Approche de sélection d'attributs pour la classification basée sur l'algorithme RFE-SVM

Yahya Slimani, Mohamed Amir Essegir, Mouhamadou Lamine Samb, Fodé Camara, Samba Ndiaye

Special issue CARI'12

Approche de sélection d'attributs pour la classification basée sur l'algorithme RFE-SVM

Mouhamadou Lamine SAMB*, Fodé CAMARA*, Samba NDIAYE*
Yahya SLIMANI**, Mohamed Amir ESSEGHIR***

* Département Mathématiques-Informatique, Faculté des Sciences et Techniques, Université Cheikh Anta Diop de Dakar SENEGAL
{mouhamadou81.samb, fode.camara, samba.ndiaye}@ucad.edu.sn
** Département d'Informatique, Faculté des Sciences Université Tunis TUNISIE yahya.slimani@fst.rnu.tn
*** Laboratoire de Génie Informatique et d'Automatique de l'Artois France mohamedemir@yahoo.fr

RÉSUMÉ. La problématique de filtrage de données ou de sélection d'attributs pour la classification représente un axe de recherche très actif dans le domaine du datamining (fouille de données) et en optimisation. La nature combinatoire du problème impose le développement de techniques spécifiques (utilisation de filtres, d'enveloppes etc.) ou hybrides combinant plusieurs procédés d'optimisation. Dans ce contexte, l'approche basée sur l'algorithme *RFE-SVM (Recursive Feature Elimination with Support Vector Machine)* s'est distinguée par l'intégration du filtrage dans le processus d'apprentissage basé sur les machines à vecteurs de supports ou *SVM*. Cependant, l'utilisation de l'algorithme *RFE-SVM* a montré quelques limites comme le fait d'être complètement glouton, c'est-à-dire ne permettant pas les retours en arrière. Dans cet article, nous avons proposé une approche de sélection d'attributs pour pallier cette limite de l'algorithme *RFE-SVM*. Notre approche consiste à combiner l'algorithme *RFE-SVM* avec des opérateurs de recherche locale, issus des domaines de la recherche opérationnelle et de l'intelligence artificielle.

ABSTRACT. The feature selection for classification is a very active research field in data mining and optimization. Its combinatorial nature requires the development of specific techniques (such as filters, wrappers, genetic algorithms, and so on) or hybrid approaches combining several optimization methods. In this context, the support vector machine recursive feature elimination (SVM-RFE), is distinguished as one of the most effective methods. However, the RFE-SVM algorithm is a greedy method that only hopes to find the best possible combination for classification. To overcome this limitation, we propose an alternative approach with the aim to combine the RFE-SVM algorithm with local search operators based on operational research and artificial intelligence.

MOTS-CLÉS : Datamining, Classification, Apprentissage supervisé, Sélection d'attributs, Machines à Vecteurs de Supports (SVM), Recursive Feature Elimination (RFE), Recherche locale.

KEYWORDS: Data mining, Classification, Supervised classification, Feature selection, Support Vector Machines, Recursive Feature Elimination (RFE), Local search.

1. Introduction

La sélection d'attributs est un sujet de recherche très actif depuis une dizaine d'années dans les domaines de l'apprentissage artificiel, de la fouille de données, du traitement d'images, et de l'analyse de données en bioinformatique [1, 2, 3]. Elle consiste à choisir parmi un ensemble d'attributs de grande taille, un sous-ensemble d'attributs intéressants pour le problème étudié. Cette problématique peut concerner différentes tâches d'apprentissage ou de fouille de données, mais nous parlerons dans ce papier de la sélection d'attributs réalisée pour la classification supervisée. Dans ce contexte, l'objectif de la sélection est de trouver un sous-ensemble optimal d'attributs qui ait les propriétés suivantes : il doit être composé d'attributs pertinents et doit chercher à éviter les attributs redondants. De plus, cet ensemble doit permettre de satisfaire au mieux l'objectif fixé, à savoir la précision de l'apprentissage, la rapidité de l'apprentissage ou bien encore l'applicabilité du classifieur proposé [3, 6].

Dans ce papier, nous avons choisi d'étudier, l'algorithme *RFE-SVM* [1] qui est un algorithme de sélection basé sur l'élimination *backward*[1] et exploitant les *SVM* [9], de façon récursive pour sélectionner un sous-ensemble d'attributs optimal. Il a été utilisé en bioinformatique pour l'analyse du niveau d'expression de gènes et dans l'analyse de données de transcriptome [10]. Ceci a montré que *RFE-SVM* [1] sélectionne de bons sous-ensembles d'attributs. Cependant, son utilisation a montré quelques limites comme le fait d'être complètement glouton, c'est-à-dire n'intégrant pas de retours en arrière, et donc dans chaque recursion, l'attribut ou le sous ensemble d'attributs éliminé ne peut plus jamais revenir dans le sous-ensemble sélectionné, ce qui aura pour effet de biaiser la recherche et la diriger vers des minima locaux. On se propose donc de donner aux attributs éliminés dans les itérations antérieures la chance de revenir dans la recherche, ce qui pourrait, peut être, améliorer la qualité de la solution retournée par *RFE-SVM*.

Notre approche consiste à combiner l'algorithme *RFE-SVM* avec des opérateurs de recherche locale, issus des domaines de la recherche opérationnelle et de l'intelligence artificielle. Afin d'évaluer les apports de notre approche, nous avons réalisé une série d'expérimentations sur trois bases de données disponibles dans le site *UCI* Machine Learning Repository [5].

Les résultats de ces expérimentations sont prometteurs et confirment l'intérêt d'utiliser des recherches locales. En effet, même s'il y'a encore des erreurs de classification, nous avons pu montrer que la réinsertion d'attributs, éliminés lors du processus de classification contribuait à l'amélioration de la qualité de la solution

[1] L'ensemble total des attributs est considéré au départ de la procédure itérative, et à chaque itération on supprime un attribut. On l'appelle également approche descendante.

retournée par l'algorithme *RFE-SVM* et ainsi, nous avons pu produire des classifieurs plus performants.

Le reste du papier est structuré comme suit. La section 2, introduit les concepts et les notions utilisés dans ce papier. Nous décrivons dans cette section, des techniques de datamining en l'occurrence la classification supervisée avec l'algorithme SVM et la sélection d'attributs avec l'algorithme RFE-SVM. Nous terminons cette section 2 par une présentation de la problématique que nous essayons de traiter dans ce papier. Notre proposition est présentée dans la section 3 où nous commençons par présenter le principe de notre approche, puis terminons par une présentation détaillée de notre algorithme de sélection d'attributs. Les operateurs de recherche locale utilisés seront aussi présentés dans cette partie.

Les résultats obtenus lors des expérimentations sur des bases de données benchmarks sont détaillés dans la section 4. La cinquième et dernière partie de ce papier est consacrée à la conclusion dans laquelle nous faisons le bilan des apports de notre contribution et terminer par la présentation rapide de quelques perspectives ouvertes par notre travail.

2. Préliminaires

2.1. L'apprentissage supervisé : La classification

Le terme classification peut désigner deux classes de méthodes distinctes : la classification supervisée et la classification non-supervisée (automatic classification et clustering en anglais). Les méthodes non supervisées ont pour but de constituer des groupes d'exemples (ou des groupes d'instances) en fonction des données observées, sans connaissance a priori. En revanche, les méthodes supervisées utilisent la connaissance a priori sur l'appartenance d'un exemple à une classe pour construire un système de reconnaissance de ces classes. Dans ce papier nous nous intéressons à la classification à partir d'exemples : la classification supervisée.

L'objectif de la classification supervisée est d'apprendre à l'aide d'un ensemble d'entraînement (ensemble d'apprentissage) une procédure de classification qui permet de prédire l'appartenance d'un nouvel exemple à une classe.

Les systèmes d'apprentissage permettant d'obtenir une telle procédure peuvent être basés sur des hypothèses probabilistes (classifieur naïf de Bayesien), sur des notions de proximité (plus proches voisins) ou sur des recherches dans des espaces d'hypothèses (arbres de décisions). Nous décrivons, dans la suite la méthode de classification *SVM* [9].

2.1.1. Formalisation du problème de classification

Le problème de classification rentre dans le cadre de l'apprentissage statistique supervisé. Le but est de prévoir la classe *y* d'un vecteur *p*-dimensionnel *x* en se basant sur les mesures des variables qui l'expliquent avec pour seule information celle contenue dans un échantillon d'apprentissage *S*.

Dans le cas d'une discrimination bi-classe, nous supposons que les données sont des couples $(x_i, y_i)_{1 \le i \le l} \in X \times Y$ où *X* désigne l'espace des variables explicatives souvent pris dans R^p, $Y = \{-1, +1\}$ et l est la taille de l'échantillon. L'appartenance d'une observation x_i à une classe ou à une autre est matérialisée ici par la valeur −1 ou +1 de son étiquette y_i . L'échantillon d'apprentissage *S* est souvent dénoté par [14] :

$$S = \{(x_1, y_1), (x_2, y_2) \cdots (x_l, y_l)\} \in (X \times Y)^l \quad (1)$$

Le problème consiste alors, en s'appuyant sur l'ensemble d'exemples *S* à prédire la classe de toute nouvelle donnée. En termes de vocabulaire, on utilise le mot « étiquette » comme synonyme de « classe ».

Définition 1 [14] : Un exemple est une donnée pour laquelle on dispose de sa classe.

On utilise donc un ensemble d'exemples classés pour prédire la classe de nouvelles données ; c'est une tâche « d'apprentissage à partir d'exemples », ou « apprentissage supervisé ».

On parle de classification binaire quand le nombre de classes *Y* est égal à 2 ; il peut naturellement être quelconque. Dans tous les cas, il s'agit d'un attribut qualitatif pouvant prendre un nombre fini de valeurs. Dans l'absolu, une donnée peut appartenir à plusieurs classes : dans ce cas nous avons à faire à un problème multi-classes. Ici, on considère que chaque donnée appartient à une et une seule classe. L'objectif de la classification est de construire un modèle sur l'ensemble d'apprentissage. Ce modèle sera ensuite utilisé pour la classification des nouveaux individus (exemples) dont la classe d'appartenance est inconnue.

Définition 2 [14] : Un « classifieur » est un algorithme qui, à partir d'un ensemble d'exemples, produit une prédiction de la classe de toute donnée.

D'une manière générale, un classifieur procède par « induction » : à partir d'exemples (donc de cas connus), on construit une connaissance plus générale[2] . La notion d'induction de connaissances implique la notion de « généralisation » de la connaissance : à partir de connaissances éparses, les exemples, on induit une connaissance plus générale. Naturellement, même si l'on suppose que la classe des étiquettes n'est pas erronée, il y a un risque d'erreur lors de la généralisation ; ce risque

[2] Dans le cas de la déduction, on part d'une connaissance générale pour obtenir une information sur un cas particulier

est quantifié par la notion de « taux d'échec », ou d'« erreur en généralisation », qui sera définie plus loin dans cet article.

2.1.2. Évaluation d'une hypothèse de classification

La classification automatique vise donc à assigner des objets à des catégories ou classes. Le principe général des systèmes de classification inclut deux étapes [14, 15]:

- 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* au cours de laquelle les performances du système de classification sont évaluées.

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. Une fois le classifieur construit, il est essentiel de le valider en essayant d'estimer les erreurs de classification qu'il peut engendrer, autrement dit, la probabilité que la classe prédite pour une donnée quelconque soit incorrecte[3]. En fonction de l'ensemble de données qui est utilisé pour la mesurer, cette quantité est donc une variable aléatoire dont il faut estimer la valeur.

Définitions 3 [14] : L'erreur de classification (erreur en généralisation = taux d'échec) E d'un classifieur est la probabilité que ce classifieur ne prédise pas correctement la classe d'une donnée de l'espace de données.

Le taux de succès[4] est égal à $1-E$.

Une première approche, naïve, consiste à faire cette estimation en comptant le nombre d'exemples qui sont mal classés.

Définition 4 [14] : L'erreur apparente[5] Eapp (erreur d'apprentissage) est mesurée avec les exemples utilisés pour la construction du classifieur : c'est la proportion d'exemples dont la classe est mal prédite.

Eapp n'est pas un bon estimateur de l'erreur qui serait commise face à de nouvelles données. En effet, l'enjeu est bien là : à partir des exemples avec lesquels le classifieur a été construit, l'apprentissage doit pouvoir être généralisé à de nouvelles données. C'est là l'objectif principal des algorithmes d'apprentissage : un algorithme ne peut être qualifié d'algorithme d'apprentissage que s'il est capable de généraliser ce qu'il a appris. L'estimation de la qualité d'un algorithme construit en tant que classifieur de nouvelles données est donc un point capital.

[3] error rate
[4] success rate
[5] resubstitution error.

Pour avoir donc une estimation non optimiste de l'erreur de classification, il faut recourir à une base d'exemples qui n'ont pas servi pour l'apprentissage : c'est ce que l'on appelle la base de test. Celle-ci contient elle aussi des exemples étiquetés permettant de comparer les prédictions d'une hypothèse avec la valeur réelle de la classe. Cette base de test est généralement obtenue en réservant une partie des exemples supervisés initiaux et qui ne seront pas utilisés lors de la phase d'apprentissage. Lorsqu'on dispose de peu d'exemples, comme c'est le cas dans le traitement des données d'expression de gènes, il est pénalisant de laisser de côté une partie des exemples pendant la phase d'apprentissage [15].

On peut alors utiliser le processus de validation croisée [15] pour proposer une estimation du risque réel. L'algorithme de validation croisée à k blocs (k *−fold* cross-validation) consiste à découper l'ensemble initial d'exemples D en k blocs. On répète alors k phases d'apprentissage-évaluation, où une hypothèse h est obtenue par apprentissage sur (k−1) blocs de données et testée sur le bloc restant. L'estimateur de l'erreur est obtenu comme la moyenne des k erreurs empiriques ainsi obtenues. L'algorithme de validation croisée est décrit dans [15].

Dans la suite de ce papier, nous utiliserons la procédure de validation avec une base de données test pour évaluer les classifieurs que nous mettrons en œuvre à partir des jeux de données dont nous disposons.

2.1.3. Machine à vecteurs de support (SVM)

Les Séparateurs à Vaste Marge ou Support Vector Machines (*SVM*) ont été proposés en 1995 par V. Vapnik dans son livre « *The nature of statistical learning theory* » [9]. Elles sont des techniques largement répandues en apprentissage statistique, elles ont eu beaucoup de succès dans quasiment tous les domaines où elles ont été appliquées.

Elles reposent sur deux idées clés : la notion de marge maximale et la notion de fonction noyau. Cette méthode est donc une alternative récente pour la classification. Elle repose sur l'existence d'un classifieur linéaire dans un espace approprié.

Étant donné que c'est un problème de classification à deux classes, cette méthode fait appel à un jeu de données d'apprentissage pour apprendre les paramètres du modèle. Elle est basée sur l'utilisation de fonctions dites noyau (kernel) qui permettent une séparation optimale des données.

La notion d'apprentissage étant importante, nous allons commencer par effectuer un rappel sur cette notion.

L'apprentissage par induction permet d'arriver à des conclusions par l'examen d'exemples particuliers. Il se divise en apprentissage supervisé et non supervisé. Le cas qui concerne les *SVM* est l'apprentissage supervisé. Les exemples particuliers sont représentés par un ensemble de couples d'*entrées/sorties*. Le but est d'apprendre une fonction qui correspond aux exemples et qui prédit les sorties pour les entrées qui n'ont

pas encore été examinées. Les entrées peuvent être des descriptions d'objets et les sorties la classe des objets donnés en entrée.

Les machines à vecteurs de support [9] (SVM pour « Support Vector Machines ») sont des techniques largement répandues en apprentissage statistique, elles ont eu beaucoup de succès dans quasiment tous les domaines où elles ont été appliquées.

Dans ce papier, nous nous limitons au cadre de la classification linéaire. Dans ce cas, l'échantillon d'apprentissage est de la forme *D* avec :

$$D = \{(x_1, y_1), \dots, (x_n, y_n)\} \; et \; y_i = \{\pm 1\}, \quad (2)$$

Le problème consiste donc à trouver un hyperplan tel que : les données des étiquettes de classe $+1$ et -1 se trouvent de chaque côté de l'hyperplan et la distance des vecteurs les plus proches de l'hyperplan (pour chacune des deux classes) est maximale. Ces vecteurs sont appelés *vecteurs de support* et la distance de ceux-ci par rapport à l'hyperplan constitue la *marge optimale*. D'une façon plus formelle, notre objectif est de trouver un hyperplan $w \cdot x + b, w \in R \; et \; b \in R$ qui sépare les deux classes avec la plus grande marge. La recherche de la marge optimale permettant de déterminer les paramètres *w* et *b* de l'hyperplan conduit à un problème d'optimisation quadratique qui consiste (dans le cadre général) à minimiser :

$$\{\| w \|^2 + C \sum \epsilon_i | y_i (w . \Phi(x_i) + b) \geq 1 - \epsilon_i, \epsilon\_i \geq 0\} \quad (3)$$

Où C est un paramètre de compromis entre la marge et les erreurs[6], ϵ_i est une variable ressort associée à l'observation x_i, et ϕ est une transformation. Le problème peut être résolu (entre autre) par la méthode Lagrangienne d'optimisation quadratique avec contraintes (formulation duale) pour maximiser la marge [9] :

$$\{\sum \alpha_i - (1/2 \sum \alpha_i \; \alpha_j \; y_i \; y_j \; K(x_i, x_j) \mid 0 \leq \alpha_i \leq C, \sum \alpha_i \; y_i = 0)\} \quad (4)$$

où α_i est le multiplicateur Lagrangien associé aux vecteurs x_i. Si la valeur de α_i est non-nul alors x_i est un vecteur de support et $K(x_i, x_j) = \phi(x_i).\phi(x_j)$ est le noyau de transformation. Le noyau [9] d'un SVM est une fonction symétrique définie-positive qui permet de projeter les données dans un espace transformé de grande dimension dans lequel s'opère plus facilement la séparation des classes.

La décision est obtenue selon (le signe de) la fonction :

$$f(x) = sign[\alpha_i y_i K(x_i, x) + b] \quad (5)$$

- **Coefficients de classement d'un *SVM***

Lorsqu'une fonction de décision s'exprime comme une fonction linéaire des entrées, les coefficients de cette combinaison représentent l'influence de chaque variable d'entrée sur la classification. Dans un classifieur *SVM* linéaire, la valeur absolue des coordonnées du vecteur *w* indique l'importance de chaque variable d'entrée pour le

[6] Le choix de C est critique si les données sont bruitées.

calcul de l'hyperplan séparateur. Par exemple, si l'hyperplan est orthogonal à une direction correspondant à un attribut α_j alors cet attribut est important pour la classification et sera affecté d'un poids fort. Ces coefficients peuvent donc être utilisés pour classer les différentes variables selon leur pertinence pour la classification.

Pour un classifieur *SVM* linéaire construit sur un ensemble d'attributs donné, et dont la fonction de décision s'écrit $w.x + b = 0$, on appellera **vecteur des coefficients de classement des attributs** le vecteur c tel que :

$$\forall\, j, c_j = (w_j) \quad (6)$$

Cette idée est utilisée dans [9] ou l'auteur propose une méthode d'élimination récursive des variables (*SVM-RFE*). Plus exactement, dans cette sélection séquentielle qui part de l'ensemble complet des attributs, on élimine progressivement l'attribut le moins pertinent.

Pour déterminer l'attribut à enlever à chaque itération, on considère l'attribut qui a l'influence la plus petite sur la fonction de coût du processus de classification, c'est-à-dire, l'attribut qui a le plus petit coefficientc_j.

Dans la suite de ce papier, nous étudierons la technique de sélection d'attributs pour la classification supervisée avec les SVM.

2.2. La sélection d'attributs

Les performances d'un classifieur dépendent fortement de la qualité de représentation des données à traiter, ce qui implique généralement l'obligation de représenter ces dernières au moyen d'un nombre élevé d'attributs représentatifs. Il est alors fréquent qu'une partie de celles-ci ne contienne que des informations non pertinentes, redondantes ou inutiles à la tâche de classification, rendant cette dernière plus complexe. Il est donc nécessaire, lors de la construction d'un système de classification, de limiter le nombre d'attributs pris en compte, de manière à optimiser ses performances. C'est exactement ce que fait le processus de « sélection d'attributs » qui a pour but de filtrer le vecteur des attributs de manière à en extraire l'information discriminante et pertinente améliorant la qualité du système [11].

La sélection d'attributs est un sujet de recherche très actif depuis une dizaine d'années dans les domaines de l'apprentissage artificiel, de la fouille de données, du traitement d'images, et de l'analyse de données en bio-informatique [12, 13]. Dans tous ces domaines, les applications nécessitent de traiter des données décrites par un très grand nombre d'attributs. Ainsi, on peut avoir à traiter des pages web décrites par plusieurs milliers de descripteurs, des images décrites par plusieurs millions de pixels ou des données en bio-informatique donnant les niveaux d'expression de plusieurs milliers de gènes [12, 13].

La sélection de sous-ensembles de variables (en anglais *feature subset selection* ou *FSS*) est un composant essentiel pour la construction de modèles d'aide à la décision. La *FSS* permet de diminuer la complexité des modèles tout en conservant une efficacité au moins aussi bonne, puisque seules les variables informatives et non-redondantes en

information seront conservées. En outre, la *FSS* permet également à un expert d'avoir une vue sur les variables pertinentes de son problème.

Récemment, la taille des bases de données a augmenté considérablement, tant en nombre d'instances qu'en nombre d'attributs considérés. Adapter ou créer de nouvelles méthodes de sélection efficaces à ces nouvelles échelles est un nouveau défi. Ce défi présente de bonnes opportunités et challenges pour la recherche en extraction de connaissances et en apprentissage machine.

La thématique de sélection de caractéristiques est un domaine de recherche actif depuis plusieurs décennies [1, 7, 8, 12]. Elle consiste à extraire de l'ensemble des variables explicatives disponibles un ensemble optimal de caractéristiques les plus importantes par rapport à un système donné afin de mener à bien la tâche pour laquelle il a été conçu. De nombreux travaux et publications traitent de ces techniques qui sont appliquées dans un grand nombre de domaines [1, 7, 8, 11, 12].

Au cours de ces quelques dernières années, de nouvelles techniques ont été proposées pour aborder cette stimulante tâche en présence de milliers de variables explicatives. Ces techniques sont essentiellement basées sur les machines à vecteurs supports [9]. Le choix des *SVM* pour faire face à ce défi s'explique par le grand succès qu'a connu cette méthode d'apprentissage dans différents domaines d'applications et surtout par la richesse de son fondement théorique.

Le fondement théorique des *SVM*, abordé dans la deuxième partie de ce papier, nous montre que l'augmentation du nombre d'attributs ne devrait pas nuire à la qualité de la discrimination qu'elles réalisent. En revanche, la qualité des données pose néanmoins des problèmes majeurs dans les applications. Cependant, la sélection appropriée d'attributs porte des avantages multiples : améliorer la performance prédictive du modèle construit, faciliter l'interprétation des données et réduire le temps de calcul.

La sélection de variables joue un rôle très important en classification lorsqu'un grand nombre p de variables sont disponibles, certaines pouvant être peu significatives, corrélées ou non pertinentes au regard de l'application considérée [1, 2]. Elle consiste à sélectionner un sous-ensemble de q variables ($q < p$) sans que les performances de la règle de classement diminuent trop voire même augmentent. La sélection permet également de faciliter l'étape d'apprentissage et de réduire la complexité des algorithmes ainsi que les temps de calcul.

Les techniques de sélection d'attributs sont divisibles en trois catégories, selon la manière dont elles interagissent avec le classifieur.

Les méthodes filtres [1] (*filter methods*) opèrent directement sur le jeu de données et fournissent une pondération, un classement ou un ensemble de variables en sortie. Ces méthodes ont l'avantage d'être rapides et indépendantes du modèle de classification, mais au prix de résultats inférieurs.

Les méthodes enveloppes [1] (*wrapper methods*) effectuent une recherche dans l'espace des sous-ensembles de variables, guidée par le résultat du modèle, par exemple les performances en validation croisée sur les données d'apprentissage. Elles ont souvent de meilleurs résultats que les méthodes de filtrage, mais au prix d'un temps de calcul plus important [2].

Enfin, les méthodes embarquées (*Embedded methods*) utilisent l'information interne du modèle de classification par exemple, le vecteur de poids dans le cas des *SVM* (*support vector machines*). Ces méthodes sont donc proches des méthodes d'enveloppes, du fait qu'elles combinent le processus d'exploration avec un algorithme d'apprentissage sans étape de validation, pour maximiser la qualité de l'ajustement et minimiser le nombre d'attributs [4]. La différence de ces dernières, avec les méthodes enveloppes est que le classifieur sert non seulement à évaluer un sous-ensemble candidats mais aussi à guider le mécanisme de sélection. Selon Elisseeff et al [4], ces méthodes seraient bien plus avantageuses en terme de temps de calcul que les méthodes de type *wrapper* et seraient robustes face au problème de sur-ajustement.

Étant donné que plusieurs approches sont possibles pour le développement d'un algorithme de sélection d'attributs, il en résulte qu'un grand nombre d'algorithmes est disponible [1, 7, 8, 11, 12].

Parmi ces algorithmes, on note l'algorithme d'élimination récursive d'attributs par machines à vecteurs de support (*Recursive Feature Elimination with Support Vector Machines RFE-SVM*), introduit dans [1].

2.2.1 Algorithme RFE-SVM

RFE-SVM [1, 13] est un algorithme de sélection d'attributs pour la classification supervisée. Cet algorithme fait partie des algorithmes de type « *Embedded* ». En effet, il intègre le filtrage dans le processus d'apprentissage *SVM* dans le but d'évaluer chaque sous-ensemble grâce à un classifieur *SVM* mais aussi pour avoir des informations sur la contribution de chaque attribut sur la construction de l'*hyperplan séparateur*.

La méthode repose sur le fait que chaque élément w_j du vecteur de poids *w* sur chaque variable *j* est une combinaison linéaire des observations et que la plupart des α_j sont nuls, exceptés pour les observations support (voire section 2). Par conséquent, la mesure *w* peut être directement reliée à l'importance des variables dans le modèle *SVM*.

En effet, la mesure w^2 est une mesure de pouvoir prédictif. Ainsi, les variables *j* de plus petit poids w_j^2 seront progressivement éliminées dans la procédure *RFE*. Ces scores sont ici simplement les composantes du vecteur de poids *w*, définissant l'hyperplan optimal obtenu, qui est, rappelons le, une combinaison linéaire des *n* vecteurs supports x_i du problème, faisant intervenir les multiplicateurs de Lagrange α_j [9] :

$$w\_j = \sum_{i=1}^{ns} \alpha_j y_j x_j$$

L'idée est que les attributs, qui correspondent à des directions de l'espace selon lesquelles le vecteur *w* 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).

Donc, à chaque récursion de l'algorithme *SVM-RFE*, l'attribut possédant le score le plus faible est éliminé. Le processus est arrêté lorsque le critère d'arrêt est atteint.

L'algorithme RFE-SVM de base s'écrit comme suit :

ALGORITHME 1 : RFE-SVM

Input :

$s = \{1,2,..,p\}$: Ensemble d'apprentissage ;

$S = s;$

d : nombre d'attributs fixés ;

Ouput :

$Sol = s$: Ensemble solution ;

1. While $s \geq d$ **do**

2. Apprentissage du SVM sur les données : x_{is} *et* y_i

3. Calcul du vecteur de poids : $w = \sum \alpha_i x_i y_i$

4. Calcul du critère de rang : $c_i = \omega_i^2$ pour indice i dans s ;

5. Trouver la variable avec le plus petit critère de rang $f = \arg\min(c_i);$

6. Eliminer la variable avec le plus petit critère de rang : $Sol = \{S\} \setminus f$;

7. end

8. Return Sol ;

Soulignons que cet algorithme ne peut utiliser que des *SVM* linéaires, pour lesquels la contribution d'un attribut x_i à la fonction de décision est un terme linéaire $w_i x_i$. Dans les cas où la surface de décision est non-linéaire, la pertinence d'un attribut peut dépendre de la région dans laquelle se trouve *x*, ce qui exclut l'utilisation des *SVM* non-linéaires à des fins de sélection d'attributs globalement pertinents.

L'étape d'apprentissage du *SVM* pouvant être coûteuse en calculs, en particulier pour les itérations initiales ou le nombre d'attributs utilisés est grand, plusieurs attributs peuvent être éliminés simultanément en une itération ; il s'agit dans ce cas de ceux ayant les poids les plus faibles [1]

Pour accélérer le temps de calcul, les auteurs [1] proposent d'éliminer plusieurs variables à la fois, bien que la performance de classification puisse en être affectée. Dans ce cas-là, on obtient non pas un critère de rang sur des variables, mais un critère de rang sur des sous-ensembles de variables qui sont imbriqués les uns dans les autres. Si les variables sont éliminées une à une comme le propose l'algorithme initial, les auteurs dans [1] mettent en garde sur la pertinence des variables du plus haut rang : seul le sous-ensemble de variables sélectionné est optimal, et non pas les variables de plus haut rang considérées individuellement. En effet, *RFE* est une méthode de type *«wrapper»* qui va avoir tendance à sélectionner des variables comportant de l'information complémentaire, améliorant ainsi la tâche de classification. Les attributs considérés un à un dans la sélection ne contiennent que peu d'information pertinente.

Il est important de noter que *RFE* ne s'intéresse pas à la recherche du sous-ensemble de taille optimale, mais donne une mesure d'importance sur chaque variable ou groupe de variables. Les auteurs dans [1] ne spécifient pas précisément comment choisir les différentes tailles des sous-ensembles récursivement éliminés dans la deuxième approche.

RFE a été utilisé en bioinformatique pour l'analyse du niveau d'expression de gènes [12, 13] et dans l'analyse de données de transcriptome [10]. Ceci a montré que *RFE-SVM* sélectionne de très bons sous-ensembles d'attributs ; par contre avec les ensembles d'apprentissage constitués de peu d'échantillons et disposant d'un ensemble d'attributs très important (environ 1200 gènes ou plus), *RFE* prend un temps considérable pour sélectionner un sous-ensemble optimal.

Ainsi avec la version originale de *RFE* où un attribut est éliminé par itération, le temps d'exécution de l'algorithme *RFE-SVM* est extrêmement coûteux.

Pour surmonter cet inconvénient afin d'essayer d'accélérer le processus de calcul, plusieurs variantes de l'algorithme ont été proposées parmi lesquelles nous allons citer l'algorithme *RFE-annealing* [16] qui est basé sur un algorithme de recuit simulé, de façon à éliminer un nombre important de variables lors des premières itérations, puis à réduire le nombre de variables éliminées. Ceci permet de réduire de façon significative le temps de calcul. Cette méthode, très proche de *RFE*, nécessite de fixer (par l'utilisateur) le nombre de variables à sélectionner. Un autre exemple, est celui de *SVM-RCE* [17] qui consiste à sélectionner des ensembles de variables corrélés. La critique principale de *RCE* concernant *RFE* est que certaines variables avec des petits poids (et donc jugés peu informatifs) peuvent être importants, mais leur rang bas est le résultat de la présence d'autres variables dominants hautement corrélés à ces derniers. Ceci soulève le problème des variables corrélés et comportant de l'information redondante.

Il y'a également l'algorithme *SQRT-RFE* [16] qui élimine $\sqrt{S}$ attributs par itération, où S est le nombre d'attributs qui restent de chaque itération. Une autre variante est l'Entropy-based *RFE* (*E-RFE*) [18], qui est basée sur une fonction d'entropie pour la

détermination de l'attribut à enlever. Cette variante se base sur la structure de la répartition du poids et elle introduit la notion de fonction d'entropie *H* qui sera utilisé comme une mesure de la répartition de poids.

Dans le cadre de ce travail, nous utilisons l'algorithme *RFE-SVM* de base mais pour plus d'informations sur les variantes de *RFE-SVM* proposées les lecteurs pourront se référer aux références [16, 17, 18, 19, 20] pour plus de détails.

2.2.2. Problématique

Comme nous l'avons énoncé dans l'introduction, la problématique de filtrage de données ou de sélection d'attributs pour la classification a fait introduire un axe de recherche sur lequel travaillent activement les spécialistes en Datamining (fouille de données) et en optimisation [1, 2, 3, 12, 13].

L'étude de l'existant réalisée dans le précédent point de ce papier, nous a permis de constater que le problème de la sélection d'attributs fait parti des problèmes *NP-difficiles* [1, 3, 11, 21, 22] puisque pour *N* attributs, nous avons 2^{N-1} sous ensembles possibles (il est considéré comme un problème d'optimisation combinatoire). Ainsi, une résolution exhaustive est exponentiellement prohibitive.

Et comme nous l'avons aussi mentionné ci-dessus, la version de base de *RFE-SVM* repose sur le principe de l'élimination itérative, « Recursive Feature Elimination » (*RFE*), c'est à dire qu'à chaque itération de l'algorithme, il y a un attribut qui est éliminé. D'autres versions qui éliminent plus qu'un attribut, ont été proposées, pour accélérer l'approche de base. Il existe même des approches qui traitent des milliers d'attributs et éliminent la moitié des attributs dans les premières itérations.

Toutes ces propositions ont été faites dans le but d'améliorer et d'adapter l'approche de base en éliminant plus d'un attribut par itération.

Dans cette étude, nous avons constaté que *RFE-SVM* est un algorithme complètement glouton, c'est-à-dire que quelque soit la variante adoptée, l'attribut ou le sous-ensemble d'attributs éliminé ne peut plus jamais revenir dans le sous-ensemble sélectionné, ce qui pourrait biaiser la recherche et la diriger vers des minima locaux.

On se propose donc de donner aux attributs éliminés dans les itérations antérieures la chance de revenir dans la recherche, ce qui pourrait, peut être, améliorer la qualité de la solution retournée par *RFE-SVM*.

3. Notre Proposition

3.1. Principe de l'approche

Le problème de la sélection d'attributs est un problème NP-difficile. C'est pourquoi, les méta-heuristiques à base de voisinage ou les méthodes de recherche locale semblent bien appropriées pour traiter ce problème. Ainsi, nous considérons le problème de la sélection de gènes comme un problème combinatoire de maximisation *(S, f)* où *S* est l'ensemble des attributs et *f* la fonction à maximiser (fonction objective), c'est-à-dire qu'on cherche un sous-ensemble d'attributs afin de construire un classifieur avec la performance maximum prédisant la classe de chaque nouvelle entrée.

Plus exactement, nous formulons ce problème de maximisation *(S, f),* tel que :

- l'espace de recherche *S* est défini par les sous-ensembles d'attributs sélectionnés possibles.
- la fonction objective $f : S \rightarrow R$ est telle que quelque soit $s \in S$, $f(s)$ représente le taux de classification d'un classifieur construit avec le sous-ensemble d'attributs *s*.

Dans la section suivante nous présenterons les adaptations des différents éléments qui constituent les méthodes de recherche locale utilisées pour aborder ce problème.

Afin d'explorer et exploiter des zones intéressantes de l'espace de recherche, nous avons envisagé de limiter cet espace. Pour cela, notre modèle débute par une phase de présélection ou de construction d'une solution initiale réalisée grâce à l'algorithme *RFE-SVM* de base. Cette présélection permet de restreindre l'espace de recherche et nous fournit un groupe initial d'attributs pertinents sur lequel nos opérateurs de recherche locale vont travailler pour essayer d'améliorer la solution déjà construite.

3.1.1. Phase 1- Construction de la solution initiale

Cette phase peut être considérée comme une phase d'initialisation, qui permet de construire une solution initiale pour débuter la recherche. La solution initiale peut être créée au hasard au risque d'avoir une solution de mauvaise qualité (cas des méthodes aléatoires). C'est pourquoi, nous proposons d'appliquer un algorithme glouton de sélection d'attributs pour la classification supervisée afin d'obtenir une solution initiale qui ne soit pas trop mauvaise. C'est dans ce sens que l'algorithme *RFE-SVM* a été choisi, car il a donné des résultats encourageants pour la construction de classifieur acceptables [1]. Nous rappelons que ce dernier est un algorithme complètement glouton et qu'il ne retourne généralement pas une solution optimale.

Pour rappel, les algorithmes gloutons (greedy algorithms en anglais) sont des algorithmes pour lesquels, à chaque itération, on fixe la valeur d'une (ou plusieurs) des

variables décrivant le problème sans remettre en cause les choix antérieurs. Le principe est donc de partir d'une solution incomplète (éventuellement totalement indéterminée) que l'on complète de proche en proche en effectuant des choix définitifs : à chaque étape, on traite une partie des variables sur lesquelles on ne reviendra plus. Certains de ces algorithmes conduisent néanmoins à une solution exacte, mais très rarement [23]. Si les algorithmes de ce type sont souvent rapides, en revanche la solution qu'ils déterminent peut être arbitrairement éloignée de la solution. On les utilise néanmoins fréquemment pour obtenir rapidement une solution réalisable. Par exemple, elles servent à initialiser une méthode itérative.

3.1.2. Phase 2 - Amélioration de la solution

Cette seconde phase est introduite afin d'essayer d'améliorer la qualité de la solution retournée par *RFE-SVM* dans la première phase.

L'idée de l'amélioration de la recherche est donc de proposer des *bactracks* occasionnels ou fréquents qui donneraient une chance supplémentaire aux attributs éliminés dans les itérations antérieures.

Le fait de considérer plusieurs solutions à la fin de chaque itération pourrait être formalisé par une recherche locale. Dans le papier [7], plusieurs techniques de recherche locales sont proposées et évaluées empiriquement (dans un schéma d'optimisation de type algorithme génétique (*AG*)). Dans la suite, nous proposons d'appliquer ces opérateurs ou procédures de recherche locale au contexte du *SVM*.

3.2. Procédure de recherche locale

Les opérateurs de recherche locale [7, 8] sont souvent présentés comme des méthodes qui peuvent être à la fois intuitives et très efficaces pour résoudre de manière approchée des instances de problèmes combinatoires, lorsque ces instances possèdent des solutions. Ces opérateurs de recherche locale s'appuient sur le principe suivant:

Partant d'une solution quelconque *s*, qui peut être prise au hasard ou peut être construite par un algorithme dans l'espace de recherche *S* (dans notre cas l'algorithme *RFE-SVM*), ces méthodes cherchent à améliorer, de façon itérative, la solution existante en explorant le voisinage de celle-ci.

Un operateur de recherche locale se caractérise par deux éléments : une fonction d'évaluation et une fonction de voisinage. D'ailleurs le nom des fonctions de voisinage est utilisé pour désigner chaque opérateur utilisé dans ce travail.

Il s'agit principalement des opérateurs *Bit-Flip[7]* et *Attribute- Flip[7]*. Nous allons dans ce qui suit donner les détails de chacun des deux opérateurs cités précédemment.

- Operateur *Attribute-Flip :* Dans le cas de l'opérateur *Attribute-Flip*, la fonction de voisinage est basée sur l'opération enlever/ajouter qui enlève

un attribut f_j de la solution X et ajoute un autre attribut f_i appartenant aux sous-ensembles d'attributs supprimés. L'opérateur *Attribute-Flip* est illustré par l'équation 5.

$$NB_{AF} = \{X|X = S \cup \{f_i\} - \{f_j\}, \forall f_i \in X, f_j \notin X\} \quad (7)$$

– Operateur *Bit-Flip :* L'opérateur *Bip-Flip* lui aussi est basé sur l'opération enlever/ajouter qui retire un attribut a_i de la solution X ou ajoute un autre attribut a_j appartenant au sous-ensemble d'attributs supprimés. Ici on ne fait pas l'échange, mais cet opérateur va explorer le voisinage de la solution retournée par *RFE-SVM* en ajoutant ou en supprimant un attribut. La fonction de voisinage de l'opérateur *Bit-Flip* (*BF*) est illustrée par l'équation 6.

$$NH_{BF}(S) = \{X|X = NH^{+}(S) \cup NH^{-}(S)\} \quad (8)$$

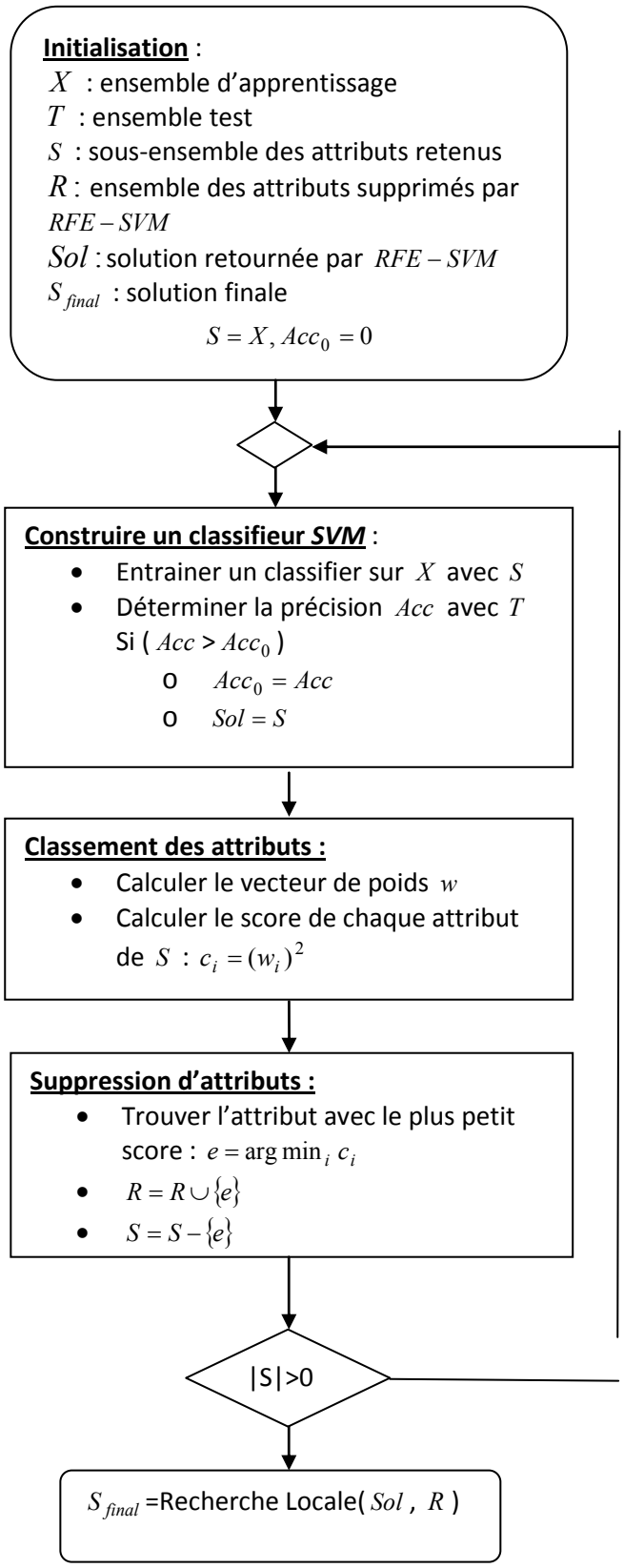
avec $NH^{+}(S)$ et $NH^{-}(S)$ qui désignent, respectivement, les solutions voisines obtenues en ajoutant ou en supprimant un attribut à la solution courante.

3.2. Notre algorithme

Nous avons proposé une démarche de sélection d'attributs suivant un processus à deux étapes (voir ci-dessous). Dans la première phase que nous appelons : phase d'initialisation, nous proposons d'appliquer l'algorithme *RFE-SVM* [1, 13] de base afin d'obtenir une solution initiale qui ne soit pas trop mauvaise.

Rappelons que *RFE-SVM* est très utilisé dans la littérature mais le caractère glouton de cet algorithme fait qu'il ne retourne généralement pas une solution optimale. A la seconde phase, que nous appelons phase d'amélioration, nous allons introduire de la recherche locale [7, 8] afin d'améliorer la qualité de la solution retournée à la première phase

L'organigramme général de notre algorithme de sélection d'attributs et le code source de l'opérateur de recherche locale que nous avons utilisé sont donnés ci-après:



ALGORITHME 2 : Recherche locale

INPUT :

S : Solution retournée par $RFE-SVM$

S' : Attributs supprimés par $RFE-SVM$

Cla : Le classifieur SVM utilisé

OUTPUT :

S_{local} : Solution retournée par la recherche locale

BEGIN

$S_1 \leftarrow S$, $S_{best} \leftarrow S_1$

$Stop \leftarrow false$

REPEAT

$Sol_{list} \leftarrow NH(S_1, S')$

$\forall Sol \in Sol_{list}, Evaluate(Sol, Cla)$

$S_1 = getBest(Sol_{list})$

IF ($S_1.fitness \succ S_{best}.fitness$ **) THEN**

$S_{best} \leftarrow S_1$

ELSE

$Stop \leftarrow true$

END IF

UNTIL ($Stop = true$)

$S_{local} \leftarrow S_{best}$

RETURN (S_{local})

END

4. Experimentation

4.1. Protocole d'expérimentation

4.1.1. Simulateur: Weka (Waikato Environment for Knowledge Analysis)

Pour la validation expérimentale, nous avons utilisé le simulateur Weka[24]. Le système Weka est développé par l'université de Waikato, en Nouvelle Zélande. Il permet de prétraiter des données, de les analyser à l'aide de méthodes de datamining et d'afficher le modèle résultats et ses performances. Weka est entièrement développé en java et traite des données au format *ARFF* (Attribute Relation Format File) ou *CSV* (Comma Separated Values). Un certain nombre de jeu de données, aux formats *ARFF*, issus du site de l'*UCI[5]* (University of Caliornia - Irvine) peuvent être téléchargés à partir de la même adresse.

4.1.2. Données de l'*UCI*

L'Université de Californie à Irvine (*UCI*) offre gracieusement à la communauté de fouille de données une série d'ensemble de données présentant des caractéristiques variées pour la validation d'algorithmes de classification. Nous avons utilisé ces jeux de données publics, facilement accessibles et qui sont utilisés dans de nombreux travaux concernant la classification des données. Ces jeux constituent en quelque sorte des jeux tests qui permettent de comparer les méthodes proposées depuis quelques années. Nous donnons ci-dessous les caractéristiques des jeux de données que nous avons choisis pour la validation de notre algorithme.

- ✓ L'ensemble de données JOHNS UNIVERSITY IONOSPHERE DATABASE est constitué de données radar permettant la détection des structures particulières dans l'ionosphère. Il s'agit d'un problème de classification binaire, avec 63 instances appartenant à une première classe appelée classe *b* et les 113 instances restantes appartiennent à une seconde classe, nommée classe *g*. Les données sont représentées par 35 attributs.
- ✓ L'ensemble de données SPAMBASE contient les informations relatives à 512 messages, contenant 202 spams. Cette collection a été prétraitée, et les textes des messages ne sont pas disponibles. Chaque message est décrit en fonction de 57 attributs, le 58è*me* correspond à la classe du message (spam ou message légitime). Les caractéristiques (55-57) mesurent la longueur des séquences de lettres majuscules consécutives.
- ✓ L'ensemble de données SPECT HEART DATA décrit le diagnostic de problèmes cardiaques par émission (SPECT) images. Cet ensemble est constitué de 80 patients issus de deux populations : normaux et anormaux. Chaque patient est décrit en fonction de 22 attributs, le 23ème correspond à la classe du patient (malade ou sain). La matrice des niveaux d'expression comporte donc 23 colonnes et 80 lignes.

- ✓ L'ensemble de données SONAR, MINES vs. ROCKS est constitué de deux classes de données. La première classe est composée de 49 instances représentant un cylindre métallique observé selon différentes conditions par un sonar. La seconde est composée de 55 instances d'une pierre selon les mêmes conditions. Les objets sont représentés par 61 attributs.

Enfin, le tableau 1 suivant résume les caractéristiques de ces trois jeux de données.

Données	***N***	***I***	***C***
Ionosphere	35	176	2
SpamBase	58	512	2
Spect Heart Data	23	80	2
Sonar	61	104	2

Tableau 1 - Description des données, *N* désigne le nombre d'attributs, *I* le nombre d'instances et *C* le nombre de classes de l'échantillon d'apprentissage.

4.1.3. Le matériel utilisé

Le matériel utilisé est un ordinateur portable avec processeur Pentium(R) Dual-Core CPU T4400 @ 2.20GHz (2 CPUs), 2.2GHz avec 4096MB de RAM, sous le système d'exploitation Windows 7.

4.2. Résultats expérimentaux

Nous avons choisi d'évaluer les performances en terme de précision du classifieur construit à l'aide de ces trois jeux de données publics présentés ci-dessus.

Dans la pratique, nous divisons chaque base de données comme suit: les deux tiers des données sont utilisées pour la sélection des caractéristiques et l'apprentissage des classifieurs, la partie restante, c'est-à-dire le tiers, étant utilisé comme ensemble test.

La performance est mesurée par le taux de bonne classification moyennée sur les ensembles de test. Les résultats obtenus comparés aux résultats de l'algorithme RFE-SVM de base sont présentés dans le tableau 2, ou *N* désigne le nombre d'attributs retenu par l'algorithme de la première ligne et *P* la précision du classifieur construit sur les *N* attributs.

Datasets	***RFE –SVM***	***RFE –SVM +BF***	***RFE –SVM +AF***
Ionosphere	N = 5, P = 85.714%	N= 6, P = 85.714%	N = 5, P = 86.286%
SpamBase	N = 39, P=89.606%	N= 38, P=89.729%	N = 39, P=89.704%
Spect Heart Data	N = 5, P = 71.123%	N= 6, P = 72.727%	N = 5, P = 73.262%
Sonar	N= 41, P = 81.731%	N= 40, P= 81.731%	N= 41, P = 81.731

Tableau 2 - ***Comparaisons des résultats de classification.***

Dans le Tableau 2, nous faisons une comparaison des résultats de classification obtenus par notre algorithme de sélection d'attributs avec ceux de l'algorithme *RFE-SVM*.

Nous remarquons que sur ces ensembles de données, notre algorithme de sélection d'attributs retourne un taux de prédiction de qualité égale ou supérieure à celui retourné par l'algorithme *RFE-SVM* de base.

Concernant l'ensemble de données *Sonar*, notre algorithme obtient un taux de prédiction égal à celui de *RFE-SVM*, mais si nous faisons la comparaison en terme de nombre d'attributs retenu dans le sous-ensemble solution, nous pouvons dire que notre algorithme est meilleur car il ne retient que 40 attributs contre 41 contenus dans le sous-ensemble solution retourné par l'algorithme *RFE-SVM*.

En nous basant sur ces résultats issus des expériences que nous avons réalisées, nous pouvons affirmer que notre approche (*RFE-SVM+BF et RFE-SVM+AT*) de sélection d'attributs est très bien fondée. En effet, sur les quatre ensembles de données utilisés dans ce papier, notre algorithme de sélection arrive à améliorer la qualité de la solution retournée par l'algorithme *RFE-SVM*, et nous avons ainsi produit des classifieurs plus performants que ceux produit par l'algorithme RFE-SVM de base.

5. Conclusion

Dans ce papier, nous avons étudié l'algorithme *RFE-SVM* qui est un algorithme de sélection d'attributs pour la classification supervisée. Celui-ci exploite essentiellement la richesse du bagage théorique sur lequel sont basées les machines à vecteurs supports (*SVM*). L'analyse de l'existant, nous a permis de noter que la principale faiblesse de *RFE-SVM* est liée au fait qu'il est glouton (i.e. ne fait pas de retour en arrière). Par conséquent la solution qu'il retourne est généralement non optimale. Nous nous sommes donné comme principal objectif d'améliorer *RFE-SVM* en autorisant les retours en arrière.

La variante que nous avons proposé est constituée de deux phases : (1) dans la première phase, appelée phase d'initialisation nous avons utilisé l'algorithme *RFE-SVM* pour la construction d'une solution initiale de bonne qualité mais non-optimale ; et (2) dans la seconde phase, nommée phase d'amélioration, nous avons utilisé des opérateurs de recherche locale qui vont autoriser les retours en arrière afin d'améliorer la qualité de la solution initiale.

Nous avons implémenté et testé notre algorithme sur des datasets benchmark d'*UCI* repository. Les résultats issus des expérimentations ont été concluants et ont montré le bien fondé de notre approche. En effet, sur quatre des bases de données que nous avons utilisées, les expériences ont montré que le fait de donner aux attributs supprimés dans les itérations antérieures la chance de revenir dans la sélection permet d'améliorer la qualité de la solution retournée par l'algorithme *RFE-SVM* et ainsi de rendre le classifieur proposé plus performant.

Bien que les résultats obtenus soient intéressants et encourageants, beaucoup de points sont susceptibles d'être étudiés dans le cadre de travaux futurs.

Donc en guise de perspectives, nous comptons, dans le court terme, continuer à expérimenter notre algorithme sur d'autres datasets benchmarks afin de voir le comportement de notre approche sur d'autres bases de très grandes tailles.

Et dans le long terme nous comptons utiliser, au sein de notre procédure de sélection d'attributs toutes les variantes de *RFE-SVM*, mais aussi d'autres opérateurs de recherche locale afin de desceller celles qui conviennent le plus pour notre approche, i.e. qui retournent le meilleur taux de classification. Aussi, nous envisageons, d'étudier la classification multi-classe, pour voir comment se comporte notre algorithme dans ce cas de figure. Enfin, nous désirons également faire une étude comparative des algorithmes de sélection d'attributs pour la classification supervisée qui existe dans la littérature afin de pouvoir situer notre approche en qualité de taux de bonne classification et de nombre d'attributs retenu.

6. Bibliographie

[1] Guyon, Weston, Barnhill, and Vapnik, "*Gene selection for cancer classification using support vector machines*," *MACHLEARN: Machine Learning*, vol. 46, 2002.

[2] P. A. Mundra and J. C. Rajapakse, "*SVM-RFE with relevancy and redundancy criteria for gene selection*," in *PRIB*, J. C. Rajapakse, B. Schmidt, and L. G. Volkert, Eds., vol. 4774. Springer, 2007, pp. 242–252.

[3] H. Liu and L. Yu, "*Toward integrating feature selection algorithms for classification and clustering*," *IEEE Trans. Knowl. Data Eng*, vol. 17, no. 4, pp. 491–502, 2005.

[4] T. N. Lal, O. Chapelle, J. Weston, and A. Elisseeff, *Embedded Methods*. Springer, Nov. 20 2004.

[5] "Uci machine learning repository," http://archive.ics.uci.edu/ml/datasets.

[6] A. K. Jain and D. E. Zongker, "*Feature selection: Evaluation, application, and small sample performance*," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 19, no. 2, pp. 153–158, 1997.

[7] M. A. Esseghir, G. Goncalves, and Y. Slimani, "*Memetic feature selection: Benchmarking hybridization schemata*," in *HAIS(1)*, 2010, pp. 351–358.

[8] J. C. H. Hernandez, B. Duval, and J.-K. Hao, "*Svm based local search for gene selection and classification of microarray data*," in *BIRD*, 2008, pp. 499-508.

[9] V. N. Vapnik, "*The nature of statistical learning theory"*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

[10] Sridhar Ramaswamy, Pablo Tamayo, Ryan Rifkin, Sayan Mukherjee, Michael Angelo, Christine Ladd, Michael Reich, P. Mesirov, Tomaso Poggio, William Gerald, Massimo Loda, Eric S. L, and Todd R. Golub. *SUPPLEMENTARY INFORMATION multi-class cancer diagnosis using tumor gene expression signatures*, December 27 (2001).

[11] Manoranjan Dash and Huan Liu. *Feature selection for classification*. Intell. Data Anal **1**(1-4), 131–156 (1997).

[12] Isabelle Guyon and André Elisseeff. *An introduction to variable and feature selection*. J. Mach. Learn. Res. **3**, 1157–1182 March (2003).

[13] F. Camara et al, Privacy Preserving RFE-SVM for Distributed Gene Selection, IJCSI, Vol. 9, Issue 1, No 2, Pages 154-159, 2012.

[14] Fouille de données notes de cours ph. preux université de lille. http://www.grappa.univ-lille3.fr/~ppreux/fouille.

[15] Ron Kohavi. Feature subset selection as search with probabilistic estimates. In *in AAAI Fall Symposium on Relevance*, pages 122–126 (1994).

[16] Yuanyuan Ding and Dawn Wilkins. Improving the performance of SVM-RFE to select genes in microarray data.

[17] Malik Yousef, Segun Jung, Louise C. Showe, and Michael K. Showe. Recursive cluster elimination (RCE) for classification and feature selection from gene expression data. BMC Bioinformatics **8** (2007).

[18] C. Furlanello, M. Serafini, S. Merler, and G. Jurman. An accelerated procedure for recursive feature ranking on microarray data. Neural Networks **16** (5-6), 641–648 (2003).

[19] Yuchun Tang, Yan-Qing Zhang, and Zhen Huang. Development of twostage SVM-RFE gene selection strategy for microarray expression data analysis. IEEE/ACM Trans. Comput. Biology Bioinform **4**(3), 365–381 (2007).

[20] Piyushkumar A. Mundra and Jagath C. Rajapakse. SVM-RFE with relevancy and redundancy criteria for gene selection. , **4774**, pages 242–252. Springer (2007).

[21] Scott Davies and Stuart Russell. NP-completeness of searches for smallest possible feature sets, October 18 (1994).

[22] Carlos Cotta and Pablo Moscato. *The -feature set problem is [2] complete*. J. Comput. Syst. Sci. **67**(4), 686–690 (2003).

[23] J.-K. Hao, P. Galinier, and M. Habib. Méthodes heuristiques pour l'optimisation combinatoire et l'affectation sous contraintes. In S. Pesty and P. Siegel, editors, *Actes des 6 es Journées Nationales du PRC-GDR Intelligence Artificielle*, pages 107–144. Hermès, Paris (1997).

[24] Ian H. Witten and Eibe Frank. *Data mining : practical machine learning tools and techniques with Java implementations*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2000).