithme d'optimisation consiste à trouver un compromis entre la maximisation de la marge et la minimisation du ressort moyen.

Les exemples $\boldsymbol{x}_i \in \mathcal{A}_n$ qui se trouvent du mauvais côté de l'hyperplan sont considérés comme des anomalies et la distance entre une anomalie et l'hyperplan est alors égale à $\xi_i / \| \boldsymbol{w} \|$ avec $\xi_i > 0$. Par ailleurs, la distance entre la marge et l'origine vaut $\rho / \| \boldsymbol{w} \|$. Le paramètre de réglage $\nu \in [0, 1]$ est une borne supérieure du taux d'anomalies, mais aussi une borne inférieure du taux de vecteurs supports.

Le problème d'optimisation (III.4) peut être résolu par l'introduction les multiplicateurs de Lagrange $\alpha_i, \beta_i \geq 0$, le lagrangien est alors donné par,

$$\begin{aligned} L(\boldsymbol{w}, \boldsymbol{\xi}, \rho, \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \frac{1}{2} \| \boldsymbol{w} \|^2 + \frac{1}{\nu n} \sum_{i=1}^{n} \xi_i - \rho \\ &\quad - \sum_{i=1}^{n} \alpha_i \left(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle - \rho + \xi_i \right) - \sum_{i=1}^{n} \beta_i \xi_i \end{aligned} \tag{III.5}$$

En annulant les dérivées partielles du lagrangien par rapport aux variables primales $\boldsymbol{w}, \boldsymbol{\xi}, \rho$, on obtient,

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i) \tag{III.6}$$

et

$$\alpha_i = \frac{1}{\nu n} - \beta_i \leq \frac{1}{\nu n}, \quad \sum_{i=1}^{n} \alpha_i = 1 \tag{III.7}$$

Les exemples $\boldsymbol{x}_i$ $(i = 1, \ldots, n)$ avec $\alpha_i > 0$ dans l'équation (III.6) sont appelés les vecteurs supports. En réinjectant les valeur (III.6,III.7) dans l'équation (III.5), le problème dual peut s'écrire,

$$\begin{cases} \min\limits_{\boldsymbol{\alpha}} & \frac{1}{2} \sum\limits_{i,j=1}^{n} \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{sous les contraintes} & 0 \leq \alpha_i \leq \frac{1}{\nu n}, \quad \sum\limits_{i=1}^{n} \alpha_i = 1 \end{cases} \tag{III.8}$$

et la fonction de décision est alors donnée par,

$$\begin{aligned} f(\boldsymbol{x}) &= sgn \left(\sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) - \rho \right) \\ &= sgn \left(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle - \rho \right) \end{aligned} \tag{III.9}$$

L'optimum est atteint si β_i est strictement positif pour tous les $\boldsymbol{x}_i$ pour lesquels ξ_i est nul. Dans ce cas, $\alpha_i < 1/(n\nu)$.

D'autre part on a $\alpha_i > 0$ lorsque $\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle - \rho$ est nul. Donc dans le cas où $\alpha_i \in]0, \frac{1}{n\nu}[$, on a $\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle - \rho = 0$ et par conséquent, le paramètre ρ peut être déterminé de façon à ce que le vecteur support correspondant à ces α_i satisfasse,

$$\rho = \langle \boldsymbol{w}, \phi(\boldsymbol{x}_i) \rangle = \sum_{j=1}^{n} \alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_i) \tag{III.10}$$

De même, le ressort peut être obtenu par le calcul pour toutes les anomalies (i.e. avec $\alpha_i = \frac{1}{\nu n}$ correspondant),

$$\xi_i = \rho - \langle\, \boldsymbol{w}, \phi(\boldsymbol{x}_i) \,\rangle = \rho - \sum_{j=1}^{n} \alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_i) \tag{III.11}$$

Pour un nouvel exemple $\boldsymbol{x}$ à classer, son label est déterminé en évaluant de quel côté de l'hyperplan il se situe dans l'espace transformé au moyen de la fonction $f(\boldsymbol{x})$ (Cf. équation III.9).

En résumé, nous pouvons donc distinguer trois types d'exemples d'apprentissage :
- Les anomalies $\boldsymbol{x}_i \in \bar{S}$ (rejet) pour lesquelles on a : $f(\boldsymbol{x}_i) < 0$, $\alpha_i = \frac{1}{\nu n}$ et $\xi_i > 0$
- Les exemples normaux $\boldsymbol{x}_i \in S$ (acceptation) pour lesquels on a : $f(\boldsymbol{x}_i) > 0$, $\alpha_i = 0$ et $\xi_i = 0$
- Les observations sur l'enveloppe de S pour lesquelles on a : $f(\boldsymbol{x}_i) = 0$ et $0 \leq \alpha_i \leq \frac{1}{\nu n}$

III.3.2 Kernel PCA

L'analyse en composante principale (en anglais *Principal Component Analysis*, PCA) est une technique puissante et assez simple à mettre en œuvre (Jolliffe - 1986; Diamantaras et Kung - 1996) pour extraire la structure de données de grande dimension. PCA est en fait une transformation orthogonale du système de coordonnées dans lequel les données sont représentées et les nouvelles coordonnées sont appelées composantes principales. L'idée de cette méthode repose sur le fait qu'un petit nombre de composantes principales est normalement suffisant pour représenter la structure de la plupart des données. Toutefois, la limitation majeure de PCA est qu'elle ne traite bien que les données situées dans un hyperplan de l'esparce initial. Pour les cas où les données appartiennent à une variété non-linéaire, Schölkopf et al. (1998b) se sont inspiré des SVM et ont développé une nouvelle méthode utilisant la technique du noyau, d'où l'appellation *Kernel Principal Component Analysis* (KPCA).

Les exemples d'un ensemble d'apprentissage $\mathcal{A}_n = \{\, (\boldsymbol{x}_i, y_i) \mid i = 1, \ldots, n \,\}$ sont d'abord projetés dans un espace induit par une transformation non-linéaire ϕ,

$$\boldsymbol{x}_i \xmapsto{\phi} \phi(\boldsymbol{x}_i)$$

Dans ce nouvel espace, on effectue une PCA classique sur les données transformées, préalablement centrées (Cf. équation III.12)

$$\tilde{\Phi}(\boldsymbol{x}_i) = \phi(\boldsymbol{x}_i) - \frac{1}{n} \sum_{r=1}^{n} \phi(\boldsymbol{x}_r) \tag{III.12}$$

La matrice de covariance à diagonaliser s'exprime alors par,

$$C = \frac{1}{n} \sum_{i=1}^{n} \tilde{\Phi}(\boldsymbol{x}_i) \tilde{\Phi}^t(\boldsymbol{x}_i) \tag{III.13}$$

La matrice C n'étant pas accessible, la recherche de ses vecteurs et valeurs propres nécessite quelques développements que nous ne détaillerons pas ici. L'idée de base repose sur l'utilisation de fonctions noyaux pour représenter les produits scalaires dans l'espace transformé. Schölkopf et al. (1998b) ont ainsi montré que la résolution de l'équation

$$\lambda \mathbf{v} = C\mathbf{v} \tag{III.14}$$

était équivalente à la résolution de l'équation

$$n\lambda \mathbf{e} = \tilde{\mathbf{K}}\mathbf{e} \tag{III.15}$$

dans l'équation III.15 $\tilde{\mathbf{K}}$ $(n \times n)$ représente la matrice de terme général,

$$\tilde{K}_{ij} = \langle\, \tilde{\Phi}(\boldsymbol{x}_i)\,,\, \tilde{\Phi}(\boldsymbol{x}_j)\,\rangle \tag{III.16}$$

Soit $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ les valeurs propres de la matrice $\tilde{\mathbf{K}}$ (i.e. les solutions de l'équation III.15) et $\mathbf{e}^{(1)}, \ldots, \mathbf{e}^{(n)}$ l'ensemble complet des vecteurs propres correspondants. En considérant les p premières valeurs propres non nulles, les vecteurs propres correspondants $\mathbf{e}^{(1)}, \ldots, \mathbf{e}^{(p)}$ doivent être normalisés, de sorte que

$$\langle\, \mathbf{v}^{(r)}\,,\, \mathbf{v}^{(r)}\,\rangle = 1 \quad \forall\, r = 1, \ldots, p \tag{III.17}$$

ce qui se traduit par,

$$\mathbf{v}^{(r)} = \lambda_r^{-\frac{1}{2}} \sum_{i=1}^{n} e_i^{(r)} \tilde{\Phi}(\boldsymbol{x}_i) \quad \forall\, r = 1, \ldots, p \tag{III.18}$$

où λ_r correspond à la valeur propre de $\tilde{\mathbf{K}}$ associée au vecteur propre $\mathbf{e}^{(r)}$

Généralement, on ne retient qu'un nombre limité q de vecteurs propres associés aux q plus grandes valeurs propres λ_k $(k = 1, \ldots, q)$ et un nouveau point test $\boldsymbol{x}$ a alors pour projection,

$$\begin{aligned} \langle\, \mathbf{v}^k\,,\, \tilde{\Phi}(\boldsymbol{x})\,\rangle &= \lambda_k^{-\frac{1}{2}} \sum_{i=1}^{n} e_i^k \langle\, \tilde{\Phi}(\boldsymbol{x}_i)\,,\, \tilde{\Phi}(\boldsymbol{x})\,\rangle \\ &= \lambda_k^{-\frac{1}{2}} \sum_{i=1}^{n} e_i^k \tilde{K}(\boldsymbol{x}_i, \boldsymbol{x}) \end{aligned} \tag{III.19}$$

Les propriétés mathématiques et statistiques de PCA (Jolliffe - 1986; Diamantaras et Kung - 1996) sont donc valables pour KPCA moyennant quelques adaptations. On peut les résumer ainsi :

- les q premières composantes principales engendrent une plus grande variance que toutes les autres combinaisons de q directions orthogonales ;
- l'erreur quadratique moyenne d'approximation résultant de la représentation des données par les q premières composantes principales est minimale ;
- les composantes principales ne sont pas corrélées ;
- Dans le cas gaussien, les q premières composantes principales portent l'information mutuelle maximale par rapport aux données d'entrée.

III.3.3 Kernel ECA

L'analyse en composante d'entropie à noyau (en anglais *Kernel Entropy Component Analysis*, KECA) est une nouvelle méthode de transformation des données et de réduction de dimension proposée récemment par Jenssen (2010), qui expose anisi le principe de cette méthode :

KECA capte la structure des données dans l'espace d'entrée sur la base de l'entropie de Rényi, qui est estimée en utilisant le noyau de Parzen. En supposant que la matrice du noyau est semi-définie positive, l'estimation d'entropie de Rényi obtenue peut alors être exprimée en termes des projections sur les axes principaux dans l'espace induit. Ces axes principaux sont en fait les axes de KPCA. Toutefois, KECA réalise la transformation des données et la réduction de dimension en projetant sur un sous-ensemble d'axes de KPCA qui contribuent le plus à l'entropie de Rényi. Ce sous-ensemble d'axes ne correspond pas, en général, aux plus grandes valeurs propres de la matrice noyau, comme c'est le cas avec la méthode de KPCA (Cf. section III.3.2).

Jenssen (2010) montre que le méthode de KECA peut produire un ensemble très différent de données transformées en comparaison avec la méthode de KPCA.

L'entropie de Rényi (Renyi - 1961) est donnée par :

$$H(p) = -\log \int p^2(\boldsymbol{x}) d\boldsymbol{x} \tag{III.20}$$

où $p(\boldsymbol{x})$ est la densité de probabilité des données de $\mathcal{A}_n$. Le logarithme étant une fonction monotone, on peut se concentrer uniquement sur la quantité

$$V(p) = \int p^2(\boldsymbol{x}) d\boldsymbol{x} = E\left[p(\boldsymbol{X})\right] \tag{III.21}$$

Afin d'estimer $V(p)$ (donc $H(p)$), un estimateur de densité à noyau de Parzen (1962) est introduit,

$$\hat{p}(\boldsymbol{x}) = \frac{1}{n} \sum_{\boldsymbol{x}_i \in \mathcal{A}_n} K_\sigma(\boldsymbol{x}, \boldsymbol{x}_i) \tag{III.22}$$

Le noyau de Parzen la plus largement utilisé est le noyau gaussien (Cf. équation III.3).

En estimant $V(p)$ dans l'équation III.21 au moyen des données de l'ensemble d'apprentissage $\mathcal{A}_n$, on obtient,

$$\begin{aligned} \hat{V}(p) &= \frac{1}{n} \sum_{\boldsymbol{x}_i \in \mathcal{A}_n} \hat{p}(\boldsymbol{x}_i) = \frac{1}{n} \sum_{\boldsymbol{x}_i \in \mathcal{A}_n} \frac{1}{n} \sum_{\boldsymbol{x}_j \in \mathcal{A}_n} K_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &= \frac{1}{n^2} \mathbf{1}^t \mathbf{K} \mathbf{1} \end{aligned} \tag{III.23}$$

où $\mathbf{K}$ est la matrice $(n \times n)$ de terme général $K_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)$ et $\mathbf{1}$ est un vecteur unitaire $(n \times 1)$. Il est évident que l'estimation de l'entropie de Rényi ainsi obtenue dépend entièrement des éléments de la matrice noyau (Girolami - 2002).

L'estimation de l'entropie de Rényi peut aussi s'exprimer en termes des valeurs et vecteurs propres de $\mathbf{K} = \mathbf{E}\mathbf{\Lambda}\mathbf{E}^t$, où $\mathbf{\Lambda}$ est la matrice diagonale des valeurs propres

$\lambda_1,\ \lambda_2,\ \ldots, \lambda_n$ stockées en ordre décroissant et $\mathbf{E}$ est la matrice des vecteurs (colonnes) propres correspondants $\mathbf{e}_1,\ \mathbf{e}_2,\ \ldots, \mathbf{e}_n$. L'équation (III.23) peut alors être reécrite par,

$$\hat{V}(p) = \frac{1}{n^2} \sum_{i=1}^{n} \left(\lambda_i^{\frac{1}{2}} \mathbf{e}_i^t \mathbf{1} \right)^2 \tag{III.24}$$

L'équation (III.24) illustre que l'estimation de l'entropie de Rényi est en fait composée de projections sur tous les axes principaux de KPCA c'est-à-dire avec les valeurs et vecteurs propres $(\lambda, \mathbf{e})$ de la matrice $\mathbf{K}$ de la méthode KPCA (Cf. équation III.15) pour une fonction noyau donnée. Chaque terme $\frac{1}{n^2} \left(\lambda_i^{\frac{1}{2}} \mathbf{e}_i^t \mathbf{1} \right)^2$ de cette expression exprime la contribution de chaque axe à l'entropie. Toutefois certains vecteurs et valeurs propres contribuent plus que d'autres. Cela implique qu'une grande valeur propre λ_i ne garantit pas une plus grande contribution à l'entropie.

En résumé, la méthode KECA se différencie de la méthode KPCA dans le mode de sélection des axes principaux utilisés et aussi par le fait que dans KECA, les données ne sont pas centrées dans l'espace tranformé. Les axes principaux qui contribuent les plus à l'estimation d'entropie de Rényi (KECA) portent plus d'information provenant de la densité de probabilité des données dans l'espace de représentation d'origine. En revanche, la méthode de KPCA repose uniquement sur les valeurs propres, et la transformation qui en résulte peut donc retenir des vecteurs propres non informatifs du point de vue de l'entropie.

III.3.4 Erreur de reconstruction

Bien que les méthode KPCA et KECA soient des techniques de transformation des données, elles peuvent aussi être utilisées à en classification (Braun et al. - 2008; Hoffmann - 2007). Hoffmann (2007) a proposé d'utiliser les surfaces ou plans équipotentiels de l'erreur de reconstruction comme frontières d'une règle de décision. Cette approche donne de meilleures performances que la méthode de one-class SVM.

La fonction noyau la plus largement utilisée est le noyau gaussien pour la méthode de one-class SVM et c'est donc celle que nous avons retenue pour l'appliquer aux méthodes KPCA et KECA. D'après la définition du noyau gaussien, $K(\boldsymbol{x}, \boldsymbol{x})$ prend la même valeur (constante) pour tous les $\boldsymbol{x}$. Il en résulte que toutes les projections $\phi(\boldsymbol{x})$ se trouvent sur la surface d'une hypersphère dans l'espace induit. Si nous exigeons que tous les points des données soient englobés par la frontière de décision (i.e. marges dures), le one-class SVM consiste à mettre un hyperplan aussi proche que possible de l'ensemble $\{\phi(\boldsymbol{x}_i)\}$ pour les séparer de l'origine (Cf. section III.3.1). Toutefois, une telle frontière de décision englobant ces données n'est pas optimale du point de vue de la compacité si ces données ont une variance anisotrope dans l'espace induit. En revanche, l'erreur de reconstruction des méthodes KPCA ou KECA tient compte de la variance de la distribution. La frontière de décision qui est orthogonale aux axes principaux déterminés par la méthode KPCA ou KECA est donc plus proche des courbes isodensité de la distribution que celle de one-class SVM. Autrement dit, pour le même nombre d'échantillons englobés, le domaine déterminé par l'erreur de reconstruction a un plus petit volume dans l'espace induit que celui déterminé par one-class SVM.

Ces arguments donnent aussi une idée pour choisir les paramètres de KPCA et KECA que sont le rayon σ du noyau et le nombre q de vecteurs propres à sélectionner.

La valeur de σ doit être choisie dans un intervalle de valeurs raisonables. Si par exemple σ est trop petit, on a $K(\boldsymbol{x}_i, \boldsymbol{x}_j) \approx 0 \, \forall i, j$ et $i \neq j$ et dans ce cas toutes les projections $\phi(\boldsymbol{x}_i)$ sont quasi deux à deux orthogonales, ce qui a pour conséquence que PCA devient inopérant. Dans le cas contraire, pour de grandes valeurs de σ, l'erreur de reconstruction dans l'espace induit se rapproche de celle de l'ACP standard dans l'espace original.

De plus, le nombre q doit être suffisamment grand afin que l'erreur de reconstruction reste dans un intervalle restreint pour l'ensemble des points.

L'erreur de reconstruction pour les deux méthodes KPCA et KECA (Diamantaras et Kung - 1996) est donnée par les formules suivantes,

$$\begin{cases} \varepsilon^{KPCA} = \langle \, \tilde{\Phi} \, , \, \tilde{\Phi} \, \rangle - \langle \, \tilde{\mathbf{U}}\tilde{\Phi} \, , \, \tilde{\mathbf{U}}\tilde{\Phi} \, \rangle \\ \varepsilon^{KECA} = \langle \, \Phi \, , \, \Phi \, \rangle - \langle \, \mathbf{U}\Phi \, , \, \mathbf{U}\Phi \, \rangle \end{cases} \tag{III.25}$$

où $\tilde{\Phi}$ (Cf. équation III.12) représente les données centrées dans l'espace induit. La matrice $\tilde{\mathbf{U}}$ contient les q vecteurs propres correspondant aux q plus grandes valeurs propres de la matrice $\tilde{\mathbf{K}}$ pour la méthode KPCA et $\mathbf{U}$ contient les q vecteurs propres de la matrice $\mathbf{K}$ qui contribuent les plus à l'entropie de Rényi pour la méthode KECA.

La projection d'une nouvelle observation $\boldsymbol{x}$ sur le $k^{ème}$ vecteur propre $\tilde{\mathbf{u}}_k$ (resp. $\mathbf{u}_k$) est mesurée par,

$$\begin{cases} \epsilon_k^{KPCA}(\boldsymbol{x}) = \langle \, \tilde{\Phi}(\boldsymbol{x}) \, , \, \tilde{\mathbf{u}}_k \, \rangle \\ \epsilon_k^{KECA}(\boldsymbol{x}) = \langle \, \Phi(\boldsymbol{x}) \, , \, \mathbf{u}_k \, \rangle \end{cases} \qquad k = 1, \ldots, q \tag{III.26}$$

En considérant l'équation III.19, cette projection est donnée par,

$$\begin{cases} \epsilon_k^{KPCA}(\boldsymbol{x}) = \lambda_k^{-\frac{1}{2}} \sum_{i=1}^{n} e_i^k \tilde{K}(\boldsymbol{x}_i, \boldsymbol{x}) \\ \epsilon_k^{KECA}(\boldsymbol{x}) = \lambda_k^{-\frac{1}{2}} \sum_{i=1}^{n} e_i^k K(\boldsymbol{x}_i, \boldsymbol{x}) \end{cases} \qquad k = 1, \ldots, q \tag{III.27}$$

À partir des équations III.25 et III.27, on peut déduire l'expression finale de l'erreur de reconstruction pour une nouvelle observation $\boldsymbol{x}$,

$$\begin{cases} \varepsilon^{KPCA}(\boldsymbol{x}) = \tilde{K}(\boldsymbol{x}, \boldsymbol{x}) - \sum_{k=1}^{q} \left(\epsilon_k^{KPCA}(\boldsymbol{x}) \right)^2 \\ \varepsilon^{KECA}(\boldsymbol{x}) = K(\boldsymbol{x}, \boldsymbol{x}) - \sum_{k=1}^{q} \left(\epsilon_k^{KECA}(\boldsymbol{x}) \right)^2 \end{cases} \tag{III.28}$$

Notons qu'il existe la relation (III.29) suivante entre les éléments des matrices $\mathbf{K}$ et $\tilde{\mathbf{K}}$ définies respectivement à partir des données non centrées et centrées (Cf. équation III.12)

$$\tilde{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \frac{1}{n} \sum_{r=1}^{n} K(\boldsymbol{x}_i, \boldsymbol{x}_r) - \frac{1}{n} \sum_{r=1}^{n} K(\boldsymbol{x}_r, \boldsymbol{x}_j) + \frac{1}{n^2} \sum_{r,s=1}^{n} K(\boldsymbol{x}_r, \boldsymbol{x}_s) \tag{III.29}$$

La classification de $\boldsymbol{x}$ se fait finalement en comparant $\varepsilon^{KPCA}(\boldsymbol{x})$ (resp. $\varepsilon^{KECA}(\boldsymbol{x})$) à un seuil donné.

III.4 Formalisme et contexte applicatif

III.4.1 Sous-espaces de représentation homogènes

Dans toute la suite, nous allons considérer le problème de la surveillance d'un système observé par m capteurs délivrant chacun un attribut. L'information disponible sera donc initialement décrite par un ensemble de données représentées chacune par une collection (ou vecteur ou point) de m mesures ou attributs décrits dans un espace de représentation $\mathcal{X}^m \subseteq \mathbb{R}^m$. Chaque donnée issue de l'observation du système sera considérée comme la réalisation $\boldsymbol{x}$ dans $\mathcal{X}^m$ d'un vecteur aléatoire $\boldsymbol{X} = [X_1\ ,\ X_2\ ,\ \ldots\ ,\ X_m]^t$. Chacune de ses composantes X_i sera considérée comme une variable aléatoire représentant l'attribut délivré par le $i^{ème}$ capteur.

Comme nous l'avons dit en introduction (section III.1) de ce chapitre, un système décisionnel idéal doit être robuste aux perturbations dues au bruit ou à des changements brutaux de comportement de certains attributs (pannes de capteurs ou changements locaux de l'état du sytème sous surveillance). Notre approche repose sur le fait qu'en situation d'environnement normal et non perturbé pour le système sous surveillance, l'ensemble des m variables aléatoires (attributs) obéissent conjointement au modèle probabiliste décrivant la classe normale. On dira dans ce cas que le modèle ou que l'espace $\mathcal{X}^m$ est homogène. Dans le cas contraire (i.e. environnement perturbé ou changements locaux de l'état du sytème), différentes situations peuvent se présenter pour lesquelles, seule une partie des m composantes de $\boldsymbol{X}$ seront aptes à décrire la classe normale. Les autres composantes de $\boldsymbol{X}$ peuvent être perturbées ou décrire un autre état du système. Dans cette dernière situation, l'espace complet $\mathcal{X}^m$ ne peut plus être qualifié d'homogène, mais on peut raisonnablement admettre qu'il existe des sous-espaces, définis à partir d'attributs non perturbés, qui pourront alors être qualifiés d'homogènes.

Lors de la phase d'apprentissage, nous supposons que l'espace initial $\mathcal{X}^m$ est homogène et que, par conséquent, tous les sous-espaces qui peuvent en être issus le sont aussi. Les classifieurs sont donc initialement tous construits dans des sous-espaces de représentation homogènes. Dans ce contexte, nous faisons l'hypothèse qu'en cas de perturbation de l'environnement ou de changement local d'état du système, les classifieurs qui opèrent dans des sous-espaces restés homogènes doivent être plus pertinents et plus fiables pour classer de nouvelles observations que les classifieurs opérant dans un sous-espace non homogène.

III.4.2 Formalisation en termes de modèles homogènes

Conformément à ce que nous avons présenté dans la section III.4.1, on peut expliciter les différentes situations en termes de modélisation probabiliste du vecteur aléatoire d'attributs $\boldsymbol{X}$. Soit donc ω_0 la classe normale (que nous représenterons parfois par l'hypothèse H_0 selon le contexte) et ω_1 (reps. hypothèse H_1) la classe de rejet. Dans

le contexte d'un environnement normal et en l'absence de perturbations et de pannes, toutes les composantes de $\boldsymbol{X}$ rendent compte de la classe normale ω_0 et l'espace de représentation initial $\mathcal{X}^m$ est donc homogène. Le vecteur aléatoire d'attributs $\boldsymbol{X}$ est alors modélisé par une densité conjointe,

$$\boldsymbol{X} \sim f(\boldsymbol{x} \mid \omega_0) \tag{III.30}$$

Partant de cette représentation, on peut envisager les situations suivantes correspondant à un environnement perturbé :

- Certains, voire tous les capteurs sont affectés par du bruit. Les composantes de $\boldsymbol{X}$ correspondant à ces capteurs vont alors contribuer à une modification du modèle conjoint de la loi de $\boldsymbol{X}$. Si on note $\boldsymbol{X}' = \boldsymbol{X} + \boldsymbol{b}$ où $\boldsymbol{b}$ représente le bruit, on aura alors
$$\boldsymbol{X}' \sim f'(\boldsymbol{x} \mid \omega_0) \tag{III.31}$$
où la densité f' n'est pas connue (ou apprise) a priori.
- Certains capteurs "captent" un nouvel état du système (changement local d'état) ou bien sont en panne et n'émettent alors que du bruit, tandis que le reste se trouve dans l'environnement de la classe normale ω_0. Dans ce cas, on peut scinder $\boldsymbol{X}$ en deux blocs $\boldsymbol{X}_0$ et $\boldsymbol{X}_1$ tels que :
$$\boldsymbol{X}_0 \sim f(\boldsymbol{x}_0 \mid \omega_0) \quad \text{et} \quad \boldsymbol{X}_1 \sim f(\boldsymbol{x}_1 \mid \omega_1) \tag{III.32}$$
et $\boldsymbol{X}$ obéit alors à un modèle intermédiaire et inconnu
$$\boldsymbol{X} = [\boldsymbol{X}_0^t, \boldsymbol{X}_1^t]^t \sim f(\boldsymbol{x} \mid \omega_0 \cup \omega_1) = f([\boldsymbol{x}_0^t, \boldsymbol{x}_1^t]^t \mid \omega_0 \cup \omega_1) \tag{III.33}$$

Bien entendu, on peut imaginer beaucoup plus de situations que celles présentées ici, mais nous avons choisi pour cette étude de nous limiter exclusivement à la seconde situation (changement local d'état) qui peut être illustrée et simulée au moyen des transitions spatiales d'une région à une autre que l'on peut observer dans les images par exemple (Cf. section III.5.1).

L'expression (III.33) illustre le fait que dans un contexte d'environnement perturbé ou de changement local d'état, il peut exister un ou plusieurs sous-espaces issus de $\mathcal{X}^m$ (ici celui auquel appartient le vecteur $\boldsymbol{X}_0$) dans lesquels les attributs ne sont pas perturbées et obéissent donc à un modèle conforme à celui qui a fait l'objet d'un apprentissage.

III.4.3 Construction de la règle de décision finale

Nous avons opté pour le principe de la méthode RSM pour générer dans un premier temps un ensemble de L sous-espaces de représentation $\{\mathcal{X}_\ell^d \subset \mathcal{X}^m \mid \ell = 1, \ldots, L\}$ en effectuant L tirages aléatoires de d attributs parmi m ($d < m$). L'ensemble d'apprentissage $\mathcal{A}_n$ est ensuite projeté dans chaque sous-espace de représentation, donnant ainsi un ensemble de L ensembles d'apprentissage $\{\mathcal{A}_n^1, \mathcal{A}_n^2, \ldots, \mathcal{A}_n^L\}$. Chaque sous-ensemble d'apprentissage $\mathcal{A}_n^\ell$ est dédié à la mise au point d'un classifieur h_ℓ dans le sous-espace

$\mathcal{X}_\ell^d$. Nous avons, pour chaque ensemble $\mathcal{A}_n^\ell$, testé les trois algorithmes d'apprentissage, one-class SVM, KPCA et KECA qui ont été présentés dans cette section.

Pour une observation $\boldsymbol{x} \in \mathcal{X}^m$ à classer et une méthode d'apprentissage donnée on va donc obtenir L décisions qui seront prises sur la base des L projections $\boldsymbol{x}^{(\ell)}$ de $\boldsymbol{x}$ dans chaque sous-espace $\mathcal{X}_\ell^d$.

- Avec la méthode de one-class SVM (que l'on notera OSVM), l'expression du classifieur est donnée par (Cf. section III.3.1),

$$h_\ell^{OSVM}(\boldsymbol{x}^{(\ell)}) = \langle\ \boldsymbol{w}^{(\ell)}\ ,\ \phi(\boldsymbol{x}^{(\ell)})\ \rangle \tag{III.34}$$

- Pour la méthode KPCA, le classifieur s'exprime par la mesure de l'erreur de reconstruction (Cf. sections III.3.2 et III.3.4),

$$h_\ell^{KPCA}(\boldsymbol{x}^{(\ell)}) = \varepsilon^{KPCA}(\boldsymbol{x}^{(\ell)}) \tag{III.35}$$

- Le classifieur de la méthode KECA est construit de la même manière que celui de KPCA que (Cf. sections III.3.3 et III.3.4),

$$h_\ell^{KECA}(\boldsymbol{x}^{(\ell)}) = \varepsilon^{KECA}(\boldsymbol{x}^{(\ell)}) \tag{III.36}$$

La décision finale est ensuite prise au moyen du vote majoritaire au moyen de la fonction décisionnelle suivante,

$$f(\boldsymbol{x}) = sgn\left(\sum_{\ell=1}^{L} h_\ell(\boldsymbol{x}^{(\ell)}) - \theta\right) \tag{III.37}$$

Dans l'expression (III.37) de f, θ est un seuil qui joue le rôle du paramètre ρ dans le cas de la méthode OSVM (Cf. équation III.9 page 45) et qui représente une borne supérieure sur l'erreur de reconstruction pour les méthodes KPCA et KECA (Cf. équations III.28 page 50).

III.4.4 Processus d'évaluation

Nous présentons ici le protocole d'évaluation des performances des trois règles de décision que nous avons conçu afin de les comparer. Pour cela on considère que l'on dispose d'un ensemble d'apprentissage $\mathcal{A}_n$, constitués de n exemples de la classe normale ω_0, ainsi que de deux ensembles de test $\mathcal{T}_{n_0}$ et $\mathcal{T}_{n_1}$, constitués respectivement de n_0 exemples de la classe normale ω_0 et n_1 exemples de la classe de rejet ω_1. Par ailleurs, pour une nouvelle réalisation $\boldsymbol{x}$ à classer et pour chacune des trois méthodes, la règle de décision globale reposant sur f (Cf. III.37) sera définie telle que :

$$\begin{cases} \boldsymbol{x} & \text{est reconnu comme élément de } \omega_0 & \text{si} & f(\boldsymbol{x}) = 0 \\ \boldsymbol{x} & \text{est rejeté (classé dans } \omega_1) & \text{si} & f(\boldsymbol{x}) = 1 \end{cases} \tag{III.38}$$

Comme l'évaluation des performances s'effectue selon le même processus pour les trois algorithmes d'apprentissage, nous ne le détaillons ici que pour la méthode OSVM. Les différentes étapes sont décrites dans le cadre du schéma III.2

- Pour un couple donné de paramètres (ν, σ) de la méthode OSVM
  1. On construit les classifieurs h_ℓ^{OSVM} :
     apprentissage de $\boldsymbol{w}^{(\ell)}$ et des $\alpha_i^{(\ell)}$ sur $\mathcal{A}_n^\ell$ pour $\ell = 1, \ldots, L$
  2. Détermination du seuil θ (Cf. équation III.37) pour une probabilité d'erreur de première espèce α (Cf. tableau II.1 page 35) donnée :
     $\theta = \rho_\alpha(\nu, \sigma)$ est ajusté au moyen des éléments $\boldsymbol{x}_{k_0} \in \mathcal{T}_{n_0}$, de sorte que

$$\alpha = P\left[f(\boldsymbol{x}) = 1 \mid \boldsymbol{x} \in \omega_0\right]$$

$$\text{avec} \quad f(\boldsymbol{x}) = sgn\left(\sum_{\ell=1}^{L} \langle\ \boldsymbol{w}^{(\ell)}\ ,\ \phi(\boldsymbol{x}^{(\ell)})\ \rangle - \rho_\alpha(\nu, \sigma)\right)$$

  3. Estimation de la probabilité d'erreur de seconde espèce β (Cf. tableau II.1) :
     β est estimé en appliquant aux éléments $\boldsymbol{x}_{k_1} \in \mathcal{T}_{n_1}$ la fonction décisionnelle f mise au point à l'étape 2 avec le seuil $\theta = \rho_\alpha(\nu, \sigma)$
- On répète les étapes 1 à 3 précédentes afin de trouver le couple optimal (ν^*, σ^*) tel que $\beta_\alpha(\nu^*, \sigma^*) = \min_{\nu,\sigma}\left(\beta_\alpha(\nu, \sigma)\right)$ à α fixé.

Pour les méthodes reposant sur l'erreur de reconstruction de KPCA et KECA, le couple de paramètres optimaux à déterminer est (q, σ), où q est le nombre d'axes principaux sélectionnés.

Schéma III.2 – Processus d'évaluation des performances pour la méthode OSVM

III.5 Expérimentations : application à la segmentation d'images texturées

La segmentation d'images texturées constitue une application tout à fait appropriée pour illustrer les méthodes de classification que nous avons décrites et pour évaluer leurs performances car elle permet de simuler facilement des distributions conjointes complexes et un environnement non-stationnaire. En effet, beaucoup de textures peuvent être considérées comme la réalisation d'un processus stochastique dont le modèle repose sur un ensemble de paramètres attachés à un domaine spatial très localisé souvent nommé voisinage (Cross et Jain - 1983; Geman et Geman - 1984; Unser - 1984). On peut dans ce cas admettre aisément que la stationnarité du modèle sera rompue à la frontière entre deux régions de textures différentes. Cette situation correspond à la formulation (III.33) proposée à la section III.4.2. Nous avons donc testé les trois méthodes présentées à la section III.3 en les appliquant à la segmentation d'images texturées selon le protocole décrit par le schéma III.2. Les images testées étant monochromes, les attributs caractérisant les données seront les niveaux de gris attachés à chaque pixel et les données à classer seront les pixels de l'image pour lesquels le classifieur devra

fournir en sortie le label de la classe de texture à laquelle il appartient. Les images sont constituées de deux classes de texture spatialement séparées par des frontières de formes quelconques et complexes.

III.5.1 Définition des espaces de représentation

Nous considérons donc une texture comme la réalisation d'un processus stochastique $\{X_s \mid s \in S\}$, où S est l'ensemble des sites occupés par les pixels de l'image à traiter et X_s est une variable aléatoire attachée au site s et représentant l'état du pixel correspondant. Les variables X_s sont discrètes et prennent leurs valeurs dans un ensemble Ω qui décrit la plupart du temps la palette des niveaux de gris pour les images monochromes.

Par ailleurs, les textures que nous avons testées sont entièrement caractérisées au niveau local. Plus précisément, dans le cadre d'un processus stochastique, cela signifie que l'état X_s d'un pixel ne dépend que de l'état d'un nombre limité de pixels distribués spatialement dans ce que l'on appelle un voisinage de s. Nous avons opté pour la structure la plus courante de voisinage, un voisinage carré de $m = p \times p$ pixels centrés sur le pixel à classer et l'espace de représentation initial qui en résulte est l'espace $\mathcal{X}^m$ des m variables X_t attachées aux pixels du voisinage. Le vecteur d'attributs caractérisant un pixel s sera donc représenté par $\boldsymbol{X}^{(s)} = \left[X_1^{(s)}\ ,\ X_2^{(s)}\ ,\ \ldots\ ,\ X_m^{(s)}\right]^t$ avec $m = p^2$.

Un ensemble de sous-espaces de représentation de dimension d pourra alors être obtenu en sélectionnant d pixels parmi les m appartenant au voisinage (Cf. exemple à la figure III.2).

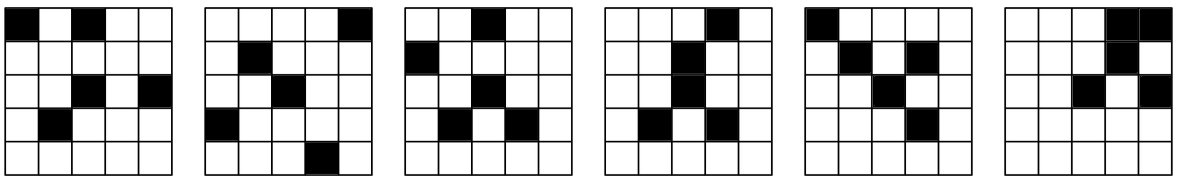


Figure III.2 – 6 sous-espaces de dimension $d = 5$ tirés d'un voisinage 5×5

Pour faire le lien avec les notions de modèle homogène et non homogène qui ont été présentées à la section III.4.2, nous sommes amenés à distinguer deux types de zones dans le plan image qui correspondent aux situations de modèle homogène et non homogène décrites respectivement par les expressions (III.30) et (III.33). En considérant deux classes ω_0 et ω_1, on distinguera donc dans la suite les deux zones suivantes :

- **« Zone cœur » :** c'est l'ensemble des pixels d'une image dont les voisins appartiennent à la même classe que lui. Elle caractérise les régions homogènes.

$$\boldsymbol{X}^{(s)} \sim f(\boldsymbol{x}^{(s)} \mid \omega_0) \qquad \text{ou bien} \qquad \boldsymbol{X}^{(s)} \sim f(\boldsymbol{x}^{(s)} \mid \omega_1) \tag{III.39}$$

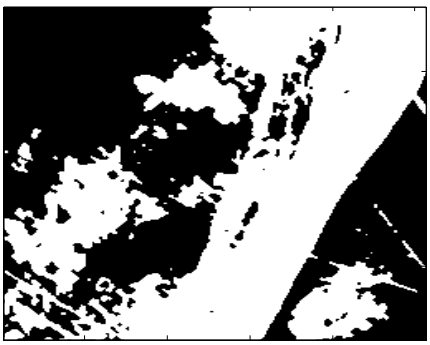
- **« Zone frontière » :** c'est l'ensemble des pixels d'une image dont une partie des voisins appartiennent à une classe et le reste des voisins appartiennent à l'autre classe. Elle caractérise les régions non homogènes.

$$\boldsymbol{X}^{(s)} \sim f(\boldsymbol{x}^{(s)} \mid \omega_0 \cup \omega_1) = f([\boldsymbol{x}_0^{(s)}, \boldsymbol{x}_1^{(s)}] \mid \omega_0 \cup \omega_1) \tag{III.40}$$

Le principe de sélection des sous-espaces de représentation illustré par la figure III.2 permet, comme nous le souhaitons, d'avoir la possibilité de prendre des décisions dans des sous-espaces homogènes lorsque l'environnement capté par l'espace complet $\mathcal{X}^m$ ne l'est pas.

III.5.2 Base d'images texturées

Les images que nous avons choisies pour nos expériences sont composées d'une texture qui représente la classe normale ω_0 (en noir sur la figure III.3(a)) et d'un bruit uniforme qui représente la classe de rejet ω_1 (en blanc sur la figure III.3(a)). Les régions contenant ω_0 et ω_1 sont séparées par des frontières complexes. La « zone frontière » (en blanc sur la figure III.3(b)) est constituée, pour un voisinage $p \times p$ d'une bande de largeur $p-1$ pixels centrée sur les frontières réelles visibles sur la figure III.3(a). La « zones cœur » est représentée en noir sur la figure III.3(b).



(a) Carte des régions : ω_0 en noir et ω_1 en blanc

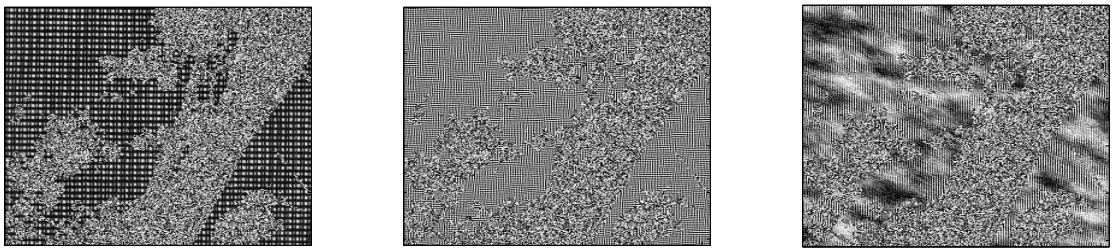
(b) Carte des zones « cœur » en noir et « frontière » en blanc

Figure III.3 – Images de la cartographie réelle des labels

Toutes les textures utilisées sont monochromes et quantifiées sur 256 niveaux de gris, $X_s \in \Omega = \{0, 1, \ldots, 255\}$. Ce sont, soit des textures naturelles tirées de l'album de Brodatz (1966), soit des textures de synthèse obtenues par simulation d'un champ de Markov (Smolarz - 1997).

Nous avons construit trois exemples d'images à segmenter qui sont présentés à la figure III.4 et décrites dans le tableau III.3. Ces images sont de taille 256×256 pixels.

Pour l'ensemble d'apprentissage $\mathcal{A}_n$, nous avons sélectionné aléatoirement $n = 500$ exemples et $n_0 = 3000$ exemples pour l'ensemble de test $\mathcal{T}_{n_0}$. Ces exemples sont des vecteurs de dimension m extraits d'une image complète de chaque texture originale (classe ω_0). Les images ayant fourni les exemples d'apprentissage ne sont pas incluses dans les images à segmenter.



(a) (b) (c)

Figure III.4 – Images à segmenter

Image de test	Origine de texture
Fig.III.4(a)	D21 de Brodatz (1966) (ω_0) et bruit uniforme (ω_1)
Fig.III.4(b)	Texture markovienne (Smolarz - 1997) (ω_0) et bruit uniforme (ω_1)
Fig.III.4(c)	Texture markovienne (Smolarz - 1997) (ω_0) et bruit uniforme (ω_1)

Tableau III.3 – Description des images testées

III.5.3 Détermination des paramètres

III.5.3.1 Paramètres du système d'ensemble via la définition des sous-espaces de représentation

Dans le cadre de l'approche que nous avons adoptée associant un ensemble de sous-espaces à un ensemble de classifieurs, il nous faut fixer 3 paramètres : la dimension m de l'espace de représentation initial (c'est-à-dire ici la taille $p \times p$ du voisinage), la dimension d des sous-espaces de représentation et le nombre total L de ces sous-espaces (i.e. nombre de classifieurs individuels).

Dimension m de l'espace de représentation initial (taille du voisinage)

Le choix de la taille du voisinage joue un rôle important pour la construction du modèle de texture, mais les simulations (Cross et Jain - 1983; Geman et Geman - 1984; Smolarz - 1997) montrent qu'un voisinage 5×5 est très souvent suffisant. Par ailleurs, le choix d'un voisinage de grande taille, s'il permet a priori de capter davantage d'information qu'un petit voisinage, s'avèrera réellement efficace dans les zones « cœur », mais risque de présenter une grande instabilité au niveau des frontières. En revanche, le choix d'un voisinage de petite taille permet de bien contrôler les perturbations aux frontières, mais risque d'engendrer en contrepartie une dégradation de la performance dans les régions homogènes (Kim et al. - 2002; Beauseroy et Smolarz - 2004). La détermination de la dimension $m = p \times p$ repose donc sur un compromis entre les deux aspects évoqués ici.

Dimension d des sous-espace de représentation

La dimension d peut être choisie aléatoirement pour chaque sous-espace ou bien être la même pour tous les sous-espaces et déterminée par l'expérience (He et al. - 2010). Lorsque m est donné, le choix de d fait face au même problème que le choix de la taille du voisinage. Une grande valeur de d peut garantir et améliorer la capacité de classification dans les régions homogènes, mais entraîner une mauvaise performance aux frontières car la probabilité de sélectionner des attributs altérés augmente avec d. Dans nos expérimentations, nous avons testé deux stratégies décrites à la fin de cette section.

Nombre L de sous-espace de représentation (classifieurs)

En ce qui concerne le dernier paramètre (le nombre L de classifieur), il est expérimentalement admis (Ho - 1993, 1995, 1998a) que le taux d'erreur de classification diminue lorsqu'on augmente du nombre de classifieurs. Toutefois, l'augmentation du nombre de classifieurs entraîne une augmentation de la durée d'exécution. Là encore on cherchera un compromis entre la performance souhaitée et le coût de calcul.

En nous inspirant des travaux de He (2009), nous avons testé deux stratégies de choix des paramètres :

Stratégie 1°

- dimension de l'espace de représentation initial : $m = p \times p = 5 \times 5 = 25$;
- dimension de chaque sous-espace de représentation : $d = 5$;
- nombre total de sous-espaces de représentation extraits aléatoirement (classifieurs) : $L = 100$.

Stratégie 2°

- dimension de l'espace de représentation initial : $m = p \times p = 5 \times 5 = 25$;
- dimensions différentes de sous-espace de représentation : $d = 3,\ 5,\ 7$;
- nombre total de sous-espaces de représentation (classifieurs) : $L = 120$ dont L_3 correspondant à $d = 3$, L_5 correspondant à $d = 5$ et L_7 correspondant à $d = 7$.
- trois combinaisons (L_3, L_5, L_7) sont testées : $(20, 40, 60)$, $(40, 40, 40)$ et $(60, 40, 20)$.

III.5.3.2 Paramètres des algorithmes d'apprentissage testés

Pour les trois algorithmes d'apprentissage testés (one-class SVM, KPCA et KECA), la fonction du noyau joue un rôle crucial dans le système décisionnel, car c'est elle qui définit l'espace induit où la classification est effectuée. D'après beaucoup d'études (Schölkopf et al. - 2001; Tax et Duin - 2004; Hoffmann - 2007), la fonction de base radiale (i.e. *RBF kernel*), particulièrement le noyau gaussien, semble mieux adaptée que les autres (noyau polynomial par exemple) et est alors largement utilisée dans les applications de SVM et l'analyse en composante principale pour le cas non-linéaire. Pour ces raisons, nous avons choisi le noyau gaussien dans nos expériences pour les

trois algorithmes testés.

$$K(\boldsymbol{x}, \boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\sigma^2}} \tag{III.41}$$

III.5.3.2.1 Paramètres de one-class SVM

Les deux paramètres associés à la méthode de one-class SVM sont le paramètre ν (Cf. équation III.4) et le paramètre σ (Cf. équation III.41) du noyau. Comme nous l'avons déjà évoqué à la section III.3.1, $\nu \in [0, 1]$ est une borne supérieure du taux d'anomalies et aussi une borne inférieure du taux de vecteurs supports. Le taux d'anomalies est en fait la fraction d'erreurs de classification acceptable de la classe positive dans l'apprentissage et le nombre de vecteurs supports contrôle souvent la performance de généralisation de l'algorithme.

Lorsque ν est donné, la valeur adéquate de σ est ensuite déterminée par une recherche d'optimisation. Dans notre problème, pour chaque image texturée, nous prédéfinissons d'abord certaines valeurs de ν et σ pour la sélection, comme le montre le tableau III.4. Puis différentes combinaisons des deux paramètres sont examinées et nous choisissons celle qui produit le minimum de la probabilité de non détection $\beta_\alpha(\nu, \sigma)$ sous la probabilité de fausse alarme α fixée a priori selon le processus de l'évaluation (Cf. section III.4.4, schéma III.2). Nous avons choisi ici de faire les tests avec deux valeurs $\alpha = 0.01$ et $\alpha = 0.05$.

Paramètres	Valeurs prédéfinies
ν	0.01, 0.05, 0.1, 0.2
σ	$[10 : 5 : 150]$

Tableau III.4 – Les valeurs prédéfinies pour les paramètres ν et σ

Stratégie 1° : Le tableau III.5 montre les combinaisons optimales des valeurs (ν, σ) pour chaque image test (Cf. figures III.4(a), III.4(b) et III.4(c)) avec $\alpha = 0.01$ et $\alpha = 0.05$, $m = 5 \times 5 = 25$, $d = 5$ pour chaque sous-espace et $L = 100$.

Image de test	Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)
$(\nu, \sigma)_{\alpha=0.01}$	(0.01, 40)	(0.01, 40)	(0.01, 55)
$(\nu, \sigma)_{\alpha=0.05}$	(0.01, 40)	(0.01, 40)	(0.01, 40)

Tableau III.5 – Combinaisons optimales des paramètres (ν, σ) avec la stratégie 1

Stratégie 2° : On utilise ici 3 valeurs différentes de d (dimension des sous-espaces de représentation). Les combinaisons optimales des valeurs (ν, σ) avec $\alpha = 0.01$ et $\alpha = 0.05$ sont données dans le tableau III.6.

Par ailleurs, afin d'illustrer l'intérêt de travailler dans des sous-espaces de représentation, nous avons également appliqué la règle de one-class SVM opérant sur l'espace complet $\mathcal{X}^m$. Les paramètres de ce classifieur sont déterminés selon le même processus d'évaluation que celui décrit dans la section III.4.4 et sont présentés dans le tableau

III.7. Dans la présentation des résultats, cette méthode sera notée OSVM_$\mathcal{X}^m$, tandis que celle opérant sur L sous-espaces sera notée simplement OSVM.

		$\alpha = 0.01$			$\alpha = 0.05$		
Image de test		Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)	Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)
$d = 3$	$L_3 = 20$	$(0.05, 40)$	$(0.01, 40)$	$(0.05, 40)$	$(0.01, 25)$	$(0.01, 40)$	$(0.01, 40)$
	$L_3 = 40$	$(0.05, 40)$	$(0.01, 40)$	$(0.05, 40)$	$(0.01, 25)$	$(0.01, 25)$	$(0.01, 40)$
	$L_3 = 60$	$(0.01, 40)$	$(0.05, 40)$	$(0.01, 40)$	$(0.01, 25)$	$(0.01, 40)$	$(0.01, 40)$
$d = 5$	$L_5 = 40$	$(0.01, 40)$	$(0.01, 40)$	$(0.01, 55)$	$(0.01, 55)$	$(0.01, 40)$	$(0.01, 40)$
$d = 7$	$L_7 = 20$	$(0.01, 40)$	$(0.01, 55)$	$(0.01, 55)$	$(0.01, 40)$	$(0.01, 55)$	$(0.01, 55)$
	$L_7 = 40$	$(0.01, 40)$	$(0.01, 55)$	$(0.01, 55)$	$(0.01, 40)$	$(0.01, 40)$	$(0.01, 55)$
	$L_7 = 60$	$(0.01, 40)$	$(0.01, 55)$	$(0.01, 55)$	$(0.01, 40)$	$(0.01, 40)$	$(0.01, 55)$

Tableau III.6 – Combinaisons optimales des paramètres (ν, σ) avec la stratégie 2

Image de test	Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)
$(\nu, \sigma)_{\alpha=0.01}$	(0.01, 90)	(0.01, 90)	(0.01, 105)
$(\nu, \sigma)_{\alpha=0.05}$	(0.01, 90)	(0.01, 90)	(0.01, 90)

Tableau III.7 – Combinaisons optimales des paramètres pour la méthode OSVM_$\mathcal{X}^m$

III.5.3.2.2 Paramètres de KPCA et KECA

Hormis le paramètre σ du noyau, l'autre paramètre spécifique aux méthodes KPCA et KECA est le nombre q de vecteurs propres utilisés. Avec le noyau gaussien, le choix d'une valeur trop faible pour σ peut entraîner que toutes les projections $\phi(\boldsymbol{x})$ d'exemples dans l'espace induit soient orthogonales entre elles et l'ACP devient donc inutile. En revanche, une grande valeur de σ donnera des résultats proches de l'ACP standard. Par ailleurs, la mise en œuvre des méthodes KPCA et KECA est assez lourde en temps de calculs. La plupart du temps d'exécution est consacré à l'extraction des vecteurs propres de la matrice du noyau $\mathbf{K}$. La complexité est de l'ordre de $O(n^3)$ (n est le nombre d'exemples d'apprentissage ou de test) si on extrait tous les vecteurs propres (Press et al. - 1993). Par conséquent, nous avons prédéfini des valeurs de q et σ qui nous semblent donner un bon compromis dans le contexte décrit ici (Cf. tableau III.8). De la même manière que pour la méthode OSVM, les paramètres optimaux ont été déterminés selon les deux stratégies (tableaux III.9 et III.10).

Paramètres	Valeurs prédéfinies
q	20, 50, 100, 150
σ	$[10 : 15 : 150]$

Tableau III.8 – Valeurs prédéfinies pour les paramètres q et σ

(a) Paramètres pour KPCA

Image de test	Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)
$(q,\sigma)_{\alpha=0.01}$	(100, 40)	(100, 40)	(50, 55)
$(q,\sigma)_{\alpha=0.05}$	(50, 25)	(100, 25)	(100, 40)

(b) Paramètres pour KECA

Image de test	Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)
$(q,\sigma)_{\alpha=0.01}$	(100, 40)	(100, 40)	(50, 55)
$(q,\sigma)_{\alpha=0.05}$	(50, 25)	(100, 25)	(100, 40)

Tableau III.9 – Combinaisons optimales des paramètres (q,σ) avec la stratégie 1

(a) Paramètres pour KPCA

		$\alpha=0.01$			$\alpha=0.05$		
Image de test		Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)	Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)
$d=3$	$L_3=20$	(100, 40)	(100, 40)	(100, 40)	(100, 40)	(100, 40)	(100, 40)
	$L_3=40$	(100, 40)	(100, 40)	(50, 55)	(100, 40)	(100, 40)	(100, 40)
	$L_3=60$	(50, 40)	(50, 40)	(50, 55)	(50, 40)	(50, 40)	(50, 40)
$d=5$	$L_5=40$	(100, 55)	(100, 40)	(50, 55)	(100, 25)	(100, 40)	(50, 55)
$d=7$	$L_7=20$	(100, 40)	(50, 55)	(100, 55)	(50, 40)	(50, 55)	(100, 55)
	$L_7=40$	(50, 40)	(100, 40)	(100, 55)	(50, 25)	(100, 40)	(100, 40)
	$L_7=60$	(100, 40)	(50, 55)	(100, 55)	(50, 40)	(50, 40)	(100, 40)

(b) Paramètres pour KECA

		$\alpha=0.01$			$\alpha=0.05$		
Image de test		Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)	Fig.III.4(a)	Fig.III.4(b)	Fig.III.4(c)
$d=3$	$L_3=20$	(100, 40)	(100, 40)	(100, 40)	(100, 40)	(100, 40)	(100, 40)
	$L_3=40$	(100, 40)	(100, 40)	(50, 55)	(100, 40)	(100, 40)	(100, 40)
	$L_3=60$	(50, 40)	(50, 40)	(50, 55)	(50, 40)	(50, 40)	(50, 40)
$d=5$	$L_5=40$	(100, 55)	(100, 40)	(50, 55)	(100, 25)	(100, 40)	(50, 55)
$d=7$	$L_7=20$	(100, 40)	(50, 55)	(100, 55)	(50, 40)	(50, 55)	(100, 55)
	$L_7=40$	(50, 40)	(100, 40)	(100, 55)	(50, 25)	(100, 40)	(100, 40)
	$L_7=60$	(100, 40)	(50, 55)	(100, 55)	(50, 40)	(50, 40)	(100, 40)

Tableau III.10 – Combinaisons optimales des paramètres (q,σ) avec la stratégie 2

III.5.4 Étude des résultats expérimentaux

Pour chaque image de test, afin d'évaluer les performances des quatre méthodes (OSVM_$\mathcal{X}^m$, OSVM, KPCA et KECA), nous distinguerons les pixels situés au cœur d'une classe (ω_0 ou ω_1) et ceux situés aux frontières entre les classes (Cf. définitions

des zones « cœur » et « frontière » à la section III.5.1). Les performances sont évaluées pour chaque méthode par l'estimation des quatre probabilités d'erreurs suivantes :

$$\begin{cases} \alpha_C = P\left[f(\boldsymbol{x}) = 1 \mid \boldsymbol{x} \in \omega_0 \cap \text{cœur}\right] \\ \beta_C = P\left[f(\boldsymbol{x}) = 0 \mid \boldsymbol{x} \in \omega_1 \cap \text{cœur}\right] \\ \alpha_F = P\left[f(\boldsymbol{x}) = 1 \mid \boldsymbol{x} \in \omega_0 \cap \text{frontière}\right] \\ \beta_F = P\left[f(\boldsymbol{x}) = 0 \mid \boldsymbol{x} \in \omega_1 \cap \text{frontière}\right] \end{cases} \tag{III.42}$$

III.5.4.1 Analyse des résultats avec la stratégie 1°

Rappelons qu'ici, nous avons fixé le même nombre d'attributs ($d = 5$) pour tous les sous-espaces de représentation avec un espace initial $\mathcal{X}^m$ de dimension $m = 25$. L'objectif est d'illustrer l'intérêt de la méthode d'ensemble de classifieurs reposant sur des sous-espaces de représentation et en particulier d'analyser sa robustesse en environnement non-stationnaire, c'est-à-dire dans les zones frontières. Le tableau III.11 présente les estimations (en %) des quatre taux d'erreurs (Cf. expressions III.42) pour chaque image de test et pour chaque méthode avec $\alpha = 0.01$ et $\alpha = 0.05$ fixés a priori. Les meilleurs résultats y figurent en gras.

(a) Avec $\alpha = 0.01$ (1%)

Fig.III.4(a)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$	**0.49**	**0.00**	72.36	**0.58**
OSVM	0.51	0.04	40.90	2.03
KPCA	0.64	0.18	25.45	4.85
KECA	0.59	0.22	**24.46**	5.34
Fig.III.4(b)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$	**0.67**	**0.00**	67.24	**0.56**
OSVM	1.11	0.09	47.09	1.34
KPCA	0.97	0.63	30.89	3.56
KECA	0.98	0.65	**30.47**	3.64
Fig.III.4(c)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$	0.97	**0.11**	58.87	**1.79**
OSVM	1.02	2.29	34.82	7.45
KPCA	0.77	4.75	23.38	11.33
KECA	**0.76**	5.01	**22.52**	11.87

(b) Avec $\alpha = 0.05$ (5%)

Fig.III.4(a)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$	**4.03**	**0.00**	82.12	**0.30**
OSVM	5.13	**0.00**	61.30	0.44
KPCA	4.31	0.08	**28.38**	4.74
KECA	4.24	0.05	29.83	4.09
Fig.III.4(b)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$	**4.60**	**0.00**	78.21	**0.27**
OSVM	4.88	0.01	61.80	0.48
KPCA	4.70	0.78	**31.12**	3.69
KECA	4.96	0.59	33.43	3.18
Fig.III.4(c)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$	5.62	**0.04**	72.07	**0.49**
OSVM	5.37	0.27	52.33	1.98
KPCA	**5.26**	0.70	40.00	3.68
KECA	5.40	0.73	**39.60**	3.72

Tableau III.11 – Taux d'erreurs (%) correspondant à la stratégie 1°

Indépendamment de α, les résultats sont similaires pour les quatre méthodes en termes des quatre probabilités d'erreurs. Si on prend par exemple $\alpha = 0.01$ (Cf. tableau

III.11(a)), nous pouvons constater que la plupart des erreurs importantes sont obtenues dans les zones frontières (i.e. les estimations $\hat{\alpha}_F$, $\hat{\beta}_F$ sont très supérieures à celles de $\hat{\alpha}_C$ et $\hat{\beta}_C$).

Si nous ne considérons ici que les valeurs des taux d'erreurs dans les zones cœurs ($\hat{\alpha}_C$ et $\hat{\beta}_C$), nous pouvons observer que la méthode OSVM_$\mathcal{X}^m$ est souvent plus performante que celles qui opèrent dans des sous-espaces. Ce résultat est assez logique puisque la texture est décrite dans ce cas par un plus grand nombre d'attributs, ce qui contribue à améliorer les perfomances du système décisionnel dans les zones cœurs.

En revanche, pour ce qui concerne la reconnaissance de la classe texture (ω_0) dans les zones frontières ($\hat{\alpha}_F$), les méthodes opérant dans des sous-espaces s'avèrent bien plus performantes. L'amélioration la plus marquée par rapport au résultat de la méthode OSVM_$\mathcal{X}^m$ est obtenue pour l'image de la figure III.4(a) (de 72.36% pour OSVM_$\mathcal{X}^m$ à 24.46% pour KECA, soit un gain relatif de 66.2%). Cela conforte notre hypothèse selon laquelle le vecteur complet d'attributs ne peut plus représenter la classe normale dans les zones frontières et illustre bien l'intérêt de la sélection d'un ensemble de sous-eapces de représentation adaptés aux situations de changement de modèle. En effet, dans cette situation, une partie seulement des attributs décrit la classe normale, tandis que le reste est altéré. La sélection de sous-espaces permet alors de prendre des décisions sur la base d'ensembles d'attributs tous conformes à l'apprentissage, réduisant ainsi les risques de mauvaise décision.

Concernant les trois méthodes OSVM, KPCA et KECA, nous pouvons constater que les taux d'erreurs $\hat{\alpha}_C$, $\hat{\beta}_C$ et $\hat{\beta}_F$ sont assez comparables, tandis que les méthodes KPCA et KECA s'avèrent plus performantes pour la reconnaissance de la classe normale dans les zones frontières ($\hat{\alpha}_F$). Toujours sur l'image de la figure III.4(a), on observe une baisse du taux d'erreur de 40.90% pour OSVM à 24.46% pour KECA, soit un gain relatif de 40.2% en faveur de KECA. Cela confirme que les frontières de décision déterminées par les méthodes KPCA et KECA reposant sur l'erreur de reconstruction sont plus compactes que celle de OSVM (Hoffmann - 2007). Nous pouvons noter également que la méthode KPCA donne des performances similaires à la méthode KECA.

III.5.4.2 Analyse des résultats avec la stratégie 2°

Dans la stratégie 2°, nous choisissons des valeurs différentes de la dimension des sous-espaces de représentation ($d = 3, 5, 7$ toujours avec $m = 25$) dont chacune est appliquée à une proportion variable (respectivement L_3, L_5, L_7) des $L = 120$ sous-espaces au total. Le but principal est d'étudier l'impact des proportions différentes (L_3, L_5, L_7) sur la performance de la méthode proposée en présence de perturbations. Les résultats des quatre taux d'erreurs reposant sur les proportions différentes pour chaque méthode sont représentés par le tableau III.12 pour $\alpha = 0.01$ et le tableau III.13 pour $\alpha = 0.05$.

Tout d'abord, notons que comme avec la stratégie 1°, il apparaît que les méthodes reposant sur des sous-espaces de représentation (OSVM, KPCA et KECA) préservent mieux la robustesse des décisions dans les zones frontières ($\hat{\alpha}_F$, $\hat{\beta}_F$) par rapport à celle utilisant l'espace initial complet (OSVM_$\mathcal{X}^m$), mais entraînent en contrepartie une

dégradation de la performance pour les données non-perturbés (c'est-à-dire dans les zones cœurs ($\hat{\alpha}_C$, $\hat{\beta}_C$).

Par ailleurs, nous pouvons observer que la performance des méthodes opérant sur des sous-espaces dépend aussi de la proportion (L_3, L_5, L_7). Dans les zones frontières, les résultats montrent que $\hat{\alpha}_F$ diminue tandis que $\hat{\beta}_F$ augmente quand la proportion de sous-espaces de petite dimenstion (L_3) augmente. En revanche, l'augmentation de la proportion de sous-espaces de grande dimension (L_7) conduit à des résultats inverses. En d'autres termes, la répartition des dimensions des sous-espaces permet de régler le compromis entre les taux d'erreur $\hat{\alpha}_F$ et $\hat{\beta}_F$ pour les données situées dans les zones frontières.

III.6 Conclusion

Dans ce chapitre, nous avons présenté et testé une méthode robuste de classification par la définition d'un ensemble de sous-espaces de représentation homogènes visant à préserver ou améliorer la performance d'un système décisionnel en présence de perturbations dans l'espace de représentation initial. Ces sous-espaces de représentation sont générés en sélectionnant aléatoirement des attributs à partir de l'espace de représentation initial, selon le principe de la méthode RSM. Nous avons alors étudié et testé trois algorithmes d'apprentissage différents (OSVM, KPCA et KECA) pour construire les classifieurs élémentaires opérant dans les sous-espaces de représentation générés. Pour les deux algorithmes KPCA et KECA, la classification a été opérée sur la base de l'erreur de reconstruction. Le problème de la segmentation d'images texturées nous a fourni un support tout à fait adéquat pour illustrer notre démarche et tester les performances du classifieur global en environnement perturbé.

Notons toutefois que notre approche et le processus d'évaluation n'avaient pas pour but de chercher à minimiser les taux d'erreur de classification globale dans la segmentation d'images, mais plutôt d'étudier et d'illustrer l'intérêt de travailler dans des sous-espaces de représentation pour améliorer les décisions dans différentes situations d'environnement des données à classer (normal et perturbé). Les résultats nous ont montré qu'en environnement normal (les zones cœur d'image), la méthode OSVM_$\mathcal{X}^m$ utilisant tous les attributs de l'espaces de représentation initial donne la meilleure performance. D'autre part, dans l'environnement perturbé (les zones frontières d'image), les méthodes opérant sur des sous-espaces de représentation (OSVM, KPCA et KECA) s'avèrent plus performantes, en particulier KPCA et KECA pour lesquelles les résultats sont assez proches. Cette étude nous a donc permis de montrer que la sélection de sous-espaces de représentation associée à des méthodes d'apprentissage performantes permet de construire un ensemble de classifieurs sur la base de données « normales » ou non perturbées et d'en déduire une règle de décision globale offrant de bonnes performances en environnement perturbé ou non-stationnaire. Il semble par ailleurs que la prise en compte de sous-espaces de dimensions différentes offre la possibilité de régler le compromis entre les probabilités d'erreurs α_F et β_F aux frontières, mais il demeure néanmoins que l'optimisation de ce type de classifieur est d'une grande complexité.

Fig.III.4(a)	(L_3,L_5,L_7)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$		0.49	**0.00**	72.36	**0.58**
OSVM	(20,40,60)	**0.46**	**0.06**	37.62	**2.44**
	(40,40,40)	**0.46**	0.17	33.93	2.97
	(60,40,20)	0.50	0.38	**31.77**	3.86
KPCA	(20,40,60)	**0.69**	**0.13**	27.45	**4.36**
	(40,40,40)	0.73	0.29	25.43	5.13
	(60,40,20)	0.73	0.62	**22.41**	6.49
KECA	(20,40,60)	**0.70**	**0.17**	26.31	**4.74**
	(40,40,40)	**0.70**	0.39	23.92	5.83
	(60,40,20)	0.71	0.95	**20.03**	7.98

Fig.III.4(b)	(L_3,L_5,L_7)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$		0.67	**0.00**	67.24	**0.56**
OSVM	(20,40,60)	0.65	**0.06**	49.17	**1.22**
	(40,40,40)	**0.53**	0.19	44.53	1.62
	(60,40,20)	0.67	0.53	**42.27**	2.46
KPCA	(20,40,60)	0.64	**0.40**	33.62	**2.93**
	(40,40,40)	0.57	0.87	30.70	3.63
	(60,40,20)	**0.44**	1.78	**28.53**	5.67
KECA	(20,40,60)	0.66	**0.39**	33.38	**2.94**
	(40,40,40)	0.59	0.87	30.89	3.61
	(60,40,20)	**0.54**	1.87	**28.09**	5.75

Fig.III.4(c)	(L_3,L_5,L_7)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$		0.97	**0.11**	58.87	**1.79**
OSVM	(20,40,60)	**0.77**	**2.12**	33.25	**7.32**
	(40,40,40)	0.80	3.28	29.99	9.01
	(60,40,20)	0.85	5.20	**28.11**	11.71
KPCA	(20,40,60)	0.65	**4.16**	23.27	**11.13**
	(40,40,40)	**0.61**	5.57	22.24	12.38
	(60,40,20)	0.62	8.78	**21.51**	15.89
KECA	(20,40,60)	0.65	**4.40**	22.69	**11.24**
	(40,40,40)	**0.58**	6.60	20.82	13.86
	(60,40,20)	0.60	10.78	**19.71**	18.19

Tableau III.12 – Taux d'erreurs (%) correspondant à la stratégie 2° avec $\alpha = 0.01$

Fig.III.4(a)	(L_3,L_5,L_7)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$		4.03	**0.00**	82.12	**0.30**
OSVM	(20,40,60)	**3.84**	**0.004**	60.81	**0.47**
	(40,40,40)	3.92	0.008	56.93	0.56
	(60,40,20)	4.22	0.01	**54.24**	0.67
KPCA	(20,40,60)	4.23	**0.01**	35.85	**2.62**
	(40,40,40)	4.04	0.07	32.56	2.98
	(60,40,20)	**3.95**	0.14	**31.95**	3.18
KECA	(20,40,60)	4.17	**0.004**	37.08	**2.41**
	(40,40,40)	4.11	0.05	33.31	2.65
	(60,40,20)	**3.86**	0.19	**30.34**	3.69
Fig.III.4(b)	(L_3,L_5,L_7)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$		4.60	**0.00**	78.21	**0.27**
OSVM	(20,40,60)	**3.93**	**0.008**	60.61	**0.48**
	(40,40,40)	4.22	0.02	58.88	0.55
	(60,40,20)	4.17	0.05	**57.33**	0.77
KPCA	(20,40,60)	4.19	**0.03**	53.08	**0.81**
	(40,40,40)	3.65	0.05	49.49	1.07
	(60,40,20)	**3.35**	0.20	**46.00**	1.73
KECA	(20,40,60)	4.28	**0.03**	52.59	**0.82**
	(40,40,40)	3.68	0.07	48.64	1.10
	(60,40,20)	**3.52**	0.27	**45.19**	1.86
Fig.III.4(c)	(L_3,L_5,L_7)	$\hat{\alpha}_C$	$\hat{\beta}_C$	$\hat{\alpha}_F$	$\hat{\beta}_F$
OSVM_$\mathcal{X}^m$		5.62	**0.04**	72.07	**0.49**
OSVM	(20,40,60)	4.85	**0.19**	55.72	**1.65**
	(40,40,40)	**4.77**	0.28	52.48	2.06
	(60,40,20)	4.83	0.57	**49.85**	3.04
KPCA	(20,40,60)	4.61	**0.41**	46.54	**2.81**
	(40,40,40)	**4.48**	0.86	41.31	3.99
	(60,40,20)	4.62	1.77	**37.87**	5.64
KECA	(20,40,60)	4.62	**0.44**	45.32	**2.92**
	(40,40,40)	**4.56**	0.94	40.30	4.23
	(60,40,20)	4.64	2.04	**36.35**	6.15

Tableau III.13 – Taux d'erreurs (%) correspondant à la stratégie 2° avec $\alpha = 0.05$

Chapitre IV

Modélisation des variables décisionnelles dans le cadre de la fusion de classifieurs en environnement non stationnaire

IV.1 Introduction

Dans le chapitre précédent, nous avons simulé un problème de surveillance d'un système dans un environnement pouvant présenter des non-stationarités au moyen de la segmentaiton d'images texturées. Toutefois, dans ce contexte, le problème de la surveillance d'un système est posé de façon particulière et propre au support des données traitées. En effet, un pixel représente à la fois un individu à classer (réalisation du vecteur d'attributs) et un capteur délivrant une donnée participant au vecteur d'attributs qui le représente ou bien à un vecteur d'attributs attaché à l'un de ses voisins (qui joue alors le rôle de l'individu à classer). Bien que traitant d'un cas très particulier de classification, l'exemple de la segmentation nous a permis d'illustrer et de mettre en évidence la pertinence et les performances des méthodes d'ensemble pour la prise de décision en environnement non-stationnaire ou perturbé.

Toutefois, afin d'étudier plus en détail les performances d'un tel système décisionnel, nous allons maintenant nous replacer dans un cadre plus général de la surveillance et du diagnostic d'un système. Pour cela, nous allons considérer chaque observation à classer comme la réalisation d'un vecteur d'attributs dans un espace de représentation donné. Les attributs sont extraits des mesures issues de capteurs et sont supposées aptes à caractériser les états du système surveillé. Nous resterons également dans un problème de classification binaire en considérant que le système ne peut présenter que deux états : « fonctionnement normal » et « fonctionnement anormal ». Par ailleurs, nous considérons que l'état des capteurs est susceptible d'évoluer au cours du temps ou selon le contexte (panne, bruit...) et les performances d'une règle de décision reposant sur leurs sorties peuvent donc également évoluer selon le contexte.

Bien que les résultats expérimentaux aient montré (Cf. chapitre III) qu'en fusionnant des règles de décision construites dans des sous-espaces de représentation, on obtenait une règle globale robuste en environnement non-stationnaire (en particulier dans les zones frontières pour la segmentation d'images texturées), on peut imaginer que lorsque la grande majorité des capteurs sont en panne à un instant donné, le

nombre de sous-espaces de représentation homogènes risque de devenir très faible, entraînant ainsi une perte de performances de la règle de décision finale obtenue par vote majoritaire.

Pour cette raison, nous proposons dans ce chapitre d'envisager des règles de combinaison plus complexes. En fait, la conception des règles de combinaison repose sur divers critères d'optimisation dont les deux plus connus sont le critère de *Bayes* et le critère de *Neyman-Pearson* dans le cadre des tests statistiques de deux hypothèses (Thomopoulos et al. - 1989; Lehmann - 1986; Varshney - 1996; Viswanathan et Varshney - 1997; Varshney - 1997; Thomopoulos et al. - 1987). Ils prennent en compte les propriétés de chaque classifieur individuel que sont les probabilités d'erreur de première (Cf. équation II.31) et de seconde espèce (Cf. équation II.32). En exprimant la performance globale de la règle de décision en fonction de ces deux probabilités d'erreur, on établit un modèle permettant de caractériser la règle de fusion. De cette manière, on peut avoir une maîtrise plus fine des performances globales de la règle de décision et on peut également les évaluer facilement si le nombre de classifieurs diminue (perte de capteurs par exemple).

Il faut toutefois noter que ce modèle ne peut être établi aisément que si les décisions locales prises par les classifieurs élémentaires sont indépendantes. Ce n'est bien entendu pas le cas dans notre problème, car chaque classifieur est défini sur un sous-espaces de mesures issues d'un sous-ensemble de capteurs sélectionnés aléatoirement parmi tous les capteurs. Les sous-espaces générés ne sont donc pas nécessairement disjoints, de sorte que les classifieurs reposant sur ces sous-espaces ne sont généralement pas indépendants. Dans le cas de la classification binaire que nous avons choisi de traiter, l'ensemble des décisions à combiner pour obtenir la décision globale peut être considéré comme un vecteur aléatoire dont les composantes sont des variables de Bernoulli corrélées. Le modèle permettant de caractériser la règle de fusion repose alors sur la loi conjointe de ce vecteur décisionnel qu'il n'est pas simple d'établir lorsque les composantes sont corrélées. Nous nous sommes néanmoins attaqués à ce problème et pour y parvenir, nous nous sommes appuyés sur le modèle de Bahadur-Lazarsfeld (Bahadur - 1961) et le modèle de maximum d'entropie proposé par Van Der Geest (2005).

Ce chapitre sera organisé comme suit : dans un premier temps, nous présenterons quelques règles de fusion de décisions reposant sur les critères de *Bayes* et de *Neyman-Pearson*, dans le cas où les classifieurs sont indépendants. S'inspirant de ces deux critères, une nouvelle approche reposant sur un test du rapport de vraisemblance sera proposée et évaluée. Puis, en considérant la situation où les classifieurs élémentaires ne sont plus indépendants, nous rappellerons les modèles et l'approche permettant d'approximer la loi conjointe pour les variables décisionnelles corrélées et nous présenterons, avant de conclure, quelques simulations et résultats obtenus par la mise en œuvre de ces modèles.

IV.2 Règles de fusion de décisions dans le cas de classifieurs indépendants

Le problème de la fusion de décisions est une étape cruciale pour la construction d'une règle de décision globale. Il couvre un domaine important de recherche dans le cadre de la surveillance des systèmes. Dans un système traditionnel surveillé par un ensemble de capteurs, chaque capteur est généralement considéré comme un classifieur, qui n'est pas seulement chargé de recueillir l'information mais aussi de prendre une décision locale au sujet de l'état du système ou de la présence/absence d'un objet cible. Les données brutes recueillies ou les décisions locales sont ensuite transmises à un processeur central, souvent appelé centre de fusion, qui les combine pour produire une décision finale. C'est sur cette étape de fusion des décisions qu'ont porté les travaux présentés ici.

IV.2.1 Formulation du problème et notations

Soit donc un espace de repésentation $\mathcal{X}^m$ et une règle de décision binaire entre deux hypothèses H_0 et H_1 représentant respectivement l'état normal et anormal du système. Cette règle repose sur L classifieurs individuels h_ℓ dont les décisions binaires sont représentées par les réalisations y_ℓ d'une variable aléatoire décisionnelle Y_ℓ selon le schéma

$$\boldsymbol{x} \in \mathcal{X}^m \xmapsto{h_\ell} h_\ell(\boldsymbol{x}) = y_\ell \in \mathcal{Y} = \{0,1\} \quad \text{pour} \quad \ell = 1,,\ldots,L \tag{IV.1}$$

avec la convention suivante :

$$\text{si} \quad y_\ell = \begin{cases} 0 & \text{le classifieur } h_\ell \text{ décide (ou vote) en faveur de } H_0 \\ 1 & \text{le classifieur } h_\ell \text{ décide en faveur de } H_1 \end{cases} \tag{IV.2}$$

Rappelons que L représente également le nombre de sous-espaces de représentation $\mathcal{X}_\ell^d$ de dimension d et dans chacun desquels opère un classifieur.

Les décisions locales représentées par les variables aléatoires décisionnelles Y_ℓ sont rassemblées dans un vecteur décisionnel $\boldsymbol{Y} = [Y_1\ ,\ Y_2\ ,\ \ldots\ ,\ Y_L]^t$. La fusion des décisions consiste ensuite en une fonction logique à plusieurs entrées binaires et donnant une sortie binaire. Pour un système avec L classifieurs binaires, on aura ainsi 2^L réalisations possibles du vecteur décisionnel $\boldsymbol{Y}$. Les fonctions logiques du type « k-out-of-L » sont souvent mises en œuvre dans la pratique, elles consistent à décider en faveur de H_0 si k classifieurs au moins votent pour l'hypothèse H_0. Les règles les plus largement utilisées, telles que « ET », « OU » et le vote majoritaire (ici $k = L/2$), sont des cas particuliers de « k-out-of-L ».

Finalement, la fusion des décisions peut être vue comme une fonction binaire h associant au vecteur décisionnel $\boldsymbol{Y}$ une variable aléatoire Y représentant la décision finale selon la convention suivante :

$$\text{si} \quad Y = h(\,\boldsymbol{Y}\,) = \begin{cases} 0 & \text{la décision finale est en faveur de } H_0 \\ 1 & \text{la décision finale est en faveur de } H_1 \end{cases} \tag{IV.3}$$

Dans de nombreuses applications, on peut arbitrairement choisir une règle de fusion parmi l'ensemble des fonctions logiques existantes. Un choix arbitraire est généralement satisfaisant, mais n'offre pas la garantie d'être optimal. Nous allons donc tout d'abord rappeler les critères de *Bayes* et de *Neyman-Pearson*, qui permettent de déterminer des règles de fusion optimales.

IV.2.2 Règles de fusion et critère de Bayes

Le critère d'optimisation de Bayes consiste à minimiser un coût moyen d'erreur sur les deux hypothèses (ce que l'on nomme aussi le risque bayésien). La mise en œuvre de ce critère nécessite de fixer des coûts pour les bonnes et mauvaises décisions et requiert la connaissance des probabilités a priori π_0 et π_1 des deux hypothèses H_0 (classe de fonctionnement normal) et H_1 (classe de fonctionnement anormal).

Chaque classifieur élémentaire h_ℓ est en outre caractérisé par ses probabilités d'erreur de première espèce (α_ℓ) et de seconde espèce (β_ℓ) que l'on peut redéfinir aussi comme suit,

$$\begin{cases} \text{probabilité de fausse alarme :} & \alpha_\ell = P\left[Y_\ell = 1 \mid H_0\right] \\ \text{probabilité de non détection :} & \beta_\ell = P\left[Y_\ell = 0 \mid H_1\right] \end{cases} \tag{IV.4}$$

Remarquons que les variables décisionnelles Y_ℓ sont des variables obéissant à des lois de Bernoulli dont les paramètres sont donnés en (IV.4) et que l'on peut écrire sous les formes :

$$\begin{cases} P\left[Y_\ell = y_\ell \mid H_0\right] &=& \alpha_\ell^{y_\ell}(1-\alpha_\ell)^{1-y_\ell} &=& p_0(y_\ell) \\ P\left[Y_\ell = y_\ell \mid H_1\right] &=& \beta_\ell^{1-y_\ell}(1-\beta_\ell)^{y_\ell} &=& p_1(y_\ell) \end{cases} \tag{IV.5}$$

On considère de plus des coûts C_{ij} $(i,j=0,1)$ associés à la décision finale. C_{ij} est le coût associé à la décision H_i (i.e. $y = h(\boldsymbol{y}) = i$, Cf. (IV.3)) lorsque l'hypothèse H_j est vraie.

Pour un exemple $\boldsymbol{X}$ à classer, conformément à la définition des classifieurs individuels que nous avons donnée en (IV.1), on obtient L décisions $Y_\ell = h_\ell(\boldsymbol{X})$ modélisées conjointement par le vecteur décisionnel $\boldsymbol{Y} = [Y_1\ ,\ Y_2\ ,\ \ldots\ ,\ Y_L]^t$.

La règle de décision reposant sur la minimisation du risque de Bayes s'exprime alors comme le test du rapport de vraisemblance $r(\boldsymbol{Y})$ suivant (Varshney - 1996; Chair et Varshney - 1986),

$$r(\boldsymbol{Y}) = \frac{P\left[Y_1,\ Y_2,\ \ldots, Y_L \mid H_1\right]}{P\left[Y_1,\ Y_2,\ \ldots, Y_L \mid H_0\right]} = \frac{L_1(\boldsymbol{Y})}{L_0(\boldsymbol{Y})} \underset{H_0}{\overset{H_1}{\gtrless}} \frac{\pi_0(C_{10}-C_{00})}{\pi_1(C_{01}-C_{11})} = \eta \tag{IV.6}$$

la fonction de fusion pour la décision finale définie en (IV.3) peut ici s'écrire :

$$Y = h(\boldsymbol{Y}) = \begin{cases} 0 & \text{si} \quad r(\boldsymbol{Y}) < \eta \\ 1 & \text{si} \quad r(\boldsymbol{Y}) > \eta \end{cases} \tag{IV.7}$$

Sous l'hypothèse d'indépendance des classifieurs, les variables décisionnelles Y_ℓ sont indépendantes et avec les notations définies en (IV.5) on a alors :

$$L_i(\boldsymbol{Y}) = P\left[Y_1,\ Y_2,\ \ldots, Y_L \mid H_i\right] = \prod_{\ell=1}^{L} p_i(Y_\ell) \quad (i = 0 \text{ ou } 1) \tag{IV.8}$$

En utilisant l'expression (IV.8) dans le terme de gauche de l'équation (IV.6), la règle de décision s'écrit :

$$r(\boldsymbol{Y}) = \prod_{\ell=1}^{L} \left(\frac{1-\beta_\ell}{\alpha_\ell}\right)^{Y_\ell} \left(\frac{\beta_\ell}{1-\alpha_\ell}\right)^{(1-Y_\ell)} \underset{H_0}{\overset{H_1}{\gtrless}} \eta \tag{IV.9}$$

En passant au logarithme, on obtient finalement :

$$\log\left(r(\boldsymbol{Y})\right) = \sum_{\ell=1}^{L} \left[Y_\ell \log\frac{1-\beta_\ell}{\alpha_\ell} + (1-Y_\ell)\log\frac{\beta_\ell}{1-\alpha_\ell} \right] \underset{H_0}{\overset{H_1}{\gtrless}} \log(\eta) \tag{IV.10}$$

$$\Longleftrightarrow \quad \sum_{\ell=1}^{L} \log\left(\frac{(1-\beta_\ell)(1-\alpha_\ell)}{\alpha_\ell\beta_\ell}\right) Y_\ell \underset{H_0}{\overset{H_1}{\gtrless}} \log(\eta) - \sum_{\ell=1}^{L}\log\left(\frac{\beta_\ell}{1-\alpha_\ell}\right) \tag{IV.11}$$

$$\Longleftrightarrow \quad \sum_{\ell=1}^{L} \theta_\ell Y_\ell \underset{H_0}{\overset{H_1}{\gtrless}} \eta' \tag{IV.12}$$

Les probabilités de fausse alarme et de non détection de la règle globale résultant de la fusion peuvent alors s'écrire ainsi :

$$\begin{cases} \alpha_G = P\left[Y = 1 \mid H_0\right] = P\left[\sum_{\ell=1}^{L} \theta_\ell Y_\ell > \eta' \mid H_0\right] & \text{(fausse alarme)} \\ \beta_G = P\left[Y = 0 \mid H_1\right] = P\left[\sum_{\ell=1}^{L} \theta_\ell Y_\ell < \eta' \mid H_1\right] & \text{(non détection)} \end{cases} \tag{IV.13}$$

On observe donc que la règle de décision finale définie en (IV.12) consiste à comparer à un seuil, une combinaison linéaire des variables décisionnelles associées aux classifieurs individuels. Les coefficients θ_ℓ de cette combinaison sont donnés par les probabilités caractéristiques (α_ℓ et β_ℓ) des classifieurs individuels. Le seuil η' est quant à lui complètement déterminé par les coûts, les probabilités a priori des classes π_0 et π_1 ainsi que par les probabilités caractéristiques (α_ℓ et β_ℓ).

Une fois fixés les coûts (très souvent choisis ainsi : $C_{00} = C_{11} = 0$ et $C_{10} = C_{01} = 1$), il apparaît que la règle finale ne dépend que des caractéristiques des classifieurs individuels et des probabilités a priori des classes ou hypothèses.

On peut noter également que les coefficients θ_ℓ prennent des valeurs importantes pour les classifieurs individuels les plus performants qui pèsent donc plus dans la décision finale. Toutefois le lien entre les performances globales (α_G et β_G) et celles des classifieurs individuels ne sont pas simples à modéliser dans le cadre de cette règle de décision. Varshney (1996) a étudié quelques extensions de ce problème.

IV.2.3 Règles de fusion et critère de Neyman-Pearson

Comme nous venons de le voir dans la section précédente, la règle de Bayes repose sur un seuil η (Cf. relation (IV.6)) qui suppose la connaissance des probabilités a priori π_0 et $\pi_1 = 1 - \pi_0$ des hypothèses. Si on connaît π_0 ou si on dispose d'une estimation $\hat{\pi}_0$, le seuil η est alors complètement déterminé une fois que l'on a fixé les coûts C_{ij}. De ce fait les performances globales α_G et β_G de la règle de Bayes sont, elles aussi, entièrement déterminées et ne peuvent être ajustées qu'en agissant sur les caractéristiques α_ℓ et β_ℓ des classifieurs individuels.

Le critère de Neyman-Pearson propose donc une alternative qui consiste à choisir la règle de décision finale qui minimise la probabilité globale de non détection β_G pour une probabilité globale de fausse alarme α_G fixée a priori.

Avec les notations définies précédemment, cette règle repose, elle aussi, sur le rapport de vraisemblance $r(\boldsymbol{Y})$ défini à l'équation (IV.6) et s'écrit ainsi :

$$r(\boldsymbol{Y}) \underset{H_0}{\overset{H_1}{\gtrless}} \gamma \tag{IV.14}$$

Ou encore, en s'inspirant des calculs de la section précédente, la règle de décision finale donnée en (IV.12) peut s'écrire ici :

$$\sum_{\ell=1}^{L} \theta_\ell Y_\ell \underset{H_0}{\overset{H_1}{\gtrless}} \gamma' \tag{IV.15}$$

$$\text{avec} \quad \gamma' = \log(\gamma) - \sum_{\ell=1}^{L} \log\left(\frac{\beta_\ell}{1-\alpha_\ell}\right) \tag{IV.16}$$

Le seuil γ est ici variable et déterminé en fonction de la probabilité de fausse alarme α_G souhaitée qui s'écrit ici :

$$\alpha_G = P\left[Y = 1 \mid H_0\right] = P\left[\sum_{\ell=1}^{L} \theta_\ell Y_\ell > \gamma' \mid H_0\right] \tag{IV.17}$$

En résumé, à une probabilité de fausse alarme α_G^* prédéfinie, on aura un seuil $\gamma'_{\alpha_G^*}$ et la règle (IV.15) s'écrit finalement :

$$\sum_{\ell=1}^{L} \theta_\ell Y_\ell \underset{H_0}{\overset{H_1}{\gtrless}} \gamma'_{\alpha_G^*} \tag{IV.18}$$

faisant ainsi apparaître que les performances de la règle globale, bien qu'étant liées aux performances des classifieurs individuels (par l'intermédiaire des coefficients θ_ℓ), peuvent être fixées a priori indépendamment de celles-ci.

Si l'on considère par exemple le cas où les classifieurs individuels ont tous les mêmes caractéristiques

$$\alpha_\ell = \alpha_{indiv} \quad \text{et} \quad \beta_\ell = \beta_{indiv} \; \forall \; \ell = 1, \ldots, L$$

la règle donnée par l'expression(IV.18) devient simplement :

$$\sum_{\ell=1}^{L} Y_\ell \underset{H_0}{\overset{H_1}{\gtrless}} \mathcal{B}(L, \alpha_{indiv}, \alpha_G^*) \tag{IV.19}$$

où
$\mathcal{B}(L, \alpha_{indiv}, \alpha_G^*) = \gamma'_{\alpha_G^*}/\theta$
$\mathcal{B}(L, \alpha_{indiv}, \alpha_G^*)$ est le quantile de niveau α_G^* d'une loi binomiale $\mathcal{B}(L, \alpha_{indiv})$
$\theta = \log\left(\frac{(1-\beta_{indiv})(1-\alpha_{indiv})}{\alpha_{indiv}\beta_{indiv}}\right)$ Cf. (IV.11)

Notons que dans les conditions de la règle donnée en (IV.19), si on fixe le seuil à $L/2$, la règle devient la règle du vote majoritaire qui s'écrit :

$$\sum_{\ell=1}^{L} Y_\ell \underset{H_0}{\overset{H_1}{\gtrless}} L/2 \tag{IV.20}$$

Quelques pistes pour la mise en œuvre de ce critère ont été proposées par Thomopoulos et al. (1987). Dans notre approche, nous nous sommes inspirés de ces règles de décision en utilisant le rapport de vraisemblance pour modéliser et optimiser une règle de fusion, là encore dans le cas où les classifieurs sont indépendants.

IV.2.4 Comparaison de règles de fusion avec capteurs en panne

IV.2.4.1 Présentation du problème

Nous nous intéressons ici aux performances d'une règle de décision globale résultant de la fusion des décisions de plusieurs classifieurs lorsque les caractéristiques des classifieurs locaux peuvent être altérées par des pannes de capteurs. Pour cela nous allons considérer que l'on dispose d'un ensemble initial de capteurs dont un certain nombre est en panne à un moment donné. Nous supposerons de plus qu'un capteur fournit un seul attribut. Les sous-espaces de représentation que nous allons considérer correspondront donc à une sélection de capteurs.

Soit donc m le nombre initial de capteurs surveillant le système et d la dimension des sous-espaces de représentation. Un sous-espace $\mathcal{X}_\ell^d \subset \mathcal{X}^m$ sera donc engendré par la sélection de d capteurs parmi les m et un classifieur sera élaboré dans chaque sous-espace.

Nous avons évoqué à la fin de la section précédente le cas particulier où les L classifieurs ont des caractéristiques α_ℓ et β_ℓ identiques. Cette hypothèse de travail n'est plus valide dès lors qu'un ou plusieurs capteurs sont altérés ou en panne par exemple. En effet, les classifieurs individuels opérant dans les sous-espaces ayant sélectionné au moins un de ces capteurs, vont voir leurs caractéristiques changer. C'est cet aspect qu'il nous semble important d'intégrer dans la modélisation d'une règle de fusion afin de disposer d'un moyen de contrôle et d'ajustement des performances de la règle globale. Nous proposons donc d'étudier un critère de fusion prenant en compte cet aspect.

IV.2.4.2 Principe général

Si on dispose de m capteurs notés $C_i\,(i = 1, \ldots, m)$, chacun reçoit une donnée à classer $\mathcal{D}$ puis délivre une sortie (i.e. un attribut) représentée par une variable aléatoire $X_i\,(i = 1, \ldots, m)$ dont l'état dépend de $\mathcal{D}$. L'espace initial de représentation est alors de dimension m et on va considérer L sous-espaces d'attributs de dimension d. Un classifieur binaire sera élaboré dans chaque sous-espace d'attributs et prendra une décision locale représentée par une variable aléatoire de Bernoulli Y_ℓ. Le processus complet est résumé dans le tableau IV.1 avec les notations déjà introduites à la section IV.2.1 page 69 et dans le cas où on considère les $N = C_m^d$ sous-espaces possibles.

donnée à classer	sorties capteurs	Sous-ensembles d'attributs (capteurs)	sorties classifieurs (variables décisionnelles)
$\mathcal{D}$	$C_1(\mathcal{D}) = X_1$	$SE_1 = (X_1, X_2)$	$Y_1 = h_1(X_1, X_2)$
(signal,	$\vdots$	$\vdots$	$\vdots$
image)	$C_m(\mathcal{D}) = X_m$	$SE_N = (X_{m-1}, X_m)$	$Y_N = h_N(X_{m-1}, X_m)$
	vecteur d'attributs $\boldsymbol{X} = [\,X_1, \cdots, X_m\,]^t$		vecteur décisionnel $\boldsymbol{Y} = [\,Y_1, \cdots, Y_N\,]^t$

Tableau IV.1 – Processus décisionnel avec m capteurs et N sous-espaces d'attributs de dimension $d = 2$

Dans toute la suite de cette étude nous admettrons par ailleurs que les m capteurs sont indépendants. Dans ce cas, pour pouvoir faire l'hypothèse que les L classifieurs pris en compte dans la fusion sont indépendants, il nous faut alors considérer des sous-espaces de représentation disjoints et pour cela nous avons procédé à des tirages aléatoires sans remise de d capteurs parmi m. Il en résulte que $L < N$ et que le vecteur décisionnel $\boldsymbol{Y} = [Y_1\ ,\ Y_2\ ,\ \ldots\ ,\ Y_L]^t$ est constitué de variables de Bernoulli indépendantes.

Afin d'intégrer la prise en compte des capteurs en panne dans la formulation du problème, nous introduisons les quelques notations complémentaires suivantes :

- $Ncp^{(m)}$ variable représentant le nombre total de capteurs en panne à un instant donné $(0 \leq Ncp^{(m)} \leq m)$
- $Ncp_\ell^{(d)}$ variable représentant le nombre de capteurs en panne dans le sous-espace $\mathcal{X}_\ell^d$ $(0 \leq Ncp_\ell^{(d)} \leq d)$
- α_{ℓ,k_ℓ} : probabilité de fausse alarme du classifieur ℓ, définie par

$$\alpha_{\ell,k_\ell} = P\left[Y_\ell = 1 \mid H_0 \text{ et } Ncp_\ell^{(d)} = k_\ell\right], \quad k_\ell = 0, 1, \ldots, d \tag{IV.21}$$

- β_{ℓ,k_ℓ} : probabilité de non détection du classifieur ℓ, définie par

$$\beta_{\ell,k_\ell} = P\left[Y_\ell = 0 \mid H_1 \text{ et } Ncp_\ell^{(d)} = k_\ell\right], \quad k_\ell = 0, 1, \ldots, d \tag{IV.22}$$

- Il existe par ailleurs 2^m scénarios possibles de panne sur l'ensemble des m capteurs, un scénario sans aucun capteur en panne et $2^m - 1$ scénarios de panne d'au moins un capteur. Nous noterons S_j $(j = 1, \ldots, 2^m)$ les 2^m scénarios possibles.

Sous l'hypothèse d'indépendance des classifieurs et conjointement avec un scénario de panne S_j, la vraisemblance donnée en (IV.8) devient :

$$\begin{aligned} L_0^{(j)}(\boldsymbol{Y}, S_j) &= P[\boldsymbol{Y} \mid S_j, H_0]\, P[S_j \mid H_0] \\ &= \prod_{\ell=1}^{L} (\alpha_{\ell,k_\ell})^{Y_\ell} (1 - \alpha_{\ell,k_\ell})^{1-Y_\ell} \cdot P[S_j] \qquad \text{(IV.23)} \\ \text{et} \quad L_1^{(j)}(\boldsymbol{Y}, S_j) &= \prod_{\ell=1}^{L} (\beta_{\ell,k_\ell})^{1-Y_\ell} (1 - \beta_{\ell,k_\ell})^{Y_\ell} \cdot P[S_j] \qquad \text{(IV.24)} \end{aligned}$$

$P[S_j]$ représente la probabilité du $j^{\text{ème}}$ scénario de panne S_j. Nous admettons que les scénarios de panne ne dépendent que de l'environnement et pas de l'état du système sous surveillance. De ce fait, dans les formules (IV.23) et (IV.24), on a

$$P[S_j \mid H_0] = P[S_j \mid H_1] = P[S_j]$$

Si l'on suppose de plus que tous les capteurs ont la même probabilité de panne p_{cp} et que le $j^{\text{ème}}$ scénario de panne S_j correspond à $Ncp^{(m)} = ncp_j$ capteurs en panne, on a alors :

$$P[S_j] = P\left[Ncp^{(m)} = ncp_j\right] = (p_{cp})^{ncp_j} \cdot (1 - p_{cp})^{m - ncp_j} \qquad \text{(IV.25)}$$

Illustration des formules (IV.23) et (IV.24)

Afin de clarifier la mise en œuvre des formules (IV.23) et (IV.24), considérons un exemple (Cf. tableaux IV.2 et IV.3) pour lequel on dispose de $m = 4$ capteurs et où l'on choisit des sous-espaces de dimension $d = 2$. On peut définir alors $N = C_m^d = 6$ sous-espaces au total et donc 6 variables décisionnelles Y_ℓ selon le schéma présenté dans le tableau IV.2 où on peut observer que les couples de variables (Y_1, Y_6), (Y_2, Y_5) et (Y_3, Y_4) sont constitués de variables indépendantes car les sous-espaces correspondants sont disjoints et les sorties X_i des capteurs sont supposées indépendantes. Il existe par ailleurs $2^m = 16$ scénarios possibles de panne dont le tableau IV.3 donne un aperçu.

Supposons maintenant que l'on sélectionne les $L = 2$ sous-espaces $\mathcal{X}_1^d$ et $\mathcal{X}_6^d$ associés aux variables Y_1 et Y_6, la formule (IV.23) donnera alors par exemple :

$$\begin{aligned} L_0^{(1)}(\boldsymbol{Y}, S_1) &= (\alpha_{1,0})^{Y_1} (1 - \alpha_{1,0})^{1-Y_1} (\alpha_{6,0})^{Y_6} (1 - \alpha_{6,0})^{1-Y_6} (1 - p_{cp})^m \\ L_0^{(2)}(\boldsymbol{Y}, S_2) &= (\alpha_{1,1})^{Y_1} (1 - \alpha_{1,1})^{1-Y_1} (\alpha_{6,0})^{Y_6} (1 - \alpha_{6,0})^{1-Y_6} p_{cp}(1 - p_{cp})^{m-1} \\ &\vdots \\ L_0^{(6)}(\boldsymbol{Y}, S_6) &= (\alpha_{1,2})^{Y_1} (1 - \alpha_{1,2})^{1-Y_1} (\alpha_{6,0})^{Y_6} (1 - \alpha_{6,0})^{1-Y_6} (p_{cp})^2(1 - p_{cp})^{m-2} \\ &\vdots \end{aligned}$$

$\mathcal{X}_\ell^d$	(C_1, C_2)	(C_1, C_3)	(C_1, C_4)	(C_2, C_3)	(C_2, C_4)	(C_3, C_4)
Y_ℓ	$Y_1 =$ $h_1(X_1, X_2)$	$Y_2 =$ $h_2(X_1, X_3)$	$Y_3 =$ $h_3(X_1, X_4)$	$Y_4 =$ $h_4(X_2, X_3)$	$Y_5 =$ $h_5(X_2, X_4)$	$Y_6 =$ $h_6(X_3, X_4)$

Tableau IV.2 – Sous-espaces de représentation et variables décisionnelles associées

	Scénarios S_j							
Capteur	S_1	S_2	S_3	S_4	S_5	S_6	...	S_{16}
C_1	OK	panne	OK	OK	OK	panne	...	panne
C_2	OK	OK	panne	OK	OK	panne	...	panne
C_3	OK	OK	OK	panne	OK	OK	...	panne
C_4	OK	OK	OK	OK	panne	OK	...	panne
ncp_j	0	1	1	1	1	2	...	4

Tableau IV.3 – Exemples de scénarios de pannes de capteurs

Les valeurs α_{ℓ,k_ℓ} et β_{ℓ,k_ℓ} sont fixées a priori (ici pour $\ell = 1$ et 6)

Règle de décision finale

Nous proposons pour la décision finale un critère reposant sur la comparaison des maximums de la vraisemblance sous H_0 et H_1, calculés sur l'ensemble des scénarios de panne. Cela conduit à la règle (IV.26) suivante :

$$\frac{\max\limits_j L_0^{(j)}(\boldsymbol{Y}, S_j)}{\max\limits_j L_1^{(j)}(\boldsymbol{Y}, S_j)} \underset{H_1}{\overset{H_0}{\gtrless}} 1 \quad j = 1, \ldots, 2^m \tag{IV.26}$$

Le seuil sur le rapport a été choisi égal à 1, les probabilités d'erreur globales α_G et β_G sont donc déterminées a posteriori en fonction de ce seuil, des caractéristiques α_{ℓ,k_ℓ} et β_{ℓ,k_ℓ} des classifieurs individuels et du nombre de capteurs en panne selon les scénarios S_j.

Cette règle de décision donne, en plus de la décision, une indication sur le scénario de panne et donc sur l'état des capteurs.

IV.2.4.3 Simulations et résultats

Afin d'estimer les performances de la règle de fusion donnée par la relation (IV.26), nous avons procédé à des simulations des sorties Y_ℓ des classifieurs individuels selon les

caractéristiques α_{ℓ,k_ℓ} et β_{ℓ,k_ℓ} des classifieurs individuels associées aux hypothèses H_0 et H_1. Le nombre k_ℓ de capteurs en panne dans chaque sous-espace de représentation est lié implicitement aux valeurs du nombre total $Ncp^{(m)}$ de capteurs en panne pour un scénario donné.

Nous avons ensuite estimé les probabilités d'erreur globales α_G et β_G de la règle reposant sur le rapport des maximums de vraisemblance proposée en (IV.26) et que nous noterons R_{MV} par la suite. Enfin pour tester un éventuel gain de performance significatif avec cette règle, nous avons également appliqué la règle du vote majoritaire que nous noterons R_{MAJ} et qui s'exprime selon la relation (IV.20) page 73.

Principe des simulations

1. Pour $Ncp^{(m)} = 0, \ldots, m$
   (a) on tire au hasard un scénario S_j parmi les $C_m^{Ncp^{(m)}}$ scénarios possibles
   (b) on simule n sorties Y_ℓ ($\ell = 1, \ldots, L$) sous H_0 et H_1, de la manière suivante :
   Sous H_0 : on tire 0 avec la probabilité $1 - \alpha_{\ell,k_\ell}$ et 1 avec la probabilité α_{ℓ,k_ℓ}
   Sous H_1 : on tire 0 avec la probabilité β_{ℓ,k_ℓ} et 1 avec la probabilité $1 - \beta_{\ell,k_\ell}$
2. on calcule les estimations $\hat{\alpha}_G$ et $\hat{\beta}_G$ de α_G et β_G obtenues avec la règle R_{MV}
3. on calcule les estimations $\hat{\alpha}_G$ et $\hat{\beta}_G$ de α_G et β_G obtenues avec la règle R_{MAJ}

Données des simulations

- nombre total de capteurs : $m = 10$;
- dimension de chaque sous-espace de représentation : $d = 2$;
- nombre total de classifieurs (sous-espaces de représentation) : $L = 5$;
- caractéristiques des classifieurs individuels :

$$\left| \begin{array}{lll} \alpha_{\ell,0} = 0.2 & \alpha_{\ell,1} = 0.4 & \alpha_{\ell,2} = 0.9 \\ \beta_{\ell,0} = 0.05 & \beta_{\ell,1} = 0.1 & \beta_{\ell,2} = 0.1 \end{array} \right. \quad \forall \ell = 1, \ldots, 5$$

- probabilité de panne d'un capteur dans le système : $p_{cp} = 0.2$
- $n = 100$ simulations sous H_0 et sous H_1

Résultats

La figure IV.1 présente les estimations $\hat{\alpha}_G$ obtenues par les deux règles R_{MV} et R_{MAJ} en fonction du nombre total $Ncp^{(m)}$ de capteurs en panne et la figure IV.2 présente les estimations $\hat{\beta}_G$.

D'après la figure IV.1, nous pouvons constater que, sous l'hypothèse H_0, la règle de fusion reposant sur le test du rapport des maximums de vraisemblance est uniformément la plus performante par rapport au nombre total de capteurs en panne. Particulièrement, jusqu'à la moitié des capteurs en panne ($Ncp^{(m)} \leq 5$), les taux d'erreur de

la règle R_{MV} restent inférieurs à 10% tandis que la performance du vote majoritaire se dégrade très vite. Quand plus de la moitié des capteurs sont en panne ($Ncp^{(m)} > 6$), la performance de la règle R_{MV} se dégrade toutefois assez brutalement et rapidement. Sur la figure IV.2, nous pouvons observer que, sous l'hypothèse H_1, c'est la règle du vote majoritaire qui apparaît comme la plus performante, mais les deux règles restent assez comparables de ce point de vue.

Les résultats des simulations illustrent donc bien que les performances de la règle de décision globale peuvent être améliorées en intégrant le nombre de capteurs en panne dans le processus décisionnel. Toutefois, la mise en œuvre de la règle R_{MV} que nous proposons est coûteuse en temps de calcul et son étude s'inscrit dans nos perspectives.

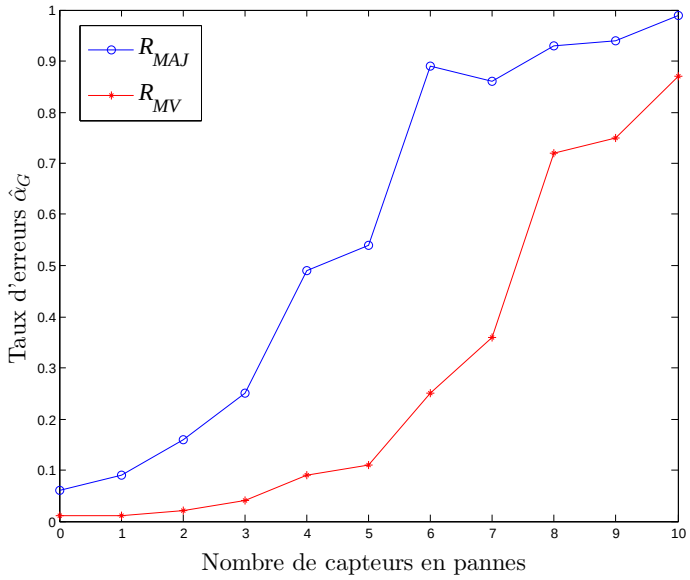


Figure IV.1 – $\hat{\alpha}_G$ en fonction de $Ncp^{(m)}$ pour les deux règles R_{MV} et R_{MAJ}

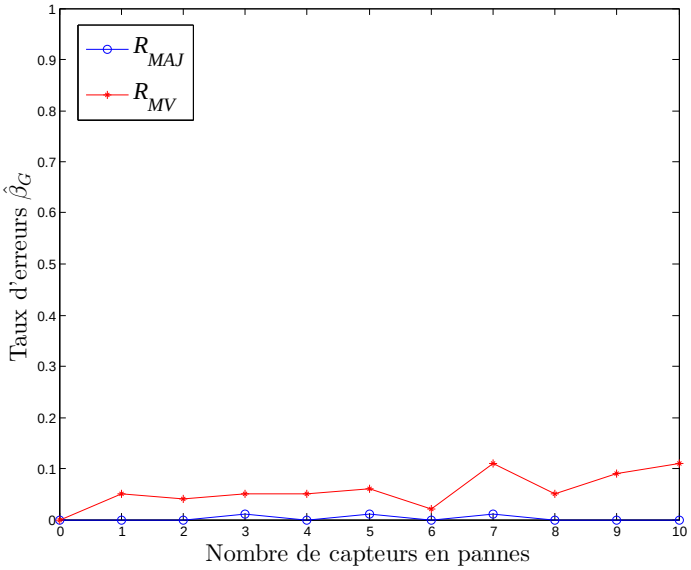


Figure IV.2 – $\hat{\beta}_G$ en fonction de $Ncp^{(m)}$ pour les deux règles R_{MV} et R_{MAJ}

IV.3 Règles de fusion de décisions dans le cas de classifieurs corrélés

Dans la section précédente, nous avons vu que l'indépendance des classifieurs individuels reposait sur la sélection de sous-espaces de représentation disjoints, à condition que les sorties des m capteurs soient indépendantes.

Très souvent, le nombre de capteurs est limité et ce mode de sélection restreint donc considérablement le nombre L de classifieurs indépendants disponibles. Par exemple, dans le cas des simulations que nous avons effectuées à la section IV.2.4.3 précédente, pour $m = 10$ capteurs et $d = 2$, nous pouvons construire au total $N = 45$ (C_{10}^{2}) classifieurs différents, mais au maximum $L = \lfloor \frac{m}{d} \rfloor = 5$ classifieurs indépendants ($L = \lfloor x \rfloor$ représente la partie entière de x et très souvent $\lfloor \frac{m}{d} \rfloor \ll C_{m}^{d}$). Toutefois, Ho (1993, 1995, 1998a) a montré expérimentalement que la performance de classification s'améliore de façon quasi monotone avec l'augmentation du nombre de classifieurs. Par ailleurs, nous souhaitons construire une règle de fusion que nous pouvons maîtriser finement et évaluer aisément en cas de panne de capteurs et plus généralement lorsque l'environnement est altéré. Pour pouvoir prendre en compte et mesurer l'impact de ces changements il nous faut alors établir un modèle permettant de lier les performances des classifieurs individuels à celles de la règle globale. Pour toutes les raisons évoquées ici, il semble crucial de pouvoir prendre en compte un nombre suffisamment important de classifieurs. Ce choix offre en outre, la possibilité de disposer de sous-espaces non altérés ou peu altérés.

Dans ce contexte, il nous faut alors abandonner la contrainte d'indépendance des classifieurs individuels. La conséquence immédiate est que le vecteur décisionnel $\boldsymbol{Y}$ est désormais constitué de variables de Bernoulli corrélées et la difficulté majeure repose sur l'expression de sa loi conjointe qui est au cœur des règles de fusion que nous avons vues précédemment, *Bayes* (IV.6), *Neyman-Pearson* (IV.14), et R_{MV} (IV.26).

Bien que les règles de fusion que nous avons vues ne reposent pas exclusivement sur la loi binomiale, nous avons passé en revue quelques travaux portant sur sa généralisation lorsque les variables de Bernoulli sont corrélées. Nous avons pu noter que dans quelques situations précises, la règle de fusion reposait sur la somme des variables décisionnelles associées aux classifieurs individuels pris en compte. C'est le cas par exemple de la règle du vote majoritaire. Dans le cas général (avec ou sans indépendance), il est donc important de connaître la loi conjointe du vecteur décisionnel $\boldsymbol{Y}$ pour pouvoir établir la loi de probabilité de la somme $W = \sum_{\ell=1}^{L} Y_\ell$ de ses composantes. En remarquant par ailleurs que pour une réalisation $\boldsymbol{y}$ d'un vecteur de variables de Bernoulli, on a :

$$\sum_{\ell=1}^{L} y_\ell = \|\boldsymbol{y}\|$$

la loi de probabilité de W s'écrit simplement :

$$P[W = w] = \sum_{\boldsymbol{y} \mid \|\boldsymbol{y}\| = w} P[\boldsymbol{Y} = \boldsymbol{y}] \quad \forall\, w \in \Omega_W = \{0, 1, , \ldots, L\} \tag{IV.27}$$

Quelques approximations reposant sur diverses approches ont été proposées pour proposer des modèles de généralisation de la loi binomiale. Citons par exemple Chen (1975) (approximation de Poisson), Yu et al. (1983) (modélisation d'arbres de dépendance) ainsi que d'autres techniques tenant compte de types particuliers de dépendances (Denuit et al. - 2002).

Altham (1978) a proposé un modèle de distribution à deux paramètres qui peut être obtenu par le produit de la distribution binomiale standard par un facteur correcteur (modèle multiplicatif) ou bien par un mélange de trois distributions binomiales standard de paramètres différents (modèle additif). Cette généralisation additive de la distribution binomiale est identique à un autre modèle binomial corrélé (Kupper et Haseman - 1978), nommé distribution CB et qui tient compte des coefficients de corrélation entre les variables de Bernoulli reposant sur l'idée de Bahadur (1961). Le modèle de Bahadur (1961), plus précisément nommé modèle de Bahadur-Lazarsfeld, propose une représentation complète de la loi conjointe de variables de Bernoulli corrélées. Ce modèle s'exprime comme le produit de la loi conjointe sous hypothèse d'indépendance par une fonction correctrice intégrant toutes les corrélations entre les variables.

Nous nous sommes inspirés de ce modèle pour modéliser la loi conjointe du vecteur décisionnel $\boldsymbol{Y}$ dans le cas où les classifieurs sont corrélés. Toutefois, dans le cadre de cette thèse nous n'avons pour le moment abordé que le cas où aucun capteur n'est en panne. L'extension de cette approche avec prise en compte des pannes de capteurs sera évoquée dans nos perspectives. Nous allons donc ici présenter le modèle de Bahadur ainsi qu'une approche proposée par Van Der Geest (2005) qui évite la prise en compte de toutes les corrélations puis nous présenterons quelques résultats obtenus sur une simulation.

IV.3.1 Modèle de Bahadur-Lazarsfeld

On considère donc un vecteur aléatoire $\boldsymbol{Y} = [Y_1\ ,\ Y_2\ ,\ \ldots\ ,\ Y_L]^t$ constitué de L variables de Bernoulli corrélées de paramètres respectifs p_ℓ définis ainsi :

$$p_\ell = E\left[Y_\ell\right] = P\left[Y_\ell = 1\right] \quad (\ell = 1, \ldots, L) \tag{IV.28}$$

Le modèle de Bahadur-Lazarsfeld, repose sur les L probabilités marginales p_ℓ et sur les $2^L - L - 1$ coefficients de corrélation d'ordre 2 à L définis comme suit,

$$\begin{aligned} &r_{ij} = E\left[Z_i Z_j\right] \qquad \forall\, i, j \ \ (1 \leq i < j \leq L) \\ &r_{ijk} = E\left[Z_i Z_j Z_k\right] \quad \forall\, i, j, k \ \ (1 \leq i < j < k \leq L) \\ &\vdots \\ &r_{12\ldots L} = E\left[Z_1 Z_2 \ldots Z_L\right] \end{aligned} \tag{IV.29}$$

les variables Z_ℓ sont les versions centrées réduites des variables Y_ℓ,

$$Z_\ell = \frac{Y_\ell - p_\ell}{\sqrt{p_\ell(1 - p_\ell)}} \tag{IV.30}$$

il y a $\complement_L^k$ coefficients de corrélation d'ordre k, donc $\sum_{k=2}^{L} \complement_L^k = 2^L - L - 1$ au total.

Sous hypothèse d'indépendance des variables Y_ℓ, la loi conjointe de $\boldsymbol{Y}$ s'écrit :

$$P_{indep}\left[\,\boldsymbol{Y}=\boldsymbol{y}\,\right]=P\left[Y_1=y_1,\ldots,Y_L=y_L\right]=\prod_{\ell=1}^{L}p_\ell^{y_\ell}(1-p_\ell)^{1-y_\ell} \tag{IV.31}$$

et le modèle géńéral sans hypothèse d'indépendance, proposé par Bahadur s'écrit :

$$P\left[\,\boldsymbol{Y}=\boldsymbol{y}\,\right]=f(\boldsymbol{y})\times P_{indep}\left[\,\boldsymbol{Y}=\boldsymbol{y}\,\right]=f(\boldsymbol{y})\times\prod_{\ell=1}^{L}p_\ell^{y_\ell}(1-p_\ell)^{1-y_\ell}=p(\boldsymbol{y}) \tag{IV.32}$$

où la fonction correctrice f s'écrit,

$$f(\boldsymbol{y})=1+\sum_{i<j}r_{ij}z_iz_j+\sum_{i<j<k}r_{ijk}z_iz_jz_k+\ldots+r_{12\ldots L}z_1z_2\ldots z_L \tag{IV.33}$$

La mise en œuvre du modèle de Bahadur-Lazarsfeld s'avère cependant très compliquée voire impossible lorsque la dimension du vecteur engendre un nombre trop important de corrélations à déterminer (fixées a priori ou estimées). Une solution habituelle consiste à supposer que les corrélations sont nulles à partir d'un certain ordre. Cependant, une telle troncature dans la représentation du modèle de Bahadur-Lazarsfeld a pour conséquence le risque de produire une distribution impropre, avec par exemple des probabilités négatives ou supérieures à 1 pour certaines réalisations $\boldsymbol{y}$ du vecteur $\boldsymbol{Y}$ (Yu et al. - 1983).

IV.3.2 Modèle du maximum d'entropie

Une alternative intéressante au problème posé par le modèle de Bahadur-Lazarsfeld consiste à utiliser une approche d'inférence reposant sur l'entropie pour estimer les distributions de probabilité à partir d'informations incomplètes (Levine et Tribus - 1979). Cette approche a été développée par Van Der Geest (2005) dans le cas de la distribution conjointe de variables de Bernoulli corrélées. L'intérêt de cette méthode est de permettre l'estimation de la distribution conjointe en requérant uniquement les probabilités marginales et l'ensemble (éventuellement incomplet) des corrélations d'ordre 2. Ce dernier point sera détaillé dans la suite.

IV.3.2.1 Principe de la construction du modèle

La méthode proposée par Van Der Geest (2005) consiste à maximiser l'entropie de la distribution conjointe $p(\boldsymbol{y})=P\left[\,\boldsymbol{Y}=\boldsymbol{y}\,\right]$ avec des contraintes limitées aux probabilités marginales et aux corrélations d'ordre 2. Le problème posé s'énonce ainsi :

$$\text{Maximiser}\qquad -\sum_{k=1}^{2^L}p(\boldsymbol{y}_k)\log(p(\boldsymbol{y}_k)) \tag{IV.34}$$

$$\text{Sous les contraintes}\quad\left|\begin{array}{ll}\displaystyle\sum_{k=1}^{2^L}p(\boldsymbol{y}_k)=1 & \text{(a)}\\ \displaystyle\sum_{k=1}^{2^L}A_{sk}p(\boldsymbol{y}_k)=b_s\quad s=1,\ldots,S & \text{(b)}\end{array}\right. \tag{IV.35}$$

- le vecteur $\boldsymbol{b} = [b_1\ ,\ b_2\ ,\ \ldots\ ,\ b_S]^t$ est constitué des L probabilités marginales p_ℓ (Cf. IV.28) et des C_L^2 moments d'ordre 2 :

$$E\left[Y_i Y_j\ \right] = p_i p_j + r_{ij}\sqrt{p_i p_j (1-p_i)(1-p_j)} \tag{IV.36}$$

  où r_{ij} est le coefficient de corrélation d'ordre 2 entre Y_i et Y_j
- $S = L + C_L^2 = L(L+1)/2$ est donc le nombre de contraintes fixées par l'utilisateur
- A_{sk} est le terme général d'une matrice $\boldsymbol{A}$ $(S \times 2^L)$ dont les lignes sont constituées de 0 et de 1 permettant de sélectionner les termes des probabilités conjointes $p(\boldsymbol{y}_k)$ qui contribuent au calcul de chaque probabilité marginale ou de chaque moment d'ordre 2 correspondant au membre de droite b_s

IV.3.2.2 Précisions sur les notations et la formulation du problème

Afin d'expliciter l'écriture de la seconde contrainte (IV.35-b) au moyen de la matrice $\boldsymbol{A}$, il est utile de préciser la convention d'indexation des réalisations $\boldsymbol{y}_k$. En effet, il n'existe pas de relation d'ordre dans l'ensemble des 2^L réalisations possibles de $\boldsymbol{Y}$, mais il faut néanmoins pouvoir numéroter ces réalisations pour que les équations (IV.35) aient un sens et puissent être mises en œuvre.

Convention de notation des réalisations $\boldsymbol{y}_k$

La convention adpotée repose sur le fait que chaque réalisation est la représentation binaire d'un entier codé sur L bits. Si on choisit de noter $\boldsymbol{y}_k$ la réalisation correspondant au nombre entier $k = 2^L - \textit{valeur de l'entier codé par } \boldsymbol{y}_k$, on définit une bijection de l'ensemble des réalisations vers l'ensemble des entiers compris entre 1 et 2^L. Une illustration de cette convention est donnée dans le tableau IV.4 ci-dessous.

y_1	y_2	y_3	entier codé	2^L-entier codé	$\boldsymbol{y}_k$
1	1	1	7	1	$\boldsymbol{y}_1$
1	1	0	6	2	$\boldsymbol{y}_2$
1	0	1	5	3	$\boldsymbol{y}_3$
1	0	0	4	4	$\boldsymbol{y}_4$
0	1	1	3	5	$\boldsymbol{y}_5$
0	1	0	2	6	$\boldsymbol{y}_6$
0	0	1	1	7	$\boldsymbol{y}_7$
0	0	0	0	8	$\boldsymbol{y}_8$

Tableau IV.4 – Illustration de la convention d'indexation des réalisations $\boldsymbol{y}_k$ pour $L = 3$

Précisions sur la matrice $\boldsymbol{A}$ et l'équation (IV.35 - b)

Reprenons l'exemple du cas présenté dans le tableau IV.4 et écrivons la probabilité marginale p_1

$$\begin{aligned} p_1 &= P[Y_1 = 1] = P[\boldsymbol{Y} = \boldsymbol{y}_1] + P[\boldsymbol{Y} = \boldsymbol{y}_2] + P[\boldsymbol{Y} = \boldsymbol{y}_3] + P[\boldsymbol{Y} = \boldsymbol{y}_4] \\ &= p(\boldsymbol{y}_1) + p(\boldsymbol{y}_2) + p(\boldsymbol{y}_3) + p(\boldsymbol{y}_4) \\ &= 1 \times [p(\boldsymbol{y}_1) + p(\boldsymbol{y}_2) + p(\boldsymbol{y}_3) + p(\boldsymbol{y}_4)] + 0 \times [p(\boldsymbol{y}_5) + p(\boldsymbol{y}_6) + p(\boldsymbol{y}_7) + p(\boldsymbol{y}_8)] \\ &= \sum_{k=1}^{8} A_{1k} \times p(\boldsymbol{y}_k) \end{aligned} \tag{IV.37}$$

La première ligne de la matrice $\boldsymbol{A}$ est donc constituée des éléments A_{1k} pour $k = 1, \ldots, 2^L$ et on peut observer qu'elle correspond à la première colonne du tableau IV.4. En exprimant la seconde probabilité marginale p_2, on observe aisément que la seconde ligne (A_{2k}, $k = 1, \ldots, 2^L$) de $\boldsymbol{A}$ correspond à la seconde colonne du tableau IV.4. On peut donc généraliser en disant que la ligne A_{sk} de $\boldsymbol{A}$ correspond à la colonne k du tableau d'indexation des réalisations de $\boldsymbol{Y}$ pour s allant de 1 à L.

Ecrivons maintenant le moment d'ordre 2 $E[Y_1 Y_2]$

$$\begin{aligned} E[Y_1 Y_2] &= P[Y_1 = 1, Y_2 = 1] = P[\boldsymbol{Y} = \boldsymbol{y}_1] + P[\boldsymbol{Y} = \boldsymbol{y}_2] \\ &= p(\boldsymbol{y}_1) + p(\boldsymbol{y}_2) \\ &= 1 \times [p(\boldsymbol{y}_1) + p(\boldsymbol{y}_2)] + 0 \times [p(\boldsymbol{y}_3) + p(\boldsymbol{y}_4) + p(\boldsymbol{y}_5) + p(\boldsymbol{y}_6) + p(\boldsymbol{y}_7) + p(\boldsymbol{y}_8)] \\ &= \sum_{k=1}^{8} A_{1k} \times A_{2k} \times p(\boldsymbol{y}_k) \end{aligned} \tag{IV.38}$$

l'équation IV.38 nous montre que la ligne 4 $(L+1)$ de $\boldsymbol{A}$ est obtenue par le produit terme à terme des lignes 1 et 2.

Plus généralement, pour s allant de $L+1$ à $L(L+1)/2$ la ligne s de $\boldsymbol{A}$ correspondant au moment $E[Y_i Y_j]$ sera obtenue par les produit terme à terme des lignes i et j de $\boldsymbol{A}$ ($1 \leq i, j \leq L$) et finalement les contraintes exprimées par l'équation (IV.35) peuvent s'écrire sous la forme équivalente :

$$\left| \begin{array}{l} \sum_{k=1}^{2^L} p(\boldsymbol{y}_k) = 1 \\ \boldsymbol{A} \cdot [p(\boldsymbol{y}_1), p(\boldsymbol{y}_2), \ldots, p(\boldsymbol{y}_{2^L})]^t = \boldsymbol{b} \end{array} \right. \tag{IV.39}$$

Si certaines corrélations r_{ij} sont inconnues et donc, les moments d'ordre 2 correspondants (Cf. équation IV.36), elles peuvent être omises dans le vecteur $\boldsymbol{b}$ (Cf. équation de contrainte IV.35-b ou IV.39), mais pas mises à zéro. Cela réduit le nombre S de contraintes mais n'empêche pas la production d'une solution. Les valeurs manquantes de corrélations d'ordre 2 ainsi que celles des corrélations d'ordre supérieur peuvent ensuite être déduites au moyen de la distribution conjointe, solution du problème de maximisation de l'entropie (Van Der Geest - 2005).

IV.3.2.3 Obtention de la solution

Jaynes (1957) a montré que pour la méthode des multiplicateurs de Lagrange, la distribution d'entropie maximum est de la forme :

$$p(\boldsymbol{y}_k) = \exp\left(-\lambda_0 - \sum_{s=1}^{S} \lambda_s A_{sk}\right) \qquad \forall\, k = 1, \ldots, 2^L \tag{IV.40}$$

Les multiplicateurs de Lagrange $(\lambda_0, \lambda_1, \ldots, \lambda_S)$ sont déterminés par les $S+1$ contraintes (Cf. équations IV.35-a et IV.35-b). D'après la condition de normalisation (IV.35-a), on peut d'abord déduire l'expression de λ_0,

$$\exp(\lambda_0) = \sum_{k=1}^{2^L} \exp\left(-\sum_{s=1}^{S} \lambda_s A_{sk}\right) \tag{IV.41}$$

En réinjectant les expressions (IV.40) et (IV.41) dans la deuxième contrainte (IV.35-b), on obtient alors,

$$\sum_{k=1}^{2^L} B_{sk} \exp\left(-\sum_{t=1}^{S} \lambda_t A_{tk}\right) = 0 \quad s = 1, \ldots, S \tag{IV.42}$$

où $B_{sk} = A_{sk} - b_s$

IV.4 Simulations et validation des modèles conjoints avec corrélations

Comme nous l'avons précisé page 79, en préambule à cette section, dans beaucoup de situations le nombre initial de capteurs est limité. De ce fait, la restriction à des sous-espaces disjoints pour construire un ensemble de classifieurs indépendants réduit considérablement le nombre de classifieurs. Cet aspect peut alors présenter un fort préjudice en termes de performances globales en cas de perte de capteurs et c'est une des raisons qui motivent notre choix de nous affranchir de la contrainte d'indépendance.

Dans ce contexte, on peut a priori envisager de considérer l'ensemble complet des $N = C_m^d$ sous-espaces de représentation et donc des $L = N$ classifieurs associés.

Toutefois la mise en œuvre de cette solution n'est acceptable en pratique que si N n'est pas trop grand. Nous avons vu par exemple que le modèle de Bahadur (Cf. section IV.3.1) requiert de fixer ou d'estimer les corrélations de tous ordres. Même en limitant le modèle aux corrélations d'ordre 3 en tronquant la fonction correctrice (Cf. équation IV.33), on aura par exemple pour un système à $m = 5$ capteurs et des sous-espaces de dimension $d = 2$, un nombre total de classifieurs égal à 10 et le nombre total de corrélations à considérer dans le modèle sera égal à 165.

Cette partie ayant pour objet principal de tester la validité des méthodes de Bahadur et du maximum d'entropie (Cf. section IV.3.2), nous nous sommes donc restreints ici à l'étude d'un cas très simple pour deux raisons principales. La première est que nous souhaitons ne pas tronquer le modèle de Bahadur afin de comparer sa solution à celle fournie par la méthode du maximum d'entropie. La seconde repose sur le fait que nous souhaitions avoir la possibilité d'établir la loi conjointe vraie de $\boldsymbol{Y}$ afin de valider les solutions des modèles de Bahadur et du maximum d'entropie.

Enfin, nous avons testé les performances de la règle de fusion en les comparant à celles d'une règle reposant sur l'espace de représentation initial $\mathcal{X}^m$.

IV.4.1 Simulations

Nous avons effectué des simulations conformément au schéma du processus décisionnel représenté dans le tableau IV.1 page 74 et dans ce contexte, nous avons simulé des échantillons de réalisations du vecteur d'attributs $\boldsymbol{X}$.

Paramètres des simulations

- nombre de capteurs : $m = 3$
- dimension des sous-espaces : $d = 2$
- nombre de classifieurs $L = N = \mathsf{C}_3^2 = 3$
- sorties des capteurs (attributs) $\left| \begin{array}{l} X_i \sim \mathcal{N}(\mu_0, 1) \text{ sous } H_0 \\ X_i \sim \mathcal{N}(\mu_1, 1) \text{ sous } H_1 \end{array} \right. \quad \forall\, i = 1, \ldots, 3$

Règles de décision des classifieurs individuels

Pour chaque classifieur nous avons adopté la règle simple suivante :

$$\begin{cases} Y_1 = h_1(X_1, X_2) = 1 & \text{si} \quad X_1 + X_2 < 0 \\ Y_2 = h_2(X_1, X_3) = 1 & \text{si} \quad X_1 + X_3 < 0 \\ Y_3 = h_3(X_2, X_3) = 1 & \text{si} \quad X_2 + X_3 < 0 \end{cases} \tag{IV.43}$$

Nous supposons par ailleurs que les capteurs sont indépendants et identiques. De ce fait, les performance individuelles de chaque classifieur seront données par des probabilités d'erreur de première et de seconde espèce identiques,

$$\alpha_\ell = P\left[Y_\ell = 1 \mid H_0\right] = \alpha \quad \text{et} \quad \beta_\ell = P\left[Y_\ell = 0 \mid H_1\right] = \beta \qquad \forall\, \ell = 1, \ldots, N$$

Les seuils des règles individuelles définies en (IV.43) ont été choisis nuls pour des raisons de simplification des calculs. Les moyennes μ_0 et μ_1 ont donc été fixées en fonction de α et β en accord avec ces règles. On a alors,

$$\begin{cases} \mu_0 = E\left[X_i \mid H_0\right] = \dfrac{\sigma u_{1-\alpha}}{\sqrt{2}} \\ \mu_1 = E\left[X_i \mid H_1\right] = \dfrac{\sigma u_{\beta}}{\sqrt{2}} \end{cases} \quad \forall i = 1, \ldots, m \tag{IV.44}$$

u_p représente le quantile de niveau p d'une loi normale centrée-réduite.

En outre, nous avons fixé $\sigma^2 = 1$ ainsi que $\alpha = 0.15$ et $\beta = 0.05$.

Loi conjointe vraie

Il est assez simple de calculer les probabilités des $2^3 = 8$ réalisations du vecteur décisionnel $\boldsymbol{Y}$. Par exemple, la probabilité de $\boldsymbol{y} = [0,\ 0,\ 0]^t$ sera donnée par,

$$\begin{aligned} P_{000} &= P\left[Y_1 = 0, Y_2 = 0, Y_3 = 0\right] = P\left[X_1 > -X_2, X_1 > -X_3, X_2 > -X_3\right] \\ &= \int_{-\infty}^{\infty} f(x_3) \int_{-x_3}^{\infty} f(x_2) \int_{\max(-x_2,-x_3)}^{\infty} f(x_1) dx_1 dx_2 dx_3 \end{aligned}$$

Les intégrales ont été évaluées numériquement pour les $2^3 = 8$ probabilités conjointes.

Loi conjointe calculée avec le modèle de Bahadur

Les variables centrées-réduites Z_i (Cf. IV.30) qui interviennent dans le calcul de la fonction correctrice (Cf. équation IV.33) s'expriment différemment selon H_0 et H_1, donnant les expressions des lois conjointes de $\boldsymbol{Y}$ résumées dans le tableau IV.5

Hypothèse	H_0	H_1
Z_ℓ	$\dfrac{Y_\ell - \alpha_\ell}{\sqrt{\alpha_\ell(1-\alpha_\ell)}}$	$\dfrac{Y_\ell - (1-\beta_\ell)}{\sqrt{\beta_\ell(1-\beta_\ell)}}$
$P\left[\boldsymbol{Y} = \boldsymbol{y} \mid H_i\right]$	$f_0(\boldsymbol{y}) \times \prod_{\ell=1}^{3} \alpha_\ell^{y_\ell}(1-\alpha_\ell)^{1-y_\ell}$	$f_1(\boldsymbol{y}) \times \prod_{\ell=1}^{3} \beta_\ell^{1-y_\ell}(1-\beta_\ell)^{y_\ell}$

Tableau IV.5 – paramètres du modèle de Bahadur

Loi conjointe calculée avec le modèle du maximum d'entropie

Dans le cas de la méthode du maximum d'entropie, ce sont les termes du vecteur $\boldsymbol{b}$ (Cf. équation IV.39 des contraintes) qui s'expriment différemment selon H_0 et H_1 (tableau IV.6)

Hypothèse	H_0	H_1
$p_\ell = P[Y_\ell = 1 \mid H_i]$	α_ℓ	$1 - \beta_\ell$
$E[Y_j Y_\ell \mid H_i]$	$\alpha_j\alpha_\ell + r_{j,\ell}^{(0)}\sqrt{\alpha_j\alpha_\ell(1-\alpha_j)(1-\alpha_\ell)}$	$(1-\beta_j)(1-\beta_\ell) + r_{j,\ell}^{(1)}\sqrt{\beta_j\beta_\ell(1-\beta_j)(1-\beta_\ell)}$

Tableau IV.6 – paramètres du modèle du maximum d'entropie

IV.4.2 Résultats

Pour l'estimation de la loi conjointe de $\boldsymbol{Y}$, nous avons constitué un ensemble d'apprentissage de $N_{app} = 100000$ réalisations de X_1, X_2 et X_3 sous chaque hypothèse afin d'estimer les coefficients $r_{j,\ell}^{(0)}$ et $r_{j,\ell}^{(1)}$ qui sont requis pour les deux méthodes.

Le tableau IV.7 représente respectivement les lois conjointes obtenues par le modèle réel (notée $P(y_1, y_2, y_3)$), le modèle de Bahadur-Lazarsfeld (notée $\hat{P}_{BL}$) et le modèle du maximum d'entropie (notée $\hat{P}_{ME}$). Les résultats montrent une très bonne adéquation des approximations de Bahadur-Lazarsfeld et du maximum d'entropie avec le modèle réel.

$\boldsymbol{Y}$			Sous H_0			Sous H_1		
y_1	y_2	y_3	$P(y_1,y_2,y_3)$	$\hat{P}_{BL}$	$\hat{P}_{ME}$	$P(y_1,y_2,y_3)$	$\hat{P}_{BL}$	$\hat{P}_{ME}$
0	0	0	0.6923463	0.692665	0.690427	0.004957	0.00499	0.005884
0	0	1	0.06535	0.065541	0.067818	0.007212	0.007047	0.006291
0	1	0	0.06535	0.065027	0.066581	0.007212	0.007063	0.006274
1	0	0	0.06535	0.065387	0.068083	0.007212	0.007676	0.005565
0	1	1	0.026953	0.026769	0.025174	0.0306	0.030893	0.031550
1	0	1	0.026953	0.026409	0.023672	0.0306	0.030281	0.032259
1	1	0	0.026953	0.026923	0.024908	0.0306	0.030265	0.032276
1	1	1	0.030742	0.031280	0.033336	0.88161	0.881778	0.879899

Tableau IV.7 – Loi de $\boldsymbol{Y}$: modèle réel et approximations de Bahadur-Lazarsfeld et du maximum d'entropie

Pour la règle de décision globale, nous avons opté pour le vote majoritaire (Cf. équation IV.20). Nous avons classé 100 fois un ensemble de $N_{test} = 1000$ données test (i.e. réalisations de X_1, X_2 et X_3) pour chaque hypothèse afin d'estimer les probabilités d'erreur du classifieur global et de les comparer à celles $\alpha_{P_{\boldsymbol{Y}}}$ et $\beta_{P_{\boldsymbol{Y}}}$ calculées avec la loi conjointe réelle de $\boldsymbol{Y}$ (i.e. $P(y_1, y_2, y_3)$ dans le tableau IV.7) selon la formule de calcul (IV.27) donnée page 79.

Par ailleurs, nous avons également estimé les probabilités d'erreur α_{CU} et β_{CU} d'un classifieur unique défini sur l'espace complet $\mathcal{X}_3$ avec la règle suivante :

$$\alpha_{CU} = P\left[X_1 + X_2 + X_3 < 0 \mid H_0\right] \quad \text{ou} \quad \beta_{CU} = P\left[X_1 + X_2 + X_3 \geq 0 \mid H_1\right] \quad \text{(IV.45)}$$

Le tableau IV.8 présente respectivement les taux d'erreurs $(\hat{\alpha}_{test}, \hat{\beta}_{test})$ estimés sur l'ensemble test, $(\alpha_{P_{\boldsymbol{Y}}}, \beta_{P_{\boldsymbol{Y}}})$ calculé avec la loi réelle de $\boldsymbol{Y}$ du classifieur global à vote majoritaire et $(\alpha_{CU}, \beta_{CU})$ reposant sur l'espace complet.

Sous H_0				Sous H_1			
α	$\hat{\alpha}_{test}$	$\alpha_{P_{\boldsymbol{Y}}}$	α_{CU}	β	$\hat{\beta}_{test}$	$\beta_{P_{\boldsymbol{Y}}}$	β_{CU}
0.15	0.111	0.1116	0.10215	0.05	0.02624	0.02659	0.02197

Tableau IV.8 – Taux d'erreur global et individuels

Les résultats (Cf. tableau IV.8) montrent que la loi conjointe du vecteur décisionnel permet effectivement d'évaluer correctement les performances de la règle de décision globale. Les modèles permettent d'établir la relation entre les performances individuelles des classifieurs et la performance globale.

Par ailleurs, si on augmente légèrement le nombre de capteurs ou si on change la dimension des sous-espaces de représentation, le nombre de classifieurs peut devenir très grand (par exemple avec $m = 10$ capteurs et des sous-espace de dimension $d = 3$, on a alors $N = 120$ classifieurs possibles).

Étant donné le coût de calcul engendré on choisira donc souvent de construire un ensemble de classifieurs choisis aléatoirement parmi tous les classifieurs possibles. Cet aspect présente un inconvénient majeur pour l'évaluation des performances de la règle globale si l'on change l'ensemble des classifieurs sur lesquels elle repose, car il faut alors procéder à nouveau à un apprentissage et faire de nouveaux tests.

Sur ce point la connaissance du modèle conjoint des variables décisionnelles de l'ensemble de tous les classifieurs peut apporter une réponse appropriée. Plus précisément, à partir du modèle conjoint complet, on peut déduire aisément les modèles conjoints sous-jacents associés à tous les sous-ensembles et cela devrait permettre d'étudier précisément l'impact de l'ajout ou du retrait de classifieurs dans le processus décisionnel final.

IV.5 Conclusion

Dans ce chapitre, nous avons tout d'abord proposé une nouvelle règle de fusion pour un ensemble de classifieurs indépendants. Cette règle est plus complexe que celle du vote majoritaire, mais elle permet de prendre en compte des perturbations affectant les caractéristiques des classifieurs individuels définis sur des sous-espaces de représentataion issus de l'espace initial. Plus précisément, lorsque la grande majorité des capteurs sont en panne dans le système, il en résulte que les attributs dans l'espace de représentation initial ne sont plus pertinents. Les classifieurs opérant dans les sous-espaces de représentation qui incluent ces attributs deviennent moins performants. Le vote majoritaire en combinant des classifieurs opérant sur des telles sous-espaces devient inopérant.

Notre règle de fusion repose sur un test du rapport des maximums de vraisemblance par rapport à l'ensemble des scénarios possibles de pannes de capteurs. Les résultats obtenus sur des simulations ont montré que les performances globales de notre règle de décision sont bien conservées en présence de pannes de capteurs. Son inconvénient majeur réside dans le coût de calcul car il faut énumérer tous les scénarios possibles de pannes de capteurs.

Il faut noter que la méthode de classification décrite ci-dessus a été conçue pour un ensemble de classifieurs indépendants. Toutefois la contrainte d'indépendance est généralement difficile à satisfaire dans les applications réelles, ce qui a pour conséquence d'engendrer des décisions locales dépendantes. Dans ce cas la modélisation de la règle de fusion devient plus complexe.

Dans un second temps, nous avons donc étudié deux modèles permettant d'estimer la loi conjointe de variables décisionnelles de Bernoulli corrélées. Les résultats ayant montré que ces modèles (Bahadur-Lazarsfeld et maximum d'entropie) offraient une très bonne adéquation avec le modèle réel et confortent notre approche consistant à utiliser la connaissance de la loi conjointe du vecteur décisionnel pour évaluer les performances de la règle globale. Nous avons ainsi pu montrer que les probabilités d'erreur de la règle globale estimées sur un ensemble de données test étaient comparables à celles données par la loi conjointe du vecteur décisionnel. Dans le cadre de ces simulations sur des données test, nous avons également utilisé une règle reposant sur un classifieur unique opérant dans l'espace de représentation initial. Les performances de cette règle sont sensiblement meilleurs que celles de la règle reposant sur plusieurs classifieurs dépendants. Ce résultat est assez normal dans la mesure où tous les capteurs sont opérationnels.

L'inconvénient majeur du modèle de Bahadur-Lazarsfeld réside dans l'évaluation exhaustive des coefficients de corrélation tandis que celle du modèle du maximum d'entropie réside dans le nombre de réalisations du vecteur décisionnel qui croît très vite avec le nombre total de classifieurs. Ces deux aspects posent donc un problème de mise en œuvre dans le cas d'un grand nombre de classifieurs.

Chapitre V

Perspectives

Dans ce chapitre, nous nous consacrons à deux extensions possibles aux travaux du chapitre IV sur la modélisation probabiliste du classifieur global. D'une part, nous proposons une piste de simplification du modèle conjoint dans le but de pouvoir l'estimer par la méthode du maximum d'entropie dans le cas d'un grand nombre de classifieurs. D'autre part, nous considérons le problème du réglage des classifieurs individuels pour l'obtention d'une performance globale donnée.

V.1 Simplification du modèle du maximum d'entropie

V.1.1 Présentation du problème

Comme nous venons de le voir dans le chapitre IV, les lois conjointes produites par les modèles de Bahadur-Lazarsfeld et du maximum d'entropie présentent une très bonne adéquation avec le modèle réel pour des variables de Bernoulli corrélées. Cependant la représentation complète du modèle de Bahadur-Lazarsfeld nécessite de fixer ou d'estimer tous les coefficients de corrélation entre les variables décisionnelles. Lorsque la dimension du vecteur décisionnel constitué de ces variables devient importante, la mise en œuvre de ce modèle n'est plus possible. Une possibilité réside dans la troncature de la représentation qui omet les coefficients de corrélation d'ordre supérieur à une certaine valeur en les considérant nuls, mais cette alternative n'est pas très satisfaisante car elle peut engendrer une solution ne respectant plus les axiomes des probabilités.

Une alternative intéressante de maximisation sous contrainte de l'entropie de la distribution conjointe a été proposée par Van Der Geest (2005), les contraintes faisant intervenir uniquement les probabilités marginales et les coefficients de corrélation d'ordre 2. Comme nous l'avons mentionné, cette méthode présente aussi l'avantage de produire une solution en cas de valeurs manquantes de corrélations.

Toutefois, le nombre de coefficients de corrélation d'ordre 2 peut s'avérer très grand et la taille de l'ensemble fondamental $\Omega_{\boldsymbol{Y}}$ des réalisations du vecteur décisionnel $\boldsymbol{Y}$ croît très vite avec le nombre de classifieurs (variables décisionnelles). Ces aspects peuvent donc rapidement constituer un problème crucial de mise en œuvre de la méthode en termes de capacité de calcul. Par conséquent, il nous apparaît indispensable de s'attarder sur la détermination des coefficients de corrélation et aussi de simplifier $\Omega_{\boldsymbol{Y}}$ afin de pouvoir exploiter cette méthode. Pour cela, nous proposons d'opérer une segmentation

de l'ensemble des couples de variables décisionnelles partageant le même coefficient de corrélation et de restructurer $\Omega_{\boldsymbol{Y}}$ en une partition d'événements regroupant les réalisations élémentaires correspondant à des situations décisionnelles similaires ou analogues. Ces deux points sont détaillés dans la suite de cette section et illustrés par un exemple simple.

V.1.2 Détermination des coefficients de corrélation

Le contexte est celui du problème de décision présenté à la section IV.2.4.2 page 74.

Soit donc m capteurs délivrant chacun un attribut et $N = C_m^d$ sous-espaces de représentation de dimension d. Nous considérons l'ensemble des N classifieurs élaborés dans chaque sous-espace et donc le vecteur décisionnel $\boldsymbol{Y} = [Y_1\ ,\ Y_2\ ,\ \ldots\ ,\ Y_N]^t$ constitué des N variables décisionnelles associées à chaque classifieur individuel (Cf. schéma IV.1 page 74). On suppose par ailleurs que les m capteurs sont identiques et opérationnels (pas de panne).

Si l'on considère tous les couples possibles de variables décisionnelles, il y a donc C_N^2 coefficients de corrélation à estimer ou à fixer a priori. Toutefois, comme nous considérons m capteurs identiques, nous suggérons que la corrélation entre deux variables Y_i et Y_j ne dépend que du nombre de capteurs (ou attributs) que les sous-espaces $\mathcal{X}_i^d$ et $\mathcal{X}_j^d$ ont en commun. Ceci permet de réduire le nombre de coefficients de corrélation à déterminer comme le montre l'exemple suivant.

Prenons par exemple, $m = 5$ et $d = 3$, il y a alors $N = C_5^3 = 10$ sous-espaces (classifieurs) différents et donc $N = 10$ variables décisionnelles Y_ℓ (Cf. tableau V.1)

Capteurs sélectionnés pour chaque sous-espace	variable décisionnelle associée
(C_1, C_2, C_3)	Y_1
(C_1, C_2, C_4)	Y_2
(C_1, C_2, C_5)	Y_3
(C_1, C_3, C_4)	Y_4
(C_1, C_3, C_5)	Y_5
(C_1, C_4, C_5)	Y_6
(C_2, C_3, C_4)	Y_7
(C_2, C_3, C_5)	Y_8
(C_2, C_4, C_5)	Y_9
(C_3, C_4, C_5)	Y_{10}

Tableau V.1 – Sélection de 3 capteurs parmi 5 pour chaque sous-espace

Il y a donc en principe $C_{10}^2 = 45$ coefficients de corrélations à déterminer si on considère que les 45 couples (Y_i, Y_j) ont des corrélations différentes. En revanche, si on

prend en compte l'hypothèse que le coefficient de corrélation est le même pour tous les couples associés à des sous-espaces ayant le même nombre de capteurs en commun, il ne reste plus que 2 coefficients de corrélation r_1 et r_2 à déterminer comme le montre le tableau V.2 des corrélations.

Dans ce tableau, r_1 représente le coefficient de corrélation des couples associés à des sous-espaces ayant exactement 1 capteur en commun (par exemple (Y_1, Y_6) dont les sous-espaces associés ont le capteur C_1 en commun) et r_2 représente le coefficient de corrélation des couples associés à des sous-espaces ayant exactement 2 capteurs en commun.

Plus généralement, le nombre de capteurs en commun pouvant être compris entre 0 et $d-1$, on aura au maximum d coefficients de corrélation à déterminer si on est dans ce contexte (m capteurs identiques et opérationnels).

	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	Y_9	Y_{10}
Y_1		r_2	r_2	r_2	r_2	r_1	r_2	r_2	r_1	r_1
Y_2			r_2	r_2	r_1	r_2	r_2	r_1	r_2	r_1
Y_3				r_1	r_2	r_2	r_1	r_2	r_2	r_1
Y_4					r_2	r_2	r_2	r_1	r_1	r_2
Y_5						r_2	r_1	r_2	r_1	r_2
Y_6							r_1	r_1	r_2	r_2
Y_7								r_2	r_2	r_2
Y_8									r_2	r_2
Y_9										r_2
Y_{10}										

Tableau V.2 – Tableau simplifié des corrélations

V.1.3 Classement des réalisations du vecteur décisionnel

Reprenons maintenant la variable aléatoire représentant la somme des variables décisionnelles $W = \sum_{\ell=1}^{N} Y_\ell$ et dont nous avons exprimé la loi de probabilité en (IV.27) page 79. Nous avons vu qu'une réalisation w de W est donnée par toutes les réalisations $\boldsymbol{y}$ de $\boldsymbol{Y}$ telles que $\|\boldsymbol{y}\| = w$

On peut alors redéfinir l'ensemble fondamental des décisions $\Omega_{\boldsymbol{Y}}$ associé initialement à $\boldsymbol{Y}$ par l'ensemble complet des réalisations $\Omega_W = \{0, 1, \ldots, N\}$ associé à W. Ce principe ainsi que les notations qui en découlent sont illustrés par un exemple simple dans le tableau V.3 où la colonne $nb(w)$ représente le nombre de réalisations dans $\boldsymbol{y} \in \Omega_{\boldsymbol{Y}}$ qui vérifient $\|\boldsymbol{y}\| = w$.

Toujours sous l'hypothèse de travail que les capteurs sont identiques et opérationnels, on peut admettre que toutes les réalisations $\boldsymbol{y}$ telles que $\|\boldsymbol{y}\| = w$ pèsent

de manière identique dans la décision globale et on aura donc :

$$P\left[\boldsymbol{Y}=\boldsymbol{y} \mid \|\boldsymbol{y}\|=w\right]=p_{\boldsymbol{Y}}(w)$$

$$\Longrightarrow \quad p_W(w)=\sum_{\boldsymbol{y} \mid \|\boldsymbol{y}\|=w} P\left[\boldsymbol{Y}=\boldsymbol{y}\right]=nb(w)\times p_{\boldsymbol{Y}}(w) \tag{V.1}$$

$\Omega_{\boldsymbol{Y}}$	y_1	y_2	y_3	$nb(w)$	Ω_W (ensemble des valeurs de w)	$p_W(w)=P\left[W=w\right]$
$\boldsymbol{y}_1$	0	0	0	1	0	$p_W(0)$
$\boldsymbol{y}_2$	0	0	1	3	1	$p_W(1)$
$\boldsymbol{y}_3$	0	1	0			
$\boldsymbol{y}_4$	1	0	0			
$\boldsymbol{y}_5$	0	1	1	3	2	$p_W(2)$
$\boldsymbol{y}_6$	1	0	1			
$\boldsymbol{y}_7$	1	1	0			
$\boldsymbol{y}_8$	1	1	1	1	3	$p_W(3)$

Tableau V.3 – Représentation de $\Omega_{\boldsymbol{Y}}$ et de Ω_W dans le cas où $m=3$ capteurs et $N=3$ sous-espaces de dimension $d=2$

V.1.4 Construction du nouveau modèle

En réinjectant l'équation V.1 dans la formulation du problème initial du maximum d'entropie (Cf. équations IV.34 et IV.35 page 81), on peut alors déduire le nouveau modèle donné par,

$$\begin{aligned}\text{Maximiser} \qquad & -\sum_{k=1}^{2^N} p(\boldsymbol{y}_k)\log\left(p(\boldsymbol{y}_k)\right) \\ & -\sum_{w=0}^{N} nb(w)\times p_{\boldsymbol{Y}}(w)\log\left(p_{\boldsymbol{Y}}(w)\right) \\ = \quad & -\sum_{w=0}^{N} p_W(w)\log\left(\frac{p_W(w)}{nb(w)}\right)\end{aligned} \tag{V.2}$$

$$\text{Sous les contraintes} \quad \left| \begin{array}{l} \sum_{w=0}^{N} p_W(w)=1 \\ \sum_{w=0}^{N} A'_{sw}p_W(w)=b'_s \quad s=1,\ldots,S' \end{array} \right. \tag{V.3}$$

où S' est le nombre de contraintes définies par l'utilisateur.

Par rapport à la construction de la matrice $\boldsymbol{A}$ et du vecteur $\boldsymbol{b}$ du problème initial (Cf. équation IV.39 page 83), la matrice $\boldsymbol{A}'$ et le vecteur $\boldsymbol{b}'$ du nouveau modèle supportent quelques changements :

- Concernant le vecteur $\boldsymbol{b}'$: le vecteur original $\boldsymbol{b}$ est constitué des N probabilités marginales p_i et des C_N^2 moments d'ordre 2

$$E\left[Y_i Y_j\right] = p_i p_j + r_{ij}\sqrt{p_i p_j (1-p_i)(1-p_j)}$$

  Sous l'hypothèse de travail que nous avons émise, on a alors $p_i = p$ $(\forall i = 1, \ldots, N)$ et nous avons au plus d coefficients de corrélation (Cf. tableau V.2 page 93). Par conséquent, la dimension du vecteur $\boldsymbol{b}'$ est réduite par rapport à celle de $\boldsymbol{b}$ (i.e. $S' < S$).
- Concernant la matrice $\boldsymbol{A}'$: on peut alors combiner les termes de $\boldsymbol{A}$ qui correspondent à la même réalisation w de W pour construire $\boldsymbol{A}'$ (Cf. construction de $\boldsymbol{A}$ à la section IV.3.2.2 page 82).

V.1.5 Exemple simple

Le contexte est celui du problème de décision présenté à la section IV.2.4.2 page 74 avec les paramètres suivants :

- $m = 3$ capteurs et $N = C_m^d = 3$ sous-espaces (classifieurs) différents de dimension $d = 2$;
- les sorties $X_i\,(i = 1, \ldots, m)$ des capteurs suivent une distribution gaussienne de même variance $\sigma^2 = 1$ sous les deux hypothèses H_0 et H_1 et des moyennes déterminées en fonction de α et β ;
- un ensemble d'apprentissage constitué de $N_{app} = 100000$ réalisations de $X_1, X_2, \ldots, X_m$ sous chaque hypothèse afin d'estimer les corrélations r_c ;
- $\alpha = 0.15$ et $\beta = 0.05$.

Pour comparaison nous rappelons aussi les résultats de la section IV.4.2 pour les lois conjointes des réalisations obtenues respectivement par le modèle réel, le modèle de Bahadur-Lazarsfeld et le modèle du maximum d'entropie (Cf. tableau V.4). Les probabilités de chaque classe $\hat{p}_W(w)\,(w = 0, \ldots, N)$ sont estimées à l'aide de la nouvelle formulation donnée en (V.3) et présentées avec les probabilités vraies $p_W(w)$ calculées avec le modèle réel.

Les résultats présentés dans le tableau V.5 montrent une très bonne adéquation des solutions du nouveau modèle simplifié avec le modèle réel. Ces résultats reposent cependant sur un cas simple et il reste à vérifier ce nouveau modèle sur des situations plus complexes. Par exemple avec un grand nombre de capteurs. Dans la section suivante nous détaillons une perspective pour étendre ces résultats au cas où certains capteurs sont en panne.

w	$\boldsymbol{Y}$			Sous H_0			Sous H_1		
	y_1	y_2	y_3	$P(y_1,y_2,y_3)$	$\hat{P}_{BL}$	$\hat{P}_{ME}$	$P(y_1,y_2,y_3)$	$\hat{P}_{BL}$	$\hat{P}_{ME}$
$w=0$	0	0	0	0.6923463	0.692665	0.690427	0.004957	0.00499	0.005884
$w=1$	0	0	1	0.06535	0.065541	0.067818	0.007212	0.007047	0.006291
	0	1	0	0.06535	0.065027	0.066581	0.007212	0.007063	0.006274
	1	0	0	0.06535	0.065387	0.068083	0.007212	0.007676	0.005565
$w=2$	0	1	1	0.026953	0.026769	0.025174	0.0306	0.030893	0.031550
	1	0	1	0.026953	0.026409	0.023672	0.0306	0.030281	0.032259
	1	1	0	0.026953	0.026923	0.024908	0.0306	0.030265	0.032276
$w=3$	1	1	1	0.030742	0.031280	0.033336	0.88161	0.881778	0.879899

Tableau V.4 – Lois conjointes des réalisations du vecteur décisionnel obtenues par trois modèles

w	Sous H_0		Sous H_1	
	$p_W(w)$	$\hat{p}_W(w)$	$p_W(w)$	$\hat{p}_W(w)$
$w=0$	0.6923463	0.6900	0.004957	0.0061
$w=1$	0.19605	0.2031	0.021636	0.0182
$w=2$	0.080859	0.0738	0.0918	0.0952
$w=3$	0.030742	0.0331	0.88161	0.8805

Tableau V.5 – Lois conjointes obtenues par le modèle réel et le nouveau modèle du maximum d'entropie

V.2 Réglage d'un classifieur d'ensemble

Cet aspect concerne la préoccupation initiale de ce travail. Il s'agit de maîtriser le lien qui existe entre le choix des couples de paramètres $(\alpha_\ell, \beta_\ell)$ des classifieurs individuels et ceux du classifieur global.

Concrètement, cela consiste, à partir des courbes COR des classifieurs individuels et d'un objectif de performance globale, à trouver le meilleur réglage de chaque classifieur de sorte à atteindre la performance globale souhaitée.

Nous disposons d'un moyen d'estimer la distribution conjointe du vecteur décisionnel pour un ensemble de couples $(\alpha_\ell, \beta_\ell)$ donné. On peut donc envisager d'utiliser ce modèle pour établir le lien avec le couple (α_G, β_G) de paramètres du classifieur global.

Il s'agit d'un problème d'optimisation qui reste complexe à traiter aussi bien dans sa formulation que dans sa mise en œuvre. Un des problèmes à règler réside dans la dépendance entre la distribution conjointe et les couples $(\alpha_\ell, \beta_\ell)$ qui nécessitent la ré-estimation des distributions marginales et des corrélations.

Dans cette perspective on peut envisager de tenir compte des pannes de capteurs pour le calcul de (α_G, β_G). La panne peut-être intégrée de deux façons différentes :

- on connaît les capteurs qui sont en panne. Dans ce cas, on connaît les composantes Y_k du vecteur décisionnel qui sont impactées, ce qui peut être pris en compte lors de l'estimation de (α_G, β_G) en les éliminant.
- on ne connaît pas les capteurs qui sont en panne. Dans ce cas, cela revient à modifier $(\alpha_\ell, \beta_\ell)$ sur certaines composantes du vecteur décisionnel. Dans les situations où les capteurs sont identiques, on peut choisir arbitrairement un scénario de panne pour construire la distribution conjointe du vecteur décisionnel.

Dans le même ordre d'idées on devrait pouvoir tenir compte de groupes de capteurs ayant des courbes COR différentes. Par exemple en choisissant un ensemble de classifieurs définis dans des sous-espaces de dimensions différentes.

Chapitre VI

Conclusions

Dans cette thèse, nous avons présenté nos travaux de recherche sur le problème de la surveillance ou du diagnostic d'un système complexe, que nous avons considéré comme un problème de classification. Notre objectif principal était de proposer des méthodes robustes de classification visant à préserver ou améliorer les performances du système décisionnel dans un environnement non-stationnaire où des changements ou des perturbations surviennent à un moment donné sur une partie des mesures du système. Dans une telle situation, l'espace de représentation initial ne peut plus représenter correctement les données, ce qui a pour conséquence une dégradation de la performance globale du système décisionnel. L'efficacité des méthodes d'ensemble dans beaucoup d'applications nous a incités, dans le cadre de nos objectifs, à nous orienter vers la sélection de sous-espaces de représentation. Cette option permet de disposer de plusieurs sous-espaces non-altérés dans lesquels les classifieurs de l'ensemble prennent des décisions conformément à l'apprentissage qui a permis de les mettre au point. La décision finale est alors prise en fusionnant les décisions des classifieurs individuels de l'ensemble. Deux facteurs cruciaux interviennent dans l'optimisation de la règle de décision finale : la construction d'un ensemble adéquat de classifieurs et la conception d'une règle de combinaison. Nos travaux se sont donc développés autour de ces deux facteurs.

Dans un premier temps, nous avons dressé un état de l'art varié et précis, bien que non-exhaustif, sur les méthodes de classification. Il nous a éclairé sur les directions prises actuellement dans ce domaine. Les méthodes existantes sont présentées en les classant schématiquement en deux branches : les méthodes individuelles (arbres de décision, réseaux de neurones, SVM etc.) et les méthodes d'ensemble (Bagging, Boosting et RSM). Pour chaque type de méthode, nous avons clairement analysé ses avantages et faiblesses. Toutefois, de nombreuses études des différentes références comfirment que la performance des méthodes d'ensemble est généralement meilleure que celle d'un seul classifieur. Ceci nous a donc amenés à choisir un système d'ensemble comme support de nos travaux. Une telle méthode permet de réduire la variance des erreurs de classification des différents classifieurs élémentaires, ce qui a pour conséquence de stabiliser la décision finale en présence de changements.

Ensuite, dans le chapitre III nous avons proposé une méthode robuste de classification par la définition d'un ensemble de sous-espaces de représentation homogènes visant à préserver ou améliorer la performance d'un système décisionnel en présence de perturbations dans l'espace de représentation initial. Nous avons basé nos méthodes sur le fait qu'il est possible de sélectionner un ou plusieurs sous-espaces de dimension inférieure à celle de l'espace initial qui soient homogènes. Notre hypothèse est qu'un système décisionnel est plus performant lorsqu'il repose sur des informations homogènes.

La méthode RSM (en anglais *Random Subspace Method*), qui revient à sélectionner aléatoirement des attributs parmi tous ceux disponibles initialement, est une technique efficace pour générer un tel ensemble de sous-espaces de représentation. Reposant sur chaque sous-espace, un ensemble de classifieurs est élaboré selon trois algorithmes d'apprentissage différents : OSVM, KPCA et KECA. Pour les deux algorithmes KPCA et KECA, la classification a été opérée sur la base de l'erreur de reconstruction. La décision finale est alors prise par le vote majoritaire. Les méthodes proposées ont été testées dans le cadre de la segmentation d'images texturées qui offre un support tout à fait adéquat pour illustrer la diversité des situations de perturbations ou de non-stationnarité de l'environnement. Les résultats expérimentaux ont montré que cette méthode est efficace et robuste en termes de taux d'erreur dans les zones frontières (là où l'environnement est perturbé ou non-stationnaire). Cette étude montre également la difficulté de faire des choix de conception efficace du classifieur d'ensemble : choix de la méthode de classification, choix des sous-espaces, nombre de sous-espaces, paramètres des classifieurs individuels etc.

Dans le chapitre IV, basé également sur des sous-espaces de représentation, nous avons d'abord développé une nouvelle règle de combinaison reposant sur le test du rapport de vraisemblance pour un ensemble de classifieurs indépendants. En comparaison du vote majoritaire, cette règle de combinaison prend en compte les performances individuelles de chaque classifieur (les probabilités d'erreur de première et de seconde espèce), ce qui a pour conséquence de maîtriser finement la performance globale du système décisionnel. Les résultats des simulations sur des données artificielles montrent que cette méthode présente une amélioration intéressante de la performance du système en cas de graves perturbations lorsque la grande majorité des capteurs du système tombent en panne. Les deux modèles conjoints (le modèle de Bahadur-Lazarsfeld et du maximum d'entropie) ont ensuite été étudiés afin d'étendre le problème au cas où les classifieurs de l'ensemble ne sont pas indépendants. Ils permettent non seulement d'estimer correctement les lois conjointes des variables décisionnelles corrélées, mais aussi d'établir un lien entre les performances individuelles des classifieurs et la performance globale de la règle de décision. Cette propriété devrait permettre de mesurer l'impact des changements de comportement, issus du dysfonctionnement de certains capteurs sur la performance globale.

Enfin, le chapitre de perspectives a été consacré aux extensions des travaux du chapitre IV. La méthode initiale du maximum d'entropie est simplifiée en restructurant l'ensemble des réalisations du vecteur décisionnel. Cette simplification a pour but de pouvoir améliorer l'efficacité de calcul dans le cas d'un grand nombre de classifieurs. Les résultats d'un exemple simple montrent une très bonne adéquation des approximations du nouveau modèle simplifié avec le modèle réel. D'autre part, nous avons précisé quelques pistes permettant d'exploiter la modélisation du vecteur décisionnel dans le but de régler le classifieur global en fonction des caractéristiques des classifieurs individuels.

Liste des publications de l'auteur

Conférences internationales avec comité de lecture

1. BEAUSEROY, P., SMOLARZ, A., DONG, Y. et HE, X. « Dynamic decision method based on contextual selection of representation subspaces ». dans *Proceedings of the Ninth International Conference on Machine Learning and Applications, ICMLA 2010*, pages 567 - 572. Washington, DC, USA. **2010.**
2. DONG, Y., BEAUSEROY, P. et SMOLARZ, A. « A Robust classification method using combined classifiers in a nonstationary environment ». dans *Proceedings of the 20th European Signal Processing Conference, EUSIPCO 2012*, pages 679 - 683. Bucharest. **2012.**

Conférences nationales avec comité de lecture

1. SMOLARZ, A., BEAUSEROY, P. et DONG, Y. « Sélection d'espaces de représentation et diagnostic contextuel ». dans *Proceedings of 22ème Colloque GRETSI*. Bordeaux, France. **2011.**
2. BEAUSEROY, P., SMOLARZ, A. et DONG, Y. « Classification robuste via la sélection d'un ensemble de sous-espaces de représentation ». dans *Proceedings of 44èmes Journées de Statistique*. Bruxelles, Belgique. **2012.**
3. SMOLARZ, A., BEAUSEROY, P. et DONG, Y. « Estimation des performances de classifieurs d'ensembles à partir des propriétés des classifieurs individuels ». dans *Proceedings of 24ème Colloque GRETSI*. Brest, France. **2013.**
4. SMOLARZ, A., BEAUSEROY, P. et DONG, Y. « Diagnostic par fusion de décisions binaires corrélées ». dans *Proceedings of 45èmes Journées de Statistique*. Toulouse, France. **2013.**

Bibliographie

M.A. Aizerman, E.M. Braverman, et L.I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25 :821–837, 1964.

H. Almuallim et T.G. Dietterich. Learning with many irrelevant features. In *Proceedings of the 9th National Conference on Artificial Intelligence*, pages 547–552, 1991.

P.M.E. Altham. Two generalizations of the binomial distribution. *Applied Statistics*, 27(2) :162–167, 1978.

J.A. Anderson. Neural models with cognitive implications. *Basic Processes in Reading : Perception and Comprehension*, pages 27–90, 1977.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3) :337–404, 1950.

P. Auer et M. Warmuth. Tracking the best disjunction. *Machine Learning*, 32(2) : 127–150, 1998.

R.R. Bahadur. A representation of the joint distribution of responses to n dichotomous items. *Studies in Item Analysis and Prediction*, pages 158–168, 1961.

E. Bauer et R. Kohavi. An empirical comparison of voting classification algorithms : bagging, boosting, and variants. *Machine Learning*, 36(1) :105–139, 1999.

P. Beauseroy. Caractérisation et surveillance : de la mesure à la décision avec contraintes. *Mémoire de HDR, Université de Technologie de Troyes*, 2007.

P. Beauseroy et A. Smolarz. Optimisation de la géométrie du voisinage pour la segmentation d'images texturées. In *Proceedings of XXXVIèmes Journées de Statistique, Montpellier, France*, pages 24–28, 2004.

M. Ben-Bassat. Use of distance measures, information measures and error bounds in feature evaluation. *Handbook of Statistics : Classification, Pattern Recognition and Reduction of Dimensionality*, 2 :773–791, 1982.

K.P. Bennett et O.L. Mangasarian. Multicategory discrimination via linear programming. *Optimization Methods and Software*, 3(1-3) :27–39, 1994.

J.C. Bezdek, S. Boggavarapu, L.O. Hall, et A. Bensaid. Genetic algorithm guided clustering. In *Proceedings of 1st IEEE Conference on Evolutionary Computation*, pages 34–39, 1994.

C.M. Bishop. *Pattern recognition and machine learning (Information science and statistics)*. Springer-Verlag New York, 2006.

A. Blum. Empirical support for winnow and weighted-majority algorithms : Results on a calendar scheduling domain. *Machine Learning*, 26(1) :5–23, 1997.

A.L. Blum et P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1) :245–271, 1997.

B.E. Boser, I.M. Guyon, et V.N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, pages 144–152, 1992.

M.L. Braun, J.M. Buhmann, et K.R. Müller. On relevant dimensions in kernel feature spaces. *The Journal of Machine Learning Research*, 9(8) :1875–1908, 2008.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2) :123–140, 1996a.

L. Breiman. Heuristics of instability and stabilization in model selection. *The Annals of Statistics*, 24(6) :2350–2383, 1996b.

L. Breiman. Stacked regressions. *Machine learning*, 24(1) :49–64, 1996c.

L. Breiman. Bias, variance, and arcing classifier. *Rapport Technique 460, Statistics Department, University of California, Berkeley, CA*, 1997.

L. Breiman, J.H. Friedman, C.J. Stone, et R.A. Olshen. *Classification and regression trees*. Wadsworth International Group, 1984.

H. Brighton et C. Mellish. Advances in instance selection for instance-based learning algorithms. *Data Mining and Knowledge Discovery*, 6(2) :153–172, 2002.

P. Brodatz. *Textures : A photographic album for artists and designers*. Dover Publications, New York, 1966.

I. Buciu, C. Kotropoulos, et I. Pitas. Demonstrating the stability of support vector machines for classification. *Signal Processing*, 86(9) :2364–2380, 2006.

Christopher J. C. Burges. Geometry and invariance in kernel based methods. In *Advances in Kernel Methods : Support Vector Learning*, pages 89–116, 1999.

L.S. Camargo et T. Yoneyama. Specification of training sets and the number of hidden neurons for multilayer perceptrons. *Neural computation*, 13(12) :2673–2680, 2001.

C. Cardie. Using decision trees to improve case-based learning. In *Proceedings of the 10th International Conference on Machine Learning*, pages 25–32, 1993.

R.A. Caruana et D. Freitag. Greedy attribute selection. In *Proceedings of the 11th International Conference on Machine Learning*, pages 28–36, 1994.

G. Castellano, A. Fanelli, et M. Pelillo. An iterative pruning algorithm for feedforward neural networks. *IEEE Transactions on Neural Networks*, 8(3) :519–531, 1997.

A. Celisse et T. Mary-Huard. Exact cross-validation for knn : application to passive and active learning in classification. *Journal de la SFDS*, 152(3) :83–97, 2011.

Z. Chair et P.K. Varshney. Optimal data fusion in multiple sensor detection systems. *IEEE Transactions on Aerospace and Electronic Systems*, 22(1) :98–101, 1986.

C.C. Chang et C.J. Lin. Training v-support vector regression : theory and algorithms. *Neural Computation*, 14(8) :1959–1977, 2002.

P. Chaovalit et L. Zhou. Movie review mining : A comparison between supervised and unsupervised classification approaches. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, pages 112–120, 2005.

L.H.Y. Chen. Poisson approximation for dependent trials. *The Annals of Probability*, 3(3) :534–545, 1975.

J. Cheng, R. Greiner, J. Kelly, D. Bell, et W. Liu. Learning bayesian networks from data : An information-theory based approach. *Artificial Intelligence*, 137(1-2) :43–90, 2002.

D.M. Chickering. Learning bayesian networks is np-complete. *Learning from Data*, pages 121–130, 1995.

C. Chow et C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3) :462–467, 1968.

W.W. Cohen. Efficient pruning methods for separate-and-conquer rule learning systems. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 988–994, 1993.

T.H. Cormen, C.E. Leiserson, et R.L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

C. Cortes et V. Vapnik. Support-vector networks. *Machine learning*, 20(3) :273–297, 1995.

R. Courant et D. Hilbert. *Methods of mathematical physics*. Interscience Publishers, 1954.

T. Cover et P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1) :21–27, 1967.

G.R. Cross et A.K. Jain. Markov Random Field Texture Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(1) :25–39, 1983.

P. Cunningham et J. Carney. Diversity versus quality in classification ensembles based on feature selection. In *Machine Learning : ECML 2000*, pages 109–116, 2000.

S.P. Curram et J. Mingers. Neural networks, decision tree induction and discriminant analysis : An empirical comparison. *Journal of the Operational Research Society*, 45 (4) :440–450, 1994.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4) :303–314, 1989.

M. Dash et H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1 (1-4) :131–156, 1997.

I. De Falco, E. Tarantino, A.D. Cioppa, et F. Fontanella. An innovative approach to genetic programming-based clustering. In *Proceedings of the 9th Online World Conference on Soft Computing in Industrial Applications*, volume 34, pages 55–64, 2004.

M. Denuit, C. Lefèvre, et S. Utev. Measuring the impact of dependence between claims occurrences. *Insurance : Mathematics and Economics*, 30(1) :1–19, 2002.

P.A. Devijver et J. Kittler. *Pattern recognition : A statistical approach*. Prentice/Hall International, 1982.

K.I. Diamantaras et S.Y. Kung. *Principal component neural networks : Theory and applications*. John Wiley & Sons, Inc., 1996.

T.G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees : Bagging, boosting, and randomization. *Machine learning*, 40(2) :139–157, 2000.

J. Doak. An evaluation of feature selection methods and their application to computer security. *Tech. Rept. CSE-92-18, Department of Computer Science, University of California*, 1992.

R.O Duda et P.E. Hart. *Pattern classification and scene analysis*. Wiley, New York, 1973.

F. Esposito, D. Malerba, et G. Semeraro. A further study of pruning methods in decision tree induction. In *Proceedings of the 5th International Workshop on Artificial Intelligence and Statistics*, pages 211–218, 1995.

U. Fayyad, G. Piatetsky-Shapiro, et P. Smyth. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11) :27–34, 1996.

J.A. Feldman et D.H. Ballard. Connectionist models and their properties. *Cognitive Science*, 6(3) :205–254, 1982.

R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Human Genetics*, 7(2) :179–188, 1936.

J. Fleiss. *Statistical methods for rates and proportions*. John Wiley & Sons, Inc., 1981.

R. Fletcher. *Practical methods of optimization*. John Wiley & Sons, Inc., 1987.

Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2) :256–285, 1995.

Y. Freund. An adaptive version of the boost by majority algorithm. *Machine Learning*, 43(3) :293–318, 2001.

Y. Freund et R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1) :119–139, 1997.

Y. Freund et R. Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3) :277–296, 1999.

Y. Freund et R.E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, pages 148–156, 1996.

J.H. Friedman. A recursive partitioning decision rule for nonparametric classification. *IEEE Transactions on Computers*, 100(4) :404–408, 1977.

J.H. Friedman et J.W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, 100(9) :881–890, 1974.

K. Fukunaga. *Introduction to statistical pattern recognition (2nd edition)*. Academic Press, 1990.

S.B. Gelfand, C.S. Ravishankar, et E.J. Delp. An iterative growing and pruning algorithm for classification tree design. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 13(2) :163–174, 1991.

S. Geman et D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6) :721–741, 1984.

G. Giacinto et F. Roli. Design of effective neural network ensembles for image classification purposes. *Image and Vision Computing*, 19(9) :699–707, 2001.

M. Girolami. Orthogonal series density estimation and the kernel eigenvalue problem. *Neural Computation*, 14(3) :669–688, 2002.

M. Girolami, A. Cichocki, et S.I. Amari. A common neural network model for unsupervised exploratory data analysis and independent component analysis. *IEEE Transactions on Neural Networks*, 9(6) :1495–1501, 1998.

E. Grall-Maes et P. Beauseroy. Optimal decision rule with class-selective rejection and performance constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 :2073–2082, 2009.

C. Guerra-Salcedo et D. Whitley. Genetic approach to feature selection for ensemble creation. In *Proceedings of Genetic and Evolutionary Computation Conference*, pages 236–243, 1999.

X. Guorong, C. Peiqi, et W. Minhui. Bhattacharyya distance feature selection. In *Proceedings of the 13th International Conference on Pattern Recognition*, pages 195–199, 1996.

L. Hall, K. Bowyer, W. Kegelmeyer, T. Moore, et C. Chao. Distributed learning on very large data sets. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 79–84, 2000.

J. Han et Micheline Kamber. *Data mining : concepts and techniques.* San Francisco, CA : Morgan Kaufmann, 2001.

T.R. Hancock, T. Jiang, M. Li, et J. Tromp. Lower bounds on learning decision lists and trees. *Information and Computation*, 126(2) :114–122, 1996.

L.K. Hansen et P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10) :993–1001, 1990.

T. Hastie, R. Tibshirani, et J. Friedman. *The elements of statistical learning : Data mining, inference and prediction.* New York : Springer-Verlag, 2001.

X. He. *Sélection d'espaces de représentation pour la décision en environnement non-stationnaire. Application à la segmentation d'images texturées.* PhD thesis, Université de Technologie de Troyes, 2009.

X. He, P. Beauseroy, et A. Smolarz. Nearest neighbour ensembles in lasso feature subspaces. *IET Computer Vision*, 4(4) :306–319, 2010.

D.P. Helmbold et M.K. Warmuth. On weak learning. *Journal of Computer and System Sciences*, 50(3) :551–573, 1995.

Tin Kam Ho. Recognition of handwritten digits by combining independent learning vector quantizations. In *Proceedings of the Second International Conference on Document Analysis and Recognition*, pages 818–821, 1993.

T.K. Ho. Random decision forests. In *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, pages 278–282, 1995.

T.K. Ho. Nearest neighbors in random subspaces. In *Proceedings of the Second International Workshop on Statistical Techniques in Pattern Recognition*, pages 640–648, 1998a.

T.K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8) :832–844, 1998b.

H. Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40(3) :863–874, 2007.

J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8) :2554–2558, 1982.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2) :251–257, 1991.

K. Hornik, M. Stinchcombe, et H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5) :359–366, 1989.

W.Y. Huang et R.P. Lippmann. Comparisons between neural net and conventional classifiers. In *Proceedings of IEEE First International Conference on Neural Networks*, pages 485–493, 1987.

L. Hyafil et R.L. Rivest. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters*, 5(1) :15–17, 1976.

A.K. Jain, M.N. Murty, et P.J. Flynn. Data clustering : A review. *ACM Computing Surveys (CSUR)*, 31(3) :264–323, 1999.

N. Japkowicz et S. Stephen. The class imbalance problem : A systematic study. *Intelligent Data Analysis*, 6(5) :429–449, 2002.

E.T. Jaynes. Information theory and statistical mechanics. *Physical Review*, 106(4) : 620–630, 1957.

R. Jenssen. Kernel entropy component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5) :847–860, 2010.

T. Joachims. Text categorization with support vector machines : Learning with many relevant features. *Machine learning : ECML-98*, pages 137–142, 1998.

G.H. John. Robust linear discriminant trees. *Lecture Notes in Statistics*, pages 375–386, 1996.

G.H. John, R. Kohavi, et K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the 11th International Conference on Machine Learning*, pages 121–129, 1994.

I.T. Jolliffe. *Principal component analysis*. Springer, New York, 1986.

B. Kim et D.A. Landgrebe. Hierarchical decision tree classifiers in high-dimensional and large class data. *IEEE Transactions on Geoscience and Remote Sensing*, 29(4) : 518, 1990.

K.I. Kim, K. Jung, S.H. Park, et H.J. Kim. Support vector machines for texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 (11) :1542–1550, 2002.

K. Kira et L.A. Rendell. The feature selection problem : Traditional methods and a new algorithm. In *Proceedings of the 10th National Conference on Artificial Intelligence*, pages 129–134, 1992a.

K. Kira et L.A. Rendell. A practical approach to feature selection. In *Proceedings of the 9th International Conference on Machine Learning*, pages 249–256, 1992b.

E.M. Kleinberg. On the algorithmic implementation of stochastic discrimination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(5) :473–490, 2000.

R. Kohavi et G.H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2) :273–324, 1997.

R. Kohavi et D.H. Wolpert. Bias plus variance decomposition for zero-one loss functions. In *Proceedings of the 13th International Conference in Machine Learning*, pages 275–283, 1996.

T. Kohonen. *Associative memory : A system-theoretical approach.* Springer-Verlag Berlin, 1977.

D. Koller et M. Sahami. Toward optimal feature selection. In *Proceedings of the 13th International Conference on Machine Learning*, pages 284–292, 1996.

M.A. Kon et L. Plaskota. Information complexity of neural networks. *Neural Networks*, 13(3) :365–375, 2000.

S.B. Kotsiantis et P.E. Pintelas. Selective voting. *Breast-Cancer*, 286(9) :397–402, 2004.

S.B. Kotsiantis, I.D. Zaharakis, et P.E. Pintelas. Machine learning : A review of classification and combining techniques. *Artificial Intelligence Review*, 26(3) :159–190, 2006.

A. Krogh et J. Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems*, pages 231–238, 1995.

M. Kubat, D. Flotzinger, et G. Pfurtscheller. Discovering patterns in EEG-signals : Comparative study of a few methods. In *Proceedings 1993 European Conference on Machine Learning*, pages 366–371, 1993.

L.I. Kuncheva et C.J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2) :181–207, 2003.

LL Kupper et Joseph K Haseman. The use of a correlated binomial model for the analysis of certain toxicological experiments. *Biometrics*, 34(1) :69–76, 1978.

G.H. Landeweerd, T. Timmers, E.S. Gelsema, M. Bins, et M.R. Halie. Binary tree versus single level tree classification of white blood cells. *Pattern Recognition*, 16 (6) :571–577, 1983.

P. Langley. *Selection of relevant features in machine learning.* AAAI Press, 1994.

P. Langley et S. Sage. Oblivious decision trees and abstract cases. In *Working Notes of the AAAI-94 Workshop on Case-Based Reasoning*, pages 113–117, 1994.

P. Langley, W. Iba, et K. Thompson. An analysis of bayesian classifiers. In *Proceedings of the 10th National Conference on Artificial Intelligence*, pages 223–228, 1992.

M. LeBlanc et R. Tibshirani. Combining estimates in regression and classification. *Journal of the American Statistical Association*, 91(436) :1641–1650, 1996.

Y. Lee, Y. Lin, et G. Wahba. Multicategory support vector machines : Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465) :67–81, 2004.

E.L. Lehmann. *Testing statistical hypotheses.* New York : Wiley, 1986.

R. Lengellé. *Décision et reconnaissance des formes en signal.* Lavoisier, 2002.

R.D. Levine et M. Tribus. *The maximum entropy formalism.* Cambridge, MA : MIT Press, 1979.

D.D. Lewis. Feature selection and feature extraction for text categorization. In *Proceedings of the Workshop on Speech and Natural Language*, pages 212–217, 1992a.

D.D. Lewis. *Representation and learning in information retrieval.* PhD thesis, UM-CS-1991-093, Department of Computer Science, University of Massachusetts, Amherst, MA, 1992b.

J. Li, X. Gao, et C. Li. A GA-based clustering algorithm for large data sets with mixed numeric and categorical values. In *Proceedings of the 5th International Conference on Computational Intelligence and Multimedia Applications*, pages 102–107, 2003.

Y.K. Lin et K.S. Fu. Automatic classification of cervical cells using a binary tree classifier. *Pattern Recognition*, 16(1) :69–80, 1983.

N. Littlestone et M. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2) :212–261, 1994.

H. Liu et H. Metoda. *Instance selection and construction for data mining.* Kluwer Academic Publishers, 2001.

H. Liu et H. Motoda. *Feature extraction, construction and selection : A data mining perspective.* The Springer International Series in Engineering and Computer Science, 1998.

Y. Liu, T. Özyer, R. Alhajj, et K. Barker. Cluster validity analysis of alternative results from multi-objective optimization. In *Proceedings of the 5th SIAM International Conference on Data Mining*, pages 496–500, 2005.

Y. Liu, H. Zhang, et Y. Wu. Hard or soft classification ? Large-margin unified machines. *Journal of the American Statistical Association*, 106(493) :166–177, 2011.

W.Y. Loh et N. Vanichsetakul. Tree-structured classification via generalized discriminant analysis. *Journal of the American Statistical Association*, 83(403) :715–725, 1988.

D. Lubinsky. Algorithmic speedups in growing classification trees by using an additive split criterion. *Lecture Notes in Statistics*, pages 435–444, 1994.

R. Meir et G. Rätsch. An introduction to boosting and leveraging. *Advanced Lectures on Machine Learning*, pages 118–183, 2003.

R.S. Michalski. On the quasi-minimal solution of the general covering problem. In *Proceedings of the 5th International Symposium on Information Processing*, pages 125–128, 1969.

D. Michie, D.J. Spiegelhalter, C.C. Taylor, et J. Campbell. *Machine learning, neural and statistical classification.* Ellis Horwood London, 1994.

S. Mika, G. Rätsch, J. Weston, B. Schölkopf, et K.R. Müllers. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop*, pages 41–48, 1999a.

S. Mika, B. Schölkopf, A. Smola, K.R. Müller, M. Scholz, et G. Rätsch. Kernel PCA and de-noising in feature spaces. In *Advances in Neural Information Processing Systems*, volume 11, pages 536–542, 1999b.

S. Mika, G. Rätsch, J. Weston, B. Schölkopf, A.J. Smola, et K.R. Müller. Invariant feature extraction and classification in kernel spaces. In *Advances in Neural Information Processing Systems*, volume 12, pages 526–532, 2000.

S. Mika, G. Rätsch, et K.R. Müller. A mathematical programming approach to the kernel fisher algorithm. In *Advances in Neural Information Processing Systems*, volume 13, pages 591–597, 2001.

J. Mingers. An empirical comparison of pruning methods for decision tree induction. *Machine learning*, 4(2) :227–243, 1989.

M. Minsky et S. Papert. *Perceptrons : An introduction to computational geometry.* MIT Press, Cambridge, MA, 1969.

T. Mitchell. *Machine learning.* McGraw Hill, 1997.

S.K. Murthy, S. Kasif, et S. Salzberg. A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*, 2(1) :1–32, 1994.

G.E. Naumov. NP-completeness of problems of construction of optimal decision trees. *Soviet Physics : Doklady*, 36(4) :270–271, 1991.

C. Neocleous et C. Schizas. Artificial neural network learning : A comparative review. *Methods and Applications of Artificial Intelligence*, pages 300–313, 2002.

T. Okada et S. Tomita. An optimal orthonormal system for discriminant analysis. *Pattern Recognition*, 18(2) :139–144, 1985.

M. Oltean et L. Dioşan. An autonomous GP-based system for regression and classification problems. *Applied Soft Computing*, 9(1) :49–60, 2009.

D. Opitz. Feature selection for ensembles. In *Proceedings of the National Conference on Artificial Intelligence*, pages 379–384. JOHN WILEY & SONS LTD, 1999.

D. Opitz et R. Maclin. Popular ensemble methods : An empirical study. *Journal of Artificial Intelligence Research*, 11 :169–198, 1999.

D. Opitz et J.W. Shavlik. Actively searching for an effective neural network ensemble. *Connection Science*, 8(3-4) :337–354, 1996.

G. Pagallo et D. Haussler. Boolean feature discovery in empirical learning. *Machine learning*, 5(1) :71–99, 1990.

R. Parekh, J. Yang, et V. Honavar. Constructive neural-network learning algorithms for pattern classification. *IEEE Transactions on Neural Networks*, 11(2) :436–451, 2000.

D. Partridge et W. Krzanowski. Software diversity : Practical statistics for its measurement and exploitation. *Information and software technology*, 39(10) :707–717, 1997.

E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3) :1065–1076, 1962.

J. Pearl. *Probabilistic reasoning in intelligent systems*. Morgan Kaufmann, 1988.

B.A. Pearlmutter. *Dynamic recurrent neural networks*. School of Computer Science, Carnegie Mellon University, 1990.

W.H. Press, S.A. Teukolsky, W.T. Vetterling, et B.P. Flannery. *Numerical recipes in C : the art of scientific computing*. Cambridge University Press, UK, 1993.

J.R. Quinlan. Induction of decision trees. *Machine learning*, 1(1) :81–106, 1986.

J.R. Quinlan. Decision trees and multi-valued attributes. In *Machine Intelligence 11*, pages 305–318. Oxford University Press, 1988.

J.R. Quinlan. *C4.5 : programs for machine learning*. Morgan kaufmann, Los Altos, 1993.

T. Reinartz. A unifying view on instance selection. *Data Mining and Knowledge Discovery*, 6(2) :191–210, 2002.

A. Renyi. On measures of entropy and information. In *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability*, pages 547–561, 1961.

C. Richard, R. Lengellé, et F. Abdallah. Optimisation de critères de contraste et des performances en détection. In *Dix-huitième Colloque sur le Traitement du Signal et des Images, FRA, 2001*. GRETSI, 2001.

M.D. Richard et R.P. Lippmann. Neural network classifiers estimate bayesian a posteriori probabilities. *Neural computation*, 3(4) :461–483, 1991.

J.H. Robert et C.J. Lakhmi. *Radial basis function networks 2 : New advances in design*. Physica-Verlag Heidelberg, 2001.

F. Roli, G. Giacinto, et G. Vernazza. Methods for designing multiple classifier systems. *Multiple Classifier Systems*, pages 78–87, 2001.

F. Rosenblatt. The perceptron : A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6) :386–407, 1958.

F. Rosenblatt. *Principles of neurodynamics*. Spartan, New York, 1962.

A. Roy. On connectionism, rule extraction, and brain-like learning. *IEEE Transactions on Fuzzy Systems*, 8(2) :222–227, 2000.

G. Sahoo et Yugal Kumar. Analysis of parametric & non parametric classifiers for classification technique using WEKA. *International Journal of Information Technology and Computer Science (IJITCS)*, 4(7) :43–49, 2012.

M.P. Sampat, A.C. Bovik, J.K. Aggarwal, et K.R. Castleman. Supervised parametric and non-parametric classification of chromosome images. *Pattern Recognition*, 38 (8) :1209–1223, 2005.

G. Saon et M. Padmanabhan. Minimum bayes error feature selection for continuous speech recognition. In *Advances in Neural Information Processing Systems*, volume 13, pages 800–806, 2001.

R.E. Schapire, Y. Singer, et A. Singhal. Boosting and rocchio applied to text filtering. In *Proceedings of the 21st Annual International Conference on Research and Development in Information Retrieval*, pages 215–223. ACM, 1998.

B. Schölkopf, P. Simard, A.J. Smola, et V. Vapnik. Prior knowledge in support vector kernels. In *Advances in Neural Information Processing Systems*, pages 640–646, 1998a.

B. Schölkopf, A. Smola, et K.R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5) :1299–1319, 1998b.

B. Schölkopf, A.J. Smola, R.C. Williamson, et P.L. Bartlett. New support vector algorithms. *Neural Computation*, 12(5) :1207–1245, 2000.

B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, et R.C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7) :1443–1471, 2001.

I.K. Sethi et J.H. Yoo. Design of multicategory multifeature split decision trees using perceptron learning. *Pattern Recognition*, 27(7) :939–947, 1994.

R. Setiono et W.K. Leow. Fernn : An algorithm for fast extraction of rules from neural networks. *Applied Intelligence*, 12(1) :15–25, 2000.

M.N.H. Siddique et M.O. Tokhi. Training neural networks : backpropagation vs. genetic algorithms. In *Proceedings of IEEE International Joint Conference on Neural Networks*, pages 2673–2678, 2001.

W. Siedlecki et J. Sklansky. On automatic feature selection. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(2) :197–220, 1988.

M. Singh et G.M. Provan. Efficient learning of selective bayesian network classifiers. In *Proceedings of the 13th International Conference on Machine Learning*, pages 453–461, 1996.

D. Skalak. The sources of increased accuracy for two proposed boosting algorithms. In *Proceedings American Association for Artificial Intelligence, AAAI-96, Integrating Multiple Learned Models Workshop*, pages 120–125, 1996.

J. Sklansky et G.N. Wassel. *Pattern classifiers and trainable machines*. 1981.

A. Smolarz. Etude qualitative du modèle auto-binomial appliqué à la synthèse de texture. *XXIXèmes Journées de Statistique*, pages 712–715, 1997.

P.-N. Tan, M. Steinbach, et V. Kumar. *Introduction to data mining*. Boston, MA : Addison-Wesley/Longman, 2005.

D.M.J. Tax et R.P.W. Duin. Support vector data description. *Machine Learning*, 54 (1) :45–66, 2004.

S.C.A. Thomopoulos, R. Viswanathan, et D.C. Bougoulias. Optimal decision fusion in multiple sensor systems. *IEEE Transactions on Aerospace and Electronic Systems*, 23(5) :644–653, 1987.

S.C.A. Thomopoulos, R. Viswanathan, et D.C. Bougoulias. Optimal distributed decision fusion. *IEEE Transactions on Aerospace and Electronic Systems*, 25(5) :761–765, 1989.

K.M. Ting et T.H. Witten. Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10(1) :271–289, 1999.

K. Torkkola. Feature extraction by non-parametric mutual information maximization. *The Journal of Machine Learning Research*, 3 :1415–1438, 2003.

K. Tumer et J. Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection Science*, 8(3-4) :385–404, 1996.

M. Unser. *Description statistique de textures : Application à l'inspection automatique*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 1984.

P.E. Utgoff. Perceptron trees : A case study in hybrid concept representations. *Connection Science*, 1(4) :377–391, 1989.

G. Valentini et T.G. Dietterich. Bias-variance analysis and ensembles of SVM. In *Proceedings of the 3rd International Workshop on Multiple Classifier Systems*, pages 222–231, 2002.

G. Valentini et T.G. Dietterich. Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods. *The Journal of Machine Learning Research*, 5 :725–775, 2004.

P.A.G. Van Der Geest. The binomial distribution with dependent bernoulli trials. *Journal of Statistical Computation and Simulation*, 75(2) :141–154, 2005.

V. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, New York, 1995.

V. Vapnik. Statistical learning theory, 1998.

V. Vapnik et S. Kotz. *Estimation of dependences based on empirical data*. Springer-Verlag New York, 1982.

P.K. Varshney. *Distributed detection and data fusion*. New York : Springer-Verlag, 1996.

P.K. Varshney. Multisensor data fusion. *Electronics and Communication Engineering Journal*, 9(6) :245–253, 1997.

K. Veropoulos, C. Campbell, et N. Cristianini. Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 55–60, 1999.

R. Viswanathan et P.K. Varshney. Distributed detection with multiple sensors : Part I-fundamentals. *Proceedings of the IEEE*, 85(1) :54–63, 1997.

F. Vivarelli et C. Williams. Comparing bayesian neural network algorithms for classifying segmented outdoor images. *Neural Networks*, 14(4) :427–437, 2001.

S.M. Weiss et C.A. Kulikowski. *Computer systems that learn : classification and prediction methods from statistics, neural nets, machine learning, and expert systems*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1991.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, et V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems*, pages 668–674, 2001.

Jason Weston et Chris Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Royal Holloway, University of London, Department of Computer Science, 1998.

D. Wettschereck, D.W. Aha, et T. Mohri. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, 11(1) :273–314, 1997.

B. Widrow et M.E. Hoff. Adaptive switching circuits. *IRE WESCON Convention Record*, pages 96–104, 1960.

D.R. Wilson et T. Martinez. Reduction techniques for instance-based learning algorithms. *Machine learning*, 38(3) :257–286, 2000.

D.H. Wolpert. Stacked generalization. *Neural Networks*, 5(2) :241–259, 1992.

C. Wu, D. Landgrebe, et P. Swain. The decision tree approach to classification. Technical report, Laboratory for Applications of Remote Sensing, School of Engineering, Purdue University, 1975.

J. Yam et W. Chow. Feedforward networks training speed enhancement by optimal initialization of the synaptic coefficients. *IEEE Transactions on Neural Networks*, 12(2) :430–434, 2001.

H. Yang et J. Moody. Data visualization and feature selection : New algorithms for nongaussian data. In *Advances in Neural Information Processing Systems*, volume 12, pages 687–693, 2000.

J. Yang, A.F. Frangi, J. Yang, D. Zhang, et Z. Jin. KPCA plus LDA : A complete kernel fisher discriminant framework for feature extraction and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2) :230–244, 2005.

Y. Yang et G. Webb. On why discretization works for naive-bayes classifiers. *Advances in Artificial Intelligence*, pages 440–452, 2003.

G.G. Yen et H. Lu. Hierarchical genetic algorithm based neural network design. In *IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks*, pages 168–175, 2000.

K.C. You et K.S. Fu. An approach to the design of a linear binary tree classifier. In *LARS Symposia, Laboratory for Applications of Remote Sensing*, 1976.

C.T. Yu, C. Buckley, K. Lam, et G. Salton. A generalized term dependence model in information retrieval. Technical report, Cornell University, Ithaca, NY, USA, 1983.

E. Yu et S. Cho. GA-SVM wrapper approach for feature subset selection in keystroke dynamics identity verification. In *Proceedings of the International Joint Conference on Neural Networks*, pages 2253–2257, 2003.

L. Yu et H. Liu. Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, 5 :1205–1224, 2004.

S. Zhang, C. Zhang, et Q. Yang. Data preparation for data mining. *Applied Artificial Intelligence*, 17(5-6) :375–381, 2003.

Z. Zhou. Rule extraction : using neural networks or for neural networks ? *Journal of Computer Science and Technology*, 19(2) :249–253, 2004.

Yuan DONG

Doctorat : Optimisation et Sûreté des Systèmes

Année 2013

Modélisation probabiliste de classifieurs d'ensemble pour des problèmes à deux classes

L'objectif de cette thèse est d'améliorer ou de préserver les performances d'un système décisionnel quand l'environnement peut impacter certains attributs de l'espace de représentation à un instant donné ou en fonction de la position géographique de l'observation. S'inspirant des méthodes d'ensemble, notre approche a consisté à prendre les décisions dans des sous-espaces de représentation résultant de projections de l'espace initial, espérant ainsi travailler dans des sous-espaces non impactés. La décision finale est alors prise par fusion des décisions individuelles. Dans ce contexte, trois méthodes de classification (one-class SVM, Kernel PCA et Kernel ECA) ont été testées en segmentation d'images texturées qui constitue un support applicatif parfaitement adéquat en raison des ruptures de modèle de texture aux frontières entre deux régions. Ensuite, nous avons proposé une nouvelle règle de fusion reposant sur un test du rapport de vraisemblance pour un ensemble de classifieurs indépendants. Par rapport au vote majoritaire, cette règle de fusion a montré de meilleures performances face à l'altération de l'espace de représentation. Enfin, nous avons établi un modèle conjoint pour l'ensemble des variables décisionnelles de Bernoulli corrélées associées aux décisions des classifieurs individuels. Cette modélisation doit permettre de lier les performances des classifieurs individuels à la performance de la règle de décision globale et d'étudier et de maîtriser l'impact des changements de l'espace initial sur la performance globale.

Mots clés : apprentissage automatique - classification - fusion multicapteurs – reconnaissance des formes (informatique).

Probabilistic Modeling of Ensemble Classifiers for Two Classes Problems

The objective of this thesis is to improve or maintain the performance of a decision-making system when the environment can impact some attributes of the feature space at a given time or depending on the geographical location of the observation. Inspired by ensemble methods, our approach has been to make decisions in representation sub-spaces resulting of projections of the initial space, expecting that most of the subspaces are not impacted. The final decision is then made by fusing the individual decisions. In this context, three classification methods (one-class SVM, Kernel PCA and Kernel ECA) were tested on a textured images segmentation problem which is a perfectly adequate application support because of texture pattern changes at the border between two regions. Then, we proposed a new fusion rule based on a likelihood ratio test for a set of independent classifiers. Compared to the majority vote, this fusion rule showed better performance against the alteration of the performance space. Finally, we modeled the decision system using a joint model for all decisions based on the assumption that decisions of individual classifiers follow a correlated Bernoulli law. This model is intended to link the performance of individual classifiers to the performance of the overall decision rule and to investigate and control the impact of changes in the original space on the overall performance.

Keywords: machine learning - classification - multisensor data fusion – pattern recognition.

Thèse réalisée en partenariat entre :

CHINA SCHOLARSHIP COUNCIL

utt université de technologie Troyes

Ecole Doctorale "Sciences et Technologies"