N° d'ordre : 29 / 2010-M / EL

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumedienne
FACULTE D'ELECTRONIQUE ET D'INFORMATIQUE

Mémoire présenté pour l'obtention du diplôme de MAGISTER
en Electronique

Option : Traitement du Signal et des Images

Par

M. *ABIDINE M'hamed Billel*

SUJET

Comparaison de méthodes à noyaux pour la sélection de descripteurs :
Application au traitement de documents audio.

Soutenu publiquement le 21/06/2010 devant le jury composé de :

Mme. R.Touhami	Professeur à l'USTHB	Présidente
M. B.Fergani	Maitre de conférences/A à l'USTHB	Directeur de Mémoire
M. Y.Chibani	Professeur à l'USTHB	Examinateur
Mme. L.Falek	Maitre de conférences/A à l'USTHB	Examinatrice
M. R.Baba Ali	Maitre de conférences/A à l'USTHB	Examinateur

Remerciements

Au terme de ce travail, je remercie DIEU Le Tout Puissant qui m'a donné la force et la volonté d'achever mon travail et nous lui rendons grâce.

Ce travail a été réalisé au sein du Laboratoire de Communication Parlée et de Traitement de Signal (LCPTS) de la faculté d'électronique et d'informatique de l'Université des Sciences et de la Technologie Houari BOUMEDIENE USTHB.

Le travail de recherche, comme tout autre travail, nécessite de la motivation pour être accompli d'une manière optimale. Pendant mes années de mémoire, j'ai été motivé, et ce grâce à la confiance que m'a donné mon Directeur de Mémoire. Je tiens à exprimer ma profonde gratitude et ma sincère reconnaissance à Monsieur Belkacem Fergani qui m'a encadré durant ces années de mémoire, pour ces conseils, son aide inestimable et son optimisme contagieux.

Je remercie les membres de jury en commençant par Mme. Touhami qui m'a fait l'honneur d'accepter de présider ce jury. Aussi je tiens également à remercier les membres du jury : Mme. Falek, M. Chibani et M. Baba Ali, pour l'honneur qu'ils me font à juger ce travail.

Le travail de recherche n'exclut pas des moments difficiles pour des raisons psychologiques, administratives, techniques, et autres. Je tiens à remercier mes parents Djamila, Djaffar qui ont veillé à ce que ces moments difficiles soient les plus courts grâce à leur écoute, encouragement, affection, et aide, sans qui ce travail n'aurait jamais pu voir le jour. Je leur dédie ce travail en témoignage de ma profonde affection pour toute la patience et les sacrifices qu'ils ont consenti pour moi et dont je serai à jamais redevable, et d'avoir porté ce travail à terme représente pour moi aujourd'hui la plus belle des récompenses.

Enfin, je ne saurais terminer cette liste sans adresser un remerciement particulier à ceux qui m'ont soutenu dans l'ombre de prés ou de loin, mes amis, et toute ma famille.

TABLE DES MATIERES :

Page

LISTE DES TABLEAUX **5**

LISTE DES FIGURES **7**

ORGANISATION DU MEMOIRE **9**

INTRODUCTION GENERALE **10**

CHAPITRE I : GENERALITES SUR LES NOYAUX. **12**

I-1- Introduction 12
I-2- Généralités sur les noyaux 13
I-2-1- L'astuce du noyau 13
I-2-2- Propriétés mathématiques 14
I-3- Conclusion 17

CHAPITRE II : METHODES DE CLASSIFICATION EN RDF: ETAT DE L'ART **19**

II-1- Introduction 19
II-2- Classification automatique des signaux audio 20
II-3- Les méthodes de classification/ discriminatives. 25
II-3-1- Les *k*-plus proches voisins (*k*-ppv) 25
II-3-2- Les réseaux de neurones : Le Perceptron Multi-Couches (PMC) 27
II-3-2-1- Le Perceptron Multi-Couches (PMC) 28
II-3-3- Méthodes à vecteurs de supports bi-classes 29
II-3-3-1- Introduction 29
II-3-3-2- Classificateur linéaire 31
II-3-3-2-1- Définition 31
II-3-3-2-2- Marge de l'hyperplan 32
II-3-3-2-3- Hyperplans canoniques 33
II-3-3-2-4- Relation avec l'apprentissage statistique 34
II-3-3-2-5- Sur et sous-apprentissage (Over and Under-fitting) . . . 35
II-3-3-2-6- Classificateur à marge maximale 36
II-3-3-3- Les fonctions noyaux (*kernels*) 39
II-3-3-3-1- Généralités 39
II-3-3-3-2- Le noyau RBF (Radial Basis Function) 43
II-3-3-4- Formulation marge douce (*Soft Margin*) 45
II-3-3-5- Effet des hyper-paramètres sur le modèle SVM 47
II-4- Conclusion 49

CHAPITRE III : SELECTION DES HYPER-PARAMETRES D'UN MODELE SVM **51**

III-1- Problématique 51
III-2- Etat de l'art 51
III-2-1- La validation croisée 52

III-2-2- Critère d'Alignement de noyaux 54
III-2-3- Critère combinant borne *Rayon-Marge* et mesure de séparabilité des classes à noyau 59
III-2-3-1- Borne sur l'erreur de généralisation pour les SVM 60
III-2-3-2- Séparabilité des classes à noyau (*kernel class separability*) . . . 62
III-2-4- Choix du paramètre *C*. 64
III-3- Conclusion 65

CHAPITRE IV LA SELECTION DE DESCRIPTEURS **66**

IV-1- Problématique 66
IV-2- Etat de l'art 67
IV-3- Algorithmes de sélection des attributs (ASA) 80
IV-3-1- Algorithme CLDA(Constrained Linear Discriminant Analysis) 80
IV-3-2- Séparabilité des classes à noyau (*Kernel Class Separability*) 83
IV-3-3- Critère d'Alignement de noyaux 84
IV-3-4- Critère de corrélation. 84
IV-3-5- Algorithme basé sur le critère d'Entropie de Représentation. 87
IV-3-6- Algorithme SVM-RFE (Elimination Récursive des Attributs). 89
IV-4- Conclusion 90

CHAPITRE V EXPERIMENTATIONS ET RESULTATS . . . . . **.91**

V-1- Cadre des expérimentations 91
V-2- Matériels et logiciels. 91
V-3- Corpus utilisé 92
V-4- Les résultats des expérimentations. 94
V-4-1- Sélection de modèle. 94
V-4-2- Discussion 103
V-4-3- Sélection des descripteurs. 104
V-4-3-1- Corpus utilisé 105
V-4-3-2- Evaluation des ASA en fonction de la taille des descripteurs . . . 106
V-4-4- Discussion 117
V-5- Récapitulatif des résultats. 117

CONCLUSION ET PERSPECTIVES. . . . . **.119**
Contributions. 119
Perspectives. 121

ANNEXES. **122**
A.1 : Développement de la trace de la matrice de dispersion *inter-classe* dans l'espace à noyau. 122
A.2 : Développement de la trace de la matrice de dispersion *intra-classe* à noyau. . . . 122
B : Sélection de descripteurs sur la base de données *UCI* 123

BIBLIOGRAPHIE. **128**

LISTE DES TABLEAUX :

Page

TABLEAU I.1 Noyaux usuels pour les SVM. 15
TABLEAU V.1 Création de la base de données avec illustration des trois partirions de données. 92
TABLEAU V.2 La sélection de modèle avec la première méthode pour la partition *P1*. . . . 96
TABLEAU V.3 La sélection de modèle avec la deuxième méthode pour la partition *P1*. . . . 96
TABLEAU V.4 La sélection de modèle avec la troisième méthode pour la partition *P1*. . . . 97
TABLEAU V.5 La sélection de modèle avec la première méthode pour la partition *P2*. . . . 97
TABLEAU V.6 La sélection de modèle avec la deuxième méthode pour la partition *P2*. . . . 98
TABLEAU V.7 La sélection de modèle avec la troisième méthode pour la partition *P2*. . . . 98
TABLEAU V.8 La sélection de modèle avec la première méthode pour la partition *P3*. . . . 99
TABLEAU V.9 La sélection de modèle avec la deuxième méthode pour la partition *P3* . . . 99
TABLEAU V.10 La sélection de modèle avec la troisième méthode pour la partition *P3*. . . 100
TABLEAU V.11 Comparaison entre les trois méthodes pour les trois partitions. 100
TABLEAU V.12 Comparaison entre TCC **(Musique(M))** et TCC **(Parole(P))** pour les trois méthodes. 103
TABLEAU V.13 Création de la base de données pour la sélection de descripteurs. 105
TABLEAU V.14 Récapitulation des descripteurs utilisés. Les descripteurs sont regroupés par paquets. Au total nous obtenons 154 attributs. 106
TABLEAU V.15 La sélection de modèle pour la base de données audio avec *d=16*. 107
TABLEAU V.16 Contribution des attributs par les différentes méthodes de sélection (*d*=16) 107
TABLEAU V.17 Les algorithmes présentant des améliorations de performances par rapport à l'ensemble complet d'attributs 108
TABLEAU V.18 Complexité des ASA en termes de temps. 111
TABLEAU V.19 La sélection de modèle pour la base de données audio avec *d=154*. 111
TABLEAU V.20 Contribution des attributs par les différentes méthodes de sélection (*d*=154). 112
TABLEAU V.21 Les algorithmes présentant des améliorations de performances par rapport à l'ensemble complet d'attributs 115
TABLEAU V.22 Complexité des ASA en termes de temps. 116
TABLEAU B.1 Description des données réelles *UCI* à deux classes : nombre de variables, nombre d'instances, tailles des échantillons d'apprentissage et de test. 124
TABLEAU B.2 La sélection de modèle avec la première méthode pour la base de données *WBC*. 124
TABLEAU B.3 Contribution des attributs par les différentes méthodes de sélection (*d*=10). 125
TABLEAU B.4 Les taux de bon classement obtenus en utilisant la combinaison des attributs sélectionnés par les différentes méthodes de sélection (*d*=10). **En gras** : meilleurs TCC plus grand ou égal à celui calculé avec les attributs d'origines pour chaque ASA.. 126
TABLEAU B.5 La sélection de modèle avec la première méthode pour la base de données *Heart*. 126

TABLEAU B.6 Contribution des attributs par les différentes méthodes de sélection (*d*=13). 127
TABLEAU B.7 Les taux de bon classement obtenus en utilisant la combinaison des attributs sélectionnés par les différentes méthodes de sélection *(d*=13). 127

LISTE DES FIGURES :

Page

FIGURE I.1. Variété de Riemann correspondant à un noyau polynomial 16
FIGURE II.1. Système de classification audio. 21
FIGURE II.2. Les principales méthodes de classification, d'un côté génératives, de l'autre discriminatives. Les méthodes indiquées par une couleur jaune sont des méthodes discriminatives qui peuvent être liées à un modèle génératif. 25
FIGURE II.3. Décision par 1-ppv (cercle pointillé) et 3-ppv (cercle en trait plein) sur un ensemble d'observations appartenant à 2 classes. 26
FIGURE II.4. Architecture d'un neurone formel à *n* entrées. 28
FIGURE II.5. Architecture d'un Perceptron Multi-Couches à une couche cachée. 29
FIGURE II.6. Représentation dans IR^2 de l'hyperplan correspondant à la fonction de décision d'un classificateur linéaire. 32
FIGURE II.7. Expression de la marge pour la donnée x_i (La marge est calculée à partir du produit scalaire entre les vecteurs situés à la frontière de chaque classe et le vecteur unitaire normal de l'hyperplan séparateur) 33
FIGURE II.8. Hyperplans canoniques. 34
FIGURE II.9. Le même jeu de données classé par un hyperplan à marge maximale (à gauche) et un hyperplan quelconque ne commettant pas d'erreur sur le *training set* (à droite). La boule grisée est une donnée du *test set*. 35
FIGURE II.10. Sur et sous-apprentissage des données, avec le modèle quadratique est le plus adapté. 35
FIGURE II.11. Un mapping ϕ rendant les données *linéairement séparables*. 40
FIGURE II.12. Frontières de décision obtenues par trois SVM à noyau linéaire , polynomial et RBF. 42
FIGURE II.13. Deux données mappées dans un *feature space* de dimension infinie. 43
FIGURE II.14. La similarité des données vue en utilisant un noyau RBF à σ tendant vers 0. .44
FIGURE II.15. La distance séparant un *outlier* et l'hyperplan canonique est : $\frac{\xi_i}{\|w\|}$. 46
FIGURE II.16. Disposition des vecteurs 2D d'entrée pour les SVM. 47
(a) Fonctions "représentantes" $k(a_i,.)$ et illustration des coefficients de Lagrange α_i associes, correspondant aux Vecteurs de Support, avec $\sigma = \sigma_o$= cte / $C = 10^2$. 48
(b) Modèles SVM appris avec un noyau Gaussien 48
FIGURE II.17. Modèles SVM vectoriels : noyaux Gaussiens. 48
FIGURE III.1. Principe de la Validation Croisée (VC) avec *K*=4. 53
FIGURE III.2. Schéma de sélection de modèle basé sur le critère d'alignement de noyaux. . . 57
FIGURE III.3. Illustration de l'obtention de ERR_{MIN} et ERR_Q. 62
FIGURE IV.1. Principe de la sélection de variables. 68
FIGURE IV.2. Principe de l'extraction de caractéristiques. 68
FIGURE IV.3. Procédure générale de sélection d'un sous-ensemble de caractéristiques. 71
FIGURE IV.4. Approches "*Filtres*". 72
FIGURE IV.5. Approches " *Wrappers* ". 74

FIGURE IV.6. Analyse discriminante à une dimension (nous cherchons une droite pour laquelle les projections des points des différentes classes sont les mieux séparées, la figure de droite montre une grande séparation entre les deux classes) 81
FIGURE IV.7. Représentation des différents types de résultats de *R* entre *X* et *Y*. 87
FIGURE V.1. Interface graphique du logiciel *Audacity*. 93
FIGURE V.2. Etiquetage manuel du signal sonore en classes parole/musique. 93
FIGURE V.3. Taux de classification correcte avec le nombre d'attributs varie de 2 à 16. . . 108
FIGURE V.4. Temps d'entrainement en fonction du nombre d'attributs sélectionnés. 109
FIGURE V.5. Nombre de vecteurs supports en fonction du nombre d'attributs sélectionnés. 109
FIGURE V.6. Taux de classification correcte avec le nombre d'attributs varie de 10 à 154. . 113
FIGURE V.7. Temps d'entrainement en fonction du nombre d'attributs sélectionnés (complexité exponentielle) 114
FIGURE V.8. Nombre de vecteurs supports en fonction du nombre d'attributs sélectionnés. . 114

ORGANISATION DU MEMOIRE :

Ce manuscrit est organisé comme suit :

L'introduction générale présente deux problématiques concernant certaines questions en vue d'une tâche de classification audio, qui donne naissance à deux chapitres III et IV.

Le chapitre I introduit des généralités sur la théorie des noyaux, en commençant pas expliquer l'astuce de noyau, ensuite étudier les propriétés mathématiques d'une fonction noyau.

Le chapitre II présente les méthodes de classification de l'état de l'art, utilisées en RDF, et s'intéresse plus particulièrement au classifieur SVM (méthodes à vecteurs de support). Il présente aussi les fondements théoriques sur lesquels reposent les SVM, à savoir les notions d'hyperplan optimal et l'astuce du noyau. Un problème artificiel en deux dimensions est alors utilisé de manière à pouvoir visualiser l'effet des hyper-paramètres qui conditionnent l'apprentissage des SVM.

Le chapitre III est consacré à la sélection des hyper-paramètres de SVM pour la résolution de problèmes bi-classes. Différentes approches à noyaux ont été développées, et y sont comparées expérimentalement à une méthode classique de l'état de l'art que nous utiliserons comme méthode de référence.

Le chapitre IV consiste à développer des méthodes à noyaux pour la sélection d'attributs, puis de comparer ces méthodes avec celles de l'état de l'art.

Le chapitre V est dédié aux expérimentations réalisées et à la présentation des résultats obtenus sur une base de signaux audio réels.

Bien entendu, en conclusion de ce manuscrit, nous dressons un bilan des contributions qu'apportent ce mémoire et les perspectives de travaux futurs qui en ressortent.

INTRODUCTION GENERALE

Au cours de la dernière décennie, de multiples algorithmes pour l'analyse et la classification de données fondés sur la théorie des espaces de Hilbert à noyau reproduisant ont été développés [1]. De nos jours les méthodes à noyaux connaissent un développement croissant, elles constituent une source de progrès, tant au niveau du coût calculatoire que des performances atteintes. Cet état de fait est d'autant plus surprenant que ces méthodes conditionnent largement l'efficacité des traitements, par leur aptitude à révéler les similarités existantes au sein des données traitées.

Elles reposent sur un principe central appelé *astuce du noyau*, exploité pour la première fois avec les Méthodes à Vecteurs de Support (SVM) [2, 3], ensuite utilisé aussi pour transformer beaucoup d'algorithmes linéaires de réduction de dimensionnalité en algorithmes non linéaires. L'astuce du noyau permet en effet de conférer un caractère non-linéaire à nombre de méthodes originellement linéaires [4, 5], sans qu'il soit nécessaire de recourir à d'importants développements théoriques, sous réserve que celles-ci puissent s'exprimer uniquement en fonction de produits scalaires des observations. Dans nos travaux, on compte par exemple des généralisations non-linéaires de la mesure de séparabilité des classes [6, 7], désignées dans la littérature en anglais *Kernel-Based Class Separability*.

Ce mémoire s'inscrit dans le cadre de l'apprentissage statistique inductif, et s'intéresse essentiellement au problème de sélection d'attributs en grande dimension pour la classification supervisée binaire. Un autre volet développé dans ce mémoire a trait à la sélection des hyper-paramètres tels que le choix des paramètres de noyaux. Une attente légitime pour un modèle d'apprentissage est d'être le plus simple possible et donnant de bonnes performances de généralisation. Les critères d'évaluations reposent sur les Séparateurs à Vaste Marge (SVM).

Nous allons essayer dans ce mémoire de répondre à deux problématiques pour la résolution de problèmes biclasses :

- **La Première :** Une des principales critiques que l'on puisse faire aux SVM est que la recherche des meilleurs hyper-paramètres (constante de régularisation *C*

en général et σ pour les noyaux RBF en particulier) inhérents à la méthode peut requérir des phases de développement assez longues afin d'obtenir un système de classification efficient, dans le sens d'une erreur de généralisation minimale. Parmi les méthodes les plus utilisées pour résoudre ce problème de sélection des hyper-paramètres, demeure celle basée sur la validation croisée de type *grid search* [7, 8, 9]. Son inconvénient majeur est le coût en temps de calcul, notamment pour des données issues du traitement audio. Afin de remédier à ces critiques, nous proposons dans ce mémoire d'exploiter deux approches utilisant des critères issus de méthodes à noyaux [10], l'une reposant sur la maximisation du critère d'alignement de noyaux [11, 12, 13], et l'autre [14] basée sur la minimisation du critère combinant la borne supérieure *Rayon-Marge* avec le critère de séparabilité des classes dans l'espace à noyau.

- **La Deuxième :** Dans la plupart des problèmes de classification ou de regroupement, le nombre d'attributs qui caractérise les signaux analysés peut atteindre dans certaines applications de la reconnaissance de formes, des centaines, voire des milliers (en bioinformatique). Cette quantité d'informations gigantesque, souvent redondante, bruitée et peu corrélées avec les étiquettes des données, induit des baisses de performances, une complexité algorithmique et des charges de calcul et de stockage. Ces inconvénients handicapent la réalisation de systèmes efficients de traitement de documents audio.

L'objet de la sélection d'attributs est de contribuer à réduire cette complexité, en diminuant la dimension des données à traiter. Il s'agit alors de choisir, selon des critères objectifs, un sous ensemble "optimal" d'attributs pertinents vis-à-vis du problème considéré, à partir des attributs sélectionnés initialement. L'objectif de notre travail consiste à développer des méthodes à noyaux, l'alignement de noyaux [15] et la séparabilité des classes dans l'espace à noyau [16], pour la sélection d'attributs, puis à mener une étude comparative de leurs performances avec des méthodes récentes de l'état de l'art. Les algorithmes de sélection d'attributs (ASA) utilisés sont : CLDA [17], SVM-RFE [18, 19], et d'autres basés sur des critères d'Entropie [20, 21], et de Corrélation [21, 22, 23, 24].

Les signaux de validation utilisés sont tous issus de bases de signaux audio radiodiffusés (parole/musique). Exceptionnellement, pour comparer les méthodes de sélection d'attributs, nous avons utilisé en plus de ces signaux, les données de bioinformatique publiques de la base *UCI* accessible sur Internet [25, 26].

CHAPITRE I : GENERALITES SUR LES NOYAUX

I-1- Introduction :

Au cours de la dernière décennie, de multiples algorithmes pour l'analyse et la classification de données fondés sur la théorie des espaces de Hilbert à noyau reproduisant ont été développés [1]. Ils reposent sur un principe central appelé *coup du noyau*, exploité pour la première fois avec les SVM (§II-3-3), qui confère un caractère non-linéaire à nombre de méthodes originellement linéaires [4].

Dans ce chapitre, nous présentons le concept de noyaux, à la base d'un groupe de méthodes appelées "méthodes à noyau" *(kernel methods)*, parmi lesquelles on compte les SVM. Nous donnons les éléments essentiels qui permettent de construire et choisir un noyau pour un problème de classification. Nous nous intéressons particulièrement à la classification de données numériques (c'est-à-dire de vecteurs).

De manière sommaire, l'astuce du noyau permet d'appliquer une grande famille d'algorithmes à tout type de données (numériques, symboliques, structurées, etc.) en tenant compte de relations complexes entre les caractéristiques mesurées. Pour la classification de données numériques, il s'agit par exemple d'exploiter les corrélations non linéaires entre les composantes d'entrée pour mieux capturer leur pouvoir discriminant. Notons que l'astuce du noyau ne sert pas seulement aux problèmes de classification ou de régression, comme c'est le cas avec les SVM. Elle peut s'appliquer dans le cadre général de l'analyse de données et permet par exemple d'analyser des corrélations non-linéaires, et les dépendances entre caractéristiques d'entrée [27]. Ainsi ont été proposées avec l'astuce du noyau des généralisations de l'Analyse en Composantes Principales [28], l'Analyse Discriminantes de Fisher [29] et de l'Analyse en Composantes Indépendantes [30], respectivement désignées dans la littérature par kernel-PCA, kernel-FDA et kernel-ICA.

Dans ce chapitre, le concept de noyau est d'abord introduit avec les principales notions mathématiques auxquelles il renvoie de manière générale (§I-2). Nous donnons ensuite des exemples de fonctions noyaux qui s'appliquent aux vecteurs de dimension fixe.

I-2- Généralités sur les noyaux :

I-2-1- L'astuce du noyau :

En pratique, l'*astuce du noyau* consiste à réécrire un algorithme où toutes les relations entre données d'entrée peuvent s'écrire sous forme de produits scalaires, en remplaçant ce produit scalaire par une fonction scalaire de deux variables ("noyau"). Ainsi la structure des algorithmes demeure-t-elle inchangée et le surcoût calculatoire dû à l'évaluation des noyaux négligeable. L'astuce du noyau permet ainsi de généraliser un algorithme linéaire manipulant des vecteurs :

1. pour traiter les vecteurs de façon non linéaire (parce que les données présentent des non-linéarités qu'il est utile d'exploiter pour le problème visé);

Ou

2. pour manipuler d'autres types d'objets que les vecteurs (ex : objets symboliques comme les chaînes de caractères ou encore des objets structurés comme les graphes).

D'un point de vue qualitatif, le noyau peut être vu comme une *mesure de similarité*, qui permet de comparer deux objets d'un même type. Pour appliquer les méthodes à noyau sur un ensemble de données, il suffit en pratique de connaître les valeurs de noyaux pour tous les couples de cet ensemble. Par exemple, pour dérouler l'algorithme d'apprentissage SVM, il suffit de connaître les valeurs de noyaux estimées sur le corpus d'apprentissage. Ces valeurs sont habituellement mémorisées dans une matrice carrée : la "matrice de Gram".

Définition I.1 (Matrice de Gram) :

Soient une fonction noyau $k : X \times X \rightarrow IR$ et un ensemble de données $A = \{a_i\}_{i=1...N}$ de taille N.

La "matrice de Gram" $\mathbf{K}_A$ est définie par la matrice carrée $N \times N$ contenant les valeurs du noyau sur les couples :

$$(\mathbf{K}_A)_{i,j} = k(a_i, a_j)$$

Cette matrice contient toutes les similarités entre toutes les données.

Dans toute la suite, nous notons k le noyau et X l'espace d'entrée.

I-2-2- Propriétés mathématiques :

Tout comme le produit scalaire, on peut attendre de la fonction noyau qu'elle soit définie positive (définition I.2).

Définition I.2 ((Semi-)défini-positivité) :

Une fonction scalaire $k: X \times X \to IR$ est "semi- définie-positive" si et seulement si $\forall\ (\psi : X \to R) \neq 0, \qquad \iint_{X^2} k(x,y)\psi(x)\psi(y)dx\,dy \geq 0$

Elle est "définie-positive" (propriété plus forte) *ssi* elle vérifie la même propriété avec une inégalité stricte ('>' au lieu des '≥').

Une matrice carrée **K** de taille $N \times N$ est "semi-définie-positive" (resp. "définie-positive") *ssi* pour tout vecteur colonne $\psi \in R^N$ non nul, $\psi^T \mathbf{K} \psi \geq 0$ (resp. $\psi^T \mathbf{K} \psi > 0$).

Ce qui revient à exiger que toutes les valeurs propres de **K** soient positives.

Notons qu'une fonction $X \times X \to IR$ générant une matrice définie positive possède les trois propriétés fondamentales du produit scalaire : $\forall x_i, x_j \in X$

1. Positivité : $k(x_i, x_i) \geq 0$
2. Symétrie : $k(x_i, x_j) = k(x_j, x_i)$
3. Inégalité de *Cauchy-Shwartz* : $\left|k(x_i, x_j)\right| \leq \left\|x_i\right\| \left\|x_j\right\|$

Les fonctions scalaires symétriques et définies positives, que l'on désigne souvent simplement par "noyaux", sont plus précisément des "noyaux de Mercer"[1]. Cette expression vient de ce que l'on appelle "théorème de Mercer" [31], qui est à rigoureusement parler un corollaire du théorème de Mercer.

Théorème I.1 (Théorème de Mercer) : *Si un noyau $k: X \times X \to IR$ est symétrique et semi-défini positif, alors il admet un développement de la forme*

$$\forall\ (x,y) \in X^2, \qquad k(x,y) = \sum_{u=1}^{D} \phi_u(x)\phi_u(y)$$

Où $D \leq +\infty$ et où les D fonctions scalaires $\phi_u : X \to R$ peuvent être choisies parmi une famille orthonormée. Les conditions de symétrie et de semi-défini-positivité sont nécessaires et suffisantes.

[1] On trouve aussi le terme de "*covariance kernels*"

Autrement dit, si $k: X \times X \to IR$ est défini positif, alors il peut s'exprimer comme un produit scalaire dans un espace vectoriel où sont projetées les données, qui est appelé "*Feature Space*". Inversement, si l'on définit une correspondance entre les données d'entrée et un espace vectoriel, alors le produit scalaire dans cet espace vectoriel sera un noyau défini positif. L'*expansion* correspondant à un noyau est une fonction $\phi : X \to R^D$ telle que :

$$\forall \ (x,y) \in X^2, \qquad k(x,y) = \phi(x)^T \phi(y)$$

Nous présentons dans le tableau I.1 quelques familles de fonctions noyaux usuels paramétrables qui vérifient la condition de Mercer. Il se trouve que si cette dernière est vérifiée, les noyaux fournissent à moindre coût de calcul le produit scalaire des images de deux observations x_i et x_j par une application ϕ. Une liste plus complète de noyaux de Mercer peut être consultée dans [32].

TAB I.1. Noyaux usuels pour les SVM.

Dénomination	**Expression de *k***	**Validation du théorème de Mercer**
Linéaire	$k(x,y) = (x \,.\, y)$	Oui
Polynomial	$k(x,y) = (a(x \,.\, y) + b)^d$	Oui
Gaussien	$k(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$	Oui
Sigmoïde	$k(x,y) = \tanh(a(x \,.\, y) + b)$	Selon les valeurs de (a,b)
Laplacien	$k(x,y) = \exp(-\lvert\gamma\rvert \|x-y\|)$	Oui
KMOD (Kernel Moderate Decreasing)	$k(x,y) = a\left[\exp\left(\frac{\gamma^2}{\|x-y\|^2 + \sigma^2}\right) - 1\right]$	Oui

Remarquons qu'à l'exception du noyau linéaire, tous les noyaux possèdent un paramètre.

Notons que les données d'entrée projetées dans le *Feature Space* R^D gisent sur un sous-ensemble ouvert de R^D qui est une variété de Riemann. A contrario, les éléments du *Feature Space* R^D n'admettent pas forcement de pré-image dans l'espace d'entrée [33]. La figure I.1 représente la variété correspondant à des données d'entrée bidimensionnelles et à un noyau polynomial de degré deux : $k(x,y) = (x^T y)^2$.

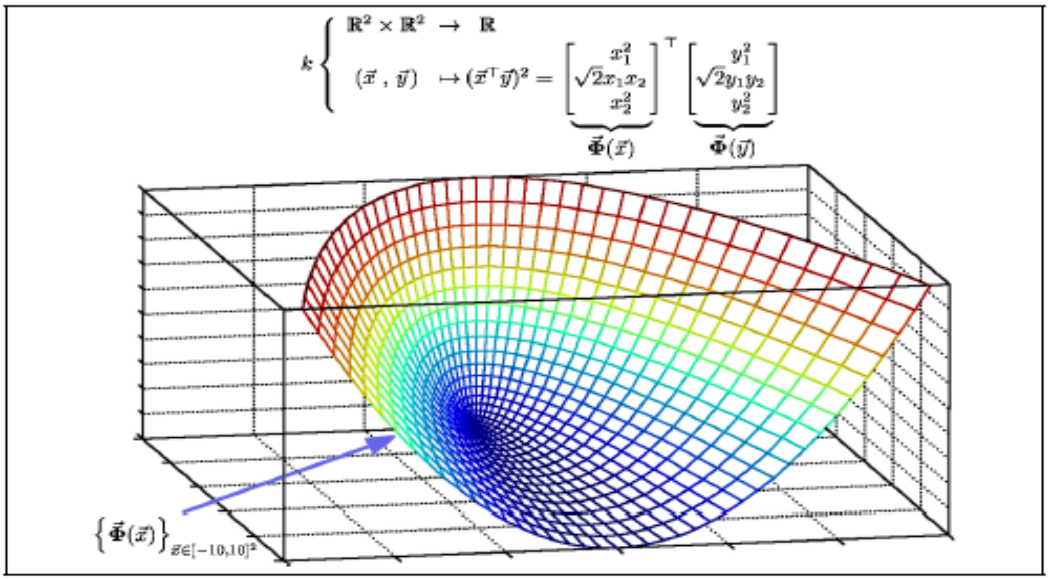


FIG I.1. Variété de Riemann correspondant à un noyau polynomial.

Pour certaines fonctions noyau symétriques et définies positives, D peut être infini et ϕ n'a pas toujours une approximation analytique connue. En fait, le théorème de Mercer est un théorème d'existence. Il ne fournit par contre aucun moyen de déduire l'expansion ϕ sous-jacente à un noyau k donné. Ainsi, il existe des noyaux pour lesquels aucune approximation analytique n'a encore été trouvée. C'est le cas du noyau Gaussien (§II-3-3-3-2), pour lequel il n'a que récemment été proposée une expression analytique de l'expansion dans le cas de données uni-dimensionnelles [34]. Cette approximation se résume au développement limité suivant :

$$\forall (x,y) \in R^2, \quad \phi^D(x)^T \phi^D(y) \overset{D\to+\infty}{\to} e^{-\frac{(x-y)^2}{2\sigma^2}}$$

$$\text{Avec } \phi^D \begin{cases} R \to R^D \\ x \to e^{\frac{-x^2}{2\sigma^2}} \left[1, \frac{x}{\sigma}, ..., \frac{x^D}{D!\sigma^D}\right]^T \end{cases}$$

Revenons maintenant à la matrice de Gram (définition I.1), notion centrale des méthodes à noyau. Il est facile de constater que les propriétés d'un noyau k se répercutent sur la matrice de Gram $\mathbf{K}_A$ quelque soit l'ensemble de données $\boldsymbol{A}$ considéré :

- Si la fonction noyau k est symétrique, alors la matrice de Gram $\mathbf{K}_A$ est symétrique.
- Si k est définie positive, alors la matrice de $\mathbf{K}_A$ est définie positive.

Inversement, si la matrice de Gram sur les données d'apprentissages est symétrique et définie positive, alors la propriété de Mercer est vérifiée *sur ces données d'apprentissage* (même si l'on a aucune garantie sur la défini-positivité du noyau) :

$$\mathbf{K}_{\boldsymbol{A}} \; sym.déf\text{-}pos. \Rightarrow \left[\exists \left\{ \phi_u \right\}_{u \in N^+} \;\; tq \;\; \forall \left(a_i, a_j \right) \in \boldsymbol{A}^2 \, , k(a_i, a_j) = \sum_{u=1}^{+\infty} \phi_u(a_i) \phi_u(a_j) \right]$$

La propriété de défini-positivité d'un noyau entraîne plusieurs propriétés qui aident à la résolution d'une grande famille d'algorithme d'optimisation, comme l'inégalité de Cauchy-Schwartz :

$$\forall \, (x, y) \in X^2, \quad k(x, y) \leq \sqrt{k(x, x) k(y, y)}$$

De manière générale, la symétrie et la défini-positivité d'une fonction noyau présentent plusieurs avantages:

1. *Avantages pratique* - Ces propriétés permettent la convergence d'algorithmes basés sur une programmation quadratique. Pour les SVM par exemple, une matrice de Gram définie positive garantit la convergence de l'algorithme d'apprentissage (quadratique) vers une solution optimale et unique. Dans le cas contraire, l'algorithme d'apprentissage peut converger vers une solution localement mais pas globalement optimale ou, pire, ne pas converger (problèmes numériques).
2. *Avantages théorique* - En considérant l'expansion sous-jacente ϕ dont l'existence est garantie par le théorème de Mercer, on peut interpréter les algorithmes à noyau comme des algorithmes linéaires dans le *Feature Space*. La matrice de Gram encode alors les positions relatives d'un ensemble de données dans le *Feature Space*.

I-3- Conclusion :

Dans le foisonnant domaine des méthodes dites à *noyau*, dont les SVMs sont le fer de lance, on constate que les efforts de recherche ont été essentiellement portés sur les applications, délaissant quelque peu l'étude des noyaux eux-mêmes. Cet état de fait est d'autant plus surprenant que ces derniers conditionnent largement l'efficacité des traitements, par leur aptitude à révéler les similarités existant au sein des données traitées. Les méthodes à noyau constituent une partie importante du présent mémoire.

La théorie des noyaux reproduisants a permis le développement fulgurant d'une classe d'algorithmes de reconnaissance des formes dont la formulation ne dépend pas de la nature des données traitées, ni de l'espace de représentation adopté pour résoudre les problèmes, partant de la constatation que ces méthodes nécessitent le stockage et la manipulation de matrices dont la taille est égale au nombre de données traitées. Au delà de ce caractère universel, celles que l'on range désormais sous le qualificatif de méthodes à noyau doivent également leur succès à l'essor de la théorie statistique de l'apprentissage, au sein de laquelle la prédiction de leurs performances en généralisation fait aujourd'hui encore l'objet d'études approfondies.

CHAPITRE II : METHODES DE CLASSIFICATION EN RDF: ETAT DE L'ART.

II-1- Introduction :

Les moyens de stockage de l'information, connaissent depuis quelques années, une forte croissance. De nos jours, il n'est plus rare de voir des disques durs atteignant le téraoctet. Cette capacité a permis de stocker une quantité importante de données dans le but d'acquérir une meilleure connaissance de l'information pour des tâches telles que la prise de décision. Toutefois, face à un tel volume de données, la recherche de l'information pertinente est devenue complexe et couteuse.

Il est aisé et courant de manipuler des documents de type texte : le stockage, la manipulation, la recherche d'information sont des opérations abordables par le grand public. Or, la société de l'information actuelle est très friande pour d'autres types de media : les nouveaux moyens de communications (téléphones portables UMTS, accès Internet à haut débit, télévision numérique) tendent à vulgariser l'usage de documents audiovisuels auprès d'un nombre d'usagers en constante augmentation. Peu développé jusqu'alors, et souvent cantonné à un monde exclusivement professionnel, le besoin de traitement de ce type de documents apparaît assez critique. En effet, il n'est pas aisé de stocker, manipuler et exploiter des documents audiovisuels. Il n'est pas encore possible d'utiliser un moteur de recherche sur une grande collection d'émissions de télévisions de la même façon que l'on recherche des mots clés sur l'ensemble des pages Internet disponibles sur le réseau. En effet, un document audiovisuel est une source d'information multimodale riche.

Si l'on ne considère que les documents audio, on se retrouve principalement devant des problèmes de classification ou de reconnaissance de formes. Les systèmes utilisés ont alors pour but de segmenter et/ou d'identifier les différentes composantes du flux sonore. C'est ainsi que sont apparus des systèmes de détection de la parole et de la musique, des systèmes d'identification de la langue, des systèmes de détection de locuteurs, des systèmes de transcription automatique des zones de parole.

II-2- Classification automatique des signaux audio :

La classification automatique vise à assigner des objets O_i, $i \in \{1,...,m\}$ à des *catégories* ou *classes* C_j ; $j = 2$. Dans notre cas c'est la classification binaire. Les objets sont des signaux ou des segments de signaux audio provenant des stations radio. Le silence est généralement très rare en radio. A contrario, la musique y est omniprésente.

La parole se retrouve aujourd'hui noyée dans un magma de musique et de bruit. La classification du flux audio est donc nécessaire; la classification d'un signal consiste pour nous à découper le signal suivant les composantes acoustiques que nous voulons décrire. Cette tâche doit permettre à la machine de faire la différence entre la parole, la musique et le bruit.

En ce qui nous concerne, nous nous intéressons dans non travaux à la séparation en deux classes **parole/musique**. Cette étape est très importante dans diverses applications du domaine multimédia, nous pouvons citer :

- La détection de la parole d'un enregistrement audio est cruciale avant l'application d'un système de reconnaissance automatique de la parole ou de reconnaissance du locuteur.

- Au sein d'un système de transcription automatique d'émissions radiophoniques. En effet, le passage au moteur de reconnaissance de tout événement sonore n'étant pas de la parole peut provoquer des erreurs de transcription, faisant ainsi baisser les performances du système.
- L'aide au sous-titrage : séparer la parole de la musique peut permettre à la personne sous-titrant des documents audiovisuels d'aller plus vite. En effet, elle pourra facilement passer d'une zone de parole à une autre et éviter ainsi toutes les zones sans discours qu'elle aurait dû écouter.
- Il est aussi connu que les codeurs spécifiques de la parole ou de la musique sont plus performants que les codeurs généraux. Il est donc intéressant de réaliser sur un signal sonore une classification automatique en parole/musique afin de pouvoir faire appel au codeur correspondant.

Dans les applications d'indexation multimédia, une classification en parole/musique est généralement considérée comme une étape préalable pour une indexation plus fine d'un

document audiovisuel. Le principe général des systèmes de classification audio (figure II.1) inclut deux étapes :

- Une *étape d'apprentissage* qui peut être vue comme une phase de développement aboutissant à la mise en œuvre d'une stratégie de classification;
- Une *étape de test (d'épreuve)* par laquelle les performances de généralisation du système de classification sont évaluées.

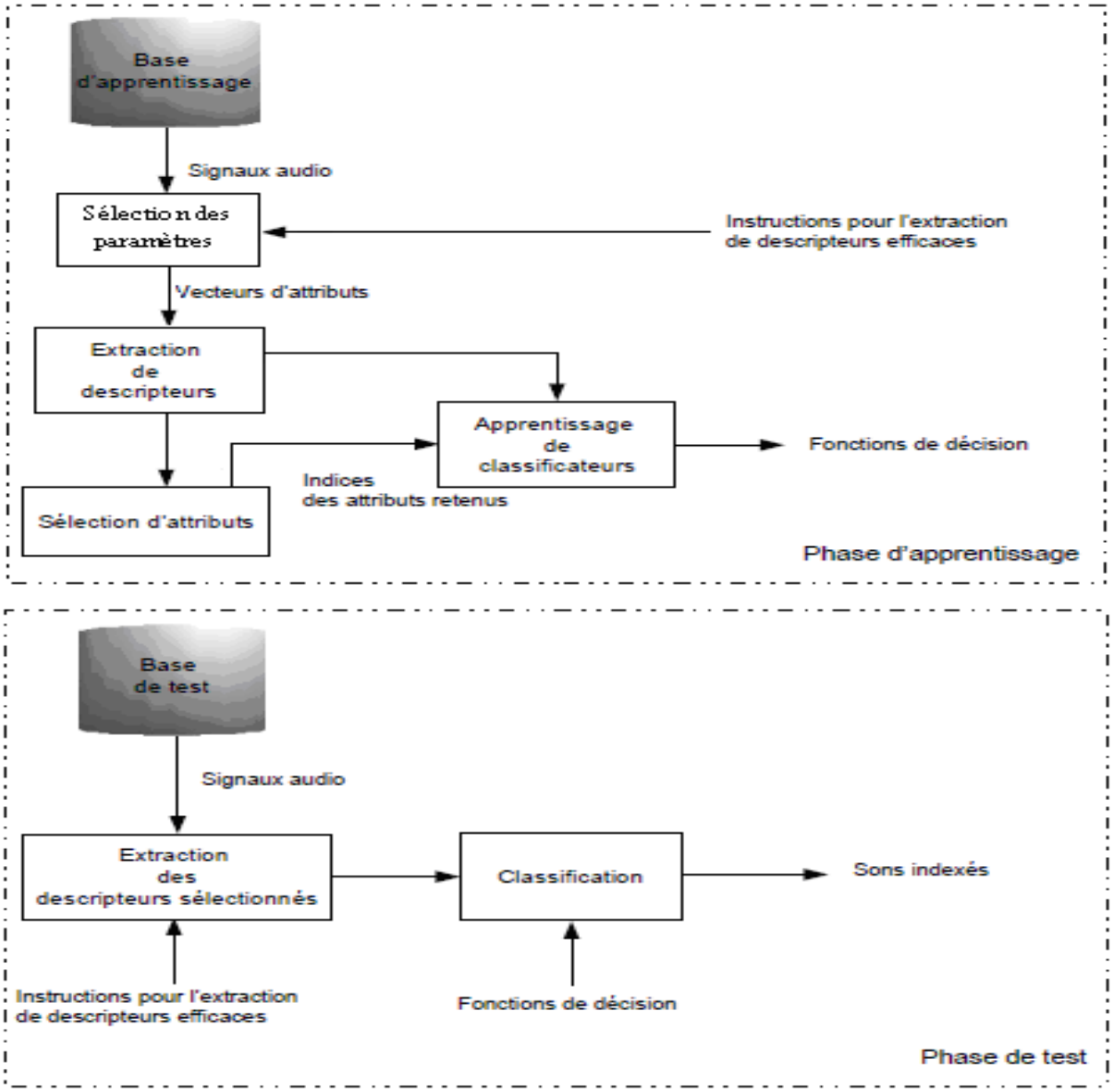


FIG II.1. Système de classification audio.

En général, un système n'est prêt pour une utilisation réelle qu'après une succession d'étapes d'apprentissage et de test permettant de mettre en place une stratégie de classification efficace. Il y a deux possibilités :

– *Suffisamment de données supervisées* disponibles **:**

- Un ensemble test maintenu à part pour l'évaluation finale.
- Seul l'ensemble d'apprentissage doit être partagé pour permettre la phase de validation sur des données indépendantes.

– *Trop peu de données supervisées* :

- Un seul ensemble de données supervisées pour validation ET évaluation finale.

La phase d'apprentissage comprend :

- L'extraction à partir d'une base de sons de référence appelée *base d'apprentissage*, de descripteurs sous forme de paramètres numériques. Ces paramètres qui sont aussi appelés trames acoustiques, traits, descripteurs ou *attributs* (*features*), extraits à un intervalle régulier, sont sensés caractériser des propriétés des signaux pouvant révéler leur appartenance à l'une des classes envisagées. Classiquement dans le domaine de l'analyse de la parole, notamment en ce qui concerne la reconnaissance de la parole, il est supposé que le signal de la parole peut être considéré comme stationnaire pour des durées de l'ordre de 20 à 30 ms. De ce fait, un signal continu de parole est généralement segmenté en des unités temporelles de 20 à 30 ms. Des caractéristiques du signal sont ensuite extraites de chaque unité temporelle.
  Pour généraliser l'espace de représentation à plusieurs classes sonores, nous combinons plusieurs caractéristiques issues de diverses paramétrisations dans un même vecteur. Cependant, ce processus se fait généralement d'une manière empirique sans aucune méthodologie claire permettant de nous guider dans le choix de caractéristiques qu'il faut inclure dans le vecteur de caractéristiques. L'étape d'extraction nous permet d'avoir une matrice de données, qui est utilisée comme entrée au classificateur.
- Généralement des prétraitements sont effectués sur la matrice de données. Les valeurs de plusieurs attributs, notamment issus de descripteurs de nature physique différente, présentent souvent des dynamiques assez hétérogènes. Les attributs possédant des valeurs

plus grandes risquent alors d'avoir une influence plus importante sur le comportement des différents traitements à suivre (sélection, classification), même si cela ne reflète pas forcement leur pertinence pour la tâche envisagée. Afin de contourner ce problème, il est classiquement fait appel à des techniques de **normalisation** permettant d'uniformiser les dynamiques des différents attributs. Habituellement, cette normalisation est réalisée de façon linéaire en ramenant la dynamique des attributs dans l'intervalle [-1,+1]. Cela s'obtient en estimant à partir de l'ensemble des données les valeurs maximales pour le *j*-ème attribut (en valeur absolue) :

$$\tilde{x}_j = \max_{1 \le j \le n} \left| x_{k,j} \right| \qquad \text{(II.1)}$$

Et en prenant $\hat{x}_{k,j} = \frac{x_{k,j}}{\tilde{x}_j}$

Cette normalisation est appelée "normalisation min-max". Ce type de normalisation sera utilisé par la suite. Cette normalisation est notamment obligatoire lors de l'utilisation de certains noyaux (ex : RBF (Radial Basis Function)) afin d'éviter leurs saturation, il n'est donc pas possible d'appliquer les SVM RBF sur des ensembles non normalisés.
De même, les points de test sont ajustés grâce à ces mêmes paramètres de normalisation.

- La sélection des paramètres de classificateur comme le noyau gaussien dans le cas des SVM. Il y a toujours un réglage nécessaire de ces paramètres pour l'apprentissage, qui permet de garantir la qualité du résultat du processus SVM.
- La sélection d'attributs efficaces; en pratique un nombre élevé de descripteurs candidats qui ne servent pas tous sur les performances de classification est considéré, il est alors intéressant d'avoir recours à des techniques permettant de retenir un sous-ensemble d'attributs (de plus petite taille) permettent alors, pour des coûts de calcul moindres, d'obtenir des performances similaires, voire meilleures que si la totalité des descripteurs était utilisée.
- L'apprentissage à partir des attributs sélectionnés de *fonctions de classification* ou *fonctions de décision*, lesquelles elles serviront à assigner des observations d'attributs de nouvelles données (de test) à l'une des classes possibles. Notons que l'apprentissage est supervisé : on connait la classe de chacun des sons de la base d'apprentissage.

Lors de l'étape de test, il n'est nécessaire d'extraire des signaux que les attributs qui ont été retenus, et de décider de l'appartenance de ces signaux aux classes possibles en utilisant les fonctions de décisions apprises.

La conception de systèmes de classification audio apparaît ainsi comme un processus complexe qui demande la coopération de techniques issues de diverses disciplines :

- L'acoustique et la perception des sons, qui fournissent les pistes nécessaires au développement de descripteurs adéquats;
- Le traitement du signal, outil incontournable pour l'extraction efficace de ces descripteurs;
- L'apprentissage automatique, qui permet de mettre en œuvre des stratégies de classification performantes en exploitant les descripteurs obtenus.

Parmi les algorithmes d'apprentissage, les méthodes à noyaux connaissent un énorme succès depuis quelques années. Les noyaux sont des fonctions de similarité vérifiant certaines propriétés mathématiques. Ils peuvent être utilisés avec des algorithmes d'apprentissage linéaires tels que les SVM discriminatives pour extraire des relations non-linéaires.

Les SVM ont l'avantage d'être discriminatives, par opposition aux approches dites "génératives" ou "informative". On peut différencier deux types d'approche pour résoudre un problème d'apprentissage supervisé :

- L'approche *discriminative* modélise directement la règle de classification P(Y|X),
- L'approche *générative* cherche à modéliser la distribution jointe P(Y, X) des entrées et des sorties. On en déduit ensuite la règle de classification par application de la loi de Bayes.

Les approches génératives présupposent une structure particulière (souvent mal justifiée) des distributions de probabilités des données mises en jeu, ce qui constitue un point fort des méthodes discriminatives du point de vue de leur fondement théorique. Récemment, les SVM ont démontré leur capacité à obtenir d'excellentes performances pour des tâches de classification, de régression ou de segmentation [35, 36].

La figure II.2 regroupe les principales méthodes de classification génératives et discriminatives.

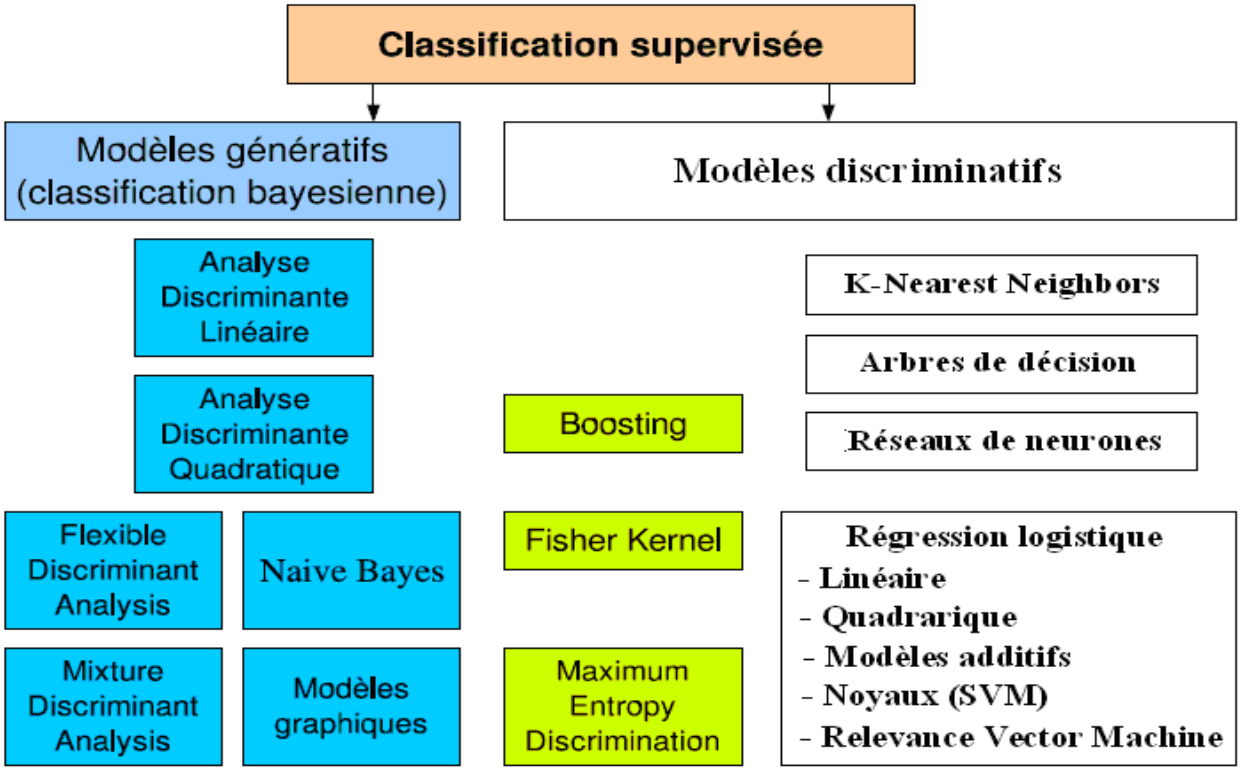


FIG II.2. Les principales méthodes de classification, d'un côté génératives, de l'autre discriminatives. Les méthodes indiquées par une couleur verte sont des méthodes discriminatives qui peuvent être liées à un modèle génératif.

Nous allons présenter dans ce qui suit, les méthodes discriminatives les plus utilisées en classification parole/musique avec apprentissage supervisé: les *k*-plus proches voisins, les réseaux de neurones et les SVM; ces deniers seront présentés plus en détails dans (§II-3-3).

II-3- Les méthodes de classification/ discriminatives :

II-3-1- Les *k*-plus proches voisins (*k*-ppv) [7] :

La méthode des *k*-plus proches voisins (*k*-ppv) fait partie des méthodes par fonction noyau, qui détermine une mesure de proximité entre les observations x_i et x_j. La règle de décision des *k*-ppv est très simple : une observation nouvelle est classée en prenant la classe majoritaire parmi les *k* observations d'apprentissage les plus proches (figure II.3). Cela revient à choisir une fonction noyau simple, constante dans l'hypersphère contenant les *k* voisins et nulle ailleurs.

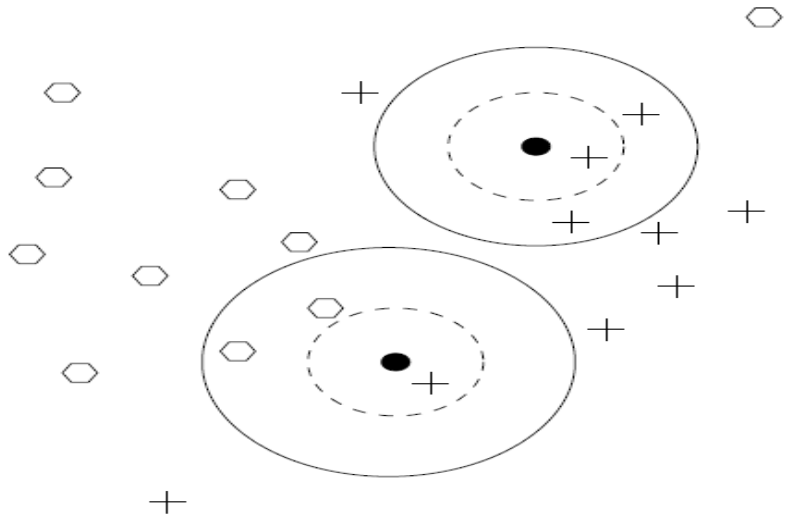


FIG II.3. Décision par 1-ppv (cercle pointillé) et 3-ppv (cercle en trait plein) sur un ensemble d'observations appartenant à 2 classes.

Cette méthode n'a pas réellement de phase d'apprentissage, c'est-à-dire qu'il n'y a pas de construction de modèle. Tout repose sur :

- L'ensemble d'apprentissage stocké en mémoire;
- Une mesure de distance, c'est-à-dire la fonction noyau. Parmi les distances existantes, on peut citer les distances euclidiennes et de Mahalanobis. Ces dernières n'étant pas toujours adaptées aux données, notamment les données audio à cause de leur caractère non linéaire, il peut être avantageux de considérer d'autre type de distances dans un espace de plus grande dimension.
- Une méthode de choix de la classe, en général, la méthode consiste à choisir la classe majoritaire parmi les *k* observations d'apprentissage les plus proches.

Notons que la capacité de généralisation de cette méthode dépend du paramètre *k*. le réglage de *k* permet de lisser la modélisation. En effet, un *k* élevé permet d'englober plus de voisins et ainsi d'être moins sensible aux erreurs d'apprentissage, mais dans le même temps les *k* plus proches voisins doivent rester très proches de *y* pour que les approximations soient fiables. Par suite, il est nécessaire de réaliser un compromis, en choisissant une valeur de *k* plus petite que le nombre de d'observations *l*. Cette méthode a l'avantage de pouvoir s'appliquer à des cas de discrimination faisant intervenir un nombre élevé de classes. L'algorithme des *k*-ppv peut être résumé ainsi :

Algorithme II.1: *k*-ppv.

Début

Soit *x* l'ensemble à classer

et $C = C_1, ..., C_n$ l'ensemble des classes

Pour chaque observation (y, C_i) de l'ensemble d'apprentissage **faire**

Calculer la distance $D(y, x)$ entre y et x

FinPour

Dans les *k* plus proches voisins de *x*

Compter le nombre d'occurrences de chaque classe

Classer *x* dans la classe apparaissant le plus souvent.

Fin

Les défauts de ces algorithmes simples sont d'une part le besoin de garder en mémoire les données d'apprentissage et d'autre part la nécessité de calculer les distances à tous les points d'apprentissage. Ils sont donc fort consommateur en espace mémoire et en temps de calcul. En plus, si les données d'apprentissage ne couvrent pas suffisamment les classes à étudier, les performances du classificateur se dégradent rapidement.

II-3-2- Les réseaux de neurones : Le Perceptron Multi-Couches (PMC)

Les réseaux de neurones (RN) constituent un domaine de recherche très intéressant et sont très couramment utilisés lorsque l'on parle de classification. Les RN ont montré des résultats remarquables dans le domaine de la reconnaissance du manuscrit. Ils ont été notamment appliqués à des problèmes tels que : la reconnaissance de visage, le contrôle de robot, la reconnaissance de la parole, l'identification du locuteur, la segmentation parole/musique, etc.

Les RN réalisent un traitement d'information distribué et sont composés d'unités de calcul primitives (les neurones formels) fonctionnant en parallèle et reliées entre elles par des connexions. Un neurone formel reçoit un nombre variable d'entrées (similaires aux signaux électrochimiques du système biologique) en provenance de neurones en amont. A chacune de ces entrées est associé un poids représentant la force de la connexion. Il est aussi doté d'une sortie unique qui se ramifié ensuite pour alimenter les neurones en aval. Le principe de fonctionnement

du neurone est simple, il calcule la somme pondérée de ses entrées et passe cette valeur à une fonction d'activation qui détermine l'excitation de ce neurone. La figure II.4 illustre l'architecture d'un neurone formel. La sortie du neurone $y = F(\sum_{i=1}^{n} w_i x_i)$ dépend de la fonction d'activation choisie : Fonction seuil, linéaire par morceaux, sigmoïde, gaussienne etc.

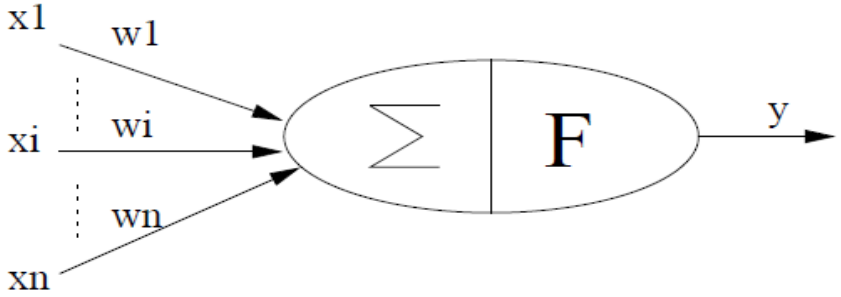


FIG II.4. Architecture d'un neurone formel à *n* entrées.

Dans un réseau, la connaissance se trouve dans la topologie même du réseau et dans les poids de connexions. L'apprentissage d'un RN est réalisé à l'aide de méthodes d'apprentissage automatique utilisant la descente du gradient de l'erreur et se fait par modification des poids des connexions du réseau en fonction des données d'apprentissage. Pendant la phase d'apprentissage, les poids changent afin d'aboutir à une solution du problème et la conservation de ces poids constitue l'expérience du réseau de neurones. Aucune hypothèse sur la distribution des données n'est nécessaire.

Enfin, les RN ont de nombreuses propriétés très intéressantes telles que leur robustesse au bruit, leur flexibilité et leur capacité importante de généralisation. Nous allons présenter rapidement le réseau de neurones le plus souvent utilisé dans le domaine de la reconnaissance automatique de la parole : le perceptron multi-couches (PMC).

II-3-2-1- Le Perceptron Multi-Couches (PMC) :

Le perceptron Multi-Couches est issu des travaux de F. Rosenblatt sur le perceptron monocouche [37]. Un PMC est un réseau dont les neurones sont disposés en plusieurs couches successives et où chaque neurone d'une couche est connecté à tous les neurones de la couche suivante et de la couche précédente mais pas aux neurones de la même couche. Le PMC est un réseau passe-avant (*feed-forward*), c'est-à-dire que les informations ou activations ne vont

circuler que dans un seul sens, des neurones de la couche d'entrée vers les neurones de la couche de sortie (figure II.5).

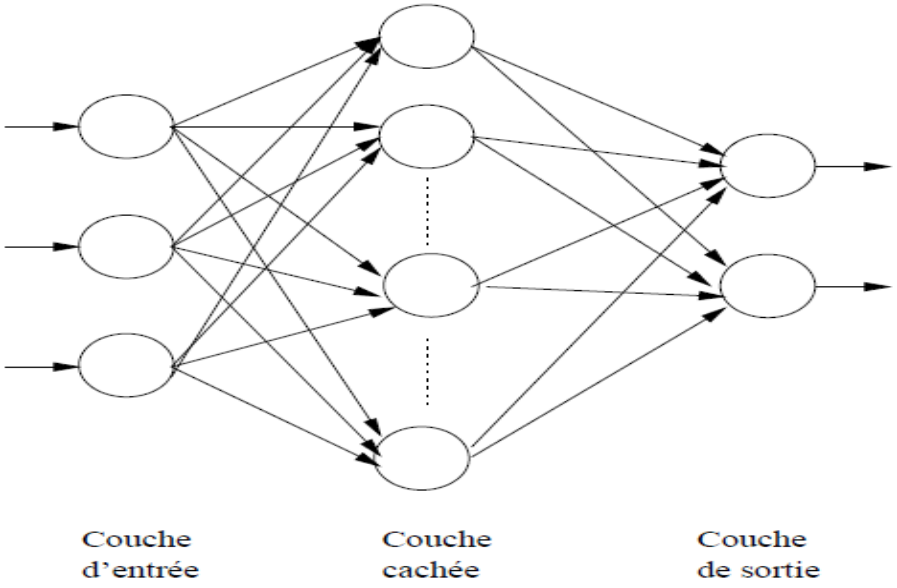


FIG II.5. Architecture d'un Perceptron Multi-Couches à une couche cachée.

Une couche cachée dans un PMC correspond à une couche qui n'est ni la couche d'entrée, ni celle de sortie. De plus, un PMC peut avoir autant de couches cachées que désiré mais il a été montré [37] que quelque soit le nombre de couches cachées dans un PMC, il existe un PMC équivalent avec une seule couche cachée. Cette couche cachée permet de modéliser des fonctions de décision complexes et non linéaires entre n'importe quels espaces d'entrée et de sortie.

L'apprentissage des PMC se fait par rétropropagation du gradient de l'erreur [37]. Le principe est d'adapter les différents poids des connexions en propageant l'erreur commise en sortie du réseau.

II-3-3- Méthodes à vecteurs de support bi-classes :

II-3-3-1- Introduction :

Ces fameuses méthodes de classification ont été inventées en 1992 par Boser *et al*. [2], mais leur dénomination par SVM n'est apparue qu'en 1995 avec Cortes *et al*. [3].
Nous avons fait le choix, pour cette méthode de classification, qui nous a paru être un outil efficace, les SVM présentent notamment deux avantages majeurs :

1. Les méthodes à vecteurs de support sont issues de la théorie de l'apprentissage statistique [32], qui définit un cadre théorique rigoureux;
2. Les SVM affichent pour des applications pratiques de très bonnes performances, et leur succès est dû à l'utilisation des noyaux pour transformer le SVM en algorithme non-linéaire pouvant être appliqué sur des données variées. Les noyaux et les SVM ont alors permis d'utiliser l'apprentissage numérique dans des problèmes traitant des données complexes.

La première affirmation semble tout à fait justifiée, Il existe en effet un lien direct entre la théorie de l'apprentissage statistique et l'algorithme d'apprentissage de SVM. La plupart des techniques du Machine Learning (ML) possèdent un trop grand nombre de paramètres d'apprentissage à fixer par l'utilisateur (structure d'un réseau de neurones, coefficient de mise à jour du gradient, ...). De plus, avec ces méthodes, le nombre de paramètres à calculer par l'algorithme d'apprentissage est en relation linéaire, voir exponentielle, avec la dimension de l'espace d'entrée (dimension des données). La formulation élégante de SVM laisse très peu de place aux paramètres utilisateurs et le nombre de paramètres est linéaire en la taille de l'ensemble d'entrainement (training set). SVM est donc une méthode de classification particulièrement bien adaptée pour traiter des données de très haute dimension telles que les textes et les images ou même de l'audio.

La seconde parait moins évidente. Les SVM ont connu ces dernières années un certain succès au travers d'applications variées. Parmi elles, nous pouvons citer, notamment, la catégorisation de textes, la détection d'événements, la classification de gestes, la détection de visages, la recherche par similarité sur des textures, la reconnaissance d'objets 3D ou 2D sous différents angles de vue, la classification de chiffres écrits à la main, la vérification d'identité, l'identification de particules, etc. Dans la plupart des cas, la performance du SVM égale ou dépasse celle des classifieurs classiques. De nombreux travaux [3], ont démontré la supériorité du SVM sur les méthodes discriminantes classique telles que le PMC, le discriminant de Fisher [7], le *k*-ppv, etc.

Nous présentons dans la section suivante, les aspects théoriques des Méthodes à Vecteurs Supports, qui sont les classificateurs utilisés dans notre système, et qui ont donné des résultats prometteurs en discrimination parole/musique [38]. Ces classificateurs biclasses sont connus pour

offrir de bonnes capacités de généralisation, même lorsque la dimension des vecteurs qu'ils doivent classifier est grande mais pertinente.

II-3-3-2- Classificateur linéaire :

II-3-3-2-1- Définition :

Le but des SVM est d'apprendre une fonction qui correspond aux données (entrée/sortie) vues et qui prédit les sorties pour les entrées qui n'ont pas encore été vues. Les entrées peuvent être des descriptions d'objets et les sorties la classe des objets donnés en entrée.

Un classificateur est dit linéaire lorsqu'il est possible d'exprimer sa fonction de décision par une fonction linéaire en x. Dans la suite, nous supposons que les données nous sont fournies dans le format vectoriel. Notre espace d'entrée X correspondant donc à IR^n où n est le nombre de composantes des vecteurs contenant les données. On peut, en toute généralité, exprimer une telle fonction comme ceci :

$$f(x) = \langle w\ , x \rangle + b \qquad \text{(II.2)}$$

$$= \sum_{i=1}^{n} w_i x_i + b$$

où $w \in IR^n$ *et* $b \in IR$ sont des paramètres à estimer de la fonction de décision.

Pour décider à quelle catégorie une donnée appartienne $\tilde{x}$, il suffit de prendre le signe de la fonction de décision : $y = \text{sgn}(f(\tilde{x}))$. Géométriquement, cela revient à considérer un hyperplan qui est le lieu des points x satisfaisant $\langle w, x \rangle + b = 0$. En orientant l'hyperplan (C.à.d., en fixant un coté pour lequel les données sont classées positivement), la règle de décision consiste à observer de quel coté de l'hyperplan se trouve la donnée $\tilde{x}$. La figure II.6 représente la situation dans IR^2 (espace d'attributs).

On voit que le vecteur $\vec{w}$ définit la pente de l'hyperplan : $\vec{w}$ est perpendiculaire à l'hyperplan. Le terme b quant à lui permet de translater l'hyperplan parallèlement à lui-même. Deux remarques à propos de la figure II.6. Il est facile d'imaginer deux nuages de points tels qu'aucune droite ne sépare les motifs ronds des croix. On dit dans ce cas que les données sont non-séparables. Nous nous limitons pour l'instant au cas séparable. La deuxième remarque est

que les droites qui séparent les motifs ronds des croix ne sont pas équivalentes du point de vue de la qualité de prédiction qui leur est associée.

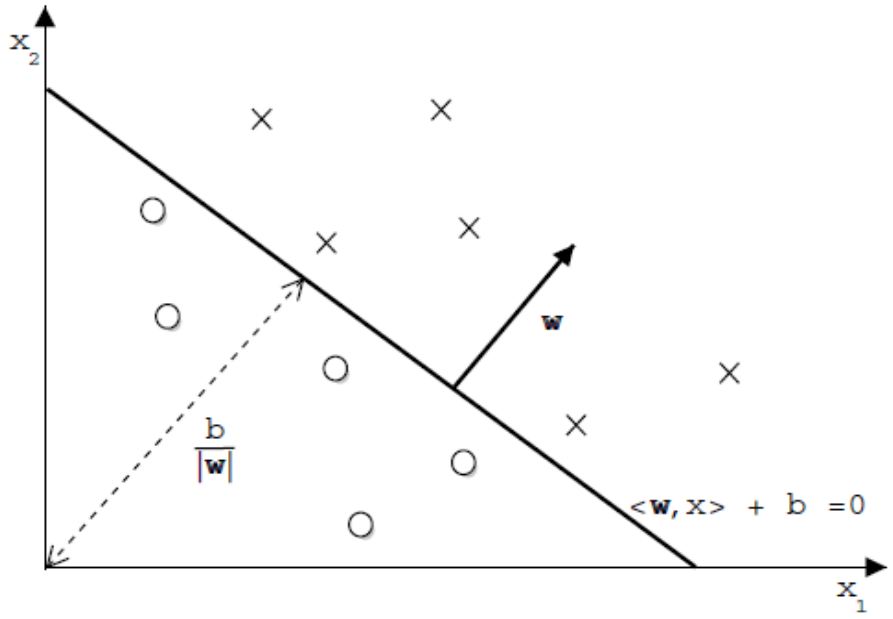


FIG II.6. Représentation dans IR^2 de l'hyperplan correspondant à la fonction de décision d'un classificateur linéaire.

II-3-3-2-2- Marge de l'hyperplan :

Avant de définir la marge, il est utile d'introduire la notion de *training set linéairement séparable*. Un training set $(x_1, y_1),(x_2, y_2),...,(x_m, y_m)$ est linéairement séparable SSI :

$$\exists w \in IR^n,\ b \in IR : y_i(\langle w\ , x_i \rangle + b) \geq 0 \quad \forall\ i \in 1...m \tag{II.3}$$

La définition consiste à dire qu'il doit exister un hyperplan laissant d'un coté toutes les données positives et de l'autre, Toutes les données négatives.

La marge géométrique représente la distance euclidienne prise perpendiculairement entre l'hyperplan et la donnée x_i. En prenant un point quelconque x_p se trouvant sur l'hyperplan, la marge géométrique peut s'exprimer $\frac{w}{\|w\|}.(x_i - x_p)$. La figure II.7 illustre la situation. On peut calculer la valeur de la marge géométrique:

$$\langle w, x_i \rangle + b = f(x_i)$$

$$- \frac{\langle w, x_p \rangle + b = 0}{w.(x_i - x_p) = f(x_i)}$$

$$\Rightarrow \quad \frac{w}{\|w\|}.(x_i - x_p) = \frac{f(x_i)}{\|w\|} \qquad \text{(II.4)}$$

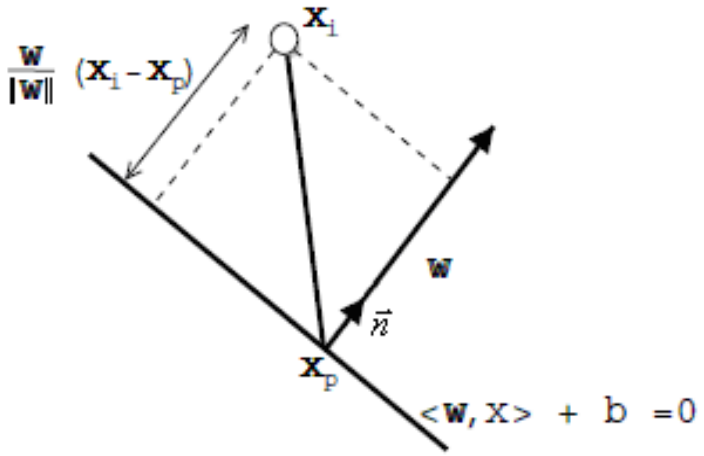


FIG II.7. Expression de la marge pour la donnée x_i (La marge est calculée à partir du produit scalaire entre les vecteurs situés à la frontière de chaque classe et le vecteur unitaire normal de l'hyperplan séparateur).

La marge est la projection du vecteur $(x_i - x_p)$ sur le vecteur normal à l'hyperplan $\vec{n} = \frac{w}{\|w\|}$. Les algorithmes d'apprentissage ayant pour critère de maximiser la marge du training set, sont appelés classificateurs à marge maximale. SVM, en particulier, en est un.

II-3-3-2-3- Hyperplans canoniques :

Dans le cadre de classificateurs à marge maximale, l'hyperplan séparateur correspond à la médiatrice du plus petit segment de droite reliant les enveloppes convexes des deux catégories. Notons que l'on suppose aussi que le training set est linéairement séparable. Dès lors, nous pouvons définir deux plans se trouvant de part et d'autre de l'hyperplan et parallèles à celui-ci, sur lesquels reposent les données les plus proches. La figure II.8 illustre cette situation :

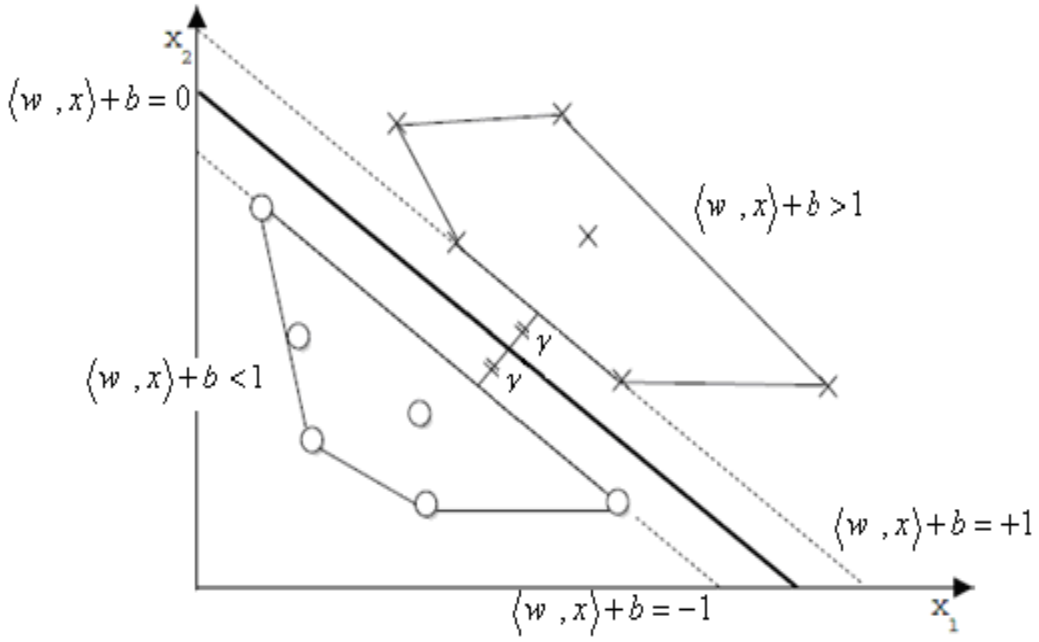


FIG II.8. Hyperplans canoniques.

$\langle w\ ,x\rangle + b = +1$ et $\langle w\ ,x\rangle + b = -1$. Ces deux hyperplans sont appelés *hyperplans canoniques*. Notons que la marge γ des hyperplans canoniques est $\frac{1}{\|w\|}$. Le vecteur w possède à présent une signification très claire.

Les motifs situés sur les lignes canoniques sont les vecteurs supports (VS) d'où l'appellation de la méthode d'apprentissage. Nous allons voir après que ces vecteurs sont les points représentatifs de toute la masse de données des deux classes.

II-3-3-2-4- Relation avec l'apprentissage statistique :

Intuitivement, le fait d'avoir une marge plus large (maximiser le pouvoir de généralisation) procure plus de sécurité lorsque l'on classe une donnée inconnue. La partie gauche de la figure II.9, nous montre qu'avec l'hyperplan optimal, une nouvelle donnée reste bien classée alors qu'elle tombe dans la marge. On constate sur la partie droite qu'avec une plus petite marge, la donnée se voit mal classé.

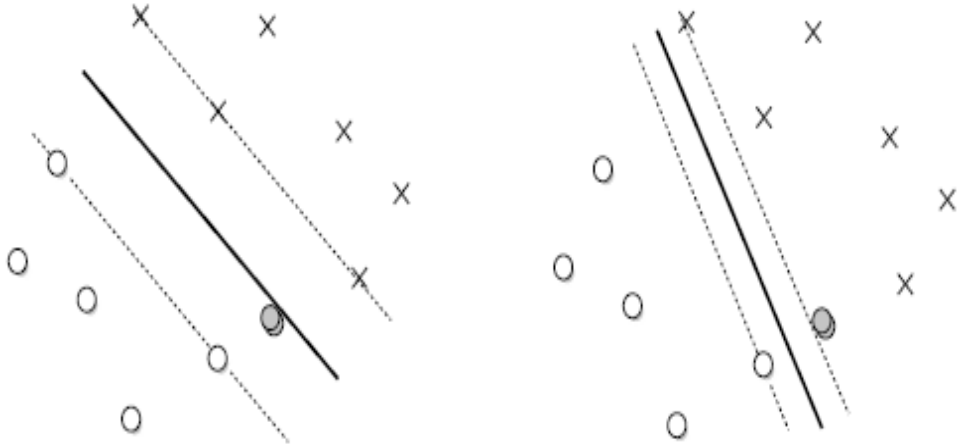


FIG II.9. Le même jeu de données classé par un hyperplan à marge maximale (à gauche) et un hyperplan quelconque ne commettant pas d'erreur sur le *training set* (à droite). La boule grisée est une donnée du *test set*.

II-3-3-2-5- Sur et sous-apprentissage (Over and Under-fitting) :

Nous présentons dans la figure suivante, deux situations d'apprentissage des données à savoir le *sur-apprentissage* et le *sous-apprentissage* que l'on essaye à éviter dans la pratique.

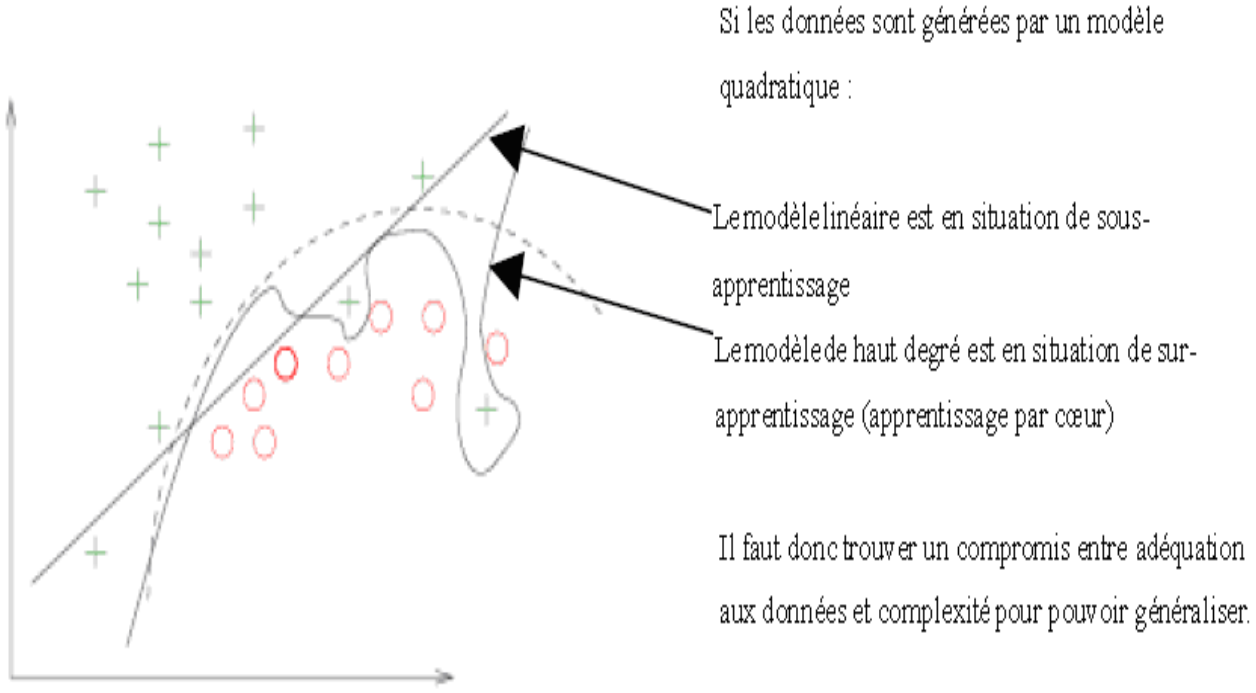


FIG II.10. Sur et sous-apprentissage des données, avec le modèle quadratique est le plus adapté.

II-3-3-2-6- Classificateur à marge maximale :

Maintenant que nous avons défini les notions de marges et d'hyperplans canoniques, nous pouvons formuler un problème d'optimisation mathématique tel que sa solution nous fournisse l'hyperplan optimal (maximisant la marge) :

$$\begin{cases} \max_{w\in IR^2, b\in IR} \dfrac{1}{\|w\|} \\ \langle w, x_i\rangle + b \geq +1 \quad quand \quad y_i = +1 \\ \langle w, x_i\rangle + b \leq -1 \quad quand \quad y_i = -1 \end{cases} \tag{II.5}$$

Pour maximiser la marge, il faut donc minimiser $\|w\|$. Par la suite, nous chercherons au fait à minimiser $(1/2).\|w\|^2$, pour simplifier les calculs. Le problème peut se transformer sous la formulation suivante équivalente :

QP1 : Minimiser $W(w,b) = \dfrac{1}{2}\|w\|^2$ (II.6)

Tel que $y_i(\langle w, x_i\rangle + b) \geq 1 \quad \forall i = 1...m$

Il s'agit d'un problème d'optimisation quadratique (QP) sous contraintes linéaires dont la fonction objective (II.6) est à minimiser. Cette fonction objective est le carré de l'inverse de la double marge. L'unique contrainte stipule que les données doivent être bien classées et qu'ils ne dépassent pas les hyperplans canoniques, c'est-à-dire qu'on ne permet aucune donnée d'apprentissage x_i dans la marge γ.

Dans cette formulation, les variables à fixer sont les composantes w_i *et* b, donc QP1 implique le réglage de n+1 paramètres (n étant la dimension de l'espace d'entrée plus la dimension du scalaire b). En gardant cette formulation telle quelle, on tombe dans le même problème critique des méthodes classiques du ML dit d'« overfitting ». Pour éviter cela, il est nécessaire d'introduire une formulation dite duale du problème. Un problème dual est un problème fournissant la même solution que le primal (initial) mais dont la formulation est différente. On appellera *variables primales*, les variables du problème primal, et *variables duales*, les variables du problème dual qui n'interviennent pas dans le primal. On utilisera souvent les termes primal et dual pour se référer aux problèmes.

Pour dualiser QP1, nous devons former ce que nous appelons le Lagrangien. Il s'agit de faire renter les contraintes dans la fonction objective et de pondérer chacune d'entre elles par une variable duale :

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i [y_i(\langle w, x_i \rangle + b) - 1] \tag{II.7}$$

Les variables duales α_i intervenant dans le Lagrangien sont appelées multiplicateurs de Lagrange, une pour chaque point de donnée. Selon le type de contraintes qu'ils représentent, et sachant que L doit être minimisé par rapport aux variables primaires w *et* b (maximisation de la marge) et maximisé par rapport aux variables duales α_i (pour s'assurer qu'un maximum de contraintes soient satisfaites), les multiplicateurs doivent respecter la condition suivante : $\alpha_i \geq 0$.

On peut donner une signification physique à ces multiplicateurs : la variable α_i représente la force avec laquelle la solution appuie sur la contrainte *i*. Ainsi, un hyperplan qui violerait la contrainte pour x_i (il classe cet exemple du mauvais coté) rendrait α_i très grand ce qui ferait fortement augmenter la fonction objective (II.7). Cette solution ne pourrait donc pas être retenue comme solution optimale. Le problème primal et sa formulation duale ont la même solution qui correspond à un point-selle du lagrangien (il faut le minimiser par rapport aux variables primaires et le maximiser par rapport aux variables duales).

A présent, nous introduisons les deux conditions de *Karush Kuhn et Tucker* (KKT), statuant sur l'optimalité d'une solution, elles peuvent être obtenues en exploitant directement les propriétés du point de selle de lagrangien. Au point de selle, la dérivée du lagrangien par rapport aux variables primaires doit s'annuler, ceci s'écrit :

$$\frac{\delta}{\delta w} L(w,b,\alpha) = 0 \quad \text{et,} \qquad \frac{\delta}{\delta b} L(w,b,\alpha) = 0 \tag{II.8}$$

Ce qui conduit à écrire :

$$\begin{cases} \dfrac{\delta L}{\delta w} = \|w\| - \sum_{i=1}^{m} \alpha_i y_i x_i = 0 \Rightarrow w = \sum_{i=1}^{m} \alpha_i y_i x_i & \text{(II.9)} \\ \dfrac{\delta L}{\delta b} = -\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \Rightarrow \sum_{i=1}^{m} \alpha_i y_i = 0 & \text{(II.10)} \\ b = -\dfrac{1}{2}(w.x_1 + w.x_2) & \text{(II.11)} \end{cases}$$

Le biais b peut ainsi être calculé à l'aide de l'équation (II.2) et en utilisant deux des vecteurs de support où x_1 est un vecteur de support de la classe positive et x_2 de la classe négative.

Remarquons qu'avec la formulation (II.9), on peut calculer w en fixant seulement m paramètres. L'idée va donc être de formuler un problème dual dans lequel w est remplacé par sa formulation (II.9). De cette façon, le nombre de paramètres à fixer est relatif au nombre de données du training set et non plus à la dimension de l'espace d'entrée n (supposée très élevée). Pour ce faire, nous substituons (II.9) et (II.10) dans le Lagrangien :

$$L(w,b,\alpha) = \frac{1}{2}\langle w,w\rangle - \sum_{i=1}^{m}\alpha_i[y_i(\langle w,x_i\rangle + b) - 1]$$

$$L(w,b,\alpha) = \frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j\langle x_i,x_j\rangle - \sum_{i=1}^{m}\alpha_i[y_i(\langle\sum_{i=1}^{m}\alpha_i y_i x_i, x_i\rangle + b) - 1]$$

$$L(w,b,\alpha) = \frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j\langle x_i,x_j\rangle + \sum_{i=1}^{m}\alpha_i - b\sum_{i=1}^{m}\alpha_i y_i - \sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j\langle x_i,x_j\rangle$$

D'après l'équation (II.10) : $\sum_{i=1}^{m}\alpha_i y_i = 0$

Donc : $L(w,b,\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y_i y_j\alpha_i\alpha_j\langle x_i,x_j\rangle$

A partir de quoi nous pouvons formuler le problème dual :

QP2 : Maximiser $W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j}^{m} y_i y_j\alpha_i\alpha_j\langle x_i,x_j\rangle$ (II.12)

Tel que $\begin{cases}\sum_{i=1}^{m}\alpha_i y_i = 0 \\ \alpha_i \geq 0 \quad \forall i = 1...m\end{cases}$

C'est une forme d'optimisation quadratique de dimension m bien étudiée, pour laquelle il existe de bons algorithmes d'optimisation. Et, ce qui est important, il est connu que les problèmes de ce type ont des optima globaux uniques. Ce qui signifie, qu'il devrait y avoir un seul ensemble de α_i qui maximise la valeur de W.

La résolution du dual permet donc de calculer la fonction de l'hyperplan. Nous avons à présent tous les éléments nécessaires pour exprimer la fonction décision de notre classificateur :

$$f(x) = \sum_{i=1}^{m} \alpha_i y_i \langle x, x_i \rangle + b \qquad \text{(II.13)}$$

La solution de l'hyperplan ne requiert que le calcul des produits scalaires entre les vecteurs de l'espace d'entrée et un nombre réduit de points : les vecteurs supports. Notons qu'un grand nombre de termes de cette somme sont nuls. En effet seuls les α_i correspondants aux (VS) sont non nuls. On peut les voir comme les représentants de leurs catégories car si le *training set* n'était constitué que des VS, l'hyperplan optimal que l'on nous trouverait serait identique. Le SVM produit alors une solution clairsemée.

Le calcul de la fonction de décision requiert la résolution d'un gros problème d'optimisation quadratique. Il existe un vaste choix d'algorithmes performants pour le résoudre. Un algorithme spécifique, l'optimisation séquentielle minimale (SMO), a été utilisé afin de pouvoir manipuler des ensembles de très grande dimension. Il optimise la fonction objective duale du problème global en opérant à chaque itération sur un ensemble réduit à deux multiplicateurs de Lagrange. Par conséquent, l'algorithme SMO n'est pas gourmand en espace mémoire vu qu'il n'utilise pas des opérations sur la totalité de la matrice de Gram. Nous utilisons dans non expériences la librairie *LIBSVM* [25] développée à l'Université Nationale de Taiwan, qui implémente cet algorithme. Cette implémentation est particulièrement efficace, particulièrement en ce qui concerne le temps de traitement.

II-3-3-3- Les fonctions noyaux (*kernels*) :

II-3-3-3-1- Généralités :

Le classificateur à marge maximale que nous venons de présenter, permet d'obtenir de très bons résultats lorsque les données sont linéairement séparables. L'intérêt principal d'un classificateur de ce type réside dans le fait que l'on contrôle facilement le pouvoir de généralisation. Naturellement, un grand nombre de jeux de données sont non-linéairement séparables. Pour classer ce genre de données nous pourrons utiliser une fonction de décision non-linéaire. Géométriquement, cela reviendrait à avoir une (hyper)courbe qui marquerait la frontière entre les données positives et négatives. Le danger de ce genre de méthode vient du fait que le nombre de paramètres à calculer est très élevé et provoque souvent du sur-apprentissage.

L'idée retenue dans SVM va dans un autre sens : on va tenter de trouver une projection (*mapping*) de l'espace d'entrée vers un autre espace, dit *transformé* ou espace des caractéristiques (*feature space*), dans lequel les données sont linéairement séparables. La figure II.11 donne une représentation imagée de ce genre de mapping. Nous pouvons alors appliquer une méthode à marge maximale dans le *feature space* et garder le contrôle que nous avions sur le pouvoir de généralisation. La dimension du *feature space* est généralement très élevée. Cela ne pose pas de problème pour notre classificateur à marge maximale vu que sa formulation duale fixe le nombre de variables à déterminer en fonction de la taille de l'ensemble d'entrainement.
Nous noterons le *feature space F*, et le mapping vers cet espace : $\phi : X \Rightarrow F$.

Ce processus de projection des données dans l'espace induit par le noyau est équivalent à une extraction de caractéristiques systématique sur les données d'entrée.

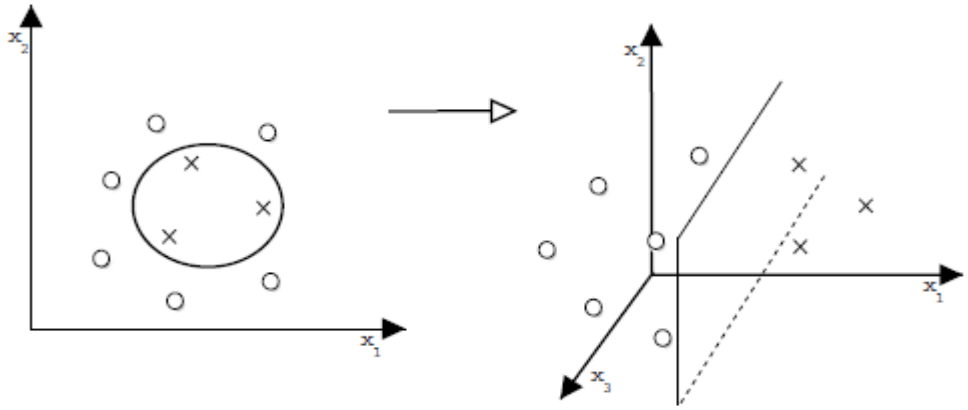


FIG II.11. Un mapping ϕ rendant les données *linéairement séparables*.

Notons ϕ une transformation non linéaire de l'espace d'entrée X vers l'espace des caractéristiques. On remplace x par $\phi(x)$ partout. Le problème dual d'optimisation devient :

QP3 : Maximiser $W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j}^{m} y_i y_j \alpha_i \alpha_j \langle \phi(x_i), \phi(x_j) \rangle$ (II.14)

Tel que $\begin{cases} \sum_{i=1}^{m} \alpha_i y_i = 0 \\ \alpha_i \geq 0 \quad \forall i = 1 \ldots m \end{cases}$

L'équation de l'hyperplan séparateur dans le nouvel espace est alors :

$$f(x) = \sum_{i=1}^{m} \alpha_i y_i \langle \phi(x), \phi(x_i) \rangle + b \qquad \text{(II.15)}$$

où les coefficients α_i et b sont obtenus comme précédemment par résolution du problème dual.

Tout cela est très bien, sauf que l'objection immédiate du praticien concerne le produit scalaire $\langle \phi(x_i), \phi(x_j) \rangle$ qui devient rapidement impossible à calculer quand la dimension n augmente. Fort heureusement, il existe la ruse qui fait vraiment la force des SVM, qui nous permet dans certains cas de court-circuiter le passage par les calculs dans l'espace des caractéristiques. En effet, il existe des fonctions bilinéaires symétriques définies positives $k(x_i, x_j)$ appelées fonctions **Noyaux**, facile à calculer et dont il a été montré dans [2], qu'elles correspondent à un produit scalaire dans un espace de grande dimension. Lorsqu'une telle correspondance est exploitable, le problème d'optimisation est équivalent au problème suivant :

QP3 : Maximiser $W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j}^{m} y_i y_j \alpha_i \alpha_j k(x_i, x_j)$ (II.16)

Tel que $\begin{cases} \sum_{i=1}^{m} \alpha_i y_i = 0 \\ \alpha_i \geq 0 \quad \forall i = 1...m \end{cases}$

Pour calculer l'hyperplan optimal dans le *feature space*, il suffit de remplacer toutes les occurrences du produit scalaire par le noyau. Plus généralement, tout algorithme d'apprentissage accédant exclusivement aux données au travers du produit scalaire (ou d'une grandeur qui en dérive) est dit *kernelisable*. Le produit scalaire lui-même peut être vu comme un noyau dont le mapping ϕ est l'identité.

L'équation de l'hyperplan séparateur dans le nouvel espace est alors :

$$f(x) = \sum_{i=1}^{m_c} \alpha_i y_i k(x, x_i) + b \quad \text{(II.17)}$$

Avec m_c le nombre de points support.

Donc tout algorithme à produit scalaire (linéaire) peut être noyauté (non linéaire) en remplaçant $\langle .,. \rangle$ par $k(.,.)$. Néanmoins ce dernier devra remplir les conditions de Mercer.

Les noyaux les plus utilisés sont les noyaux linéaires et RBF. Le choix du noyau RBF est souvent recommandé [39]. En effet, il correspond à un meilleur "éclatement" des données; alors que le noyau linéaire ne permet pas d'augmenter la dimension de l'espace des descripteurs. Il

permet aussi de traiter des cas non-linéaires, et dans [40] ont montré que la classification linéaire était une simple restriction de la classification RBF. Une autre raison est le faible nombre de paramètres à régler pour le RBF, qui influe dans la complexité du modèle, un atout que n'ont pas les autres noyaux.

Enfin, le RBF, de module ≤ 1, présente moins de problèmes numériques que les noyaux linéaires et polynomiaux et donc l'avantage principal de ce type de noyau est la rapidité de la phase d'apprentissage. Cependant il y a des situations ou l'utilisation du noyau RBF n'est pas nécessaire. En particulier quand le nombre de caractéristiques est très grand, dans ce cas on peut utiliser juste le noyau linéaire.

La figure II.12 montre des exemples de frontières de décision obtenues avec différents types de noyaux :

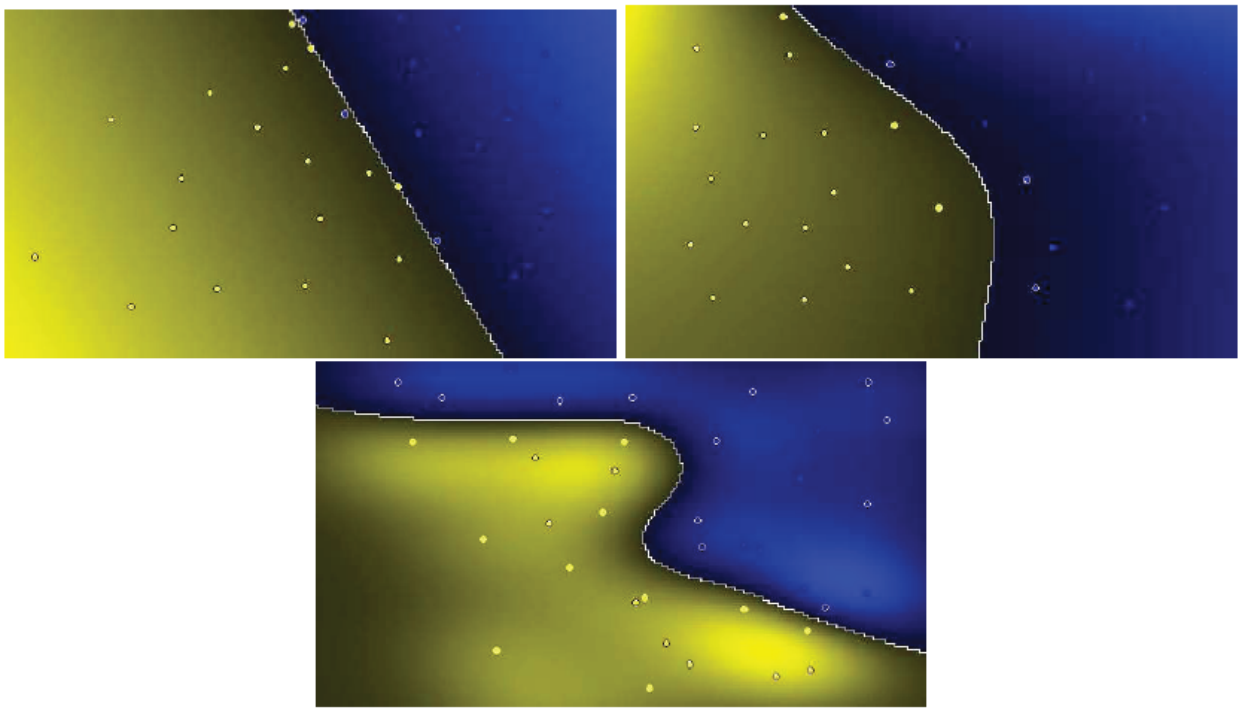


FIG II.12. Frontières de décision obtenues par trois SVM à noyau linéaire, polynomial et RBF.

A la section suivante, nous présentons un noyau considéré comme standard. Nous verrons la nature du *feature space* est un peu particulière.

II-3-3-3-2- Le noyau RBF (Radial Basis Function) :

La forme générique de ce noyau est : $k(x,z) = \exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)$ (II.18)

Cette fonction noyau est une exponentielle du carré de la distance euclidienne entre les vecteurs, divisé par sigma au carré. C'est la formule d'une gaussienne dans l'espace des caractéristiques, où sigma est la l'écart type de la gaussienne, c'est est un paramètre de l'apprentissage qui détermine les propriétés de la solution.

Le mapping ϕ induit par ce type de noyau est un peu spécial. En fait, une donnée va être mappée sur une fonction gaussienne représentant la similarité de la donnée avec toutes les données de X. La figure II.13 représente l'application du noyau RBF à deux données x et x'.

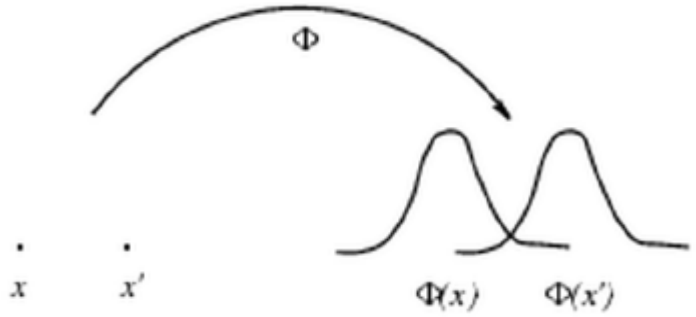


FIG II.13. Deux données mappées dans un *feature space* de dimension infinie.

L'espace d'arrivée *F* (le feature space connu comme espace de Hilbert) de la fonction ϕ est de dimension infinie étant donné que ϕ fait correspondre une fonction continue à chaque donnée, le noyau RBF permet donc de calculer des similarités dans un espace de dimension infinie !

Avec le noyau RBF, toutes les données sont placées sur une hyper-sphère de rayon 1 :

$$k(x,x) = \exp\left(-\frac{\|x-x\|^2}{2\sigma^2}\right) = 1 \quad \text{(II.19)}$$

Le paramètre σ permet de régler la largeur de la gaussienne. En prenant un σ grand, la similarité d'une donnée par rapport à ceux qui l'entoure sera assez élevée, alors qu'en prenant un

σ tendant vers 0, la donnée ne sera similaire à aucune autre (voir figure II.14). En resserrant fortement la gaussienne, un classificateur (faisant usage de ce noyau) peut arriver à apprendre n'importe quel ensemble d'entrainement sans commettre d'erreur. On sent tout de suite que le danger de l'apprentissage par cœur n'est pas loin. En fait, σ est un paramètre permettant de contrôler le pouvoir de généralisation d'un classificateur. La valeur du paramètre noyau affecte directement la complexité de la frontière de décision du classifieur.

FIG II.14. La similarité des données vue en utilisant un noyau RBF à σ tendant vers 0.

Pour connaître le domaine de variation de σ (avec : $\sigma \in [0, \sigma_{\max}]$), on utilise la distance maximale d_{max} euclidienne qui sépare deux vecteurs $\phi(x_i)$ dans l'espace transformé de rayon unité, qui est égale à :

$$d_{\max} = \left\| \phi(x_i) - \phi(x_j) \right\|^2_{\max} = 2\sigma^2_{\max} \tag{II.20}$$

Sachant que : $\left\| \phi(x_i) - \phi(x_j) \right\|_{\max} = \sqrt{2}$ Alors : $\sigma_{\max} = \sqrt{2} = 1.4$

On note qu'utiliser un noyau à base radiale avec un sigma très petit donne un nombre de vecteurs support attendu élevé et ainsi une erreur de généralisation attendue élevée. Et encore, un noyau à base radiale avec un sigma grand, bien que de dimensionnalité similaire, à peu de vecteurs support attendus et est à même de mieux généraliser. Nous ne devrions pas prendre cette limite trop au sérieux. Ce n'est pas en effet aussi prédictif des performances de généralisation en pratique mais cela montre une propriété importante des SVM, la performance de généralisation est plus liée au nombre de vecteurs supports attendus qu'à la dimensionnalité de l'espace des caractéristiques transformées.

II-3-3-4- Formulation marge douce (*Soft Margin*) :

Il reste cependant un problème : nous avons toujours conservé l'hypothèse que les données sont linéairement séparables. Notons que dans ce cadre, la notion de linéairement séparable s'applique dans le *feature space*. La plupart du temps, les données d'entrainement comportent du bruit (erreur d'acquisition, erreur sur la catégorie ...). Par conséquent, même s'il existe une relation linéaire (dans *F*) entre les données et leur catégorie, un classificateur linéaire pourrait commettre des erreurs.

On pourrait trouver un *feature space* induit par un noyau RBF suffisamment resserré qui rendrait les données linéairement séparables. Cependant, cela reviendrait à apprendre le bruit des données et donc à perdre une grande partie du pouvoir de généralisation. Au lieu de cela, il parait plus raisonnable d'admettre que certaines données (supposées bruitées) soient mal classées par notre classificateur. On appelle souvent ces données des *outliers*.

Du point de vue de notre problème primal, cela revient à relaxer la contrainte imposant que toutes les données soient bien classées. Pour ce faire, on va introduire ce que l'on appelle des *variables d'écart*.

$$\begin{aligned} y_i(\langle w, x_i\rangle + b) &\geq 1 - \xi_i \quad \forall i = 1 \ldots m \\ \xi_i &\geq 0 \qquad \forall i = 1 \ldots m \end{aligned} \tag{II.21}$$

Bien entendu, si les valeurs des ξ_i sont assez grandes, les contraintes (II.21) seront vérifiées quelque soit l'hyperplan, on peut minimiser $\|w\|^2$ jusqu'à le rendre nul. Il est donc nécessaire de pénaliser les valeurs fortes de ξ_i en introduisant le terme $\sum_{i=1}^{m} \xi_i$ dans la fonction objective (II.6). Le SVM comporte principalement deux variantes : SVM *L1* et SVM *L2*. La première applique une pénalisation quasi-linéaire sur l'erreur d'apprentissage alors que la deuxième opère une pénalisation quadratique de l'erreur. Dans ce mémoire, nous avons opté pour la variante *L1*.

$$\text{Minimiser} \quad \frac{1}{2} k(w, w) + C \sum_{i=1}^{m} \xi_i \tag{II.22}$$

Le paramètre de régularisation[2] C est souvent appelée la constante de *trade-off* (compromis), parce qu'elle permet d'indiquer l'importance que nous accordons aux erreurs commises sur le training set par rapport au fait de maximiser la marge. Si on sait que les données d'apprentissage sont très bruitées, on accordera davantage d'importance à la marge en utilisant un C petit. Par contre, si l'intérêt se porte plutôt sur les résultats obtenus sur le training set, on utilisera un C de grande valeur. Plus est grand, plus nous pénalisons les mauvaises classifications et la complexité de la classe des fonctions de décision sera plus grande.

Géométriquement, la variable d'écart ξ_i divisée par $\|w\|$, correspond à la distance euclidienne prise perpendiculairement entre l'hyperplan canonique du coté de la catégorie de la donnée et cette donnée. Notons que pour les données correctement classées, ξ_i est nul.

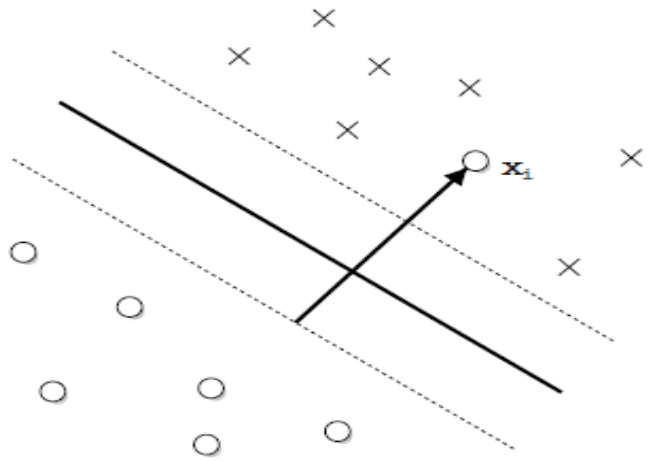


FIG II.15. La distance séparant un *outlier* et l'hyperplan canonique est : $\frac{\xi_i}{\|w\|}$.

La formulation du problème que nous avons présenté est souvent reprise sous la dénomination de *marge douce* (*soft margin*) dans la littérature. Pour les mêmes raisons qu'en la section (§II-3-3-2-6), il est intéressant de dualiser le problème QP1. Reprenons le primal :

QP1 : Minimiser $W(w,b) = \frac{1}{2}k(w,w) + C\sum_{i=1}^{m}\xi_i$ (II.23)

Tel que $\begin{cases} y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i & \forall i = 1...m \\ \xi_i \geq 0 & \forall i = 1...m \end{cases}$

[2] Le fait que l'algorithme SVM est directement la solution d'un problème régularisé [5]

En formant le Lagrangien, puis en appliquant les *conditions KKT*, on trouve la même forme duale à l'exception d'une constante *C*, la borne supérieure sur les α_i :

QP4 : Maximiser $W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j}^{m} y_i y_j \alpha_i \alpha_j k(x_i, x_j)$ (II.24)

Tel que $\begin{cases} \sum_{i=1}^{m} \alpha_i y_i = 0 \\ C \geq \alpha_i \geq 0 \quad \forall i = 1...m \end{cases}$

La constante *C* est maintenant un paramètre de l'algorithme. Il faut donc en pratique un moyen de le choisir de façon adéquate (très souvent, nous utilisons la cross-validation ou des approximations de celle-ci). Le seul inconvénient dans la version ***C*-SVM** est le domaine de variation du paramètre *C*, car ce dernier n'as pas de signification réelle. Le plus souvent, on fait varier *C* de 0.1 à 10000 par multiple de 10 (ou de 100 pour aller plus vite), puis affiner ensuite dans les zones intéressantes. Nous terminons en mentionnant que l'on peut déterminer le statut d'une donnée x_i en regardant sa variable duale α_i. Il existe trois statuts différents :

$\alpha_i = 0 \quad \Rightarrow \quad y_i f(x_i) > 1 \quad et \quad \xi_i = 0$, On dira que x_i est un non - VS.
$0 < \alpha_i < C \Rightarrow \quad y_i f(x_i) = 1 \quad et \quad \xi_i = 0$, Il s'agit donc d'un VS.
$\alpha_i = C \quad \Rightarrow \quad y_i f(x_i) < 1 \quad et \quad \xi_i > 0$, Il s'agit d'un outlier VS.

Ainsi, les seuls coefficients α_i non-nuls sont ceux associés aux échantillons x_i de l'ensemble d'apprentissage qui sont sur la marge (i.e. $0 < \alpha_i < C \; et \; y_i f(x_i) = 1$) ou à l'intérieur de l'espace délimité par la marge (i.e. $\alpha_i = C \; et \; y_i f(x_i) < 1$). Ces échantillons sont appelés vecteurs de support.

II-3-3-5- Effet des hyper-paramètres sur le modèle SVM :

De manière à essayer de mieux comprendre l'effet du paramètre noyau et celui de régularisation *C*, nous avons visualisé les frontières de décision que construisent les SVM à

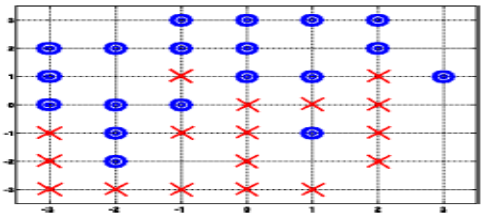
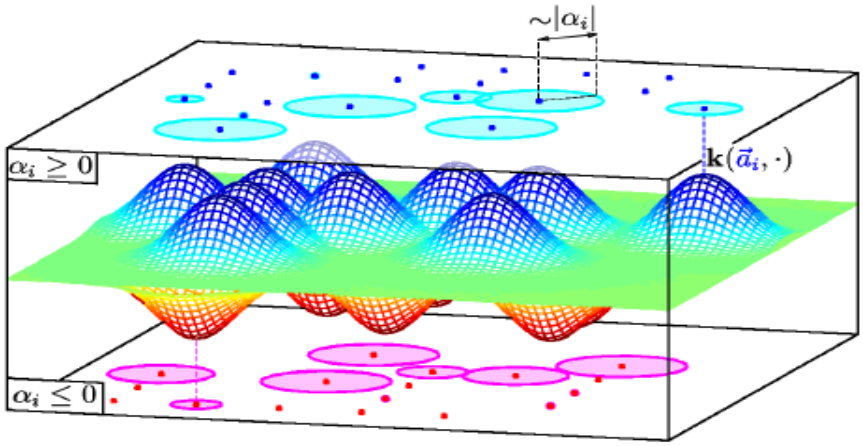
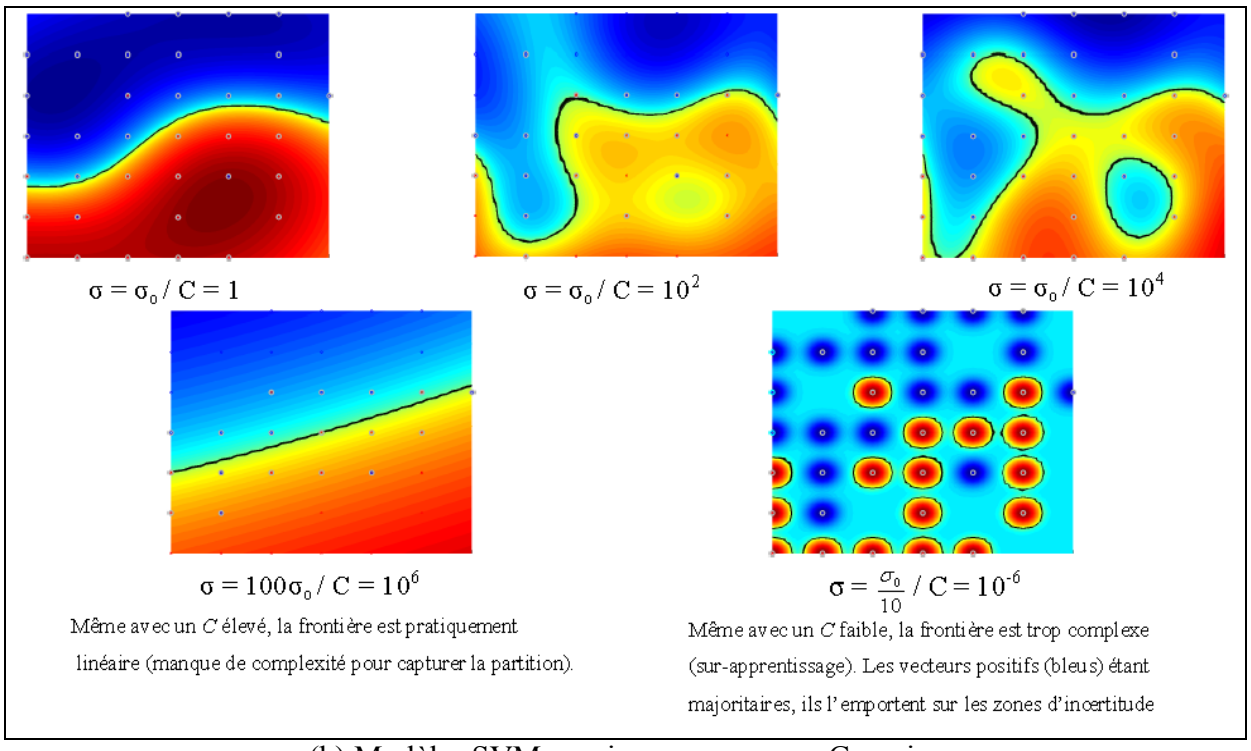


FIG II.16. Disposition des vecteurs 2D d'entrée pour les SVM.

noyau gaussien sur un cas bidimensionnel représenté en figure II.16. Les valeurs des fonctions discriminantes apprises sont représentées par les intensités de couleur (voir figure II.17 (b)), et la frontière de décision est représentée par une courbe noire.



(a) Fonctions "représentantes" $k(a_i, .)$ et illustration des coefficients de Lagrange α_i associes, correspondant aux Vecteurs de Support, avec $\sigma = \sigma_o$= cte / $C = 10^2$.



(b) Modèles SVM appris avec un noyau Gaussien.

FIG II.17. Modèles SVM vectoriels : noyaux Gaussiens.

La figure II.17 représente l'allure des solutions (frontières de décisions obtenues par apprentissage SVM) obtenues avec les noyaux RBF Gaussiens. L'effet du sur-apprentissage par les noyaux radiaux lorsque la zone d'influence σ est trop faible est visible dans figure II.17 (b) (en bas à droite). Lorsqu'au contraire σ est élevé, la frontière de décision devient quasiment linéaire (même figure, en bas à gauche) situation de sous-apprentissage. Il est donc important de bien choisir le noyau qui sera utilisé et de surtout bien optimiser le ou les paramètres du noyau choisi.

Le paramètre *C* est la borne supérieure des multiplicateurs de Lagrange, il permet d'avoir des fonctions plus au moins lisses. Si on prend un *C* petit, la sortie de SVM est plutôt lisse et ne va pas suivre les points trop décalés par rapport au reste des données. Par contre si nous mettons *C*=infini, la sortie va tenter de s'approcher de tout les points (un nombre de vecteurs supports plus important). D'après la figure II.17 (b) (les deux cas en bas), nous pouvons constater que le noyau gaussien a plus d'effet sur l'allure de la fonction de décision que le paramètre de régularisation *C*. La figure II.17 (a) illustre comment la fonction discriminante est construite à partir du noyau et des vecteurs de support.

II-4- Conclusion :

Dans ce chapitre, nous avons présenté un état de l'art des principales méthodes de classification discriminante utilisées pour la reconnaissance parole/musique : les *k*-plus proches voisins, les réseaux de neurones et les SVM. Les deux premières ne seront pas utilisées dans nos travaux. Bien que performantes sur beaucoup de problèmes de classification, le défaut principal des *k*-plus proches voisins est la nécessité de stocker les données d'apprentissage. L'autre maillon faible des réseaux de neurones, est leur sensibilité à la dimensionnalité des données. Le SVM procure un cadre théorique robuste qui résout ces problèmes puisque il produit un modèle clairsemé avec une insensibilité à la dimensionnalité des données.

Les SVM réalisent des séparations non-linéaires dans l'espace des données d'apprentissage à partir de séparations linéaires dans un espace transformé de dimension potentiellement grande, et ce grâce à l'idée des noyaux de Mercer. Depuis leur apparition les SVM ont connu beaucoup de succès sur des applications provenant de domaines très variés. Son comportement est, par ailleurs, conditionné par le paramètre du noyau σ et le paramètre de

régularisation *C*. Les valeurs de ces hyper-paramètres jouent un rôle très important dans la maximisation de la marge. La maximisation de celle-ci dans le SVM est une procédure de sélection des hyper-paramètres implicite à l'apprentissage, qui permet de minimiser l'erreur d'apprentissage tout en réduisant la complexité du classifieur. Malheureusement, les hyper-paramètres ne sont pas pris en compte pendant l'apprentissage. N'ayant pas d'information à priori sur le problème, le classifieur entraîné est loin des performances escomptées. Le cas échéant, la fonction du classifieur n'est pas adaptée aux données considérées. Dans le prochain chapitre, nous allons nous intéresser au problème de sélection optimale des hyper-paramètres SVM.

CHAPITRE III : SELECTION DES HYPER-PARAMETRES D'UN MODELE SVM

III-1- Problématique :

Bien que l'entrainement des SVM soit «fortement automatique», l'intervention de l'utilisateur est encore nécessaire pour l'adaptation des hyper-paramètres conditionnant le processus d'apprentissage. Convenablement choisies, elles permettent d'éviter les situations de sur-apprentissage fréquentes lorsque les données sont bruitées. Parmi les hyper-paramètres du SVM, on trouve les paramètres du noyau à ajuster que le classificateur en dépend fortement et le paramètre C à déterminer, qui permet de contrôler le compromis entre nombre d'erreurs de classement, et la largeur de la marge. Donc l'apprentissage comprend à la fois la détermination des paramètres optimums (α_i, b) de l'hyperplan séparateur, et le réglage des hyper-paramètres. Si la mise en ouvre d'un algorithme SVM est en général peu couteuse en temps; il faut cependant compter que la recherche des meilleurs paramètres peut requérir des phases de test assez longues.

III-2- Etat de l'art :

Comme pour la plupart des algorithmes d'apprentissage, l'utilisation de SVM nécessite de fixer au préalable certains paramètres qui ne peuvent pas être modifiés lors de l'apprentissage, cette étape préalable est appelée "sélection de modèle" [41] et ses résultats influent fortement sur les performances du classifieur. De manière à différentier ce type de paramètres, des paramètres qui sont optimisés durant l'apprentissage, nous parlons généralement d'hyper-paramètres. La constante de régularisation C en est un bon exemple. Un autre exemple est celui du choix du noyau qui sera utilisé par le SVM. En effet, non seulement il existe différents types de noyaux, mais chacun d'entre eux est généralement caractérisé par une ou plusieurs constantes (σ dans le cas d'un noyau RBF). Dans ce cas, le nombre d'hyper-paramètres est lié au type de noyau.

Dans le but de choisir convenablement les paramètres de SVM, une sélection manuelle des valeurs des hyper-paramètres doit être entreprise. Dans ce cas, il est important de pouvoir définir au préalable une zone d'intérêt dans laquelle plusieurs valeurs sont testées. Sans aucune

information à priori, celle-ci n'est pas facile à trouver. Aussi considérer une grande zone d'intérêt est très prohibitif en temps de calcul. En outre, il est nécessaire d'échantillonner cet intervalle de façon à assurer la plus grande résolution.

Si le modèle utilise un seul hyper-paramètre, il est possible d'essayer un nombre fini de valeurs et choisir celle qui minimise le critère de sélection. Cette technique devient toutefois difficile à implémenter pour deux hyper-paramètres ou plus. On s'intéresse en particulier à la distribution à Noyau Radialement Gaussien qui nécessite un seul paramètre. De plus, si ce paramètre est plus délicat à ajuster, une optimisation fine de celui-ci permet généralement d'obtenir de bons résultats.

Pendant longtemps, la sélection de modèle a été effectuée par une méthode de type "Grid Search" [4, 40], où une recherche systématique est mise en œuvre en discrétisant l'espace des paramètres à l'aide d'un pas fixe plus ou moins grand. Il a été montré que ces approches fonctionnaient mal et qu'elles étaient très gourmandes en temps de calcul. Afin de réduire les charges de calculs et d'améliorer les performances de classification, nous exploitons deux approches à noyaux.

Nous détaillerons dans la section suivante les méthodes de sélection de modèles du SVM, en commençant par la méthode traditionnelle de l'état de l'art que nous utiliserons comme méthode de référence, puis nous présenterons les deux méthodes à noyaux que nous comparerons avec la méthode de référence. Pour une étude bibliographique approfondie des différentes méthodes de sélection de modèle, le lecteur est renvoyé à la référence [51].

III-2-1- La validation croisée :

Dans le but de trouver le SVM qui généralise le mieux, le plus simple serait d'utiliser un ensemble indépendant de celui de l'apprentissage pour l'évaluation du classifieur. Plusieurs SVM peuvent donc être entraînés sur un même ensemble d'apprentissage avec toutefois différentes valeurs d'hyper-paramètres. Leurs performances sont par la suite comparées en estimant l'erreur de chacun d'eux sur un ensemble de validation indépendant de celui de l'apprentissage. En faisant plusieurs classifications SVM avec différents couples σ et C, une **matrice de confusion** sera fournie en sortie; le SVM qui fournit la plus faible erreur de classification est le plus approprié. Cette approche est aussi connue sous le nom ***«hold out method»*** **[7]**.

Afin d'éviter le sur-apprentissage des données de validation, le classifieur choisi est à son tour évalué sur un ensemble de test indépendant. Cependant en pratique, nous ne disposons pas toujours d'un nombre suffisant de données étiquetées pour s'en réserver une partie pour la sélection de modèle. Dans ce cas, on utilise ceux que l'on appelle la **Validation Croisée** (*Cross Validation*) ***K*-fold** qui consiste à diviser l'ensemble de données en K partitions distinctes de taille approximativement égale pour en utiliser $K-1$ partitions pour l'apprentissage et la dernière pour évaluer la performance du classifieur. A chaque test de validation, les paramètres (σ, C) changent suivant une grille prédéfinie des valeurs les plus probables ***Grid search method*** en anglais.

Ce processus est répété K fois pour chacun des K segments disponibles pour le test. L'erreur de test finale est la moyenne sur les tests croisés des K estimations possibles. Les paramètres SVM (σ, C) qui ont obtenu le plus faible score en moyenne, seront sauvegardés pour effectuer l'apprentissage final sur les signaux test. Cette méthode requiert K procédures d'apprentissage. Elle est relativement lente pour un grand ensemble de données mais permet d'acquérir une certaine robustesse dans le choix des paramètres de la méthode SVM. Le cas limite où K est égal à la taille m de l'ensemble de données D est appelé "*LOO*" (de l'anglais "*leave-one-out*"), dans ce cas, l'estimation de l'erreur est la plus fidèle possible de l'erreur de généralisation. Cependant la procédure (*LOO*) est couteuse en calcul, car nécessitant m apprentissage. Donc elle est bien adaptée seulement pour des données en faible nombre de m.

Ex : pour K=4 :

FIG III.1. Principe de la Validation Croisée (VC) avec K=4.

– Avantage de la validation croisée :

- Tous les échantillons disponibles sont pris en compte pour la classification.

- Presque tous les échantillons disponibles sont pris en compte pour l'apprentissage (si K>>1).
- Aucun couplage entre les échantillons utilisés pour l'apprentissage et ceux utilisés pour la classification → **pas de sur-apprentissage (en principe)**.

– Inconvénients de la validation croisée :

- Temps de calcul important si K>>1.
- Devient non applicable pour un très grand nombre de paramètres.

– **Peut être intégré dans un système automatique (ex : SVM) :**

– Un seul paramètre à définir : K.

– En pratique, K varie le plus souvent entre 4 et 10.

Dans [8] ont montré que la procédure validation croisée k-fold permet un réglage des hyper-paramètres de la méthode SVM et donne une estimation de l'erreur de généralisation la plus fidèle possible. Ces techniques (dites de "ré-échantillonnage") permettent en fait d'estimer le pouvoir de généralisation d'un certain type d'architecture de modèle, plutôt que d'un modèle particulier, leur principe est purement empirique. Un défaut évident de ces techniques est leur consommation très importante du temps de calcul.

III-2-2- Critère d'Alignement de noyaux :

On exploite le critère d'alignement développé pour la sélection du paramètre noyau reproduisant dans le cadre des méthodes à noyaux pour un problème à deux classes, et étant donné un ensemble de m données d'apprentissage et leurs étiquettes associés avec : $D = \{(x_1, y_1),(x_2, y_2),...(x_m, y_m)\}$. L'apprentissage statistique inductif permet d'exploiter la relation existante entre la distribution géométrique des données et celle des étiquettes de classes. Dans ce cadre, il est aussi démontré que la matrice de Gram K ou matrice noyau, symétrique semi-définie positive permet de caractériser les ressemblances (ou dissemblances) entre les paires de données d'entrée en rapport avec la métrique induite par une fonction noyau (un exponentiel d'une distance euclidienne entre les données). Chaque terme k_{ij} est le résultat de l'application d'une fonction noyau à la paire de donnée (x_i, x_j). D'autre part, si nous considérons que les

étiquettes de classes notées (± 1) sont contenues dans un vecteur colonne y, nous pouvons définir la matrice $Y = y \times y^T$. Y est une matrice carrée symétrique de même dimension que la matrice K telle que : $y_{ij} = +1$ *si* $y_i = y_j$ *et* $y_{ij} = -1$ *si* $y_i \neq y_j$. Ainsi Y résume toute l'information véhiculée par les données d'apprentissage représentées par leurs étiquettes. Puisque K représente la similarité géométrique des données et Y représente la similarité des étiquettes de ces données, il est raisonnable de penser que ces deux matrices présentent aussi des ressemblances entre elles. Cette propriété est alors mesurée et quantifiée par le critère d'alignement. D'autres critères existent, nous citons par exemple le *class separability criterion* [6] et le *data centering criterion* [42].

Le critère d'alignement n'est en fait qu'une généralisation aux matrices, du produit scalaire normalisé de deux vecteurs (x, y). La mesure de similarité entre deux vecteurs est souvent utilisée en mathématique comme le cosinus de l'angle entre deux vecteurs qui moyennant une normalisation du produit scalaire :

$$\cos(\varphi) = \frac{x^T y}{\sqrt{x^T x * y^T y}} \tag{III.1}$$

Dans le cas de deux matrices A et B cette généralisation traduit le critère d'alignement qui s'écrit :

$$\mathrm{A}(A,B) = \frac{\langle A,B\rangle_F}{\sqrt{\langle A,A\rangle_F} * \sqrt{\langle B,B\rangle_F}} \tag{III.2}$$

où $\langle .,. \rangle_F$ et $\|.\|_F$ sont respectivement le produit scalaire généralisé de Frobénius et la norme de Frobénius. Ainsi la formule précédente de l'alignement se réécrit comme :

$$\mathrm{A}(A,B) = \frac{\langle A,B\rangle_F}{\left(\|A\|_F * \|B\|_F\right)} \tag{III.3}$$

Pour un couple de matrices A et B de taille identique, nous pouvons écrire le produit généralisé de Frobenius :

$$\langle A,B\rangle_F = \sum_i \sum_j a_i b_j \tag{III.4}$$

De ce fait pour mesurer l'alignement entre la matrice des données K et la matrice Y des données d'apprentissage étiquetées, nous évaluons la quantité $A(Y,K)$ telle que définie par l'expression (III.2) ou (III.3) :

$$A(Y,K) = \frac{\langle Y,K \rangle_F}{\|Y\|_F * \|K\|_F} \tag{III.5}$$

Plusieurs utilisations ont été proposées pour la mesure d'alignement de noyaux dans les deux cas d'apprentissages supervisé et non supervisé, par la suite on s'intéresse au premier apprentissage puisque les classes de l'ensemble d'apprentissage sont connues.

Dans [12], les auteurs proposent de quantifier la qualité *à priori* de l'espace de représentation F à l'aide de la mesure d'*alignement* avec :

$$-1 \leq A(Y, K) \leq 1 \tag{III.6}$$

Pour une fonction noyau $(K(x_i,x_j))_{i,j=1}^m$ symétrique définie positive :

$$0 \leq A(Y, K) \leq 1 \tag{III.7}$$

Maintenant si on développe l'expression de l'alignement, on trouve :

$$A(Y,K) = \frac{\langle K, yy^T \rangle_F}{\sqrt{(\langle K,K \rangle_F * \langle yy^T, yy^T \rangle_F)}} = \frac{y^T K\ y}{m\sqrt{\langle K,K \rangle}} = \frac{\sum_{i,j=1}^m y_i y_j K(x_i,x_j)}{m\sqrt{(\sum_{i,j=1}^m K(x_i,x_j)^2)}} \tag{III.8}$$

La paramétrisation optimale est obtenue par maximisation de l'alignement, à savoir $\sigma = \arg\ \max_\sigma A(Y,K)$, ce qui correspond à la maximisation du numérateur (*A(Y,K)* sans normalisation), sous contraintes que son dénominateur soit constant. On peut alors écrire le problème d'optimisation suivant :

$$\max_\sigma \sum_{i,j=1}^m y_i y_j K(x_i,x_j),$$

$$\langle Y,K \rangle = \sum_{y_i=y_j} K(x_i,x_j) - \sum_{y_i \neq y_j} K(x_i,x_j) \tag{III.9}$$

On voit bien la correspondance :

– Si x_1 est similaire à x_2 $(yy^T = +1)$ $\Rightarrow \|x_1 - x_2\|^2$ petit $\Rightarrow K(x_i,x_j) \uparrow$.

– Si x_1 est différent de x_2 $(yy^T = -1) \Rightarrow \|x_1 - x_2\|^2$ grand $\Rightarrow K(x_i,x_j) \downarrow$.

Donc $\langle yy^T, K \rangle$ est grande si la mesure de similarité introduite par le noyau est grande pour les données de même classe et petite pour les données de différentes classes.

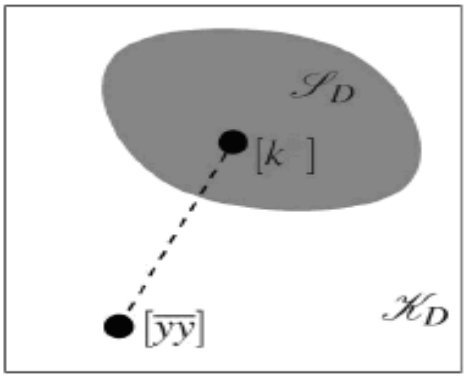


FIG III.2. Schéma de sélection de modèle basé sur le critère d'alignement de noyaux.

Le but est de trouver le noyau K à l'intérieur d'une famille restreinte de fonctions noyaux φ_D dans laquelle la matrice de gram normalisée introduit par K a la plus petite distance avec la matrice des étiquettes normalisée empirique yy^t, voir la figure III.2. Il a été montré dans [12] que si le noyau K est très aligné avec l'information des étiquettes Y, alors il existe une bonne séparation entre les données des différentes classes avec une faible borne de l'erreur de généralisation. Cette dernière est expliquée dans la section III-2-3-1.

Dans [12], les auteurs proposent de maximiser l'indice d'alignement pour le réglage de la largeur du noyau gaussien (un usage lequel a été indiqué dans la littérature mais n'as pas été reporté en détails) et un nouvel usage possible pour la sélection de l'ensemble de caractéristiques [15]. Le critère d'alignement repose sur un fondement purement théorique (Heuristique).

Nous calculons simplement l'alignement du noyau avec les données d'entrainement pour une gamme de valeurs du paramètre σ. L'allure qui en résulte à la forme d'une gaussienne (voir [15] appliquée sur le jeu de donnée "Breast-Cancer Wisconsin" de la base *UCI*), en général les résultats sont similaires pour d'autres ensembles de données. La forme de l'allure contribue à la robustesse de la méthode car l'alignement admet un seul maximum global.

La complexité (temps de calcul) de l'algorithme du critère d'alignement dépend de façon importante de la matrice du noyau RBF. Elle évolue en O (m^2), où m est le nombre de points de données. Ces problèmes de temps de calcul sont d'autant plus gênants lorsqu'on effectue un apprentissage en temps réel.

Pour faciliter son implémentation et sa vitesse d'exécution, on peut exploiter les propriétés de la matrice de Gram. Sachant que la matrice est symétrique, donc on peut calculer que le triangle supérieur ou le triangle inférieur de cette matrice K, c'est-à-dire seulement pour une moitie de la matrice de gram. On peut aussi initialiser la diagonale ce cette matrice à 1, elle représente le maximum de similarité $K_{ij} = 1$ puisque c'est une mesure de distance entre un vecteur avec lui même. Finalement, on évite le problème de stockage de la matrice de Gram qui devient impossible pour un nombre élevé de données (*Out of memory*). Par exemple pour 30000 données de format double précision nous aurons ($30000^2.8/2$=3.6GB) de RAM pour stocker K.

Le gain de temps ne sera peut être pas extraordinaire avec une grosse matrice de Gram, nous proposons alors de simplifier la formule de noyau gaussien (RBF).

$$K(x_i, x_j) = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right) \tag{III.10}$$

Nous utilisons $r = \|x_i - x_j\|$ pour indiquer la distance euclidienne entre les vecteurs x_i et x_j, et écrire la fonction noyau comme $k(r) = \exp\left(\frac{-r^2}{2\sigma^2}\right)$. Ce dernier calcule la norme pour ensuite l'élever au carré. Hors ici, la norme est calculée avec une racine carrée. Donc on peut remplacer directement cette opération par la somme des termes au carré :

$$r_{ij}^2 = \|x_i - x_j\|^2 \tag{III.11}$$

$$= (x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{in} - x_{jn})^2$$

Si n est très grand, nous utilisons le principe de la méthode [43] qui consiste à partitionner la matrice des données X, comme suit :

$$X = (X^1 \mid X^2 \mid ... \mid X^p) \tag{III.12}$$

Chaque partition X^k est de dimension $m \times l$, ou $l = \frac{n}{p}$ et p est le nombre total des nœuds de traitement. Nous supposons l est un entier. La i^{eme} ligne de X^k est : $X_i^k = (X_{i1}^k, X_{i2}^k, ..., X_{il}^k)$. L'équation (III.11) devient :

$$r_{ij}^2 = (X_{i1}^1 - X_{j1}^1)^2 + ... + (X_{il}^1 - X_{jl}^1)^2$$
$$+ (X_{i1}^2 - X_{j1}^2)^2 + ... + (X_{il}^2 - X_{jl}^2)^2 +$$
$$... + (X_{i1}^p - X_{j1}^p)^2 + ... + (X_{il}^p - X_{jl}^p)^2$$

$$r_{ij}^2 = \sum_{k=1}^{p} N_{ij}^k \tag{III.13}$$

où N_{ij}^k est la distance au carrée partielle calculée pour le k^{ieme} nœud,

$$N_{ij}^k = \sum_{d=1}^{l} (X_{id}^k - X_{jd}^k)^2 \tag{III.14}$$

Alors la fonction noyau peut être réécrite comme suit :

$$K(x_i, x_j) = \exp\left(\frac{-\sum_{k=1}^{p} N_{ij}^k}{2\sigma^2} \right)$$

$$= \prod_{k=1}^{p} \exp\left(\frac{-N_{ij}^k}{2\sigma^2} \right) = \prod_{k=1}^{p} K_{ij}^{(k)} \tag{III.15}$$

Où $K_{ij}^{(k)}$ est l'élément calculé par le nœud k, et la multiplication est élément par élément (*the Schur product*). Le gain devrait être très significatif avec toutes ces améliorations.

III-2-3- Critère combinant borne *Rayon-Marge* et mesure de séparabilité des classes à noyau :

Cette approche proposée dans [14], combine deux critères Q et Q', l'un est basé sur la minimisation de la borne *Rayon-Marge* et l'autre sur la maximisation de la mesure de séparabilité des classes dans l'espace transformé, pour former un nouveau critère défini :

$$Q^* = Q / Q' \tag{III.16}$$

Q est dérivé de la borne supérieure de l'erreur de généralisation pour les classifieurs linéaires à vaste marge [44]. Elle représente un bon compromis entre qualité et temps de calcul. Plus précisément, elle s'avère presque aussi efficace pour déterminer la valeur de σ, tandis que les calculs qu'elle nécessite sont moins lourds. La formulation de cette borne fait intervenir une SVM bi-classe à marge dure.

Q' est la mesure de séparabilité des classes dans l'espace de caractéristiques de plus grande dimension, elle emploi les deux traces de la matrice de covariance inter-classe S_b et de la matrice de covariance intra-classe S_w dans cet espace. Dans ce qui suit, nous développerons Q et Q'.

III-2-3-1- Borne sur l'erreur de généralisation pour les SVM :

La sélection des paramètres noyaux est un problème ouvert dans l'entraînement des méthodes à vecteurs de support non linéaire. Quel que soit le noyau, le paramètre devrait être choisi comme une minimisation de l'erreur de généralisation. Les performances prédictives d'un modèle SVM sont souvent évaluées par des bornes de risque [32] majorant ses taux d'erreurs. L'estimateur du taux d'erreur utilisé pour les bornes de risque des SVM est celui obtenu par *leave-one-out* (Validation croisée ou les échantillons tests sont constitués d'une seule observation à la fois), non biaisée. Les plus connues de ces bornes sont basées sur la *marge* de l'hyperplan séparateur. Dans [32] Vapnik a fourni la borne supérieure suivante :

$$\varepsilon \leq \frac{1}{m}\frac{R^2}{\gamma^2} = \frac{1}{m} R^2 \|w\|^2 \tag{III.17}$$

où ε est le nombre d'observations mal classées par leave-one-out (LOO), γ est la largeur de la marge interclasses, m est le nombre de réitérations d'apprentissage de la méthode (LOO) et R le rayon de la plus petite sphère englobant l'ensemble des échantillons d'apprentissage (vecteurs supports) dans l'espace transformé.

Au fait dans le cas des C-SVM, où les données sont non séparables dans l'espace des caractéristiques, la borne supérieure de l'erreur de généralisation est :

$$\varepsilon \leq \frac{1}{m}\frac{(R^2 + \sqrt{\sum \xi^2})^2}{\gamma^2} \tag{III.18}$$

Dans le cas d'une SVM séparant sans erreurs tous les points de l'échantillon d'apprentissage avec une marge γ, avec une seule partition (m=1) apprentissage et test, nous se retrouvons avec la borne suivante :

$$\varepsilon \leq R^2 \|w\|^2 \tag{III.19}$$

Nous avons vu d'après l'équation (II.17) que la solution SVM est un hyperplan avec :

$$f(x) = \sum_{i=1}^{m} \alpha_i y_i k(x, x_i) + b \tag{III.20}$$

où α_i sont des coefficients de Lagrange non négatifs, ils sont obtenus en utilisant la programmation quadratique en maximisant $1/|w|^2$, et b est choisi tels que l'hyperplan se trouve dans le centre de la marge. La norme du vecteur poids est :

$$|w|^2 = \sum_{i,j}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \tag{III.21}$$

Étant donné, dans l'entrainement SVM, il a été montré dans [3, 44] que :

$$\gamma^2 = |w|^{-2} = \left(\sum_{i}^{m} \alpha_i \right)^{-1} \tag{III.22}$$

Bien que l'entrainement SVM soit presque entièrement automatique, l'intervention de l'utilisateur est encore nécessaire dans la sélection du paramètre noyau, par exemple σ dans le noyau gaussien et d dans le noyau polynomial. La sélection est généralement basée sur le critère Q qui est le critère courant et recommandé pour la sélection de modèle (quotient à minimiser pour lequel correspond le meilleur paramètre du noyau) :

$$Q = R^2 |w|^2 \tag{III.23}$$

Le calcul de $|w|^2$ est de complexité $O(V)$; V étant le nombre de vecteurs de support.

En général, R peut être calculé exactement en utilisant la programmation quadratique similaire à celui correspondant à l'apprentissage d'une SVM. Cependant, comme indiqué dans [45], nous pouvons obtenir la borne supérieure en notant que les noyaux gaussiens projettent toujours les données d'entrées sur la surface d'une hypersphère de rayon unité dans l'espace des caractéristiques. En d'autres termes, tous les vecteurs de caractéristiques sont de norme unitaire, depuis :

$$|\phi(x)|^2 = \phi(x).\phi(x) = k(x,x) = \exp\left(\frac{-|x-x|^2}{2\sigma^2} \right) = 1 \tag{III.24}$$

D'ici, pour les noyaux gaussiens, nous approximons $R^2 = 1$. Dans [45] ont montré que la courbe de la fonction Q en fonction du noyau utilisant cette approximation suit la courbe réelle (avec programmation quadratique).

D'après les études empiriques [14] faites sur des ensembles de données réels de la base de données *UCI*, utilisant les deux noyaux gaussien et polynomial, l'erreur de test due à ce critère Q est souvent beaucoup plus grande que l'erreur de test minimum (voir figure III.3). En d'autre terme ce critère peut être sous-optimal ou encore inadéquat. Cela à amener à explorer les façons d'améliorer le critère afin que nous puissions prédire mieux le paramètre noyau optimal.

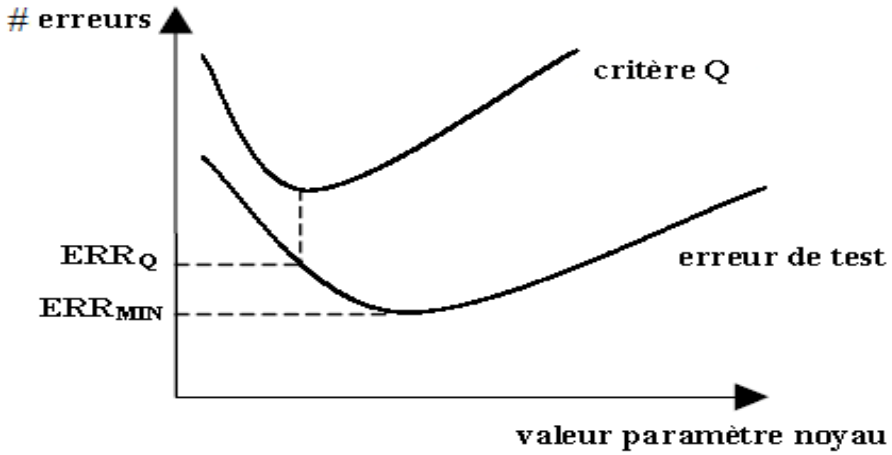


FIG III.3. Illustration de l'obtention de ERR_{MIN} et ERR_Q.

III-2-3-2- Séparabilité des classes à noyau (*Kernel Class Separability*) :

D'ici est venue l'idée d'améliorer le critère Q en intégrant la mesure de séparabilité des classes utilisée dans la sélection des caractéristiques [16], connu également sous l'appellation anglophone KCS(*Kernel Class Separability*). Exemples de telles mesures sont la divergence des classes, la distance de Bhattacharyya, et plusieurs autres mesures basées sur les matrices de dispersion. La mesure de séparabilité des classes appartient à ce dernier type de mesures, elle emploi la matrice de covariance inter-classe et la matrice de covariance intra-classe (les deux matrices sont développées dans (§IV-3-1)) dans l'espace des caractéristiques de plus grande dimension.

Le calcul de la matrice de covariance n'est pas faisable dans le contexte des méthodes à vecteurs de support à cause de la dimension de l'espace des caractéristiques qui est généralement

très grande ou peut être infinie. Néanmoins on peut calculer les traces des deux matrices S_b et S_w via la fonction noyau et par conséquent trouver la mesure de séparabilité des classes :

$$\overline{Q} = \frac{tr(S_b)}{tr(S_w)} \tag{III.25}$$

où S_b est la matrice de dispersion inter-classe et S_w est la matrice de dispersion intra-classe. Le principe du ratio $\overline{Q}$ consiste à maximiser la variance inter-classe et minimiser la variance intra-classe. Pour deux classes, la trace de la matrice de dispersion inter-classe est équivalente à la distance euclidienne au carrée entre les moyennes des classes :

$$D^2 = \left|\mu_+ - \mu_-\right|^2 \tag{III.26}$$

$$= \left(\frac{1}{m_+} \sum_i^+ \phi(x_i) - \frac{1}{m_-} \sum_i^- \phi(x_i) \right)^2$$

où m_+ est le nombre d'échantillons avec la classe d'étiquettes +1; de même pour m_-.

En développant ceci (annexe **A.1**) et en remplaçant tous les produits internes par des fonctions noyaux, nous obtenons :

$$D^2 = \frac{1}{m_+^{\,2}} \sum_{i,j}^+ k_{ij} + \frac{1}{m_-^{\,2}} \sum_{i,j}^- k_{ij} - \frac{2}{m_+ m_-} \sum_{i,j}^{y_i \neq y_j} k_{ij} \tag{III.27}$$

où $k_{ij} = k(x_i, x_j)$.

La trace de la matrice de dispersion intra-classe est la somme des moyennes des variances (distances au carrée) des classes :

$$V^2 = \pi_+ E_+(\left|\phi(x) - \mu_+\right|^2) + \pi_- E_-(\left|\phi(x) - \mu_-\right|^2) \tag{III.28}$$

où π_+ et π_- sont les probabilités à priori des classes C_i. Nous estimons chaque probabilité comme le rapport du nombre d'échantillons dans chaque classe sur le nombre total d'échantillons d'apprentissage. Comme précédemment, nous développons V^2 (annexe **A.2**) et nous remplaçons tout les produits internes par les fonctions noyaux afin d'obtenir :

$$V^2 = \frac{1}{m} \left(\sum_i^m k_{ii} - \frac{1}{m_+} \sum_{i,j}^+ k_{ij} - \frac{1}{m_-} \sum_{i,j}^- k_{ij} \right) \tag{III.29}$$

La mesure de séparabilité des classes à maximiser est alors :

$$Q' = D^2 / V^2 \tag{III.30}$$

D'après les études empiriques [14] faites sur la base de données *UCI*, Q' est une fonction monotone (augmente ou décroît avec le paramètre noyau). D'ici, Q' n'est pas un critère approprié pour sélectionner le paramètre noyau optimal.

Bien que le critère original Q est basé sur la minimisation de la borne "rayon-marge", il ignore la distribution des classes; alors que Q' emploi la simple distribution statistique en maximisant la séparabilité des classes. Par conséquent, nous combinons Q et Q' pour former un nouveau critère comme suit :

$$Q^* = \frac{Q}{Q'} = \frac{R^2 |w|^2 V^2}{D^2} \tag{III.31}$$

Avec la minimisation de Q^* en fonction du paramètre noyau, nous sommes capables d'obtenir de meilleures généralisations, c'est-à-dire que l'erreur de test due à ce critère doit être réduite par rapport à l'erreur de test due au critère Q.

Nous remarquons d'après l'équation (III.31) que l'adaptation du paramètre noyau nécessite un apprentissage de la statistique de décision, ce qui laisse à penser que l'on puisse adapter le paramètre C en utilisant ce critère. Hélas la formulation de la borne *rayon-marge* fait intervenir une SVM bi-classe à marge dure. Nous avons attribué une grande valeur à C, pour trouver le paramètre noyau gaussien.

III-2-4- Choix du paramètre *C* :

Nous avons vu que le critère combinant la borne supérieure *rayon-marge* avec la mesure de séparabilité des classes dans l'espace transformé ne permettait pas de trouver le paramètre C_{opt}. Ce dernier ne peut pas non plus être optimisé en utilisant le critère d'alignement qui est indépendant de l'algorithme d'apprentissage appliqué dans l'espace transformé.

Par conséquent, l'adaptation des hyper-paramètres (σ_{opt}, C_{opt}) pour les deux méthodes à noyaux est effectuée en deux étapes :

1. Optimisation du paramètre noyau par $A_{\max}(Y,K)$ ou Q^*_{min} sur toute les données de l'entrainement et
2. Adaptation du paramètre de régularisation C par minimisation de l'estimation de l'erreur de classification par validation croisée utilisant la grille de recherche (Grid-Search). La

procédure générale de la méthode grille de recherche est seulement pour une dimension, c'est-à-dire une recherche ligne (*Line-Search*). L'adaptation du paramètre noyau est complètement découplée de la grille de recherche.

III-3- Conclusion :

Dans ce chapitre, nous discutons de la problématique de sélection de modèle de type méthodes à vecteurs supports (SVM). La validation croisée de type *grid search* est la méthode de l'état de l'art dédiée traditionnellement à cette tâche. Son inconvénient majeur est que la recherche des valeurs adéquates des hyper-paramètres (σ, C) inhérents à la méthode peut requérir des phases de développement assez longues afin d'obtenir un système de classification efficient, dans le sens d'une erreur de généralisation minimale. En outre, la sélection de modèle automatique nécessite une procédure d'optimisation qui minimise ou maximise un critère de performance.

Aujourd'hui encore, peu de travaux exploitent les méthodes à noyaux pour trouver les hyper-paramètres adaptés pour une tache de classification donnée. Nous avons analysé ici deux approches pour le réglage du paramètre de noyau gaussien, basées respectivement sur la maximisation du critère d'alignement de noyaux et sur la minimisation du critère combinant la borne "Rayon-Marge" avec le critère de séparabilité des classes dans l'espace de dimension supérieure induit par le noyau utilisé par le classificateur. La régularisation du paramètre C ne peut pas être optimisée en utilisant les méthodes à noyaux qui sont indépendantes de l'algorithme d'apprentissage appliqué dans l'espace transformé. L'adaptation du paramètre C sera effectuée par la minimisation de l'erreur de classification par validation croisée utilisant la grille de recherche pour une dimension (Line-Search).

CHAPITRE IV : LA SELECTION DE DESCRIPTEURS

IV-1- Problématique :

En apprentissage statistique supervisé, on cherche à modéliser puis prédire un descripteur des étiquettes- ou variable dépendante (catégorielle) à partir d'un ensemble de variables explicatives (descripteurs). Le problème de "*malédiction de la dimensionnalité*" en statistique désigne les situations où l'on dispose de peu d'observations alors que le nombre de descripteurs est très grand. Cette situation est de plus en plus fréquente dans les applications, en particulier celles liées aux biopuces. Une biopuce fournit une seule observation de plusieurs milliers de gènes simultanément. Cette observation correspond en général à une seule condition expérimentale (une cellule par exemple) et à une seule classe parmi deux ou plusieurs (cellule saine ou cellule cancéreuse par exemple). Les gènes jouent le rôle des variables, et le nombre d'observations (donc de biopuces) est très faible pour des raisons de coût.

Dans un contexte différent de celui des biopuces, qui est les signaux audio, on cherche à réduire le nombre de variables explicatives, parmi lesquelles, peu sont informatives en général c'est-à-dire indépendantes de la classe, et les autres constituent essentiellement du bruit. Le problème auquel nous nous intéressons ici est celui de la détermination des variables explicatives importantes pour un modèle de type méthodes à vecteurs supports. Bien que théoriquement considéré comme insensible à la dimensionnalité du problème de discrimination (voir l'équation (II.9)), il s'avère en pratique que les SVM souffrent également de la redondance et de la non-pertinence de certaines variables, on donc voit leurs performances chuter. C'est la raison pour laquelle des méthodes de sélection de variables sont souvent mises en œuvre. En particulier, nous traitons cette question dans le cadre de la classification supervisée (cas où la variable à prédire des étiquettes est catégorielle) binaire. Notre objectif est donc de réduire la dimension de l'espace des variables explicatives sans perte significative de performance pour le modèle utilisé.

IV-2- Etat de l'art :

La taille des données peut être mesurée selon deux dimensions, le nombre de variables et le nombre de données. Ces deux dimensions peuvent prendre des valeurs très élevées, ce qui peut poser un problème lors de l'exploration et l'analyse de ces données. Pour cela, il est fondamental de mettre en place des outils de traitement de données permettant une meilleure compréhension de la valeur des connaissances disponibles dans ces données. La réduction des dimensions est l'une des plus vieilles approches permettant d'apporter des éléments de réponse à ce problème. Son objectif est de sélectionner ou d'extraire un sous-ensemble optimal de caractéristiques pertinentes pour un critère fixé auparavant. La réduction de ce sous-ensemble de caractéristiques permet d'éliminer les informations non-pertinentes et redondantes. Elle permet donc de réduire la dimension de l'espace d'entrée et de rendre l'ensemble des données plus représentatif du problème. En effet, les principaux objectifs de la réduction de dimension sont :

- Elle facilite la visualisation et augmente la compréhensibilité des données.
- Réduire l'espace de stockage nécessaire.
- Identifier les facteurs pertinents.
- Lorsque le nombre de variables est vraiment trop grand (il peut aller jusqu'à plusieurs dizaines de milliers dans certaines applications), l'algorithme d'apprentissage ne peut pas terminer en un temps convenable. La réduction de dimensionnalité diminue les charges de calcul dû à l'apprentissage.
- D'un point de vue intelligence artificielle, créer un classificateur revient à créer un modèle pour les données. Or, une attente légitime pour un modèle est d'être le plus simple possible (principe du Razoir d'Occam [7]) pour éviter le problème de sur-apprentissage. La réduction de la dimension de l'espace des caractéristiques permet alors de réduire le nombre de paramètres nécessaires à la description de ce modèle.
- Elle améliore la vitesse de la classification.
- Enfin, mais pas toujours, nous améliorons la qualité de la prédiction, le ratio nombre d'observations / dimension de représentation étant plus favorable.

Voir aussi [46], pour les différents objectifs.

Les algorithmes d'apprentissage artificiel requièrent typiquement peu de traits - *features* - ou de variables - attributs - (ou colonnes) très significatives caractérisant le processus étudié. Dans le domaine de la reconnaissance des formes et de la fouille de données, il pourrait encore être bénéfique d'incorporer un module de réduction de la dimension dans le système global avec comme objectif d'enlever toute information inconséquente et redondante. Cela a un effet important sur la performance du système. En effet le nombre de caractéristiques utilisées est directement lié à l'erreur finale. L'importance de chaque caractéristique dépend de la taille de la base d'apprentissage, pour un échantillon de petite taille, l'élimination d'une caractéristique importante peut diminuer l'erreur. Il faut aussi noter que des caractéristiques individuellement peu pertinentes peuvent être très informatives si nous les utilisons conjointement.

Nous pouvons classer toutes les techniques mathématiques de réduction de dimension en deux grandes catégories :

- **La sélection de variables :** qui consiste à choisir des caractéristiques dans l'espace de mesure n'impose pas d'effort d'interprétation de nouvelles variables à l'utilisateur, (Figure IV.1).
- Et **l'extraction de traits :** qui vise à sélectionner des caractéristiques dans un espace transformé -dans un espace de projection (figure IV.2).

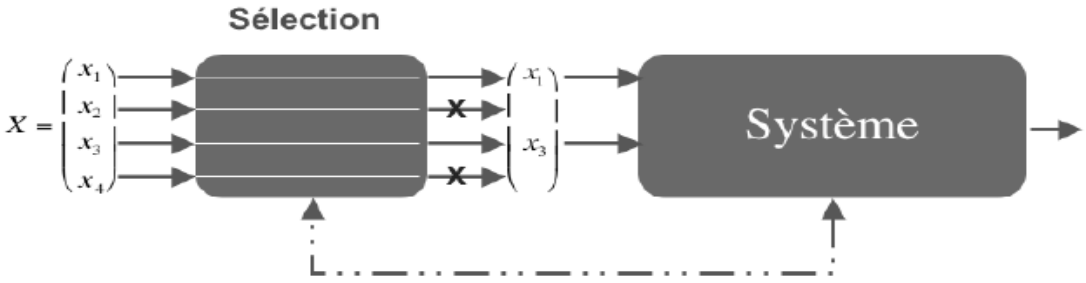


FIG IV.1. Principe de la sélection de variables.

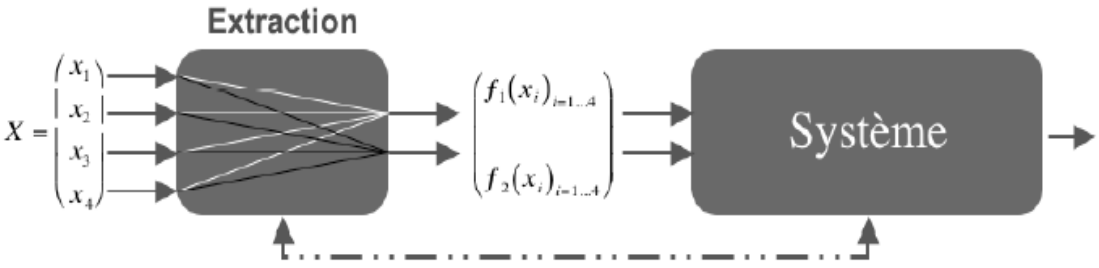


FIG IV.2. Principe de l'extraction de caractéristiques.

Dans ce mémoire, nous nous intéressons à la sélection de variables dans le cadre de la classification supervisée, en présence d'une variable des étiquettes Y binaire.

Définition IV.1 (Variables et Traits [47]) : *Nous appelons "variables" les descripteurs d'entrée et "traits" des caractéristiques construites à partir des variables d'entrée.*

La distinction est nécessaire dans le cas des méthodes à noyaux pour lesquelles les traits (*features*) ne sont pas explicitement calculés.

La première catégorie est appropriée quand l'acquisition de mesures des formes est coûteuse. Ainsi l'objectif principal de la sélection de caractéristiques dans ce cas est de réduire le nombre de mesures requises. Elles consistent à éliminer les variables considérées moins importantes selon un certain critère et en suivant une certaine stratégie de sélection. Les performances de ces approches dépendent fortement de la nature du critère et de la stratégie utilisée. Par contre, les techniques d'extraction de traits (deuxième catégorie) utilisent toute l'information (espace de mesures) contenue dans les formes pour la compresser et produire un vecteur de plus petite dimension. Ces techniques projettent un vecteur forme de l'espace de représentation dans un espace de dimension plus petite comme les méthodes linéaires (ACP (Analyse en Composantes Principales), AF(Analyse Factorielle), MDS(Multidimensional Scaling) [7, 48].

Nous nous retrouvons de plus en plus dans des situations où le nombre de variables n a tendance à être beaucoup plus grand que le nombre d'observations m. Dans un ensemble de mesures effectuées sur le phénomène originel, toutes ne sont pas aussi pertinentes. Il est possible que certaines variables correspondent à du bruit ou qu'elles soient peu informatives, corrélées et redondantes ou même inutiles au problème de classification. Il est très naturel que le succès d'un classifieur dans ce genre de situation soit fortement conditionné par la qualité des données et des variables qui les caractérisent. De ce fait il est devenu indispensable de proposer des méthodes efficaces pour sélectionner les variables pertinentes sans que les performances de la règle de classement se détériorent.

La thématique de sélection de caractéristiques est un problème difficile qui a été étudié depuis les années 70. Il revient dans l'actualité scientifique avec l'apparition des grandes bases de données et les systèmes de fouille de données "*Data Mining*" [49], devenu un domaine de recherche actif. Elle consiste à extraire de l'ensemble des variables explicatives disponibles un

ensemble optimal des caractéristiques les plus importantes à un système donné afin de mener à bien la tâche pour laquelle il a été conçu. De nombreux travaux et publications dans la littérature (voir [51]), traitent de ces techniques qui sont appliquées dans un grand nombre de domaines là où le nombre de variables ne devrait pas dépasser quelques centaines. Pendant ces quelques dernières années de nouvelles techniques [46] ont été proposées pour aborder cette stimulante tâche en présence de milliers de variables explicatives.

La sélection de variables est une problématique complexe et d'une importance cruciale pour les systèmes d'apprentissage. Afin de mettre en évidence les deux aspects du processus de la sélection de variables, difficulté et importance, nous allons présenter les éléments essentiels que nécessite généralement ce processus. Une définition de la sélection de variables peut s'énoncer de la façon suivante :

Définition IV.2 (Pertinence d'une variable [47]) : *La sélection de variables est un procédé permettant de choisir un sous ensemble optimal de variables pertinentes, à partir d'un ensemble de variables original, selon un certain critère de performance.*

A partir de cette définition, on peut se poser trois questions essentielles :

- Comment mesurer la pertinence des variables ?
- Comment former le sous-ensemble optimal ?
- Quel critère d'optimalité utiliser ?

Ces trois questions définissent les éléments essentiels d'une procédure de sélection de variables. En général, le problème de la sélection de variables consiste à identifier les variables permettant une meilleure séparation entre les différentes classes dans le cas d'un classement. On parle alors de "pouvoir discriminant" pour désigner la pertinence d'une variable. Jusqu'à ce qu'un certain critère soit satisfait, des sous-ensembles sont générés en parcourant l'espace des sous-ensembles, et ils sont évalués.

La réponse à la première question consiste à trouver une mesure de pertinence ou un **critère d'évaluation** $J(X)$ permettant de quantifier l'importance d'une variable ou d'un ensemble de variables X. La deuxième question évoque le problème du choix de la **procédure de recherche** ou de constitution du sous-ensemble optimal des variables pertinentes. La dernière question demande la définition d'un critère d'arrêt de la recherche. Le **critère d'arrêt** est

généralement déterminé à travers une combinaison particulière entre la mesure de pertinence et la procédure de recherche. Une fois le "meilleur" sous ensemble déterminé, reste la **validation** de celui-ci pour évaluer la qualité des modèles obtenus. Une structure générale des algorithmes de sélection de caractéristiques peut être proposée de cette façon, d'après la figure IV.3 [49] :

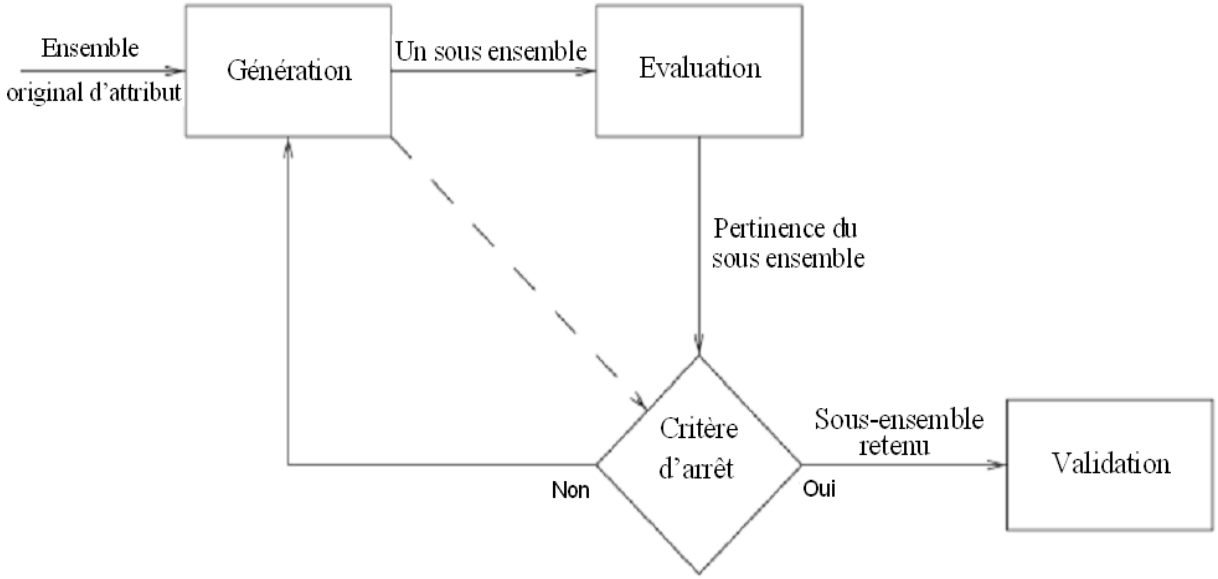


FIG IV.3. Procédure générale de sélection d'un sous-ensemble de caractéristiques.

Nous décrivons successivement chaque phase :

1. Critères d'évaluation :

L'amélioration des performances d'un système d'apprentissage par une procédure de sélection de variables nécessite dans un premier temps la définition d'une mesure de pertinence. Dans le cas d'un problème de classement, on teste, par exemple, la qualité de discrimination du système en présence ou en absence d'une variable. Commençons d'abord par définir ce qui est la pertinence d'une variable (ou d'un ensemble de variables).

Définition IV.3 (Pertinence d'une variable [47]) : *Une variable pertinente est une variable telle que sa suppression entraîne une détérioration des performances pouvoir de discrimination en classement du système d'apprentissage*.

Le problème est que cette approche est liée à un classifeur donné : elle doit donc être répétée lorsque l'on teste plusieurs classifeurs. Par ailleurs, l'estimation des performances demande un apprentissage du classifeur, ce qui peut induire un temps de calcul important.

Les approches de sélection se différent selon leur critère de sélection et la façon dont il est pris en compte dans la procédure de classification, voir [46, 50]. Il existe trois grandes familles d'approches :

Approches "Filtres" – *Filters* [19] : ces méthodes sélectionnent les variables indépendamment de la méthode qui va les utiliser, elles se basent sur les caractéristiques de l'ensemble des données afin de sélectionner certaines variables et d'éliminer d'autres sous forme de pré-traitement des données (ex : *critère de séparabilité des classes*).

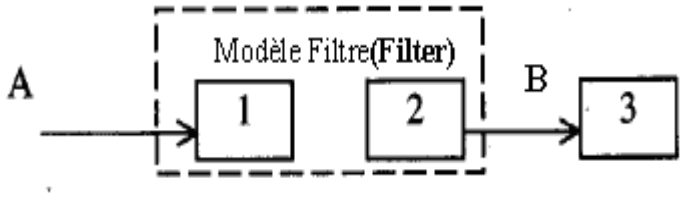


FIG IV.4. Approches "*Filtres*".

Où **A :** Ensemble de caractéristiques (*Features*) d'entrée.

B : Sous ensemble de caractéristiques de sortie.

1 : Générateur d'un sous ensemble de caractéristiques.

2 : Evaluateur d'un sous ensemble de caractéristiques.

3 : Méthode d'apprentissage pour la classification.

L'approche la plus simple consiste à évaluer le pouvoir discriminant de chaque variable prise individuellement avec complexité $O(D)$, en attribuant des scores pour chacune d'elles (*feature ranking*), puis de sélectionner les n meilleures variables. Elles sont aussi présentes sous l'appellation méthodes *univariées*.

Pour donner un score à un sous-ensemble (*subset ranking*), appelé méthodes *multivariées*, une première solution est de donner un score à chaque variable indépendamment des autres et de faire la somme de ces scores. Pour évaluer une variable, l'idée est de déterminer sa corrélation avec la variable de classe. Mais dans [46], les auteurs proposent des exemples simples montrant que cette approche nommée (*feature ranking*) pose des problèmes dans le cas général. Le problème est que cette méthode ne tient pas compte des interactions entre les variables :

- Il se peut qu'une caractéristique soit peu discriminante (peu corrélée avec la classe) considérée isolement, mais très discriminante lorsqu'elle est associée à d'autres variables;

- Par ailleurs, deux variables très corrélées risquent d'être choisies si elles sont discriminantes, alors qu'il vaudrait mieux de prendre une seule des deux variables pour éliminer la redondance, et en sélectionner une autre apportant une information complémentaire.

Il existe un intermédiaire entre "*feature ranking*" et "*subset ranking*" basé sur une idée de Ghiselli [23] et utilisé par Hall [22] dans le cadre de la *CFS* (*Correlation based feature selection*). Le score d'un sous-ensemble est construit en fonction des corrélations variable-classe et des corrélations variable-variable selon la formule suivante :

$$r_{\theta s} = \frac{k\,\bar{r}_{\theta i}}{\sqrt{k + k(k-1)\bar{r}_{ij}}} \qquad \text{(IV.1)}$$

Avec $r_{\theta s}$ le score du sous-ensemble de cardinal *k*, $\bar{r}_{\theta i}$ la moyenne arithmétique des corrélations entre θ l'attribut des étiquettes et les variables *i*, et $\bar{r}_{ij}$ la moyenne des k^2 intercorrélations entre variables. Cette équation exprime que le score du sous-ensemble augmente si les variables sont fortement corrélées avec θ et diminue si elles sont fortement corrélées entre elles. De plus cette formule est valable dans le cadre de scores normalisés (*i.e.* de variance unitaire). L'idée est de dire qu'un bon sous-ensemble est constitué de variables fortement corrélées avec la classe (pour ne pas garder les indépendantes), et peu corrélées entre elles (pour éviter la redondance). Il s'agit d'une approximation car on ne prend en compte que les interactions d'ordre 1.

La corrélation ou dépendance entre deux variables peut être définie de plusieurs façons. La plus utilisé est le coefficient de corrélation statistique [22, 46] qui est dans certains cas restrictif car il ne capture que les dépendances linéaires, celles-ci étant présentes entres les attributs et l'attribut des étiquettes de classes, le critère de corrélation donnerai de bon résultats.

Plusieurs critères d'évaluation ont été proposés, basés sur des hypothèses statistiques ou sur des heuristiques. Pour un problème de classement - discrimination -, les critères d'évaluation sont souvent basés sur les matrices de dispersion intra et inter classes. En effet, ces matrices sont directement liées à la géométrie des classes et donnent une information significative sur la répartition des classes dans l'espace des formes. Ex : *critère de séparabilité des classes*. Le meilleur sous-ensemble de variables est alors celui qui définit un espace de représentation dans lequel les classes sont les plus séparées possibles.

On trouve aussi des critères d'évaluation qui utilisent des mesures de distances probabilistes (ex : Mahalanobis, Battacharyya), d'information (ex : entropie) ou de dépendance (ex : corrélation, information mutuelle).

Les approches filtres sont beaucoup plus rapides, puisque la sélection se fait avant apprentissage, ils reposent sur des considérations plus théoriques, ils nous permettent de mieux comprendre les relations de dépendance entre variables. Mais, comme ils ne prennent pas en compte les biais de l'algorithme de classification, au principe les sous-ensembles de variables générés donnent un taux de reconnaissance plus faible que les approches *Wrappers*.

Approches "Symbioses" ou **"enveloppantes"** - *Wrappers* : ils ont été introduits par John *et al*. en 1994 [50, 53], contrairement aux approches filtres qui ignorent totalement l'influence des variables sélectionnées sur la performance de l'algorithme d'apprentissage, les approches "enveloppantes" utilisent l'algorithme d'apprentissage comme une fonction d'évaluation.

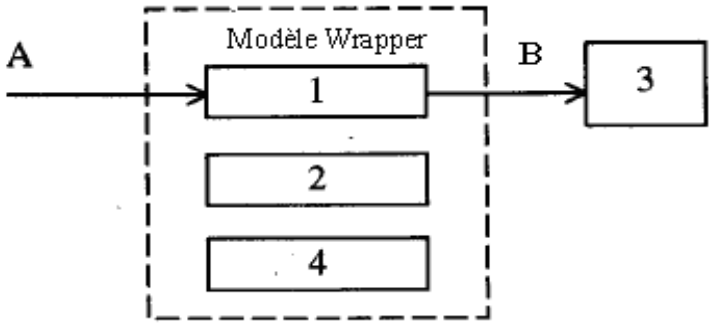


FIG IV.5. Approches " *Wrappers* ".

Avec **: 4 :** Méthode d'apprentissage centrale de sélection.

Leur principe est de générer des sous-ensembles candidats et de les évaluer grâce à l'algorithme de classification. Le score d'un ensemble sera par exemple un compromis entre le nombre de variables éliminées et le taux de réussite de la classification sur un fichier de test. Ainsi la phase "évaluation" du cycle de sélection est constituée par un appel à l'algorithme de classification. En fait, celui-ci est appelé plusieurs fois à chaque évaluation car un mécanisme de validation croisée est fréquemment utilisé.

Elles utilisent aussi certaines propriétés du modèle utilisé. Plus récemment Guyon *et al* [18] ont suggéré des scores pour chaque variable explicative utilisée dans un modèle de type méthode à vecteurs supports, permettant ainsi d'établir une hiérarchie des variables. Une fois une hiérarchie des variables obtenue, il est nécessaire de choisir celles à garder dans le modèle. En se basant sur un score calculé à partir des SVM, Guyon *et al*. ont proposé un algorithme d'élimination récursive des variables, nommé SVM-RFE, qui consiste à éliminer itérativement les variables qui font le moins décroître la marge, voir aussi [46].

De par son principe même, cette méthode génère des sous-ensembles bien adaptés à l'algorithme de classification. Les taux de reconnaissance sont hauts car la sélection prend en compte le biais intrinsèque de l'algorithme de classification. Un autre avantage est sa simplicité conceptuelle : il n'y a nul besoin de comprendre comment l'induction est affectée par la sélection de variables, il suffit de générer et de tester.

Cependant, trois raisons font que les *wrappers* ne constituent pas une solution parfaite. D'abord, ils n'apportent pas vraiment de justification théorique à la sélection et ils ne nous permettent pas de comprendre les relations de dépendances conditionnelles qu'il peut y avoir entre les variables. D'autre part la procédure de sélection est spécifique à un algorithme de classification particulier et les sous-ensembles trouvés ne sont pas forcement valides si on change de méthode d'induction (dans notre cas les SVM). Finalement et c'est le défaut principal de la méthode, les calculs deviennent vite très longs, voire irréalisables lorsque le nombre de variables croit. Des méthodes récentes combinant *wrapper* et *filter* sont présentées dans [46].

Approches "Intégrées" - *Embedded* : ces méthodes exécutent la sélection de variable pendant le processus de l'apprentissage (l'estimation du modèle); en d'autre termes elles combinent la sélection de variables et l'estimation du modèle en une seule tache (comme le font par exemple [19]). Le processus de la sélection de variables est effectué parallèlement au processus de classement - découvrir à la fois le nombre de groupes d'objets présents dans les données mais aussi un sous ensemble associé d'attribut pertinent -. Le sous-ensemble de variables ainsi sélectionné sera choisi de façon à optimiser le critère d'apprentissage utilisé. Cette dernière approche ne sera pas utilisée dans nos travaux.

2. Procédures de recherche (Génération de sous ensembles) :

En général, nous ne connaissons pas le nombre optimal *m* de variables à sélectionner. Ce nombre dépendra de - la taille et de la qualité de la base d'apprentissage - la quantité et la qualité d'information disponible - et de la règle de décision utilisée- le modèle. Pour un ensemble de *n* variables il existe (2^n −1) combinaisons de variables possibles où 2 représente deux choix : sélectionner ou ne pas sélectionner la variable. La recherche d'un sous-ensemble de *m* variables parmi *n* engendre un nombre de combinaison égal à :

$$\binom{n}{m} = \frac{n!}{(n-m)!m!} \tag{IV.2}$$

Le problème est que l'espace des solutions est très grand. Par exemple, pour *n*=24 et *m*=12, il y a 2.704.156 choix possibles. Si nous ne fixons pas le nombre *m* de variables à sélectionner, il faut considérer les (2^n − 1) sous ensembles non vides de variables (soit 16.777.216 pour *n*=24).

En grande dimension - très grande -, le nombre de combinaison à examiner devient très élevé et un **parcours exhaustif** n'est pas envisageable. La recherche d'un sous ensemble optimal de variables est un problème NP (Non déterministe Polynomial)-difficile, sauf lorsque *n* est déjà petit (mais l'intérêt d'une sélection de variable est alors moins évident). Une alternative consiste à utiliser une méthode de recherche de type *Branch & Bound* [49]. Cette méthode de recherche permet de restreindre la recherche et donne le sous ensemble optimal de variables, sous l'hypothèse de monotocité du critère de sélection *J(X)*. Le critère *J(X)* est dit monotone si :

$$X_1 \subset X_2 \subset ... \subset X_m : J(X_1) < J(X_2) < ... < J(X_m) \tag{IV.3}$$

Où X_K est l'ensemble contenant *k* variables sélectionnées.

Cependant, la plupart des critères d'évaluation utilisés pour la sélection ne sont pas monotones et dans ce cas nous avons recours à la seule alternative basée sur des méthodes sous-optimales comme les algorithmes gloutonnes ou séquentielles. Une meilleure approche consiste à utiliser les méthodes séquentielles qui utilisent un **parcours heuristique,** Le coût devient $\leq O(n^2)$:

- Stratégie ascendante : *Forward Selection (FS)*,
- Stratégie descendante : *Backward Selection (BS)*,
- Stratégie bidirectionnelle : *Bidirectional Selection (BiS)*.

La méthode *FS* procède par agrégations successives - par ajouts successifs de variables. Au départ l'ensemble des variables sélectionnées est initialisé à l'ensemble vide. À chaque étape *k*, on sélectionne la variable qui optimise le critère d'évaluation $J(X_K)$ et on la rajoute à l'ensemble des variables sélectionnées X_K sachant que la première variable sélectionnée est celle la plus discriminante de l'ensemble initial des variables. D'une manière plus simple on commence par rechercher la variable X_1 la plus discriminante, en parcourant toute les variables. Puis, nous recherchons la variable X_2 qui forme avec X_1 le couple de variables le plus discriminant en parcourant toute les variables sauf X_1. Puis nous cherchons la variable X_3 qui, associée à X_1 et X_2, est la plus discriminante, etc.

Soit X l'ensemble des variables, nous sélectionnons la variable x_i telle que :

$$J(X_k) = \max_{x_i \in X \setminus X_{k-1}} J(X_{k-1} \setminus \{x_i\}) \qquad \text{(IV.4)}$$

L'ordre d'adjonction des variables à l'ensemble des variables sélectionnées produit une liste ordonnée des variables selon leur importance. Les variables les plus importantes sont les premières variables ajoutées à la liste. Néanmoins, il faut aussi se rappeler que des variables individuellement peu pertinentes peuvent être très informatives si on les utilise conjointement.

La méthode *BS* est une procédure inverse de la précédente - par retraits successifs de variables. On part de l'ensemble *X* complet des variables et on procède par élimination. A chaque étape la variable la moins importante selon le critère d'évaluation est éliminée. Le procédé continue jusqu'à ce qu'il reste qu'une seule variable dans l'ensemble des variables de départ. A l'étape *k*, on supprime la variable x_i telle que :

$$J(X_k) = \max_{x_i \in X \setminus X_{k-1}} J(X_{k+1} \setminus \{x_i\}) \qquad \text{(IV.5)}$$

Une liste ordonnée selon l'ordre d'élimination des variables est ainsi obtenue. Les variables les plus pertinentes sont alors les variables qui se trouvent dans les dernières positions

de la liste. On reproche généralement à la méthode de sélection avant de ne pas prendre en compte le problème de la pertinence mutuelle : deux variables peuvent être intéressantes lorsqu'elles sont présentes conjointement dans le sous-ensemble sélectionné bien que chacune prise séparément n'apporte aucune information pertinente (un exemple classique de cette situation est le cas du XOR). La méthode d'élimination arrière permet de traiter correctement ce problème mais n'est malheureusement pas efficace du point de vue calculatoire. Les auteurs dans [52] avancent que la méthode *Backward Selection* est mieux adaptée mais les résultats expérimentaux de [53] montrent que les performances sont équivalentes. La méthode *Forward Selection* sélectionne néanmoins moins de variables.

La procédure *BiS (*STEPWISE) qui mixe ces deux approches précédentes effectue sa recherche dans les deux directions - *Forward* et *Backward* - d'une manière concurrentielle. Après avoir ajouté une variable, nous regardons s'il n'est pas nécessaire de retirer certaines variables parmi celles qui ont déjà été introduites. Puis nous regardons à nouveau s'il n'est pas possible d'en ajouter de nouvelles, etc. La procédure s'arrête dans deux cas : (1) quand une des deux directions a trouvé le meilleur sous-ensemble de variables avant d'atteindre le milieu de l'espace de recherche ; ou (2) quand les deux directions arrivent au milieu. Il est clair que les ensembles de variables sélectionnées trouvés respectivement par *FS* et par *BS* ne sont pas égaux à cause de leurs différents principes de sélection. Néanmoins, cette méthode réduit le temps de recherche puisque la recherche s'effectue dans les deux directions et s'arrête dès qu'il y a une solution quelle que soit la direction. *Sequential Forward Floating Search* (SFFS) et *Sequential Backward Floating Search* (SBFS) sont des généralisations de la méthode *BiS* et sont les procédures les plus récentes de cette catégorie [54], par exemple l'algorithme SFFS permet à chaque pas, d'ajouter une variable et d'en supprimer plusieurs tant que le sous ensemble résultant améliore le critère d'évaluation, mais elles ont été vite abandonnées [19] en raison de leur forte dépendance de l'ordre des variables dans les données d'origine.

D'autres types de recherche sont possibles (*greedysearch,* voir [50] pour une nomenclature). Il y a aussi le **parcours non déterministe (une recherche aléatoire)** : Les sous-ensembles sont générés lors d'un processus comprenant une part de hasard. Typiquement, [7, 55, 56] utilisent des algorithmes évolutionnistes (algorithmes génétiques) qui sont difficiles à mettre en œuvre et souvent coûteux en temps de calcul.

Tout algorithme fait partie d'une catégorie, doit sa nomenclature soit par rapport au type de la méthode de recherche qu'il utilise pour trouver le sous ensemble pertinent, ou par la mesure de pertinence d'attribut. Il est à noté toute fois, que nous ne pouvons pas utiliser n'importe quelle méthode de recherche avec n'importe quel critère d'évaluation. Donnant l'exemple de critère de corrélation statistique, il ne pas être étendu à des groupes de variables, dans des procédures de sélection de type "greedy" (procédures itératives "forward", "forward-backward", etc.).

3. Critères d'arrêt :

Le nombre optimal de variables n'est pas connu à priori, l'utilisation d'une règle pour contrôler la sélection-élimination de variables permet d'arrêter la recherche lorsqu'aucune variable n'est plus suffisamment informative. Le critère d'arrêt est souvent défini comme une combinaison de la procédure de recherche et du critère d'évaluation.

Le critère d'arrêt peut être de diverses natures : un temps de calcul, un seuil sur les performances du classifieur, un nombre de générations (pour un algorithme génétique), un nombre de variables sélectionnées, une évaluation heuristique de la "valeur" du sous-ensemble ou encore une recherche complète des variables dans ce cas le critère d'arrêt est le nombre d'attributs total atteint.

4. Validation :

Dans la littérature, la sélection de caractéristiques est souvent un prétraitement pour faire de la classification. Les auteurs valident donc leurs sous-ensembles de variables par les performances du programme d'apprentissage utilisé ensuite. En général, ils comparent les taux de réussite de la classification, avant et après la sélection. Une heuristique, souvent utilisée, consiste à calculer pour les différents sous ensembles de variables sélectionnées une estimation de l'erreur de généralisation par validation croisée en créant un ensemble de validation en plus de l'ensemble d'entrainement, que nous utilisons pour estimer l'erreur de classification. Ceci est valable seulement pour une large base de données. Le sous-ensemble de variables sélectionnées est celui qui minimise cette erreur comme pour les méthodes enveloppantes. Toutefois, utiliser un grand ensemble validation parmi nos données peut apparaitre comme une perte d'information lorsque le nombre de données est restreint. Un petit ensemble de validation conduirait en revanche à des conclusions peu fiables. Autrement, pour les petits ensembles

d'entrainement, nous pouvons effectuer *K*-fold cross-validation. Notons par la suite qu'on va utiliser un seul ensemble de données supervisées pour validation ET évaluation finale (test).

IV-3- Algorithmes de sélection des attributs (ASA) :

Un grand nombre d'algorithmes de sélection d'attributs est disponible dans la littérature mais rares sont les méthodes capables de relever le défi sur lequel nous nous focalisons. Dans cette partie nous donnons un bref aperçu des algorithmes de sélection étudiés et testés pour la tache de classification parole/musique. La plupart de ces algorithmes sont des approches de types *filters* car ces derniers ont prouvé d'être plus pratique que les *Wrappers* à cause de leur vitesse de déroulement ne nécessitant aucun apprentissage, ils produisent un score w_i relatif à chaque attribut i, $1 \leq i \leq D$, pour garder ceux qui présentent les scores les plus élevés (les d attributs les mieux classés).

Les ASA sont de deux types linéaires et non linéaires. Les méthodes linéaires reposent (au moins implicitement) sur l'utilisation d'une distance euclidienne (liée au produit scalaire ordinaire). Dans de nombreuses applications, la distance euclidienne n'a pas grand sens ; elle suppose en particulier que toutes les variables sont comparables entre elles (elles doivent donc avoir été convenablement normalisées). La théorie des espaces de Hilbert permet de définir d'autres produits scalaires, basés sur des fonctions noyaux $k(x, y)$. Le noyau k défini implicitement une application de l'espace d'origine vers un "espace de caractéristiques" F. La dimension de l'espace F est éventuellement infinie. De nombreuses méthodes statistiques peuvent s'exprimer en ne recourant qu'à des produits scalaires entre les points à traiter et les données d'apprentissage. Si l'on remplace le produit scalaire habituel par un noyau k, on rend la méthode non-linéaire ; c'est le "truc du noyau" - *kernel trick* -, qui a fait l'objet de nombreuses recherches depuis son introduction par Vapnik [2] dans le cadre des méthodes à vecteurs de support (SVM).

IV-3-1- Algorithme CLDA (Constrained Linear Discriminant Analysis) :

Cet algorithme s'inspire de l'*Analyse Linéaire Discriminante* (sous l'appellation anglophone LDA), également appelée Analyse Discriminante de Fisher [7], s'applique lorsque les classes des individus sont connues. C'est une excellente méthode de réduction de

dimensionnalité dans laquelle elle transforme les données originales dans un sous espace d'attributs de plus petite dimension. Cependant elle change l'espace physique original d'attributs, d'où l'apparition de l'approche CLDA qui consiste à sélectionner sans transformation les attributs afin de préserver la structure de l'information. Cette méthode est jugée meilleure que le gain d'information et le test du Chi2 avec une complexité de calcul inferieure (voir [17]).

Nous commençons par expliquer le principe de la LDA, ensuite nous développerons l'algorithme CLDA. La LDA permet de trouver les directions "utiles à une bonne discrimination des classes" (La première est le discriminant de Fisher généralisé) dans lesquelles les données originales sont projetées dans l'espace transformé en maximisant le rapport :

$$J(D) = \arg\max_{D} \frac{|D^T B D|}{|D^T W D|} \qquad \text{(IV.6)}$$

$B_{p\times p} = \frac{1}{n}\sum_{k=1}^{q} n_k (m_k - m)(m_k - m)^T$ avec $(m_k - m)(m_k - m)^T$ est une matrice carrée

et $W_{p\times p} = \frac{1}{n}\sum_{k=1}^{q} n_k S_k$ avec $S_k = (X_k - m_k)(X_k - m_k)^T$

Où $m_k = (m_{k1}, m_{k2}, ..., m_{kj}, ...m_{kp})^T$ et m_{kj}=moyenne de X_j dans w_k avec: $m_{kj} = \frac{1}{n_k}\sum_{j}^{n_k} X_{kj}$

B et *W* sont les matrices respectives de dispersion *inter-classe* (entre les centres de gravité m_1 et m_2) et *intra-classe* (entre les points et le centre de gravité correspondant). Le nuage de point de centre de gravité m est partagé en *q* (2 dans notre cas) sous-nuages par la variable "classe". Chaque sous-nuage (classe w_k) d'effectif n_k est caractérisé par son centre de gravite (ou *centroide*) m_k et sa matrice de variance-covariance S_k .

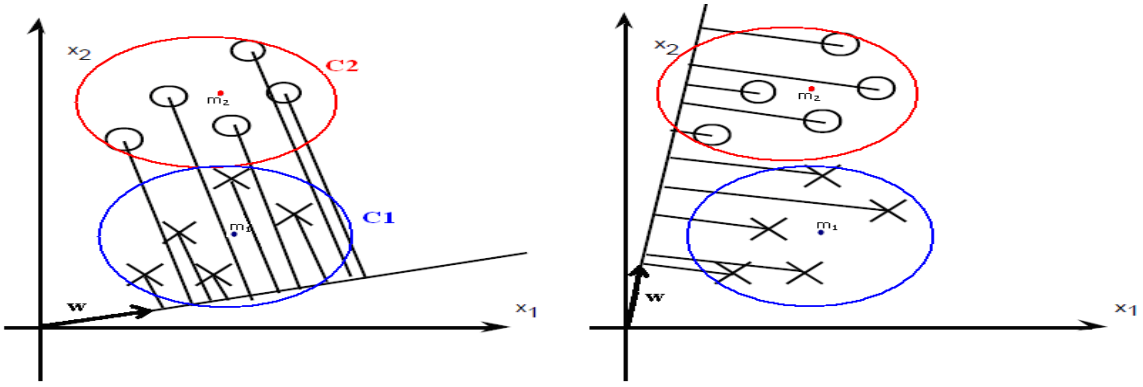


FIG IV.6 Analyse discriminante à une dimension (nous cherchons une droite pour laquelle les projections des points des différentes classes sont les mieux séparées, la figure de droite montre une grande séparation entre les deux classes).

La figure IV.6 montre la disposition des classes dans un espace à deux variables. Soient deux classes C1 et C2 de centres de gravité $m_1(X_1, X_2)$ et $m_2(X_1, X_2)$, les résultats de la transformation appliquée correspondent à des nuages de points de chaque classe très regroupés et que les centres de gravité des classes bien séparés.

Soit D^T la transformation linéaire permettant d'établir le nouvel espace de projection. Les vecteurs lignes de la matrice D^T définissent les directions des axes de la nouvelle base. En général, s'il y a q classes, nous allons chercher q-1 directions (vecteurs discriminants) qui correspondent aux q-1 plus grandes valeurs propres. La solution est trouvée en cherchant les racines du polynôme caractéristique et en résolvant l'équation (IV.7) :

$$(B - \lambda_i W)d_i = 0 \tag{IV.7}$$

L'algorithme de sélection que nous utiliserons, s'inspire de ce principe, sans qu'aucune projection des données ne soit réalisée, donc l'espace original des caractéristiques reste inchangé. CLDA est une des techniques de "*scoring*" essaye de déterminer la contribution des variables qui expliquent l'appartenance des individus à des groupes. Si nous supposons que le sous ensemble sélectionné de caractéristiques est $F^k = \left\{f_{t_1}, f_{t_2}, \ldots, f_{t_k} \mid 1 \le t_j \le p\right\}$, le rôle de la sélection d'attribut est de trouver la transformation $\widetilde{D} \in R^{k \times p}$ tels que la valeur de critère de pertinence $J(.)$ est maximisé. Les matrices de dispersion *inter-classe* et *intra-classe* dans l'espace de dimension k sont définies comme suit :

$$\widetilde{B} = \widetilde{D}^T B \widetilde{D}, \quad \widetilde{W} = \widetilde{D}^T W \widetilde{D} \tag{IV.8}$$

La transformation optimale qui conserve la structure des classes est donnée par maximisation de la $trace(\widetilde{B})$ et minimisation de la $trace(\widetilde{W})$ simultanément:

$$J(\widetilde{D}) = \arg\max_{\widetilde{D}} \frac{trace(\widetilde{D}^T B \widetilde{D})}{trace(\widetilde{D}^T W \widetilde{D})} \tag{IV.9}$$

L'optimisation au-dessus est égal à :

$$J(\widetilde{D}) = \arg\max_{\widetilde{D}} \left\{ trace(\widetilde{D}^T B \widetilde{D}) - trace(\widetilde{D}^T W \widetilde{D}) \right\} \tag{IV.10}$$

$$J(\widetilde{D}) = \arg\max_{\widetilde{D}} \left\{ \sum_{j=1}^{k} (B[t_j, t_j]) - (W[t_j, t_j]) \right\}$$

A partir de cette équation (IV.10), nous pouvons voir que $J(\widetilde{D})$ est déterminé par le signe de A :

$$A = (B[t_j, t_j]) - (W[t_j, t_j]), \qquad 1 \le j \le p \qquad \text{(IV.11)}$$

Quand :

– A>0, alors le $t_j^{ième}$ attribut contribue dans le critère d'évaluation $J(\widetilde{D})$.

– A=0, alors le $t_j^{ième}$ attribut ne contribue pas dans $J(\widetilde{D})$.

– A<0, alors le $t_j^{ième}$ attribut diminue $J(\widetilde{D})$.

Les axes les plus pertinents sont ceux qui possèdent un pouvoir discriminant A positif et maximal. Bien évidement le sous ensemble d'attributs correspondant au score A supérieur à zéro est la solution optimale pour une sélection d'attribut idéale. Pour une sélection d'attribut classique, k_1 est donné, nous devons juste choisir les attributs avec les plus grands scores.

Algorithme IV.1 : CLDA.

Entrées : k_1 *(nombre d'attributs)*; Sélection d'attribut idéale si k_1=0
Sélection d'attribut classique si $1 \le k_1 \le p$

x ←[x1, x2, . . .xn] //données d'apprentissage
y ←[y1, y2, . . .yn] //Étiquettes de classe

Sorties : F^{k_1}

1. Calcul du centroide m pour toutes les données d'apprentissage et m_k pour chaque classe
2. Calcul de B et W dans l'espace original de dimension p
3. $Score[t_j] = (B[t_j, t_j] - W[t_j, t_j])$ pour $1 \le t_j \le p$
4. Si k_1=0 alors $F^{k_1} = \{ f_{t_j} \mid score(t_j) > 0 \}$
5. Si $1 \le k_1 \le p$ alors $F^{k_1} = k_1$ attributs avec k_1 plus grands scores. Dans notre cas $k_1 = p$ alors F^{k_1} est l'ensemble de tous les attributs selon l'ordre d'importance.

IV-3-2- Séparabilité des classes à noyau (*Kernel Class Separability*) :

Le critère de séparabilité des classes à noyau (§III-2-3-2) à été développé dans un espace de grande dimension à noyau pour la sélection de l'hyperparamètre σ. Nous allons utiliser ce même critère pour la sélection d'attributs. Cette dernière est réalisée par la maximisation du ratio suivant : $\overline{Q} = \dfrac{tr(S_b)}{tr(S_w)}$ où S_b est la matrice de dispersion inter-classe et S_w est la matrice de dispersion intra-classe.

Nous rappelons depuis (§III-2-3-2), les études empiriques [14] faites sur des ensembles de données réels *UCI*, montrent que Q' est une fonction monotone (augmente ou décroît avec le paramètre noyau) mais cela n'affecte pas la sélection puisque ce ratio sera calculé en fonction des attributs et non des valeurs du paramètre noyau. Une liste de scores sera obtenue à la fin. Nous allons ordonner cette liste selon l'ordre décroissant, les plus importants (attributs) occuperont les premières positions.

IV-3-3- Critère d'Alignement de noyaux :

Dans [15], *Greene* propose de maximiser l'indice d'alignement de noyaux pour le réglage de la largeur du noyau gaussien σ et annoncent un nouvel usage possible pour la recherche du meilleur sous ensemble d'attribut.

L'alignement est considéré comme un critère consistant pour leur évaluation, en attribuant des scores pour chaque attribut. Le score qui est simplement la valeur de l'alignement pour un sigma fixé auparavant. Le noyau (matrice de gram) est calculé à chaque fois avec des vecteurs d'une seule dimension. L'algorithme étudié repose donc sur l'estimation de poids (scores) correspondant à chaque caractéristique. Ces poids sont utilisés pour ordonner puis sélectionner les K (parmi D) descripteurs les plus pertinents. La complexité de calcul de l'algorithme est O ($m^2 n$), dépend de façon importante de la dimension matrice du noyau RBF et de n la dimension de l'espace d'entrée.

IV-3-4- Critère de corrélation :

Le critère de corrélation basé sur coefficient de *Pearson* est souvent utilisé pour la sélection d'attributs comme approche de type *filter*, il est aussi nommé sous l'appellation anglophone *CFS* (*Correlation-based Feature Selection*) dans le cadre de sélection de sous ensemble de caractéristiques qui est simple à mettre en ouvre et ne nécessite pas un temps calculatoire énorme comparativement avec les approches de type *Wrappers*. Il a été montré dans [23] à travers les expériences faites sur la collection *UCI* que l'approche sélection d'attribut basée sur le critère de corrélation pour différents types de méthodes d'apprentissages, donné des résultats comparables même parfois meilleurs dans certains cas que les approches *Wrappers*. Ces dernières paraissent avoir des difficultés pour les ensembles de données avec peu

d'observations contrairement à la CFS que lui donné des résultats prometteurs même avec un faible ensemble de données.

Le critère de corrélation que nous allons utiliser dans nos travaux (voir (IV.12)) est un peu différent de celui utilisé en [24] (voir (IV.1)) car il ne prend pas en compte l'intercorrélation entre variables. Dans [24] *Hall* a montré que ce critère peut identifier les attributs utiles pour la classification en considérant seulement les interactions entre les attributs et l'attribut des étiquettes Y :

$$R = \frac{\sigma_{XY}}{\sigma_X \sigma_Y} \tag{IV.12}$$

- L'hypothèse dans laquelle cette heuristique est basée peut être énoncée comme suit : "Un sous ensemble d'attribut pertinent contient les attributs fortement corrélés avec la variable classe". La **corrélation** sert à quantifier la liaison linéaire mathématique entre 2 variables.

Le coefficient de corrélation de *Pearson* est défini comme la corrélation entre attribut X et un autre Y, Il fait jouer un rôle symétrique à X et Y.

$$R(i) = \frac{\mathrm{cov}(X_i, Y)}{\sqrt{\mathrm{var}(X_i)\,\mathrm{var}(Y)}} \tag{IV.13}$$

Où *cov* désigne la covariance et *var* la variance.
C'est un nombre sans dimension, indépendant des unités de mesure des observations puisque nous avons "standardisé" la covariance. *R(i)* est donné par:

$$R(i) = \frac{\sum_{k=1}^{m}(x_{k,i} - \overline{x_i})(y_k - \overline{y})}{\sqrt{\left(\sum_{k=1}^{m}(x_{k,i} - \overline{x_i})^2 \sum_{k=1}^{m}(y_k - \overline{y})^2\right)}} \tag{IV.14}$$

$\overline{x_i} = \frac{1}{m}\sum_{i=1}^{m} x_i$ est la moyenne de X Et $\overline{y} = \frac{1}{m}\sum_{i=1}^{m} y_i$ est la moyenne de Y.

– Interprétation géométrique :

Les deux séries de valeurs $X(x_1,...,x_m)$ *et* $Y(y_1,...,y_m)$ peuvent être considérées comme des vecteurs dans un espace à m dimensions. Remplaçons-les par des vecteurs centrés :

$X(x_1 - \bar{x},...,x_m - \bar{x})$ *et* $Y(y_1 - \bar{y},...,y_m - \bar{y})$. Le cosinus de l'angle α entre ces vecteurs est donné par la formule suivante (produit scalaire normé) :

$$Cos(\alpha) = \frac{\sum_{i=1}^{m}(x_k - \bar{x}).(y_k - \bar{y})}{\sqrt{\sum_{i=1}^{m}(x_k - \bar{x})^2}.\sqrt{\sum_{i=1}^{m}(y_k - \bar{y})^2}} \qquad \text{(IV.15)}$$

Donc *cos(α)* = *R*. Il nous renseigne de façon précise sur leur distance angulaire dans l'hypersphère à *m* dimensions. L'amplitude d'un coefficient de corrélation *R* varie entre +1 et -1:

- Si *R*=0 l'angle *α* = 90°, les deux vecteurs sont orthogonaux, alors il n'y a pas de relation entre les variables (linéairement indépendantes).
- Si *R*=-1 l'angle *α* vaut 180°, les deux vecteurs sont colinéaires de sens opposé, la relation est négative (et parfaite).
- Si *R*=+1 l'angle *α* = 0, les deux vecteurs sont colinéaires (parallèles), la relation est positive (et parfaite).

Les valeurs intermédiaires renseignent sur le degré de dépendance linéaire entre les deux variables. Il est toujours possible de calculer un coefficient de corrélation mais un tel coefficient n'arrive pas toujours à rendre compte de la relation qui existe en réalité entre les variables étudiées. En effet, il suppose que l'on essaye de juger de l'existence d'une relation linéaire entre nos variables. Il n'est donc pas adapté pour juger de corrélations qui ne seraient pas linéaires et non linéarisables. Si les deux variables sont totalement indépendantes, alors leur corrélation est égale à 0. La réciproque est cependant fausse, car le coefficient de corrélation indique uniquement une dépendance *linéaire*. D'autres phénomènes, par exemple, peuvent être corrélés de manière exponentielle, ou sous forme de puissance.

Supposons que la variable aléatoire *X* soit uniformément distribuée sur l'intervalle [-1,1], et que $Y = X^2$; alors *Y* est complètement déterminée par *X*, de sorte que *X* et *Y* ne sont pas indépendants, mais leur corrélation vaut 0.

Calculer le coefficient de corrélation entre 2 variables revient à chercher à résumer la liaison qui existe entre les variables à l'aide d'une droite. Plus la qualité globale de représentation de la liaison entre nos variables par cette droite est bonne, et plus le coefficient de corrélation linéaire associé l'est également, la figure IV.7 illustre bien cette situation :

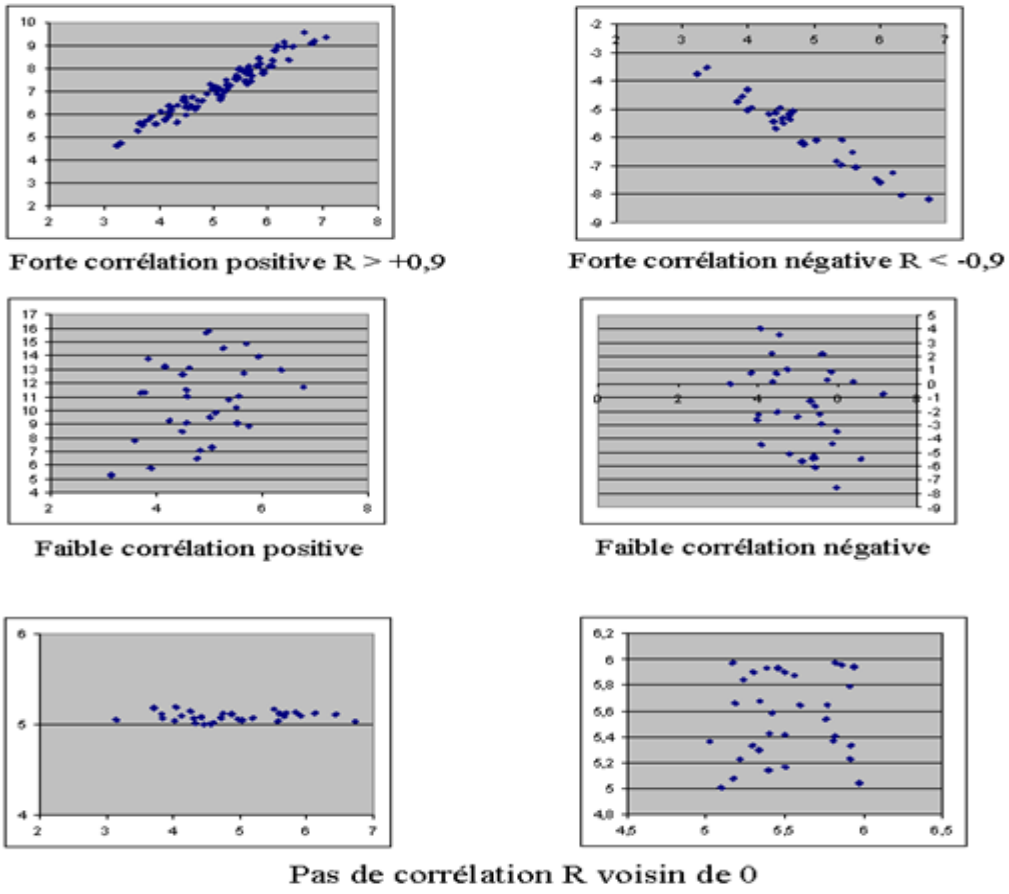


FIG IV.7 Représentation des différents types de résultats de *R* entre *X* et *Y*.

IV-3-5- Algorithme basé sur le critère d'Entropie de Représentation :

L'entropie de représentation *H* s'obtient en calculant les valeurs propres λ_i de la matrice de covariance des attributs de taille $D \times D$. Après normalisation de ces valeurs propres selon :

$$\tilde{\lambda}_i = \frac{\lambda_i}{\sum_{j=1}^{D} \lambda_j} \qquad \text{(IV.16)}$$

H se calcule selon :

$$H = -\sum_{j=1}^{D} \tilde{\lambda}_j \log \tilde{\lambda}_j \qquad \text{(IV.17)}$$

Avec $\tilde{\lambda}_i$ a des propriétés similaire avec la probabilité à savoir $0 \le \tilde{\lambda}_i \le 1$ et $\sum_{j=1}^{d} \tilde{\lambda}_j = 1$.

H est minimale lorsque la variance est concentrée sur un axe (toutes les valeurs propres sont faibles, sauf une). Elle est maximale lorsque la variance est repartie sur l'ensemble des

caractéristiques. L'entropie est donc d'autant plus grande que la variance se repartit uniformément sur l'ensemble des attributs. De ce fait, *H* peut être vu comme une mesure de la redondance de l'ensemble d'attributs considéré. Par conséquent, nous obtenons un sous-ensemble d'attributs non-redondants en maximisant *H* (voir [20, 21]).

Nous proposons la méthode de sélection suivante :

$$\begin{cases} \text{Critère d'évaluation} & : \text{Entropie} \\ \text{Procédure de recherche} & : \text{Backward} \\ \text{Critère d'arrêt} & : \text{le nombre total dattributs} \end{cases}$$

Nous avons choisi une méthode de sélection descendante ou méthode d'élimination arrière dans laquelle la variable la moins pertinente (sa suppression est la cause d'augmentation de *H*), est éliminée à chaque itération suivant le critère d'entropie, avec le critère d'arrêt utilisé est le nombre total d'attributs. A chaque itération, nous aurons plusieurs sous ensembles d'attributs de même taille, nous choisissons donc le meilleur (moins redondant) pour lequel l'entropie de représentation qui est lui associée est la plus grande. A la fin on aura une liste d'attributs dans l'ordre croissant d'importance.

Algorithme IV.2 :

Entrées : $X \leftarrow [x_1, x_2, \ldots x_l]^T$ //données d'apprentissage

$Y \leftarrow [y_1, y_2, \ldots y_l]^T$ //Étiquettes de classe

//Initialisation

$R \leftarrow \{\}$ //Classement des attributs

$H' = \{\ ,\ \}$ //Matrice contient l'entropie H_a, et l'indice d'attribut correspondant

$S \leftarrow \{1, 2, \ldots D\}$ //Sous-ensemble des attributs

Tant que card(S) ≠ 2 **faire**

$S_1 \leftarrow \{1, 2, \ldots D\}$

Pour a=1:D *et* $a \neq R$ //L'indice de l'attribut a va être comparé avec tous les indices de R

$X' \leftarrow X(:, S_1 - \{a\})$

Évaluer H_a sur X' en utilisant (**IV.17**)

$H' \leftarrow [H_a, a]$

Fin Pour

$f \leftarrow a = \arg\max \{H_a\}$ //Trouver l'attribut a dont sa suppression augmente H_a

$R \leftarrow R \cup f$ //Mise à jour du classement des attributs

$S \leftarrow S - \{f\}$//Retirer l'attribut f

Fin tant que

$R \leftarrow S_1 - R$ //Ajout des deux derniers attributs

Sorties: Classement des attributs R

IV-3-6- Algorithme SVM-RFE (Elimination Récursive des Attributs) :

SVM-RFE est un algorithme de type *wrapper* exploitant les SVM de façon récursive pour estimer des scores $\{w_i\}_{1\le i\le D}$ relatifs à chaque attribut. Le score w_i correspondant à l'attribut *i* est obtenu en moyennant les scores $\{w_i^{pq}\}_{1\le p<q\le Q}$ calculés pour chaque problème bi-classes linéairement séparables $(\Omega_p \; vs \; \Omega_q)$ à partir de la méthode à vecteurs supports correspondante. Ces scores sont ici simplement les composantes du vecteur de poids w^{pq}, définissant l'hyperplan optimal obtenu pour la paire $\{\Omega_p, \Omega_q\}$, qui est, nous le rappelons, une combinaison linéaire des n_s vecteurs supports x_i du problème, faisant intervenir les multiplicateurs de Lagrange α_i :

$$w^{pq} = \sum_{j=1}^{n_s} \alpha_j y_j x_j \qquad \text{(IV.18)}$$

L'idée est que les attributs qui correspondent à des directions de l'espace selon lesquelles le vecteur w^{pq} admet une faible énergie, ne sont pas aussi utiles au problème que les autres attributs (puisqu'ils contribuent faiblement à la définition de l'hyperplan optimal). A chaque récursion de l'algorithme SVM-RFE, l'attribut possédant le score le plus faible est éliminé (*Backward Selection*). Le processus est arrêté lorsque le nombre d'attributs restant atteint *d*. Il est possible d'éliminer plus d'un attribut à la fois pour réduire la complexité de l'algorithme, qui est assez élevée ($O(l^2 D)$). Nous proposons une description de cette approche sous forme d'algorithme, d'après [18] :

Algorithme IV.3 : SVM-RFE pour un problème bi-classes.

Entrées : $X \leftarrow [x_1, x_2, \ldots x_l]^T$ // données d'apprentissage
$Y \leftarrow [y_1, y_2, \ldots y_l]^T$ //Étiquettes de classe
d //Nombre d'attributs cible
//Initialisation
$S \leftarrow \{1, 2, \ldots D\}$ //Sous-ensemble des attributs "survivants"
$R \leftarrow \{\}$ //Classement des attributs
Tant que $s \neq \{\}$ **faire**
$X \leftarrow X(:, S)$ //Restreindre aux attributs utiles
$\alpha \leftarrow$ SVM-train(X, y) // Apprentissage SVM linéaire
$w \leftarrow \sum_k \alpha_k y_k x_k$ //Calcul du vecteur de poids de dimension égale à card(S)
$f \leftarrow \arg\min \{w_i^2 \mid i \in S\}$ //Trouver l'attribut ayant le score (w_i^2) le plus bas
$R \leftarrow R \cup S(f)$ //Mise à jour du classement des attributs
$S \leftarrow S \setminus f$ //Suppression de l'attribut ayant le score le plus bas
Fin tant que
Sorties: Classement des attributs R

IV-4- Conclusion :

Afin d'introduire les mises en œuvre des chapitres III et IV, nous avons présenté dans ce chapitre un aperçu de la problématique de sélection d'attributs dans le cadre de la classification supervisée SVM, l'application ciblée par ce travail.

En premier lieu, nous avons tenu à présenter un historique des principaux apports ayant influencés le développement du domaine. Ensuite nous avons exposé les éléments essentiels d'une procédure de sélection d'attributs. En général, le problème de la sélection d'attributs consiste à identifier les attributs permettant une meilleure séparation entre les différentes classes dans le cas d'un classement.

Dans la dernière partie du chapitre, nous avons développé les ASA étudiés et testés dans nos travaux. L'objectif de notre mémoire consiste à développer des méthodes à noyaux pour la sélection d'attributs puis de comparer ces méthodes avec celles de l'état de l'art. La plupart de ces algorithmes sont des approches de types *filtres* ne nécessitant aucun apprentissage de la statistique de décision, à l'exception de SVM-RFE qui est un algorithme de type *wrapper*.

CHAPITRE V : EXPERIMENTATIONS ET RESULTATS

V-1- Cadre des expérimentations :

Nous présentons dans ce chapitre les résultats des expérimentations obtenus avec les méthodes proposées et discutées dans les chapitres III et IV. Cette étude s'articule autour de deux parties à savoir la sélection des hyperparamètres et la sélection des descripteurs dans le cadre de l'apprentissage supervisé.

Nous avons constitué un corpus audio, nous avons extrait les descripteurs adaptés en générale de type MFCC qui sont largement utilisés en audio [57] dans la plupart des systèmes de reconnaissance automatique et qui montrent de bonne performances. A la fin nous l'avons étiqueté manuellement en classes parole/musique.

Nous avons donc cherché en premier temps à quantifier l'intérêt de l'utilisation des méthodes à noyaux par rapport à celle de la méthode classique de l'état de l'art pour la sélection de modèle. Dans un deuxième temps, nous avons effectué une sélection de descripteurs sur un autre corpus de taille réduite afin d'améliorer le taux de bonne classification et diminuer le temps d'entrainement en utilisant une combinaison de diverses stratégies de paramétrisations acoustiques. Cette combinaison est réalisée en concaténant plusieurs vecteurs acoustiques résultants de diverses paramétrisations.

V-2- Matériels et logiciels :

Les expérimentations ont été réalisées dans l'environnement suivant :

CPU : Pentium Core 2 DuoT1350

RAM : DDR II 1GB

OS : Microsoft Windows XP SP2

Langage : MATLAB Version 7.5.0.342 (R2007b)

SVM tool : LibSVM 2.88 (released on October 30, 2008)

V-3- Corpus utilisé :

Nous avons utilisé un enregistrement d'une durée de 46.5minutes en utilisant les deux stations *Radio France Info* et *Radio France Musique* effectué le 17 Février 2009. La première contient de la parole pure composée d'émissions (en général tours de parole) d'une durée de 25.33minutes (10 fichiers) et la deuxième de la musique pure (instrument sans parole (piano, violon, violoncelle, pipeau, clarinette)) de genre *Classique* d'une durée de 21.17minutes (8 fichiers), voir tableau V.1. L'avantage des deux stations est qu'elles présentent des longues périodes de parole et de musique.

TAB. V.1

Création de la base de données avec illustration des trois partitions de données.

		Partition 1		**Partition 2**		**Partition 3**	
Fichier (.Wav)	**Durée (min : sec)**	**App1**	**Test1**	**App2**	**Test2**	**App3**	**Test3**
Parole (1)	2 :59	×		×		×	
Musique (1)	1 :07	×		×		×	
Parole (2)	3		×	×		×	
Musique (2)	1 :06		×	×		×	
Parole (3)	2 :15		×	×		×	
Musique (3)	4		×		×	×	
Parole (4)	3		×		×	×	
Musique (4)	2 :59		×		×	×	
Parole (5)	3		×		×		×
Musique (5)	4 :35		×		×		×
Parole (6)	2 :01		×		×		×
Musique (6)	3 :56		×		×		×
Parole (7)	2 :58		×		×		×
Musique (7)	3 :01		×		×		×
Parole (8)	1 :59		×		×		×
Parole (9)	1 :59		×		×		×
Parole (10)	2 :09		×		×		×
Musique (8)	0 :26		×		×		×

Donc la base de données est issue d'un mixage à partir de différents enregistrements (parole/musique) à l'aide du logiciel **"*Audacity*" Version 1.2.6**, disponible sur le web [58]. La figure V.1 montre l'interface graphique du logiciel :

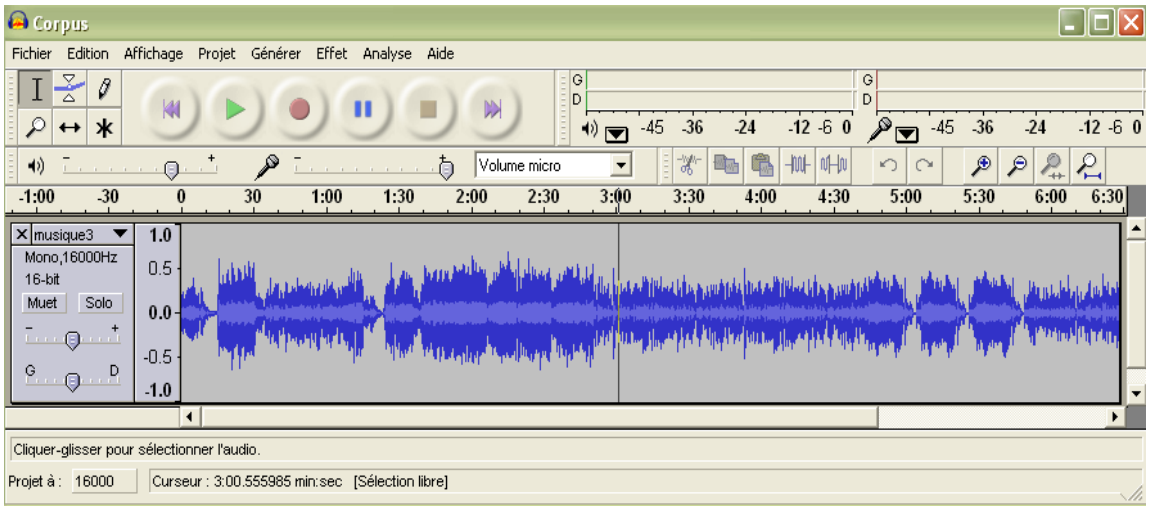


FIG V.1. Interface graphique du logiciel *Audacity*.

Toutes les données sont sous-échantillonnées en 16khz, pour permettre de réduire la complexité des calculs (par rapport à la fréquence d'échantillonnage initiale de 44.1 kHZ) tout en conservant une bonne qualité audio, encodés en mono au format PCM (*Pulse Coded Modulation*). Nous avons utilisé pour l'extraction de paramètres l'outil *HCOPY* du paquetage *HTK*, 2002, disponible sur internet [59], avec les coefficients MFCC extraits à partir de 22 filtres Mel calculés toutes les 10 ms sur des fenêtres de Hamming de 30 ms.

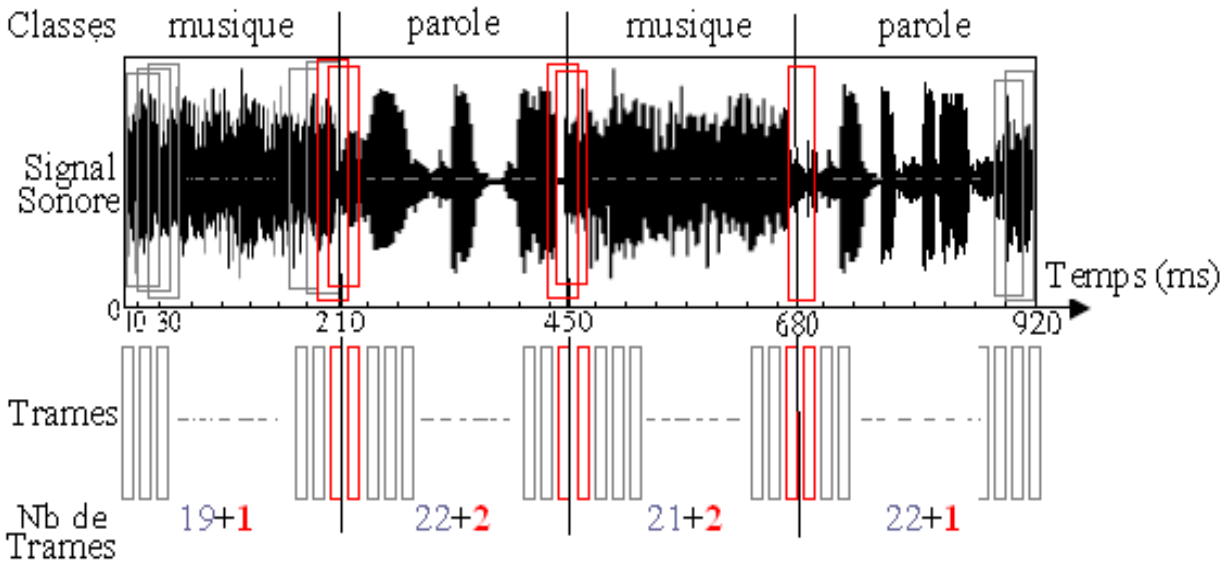


FIG V.2. Etiquetage manuel du signal sonore en classes parole/musique.

L'ensemble des données a été paramétré avec la configuration 16MFCC. Chaque vecteur de paramètres est normalisé linéairement entre –1 et +1. Ensuite l'ensemble de données a été classifié manuellement en classes parole/musique. La paramétrisation acoustique consiste à découper le signal de parole en fenêtres (trames) glissantes de taille fixe 30ms reparties de façon uniforme le long du signal (toutes les 10ms) comme le montre la figure V.2, ensuite à extraire les descripteurs sur chacune des fenêtres. Le signal audio est donc considéré comme stationnaire sur la durée de la fenêtre. Chaque trame est étiquetée suivant le contenu du segment auquel elle appartient. Les trames peuvent ainsi être étiquetées en :

- Parole (P), si le segment ne contient que de la parole;
- Musique (M), si le segment ne contient que de la musique.

Toute fois il y aura des zones où la trame va être composée de (P et M), les trames rouges dans la figure V.2, constituant une troisième catégorie le bruit (B). Cette dernière catégorie ne fait pas l'objet de notre étude. Dans ce cas la, le segment (rouge) va être affecté à la classe majoritaire qui couvre 20 ms du signal (P ou M).

Nous prenons comme exemple la portion de durée 210 ms d'après la figure V.2. Pour trouver le nombre de trames acoustiques (N_t=19) dans cette portion nous avons utilisé la formule suivante :

$$N_t=(T(\text{ms})-20(\text{ms}))/10(\text{ms}) \qquad \text{(V.1)}$$

Avec : T est la taille de la portion du signal considéré.

V-4- Les résultats des expérimentations :

V-4-1- Sélection de modèle:

Dans un premier temps, nous devons partitionner le fichier de données : une première partie, dite d'apprentissage, elle est utilisée pour la construction des modèles; une seconde partie dite test est réservée pour leur évaluation. Cette subdivision est toujours souhaitable dès lors que nous voulons obtenir une évaluation crédible des performances. Cette subdivision est faite en variant à chaque fois la durée de l'ensemble d'apprentissage (voir tableau V.1), on identifié ainsi trois partirions P1, P2, P3. Nous pouvons dorénavant faire une recherche des meilleurs hyper-paramètres avec les trois méthodes de sélection de modèle décrites précédemment. La méthode1

qui utilise le critère d'alignement, la méthode2 qui se base sur un nouveau critère qui minimise à la fois la borne "rayon-marge" et maximise la séparabilité des classes dans l'espace à noyau pour trouver sigma. La validation croisée sera utilisée dans ces deux méthodes pour trouver le paramètre C utilisant une ligne de recherche de dimension (1×7). Et finalement la methode3 est la validation croisée classique discrétisée en grille du plan (σ, C), en anglais appelée *Grid-Search* de dimension 7×10.

On utilise pour la validation croisée (*VC*) 5-fold-cross-validation $k = 5$.

La sélection des hyper-paramètres étant pratiquée à priori. Donc le meilleur modèle, que nous avons utilisé pour prédire les sorties des observations de la base de test, est celui établi en utilisant les valeurs optimales des deux paramètres (σ_{opt}, C_{opt}). Un lien direct avec l'erreur de généralisation assure la pertinence des méthodes de sélection de modèle.

Après avoir obtenus les hyper-paramètres, on évalue les performances par le Taux de Classification Correcte **(TCC)** renvoyé par l'outil *Libsvm* [25], il est exprimé en pourcentage des données correctement classifiées par le système. On divise le TCC selon notre travail de classification à trois types :

- **TCC :** Taux de Classification Correcte sur les données test après avoir obtenu la structure du modèle approprié à partir d'échantillons d'apprentissages.
- **TCC_{vc} :** Taux de Classification Correcte de test finale, c'est une moyenne sur les tests croisés des K estimations possibles.
- **TCC[training] :** Taux de Classification Correcte apparent (ou de re-substitution) obtenu en re-classant les individus ayant servi à apprendre la règle de décision. Ce taux est très souvent trop optimiste. Optimisme d'autant plus marqué qu'il y a sur-apprentissage : le modèle "colle" trop aux données.

On note que plusieurs couples (σ, C) peuvent avoir la même précision dans l'étage de validation croisée, dans ce cas la, nous pouvons appliquer tout d'eux sur l'ensemble de test, et on reporte celui qui donne le plus grand TCC. Maintenant si plusieurs couples donnent le même TCC, nous reporterons celui qui donne le temps d'entrainement minimal. Dans nos expérimentations nous avons décidé de choisir l'une des valeurs de couple (σ, C).

Dans nos expérimentations nous utilisons la fonction à base radiale (RBF) :

$k(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$ avec : $\sigma \in \{.1\ .2\ .3\ .4\ .5\ .6\ .7\ .8\ .9\ 1\}$ et

$C \in \{1\ 5\ 50\ 500\ 1000\ 5000\ 10000\}$

Les tableaux V.2, .3, .4, .5, .6, .7, .8, .9, .10 présentent les résultats d'expérimentations des trois procédures de sélection des hyperparamètres (σ_{opt}, C_{opt}) pour différentes tailles de corpus de données. Tous les réglages ont été effectués sur l'ensemble d'apprentissage seul (sans recours à une étape de validation).

1. Partition1 :

– Méthode1 (A_{max}, C_{vc}) :

TAB V.2

La sélection de modèle avec la première méthode pour la partition *P1*.

σ	.1	.2	0.3	.4	.5	**.6**	.7	.8	.9	1
A	0.0069	0.0396	0.1454	0.2297	0.2669	**0.2753**	0.2718	0.2646	0.2570	0.2501
(σ_{opt}=0.6,C)	.1	**5**	50	500	1000	5000	10000			
TCC_{vc}(%)	99.6687	**99.8667**	99.8384	99.7858	99.7858	99.7858	99.7858			

Les meilleurs hyper-paramètres sont obtenus avec σ_{opt} = 0.6 pour A_{max} = 0.27 et C_{opt} = 5 pour TCC_{vc}=99.86%.

– Méthode2 (Q^*_{min}, C_{vc}) :

TAB V.3

La sélection de modèle avec la deuxième méthode pour la partition *P1*.

σ	.1	.2	0.3	**.4**	.5	.6	.7	.8	.9	1
Q ($\times 10^4$)	1.7198	0.34159	0.09475	**0.08492**	0.14434	0.28026	0.53374	0.97103	1.68320	2.79062
$\overline{Q}$ ($\times 10^{-4}$)	2.3945	32.468	288.78	1010.1	2179.8	3669.3	5393	7317.5	9435	11750
Q^*_{min} ($\times 10^7$)	7.1825	0.10521	0.03281	0.008408	**0.006622**	0.007638	0.009897	0.01327	0.01784	0.02375
ERR_{test}(%)	48.9994	48.7441	46.2619	42.8387	40.8351	40.5102	40.3231	40.341	40.3256	40.5757
(σ_{opt}=0.5,C)	.1	**5**	50	500	1000	5000	10000			
TCC_{vc}(%)	99.7333	**99.8545**	99.8465	99.8424	99.8424	99.8424	99.8424			

Les meilleurs hyperparamètres sont obtenus avec σ_{opt} = 0.5 pour Q^*_{min} = 66220 et C_{opt} = 5 pour TCC_{vc}=99.85%. Le calcul du ratio Q^* nécessite l'entrainement du classificateur, nous choisissons la valeur de paramètre C égale à 10000 et ceux pour les trois partitions.

L'erreur de test due à ce critère Q est égale à 42.83% pour σ_{opt}=0.4 est donc souvent beaucoup plus grande que l'erreur de test minimum qui est égale à 40.32% pour σ_{opt}=0.7. Le critère Q^* est plus performant par rapport à Q car il permet une diminution de l'erreur de généralisation avec 40.83% pour σ_{opt}=0.5.

– Méthode3 (*VC*) :

TAB V.4

La sélection de modèle avec la troisième méthode pour la partition *P1*.

(c, σ)	.1	.2	.3	.4	.5	.6	.7	.8	.9	1
.1	72.6292%	87.0500%	99.1272%	99.7212%	99.7333%	99.6687%	99.5879%	99.4949%	99.4464%	99.3899%
5	75.4536%	98.9818%	99.8707%	**99.8747%**	99.8545%	99.8667%	99.8707%	99.8505%	99.8424%	99.8101%
50	75.4536%	98.9818%	99.8707%	99.8707%	99.8465%	99.8384%	99.8101%	99.8222%	99.8343%	99.8263%
500	75.4536%	98.9818%	99.8707%	99.8707%	99.8424%	99.7858%	99.7576%	99.7576%	99.7576%	99.7454%
1000	75.4536%	98.9818%	99.8707%	99.8707%	99.8424%	99.7858%	99.7576%	99.7252%	99.7374%	99.7333%
5000	75.4536%	98.9818%	99.8707%	99.8707%	99.8424%	99.7858%	99.7576%	99.7252%	99.7131%	99.6970%
10000	75.4536%	98.9818%	99.8707%	99.8707%	99.8424%	99.7858%	99.7576%	99.7252%	99.7131%	99.6970%

Nous visualisons les résultats de la méthode Grid-Search avec σ_{opt}=0.4, C_{opt}=5, ils sont obtenus en maximisant le TCC_{vc}=99.87%.

2. Partition2 :

– Méthode1 (A_{max}, C_{vc}) :

TAB V.5

La sélection de modèle avec la première méthode pour la partition *P2*.

σ	.1	.2	0.3	.4	.5	.6	.7	**.8**	.9	1
A	0.0044	0.0504	0.1728	0.2714	0.3260	0.3501	0.3589	**0.3607**	0.3596	0.3573

$(\sigma_{opt}$=**0.8**,$C)$	.1	5	**50**	500	1000	5000	10000
TCC_{vc}(%)	99.4346	99.8729	**99.8857**	99.8142	99.784	99.7411	99.7411

Les meilleurs hyper-paramètres sont obtenus avec σ_{opt} = 0.8 pour A_{max} = 0.36 et C_{opt} = 50 pour TCC_{vc}=99.88%.

– Méthode2 (Q^*_{min}, C_{vc}) :

TAB V.6

La sélection de modèle avec la deuxième méthode pour la partition *P2*.

σ	.1	.2	**.3**	**.4**	.5	.6	.7	.8	.9	1
Q ($\times 10^6$)	0.03554	0.005363	**0.001669**	0.002094	0.004563	0.01031	0.022	0.04385	0.08211	0.1457
$\overline{Q}$ ($\times 10^{-4}$)	1.1731	33.889	335.83	1151.7	2428.7	4026.8	5866.4	7920.7	10186	12668
$Q^*_{\min}$ ($\times 10^6$)	303	1.5827	0.04971	**0.01818**	0.01879	0.02562	0.0375	0.05536	0.08061	0.115
ERR_{test}(%)	54.8126	54.659	52.8015	50.286	48.6637	48.5587	48.4639	48.7027	48.8486	49.2775
(σ_{opt}=0.4,C)	.1	**5**	50	500	1000	5000	10000			
TCC_{vc}(%)	99.7538	**99.9238**	99.8888	99.8872	99.8872	99.8872	99.8872			

L'erreur de test due à ce critère Q est égale à 52.8% pour σ_{opt}=0.3 est donc souvent beaucoup plus grande que l'erreur de test minimum qui est égale à 48.46% pour σ_{opt}=0.7. Le critère Q^* est plus performant par rapport à Q car il permet une diminution de l'erreur de généralisation avec 50.28% pour σ_{opt}=0.4.

– Méthode3 (VC) :

TAB V.7

La sélection de modèle avec la troisième méthode pour la partition *P2*.

(c, σ)	.1	.2	.3	**.4**	.5	.6	.7	.8	.9	1
.1	78.677%	93.0199%	99.3806%	99.7538%	99.73%	99.649%	99.5648%	99.4346%	99.3123%	99.1868%
5	80.9084%	99.3631%	99.9095%	**99.9238%**	99.919%	99.9047%	99.8777%	99.8729%	99.8507%	99.8459%
50	80.9084%	99.3631%	99.9015%	99.8888%	99.8936%	99.8872%	99.8872%	99.8857%	99.8888%	99.8888%
500	80.9084%	99.3631%	99.9015%	99.8872%	99.8539%	99.7935%	99.8078%	99.8142%	99.8189%	99.838%
1000	80.9084%	99.3631%	99.9015%	99.8872%	99.8539%	99.7951%	99.7602%	99.784%	99.7888%	99.8078%
5000	80.9084%	99.3631%	99.9015%	99.8872%	99.8539%	99.7951%	99.7586%	99.7411%	99.7348%	99.7252%
10000	80.9084%	99.3631%	99.9015%	99.8872%	99.8539%	99.7951%	99.7586%	99.7411%	99.7348%	99.7205%

Nous visualisons les résultats de la méthode Grid-Search avec σ_{opt}=0.4, C_{opt}=5, ils sont obtenus en maximisant le TCC_{vc}=99.92%.

3. Partition3 :

– Méthode1 (A_{max}, C_{vc}) :

TAB V.8

La sélection de modèle avec la première méthode pour la partition *P3*.

σ	.1	.2	.3	**.4**	.5	.6	.7	.8	.9	1
A	0.0036	0.0763	0.1726	**0.1976**	0.1792	0.1501	0.1232	0.1015	0.0847	0.0717
(σ_{opt}=0.4,C)	.1	**5**	50	500	1000	5000	10000			
TCC_{vc}(%)	98.7645	**99.4636**	99.3228	99.2528	99.2528	99.2528	99.2528			

Les meilleurs hyper-paramètres sont obtenus avec σ_{opt} = 0.4 pour A_{max} = 0.19 et C_{opt} = 5 pour TCC_{vc}=99.46%.

– Méthode2 (Q^*_{min}, C_{vc}) :

TAB V.9

La sélection de modèle avec la deuxième méthode pour la partition *P3*.

σ	.1	.2	**.3**	**.4**	.5	.6	.7	.8	.9	1
Q ($\times 10^7$)	0.01014	0.00167	**0.001206**	0.0038	0.01409	0.0472	0.1319	0.2762	0.4799	0.7196
$\overline{Q}$ ($\times 10^{-3}$)	0.0428	2.5642	28.5853	101.69	221.12	378.31	567.75	786.94	1034.9	1311.3
Q^*_{min} ($\times 10^8$)	23.688	0.06517	0.00422	**0.00374**	0.006372	0.01247	0.02324	0.03509	0.04637	0.05487
ERR_{test}(%)	48.5811	42.9605	38.3615	36.3371	35.5366	35.0792	34.6205	34.7308	34.4189	34.4243
(σ_{opt}=0.4,C)	.1	**5**	50	500	1000	5000	10000			
TCC_{vc}(%)	98.7645	**99.4636**	99.3228	99.2528	99.2528	99.2528	99.2528			

L'erreur de test due à ce critère Q est égale à 38.36% pour σ_{opt}=0.3 est donc souvent beaucoup plus grande que l'erreur de test minimum qui est égale à 34.41% pour σ_{opt}=0.9. Le critère Q^* est plus performant par rapport à Q car il permet une diminution de l'erreur de généralisation avec 36.33% pour σ_{opt}=0.4.

Bien que nos expérimentations aient montré que le nouveau critère donne de meilleures généralisations par rapport au critère *Q*, cependant l'erreur de test n'est pas encore proche du minimum que nous aimerions avoir. Ceci en est la cause de plusieurs raisons. Par exemple, la mesure de séparabilité des classes utilisée dans le nouveau critère *Q** emploi seulement les moyennes et les variances des classes (les deux premiers ordres), c.-à-d, on considère que les nuages de points associés à chaque groupe sont distribués selon une loi normale multidimensionnelle. Une hypothèse supplémentaire est que les nuages conditionnels doivent avoir une forme identique, uni-modale. En général, les distributions des classes sont multimodales.

– **Méthode3 (*VC*) :**

TAB V.10. La sélection de modèle avec la troisième méthode pour la partition *P3*.

(c, σ)	.1	.2	.3	.4	.5	.6	.7	.8	.9	1
.1	54.9688%	90.0591%	98.386%	98.7645%	98.5138%	98.2404%	97.8985%	97.5168%	97.1196%	96.7224%
5	76.9212%	99.0429%	**99.5222%**	99.4636%	99.3578%	99.2325%	99.1186%	98.9826%	98.8223%	98.7018%
50	76.9212%	99.0355%	99.4604%	99.3228%	99.2992%	99.3123%	99.2789%	99.2235%	99.1576%	99.0738%
500	76.9212%	99.0355%	99.4571%	99.2528%	99.0575%	99.0201%	99.0844%	99.1674%	99.2065%	99.2008%
1000	76.9212%	99.0355%	99.4571%	99.2528%	99.0258%	98.9452%	98.9859%	99.0868%	99.1438%	99.1828%
5000	76.9212%	99.0355%	99.4571%	99.2528%	99.0241%	98.8589%	98.8174%	98.8687%	98.9346%	99.0347%
10000	76.9212%	99.0355%	99.4571%	99.2528%	99.0241%	98.8597%	98.7604%	98.7726%	98.8443%	98.9167%

Nous visualisons les résultats de la méthode Grid-Search avec σ_{opt}=0.3, C_{opt}=5, ils sont obtenus en maximisant le TCC_{vc}=99.52%.

TAB. V.11. Comparaison entre les trois méthodes pour les trois partitions.

Durée (min)		Nombre d'échantillons		Méthode de recherche	(σ_{opt} c_{opt})		Temps de calcul nécessaire pour réglage des paramètres			TCC(%) [training]	Nb VS	TCC(%)
App	Test	App	Test		σ_{opt}	c_{opt}	$T(\sigma_{opt})$	$T(C_{opt})$	T_{total}			
4.1	42.4	24749	245911	1(A_{max}, C_{vc})	0.6	5	17.39min	2.9min	20.29min	99.94	457	59.48
				2(Q^*_{min}, C_{vc})	0.5	5	44.78min	3.77min	48.55min	99.97	687	59.16
				3(*VC*)	0.4	5			12.92h	99.99	1933	57.16
10.45	36.05	62965	207695	1(A_{max}, C_{vc})	0.8	50	1.93h	18min	2.23h	99.97	491	52.54
				2(Q^*_{min}, C_{vc})	0.4	5	5.21h	1.34h	7.55h	99.99	2491	49.71
				3(*VC*)	0.4	5			4.77jours	99.99	2491	49.71
20.43	26.06	122866	147794	1(A_{max}, C_{vc})	0.4	5	7.98h	2.78h	10.76h	99.85	6565	63.66
				2(Q^*_{min}, C_{vc})	0.4	5	21.54h	2.78h	1.01jours	99.85	6565	63.66
				3(*VC*)	0.3	5			11.8jours	99.95	15638	61.63

Ici nous comparons trois approches différentes **1**, **2**, **3** pour trois partitions de données différentes pour parvenir au même objectif : évaluer la capacité de chaque méthode à établir le bon modèle. Un aspect important pour comparer les différentes méthodes testées est la complexité de temps de calcul total qu'elles induisent pour le réglage des hyper-paramètres et ceux pour les trois partitions.

Les méthodes à noyaux **1** et **2** ne nécessitent pas un aussi long apprentissage des hyperparamètres, par rapport à la méthode **3** qui est lourde en temps de calcul, où la recherche des meilleurs paramètres peut requérir des phases d'apprentissages assez longues. Plus précisément la méthode **1** est la plus efficace car elle est indépendante de l'entrainement de la méthode d'apprentissage et par conséquent elle est peu couteuse en temps.

Concernant la précision de classification, les résultats sont synthétisés dans le tableau V.11. On remarque que le meilleur taux de classification pour les trois partitions de données est atteint avec la méthode **1** avec **TCC1=59.48%**, **TCC1=52.54%** et **TCC1=63.66%**, par rapport à la méthode3 avec les plus faibles performances à savoir **TCC3=57.16%, TCC3=49.71%, TCC3=61.63%**.

Donc la méthode **1** est la plus adaptée à la recherche des meilleurs paramètres, et son succès est dû majoritairement à son utilisation du **critère d'alignement** qui fixe le paramètre optimal sigma, ce qui est crucial sur les performances de la méthode d'apprentissage. Après vient la recherche du paramètre *C* qui dépend de façon importante du paramètre noyau fixé (principe de *Line-Search*). SVM est beaucoup plus sensible à ce dernier qu'au paramètre de régularisation *C*.

Nous remarquons également dans le tableau V.11 que les TCC sur l'ensemble d'apprentissage (training set) sont globalement dégradés avec la méthode **1**, **TCC1[training]= 99.94%**, **TCC1[training]= 99.97%** et **TCC1[training]= 99.85%**, par rapport aux deux autres méthodes présentant un ordre de **sur-apprentissage** (over-fitting) des données relativement plus élevé. Par conséquent la méthode **1** utilisant le critère d'alignement a un meilleur pouvoir de généralisation de performances, donnant ainsi de meilleurs résultats sur les données test, indépendant de la méthode d'apprentissage, par conséquent c'est une méthode robuste pour la sélection de modèle.

Pour ce qui est de la comparaison des TCC par rapport aux trois partitions des trois méthodes, les meilleures performances sont obtenues avec la plus grande durée d'entrainement de 20.43minutes pour laquelle correspond respectivement **6565**(méthodes1, 2), **15638**(méthode3) vecteurs de support (VS). Au principe les performances de classification augmentent avec la durée de l'ensemble d'apprentissage. Cependant cette hypothèse n'est pas vérifiée dans notre cas, nous voyons bien que *P1* donne de meilleurs résultats par rapport à *P2*. Ceci peut s'expliquer par le fait que dans cette dernière nous avons effectué un apprentissage de 2 minutes de musique contre 8 minutes de parole, les résultats sont détériorés par un training set non équilibré. On conclut donc que dans certains cas, l'augmentation de la durée du fichier d'entrainement n'améliore pas forcement les performances du classificateur, donc il faudrait essayer plusieurs partitions et choisir celle donnant la meilleure précision.

Il existe un autre résultat concernant la relation entre le paramètre σ, le nombre de vecteurs de support et la précision de généralisation TCC.

L'examen du nombre des vecteurs de supports utilisés dans les différents cas étudiés donne une indication sur la configuration des données dans l'espace augmenté par le noyau RBF utilisé. Les nombres de vecteurs de support (resp. les TCC) résultants de la classification SVM par la méthode **1** pour les trois partirions sont inferieures (resp. supérieures) à ceux obtenus avec les deux autres méthodes, sauf exception dans la partition3 dans laquelle $\mathbf{Nb_{VS1} = Nb_{VS2} = 6565}$ (resp. **TCC1= TCC2= 63.66%)**.

Conclusion, plus le nombre de vecteurs supports est petit, meilleure est la classification. Ceci s'explique par le fait que quand l'hyperplan est difficile à déterminer, beaucoup de vecteurs de support doivent être utilisés (risque de sur-apprentissage); la méthode **3** illustre parfaitement cette situation avec un sigma très petit, et donc une grande complexité du modèle, alors que peu de vecteurs supports (méthode **1**), indique que l'hyperplan est relativement facile à identifier et par conséquent, les performances seront relativement bonnes. On va utiliser le terme "complexité spatiale" du modèle quand nous voulons décrire les charges de calcul nécessaire liée à l'utilisation mémoire pour stoker les VS par le fait que la fonction de décision ne dépend que de ces points de support et pas des autres données. La complexité spatiale de l'algorithme de résolution des SVM croit très sensiblement avec le nombre d'observations **(24749, 62965, 122866)** et ceux pour les trois méthodes $\mathbf{(Nb_{VS1} < Nb_{VS2} < Nb_{VS3})}$.

Nous nous sommes enfin intéressés aux temps **T1(C_{opt1})= 2.9 min; T2(C_{opt1})=18 min; T3(C_{opt1})= 2.78 h** ; qui sont les temps de calculs nécessaires pour la recherche du meilleur C_{opt1} pour σ_{opt1} fixé avec la méthode d'alignement, ils sont assez faibles. Au premier coud d'œil nous voyons que plus le paramètre σ est optimal, moins les temps de calculs sont longs et ce quelque soit la partition de données, ceci est du au fait qu'en choisissant le bon sigma, la validation croisée utilisée pour trouver le paramètre C avec une ligne de recherche de (1×7) crée pour l'apprentissage des modèles moins complexes (Nb VS petit) et donc des temps calculatoires relativement faibles. Dans notre cas il y aura 5 modèles pour chaque paramètre C.

TAB V.12

Comparaison entre TCC **(Musique(M))** et TCC **(Parole(P))** pour les trois méthodes.

	Méthode 1			**Méthode2**			**Méthode3**		
	TCC(%)	**TCC_M(%)**	**TCC_P(%)**	**TCC(%)**	**TCC_M(%)**	**TCC_P(%)**	**TCC(%)**	**TCC_M(%)**	**TCC_P(%)**
P1	59.4898	17.6961	99.6435	59.1649	17.0704	99.6077	57.1613	12.9482	99.6395
P2	52.5405	13.6073	99.7666	49.7142	8.4256	99.7975	49.7142	8.4256	99.7975
P3	63.6629	28.5789	96.9374	63.6629	28.5789	96.9374	61.6385	23.9043	97.4266

Par ailleurs, on constate d'après le tableau V.12 que le classificateur commet beaucoup plus d'erreurs pour la classification de la musique par rapport à la classification de la parole, ceci peut être vérifié en comparant la classe connue d'une instance test avec le résultat du modèle. Ce phénomène peut être expliqué par l'existence d'une grande variété des instruments de musique, aussi la paramétrisation de type MFCC seule n'est pas suffisante pour la représentation de la classe musique. Pour la parole les **TCC** sont très performants. En effet, les MFCC sont quasiment incontournables en reconnaissance de la parole car ils modélisent très bien la parole.

V-4-2- Discussion :

Nous retenons des tests présentés plus haut que dorénavant le réglage des hyperparamètres peut s'effectuer à l'aide d'une nouvelle méthode novatrice en matière de classification basée sur le critère d'alignement de noyaux; ce qui permet à la fois d'améliorer la précision; et le temps d'exécution CPU.

V-4-3- Sélection des descripteurs :

Nous proposons ici une étude comparative des performances des méthodes à noyaux avec des algorithmes récents de sélection d'attributs (ASA), décrits précédemment et de leur comportement (complexité) et leur efficacité en relation avec la technique de classification SVM. Les algorithmes que nous avons utilisé sont issus de la littérature et appartenant soit à l'approche *filtrage* (CLDA, KCS, Alignement, Corrélation, Entropie) ou *enveloppage* (SVM-RFE).
La plupart des approches proposées pour la sélection d'attributs sont des approches filtres qui se basent sur la similarité ou la redondance des attributs, ces derniers sont évalués individuellement afin d'avoir une complexité algorithmique raisonnable.

L'objectif est de montrer leurs capacité à retrouver d'une part le bon ordre d'attributs, et d'autre part le "bon modèle", au sens du bon nombre d'attributs à conservé. L'effet de la taille du nombre d'attributs sur les performances de classification est aussi analysé.

Pour interpréter les résultats et comparer les algorithmes, nous nous sommes basés sur les critères suivants :

- Le critère d'arrêt : Dés que le nombre total d'attributs est atteint.
- Taux de reconnaissance sur un fichier similaire mais différent de celui ayant servi à l'apprentissage.
- Nombre de paramètres à fixer : Dans un premier temps, il y a des paramètres à régler pour le déroulement des différents ASA. Les deux algorithmes de sélection d'attributs non linéaires à savoir KCS et le critère d'Alignement utilisent comme paramètre le noyau gaussien σ donc ce dernier doit être fixé (méthode d'alignement). Pour l'algorithme SVM-RFE, la valeur du paramètre pénalisation C=1000, a été fixé à une valeur suffisamment élevée pour que l'erreur d'apprentissage reste faible.
- Ensuite pour l'évaluation nous cherchons le couple (σ_{opt}, C_{opt}) à l'aide de la méthode de recherche **1** qui va nous aider à valider le plus petit sous-ensemble obtenu avec les meilleures performances en termes de classification et cela pour tout les ASA.

Pour comparer les méthodes de sélection décrites ci-dessus nous avons utilisé les données audio pour la validation des méthodes ASA. L'annexe **B** présente la procédure de sélection d'attributs

en détails et les résultats sur une autre base de données bioinformatique *UCI* accessible sur Internet [25, 26].

V-4-3-1- Corpus utilisé :

Les expérimentations présentées ici ont été menées sur un corpus audio. Afin d'expérimenter un grand nombre de variations sur les algorithmes dans des délais acceptables, nous allégeons la charge totale de calcul en considérant la base de donnée employée dans la première partie pour la sélection de modèle de taille réduite d'une durée 10minutes avec une partition comprenant deux classes (9 fichiers chacune) parole-musique pour un total de 26998 données d'apprentissage (4.5minutes) et 32998 données de test (5.5minutes). Les données de la base d'apprentissage et de test ont été aléatoirement sélectionnées, l'ensemble d'entrainement est non équilibré numériquement 1.83minutes de parole contre 2.66minutes de musique, voir le tableau V.13. L'étape de la sélection d'attributs est appliquée à l'ensemble des données paramétrées et normalisées entre [-1, 1] pour la base d'apprentissage et de test.

TAB V.13

Création de la base de données pour la sélection de descripteurs.

Fichier (.Wav)	Durée (min : sec)	App	Test
Parole (1)	0 :10	×	
Musique (1)	0 :30	×	
Parole (2)	0 :20	×	
Musique (2)	0 :40	×	
Parole (3)	0 :20	×	
Musique (3)	0 :30	×	
Parole (4)	0 :30	×	
Musique (4)	0 :50	×	
Parole (5)	0 :10	×	
Musique (5)	0 :10	×	
Parole (6)	0 :20	×	
Musique (6)	0 :40		×
Parole (7)	1 :00		×
Musique (7)	0 :50		×
Parole (8)	1 :20		×
Musique (8)	1 :00		×
Parole (9)	0 :30		×
Musique (9)	0 :10		×

En premier lieux la sélection sera effectuée sur 16 attributs correspond à une situation que nous pourrons qualifier d'intermédiaire : le nombre d'attributs disponibles n'est pas très élevé, cependant la puissance de calcul des machines, sans cesse grandissante, nous invite de plus en plus à considérer une sélection d'un grand nombre d'attributs (jusqu'à une centaine). Par la suite on augmente le nombre d'attributs jusqu'à 154, ceux-ci sont issus de 6 descripteurs différents (voir tableau V.14).

TAB V.14

Récapitulation des descripteurs utilisés. Les descripteurs sont regroupés par paquets. Au total nous obtenons 154 attributs.

Descripteur ou paquet d'attributs	Taille	Synopsis
MFCC16_D_A_E	51	Coefficients cepstraux plus l'énergie et leurs dérivées premières et secondes
FBANK20	20	Coefficients spectraux suivant une échelle linéaire
LPC20	20	Coefficients du model autorégressif (AR) à l'ordre 20 du signal
LPCEPSTRA10_D_A_E	33	Coefficients cepstra issus de la prédiction linéaire plus l'énergie et leurs dérivées premières et secondes
LPREFC10	10	Coefficients de réflexion
MELSPEC20	20	Coefficient spectraux sur une échelle MEL

V-4-3-2- Evaluation des ASA en fonction de la taille des descripteurs :

– *Pour d=16* :

Les meilleurs hyperparamètres sont obtenus avec σ_{opt} = 0.4 pour A_{max} = 0.25 et C_{opt} = 5 pour TCC_{vc}=95.36%.

TAB V.15

La sélection de modèle pour la base de données audio avec *d=16*.

σ	.1	.2	.3	**.4**	.5	.6	.7	.8	.9	1
A	0.0068	0.0933	0.2179	**0.2568**	0.2474	0.2206	0.1916	0.1658	0.1445	0.1272
$(\sigma_{opt}=0.4,C)$	.1	**5**	50	500	1000	5000	10000			
$TCC_{vc}(\%)$	98.5443	**99.3629**	99.2259	99.1666	99.1666	99.1666	99.1666			

Nous obtiendrons TCC=85.27% (95.72% de parole + 74.17% de musique). Les performances de classification sont assez bonnes, on remarque même une augmentation dans les performances de classification de musique par rapport à ceux obtenu dans la partie sélection de modèle.

Cette augmentation de performance est due à l'utilisation d'une base de donnée avec un ensemble d'entrainement relativement équilibré comparativement à celui utilisé dans la sélection de modèle (partition1 : avec 2.98minutes de parole contre 1.11minutes de musique) qui a donné un TCC=59.48% (99.64% de parole + 17.69% de musique). Afin de confirmation, nous allons recalculer le TCC pour la première partition de donnée utilisée dans la sélection de modèle en changeant seulement l'ensemble d'entrainement par celui utilisé dans cette partie et on trouve TCC=81.07%(93.86% de parole + 67.76% de musique).

TAB V.16

Contribution des attributs par les différentes méthodes de sélection (*d*=16).

ASA	Ordre d'importance des attributs															
CLDA	12	9	2	6	14	4	11	7	13	8	3	5	10	16	15	1
KCS (σ=0.4)	2	12	9	3	6	4	14	7	8	13	11	5	10	1	16	15
Alignement (σ=0.4)	2	3	1	4	7	6	8	5	16	15	9	10	12	11	14	13
Corrélation	2	3	4	7	8	15	10	16	12	1	14	5	6	9	13	11
Entropie	6	12	9	5	4	14	8	13	11	16	15	2	7	3	1	10
SVM-RFE (C=1000)	2	7	3	15	12	1	16	8	6	13	14	5	11	10	9	4

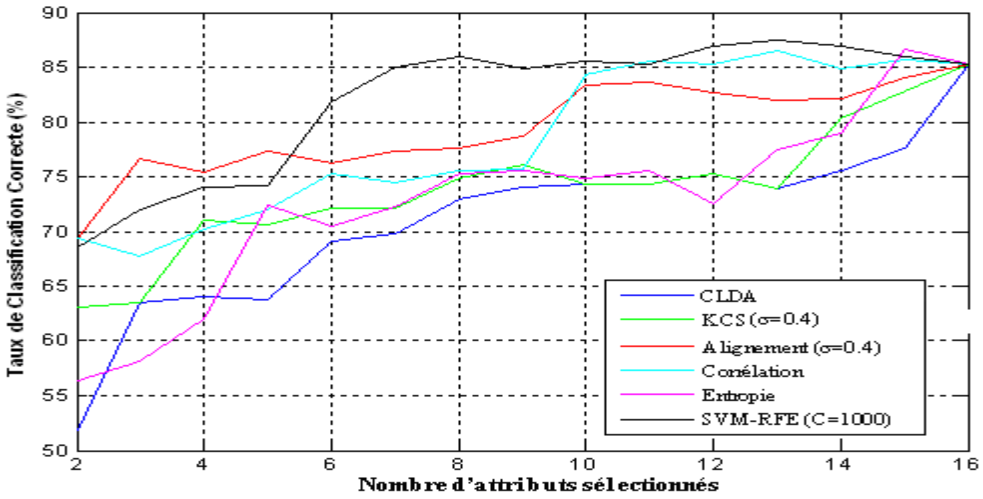


FIG V.3. Taux de classification correcte avec le nombre d'attributs varie de 2 à 16.

A partir des figures V.3, .4, .5, nous observons des améliorations dans : les performances de classification (TCC); la complexité du modèle (VS) et le cout de l'apprentissage (T), obtenues avec les algorithmes de sélection du tableau suivant :

TAB V.17

Les algorithmes présentant des améliorations de performances par rapport à l'ensemble complet d'attributs.

	Nb d'attributs	TCC(%)	VS	T(s)
Corrélation	A11	85.49	1869	18.5
	A13	86.44	1982	30.26
	A15	85.69	2450	45.16
Entropie	A15	86.68	2453	43.97
SVM-RFE	A8	85.99	2173	14.39
	A10	85.55	1885	15.9
	A12	86.89	1950	24.86
	A13	87.36	2183	33.86
	A14	86.96	2435	42.98
	A15	85.96	2666	49.21
Nb total d'attributs	**A16**	**85.27**	**2842**	**54.58**

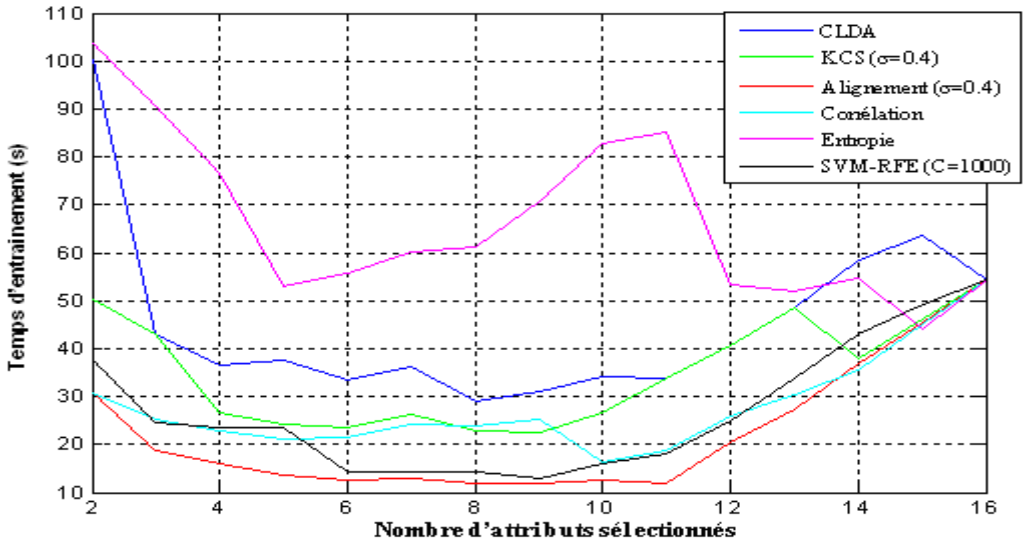


FIG V.4. Temps d'entrainement en fonction du nombre d'attributs sélectionnés.

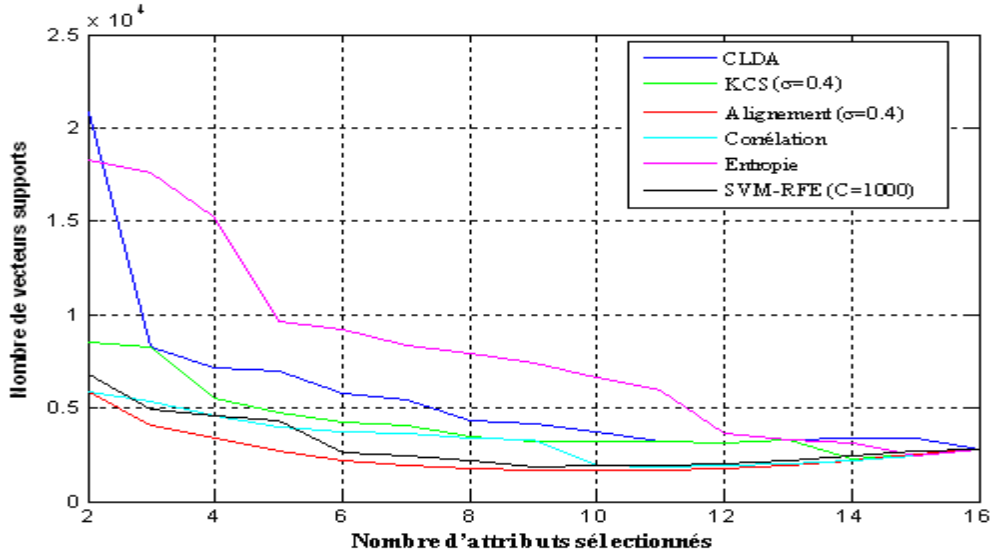


FIG V.5. Nombre de vecteurs supports en fonction du nombre d'attributs sélectionnés.

SVM-RFE exhibe la plus forte augmentation de performances : en réduisant le nombre d'attributs sélectionnés à 13, le taux de reconnaissance s'élève à 2.1%.

Les performances maximales obtenues en effectuant la sélection avec le critère d'alignement sont globalement inferieures à celles atteintes en exploitant l'intégralité des données, A15**(TCC=84.11%, VS=2496, T=45.50s)**, tout comme la **CLDA [**A15**(TCC=77.60%, VS=3394, T=63.37s)]** et **KCS[**A15**(TCC=82.84%, VS=2537, T=46.12s)]**. Néanmoins l'alignement présente une certaine robustesse car si on admet une baisse du taux de

reconnaissance autour de <2%, seulement 10 attributs peuvent être utilisés lors de la classification, ceci est valable pour le critère de corrélation avec aussi 10 attributs sans que les performances de la règle de classement diminuent trop.

La complexité d'entrainement effectuée sur les groupes d'attributs sélectionnés par les différents ASA est une alternative intéressante qu'on cherche à alléger (classification effectuée sur moins d'exemples(VS) et donc des temps d'entrainements relativement faibles), il ne faut pas surtout confondre avec la complexité d'un ASA qui est le temps écoulé lors de la sélection. Nous remarquons d'après la figure V.5, que les algorithmes de sélection à savoir l'alignement, Corrélation, SVM-RFE présentent une complexité d'entrainement moindre à celles des autres approches ne dépassant pas les VS = 6789, et ceux pour les différents sous ensembles d'attributs sélectionnés. Ceci s'explique pourquoi ces trois algorithmes admettent en général de meilleurs taux de reconnaissance en fonction du nombre d'attributs sélectionnés (voir figure V.3), car plus le nombre de vecteurs de support est réduit plus les capacités de généralisation de l'hyperplan construit sont meilleures. En outre les ASA (CLDA, KCS, Entropie) pour les différents sous ensembles d'attributs sélectionnés produisent des taux de reconnaissances dégradés (voir figure V.3) avec des nombres de vecteurs supports considérables (voir figure V.5) et par conséquent des temps d'entrainements importants (voir figure V.4). Notons cependant que les deux algorithmes CLDA, KCS pour les différentes solutions d'attributs sélectionnés A3, A10, A11, A12, A13 ont des TCC aux performances égales, alors que pour les autres sous ensembles d'attributs, KCS est meilleur.

Au-delà de ce fait, nous remarquons d'après les figures V.3, .4, .5 que l'approche Entropie a une dégradation beaucoup plus nette des taux de classification et donc une complexité d'entrainement importante même s'il donne de bons résultats pour A15 mais cela reste insuffisant puisque le TCC ne dépasse pas 78.87% pour A14.

Le tableau V.18 donne les temps CPU relatifs au déroulement des différents ASA. L'approche basée sur le critère Corrélation s'avère nettement avantageuse car elle réalise un excellent compromis performances/complexité. Soulignons que l'algorithme SVM-RFE présente une complexité largement supérieure à celles des autres approches, ce qui n'est pas souhaitable lors de la sélection d'attribut.

TAB V.18

Complexité des ASA en termes de temps.

ASA	Temps
CLDA	47.84sec
KCS	10.20 min
Alignement	10.33 min
Corrélation	0.11 sec
Entropie	14.73 sec
SVM-RFE	2.61 h

Nous nous intéressons maintenant à un nombre d'attributs plus élevé (d=154), on présente tout d'abord l'impact d'une nouvelle stratégie de paramétrisation acoustique décrite dans le tableau V.14 sur les performances de classification, ensuite on fait une sélection d'attributs avec les différents algorithmes présentés si dessus avec la même procédure de sélection et de validation utilisée précédemment (d=16).

– *Pour d=154* :

TAB V.19

La sélection de modèle pour la base de données audio avec *d=154*.

σ	.1	.2	.3	.4	.5	.6	.7	.8	.9	1
A	0.0061	0.0061	0.0062	0.0069	0.0142	0.0528	0.1366	0.2149	0.2648	0.2943

(σ_{opt}=1.2,C)	.1	**5**	50	500	1000	5000	10000
TCC_{vc}(%)	98.4740	**99.8667**	99.8667	99.8667	99.8667	99.8667	99.8667

σ	1.1	**1.2**	1.3	1.4	1.5
A	0.3098	**0.3153**	0.3134	0.3065	0.2962

La précision de classification dans l'étage de validation croisée est la même pour $C \in \{5\ 50\ 500\ 1000\ 5000\ 10000\}$ à savoir 99.86%. Nous choisissons C=5. Donc le meilleur couple obtenu est (σ_{opt}=1.2, C_{opt}=5) qui donne un TCC=95.37%(99.17% de parole + 91.33% de musique). Nous observons une amélioration significative de

10.09%(3.44%parole+17.15%musique) avec cette nouvelle configuration, en termes de taux de reconnaissance, par rapport à la paramétrisation précédente 16MFCC.

Bien que l'ajout de ces caractéristiques permette en effet d'améliorer les performances globales de classification, on ne peut s'empêcher de noter certains effets néfastes dus à cette augmentation. En effet, le nombre de caractéristiques ne revêt plus la même importance et la tendance est à en utiliser autant que possible tant que les performances de classification sont augmentées. Toutefois, la dimensionnalité de l'espace dans lequel est entrainé le classifieur peut avoir une incidence cruciale sur ses performances, relativement au nombre de variables utilisées. La complexité en termes de temps de calcul (espace de calcul) du processus d'entrainement est un autre aspect important qu'on cherche à alléger. Nous essayerons de sélectionner un nombre réduit d'attributs sans que les performances de classification baissent trop voire même augmentent.

TAB V.20

Contribution des attributs par les différentes méthodes de sélection (d=154).

ASA	Ordre d'importance des attributs
CLDA	80, 78, 88, 87, 148, 82, 76, 79, 153, 150, 81, 89, 84,154, 77, 143, 74, 151, 146, 83, 144, 152, 145, 75, 140, 85, 139, 103, 149, 34, 113, 51, 124, 135, 86, 20, 19, 147, 117, 114, 141, 115, 50, 23, 18, 106, 119, 21, 37, 104, 36, 142, 90, 38, 35, 40, 105, 107, 108, 30, 33, 22, 116, 27, 29, 123, 118, 45, 12, 26, 9, 91, 44, 24, 41, 43, 46, 112, 39, 2, 28, 25, 111, 110, 138, 42, 122, 48, 31, 100, 120, 47, 73, 121, 95, 98, 109, 6, 14, 136, 137, 96, 4, 11, 32, 7, 13, 8, 72, 101, 3, 49, 93, 92, 5, 134, 99, 10, 16, 131, 94, 130, 97, 15, 133, 132, 1, 129, 60, 61, 62, 68, 128, 63, 69, 127, 67, 58, 59, 65, 57, 64, 56, 70, 66, 17, 102, 71, 126, 55, 54, 125, 53, 52
KCS (σ=1.2)	80, 78, 88, 148, 87, 82, 153, 76, 79, 150, 81, 154, 84, 89, 143, 77, 151, 144, 74, 146, 152, 83, 145, 140, 75, 85, 139, 149, 103, 34, 113, 51, 124, 135, 147, 19, 20, 86, 141, 117, 114, 18, 142, 115, 23, 50, 106, 119, 21, 37, 36, 104, 90, 35, 38, 40, 105, 107, 108, 30, 22, 33, 116, 27, 29, 118, 123, 45, 12, 26, 9, 91, 24, 44, 2, 41, 43, 46, 39, 112, 138, 25, 28, 111, 110, 42, 122, 100, 48, 31, 120, 95, 73, 47, 98, 121, 137, 109, 6, 136, 14, 96, 4, 72, 11, 32, 3, 7, 13, 8, 101, 92, 49, 93, 5, 134, 99, 94, 10, 131, 16, 130, 97, 15, 133, 132, 1, 129, 60, 62, 61, 128, 63, 68, 58, 127, 59, 69, 67, 57, 65, 64, 56, 126, 70, 66, 17, 102, 71, 55, 125, 54, 53, 52
Alignement (σ=1.2)	58, 59, 57, 60, 62, 3, 96, 2, 61, 63, 56, 125, 64, 128, 72, 95, 126, 92, 55, 65, 100, 127, 1, 99, 4, 94, 131, 68, 66, 69, 7, 67, 54, 132, 97, 98, 53, 130, 129, 136, 75, 8, 70, 35, 73, 133, 93, 17, 102, 6, 134, 18, 74, 36, 116, 114, 105, 19, 51, 124, 39, 103, 34, 113, 24, 101, 38, 22, 21, 104, 41, 71, 5, 20, 37, 42, 25, 81, 79, 40, 115, 23, 83, 33, 107, 76, 118, 16, 77, 141, 47, 106, 26, 43, 109, 28, 46, 50, 45, 27, 44, 29, 117, 48, 120, 15, 154, 30, 31, 145, 153, 49, 152, 85, 108, 32, 52, 140, 86, 10, 149, 137, 78, 110, 82, 9, 119, 151, 147, 84, 121, 80, 111, 146, 122, 148, 150, 135, 139, 88, 12, 144, 89, 87, 143, 91, 142, 138, 90, 11, 112, 123, 14, 13
Corrélation	2, 96, 59, 58, 3, 57, 60, 95, 62, 61, 100, 63, 72, 75, 92, 81, 74, 64, 79, 56, 141, 76, 128, 99, 4, 83, 140, 77, 146, 142, 7, 131, 143, 145, 126, 98, 65, 97, 101, 125, 144, 55, 80, 82, 73, 136, 8, 85, 86, 78, 68, 15, 94, 84, 147, 132, 134, 67, 139, 69, 133, 91, 127, 88, 53, 10, 130, 66, 16, 90, 138, 12, 1, 154, 93, 153, 89, 54, 148, 14, 135, 150, 87, 5, 17, 102, 70, 52, 151, 137, 6, 152, 9, 13, 149, 129, 36, 71, 49, 114, 118, 121, 11, 37, 116, 50, 40, 38, 122, 35, 19, 44, 117, 42, 120, 47, 123, 105, 103, 41, 48, 39, 28, 119, 45, 106, 46, 43, 18, 24, 104, 33, 29, 25, 108, 32, 34, 113, 31, 20, 110, 30, 27, 111, 51, 124, 26, 23, 22, 21, 107, 115, 109, 112

Entropie	75, 140, 154, 89, 148, 153, 143, 150, 146, 76, 74, 139, 144, 21, 50, 69, 147, 79, 51, 106, 104, 35, 70, 82, 151, 86, 142, 117, 149, 115, 114, 152, 9, 19, 33, 145, 123, 37, 138, 77, 46, 36, 113, 118, 39, 88, 137, 136, 110, 42, 141, 16, 48, 135, 108, 83, 120, 12, 107, 84, 73, 62, 111, 103, 6, 30, 26, 45, 109, 24, 25, 91, 100, 112, 41, 34, 2, 31, 38, 18, 4, 80, 43, 121, 14, 23, 59, 53, 11, 67, 49, 40, 92, 116, 47, 8, 5, 54, 61, 93, 10, 97, 58, 105, 7, 55, 32, 44, 134, 98, 52, 29, 96, 71, 81, 68, 130, 72, 63, 131, 132, 87, 20, 90, 3, 94, 119, 85, 15, 65, 102, 57, 133, 99, 28, 95, 126, 13, 66, 64, 129, 122, 27, 60, 1, 78, 101, 128, 127, 56, 124, 22, 125, 17
SVM-RFE (C=1000)	69, 16, 15, 12, 10, 14, 9, 11, 13, 70, 71, 6, 7, 8, 4, 76, 5, 3, 77, 93, 1, 68, 67, 57, 73, 58, 62, 79, 36, 2, 52, 114, 35, 64, 88, 72, 125, 115, 89, 37, 126, 127, 128, 130, 129, 60, 131, 132, 92, 97, 94, 145, 133, 96, 59, 134, 95, 80, 78, 101, 143, 99, 144, 81, 140, 151, 38, 86, 98, 41, 154, 123, 61, 66, 75, 121, 46, 40, 65, 150, 118, 146, 82, 153, 90, 147, 49, 102, 117, 135, 91, 83, 139, 42, 17, 152, 141, 116, 100, 137, 18, 44, 55, 142, 122, 53, 149, 119, 103, 19, 148, 27, 43, 84, 74, 25, 39, 136, 56, 124, 51, 22, 47, 33, 32, 120, 45, 54, 50, 113, 34, 48, 20, 105, 31, 23, 107, 30, 87, 29, 104, 24, 138, 110, 108, 28, 26, 85, 109, 21, 111, 106, 63, 112

L'ordre d'importance des attributs étant obtenu (voir tableau V.20), l'introduction séquentielle des attributs a été réalisée par paquets. La taille de ces derniers a été choisie telle que leur nombre soit constant, dans notre cas 10.

En raison de la complexité importante de l'algorithme SVM-RFE, celui-ci n'a pu être testé sur la base de données avec 154 attributs. En outre, cette approche n'as pas produit de solutions (dans un délai acceptable). Notons qu'avec l'algorithme SVM-RFE, l'attribut possédant le score le plus faible est éliminé. Le processus est arrêté lorsque le nombre d'attributs restant atteint d=154. Pour accélérer la procédure, il est possible d'éliminer plus d'un attribut à la fois pour réduire la complexité de l'algorithme, qui est assez élevée $O(l^2 \times d)$ avec l= 26998 instances. Nous avons choisit de supprimer 5 attributs à chaque récursion correspondants aux poids les plus faibles.

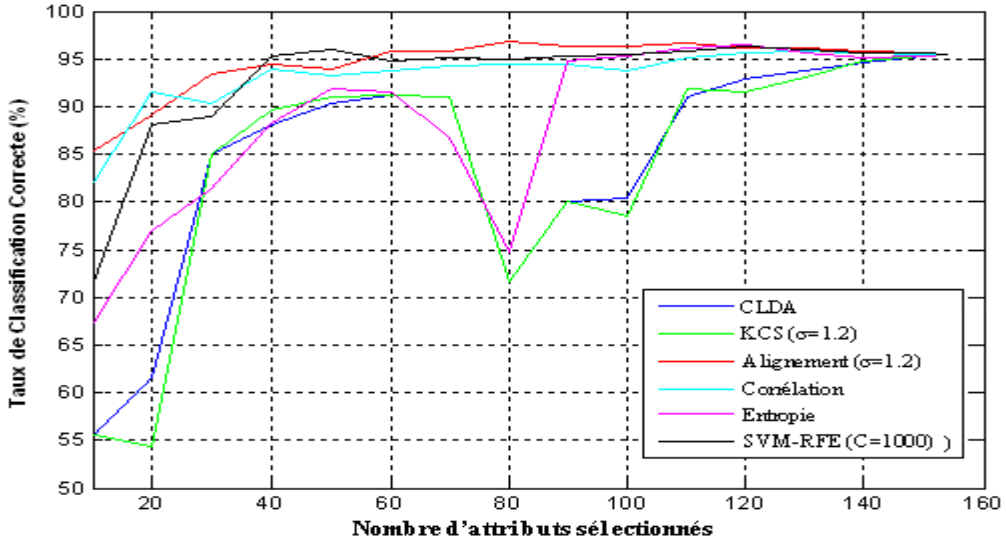


FIG V.6. Taux de classification correcte avec le nombre d'attributs varie de 10 à 154.

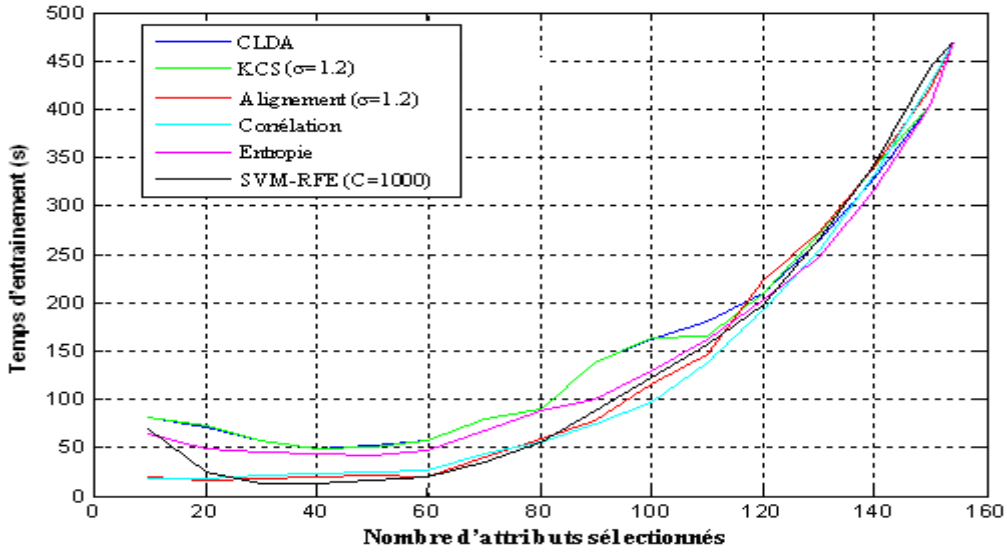


FIG V.7. Temps d'entrainement en fonction du nombre d'attributs sélectionnés (complexité exponentielle).

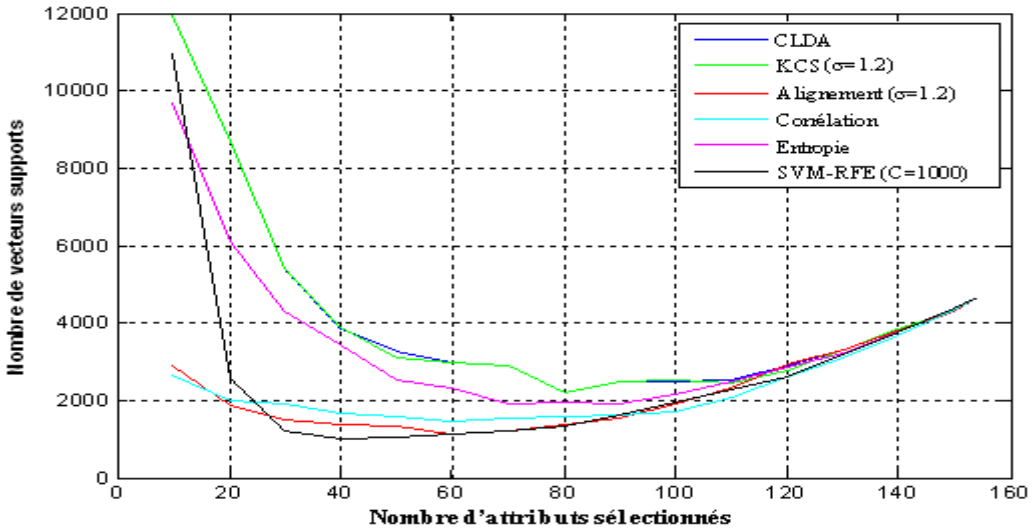


FIG V.8. Nombre de vecteurs supports en fonction du nombre d'attributs sélectionnés.

Eu égard aux bonnes performances de classification **(TCC=95.37%, VS=4647, T=471.14s)** avec la nouvelle parametrisation pour un ensemble de 154 attributs, nous nous sommes attachés à l'améliorer.

A partir des figures V.6, .7, .8, nous remarquons que le nombre de caractéristiques peut être réduit avec une amélioration des performances de classification, ce qui est significatif de leur capacité à effectuer un apprentissage moins couteux obtenu avec les algorithmes de sélection du tableau suivants :

TAB V.21

Les algorithmes présentant des améliorations de performances par rapport à l'ensemble complet d'attributs.

	Nb d'attributs	TCC(%)	VS	T(s)
Alignement	A60	95.95	1118	18.38
	A70	95.93	1219	39.7
	A80	96.96	1382	59.41
	A90	96.26	1521	76.95
	A100	96.27	1900	115.99
	A110	96.56	2308	145.53
	A120	96.17	2931	223.3
	A130	96.1	3313	272.89
	A140	95.86	3776	339.14
	A150	95.55	4280	419.49
Corrélation	A120	95.76	2610	192.45
	A130	95.91	3098	253.84
	A140	95.68	3679	332.08
	A150	95.41	4338	426.24
Entropie	A110	96.22	2486	162.66
	A120	96.46	2841	202.75
	A130	95.61	3201	247.16
SVM-RFE	A50	95.96	1064	16.05
	A100	95.47	1946	121.53
	A110	95.94	2266	156.88
	A120	96.27	2605	198.45
	A130	96.03	3173	266.13
	A140	95.79	3737	342.4
	A150	95.52	4349	442.62
Nb total d'attributs	**A154**	**95.37**	**4647**	**471.14**

De ce fait l'Alignement présente la plus forte augmentation des performances : en réduisant le nombre d'attributs sélectionnés à 80, le taux de reconnaissance s'élève à 1.59%.

Les algorithmes CLDA et KCS avec les performances maximales A150**(TCC=95.24%, VS=4290, T=402.88s)** n'ont pas permis d'améliorer le TCC par rapport à celui obtenu avec de l'ensemble complet d'attributs. Ces deux algorithmes ont les plus faibles performances avec un nombre de vecteurs de support élevé et donc nécessite des temps d'entrainements couteux par

rapport aux autres algorithmes en fonction des sous ensembles d'attributs retenus. Notons aussi qu'ils produisent des taux de reconnaissances comparables entre eux, même nombre de vecteurs de supports et les même temps d'entrainements pour les différents sous ensembles d'attributs sélectionnés, voir figures V.6, .7, .8.

L'algorithme d'entropie quant à lui a donné des performances en général meilleures que les deux algorithmes précédents mais ses performances reste inferieures à celles des autres algorithmes avec une complexité de vecteurs supports relativement grande.

Maintenant si nous permettons une diminution du TCC de seulement <2%, cela traduit le fait que le choix A30(Alignement), A40(Corrélation), A40(SVM-RFE) est un choix plus convenable pour notre schéma de classification (une réduction significative du nombre des attributs d'origines). Globalement l'alignement présente les plus grands TCC pour les différents groupes des attributs sélectionnés (voir figure V.6) avec une complexité d'entrainement assez faible (voir figures V.7, .8).

TAB V.22

Complexité des ASA en termes de temps.

ASA	Temps
CLDA	9.94 min
KCS	1.5 h
Alignement	1.63 h
Corrélation	1.03 sec
Entropie	14.9 h
SVM-RFE	29.98 h

En examinant les différents algorithmes et en admettant une certaine erreur de classification qui n'était pas trop importante inferieure à 2%, on peut réaliser que l'alignement réalise de très grande performances au détriment d'une complexité de sélection 1.63h, beaucoup plus grande que le critère de corrélation que lui présente des performances légèrement inferieures à l'alignement mais aussi bonnes avec une importante réduction de la complexité 1.03sec ! ceux qui laisse à l'utilisateur le choix de faire un compromis selon l'importance que l'on accorde à la

précision de classification par rapport au fait de minimiser le temps CPU de l'algorithme de sélection.

SVM-RFE réalise un bon compromis performances/complexité d'entrainement aux dépens d'un temps calculatoire significatif qui est de 29.98h. Cet algorithme est une méthode performante dans un schéma de classification par SVM, car il ya un lien direct entre la méthode de sélection et le classificateur utilisé, donc c'est une méthode élaborée mais qui pèche par une complexité très élevée, rendant très difficile son application à des bases d'apprentissages de dimensions typiques de l'indexation audio.

V-4-4- Discussion :

Certains algorithmes sont moins sensibles que d'autres à l'effet de la réduction du nombre d'attributs sélectionnés, en particulier les méthodes de l'alignement de noyaux, Corrélation et SVM-RFE, ce qui indique qu'ils réalisent un classement plus fiable des attributs les plus utiles en positionnant les plus efficaces aux premiers rangs.

Ainsi avec les trois algorithmes ASA et en utilisant une configuration binaire nous nous sommes parvenu à atteindre une dimension réduite inferieure à la moitié de l'ensemble total d'attributs, avec des performances proches de celles obtenues avec un ensemble particulièrement exhaustif de caractéristiques.

Le **critère de corrélation** est considéré comme le plus efficace parmi les algorithmes de sélection d'attributs utilisés. Il réalise un classement fiable des attributs les plus utiles, en positionnant les plus efficaces aux premiers rangs avec une complexité de calcul négligeable. Cela parait très avantageux du point de vue à rendre plus facile l'étape d'apprentissage et de réduire de la complexité de la classification ainsi rendant l'entrainement et le test beaucoup plus rapides, cela est de première importance pour ce type d'applications (classification) avec une contrainte de temps réel.

V-5- Récapitulatif des résultats :

Globalement, nous sommes arrivés à diminuer les charges de calcul dû à l'apprentissage SVM en utilisant le critère d'alignement de noyaux pour trouver le paramètre noyau gaussien optimal σ_{opt} et le critère de corrélation pour la sélection d'attributs. Le seul point contraignant est

la recherche du paramètre *C* qui est basée sur la validation croisée utilisant une ligne de recherche (1×7) qui nécessite l'apprentissage de la statistique de décision. Une valeur de *C*, notée C_{dat}, fixée de façon adaptative à partir des données d'apprentissage a été envisagée : elle est obtenue comme l'inverse de la longueur moyenne des *m* données d'apprentissage transformés $\phi(x_i)$ [60], en prenant

$$C_{dat} = \frac{1}{\frac{1}{m}\sum_{i=1}^{m} k(x_i, x_i)} \tag{V.2}$$

Pour un noyau gaussien $C_{dat} = 1$, donc la valeur de C_{dat} dépend du noyau utilisé.

Nous avons mené les dernières expérimentations sur le corpus audio utilisé précédemment pour la sélection d'attribut avec *d*=154. Les meilleurs hyper-paramètres ont été obtenus avec σ_{opt} = 1.2 et C_{opt} = C_{dat} = 1 avec T(σ_{opt}, C_{opt})= 16.7h. Nous obtiendrons TCC =95.24% avec T(training)=7.3minutes et T(test)=5.76minutes. Avec une sélection d'attribut *d* = 40, nous avons obtenus avec σ_{opt} = 0.8 et C_{opt} = 1, des performances inferieures TCC = 92.58% avec une rapidité de mise en œuvre T(σ_{opt}, C_{opt})=1.26h et T(training)= 27.74sec puis de décision T(test)= 38.16sec qui traduit par conséquent une faible charge de calcul.

CONCLUSIONS ET PERSPECTIVES

L'objectif assigné à ce travail développé dans le cadre d'un diplôme de magister en électronique, option Traitement de Signal et des Images était le développement de méthodes à noyaux pour la sélection de descripteurs pour des problèmes de classification de signaux audio.

L'analyse bibliographique nous a permis de relever que cette tâche fait apparaître deux volets: le premier concerne les méthodes de classifications et le deuxième à trait aux descripteurs utilisés et à la problématique de leur sélection pour des fins de réduction de la dimensionnalité des données. Étant donné que notre choix s'est porté sur les Méthodes à Vecteurs de Support (SVM) comme méthode de classification, notre attention s'est focalisée alors sur les difficultés soulevées par le choix souvent empirique des hyperparamètres inhérents à cette méthode. Les deux paramètres concernés sont les paramètres du noyau gaussien et de régularisation (σ, C). Leur choix optimal conditionne les performances des méthodes SVM. Dans la littérature cette tâche porte le nom de sélection de modèle SVM.

Nous avons donc apporté un certain nombre d'éléments de réponse concernant certaines questions en vue d'une tache de classification audio. L'une est liée à la sélection de modèle automatique des SVM au sein de systèmes de reconnaissance de formes et l'autre à la sélection de descripteurs. Notons que nous parlons bien d'éléments de réponse et non de réponses, car comme nous le rappellent [7] en s'appuyant sur le "*No Free Lunch Theorem*", si une approche est supérieure à une autre pour un problème de classification donné, rien ne permet de prédire qu'elle le sera aussi pour un problème différent. Ainsi, le choix de l'approche à utiliser dépendra du nombre de classes, du nombre de données d'apprentissage, du type de distribution, mais aussi des contraintes liées à l'application visée.

Pour conclure ce manuscrit, nous proposons donc de faire un bilan des contributions qui ressortent de nos travaux de recherche et des perspectives futures qui en découlent.

Contributions :

Partant de la constatation que la revue de la littérature utilise la validation croisée comme le moyen typique pour le réglage des hyperparamètres SVM, nonobstant son inconvénient majeur

lié aux charges de calculs importantes et donc d'un temps d'exécution conséquent. Nous avons présenté une nouvelle méthodologie de sélection des hyper-paramètres. L'approche permet d'optimiser les paramètres de noyau σ dans un délai acceptable et de réduire efficacement la complexité de classifieur en minimisant le nombre de vecteurs de support. Ceci s'accompagne d'une augmentation de la précision de généralisation. La méthodologie proposée est basée sur un nouveau critère de sélection de modèle qui est l'alignement de noyaux. Ce critère est ensuite maximisé pour trouver paramètre de noyau gaussien adapté à la densité des données.

La valeur du paramètre de compromis C est aussi un hyper-paramètre qui régit la performance du SVM. Ce paramètre sert à fixer le compromis entre la minimisation de l'erreur d'apprentissage et la maximisation de la marge. En pratique, le comportement du SVM est sensible à la valeur de C uniquement si les données d'apprentissage ne sont pas séparables. La régularisation du paramètre C ne peut pas être optimisée en utilisant le critère d'alignement qui est indépendant de l'algorithme d'apprentissage appliqué dans l'espace augmenté (*feature space*). Nous nous sommes basés un nouveau critère qui fixe C à 1 pour les noyaux gaussiens. Étant donnée la grande variabilité des données audio, de bons résultats ont été obtenus, comme prévu, en gardant un maximum de marge, c'est-à-dire les valeurs de C les plus petites. Le choix $C_{opt} = C_{dat}$ fournit des résultats acceptables.

Nous nous sommes focalisés par la suite sur le deuxième volet de notre travail: la sélection d'attributs pour la classification binaire SVM. Le fondement théorique de ce dernier, abordé dans le premier chapitre de ce mémoire, nous apprend que l'augmentation du nombre d'attributs ne devrait pas nuire à la qualité de la discrimination qu'elles réalisent. En revanche les temps d'entrainements et de test sont directement affectés, en plus la qualité des données pose néanmoins des problèmes majeurs dans les applications. Nous préconisons l'usage d'une procédure de sélection, non seulement pour réduire l'espace de stockage, mais aussi pour sélectionner le sous ensemble d'attributs qui a des performances équivalentes, même dans certains cas qui donne des améliorations de la précision et la complexité d'entrainement. Aussi la stratégie de sélection doit être peu couteuse en termes de temps de calcul.

Nous avons ainsi développé différents algorithmes de sélection supervisés de descripteurs dits à noyaux que nous avons confrontés aux méthodes supervisées de l'état de l'art, sauf exception l'Entropie qui nécessite pas la connaissance des classes. Nous avons alors vérifié

expérimentalement que les méthodes d'alignement de noyaux, de corrélation et le SVM-RFE permettent chacun avec des performances diverses, d'accélérer notablement la phase d'apprentissage tenant en compte la qualité des performances de classification. L'alignement a donné de bons résultats de sélection en dépit d'un temps d'exécution significatif. Le SVM-RFE semble avoir des difficultés plus importantes non seulement pour les ensembles de données avec peu de données mais aussi il est très gourmand en temps de calcul pour la sélection. Le critère de corrélation qui est un critère basé sur des relations linéaires entre les attributs et l'attribut des étiquettes a permis de donner une sélection efficace sur les différentes bases de données, avec un temps de sélection négligeable. Nous ajoutant que ce critère ne nécessite le réglage d'aucun paramètre contrairement aux autres algorithmes.

Les problèmes de temps de calcul étant améliorés avec le critère d'alignement pour la sélection de modèle et le critère de corrélation pour las sélection d'attributs, l'apprentissage peut être dorénavant effectué en temps plus ou moins raisonnable. Toute fois, nous sommes loin du temps réel.

Perspectives :

Les approches que nous avons proposées sont encore perfectibles et plusieurs perspectives de travaux de recherches sont envisageables. Une perspective très intéressante est l'amélioration du temps de calcul du critère d'alignement en développant une implémentation en langage C afin d'optimiser l'hyperparamètre du noyau, ceci permettrait d'accélérer grandement le calibrage du classifieur et de l'intégrer dans l'apprentissage permettant ainsi une sélection de modèle en ligne.

En ce qui concerne la sélection d'attributs, il est nécessaire de disposer d'un critère qui permette de déterminer automatiquement le nombre d'attributs à sélectionner. Cela serait particulièrement profitable à l'approche de sélection binaire et multiclasses, cette perspective fait déjà l'objet d'un travail en cours de réalisation.

Signalons enfin, les méthodes que nous avons proposées doivent pouvoir être utilisées pour d'autres tâches que la classification parole/musique. Il serait intéressant de tester leurs performances dans des contextes d'applications différents.

ANNEXES :

A.1 : Développement de la trace de la matrice de dispersion *inter-classe* dans l'espace à noyau

$$D^2 = \left|\mu_+ - \mu_-\right|^2$$

$$= \left(\frac{1}{m_+} \sum_i^+ \phi(x_i) - \frac{1}{m_-} \sum_i^- \phi(x_i) \right)^2$$

$$= \left(\frac{1}{{m_+}^2} (\sum_i^+ \phi(x_i))^2 + \frac{1}{{m_-}^2} (\sum_{i,j}^- \phi(x_i))^2 - \frac{2}{m_+ m_-} \sum_i^+ \phi(x_i) \sum_i^- \phi(x_i) \right)$$

$$= \left(\frac{1}{{m_+}^2} \sum_{i,j}^+ \phi(x_i)\phi(x_j) + \frac{1}{{m_-}^2} \sum_{i,j}^- \phi(x_i)\phi(x_j) - \frac{2}{m_+ m_-} \sum_{i,j}^{y_i \neq y_j} \phi(x_i)\phi(x_j) \right)$$

$$D^2 = \frac{1}{{m_+}^2} \sum_{i,j}^+ k_{ij} + \frac{1}{{m_-}^2} \sum_{i,j}^- k_{ij} - \frac{2}{m_+ m_-} \sum_{i,j}^{y_i \neq y_j} k_{ij}$$

A.2 : Développement de la trace de la matrice de dispersion *intra-classe* à noyau

$$V^2 = \pi_+ E_+ (\left|\phi(x) - \mu_+\right|^2) + \pi_- E_- (\left|\phi(x) - \mu_-\right|^2)$$

$$V^2 = \frac{m_+}{m} E_+ (\phi(x)^2 + {\mu_+}^2 - 2\mu_+ \phi(x)) + \frac{m_-}{m} E_- (\phi(x)^2 + {\mu_-}^2 - 2\mu_- \phi(x))$$

$$V^2 = \frac{m_+}{m} E_+ (\phi(x)^2) + \frac{m_+}{m} {\mu_+}^2 - 2\frac{m_+}{m} \mu_+ E_+ (\phi(x)) + \frac{m_-}{m} E_- (\phi(x)^2) + \frac{m_-}{m} {\mu_-}^2 - 2\frac{m_-}{m} \mu_- E_- (\phi(x))$$

$$V^2 = \frac{m_+}{m} \frac{1}{m_+} \sum_i^+ \phi(x_i)\phi(x_i) + \frac{m_+}{m} \frac{1}{{m_+}^2} \sum_{i,j}^+ \phi(x_i)\phi(x_j) - 2\frac{m_+}{m} \frac{1}{{m_+}^2} \sum_{i,j}^+ \phi(x_i)\phi(x_j) + \frac{m_-}{m} \frac{1}{m_-} \sum_i^- \phi(x_i)\phi(x_i) +$$

$$\frac{m_-}{m} \frac{1}{{m_-}^2} \sum_{i,j}^- \phi(x_i)\phi(x_j) - 2\frac{m_-}{m} \frac{1}{{m_-}^2} \sum_{i,j}^- \phi(x_i)\phi(x_j)$$

$$V^2 = \frac{1}{m} \sum_i^+ k_{ii} + \frac{1}{m_+ m} \sum_{i,j}^+ k_{ij} - \frac{2}{m_+ m} \sum_{i,j}^+ k_{ij} + \frac{1}{m} \sum_i^- k_{ii} + \frac{1}{m_- m} \sum_{i,j}^- k_{ij} - \frac{2}{m_- m} \sum_{i,j}^- k_{ij}$$

$$V^2 = \frac{1}{m} \sum_{i,i}^+ k_{ii} + \frac{1}{m} \sum_{i,i}^- k_{ii} - \frac{1}{m_+ m} \sum_{i,j}^+ k_{ij} - \frac{1}{m_- m} \sum_{i,j}^- k_{ij}$$

$$V^2 = \frac{1}{m}\left(\sum_{i}^{m} k_{ii} - \frac{1}{m_+} \sum_{i,j}^{+} k_{ij} - \frac{1}{m_-} \sum_{i,j}^{-} k_{ij} \right)$$

B : Sélection de descripteurs sur la base de données *UCI*

Nous avons testé les algorithmes sur deux jeux de données réels issus de la base de données mise à la disposition de la communauté d'apprentissage artificiel par l'université de Californie à Irvine (UCI), et très répandues dans la littérature portant sur l'apprentissage. Les deux jeu soulèvent un problème de discrimination binaire, et disposent de peu d'observations. Avant de pouvoir utiliser certains jeux de données, il est souvent nécessaire d'effectuer une normalisation de celui-ci.

L'une des techniques classiques pour éliminer l'effet d'échelle est d'effectuer une normalisation linéaire entre [-1,1] des jeux de données, ces derniers sont normalisés disponibles dans [25]. Les deux jeux de données sont *Breast-Cancer Wisconsin,* et *Heart*, utilisés dans le diagnostic médical. Voici une brève description des données que nous avons utilisées, leur origine, les prétraitements qu'elles ont subis par les fournisseurs et le problème qu'elles soulèvent. (Pour une description plus détaillée des traitements statistiques préalables appliqués sur les données voir [26]).

- ***Wisconsin Breast-Cancer*** **(WBC) :** Les données de cette base de données ont été recueillies à partir d'images numérisées d'un prélèvement par biopsie d'une masse éventuellement cancéreuse. Le problème considéré traite de la détection des cellules cancéreuses pour le diagnostic du cancer du sein. Les attributs extraits, au nombre de 10, proviennent d'une série d'observations annotées de 0 à 10 par un expert concernant la forme, la taille, l'opacité de la cellule. Le nombre total des instances est de 699 dont 16 avec des valeurs manquantes ont été supprimées selon la technique "*Listwise data deletion*". Ce problème à deux classes selon qu'il s'agit de tumeurs (maligne/bénigne) est plutôt considéré comme facile : les classes sont linéairement séparable et l'état de l'art fait mention d'une précision supérieure à 97% en classement.
- ***Heart*** **:** Cet ensemble de données est une base de données d'une maladie cardiaque, elle contient 76 attributs bruts, mais toutes les expériences publiées se réfèrent à l'utilisation d'un sous-ensemble de 14 d'entre eux, on note par exemple l'âge(1), sexe(2), taux de

cholestérol(12), la pression artérielle au repos(10) ...etc. Son but est de renvoyer la présence ou l'absence d'une maladie cardiaque chez le patient, donc c'est un problème à deux classes.

L'échantillon test n'étant pas disponible, nous aurons recours à une division aléatoire (App- Test) de l'échantillon de données comme le montre le tableau suivant :

TAB B.1

Description des données réelles *UCI* à deux classes : nombre de variables, nombre d'instances, tailles des échantillons d'apprentissage et de test.

Données	# de variables	# d'observations	apprentissage	test
Heart	13	270	150	120
Breast-Cancer Wisconsin	10	683	375	308

Notre objectif ici n'est pas d'interpréter les résultats obtenus par les différentes approches de sélection d'attributs, mais juste pour en avoir une idée de leur capacité à repérer les sous-ensembles optimaux d'attributs et repérer les algorithmes les plus efficaces en terme d'une sélection du plut petit sous ensemble avec l'obtention des meilleures performances de classification.

– ***Wisconsin Breast-Cancer*** **:**

TAB B.2

La sélection de modèle avec la première méthode pour la base de données *WBC*.

σ	.1	.2	.3	.4	.5	.6	.7	.8	.9	**1**
A	0.1096	0.2338	0.3305	0.3936	0.4348	0.4620	0.4800	0.4920	0.4994	**0.5027**
$(\sigma_{opt}=1, C)$	**.1**	5	50	500	1000	5000	10000			
$TCC_{vc}(\%)$	**95.7333**	94.4000	93.0667	93.0667	93.0667	93.0667	93.0667			

σ	1.1	1.2	1.3	1.4	1.5
A	0.5021	0.4977	0.4897	0.4785	0.4645

Les meilleurs hyper-paramètres sont obtenus avec $\sigma_{opt} = 1$ pour $A_{max} = 0.5$ et $C_{opt} = 0.1$ pour TCC_{vc}=95.73%.

TAB B.3

Contribution des attributs par les différentes méthodes de sélection (d=10).

ASA	Ordre d'importance des attributs										Temps CPU (sec)
CLDA	1	8	10	4	6	3	2	5	7	9	0.15
KCS (σ=1)	1	8	4	10	3	7	6	2	5	9	0.07
Alignement (σ=1)	7	4	3	2	9	5	6	8	10	1	0.07
Corrélation	7	4	3	2	9	5	8	6	10	1	0.001
Entropie	8	10	6	5	1	9	7	4	3	2	0.04
SVM-RFE (C=1000)	5	2	7	10	1	8	4	6	9	3	1.24

Le critère d'arrêt utilisé ici est le nombre d'attributs total atteint et sachant que le nombre de descripteurs utilisés est directement lié à l'erreur finale, nous allons donc choisir pour la validation le plus petit sous-ensemble obtenu avec les meilleures performances en termes de classification.

On sélectionne manuellement les bons attributs (avec une méthode empirique très basique) en essayant ceci pour plusieurs combinaisons de caractéristiques **A**{2,**3**,4,5,6,7,8,9,10}, les attributs sont introduits séquentiellement, un par un, dans le modèle selon l'ordre d'importance décroissant (c'est-à-dire dans le cas de l'algorithme KCS par exemple, **A3** selon l'ordre d'importance veut dire la combinaison des attributs **{1,8,4}**), et par la suite, on évalue les performances en prédiction en partageant l'échantillon dont on dispose en deux parties qui ont comme attributs **{1,8,4}**, une partie dédiée à l'apprentissage et une partie dédiée au test que l'on utilise pour estimer la précision de classification, voir tableau B.4.

Nous constatons d'après le tableau B.4 que le nombre de variable minimal qui donne les performances maximales TCC=99.35% est atteint avec le critère d'alignement et le critère de corrélation pour A4= {7, 4, 3,2}.

TAB B.4

Les taux de bon classement obtenus en utilisant la combinaison des attributs sélectionnés par les différentes méthodes de sélection (d=10). **En gras** : meilleurs TCC plus grand ou égal à celui calculé avec les attributs d'origines pour chaque ASA.

ASA \ Nb d'attributs Sélectionnés(A)	Taux de Classification Correcte (%)								
	2	3	4	5	6	7	8	9	Tous les attributs
CLDA	96.7532	96.7532	**97.4026**	**97.4026**	**97.4026**	97.7273	97.0779	**98.0519**	**97.4026**
KCS	96.7532	**97.4026**	**97.4026**	**97.4026**	**98.3766**	97.7273	**98.3766**	**98.0519**	
Alignement	**98.0519**	**98.7013**	**99.3506**	**98.7013**	**98.0519**	**98.0519**	**98.0519**	97.7273	
Corrélation	**98.0519**	**98.7013**	**99.3506**	**98.7013**	**98.0519**	**98.0519**	**98.0519**	97.7273	
Entropie	96.7532	**97.4026**	96.1039	95.7792	95.7792	97.0779	97.0779	97.0779	
SVM-RFE	94.4805	96.4286	96.1039	95.7792	97.7273	**98.3766**	**98.0519**	**97.4026**	

***– Heart* :**

TAB B.5

La sélection de modèle avec la première méthode pour la base de données *Heart*.

σ	.1	.2	.3	.4	.5	.6	.7	.8	.9	1
A	0.0817	0.0820	0.0837	0.0889	0.0978	0.1090	0.1215	0.1353	0.1500	0.1647

$(\sigma_{opt}$=**1.4**,$C)$	.1	5	**50**	500	1000	5000	10000
TCC_{vc}(%)	76.6667	76.0000	77.3333	77.3333	77.3333	77.3333	77.3333

σ	1.1	1.2	1.3	**1.4**	1.5
A	0.1776	0.1874	0.1932	**0.1947**	0.1926

La précision de classification dans l'étage de validation croisée est la même pour $C \in \{50\ 500\ 1000\ 5000\ 10000\}$ à savoir 77.33%, ainsi nous reporterons l'une des valeurs de C, nous choisissons C=50. Donc le meilleur couple obtenu est (σ_{opt}=1.4, C_{opt}=50) qui donne un TCC=75.83%.

TAB B.6

Contribution des attributs par les différentes méthodes de sélection (*d*=13).

ASA	Ordre d'importance des attributs													Temps CPU (sec)
CLDA	5	10	4	8	1	12	3	11	6	13	2	9	7	0.0039
KCS (σ=1.4)	5	10	4	8	1	12	3	11	6	13	2	9	7	0.0152
Alignement (σ=1.4)	13	12	9	2	3	7	11	8	10	4	1	5	6	0.01
Corrélation	13	12	3	8	10	2	11	9	7	4	1	5	6	0.001
Entropie	6	11	3	2	12	1	13	4	5	9	10	8	7	0.0875
SVM-RFE (C=1000)	12	8	4	3	1	5	2	13	6	7	10	11	9	3.96

A la lumière des résultats exposés dans le tableau B.7, nous remarquons que les algorithmes de sélection, Alignement et Corrélation respectivement avec le sous ensemble de variables V3= {9,12,13} et V3= {3,12,13} ont permis des améliorations importantes dans les performances de classification avec plus de 9% du taux de classification correcte. Les autres algorithmes ont permis aussi d'améliorer les TCC mais avec un nombre d'attribut plus élevé. Nous allons voir dans les expérimentations qui suivent (§V-4-3-2) qu'en augmentant la taille de la base d'apprentissage, l'algorithme SVM-RFE donnerai de bon résultats.

TAB B.7

Les taux de bon classement obtenus en utilisant la combinaison des attributs sélectionnés par les différentes méthodes de sélection (*d*=13).

ASA / Nb d'attributs Sélectionnés(A)	Taux de Classification Correcte (%)										
	2	3	4	5	6	7	8	9	10	11	12
CLDA	67.5	69.1667	74.1667	70.8333	73.3333	74.1667	75.8333	76.6667	78.3333	78.3333	78.3333
KCS	67.5	69.1667	74.1667	70.8333	73.3333	74.1667	75.8333	76.6667	78.3333	78.3333	78.3333
Alignement	79.1667	85	82.5	82.5	72.5	77.5	80.8333	82.5	82.5	77.5	76.6667
Corrélation	79.1667	85	84.1667	84.1667	84.1667	81.6667	80.8333	82.5	82.5	77.5	76.6667
Entropie	65.8333	70.8333	74.1667	74.1667	75.8333	77.5	79.1667	80	75.8333	80	78.3333
SVM-RFE	73.3333	73.3333	78.3333	73.3333	73.3333	78.3333	79.1667	79.1667	76.6667	80	75

	Tous les attributs
CLDA	75.8333
KCS	
Alignement	
Corrélation	
Entropie	
SVM-RFE	

BIBLIOGRAPHIE :

[1] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American Mathematical Society*, vol. 68, pp. 337-404, 1950.

[2] B. Boser, I. Guyon, et V. Vapnik, "An training algorithm for optimal margin classifiers," in *Proc. Fifth Annual Workshop on Computational Learning Theory*, 1992, pp. 144-152.

[3] Cortes, C.; and Vapnik, V. 1995. Support Vector Networks. *Machine Learning* 20, 273-297.

[4] J. Vert, K. Tsuda, et B. Schölkopf, "A primer on kernel methods," in *Kernel Methods in Computational Biology*, B. Schölkopf, K. Tsuda, and J. Vert, Eds. Cambridge, MA: MIT Press, 2004, pp. 35-70.

[5] Schölkopf, B. and A. J. Smola*:* Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. 187-222; 405-468, MIT Press, Cambridge, MA, USA (12- 2001).

[6] L. WANG ET K. LUK CHAN. Learning kernel parameters by using class separability measure. *Neural Information Processing Systems (NIPS)*, Canada, 2002.

[7] R.O. Duda, P. E. Hart and D.G. Stork. *Pattern Classification (2nd edition) John Wiley and Sons*, 2000.

[8] K. Duan, S.S. Keerthi, and A.N. Poo. Evaluation of simple performance measures for tuning SVM hyperparameter, *Neurocomputing*, vol.51, pp. 41-59, 2003.

[9] O. Chapelle, V.N. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131-159, (2002).

[10] Cheng Soon Ong, Alexander J. Smola, and Robert C. Williamson. Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, 6:1043–1071, 2005.

[11] Christian Igel, Tobias Glasmachers, Britta Mersch, Nico Pfeifer, and Peter Meinicke. Gradient-based Optimization of Kernel-Target Alignment for Sequence Kernels Applied to Bacterial Gene Start Detection. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 4(2), pp. 216-226, 2007.

[12] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola, "On Kernel-Target Alignment," *Advances in Neural Information Processing Systems* , vol. 14, pp. 367-373, 2002.

[13] J.-B. Pothin and C. Richard, "Kernel machines : une nouvelle méthode pour l'optimisation de l'alignement des noyaux et l'amélioration des performances," in *Proc. XX colloque GRETSI*, Louvain-la-Neuve, Belgium, 2005, pp. 1133–1136.

[14] Loo-Nin Teow, Kia-Fock Loe: Selection of Support Vector Kernel Parameters for Improved Generalization. ICML 2000: pp. 967-974.

[15] Greene J.R. *Feature subset selection using Thornton's separability index and its applicability to a number of sparse proximity-based classifiers*. In proceedings of the Pattern Recognition Association of South Africa, 2001.

[16] Lei Wang, *Feature Selection with Kernel Class Separability*. IEEE Trans. Pattern Anal. Mach. Intell. 30(9): 1534-1546 (September 2008).

[17] Zifeng Cui, Baowen Xu, Weifeng Zhang, Jiang Dawei, Junling Xu: CLDA: Feature Selection for Text Categorization Based on Constrained LDA. ICSC 2007: 702-712.

[18] GUYON I., J. WESTON, S. BARNHILL, and V. VAPNIK (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3) :389-422.

[19] Ghattas, Badih; Ben Ishak, Anis, Sélection de variables pour la classification binaire en grande dimension : comparaisons et application aux données de biopuces. *Journal de la société française de statistiques,* 149 no. 3 (2008), p. 43-66.

[20] Slim Essid. *Classification automatique des signaux audio-fréquences : reconnaissance des instruments de musique*. PhD thesis, Telecom Paristech, 2005.

[21] P. Mitra, C.A. Murthy, and S.K. Pal. Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):301–312, 2002.

[22] Mark A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 359-366). Morgan Kaufmann, San Francisco, CA, 2000.

[23] Mark A. Hall and Lloyd A. Smith. Feature Selection for Machine Learning: Comparing a Correlation-Based Filter Approach to the Wrapper. FLAIRS Conference. 1999.

[24] Hall, M. A. (1998). *Correlation-based feature selection for machine learning*. PhD thesis, Department of Computer Science, University of Waikato, Hamilton, New Zealand.

[25] C.C. Chang, C.J. Lin. Software disponible sur http ://www.csie.ntu.edu.tw/~ cjlin/libsvm/. *LIBSVM : a library for support vector machines*, 2001.

[26] http://archive.ics.uci.edu/ml/.

[27] Gretton, A., Herbrich, R., Smola, A., Bousquet, O., et Schölkopf, B. (2005). Kernel methods for measuring independance. *Journal of Machine Learning Research*, 6 :2075-2129.

[28] Schölkopf, B. et Smola, A. (2002). *Learning with kernels*. MIT Press.

[29] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, et K. R. Müller, "Fisher discriminant analysis with kernels.," in *Advances in neural networks for signal processing,* Y. H. Hu, J. Larsen, E. Wilson, and S. Douglas, Eds. San Mateo, CA: Morgan Kaufmann, 1999, pp. 41-48.

[30] Bach, F. et Jordan, M. (2002). Kernel independent component analysis. *Journal of Machine Learning Research*, 3 :1-48.

[31] Shawe- Taylor, J. et Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

[32] V. N. Vapnik, *The nature of statistical learning theory*. New York, NY: Springer, 1995.

[33] Mika, S., Schölkopf,, B., Smola, A., Müller, K., M., S., et G., R. (1999). *Advances in Neural Information Processing Systems*, volume 11, chapter Kernel PCA and de-noising in feature spaces, pages 536-542. MIT Press.

[34] Xu, J.-W., Pokharel, P., Jeong, K.-H, et C. Principe, J. (2006). An explicit construction of a reproducing gaussian kernel hilbert space. Dans *Proc. ICASSP*.

[35] CRISTIANNINI N., SHAWE-TAYLOR J. (2000). *An Introduction to Support Vector machine and Other Kernel-based Learning Methods*. Cambridge University Press.

[36] SCHÖLKOP B., BURGES C. J. C., SMOLA A.J. (1999). *Advances in Kernel Methods--Support Vector Learning*. MIT Press, Cambridge, MA.

[37] Emmanuel Didiot. *Segmentation parole/musique pour la transcription automatique de parole continue*, thèse de doctorat [S.I], Université Henri Poincaré -Nancy 1 : 2007 , pages 27-30.

[38] N. Mesgarani, M. Slaney & S.A Shamma. *Discrimination of Speech from Nonspeech based on multiscale spectro-temporal modulations*. IEEE trans. On Audio, Speech, and Language Processing, vol. 14, no. 3, pages 920-930, 2006.

[39] *A practical guide to Support Vector Classifcation*, Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin.

[40] S. Sathiya Keerthi and Chih-Jen Lin. *Asymptotic behaviors of support vector Machines with Gaussian kernel*. Neural Computation, 15(7):1667-1689, 2003.

[41] C. Gold & P. Sollich. *Model selection for support vector machine classification*. Neurocomputing, vol. 55, pages 221-249, 2003.

[42] M. MEILA. Data centering in feature space. *Proceedings of Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[43] Shibin Qiu and Terran Lane. "*Parallel Computation of RBF Kernels for Support Vector Classifiers*", SIAM International Conference on Data Mining, Newport Beach, USA, April 2005.

[44] Bartlett, P., & Shawe-Taylor, J. (1999). Generalization performance of support vector machines and other pattern classifiers. In B. Scholkopf, C.J.C. Burges &A.J. Smola (Eds.), *Advances in kernel methods – support vector learning*. Cambridge: MIT-Press.

[45] Cristianini, N., Campbell, C., & Shawe-Taylor, J. (1998). Dynamically adapting kernels in support vector machines. In M. S. Kearns, S. A. Solla, & D. A. Cohn (Eds.), *Advances in Neural Information Processing Systems, 11*, 204-210.

[46] Isabelle Guyon, André Elisseeff; An Introduction to Variable and Feature Selection, JMLR Special Issue on Variable and Feature Selection, 3(Mar):1157-1182, 2003.

[47] Younès Bennani. *Systèmes d'apprentissage connexionnistes : sélection de variables*, volume15(3-4) of *Revue d'Intelligence Artiticielle*. Hermes Science Publications, Paris, France, 2001.

[48] Guérif S. (2006). Réduction de Dimension en Apprentissage Numérique Non Supervisé, Thèse de doctorat, Université Paris 13, Villetaneuse, Décembre 2006.

[49] H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 1998.

[50] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2) :273-324, 1997.

[51] http://www.cs.ucl.ac.uk/staff/M.Sewell/publications.html.

[52] Daphne Koller and Mehran Sahami. Toward optimal feature selection. *In International Conference on Machine Learning*, pages 284-292, 1996.

[53] George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. *In International Conference on Machine Learning*, pages 121-129, 1994. Journal version in AIJ, available at http://citeseer.nj.nec.com/13663.html.

[54] Somol P., Pudil P., Novovicova J., Paclik P.: *Adaptive floating search methods in feature selection.* Pattern Recognition Letters, 20 (1999), 11/13, 1157-1163.

[55] Oliver Ritthoff and et al. A hybrid approach to feature selection and generation using an ea, 2002.

[56] Jihoon Yang and Vasant Honavar. Feature Subset Selection Using a Genetic Algorithm. *IEEE Intelligent Systems*, 13 :44-49, 1998.

[57] B. Logan. *Mel Frequency Cepstral Coefficients for Music Modeling*. In International Symposium on Music Information Retrieval (ISMIR), 2000.

[58] http://logiciel.codes-sources.com/logiciels/Audacity-127.aspx#

[59] http://htk.eng.cam.ac.uk/.

[60] T. Joachims. Svm light support vector machine. http://svmlight.joachims.org/ .