Application de l'apprentissage automatique

2.1 Introduction

Ce chapitre sera consacré aux résultats découlant de l'application de l'apprentissage automatique aux données de spectrométrie de masse. Il contient aussi les résultats d'une nouvelle méthode qui a été développée au cours de cette maîtrise.

Méthodes à noyau

Il faut par contre introduire les différents algorithmes utilisés. Un type d'algorithme qui fut utilisé au cours des travaux présentés ici est la classe des méthodes à noyau. Ces méthodes furent utilisées malgré le fait qu'elles ne sont pas parcimonieuses et ni interprétables, ce qui est une des caractéristiques recherchées dans ce projet. Par contre, les méthodes à noyau sont aisément adaptées aux problèmes auxquelles elles sont appliquées. Dans ce type de méthode, l'élément principal est la fonction de noyau (*kernel function*) qui projette les exemples d'apprentissage dans une dimensionnalité supérieure afin de pouvoir mieux séparer les classes d'exemples Boser et collab. (1992). Un exemple de ce procédé est présenté à la figure 2.1.

On cherche à avoir une fonction qui permet de faire le produit scalaire entre les vecteurs de caractéristiques des exemples sans avoir en mémoire les vecteurs de caractéristiques dans l'espace de dimensionnalité supérieure (que l'on dénote usuellement ϕ), ce qui est une caractéristique additionnelle des méthodes à noyau. On cherche en fait à obtenir directement la matrice de taille n par n, où n est le nombre d'exemples dans l'ensemble d'entraînement. On nomme cette matrice la matrice de Gram. C'est en se basant sur cette matrice que les algorithmes utilisant les méthodes à noyau peuvent faire de la classification. Cette matrice représente en fait une mesure de similarité entre les exemples, c'est-à-dire que si $k(x,x')$ est grand alors les deux exemples x et x' sont similaires et proches dans l'espace du noyau et si $k(y,y')$ est petit, ou même négatif dans certains cas, alors les exemples y et y' sont

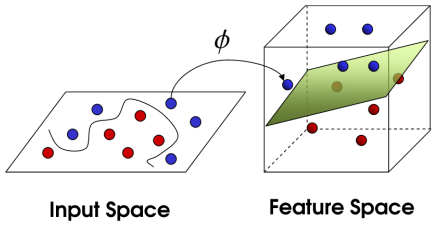

FIGURE 2.1 – **Exemple de méthode à noyau.** On projette les exemples comportant deux dimensions dans un espace de dimensionnalité supérieure, soit trois dans cet exemple, afin de pouvoir y tracer un séparateur linéaire. Ce séparateur devient non linéaire dans l'espace source des exemples.

dissimilaires et donc éloignés dans l'espace du noyau.

$$k(x,x') = \phi(x) \cdot \phi(x') \tag{2.1}$$

L'algorithme utilisant les méthodes à noyau le plus répandu est l'algorithme des machines à vecteur de support (*Support Vector Machine*, SVM) Cortes et Vapnik (1995); Chang et Lin (2011); Fan et collab. (2008). Cet algorithme cherche à tracer un séparateur linéaire entre les exemples de l'ensemble d'entraînement des différentes classes. Un séparateur linéaire est un séparateur ayant une dimension de moins que l'espace des caractéristiques des exemples. Par exemple, un séparateur linéaire est une ligne droite dans le cas où les exemples ont deux dimensions ou caractéristiques. Un séparateur linéaire est un hyperplan à deux dimensions dans le cas où les exemples ont trois dimensions pour les décrire. Par contre, l'algorithme du SVM tente de tracer le séparateur linéaire dans l'espace du noyau fourni. Comme un noyau est de plus haute dimensionnalité des données, le séparateur devient non-linéaire dans l'espace de base des exemples. On peut voir un exemple de cet effet dans les figures 2.1 et 2.2.

De plus, l'algorithme du SVM cherche à maximiser la marge du séparateur avec les exemples. La notion de marge est une notion importante pour minimiser les risques de surapprentissage. L'algorithme cherche à trouver le séparateur linéaire qui divise les deux classes dans l'espace du noyau et qui maximise la marge, soit la distance entre le séparateur et les exemples qui en sont le plus près. Le séparateur se retrouvera donc à une distance égale entre au moins un exemple de chacune des deux classes. Un exemple visuel est présenté dans la figure 2.2. Les exemples qui se retrouvent à la distance minimale du séparateur et qui définissent le séparateur sont appelés les vecteurs de support.

Un avantage que comportent le SVM et les algorithmes utilisant les noyaux est qu'il est possible de choisir différents noyaux plus adaptés au problème étudié, voire d'en mettre au point de nouveaux. Par contre, cet algorithme présente aussi un inconvénient. Alors que, dans le projet présent, les algorithmes plus parcimonieux sont privilégiés, l'algorithme du SVM n'est pas du tout parcimonieux. Le fait d'utiliser un noyau fait qu'il faut considérer toutes les caractéristiques d'un exemple et la classification va en fait dépendre de la comparaison d'un nouvel exemple avec les vecteurs de support. Malgré cet

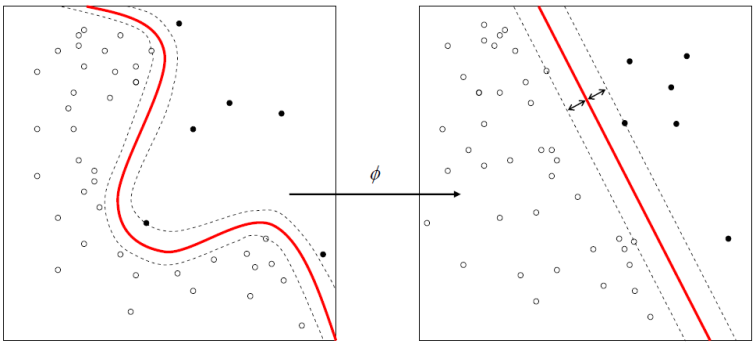

FIGURE 2.2 – **Exemple d'un SVM sur un ensemble de données.** On remarque que, sur l'image de droite, qui représente l'espace projeté du noyau, le séparateur est linéaire. Il devient non-linéaire dans l'espace des caractéristiques à gauche. On remarque aussi la marge qui place le séparateur entre des exemples des deux classes différentes dans l'image de droite.

inconvénient, cette méthode est répandue et performante, en plus d'avoir l'avantage d'être flexible. Elle a donc été incluse dans les tests de ce projet.

2.1.1 Machine à couverture d'ensembles

L'algorithme qui a été le plus privilégié pour ce projet est sans doute l'algorithme de la machine à couverture d'ensembles (*Set Covering Machine*, abbrévié SCM) Marchand et Taylor (2002). La force de cet algorithme et la raison pour laquelle il est privilégié dans ces travaux est qu'il met l'accent sur la parcimonie. Cet algorithme est très parcimonieux au niveau du nombre de caractéristiques des exemples utilisés pour la prédiction, ce qui rend la classification facilement interprétable. Le SCM cherche à classifier les exemples en utilisant des règles simples. L'exemple le plus simple de ces règles est la souche de décision, qui a été utilisée au cours de ce projet avec le SCM. Une souche de décision est une règle où l'on prend une caractéristique d'un exemple et on y applique un seuil. On obtient la forme $x_i > z$ où z est une valeur soit observée ou arbitraire. x_i dénote la caractéristique i d'un exemple x. Le SCM cherche donc la règle ayant la plus grande *utilité* sur les exemples de l'ensemble d'entraînement. L'utilité est définie comme le nombre d'exemples négatifs qui sont bien classés moins le nombre d'exemples positifs qui sont mals classés par cette règle, pondérée par un paramètre p qui est déterminé par validation croisée.

$$U_h = |Q_h| - p \cdot |R_h|,$$

Où U_h est l'utilité de la règle h, Q_h est le nombre d'exemples négatifs bien classifiés et R_h est le nombre d'exemples positifs mal classés. Lorsque la règle optimale a été trouvée, le SCM va ensuite retirer tous les exemples négatifs qui sont bien classés et les exemples positifs mal classés de l'ensemble d'entraînement, avant de faire une autre itération. L'algorithme peut s'arrêter sous deux conditions : soit il n'y a plus d'exemples négatifs à classifier, soit l'algorithme atteint le maximum de règles permises, qui

est aussi un paramètre déterminé par validation croisée. Le fait d'avoir un nombre maximal de règles permises est aussi une manière de réguler le surapprentissage.

Finalement, les règles retenues par l'algorithme sont combinées de deux façons possibles. La première façon est par la conjonction, c.-à-d., le prédicteur ainsi produit est le prédicteur booléen consistant en la conjonction de toutes les règles retenues. La seconde façon consiste à produire une disjonction de toutes ces règles. Une disjonction correspond à un ou exclusif en logique. Par exemple, afin de classifier un exemple par un ensemble de règles disjointes, cet exemples doit répondre à une et une seule de ces règles.

L'algorithme du SCM a donc un avantage clair pour le projet décrit ici qui est sa parcimonie et son interprétabilité. La parcimonie du SCM résulte en un très petit nombre de caractéristiques, soit des pics dans un spectre de masse dans notre cas, qui établit un lien prédictif entre les classes. De plus, l'algorithme fournit une organisation logique entre les caractéristiques utilisées par la classification. Effectivement, les deux types d'organisations logiques des règles du SCM fonctionnent très bien dans un contexte biologique. Une conjonction de caractéristiques indique que plusieurs molécules sont nécessaires afin de faire la différence entre les deux classes, ce qui est un cas courant en biologie. Le cas d'une disjonction est aussi très compatible. De trouver un classificateur ayant une disjonction entre deux pics par exemple pourrait indiquer que deux voies métaboliques distinctes sont impliquées dans la différence entre les classes.

2.1.2 Arbres de décision

Un autre algorithme utilisé dans ce projet est l'algorithme des arbres de décision (*decision trees*) Breiman et collab. (1983). Plus précisément, les arbres de classification sont utilisés ici. L'algorithme d'apprentissage par arbre de décision fonctionne en séparant les données par une série de règles du même type que celles utilisées pour le SCM. On peut en fait voir les souches de décisions, mentionnées dans le cadre de l'algorithme du SCM, comme des arbres de décision avec une seule règle, soit une profondeur de 1. Un exemple d'un de ces arbres est présenté à la figure 2.3. L'algorithme des arbres de décision va ainsi faire un arbre de la forme suivante. On commence à la racine de l'arbre. On a une première décision basée sur une caractéristique des exemples, de la forme $x_i > z$. Selon le résultat de la comparaison, on suit l'arc correspondant dans l'arbre. On peut ensuite arriver à un nouveau noeud. Si c'est le cas, on répète le processus avec la règle encodée dans ce noeud. Si l'on arrive à une feuille, c.-à-d. un noeud terminal sans arc sortant, cette feuille indique dans quelle classe est l'exemple considéré.

Comme tous les autres algorithmes, il y a un risque de surapprentissage avec cet algorithme. On peut facilement former un arbre avec trop de noeuds, trop profond et qui ne généralisera pas à d'autres exemples. Pour pallier ce problème, on utilise la validation croisée pour déterminer les meilleurs paramètres pour l'algorithme. Les paramètres disponibles dans l'implémentation de cet algorithme qui fut utilisé sont la profondeur maximale de l'arbre et le nombre d'exemples minimum à séparer par noeud

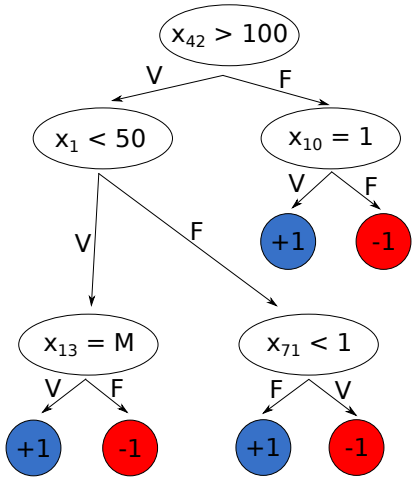

FIGURE 2.3 – **Exemple de la structure d'un arbre de décision.** Les noeuds représentent des règles de décision. Selon si elle est respectée ou pas dans un exemple, on suit le chemin vers le prochain noeud. Les noeuds colorés sont les feuilles, qui représentent la classe majoritaire des exemples se retrouvant dans cette feuille.

Pedregosa et collab. (2011). On utilise donc la validation croisée afin de trouver une combinaison de ces paramètres qui sera performante sur l'ensemble d'entraînement et sera capable de généraliser.

De manière semblable au SCM, on peut voir l'arbre de décision comme un mélange plus complexe de disjonctions et conjonctions de règles. L'algorithme des arbres de décision est aussi un algorithme avantageux pour le projet étant donné qu'il est aussi très parcimonieux et interprétable, surtout comparé aux méthodes à noyaux ou aux réseaux de neurones. L'arbre de décision a par contre tendance à être moins parcimonieux que le SCM. De plus, sa capacité à organiser les règles de décision de manière plus complexe peut l'aider dans certains cas ou rendre le classificateur moins interprétable.

2.1.3 Forêt d'arbres décisionnels

Un dernier algorithme d'apprentissage utilisé au courant de ce projet est celui des forêts d'arbres décisionnels, parfois appelés forêts d'arbres aléatoires, ou *random forest* Breiman (2001). Cet algorithme fait partie de la classe d'algorithmes d'apprentissage que l'on nomme les méthodes d'ensemble. Les méthodes d'ensemble sont des méthodes qui tentent d'obtenir une meilleure performance en prédiction en utilisant un vote de majorité de plusieurs autres algorithmes. Dans le cas des forêts d'arbres décisionnels, un vote de majorité est fait sur un ensemble d'arbres de décisions.

L'algorithme génère un nombre d'arbres de décisions. Les arbres sont générés sur un sous-ensemble d'exemples, tirés par méthode *bootstrap*, c'est-à-dire en pigeant aléatoirement des exemples avec

remise, dans l'ensemble d'entraînement. Ensuite, au moment de calculer les règles à chaque noeud, l'algorithme va prendre un sous-ensemble aléatoire des caractéristiques des exemples et l'utiliser pour choisir la règle. On entraîne ainsi un nombre d'estimateurs spécifié. Pour faire une prédiction, on montre l'exemple à classifier à chacun des arbres estimateurs, qui vont chacun prédire une classe. On fait ensuite un vote de majorité de ces classificateurs avec un poids uniforme, c'est-à-dire que chacun des votants a un poids équivalent aux autres.

Les forêts d'arbres décisionnels sont utilisées dans ce projet pour les raisons suivantes. Une première est que c'est un algorithme populaire et généralement performant. Il est aussi très rapide à entraîner. Cet algorithme a aussi une tendance assez faible à surapprendre et son design est fait pour diminuer le surapprentissage des arbres le composant. Un inconvénient par rapport aux arbres de décision et au SCM est que cet algorithme n'est pas très parcimonieux et n'est pas du tout interprétable.

2.2 Le noyau à boîtes chevauchantes pour l'algorithme du SVM

2.2.1 Motivation et preuve de noyau

Au cours du projet, l'idée d'un nouveau noyau adapté aux défis du travail avec des données de spectromètre de masse est revenue à plusieurs reprises. Finalement, une idée s'est imposée. Elle est inspirée d'une méthode déjà existante en traitement de données de spectrométrie de masse. Cette technique se nomme le *binning*. Simplement, cette méthode consiste à compenser pour le désalignement des pics de plusieurs spectres en regroupant plusieurs pics dans une distance spécifiée en un seul, que l'on nomme un *bin*, ou une boîte. Un exemple est présenté à la figure 2.4. Une manière simple de l'implémenter est d'arrondir d'une ou deux décimales les masses en uma mesurées par un spectromètre et de sommer les pics qui se retrouvent à la même masse.

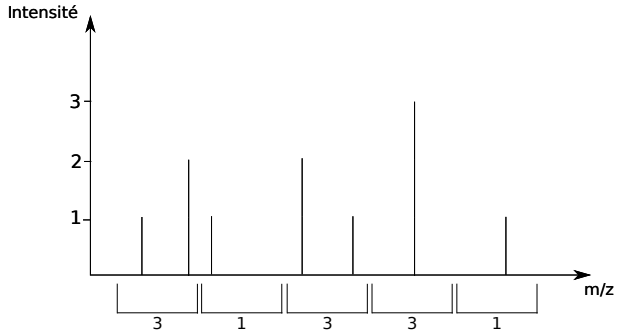

FIGURE 2.4 – **Exemple de la méthode du binning.** Si l'on considère un spectre donné, on regroupe et additionne les pics retrouvés dans un *bin*, ou une boîte.

Certains problèmes sont présents dans cette méthode. Un premier est la possibilité qu'un pic homologue entre deux spectres se retrouve dans des bins différents. En reprenant l'exemple du paragraphe précédent, on peut imaginer un pic x_1 de masse 201.1234 Da qui ait une déviation dans un autre spectre x_2 avec une masse de 201.1239 Da. Le fait d'arrondir pourrait amener la masse de x_1 à 201.123 Da et

celle de x_2 à 201.124 Da. Un autre problème avec l'approche par arrondissement est que les variances aléatoires d'un spectre sont relatives à la masse et donc on aura une déviation plus grande avec de plus grandes masses. Cet inconvénient peut être pallié si l'on imagine les bins comme étant d'une taille fixe en ppm, mais il restera tout de même le fait que des pics peuvent se retrouver d'un bord et de l'autre de la frontière entre deux bins.

C'est donc dans cette optique que nous avons eu l'idée de mettre au point un nouveau noyau pour l'algorithme du SVM basé sur l'idée des bins. Pour régler le problème des pics sur la frontière entre les bins, un nouveau paramètre de chevauchement a été introduit. Dans ce noyau, on représente donc un spectre par une série de boîtes d'une taille constante en ppm, donc dont la taille va augmenter à travers le spectre lorsque la masse augmente, et qui se chevauchent partiellement. Un exemple de cette représentation est montré à la figure 2.5.

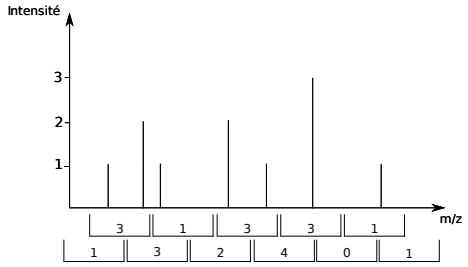

FIGURE 2.5 – **Exemple de la méthode par boîtes chevauchantes (*overlapping bins*).** On considère ici le même spectre qu'à la figure précédente. On considère aussi des boîtes de la même taille, et un chevauchement de 50%. On remarque qu'on a les mêmes boîtes qu'à la figure précédente. En plus, on regroupe dans les nouvelles boîtes des pics qui étaient auparavant séparés par la frontière d'une boîte, malgré une petite distance entre ces pics.

Afin d'être utilisable avec les méthodes à noyau tel que l'algorithme du SVM, il faut faire la preuve que le nouveau noyau, baptisé le noyau à boîtes chevauchantes (*Overlapping Bin Kernel*, abbrévié OBK), est effectivement un noyau. La caractéristique principale des noyaux est présentée à l'équation 2.1. Dans le cas de l'OBK, la preuve est simple. Avec les paramètres de taille de bin et de chevauchement, on peut décrire explicitement le vecteur de caractéristique dans le noyau, puisqu'on connaît le point de départ (la masse la plus petite que l'on a acquise dans l'ensemble des spectres), la taille de chaque boîte et le chevauchement entre chaque boîte. Ainsi, si l'on sait exactement la forme du vecteur $\phi(x)$ pour tout x et donc que $\phi(x')$ aura la même forme pour tout x', il s'ensuit directement que pour $k(x,x') = \phi(x) \cdot \phi(x')$, la mesure de similarité k est donc bien une fonction noyau (c.-à-d., une fonction représentant un produit scalaire dans l'espace vectoriel des caractéristiques). Il peut donc être utilisé par l'algorithme du SVM.

Il est à noter qu'une conséquence de l'utilisation de ce noyau est qu'il n'est plus nécessaire d'appliquer l'algorithme d'alignement décrit au chapitre précédent. Il est aussi théoriquement possible que ce noyau rende redondante l'application de masses de verrouillages, réelles ou virtuelles. Par contre, il faut utiliser des tailles de boîtes beaucoup plus grandes afin de compenser pour les déviations corrigées

par masses de verrouillage. Cela entraîne un risque que la taille de boîte soit trop grande et plus assez spécifique pour servir de bonne fonction de comparaison. En général, la taille des boîtes et le chevauchement seront des hyperparamètres qui seront choisis par validation croisée afin d'éviter un surapprentissage.

Il est aussi à noter que, malgré que le fait que les objectifs de ce projet favorisent l'application d'algorithmes parcimonieux et dont le modèle est interprétable, cette méthode est utilisée en conjonction avec l'algorithme du SVM et n'est donc pas du tout parcimonieuse ou interprétable. Par contre, l'objectif plus général du projet est de classifier des échantillons de produits sanguins selon leur spectre de masse. Il semble alors qu'il ne faut pas laisser passer une opportunité de mettre au point une nouvelle méthode adaptée aux données et donc prometteuse pour la classification, même si elle ne répond pas à toutes les caractéristiques privilégiées.

2.2.2 Présentation de l'algorithme

De premiers tests ont été pratiqués avec une première version très directe et peu optimisée de l'algorithme, afin de vérifier si l'approche avait un bon potentiel. L'algorithme 6 présente une partie commune à la première implémentation et aux versions subséquentes, c'est-à-dire la construction de la matrice de Gram entre les différents exemples. Cette matrice est la matrice de comparaison entre les échantillons, nécessaire et utilisée par l'algorithme du SVM.

Algorithm 6 Construction de la matrice de Gram

Require: Un ensemble de m spectres
  Créer une matrice de taille $m \times m$, la matrice de Gram
  for $i = 0$ to m **do**
    for $j = i$ to m **do**
      Comparer le spectre i et le spectre j avec le noyau à boîtes chevauchantes
      Stocker la valeur de noyau retournée aux indices $[i,j]$ et $[j,i]$
    end for
  end for
  return La matrice de Gram

La méthode de construction de matrices de Gram présentée à l'algorithme 6 est standard. Il faut construire une matrice de comparaison entre toutes les paires d'exemples, où chaque exemple est un spectre. On construit donc un tableau de taille $m \times m$ pour stocker les valeurs du noyau lorsqu'on a m spectres. On parcourt ensuite les exemples, en faisant toutes les comparaisons. Notez qu'Algorithme 6 est sous-optimal, le fait de faire toutes les comparaisons n'est pas la meilleure façon de calculer le noyau OBK. Nous allons d'abord proposer une façon légèrement plus efficace (et surtout mieux détaillée) de faire ce calcul (Algorithme 7) que nous améliorerons à la sous-section suivante. En fait, pour accélérer la méthode, on ne calcule que les valeurs pour la moitié supérieure du tableau, puisque la comparaison entre un exemple x et y sera la même qu'entre y et x. Cela permet d'accélérer cette étape d'un facteur près de deux. L'algorithme 7 va donc présenter la première implémentation du noyau.

Algorithm 7 Première implémentation du noyau à boîtes chevauchantes

Require: Deux spectres, s_1 et s_2
Require: Un paramètre de taille de boîtes, en ppm, σ
Require: Un paramètre de chevauchement, ω
Require: Une masse de départ et une masse finale, μ_d et μ_f
▷ La masse de départ est déterminée par la masse minimale de l'acquisition des spectres. La masse finale doit être déterminée sur l'ensemble des spectres utilisés. Il nous faut donc la masse de plus grande taille observée dans tous les spectres.
for $i = 1$ to 2 **do**
Créer un vecteur contenant les valeurs de boîtes v_i
Trouver les pics de s_i dans la boîte $b_1 = [\mu_d, \mu_d + \mu_d \cdot \sigma]$ et ajouter la somme de leurs intensités au vecteur v_i
while $\mu_f \notin b_j$ **do**
$b_j = [\mu_j, \mu_j + \mu_j \cdot \sigma]$ où $\mu_j = \mu_{i-1} + (\mu_{i-1} \cdot \sigma) \cdot \omega$
Trouver les pics de s_i dont la masse est dans la boîte b_j et ajouter la somme de leurs intensités au vecteur v_i
end while
end for ▷ On a à ce point les vecteurs des valeurs des boîtes des deux spectres
▷ La valeur à tout indice donné j dans chacune de ces listes équivaut à la même boîte dans les deux spectres.
return $v_1 \cdot v_2$

Décrivons ici l'algorithme 7 en plus de détail. Une première spécification à apporter est au niveau des masses de départ et de fin des boîtes. Il faut que, dans toutes les comparaisons de spectres, on ait exactement les mêmes boîtes définies. Étant donné que les boîtes correspondent aux éléments du vecteur $\phi(x)$, il nous faut absolument que tous les ϕ soient identiques au niveau de leur composition. Si l'on définissait les boîtes selon les plus petites masses et plus grandes masses observées dans les deux exemples d'une comparaison, on aurait des vecteurs différents à chaque comparaison et donc nous n'aurions pas un noyau.

Ensuite, on crée la première boîte et l'on somme tout pic se trouvant dans cette boîte. Si aucun ne s'y trouve, on confère une valeur de 0 à cette boîte. On cherche les pics selon une recherche binaire où les bornes sont les masses limites de la boîte. Cela fonctionne puisque nous gardons la liste des pics ordonnée par leurs masses. De plus, la recherche binaire s'opère de manière efficace, avec un pire cas de $O(\log n)$ où n est le nombre de pics dans le spectre. Une fois le contenu de la boîte sommé, on stocke cette somme en mémoire.

Après qu'on ait terminé la première boîte, on avance à la prochaine. On peut calculer la masse de départ de la nouvelle boîte à partir de la taille et du chevauchement. On répète alors la même procédure qu'à la première boîte à cette nouvelle boîte et toutes les boîtes subséquentes. Le critère d'arrêt est que l'on atteigne une boîte qui inclut dans sa fenêtre la masse finale générale aux exemples.

Lorsque ce procédé a été fait sur les deux spectres à comparer, on fait le produit scalaire entre les deux vecteurs de caractéristiques. On itère sur chacune des caractéristiques des vecteurs et on multiplie

chacune des boîtes homologues ensemble avant de sommer tous les résultats. On a ainsi le produit scalaire entre les vecteurs. Ce produit est la mesure de similarité entre les deux spectres.

Cette implémentation contient plusieurs inconvénients. Le premier est d'ailleurs que l'on explicite le vecteur ϕ des exemples en mémoire. Cela requiert une grande quantité de mémoire vive et de temps de calcul afin d'ajouter des éléments et en temps d'accès. De plus, la complexité de l'algorithme dépend fortement, dans cette implémentation, du nombre de boîtes nécessaire pour couvrir les spectres, donc essentiellement des paramètres de taille et de chevauchement. Un autre problème de cette implémentation est qu'il faut constamment aller chercher les pics dans le spectre. Même avec un algorithme de recherche efficace tel que la recherche binaire, une recherche est nécessaire pour chaque boîte. Plusieurs seront d'ailleurs inutile si aucun pic ne se trouve dans la boîte cible.

Plusieurs améliorations doivent donc être apportées à l'algorithme afin d'accélérer son fonctionnement, puisque les premiers tests se sont avérés prometteurs. Une première amélioration était au niveau de l'implicitation du vecteur de caractéristiques. Au lieu de créer un vecteur avec b entrées où b est le nombre de boîtes, on crée un dictionnaire vide. Si une boîte ne contient aucun pic, on ne crée pas d'entrée dans le dictionnaire. Si une boîte contient des pics, on somme alors leur intensité et on stocke la valeur dans le dictionnaire, avec la clé d'accès étant les bornes de la boîte. Cette amélioration est avantageuse au niveau de la mémoire utilisée et du temps de calcul pour la gestion de la mémoire. Étant donnée que les dictionnaires en Python sont des tables de hashage [1], on a alors un temps d'accès constant et un temps d'insertion constant en moyenne (en l'absence de collisions). Cette amélioration est aussi bénéfique pour l'étape finale du produit scalaire. En effet, vu le fait qu'un grand nombre de boîtes auront généralement une valeur de 0, nous n'avons pas à nous soucier de leur contribution au produit scalaire. De plus, il faut que les deux boîtes homologues des deux spectres comparés soient non-nulles pour que leur partie du produit scalaire puisse contribuer. On itère alors sur les clés d'un des dictionnaires. Si cette clé n'est pas présente dans le dictionnaire de l'autre spectre, alors on sait que le second spectre n'a pas de pics dans la boîte homologue.

Une autre amélioration à l'algorithme est liée au fait qu'il est possible de construire des équations permettant de calculer dans quelles boîtes se retrouvera un pic d'une masse donnée, connaissant le point de départ des boîtes, la taille des boîtes et le chevauchement. Cette amélioration va donc permettre de rendre la complexité de l'algorithme du noyau dépendant plus fortement du nombre de pics dans un spectre, qui est typiquement inférieur au nombre de boîtes, tout dépendant des paramètres de taille t de chevauchement. L'algorithme devient alors O(n) où n est le nombre de pic dans le spectre à comparer.

1. Une table ed hashage est une structure de données de base. Sa caratéristique principale qui nous intéresse pour cette application est son temps d'accès optimal ainsi que son temps d'insertion très rapide en moyenne. Une table de hashage est représentée en mémoire par un long vecteur, et on "hash" la clé par une fonction afin d'obtenir l'index de la case où se retrouvera l'entrée. Pour plus d'information, consultez tout manuel de base de structures de données.

Amélioration mathématique du noyau à boîtes chevauchantes

Soit $\lambda_1 \in \mathbb{R}$, la masse la plus petite que l'on considère. On considère que chaque boîte est représentée par $(\lambda_i, \varepsilon_i)$ où $\lambda_i, \varepsilon_i \in \mathbb{R}$. Elles sont ordonnées en ordre crossant de λ. On définit aussi k comme étant $k = \frac{\sigma}{10^6}$, où σ est la taille en ppm de la boîte. On a aussi δ qui est défini comme $\delta = 1 - \omega$, où ω est le chevauchement entre les boîtes désiré.

$$\begin{aligned} \lambda_i &= \lambda_{i-1} + k \cdot \delta \\ &= \lambda_{i-1}(1 + k\delta) \end{aligned}$$

On pose alors que $c = 1 + k\delta$. On obtient alors que :

$$\begin{aligned} \lambda_i = \lambda_{i-1} \cdot c \quad \forall i \in \{2,3,4,...\} \subseteq \mathbb{N} \\ \lambda_i = \lambda_1 \cdot c^{(i-1)} \end{aligned} \tag{2.2}$$

On retrouve donc l'équation (2.2) qui détermine simplement la masse de départ λ_i de toute boîte i. On va donc maintenant avoir un moyen de calculer la fin de chaque boîte.

$$\begin{aligned} \varepsilon_i &= \lambda_i + k \cdot \lambda_i \\ &= \lambda_i(1 + k) \end{aligned}$$

$$\varepsilon_i = \lambda_1 \cdot c^{i-1} \cdot (1 + k) \quad \forall i \in \{1,2,3,...\} \subseteq \mathbb{N} \tag{2.3}$$

L'équation (2.3) nous indique alors la borne supérieure en masse de chaque boîte i. Nous cherchons par contre un moyen de calculer dans quelle boîte(s) sera un pic étant donné sa masse. Il nous faut donc supposer que l'on a une masse μ d'un pic quelconque et que $\mu \in \mathbb{R}$. Alors, on cherche à trouver les boîtes où μ peut se retrouver. On commence par déterminer dans quelles boîtes μ est supérieur à λ_i.

$$\begin{aligned} \mu &\geq \lambda_i \\ &\geq \lambda_1 \cdot c^{i-1} \end{aligned}$$

$$\begin{aligned} \log_c(\mu) &\geq \log_c[\lambda_1 \cdot c^{(i-1)}] \\ &\geq \log_c(\lambda_1) + \log_c(c^{(i-1)}) \\ &\geq \log_c(\lambda_1) + (i-1) \end{aligned}$$

$$\log_c(\mu) - \log_c(\lambda_1) + 1 \geq i \tag{2.4}$$

On fait le même exercice avec les masses finales, afin d'avoir les indices i des boîtes dont la masse finale est plus grande que la masse μ.

$$\begin{aligned}\mu &\leq \varepsilon_i \\ &\leq \lambda_1 \cdot c^{(i-1)} \cdot (1+k)\end{aligned}$$

$$\begin{aligned}\log_c(\mu) &\leq \log_c[\lambda_1 \cdot c^{(i-1)} \cdot (1+k)] \\ &\leq \log_c(\lambda_1) + \log_c(c^{(i-1)}) + \log_c(1+k) \\ &\leq \log_c(\lambda_1) + i - 1 + \log_c(1+k)\end{aligned}$$

$$\log_c(\mu) - \log_c(\lambda_1) - \log_c(1+k) + 1 \leq i \tag{2.5}$$

Avec les équations (2.4) et (2.5), on peut alors déterminer les indices i de toutes les boîtes dont la masse μ fera partie.

$$1 + \log_c(\mu) - \log_c(\lambda_1) - \log_c(1+k) \leq i \leq 1 + \log_c(\mu) - \log_c(\lambda_1) \tag{2.6}$$

On peut simplifier l'équation (2.6) à la forme suivante :

$$\log_c(\mu) + b_{low} \leq i \leq \log_c(\mu) + b_{high} \tag{2.7}$$

On définit les constantes b_{low} et b_{high} à l'équation (2.7) aux formes suivantes :

$$b_{low} = 1 - \log_c(\lambda_1) - \log_c(1+k)$$

$$b_{high} = 1 - \log_c(\lambda_1)$$

On considère que b_{low} et b_{high} sont des constantes puisqu'ils dépendent simplement de la masse de départ, de la taille des boîtes et du chevauchement. Tous ces paramètres sont fixés au moment de calculer le noyau et ne varient pas pour toute la matrice de Gram. Il est donc possible de précalculer les valeurs de c, k, λ_1 et donc de b_{low} et b_{high} au début de l'algorithme.

L'application de cette forme de recherche améliore grandement l'efficacité de l'algorithme, lorsqu'appliquée sur des spectres. Si l'on prend b comme étant le nombre de boîtes nécessaire pour couvrir le spectre et n comme étant le nombre de pics par spectre, la première implémentation de l'algorithme avait une complexité de l'ordre de $O(b \cdot \log(n))$ pour faire la comparaison entre deux spectres. Avec la

méthode décrite ci-dessus, on a une complexité de l'ordre de O(n) par comparaison. Étant donné que le nombre de boîtes peut devenir beaucoup plus grand que le nombre de pics par spectre, on a un très bon gain de performance.

La nouvelle implémentation de l'algorithme, décrite à l'algorithme 8, inclut les changements décrits plus haut.

Algorithm 8 Implémentation améliorée du noyau à boîtes chevauchantes

Require: Deux spectres, s_1 et s_2
Require: Un paramètre de taille de boîtes, en ppm, σ
Require: Un paramètre de chevauchement, ω
Require: Une masse de départ, λ_1
  Calculer les constantes c, b_{low}, b_{high}
  for $i = 1$ to 2 **do**
    Créer le vecteur v_i
    for $j = 1$ to n où $n = |s_i|$ **do**
      Calculer dans quelle(s) boîte(s) se retrouve le pic $p_i j$ à partir de sa masse $\mu_i j$
      Ajouter l'intensité $\iota_i j$ du pic dans chacune des boîtes où il se retrouve
    end for
  end for
  return $v_1 \cdot v_2$

Décrivons en détail les changements dans l'algorithme 8. Un premier changement notoire est qu'il n'y a pas de masse finale prédéfinie. Étant donné la nouvelle méthode mathématique, il n'y a pas nécessairement de boîte maximale à définir. Comme mentionné plus haut, on remarque aussi que l'on itère sur les pics du spectre lorsque l'on calcule le contenu des différentes boîtes au lieu d'itérer sur les boîtes mêmes. En utilisant l'équation (2.7), on obtient simplement l'index (ex : la i^{eme} boîte, $\forall i \in \mathbb{N}$) de toutes les boîtes dans lequel le pic se retrouve. Cela donne lieu d'ailleurs à un second changement. On dénote les boîtes dans le dictionnaire avec leur index au lieu de leurs masses frontière. Il serait aisé de calculer les masses frontières à partir des équations (2.2) et (2.3), mais cela n'est pas nécessaire au fonctionnement du noyau et ne nous fournirais aucune information supplémentaire.

Avec ces changements, on remarque une amélioration marquée de la performance. Alors que la première implémentation prenait un temps de calcul d'environs 3h30 (environ 210 minutes) afin de construire la matrice de Gram de 96 exemples, la seconde implémentation peut comparer 185 exemples en moins de 15 minutes [2]. La performance de la nouvelle implémentation est aussi beaucoup moins dépendante des paramètres de taille et de chevauchement que la première version, même si ces paramètres ont encore une influence. Effectivement, le paramètre de taille va faire augmenter le temps de calcul à mesure que la taille diminue, car il y aura plus de boîtes à mettre à jour lors du calcul du vecteur de caractéristiques et il y aura plus de caractéristiques sur lesquelles itérer au moment du produit scalaire. On remarque aussi que le temps de calcule augmente plus il y a de chevauchement.

2. Ces temps de calcul correspondent au temps nécessaire sur un noeud de calcul du supercalculateur Colosse, ayant un processeur Intel Xeon X5560 Nehalem de 2.8 GHz.

De manière comparable à la taille, plus il y a de chevauchement, plus il faut mettre de boîtes à jour par pic. Similairement, le produit scalaire sera plus long à faire puisque plus de boîtes seront non nul. Il faut aussi noter que le chevauchement fait rapidement augmenter le nombre de boîtes nécessaire afin de couvrir le spectre.

Une détail est à noter à propos de l'utilisation de l'algorithme et de ses paramètres. Il semble plus équitable d'utiliser des valeurs répondant à la forme suivante pour ce paramètre :

$$\omega = 1 - \frac{1}{\alpha} \quad \forall \alpha \in \{1,2,3,...\} \subseteq \mathbb{N}$$

La raison est la suivante. On considère par exemple le cas où l'on a un chevauchement de 50%. Cela répond au critère puisque $50\% = 0.5 = 1 - \frac{1}{2}$. On peut voir que, si chaque nouvelle boîte commence à la moitié de la boîte précédente, alors tout endroit dans le spectre couvert sera simultanément dans deux boîtes (à l'exception de la première moitié de la première boîte et deuxième moitié de la deuxième boîte). Un exemple de cela est montré à la figure 2.6. Donc, tout pic sera représenté deux fois dans le spectre. Il en est de même pour toute autre valeur α qui fait partie de l'ensemble $\{1,2,3,...\} \subseteq \mathbb{N}$, qui aura chaque pic représenté dans α boîtes à tout moment.

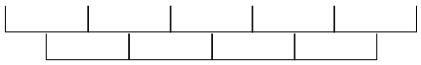

FIGURE 2.6 – **Exemple de boîtes chevauchantes dont le paramètre de chevauchement est à 50%.** On remarque que chaque endroit du spectre est inclus dans les limites de deux boîtes à tout moment, sauf aux extrémités.

Prenons par contre un exemple où le chevauchement a une valeur de 33%. On peut voir un exemple de l'arrangement des boîtes dans ce cas est présenté à la figure 2.7. Si l'on a une boîte de départ et que l'on ajoute seulement la boîte suivante avec 33% de chevauchement alors on peut remarquer dans la figure que certaines zones seront couvertes par deux boîtes, mais d'autres zones auront seulement une boîte pour les couvrir (dans leur centre). Il semble donc que d'avoir une couverture inégale à travers tout le spectre pourrait induire des biais dans la valeur du noyau. Il semble donc préférable d'avoir un paramètre de chevauchement répondant au critère décrit ci-dessus, soit d'utiliser des valeurs de chevauchement de la forme $\omega = 1 - \frac{1}{\alpha}$ pour $\alpha \in \{1,2,3,...\}$.

2.2.3 Résultats expérimentaux

Jeu de donnée Clomiphène-Acétaminophène Le noyau a été testé sur un ensemble de données généré sur des échantillons de plasma. Cet ensemble de données contenait deux plaques de 96 échantillons. La première de ces deux plaques a eu ses échantillons mis en communs et reséparés en 96 aliquots. Ces échantillons seront alors uniformes. Les échantillons de la deuxième plaque n'ont pas

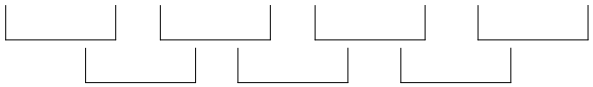

FIGURE 2.7 – **Exemple de boîtes chevauchantes dont le paramètre de chevauchement est à 33%.** On remarque que, contrairement à la figure précédente, la couverture n'est pas uniforme à travers le spectre et certaines zones ne sont couvertes que par une seule boîte.

été mis en commun, mais gardés individuels. Sur le tiers des échantillons de chaque plaque (32 par plaque), les échantillons (ou les aliquots) ont été gardés tels quels. Dans un second tiers des échantillons, du clomiphène a été ajouté aux échantillons. Dans le dernier tiers, une solution de comprimé d'acétaminophène dilué a été ajoutée. On aura donc, par plaque, 32 échantillons sans aucun ajout, 32 échantillons avec du clomiphène ajouté et 32 échantillons avec de l'acétaminophène ajouté.

Ce jeu de données permettra donc de tester les algorithmes d'apprentissage de différentes manières. Étant donné que la première plaque consiste d'échantillons uniformes après la mise en commun, alors les seules différences entre les spectres attendus sont les pics associés aux ajouts. Pour les échantillons de la seconde plaque, on a à la fois les variations interindividuelles du plasma et les molécules ajoutées. Ainsi, on peut tester si l'algorithme est capable d'apprendre sur des échantillons sans variations et de prédire sur les échantillons avec variation ou vice-versa. De plus, les deux molécules ajoutées vont induire différents changements dans les spectres. L'ajout de clomiphène va ajouter un seul pic de haute intensité dans une région du spectre qui n'a normalement pas ou peu de pics. L'ajout de la dilution de comprimés d'acétaminophène de son côté va ajouter plusieurs pics dans le spectre, comme les comprimés ne sont pas composés d'une seule molécule. De plus, il est possible que les donneurs aient consommé de l'acétaminophène avant leurs dons. Étant donné que l'on ajoute une très forte quantité d'acétaminophène, plus forte que ce qu'on retrouverait dans du plasma humain, on va alors observer une forte hausse d'intensité dans des pics avec cet ajout. Il est par contre possible que ce pic soit déjà présent dans des spectres sans ajout, mais à plus faible intensité.

Les échantillons (ou aliquots de la plaque mise en commun) ont premièrement subi une extraction à l'ACN :MeOH. La solution d'extraction était composée de 75% d'ACN et 25% de méthanol. On fait l'extraction avec 10 μL d'échantillon et 90 μL de solution d'extraction. Les solutions subissent ensuite une sonication de 5 minutes. On fait ensuite une centrifugation des échantillons pour 5 minutes à 5000 RPM. On conserve alors le surnageant de l'extraction, laissant les protéines précipitées. On ajoute à ce point le clomiphène ou la solution d'acétaminophène dans les échantillons. On fait finalement une dilution 1 :10 des échantillons afin d'éviter la suppression ionique durant l'acquisition des spectres.

Les spectres ont été acquis avec une source LDTD et un spectromètre Synapt G2 de Waters Corporation. Le patron laser du LDTD était d'une durée de 11 secondes. Le laser reste fermé 2 secondes au

départ. Il s'ouvre ensuite et monte à 65% de sa puissance maximale sur un gradient de 6 secondes. Le laser maintient ensuite son intensité pendant 3 secondes avant de se fermer. Les spectres ont été acquis en mode MS^e sur le spectromètre, qui est un mode d'acquisition data-indépendant. Dans ce mode, on acquiert deux spectres sur l'échantillon. Le premier est une fonction de basse énergie, où l'on n'ajoute aucune énergie de collision dans la cellule de collision du spectromètre. On a donc le spectre contenant les molécules ionisées, mais pas ou peu fragmentées. On acquiert ensuite une fonction de haute énergie où l'on rajoute une énergie de collision, qui causera la fragmentation des molécules ionisées. On acquiert donc un second spectre où l'on mesure les ions filles du premier spectre. Le spectromètre était en mode *high resolution* pour l'acquisition, donnant une précision accrue sur les masses des pics contre une certaine diminution de la sensibilité.

Il est à noter que 7 des échantillons de la plaque mis en communs ont été exclus vu des problèmes expérimentaux et que leurs spectres n'avaient pas été acquis convenablement. On a donc 96 échantillons individuels et 89 exemples de la plaque d'échantillons mis en communs.

Les spectres ont ensuite subi un prétraitement. Ils ont d'abord été prétraités par le logiciel MassLynx de Waters Corporation afin de faire la centroïdation des pics. Ils ont ensuite subi une correction par masses de verrouillage virtuelles (*virtual lock mass*, ou VLM). Les pics ont ensuite subi une filtration par un seuil d'intensité fixé à 500. Les spectres n'ont pas été alignés, car l'on veut vérifier que le noyau OBK soit capable de compenser pour le désalignement aléatoire. Seuls les spectres de la fonction de basse énergie sont utilisés dans la classification.

Les résultats ont été générés avec le noyau OBK et avec l'algorithme du SVM afin de construire un prédicteur basé sur les valeurs de comparaison du noyau. Les hyperparamètres à valider dans ce cas sont donc la valeur C de l'algorithme du SVM, la taille des boîtes ainsi que le chevauchement entre les boîtes. Afin de mieux constater l'influence des paramètres du noyau, les tableaux de résultats présentés ci-dessous montreront une grille des possibilités explorées sur le plan de taille de boîte et de chevauchement entre les boîtes alors que les scores indiqués correspondront aux résultats du meilleur paramètre C, déterminé par validation croisée à 5 plis.

Étant donné que les exemples comportent trois classes, soit les échantillons sans ajout, les échantillons avec clomiphène ajouté et les échantillons avec acétaminophène ajouté, le problème de classification a été séparé en deux. On considérera que soit les échantillons avec clomiphène ou bien les échantillons avec acétaminophène seront la classe positive (+1) et que les échantillons sans ajout et l'autre type d'échantillon avec ajout seront la classe négative (-1). On cherchera donc à classifier acétaminophène (+1) vs sans ajout et clomiphène (-1) ou bien clomiphène (+1) vs sans ajout et acétaminophène (-1). On peut alors toujours considérer le problème comme un problème de classification binaire.

Pour les premiers résultats présentés dans les prochaines tables, l'ensemble d'entraînement était les 89 échantillons provenant de la plaque mise en commun. On s'attend donc à ce que les seules différences entre les spectres soient dues aux ajouts et aux variations de mesure. L'ensemble de test était composé des 96 échantillons provenant de la seconde plaque, restés individuels. On cherche donc à constater si

l'on peut apprendre sur les échantillons avec aucune variation interindividuelle et quand même prédire sur les échantillons individuels, c'est-à-dire si on peut arriver à de bonnes performances prédictives.

Les paramètres utilisés pour ces tests sont les suivants. Les tailles de boîtes testées sont de $\{100, 50, 20, 10, 5, 2\}$ ppm. Les valeurs de chevauchement possibles sont de $\{0\%, 50\%, 66\%, 75\%, 80\%\}$, correspondant à la forme $\omega = 1 - \frac{1}{n}$ où $n \in \{1, 2, 3, 4, 5\}$. Les valeurs du paramètre C du SVM se retrouvaient dans un espace logarithmique de 30 valeurs dont les limites sont de 10^{-20} à 10^{5}. Les résultats indiqués sont la précision des prédictions, soit $precision = 1 - risque$, où le risque est la proportion d'erreurs dans les prédictions. Les paramètres déterminés comme optimaux par validation croisée seront indiqués en gras.

TABLE 2.1 – **Précision de la classification par l'OBK.** La classe positive était les échantillons auxquels a été ajouté de l'acétaminophène. L'ensemble d'entraînement est la plaque d'échantillons mis en communs et l'on prédit sur les échantillons individuels. La valeur en gras représente la valeur optimale déterminée par validation croisée.

Taille / Chevauchement	**0%**	50%	66%	75%	80%
100 ppm	65.17%	64.04%	69.66%	64.04%	70.79%
50 ppm	64.04%	64.04%	89.89%	65.17%	96.63%
20 ppm	70.79%	96.63%	98.88%	96.63%	98.88%
10 ppm	**97.75%**	93.26%	97.75%	97.75%	98.88%
5 ppm	97.75%	96.63%	97.75%	98.88%	98.88%
2 ppm	98.88%	98.88%	98.88%	98.88%	98.88%

On voit à la table 2.1 les résultats lorsqu'on apprend sur une plaque d'échantillons mis en communs afin de prédire si un échantillon contient un ajout d'acétaminophène. On remarque que la validation croisée a obtenu comme résultat optimal la combinaison de paramètres de 10 ppm comme taille des boîtes et aucun chevauchement, ce qui donne une précision de presque 98% sur l'ensemble de test. On remarque également que ce n'est pas la précision optimale sur l'ensemble de test. Ceci est un risque de la validation croisée et seulement visible car on présente des résultats sur l'ensemble de test sur plusieurs paramètres au lieu d'un seul, ce qui n'est pas une pratique standard.

Comparons aux résultats des autres algorithmes, présentés à la table 2.8 plus loin dans ce chapitre. Les algorithmes parcimonieux et le Random Forest ont tous des précisions de 100%. Le SVM à noyau linéaire a une précision de 77% sur cet ensemble de donnée. Le SVM à noyau RBF a quant à lui une précision de 66%. Avec sa précision de 97.75% (choisie par validation croisée), le noyau OBK se compare favorablement aux autres noyaux testés pour l'algorithme du SVM.

On remarque que la précision a tendance à augmenter plus la taille des boîtes diminue. Il est probable que les tailles trop élevées contiennent trop de pics et soient donc trop peu précises et caractéristiques par rapport à certains pics importants. De plus, les boîtes plus grandes ont moins de chances de se retrouver vides, même s'il y a désalignement entre les spectres comparés. On observe aussi que la précision a tendance à augmenter si l'on augmente le chevauchement entre les boîtes. Comparons

maintenant avec les mêmes ensembles d'entraînement et de test, mais en tentant de prédire la présence de clomiphène dans les échantillons.

TABLE 2.2 – **Précision de la classification par l'OBK.** La classe positive était les échantillons auxquels a été ajouté du clomiphène. L'ensemble d'entraînement est la plaque d'échantillons mis en communs et l'on prédit sur les échantillons individuels. La valeur en gras représente la valeur optimale déterminée par validation croisée.

Taille / Chevauchement	**0%**	50%	66%	75%	80%
100 ppm	**28.09%**	28.09%	28.09%	32.58%	32.58%
50 ppm	28.09%	32.58%	49.44%	30.34%	59.55%
20 ppm	28.09%	61.80%	46.07%	29.22%	28.09%
10 ppm	65.17%	28.09%	70.79%	57.30%	29.22%
5 ppm	29.22%	28.09%	29.22%	29.22%	30.34%
2 ppm	39.33%	28.09%	28.09%	55.06%	30.34%

Considérons la table 2.2. Les paramètres choisis par validation croisée sont une taille de boîte de 100 ppm et aucun chevauchement. Cette combinaison de paramètre fait très peu de sens physiquement, puisque une taille de boîte de 100 ppm est extrèmement grand. Par contre, on remarque que la précision avec ces paramètres est très mauvaise. On observe en fait que pratiquement l'entièreté des précisions montrées ici sont mauvaises, puisque le score correspondant à simplement prédire la classe majoritaire est 66% de précision, vu que l'on a 33% d'exemples positifs et 66% d'exemples négatifs. Seule une combinaison de paramètres arrive à battre ce score minimal, soit 70.79% pour des boîtes de 10 ppm et un chevauchement de 66%, ce qui n'est pas un score très performant.

On remarque quand même une certaine tendance où les boîtes plus grandes ont de moins bonnes performances. Cette fois, les performances retombent aussi dans les plus petites boîtes comme 5 ppm et 2 ppm. Les tailles de 10 ppm à 50 ppm contiennent de meilleurs scores. Une légère tendance se dessine au niveau des chevauchements dans ces tailles de boîtes, où les chevauchements de 50% à 75% semblent favoriser la performance.

Comparons les résultats du SVM avec le noyau OBK avec d'autres algorithmes. Ces résultats sont présentés en plus de détails à la table 2.8, plus tard dans ce chapitre. Dans le cas du noyau à boîtes chevauchantes, le classificateur choisi par validation croisée a une précision de 28% environ. Sur le même ensemble de données, un SVM linéaire a un résultat similaire avec 33% de précision. Un SVM avec un noyau RBF ainsi qu'un classificateur Random Forest ont des précisions de 66%. Deux classificateurs parcimonieux, l'arbre de décision et le SCM, ont respectivement 75% et 92% de précision. Le noyau OBK a donc de mauvaises performances sur ce problème comparé aux autres algorithmes.

Une explication possible pour la différence de performance du noyau sur les deux classes est l'effet des différents ajouts sur le spectre. L'acétaminophène, sur lequel l'algorithme performe très bien, va ajouter plusieurs pics et faire de grandes variations d'intensité à plusieurs points. Les pics associés à l'acétaminophène sont aussi possiblement déjà présents dans les échantillons à plus faible intensité. En

contraste, le clomiphène cause l'apparition d'un seul pic de haute intensité. Il n'y a aucun pic présent à cet endroit autrement. Il est possible que, vu l'absence de pics à cet endroit, les boîtes contenant le pic de clomiphène n'aient que des boîtes homologues vides quand on compare un spectre avec clomiphène ajouté et un autre sans clomiphène.

Considérons maintenant le cas où l'algorithme apprend sur les échantillons individuels où l'on a ajouté les molécules pharmaceutiques et où l'on essaie de prédire sur les échantillons mis en communs.

TABLE 2.3 – **Précision de la classification par l'OBK.** La classe positive était les échantillons auxquels a été ajouté de l'acétaminophène. L'ensemble d'entraînement est la plaque d'échantillons individuels et l'on prédit sur les échantillons mis en communs. La valeur en gras représente la valeur optimale déterminée par validation croisée.

Taille / Chevauchement	**0%**	50%	66%	75%	80%
100 ppm	**92.13%**	92.13%	92.13%	91.00%	94.38%
50 ppm	92.13%	91.00%	89.89%	91.00%	91.00%
20 ppm	91.00%	91.00%	89.89%	89.89%	89.89%
10 ppm	87.64%	88.76%	88.76%	88.76%	94.38%
5 ppm	86.52%	95.51%	89.89%	89.89%	89.89%
2 ppm	88.76%	88.76%	88.76%	88.76%	88.76%

La table 2.3 contient les résultats de prédictions sur l'ajout d'acétaminophène, lorsqu'entraîné sur les échantillons individuels. On remarque encore une bonne performance de prédiction, malgré qu'elle soit plus basse que dans le cas précédent. La combinaison de paramètres de taille de boîtes de 100 ppm et aucun chevauchement a été choisie comme étant optimale par validation croisée. Par contre, il faut noter que la plupart des combinaisons de paramètres obtenaient à un risque de validation croisée de 0 (soit une précision de 100%) également. Dans ce cas, il n'y a pas vraiment de tendances claires relié à la taille des boîtes ou le chevauchement, sauf peut-être le fait que les boîtes de 2 ppm aient des performances moindres que les autres tailles.

Comparons aux résultats de la table 2.10, présentée plus loin dans ce chapitre. La précision de 92% du noyau OBK est comparable ou supérieure à celles du Random Forest (76%), Decision Tree (94%), SVM à noyau linéaire (91%) et SVM à noyau RBF (72%). Le seul algorithme ayant une performance clairement supérieure est le SCM, ayant une précision de presque 98%.

Considérons maintenant la table 2.4, présentant les résultats de prédiction de l'ajout de clomiphène. On a encore de mauvais résultats pour la prédiction sur cette molécule. Il est à noter que le score de 64.04% de précision revient très souvent, car il correspond à un classificateur qui prédit seulement négatif sur l'ensemble d'entraînement. Tel quel mentionné précédemment, certains spectres ont du être retirés du jeu de données. L'absence de ces spectres cause ce changement. Encore une fois, certaines combinaisons de paramètres ont une performance meilleure que la performance naive de 64%, mais rien de meilleur que 69.66%. Il semble clair que le noyau n'est pas adapté au problème de la prédiction de l'ajout de clomiphène.

TABLE 2.4 – **Précision de la classification par l'OBK.** La classe positive était les échantillons auxquels a été ajouté du clomiphène. L'ensemble d'entraînement est la plaque d'échantillons individuels et l'on prédit surs les échantillons mis en communs. La valeur en gras représente la valeur optimale déterminée par validation croisée.

Taille / Chevauchement	0%	50%	**66%**	75%	80%
100 ppm	64.04%	64.04%	64.04%	64.04%	64.04%
50 ppm	64.04%	64.04%	64.04%	66.29%	64.04%
20 ppm	64.04%	64.04%	**64.04%**	65.17%	64.04%
10 ppm	64.04%	67.42%	62.92%	64.04%	64.04%
5 ppm	65.17%	65.17%	69.66%	62.92%	67.42%
2 ppm	69.66%	60.67%	64.04%	57.30%	52.81%

Si l'on compare encore une fois aux différents autres algorithmes, dont on peut voir les résultats à la table 2.11 plus loin dans ce chapitre, on remarque que le noyau OBK a la même performance que les autres noyaux du SVM. Les trois autres ont par contre des performances supérieures se trouvant entre 86% de précision (Random Forest) et 97% (Arbre de décision, SCM).

Finalement, le noyau OBK a été testé sur le même ensemble de données, mais avec les ensembles d'entraînement et de tests séparés aléatoirement sur le nombre total d'échantillons au lieu de tester explicitement une plaque sur une autre.

TABLE 2.5 – **Précision de la classification par l'OBK.** La classe positive était les échantillons auxquels a été ajouté de l'acétaminophène. L'ensemble d'entraînement et l'ensemble de tests sont pris aléatoirement parmi les échantillons. La valeur en gras représente la valeur optimale déterminée par validation croisée.

Taille / Chevauchement	**0%**	50%	66%	75%	80%
100 ppm	**98.96%**	98.96%	97.92%	98.96%	97.92%
50 ppm	98.96%	98.96%	97.92%	98.96%	98.96%
20 ppm	96.87%	98.96%	97.92%	97.92%	97.92%
10 ppm	98.96%	98.96%	97.92%	97.92%	97.92%
5 ppm	96.87%	97.92%	95.83%	96.87%	97.92%
2 ppm	97.92%	95.83%	95.83%	95.83%	95.83%

La table 2.5 contient les résultats de la prédiction de l'ajout d'acétaminophène dans les ensembles d'entraînement et de test aléatoires. Encore une fois, on observe une bonne performance de l'OBK afin de faire la prédiction pour l'acétaminophène. La combinaison de paramètres optimaux par validation croisée était une taille de boîtes de 100 ppm et un chevauchement de 0%. Il est à noter qu'encore une fois, plusieurs autres combinaisons de paramètres ont égalé le score de validation croisée de cette combinaison. Étant donnée que les ensembles d'entraînement et de test sont aléatoires dans ce cas, on constate que l'algorithme maintient sa performance.

Comparons aux résultats des autres algorithmes sur le même ensemble de données, qui se trouve à a table 2.12. Encore une fois, pour la prédiction de la présence d'acétaminophène, le noyau OBK a une performance comparable aux autres algorithmes. Le Random Forest a une précision de 96%, l'arbre de décision une précision de 98% et deux algorithmes ont une précision de 99% : le SVM à noyau linéaire et le SCM. Le SVM à noyau RBF a une moins bonne performance avec une précision de 73%.

TABLE 2.6 – **Précision de la classification par l'OBK.** La classe positive était les échantillons auxquels a été ajouté du clomiphène. L'ensemble d'entraînement et l'ensemble de tests sont pris aléatoirement parmi les échantillons. La valeur en gras représente la valeur optimale déterminée par validation croisée.

Taille / Chevauchement	0%	**50%**	66%	75%	80%
100 ppm	70.83%	**68.75%**	70.83%	70.83%	65.62%
50 ppm	66.67%	71.87%	64.58%	67.71%	64.58%
20 ppm	61.46%	67.71%	61.46%	73.96%	65.62%
10 ppm	64.58%	71.87%	64.58%	67.71%	64.58%
5 ppm	61.46%	58.33%	60.42%	61.46%	62.50%
2 ppm	61.46%	61.46%	64.58%	61.46%	61.46%

Comparons à la performance de l'algorithme à la prédiction de la présence de clomiphène sur les mêmes ensembles d'entraînement et de test, présenté à la table 2.6, qui se retrouve plus loin dans ce chapitre. On remarque encore une fois une mauvaise performance de l'algorithme à ce problème. Étant donné les nouveaux ensembles d'entraînement et de test, il semble alors que l'algorithme est effectivement mal adapté à classifier la différence introduite par le clomiphène. On note plusieurs scores de 61.46%, ce qui correspond à la séparation des classes, signifiant que le classificateur dans ces cas ne fait que prédire des exemples négatifs. On observe quelques combinaisons pouvant atteindre une précision supérieure à 70%, mais même ces meilleures performances ne sont pas excellentes ou significativement différentes des autres.

Comparons une dernière fois aux performances des autres algorithmes, présentés à la table 2.13. Avec sa précision d'environ 69%, le SVM avec noyau OBK est moins performant que la plupart des autres algorithmes. Le SVM à noyau RBF a une moins bonne précision, de 63%. Les autres algorithmes ont des précisions allant de 73% jusqu'à 98%. Le noyau OBK semble toujours avoir de la difficulté à prédire la présence de clomiphène.

2.3 Application d'algorithmes existants à la spectrométrie de masse

En plus de mettre au point un nouveau noyau, une grande partie des travaux présentés ici concerne l'application de plusieurs algorithmes déjà existants aux données de spectrométrie de masse. Principalement, on recherche à appliquer des algorithmes dont les modèles de classification sont parcimonieux et interprétables afin de pouvoir trouver des biomarqueurs dans les spectres.

Une grande partie des travaux a été appliquée à des spectres qui ont été générés à partir de technologie LDTD. Les résultats associés à ces jeux de données seront présentés en premier. Une seconde partie concerne des spectres acquis par LCMS. Ils seront présentés par la suite.

2.3.1 Application sur des données générées par LDTD

Jeu de données de plasma dégradé

Un premier test fait était sur deux plaques d'échantillons de plasma sanguin. Une première plaque fut décongelée et laissée à température pièce pendant 7 jours. Cette plaque forme les échantillons de plasma dégradés, formant les exemples de la classe négative. L'autre plaque fut décongelée et traitée immédiatement, formant la classe positive des échantillons frais. L'objectif de cette expérience est de voir si l'on est capable d'apprendre la différence entre les échantillons frais et dégradés à température pièce avec l'apprentissage automatique et l'acquisition de spectres avec la technologie LDTD.

Afin d'éviter tout biais, la moitié des échantillons frais ont été placé sur une plaque avec la moitié des échantillons dégradés. On partage la seconde plaque de la même manière. Ainsi, on évite qu'un biais dans les variations de masses lié aux plaques soit directement corrélé avec les classes.

La méthodologie d'acquisition de ces spectres a déjà été décrite à la Section 1.3.1 de ce mémoire, en tant que *jeu de données d'évaluations des déviations*.

Les spectres ont subi un premier traitement informatique par le logiciel MassLynx de Waters Corporation. Ce premier traitement était la centroïdation des pics. Après cela, une correction par VLM a été appliquée aux spectres des ions positifs de l'extraction ACNMeOH et aux spectres des ions négatifs de l'extraction Chlbut. Les points de correction VLM sont les mêmes qui ont été utilisés à la table1.6 du dernier chapitre. Une filtration des pics par un seuil d'intensité de 500 fut aussi appliquée. On a ensuite fait l'alignement des pics avec une distance maximale de 5 ppm. Le spectre repère fut bâti sur l'ensemble d'entraînement (voir le prochain paragraphe). Finalement, les 500 pics les plus intenses en moyenne par type de spectres furent sélectionnés. On a donc 1000 pics par échantillons afin de faire la comparaison (500 pics de la condition ACNMeOH-positive et 500 de la condition Chlbut négative). Cette dernière filtration a été appliquée afin de rendre l'apprentissage computationnellement assez rapide pour l'algorithme du SCM.

On a alors un total de 141 échantillons. Plusieurs des échantillons acquis ont dû être retirés à cause de problèmes lors de l'acquisition soit du spectre ACNMeOH-positif ou du spectre Chlbut négatif. On compte 73 échantillons de classe "frais" et 68 échantillons de classe "dégradée" dans l'ensemble des données. L'ensemble des données a ensuite été séparé aléatoirement en un ensemble de tests de 30 échantillons et un ensemble d'entraînement de 111 échantillons. Les hyperparamètres optimaux à utiliser furent déterminés par validation croisée à 5 plis.

Plusieurs algorithmes ont été appliqués à ces données. Le premier est celui des forêts d'arbres aléatoires (*Random Forest*). Cet algorithme n'a qu'un hyperparamètre à sélectionner, le nombre d'estima-

teurs. Ce nombre d'estimateurs pourrait se trouver dans l'ensemble $\{1,5,10,20,30,40,50,60,70,80,90,100,150\}$. L'arbre de décision (*decision tree*) a quant à lui deux hyperparamètres à sélectionner. Le premier est la profondeur maximale de l'arbre, qui pouvait se retrouver dans l'ensemble $\{2,3,4,5,6,7,8,9,10\}$. Le second paramètre est le nombre minimal que chaque décision doit séparer. Ce nombre pouvait être choisi dans $\{2,5,10,15,20\}$.

Deux formes de l'algorithme des machines à vecteur de support (*Support Vector Machine*, SVM) ont été essayées sur cet ensemble de données. Le premier est le SVM à noyau linéaire. Dans ce cas, il n'y a pas de fonction de noyau projetant dans une dimension supérieure, seulement l'algorithme SVM qui tente de séparer les données dans l'espace des caractéristiques. Son seul paramètre est le paramètre C, qui peut prendre une valeur dans un espace logarithmique de 7 valeurs entre 10^{-3} et 10^{3}. Le second noyau utilisé est le noyau *Radial Basis Function* (RBF). Vapnik (1995) C'est un noyau utilisé couramment en apprentissage automatique, qui projette les exemples dans un espace de dimensionnalité infinie. Cette forme de l'algorithme a deux paramètres à sélectionner, soit le paramètre C et le paramètre γ. Les deux peuvent choisir des valeurs dans un espace logarithmique de 7 valeurs contenues entre 10^{-3} et 10^{3}. Un espace logarithmique à 7 valeurs entre 10^{-3} et 10^{3} correspond à l'ensemble $\{0.001,0.01,0.1,1,10,100,1000\}$.

Le dernier algorithme utilisé est le SCM. Les règles de classifications pour l'algorithme correspondent à des souches de décision. On définit les souches de décision sur chacune des caractéristiques des exemples. On définit deux souches de décision pour chaque valeur différente dans l'ensemble d'entraînement sur chacune des caractéristiques, une qui est plus grande ou égale à la valeur et une qui est inférieure à la valeur. Par exemple, si l'on retrouve 5 valeurs différentes sur le pic de ratio m/z 123.4567, alors on définira 10 souches qui correspondent à $x \geq val_1$, $x < val_1$, $x \geq val_2$, $x < val_2$, etc. On peut alors avoir jusqu'à un maximum de $2m$ souches de décision par caractéristique, où m est le nombre d'exemples dans l'ensemble d'entraînement. C'est pourquoi on applique une sélection sur les pics, car le nombre de souches serait très grand avec un grand nombre d'exemples et de pics. Le SCM a trois paramètres à sélectionner. Le premier est le type de modèle à bâtir, soit une conjonction de règles ou une disjonction de règles. Un autre paramètre est le nombre de règles maximum à considérer. Ce paramètre est sélectionné dans l'ensemble $\{1,2,3,4,5,6,7,8,9,10,12,15\}$. Finalement, le SCM a un paramètre p. Ce paramètre p pouvait être dans un espace logarithmique de 5 valeurs entre 10^{-2} et 10^{2} ($\{0.01,0.1,1,10,100\}$).

La table 2.7 présente les résultats de classification. On remarque que chaque algorithme n'a pas de bonne précision sur l'ensemble de tests. Les scores de 43.33% et 56.67% correspondent aux classificateurs qui tentent, respectivement, de prédire que tout l'ensemble est fait d'échantillons "dégradés" et "frais". On remarque pourtant que plusieurs des algorithmes ont un score parfait ou presque sur l'ensemble d'entraînement. Le SCM est celui avec une moins bonne performance sur l'ensemble d'entraînement avec environ 83% de performance. Cela est une performance assez basse sur un ensemble d'entraînement. Cela indique que les algorithmes ont pu apprendre des séparations des exemples de l'ensemble d'entraînement, mais que celles-ci ne généralisent pas sur l'ensemble de test. Il semble

TABLE 2.7 – **Résultats de la classification d'échantillons de plasma frais vs des échantillons de plasma dégradés sur plaques mixtes**. Les scores de classification sont indiqués en précision, qui correspond à $precision = 1 - risque$ où le risque est la proportion d'erreurs faites par le classificateur.

Algorithme	Paramètres	Précision empirique	Précision de test
Random Forest	Estimateurs = 100	100.00%	56.67%
Décision Tree	Profondeur = 9, exemples séparés = 2	100.00%	43.33%
SVM Linéaire	$C = 1$	100.00%	56.67%
SVM Noyau RBF	$C = 1$, $\gamma = 0.01$	99.10%	56.67%
SCM	Conjonction, Attributs = 12, $p = 10$	82.88%	43.33%

donc que tous les algorithmes, y compris le SCM, subissent une forme de surapprentissage.

Comme mentionné au paragraphe précédent, on peut remarquer que le SCM a utilisé un grand nombre d'attributs et a quand même une précision de seulement 83% sur l'ensemble d'entraînement. Cela suggère que c'est une tâche très difficile de séparer les échantillons avec peu de règles dans ce cas. Il est également possible qu'un plus grand nombre d'échantillons soit nécessaire à l'apprentissage. De plus, tous les algorithmes n'arrivent pas à généraliser.

Une explication possible pour ces résultats est qu'il n'y a pas de pic permettant de faire la distinction entre les classes "frais" et "dégradé" dans les spectres. L'hypothèse posant que l'on pourrait faire la différence entre ces classes à partir seulement des petits métabolites est possiblement erronée. Une autre possibilité est qu'il y a trop de variation, en termes de masse ou d'intensité, pour que les algorithmes d'apprentissage puissent apprendre et généraliser sur les spectres de masse générés par la méthode décrite plus haut. Il semble alors qu'il faille tester si les méthodes de traitement de données et l'apprentissage automatique sont capables d'apprendre et de généraliser sur des spectres de masse.

Jeu de données de plasma avec molécules pharmaceutiques ajoutées

Étant donné l'incapacité à bien prédire dans l'expérience précédente peut provenir d'un problème au niveau des spectres et leur contenu ou bien du traitement des données et de l'apprentissage automatique, il nous faut tester une de ces deux hypothèses. Pour ce faire, un nouvel ensemble de données a été généré. Il sert à faire une preuve de concept afin de savoir si l'apprentissage automatique est applicable à des spectres de masse acquis par source LDTD. Afin de se faire, il nous faut avoir des spectres dont on connaît le contenu et dont on est certain qu'il contienne un moyen de faire la différence entre les classes.

Le nouvel ensemble de données utilisé pour ce faire était celui d'échantillons de plasma avec molécules pharmaceutiques ajoutées, qui a déjà été mentionné dans la section à propos du noyau à boîtes chevauchantes. Son acquisition est donc décrite à la Section 2.2.3. Ce jeu de données est composé de

deux plaques d'échantillons de plasma. Une première plaque a vu tous ses échantillons mis en communs, c'est-à-dire que les 96 échantillons furent mélangés et re séparés en 96 aliquots. La seconde a gardé ses échantillons individuels, soit 1 échantillon différent par puits. Dans le tiers des échantillons a été ajoutée une solution contenant du clomiphène. Une autre solution à base de comprimés d'acétaminophène a été faite et ajoutée à un second tiers des échantillons. Le dernier tiers est resté sans ajouts. Parmis les 96 échantillons mis en communs, il n'y aura que les ajouts et les différences de mesure expérimentales qui seront différents de puits en puits pour cette portion du jeu de données. Dans un second temps, il y a également 96 échantillons de plasma avec variations interindividuelles en plus des ajouts introduits. Ainsi, on détient un jeu de données avec trois classes d'échantillons où l'on sait que la différence entre les classes se retrouve dans le spectre de masse, soit une classe sans ajout, une classe avec ajout de clomiphène et une classe avec ajout d'acétaminophène. Il existe alors une façon de vérifier si les algorithmes d'apprentissage peuvent retrouver une vérité qui nous est connue.

Les spectres ont reçu un premier traitement informatique dans le logiciel MassLynx de Waters Corporation. Ce traitement était la centroïdation des pics. Une correction par VLM a ensuite été appliquée, utilisant les points de corrections indiqués à la table 1.6 pour les spectres ACNMeOH-positifs. Une filtration des spectres a ensuite été appliquée avec un seuil d'intensité de 500 afin d'éliminer le bruit de fond des spectres. Finalement, un alignement a été appliqué sur les spectres. La distance maximale était de 15 ppm. Le spectre de repère a été bâti sur l'ensemble d'entraînement seulement, puis l'alignement a été appliqué à tous les spectres.

L'ensemble des données est alors composé de 89 échantillons mis en communs, comme certains spectres ont eu des problèmes lors de l'acquisition et ont dû être retirés [3]. Ces échantillons retirés étaient tous des ajouts d'acétaminophène, résultant en 25 de ce type d'échantillon au lieu de 32 sur cette plaque. Nous avons 96 échantillons individuels comprenant 32 exemples de chacune des trois classes.

Trois types d'expériences ont été faites à partir de cet ensemble de données. La première (Expérience 1) est que l'on déclare les échantillons mis en commun comme ensemble d'entraînement et les échantillons individuels comme ensemble de tests. La deuxième expérience (Expérience 2) est l'inverse, avec les échantillons individuels servant d'ensemble d'entraînement. On peut ainsi observer l'effet des variations interindividuelles sur l'apprentissage. Finalement, la troisième expérience (Expérience 3) consiste à séparer l'ensemble des données en deux, aléatoirement. On a alors un ensemble de 93 échantillons pour l'entraînement et 92 échantillons pour le test, chacun des ensembles composés d'échantillons mis en communs et individuels. Dans ces trois expériences, l'objectif est de classifier si un échantillon contient un ajout d'acétaminophène.

Chaque algorithme avait des hyperparamètres à valider par validation croisée à 5 plis. L'algorithme des forêts d'arbres de décision pouvait avoir un nombre d'estimateurs dans l'ensemble $\{1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200\}$. L'arbre de décision pouvait avoir une profondeur maximale d'entre 2

3. Problèmes

et 10 ($\{2,3,4,5,6,7,8,9,10\}$) et un nombre minimal d'exemples à séparer par règle dans l'ensemble $\{2,5,10,15,20\}$. Le SVM à noyau linéaire n'a que le paramètre C à choisir, qui pouvait adopter une valeur dans $\{0.001,0.01,0.1,1,10,100,1000\}$. Ce même ensemble était utilisé pour les deux paramètres du SVM à noyau RBF, soit les paramètres C et γ. Finalement, le SCM avait trois paramètres à valider. Le premier est son modèle d'apprentissage, soit par conjonction ou disjonction. Le nombre maximal de règles utilisé par le SCM pouvait être de $\{1,2,3,4,5,6,7,8,9,10,12,15\}$. Finalement, le paramètre p du SCM pouvait prendre des valeurs de $\{0.01,0.1,1,10,100\}$.

Un changement fut également appliqué à l'algorithme du SCM. En cas d'égalité de l'utilité de deux règles, une fonction de bris d'égalité a été mise au point. Cette fonction favorise les règles sur les caractéristiques ayant, en moyenne, une plus forte intensité. Ce choix a été fait afin de favoriser les pics de plus haute intensité pour la décision.

Expérience 1 - Entraînement sur les échantillons mis en commun

TABLE 2.8 – **Résultats de la classification de l'ajout d'acétaminophène dans les échantillons de plasma.** Les algorithmes sont entraînés sur la plaque d'échantillons mis en commun et prédit sur les échantillons individuels (Expérience 1). Les scores de classification sont indiqués en précision, qui correspond à $precision = 1 - risque$ où le risque est la proportion d'erreurs faites par le classificateur.

Algorithme	Paramètres	Précision empirique	Précision de test
Random Forest	Estimateurs = 200	100%	100%
Décision Tree	Profondeur = 2, exemples séparés = 20	100%	100%
SVM Linéaire	$C = 0.001$	100%	77.08%
SVM Noyau RBF	$C = 0.001$, $\gamma = 0.001$	71.91%	66.67%
SCM	Disjonction, Attributs = 1, $p = 0.01$	97.75%	100%

Les résultats de la classification de l'ajout d'acétaminophène alors qu'on s'entraîne sur les échantillons mis en commun (Expérience 1) sont visibles à la table 2.8. On remarque d'excellentes performances sur ce problème par les algorithmes du SCM, de l'arbre de décision et des forêts d'arbres aléatoires. Le SVM à noyau linéaire a aussi réussi à séparer les classes dans l'ensemble d'entraînement, mais le classificateur n'a pas très bien généralisé aux échantillons individuels. On peut remarquer que le SVM à noyau RBF a de mauvaises performances et tend à ne prédire que la classe majoritaire. Malgré des tentatives d'élargir l'espace des paramètres explorés jusqu'à 10^{-20} et 10^{10}, les classificateurs sont toujours restés avec les mêmes risques empiriques et de validation croisée. Les résultats présentés pour cet algorithme dans la table 2.8 et les tables suivantes seront simplement ceux de l'espace des paramètres de base (10^{-3} à 10^{3}).

On peut voir à la table 2.9 les résultats reliés à la classification de l'ajout de clomiphène pour l'expérience 1, appris sur les échantillons mis en communs. Dans ce cas, on observe de moins bons résultats

TABLE 2.9 – **Résultats de la classification de l'ajout de clomiphène dans les échantillons de plasma.** Les algorithmes sont entraînés sur la plaque d'échantillons mis en commun et prédit sur les échantillons individuels (Expérience 1). Les scores de classification sont indiqués en précision, qui correspond à $precision = 1 - risque$ où le risque est la proportion d'erreurs faites par le classificateur.

Algorithme	Paramètres	Précision empirique	Précision de test
Random Forest	Estimateurs = 60	100%	66.67%
Décision Tree	Profondeur = 3, exemples séparés = 5	100%	75.00%
SVM Linéaire	$C = 0.001$	100%	33.33%
SVM Noyau RBF	$C = 0.001$, $\gamma = 0.001$	64.04%	66.67%
SCM	Conjonction, Attributs = 1, $p = 1$	97.75%	92.71%

que la classification de l'ajout d'acétaminophène. On remarque que l'arbre de décision et les forêts d'arbres aléatoires ont une bonne performance sur l'ensemble d'entraînement, mais n'arrivent pas à très bien généraliser sur les nouveaux échantillons. Le score de test des forêts d'arbres aléatoires indique que le classificateur a tenté de classer tous les échantillons en une seule classe. Le SVM a également été incapable de généraliser ses prédicteurs. Finalement, le SCM est le seul algorithme avec une bonne performance sur ce problème. Avec un seul attribut, soit le pic de clomiphène ajouté, ce classificateur arrive à bien apprendre sur les spectres.

Expérience 2 - Entraînement sur les échantillons individuels

TABLE 2.10 – **Résultats de la classification de l'ajout d'acétaminophène dans les échantillons de plasma.** Les algorithmes sont entraînés sur la plaque d'échantillons individuels et prédits sur les échantillons mis en commun (Expérience 2). Les scores de classification sont indiqués en précision, qui correspond à $precision = 1 - risque$ où le risque est la proportion d'erreurs faites par le classificateur.

Algorithme	Paramètres	Précision empirique	Précision de test
Random Forest	Estimateurs = 90	100%	76.40%
Décision Tree	Profondeur = 2, exemples séparés = 2	100%	94.38%
SVM Linéaire	$C = 0.001$	100%	91.01%
SVM Noyau RBF	$C = 0.001$, $\gamma = 0.001$	66.67%	71.91%
SCM	Conjonction, Attributs = 1, $p = 0.01$	100%	97.75%

Considérons maintenant les résultats si l'on inverse les ensembles d'entraînement et de test, apprenant sur les échantillons individuels, soit l'expérience 2. La table 2.10 montre les résultats sur la classification de l'ajout d'acétaminophène dans ce cas. On remarque de bonnes performances des algorithmes de l'arbre de décision et du SCM. Dans les deux cas, la prédiction dépend d'un seul pic. Le SVM

à noyau linéaire arrive aussi à bien généraliser son classificateur entre les deux types d'échantillons. Les forêts d'arbres aléatoires ont par contre une certaine difficulté à généraliser le prédicteur. Il est possible que, vu le principe de fonctionnement à partir d'arbres décisionnels basés sur des exemples aléatoires et des caractéristiques aléatoires, le classificateur ait possiblement trop appris à partir des variations interindividuelles et soit donc incapable de prédire sur des exemples sans ce bruit.

TABLE 2.11 – **Résultats de la classification de l'ajout de clomiphène dans les échantillons de plasma.** Les algorithmes sont entraînés sur la plaque d'échantillons individuels et prédits sur les échantillons mis en commun (Expérience 2). Les scores de classification sont indiqués en précision, qui correspond à $precision = 1 - risque$ où le risque est la proportion d'erreurs faites par le classificateur.

Algorithme	Paramètres	Précision empirique	Précision de test
Random Forest	Estimateurs = 70	100%	86.52%
Décision Tree	Profondeur = 2, exemples séparés = 2	100%	96.63%
SVM Linéaire	$C = 0.001$	100%	64.04%
SVM Noyau RBF	$C = 0.001$, $\gamma = 0.001$	66.67%	64.04%
SCM	Conjonction, Attributs = 1, $p = 0.01$	100%	96.63%

Comparons maintenant avec la classification de l'ajout de clomiphène pour l'expérience 2 à la table 2.11. On remarque de bonnes performances principalement de l'arbre de décision et du SCM. Les forêts d'arbres aléatoires ont une performance plus mitigée, sous 90% de précision. Les différents noyaux du SVM n'ont pu apprendre de classificateur qui généralise pour ce problème. Ce n'est pas étonnant, étant donné que l'ajout de clomiphène ajoute seulement un pic au spectre. Il est donc cohérent que les algorithmes plus parcimonieux aient une meilleure performance pour ce problème.

Expérience 3 - Entraînement pigé aléatoirement

TABLE 2.12 – **Résultats de la classification de l'ajout d'acétaminophène dans les échantillons de plasma.** Les ensembles d'entraînement et de tests sont tirés aléatoirement (Expérience 3). Les scores de classification sont indiqués en précision, qui correspond à $precision = 1 - risque$ où le risque est la proportion d'erreurs faites par le classificateur.

Algorithme	Paramètres	Précision empirique	Précision de test
Random Forest	Estimateurs = 90	100%	95.65%
Décision Tree	Profondeur = 2, exemples séparés = 2	100%	97.83%
SVM Linéaire	$C = 0.001$	100%	98.91%
SVM Noyau RBF	$C = 0.001$, $\gamma = 0.001$	65.59%	72.83%
SCM	Conjonction, Attributs = 1, $p = 0.01$	100%	98.91%

Finalement, comparons les résultats lorsque les ensembles d'entraînement et de test sont tirés aléatoirement dans l'ensemble de données total (Expérience 3). La table 2.12 présente la classification de l'ajout d'acétaminophène dans cette situation. On remarque que tous les algorithmes, excepté le SVM à noyau RBF, ont une bonne performance de classification dans ce cas. On peut également observer que les algorithmes du SCM et de l'arbre de décision arrivent à de bonnes performances avec encore une fois un seul pic utilisé pour la classification.

TABLE 2.13 – **Résultats de la classification de l'ajout de clomiphène dans les échantillons de plasma.** Les ensembles d'entraînement et de tests sont tirés aléatoirement (Expérience 3). Les scores de classification sont indiqués en précision, qui correspond à $precision = 1 - risque$ où le risque est la proportion d'erreurs faites par le classificateur.

Algorithme	Paramètres	Précision empirique	Précision de test
Random Forest	Estimateurs = 40	100%	72.83%
Décision Tree	Profondeur = 4, exemples séparés = 2	100%	94.57%
SVM Linéaire	$C = 0.001$	100%	77.17%
SVM Noyau RBF	$C = 0.001$, $\gamma = 0.001$	67.74%	63.04%
SCM	Conjonction, Attributs = 1, $p = 0.01$	98.91%	97.83%

La table 2.13 quant à elle présente les résultats de la classification de l'ajout de clomiphène pour l'expérience 3. Comme toujours, le SVM à noyau RBF reste incapable d'apprendre sur cet ensemble de données. On remarque de moins bonnes performances que pour l'acétaminophène sur les mêmes ensembles d'apprentissages pour les algorithmes de forêts d'arbres aléatoires et pour le SVM à noyau linéaire. On voit par contre que l'arbre de décision et le SCM gardent de bonnes performances de prédiction pour ce problème. Le SCM n'a encore une fois eu besoin de seulement un pic pour faire la classification.

Les résultats sur cet ensemble de données établissent donc une preuve de concept que les algorithmes d'apprentissage sont capables d'apprendre sur le contenu d'un spectre de masse généré par LDTD et avec les méthodes de prétraitement développées au cours du projet. Plus exactement, cela fournit une preuve que les algorithmes peuvent apprendre si l'on sait que les classes ont une différence observable dans leurs spectres de masse. Par contre, l'ensemble de données a été généré avec des molécules ajoutées dans les échantillons de plasma et ne correspond pas exactement à la réalité.

2.3.2 Application sur des données générées par LC-MS

Jeu de données LC-MS Comme preuve de concept supplémentaire pour l'application de l'apprentissage automatique à la spectrométrie de masse, un nouvel ensemble de données a été généré. Il s'agit d'un ensemble de 48 échantillons de plasma qui ont été acquis par LC-MS. Comme on ne veut pas ajouter de molécules dans les échantillons et classifier les échantillons à partir de caractéristiques de leurs métabolites, nous avons utilisé les métadonnées anonymes de l'ensemble pour séparer ces

échantillons en deux classes. 28 des échantillons proviennent de donneurs masculins et 20 de donneurs féminins. On va ainsi tenter de classifier les échantillons selon le sexe du donneur à partir des spectres de masse du plasma, c'est-à-dire à partir des métabolites sanguins. Une molécule qui devrait nous fournir la différence entre ces classes est l'hormone de la testostérone. On s'attend à voir dans les spectres de plasma masculins des quantités importantes de cette hormone et den observer des niveaux fortement moindres chez les spectres de donneurs féminins.

Les échantillons ont premièrement été décongelés. Ils ont ensuite subi une extraction à l'ACN :MeOH en proportion 75 :25. On extrait 10 μL d'échantillon avec 90 μL de solution d'extraction ACN :MeOH. Les échantillons subissent alors une sonication de 5 minutes. On centrifuge alors les échantillons pendant 5 minutes à 5000 RPM. On récupère alors le surnageant. On a ainsi les petits métabolites du plasma sans les protéines. On prend alors 10 μL d'échantillon pour l'injecter en chromatographie liquide.

Les échantillons ont été acquis par LC-MS dans un spectromètre Synapt G2 de Waters Corporation. La chromatographie liquide s'est faite dans une colonne Acquity UPLC BEH C18 de Waters Corporation. Cette colonne contient des billes de 130 Å et est d'une taille de 2.1mm par 100 mm. La chromatographie s'est effectuée avec un gradient de solvant de 10 minutes. Le solvant était, au début de la chromatographie, composé de 100% d'eau. À la fin de la chromatographie, le solvant est de 100% d'ACN. La composition varie graduellement de 100% d'eau à 100% d'ACN à travers le temps de la chromatographie.

Les spectres ont été acquis en mode MS^e sur le spectromètre, c'est-à-dire un mode d'acquisition data-indépendant. Le spectromètre acquiert alternativement deux spectres sur les ions. Le premier est une fonction de basse énergie où l'on voit les molécules peu ou pas fragmentées. Le second ajoute une énergie de collision dans le spectromètre qui fait fragmenter les molécules. On a ainsi un spectre avec les fragments filles des molécules du premier spectre. Les spectres ont aussi été acquis en mode *resolution* par le spectromètre. Ce mode est moins précis sur les ratios de masse sur charge que le mode *high resolution* utilisée précédemment, mais est plus sensible.

Le prétraitement des spectres a été fait par le logiciel Progenesis QI de Nonlinear Dynamics. Ce logiciel fait la centroïdation des pics, l'alignement des pics ainsi que la normalisation des intensités automatiquement. Le logiciel fait aussi une sélection des pics. Cette sélection a été faite avec le seuil automatique du logiciel.

Dans cette expérience, l'algorithme des masses de verrouillage virtuelles ne fut pas appliqué. L'algorithme n'est effectivement pas nécessaire afin de rendre les spectres comparables dans ce contexte. La méthodologie LC-MS est très répandue dans le domaine de la spectrométrie de masses et des algorithmes de correction et d'alignement existent déjà afin de rendre les spectres comparables. De plus, vu que l'on dispose d'une information supplémentaire sur les pics, soit leur temps d'élution lors de la chromatographie, alors les pics sont caractérisés par le temps d'élution et le ratio de masse sur charge. L'algorithme des masses de verrouillage virtuelles n'est donc pas applicable à ce type de donné sans

changement majeur, puisqu'il est conçu pour corriger *seulement* le ratio de masse sur charge.

L'ensemble des données a été séparé aléatoirement en deux ensembles de 24 exemples. Le premier est l'ensemble d'entraînement et le second l'ensemble de tests. Les hyperparamètres ont été choisis sur l'ensemble d'entraînement par validation croisée à 5 plis. Comme précédemment, les forêts d'arbres aléatoires pouvaient avoir un nombre d'estimateurs parmi $\{1,5,10,20,30,40,50,60,70,80,90,100,150,200\}$. Les arbres décisionnels pouvaient avoir une profondeur d'entre 2 et 10 ($\{2,3,4,5,6,7,8,9,10\}$) et un minimum d'exemples séparés par règle de $\{2,5,10,15,20\}$. Le SCM avait trois hyperparamètres à valider. Le premier est le type de modèle d'apprentissage, soit une conjonction ou une disjonction. Le second est le nombre maximal de règles à utiliser, qui pouvait être dans l'ensemble $\{1,2,3,4,5,6,7,8,9,10,12,15\}$. Finalement, le paramètre p pouvait prendre des valeurs de $\{0.01,0.1,1,10,100\}$.

Pour les SVM, l'étendue des paramètres à tester a dû être grandement élargie. Pour le SVM à noyau linéaire, un espace logarithmique de 50 valeurs entre 10^{-12} et 10^{12} a été testé pour la paramètre C. Pour le SVM à noyau RBF, les deux paramètres avaient le même espace à explorer. Le paramètre C et le paramètre γ pouvaient se retrouver dans un espace logarithmique de 50 valeurs entre 10^{-12} et 10^{12}. Étant donné que l'on valide toutes les combinaisons possibles, on obtient 2500 combinaisons possibles pour le SVM RBF.

TABLE 2.14 – **Résultats de la classification du sexe de donneurs de plasma sur des données acquises par LCMS.** Les scores de classification sont indiqués en précision, qui correspond à $precision = 1 - risque$ où le risque est la proportion d'erreurs faites par le classificateur.

Algorithme	Paramètres	Précision empirique	Précision de test
Random Forest	Estimateurs = 1	83.33%	54.17%
Décision Tree	Profondeur = 2, exemples séparés = 2	100%	91.67%
SVM Linéaire	$C = 9.103 \times 10^{-11}$	100%	54.17%
SVM Noyau RBF	$C = 1 \times 10^{-12}$, $\gamma = 1 \times 10^{-12}$	58.33%	58.33%
SCM	Conjonction, Attributs = 1, $p = 0.01$	100%	95.83%

Les résultats de la classification homme-femme sur les plasmas acquis en LCMS sont présentés à la table 2.14. On remarque que, pour l'algorithme des forêts d'arbres aléatoires, on a le paramètre du nombre d'estimateurs fixé à 1. Cela signifie que ce paramètre a le meilleur risque de validation croisée, ou du moins qu'aucun autre nombre d'estimateurs essayé n'a pu avoir un meilleur risque. Avec ce paramètre, l'algorithme a une assez mauvaise performance sur l'ensemble de tests avec seulement 54% de précision. Les SVM, malgré le très large espace de paramètres exploré, ont aussi de mauvaises performances. Dans le cas du noyau linéaire, le classificateur a pu apprendre sur l'ensemble d'entraînement, mais pas généraliser sur l'ensemble de tests. Pour le noyau RBF, on remarque que les

plus petites valeurs de l'espace ont été choisies et que l'algorithme n'a pas appris ni sur l'ensemble de tests ni sur l'ensemble d'entraînement. Il existait des combinaisons de paramètres où le classificateur pouvait atteindre de bonnes performances sur l'ensemble d'entraînement. Par contre, le risque de validation croisé est égal ou pire que celui des paramètres sélectionnés ici. Il semble alors logique de conclure que le SVM à noyau RBF n'arrive pas à apprendre de classificateur qui généralise sur ce problème.

Les résultats sont par contre bien meilleurs avec les algorithmes parcimonieux de l'arbre de décision et du SCM. Dans le cas de l'arbre de décision, le classificateur utilise deux attributs (pics) pour faire la prédiction. On arrive à environ 92% de précision sur l'ensemble de tests, soit 2 erreurs. Le SCM quant à lui utilise un seul attribut afin de faire la classification. Après vérification, il s'agit du pic putatif de la testostérone. Avec une règle sur ce pic, le classificateur prédit parfaitement l'ensemble d'entraînement et fait une seule erreur sur l'ensemble de tests. Une vérification des données démontre alors qu'un exemple aberrant est présent dans l'ensemble de tests, c'est-à-dire que l'on a un exemple "masculin" dont l'intensité du pic de testostérone est inférieure à l'intensité du pic de testostérone sur deux exemples "féminin". Il est également possible que cela résulte d'une erreur de mesure sur ce pic dans le spectre "masculin". Le classificateur de l'arbre de décision utilise aussi le pic putatif de testostérone dans ses règles de classification.

Cette expérience semble donc indiquer que l'apprentissage automatique peut effectivement être appliqué à des spectres de masse d'échantillons de plasma et apprendre sur des composantes intrinsèques du plasma. De plus, les résultats de cette expérience suggèrent que les algorithmes parcimonieux sont mieux adaptés à ce type de problème.

2.4 Conclusion

En conclusion de ce chapitre, l'apprentissage automatique a été testé sur des données provenant de spectrométrie de masse. Malgré certaines embûches, on arrive à la conclusion que l'application de l'apprentissage automatique est possible sur ce type de données.

Malgré de premiers tests inconclusifs, une série d'autres tests sur des ensembles de données et également l'ajout de la technologie des VLM a prouvé que plusieurs algorithmes d'apprentissage déjà existant sont compatibles avec l'apprentissage automatique et capable de prédire sur des spectres acquis sur des échantillons de plasma et des petits métabolites non protéiques qui s'y retrouvent. De plus, plusieurs de ces tests suggèrent que non seulement les algorithmes utilisant des modèles parcimonieux ont la capacité d'apprendre sur ce type de données, mais que ce sont les algorithmes ayant le plus de facilité à apprendre et à généraliser.

De plus, une nouvelle méthode à noyau a été conçue spécifiquement avec les défis de la spectrométrie de masse avec source LDTD. Lorsque testé sur un jeu de données, ce nouveau noyau a eu de bonnes performances quant à l'identification de l'ajout d'un des deux composés, mais n'a pas bien performé

pour le second. Au total, cette méthode reste encore une piste à explorer.

Il y a de multiples avenues afin de continuer les travaux présentés dans ce chapitre. Au niveau du noyau à boîtes chevauchantes (OBK), plusieurs variantes sont imaginables et qui pourraient améliorer ses performances. Par exemple, on pourrait remplacer la simple somme que l'on fait des pics dans chaque boîte individuelle pour une autre forme de comparaison et de calcul, telle que la somme des logarithmes de l'intensité des pics, la conservation du seul pic maximum ou l'introduction de fonctions non linéaires telles que des sigmoïdes.

D'autres avenues de recherches sont aussi ouvertes dans d'autres champs de recherche de l'apprentissage automatique. Une telle branche est celle de la détection d'anomalie. Alors que les classificateurs de détection d'anomalie sont typiquement très difficiles à entraîner, on ne peut nier qu'il serait intéressant de voir ce que ces algorithmes peuvent faire. Ils apporteraient aussi un bénéfice en ce qui concerne le travail en laboratoire, puisqu'on recherche un grand nombre d'échantillons normaux au lieu de rechercher à faire des ensembles de données relativement équilibrés entre des exemples positifs et négatifs, ce qui peut être ardu lorsque la classe négative est composée d'échantillons correspondant à des pathologies.