
LOGIQUE FLOUE ET ALGORITHMES GÉNÉTIQUES POUR LE PRÉ-TRAITEMENT DE DONNÉES DE BIOPUCES ET LA SÉLECTION DE GÈNES

THÈSE DE DOCTORAT

Spécialité : Informatique

ÉCOLE DOCTORALE STIM

Présentée et soutenue publiquement

Le 13 novembre 2008

À Angers

Par **Edmundo BONILLA HUERTA**

Devant le jury ci-dessous :

<i>Rapporteurs :</i>	Cyril FONLUPT,	Professeur à l'Université du Littoral Côte d'Opale, France
	Gilles VENTURINI,	Professeur à l'Université François-Rabelais Tours, France
<i>Examineurs :</i>	Pascal NICOLAS	Professeur à l'Université d'Angers
	Carlos A. REYES GARCIA ,	Professeur au INAOEP, Mexique
<i>Directeur de thèse :</i>	Jin-Kao HAO,	Professeur à l'Université d'Angers
<i>Codirectrice de thèse :</i>	Béatrice DUVAL,	Maître de conférences à l'Université d'Angers

Remerciements

En premier lieu, je tiens à exprimer ma gratitude et reconnaissance envers mon directeur de thèse, Professeur Jin-Kao Hao, pour m'avoir accueilli au sein du laboratoire LERIA, encadré et soutenu tout au long des quatre années de mon doctorat à l'Université d'Angers. Je le remercie de m'avoir apporter toute son expérience et ses compétences sans lesquelles ce travail n'aurait pas pu aboutir.

J'offre également de sincères remerciements à Madame Béatrice Duval ma codirectrice de thèse d'y avoir consacré tant de temps et de patience, pour sa disponibilité, son attention et son soutien qui sont sans doute des éléments majeurs qui m'ont permis de mener à bien cette thèse.

J'adresse mes profonds remerciements à Monsieur Cyril Fonlupt Professeur à l'Université du Littoral et à Monsieur Gilles Venturini Professeur à l'Université François-Rabelais pour avoir accepté d'être les rapporteurs de cette thèse.

Je voudrais remercier, de façon particulière, Monsieur Pascal Nicolas professeur à l'université d'Angers examinateur interne, pour ses nombreuses suggestions au cours de la phase finale de la rédaction de ce manuscrit. Merci de m'avoir permis de terminer ma thèse dans les délais prévus.

Je souhaite aussi remercier Dr. Carlos Alberto Reyes Garcia qui, malgré son absence à cette soutenance, m'a envoyé des mots d'encouragement et de soutien tout au long de cette thèse.

Mes remerciements s'adressent aussi à tous les membres du LERIA. Je remercie tout particulièrement Eduardo Rodriguez-Tello et son épouse et ainsi que Frédéric Lardeaux pour le soutien dans mes premiers pas dans cet pays. Je remercie par la même occasion mes collègues thésards latinos : Giglia GÓMEZ VILLOUTA, José C. HERNANDEZ HERNANDEZ et Jorge MATURANA ORTIZ et aussi à tous les membres du laboratoire.

Je remercie DGEST et l'Instituto Tecnológico de Apizaco pour cette opportunité et pour avoir mis à ma disposition tous les moyens financiers pour le développement de cette thèse.

Finalement je voudrais exprimer toute ma gratitude à ma famille de m'avoir supporté et aidé, GRACIAS TOTALES!!!.

Sommaire

Introduction générale	1
1 Classification des données de puces à ADN	5
1.1 Technologie des puces à ADN	7
1.2 Jeux de données utilisés dans cette thèse	10
1.2.1 Leucémie	10
1.2.2 Cancer du Colon	11
1.2.3 Tumeurs du Cerveau	12
1.2.4 Lymphome	12
1.2.5 Cancer du Poumon	12
1.2.6 Cancer Ovarien	13
1.2.7 Cancer du Sein	13
1.2.8 Cancer de la Prostate	13
1.3 La classification supervisée	14
1.3.1 Notions de classification	14
1.3.2 Le problème de la généralisation	15
1.3.3 Évaluation d'une hypothèse de classification	16
1.4 Méthodes de classification supervisée	16
1.4.1 Méthodes à noyau et SVM	16
1.4.2 Classifieur bayésien naïf	17
1.4.3 k-PPV	18
1.4.4 Réseaux de neurones	18
1.4.5 Arbres de décision	19
1.4.6 Forêts aléatoires	19
1.4.7 Bagging	20
1.4.8 Boosting	20
1.5 Méthodes de classification non-supervisée	21
1.5.1 K-moyennes	22
1.5.2 ISODATA	22
1.5.3 Classification hiérarchique	22
1.5.4 Cartes auto-organisatrices	23
1.6 Synthèse du chapitre	24

2	Sélection d'attributs	25
2.1	Présentation informelle du problème	26
2.2	Notions de pertinence, non pertinence et redondance	27
2.2.1	Pertinence d'attributs	27
2.2.2	Redondance d'attributs	28
2.3	Sélection d'attributs	30
2.3.1	La sélection vue comme un problème d'optimisation	30
2.3.2	Schéma général de la sélection d'attributs	31
2.3.3	Génération de sous-ensembles et procédures de recherche	33
2.4	Les approches pour la sélection d'attributs	34
2.4.1	Méthodes de Filtre	34
2.4.2	Méthodes Enveloppes	36
2.4.3	Méthodes Intégrées	36
2.5	Validation des méthodes	36
2.6	Synthèse du chapitre	37
3	Pré-traitement et pré-sélection des données de puces à ADN à l'aide de la logique floue	39
3.1	Logique floue	40
3.1.1	Sous-ensembles flous	40
3.1.2	Variables linguistiques	41
3.1.3	Système d'Inférence Floue	43
3.1.4	Représentation floue des variables d'entrée	44
3.1.5	Représentation floue des variables de sortie	45
3.1.6	Définition des règles floues	45
3.1.7	Inférence à partir de règles floues	46
3.1.8	Défuzzification	49
3.2	Pré-traitement flou pour les données de puces à ADN	49
3.2.1	Représentation floue de la variable d'entrée	50
3.2.2	Représentation floue de la variable de sortie	50
3.3	Schéma de pré-sélection de gènes par la logique floue	54
3.3.1	Mesure de similarité	55
3.3.2	Relation d'équivalence floue	55
3.3.3	α -coupes	58
3.3.4	Sélection de la valeur de α -coupe "optimale"	58
3.3.5	Choix d'un représentant de chaque groupe	59
3.4	Evaluation du modèle flou de réduction de dimension	61
3.4.1	Évaluation avec un petit nombre de gènes	62
3.4.2	Evaluation avec un grand nombre de gènes	64
3.5	Synthèse du chapitre	66

4	Explorations génétiques pour la sélection de gènes	67
4.1	Généralités sur les algorithmes génétiques	68
4.1.1	Codage des individus	69
4.1.2	Génération de la population initiale	70
4.1.3	Fonction d'aptitude	70
4.1.4	Croisement	70
4.1.5	Mutation	71
4.1.6	Élitisme	72
4.1.7	Probabilités des opérateurs génétiques	73
4.1.8	Mécanisme de sélection	73
4.1.9	Critère d'arrêt	75
4.1.10	Construction d'un AG	75
4.2	Schéma général de double exploration génétique pour la sélection de gènes	76
4.2.1	Étape 1 : Première exploration génétique	78
4.2.2	Étape 2 : Analyse de la fréquence des gènes	82
4.2.3	Étape 3 : Deuxième exploration génétique	85
4.3	Résultats sur 5 exécutions	86
4.4	Résultats sur 10 exécutions	87
4.5	Synthèse du chapitre	89
5	Sélection d'attributs à l'aide d'une méthode d'approche intégrée	91
5.1	Analyse Discriminante Linéaire Généralisée	92
5.1.1	Analyse Discriminante Linéaire (ADL)	92
5.1.2	Cas d'un petit nombre d'échantillons	93
5.1.3	Pseudo-inverse	94
5.1.4	Décomposition en valeurs singulières (DVS)	94
5.2	Une approche intégrée pour la sélection de gènes basée sur ADL	95
5.2.1	Caractéristiques générales de l'AG	96
5.2.2	Représentation des individus	97
5.2.3	Fonction d'évaluation	98
5.2.4	Croisement ET basé sur ADL	99
5.2.5	Mutation basée sur ADL	101
5.3	Protocole expérimental et résultats	102
5.3.1	Évaluation des performances par un classifieur ADL	103
5.3.2	Évaluation des performances SVM	105
5.4	Comparaison de nos résultats avec d'autres travaux	107
5.5	Validation biologique des résultats	110
5.6	Synthèse du chapitre	111
	Conclusion générale	113
	Index	115

Références bibliographiques	119
Liste des publications personnelles	131
Résumé / Abstract	134

Liste des figures

1.1	Étapes d'une analyse par puces à ADN.	7
1.2	Acquisition d'image dans une analyse par puces à ADN.	9
2.1	Catégorisation d'attributs.	29
2.2	Processus de la sélection d'attributs.	32
2.3	Schéma de validation croisée pour l'évaluation d'un processus de sélection-classification.	38
3.1	Fonctions caractéristiques d'un sous-ensemble classique (a) et un sous-ensemble flou (b) pour l'exemple 3.1.	42
3.2	Fonctions d'appartenance de la variable taille.	43
3.3	Structure d'un SIF.	44
3.4	Méthode d'inférence d'une règle floue.	47
3.5	Inférence de deux règles activées	48
3.6	Inférence à partir de 2 règles floues.	48
3.7	Illustration du pré-traitement flou.	50
3.8	Fuzzification de la variable d'entrée Niveau d'Expression (NE).	51
3.9	Fuzzification de la variable de sortie Nouveau Niveau d'Expression (NNE).	51
3.10	Processus d'inférence floue pour la normalisation des niveaux d'expression.	53
3.11	Défuzzification par centre de gravité.	54
3.12	Matrice de similarité.	56
3.13	Application de la fermeture à partir de la matrice de similitude (S).	57
3.14	Décomposition d'un sous-ensemble flou par alpha-coupes.	58
3.15	Représentation d'une matrice de données de puces à ADN et son pré-traitement flou.	59
3.16	Relation d'équivalence floue et les niveaux d'alpha-coupe.	60
3.17	Processus d'évaluation. a) Modèle classique de filtrage utilisé pour faire la comparaison, b) Modèle combiné utilisant un pré-traitement flou	63
3.18	Comparaison entre notre modèle combiné CM1 et la méthode de filtrage BW pour le jeu de données du Poumon.	65
3.19	Comparaison entre notre modèle combiné CM1 et la méthode de filtrage BW pour le jeu de données Ovarien.	66
4.1	Optima locaux et optimum global.	69

4.2	Exemple de croisement à 1 point.	71
4.3	Exemple de croisement en 2 points.	72
4.4	Exemple de croisement uniforme.	72
4.5	Méthode de la roulette pour l'exemple 3.1.	74
4.6	Procédure générale pour le pré-traitement, la pré-sélection et la sélection des sous-ensembles de gènes à l'aide d'une méthode d'enveloppement.	77
4.7	Division des données en 10-blocs. On construit le modèle classifieur sur 9 groupes et on le teste sur le 10-ième groupe. Puis on change de groupe test et on répète le même procédé jusqu'à avoir réalisé 10 combinaisons. On considère alors la moyenne des validations comme le taux de classification.	79
4.8	Croisement à 1 points.	80
4.9	Exemple de mutation en 3 points.	81
4.10	Gènes retenus pour chaque sous-ensemble sauvegarde dans le archive de haute qualité.	83
4.11	Fonction d'aptitude des sous-ensembles de gènes de la Leucémie qui ont été sauvegardés dans le archive de haute qualité.	84
4.12	Exemple de la fréquence des gènes pour la Leucémie qui ont été sauvegardés dans l'archive de sous-ensembles pertinents.	84
5.1	Codage d'un individu en utilisant les coefficients de ADL.	98
5.2	Sélection des deux parents de la population initiale.	100
5.3	Détermination de nombre de gènes à enlever de chaque parent.	101
5.4	Croisement des parents avec l'opérateur ADL-ET-X.	102

Liste des tables

1.1	Matrice d'expression des gènes.	10
3.1	Réduction des données à partir des relations d'équivalence floue.	61
3.2	Résultats obtenus par le modèle hybride avec $p = 30$ et $k = 5$	64
3.3	Résultats du protocole de comparaison avec $p = 100$ et $k = 5$	65
4.1	Individus d'une population fictive avec leurs fonctions d'aptitude.	73
4.2	Paramètres pour la première exploration génétique (Étape 1) de l'exemple 1.	82
4.3	Extrait du rang des 100 premiers gènes avec la plus haute fréquence pour le jeu de données de la Leucémie.	85
4.4	Paramètres pour deuxième exploration génétique (Étape 3) de l'exemple 1.	85
4.5	Paramètres pour la première exploration génétique (Étape 1).	86
4.6	Paramètres pour la deuxième exploration génétique (Étape 3).	86
4.7	Comparaison des plus pertinents travaux sur la classification de puces à ADN avec notre approche d'enveloppe.	87
4.8	Résultats de approche de double exploration génétique (dernière ligne) comparé avec les travaux les plus importants de la classification de données de puces à ADN. Taux de classification en parenthèse si le nombre de gènes est disponible.	88
5.1	conditions initiales pour les expérimentations.	103
5.2	Résultats de notre approche intégrée avec différentes tailles de population et comme classifieur ADL.	104
5.3	Résultats de notre approche intégrée avec différents tailles de population et comme classifieur SVM.	106
5.4	Résultats de ADL basé sur AG (dernières lignes) comparé avec les travaux les plus importants de la classification de données de puces à ADN. Taux de classification à gauche et entre parenthèses le nombre de gènes s'il est disponible à droite.	108
5.5	Les 3 Gènes sélectionnés pour la Leucémie par AG-M1/Exp2 qui fournissent un taux de classification de 100% (pop=100, classifieur ADL).	110
5.6	Les 7 Gènes sélectionnés par AG-M1/Exp2 qui donnent une performance de 100% pour le jeu de données DLBCL (pop=50, classifieur SVM).	110

5.7	Les 13 Gènes sélectionnés par AG-M1/Exp2 pour le jeu du Colon qui fournissent 100% de classification (pop=50, classifieur SVM).	111
5.8	Les 20 Gènes sélectionnés par AG-M2/Exp2 pour le jeu de données CNS en donnant un taux de classification de 100%(pop=30, classifieur SVM). .	111

Liste des algorithmes

1.1	k-moyennes	22
1.2	ISODATA	23
3.1	Fermeture Transitive max-min	56
4.1	Algorithme génétique	75
5.1	Obtention du Vecteur Propre	98

Introduction générale

Contexte de travail

L'évolution de l'informatique et des technologies de stockage a vu de nos jours une explosion de grands volumes des données. Il est maintenant possible d'analyser des grandes quantités de données de dimension élevée grâce aux performances accrues des ordinateurs. Néanmoins si l'on doit traiter des données décrites par un très grand nombre d'attributs, les méthodes classiques d'analyse, d'apprentissage ou de fouille de données peuvent se révéler inefficaces ou peuvent conduire à des résultats inexacts. Plusieurs domaines qui intéressent beaucoup la communauté de la fouille de données fournissent des données qui sont décrites par des milliers d'attributs. C'est le cas par exemple pour le traitement des textes dont les applications issues du web sont très nombreuses. C'est aussi le cas lorsqu'on veut analyser des images de haute résolution. Enfin un domaine plus récent, celui de la bioinformatique fournit également des données de très grande dimension où il n'est pas rare d'avoir à manipuler plusieurs milliers d'attributs.

Afin d'appliquer efficacement les techniques d'analyse ou d'apprentissage dans ces différents domaines, il est nécessaire de réduire la dimension des données en sélectionnant les attributs les plus intéressants pour le problème étudié [Kohavi et John, 1997; Blum et Langley, 1997; Duch, 2006; Cunningham, 2007; Cios *et al.*, 2007]. Les méthodes de sélection d'attributs et de réduction dimensionnelle permettent de faciliter la compréhension des données, de réduire le temps d'apprentissage et en éliminant les données non pertinentes et redondantes, elles permettent aussi d'améliorer la performance de l'apprentissage.

Motivation et objectifs

Ce travail s'intéresse aux données issues des puces à ADN. Cette technologie permet de mesurer simultanément les niveaux d'expression de gènes au sein d'échantillons de tissus dans des conditions expérimentales données. Les premiers jeux de données du domaine oncologique ont été publiés à la fin des années 90 [Golub *et al.*, 1999; Alizadeh *et al.*, 2000]. Le travail de Golub par exemple a montré que les seules données issues des puces à ADN permettaient de discriminer deux formes de leucémie. De plus, parmi les quelques 7000 gènes testés sur les puces pour cette expérience, un petit nombre de gènes (environ 50) apparaît comme très important pour la reconnaissance des deux formes de la

maladie. Depuis le début des années 2000, un grand nombre de travaux se sont intéressés au problème de la classification des données issues des puces à ADN avec l'espoir de proposer des outils de diagnostic des différents cancers et aussi de compréhension des mécanismes de ces pathologies.

Comme les données analysées présentent plusieurs milliers d'attributs, il est nécessaire de proposer des méthodes innovantes pour la réduction de dimension ou la sélection de gènes. Les données issues des biopuces sont obtenus à partir d'un protocole complexe (décrit dans le chapitre 1) où plusieurs étapes peuvent introduire du bruit dans les données. Nous nous sommes donc intéressés au problème du pré-traitement de ces données. Nous proposons une méthode de pré-traitement basée sur la logique floue qui gère bien les éventuels erreurs imprécises des données.

Le problème de la sélection d'attributs consiste ensuite à choisir les attributs les moins redondants et les plus pertinents pour accomplir la tâche de classification et cette thèse propose plusieurs méthodes pour réaliser au mieux cette sélection.

Organisation de la thèse

Cette thèse est composée de cinq chapitres dont nous donnons une brève description dans les lignes suivantes :

- Dans le chapitre 1, nous donnons d'abord une brève présentation de la technologie des puces à ADN pour expliquer les caractéristiques des jeux de données qu'elles fournissent. Nous présentons également les principaux jeux qui sont disponibles et sur lesquels nous avons travaillé. Nous rappelons ensuite la problématique de la classification supervisée et de la classification non-supervisée, en se concentrant sur les techniques qui sont utilisées pour le diagnostic à partir des données de puces à ADN.
- Le chapitre 2 aborde le problème de la sélection d'attributs en rappelant les principales approches qui peuvent être appliquées. On se concentre sur les méthodes qui sont applicables pour la sélection de gènes dans les jeux de données issues des puces à ADN.
- Le chapitre 3 présente la méthode que nous proposons pour une réduction de dimension des données de puces à ADN. Notre approche est basée sur la logique floue et prend donc en compte le caractère imprécis des données manipulées. Il s'agit d'une phase de pré-sélection qui permet de réduire la redondance présente dans les données et de retenir des gènes pertinents pour la tâche de classification. L'intérêt de cette étape de pré-sélection est évaluée grâce à un protocole expérimental qui permet de constater son impact dans un processus de sélection et de classification.
- Le chapitre 4 introduit une première approche de sélection de gènes qui est basée sur une double exploration génétique afin d'obtenir des sous-ensembles de gènes de petite taille avec une bonne performance. La méthode proposée est une méthode

de type enveloppe (wrapper) qui est utilisée à la suite de l'étape de pré-sélection présentée dans le chapitre 3. La recherche d'un sous-ensemble de gènes "optimal" se fait en deux temps.

Une première exploration génétique d'un espace de grande taille permet de trouver des sous-ensembles de gènes qui garantissent une bonne performance en classification. Ces ensembles sont sauvegardés dans des archives qui sont ensuite analysées pour déterminer quels gènes interviennent avec la plus haute fréquence. Les gènes ainsi isolés sont la base d'une deuxième exploration génétique qui permet une recherche plus intensive pour trouver un sous-ensemble de petite taille avec une bonne performance. Les résultats de cette approche sont ensuite présentés et comparés avec les travaux similaires présentés dans la littérature.

- Le chapitre 5 présente une autre approche évolutionnaire pour la sélection et classification des gènes de puces à ADN. Nous utilisons une méthode intégrée (embedded) dans laquelle un algorithme génétique explore l'espace des sous-ensembles candidats tandis qu'un classifieur fournit l'évaluation des candidats mais aussi des informations permettant de guider le processus de recherche. Nous proposons notamment des opérateurs génétiques spécialisés qui utilisent l'information fournie par une analyse discriminante linéaire pour identifier les gènes pertinents. Plusieurs expériences sont menées pour valider cette approche et la comparer aux travaux présentés dans la littérature.

Le document se termine par une synthèse de nos différentes contributions, et présente également quelques perspectives de recherche.

Chapitre 1

Classification des données de puces à ADN

Sommaire

1.1	Technologie des puces à ADN	7
1.2	Jeux de données utilisés dans cette thèse	10
1.2.1	Leucémie	10
1.2.2	Cancer du Colon	11
1.2.3	Tumeurs du Cerveau	12
1.2.4	Lymphome	12
1.2.5	Cancer du Poumon	12
1.2.6	Cancer Ovarien	13
1.2.7	Cancer du Sein	13
1.2.8	Cancer de la Prostate	13
1.3	La classification supervisée	14
1.3.1	Notions de classification	14
1.3.2	Le problème de la généralisation	15
1.3.3	Évaluation d'une hypothèse de classification	16
1.4	Méthodes de classification supervisée	16
1.4.1	Méthodes à noyau et SVM	16
1.4.2	Classifieur bayésien naïf	17
1.4.3	k-PPV	18
1.4.4	Réseaux de neurones	18
1.4.5	Arbres de décision	19
1.4.6	Forêts aléatoires	19
1.4.7	Bagging	20
1.4.8	Boosting	20
1.5	Méthodes de classification non-supervisée	21
1.5.1	K-moyennes	22

Chapitre 1. Classification des données de puces à ADN

1.5.2	ISODATA	22
1.5.3	Classification hiérarchique	22
1.5.4	Cartes auto-organisatrices	23
1.6	Synthèse du chapitre	24

1.1 Technologie des puces à ADN

La technologie des puces à ADN ou biopuces, connaît à l'heure actuelle un essor exceptionnel et suscite un formidable intérêt dans la communauté scientifique. Cette technologie a été développée au début des années 1990 et permet la mesure simultanée des niveaux d'expression de plusieurs milliers de gènes, voire d'un génome entier, dans des dizaines de conditions différentes, physiologiques ou pathologiques. L'utilité de ces informations est scientifiquement incontestable car la connaissance du niveau d'expression d'un gène dans ces différentes situations constitue une avancée vers sa fonction, mais également vers le criblage de nouvelles molécules et l'identification de nouveaux médicaments et de nouveaux outils de diagnostic. [Lander, 1999; Speed, 2000; Baldi et Brunak, 2001; Godoy, 2001; Grody, 2001; Dudoit *et al.*, 2002; Dubitzky *et al.*, 2003; Stekel, 2003; Knudsen, 2004; Schoemaker et Lin, 2005]. Le fonctionnement des puces à ADN repose sur le principe de complémentarité des brins de la double hélice d'ADN et la propriété d'hybridation entre deux séquences complémentaires d'acides nucléiques [Lander, 1999; Southern, 2001a; Soularue et Gidrol, 2002; Dubitzky *et al.*, 2003]: une séquence d'ADN ou d'ARN peut donc servir de **sonde** pour capturer son complémentaire (**cible**) dans un mélange d'acides nucléiques

Une puce ADN (appelée *DNA microarray* en anglais) est constituée de fragments d'ADN immobilisés sur un support solide, de manière ordonnée. Chaque emplacement de séquence est soigneusement repéré: la position (x_i, y_i) correspond au gène i . Un emplacement est souvent appelé *spot* ou *sonde*. L'hybridation de la puce avec un échantillon biologique qui a été marqué par une substance radioactive ou fluorescente permet de quantifier l'ensemble des *cibles* qu'il contient; l'intensité du signal émis est proportionnel à la quantité de gènes cibles qu'il contient.

Les différentes phases d'une analyse par puces ADN sont indiquées dans la figure 1.1.



Figure 1.1 – Étapes d'une analyse par puces à ADN.

La préparation des cibles et l'hybridation : pour comparer les niveaux d'expression dans deux échantillons biologiques ou deux conditions (référence et pathologique), la première étape consiste en la préparation du génome exprimé dans ces deux échantillons. Il s'agit d'extraire les ARNm d'un échantillon biologique à analyser et la qualité de l'extraction est bien sûr primordiale pour la réussite de l'hybridation qui va suivre. Une mauvaise purification peut conduire à une augmentation des bruits de fond sur la lame. La deuxième étape consiste à marquer les deux échantillons pour ensuite les hybrider en utilisant un four et à les nettoyer en utilisant une station de lavage. Les

échantillons sont marqués par des substances fluorescentes (Cy3 et Cy5), c'est-à-dire qu'une culture est marquée avec un fluorochrome vert, tandis que la seconde est marquée avec un fluorochrome rouge. L'hybridation est ensuite réalisée sur une seule puce (simple marquage) ou sur deux puces (double marquage : un échantillon sur chaque puce). Les ADN marqués sont mélangés (cible) et placés sur la puce à ADN (sonde). Ce processus d'hybridation est réalisé dans une station fluïdique (four) pour favoriser les liaisons entre séquences complémentaires [Southern, 2001b; Soularue et Gidrol, 2002; Stekel, 2003]. La durée oscille entre 10 à 17 heures en milieu liquide à 60 degrés, en fait à cette température un fragment d'ADN simple brin ou d'ARN messenger reconnaît son brin complémentaire (ADNc) parmi des milliers d'autres pour former un ADN de double brin (duplex ou double hélice). L'étape de nettoyage ou lavage des puces a pour but d'ôter de la puce des cibles non hybridées. La puce est lavée à plusieurs reprises afin qu'il ne reste sur la lame que les brins parfaitement appariés.

Acquisition et analyse des images : suite à l'hybridation, une étape de lecture de la puce permet de repérer les sondes ayant réagi avec l'échantillon testé. Cette lecture est une étape clé [Southern, 2001a]. En effet, sa qualité conditionne de façon importante la précision des données et donc, la pertinence des interprétations. L'obtention des images est réalisée par lecture des puces sur des scanners de haute précision, adaptés aux marqueurs utilisés. Le procédé de détection combine deux lasers, pour exciter les fluorochromes *Cy3* et *Cy5*. On obtient alors deux images dont le niveau de gris représente l'intensité de la fluorescence lue. Si on remplace les niveaux de gris par des niveaux de vert pour la première image et des niveaux de rouge pour la seconde, on obtient en les superposant une image en fausses couleurs composée de spots allant du vert au rouge quand un des fluorophores domine, en passant par le jaune (même intensité pour les deux fluorophores). Le noir symbolise l'absence de signal. L'intensité du signal de fluorescence pour chaque couple (gène,spot) est proportionnel à l'intensité d'hybridation donc à l'expression du gène ciblé. Les images sont traitées par des logiciels d'analyse qui permettent de mesurer la fluorescence de chaque spot sur la lame (estimant les niveaux d'expression pour chacun des gènes présents sur la puce), mais aussi de relier chaque sonde à l'annotation correspondante (nom de gène, numéro de l'ADNc utilisé, séquence de l'oligonucléotide, etc.). Ainsi, pour chaque spot, l'intensité de chaque marqueur est calculée puis comparée au bruit de fond. **Transformation des données** : les rapports des intensités de fluorescences en rouge et vert sont généralement utilisés pour mesurer une variation d'expression d'un gène entre deux conditions (référence et pathologique, par exemple). Les données d'intensité sont rarement manipulées sans transformation et la transformation la plus couramment employée est celle qui utilise le logarithme à base deux. Il existe plusieurs raisons pour justifier cette transformation. D'une part, la variation du logarithme des intensités est moins dépendante de la grandeur des intensités, et d'autre part, cette transformation permet de se rapprocher d'une distribution symétrique et d'obtenir une meilleure dispersion avec moins de valeurs extrêmes, dans [Dudoit *et al.*, 2002] les auteurs utilisent cette transformation. La normalisation consiste à ajuster l'intensité globale des images acquises sur chacun des deux canaux rouge et vert, de manière à corriger les différences systématiques entre les échantillons sur la même lame, qui ne représentent pas de variations biologiques entre les échantillons et qui tendent à déséqui-

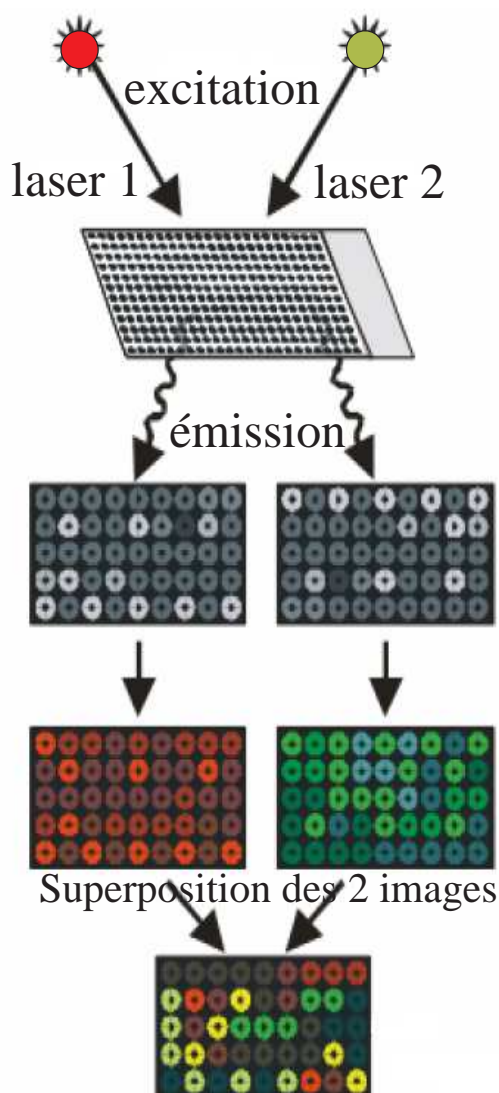


Figure 1.2 – Acquisition d’image dans une analyse par puces à ADN.

librer le signal de l’un des canaux par rapport à l’autre. Cette procédure de normalisation est définie par les gènes de référence. Les gènes de référence en moyenne ne doivent pas changer d’expression entre deux conditions.

La normalisation est effectuée à partir de toutes les sondes présentes sur le support pour éliminer les différences entre les différentes puces liées aux variations de quantité de départ, aux biais de marquage ou d’hybridation et aux variations du bruit de fond [Southern, 2001a; Yang *et al.*, 2002; Stekel, 2003; Jiang *et al.*, 2008; Stafford, 2008].

Présentation des données de puces à ADN : après les transformations décrites ci-

dessus, les données recueillies pour l'étude d'un problème donné sont regroupées sous forme de matrice avec une ligne par couple (gène, sonde) et une Colonne par échantillon (voir table 1.1). Chaque valeur de m_{ij} est la mesure du niveau d'expression du i -ième gène dans le j -ème échantillon, où $i = 1, \dots, M$ et $j = 1, \dots, N$ [Dudoit *et al.*, 2002; Dubitzky *et al.*, 2003].

$G\grave{e}ne_{id}$	$\acute{E}chantillon_1$	$\acute{E}chantillon_2$	\dots	$\acute{E}chantillon_M$
$G\grave{e}ne_1$	m_{11}	m_{12}	\dots	m_{1N}
$G\grave{e}ne_2$	m_{21}	m_{22}	\dots	m_{2N}
$G\grave{e}ne_3$	m_{31}	m_{32}	\dots	m_{3N}
\vdots	\vdots	\vdots	\ddots	\vdots
$G\grave{e}ne_N$	m_{M1}	m_{M2}	\dots	m_{MN}

Table 1.1 – Matrice d'expression des gènes.

Dans les étapes que l'on vient de voir, plusieurs d'entre elles peuvent être source d'imprécision ou d'erreurs dans les mesures obtenues. De plus, le coût d'une puce à ADN et le coût d'une analyse étant très élevé l'on ne dispose à l'heure actuelle que de quelques dizaines d'expériences pour l'étude d'un problème donné (une pathologie par exemple). Pourtant chaque expérience a permis de relever le niveau d'expression pour plusieurs milliers de gènes. Les matrices de données qui sont actuellement disponibles ont donc les caractéristiques suivantes :

1. Grande dimensionnalité due au nombre élevé de descripteurs (gènes)
2. Nombre limité d'échantillons

1.2 Jeux de données utilisés dans cette thèse

Dans la suite de ce document, nous allons présenter des méthodes de sélection d'attributs que nous avons développées sur un certain nombre de jeux de données. Nous avons utilisé des jeux de données publics, facilement accessibles et qui sont utilisés dans de nombreux travaux concernant la classification des données de puces à ADN. Ces jeux constituent en quelque sorte des jeux tests qui permettent de comparer les méthodes proposées depuis quelques années dans le domaine de la bioinformatique. Nous donnons ci-dessous les caractéristiques de ces jeux de données qui concernent tous des problèmes de reconnaissance de cancers et ou de prévision de diagnostic en oncologie.

1.2.1 Leucémie

Le jeu de données appelé dans la suite "données de la leucémie" ou "leucémie" est constitué de 72 échantillons représentant deux types de Leucémie aiguë. 47 tissus sont du type Leucémie lymphoblastique aiguë (ALL) et 25 sont du type Leucémie myéloïde aiguë (AML). Pour chaque échantillon, on a relevé les niveaux d'expression de 7129 gènes. Une

1.2 Jeux de données utilisés dans cette thèse

description plus complète de ce jeu de données peut être trouvée dans l'article [Golub *et al.*, 1999] et les données peuvent être téléchargées à partir de :

<http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi>

sous la rubrique "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression".

L'article [Golub *et al.*, 1999], publié en 1999, a été le premier à démontrer la possibilité d'un diagnostic à partir de données de puces à ADN et a donc été à l'origine d'un intérêt particulier pour cette problématique de la classification à partir des données de biopuces. C'est pourquoi nous résumons ici les principes de cette étude.

Dans cette recherche, Golub et ses collègues ont d'abord considéré 38 échantillons de moëlle osseuse pour 27 patients atteints de la forme ALL et 11 patients atteints de la forme AML de la Leucémie. La distinction entre ces deux formes de la maladie est bien connue mais nécessite un ensemble de tests de hauts niveaux (morphologie de la tumeur, histologie, ...) qui doivent être réalisés dans différents laboratoires spécialisés. L'objectif de ce travail était de voir si les seules données d'expression de gènes pouvaient conduire à une classification précise des Leucémies. Pour cela, Golub et son équipe ont d'abord identifié un ensemble de gènes corrélés à la distinction de classe ALL/AML. Ils ont retenu les 50 gènes les plus pertinents pour construire un système de prédiction permettant d'affecter un nouvel échantillon à la classe ALL ou AML. Le classifieur proposé était en fait la combinaison de 50 votes, chaque gène votant, d'après son niveau d'expression, pour l'une ou l'autre classe. Ce classifieur a été appliqué à un jeu de test de 34 échantillons issus de prélèvement de moëlle osseuse ou de sang. Pour 29 de ces 34 cas, le classifieur a proposé des prédictions "fortes" (convergence des votes des différents gènes) et ces 29 prédictions étaient correctes. Cette étude a donc montré que l'information issue des puces à ADN était suffisante pour construire un système de diagnostic précis, et que de plus un petit nombre de gènes suffisait à construire un prédicteur fiable.

Signalons enfin que généralement, avant toute analyse statistique, les données "brutes" de ce jeu de données sont pré-traitées selon un protocole comportant une étape de seuillage, de filtrage et de transformation logarithmique [Dudoit *et al.*, 2002]. Le seuillage consiste à ne conserver que les valeurs comprises entre 100 et 16000. Le filtrage a pour but d'éliminer avant tout traitement les gènes dont l'expression est trop uniforme et dont la variation n'est pas significative par rapport à la de mesure des niveaux d'expression. Pour chaque gène, on relève la valeur minimale s_{min} et la valeur maximale s_{max} du niveau d'expression parmi les tissus disponibles et on ne garde que les gènes tels que $s_{max}/s_{min} > 5$ et $s_{max}-s_{min} > 500$. Et on effectue une transformation logarithmique en base 10 des données.

1.2.2 Cancer du Colon

Ce jeu de données qui concerne le cancer du Colon, est constitué de 62 échantillons dont 40 sont des tissus tumoraux et 22 sont des tissus sains ou normaux. Les expériences ont été menées avec des puces relevant les valeurs d'expression pour plus de 6500 gènes

humains mais seuls les 2000 gènes ayant les plus fortes intensités minimales ont été retenus. La matrice des niveaux d'expression comporte donc 2000 Colonnes et 62 lignes. Dans le papier [Alon *et al.*, 1999], les auteurs utilisaient un jeu d'apprentissage de 40 échantillons (26 tumeurs, 14 normaux) et un jeu de test de 22 cas (14 tumeurs, 8 normaux). Ce jeu de données peut être téléchargé à l'adresse:

<http://microarray.princeton.edu/oncology/affydata/index.html>

1.2.3 Tumeurs du Cerveau

Cette base de données relative aux tumeurs embryonnaires du système nerveux central ou CNS a été utilisée par Pomeroy et al [Pomeroy *et al.*, 2002]. Ce jeu contient les échantillons obtenus de 60 patients qui ont été traités pour des médulloblastomes. Parmi ces 60 patients, 39 sont survivants et pour les 21 autres le traitement est un échec. Chaque échantillon est décrit par 7129 gènes. Le jeu de données est accessible à :

<http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi>

sous la rubrique "Gene Expression-Based Classification and Outcome Prediction of Central Nervous System Embryonal Tumors"

1.2.4 Lymphome

Ce jeu de données provient du site <http://llmpp.nih.gov/lymphoma/data.shtml>, associé à la publication de Alizadeh et al. [Alizadeh *et al.*, 2000]. Il contient les informations médicales de 96 échantillons de lymphocytes relatifs à différents types de Lymphome. Dans la littérature, on trouve des études regroupant les échantillons de deux manières différentes.

1. 62 échantillons de lymphocytes malins (dont 42 proviennent de Lymphomes diffus à grandes cellules B ou DLBCL, 9 sont des Lymphomes folliculaires ou FL et 11 sont des Lymphomes diffus lymphocytiques de type Leucémie lymphoïde chronique ou CLL) et 34 échantillons de lymphocytes normaux. Ce groupement a été proposé par Weston et al. et également utilisé par Rakotomamonjy [Weston *et al.*, 2002; Rakotomamonjy, 2003]. Il est disponible sur le site : <http://www.kyb.tuebingen.mpg.de/bs/people/weston/10/>
2. 24 échantillons de DLBCL de cellules de centre germinatif (*GC B-like*) et 23 échantillons de DLBCL activés (*activated B-like*). Ce groupement a été identifié par Alizadeh et al.

Ce jeu contient des données manquantes. Pour le traiter, nous avons remplacé les valeurs manquantes selon la méthode suggérée par [Troyanskaya *et al.*, 2001].

1.2.5 Cancer du Poumon

Le jeu de données concernant le cancer du Poumon, il a été traité par [Gordon *et al.*, 2002]. Ce jeu décrit deux types de pathologie du cancer du Poumon: le cancer de

1.2 Jeux de données utilisés dans cette thèse

type adénocarcinome *ADCA* et le cancer du mésothéliome malin de la plèvre *MPM*. Le jeu contient 181 instances décrites pour 12533 gènes. Les instances sont généralement divisées en 16 *MPM* et 16 *ADCM* pour l'apprentissage, et 15 *MPM* et 134 *ADCA* pour le test.

<http://www-genome.wi.mit.edu/cgi-bin/cancer/datasets.cgi>

1.2.6 Cancer Ovarien

Le jeu de données du cancer Ovarien comporte 253 échantillons dont 91 dits normaux (sans maladie) et 162 avec cancer. Ces données ont été obtenues par spectrométrie à partir d'échantillons de sérum de femmes saines et de sérum de femmes atteintes d'un cancer de l'ovaire. Chaque spectre représente l'expression de 15154 peptides. Pour une description complète consulter [Petricoin *et al.*, 2002]. Lorsque nous avons téléchargé ce jeu de données, il était disponible à l'adresse suivante :

<http://research.i2r.a-star.edu.sg/rp/>

1.2.7 Cancer du Sein

L'équipe de Van't Veer a abordé le problème du sur-traitement des patientes par la chimiothérapie adjuvante du cancer du Sein [Veer *et al.*, 2002]. Il existe des patientes ayant une tumeur du Sein sans envahissement ganglionnaire axillaire (N^-) et traitées par chimiothérapie, qui survivraient avec un traitement loco-régional seul. Cependant, l'imperfection des facteurs pronostiques actuels et le bénéfice apporté par cette chimiothérapie font qu'il faut sur-traiter de nombreuses patientes. L'objectif est de différencier, parmi les tumeurs N^- , celles qui ne vont pas rechuter de celles qui vont rechuter. La base d'apprentissage contient 78 échantillons de patientes, dont 34 avaient développé des métastases dans les 5 ans (rechute) et 44 étaient indemnes au-delà de 5 ans (non-rechute). La base de test contient 12 patientes avec rechute et 7 patientes avec non rechute. Le nombre de gènes utilisés dans cette étude étaient 24481. Nous remarquons qu'il existe 283 gènes avec bruit qui ont été enlevés. Ce jeu de données est disponible à l'adresse :

<http://homes.esat.kuleuven.be/~npochet/Bioinformatics/>

1.2.8 Cancer de la Prostate

Dans ce jeu de données, le niveau d'expression de 12600 gènes est mesuré sur 102 tissus. L'objectif initial est de distinguer les tissus normaux (52) des tissus cancéreux (50). Pour une description complète de ce jeu de données consulter [Singh *et al.*, 2002a]. Ce jeu de données est disponible à l'adresse :

<http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi>

sous la rubrique "Gene Expression Correlates of Clinical Prostate Cancer Behavior".

Nous venons de décrire les jeux de données sur lesquelles nous allons travailler. L'analyse de ces données a pour but de mettre en évidence l'importance de certains gènes dans les pathologies considérées et de construire un système de diagnostic s'appuyant uniquement sur les données de puces à ADN. Il s'agit là d'un problème de *classification supervisée* dont nous allons rappeler dans la section suivante les points utiles pour notre étude.

1.3 La classification supervisée

Le terme de classification peut désigner deux approches distinctes : la classification supervisée et la classification non-supervisée (automatic classification et clustering en anglais). Les méthodes non supervisées ont pour but de constituer des groupes d'exemples (ou des groupes d'attributs) en fonction des données observées, sans connaissance a priori. En revanche les méthodes supervisées utilisent la connaissance a priori sur l'appartenance d'un exemple à une classe pour construire un système de reconnaissance de ces classes. L'objectif de la classification supervisée est d'apprendre à l'aide d'un ensemble d'entraînement une procédure de classification qui permet de prédire l'appartenance d'un nouvel exemple à une classe. Les systèmes d'apprentissage permettant d'obtenir une telle procédure peuvent être basés sur des hypothèses probabilistes (classifieur naïf de Bayes), sur des notions de proximité (plus proches voisins) ou sur des recherches dans des espaces d'hypothèses (arbres de décisions, ...) Nous décrivons dans la suite quelques méthodes de classification supervisée bien connues dans la littérature. Nous n'aborderons ici que les méthodes de classification utilisées dans les travaux concernant la classification des données de puces à ADN.

1.3.1 Notions de classification

Nous rappelons tout d'abord une définition du problème de classification supervisée ou reconnaissances de formes. L'objectif de la classification est d'identifier les classes auxquelles appartiennent des objets à partir de leurs caractéristiques ou attributs descriptifs. Cette section emprunte des notations et différents éléments aux livres [Cristianini et Shawe-Taylor, 2000; Cornuéjols et Miclet, 2002] auxquels nous invitons le lecteur à se reporter.

Dans le cadre de la classification supervisée, les classes sont connues et l'on dispose d'exemples de chaque classe.

Définition 1.1 (Exemple) *Un exemple est un couple (x, \mathbf{y}) , où $x \in \mathcal{X}$ est la description ou la représentation de l'objet et $\mathbf{y} \in \mathcal{Y}$ représente la supervision de x .*

Dans un problème de classification, \mathbf{y} s'appelle la classe de x . Pour la classification binaire nous utilisons typiquement \mathcal{X} pour dénoter l'espace d'entrées tel que $\mathcal{X} \subseteq \mathbb{R}^p$ et \mathcal{Y} l'espace de sortie tel que $\mathcal{Y} = \{-1, 1\}$ ¹.

¹Parfois $\mathcal{Y} = \{+, -\}$, $\mathcal{Y} = \{1, 0\}$, $\mathcal{Y} = \{1, 2\}$

Définition 1.2 (Classification supervisée) Soit un ensemble d'exemples de n données étiquetées : $\mathcal{S} = \{(x_1, y_1), \dots, (x_n, y_n)\}$. Chaque donnée x_i est caractérisée par p attributs et par sa classe y_i . On cherche une hypothèse h telle que :

1. h satisfait les échantillons

$$\forall_i \in \{1, \dots, n\} h(x_i) = y_i$$

2. h possède de bonnes propriétés de généralisation.

Le problème de la classification consiste donc, en s'appuyant sur l'ensemble d'exemples à prédire la classe de toute nouvelle donnée $x \in \mathbb{R}^p$.

1.3.2 Le problème de la généralisation

L'objectif de la classification est de fournir une procédure ayant un bon pouvoir prédictif c'est-à-dire garantissant des prédictions fiables sur les nouveaux exemples qui seront soumis au système. La qualité prédictive d'un modèle peut être évaluée par le risque réel ou espérance du risque, qui mesure la probabilité de mauvaise classification d'un hypothèse h .

Définition 1.3 (Risque réel) Soit h une hypothèse apprise à partir d'un échantillon \mathcal{S} d'exemples de $\mathcal{X} \times \mathcal{Y}$

$$R(h) = \int_{x \in \mathcal{X}, Y} l[h(x_i), y_i] dF(x, y) \quad (1.1)$$

où l est une fonction de perte ou de coût associé aux mauvaises classifications et où l'intégrale prend en compte la distribution F de l'ensemble des exemples sur le produit cartésien de $\mathcal{X} \times \mathcal{Y}$.

La fonction de perte la plus simple utilisée en classification est définie par :

$$l[h(x_i), y_i] = \begin{cases} 0 & \text{si } y_i = h(x_i), \\ 1 & \text{si } y_i \neq h(x_i), \end{cases}$$

La distribution des exemples est inconnue, ce qui rend impossible le calcul du risque réel. Le système d'apprentissage n'a en fait accès qu'à l'erreur apparente qui est mesurée sur l'échantillon d'apprentissage.

Définition 1.4 (Risque empirique) Soit un ensemble d'apprentissage $\mathcal{S} = \{(x_1, y_1), \dots, (x_n, y_n)\}$ de taille n et une hypothèse h . Le risque empirique de h calculé sur \mathcal{S} est défini par :

$$R_{emp} = \frac{1}{n} \sum_{i=1}^n l[h(x_i), y_i] \quad (1.2)$$

Avec la fonction de perte présentée ci-dessus, le risque empirique ou apparent est simplement le nombre d'exemples de \mathcal{S} qui sont mal classés.

On peut montrer que, lorsque la taille de l'échantillon tend vers l'infini, le risque apparent converge -en probabilité si les éléments de \mathcal{S} sont tirés aléatoirement- vers le risque réel. Malheureusement on ne dispose que d'un échantillon limité d'exemples; le risque empirique est très optimiste et n'est pas un bon indicateur des performances prédictives de l'hypothèse h .

1.3.3 Évaluation d'une hypothèse de classification

Pour avoir une estimation non optimiste de l'erreur de classification, il faut recourir à une base d'exemples qui n'ont pas servi pour l'apprentissage : il s'agit de la base de test. La base de test contient elle aussi des exemples étiquetés qui permettent de comparer les prédictions d'une hypothèse h avec la valeur réelle de la classe. Cette base de test est généralement obtenue en réservant une partie des exemples supervisés initiaux et en ne les utilisant pas pour la phase d'apprentissage. Lorsqu'on dispose de peu d'exemples, comme c'est le cas dans le traitement des données d'expression de gènes, il est pénalisant de laisser de côté une partie des exemples pendant la phase d'apprentissage. On peut alors utiliser le processus de *validation croisée* pour proposer une estimation du risque réel. L'algorithme de validation croisée à k blocs (k -fold cross-validation) consiste à découper l'ensemble initial d'exemples \mathcal{D} en k blocs. On répète alors k phases d'apprentissage-évaluation où une hypothèse h est obtenue par apprentissage sur $k - 1$ blocs de données et testée sur le bloc restant. L'estimateur de l'erreur est obtenu comme la moyenne des k erreurs empiriques ainsi obtenues. L'algorithme est alors :

1. Partitionner l'ensemble d'exemples \mathcal{D} en k sous-ensembles disjoints: $\mathcal{D}_1, \dots, \mathcal{D}_k$
2. Pour tout i de 1 à k
 - Appliquer l'algorithme d'apprentissage sur le jeu d'apprentissage $\mathcal{D} - \mathcal{D}_i$ pour obtenir une hypothèse h_i
 - Calculer R_i l'erreur de h_i sur \mathcal{D}_i
3. Retourner $R = \frac{\sum_{1 \leq i \leq k} R_i}{k}$ comme estimation de l'erreur

Même s'il n'existe pas pour cela de justifications théoriques claires, l'usage montre que l'évaluation par validation croisée fournit de bons résultats pour $k=10$. Il faut noter que lorsque le nombre d'échantillons dont on dispose est limité on peut également appliquer le processus appelé *leave-one-out cross validation* où la validation croisée est appliquée avec $k = n$ le nombre d'échantillons.

Nous utiliserons la procédure de validation croisée dans la suite de cette thèse pour évaluer les classifieurs que nous mettrons en oeuvre sur les jeux de données présentés plus haut. Nous précisons à la fin du chapitre suivant le protocole d'expérimentation à respecter pour évaluer à la fois une méthode de sélection d'attributs et un classifieur.

1.4 Méthodes de classification supervisée

1.4.1 Méthodes à noyau et SVM

Les Séparateurs à Vaste Marge ou Support Vector Machines (SVMs) ont été introduits dès 1992 [Boser *et al.*, 1992; Cortes et Vapnik, 1995; Vapnik, 1998]. Les SVMs sont des classifieurs qui reposent sur deux idées clés : la notion de marge maximale et la notion de fonction noyau.

La première idée clé est la notion de marge maximale. On cherche l'hyperplan qui sépare les exemples positifs des exemples négatifs, en garantissant que la distance entre

la frontière de séparation et les échantillons les plus proches (marge) soit maximale. Ces derniers sont appelés vecteurs supports. Ceci est fait en formulant le problème comme un problème d'optimisation quadratique, pour lequel il existe des algorithmes connus.

Afin de pouvoir traiter des cas où les données ne sont pas linéairement séparables, la deuxième idée clé des SVM est de transformer l'espace de représentation des données d'entrées en un espace de plus grande dimension appelée espace de caractéristiques (possiblement de dimension infinie), dans lequel il est probable qu'il existe une séparatrice linéaire. Ceci est réalisé grâce à une fonction noyau, qui doit respecter certaines conditions, et qui a l'avantage de ne pas nécessiter la connaissance explicite de la transformation à appliquer pour le changement d'espace. Les fonctions noyau permettent de transformer un produit scalaire dans un espace de grande dimension, ce qui est coûteux, en une simple évaluation ponctuelle d'une fonction. Pour plus détails voir notamment dans [Scholkopf et Smola, 2001; Cristianini et Shawe-Taylor, 2000; Burges, 1998]

Points positifs : les caractéristiques clés des SVMs sont l'utilisation des noyaux, l'absence de minimums locaux, la faible quantité de la solution et le contrôle de capacité obtenu en optimisant la marge.

Point négatif : le réglage du paramétrage tels que l'influence du noyau contrôlée par b (la largeur de bande), la complexité du modèle contrôlée par le paramètre C et le critère d'arrêt du solveur utilisé ou un paramètre de pré-conditionnement associé au système linéaire à résoudre.

1.4.2 Classifieur bayésien naïf

Le classifieur bayésien naïf est une méthode qui trouve son fondement théorique dans le théorème de Bayes [Mitchell, 1997; Cornuéjols et Miclet, 2002; Wu *et al.*, 2008]. Il permet les inférences probabilistes et repose sur l'hypothèse que les solutions recherchées peuvent être trouvées à partir de distributions de probabilité dans les données et dans les hypothèses. Cette méthode permet de déterminer la classification d'un exemple quelconque spécifiée en termes d'attributs en supposant que les attributs $\{x_1, \dots, x_n\}$ de l'espace d'entrée X sont indépendants les uns des autres et $y \in Y$ tel que $Y = \{0, 1\}$ pour la classification binaire. La règle de classification de Bayes s'écrit :

$$C_{NBayes}(x) = \operatorname{argmax}_{y \in Y} P(x_1, \dots, x_n | y) \times P(y) \quad (1.3)$$

On peut remplacer $P(x_1, \dots, x_n)$ et $P(y)$ par des estimations faites sur l'ensemble d'échantillons X (telles que loi de Bernoulli, normale ou bien d'autres). Pour toute classe y on estime $P'(y)$ par la proportion d'éléments de la classe y dans X . Étant donné que l'estimation des $P(x_1, \dots, x_n)$ n'est pas évidente car le nombre de descriptions possibles peut être grand, il faudrait un échantillon X de taille trop importante pour pouvoir estimer correctement ces quantités. Pour cela on utilise l'hypothèse suivante : les valeurs des attributs sont indépendantes connaissant la classe. Cette hypothèse permet d'utiliser l'égalité suivante :

$$P(x_1, \dots, x_n | y) = \prod_{y \in Y} P(x_i | y) \quad (1.4)$$

Pour cela il suffit d'estimer, pour tout i et toute classe y , $P'(x_i|y)$ par la proportion d'éléments de classe y ayant la valeur x_i pour le i -ème attribut. Finalement, le classifieur naïf de Bayes associé à toute description x la classe :

$$C_{NBayes}(x) = \operatorname{argmax}_i \prod_i P'(x_1, \dots, x_n|y) \times P(y) \quad (1.5)$$

On peut voir les probabilités qui sont estimées sur l'ensemble d'échantillons X .

Points positifs : simple et facile à programmer, souvent efficace. Point négatif : très sensible à la présence d'attributs corrélés.

1.4.3 k-PPV

L'algorithme des k plus proches voisins (noté k-PPV) [Duda *et al.*, 2000; Bishop, 2006; Wu *et al.*, 2008] est une méthode basée sur la notion de proximité (voisinage) entre exemples et sur l'idée de raisonner à partir de cas similaires pour prendre une décision. Autrement dit des entrées x_i semblables devraient avoir des valeurs y_i semblables. Le principe est le suivant : on note x un nouvel exemple décrit par un vecteur de p attributs. On trouve alors, parmi l'ensemble d'exemples d'apprentissage, les k plus proches voisins de x et on associe à x la classe majoritaire parmi ses k voisins lui ressemblant le plus dans la base d'apprentissage. Cette méthode dépend donc des trois éléments suivants:

1. Le nombre de voisins retenus.
2. La mesure de distance entre exemple.
3. La combinaison des classes.

Le résultat dépend du réglage de ces paramètres. Pour le premier critère, on utilise généralement un nombre de voisins compris entre 1 et 7. Pour le deuxième paramètre, la méthode nécessite une métrique pour mesurer la proximité entre l'exemple à classifier x et chacun des exemples de l'ensemble d'apprentissage. Lorsque les attributs sont numériques la distance euclidienne est généralement utilisée. Le troisième paramètre indique de quelle manière on combine les valeurs associées aux voisins pour obtenir la valeur associée à x . la combinaison des classes des k plus proches voisins en une classe C . Pour la classification, la classe retenue pour x est la classe majoritaire chez ses voisins.

Les points positifs de cette méthode sont qu'elle ne pose aucune hypothèse sur la forme des classes à apprendre. La méthode est simple puisqu'il n'y a pas besoin d'apprentissage d'un modèle de classification et son pouvoir prédictif est souvent bon.

Mais la performance de cette méthode diminue lorsque la dimension augmente, puisque pour chaque nouvelle classification, il est nécessaire de calculer toutes les distances de x à chacun des exemples d'apprentissage. De plus, la performance dépend fortement de k , le nombre de voisins choisi et il est nécessaire d'avoir un grand nombre d'observations pour obtenir une bonne précision des résultats.

1.4.4 Réseaux de neurones

Le principe consiste à apprendre à classifier correctement des données à partir d'un jeu d'exemples déjà classifiés, c'est-à-dire l'apprentissage par l'expérience. Un réseau

de neurones est constitué d'un graphe pondéré orienté dont les noeuds symbolisent les neurones. Ces neurones possèdent une fonction d'activation qui permet d'influencer les autres neurones du réseau, les fonctions, les plus souvent utilisées, sont la fonction signe ou la fonction sigmoïde.

Les connexions entre les neurones, que l'on nomme liens synaptiques, propagent l'activité des neurones avec une pondération caractéristique de la connexion. On appelle poids synaptique la pondération des liens synaptiques.

Les neurones peuvent être organisés de différentes manières, c'est ce qui définit l'architecture et le modèle du réseau. L'architecture la plus courante est celle dite du perceptron multicouches.

Points positifs : excellentes performances, minimise la fréquence d'erreur. Point négatif : difficulté de mise en oeuvre.

1.4.5 Arbres de décision

Les arbres de décision sont une des méthodes les plus connues en classification. Le principe des arbres de décision est de réaliser la classification d'un exemple par une suite de tests sur les attributs qui le décrivent. Concrètement, dans la représentation graphique d'un arbre,

1. Un noeud interne correspond à un test sur la valeur d'un attribut.
2. Une branche part d'un noeud et correspond à une ou plusieurs valeurs de ce test.
3. Une feuille est un noeud d'où ne part aucune branche et correspond à une classe.

Une règle de décision (de la forme si ... alors ...) est créée pour chaque chemin partant de la racine de l'arbre et parcourant les tests (en faisant des conjonctions) jusqu'à la feuille qui est l'étiquette de la classe. Les arbres de décision sont particulièrement appréciés car ils permettent une compréhension aisée, mais lors d'une tâche de classification relativement complexe l'utilisateur n'est plus capable d'explorer efficacement les résultats obtenus sous forme textuelle.

Les principaux algorithmes de construction d'un arbre de décision sont : C4.5 [Quinlan, 1993], et CART [Breimann *et al.*, 1984]. Dans ces algorithmes, à chaque étape de création d'un noeud, un critère de séparation entre les classes est utilisé pour décider quel attribut est le plus pertinent pour la classification

1.4.6 Forêts aléatoires

Ils ont été introduits par [Breimann *et al.*, 1984] et sont une extension des arbres de décision avec des particularités. Notamment les feuilles de chaque arbre sont générées au hasard à partir des échantillons bootstrap et d'échantillons d'apprentissage. Il y a deux manières de faire la construction de forêts : soit par entrée aléatoire ou soit par variables aléatoires. Dans le premier cas une seule variable pour chaque règle de décision est utilisée. Dans le deuxième cas une combinaison linéaire des variables sélectionnés à chaque feuille est utilisée.

Afin de construire des forêts aléatoires il faut régler trois paramètres :

1. Nombre d'arbres.
2. Nombre de variables testées à chaque feuille d'un arbre.
3. Nombre d'observations minimal dans les feuilles des arbres.

La performance des forêts est comparable à celle des SVM, il n'y a pas besoin de réglage sophistiqué pour la construction de forêts aléatoires, utilisable même si $p > n$, il n'y a pas de risque de sur-apprentissage, les forêts aléatoires retournent toujours une mesure d'importance de chaque attribut cela permet de considérer une bonne sélection d'attributs. Elles sont plus performantes que les arbres de décision car elles exploitent mieux les échantillons de petites et grandes tailles.

1.4.7 Bagging

Cette méthode a été proposée par L. Breiman [Breiman, 1996]. Le principe du bagging est d'entraîner un algorithme d'apprentissage sur plusieurs bases d'apprentissage obtenues par tirage avec remise (ou bootstrap) de m' exemples d'apprentissage dans l'échantillon d'apprentissage S . Pour chaque tirage b , une hypothèse h_b est obtenue. L'hypothèse finale est basée sur les moyennes des hypothèses obtenues. Son avantage est qu'on améliore la performance des classifieurs instables en calculant la moyenne leurs réponses, ainsi, si les hypothèses h_b calculées pour chaque tirage b ont une variance importante, alors l'hypothèse finale aura une variance réduite. Un classifieur est dit instable si un petit changement dans les données d'apprentissage provoque un gros changement dans le comportement du classifieur. Le but du bagging est d'atténuer l'instabilité inhérente à certains classifieurs.

Le point positif : cette méthode est facile à mettre en place et s'adapte à tout type de classifieur. Les points négatifs : elle est coûteuse en temps de calcul, il faut stocker tous les classifieurs, une amélioration de la qualité de classification se fait au détriment de l'interprétation.

1.4.8 Boosting

Le boosting est une méthode d'agrégation de classifieurs faibles pour obtenir un classifieur performant [Schapire, 1990]. Son principe consiste à assigner un poids égal à chaque exemple d'apprentissage ($1/n$) (appelé pondération) où n est le nombre d'échantillons. Ce poids indique la difficulté de prédire la classe de cet exemple. L'algorithme s'exécute T fois construisant T classifieurs sur les exemples d'échantillons d'apprentissage pondérés. Chaque fois qu'on produit un classifieur on change les poids des nouveaux exemples utilisés pour le classifieur suivant, on augmente le poids de ceux dont la classe est mal prédite et on diminue le poids des autres. Ensuite, on calcule l'erreur (e) du modèle sur l'ensemble pondéré. Si e est égale à zéro ou e est supérieure à 0.5 on termine la construction du classifieur. L'idée est de forcer le nouveau classifieur à diminuer l'erreur attendue. Le classifieur final se forme en utilisant un schéma de vote.

Les points positifs : ce type de procédure permet de renforcer n'importe quel algorithme d'apprentissage, il faut régler un seul paramètre (le nombre T d'itérations), les

performances sont garanties, la procédure est immune contre le sur-apprentissage (l'erreur empirique sur l'ensemble d'apprentissage décroît exponentiellement avec le nombre d'itérations).

Les points négatifs : difficile d'incorporer des connaissances a priori, difficile de savoir comment régulariser, et les frontières de décision sont souvent très irrégulières (non interprétables).

1.5 Méthodes de classification non-supervisée

Le but de la classification non-supervisée (ou du clustering) est de créer un ensemble de clusters regroupant des données de mêmes caractéristiques ou semblables. Le regroupement de données est basé sur la notion de distance par rapport à des centroïdes, c'est-à-dire les centres de masse de chaque classe. Les classes sont a priori inconnues et sont créées selon la ressemblance des données afin d'expliquer ou résumer les données. Les principales méthodes de la classification non-supervisée sont les algorithmes de classification hiérarchique et les méthodes de regroupement.

Les méthodes de classification non-supervisée ont été appliquées à la classification des données de puces à ADN [Golub *et al.*, 1999; Alizadeh *et al.*, 2000; Jaeger *et al.*, 2003; Liu *et al.*, 2004; Fu *et al.*, 2006]. Ces travaux ont mis l'accent sur le classement des gènes en fonction de leurs profils d'expression pour mieux comprendre leur signification biologique et ainsi envisager un outil qui permet d'associer à ces nouvelles classifications une valeur pronostic d'une maladie sans avoir une connaissance du diagnostic au préalable.

Le principe du groupement est d'associer un profil d'expression à chaque gène, puis de trier ces profils en fonction de leur ressemblance et finalement d'utiliser de méthodes statistiques pour ainsi construire des groupes de gènes similaires à partir d'une matrice de distances.

Définition 1.5 (Clustering) Soit un ensemble d'exemples de n données : $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$. Un ensemble de M clusters de \mathcal{X} est une partition de \mathcal{X} dans M nuages (clusters) $\mathcal{C}_1, \dots, \mathcal{C}_M$ telle que :

- $\mathcal{C}_i \neq \emptyset, i = 1, \dots, M$
- $\cup_{i=1}^M \mathcal{C}_i = \mathcal{X}$
- $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset, i \neq j, j = 1, \dots, M$
- Les données de \mathcal{C}_i sont similaires entre elles et dissimilaires par rapport aux données de $\mathcal{C}_j (j \neq i)$ selon une mesure de similarité donnée.

Pour approfondir sur les techniques de clustering nous suggérons consulter [Jain *et al.*, 1999; Grabmeier et Rudolph, 2002; Xiu et Wunsch, 2005].

Nous présentons dans la suite quatre méthodes de classification non-supervisée dont deux non hiérarchique (k-moyennes et ISODATA) et deux hiérarchiques (les cartes auto-organisatrices et la classification hiérarchique).

1.5.1 K-moyennes

La méthode des k-moyennes est une méthode qui permet de partitionner ou segmenter les données en k nuages (appelés clusters) homogènes (c.f. Algorithme 1.1) [Theodoridis et Koutroumbas, 1999] :

Algorithme 1.1 : k-moyennes

```

1 Paramètre: le nombre de clusters à construire
2 début
3   Initialisation des  $k$  centroïdes avec les valeurs initiales
4   fin=faux
5   tant que non fin faire
6     pour chaque donnée faire
7       Trouver le centroïde le plus proche
8       Placer la donnée dans le cluster le plus proche;
9     fin
10    si aucun changement des valeurs des centroïdes alors
11      fin=vrai
12    fin
13    sinon
14      Calculer les nouveaux centroïdes
15    fin
16  fin
17 fin

```

Nous remarquons que dans le pas 2 normalement l'on prend k points dans l'espace de données de manière aléatoire. Le critère d'arrêt de l'algorithme k-moyennes est basé sur la stabilité des centres des classes. Une des limitations de cette approche est la difficulté de déterminer k a priori.

1.5.2 ISODATA

L'algorithme ISODATA (Iterative Self-Organizing Data Analyses Techniques) est un algorithme auto-organisateur et itératif parce qu'il effectue plusieurs itérations à travers l'ensemble de données, jusqu'à ce que les résultats spécifiés soient obtenus [Theodoridis et Koutroumbas, 1999]. Il s'agit d'une méthode de clustering dont la base repose sur l'algorithme de k-moyennes. Le but est de minimiser l'erreur quadratique en associant chaque donnée au centroïde le plus proche. Une caractéristique est d'éliminer les clusters avec peu d'éléments et regrouper des clusters qui sont trop proches (voir algorithme 1.2 ci-dessous).

1.5.3 Classification hiérarchique

La classification hiérarchique est une méthode d'agrégation qui consiste à regrouper les données dans de petits groupements qui sont eux-même hiérarchiquement assembles en groupements plus grands.

Algorithme 1.2 : ISODATA

```
Entrées :  $k$ 
1 début
2   Initialisation des  $k$  centroïdes avec les valeurs initiales
3   stabilisation=faux
4   tant que non stabilisation faire
5     Effectuer un groupement par l'algorithme k-moyennes
6     Éliminer les clusters trop petits
7     Scinder les clusters trop dispersés
8     Fusionner les clusters plus proches;
9   fin
10 fin
```

Cette technique d'agrégation peut être représentée par un arbre hiérarchique [Theodoridis et Koutroumbas, 1999]. Le plus petit regroupement se trouve dans la partie la plus basse de l'arbre et le plus grand dans la partie la plus haute de l'arbre, où chaque donnée est un groupe qui est progressivement absorbé par le groupe le plus proche.

Les méthodes hiérarchiques sont récursives, ces méthodes sont de deux types : 1) ascendantes, et 2) descendantes. Nous montrons la définition de la classification hiérarchique ascendante.

Définition 1.6 (Classification hiérarchique ascendante) Soit un ensemble d'exemples $\mathcal{E} = \{(\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_n)\}$. L'algorithme se déroule en les étapes suivantes :

1. On commence par une première partition des données, où chaque objet représente une classe.
2. On regroupe les deux classes les plus proches, pour constituer une nouvelle classe.
3. On recalcule la distance de cette classe avec les autres, selon un critère de distance.
4. On itère les pas 2 et 3 jusqu'à l'obtention d'une classe unique,

Les algorithmes de clustering utilisent une mesure de similarité pour grouper les différences entre les données.

1.5.4 Cartes auto-organisatrices

Les cartes auto-organisatrices de Kohonen (SOMs) sont des réseaux de neurones non-supervisés qui permettent d'ordonner et de classifier des données en fonction de leur similarité [Kohonen, 2001].

Le réseau est composé de deux couches (une couche d'entrée et une couche de sortie). La couche d'entrée est généralement de grande dimension et la couche de sortie (appelé espace des représentations) est de dimension réduite (la carte auto-organisatrice). La couche d'entrée contient un noeud (neurone) pour chaque donnée ou attribut qui est connecté de façon non-linéaire aux cellules géométriques de la couche de sortie.

Les neurones de la carte sont disposés géométriquement selon une topologie fixée a priori (e.g hexagonal ou rectangulaire). Chaque cellule géométrique de la matrice est connectée aux autres cellules avec une pondération (poids) qui décroît en fonction de

la distance relative entre celles-ci. L'ensemble de ces cellules forme la carte de Kohonen (couche de sortie) dont l'objectif est de trouver la projection entre les deux couches tout en conservant la topologie des données.

1.6 Synthèse du chapitre

Nous avons présenté dans ce chapitre les notions qui s'articulent autour de la technologie des puces à ADN et ainsi que les jeux de données basées sur cette technologie. Ensuite nous avons présenté les deux familles de classification, c'est-à-dire la classification supervisée et la classification non-supervisée. Nous avons introduit ces notions de classification, car nous considérons le problème de sélection d'attributs et de classification dans le domaine des données de puces à ADN.

Les différentes techniques de classification que nous avons décrit ont été utilisées dans différents travaux concernant le diagnostic à partir de données de puces à ADN et notamment sur les jeux de données particuliers que nous avons décrits.

Chapitre 2

Sélection d'attributs

Sommaire

2.1	Présentation informelle du problème	26
2.2	Notions de pertinence, non pertinence et redondance	27
2.2.1	Pertinence d'attributs	27
2.2.2	Redondance d'attributs	28
2.3	Sélection d'attributs	30
2.3.1	La sélection vue comme un problème d'optimisation	30
2.3.2	Schéma général de la sélection d'attributs	31
2.3.3	Génération de sous-ensembles et procédures de recherche	33
2.4	Les approches pour la sélection d'attributs	34
2.4.1	Méthodes de Filtre	34
2.4.2	Méthodes Enveloppes	36
2.4.3	Méthodes Intégrées	36
2.5	Validation des méthodes	36
2.6	Synthèse du chapitre	37

LA sélection d'attributs est devenue un sujet de recherche très actif depuis une dizaine d'années dans les domaines de l'apprentissage artificiel [Jain et Zongker, 1997; Dash et Liu, 1997; Kohavi et John, 1997; Blum et Langley, 1997; Duch, 2006], de la fouille de données, du traitement d'images [Singh *et al.*, 2002b; Fukunaga, 1990], et de l'analyse de données en bioinformatique. Dans tous ces domaines, les applications nécessitent de traiter des données décrites par un très grand nombre d'attributs. Ainsi on peut avoir à traiter des pages web décrites par plusieurs centaines de descripteurs, des images décrites par plusieurs milliers de pixels ou des données en bioinformatique donnant les niveaux d'expression de plusieurs milliers de gènes [Guyon et Elisseeff, 2003; Dai *et al.*, 2006]. [Guyon et Elisseeff, 2003; Dai *et al.*, 2006].

Dans ce chapitre, nous présentons d'abord de manière informelle le problème de la sélection d'attributs pour situer le travail et l'intérêt de la thèse. Dans la section 1, nous abordons les notions de pertinence, non-pertinence et redondance autour desquelles s'articulent la sélection d'attributs. Dans la section 2, nous présentons différents points de vue du processus de sélection d'attributs et la section 3 rappelle les différents types de méthodes pour la sélection d'attributs. Enfin dans la section 4, nous présentons le protocole expérimental qui doit être utilisé pour obtenir une estimation non biaisée d'une méthode de sélection.

2.1 Présentation informelle du problème

La sélection d'attributs consiste à choisir parmi un ensemble d'attributs de grande taille un sous-ensemble d'attributs intéressants pour le problème étudié. Cette problématique peut concerner différentes tâches d'apprentissage ou de fouille de données mais nous parlerons seulement ici de la sélection d'attributs réalisée pour la classification supervisée. Dans ce contexte, les principales motivations de la sélection d'attributs sont les suivantes [Jain et Zongker, 1997; Yang et Honovar, 1998] :

1. Utiliser un sous-ensemble plus petit permet d'améliorer la classification si l'on élimine les attributs qui sont source de bruit. Cela permet aussi une meilleure compréhension des phénomènes étudiés.
2. Des petits sous-ensembles d'attributs permettent une meilleure généralisation des données en évitant le sur-apprentissage.
3. Une fois que les meilleurs attributs sont identifiés, les temps d'apprentissage et d'exécution sont réduits et en conséquence l'apprentissage est moins coûteux.

En présence de centaines voire de milliers d'attributs il y a beaucoup de chances pour que des attributs soient corrélés et expriment des informations similaires, on dira alors qu'ils sont redondants. D'un autre côté, les attributs qui fournissent le plus d'information pour la classification seront dits pertinents. L'objectif de la sélection est donc de trouver un sous-ensemble optimal d'attributs qui ait les propriétés suivantes : il doit être composé d'attributs pertinents et il doit chercher à éviter les attributs redondants. De plus cet ensemble doit permettre de satisfaire au mieux l'objectif fixé c'est-à-dire la précision

2.2 Notions de pertinence, non pertinence et redondance

de l'apprentissage, la rapidité de l'apprentissage ou bien encore l'explicabilité du classifieur proposé [Dash et Liu, 1997; Kohavi et John, 1997; Blum et Langley, 1997; Guyon *et al.*, 2002; Yu et Liu, 2004b; Yu et Liu, 2004a; Molina *et al.*, 2002b; Molina *et al.*, 2002a; Liu et Motoda, 2008a].

On peut illustrer cette problématique par le petit exemple suivant: [Liu et Yu, 2005]

Exemple : Considérons un problème de classification, où les descripteurs sont 5 attributs booléens F_1, \dots, F_5 . Supposons que la fonction de classification s'écrit $C = g(F_1, F_2)$ où g est une fonction booléenne. De plus, nous supposons qu'il existe les relations suivantes entre les attributs: $F_2 = \bar{F}_3$ et $F_4 = \bar{F}_5$ (notre problème concerne donc huit exemples possibles). Afin de déterminer la fonction cible, F_1 est indispensable; un des attributs F_2 ou F_3 peut être enlevé, car on remarque que C peut également s'écrire $g(F_1, \bar{F}_3)$, mais un des attributs F_2 ou F_3 doit être utilisé pour définir C . Quant à F_4 et F_5 , ils peuvent être éliminés car ils ne sont pas utiles pour définir C . On peut donc sélectionner pour ce problème deux sous-ensembles d'attributs optimaux: $\{F_1, F_2\}$ ou $\{F_1, F_3\}$.

Nous voyons que les notions de pertinence et redondance jouent un rôle fondamental dans la sélection d'attributs et nous allons donc les détailler dans la section suivante.

2.2 Notions de pertinence, non pertinence et redondance

Nous présentons d'abord une formalisation de la pertinence des attributs qui permet de distinguer les différentes contributions d'un attribut à la définition d'une fonction de classification. De même, nous présentons une définition de la redondance qui permet de proposer un mécanisme d'élimination des attributs redondants.

2.2.1 Pertinence d'attributs

Dans [Liu et Yu, 2005], les auteurs définissent la pertinence dans le cas d'attributs et de fonctions booléennes, et en supposant que les données sont non bruitées. Une définition plus large proposée par [Kohavi et John, 1997] définit les attributs pertinents comme ceux dont les valeurs varient systématiquement avec les valeurs de classe. Autrement dit, un attribut F_i est pertinent si connaître sa valeur change les probabilités sur les valeurs de la classe C . Mais cette définition peut être précisée pour distinguer les attributs fortement pertinents et les attributs faiblement pertinents grâce aux définitions suivantes. Soit F un ensemble complet d'attributs, F_i un attribut, et $S_i = F - \{F_i\}$. On suppose que l'on travaille avec un espace probabilisé où la probabilité est notée P . $P(C|S)$ est la probabilité de la classe C connaissant les attributs de l'ensemble S .

Définition 2.1 Un attribut F_i est fortement pertinent ssi :

$$P(C|F_i, S_i) \neq P(C|S_i) \quad (2.1)$$

Définition 2.2 Un attribut F_i est faiblement pertinent ssi :

$$P(C|F_i, S_i) = P(C|S_i) \text{ et } \exists S'_i \subset S_i \text{ tel que } P(C|F_i, S'_i) \neq P(C|S'_i) \quad (2.2)$$

Définition 2.3 Un attribut F_i est non pertinent ssi :

$$P(C|F_i, S_i) = P(C|S_i) \text{ et } \forall S'_i \subseteq S_i (C|F_i, S'_i) = P(C|S'_i) \quad (2.3)$$

D'après ces définitions, les attributs fortement pertinents sont donc indispensables et devraient figurer dans tout sous-ensemble optimal sélectionné, car leur absence devrait conduire à un défaut de reconnaissance de la fonction cible.

La faible pertinence suggère que l'attribut n'est pas toujours important, mais il peut devenir nécessaire pour un sous-ensemble optimal dans certaines conditions.

La non-pertinence d'un attribut se définit simplement par rapport à 2.1 et 2.2 et indique qu'un attribut n'est pas du tout nécessaire dans un sous-ensemble optimal d'attributs. Si l'on revient sur l'exemple 2.1, on peut dire que l'attribut F_1 est fortement pertinent, que F_2, F_3 sont faiblement pertinents, et que F_4, F_5 sont non pertinents. Un sous-ensemble d'attributs optimal ne devrait inclure que les attributs fortement pertinents, aucun attribut non pertinent, et un sous-ensemble d'attributs faiblement pertinents.

Pour savoir comment choisir quels attributs faiblement pertinents on doit sélectionner, il faut mettre en évidence la notion de redondance.

2.2.2 Redondance d'attributs

La notion de la redondance d'attributs se comprend intuitivement et elle est généralement exprimée en termes de corrélation entre attributs. On peut dire que deux attributs sont redondants (entre eux) si leurs valeurs sont complètement corrélées (par exemple, les attributs F_2 et F_3 de l'exemple 1). Cette définition ne se généralise pas directement pour un sous-ensemble d'attributs. On trouve dans [Koller et Sahami, 1996], une définition formelle de la redondance qui permet de concevoir une approche pour identifier et éliminer les attributs redondants. Cette formalisation repose sur la notion de couverture de Markov (Markov blanket) d'un attribut qui permet d'identifier les attributs non pertinents et redondants [Koller et Sahami, 1996; Yu et Liu, 2004a].

Définition 2.4 . Soit F l'ensemble total d'attributs et C la classe. Soit F_i un attribut, et M_i un sous-ensemble d'attributs qui ne contient pas F_i , c'est-à-dire : $M_i \subseteq F$ et $F_i \notin M_i$.

M_i est une couverture de Markov pour F_i ssi

$$P(F - M_i - \{F_i\}, C|F_i, M_i) = P(F - M_i - \{F_i\}, C|M_i) \quad (2.4)$$

La définition de couverture de Markov impose que M_i subsume non seulement l'information que F_i apporte sur C mais aussi l'information qu'il apporte sur tous les autres attributs. Dans [Koller et Sahami, 1996], il est montré qu'un sous-ensemble d'attributs optimal peut être obtenu par une procédure d'élimination descendante, appelée filtrage par couverture de Markov (Markov blanket Filtering) et définie comme suit :

Filtrage par couverture de Markov :. Soit G l'ensemble d'attributs courant ($G = F$ au départ). A chaque étape de la procédure, s'il existe une couverture de Markov pour l'attribut F_i dans l'ensemble G courant, F_i est enlevé de G .

On peut montrer que ce processus garantit qu'un attribut enlevé dans une étape précédente peut trouver une couverture de Markov dans une étape postérieure.

2.2 Notions de pertinence, non pertinence et redondance

Selon les définitions précédentes de la pertinence d'attributs, on peut également montrer que les attributs fortement pertinents ne peuvent trouver aucune couverture de Markov. Par contre, les attributs non pertinents doivent être enlevés de toute façon, et il n'est donc pas nécessaire de s'y intéresser dans la définition des attributs redondants. Cela conduit à la définition suivante de la redondance [Yu et Liu, 2004a].

Définition 2.5 *Attribut! redondant*[Yu et Liu, 2004a]. Soit G l'ensemble d'attributs courant, un attribut F_i est redondant et par conséquent peut être enlevé de G ssi il est faiblement pertinent et qu'il possède une couverture de Markov dans G .

Afin de synthétiser les différentes notions de pertinence et redondance que l'on vient de présenter, on peut proposer une catégorisation des attributs présentée dans la Figure 2.1. Dans ce schéma, un ensemble initial d'attributs peut être partitionné en quatre catégories: attributs non pertinents (partie I), attributs redondants (partie II) (qui sont faiblement pertinents comme on l'a vu), attributs faiblement pertinents et non-redondants (partie III), et attributs fortement pertinents (partie IV). Un sous-ensemble d'attributs optimal contient essentiellement tous les attributs des parties III et IV. Il est important de préciser que pour un ensemble initial donné, le processus de filtrage par couverture de Markov peut conduire à différents découpages donnant les parties II et III (qui sont disjointes). Ainsi, dans l'exemple précédent 2.1, l'attribut F_2 ou l'attribut F_3 devraient être enlevés comme attributs redondants et donc figurer dans (II) mais pas les deux à la fois.

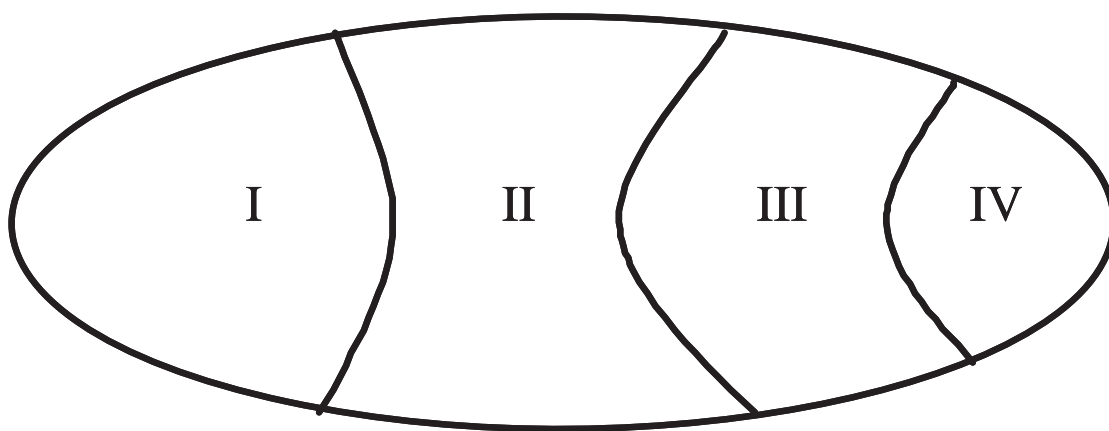


Figure 2.1 – Catégorisation d'attributs.

Une discussion plus générale au sujet des notions de pertinence, et de redondance d'attributs peut être trouvée dans : [John *et al.*, 1994; Jaeger *et al.*, 2003; Guyon et Elisseeff, 2003; Yu et Liu, 2004b; Yu et Liu, 2004a; Dash et Liu, 2006; Mundra et Rajapaks, 2007; Mao et Tang, 2007; Liu et Motoda, 2008b].

Les formalisations précédentes sont intéressantes pour mieux cerner les notions de redondance et de pertinence. Néanmoins les définitions probabilistiques que nous avons vues ne permettent pas de proposer un processus de sélection d'attributs applicable sur des données de grande dimension. Dans la suite de ce chapitre nous allons donc

présenter différents points de vue permettant de comprendre le processus de la sélection d'attributs, d'un point de vue plus opérationnel.

2.3 Sélection d'attributs

2.3.1 La sélection vue comme un problème d'optimisation

Le problème de la sélection d'un sous-ensemble d'attributs peut être vu comme une recherche dans un espace d'hypothèses (appelé ensemble de solutions possibles) [Blum et Langley, 1997]. Étant donné un ensemble initial \mathcal{X} de n attributs, la sélection d'un "bon" sous-ensemble d'attributs nécessite d'examiner potentiellement $2^n - 1$ sous-ensembles possibles. La qualité d'un sous-ensemble sélectionné est évaluée selon un critère de performance que l'on notera \mathcal{J} . Dans le cas d'un problème de classification supervisée, ce critère est très souvent la précision d'un classifieur construit à partir de l'ensemble d'attributs sélectionnés.

La recherche d'un sous-ensemble d'attributs, optimal pour le critère \mathcal{J} que l'on s'est donné, est alors un problème NP -difficile [Davies et Russell, 1994; Cotta et Moscato, 2003]. Plusieurs approches peuvent être envisagées pour contourner cette difficulté. Elles sont formalisées dans la définition suivante:

Définition 2.6 (Sélection d'attributs) [Molina et al., 2002a] Soit X un ensemble d'attributs. Soit \mathcal{J} une mesure d'évaluation qui attribue à tout sous-ensemble de X un score: $\mathcal{J} : \bar{X} \subseteq X \rightarrow \mathbb{R}$. \mathcal{J} doit être optimisée (maximisée ou minimisée suivant la nature de \mathcal{J}), on supposera dans la suite que \mathcal{J} doit être maximisée .

La sélection d'un sous-ensemble d'attributs peut se faire suivant un des schémas suivants :

- Nombre d'attributs fixé: Pour un nombre m fixé, avec $m < n$, on cherche à trouver $\bar{X} \subset X$ tel que $|\bar{X}| = m$ et que $\mathcal{J}(\bar{X})$ soit maximum.
- Seuil de performance fixé: On se donne une valeur seuil \mathcal{J}_{opt} , c'est-à-dire, le minimum acceptable pour \mathcal{J} , et on cherche à trouver $\bar{X} \subseteq X$ tel que le cardinal de \bar{X} soit le plus petit possible et que $\mathcal{J}(\bar{X}) \geq \mathcal{J}_{opt}$.
- Compromis performance et nombre d'attributs. Trouver un compromis entre le fait de minimiser le nombre d'attributs $|\bar{X}|$ et le fait d'optimiser $\mathcal{J}(\bar{X})$.

La première stratégie de la sélection consiste à passer d'un ensemble initial de n attributs à un sous-ensemble de m attributs sélectionnés qui donne une performance au moins égale ou meilleure à celle obtenue avec l'ensemble complet. Cela suppose qu'on connaît le nombre optimal des attributs à sélectionner. La première difficulté est de définir a priori ce nombre m . Ce nombre dépend de la taille, de la quantité et de la qualité de l'information disponible. Si m est fixé, une deuxième difficulté consiste alors à examiner toutes les combinaisons possibles. La recherche d'un sous-ensemble de m attributs parmi n donne un nombre de combinaisons $\binom{n}{m}$. A titre indicatif, le nombre de combinaisons sans répétition C_{49}^6 vaut 13983816 (il s'agit là du nombre de combinaisons possibles pour un tirage du loto). La croissance exponentielle de $\binom{n}{m}$ rend la recherche très coûteuse et une exploration exhaustive n'est pas envisageable, même pour des valeurs modérées de m et n .

Dans le deuxième cas on fixe un seuil de performance \mathcal{J}_{opt} à respecter. On cherche donc un sous-ensemble de cardinalité minimale dont la performance soit meilleure que \mathcal{J}_{opt} . La valeur \mathcal{J}_{opt} peut être une valeur observée avec une certaine représentation du problème et on se fixe l'objectif de trouver une représentation utilisant un nombre minimum d'attributs mais garantissant une performance au moins égale à \mathcal{J}_{opt} . Nous verrons dans les chapitres suivants que des méthodes évolutionnaires comme les algorithmes génétiques peuvent être utilisés pour cet objectif.

Dans le troisième cas, on considère un problème d'optimisation bi-critère où l'on cherche à la fois à maximiser la fonction \mathcal{J}_{opt} tout en minimisant le nombre d'attributs retenus.

Dans le cadre de la sélection d'attributs de puces à ADN, il faut considérer le bon compromis entre la performance et la taille du sous-ensemble final en prenant les critères précédemment cités.

Nous verrons dans les chapitres suivants que dans les méthodes que nous proposons ou dans les expériences réalisées, nous utilisons chacun de ces trois cas de figures.

2.3.2 Schéma général de la sélection d'attributs

Les différentes méthodes proposées dans la littérature pour la sélection d'attributs peuvent être décrites par un schéma général [Dash et Liu, 1997] (voir Figure 2.2) dans lequel on trouve les éléments clés suivants :

1. Une procédure de génération de sous-ensembles candidats qui détermine l'exploration de l'espace de recherche.
2. Une fonction d'évaluation donnant la qualité des sous-ensembles candidats.
3. Une condition d'arrêt.
4. Un processus de validation pour vérifier si l'objectif souhaité est atteint.

Nous détaillons ci-dessous les étapes importantes de ce schéma.

2.3.2.1 Initialisation

Pour former un ensemble de départ dans le processus de sélection d'attributs, on peut envisager différents cas qui sont fortement liés à la direction de recherche qu'on va utiliser. L'ensemble considéré comme point de départ peut être a) l'ensemble de tous les attributs disponibles, b) l'ensemble vide, c) un ensemble d'attributs tirés de manière aléatoire ou e) une population de sous-ensembles d'attributs lorsqu'on travaillera avec des méthodes à base de population.

2.3.2.2 Critère d'évaluation

On distingue deux grandes approches pour la sélection d'attributs suivant que l'on va évaluer les attributs individuellement ou évaluer des sous-ensembles d'attributs.

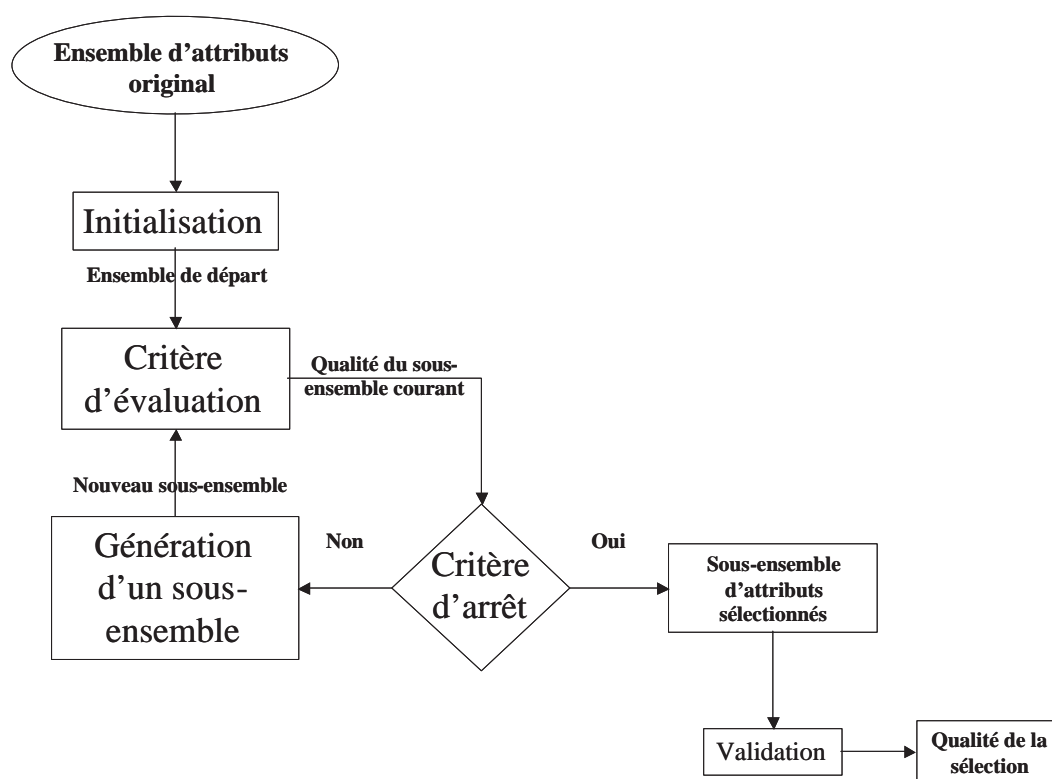


Figure 2.2 – Processus de la sélection d'attributs.

a) L'évaluation individuelle

Dans l'évaluation individuelle, chaque attribut est évalué indépendamment des autres à partir des seules informations fournies par les données et on lui attribue un poids (score) selon son degré de pertinence. Ce score permet d'établir une liste classée des attributs qui donnera un ensemble de m attributs en sélectionnant les m premiers. On garantit ainsi d'avoir des attributs pertinents (voir les catégories II, III et IV de la figure 2.1) mais cette approche ne peut enlever les attributs redondants car ces attributs ont probablement des scores similaires. Pour les données de grande dimension contenant un grand nombre d'attributs corrélés, cette approche produit des résultats éloignés des optimaux car elle ne gère pas du tout la redondance. Le principal intérêt de cette méthode est sa complexité linéaire qui lui permet de traiter des données de grande dimension. On peut remarquer que cette approche peut être utilisée dans un premier temps pour réduire la taille de l'ensemble de départ.

b) L'évaluation d'un sous-ensemble

Beaucoup de méthodes de sélection s'appuient sur l'évaluation des sous-ensembles pour gérer à la fois la redondance et la pertinence. Dans la figure 2.2, la génération

de sous-ensembles produit des sous-ensembles candidats suivant une stratégie de recherche [Liu et Yu, 2005]. Chaque sous-ensemble d'attributs candidat est évalué par une certaine mesure d'évaluation et comparé avec le meilleur sous-ensemble d'attributs obtenu précédemment par rapport à cette mesure. Si le sous-ensemble courant est meilleur, il remplace le meilleur sous-ensemble d'attributs mémorisé. Le processus de génération et d'évaluation d'un sous-ensemble est répété jusqu'à ce qu'un critère d'arrêt soit satisfait. On peut noter que les mesures d'évaluation utilisées par cette approche prennent en compte l'existence et l'effet d'attributs redondants à la différence des méthodes d'évaluation individuelle. Un sous-ensemble d'attributs sélectionné par cette approche se rapproche d'un sous-ensemble optimal (voir les catégories III et IV dans la figure 2.1). Plusieurs méthodes se sont montrées performantes en enlevant les attributs non pertinents et les attributs redondants [Koller et Sahami, 1996]. Néanmoins, le problème de ces méthodes est qu'elles doivent explorer l'espace des sous-ensembles d'attributs.

2.3.3 Génération de sous-ensembles et procédures de recherche

L'espace de recherche comporte 2^n sous-ensembles candidats si n est le nombre d'attributs initial. Plusieurs stratégies de recherche heuristiques peuvent être envisagées. Dans une stratégie de recherche séquentielle ascendante (forward selection), l'ensemble de départ est l'ensemble vide et chaque étape de génération de sous-ensembles considère tous les attributs qui ne sont pas encore sélectionnés pour retenir le meilleur d'entre eux et l'ajouter à l'ensemble courant. De la même façon, une recherche descendante part de l'ensemble complet d'attributs et retire à chaque étape un attribut.

Ces méthodes de nature gloutonne ne font pas de retour arrière (et ne sont donc pas complètes). Leur principal avantage est qu'elles sont de complexité quadratique. Sur des données de taille modérée, elles peuvent donc être appliquées efficacement. Pour les données de très grande dimension comme celles issues de la bioinformatique, il faut adapter ces procédures. On peut par exemple proposer une procédure de recherche descendante où dans les premières étapes de la recherche, lorsqu'on travaille avec un grand nombre d'attributs, on élimine non pas un seul attribut à la fois mais plusieurs attributs. Il faut pour cela pouvoir estimer efficacement (en un seul calcul) quels sont les attributs à retirer. On trouvera une illustration de ce schéma dans la méthode SVM-RFE [Guyon *et al.*, 2002].

La recherche peut aussi se faire grâce à des procédures évolutionnaires ou métaheuristiques. C'est ce que nous présenterons dans les chapitres suivants. Dans ces approches, la génération de sous-ensembles candidats se fait généralement suivant un processus aléatoire (croisement par exemple).

Pour appliquer ces stratégies sur des données de grande dimension, il sera nécessaire de guider efficacement la recherche et un des apports de notre travail sera de proposer des informations utiles pour guider au mieux les stratégies de recherche.

2.3.3.1 Critère d'arrêt

L'initialisation et le critère d'arrêt définissent les bornes de la recherche. Dans le cas de méthodes basées sur un critère d'évaluation individuelle la condition d'arrêt peut être un nombre fixé d'attributs à retenir. Dans le cas des méthodes d'évaluation de sous-ensembles, le critère d'arrêt peut être un temps de calcul fixé, un nombre d'itérations fixé, l'absence de gain de performance par rapport aux solutions déjà trouvées ou encore le fait que les sous-ensembles candidats deviennent trop homogènes (dans le cas d'algorithmes à base de populations).

2.4 Les approches pour la sélection d'attributs

Le schéma général présenté dans la section précédente se retrouve dans les 3 grands types d'approches que nous décrivons ci-dessous. Ces approches se distinguent par la manière dont la sélection d'attributs interagit (ou non) avec le mécanisme de classification.

2.4.1 Méthodes de Filtre

Dans les méthodes traditionnelles appelées méthodes filtres ou de filtrage, le principe consiste à évaluer chaque attribut (cas univarié) pour lui assigner un score de pertinence. Ce score permet un classement des attributs et in fine la sélection des attributs les mieux classés c'est-à-dire les plus pertinents. Notons que l'on trouve aussi quelques méthodes multivariées qui attribuent des scores à des groupes d'attributs.

L'avantage des méthodes de filtrage est qu'elles peuvent être utilisées lorsqu'on travaille avec un très grand nombre d'attributs car elles sont de complexité raisonnable. Elles ne tiennent compte que des informations présentes dans les données et sont indépendantes du processus de classification.

Le principal point négatif de ces méthodes est qu'elles évaluent les attributs individuellement en négligeant les interactions possibles avec les autres attributs. Ces approches ont donc du mal à éliminer les attributs qui sont redondants. Enfin, ces méthodes reposent généralement sur le choix d'un seuil pour le critère de pertinence choisi ou d'un nombre d'attributs à sélectionner qui doit être fixé a priori. Le choix de ces paramètres n'est pas facile à réaliser.

Nous détaillons dans la suite de cette section les critères de filtre qui ont été utilisés dans le domaine de la bio-informatique pour la sélection de gènes. La mesure de pertinence utilisée dans une méthode filtre peut être une mesure statistique classique telle que la t-statistique, le test de Fisher ou le test de Wilcoxon. Certaines mesures de filtrage ont été proposées spécifiquement pour la sélection de gènes telles que B/W [Dudoit *et al.*, 2002] ou SNR [Golub *et al.*, 1999].

a) t-statistique

2.4 Les approches pour la sélection d'attributs

Le critère t-statistique est utilisé dans [Nguyen et Rocke, 2002]:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s_1^2/n_1 + s_2^2/n_2}} \quad (2.5)$$

où n_k , \bar{x}_k et s_k^2 sont la taille, la moyenne la variance des classes $k = 1, 2$. Pour chaque gène une t – valeur est calculée et si on souhaite sélectionner p gènes, on retient $p/2$ gènes avec les plus grandes valeurs positives (gènes fortement exprimés dans la classe 1) et les $p/2$ gènes avec les plus “grandes” valeurs négatives (gènes fortement exprimés dans la classe 2).

b) test de Wilcoxon

La statistique de la somme des rangs de Wilcoxon est utilisée pour déterminer si un gène est différentiellement exprimé dans deux classes [Deng *et al.*, 2004; Chu *et al.*, 2005]. Pour un gène donné, les valeurs d'expression sont triés dans l'ordre croissant et on attribue un rang à chaque échantillon. La valeur W est définie comme la somme des rangs de la classe 1, et on associe à cette valeur une p – valeur indiquant si l'hypothèse que les expressions des deux classes sont différentes peut être retenue.

c) Fisher

Le test de Fisher [Duda *et al.*, 2000] est défini comme suit :

$$P = \frac{(\bar{x}_1 - \bar{x}_2)^2}{(s_1^2 + s_2^2)} \quad (2.6)$$

où \bar{x}_k , $(s_k)^2$ sont la moyenne et l'écart-type de l'attribut pour la classe $k = 1, 2$. Un score important indique donc que les moyennes des 2 classes sont significativement différentes.

d) BW

Le score discriminant BW a été proposé par [Dudoit *et al.*, 2002]. Il est basé sur le rapport entre dispersion entre classes et dispersion intra-classes. Pour un attribut j , ce rapport est obtenu comme suit :

$$BW(j) = \frac{\sum_i \sum_k I(y_i = k)(\bar{x}_{kj} - \bar{x}_j)^2}{\sum_i \sum_k I(y_i = k)(x_{ij} - \bar{x}_{kj})^2} \quad (2.7)$$

où \bar{x}_j et \bar{x}_{kj} dénotent respectivement la moyenne d'un attribut j à travers tous les échantillons et à travers les échantillons appartenant à la classe k seulement.

e) SNR ou S/N

Ce critère a été proposé par [Golub *et al.*, 1999] et est défini comme suit :

$$P(j) = \frac{\bar{x}_{1j} - \bar{x}_{2j}}{s_{1j} + s_{2j}} \quad (2.8)$$

où \bar{x}_{kj} , s_{kj} dénotent la moyenne et l'écart-type de l'attribut j pour les échantillons de classes $k = 1, 2$. De grandes valeurs de $|P(j)|$ indiquent une forte corrélation entre les valeurs de l'attribut et la distinction de classes.

2.4.2 Méthodes Enveloppes

Dans le deuxième type d'approche pour la sélection d'attributs, le mécanisme de sélection interagit avec un classifieur pour trouver un sous-ensemble d'attributs qui sera optimal pour ce modèle d'apprentissage [Kohavi et John, 1997]. Comme on l'a présenté auparavant, la sélection peut être vue comme une exploration de sous-ensembles candidats et dans les méthodes enveloppes, un algorithme de recherche "enveloppe" le processus de classification. Le processus d'apprentissage effectue la tâche de l'évaluation des sous-ensembles d'attributs.

L'intérêt de ces méthodes est que le sous-ensemble choisi est parfaitement adapté au classifieur.

En revanche, un risque de sur-apprentissage existe. De plus, ces méthodes enveloppes sont beaucoup plus coûteuses en temps de calcul puisqu'un classifieur doit être construit chaque fois que l'on doit évaluer un sous-ensemble candidat. Leur complexité de calcul est donc dépendante de la complexité du modèle d'apprentissage utilisé.

2.4.3 Méthodes Intégrées

Ces méthodes sont proches des méthodes d'enveloppe, car elles combinent le processus d'exploration avec un algorithme d'apprentissage [Navin *et al.*, 2006]. La différence avec les méthodes enveloppes est que le classifieur sert non seulement à évaluer un sous-ensemble candidats mais aussi à guider le mécanisme de sélection. Selon cette définition, les méthodes de construction d'arbres de décision comme C4.5 de Quinlan [Quinlan, 1993] ou CART (Classification and Regression Tree) de Breiman [Breiman *et al.*, 1984] relèvent de cette approche puisque la sélection des attributs se fait en même temps que la construction du modèle. Parmi les derniers travaux qui utilisent cette approche, on peut citer Weighted naive Bayes [Duda *et al.*, 2000] et les travaux dans lesquels le mécanisme d'apprentissage fournit des poids aux attributs pour faire la sélection. On peut citer notamment RFE-SVM [Guyon *et al.*, 2002; Rakotomamonjy, 2003], RFE+PKLR [Zhu et Hastie, 2004] et A-ROM [Weston *et al.*, 2002].

L'avantage de ces méthodes est que le processus de recherche est guidé par des informations intéressantes fournies par le classifieur, ce qui rend ces méthodes plus efficaces que les méthodes enveloppes [Saeys *et al.*, 2007].

2.5 Validation des méthodes

Pour évaluer la sélection d'attributs, on aura besoin d'appliquer un classifieur et d'examiner les performances de la classification. On a vu dans le chapitre précédent comment un classifieur évalue le risque empirique soit sur un ensemble de test, soit par validation croisée. L'idée la plus naturelle pour évaluer une méthode de sélection d'attributs est donc d'appliquer cette méthode sur l'ensemble complet des échantillons disponibles ce qui conduit à un sous-ensemble de gènes sélectionnés. Ensuite on retient ce sous-ensemble de gènes comme représentation du problème et on évalue par valida-

tion croisée par exemple la performance d'une règle de classification entraînée sur cette représentation.

Ce schéma a été appliqué par exemple dans le travail de [Guyon *et al.*, 2002] où de très bonnes performances sont obtenues pour les jeux de la Leucémie et du cancer du Colon. Ainsi, dans le cas de la Leucémie, la méthode SVM-RFE sélectionne 2 gènes à partir desquels on obtient pour un classifieur SVM un taux de bonne classification de 100%. Mais ces résultats ainsi que ceux que l'on trouve dans d'autres travaux de la même époque sont très optimistes et souffrent d'une erreur d'estimation appelée "biais de sélection" et mis en évidence par [Ambroise et McLachlan, 2002].

Les auteurs de Ambroise et McLachlan ont étudié le protocole expérimental généralement employé pour évaluer les méthodes de sélection de gènes pour les données de puces à ADN et ils ont observé que la procédure appliquée pour ces travaux pouvait fournir des conclusions erronées concernant la performance des algorithmes. Le biais de sélection se produit lorsque le processus de sélection est effectué à partir de tous les échantillons disponibles et que l'on applique ensuite un processus de validation pour estimer l'erreur en classification.

L'information de tous les échantillons est utilisée pour la sélection, et la sélection est donc mise à l'écart du processus de validation croisée généralement employé. Pour corriger ce problème de biais de sélection les auteurs proposent que les méthodes de sélection d'attributs utilisent un schéma de validation croisée externe à la procédure de sélection. Ainsi on doit d'abord découper le jeu de données en k blocs. $(k - 1)$ blocs sont utilisés pour sélectionner un ensemble de gènes et construire un classifieur et on évalue cet ensemble sélectionné et le classifieur appris sur le bloc de données restant (voir Figure 2.3).

Le modèle de validation pour la sélection et classification d'attributs est présenté dans la Figure 2.3, qui montre la partition de données en un ensemble d'échantillons d'apprentissage et un ensemble d'échantillons de test. Cette méthode peut être utilisée pour évaluer n'importe quel type de méthode de sélection, soit de filtrage, soit d'enveloppe ou soit d'approche intégrée.

Notons tout de même que cette validation exige de recommencer le processus de la sélection à chaque itération de la validation croisée ce qui peut être coûteux en temps de calcul pour une méthode enveloppe ou intégrée.

2.6 Synthèse du chapitre

Nous avons présenté dans ce chapitre la sélection d'attributs et les notions qui s'articulent autour d'elle. Lorsque le nombre d'attributs est élevé et le nombre d'échantillons est très limité, il semble naturel d'envisager des méthodes pour la sélection d'attributs pour réduire la dimension des données voire pour construire un modèle performant.

Nous avons présenté aussi les techniques plus connues sur la sélection d'attributs et qui se placent dans le cadre de la recherche du sous-ensemble optimal, à savoir les méthodes de filtre, enveloppe et intégrée.

Nous donnons aussi la manière pour éviter un biais de sélection sur le vrai contenu

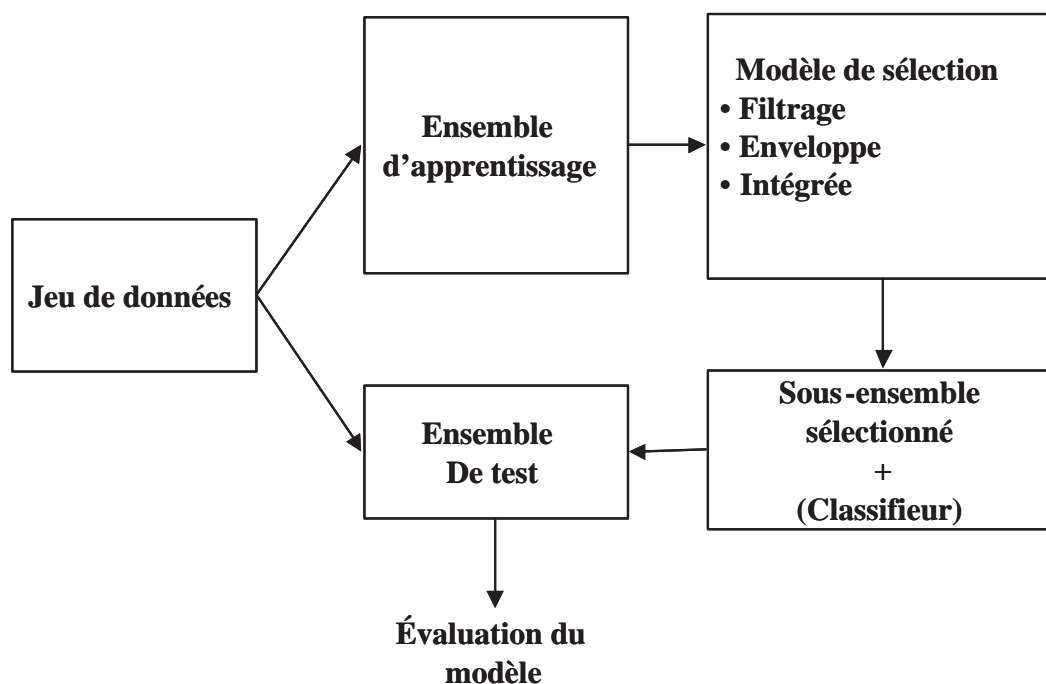


Figure 2.3 – Schéma de validation croisée pour l'évaluation d'un processus de sélection-classification.

informatif des attributs sélectionnés. Ainsi, pour avoir une estimation non optimiste de l'erreur et ainsi éviter le biais de sélection il est nécessaire d'appliquer la procédure de sélection dans une étape externe à l'étape d'évaluation. Pour plus de détails consulter [Ruschhaupt *et al.*, 2004; McLachlan *et al.*, 2008].

Chapitre 3

Pré-traitement et pré-sélection des données de puces à ADN à l'aide de la logique floue

Sommaire

3.1	Logique floue	40
3.1.1	Sous-ensembles flous	40
3.1.2	Variables linguistiques	41
3.1.3	Système d'Inférence Floue	43
3.1.4	Représentation floue des variables d'entrée	44
3.1.5	Représentation floue des variables de sortie	45
3.1.6	Définition des règles floues	45
3.1.7	Inférence à partir de règles floues	46
3.1.8	Défuzzification	49
3.2	Pré-traitement flou pour les données de puces à ADN	49
3.2.1	Représentation floue de la variable d'entrée	50
3.2.2	Représentation floue de la variable de sortie	50
3.3	Schéma de pré-sélection de gènes par la logique floue	54
3.3.1	Mesure de similarité	55
3.3.2	Relation d'équivalence floue	55
3.3.3	α -coupes	58
3.3.4	Sélection de la valeur de α -coupe "optimale"	58
3.3.5	Choix d'un représentant de chaque groupe	59
3.4	Evaluation du modèle flou de réduction de dimension	61
3.4.1	Évaluation avec un petit nombre de gènes	62
3.4.2	Évaluation avec un grand nombre de gènes	64
3.5	Synthèse du chapitre	66

DANS le cadre de données de puces à ADN, le pré-traitement est une procédure qui a comme objectif de pré-filtrer les gènes qui ne sont pas informatifs, c'est-à-dire des gènes dont les niveaux d'expression est uniforme quelle que soit la classe. Pour cela une condition de seuillage et de filtrage est généralement introduite pour limiter les effets du bruit et sélectionner des gènes sans bruit. La sortie de ce pré-traitement est une légère réduction des données de puces à ADN, mais l'on peut considérer ce processus comme une méthode de réduction de données, car il s'agit d'un processus pour travailler seulement avec les gènes sans bruit. Dans ce chapitre nous envisageons deux tâches celle de la normalisation des données de puces à ADN et celle de la réduction de données à l'aide de la logique floue afin de faciliter et d'améliorer la performance de classification.

Dans le paragraphe 1, nous décrivons les notions de base et les connaissances préliminaires de la logique floue et les Systèmes d'inférence Flous (SIF). Dans le paragraphe 2, nous détaillons les composants principaux des SIF tels que la fuzzification, l'inférence et la défuzzification pour faire le pré-traitement des données de puces à ADN.

Dans le paragraphe 3, nous expliquons l'approche floue que nous proposons pour l'élimination d'informations redondantes. Cette approche est basée sur une relation de similitude entre les expressions des gènes, qui conduit à une partition des gènes en groupes d'expression similaire. Les différents groupes de gènes similaires peuvent être remplacés par un seul membre représentatif. Le choix de ce membre est basé sur différents critères pour évaluer la pertinence d'un gène.

Dans le paragraphe 4, nous proposons un protocole expérimental pour évaluer l'impact de notre pré-traitement flou sur un processus de sélection et classification.

3.1 Logique floue

La logique floue fut développée par Lofti A. Zadeh en 1965 à partir de sa théorie des sous-ensembles flous [Zadeh, 1965]. Les sous-ensembles flous sont une manière mathématique de représenter l'imprécision de la langue naturelle, ils peuvent être considérés comme une généralisation de la théorie des ensembles classiques [Zadeh, 1965; Kaufmann, 1973; C-T.Lin et Lee, 1996; Jang *et al.*, 1997; Ross, 2005; Tong-Tong, 1995]. La logique floue est aussi appelée "logique linguistique" car ses valeurs de vérité sont des mots du langage courant : "plutôt vrai, presque faux, loin, si loin, près de, grand, petit...". La logique floue a pour objectif l'étude de la représentation des connaissances imprécises, des raisonnements approchés [Gacogne, 1997] et elle cherche à modéliser les notions vagues du langage naturel pour pallier l'inadéquation de la théorie des ensembles classiques dans ce domaine [Tong-Tong, 1995].

3.1.1 Sous-ensembles flous

En théorie des ensembles classiques, l'appartenance d'un élément à un sous-ensemble est booléenne. Les sous-ensembles flous permettent en revanche de connaître le degré

3.1 Logique floue

d'appartenance d'un élément au sous-ensemble. Un sous-ensemble flou A d'un univers du discours U est caractérisé par une fonction d'appartenance [Zadeh, 1965] :

$$\mu_A : U \rightarrow [0, 1] \quad (3.1)$$

où μ_A est le niveau ou degré d'appartenance d'un élément de l'univers de discours U dans le sous-ensemble flou. On peut définir aussi un sous-ensemble flou \bar{A} dans l'univers du discours U comme suit [Zadeh, 1965; Tong-Tong, 1995; Jang *et al.*, 1997; C-T.Lin et Lee, 1996; Ross, 2005] :

$$\bar{A} = \{(x, \mu_{\bar{A}}(x)) | x \in U\}, \quad (3.2)$$

avec $\mu_{\bar{A}}(x)$ comme le degré d'appartenance de x dans \bar{A} .

Exemple 3.1 Soit U défini sur \mathbb{R} et A le sous-ensemble classique pour représenter les nombres réels supérieurs au égaux à 5; alors, nous avons :

$$A = \{(x, \mu_A(x)) | x \in U\},$$

où la fonction caractéristique est définie par :

$$\mu_A(x) = \begin{cases} 0, & \text{si } x < 5, \\ 1, & \text{si } x \geq 5, \end{cases}$$

qui est montrée dans la Figure 3.1(a). Soit, alors, un sous-ensemble flou \bar{A} qui représente les nombres réels proches de 5, nous avons donc comme fonction caractéristique :

$$\bar{A} = \{(x, \mu_{\bar{A}}(x)) | x \in U\},$$

où la fonction caractéristique est définie comme suit :

$$\mu_{\bar{A}}(x) = \frac{1}{1 + 10(x - 5)^2}$$

et qui est montrée dans la Figure 3.1(b).

3.1.2 Variables linguistiques

En logique floue les concepts des systèmes sont normalement représentés par des variables linguistiques. Une variable linguistique est une variable dont les valeurs sont des mots ou des phrases utilisées couramment dans une langue naturelle ou un langage artificiel [Zadeh, 1975]. Une variable linguistique est définie par [C-T.Lin et Lee, 1996]:

$$(X, U, T(X), \mu_x) \quad (3.3)$$

où X désigne le nom de la variable, U est l'univers du discours associé à la variable X (appelé aussi référentiel [Tong-Tong, 1995]), $T(X) = \{T_1, T_2, \dots, T_n\}$ est l'ensemble des valeurs linguistiques de la variable X (appelé également termes linguistiques ou étiquettes linguistiques), et finalement μ_x sont les fonctions d'appartenance associées à l'ensemble de termes linguistiques.

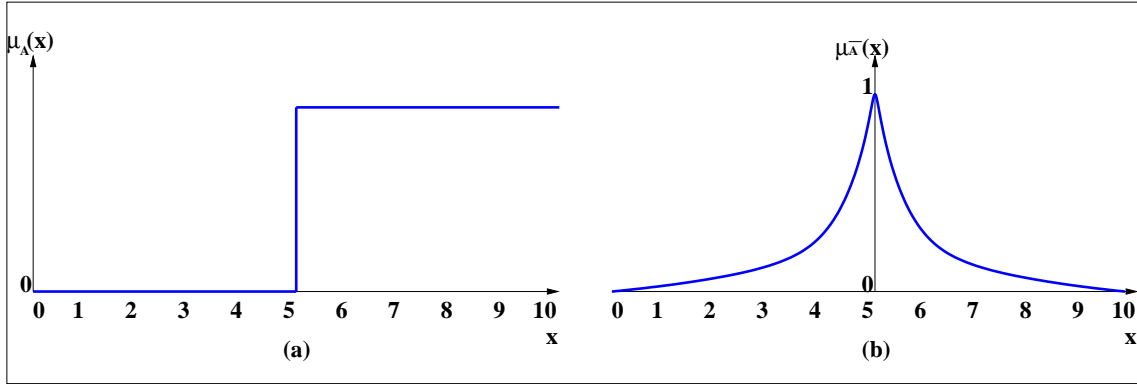


Figure 3.1 – Fonctions caractéristiques d'un sous-ensemble classique (a) et un sous-ensemble flou (b) pour l'exemple 3.1.

Exemple 3.2 On considère la taille d'un être humain, donc la variable linguistique *taille* sera définie comme suit:

$$(Taille, U = \{40, 220\}, T(X) = \{Tx_1, Tx_2, Tx_3\}, \mu = \{\mu_{x_1}, \mu_{x_2}, \mu_{x_3}\}) \quad (3.4)$$

où U est l'univers des tailles humaines défini en centimètres, l'ensemble T est constitué de trois étiquettes linguistiques : $Tx_1 = petite$, $Tx_2 = moyenne$ et $Tx_3 = grande$, et les fonctions d'appartenance définies par chaque terme linguistique sont :

$$\mu_{petite} = trapézoïde(x, a', 40, 80, 140)$$

$$\mu_{moyenne} = trapézoïde(x, 120, 160, 180, 210)$$

$$\mu_{grande} = trapézoïde(x, 170, 200, 200, d')$$

Par convention nous notons dorénavant a' et d' comme les limites des trapézoïdes qui sont reportées au-delà de l'univers de discours. Cela veut dire que les valeurs qui sont inférieures à la valeur a' et supérieures à la valeur d' ont toujours un degré d'appartenance de 1.

La fonction d'appartenance triangulaire est définie comme suit [Jang *et al.*, 1997]:

$$triangulaire(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \quad (3.5)$$

avec $(a < b < c)$ où b est le sommet du triangle tandis que a et c imposent la largeur du domaine de la valeur à fuzzifier.

La fonction d'appartenance trapézoïde est définie comme suit [Jang *et al.*, 1997]:

$$trapézoïde(x; a, b, c, d) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-b}\right), 0\right) \quad (3.6)$$

avec $(a < b \leq c < d)$ où b et c sont le sommet du trapézoïde tandis que a et d fournissent la largeur du domaine de la valeur à fuzzifier.

3.1 Logique floue

Nous illustrons l'exemple 3.2 dans la Figure 3.2 où on représente la variable linguistique taille de l'être humain. La définition de chaque sous-ensemble flou repose sur l'intuition des tailles humaines. Si une personne mesure 1m70 cela se traduira par différents degrés d'appartenances à chacun des sous-ensembles flous :

$$\mu_{\text{petit}} = 0, \mu_{\text{moyen}} = 1, \mu_{\text{grand}} = 0$$

Nous pouvons conclure que cette personne appartient au sous-ensemble des personnes de taille moyenne. Par contre une autre personne de taille 1m72 a les appartenances suivantes:

$$\mu_{\text{petit}} = 0, \mu_{\text{moyen}} = 0.95, \mu_{\text{grand}} = 0.04$$

Cette personne peut donc être considérée à la fois de taille moyenne et grande avec une plus forte appartenance à la taille "moyenne". Si nous voulons traiter cet exemple pour un domaine en particulier, la définition de sous-ensembles changera, par exemple pour la taille des espagnols ou bien pour les joueurs du basket-ball de la NBA.

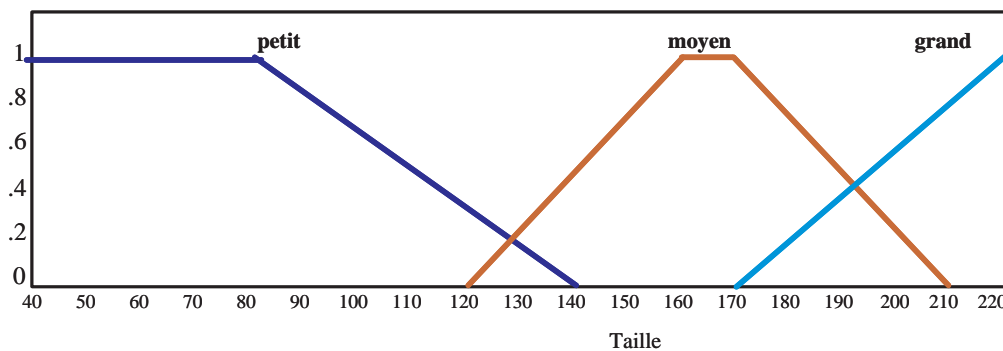


Figure 3.2 – Fonctions d'appartenance de la variable taille.

3.1.3 Système d'Inférence Floue

Un Système d'Inférence Floue (SIF) a comme but de transformer les données d'entrée en données de sortie à partir de l'évaluation d'un ensemble des règles. Les entrées sont issues du processus de fuzzification et l'ensemble de règles normalement sont définies par le savoir-faire de l'expert. Un SIF (voir Figure 3.3) est constitué de trois étapes: a) Fuzzification, b) Inférence et c) Défuzzification.

La première étape est la fuzzification, qui consiste à caractériser les variables linguistiques utilisées dans le système. Il s'agit donc d'une transformation des entrées réelles en une partie floue définie sur un espace de représentation lié à l'entrée. Cet espace de représentation est normalement un sous-ensemble flou. Durant l'étape de la fuzzification, chaque variable d'entrée et de sortie est associée à des sous-ensembles flous.

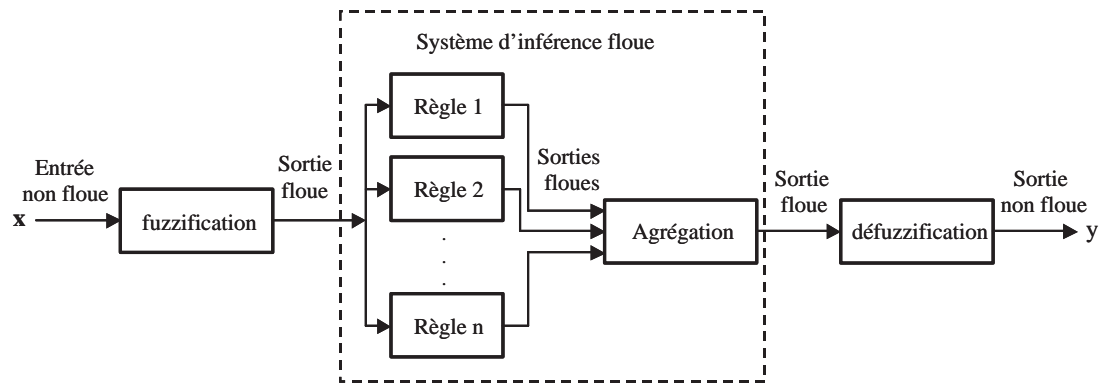


Figure 3.3 – Structure d'un SIF.

La deuxième étape est le moteur d'inférence, qui est un mécanisme permettant de condenser l'information d'un système au travers d'un ensemble de règles définies pour la représentation d'un problème quelconque. Chaque règle délivre une conclusion partielle qui est ensuite agrégée aux autres règles pour fournir une conclusion (agrégation). Les règles constituent le système d'inférence floue, dans la suite de ce chapitre nous donnons une description de règles floues dans un cadre plus formel.

La troisième étape est la défuzzification, cette opération est l'inverse de la fuzzification et permet de transformer les sorties floues de l'inférence en une valeur non floue comme réponse finale du SIF.

3.1.4 Représentation floue des variables d'entrée

Pour montrer le processus de fuzzification il faut d'abord savoir combien des variables d'entrée seront définies dans le SIF. Nous rappelons qu'une variable d'entrée (taille, température, pression, angle, vitesse, humidité, etc.) est un paramètre réel qui prend ses valeurs dans un univers bien déterminé [Tong-Tong, 1995].

Exemple 3.3 Prenons deux variables d'entrée : Vitesse et Distance qui décrivent les règles de conduite automobile à l'approche d'un carrefour contrôlé par des feux tricolores. La représentation (définition) de chaque variable suit le quadruple :

$$(Vitesse, [0, 80], \{Minimale, Normale, Maximale\}, \{\mu_{Minimale}, \mu_{Normale}, \mu_{Maximale}\})$$

$$(Distance, [0, 70], \{Courte, Moyenne, Longue\}, \{\mu_{Courte}, \mu_{Moyenne}, \mu_{Longue}\})$$

Les fonctions d'appartenance définies pour le terme linguistique Distance sont :

$$\mu_{Courte} = trapézoïde(x, a', 0, 5, 35)$$

$$\mu_{Moyenne} = triangulaire(x, 0, 35, 70)$$

$$\mu_{Longue} = trapézoïde(x, 35, 60, 70, d')$$

et celles du terme linguistique Vitesse :

$$\mu_{Minimale} = \text{trapézoïde}(x, a', 0, 10, 40)$$

$$\mu_{Normale} = \text{triangulaire}(x, 0, 40, 80)$$

$$\mu_{Maximale} = \text{trapézoïde}(x, 40, 70, 80, d')$$

La caractérisation des variables nécessite de bien connaître le domaine d'application afin de proposer un modèle adéquat à l'aide de la logique floue.

3.1.5 Représentation floue des variables de sortie

Toute variable de sortie doit être fuzzifiée car les sorties sont liées aux variables d'entrées. Pour cela il faut également savoir le nombre de variables de sortie et définir correctement l'univers du discours.

Exemple 3.4 Nous définissons une variable de sortie: Freiner. Le quadruplet pour cette variable de sortie est défini de la même manière que pour les variables d'entrée :

$$(\text{Freiner}, [0, 50], , \{\text{Douxment}, \text{Fortement}\}, \{\mu_{\text{Douxment}}, \mu_{\text{Fortement}}\})$$

qui représente l'univers $[0,50]$, c'est-à-dire la force de freinage notée en newtons et ses fonctions d'appartenances sont définies comme suit :

$$\mu_{\text{Douxment}} = \text{trapézoïde}(x, a', 0, 5, 35)$$

$$\mu_{\text{Fortement}} = \text{trapézoïde}(x, 15, 40, 50, d')$$

3.1.6 Définition des règles floues

Le nombre de règles dans un SIF dépend du nombre de variables (d'entrée et de sortie) [Tong-Tong, 1995]. Les règles floues sont généralement du type "SI ... ALORS" et permettent de représenter les relations entre les variables d'entrée et de sortie. Plus précisément une règle floue R est définie de la forme suivante [Jang *et al.*, 1997] :

$$\text{Si } x \text{ est } A \text{ Alors } y \text{ est } B \quad (3.7)$$

où A et B sont des variables linguistiques définies dans un univers du discours X et Y . La première partie de la règle " x est A " est l'antécédent et la deuxième partie de la règle " y est B " est le conséquent.

Les règles floues, peuvent être simples avec antécédent et conséquent simples ou bien composées, avec la combinaison de plusieurs prémisses de la forme conjonctive suivante :

$$R : \text{Si } x_1 \text{ est } A_1 \text{ et } x_2 \text{ est } A_2 \text{ et } \dots \text{ et } x_n \text{ est } A_n \text{ Alors } y \text{ est } B \quad (3.8)$$

ou bien de la forme :

$$R : \text{Si } x_1 \text{ est } A_1 \text{ et } x_2 \text{ est } A_2 \text{ et } \dots \text{ et } x_n \text{ n'est pas } A_n \text{ Alors } y \text{ est } B \quad (3.9)$$

Considérons à titre illustratif une règle floue : "Si la distance au feu est courte et la vitesse de la voiture est maximale alors il faut freiner Fortement".

3.1.7 Inférence à partir de règles floues

Le but de l'inférence floue est de déterminer les sorties du système à partir des entrées floues issues de la fuzzification des entrées réelles [Tong-Tong, 1995; Ross, 2005]. Étant donné une collection de règles, le mécanisme d'inférence consiste à dériver un ensemble flou de sorties à partir de l'agrégation des conclusions de l'ensemble des règles floues.

3.1.7.1 Inférence avec une seule règle

Dans le cas où une seule règle floue serait activée l'inférence repose sur la valeur d'appartenance (μ) (appelé poids) associé à la variable linguistique d'entrée. La définition pour ce cas est comme suit :

Règle 1 : **Si** x_1 est A_1 et **Si** x_2 est A_2 **Alors** y est B

Dans le cas d'inférence d'une seule règle le degré d'appartenance de la variable linguistique de sortie (B) est défini comme suit :

$$\mu_B(y) = \text{poids de la règle 1} = \min(\mu_{A_1}(x_1), \mu_{A_2}(x_2))$$

Exemple 3.5 Pour illustrer le cas d'une inférence avec une seule règle, nous supposons que nous voulons connaître la force de freinage d'une voiture qui roule dans une ville si le feu est rouge. Pour cela nous avons deux caractéristiques associées : Vitesse et Distance. Ces caractéristiques sont alors nos variables d'entrée et la variable de sortie est freinage. Si nous considérons une voiture qui roule à une vitesse de 65 Km/h et le feu se trouve à 20 mètres, la règle activée est la suivante :

R_1 : Si Distance est Courte et Vitesse est Maximale Alors Freinage est Fortement

Le degré d'accomplissement ou degré d'appartenance (μ) pour la variable Distance ($x_1 = 20$) et Vitesse ($x_2 = 65$) est comme suit :

$$x_1 = 20 \rightarrow \mu_{Courte}(20) = 0.42, \mu_{Moyenne}(20) = 0.57, \mu_{Longue}(20) = 0.0$$

$$x_2 = 65 \rightarrow \mu_{Minimale}(65) = 0.0, \mu_{Normale}(65) = 0.375, \mu_{Maximale}(65) = 0.833$$

Le résultat de l'inférence se fait seulement en prenant la valeur la plus petit des prémisses (Voir Figure 3.5) c'est-à-dire que nous obtenons :

$$R_1 : \text{Freiner}_{\mu_{Fortement_1}} = [\min(\mu_{Courte}(20), \mu_{Maximale}(65))] = \min(0.42, 0.833) = 0.42$$

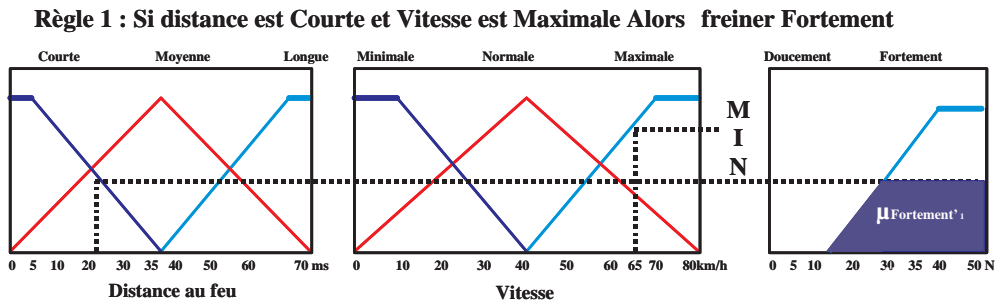


Figure 3.4 – Méthode d'inférence d'une règle floue.

3.1.7.2 Inférence avec plusieurs règles

Dans le cas où plusieurs règles floues seraient activées l'inférence repose sur les différentes valeurs d'appartenance (μ) associés aux variables linguistiques d'entrée. La définition pour l'activation de plusieurs règles est comme suit :

Règle 1 : **Si** x_1 est A_{11} et **Si** x_2 est A_{12} **Alors** y est B_1

Règle 2 : **Si** x_1 est A_{21} et **Si** x_2 est A_{22} **Alors** y est B_2

Dans le cas où B_1 et B_2 sont la même valeur de la variable de sortie y , on combine les inférences des 2 règles à l'aide de l'opérateur max. Si B_1 et B_2 sont 2 valeurs différentes, chaque règle donne un sous-ensemble flou sur la valeur de sortie y et on agrège les conclusions des 2 règles. Cela peut se représenter géométriquement comme on va le voir sur l'exemple suivant.

Exemple 3.5 Nous continuons notre exemple de la voiture avec les mêmes valeurs, c'est-à-dire 20 mètres pour la distance au feu et 65 km/h pour la vitesse de la voiture (Voir Figure 3.5).

Nous considérons les deux règles suivantes :

R_1 : Si Distance est Courte et Vitesse est Maximale Alors Freinage est Fortement

R_2 : Si Distance est Moyenne et Vitesse est Normale Alors Freinage est Doucement

Comme on a vu que :

$$x_1 = 20 \rightarrow \mu_{Courte}(20) = 0.42, \mu_{Moyenne}(20) = 0.57, \mu_{Longue}(20) = 0.0$$

$$x_2 = 65 \rightarrow \mu_{Minimale}(65) = 0.0, \mu_{Normale}(65) = 0.375, \mu_{Maximale}(65) = 0.833$$

on a :

$$R_1 : \mu_{Fortement_1} = [\min(\mu_{Courte}(0.42), \mu_{Maximale}(0.833))] = \min(0.42, 0.833) = 0.42$$

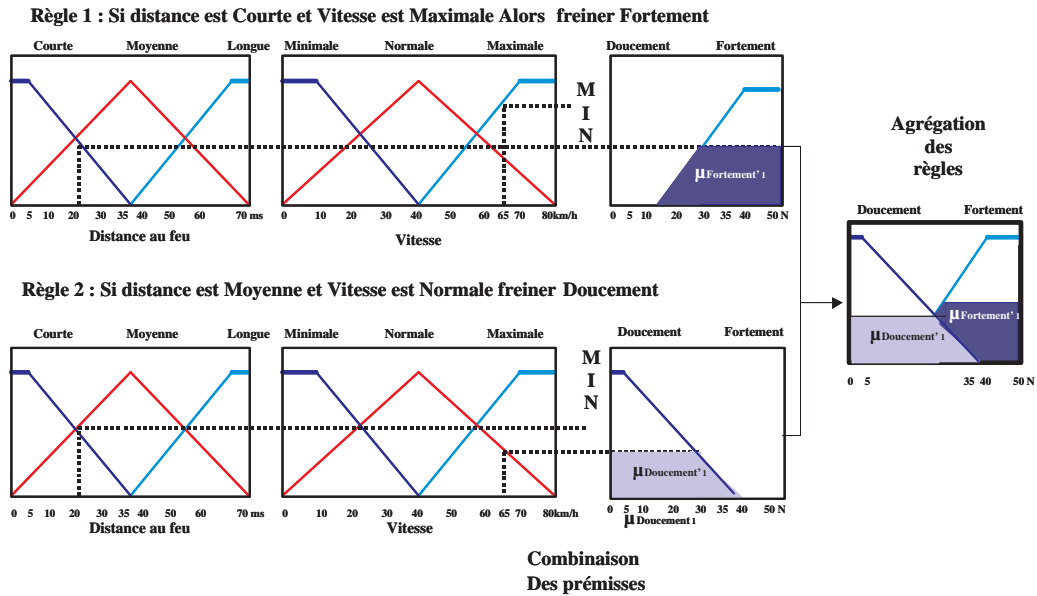


Figure 3.5 – Inférence de deux règles activées

$$R_2 : \mu_{Doucement_1} = [\min(\mu_{Moyenne}(0.57), \mu_{Normale}(0.375))] = \min(0.57, 0.375) = 0.375$$

Le sous-ensemble obtenu indique que la force de freinage doit être à la fois forte et douce. Pour connaître avec quelle force (entre 0 et 35 Newtons) le conducteur doit appuyer sur la pédale il faut défuzzifier ce résultat pour obtenir y^* (Voir Figure 3.6) comme nous l'expliquons dans le paragraphe suivant.

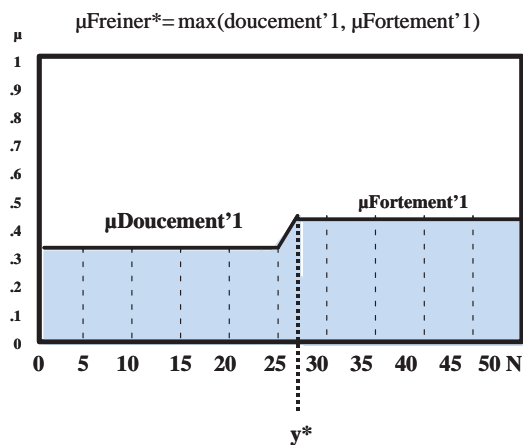


Figure 3.6 – Inférence à partir de 2 règles floues.

3.1.8 Défuzzification

La défuzzification permet d'avoir un résultat numérique non flou à partir de la sortie de l'inférence. Cette sortie est un sous-ensemble représentant l'union des conclusions ($\mu_{Freiner^*}$). La méthode de défuzzification la plus utilisée pour faire cette transformation est celle de la détermination du centre de gravité de ce sous-ensemble comme suit :

$$y^* = \frac{\int_a^b \mu_{Freiner^*}(y) dy}{\int_a^b \mu_{Freiner^*} dy} = \frac{\sum_a^b \mu(y)y}{\sum_a^b \mu(y)} \quad (3.10)$$

où y^* est une valeur qui va se transmettre à l'extérieur du système comme résultat de ce mécanisme flou. Les bornes de l'intégrale a et b correspondent alors à la valeur minimale et à la valeur maximale du sous-ensemble Freiner, i.e. $a = 0.0$ et $b = 50$. Nous voulons remarquer que l'intégrale du dénominateur donne la surface à défuzzifier, tandis que l'intégrale du numérateur correspond au moment de la surface.

La sortie (y^*) obtenue par la méthode de défuzzification est:

$$y^* = \frac{\int_{0.0}^{50} \mu_{Freiner^*}(y) dy}{\int_{0.0}^{50} \mu_{Freiner^*} dy} = \frac{112.1250}{4.35} = 25.77 \quad (3.11)$$

Étant donné que les valeurs de freiner sont notées en newtons, le conducteur doit donc appuyer sur la pédale de frein avec une force de 25.77 Newtons.

Il existe d'autres méthodes pour faire la défuzzification [Ross, 2005] : méthodes des maximums, somme-prod, moyenne-pondérée, moyenne des maximums, entre autres. La méthode du centre de gravité est normalement utilisée avec le mécanisme d'inférence max-min de Mamdani, car cette méthode fournit une interpolation proportionnelle à la taille des sous-ensembles individuels des conséquents.

3.2 Pré-traitement flou pour les données de puces à ADN

En raison des procédures expérimentales (préparation des cibles, marquage, hybridation, lecture des spots) les données des puces à ADN contiennent du bruit. Pour réduire le bruit des données des puces à ADN, un pré-traitement est appliqué comme une partie intégrante du processus de discrimination de gènes. Afin de réduire au minimum l'effet négatif du bruit, nous proposons une méthode de pré-traitement des données à l'aide de la logique floue permettant la normalisation floue des données de biopuces (voir Figure 3.7).

Représentons un jeu de données de puces à ADN par une matrice D de dimension $m \times n$, où m est le nombre d'échantillons et n est le nombre de gènes. Chaque nombre réel d_{ij} est le niveau d'expression du gène j mesuré dans l'échantillon i . Notre approche commence par une étape de pré-traitement qui s'appuie sur la logique floue pour transformer les données d'entrées (non floues) D en matrice floue des niveaux d'expression de gènes D^f . Ce pré-traitement peut être décomposé en plusieurs opérations :

1. Effectuer une discrétisation floue dans le domaine d'expression des gènes.

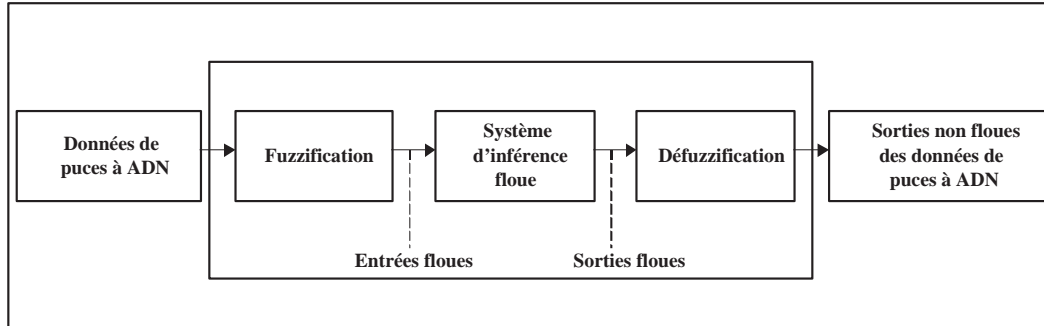


Figure 3.7 – Illustration du pré-traitement flou.

2. Appliquer un Système d'Inférence Flou (SIF) pour normaliser les valeurs d'expression de gène.
3. Appliquer la défuzzification pour transformer chaque nouveau niveau d'expression en valeurs non floues.

3.2.1 Représentation floue de la variable d'entrée

Pour le pré-traitement flou, d'abord on va représenter les valeurs de niveaux d'expression pour les différents jeux de données en utilisant un SIF. Nous présentons un exemple de fuzzification de la variable "niveau d'expression" (notée dorénavant NE) pour le jeu de données du cancer du Colon. Cette discrétisation floue devra s'effectuer dans le domaine d'expression de gènes (U). Nous représentons l'entrée dans le quadruplet :

$$("NE", U, \{Petit, Moyen, Grand\}, \mu = \{\mu_{Petit}, \mu_{Moyen}, \mu_{Grand}\})$$

L'univers du discours (U) pour le jeu du cancer du Colon se trouve dans l'intervalle [5.81, 20903.177]. Les fonctions d'appartenances définies pour chaque terme linguistique sont divisées de manière symétrique (voir Figure 3.8) comme suit :

$$\mu_{petit} = triangulaire(x, -10450, 5.816, 10450)$$

$$\mu_{moyen} = triangulaire(x, 5.816, 10450, 20900)$$

$$\mu_{grand} = triangulaire(x, 10450, 20903.177, 31350)$$

3.2.2 Représentation floue de la variable de sortie

Notre système a seulement une variable d'entrée et une variable de sortie, pour la variable de sortie nous définissons le nouveau niveau d'expression (NNE), qui est représentée dans le quadruplet :

3.2 Pré-traitement flou pour les données de puces à ADN

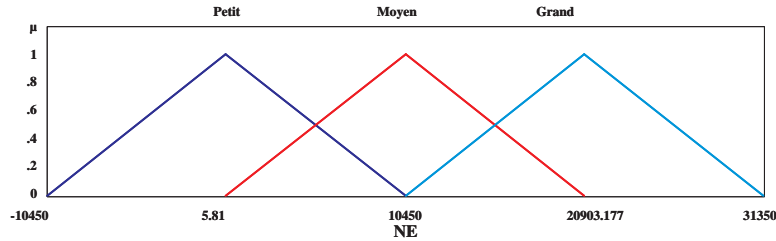


Figure 3.8 – Fuzzification de la variable d’entrée Niveau d’Expression (NE).

$$("NNE", U, \{Petit, Moyen, Grand\}, \mu = \{\mu_{Petit'}, \mu_{Moyen'}, \mu_{Grand'}\})$$

L’univers du discours (U) de la variable de sortie NNE qui est normalisée se trouve dans l’intervalle $[0, 1]$. Les fonctions d’appartenances définies pour chaque terme linguistique sont divisées de la façon suivante :

$$\mu_{Petit'} = trapézoïde((x, a', 0, 0.25, 0.5))$$

$$\mu_{Moyen'} = triangulaire(x, 0.25, 0.5, 0.75)$$

$$\mu_{Grand'} = trapézoïde((x, 0.5, 0.75, 1, b'))$$

La fuzzification de la variable NNE est illustrée dans la Figure 3.9

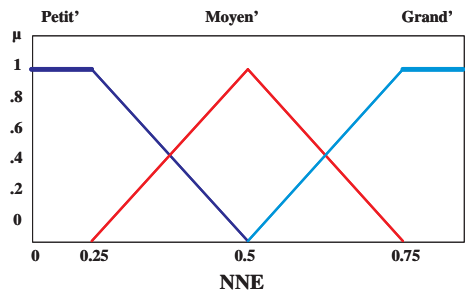


Figure 3.9 – Fuzzification de la variable de sortie Nouveau Niveau d’Expression (NNE).

3.2.2.1 Système d’inférence floue

Nous appliquons un système d’inférence flou pour pré-traiter les données bruitées et ainsi avoir une normalisation des niveaux d’expression. Nous devons tout d’abord

généraliser les règles floues pour faire l'inférence entre l'entrée (NE) et la sortie (NNE). Pour réaliser cette étape, nous utilisons les règles floues suivantes :

$$R_1 : \text{Si NE est } \textit{Petit} \text{ Alors NNE est } \textit{Petit}'$$

$$R_2 : \text{Si NE est } \textit{Moyen} \text{ Alors NNE est } \textit{Moyen}'$$

$$R_3 : \text{Si NE est } \textit{Grand} \text{ Alors NNE est } \textit{Grand}'$$

Chaque règle floue retourne une conclusion partielle du nouveau niveau d'expression pour chacun des gènes qui constituent le jeu de données à pré-traiter.

Ainsi le degré d'accomplissement de chaque règle active une conclusion sur les sous-ensembles flous de sortie. Ensuite elles sont agrégées dans la sortie, puis il faut obtenir un résultat numérique à partir de cette agrégation pour connaître le nouveau niveau d'expression des gènes.

Nous désirons montrer le processus de l'inférence pour la normalisation d'un gène du cancer du Colon d_j . Nous rappelons la variable d'entrée (NE) et la variable de sortie (NNE), alors nous prenons de lui un seul niveau d'expression (par exemple l'échantillon 53) pour visualiser le processus d'inférence. Soit donc le gène d_{ij} avec $i = 11$ et $j = 53$ qui correspond au niveau d'expression $d_{ij} = 6527.65$. Pour normaliser ce niveau d'expression nous utilisons pour le système d'inférence les règles "Si . . . Alors" et nous notons que ce gène active deux règles :

$$R_1 : \text{Si Niveau d'expression est } \textit{Petit} \text{ Alors Nouveau Niveau d'expression est } \textit{Petit}'$$

$$R_2 : \text{Si Niveau d'expression est } \textit{Moyen} \text{ Alors Nouveau Niveau d'expression est } \textit{Moyen}'$$

Nous illustrons ce processus de manière graphique pour mieux comprendre la méthode d'inférence (voir Figure 3.10). Le niveau d'expression pour d_{ij} indique qu'il appartient à deux sous-ensembles flous "Petit" et "Moyen", alors l'appartenance à chacun de ces sous-ensembles est :

$$x_1 \rightarrow \mu_{\textit{Petit}}(6527.65) = 0.3755, \mu_{\textit{Moyen}}(6527.65) = 0.6244, \mu_{\textit{Grand}}(6527.65) = 0.0$$

Étant donné qu'il existe une seule prémisse pour chaque règle comme variable d'entrée la valeur de chaque règle sera la projection de cette valeur sur la conclusion NNE comme suit :

$$\mu_{NNE_{\textit{Petit}'}} = \mu_{\textit{Petit}}(6527.65) = 0.3755$$

$$\mu_{NNE_{\textit{Moyen}'}} = \mu_{\textit{Moyen}}(6527.65) = 0.6244$$

Pour obtenir la nouvelle valeur d'expression pour le gène d_{ij} , nous devons défuzzifier la sortie. Dans la suite nous expliquons ce processus.

3.2 Pré-traitement flou pour les données de puces à ADN

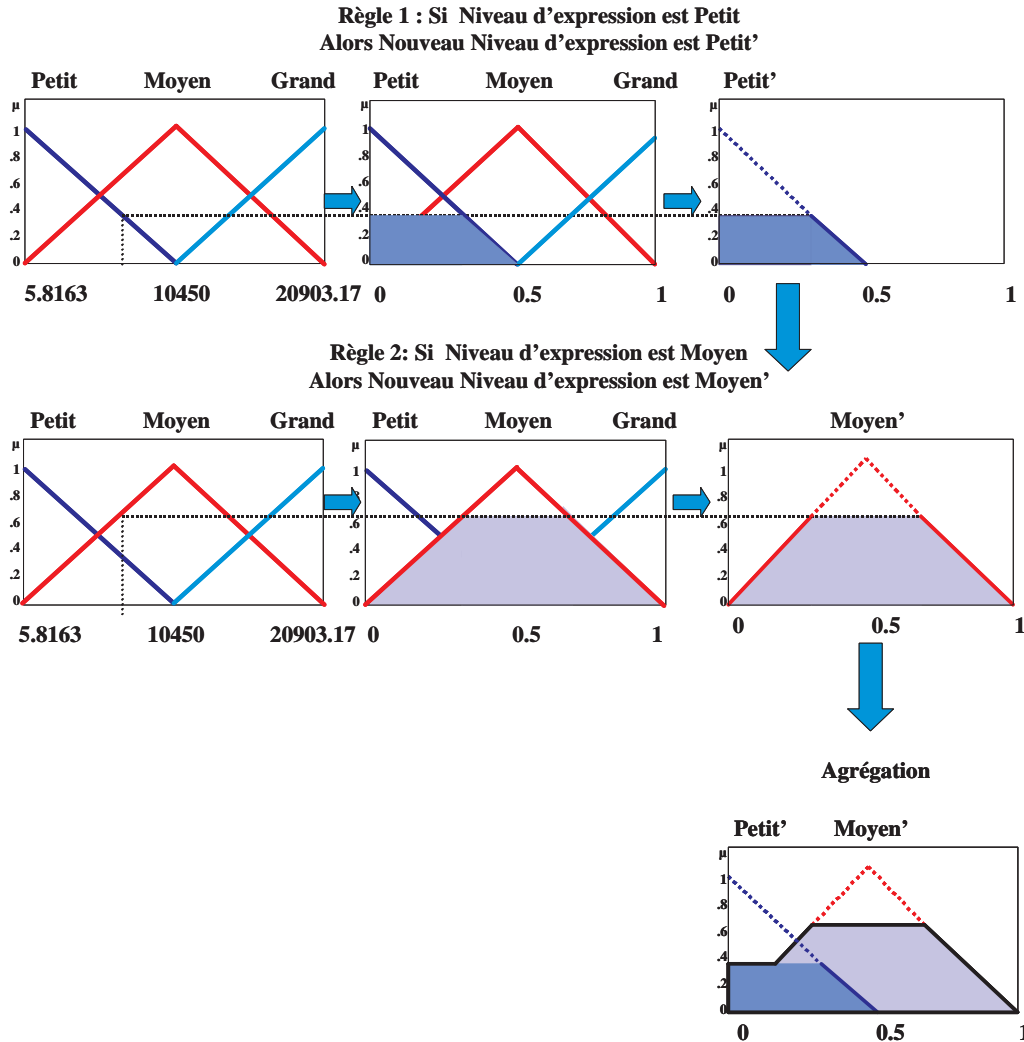


Figure 3.10 – Processus d'inférence floue pour la normalisation des niveaux d'expression.

3.2.2.2 Défuzzification

Le processus de défuzzification est appliqué pour transformer chaque nouveau niveau d'expression (NNE) en valeurs non floues. Ces valeurs forment la matrice floue $D^f = (NNE)$ contenant la discrétisation des niveaux d'expression de chaque gène. La défuzzification va permettre de connaître les nouveaux niveaux d'expression normalisés pour notre modèle flou. La Figure 3.11 illustre de manière graphique le processus de défuzzification en utilisant le centre de gravité.

Pour obtenir (y^*), nous calculons une valeur approchée de l'intégrale en la remplaçant par une somme sur des intervalles de $[0, 1]$.

$$\int_0^{.01} (\mu_{NNE})ydy, \int_{.01}^{.02} (\mu_{NNE})ydy, \dots, \int_{.99}^1 (\mu_{NNE})ydy$$

Le nouveau niveau d'expression (NNE) obtenu pour $NE_{dij} = 6527.65$ est égale à $NNE_{dij} = 0.5369$.

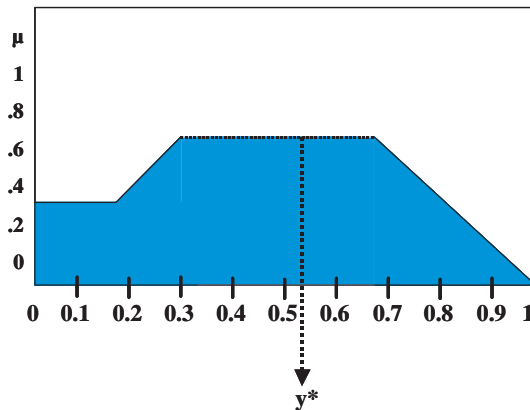


Figure 3.11 – Défuzzification par centre de gravité.

La valeur y^* obtenue par le processus de défuzzification est le centre de gravité de la figure géométrique produite par l'activation des règles floues sur la sortie. Nous avons alors une valeur non floue, dans notre cas cette valeur non floue sera le nouveau niveau d'expression normalisé par un mécanisme d'inférence floue.

3.3 Schéma de pré-sélection de gènes par la logique floue

La réduction de dimension des données de puces à ADN en utilisant un critère qui permet de sélectionner les gènes moins redondants est proposée pour améliorer la classification. Nous considérons notre modèle comme une étape de pré-sélection de gènes car nous pouvons appliquer ensuite une sélection de gènes ou bien une sélection de sous-ensembles de gènes en utilisant d'autres approches telles que les méthodes d'enveloppe ou des méthodes intégrées.

Beaucoup de travaux ont été proposés pour réduire la dimension des données de bio-puces. Tous ces travaux reposent sur l'idée du filtrage, c'est-à-dire la sélection des gènes les plus pertinents mais on obtient des gènes qui contiennent de l'information redondante. Nous proposons une technique basée sur la logique floue permettant de supprimer la redondance entre les gènes qui ont des niveaux d'expression similaire en les groupant pour ne choisir qu'un représentant de chaque groupe et ainsi réduire la dimension initiale des données.

Nous remplaçons la technique classique de groupement par des relations d'équivalence floue [Ross, 2005; Tong-Tong, 1995; C-T.Lin et Lee, 1996]. Tout d'abord, nous déterminons une relation de similarité floue pour fournir un degré de similarité entre les

3.3 Schéma de pré-sélection de gènes par la logique floue

gènes. Une relation de similarité est fréquemment utilisée pour modéliser les notions vagues de similitude. Cette relation doit être réflexive et symétrique pour générer une relation d'équivalence floue. A partir de la relation de similarité floue nous appliquons la fermeture transitive max-min pour obtenir une relation d'équivalence qui nous permet de former des classes de gènes qui ont des niveaux d'expression similaire. Nous pensons qu'il est préférable de sélectionner un gène de chaque classe pour éviter d'introduire des gènes redondants dans l'étape de sélection. Trouver une partition correcte des classes n'est pas évident, mais nous proposons une technique originale pour choisir la valeur de α -coupe optimale. Finalement nous choisissons un gène représentant de chaque classe et ainsi nous réduisons la taille initiale des données.

3.3.1 Mesure de similarité

Afin de former des groupes de gènes aux profils proches, nous mesurons la similitude des expressions des gènes contenues dans D^f . Nous utilisons la mesure du cosinus [Ross, 2005] (l'angle entre deux vecteurs) pour obtenir la corrélation entre les gènes. La colonne j de la matrice D_f est un vecteur qui contient les niveaux d'expression du gène fuzzifié j à travers tous les échantillons. Donc, la similarité entre les paires des gènes j et k est définie comme suit :

$$S_{jk} = \frac{\sum_{i=1}^m g_{ij}g_{ik}}{\sqrt{\sum_{i=1}^m g_{ij}^2} \cdot \sqrt{\sum_{i=1}^m g_{ik}^2}} \quad (3.12)$$

Une valeur de cosinus proche de 1 indique une similarité très forte entre deux gènes, en revanche une valeur de cosinus éloignée de 1 indique une similarité faible entre deux gènes.

En appliquant cette mesure de corrélation pour chaque paire de gènes, nous obtenons une matrice de similarité S de dimension $n \times n$, qui représente une relation floue entre les gènes, définie par S .

À titre d'exemple, considérons un exemple pour montrer la mesure de similarité (S) à partir de sa normalisation floue (D_f) d'un jeu de données de puces à ADN (D). Nous avons pris les 7 premiers gènes du cancer du Colon, et les 31 premiers échantillons. La taille de dimension pour considérer ces données est $D_f^{7 \times 31}$ et ainsi nous obtenons la matrice de similarité en utilisant la mesure du cosinus (voir Figure 3.12).

3.3.2 Relation d'équivalence floue

La relation de similarité S est une relation de tolérance [Ross, 2005], puisqu'elle satisfait seulement les propriétés de réflexivité et de symétrie, mais pas la transitivité. Rappelons les définitions de la réflexivité, de la symétrie et de la transitivité pour une relation floue E .

- E est une relation réflexive ssi $\forall j \in \{1, \dots, n\}, E(g_j, g_j) = 1$
- E est une relation symétrique ssi $\forall j, k \in \{1, \dots, n\}, j \neq k, E(g_j, g_k) = E(g_k, g_j)$
- E est une relation transitive ssi $\forall i, j, k \in \{1, \dots, n\}, i \neq j \neq k,$
 $E(g_i, g_j) = \lambda_1$ et $E(g_j, g_k) = \lambda_2 \rightarrow E(g_i, g_k) = \lambda$ où $\lambda \geq \min[\lambda_1, \lambda_2]$

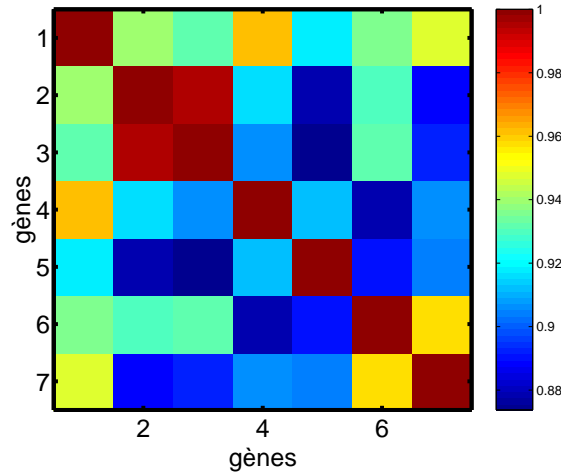


Figure 3.12 – Matrice de similarité.

À partir d'une relation de tolérance S , nous pouvons obtenir une relation d'équivalence floue E entre les gènes en calculant la fermeture transitive de S (voir l'algorithme 3.1):

Algorithme 3.1 : Fermeture Transitive max-min

```

Entrées :  $S$ 
1 début
2    $R \leftarrow S$ 
3    $R' = R \cup (R \circ R)$ 
4   tant que  $R' \neq R$  faire
5      $R \leftarrow R'$ 
6      $R' \leftarrow R \cup (R \circ R)$ 
7   fin
8    $R^t \leftarrow R'$ 
9   retourner  $R^t$ 
10 fin
    
```

où \circ est l'opérateur de composition, et nous utilisons le plus connu "max-min" :

$$\mu_{S \circ R}(x, z) = \max(\min(\mu_R(x, y), \mu_R(y, z))) \quad (3.13)$$

À titre d'exemple nous montrons ce processus en utilisant le petit jeu de données de la section précédente. Sur la Figure 3.13(a) est appliquée la première composition $R^1 = R \cup (R \circ R)$ à partir de la matrice de similarité (S). Nous pouvons constater que la relation obtenue n'est pas encore transitive car il y a des éléments dans la matrice (niveaux d'expression) qui ne vérifient pas la définition de transitivité. Voyons en détail la matrice

3.3 Schéma de pré-sélection de gènes par la logique floue

S :

$$S = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 1.0000 & & & & & \\ 2 & 0.9399 & 1.0000 & & & & \\ 3 & 0.9319 & 0.9965 & 1.0000 & & & \\ 4 & 0.9620 & 0.9168 & 0.9065 & 1.0000 & & \\ 5 & 0.9183 & 0.8780 & 0.8737 & 0.9128 & 1.0000 & \\ 6 & 0.9360 & 0.9306 & 0.9337 & 0.8797 & 0.8898 & 1.0000 \\ 7 & 0.9480 & 0.8896 & 0.8927 & 0.9074 & 0.9052 & 0.9582 & 1.0000 \end{bmatrix}$$

La relation S n'est pas transitive car avec $i = 3, j = 1$, et $k = 4$ nous avons :

$$S_{31} = 0.9319, S_{14} = 0.9620$$

et on constate que :

$$S_{34} = 0.9065 \text{ et } \min(S_{31}, S_{14}) = \min(0.9319, 0.9620) = 0.9319$$

La relation $S_{34} \geq \min(S_{31}, S_{14})$ n'est pas vérifiée. On applique donc la fermeture transitive max-min jusque ce que la relation de transitivité soit bien vérifiée. La Figure 3.13(a) montre la première itération $R^1 = R \cup (R \circ R)$ obtenue à partir de la matrice (S). Comme la nouvelle matrice ne vérifie pas encore la transitivité, une nouvelle itération est effectuée (voir Figure 3.13(b)). La nouvelle relation obtenue est une relation d'équivalence floue, car elle satisfait les propriétés de réflexivité, symétrie et de transitivité.

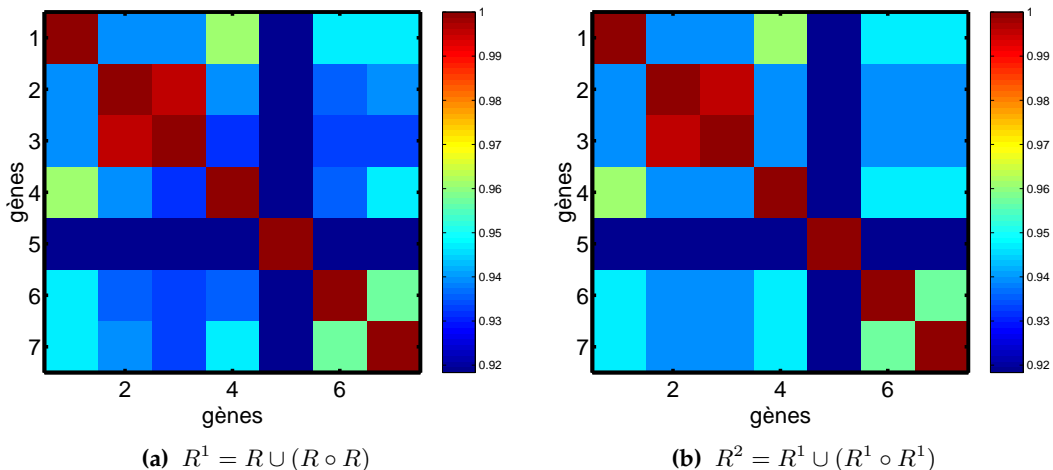


Figure 3.13 – Application de la fermeture à partir de la matrice de similitude (S).

Une fois que nous avons obtenu la relation d'équivalence floue (notée E), nous pouvons naturellement obtenir des groupes de gènes similaires en appliquant des α -coupes. Dans la section suivante, nous décrivons le processus d'obtention de classes de gènes similaires.

3.3.3 α -coupes

Une α -coupe d'un ensemble flou A est l'ensemble non flou A_α de tous les éléments qui ont un degré d'appartenance supérieur ou égal à la valeur de α [C-T.Lin et Lee, 1996; Ross, 2005; Tong-Tong, 1995] (voir Figure 3.14). Si nous considérons la relation d'équivalence floue représentée par la matrice E , pour chaque valeur α dans la matrice, nous définissons les α -coupes de E par :

$$E_\alpha = \{(x, y), \mu_{E(i,j)} \geq \alpha\}, \alpha \in [0, 1] \quad (3.14)$$

E_α est un sous-ensemble classique de E . On peut ainsi reconstruire E à partir des ses α -coupes.

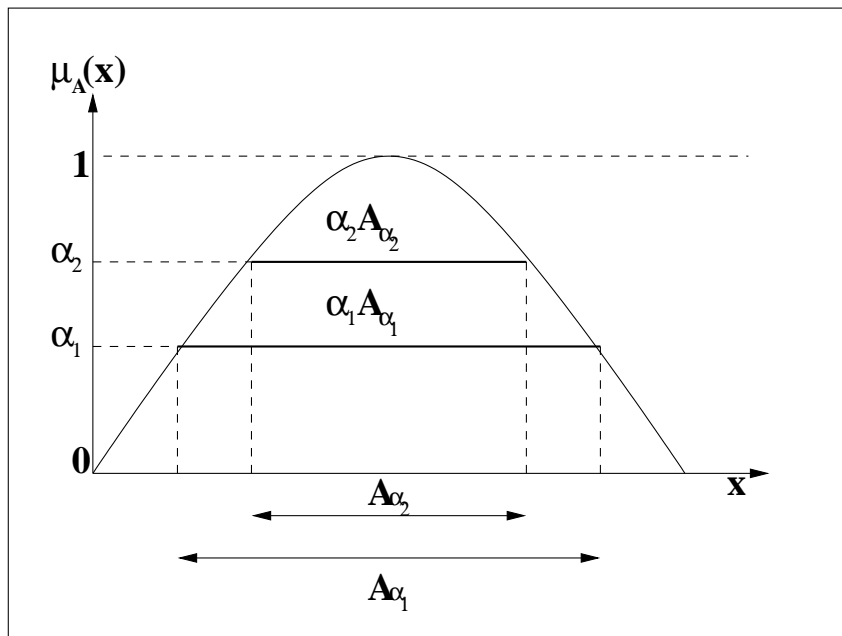


Figure 3.14 – Décomposition d'un sous-ensemble flou par alpha-coupes.

3.3.4 Sélection de la valeur de α -coupe "optimale"

Nous cherchons alors une valeur de α -coupe permettant de fournir une grande variété de groupes de gènes aux profils similaires (optimale). Afin de montrer en détail la sélection de cette valeur, nous avons sélectionné 7 gènes du cancer du Colon au hasard (voir Figure 3.15(a)) pour appliquer le pré-traitement floue (voir Figure 3.15(b)) et ensuite obtenir une relation d'équivalence floue. Pour avoir une partition des groupes des gènes similaires, nous devons choisir une valeur α "optimale" c'est-à-dire une valeur qui donne une partition ($P(E_\alpha)$) des gènes dont le niveau d'expression est similaire. Remarquons tout d'abord que la valeur ($\alpha = 0$) et la valeur ($\alpha = 1$) ne sont pas considérées, car elles

3.3 Schéma de pré-sélection de gènes par la logique floue

représentent des cas extrêmes qui constituent respectivement un seul groupe si la valeur de α est 1 et autant de groupes que de gènes si la valeur de α est 0.

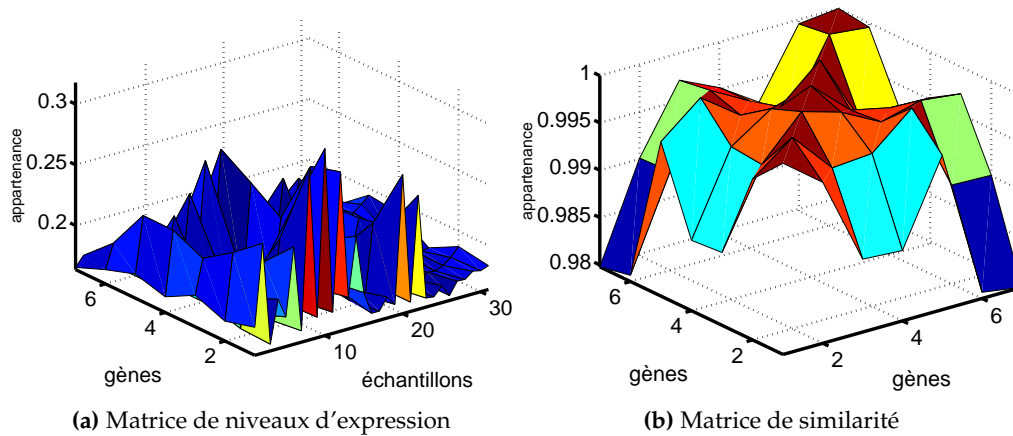


Figure 3.15 – Représentation d'une matrice de données de puces à ADN et son pré-traitement flou.

Pour rechercher la valeur de α -coupe "optimale" nous proposons de comparer les différences entre deux α -coupes successives. D'abord les valeurs des α -coupes sont des niveaux de similarité que nous ordonnons et ensuite nous cherchons l'écart le plus grand entre deux niveaux de similarité consécutifs :

$$O_i = \alpha_i - \alpha_{i-1}$$

On cherche le i tel que O_i soit le maximal et on garde α_i .

À titre illustratif nous montrons dans la Figure 3.16, la sélection de la valeur de α -coupe "optimale", qui sert pour construire les partitions (P) de la relation d'équivalence floue E .

Dans la Figure 3.16(b) il existe 5 partitions (P) (les valeurs de $\alpha = 0$ et $\alpha = 1$ ne sont pas considérées) et différentes classes, dont nous trouvons que :

$$E_{.99875} = \{g_1\}, \{g_2, g_5\}, \{g_3, g_4\}, \{g_6, g_7\} \rightarrow P(E_{.99875}) = 4 \text{ groupes}$$

nous notons que la valeur de α -coupe qui a la plus grande "chute" dans la Figure 3.16(b) est $E_{\alpha=0.99875}$ qui réalise un partitionnement de 4 classes d'équivalence.

3.3.5 Choix d'un représentant de chaque groupe

A partir des groupes de gènes dont les niveaux d'expression sont similaires on peut proposer une réduction de la dimension des données. Sur le petit exemple considéré, pour ($\alpha = 0.99875$), nous obtenons la partition suivante :

$$E_{.99875} = \{g_1\}, \{g_2, g_5\}, \{g_3, g_4\}, \{g_6, g_7\} \rightarrow P(E_{.99875}) = 4 \text{ groupes}$$

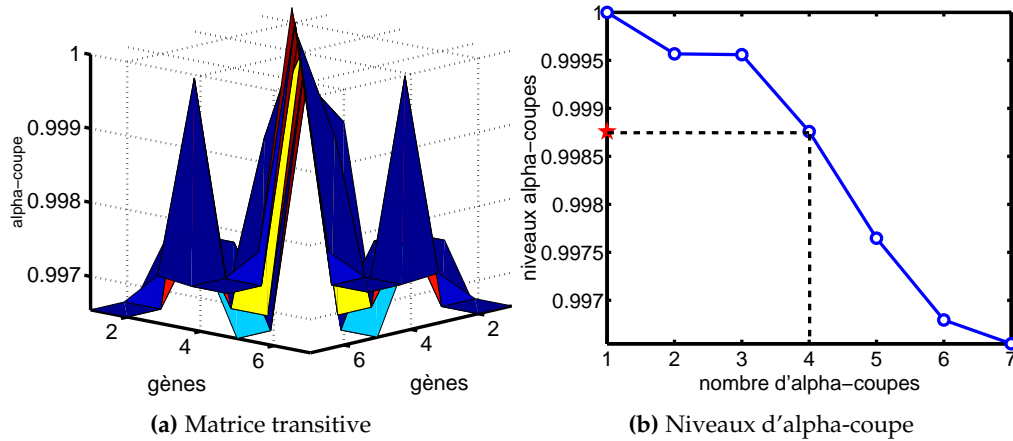


Figure 3.16 – Relation d'équivalence floue et les niveaux d'alpha-coupe.

Puisque les gènes d'un même groupe ont des niveaux d'expression très similaires, il est légitime de n'en retenir qu'un par groupe. Cela permet d'éliminer des informations redondantes et conduit donc à une réduction de dimension.

La Table 3.1 présente la réduction de dimension obtenue par cette méthode sur les jeux de données présentés dans la section 1.8. Nous pouvons constater que notre approche permet une réduction considérable pour ces jeux de données.

A titre d'exemple nous prenons le jeu de la Leucémie qui est un jeu qui a beaucoup de bruit. Pour cela plusieurs auteurs ont proposé d'ôter les gènes dont la variation de niveau d'expression est très petite. Les gènes retenus par le pré-traitement proposé par [Dudoit *et al.*, 2002] sont au nombre de 3571. Notre modèle n'a retenu que 19.07% de gènes, c'est-à-dire 5769 gènes redondants ont été ôtés.

Si nous prenons le jeu du cancer de Colon, on peut noter que la réduction de dimension est de 47.15% presque la moitié des gènes sont retenus. Nous observons que les données originales sont 2000 et que nous avons obtenu 943 gènes après l'application de la relation d'équivalence floue.

En ce qui concerne le jeu de données du Lymphome, la réduction est significative car nous avons un pourcentage de réduction de gènes de 10.80%, soit 435 conservés sur les 4026. Nous avons donc éliminé 3591 gènes redondants qui peuvent empêcher une bonne performance du classifieur appris ensuite.

3.3.5.1 Sélection du gène le plus pertinent de chaque groupe

Pour sélectionner le représentant le plus pertinent de chaque groupe, nous pouvons utiliser l'information mutuelle (IM). Soit H la fonction d'entropie et G un gène représenté par un vecteur de dimension n . Nous rappelons que dans notre cas G est une colonne de la matrice D^f et la classe est représenté par un vecteur C de dimension n , où $C_i = \{C_1, C_2, \dots, C_n\}$ est une variable discrète avec s valeurs. Soit $p(C_i)$ la probabilité de

3.4 Evaluation du modèle flou de réduction de dimension

Table 3.1 – Réduction des données à partir des relations d'équivalence floue.

jeu de Données de puces à ADN	Nombre de gènes	Nombre réduit de gènes	(%) de réduction de gènes
Leucémie	7129	1360	19.07
Colon	2000	943	47.15
Lymphome	4026	435	10.80
CNS	7129	5668	79.50
Poumon	12533	7271	58.01
Prostate	12600	8439	66.97
Ovarien	15154	938	6.18

chaque classe, la fonction d'entropie $H(C)$ est définie par :

$$H(C) = - \sum_i p(C_i) * \log p(C_i)$$

Dans le cas continu de la variable G , soit $p(g)$ la probabilité de densité, $H(G)$ est défini comme suit :

$$H(G) = - \int p(g) * \log p(g) dg$$

L'information mutuelle entre un gène G et la classe C est définie par :

$$MI(G, C) = H(G) + H(C) - H(G, C) = \sum_i \int p(g, C_i) \log \frac{p(g, C_i)}{p(C_i)p(g)} dg$$

Comme g est composé de valeurs continues, nous nous limiterons seulement à utiliser le calcul proposé dans [Schlogl *et al.*, 2002] pour évaluer l'information mutuelle. Avec ce critère on peut identifier pour chaque groupe de gènes similaires les gènes qui sont différentiellement exprimés entre les classes. Le gène G avec la valeur la plus élevée de l'information mutuelle est choisie comme un représentant de son groupe.

En choisissant le gène qui maximise l'information mutuelle nous prenons le représentant le plus intéressant dans chaque groupe.

3.4 Evaluation du modèle flou de réduction de dimension

Afin d'évaluer notre modèle flou, nous proposons de le comparer avec d'autres méthodes qui sont utilisées pour faire la réduction de dimension (voir Figure 3.17). Dans le protocole d'expérimentations notre modèle flou est appliqué avant la tâche de filtrage et de la classification et puis nous comparons les résultats obtenus avec ceux obtenus seulement avec un processus de sélection suivi d'une classification. Nous voulons connaître la performance d'ajouter notre "brique" floue dans le processus classique de sélection et classification. Dans la partie droite de la Figure 3.17 (b) nous montrons le processus d'évaluation en utilisant notre approche floue, qui est composée des pas suivants :

1. Nous appliquons le schéma flou aux jeux de données.

2. Nous appliquons une méthode de filtrage pour assigner un score à chacun de ces k gènes.
3. Nous prenons les premiers p gènes selon leur score.
4. Nous appliquons aux p gènes sélectionnés un classifieur et nous calculons son taux de classification correcte.

Les méthodes que nous proposons comme méthodes de filtre pour le pas 2 sont trois critères bien connus dans la littérature : BSS/WSS(BW) [Dudoit *et al.*, 2002], t-statistic (TT) [Nguyen et Rocke, 2002] et Wilcoxon test (WT) [Jaeger *et al.*, 2003].

Dans le pas 1, notre schéma flou produit un nombre réduit de k gènes obtenus à partir des k classes de gènes similaires.

Pour le 3, nous fixons $p = 100$ gènes.

Pour le pas 4, nous utilisons un classifieur k -plus proches voisins (kPPV) pour obtenir le taux de classification. Pour estimer ce taux nous appliquons une méthode de validation croisée (LOOCV).

3.4.1 Évaluation avec un petit nombre de gènes

Nous conduisons une première expérience pour évaluer la performance de notre modèle combiné (voir Figure 3.17 b)). D'abord pour faciliter la lecture des différentes combinaisons avec une méthode de filtrage nous nommons chaque combinaison comme suit : CM1=Schéma flou+BW, CM2=Schéma flou+TT et CM3=Schéma flou+WT.

A titre de comparaison des nos modèles combinés nous utilisons la procédure de sélection et classification (voir Figure 3.17 (a)). Les méthodes de filtre sont BW, TT et WT. Chaque couple de modèles (BW, CM1), (TT, CM2) et (WT, CM3) est appliqué sur 7 jeux de données de puces à ADN. Nous avons donc, plusieurs modèles pour savoir si notre approche permet d'améliorer la précision de la classification.

Nous voulons remarquer que la procédure classique de sélection et classification utilise une technique de pré-traitement différent pour chaque jeu de données Voir Figure 3.17 a). Nous utilisons celle qui est présentée dans [Dudoit *et al.*, 2002] comme suit:

1. Seuillage. Pour éliminer les gènes dont les niveaux sont toujours inférieurs ou supérieurs à un seuil.
2. Filtrage: Pour éliminer les gènes qui ont un niveau d'expression trop uniforme. Cela veut dire qu'on relève la valeur minimale (v_{min}) et maximale (v_{max}) du niveau d'expression parmi les tissus disponibles et on ne garde que les gènes tels que $v_{max}/v_{min} > \theta_1$ et $v_{max} - v_{min} > \theta_2$, où θ_1 et θ_2 varient pour chaque jeu de données.
3. Transformation logarithmique en base 10.
4. Standardisation.

Dans le cas de la Leucémie nous constatons qu'il existe trop de "bruit" et pour cela nous appliquons le seuillage, le filtrage et la transformation logarithmique. L'intervalle de seuillage qui nous utilisons est [100, 16000] et pour les valeurs $\theta_1 = 5$ et $\theta_2 = 500$. En ce qui concerne les jeux de données Lymphome, CNS, Poumon, Prostate et Ovarien nous avons utilisé seulement la standardisation.

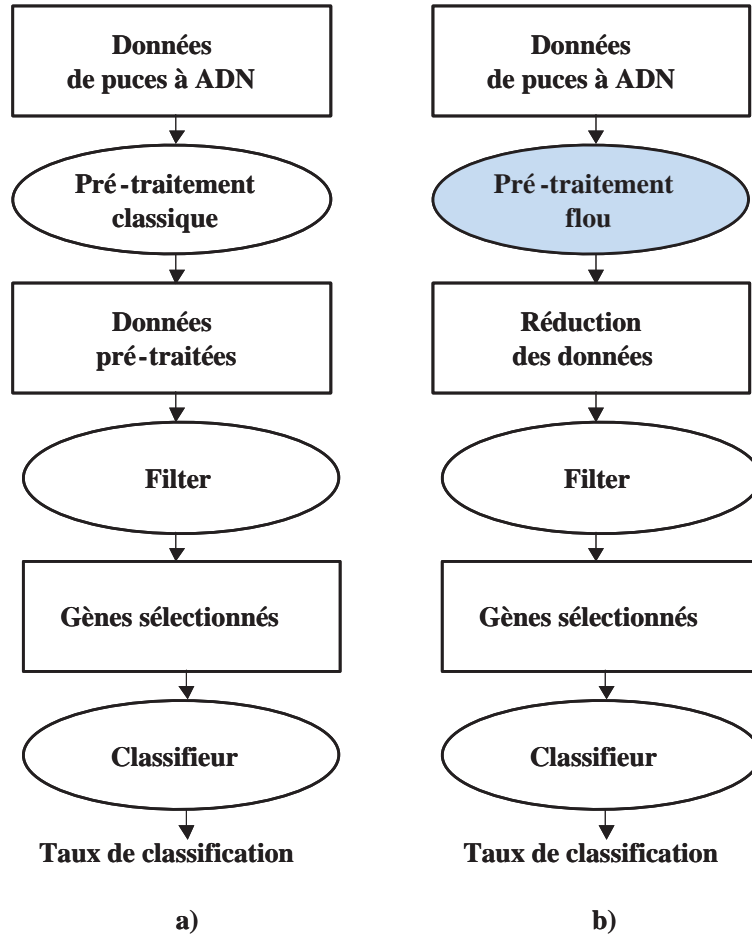


Figure 3.17 – Processus d’évaluation. a) Modèle classique de filtrage utilisé pour faire la comparaison, b) Modèle combiné utilisant un pré-traitement flou

Le tableau 3.2 récapitule les résultats obtenus par les couples (BW, CM1), (TT, CM2) et (WT, CM3). Nous avons fixé un nombre de voisins $k = 5$ pour cette première expérimentation en utilisant un nombre de gènes $p = 30$.

Nous trouvons une bonne performance de notre approche pour les jeux de données de la Leucémie et celui du cancer du Poumon. Pour la Leucémie nous trouvons un taux de classification de 100% en utilisant 30 gènes avec le modèle CM1. Nous avons donc une meilleure performance que BW (98.6%). Pour le dernier couple (TT, CM2) nous notons que CM2 a un taux de classification supérieure (97.2%) que TT (95.8%). Pour le dernier couple (WT, CM3) notre modèle a une très bonne performance, nous notons que c’est la deuxième meilleure performance après CM1.

Jeu de données	BW	CM1	TT	CM2	WT	CM3
Leucémie	95.8	100	95.8	97.2	87.5	98.6
Colon	88.7	90.3	80.6	85.4	82.2	85.4
Lymphome	92.7	92.7	87.5	88.5	93.7	86.4
CNS	81.6	73.3	71.6	80	68.3	80
Poumon	99.3	100	98.6	97.9	89.9	97.9
Prostate	93.1	92.1	92.1	92.1	92.1	93.1
Ovarien	99.6	99.6	96.0	99.2	96.4	96.4

Table 3.2 – Résultats obtenus par le modèle hybride avec $p = 30$ et $k = 5$.

En utilisant CM1 nous obtenons la plus haute performance (90.3%) pour le jeu de données du cancer du Colon. La deuxième meilleure performance est BW avec 88.7%. En revanche pour le Lymphome WT montre le taux de classification le plus haut avec 93.7%. BW et CM1 ont les deux le même résultat 92.7%. Pour CNS, BW offre la meilleure performance avec 81.6% car CM2 et CM3 obtiennent les deux un taux de classification de 80.0%. En continuant avec la Prostate BW et CM3 ont la même précision de 93.1%, tous les autres 92.1%. Finalement pour le cancer de l'ovaire nous observons que BW et CM1 arrivent à trouver la même performance 99.6%.

Nous observons que les résultats obtenus par nos modèles combinés sont très compétitifs, avec notamment une bonne performance de notre approche pour les jeux de données de la Leucémie et du cancer du Poumon. Nous rappelons que nos modèles ont obtenu la meilleure performance pour 3 de 7 jeux de données (Leucémie, Colon, et Poumon). Dans la suite de cette section nous évaluons la performance de nos modèles en utilisant plus de 30 gènes afin de tester l'influence de ce paramètre sur les résultats.

3.4.2 Evaluation avec un grand nombre de gènes

Nous avons conduit une deuxième expérience avec plus de gènes pour vérifier si nos modèles arrivent à améliorer leur performance lorsqu'on travaille avec plus de gènes. Pour cela nous fixons une valeur de $p = 100$ (les premiers 100 gènes filtrés). Les conditions d'expérimentation restent les mêmes, un classifieur kNN avec $k = 5$ et LOOCV comme méthode de validation.

Le tableau 3.3 récapitule les résultats obtenus par les différentes combinaisons de nos modèles. Pour la Leucémie nous trouvons un taux de classification de 100% avec le modèle CM1. La deuxième meilleure performance pour la Leucémie est obtenue par le filtre BW et CM3 avec 98.6%.

Dans le cas du cancer du Colon nous maintenons la même performance qu'avec $p = 30$, notre approche combinée n'arrive pas à améliorer son résultat, mais nous constatons que les autres modèles de filtre ont obtenu une meilleure performance avec les premiers 30 gènes, car ensuite la performance diminue. En revanche pour le Lymphome CM1 a obtenu un meilleur taux de classification (93.7%) le même que WT. Nous notons que pour le cancer du Poumon BW arrive à trouver une parfaite classification de 100%, sauf

3.4 Evaluation du modèle flou de réduction de dimension

Jeu de données	BW	CM1	TT	CM2	WT	CM3
Leucémie	98.6	100	97.2	97.2	95.8	98.6
Colon	88.7	90.3	80.6	85.4	82.2	85.4
Lymphome	92.7	93.7	87.5	89.5	93.7	90.6
CNS	81.6	80	71.6	80	68.3	80
Poumon	100	100	98.6	97.9	95.9	97.9
Prostate	92.2	92.2	92.1	92.1	95.9	93.1
Ovarien	99.6	99.6	96.0	99.2	98.4	96.4

Table 3.3 – Résultats du protocole de comparaison avec $p = 100$ et $k = 5$.

que la méthode BW utilise plus de gènes (entre 40 et 70 gènes) que notre modèle qui utilise seulement 2 gènes (Voir Figure 3.18).

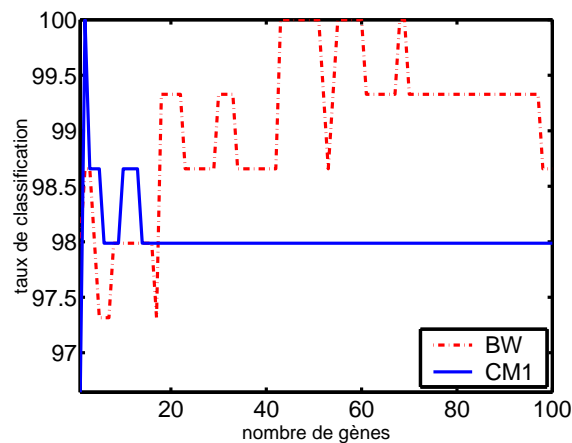


Figure 3.18 – Comparaison entre notre modèle combiné CM1 et la méthode de filtrage BW pour le jeu de données du Poumon.

Nous trouvons que la performance de nos modèles avec le cancer CNS s'améliore en utilisant CM1 (80.0%) et atteint le même taux de classification que CM2 et CM3 avec $p = 30$. La méthode de filtrage BW a la meilleure performance 81.6% et la plus faible est 71.6% avec TT. Nous constatons que ce jeu de données et celui du cancer du colon sont les deux des plus difficiles à classer.

Pour le jeu de données de la Prostate le filtre WT a le meilleur taux de classification 95.9%. Si l'on augmente le nombre de gènes, nous notons que cela n'améliore pas la performance de notre modèle. Finalement pour le cancer Ovarien CM1 et BW obtiennent le même taux de classification : 99.6% (voir Figure 3.19). Nous avons constaté que seulement dans le cas de Lymphome CM1 améliore sa performance en utilisant plus de 30 gènes et que pour les quatre jeux de données nous gardons la même performance. Nous remarquons le fait que BW égalise la performance de CM1 pour plus de 21 gènes contre 20 gènes fournis par notre modèle.

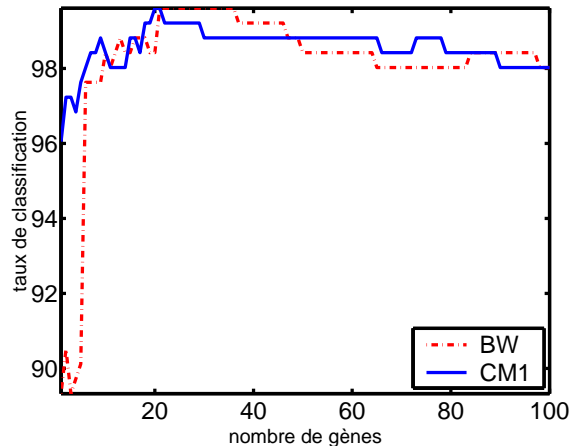


Figure 3.19 – Comparaison entre notre modèle combiné CM1 et la méthode de filtrage BW pour le jeu de données Ovarien.

Nous constatons que nous avons trouvé une bonne performance pour le jeu de données de la Leucémie par rapport aux autres méthodes de filtrage. Nous rappelons que ce jeu de données présente beaucoup de "bruit" et cela nous permet de considérer notre modèle de normalisation à l'aide de la logique floue comme une bonne stratégie pour normaliser les données de puces à ADN avec "bruit".

3.5 Synthèse du chapitre

Nous avons présenté dans ce chapitre une nouvelle méthode pour normaliser les données des biopuces à l'aide de la logique floue. Nous pensons que la normalisation des données de puces à ADN est un pas important de pré-traitement pour avoir une bonne qualité des données afin d'améliorer les protocoles expérimentaux. Nous avons également proposé dans ce chapitre une méthode originale pour réduire la dimension des données en appliquant des relations d'équivalence floues permettant de fournir un groupement de gènes qui ont un niveau d'expression similaire. Cette approche est basée sur l'idée de choisir parmi ces groupes le gène le plus pertinent pour la tâche de sélection. Pour mieux guider cette sélection des gènes, nous avons présenté plusieurs combinaisons avec trois méthodes de filtrage: BW, t-statistique et Wilcoxon test.

Pour évaluer ce pré-traitement nous avons présenté un protocole qui donne les différences de performances observées quand on ajoute ce pré-traitement floue à un processus de sélection et classification.

Chapitre 4

Explorations génétiques pour la sélection de gènes

Sommaire

4.1 Généralités sur les algorithmes génétiques	68
4.1.1 Codage des individus	69
4.1.2 Génération de la population initiale	70
4.1.3 Fonction d'aptitude	70
4.1.4 Croisement	70
4.1.5 Mutation	71
4.1.6 Élitisme	72
4.1.7 Probabilités des opérateurs génétiques	73
4.1.8 Mécanisme de sélection	73
4.1.9 Critère d'arrêt	75
4.1.10 Construction d'un AG	75
4.2 Schéma général de double exploration génétique pour la sélection de gènes	76
4.2.1 Étape 1 : Première exploration génétique	78
4.2.2 Étape 2 : Analyse de la fréquence des gènes	82
4.2.3 Étape 3 : Deuxième exploration génétique	85
4.3 Résultats sur 5 exécutions	86
4.4 Résultats sur 10 exécutions	87
4.5 Synthèse du chapitre	89

NOUS avons vu dans le chapitre précédent que nous pouvions proposer grâce à un pré-traitement flou une première réduction de la dimensionnalité des données de puces à ADN. Les ensembles de gènes retenus restent de très grande taille. Par exemple pour la Leucémie de 7129 gènes originaux nous avons 1360 gènes avec le pré-traitement flou. Nous présentons maintenant une méthode de sélection pour proposer un sous-ensemble de gènes de petite taille et qui fournit de bonnes performances en classification. Dans ce chapitre nous proposons une méthode évolutionnaire pour réaliser cela. Nous utilisons une approche de type enveloppe (appelé parfois dans cette thèse wrapper) où un algorithme génétique est couplé à un classifieur SVM. L'originalité de notre méthode est de proposer une double exploration génétique afin d'atteindre le double objectif de minimiser le nombre de gènes sélectionnés tout en maximisant les performances en classification. Une première exploration génétique de cet espace de recherche est effectuée pour identifier des sous-ensembles de gènes intéressants qui sont sauvegardés dans des archives. Ensuite une analyse de fréquence de ces archives de gènes de bonne qualité permet d'identifier un nombre encore plus réduit de gènes intéressants. Finalement une deuxième exploration génétique est effectuée pour trouver un sous-ensemble de gènes le plus pertinent et de petite taille.

Dans la section 1, nous décrivons les notions de base et les connaissances préliminaires des algorithmes génétiques. Dans la section 2, nous donnons une description des Séparateurs à Vaste Marge (SVM). Dans la section 3, nous détaillons notre modèle de double exploration génétique en utilisant une méthode d'enveloppe (wrapper) pour la sélection des sous-ensembles de gènes pertinents. Finalement dans la section 4, nous présentons les expérimentations réalisées et ainsi que les résultats obtenus par notre modèle.

4.1 Généralités sur les algorithmes génétiques

Les Algorithmes Génétiques (AG) sont des méthodes d'optimisation stochastiques qui ont été initialement développées par [Holland, 1975] et popularisées grâce à l'ouvrage de [Goldberg, 1989]. Les algorithmes génétiques sont proposés comme des heuristiques pour résoudre des problèmes complexes. Ils sont basés sur l'analogie avec le mécanisme Darwinien de la sélection naturelle dans laquelle les individus les mieux adaptés d'une population ont plus de chances de survivre dans les générations suivantes. Cela se traduit par le fait que les meilleurs individus d'une population doivent se reproduire pour engendrer de nouveaux individus de plus en plus adaptés. En revanche les individus les moins adaptés sont condamnés à disparaître.

L'objectif des algorithmes génétiques est de déterminer l'évaluation d'une fonction objectif (appelée parfois fonction d'évaluation ou fonction d'aptitude) $f : \mathcal{X} \rightarrow \mathbb{R}$ où \mathcal{X} est un ensemble de points qui définissent l'espace de recherche. Un point de cet espace représente un individu. La fonction objectif alors est définie pour mesurer la performance de chaque individu.

Les algorithmes génétiques sont normalement utilisés comme une bonne alternative pour l'optimisation de fonctions. La procédure stochastique utilisée dans un AG repose sur les points suivants :

4.1 Généralités sur les algorithmes génétiques

- un principe de codage pour chaque individu d'une population,
- une fonction à optimiser,
- un mécanisme de sélection,
- des opérateurs génétiques tels que : le croisement, la mutation ou l'élitisme,
- des paramètres initiaux tels que la taille initiale de la population, le(s) critères(s) d'arrêt, et la probabilité d'application des opérateurs génétiques.

Un AG est défini par une population initiale qui au cours de son évolution tend à converger, c'est-à-dire que les individus les plus forts tendent à se ressembler de plus en plus entre eux. Dans ce cas, nous avons une population dominée en grande partie par des individus les plus forts qui sont capables de fournir une bonne approximation de la solution du problème à résoudre. Bien entendu, l'approximation optimale fournie peut être locale ou globale (voir Figure 4.1) car la nature des algorithmes génétiques est stochastique.

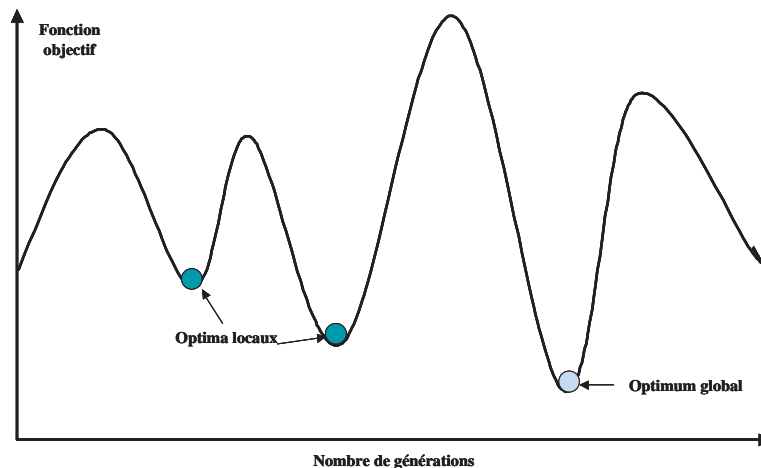


Figure 4.1 – Optima locaux et optimum global.

Dans ce qui suit nous donnons une description de chacun des éléments d'un AG.

4.1.1 Codage des individus

La première étape pour construire un AG est de choisir la représentation génétique la plus appropriée pour coder le domaine du problème à résoudre, c'est-à-dire associer à chacun des points de l'espace de recherche une structure contenant l'information du problème. Étant donné la grande diversité des problèmes, le codage est une opération délicate, car la solution du problème dépend fortement du codage des individus. Il existe deux types de codages [Goldberg, 1989] : le codage binaire et le codage symbolique.

Le codage binaire est la manière la plus connue pour représenter un individu. Un des avantages de cette méthode est que l'on peut représenter tout type d'information sous forme binaire.

Le codage symbolique est utilisé lorsque l'on garde le problème dans sa représentation naturelle, c'est-à-dire un nombre réel reste réel et un nombre entier reste entier. L'avantage est la transparence du codage de la variable du problème pour l'espace de recherche, car il est le même que l'espace du problème. L'inconvénient est l'impossibilité d'utiliser les opérateurs génétiques développés pour le codage binaire, mais il y a des opérateurs spécialisés pour ce type de codage.

Notons qu'il est aussi possible d'utiliser d'autres formes de codage telles que le codage réel et le codage de Gray [Mitchell, 1999; Haupt et Haupt, 2004]. Dans la suite de ce chapitre nous utilisons uniquement le codage binaire.

4.1.2 Génération de la population initiale

La première étape de l'algorithme consiste à générer la population initiale. Une population est constituée de plusieurs individus. Les individus de la population sont des chaînes de bits de longueur λ , et tous les individus ont la même longueur. Une population a N_{pop} individus et est une matrice de taille $N_{pop} \times \lambda$. Chaque ligne de cette matrice correspond donc à un individu initialisé de façon aléatoire avec une chaîne de 1 et 0 [Goldberg, 1989; Mitchell, 1999; Chambers, 2001; Haupt et Haupt, 2004]. La population initiale ne représente alors qu'une petite partie de l'ensemble des solutions possibles du problème à résoudre.

L'unique condition pour générer la population initiale est de créer une population avec beaucoup de diversité. Cela permet de garantir une zone de recherche riche dans laquelle il y aura une grande variété d'individus. On peut envisager la génération d'une population aléatoire ou bien pour des situations bien spécifiques on peut introduire de manière directe la population. La population initiale indique le point de départ pour la génération des nouvelles populations à partir de plusieurs itérations ou cycles génétiques. Dans chaque cycle évolutif, une nouvelle population devient de plus en plus adaptée. Cette évolution de la population est effectuée par des opérateurs génétiques tels que le croisement et la mutation.

4.1.3 Fonction d'aptitude

La clé principale de l'application des AG est de savoir choisir une représentation appropriée, une bonne fonction d'aptitude ainsi que les opérateurs génétiques [Vafaie et Jong, 1993; Goldberg, 1989; Mitchell, 1999; Haupt et Haupt, 2004]. La fonction d'aptitude doit mesurer la qualité de chacune des solutions générées par l'AG. Étant donnée la nature de sélection stochastique, la fonction d'aptitude devra déterminer la probabilité de sélection attribuée à chaque solution.

4.1.4 Croisement

L'évolution de la population à chaque itération (génération) est obtenue à partir du principe darwinien de la sélection des meilleurs individus pour la phase de reproduction. Ce mécanisme est basé sur la fonction d'aptitude, pour cette raison, la fonction d'aptitude

4.1 Généralités sur les algorithmes génétiques

est aussi appelée "habilité à survivre". Typiquement, les individus les meilleurs adaptés ont plus de chances d'être sélectionnés et de participer au croisement.

Le croisement est un opérateur essentiel car il permet l'échange du matériel génétique entre 2 individus de bonne qualité dans le but d'obtenir un ou deux nouveau(x) individu(s) appelé(s) enfant(s).

Le croisement entre 2 individus sélectionnés peut être un croisement à 1 point que nous illustrons dans la Figure 4.2).

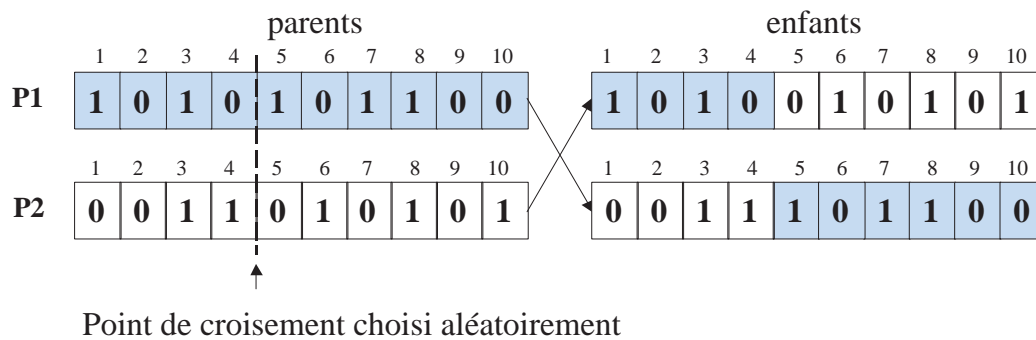


Figure 4.2 – Exemple de croisement à 1 point.

Ce croisement à 1 point suit le mécanisme [Holland, 1975] :

- Partition des individus des parents selon un critère de croisement aléatoire
- Combinaison croisée des parties d'individus des parents pour obtenir deux nouvelles solutions (enfants)

Il existe d'autres façons d'appliquer le croisement :

- Le croisement à 2-points, où les individus (parents) sélectionnés sont coupés en deux points. Chaque point de coupure est choisi de manière aléatoire et les parties centrales des deux parents sont échangés. (Voir Figure 4.3),
- Le croisement à n-points est une généralisation du mécanisme de croisement à 2 points,
- Le croisement uniforme utilise une chaîne de bits (masque) générée de manière aléatoire qui est de la même longueur que les individus. Les gènes des individus sont échangés selon cette chaîne aléatoire. (Voir Figure 4.4)

4.1.5 Mutation

Le rôle de la mutation est d'introduire de nouvelles informations dans les individus d'une population afin d'augmenter la diversité. La mutation cherche donc à éviter la convergence prématurée vers un optimum local de l'algorithme génétique. Cet opérateur change de façon aléatoire un bit ou bien plusieurs sur un individu. La mutation est

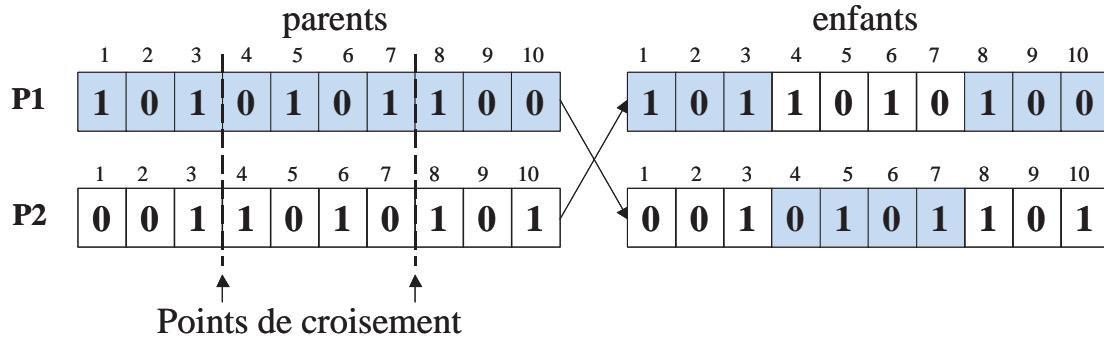


Figure 4.3 – Exemple de croisement en 2 points.

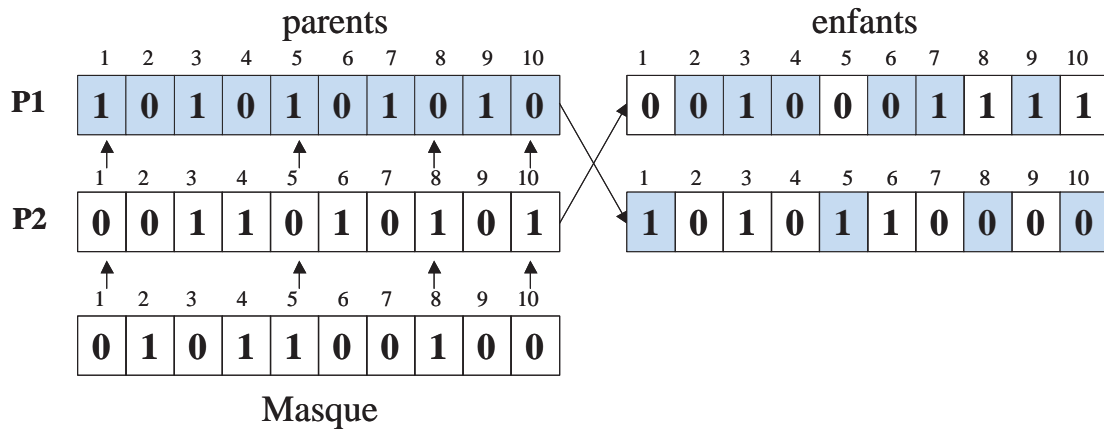


Figure 4.4 – Exemple de croisement uniforme.

souvent appliquée avec une probabilité faible P_m , normalement comprise entre 0.001 et 0.01, cela pour éviter que les individus d'une population évoluent de manière chaotique.

4.1.6 Élitisme

L'opérateur d'élitisme est une méthode de sélection qui permet de retenir seulement les meilleurs individus d'une population pour la génération suivante. Si les individus ne sont pas gardés, ils risquent de ne pas survivre pour les prochains cycles après les opérations de croisement ou la mutation [Mitchell, 1999]. Concrètement l'élitisme consiste donc à copier un pourcentage (normalement choisi comme un paramètre de l'AG) des meilleurs individus dans la nouvelle génération pour éviter qu'ils soient effacés de la population.

4.1 Généralités sur les algorithmes génétiques

Nombre	Individu	fonction d'aptitude (f)	% de surface= $f_i/\sum f$	somme cumulée
1	1000	8	0.1509	0.1509
2	1111	15	0.2830	0.4339
3	0010	2	0.0377	0.4716
4	0011	3	0.0566	0.5283
5	0101	5	0.0943	0.6226
6	1100	12	0.2264	0.8490
7	1000	8	0.1509	1
		somme=53		

Table 4.1 – Individus d'une population fictive avec leurs fonctions d'aptitude.

4.1.7 Probabilités des opérateurs génétiques

Les probabilités de croisement (P_c), de mutation (P_m) et d'élitisme (P_e) sont fixés au début de l'algorithme génétique. Ces probabilités normalement sont déterminées de manière empirique. Cependant, la valeur de probabilité de P_c doit être supérieure à la probabilité de P_m [Goldberg, 1989]. P_m peut être fixé avec une valeur faible afin d'éviter une introduction trop grande de nouvelle information dans la population.

4.1.8 Mécanisme de sélection

Le processus de sélection joue un rôle très important, car il choisit les individus de la génération (t) qui vont se reproduire à la génération ($t + 1$). Il existe plusieurs méthodes de sélection telles que la méthode de la roulette, la méthode du rang ou la méthode de tournoi que nous décrivons ci-dessous.

4.1.8.1 Méthode de sélection de roulette

Cette méthode consiste à simuler une roulette biaisée [Goldberg, 1989]. Chaque individu est représenté par une section de surface de la roue proportionnelle à sa fonction d'évaluation. Ainsi un individu c_i a la probabilité d'être sélectionné comme suit [Lin et Lee, 1996]:

$$P_{\text{sélection-roulette}}(c_i) = \frac{f(c_i)}{\sum_{j=1}^{N_{pop}} f(c_j)} \quad (4.1)$$

où N_{pop} est la taille de la population de l'AG et f est la fonction d'aptitude. À titre d'exemple nous montrons la méthode de la roulette basée sur une population fictive (voir Tableau 4.1).

Le tirage selon cette roulette biaisée se fait en générant un nombre aléatoire \mathcal{A} et en sélectionnant l'individu \mathcal{I} dont la somme cumulée est supérieur égal à \mathcal{A} .

Lorsque l'on fait tourner la roulette en générant un nombre aléatoire, la méthode choisit l'individu dont la somme cumulée est supérieure ou égale au nombre aléatoire.

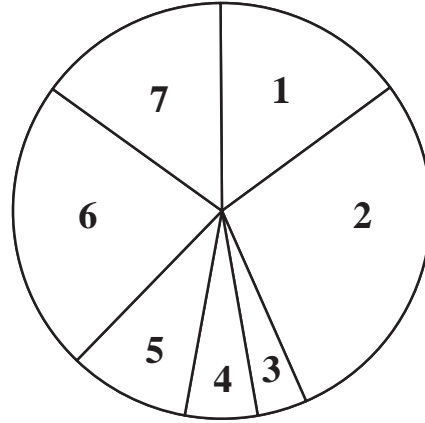


Figure 4.5 – Méthode de la roulette pour l'exemple 3.1.

4.1.8.2 Méthode du rang

La méthode de sélection par rang est une variante de la méthode de la roulette. C'est une méthode alternative pour prévenir la convergence trop rapide de l'algorithme génétique [Mitchell, 1999]. La différence est que la méthode du rang est divisée en deux étapes. La première étape consiste à trier les individus selon leur fonction d'aptitude. Chaque individu est représenté par un rang en fonction de sa position, c'est-à-dire le meilleur individu aura le rang N (taille de la population), le deuxième meilleur aura le rang $N - 1$ et le moins bon aura le rang 1. La deuxième étape consiste en l'implémentation d'une roulette basée sur les rangs des individus. La section de surface de la roue de chaque individu est proportionnelle au rang de l'individu qu'il représente. Cette procédure permet d'attribuer à chaque individu une probabilité de sélection selon son rang.

4.1.8.3 Sélection binaire par tournoi

Deux individus sont tirés de façon aléatoire dans une population P , et celui dont la fonction d'évaluation est la plus haute est sélectionné. Cette approche permet de préserver plus de diversité génétique qu'avec la méthode de la roulette [Chambers, 2001].

4.1.8.4 Sélection steady-state

L'idée principale de cette procédure repose sur l'idée qu'une grande partie de la population puisse survivre dans la prochaine génération. À chaque génération on sélectionne quelques individus parents (parmi ceux qui ont la meilleure fonction d'aptitude) pour créer des individus enfants. Ensuite les individus avec la plus mauvaise fonction d'aptitude sont ôtés et remplacés par les nouveaux enfants, en revanche les autres individus

4.1 Généralités sur les algorithmes génétiques

restants de la population survivent à la nouvelle génération.

4.1.9 Critère d'arrêt

Le processus évolutif d'un AG se termine lorsqu'un critère d'arrêt est atteint. Le critère d'arrêt peut être une des conditions suivantes [Reeves, 2003] :

- Un nombre de générations maximal fixé a été atteint.
- La valeur de la fonction d'aptitude a atteint une valeur fixée a priori s'il s'agit d'un problème de maximisation ou d'un problème de minimisation.
- Pendant plusieurs générations la valeur de la fonction d'aptitude ne change pas, ce qui correspond à un cas de convergence reposant sur l'évolution de la fonction d'aptitude.
- Les individus de la population atteignent un certain degré d'homogénéité.
- Un temps de calcul maximal est atteint.
- Un nombre donné d'évaluations de la fonction est atteint.

Une fois que la condition d'arrêt est satisfaite, les meilleurs individus de la population sont retenus comme les solutions au problème initial.

4.1.10 Construction d'un AG

Algorithme 4.1 : Algorithme génétique

```
1 début
2    $t \leftarrow 1$ 
3   Générer une population aléatoire  $P(t)$  de  $N_{pop}$  individus
4   tant que Continue faire
5     Calculer la fonction d'aptitude  $f(x)$ , pour tout individu  $x$ 
6     tant que NouvellePopulationARemplir faire
7       Sélectionner deux individus parents selon la fonction d'aptitude
8       Appliquer le croisement avec une probabilité ( $Pc$ ) pour avoir un ou deux nouveaux
       enfants
9       Appliquer la mutation sur les enfants avec une probabilité ( $Pm$ )
10      Ajouter le(s) nouveau(x) individu(s) à la nouvelle population  $P(t + 1)$ 
11    fin
12    Remplacer l'ancienne population  $P(t)$  par la nouvelle population  $P(t + 1)$ 
13     $t \leftarrow t + 1$ 
14  fin
15  retourner le ou les meilleurs individus de  $P(t)$ 
16 fin
```

Pour terminer cette présentation générale des AG (Un exemple de la construction d'un AG est montré par l'algorithme 4.1), on peut dire qu'ils sont considérés comme des méthodes d'optimisation applicables à une grande variété de problèmes. L'évolution de la population représente ainsi l'optimisation du problème à résoudre. Les individus de la population sont des points de l'espace de recherche et le codage génétique représente la structure du problème donné. Les opérateurs génétiques permettent une amélioration au

cours du temps de la qualité des individus. Enfin à la différence de l'évolution en biologie les opérateurs tels que le remplacement et l'élitisme sont utilisés fréquemment pour ne pas perdre les meilleures solutions au cours de l'évolution de l'AG.

Un AG standard est donc une méthode un peu aveugle. un AG produit des résultats réellement intéressants lorsqu'on arrive à guider son parcours dans l'espace de recherche. Cela est réalisé grâce à des codages et des opérateurs spécialisés que prennent en compte le savoir-faire du problème. Ce qui se traduira en la construction d'algorithmes génétiques plus adaptatifs et plus efficaces. Actuellement il est courant de les combiner avec d'autres méthodes (par exemple la recherche locale) afin d'obtenir des méthodes plus performantes et applicables à une plus grande variété de problèmes.

4.2 Schéma général de double exploration génétique pour la sélection de gènes

Nous proposons d'utiliser une méthode de double exploration génétique pour la sélection de sous-ensembles de gènes. Pour cela nous utilisons une méthode enveloppe basée sur un classifieur SVM. L'AG est conçu pour générer différents sous-ensembles dans l'espace de recherche qui est de cardinal 2^n . Nous rappelons que nous travaillons avec les n gènes retenus dans la phase de réduction du chapitre précédent. La sélection d'un sous-ensemble de gènes plus petit est l'objectif principal de la tâche de sélection.

Nous souhaitons que l'AG fournisse des sous-ensembles de taille assez petite tout en permettant une haute performance de classification. Si nous avons un sous-ensemble de petite taille la prédiction à partir de données de biopuces sera plus rapide et plus intéressante pour les biologistes. En effet, l'ensemble de gènes sélectionnés donne des pistes de recherche pour la compréhension des maladies étudiées.

La technique centrale dans ce chapitre est une méthode enveloppe où un AG explore des sous-ensembles candidats et chaque candidat est évalué grâce à un classifieur SVM. Le taux de classification indique si le sous-ensemble candidat permet une bonne discrimination des deux classes : cette information est donc la fonction d'aptitude retenue dans l'AG.

On construit une population génétique à partir de l'ensemble de départ et l'on cherche avec notre schéma wrapper de réaliser une première exploration génétique pour trouver des bons sous-ensembles et retenir ceux qui ont un bon taux de classification. Nous pensons que ces sous-ensembles de bonne qualité peuvent être améliorés si nous prenons d'eux les gènes qui ont la plus haute fréquence d'apparitions. Un gène qui est présent dans différents sous-ensembles est un bon candidat pour être un gène pertinent. En plus nous aimerons aussi réduire le nombre de gènes dans l'ensemble final, pour cela nous envisageons d'effectuer une deuxième exploration génétique afin d'utiliser les gènes avec la plus haute fréquence et ainsi fournir un sous-ensemble de bonne qualité.

Nous rappelons que l'évaluation de la performance des sous-ensembles de gènes pendant les deux explorations génétiques est réalisée par le classifieur SVM, lequel permettra de juger si un sous-ensemble de gènes est de bonne qualité ou non.

La procédure générale de notre approche peut être caractérisée comme un processus

4.2 Schéma général de double exploration génétique pour la sélection de gènes

séquentiel en trois-étapes qui utilise des techniques complémentaires pour réduire graduellement l'espace de recherche (voir Figure 4.6). Le reste de cette section donne une description complète de ces trois étapes :

1. Étape 1 "Première exploration génétique pour la sélection de sous-ensembles des gènes à l'aide d'une méthode wrapper AG-SVM".
2. Étape 2 "Analyse d'archives de sous-ensembles de gènes de bonne qualité".
3. Étape 3 "Deuxième exploration génétique de la méthode wrapper".

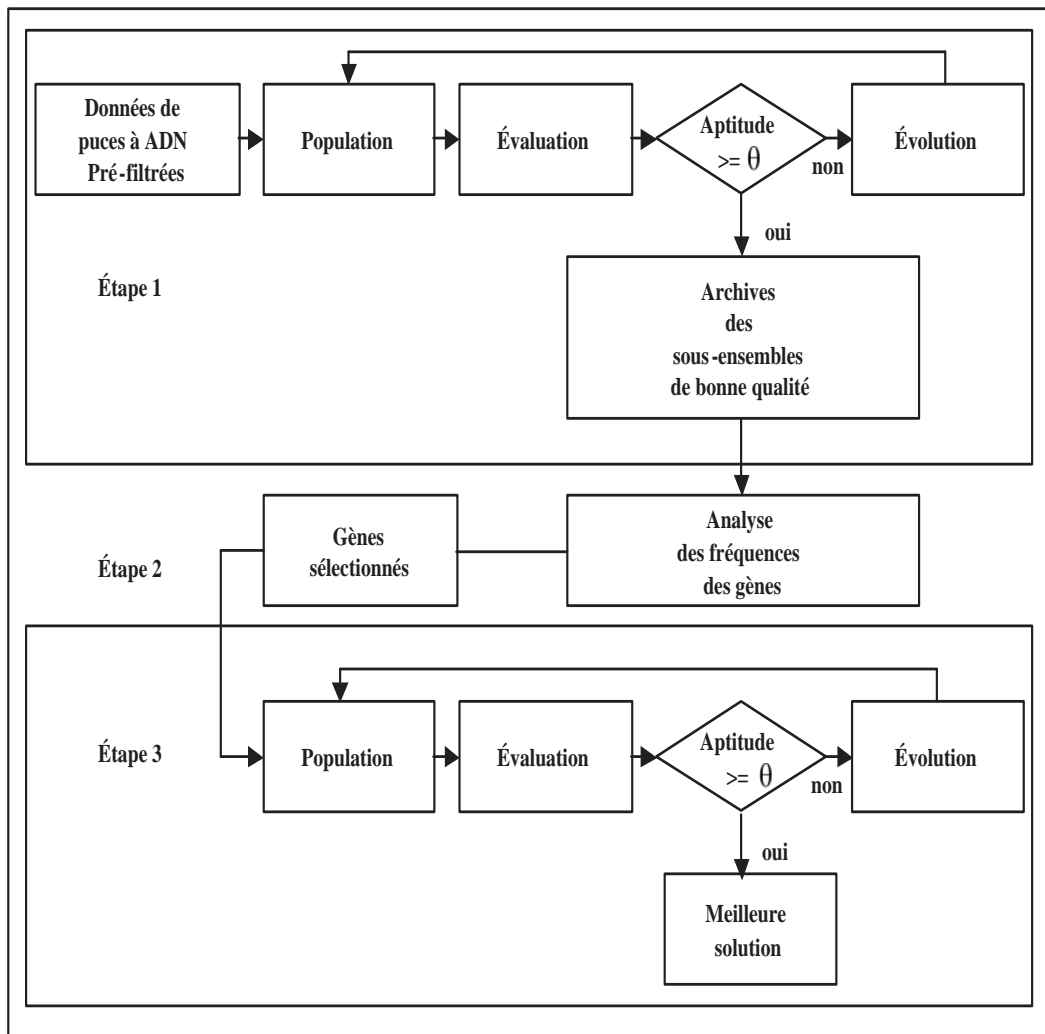


Figure 4.6 – Procédure générale pour le pré-traitement, la pré-sélection et la sélection des sous-ensembles de gènes à l'aide d'une méthode d'enveloppement.

Dans la suite de cette section nous détaillons chacune de ces étapes.

4.2.1 Étape 1 : Première exploration génétique

Un AG est un bon candidat pour étudier le problème de la sélection de gènes. En effet on a vu que la sélection est un problème d'optimisation pour les données de grande taille, dans notre cas les données de puces à ADN sont d'une grande dimension. Lorsque le nombre de gènes est trop élevé et que le nombre d'échantillons est limité, il est important de procéder à une phase de sélection de gènes pour une tâche de classification supervisée. Les algorithmes génétiques ont une grande capacité d'effectuer des recherches dans un grand espace de solutions. Il est légitime d'utiliser des heuristiques pour parcourir cet espace de recherche avec une complexité limitée. Pour cela nous proposons d'utiliser un algorithme génétique pour la recherche d'un bon sous-ensemble parmi une population de sous-ensembles de gènes qui évoluent au cours de temps afin de maximiser la performance de classification.

De plus un codage binaire très naturel permet de représenter un sous-ensemble sélectionné et les opérateurs génétiques permettent de manipuler les sous-ensembles pour obtenir des sous-ensembles de gènes de meilleure qualité.

Pour évaluer chacun de ces sous-ensembles il faudra lancer plusieurs fois le classifieur à utiliser (dans notre cas SVM) afin de déduire une mesure de performance. Pour cela nous utilisons un mécanisme de validation croisée.

Dans cette étape l'espace de recherche est de grande taille et nous voulons chercher seulement des sous-ensembles de gènes de bonne qualité qui sont évalués grâce à l'algorithme de classification SVM. Ainsi, la tâche de l'algorithme génétique sera de générer de sous-ensembles, et la tâche du classifieur SVM sera d'évaluer la qualité de chaque sous-ensemble afin de sauvegarder dans des archives seulement ceux qui ont un bon taux de classification.

Nous détaillons dans la suite les composants de l'approche wrapper AG-SVM.

4.2.1.1 Population initiale et individus

Comme le parcours de l'espace de recherche est effectué par l'AG, dans une population (P) un individu représente un sous-ensemble de gènes qui sont codés de manière binaire, où chaque allèle (bit) de l'individu représente un gène des données de puces à ADN. Si un allèle dans un sous-ensemble a une valeur de "1" cela signifie que ce gène a été sélectionné, par contre une valeur de "0" indique que le gène n'est pas sélectionné dans le sous-ensemble.

Étant donné que la taille des individus dépend de la phase de réduction du chapitre précédent, le nombre de gènes retenus pour chaque jeu de données est différent. Cela indique que pour chaque jeu de données nous aurons différentes longueurs d'individus. Par exemple pour le jeu de données de la Leucémie la longueur de chaque individu est égale à 1360 qui correspond au nombre de gènes retenus par le pré-traitement flou.

Les individus de la population initiale sont générés de façon complètement aléatoire. Il faut aussi dire que la taille de la population doit être constante.

4.2.1.2 Fonction d'évaluation

La fonction d'évaluation f dans notre approche a comme but de mesurer la qualité de classification fournie par un classifieur SVM. Autrement dit un sous-ensemble de gènes permettant un taux élevé de classification est considéré comme un meilleur sous-ensemble que celui donnant un taux faible de classification. Pour chaque individu SG de la population on calcule donc sa fonction d'aptitude en entraînant un SVM avec la représentation associée à ce sous-ensemble de gènes sélectionnés. Pour savoir si cet individu permet de fournir un bon classifieur SVM, il est nécessaire d'appliquer un mécanisme de validation. Nous utilisons un schéma de validation croisée (10-fold) de cet algorithme de classification sur SG . (voir Figure 4.7). Cette technique de validation repose sur l'idée d'un découpage (aléatoire) des données en deux sous-échantillons. Cette méthode simule un processus consistant à développer séparément un modèle sur un ensemble de données (jeu d'apprentissage) et prédire sur un ensemble de données non utilisé (jeu de test).

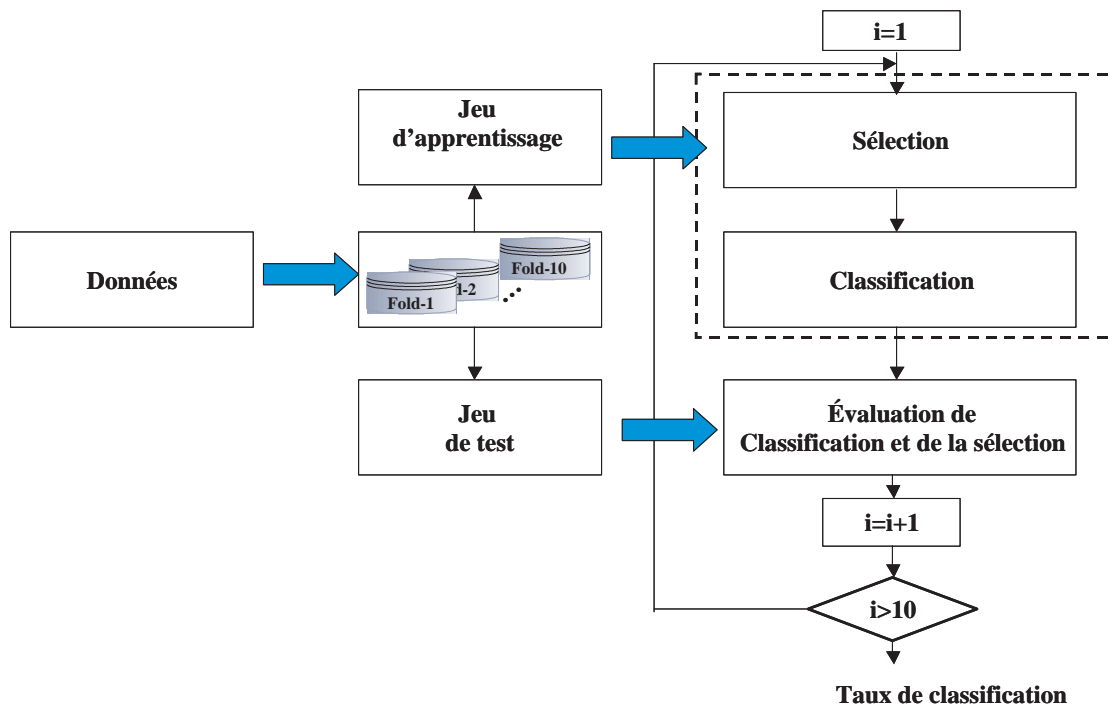


Figure 4.7 – Division des données en 10-blocs. On construit le modèle classifieur sur 9 groupes et on le teste sur le 10-ième groupe. Puis on change de groupe test et on répète le même procédé jusqu'à avoir réalisé 10 combinaisons. On considère alors la moyenne des validations comme le taux de classification.

4.2.1.3 Croisement

La méthode de sélection des parents pour appliquer l'opérateur de croisement est celle de la roulette et le croisement que nous avons implémenté dans l'AG est le croisement en un point (appelé parfois croisement simple). À partir de sous-ensembles de gènes parents, nous fabriquons deux nouveaux sous-ensembles enfants. Dans la figure 4.8 un gène est choisi de manière aléatoire sur la longueur des parents. Dans notre exemple c'est le gène 3 qui est considéré comme le point de coupure. À partir de ce point de coupure on va produire deux morceaux que l'on échange entre les deux parents sélectionnés ($P1$ et $P2$). Les enfants produits par cet échange contiennent chacun donc un morceau qui a été hérité des ces parents.

On peut noter que le nombre de gènes en commun entre les deux parents est égal à 2. Cela veut dire qu'il y a deux gènes pertinents qui sont communes à $P1$ et $P2$. Dans notre exemple, les gènes 3 et 8 figurent dans les deux parents, alors ils sont pertinents dans 2 sous-ensembles de bonne qualité. Notre opérateur tient en compte de cela et nous retrouvons ces 2 gènes dans les 2 enfants. Ils ont survécu grâce à leur capacité de fournir pour deux sous-ensembles de gènes différents une bonne performance de classification. Les enfants donc conserveront les gènes pertinents des parents pour participer à de nouvelles opérations génétiques.

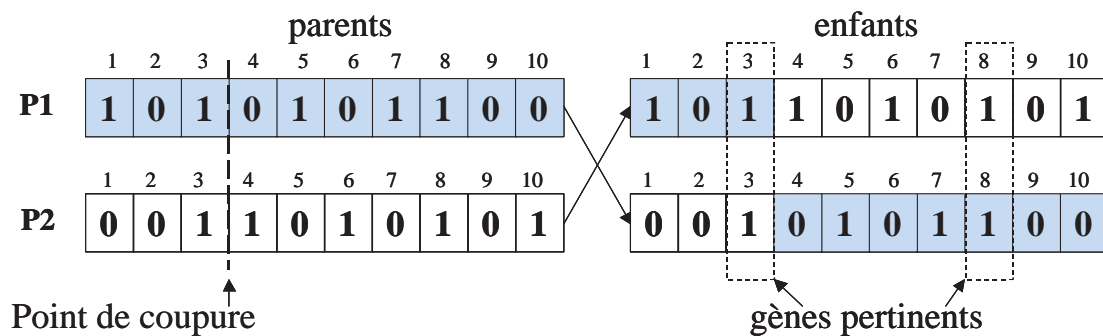


Figure 4.8 – Croisement à 1 points.

On peut remarquer que notre opération de croisement préserve donc les gènes communs aux deux parents.

4.2.1.4 Mutation

La mutation que nous avons implémentée est la mutation en n points choisis au hasard (voir Figure 4.9). La mutation permet ainsi d'insérer ou ôter des gènes dans un sous-ensemble en inversant la valeur du bit associé. Les points de mutation choisis de façon aléatoire (boîtes foncées) indiquent les gènes qui vont changer de valeur. La mutation permet donc de considérer des sous-ensembles différents pour explorer des sous-ensembles de gènes pertinents un peu différents.

4.2 Schéma général de double exploration génétique pour la sélection de gènes

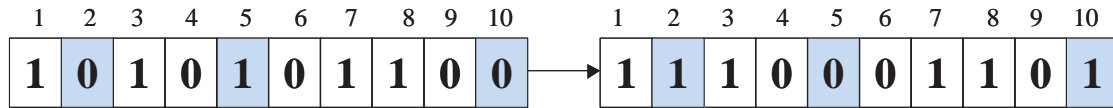


Figure 4.9 – Exemple de mutation en 3 points.

4.2.1.5 Élitisme

Comme on avait expliqué dans la section 4.1.6, l'opérateur élitiste a comme but de préserver les meilleurs sous-ensembles de gènes. Pendant l'évolution d'une population, les opérations de croisement et surtout de la mutation peuvent détruire les individus de bonne qualité. Afin de conserver les individus dont la fonction d'aptitude est haute, nous recopions un pourcentage de la population dans la prochaine génération (Élitisme).

4.2.1.6 Critère d'arrêt

Comme nous l'avons vu dans la présentation générale il est nécessaire de définir un critère d'arrêt de l'exploration. Nous avons défini un seul critère d'arrêt qui est un nombre pré-fixé de générations. Nous remarquons que pour la première exploration génétique le nombre de générations est grand car la longueur des individus est aussi grande pour les jeux de données utilisées. Pour la deuxième exploration ce nombre diminue en même temps que la longueur des individus.

4.2.1.7 Paramètres de l'AG

Dans le cadre de l'étude de notre approche enveloppe nous montrons un exemple de la sélection et classification de sous-ensemble de gènes pour les données de puces à ADN.

Exemple 1 Nous illustrons notre modèle de double exploration génétique en utilisant le jeu de données de la Leucémie. Nous rappelons que la dimension originale pour le jeu de données sur le cancer de la Leucémie est de 7129 gènes, et après notre réduction de dimension à l'aide de la logique floue (voir chapitre 3) nous avons 1360 gènes non-redondants. Les paramètres que nous avons fixés pour la première exploration de la méthode wrapper sont montrés dans le tableau 4.2. Les paramètres du classifieur SVM que nous avons choisis sont le noyau Linéaire avec $C = 100$ pour le paramètre de régularisation. Nous utilisons une taille de population initiale petite (30) et un nombre de générations aussi petit (100). Cela pour mieux comprendre les pas de chaque étape de notre modèle et afin d'illustrer de manière plus claire les résultats graphiques obtenus de notre approche.

4.2.1.8 Archives de sous-ensembles de très bonne qualité

Une caractéristique importante de l'AG développé dans ce travail est l'utilisation d'archives pour enregistrer les "meilleurs sous-ensembles de gènes" découverts pendant

Paramètre	valeur
Longueur Chromosome	1360
Taille de la population	30
Nombre de générations	100
Pc	0.98
Pm	0.01
Elitisme	10%

Table 4.2 – Paramètres pour la première exploration génétique (Étape 1) de l'exemple 1.

la première étape d'exploration génétique. Ces archives seront analysées avant la deuxième exploration génétique pour identifier un plus petit nombre de gènes apparaissant fréquemment dans ces archives.

L'idée de base des archives de haute qualité repose sur le fait que nous voulons conserver les sous-ensembles des gènes plus pertinents de l'AG. Étant donné un individu (un sous-ensemble candidat), la fonction d'aptitude est fournie pour le classifieur SVM qui évalue la qualité d'un sous-ensemble des gènes. Si le taux de classification est assez haut, défini par un seuil de θ , (voir la Figure 4.6 bloc haut), le sous-ensemble de gènes est enregistré dans des archives de haute qualité. En revanche, si le taux de classification ne dépasse pas le seuil fixé, alors le processus d'évolution continue tout en cherchant des nouvelles solutions candidats. Nous fixons un seuil permettant seulement de sauvegarder les meilleurs individus de chaque jeu de données de biopuces. Ce seuil choisi dépend du jeu de données.

Notre modèle envisage la possibilité de chercher parmi les meilleures solutions un sous-ensemble "optimal", c'est-à-dire qu'on va construire une nouvelle population de sous-ensembles de haute qualité pour faire une nouvelle sélection de sous-ensembles. Dans la suite de ce chapitre nous donnerons une description détaillée de ce processus. Avant nous décrivons comment choisir les gènes les plus pertinents des meilleurs sous-ensembles de gènes dans la section suivante.

4.2.2 Étape 2 : Analyse de la fréquence des gènes

Pour analyser les meilleurs sous-ensembles de gènes sauvegardés dans l'archive de haute qualité, nous voulons sélectionner seulement ceux qui ont été considérés à plusieurs reprises dans l'évolution de l'AG. Un gène intéressant sera trouvé plusieurs fois dans différents sous-ensembles de gènes. Chaque sous-ensemble de gènes a un comptage des gènes qui ont été trouvés à plusieurs reprises par la première exploration génétique.

Pour notre exemple de la Leucémie nous avons fixé la valeur de θ (seuil de qualité) à 0.85 et nous présentons pour les résultats offerts avec seulement 100 générations. Cette valeur dans notre modèle permet de filtrer en quelque sorte les sous-ensembles de gènes qui ont une fonction d'aptitude supérieure ou égale à 0.85. Le résultat obtenu par la première exploration génétique donne une archive de sous-ensembles des gènes perti-

4.2 Schéma général de double exploration génétique pour la sélection de gènes

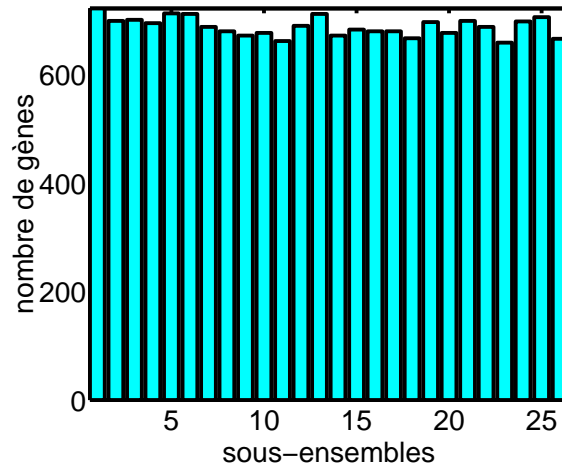


Figure 4.10 – Gènes retenus pour chaque sous-ensemble sauvegarde dans le archive de haute qualité.

nents où il y a 26 sous-ensembles de bonne qualité qui ont été filtrés par notre méthode d'enveloppe. Ces individus de l'archive de sous-ensembles pertinents ont une fonction d'aptitude qui oscille entre 0.8611% et 0.9028% (voir Figure 4.11). Pour cet exemple dans l'archive de haute qualité nous trouvons que le sous-ensemble 17 a la plus haute aptitude égale à 0.9028%. Le sous-ensemble qui a la deuxième meilleure performance pour l'étape 1 est le numéro 5. Nous trouvons aussi qu'il y a 7 individus qui ont une fonction d'aptitude égale à 0.875% et 17 individus avec une performance fournie pour le classifieur de SVM égale à 0.861%.

Pour chaque sous-ensemble il existe un nombre de gènes retenus (voir Figure 4.10), avec une moyenne de 688.26 pour ces 26 sous-ensembles. Nous avons fixé un nombre petit de générations (100) pour notre exemple, cela explique pourquoi on obtient un nombre de gènes de telle taille. Pour réduire encore le nombre de gènes sélectionnés nous utilisons une deuxième exploration génétique, que nous détaillons dans la suite de cette section.

L'analyse de fréquences de cette archive de 26 sous-ensembles de gènes pour la Leucémie montre que le gène 1048 figure dans 24 sous-ensembles et un seul gène (655) ne figure qu'une seule fois dans l'archive (voir Figure 4.12).

Ensuite nous ordonnons la fréquence des gènes et nous choisissons les cent premiers gènes (voir Table 4.3). Nous savons ainsi quels sont les gènes qui vont être considérés pour l'étape 3 (deuxième exploration génétique). Nous rappelons qu'avec le prétraitement flou nous avons réduit la dimension de données et avec cette procédure nous réduisons encore plus la dimension des données.

En raison de nombre encore trop grand de gènes dans chaque sous-ensemble de gènes de bonne qualité, nous proposons une nouvelle recherche génétique. Autrement dit les cent gènes avec la plus haute fréquence seront sélectionnés pour participer à une deuxième exploration afin de trouver des meilleurs sous-ensembles de gènes de bonne

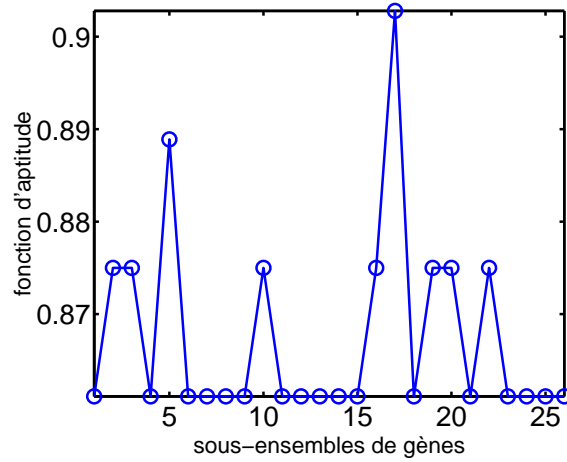


Figure 4.11 – Fonction d'aptitude des sous-ensembles de gènes de la Leucémie qui ont été sauvegardés dans le archive de haute qualité.

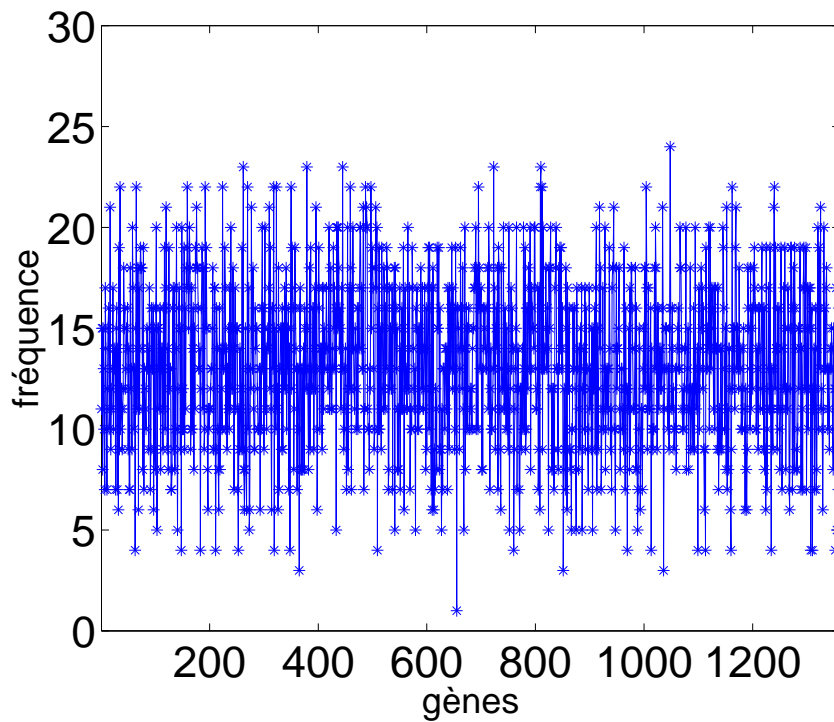


Figure 4.12 – Exemple de la fréquence des gènes pour la Leucémie qui ont été sauvegardés dans l'archive de sous-ensembles pertinents.

qualité et de petit taille. Nous détaillons le processus de sélection des meilleurs sous-ensembles dans la suite de cette section.

4.2 Schéma général de double exploration génétique pour la sélection de gènes

Numéro	Fréquence	gène
1	24	3819
2	23	7123
3	23	6509
4	23	3885
5	23	5060
6	23	6127
7	22	6918
8	22	5634
9	22	7110
.	.	.
.	.	.
.	.	.
100	19	3858

Table 4.3 – Extrait du rang des 100 premiers gènes avec la plus haute fréquence pour le jeu de données de la Leucémie.

4.2.3 Étape 3 : Deuxième exploration génétique

Dans cette deuxième exploration génétique nous utilisons le même principe de sélection par la méthode enveloppe et l'ensemble initial de gènes est fixé à 100 gènes retenus grâce à l'exploitation de l'archive. Parmi ces cents gènes avec la plus haute fréquence nous devons trouver un sous-ensemble plus performant avec des gènes de meilleure qualité. Pour cela notre GA-SVM travaille avec des individus de longueur égale à 100. La procédure de sélection et ainsi que les opérateurs génétiques utilisés dans cette deuxième exploration sont les mêmes que dans la première exploration (plus de détails voir tableau 4.4).

Paramètre	valeur
Longueur Chromosome	100
Taille de la population	30
Nombre de générations	50
Pc	0.98
Pm	0.01
Elitisme	10%

Table 4.4 – Paramètres pour deuxième exploration génétique (Étape 3) de l'exemple 1.

Pour cette deuxième exploration génétique l'espace de recherche est plus petit (2^{100}) que dans la première exploration (2^{1360}), cela permet de trouver plus rapidement un sous-ensemble de bonne qualité. De plus la génération de sous-ensembles de gènes est effectuée avec les gènes les plus pertinents de la première exploration. Autrement dit le

sous-ensemble qu'on va trouver à la fin de cette recherche doit être composé de gènes très intéressants pour le jeu de données utilisé. On constate en effet que la performance de sous-ensemble de gènes obtenu après la deuxième exploration est plus haute que celle des sous-ensembles de gènes sauvegardés dans l'archive de haute qualité. Dans la première exploration nous avons trouvé un sous-ensemble de bonne qualité avec une performance de 90.28% avec 680 gènes. Pour la deuxième exploration génétique le sous-ensemble trouvé avec la plus haute performance est de 93.06% avec seulement 49 gènes. Nous constatons une réduction significative de nombre de gènes ainsi qu'une amélioration du taux de classification. Cela justifie l'intérêt des archives de haute qualité.

4.3 Résultats sur 5 exécutions

Pour l'évaluation de notre approche enveloppe, nous avons utilisé d'abord deux jeux de données de puces à ADN. Nous avons implémenté l'AG dans Matlab (Version 5.3.1 for Windows). Le classifieur SVM que nous avons choisi est celui développé par Gavin Cawley¹. Les tableaux 4.5 et 4.6 récapitulent le paramétrage utilisé pour l'AG.

Paramètres	Leucémie	Colon
Longueur chromosome	1360	943
Taille de la population	500	500
Nombre de générations	2500	2500
Pc	0.95	0.98
Pm	0.02	0.01
Élitisme	10%	15%
Gènes filtrés de l'archive	100	100

Table 4.5 – Paramètres pour la première exploration génétique (Étape 1).

Paramètres	Leucémie	Colon
Longueur chromosome	100	100
Taille de la population	50	50
Nombre de générations	500	500
Pc	0.985	0.985
Pm	0.02	0.01
Elitisme	15%	15%

Table 4.6 – Paramètres pour la deuxième exploration génétique (Étape 3).

Le tableau 4.7 montre les résultats obtenus par notre approche. Nous montrons la moyenne sur 5 exécutions indépendantes qui sont comparés avec les résultats de sept

¹<http://theoval.sys.eua.uk/~gcc/svm/toolbox>

4.4 Résultats sur 10 exécutions

méthodes représentatives dans la littérature de la sélection et la classification des données de puces à ADN. Le critère de comparaison est le taux de classification obtenu (premier nombre) ainsi que le nombre de gènes (le nombre avec parenthèse "-" indique que le nombre de gènes n'est pas disponible). Il faut souligner que nous utilisons la méthode de validation croisée 10-FOLD, ce qui n'est pas nécessairement le cas dans tous les travaux de références.

Auteur	Leucémie	Colon
[Furey <i>et al.</i> , 2000]	94.10(-)	90.3(-)
[Wang <i>et al.</i> , 2005]	100(8)	91.9(-)
[Peng <i>et al.</i> , 2003]	100(6)	93.5(12)
[Shi et Chen, 2005]	95.0(-)	91.0(-)
[Reddy et Deb, 2003]	100(4)	97.0(7)
Notre Approche	100(14)	99.4(10)

Table 4.7 – Comparaison des plus pertinents travaux sur la classification de puces à ADN avec notre approche d'enveloppe.

Nous observons que pour le jeu de données de la Leucémie nous obtenons un taux de classification de 100%, qui est meilleur que ceux rapportés dans [Furey *et al.*, 2000; Shi et Chen, 2005]. Nous trouvons la même performance dans [Wang *et al.*, 2005; Peng *et al.*, 2003; Reddy et Deb, 2003; Guyon *et al.*, 2002], avec moins de gènes. Dans les travaux de [Reddy et Deb, 2003] et [Guyon *et al.*, 2002] est rapporté le nombre minimal de gènes pour ce jeu de données (4). Cependant dans [Reddy et Deb, 2003] ils utilisent les premiers 50 gènes qui ont été proposés dans [Golub *et al.*, 1999], et son approche est basée sur l'idée de trouver la meilleure combinaison minimale de gènes à partir des gènes déjà sélectionnés.

Le résultat le plus intéressant est celui qui concerne le jeu de données du Colon, car avec notre approche nous arrivons à trouver la plus haute performance (99.41%). Nous notons que le nombre de gènes est plus grand que ceux obtenus par [Reddy et Deb, 2003] ou ceux obtenus par [Wang *et al.*, 2005; Guyon *et al.*, 2002], mais il est plus petit que ceux rapportés dans [Peng *et al.*, 2003].

Pour les autres jeux de données nous ne pouvons pas se comparer avec ces travaux car ils seulement ont travaillé avec les jeux concernant la Leucémie et le cancer du Colon.

4.4 Résultats sur 10 exécutions

Plus récemment nous avons conduit des nouvelles expériences avec plus de jeux de données de puces à ADN et en effectuant 10 exécutions indépendantes pour se comparer aux autres travaux publiés dans la littérature (Voir tableau 4.8). Ces résultats selon le taux de classification obtenu (premier nombre) ainsi que le nombre de gènes. Nous remarquons que nous utilisons la méthode de validation croisée 10-FOLD, avec le classifieur linéaire SVM SPIDER pour matlab développé par Jason Weston, Andre Elisseeff,

Gökhan Bakır et Fabian Sinz ².

Pour faciliter la visualisation du tableau nous proposons des abréviations pour les jeux de données de puces à ADN :

- Leucémie : LEU.
- Colon : COL.
- DLBCL : DLB.
- CNS : CNS.
- Poumon : PMN.
- Prostate : PRT.
- Ovarien : OVA.

Auteur	LEU	COL	PMN	PRT	CNS	OVA	DLB
[Ye <i>et al.</i> , 2004]	97.5	85.0	-	92.5	-	-	-
[Liu <i>et al.</i> , 2004]	100 (30)	91.9(30)	100 (30)	97.0 (30)	-	99.2(75)	98(30)
[Tan et Gilbert, 2003]	91.1	95.1	93.2	73.5	88.3	-	-
[Ding et Peng, 2005]	100	93.5	97.2	-	-	-	-
[Hu <i>et al.</i> , 2006]	94.1	83.8	-	-	-	-	-
[Cho et Won, 2007]	95.9 (25)	87.7(25)	-	-	-	-	93.0(25)
[Yang <i>et al.</i> , 2006]	73.2	84.8	-	86.88	-	-	-
[Peng <i>et al.</i> , 2006]	98.6 (5)	87.0(4)	100 (3)	-	-	-	-
[Wang <i>et al.</i> , 2006]	95.8 (20)	100 (20)	-	-	-	-	95.6(20)
[Huerta <i>et al.</i> , 2006]	100	91.4	-	-	-	-	-
[Pang <i>et al.</i> , 2007]	94.1(35)	83.8(23)	91.2(34)	-	65.0(46)	98.8(26)	-
[Li <i>et al.</i> , 2007]	97.1(20)	83.5(20)	-	91.7(20)	68.5(20)	99.9 (20)	93.0(20)
[Zhang <i>et al.</i> , 2007]	100 (30)	90.3(30)	100 (30)	95.2(30)	80(30)	-	92.2(30)
[Yue <i>et al.</i> , 2007]	83.8(100)	85.4(100)	-	-	-	-	-
[Hernandez <i>et al.</i> , 2007]	91.5(3)	84.6(7)	-	-	-	-	-
[Wang <i>et al.</i> , 2007]	100 (375)	93.5(35)	-	-	-	-	-
[Li <i>et al.</i> , 2008a]	100 (7)	93.6(15)	-	-	-	-	-
Notre Approche	100 (14)	95.1(5)	100 (9)	95.0(20)	98.3 (7)	95.5(10)	98.8 (16)

Table 4.8 – Résultats de approche de double exploration génétique (dernière ligne) comparé avec les travaux les plus importants de la classification de données de puces à ADN. Taux de classification en parenthèse si le nombre de gènes est disponible.

Les performances conduites dans cette deuxième expérimentation sur les 7 jeu de données de biopuces montrent que une l'utilisation de la technique de validation 10-FOLD sont plus précises aux celles rapportes dans le tableau 4.7. En effet il faut relancer le processus de sélection pour les 10 itérations de la validation croisée afin de valider les résultat et cela est une tâche très lourde pour notre approche de double exploration génétique.

Nous notons que 4 de 7 résultats donnent une très bonne performance, c'est le cas de la Leucémie, Poumon, CNS et DLBCL. Nous notons aussi que pour les autre jeux comme celui du Colon nous obtenons la deuxième meilleure performance avec seule-

²<http://www.kyb.tuebingen.mpg.de/bs/people/spider/>

4.5 Synthèse du chapitre

ment 5 gènes, la meilleure est obtenu par [Wang *et al.*, 2006] mais ils utilisent plus de gènes (20) que notre approche.

En ce qui concerne le jeu de données de la Prostate nous avons la troisième meilleure performance 95.0% avec 20 gènes. Nous notons que la première et deuxième meilleures performances nécessitent plus de gènes que notre approche (30). En utilisant 20 gènes nous avons un taux de classification meilleur que celui listé dans [Li *et al.*, 2007].

Dans le cas Ovarien nous avons une bonne performance par rapport aux autres travaux, si nous nous comparons avec le nombre de gènes utilisés nous obtenons la meilleure sélection de gènes (10) avec notre approche.

Nous considérons qu'il est plus souhaitable pour les biologistes avoir un petit nombre de gènes que un grand nombre de gènes, car cela permet d'analyser mieux la signification biologique de chaque gène.

4.5 Synthèse du chapitre

Nous avons montré les résultats obtenus de notre modèle d'enveloppe de double exploration génétique pour la sélection et classification de gènes. Nous avons confirmé qu'en retenant les gènes les plus pertinents de la première exploration génétique dans des archives de bonne qualité, il était possible de trouver des sous-ensembles de gènes plus pertinents dans une deuxième exploration génétique. Cette double exploration a permis de conduire à une réduction considérable de la taille du sous-ensemble final. Obtenir un petit nombre de gènes dans le sous-ensemble final est très important, car cela permet une analyse biologique cherchant à comprendre le rôle de chaque gène.

Dans la mesure où il est important de connaître la pertinence des gènes pour avoir une bonne performance nous fournissons dans le chapitre suivant une approche qui introduit une valeur de pertinence à chaque gène en utilisant les opérateurs génétiques pour mieux guider le parcours de l'AG dans une méthode de type enveloppe comme celui présenté dans ce chapitre.

Pour des explications plus détaillées des AG, le lecteur intéressé pourra consulter aux références suivantes : [Mitchell, 1999; Chambers, 2001; Haupt et Haupt, 2004].

Chapitre 5

Sélection d'attributs à l'aide d'une méthode d'approche intégrée

Sommaire

5.1	Analyse Discriminante Linéaire Généralisée	92
5.1.1	Analyse Discriminante Linéaire (ADL)	92
5.1.2	Cas d'un petit nombre d'échantillons	93
5.1.3	Pseudo-inverse	94
5.1.4	Décomposition en valeurs singulières (DVS)	94
5.2	Une approche intégrée pour la sélection de gènes basée sur ADL	95
5.2.1	Caractéristiques générales de l'AG	96
5.2.2	Représentation des individus	97
5.2.3	Fonction d'évaluation	98
5.2.4	Croisement ET basé sur ADL	99
5.2.5	Mutation basée sur ADL	101
5.3	Protocole expérimental et résultats	102
5.3.1	Évaluation des performances par un classifieur ADL	103
5.3.2	Évaluation des performances SVM	105
5.4	Comparaison de nos résultats avec d'autres travaux	107
5.5	Validation biologique des résultats	110
5.6	Synthèse du chapitre	111

L'ANALYSE Discriminante Linéaire (ADL) est une technique utilisée dans de nombreux domaines qui sert à déterminer la contribution des variables qui expliquent l'appartenance d'éléments à un groupe. Nous proposons l'utilisation de ADL pour la sélection de sous-ensemble de gènes les plus discriminants dans une approche intégrée (embedded). Pour explorer l'espace de sous-ensembles de gènes ayant une bonne performance et de petite taille notre approche intégrée utilise un Algorithme Génétique. ADL est proposée comme classifieur et aussi comme une méthode de discrimination qui permet d'introduire une connaissance propre au problème de la sélection dans les opérateurs de croisement et mutation. Cela permet une meilleure exploration génétique des sous-ensembles dans l'espace des sous-ensembles de gènes de puces à ADN.

Nous pouvons voir notre approche comme une interaction intéressante entre deux méthodes (Wrapper-ADL): Le wrapper est conçue comme une stratégie pour mieux guider l'exploration de l'espace de solutions à l'aide d'un classifieur pour obtenir des sous-ensembles d'attributs pertinents. ADL est conçu comme une stratégie de discrimination qui aide à l'exploitation de l'information contenue dans chaque sous-ensemble d'attributs trouvés par le wrapper. Cette exploitation est effectuée par les opérateurs génétiques tels que le croisement et la mutation afin de ne retenir que les gènes moins redondants dans un sous-ensemble, c'est-à-dire ceux qui s'expriment différemment selon le critère ADL. Notre approche se focalise sur la sélection de sous-ensembles de gènes vis-à-vis des notions de pertinence et de redondance.

Dans la première section nous présentons d'abord la théorie de l'analyse discriminante. La deuxième section est consacrée à expliquer l'approche pour la sélection d'attributs à l'aide d'une méthode intégrée où l'analyse discriminante linéaire de Fisher (ADL) est combinée avec le schéma des algorithmes génétiques (AG). Nous montrons dans la troisième section le cadre d'expérimentations effectuées sur différents jeux de données de puces à ADN.

5.1 Analyse Discriminante Linéaire Généralisée

5.1.1 Analyse Discriminante Linéaire (ADL)

Le but de l'analyse discriminante est de trouver un nouvel axe, c'est-à-dire une combinaison linéaire des variables, permettant de séparer au mieux deux groupes.

ADL est la méthode de classification linéaire la plus couramment employée pour la réduction de dimensions. Les données d'entrée sont projetées dans un espace de dimensions plus petit où les classes sont bien séparées. Nous emploierons cette technique pour des problèmes de classification binaire. Nous considérons donc un ensemble de n échantillons qui appartiennent à deux classes C_1 et C_2 , parmi lesquels n_1 échantillons appartiennent à la classe C_1 et n_2 échantillons appartiennent à la classe C_2 . Chaque échantillon est décrit par p variables. Ainsi les données forment une matrice $(X = (x_{ij}), i = 1, \dots, n; j = 1, \dots, p)$. Nous définissons μ_k comme la moyenne de la classe C_k et μ la moyenne (ou centroïde) de tous les échantillons comme suit :

$$\mu_k = \frac{1}{n_k} \sum_{x_i \in C_k} x_i$$

$$\mu = \frac{1}{n} \sum_i x_i = \frac{1}{n} \sum_k n_k \mu_k$$

Les données sont décrites par deux matrices S_B et S_W , où S_B est la matrice de dispersion inter-classe et S_W la matrice de dispersion intra-classe, définies comme suit :

$$S_B = \sum_k n_k (\mu_k - \mu)(\mu_k - \mu)^t \quad (5.1)$$

$$S_W = \sum_k \sum_{x_i \in C_k} (x_i - \mu_k)(x_i - \mu_k)^t \quad (5.2)$$

Si nous notons S_V la matrice de covariance pour toutes les données nous avons $S_V = S_B + S_W$ (Théorème de Huygens). ADL recherche une combinaison linéaire des variables initiales de telle sorte que les moyennes des deux classes soient séparées le mieux possible. Pour cela ADL détermine un vecteur w tel que $w^t S_B w$ soit maximisé alors que $w^t S_W w$ est minimisé. Pour réaliser ce double objectif, on cherche à maximiser la quantité :

$$J = \frac{w^t S_B w}{w^t S_W w} \quad (5.3)$$

On montre que le vecteur w_{opt} qui maximise J est le vecteur propre associé à la plus grande valeur propre de $S_{W^{-1}} S_B$ si $S_{W^{-1}}$ est inversible. Ce vecteur w_{opt} détermine un axe de projection qui donne la meilleure séparation des données. Une fois cet axe w_{opt} obtenu, ADL fournit une procédure de classification : une nouvelle donnée est projetée sur cet axe et classée dans C_1 ou C_2 selon la proximité avec les centroïdes de C_1 et C_2 . ADL nous fournit également des informations sur la contribution de chacun des p attributs à la discrimination des classes. En effet, w_{opt} s'exprime comme une combinaison linéaire des p variables : $w_{opt} = (w_{opt}^i)_{i=1}^p$, la valeur absolue de w_{opt}^i indique l'importance de la i -ème variable pour l'axe de discrimination w_{opt} . Les valeurs absolues des coordonnées de w_{opt} sont donc des informations sur l'importance de chaque variable pour la discrimination. Ces informations seront utilisées par notre processus de sélection.

5.1.2 Cas d'un petit nombre d'échantillons

Le vecteur w_{opt} ne peut être calculé si la matrice S_W est singulière (non inversible) car on ne peut donc pas calculer $S_{W^{-1}}$. Cela se produit notamment si le nombre d'attributs (p) est inférieur au nombre d'échantillons (n). Pour les données de puces à ADN ce problème est présent et est appelé "Small Sample Size (SSS)" [Ye et Li, 2004; Ye *et al.*, 2004; Cevikalp *et al.*, 2005; Ye, 2005]. Pour résoudre le problème de singularité présente dans ADL, différentes approches ont été proposées dans la littérature comme "ACP+LDA" [Belhumeur *et al.*, 1997], la perturbation aux valeurs singulières [Ben-Israel et Greville, 2003], "uncorrelated LDA" [Ye *et al.*, 2004; Ye, 2005], "orthogonal LDA" [Ye, 2005], "null

space" [Yue *et al.*, 2007] entre autres. Les approches de [Ye *et al.*, 2004; Ye, 2005] utilisent la technique de la pseudo-inverse pour résoudre le problème de singularité, et nous utilisons cette technique dans notre approche intégrée.

5.1.3 Pseudo-inverse

La pseudo-inverse est une généralisation de l'inverse d'une matrice dans les cas de matrices singulières. Pour toute matrice $A \in \mathbb{R}^{m \times n}$ il existe une matrice nommée pseudo inverse de A notée $A^+ \in \mathbb{R}^{n \times m}$, où le sigle + décrit l'inversion matricielle de Moore-Penrose [Golub et Loan, 1996]. Cette matrice doit satisfaire aux conditions suivantes [Golub et Reinsch, 1970] :

1. $AA^+A = A$
2. $A^+AA^+ = A^+$
3. $(AA^+)^t = AA^+$
4. $(A^+A)^t = A^+A$

La matrice pseudo-inverse A^+ a la propriété d'être égale à l'inverse de A si elle existe. Dans notre cas la pseudo inverse S_W^+ de S_W sera utilisée pour obtenir le vecteur propre associé à la valeur propre de $S_W^+S_B$ dans l'équation 5.3. La pseudo-inverse de la matrice S_W peut être calculée par Décomposition en Valeurs singulières (DVS) [Golub et Loan, 1996; Golub et Reinsch, 1970; Ben-Israel et Greville, 2003]. Dans la suite nous décrivons le processus DVS.

5.1.4 Décomposition en valeurs singulières (DVS)

Avant d'introduire les notions de la décomposition en valeurs singulières nous donnons dans la suite la définition de valeurs singulières.

Les valeurs singulières d'une matrice rectangulaire $A \in \mathbb{R}^{m \times n}$ sont les racines carrées non négatives (ou bien nulles) des valeurs propres de la matrice A^tA , elles sont notées σ_i . Ces racines sont rangées par ordre décroissant [Golub et Reinsch, 1970]. Si toutes les valeurs singulières sont supérieures à zéro, alors la matrice A sera inversible.

Théorème 1.1 Décomposition en valeurs singulières d'une matrice. Pour toute matrice $A \in \mathbb{R}^{m \times n}$, alors il existe deux matrices unitaires orthogonales $U \in \mathbb{R}^{m \times m}$ et $V \in \mathbb{R}^{n \times n}$ telles que [Golub et Loan, 1996; Golub et Reinsch, 1970; Ben-Israel et Greville, 2003] :

$$U^tAV = \text{diag}(\sigma_1, \dots, \sigma_l) \in \mathbb{R}^{m \times n}$$

avec $l^1 = \min\{m, n\}$. Alors la décomposition en valeurs singulières de la matrice A est :

$$A = U\Sigma V^t \tag{5.4}$$

avec Σ définie comme suit:

¹ Une matrice $A \in \mathbb{R}^{m \times n}$ possède au moins 1 et au plus $l = \min(m, n)$ valeurs singulières distinctes

$$\Sigma = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} = \left[\begin{array}{cccc|c} \sigma_1 & 0 & \dots & 0 & \\ 0 & \sigma_2 & & & \\ \vdots & & & \ddots & \vdots \\ 0 & & & & \sigma_l \\ \hline 0 & \dots & 0 & 0 & 0 \\ 0 & \dots & 0 & 0 & 0 \end{array} \right]$$

où D est une matrice diagonale de dimension $l \times l$ dont les éléments diagonaux sont les valeurs (σ_k) . Pour obtenir la pseudo-inverse de A il suffit de calculer sa décomposition en valeurs singulières pour former la matrice diagonale Σ^+ . Cette matrice prend comme éléments diagonaux les inverses des valeurs singulières (σ_i) de la matrice Σ s'ils ne sont pas nuls sinon tous les éléments nuls de la diagonale Σ sont inchangés dans Σ^+ . La pseudo inverse de la matrice A est obtenue par :

$$A^+ = V\Sigma^+U^t. \quad (5.5)$$

où Σ^+ est une matrice diagonale $l \times l$ définie comme suit :

$$\Sigma^+ = \text{diag}(\sigma_1^+, \sigma_2^+, \dots, \sigma_p^+) \quad (5.6)$$

Avec $l = \min(m, n)$.

et

$$\sigma_i^+ = \begin{cases} 1/\sigma_i & \text{pour } \sigma_i > 0 \\ 0 & \text{pour } \sigma_i = 0 \end{cases}$$

On peut vérifier que :

$$((A)^t A)V = V\Sigma^+ \quad (5.7)$$

$$(AA)^t U = U\Sigma^2 \quad (5.8)$$

Ce qui montre que les valeurs singulières sont bien les racines carrées des valeurs propres de $(A^t A)$ et de AA^t . U est la matrice composée de n vecteurs propres orthonormés liés aux valeurs propres AA^t et V la matrice composée des vecteurs propres de $A^t A$.

Nous utilisons la pseudo-inverse comme une technique pour résoudre le problème de singularité lorsque S_W n'est pas inversible.

5.2 Une approche intégrée pour la sélection de gènes basée sur ADL

Dans ce chapitre, nous voulons proposer une approche exploratoire à base de méthodes évolutives réalisant en une seule passe la sélection d'un sous-ensemble de gènes. Pour cela nous devons proposer des informations pour mieux guider les processus d'évolution, en particulier proposer des opérateurs spécifiques. Nous proposons

donc une approche intégrée s'appuyant sur le classifieur ADL. Le classifieur permettra l'évaluation des sous-ensembles candidats mais de plus il offrira des informations sur l'importance des gènes pour la classification. Cette information de pertinence sera utilisée dans l'opérateur de croisement et dans l'opérateur de mutation. Une autre contribution de ce chapitre est de proposer une fonction d'évaluation bi-critère qui permettra de sélectionner des sous-ensembles de gènes de bonne performance et de petite taille. Nous voulons ainsi établir un compromis entre deux critères : le premier indique la performance des sous-ensembles dans l'espace de solutions et le second se focalise sur les sous-ensembles de bonne performance qui ont un nombre réduit de gènes. Le nombre de gènes est un critère important car il est plus facile pour les biologistes d'analyser un petit nombre de gènes de bonne qualité. ADL permet pendant tout le processus de sélection et de classification l'obtention de ces gènes de bonne qualité, comme nous allons le décrire dans la suite en présentant les opérateurs génétiques utilisés.

Nous voulons rappeler qu'avant la recherche de la méthode intégrée, une méthode de filtrage est d'abord appliquée pour obtenir un groupe ordonné G_p des gènes selon leur score de pertinence (typiquement $p = 100$) afin de réduire la dimension initiale des données. Le critère qui nous avons choisi est la méthode t -statistique car nous avons conduit différentes expériences de classification en utilisant B/W, t -statistique et le test de Wilcoxon et nous trouvons qu'en sélectionnant 100 gènes avec un classifieur ADL et SVM nous avons trouvé que ma méthode de filtrage t -statistique est la plus performante.

Ensuite, un AG est utilisé pour conduire une recherche combinatoire dans l'espace 2^p défini par ces gènes. Le but de cette recherche est de sélectionner des sous-ensembles de gènes de petite taille ayant le taux de classification le plus élevé dans le jeu d'apprentissage. Dans notre approche, ADL est employée non seulement pour évaluer un sous-ensemble de gènes candidat, mais également pour fournir aux opérateurs génétiques une information utile pour qu'ils puissent effectuer des opérations intéressantes pendant le processus de sélection.

5.2.1 Caractéristiques générales de l'AG

Les principaux composants de notre approche sont décrits ci-dessous :

- a) **Population initiale** La population initiale de l'AG est produite de façon aléatoire. Dans chaque individu de la population entre 60% et 70% de gènes sont sélectionnés. Dans notre cas nous générons une population avec cette probabilité pour avoir une diversité de sous-ensembles de grande taille et éviter la convergence très rapide vers un minimum local.
- b) **Évolution** Les individus de la population courante P sont triés selon la fonction d'évaluation (voir la section 4.3). Les 10% des "meilleurs" individus de P sont directement insérés dans la prochaine population P' . Cette stratégie élitiste a pour but de protéger les bons individus des effets négatifs des opérations génétiques, en particulier de la mutation. Les 90% des individus restants dans P sont alors produits en employant le croisement et la mutation. Remarquons que les individus de la population élitiste participent aux opérations génétiques.

- c) **Opérateurs génétiques** Les opérateurs de croisement et de mutation s'articulent autour de ADL afin de donner une importance aux différents attributs trouvés dans différents sous-ensembles de gènes obtenus au cours de la recherche dans l'espace de solutions.
- d) **Fonction d'évaluation** Pour évaluer la qualité d'un sous-ensemble nous évaluons deux critères : 1) la performance de classification du sous-ensemble et 2) le nombre de gènes.
- e) **Condition d'arrêt** Le processus d'évolution s'arrête lorsqu'un nombre prédéfini de générations est atteint ou quand on trouve un sous-ensemble réduit à un seul gène dans la population élitiste (les 10% des meilleurs individus). Cette dernière condition signifie que l'on a trouvé un unique gène qui assure une très bonne précision de la classification. Pour certains jeux de données cela peut arriver.

Dans la suite de cette section nous décrivons la procédure d'interaction entre la méthode intégrée et le classifieur dans le parcours d'exploration de bons sous-ensembles de gènes.

5.2.2 Représentation des individus

Nous commençons par décrire la première partie de la construction d'un AG : le codage. Normalement, un individu est simplement employé pour représenter un sous-ensemble candidat de gènes. Dans notre AG, un individu code plus d'informations. Pour mieux comprendre cela, nous définirons un individu comme un couple:

$$I = [\tau|\phi]$$

où τ et ϕ sont définis comme suit : d'abord la première partie de l'individu (τ) est un codage binaire qui représente un sous-ensemble de gènes. Chaque bit τ_i indique si un gène g_i a été sélectionné pour participer à l'évolution de l'AG ($\tau_i = 1$) ou non ($\tau_i = 0$). La deuxième partie de l'individu (ϕ) est un vecteur de valeurs réelles qui correspondent au coefficient de discrimination des gènes figurant dans le sous-ensemble τ . Nous appliquons ADL pour attribuer un coefficient ϕ_i au gène g_i . Nous rappelons que ce coefficient est obtenu à partir du vecteur propre associé à la plus grande valeur propre de la matrice $S_{W-1}S_W$:

$$I = [\tau_1, \tau_2, \dots, \tau_p | \phi_1, \phi_2, \dots, \phi_p]$$

La longueur des codages : τ et ϕ est définie par p , c'est-à-dire le nombre de gènes pré-sélectionnés par une méthode de filtre. Il faut noter que ce codage est plus général et plus riche que ceux utilisés dans la plupart des algorithmes génétiques pour la sélection d'attributs, dans le sens où il inclut une information importante : les coefficients discriminants de ADL, qui sont utiles pour concevoir des opérateurs spécialisés de croisement et de mutation (voir Figure 5.1).

Au cours de l'évolution de l'AG, nous utilisons les coefficients discriminants pour faciliter la recherche vers le sous-ensemble de gènes optimal et de petite taille. Nous rappelons que les AG sont des méthodes stochastiques qui parcourent l'espace de recherche

g_1	g_2	g_3	g_4	g_5	g_6	...	g_n	g_1	g_2	g_3	g_4	g_5	g_6	...	g_n
1	0	0	1	0	1	0	1	Φ_1	0	0	Φ_2	0	Φ_3	0	Φ_n

Figure 5.1 – Codage d'un individu en utilisant les coefficients de ADL.

de façon aveugle. Il est donc souhaitable qu'un AG ait le plus d'informations possibles pour avoir un meilleur parcours d'évolution.

L'algorithme pour obtenir les coefficients de ADL est montré ci-dessous (voir algorithme 5.1). Nous rappelons que nous appliquons ADL sur les sous-ensembles candidats qui contiennent s gènes avec $s \leq n$. La ligne 4 traite le cas où une matrice S_B n'est pas inversible, lorsque $s \geq n$. Pour cela nous appliquons la pseudo-inverse en appliquant la décomposition en valeurs singulières (voir section 5.1.4) pour obtenir S_{W^+} . La ligne 11 nous donne le vecteur propre ayant la valeur propre la plus grande. Ces valeurs sont alors assignées comme les coefficients discriminants d'un sous-ensemble de gènes et nous permettent de connaître les gènes les plus discriminants ainsi que les moins discriminants de chaque individu.

Algorithme 5.1 : Obtention du Vecteur Propre

```

1 début
2   Soit  $s$  le nombre de gènes sélectionnés dans l'individu candidat
3   Soit  $n$  le nombre d'échantillons
4   Calculer la matrice de dispersion inter-classe  $S_B$ 
5   Calculer la matrice de dispersion intra-classe  $S_W$ 
6   si  $s \leq n$  alors
7     | Calculer  $S_W^{-1} S_B$  pour obtenir  $w_{opt}$ 
8   fin
9   sinon
10    | Remplacer  $S_W^{-1}$  par sa pseudo-inverse  $S_W^+$  en utilisant SVD
11    | Calculer  $S_W^+ S_B$  pour obtenir  $w_{opt}$ 
12  fin
13  retourner  $\phi = |w_{opt}|$ 
14 fin

```

5.2.3 Fonction d'évaluation

Le but de la recherche génétique dans notre approche de ADL-AG est de chercher les "bons" sous-ensembles de gènes ayant la taille minimale et le taux de classification le plus élevé. Normalement ces deux objectifs sont difficiles à concilier, mais pour cela nous proposons une fonction d'aptitude prenant en considération ces deux critères.

Pour évaluer un individu $I = (\tau, \phi)$, la fonction d'évaluation considère l'exactitude d'évaluation de l'individu (f_1) et le nombre de gènes choisis dans l'individu (f_2). Plus

5.2 Une approche intégrée pour la sélection de gènes basée sur ADL

précisément, f_1 est obtenu en évaluant l'exactitude de classification du sous-ensemble de gènes τ en utilisant le classifieur ADL sur l'ensemble de données d'apprentissage. Pour cela nous utilisons la mesure d'exactitude (c.f 4.2.1.2).

Quant à la deuxième partie de la fonction d'évaluation f_2 , elle est calculée par la formule suivante :

$$f_2(I) = \left(1 - \frac{m_\tau}{p}\right) \quad (5.9)$$

où m_τ est le nombre de bits qui ont la valeur "1", c'est-à-dire le nombre de gènes sélectionnés dans cet individu; p est la longueur de l'individu qui correspond au nombre de gènes pré-sélectionnés à partir de la méthode de filtrage.

La fonction d'évaluation f est définie par la fonction d'agrégation suivante :

$$f(I) = \alpha f_1(I) + (1 - \alpha) f_2(I) \\ \text{avec } 0 < \alpha < 1$$

où α est le paramètre qui permet de choisir l'importance relative donnée à f_1 et f_2 . Assignant à α une valeur plus grande que 0.5, on poussera la recherche génétique vers des solutions avec un taux de classification plus haut (probablement aux dépens d'avoir plus de gènes). Inversement, employer de petites valeurs de α aide la recherche vers de petits sous-ensembles de gènes. Donc, une variation de α changera l'objectif de la recherche de l'algorithme génétique.

5.2.4 Croisement ET basé sur ADL

Les opérateurs "génétiques" intégrant des connaissances du problème sont préférables aux opérateurs standards. Dans notre cas, nous employons les coefficients discriminants du classificateur ADL pour concevoir nos opérateurs de croisement et de mutation. Ici, nous expliquons comment notre croisement basé sur ADL fonctionne (dénote par ADL-ET-X dorénavant).

L'opérateur génétique du croisement a été construit afin de produire un enfant ayant des gènes plus discriminants que ceux de ces parents. Pour connaître le coefficient de discrimination de chaque gène des parents, notre approche interagit avec ADL. ADL est une mesure de corrélation utilisée pour ordonner les différents gènes contenus dans chaque parent. En appliquant notre opérateur spécialisé de croisement un enfant est créé. Ensuite cet enfant subit une opération de mutation (voir 5.2.5) avant de rejoindre la prochaine population P' .

ADL-ET-X combine deux parents P^1 et P^2 pour obtenir un enfant E de telle manière que les meilleurs gènes soient conservés dans l'enfant. Pour renforcer l'importance des bons gènes, notre croisement commence par supprimer des parents P^1 et P^2 les gènes qui sont considérés d'après ADL comme peu discriminants. Cela signifie que seule les "bons" gènes seront transmis d'une génération à l'autre. Nous voulons remarquer que

ce processus est l'application d'un simple opérateur logique ET entre ces deux parents sélectionnés.

Cet opérateur a la particularité de transmettre seulement les gènes plus pertinents des parents à l'enfant, autrement dit les gènes communs aux parents. Le croisement ADL-ET-X exécute les étapes suivantes pour générer l'enfant E :

1. Sélectionner deux parents de la population (P).
2. Choisir un nombre aléatoire κ qui détermine le nombre de gènes de τ^1 et τ^2 qui seront ôtés, notés par n_1 et n_2 ;
3. Enlever respectivement de τ^1 et τ^2 , le n_1 et n_2 gènes moins discriminants selon ADL;
4. Fusionner τ^1 et τ^2 modifiés par l'opérateur logique ET pour produire τ ;
5. Appliquer le classificateur ADL à τ , et remplir ϕ par les coefficients discriminants de ADL ;
6. Créer l'enfant $E=(\tau, \phi)$.

Ainsi on peut noter que cet opérateur de croisement à l'aide de ADL a la particularité d'être consacré à la problématique centrale de ce chapitre : la sélection de gènes dont l'objectif est de trouver et de conserver les gènes les plus discriminants de chaque sous-ensemble candidat afin de produire de sous-ensembles de gènes de très bonne qualité.

Exemple 1 Nous prenons deux individus de la population (voir Figure 5.2) pour évaluer l'opérateur de croisement ADL-ET-X. Tout d'abord nous calculons la valeur de κ pour obtenir le nombre de gènes à enlever de chaque individu (Voir Figure 5.3 a)).

	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	Φ_1	Φ_2	Φ_3	Φ_4	Φ_5	Φ_6	Φ_7	Φ_8	f	
P^1	1	0	1	1	0	1	0	1	7.4	0	3.8	3.1	0	1.1	0	1.2	0.98	$n_1=5$
P^2	1	0	1	1	0	1	0	0	3.7	0	0.8	3.0	0	1.8	0	0	0.97	$n_2=4$

Figure 5.2 – Sélection des deux parents de la population initiale.

Nous générons un nombre aléatoire κ entre $[0, 1)$ et nous obtenons le nombre de gènes à ôter de chaque individu comme suit :

$$\text{nombre de gènes} - \lfloor (\kappa \times \text{nombre de gènes}) \rfloor$$

où κ est égal à 0.95 alors nous avons $n_1 = (5 - \lfloor (0.95 \times 5) \rfloor) = 1$ et $n_2 = (4 - \lfloor (0.95 \times 4) \rfloor) = 1$. Nous ôtons 1 gène de chaque individu afin de retenir seulement les gènes les plus pertinents. Après avoir ôté les gènes les moins discriminants τ_6 et τ_3 (voir Figure 5.3 b) de chaque parent, nous procédons à l'étape suivante, appliquer l'opérateur de croisement ADL-ET-X.

5.2 Une approche intégrée pour la sélection de gènes basée sur ADL

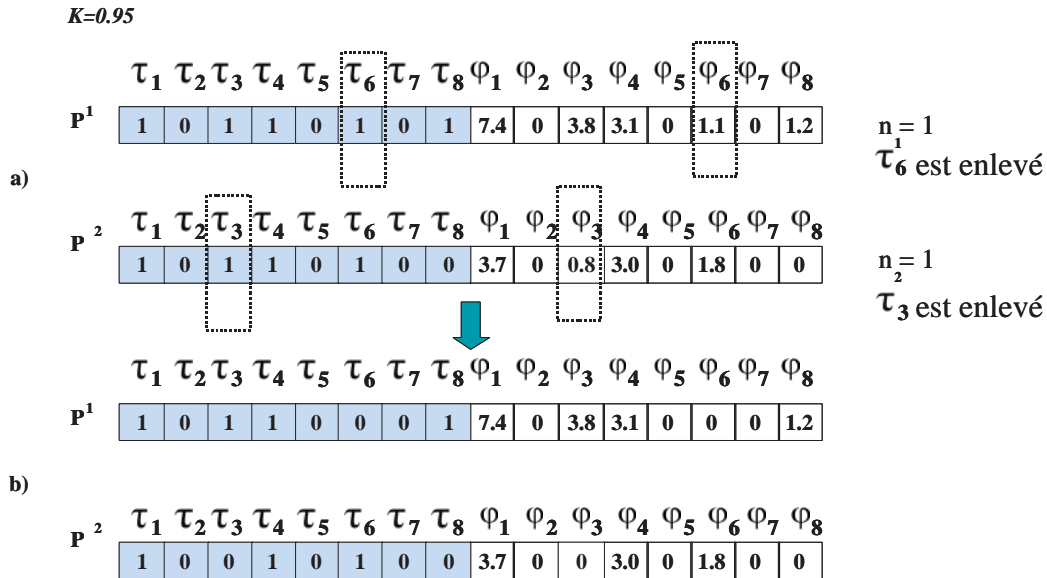


Figure 5.3 – Détermination de nombre de gènes à enlever de chaque parent.

Dans la figure 5.4 nous observons le processus de croisement ADL-ET-X pour produire un enfant de haute qualité en lui transmettant de l'information discriminante. Chaque parent P^1 et P^2 contenant des gènes informatifs vont produire un enfant en lui donnant seulement les gènes plus discriminants. Nous rappelons ce fait car dans le pas précédent nous avons ôté le plus faible de chaque parent.

On remarque que dans le parcours évolutif de l'AG, la taille des enfants est significativement plus petite que celle des parents puisque le croisement repose sur un ET logique qui ne retient que les gènes communs entre 2 parents. Un gène commun signifie un gène qui est important pour l'obtention d'un sous-ensemble de gènes de bonne qualité.

5.2.5 Mutation basée sur ADL

L'opérateur de mutation évalue les gènes de l'enfant pour éliminer le moins discriminant afin de rendre plus précise la sélection de sous-ensembles de gènes.

Étant donné que la fonction d'évaluation que nous avons définie a deux critères, celui du nombre minimal de gènes dans un sous-ensemble et celui de la meilleure performance, notre approche se focalise donc à chercher un bon sous-ensemble candidat pour réaliser la sélection du sous-ensemble de gènes. A la différence de la mutation classique, la mutation que nous proposons repose sur le fait que nous éliminons les gènes les moins discriminants selon le critère ADL. La mutation est employée pour assurer la réduction de dimension (en fait c'est la deuxième partie de la fonction d'évaluation). Chaque application de mutation élimine un seul gène. Pour déterminer quel gène sera retiré du

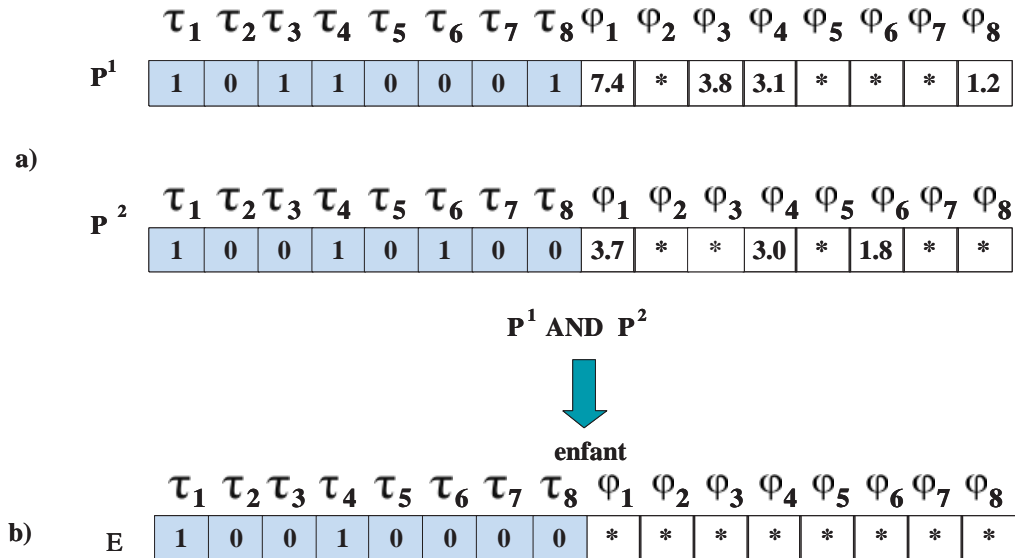


Figure 5.4 – Croisement des parents avec l'opérateur ADL-ET-X.

sous-ensemble, deux critères sont utilisés, menant à deux opérateurs de mutation. Nous décrivons dans la suite chacun de ces opérateurs de mutation.

5.2.5.1 Mutation en utilisant les coefficients discriminants (M1)

Étant donné un individu $I=(\tau, \phi)$, nous identifions le plus petit coefficient discriminant obtenu par ADL dans la partie de l'individu (ϕ) et nous enlevons le gène correspondant, c'est-à-dire celui qui est le moins informatif dans le sous-ensemble (partie (τ)).

5.2.5.2 Mutation par coefficients discriminants et fréquence (M2)

Cet opérateur de mutation a comme but de prendre en compte la fréquence des gènes qui ont été sélectionnés par différents sous-ensembles comme les plus faibles. Nous comptons le nombre de fois qu'un gène est considéré (selon le classificateur de ADL) comme le moins informatif dans un sous-ensemble de gènes. A partir de cette information, nous enlevons le gène qui a le compteur le plus grand, c'est-à-dire que le gène qui a été le plus de fois placé en dernière position va être ôté du sous-ensemble de gènes.

5.3 Protocole expérimental et résultats

Pour évaluer notre approche intégrée nous avons conduit diverses expériences. Ces expériences ont comme objectif de mesurer l'impact de la taille de la population et des

opérateurs génétiques basés sur ADL. Les expériences évaluent :

- 2 types de mutation,
- l’influence de la fonction d’évaluation,
- la taille de la population,
- le nombre de générations,

Nous avons deux types de mutation à évaluer : la mutation par le pouvoir de discrimination fournie par ADL (M1) et la mutation selon la fréquence des gènes les plus faibles trouvés dans différents sous-ensembles (M2).

Étant donné que notre fonction d’aptitude 5.10 dépend fortement du paramètre α , nous avons conduit deux expériences basées sur deux valeurs. Dans première, la valeur est fixée à $\alpha = 0.25$ pour pousser la recherche vers de petits sous-ensembles de gènes (noté Exp1). Dans la deuxième, la valeur est fixée à $\alpha = 0.75$ pour s’intéresser au taux de classification (noté Exp2).

Nous avons lancé diverses expériences qui prennent en compte la taille de population égale de 30, 50 et 100. Cela permet d’évaluer si la taille initiale de la population dans l’AG joue un rôle important. Nous avons fixé un nombre de générations de 500 pour chaque expérience.

Pour conduire nos expériences, nous avons utilisé 7 jeux de données de puces à ADN. Nous utilisons pour tous les jeux une méthode de validation croisée (10-fold). Ce schéma de validation de la précision du classifieur est très important pour obtenir des résultats non biaisés (c.f chapitre 2, section 2.5).

Nous rapportons les résultats de quatre expériences pour chaque jeu de données : GA-M1 est notre approche à l’aide d’un AG avec la mutation M1 et nous sélectionnons le sous-ensemble de gènes selon la fonction d’évaluation ($\alpha = 0.25$) : Exp1 qui se focalise sur le nombre de gènes et Exp2 qui se focalise sur le taux de classification ($\alpha = 0.75$). Alors nous avons pour le premier type de mutation : AG-M1/Exp1 et AG-M2/Exp2. Pour le deuxième type de mutation nous avons les expériences : AG-M2/Exp1 et AG-M2/Exp2.

Dans les deux expérimentations nous avons utilisé les paramètres décrits ci-dessous :

Paramètre	valeur
Taille(s) de la population	30-50-100
Nombre de gènes sélectionnés (p)	100
Nombre d’exécutions	10
Élitisme	10%

Table 5.1 – conditions initiales pour les expérimentations.

5.3.1 Évaluation des performances par un classifieur ADL

Dans cette section nous donnons les résultats où la performance de classifieur est évaluée par un classifieur ADL. Le tableau 5.2 fournit les résultats donnés par notre approche en utilisant différentes tailles de population pour les différents jeux de données de bio-puces. Nous mettons l’accent sur les résultats mettant entre parenthèses la priorité de

chaque expérience conduite. Dans le tableau la performance est montrée à gauche et le nombre de gènes à droite.

Modèle	LEU	COL	PMN	PRT	CNS	OVA	DLB
taille population = 30							
AG-M1/Exp1	93.0(1)	82.2(1)	95.9(1)	92.1(1)	73.3(1)	84.1(1)	80.8(1)
AG-M2/Exp1	94.4(1)	82.2(1)	95.9(1)	91.1(1)	76.6(1)	84.5(1)	80.8(1)
AG-M1/Exp2	(100)4	(98.3)10	(99.3)8	(97.0)10	(98.3)12	(100)28	(100)6
AG-M2/Exp2	(100)5	(98.3)19	(99.3)7	(98.0)35	(98.0)35	(100)25	(100)5
taille population = 50							
AG-M1/Exp1	94.4(1)	64.5(1)	95.3(1)	92.1(1)	73.3(1)	79.8(1)	91.4(1)
AG-M2/Exp1	94.4(1)	64.5(1)	93.2(1)	92.1(1)	73.3(1)	80.6(1)	87.2(1)
AG-M1/Exp2	(100)13	(91.9)14	(99.3)9	(94.1)18	(88.3)44	(98.8)35	(100)6
AG-M2/Exp2	(100)28	(98.3)10	(99.3)10	(95.0)38	(88.3)44	(100)34	(100)6
taille population = 100							
AG-M1/Exp1	83.3(2)	83.8(2)	93.9(2)	75.4(1)	60.0(1)	88.5(1)	70.2(2)
AG-M2/Exp1	94.4(1)	83.8(2)	95.3(1)	88.2(1)	66.6(1)	87.7(1)	72.3(1)
AG-M1/Exp2	(100)3	(91.9)4	(99.3)10	(96.0)53	(86.6)34	(100)18	(100)4
AG-M2/Exp2	(100)5	(98.3)14	(99.3)7	(96.0)8	(100)24	(98.8)17	(100)3

Table 5.2 – Résultats de notre approche intégrée avec différentes tailles de population et comme classifieur ADL.

Nous notons que pour la Leucémie, AG-M1/Exp1 et AG-M2/Exp sont très performantes avec un sous-ensemble de gènes de 4 et 5 gènes respectivement avec une taille de population de 30 et 100. En revanche en utilisant une taille de population de 100, les sous-ensembles sont meilleurs car ils ont moins de gènes avec un taux de classification de 100%.

Nos meilleures performances pour le jeu de Colon sont obtenus avec une taille de population égalée à 30 avec 98.3% en utilisant AG-M1/Exp2 (17 gènes) et AG-M1/Exp2 (10 gènes). Si l'on utilise une taille de 50 on obtient la même performance pour AG-M2/Exp2 sauf que le nombre de gènes est différent de 10.

En ce qui concerne le jeu de Poumon, toutes nos expériences nous donnent de bons résultats. Si l'on veut utiliser un sous-ensemble de petite taille, on peut utiliser celui qui a un seul gène avec une performance de 96.1, c'est-à-dire AG-M1/Exp1 et une taille de population de 50. En revanche si le biologiste a besoin de plus de gènes on peut lui offrir les sous-ensembles de AG-M1/Exp1 (8 gènes) et AG-M2/Exp1 (7 gènes) obtenus avec une population égale à 30. On peut lui fournir les autres sous-ensembles de gènes obtenus des autres expériences car aucun ne dépasse 10 gènes. Il est donc possible pour un expert biologiste d'analyser ces petits sous-ensembles pour mieux comprendre le rôle des gènes retenus.

Pour le jeu de données de Prostate le sous-ensemble ayant la plus haute performance est celui obtenu par AG-M2/Exp2 98.0% avec 35 gènes et une population initiale de 30. Nous constatons que le jeu de données de Prostate nécessite plus de gènes pour obtenir

de bonnes performances.

En utilisant le jeu de données CNS on arrive à obtenir un taux de classification parfait (100%) avec AG-M2/Exp2 (taille population égale à 100 avec un sous-ensemble de 24 gènes). Nous pensons que c'est un bon résultat malgré le nombre de gènes obtenus. Nous constatons que ce jeu de données est l'un des plus difficiles à discriminer comme celui du cancer du Colon. Nous analyserons nos résultats avec ceux fournis dans la littérature dans la section de comparaisons des résultats 5.4.

Pour le jeu de données Ovarien les meilleures performances ont été obtenues avec une taille de population égale à 30 pour les modèles AG-M1/Exp2 (28 gènes) et AG-M2/Exp2 (25 gènes). Une autre performance parfaite (100%) est fournie par le modèle AG-M1/Exp2 avec moins de 18 gènes et une population initiale de 100. Nous notons qu'avec un seul gène nous obtenons des résultats médiocres, mais il suffit simplement d'augmenter le nombre de gènes pour arriver à obtenir une combinaison de gènes qui donne une très bonne performance. Nous rappelons que dans ce jeu le nombre d'échantillons est plus grand (253) dont 150 échantillons servent comme jeu d'apprentissage et 103 échantillons comme jeu de test.

Finalement pour le jeu de données DLBCL nous obtenons de très hautes performances (100%) avec nos quatre modèles dans chacune des différentes expériences en utilisant différentes tailles de population. Nous notons aussi que le nombre de gènes de tous ces sous-ensembles ne dépasse pas 6.

Dans ce qui suit on va évaluer la performance de classification d'un classifieur SVM sur les mêmes jeux de données et les mêmes conditions pour l'AG, c'est-à-dire utiliser différentes tailles de population initiale.

5.3.2 Évaluation des performances SVM

Dans le cadre d'évaluation de la performance de notre méthode d'approche intégrée, nous voulons changer de classifieur pour analyser si cela permet d'améliorer l'apprentissage ou non. On veut tester si l'ensemble trouvé par ADL s'adapte pour un autre classifieur linéaire tel que SVM ce qui donnerait plus de poids aux résultats obtenus. Pour ces expériences nous reprenons le tableau 5.1 et nous utilisons le noyau linéaire du classifieur SVM avec $C = 100$.

Pour le jeu de données de la Leucémie nous obtenons dans les trois expérimentations (tailles initiales de population) une performance qui oscille entre 95.8 et 97.2 lorsqu'on obtient un sous-ensemble de taille minimale (2). En revanche lorsque l'on augmente le nombre de gènes on peut arriver à des classifications parfaites. Par exemple, dans la troisième expérimentation, nous obtenons avec seulement 5 gènes un taux de classification de 100%. Nous remarquons que ce résultat est produit par l'utilisation de l'opérateur de mutation $M2$ celui qui prend en compte la fréquence des gènes les moins discriminants dans un sous-ensemble donné pour AG-SVM. Dans la fin de ce chapitre, nous présentons les meilleurs résultats obtenus, ainsi que les interprétations biologiques trouvées à plusieurs reprises dans la littérature.

En ce qui concerne le jeu de données du Colon, nous trouvons une bonne performance (90.3%) pour AG-M1/ et AG-M2 dans la première expérimentation en utilisant un sous-

Modèle	LEU	COL	PMN	PRT	CNS	OVA	DLB
taille = 30							
AG-M1/Exp1	97.2(2)	90.3(2)	97.3(2)	92.1(2)	78.3(2)	81.0(2)	91.4(2)
AG-M2/Exp1	97.2(2)	90.3(2)	97.9(2)	93.1(2)	76.6(2)	81.4(2)	91.4(2)
AG-M1/Exp2	(100) 14	(98.3)17	(99.3)11	(98.0)38	(96.6)20	(99.2)27	(100) 13
AG-M2/Exp2	(100) 32	(98.3)12	(99.3)9	(98.0)15	(100) 20	(99.2)18	(100) 6
taille = 50							
AG-M1/Exp1	95.8(2)	85.4(2)	97.9(2)	92.1(2)	78.3(2)	79.0(2)	93.6(2)
AG-M2/Exp1	97.2(2)	85.4(2)	97.9(2)	92.1(2)	78.3(2)	81.4(2)	85.1(2)
AG-M1/Exp2	(100) 15	(100) 13	(99.3)12	(98.0)19	(96.6)7	(99.2)31	(100) 7
AG-M2/Exp2	(100) 16	(98.3)15	(99.3)10	(97.0)15	(98.3)17	(99.2)17	(100) 10
taille = 100							
AG-M1/Exp1	97.2(2)	90.3(2)	97.7(2)	94.1(2)	73.3(2)	96.0(2)	91.4(2)
AG-M2/Exp1	97.2(2)	90.3(2)	98.3(2)	94.1(2)	73.3(2)	96.4(2)	93.6(2)
AG-M1/Exp2	(98.6)5	(91.9)3	(97.7)2	(94.8)6	(81.6)8	(98.4)4	(100) 8
AG-M2/Exp2	(100) 5	(93.5)9	(98.3)2	(95.5)18	(86.6)7	(98.8)19	(100) 4

Table 5.3 – Résultats de notre approche intégrée avec différents tailles de population et comme classifieur SVM.

ensemble de taille égale à deux. Le premier résultat est obtenu avec la combinaison des gènes (377) et (173). Dans la deuxième expérimentation nous avons une diminution par rapport à la première expérimentation. En revanche dans la troisième expérimentation nous obtenons la même performance que la première. Si on compare les taux de classification le plus élevés de nos trois expériences nous obtenons 100% avec 13 gènes dans la deuxième expérience.

Si nous employons le jeu du Poumon nous constatons que nos modèles AG-M1/Exp1 et AG-M2/Exp1 offrent deux résultats différents avec une taille minimale de deux gènes dans un sous-ensemble. Par contre, avec la deuxième expérience nous trouvons la même performance que la première avec les mêmes gènes. En revanche, il existe une petite amélioration dans la troisième expérimentation (98.3%) avec deux gènes. Notre meilleure performance avec plus de gènes est fournie par AG-M1/Exp2 dans la deuxième expérience (99.3%) avec 11 gènes.

En utilisant le jeu de la Prostate nous pouvons observer qu'avec deux gènes nos modèles offrent des bons résultats. Le meilleur est obtenu dans la troisième expérimentation 94.1% avec deux gènes en utilisant les modèles AG-M1/Exp1 et AG-M2/Exp1. Pour avoir une meilleure performance, il faut sélectionner plus de gènes et donc le taux le plus haut est trouvé avec AG-M2/Exp2 (98.0%) et un sous-ensemble de 15 gènes

En ce qui concerne le jeu de données de CNS, la première et la deuxième expérience offrent des résultats similaires en utilisant deux gènes. C'est le cas de AG-M1/Exp1 avec une taille de 30 individus et AG-M1/Exp1 et AG-M2/Exp1 avec une taille de 50 individus. En revanche, si l'on augmente le nombre d'individus, on obtient une légère dégradation par rapport à la première et la deuxième expérience. Il est intéressant de noter

5.4 Comparaison de nos résultats avec d'autres travaux

que nous obtenons une parfaite classification en utilisant 20 gènes avec le modèle AG-M2/Exp2.

En appliquant nos modèles sur le jeu de données du cancer de l'ovarie nous avons un taux de classification de 96.0% pour AG-M1 et de 96.4% pour AG-M2 avec deux gènes dans la troisième expérience. La première et la deuxième expérience sont moins performantes. Nous notons que si l'on utilise plus de gènes on obtient une meilleure performance comme c'est le cas de AG-M2/Exp2 (99.2%) avec 17 gènes avec une taille de population de 50. En revanche AG-M1/Exp2 avec 4 gènes nous obtenons une performance inférieure (98.4%).

Le cancer de DLBCL est le dernier jeu de données à évaluer par notre approche. Nous constatons que les performances des trois expériences sont bonnes lorsque l'on utilise deux gènes, ces résultats s'améliorent si l'on augmente le nombre de gènes. Nous obtenons dans nos 2 modèles AG-M1/Exp2 et AG-M2/Exp2 un taux de classification de 100%. Etant donné que nous sommes intéressés par les sous-ensembles de gènes de petite taille, AG-M2/Exp2 est considéré comme le plus approprié pour cette tâche car il utilise seulement 4 gènes.

Nous voulons mettre en évidence qu'avec une autre valeur de C compris entre $[1, 90]$, les résultats obtenus font apparaître une diminution significative des performances sur les jeux de données de Colon, Prostate, CNS et Ovarien. Nous obtenons que sur la Leucémie et DLBCL nous conservons les mêmes taux de classification sauf que celles sont fournies par des autres sous-ensembles de gènes de taille plus grand que ceux qui ont été obtenus avec $C = 100$.

5.4 Comparaison de nos résultats avec d'autres travaux

Nous proposons de faire une comparaison avec les travaux les plus importants dans le domaine de la sélection et de la classification des données de puces à ADN. Nous comparons nos résultats seulement si le processus d'expérimentation est similaire. On prend les résultats du tableau 5.4 avec la taille population=100 car nous trouvons les meilleures performances de notre approche avec des sous-ensembles de gènes de petite taille. Nous comparons les résultats obtenus avec le classifieur ADL car nous considérons que ce classifieur nous a donné les meilleurs résultats en utilisant des sous-ensembles de petite taille: en effet nous avons un taux de classification de 100% sur 3 de 7 jeux de données de puces à ADN.

Nous notons avec le symbole (-) dans le tableau le fait qu'un jeu de données n'est pas traité dans l'article consulté. Nous remarquons que les résultats du tableau 5.4 pour notre approche (quatre dernières lignes) sont très compétitifs par rapport aux méthodes les plus représentatives des dernières années. Nous avons choisi de comparer les résultats de nos modèles AG-M1-Exp1, AG-M2-Exp1, AG-M1-Exp2 et AG-M2-Exp2, en utilisant différentes tailles de population initiale (30,50 et 100).

Tout d'abord pour le jeu de données de la Leucémie nous avons un taux de classification entre 83.3% à 100%. Le sous-ensemble le plus petit de notre modèle AG-M2-MIN fournit une performance de 83.3% avec 2 gènes. En revanche si nous utilisons un nombre

de gènes égale à 3 que pour AG-M1-Exp2 nous arrivons à trouver une parfaite classification de 100%, et pour AG-M2-Exp2 une parfaite classification (100%) avec 5 gènes. Nous constatons que la meilleure performance pour ce jeu de données est reportée dans [Li *et al.*, 2008a] avec seulement 7 gènes. Nous avons donc la deuxième meilleure performance concernant la Leucémie.

Auteur	LEU	COL	PMN	PRT	CNS	OVA	DLB
[Ye <i>et al.</i> , 2004]	97.5	85.0	–	92.5	–	–	–
[Liu <i>et al.</i> , 2004]	100(30)	91.9(30)	100(30)	97.0(30)	–	99.2(75)	98(30)
[Tan et Gilbert, 2003]	91.1	95.1	93.2	73.5	88.3	–	–
[Ding et Peng, 2005]	100	93.5	97.2	–	–	–	–
[Hu <i>et al.</i> , 2006]	94.1	83.8	–	–	–	–	–
[Cho et Won, 2007]	95.9 (25)	87.7(25)	–	–	–	–	93.0(25)
[Yang <i>et al.</i> , 2006]	73.2	84.8	–	86.88	–	–	–
[Peng <i>et al.</i> , 2006]	98.6 (5)	87.0(4)	100(3)	–	–	–	–
[Wang <i>et al.</i> , 2006]	95.8 (20)	100(20)	–	–	–	–	95.6(20)
[Huerta <i>et al.</i> , 2006]	100	91.4	–	–	–	–	–
[Pang <i>et al.</i> , 2007]	94.1(35)	83.8(23)	91.2(34)	–	65.0(46)	98.8(26)	–
[Li <i>et al.</i> , 2007]	97.1(20)	83.5(20)	–	91.7(20)	68.5(20)	99.9(20)	93.0(20)
[Zhang <i>et al.</i> , 2007]	100(30)	90.3(30)	100(30)	95.2(30)	80(30)	–	92.2(30)
[Yue <i>et al.</i> , 2007]	83.8(100)	85.4(100)	–	–	–	–	–
[Hernandez <i>et al.</i> , 2007]	91.5(3)	84.6(7)	–	–	–	–	–
[Wang <i>et al.</i> , 2007]	100(375)	93.5(35)	–	–	–	–	–
[Li <i>et al.</i> , 2008a]	100(7)	93.6(15)	–	–	–	–	–
AG-M1/Exp1	83.3(2)	83.8(2)	93.9(2)	75.4(1)	60.0(1)	88.5(1)	70.2(2)
AG-M2/Exp1	94.4(1)	83.8(2)	95.3(1)	88.2(1)	66.6(1)	87.7(1)	72.3(1)
AG-M1/Exp2	100(3)	91.9(4)	99.3(10)	96.0(53)	86.6(34)	100(18)	100(4)
AG-M2/Exp2	100(5)	98.3(14)	99.3(7)	96.0(8)	100(24)	98.8(17)	100(3)

Table 5.4 – Résultats de ADL basé sur AG (dernières lignes) comparé avec les travaux les plus importants de la classification de données de puces à ADN. Taux de classification à gauche et entre parenthèses le nombre de gènes s'il est disponible à droite.

Dans les cas du jeu de données du cancer du Colon nous obtenons une performance de 83.8% avec 2 gènes en utilisant le modèle AG-M1/Exp1 et AG-M2/Exp1. Par contre nous trouvons pour AG-M1/Exp2 une performance de 98.3% (la deuxième plus haute) avec un sous-ensemble de 14 gènes. Dans [Wang *et al.*, 2006] est listée la deuxième meilleure performance avec 20 gènes. Si nous cherchons qui a le plus petit sous-ensemble de gènes avec une bonne performance nous trouvons le travail de [Peng *et al.*, 2006] avec 4 gènes. Parmi les approches dont le nombre de gènes est petit nous maintenons la meilleure position pour le cancer du Colon. Nous signalons que si l'on utilise SVM comme classifieur nous obtenons une classification parfaite (100%) avec seulement 13 gènes (voir tableau 5.3) et une population initiale égale à 50.

En continuant avec le jeu de données de Poumon, nous trouvons que dans la plupart des travaux les résultats sont très performants par rapport à nos modèles. Dans les travaux de [Liu *et al.*, 2004; Zhang *et al.*, 2007] un taux de classification de 100% est ob-

5.4 Comparaison de nos résultats avec d'autres travaux

tenu avec 30 gènes. Il convient de noter que nous ne pouvons pas savoir si les gènes entre les deux travaux sont différents ou pas, car ils ne les montrent. Dans [Peng *et al.*, 2006] est présenté une performance de 100% avec seulement 3 gènes. Nous obtenons une performance de 95.3% avec un seul gène (AG-M2/Exp2) et 93.9% avec deux gènes (AG/M1/Exp1).

En ce qui concerne le cancer de la Prostate nous avons la deuxième plus haute performance (96.0%) avec 8 gènes. La deuxième meilleure performance a été présentée dans [Liu *et al.*, 2004] avec 97.0% en utilisant 30 gènes. Si nous considérons la taille la plus petite dans le sous-ensemble de gènes nous avons une bonne performance pour AG-M2/Exp1 (88.2%).

Un autre jeu de données difficiles à classer est celui du CNS. Normalement les taux de classification sont bien faibles comme dans les travaux de [Pang *et al.*, 2007] et [Li *et al.*, 2007] qui ne dépassent pas 70%, et pour le premier article nous trouvons que le nombre de gènes est très élevé. Nous constatons que notre modèle AG-M2/Exp2 est capable de fournir une parfaite classification de 100% avec 24 gènes. Le deuxième meilleur taux de classification est obtenu par la méthode de [Tan et Gilbert, 2003] de 88.3%. Nous avons alors la meilleure performance pour CNS avec un petit nombre de gènes. Par contre si l'on utilise SVM comme classifieur on fait mieux, car nous arrivons à une classification parfaite (100%) avec seulement 20 gènes (voir tableau 5.3) avec une population initiale égale à 30.

Dans le jeu de données du cancer de l'ovaire nous trouvons la plus haute performance 100% avec 18 gènes. Nous notons que dans [Li *et al.*, 2008a] un taux de classification de 99.9% est obtenu avec 20 gènes, et dans [Liu *et al.*, 2004] avec 75 gènes. Nous avons donc la meilleure performance avec un nombre minimal de gènes avec AG-M1/Exp2.

Pour le jeu de données de DLBCL nous avons la meilleure performance avec le plus petit nombre de gènes. En utilisant le modèle AG-M1/Exp2, AG-M2/Exp2 nous avons une parfaite classification (100%) avec 4 gènes pour le premier modèle et 3 pour le deuxième. Nous notons que le nombre de gènes rapportés dans les autres travaux sont au moins de 20 gènes pour avoir un bon taux de classification et ils n'arrivent pas à fournir une haute performance. La deuxième meilleure performance pour ce jeu a été présentée dans [Liu *et al.*, 2004] avec 30 gènes.

Les résultats de cette analyse comparative avec d'autres méthodes proposées pour la sélection et la classification des données de puces à ADN nous ont permis de savoir à quel point notre approche est compétitive. Nous montrons que la méthode d'approche intégrée est capable de fournir des sous-ensembles de petite taille avec une haute performance.

Nous avons d'abord utilisée ADL comme classifieur, et puis nous nous sommes demandés si notre approche pouvait être adaptée à un autre classifieur tel que SVM. Les résultats obtenus avec SVM sont presque similaires à ceux rapportés avec ADL avec quelques différences pour certains jeux de données.

ADL-AG permet ainsi d'envisager des méthodes d'approche intégrée en utilisant des autres méthodes de filtrage et de prévoir aussi la possibilité d'introduire peut-être de nouveaux facteurs de discrimination tels que Waveletes (Ondelettes) ou LPC's (Composants de prédiction Linéaire) permettant d'améliorer les résultats présentés dans ce

chapitre.

5.5 Validation biologique des résultats

Nous avons présenté dans la section précédente une manière d'évaluer et comparer la performance de nos modèles avec d'autres méthodes pour la sélection et la classification de puces à ADN. Dans cette section nous proposons d'évaluer la qualité des gènes d'un point de vue biologique. Nous montrons seulement les sous-ensembles de gènes qui fournissent un taux de classification de 100% pour les différents jeux des données de bio-puces et nous indiquons les publications où ces gènes ont été signalés comme pertinents pour le problème de discrimination étudié.

Nous remarquons que pour le jeu de données du CNS (voir tableau 5.8) il n'existe pas beaucoup de travaux concernant ce jeu, et ceux qui l'ont utilisé ne présentent pas la liste des gènes les plus pertinents car ses performances sont très faibles par rapport aux autres jeux de données. Cela ne permet pas de faire une comparaison directe des gènes obtenus.

ID	Gène Code	Description	Références
1834	M23197	Cd33 cd33 antigen (differentiation antigen)	[Golub <i>et al.</i> , 1999; Roth, 2002; Yang et Zhang, 2007; Draminski <i>et al.</i> , 2008]
4407	X64364	Bsg basigin	[Roth, 2002; Draminski <i>et al.</i> , 2008]
157	AJ000480	Gb def=c8fw phosphoprotein	

Table 5.5 – Les 3 Gènes sélectionnés pour la Leucémie par AG-M1/Exp2 qui fournissent un taux de classification de 100% (pop=100, classifieur ADL).

Pos	ID	Gène Code	Description	Références
4	2417	GENE3985X	T-cell protein-tyrosine	[Li <i>et al.</i> , 2001]
40	77	GENE3945X	Ptp-1b=phosphotyrosyl-protein	[Troyanskaya <i>et al.</i> , 2001]
48	379	GENE3640X	Neurotrophic tyrosine kinase, receptor	
67	493	GENE3093X	Abl tyrosine-protein kinase	
81	548	GENE1912X	Unknown ug hs.140628 ests	
87	2721	GENE611X	Thioredoxin	[Draminski <i>et al.</i> , 2008; Aguilar-Ruiz <i>et al.</i> , 2004]
97	1599	GENE301X	Hpak65=ser/thr-protein kinase	

Table 5.6 – Les 7 Gènes sélectionnés par AG-M1/Exp2 qui donnent une performance de 100% pour le jeu de données DLBCL (pop=50, classifieur SVM).

5.6 Synthèse du chapitre

ID	Code	Description	Références
493	R87126	Myosin heavy chain, nonmuscle (gallus gal- lus)	[Furlanello <i>et al.</i> , 2003; Kim et Cho, 2004; Shena <i>et al.</i> , 2007; Li <i>et al.</i> , 2008b]
245	M76378	Human cysteine-rich protein (crp) gene	[Furlanello <i>et al.</i> , 2003; Kim et Cho, 2004; Li <i>et al.</i> , 2008b]
739	X12369	Tropomyosin alpha chain, smooth muscle	[Li <i>et al.</i> , 2008b]
419	R44418	Ebna-2 nuclear protein (epstein-barr virus)	[Furlanello <i>et al.</i> , 2003]
389	H41017	Creatine kinase, ubiquitous mitochondrial	
1033	M69066	Moesin (human);contains ptr5 repetitive ele- ment	
758	T78104	Human proline- arginine-rich end leucine- rich	
203	L12168	Homo sapiens adenylyl cyclase-associated	
173	M19311	Human calmodulin mrna, complete cds.	
1496	R35665	Epidermal growth factor receptor precursor	
792	R88740	Atp synthase coupling factor 6	[Guyon <i>et al.</i> , 2002; Furlanello <i>et al.</i> , 2003]
211	T47424	Insulin receptor substrate-1 (homo sapiens)	[Furlanello <i>et al.</i> , 2003]
1816	R77794	15-Hydroxyprostaglandin dehydrogenase	

Table 5.7 – Les 13 Gènes sélectionnés par AG-M1/Exp2 pour le jeu du Colon qui four-
nissent 100% de classification (pop=50, classifieur SVM).

Pos	ID	Gène Code	Description	Références
3	538	D63880	Kiaa0159 gene	
4	2149	M64347	Fgfr3 fibroblast growth factor receptor 3	
14	4010	V00572	Pgk1 phosphoglycerate kinase 1	
27	3147	U39318	Af-4 mrna	
31	2803	U15008	Snrnp core protein	
39	5302	L11066	Mitochondrial stress-70	
48	4078	X05360	Cdc2 cell division cycle 2	
51	2929	U23143	Mitochondrial serine	
52	3984	U94855	Translation initiation factor 3	
64	4076	X05299	Cenpb centromere protein b (80kd)	
68	6077	X64624	Homeobox/pou domain protein rdc-1	
73	2094	M60858	Nucleolin gene	
76	1473	L32977	Ubiquinol-cytochrome	
78	4564	X76013	Glutaminyl-trna synthetase	
83	3575	U66468	Cell growth regulator cgr11 mrna	
88	1804	M21259	Snrpe small nuclear ribonucleoprotein	
91	1515	L36818	Inpp1l inositol polyphosphate	
100	1421	L25080	Arh12 aplysia ras-related homolog 12	

Table 5.8 – Les 20 Gènes sélectionnés par AG-M2/Exp2 pour le jeu de données CNS en
donnant un taux de classification de 100%(pop=30, classifieur SVM).

5.6 Synthèse du chapitre

Nous avons présenté dans ce chapitre notre méthode d'approche intégrée et avons évalué sa performance dans quatre types d'expérimentations. Étant donné que ADL a des limitations lorsque le nombre de gènes dépasse le nombre d'échantillons, nous avons utilisé la pseudo-inverse à l'aide de la décomposition en valeurs singulières pour résoudre le problème. Pour limiter l'espace de recherche, nous avons d'abord sélectionné

avec un test statistique les cents premiers gènes pertinents. Ensuite nous avons utilisé un schéma génétique pour trouver dans l'espace de recherche les sous-ensembles de gènes avec la plus haute performance et la taille minimale.

ADL fournit pour chaque gène contenu dans un individu une information auxiliaire sur l'importance du gène. Cela permet d'ôter les gènes les moins discriminants afin de mieux guider le parcours de recherche génétique. Nous avons construit des opérateurs génétiques basés sur le critère de discrimination de gènes. Notre approche est performante et compétitive et produit des sous-ensembles de petite taille car si nous utilisons moins de 20 gènes nous pouvons avoir de bons taux de classification sur 6 des 7 jeux de données de puces à ADN.

Dans notre approche intégrée, il faut remarquer que l'évaluation de chaque individu est très coûteuse, surtout dans les premières générations de l'AG, lorsque la population contient beaucoup de gènes. Malgré cet inconvénient du temps de calcul, notre approche réduit dans chaque itération le nombre de gènes et cela permet ainsi d'assurer la pertinence et la non-redondance du sous-ensemble final.

Conclusion générale

La sélection d'attributs dans le domaine des données de puces à ADN est un domaine de recherche qui donne lieu à de nombreuses études et à de nouvelles approches.

Nous avons réalisé de nombreuses expérimentations exhaustives pour évaluer les méthodes proposées. Nous avons conçu notre protocole expérimental pour éviter le biais de sélection. Ce protocole facilite également la comparaison entre les différents travaux.

Les contributions de cette thèse reposent à la fois sur une méthode de pré-traitement originale et sur plusieurs propositions de méthodes évolutives de type enveloppe (wrapper) ou intégrée (embedded) pour traiter la sélection de gènes.

La *première contribution* développée dans cette thèse est l'introduction d'une technique originale basée sur la logique floue pour normaliser les données de puces à ADN et réduire la dimension des données initiales.

Dans un premier temps nous nous sommes intéressés au pré-traitement des données. Nous pensons que la qualité des données de puces à ADN joue un rôle fondamental dans les traitements indépendamment de la technique mise en place. Nous avons présenté une méthode permettant de normaliser les données "bruitées" à l'aide de la logique floue.

Dans un deuxième temps nous avons proposé une méthode originale de réduction des données en utilisant un modèle flou. Pour cela nous construisons des groupes de gènes dont le niveau d'expression est similaire à partir de relations d'équivalence floue. Nous avons utilisé l'information mutuelle pour retenir un seul représentant de chaque groupe et ainsi réduire la taille des données.

Notre algorithme présente certaines limites comme le temps de calcul consommé qui peut s'avérer assez long pour certains jeux de données, à savoir ceux qui dépassent 10,000 gènes.

La *deuxième contribution* de cette thèse est l'introduction d'une double technique d'exploration génétique pour la recherche d'un sous-ensemble optimal. Étant donné qu'un sous-ensemble optimal n'est pas forcément unique, nous cherchons dans une première exploration génétique à l'aide d'une méthode wrapper de bons sous-ensembles pour la classification. Ils nous permettent de mettre en évidence un ensemble de gènes intéressants : ceux que l'on a trouvés fréquemment dans la première exploration. Dans une deuxième phase d'exploration génétique, nous intensifions la recherche dans ce sous-espace restreint, ce qui permet d'obtenir de bonnes performances en classification.

Les études expérimentales réalisées nous ont montré que la double exploration amé-

liore nettement les résultats.

Le *troisième contribution* de cette thèse correspond à l'introduction d'une méthode intégrée. Cette méthode utilise un algorithme génétique pour chercher un sous-ensemble pertinent d'attributs parmi l'espace de solutions. Pour mieux guider ce parcours nous introduisons une méthode de discrimination, à savoir le classifieur LDA, qui nous fournit des informations intéressantes pour les opérateurs de croisement et de mutation. L'information de discrimination donnée sur chaque attribut permet de retenir les attributs les plus pertinents.

Notre approche privilégie ainsi les sous-ensembles ayant une bonne performance avec une petite taille et avec les attributs les plus pertinents. Dans les expérimentations effectuées nous avons constaté que cette approche permet une amélioration des performances par rapport à la deuxième contribution et par rapport aux travaux publiés dans la littérature.

Perspectives de recherche

Nous sommes conscients que les travaux réalisés au cours de cette thèse comportent des limites. Pour cela nous envisageons des techniques complémentaires qui pourraient améliorer nos résultats. Nous présentons dans la suite une discussion ouvrant sur quelques perspectives de recherche.

Pour la méthode proposée dans le chapitre 3, nous pourrions poursuivre cette idée de groupement pour la sélection et classification en utilisant une autre approche comme les sous-ensembles rugueux [Parthalain *et al.*, 2007] ou les sous-ensembles rugueux flous [Jensen et Shen, 2007].

En ce qui concerne le chapitre sur les explorations génétiques, nous pouvons utiliser un critère de filtrage pour réduire la taille initiale des données. Plusieurs critères de filtrage (BW, t-statistique et Wilcoxon test ou bien des autres) existent dans la littérature et ne conduisent pas au même ensemble des gènes. Nous pourrions construire différentes populations en utilisant ces différents critères de filtrage et les combiner par un mécanisme de double exploration comme celui que nous avons décrit dans le chapitre 4.

Dans le chapitre 5 nous avons proposé des opérateurs génétiques spécialisés. D'autres opérateurs pourraient être proposés (OU logique par exemple) pour combiner les informations sur les gènes discriminants. Des techniques comme les ondelettes pourraient également servir pour obtenir des informations sur le pouvoir de discrimination des gènes.

Nous pouvons envisager d'autres méthodes heuristiques pour la sélection d'attributs inspirées de la nature telles que les colonies de fourmis [Dorigo *et al.*, 2002; Jensen, 2006] ou les essaims de particules [Kennedy et Eberhart, 2001].

Index

- α -
 - coupe, 58
 - coupes, 58
- Étapes d'une analyse de biopuces, 7
 - Analyse des images, 8
 - Hybridation, 8
 - Lecture de la puce, 8
 - Marquage, 7
 - Nettoyage, 8
 - Normalisation, 8
 - Préparation des cibles, 7
 - Transformation des données, 8
- Étiquette
 - linguistique, 41
- Évaluation
 - hypothèse de classification, 16
- Évolution naturelle, 69
- ADL, 92
- Algorithme
 - validation croisée, 16
- algorithme
 - génétique, 68
- algorithmes
 - génétiques, 68
- Alpha-
 - coupes, 58
- analyse
 - discriminante linéaire de Fisher, 92
- Attribut
 - faiblement pertinent, 27
 - fortement pertinent, 27
 - Non pertinent, 28
- Classification
 - hiérarchique ascendante, 23
- classification
 - notions, 14
 - objectif, 14
 - supervisée, 14, 15
 - supervisée, définition de, 14
- clustering, 21
- Codage
 - binaire, 69
- coefficients
 - discriminants, 92
- décomposition
 - en valeurs singulières, 94
- Défuzzification, 49
- Degré
 - d'appartenance, 40
 - d'accomplissement, 46
- discrimination
 - des classes, 93
- Ensembles
 - classiques, 40
- exemple, 14
- Fermeture transitive, 56
- Filtrage, 62
- Filtrage par couverture de Markov, 28
- Fonction
 - caractéristique, 41
 - d'appartenance trapézoïde, 42
 - d'appartenance triangulaire, 42
 - d'aptitude, 69
 - d'entropie, 61
 - de fitness, 69
- fonction
 - d'aptitude, 70
- Individu, 69
- Inférence
 - avec plusieurs règles, 47
 - avec une seule règle, 46
- Information
 - redondant, 40
- informatique, 68
- intelligence artificielle, 68
- Jeux de données, 10

- Cerveau, 12
- Colon, 11
- Leucémie, 10
- Lymphome, 12
- Ovarien, 13
- Poumon, 12
- Prostate, 13
- Sein, 13
- Logique
 - floue, 40
- Mécanisme
 - d'évolution, 69
 - d'inférence flou, 46
 - de sélection, 73
- Méthode
 - de filtre, 34
 - Enveloppe, 36
 - Intégrées, 36
- Méthode de sélection
 - du rang, 74
 - par tournoi, 74
 - steady-ready, 74
- méthode de sélection
 - de roulette, 73
- Méthodes de classification non supervisée, 21
 - cartes auto-organisatrices, 23
 - classification hiérarchique, 22
 - clustering, 21
 - ISODATA, 22
 - K-moyennes, 22
- Méthodes de classification supervisée, 16
 - Arbres de décision, 19
 - Bagging, 20
 - Boosting, 20
 - classifieur bayésien naïf, 17
 - Forêts aléatoires, 19
 - k-PPV, 18
 - Réseaux de neurones, 18
 - SVM, 16
- Markov blanket, 28
- Matrice
 - d'expression des gènes, 10
 - inter-classe, 93
 - intra-classe, 93
 - singulière, 94
- Mesure
 - de similarité, 55
 - du cosinus, 55
- Niveaux
 - d'expression, 7
- Opérateur
 - de croisement, 70
 - d'élitisme, 72
 - de composition max-min, 56
 - génétique, 69
- opérateur
 - croisement ET, 99
 - d'évaluation, 70
 - de mutation, 71
 - mutation basée sur ADL, 101
 - mutation par coefficients discriminants, 102
- opérateurs
 - génétiques, 69
 - spécialisés, 97
- Population, 69
 - initiale, 70
- Pré-traitement, 49
 - flou, 49
- Problème
 - de singularité, 94
 - généralisation, 15
- Pseudo-inverse, 94
- Règles
 - floues, 45
- Relation
 - de similitude, 40
 - d'équivalence floue, 55
 - réflexive, 56
 - symétrique, 56
 - transitive, 56
- Représentation
 - floue, 44
- risque
 - empirique, 15
 - réel, 15
- Sélection d'attributs, 30
 - évaluation d'un sous-ensemble, 32
 - évaluation individuelle, 31
 - approches, 34
 - critère d'évaluation, 31
 - critère d'arrêt, 34

Index

- Génération de sous-ensembles, 33
 - initialisation, 31
 - Procédures de recherche, 33
 - schéma général, 31
- sélection d'attributs
 - definition, 30
- Seuillage, 62
- Sous-ensemble
 - classique, 41
 - flou, 40
- Standardisation, 62
- Systeme
 - d'inférence floue, 43

- Technologie
 - de biopuces, 7
 - de puces à ADN, 7
- Terme
 - linguistique, 41

- valeurs
 - propres, 95
 - singulières, 94
- Validation
 - des méthodes, 36
- Variable
 - linguistique, 41
- vecteurs
 - propres, 95

Références bibliographiques

- [Aguilar-Ruiz *et al.*, 2004] cité p. 110
J. Aguilar-Ruiz, F. Azuaje, and J.C. Riquelme Santos. Data mining approaches to diffuse large b-cell lymphoma gene expression data interpretation. In *DaWaK*, pages 279–288, 2004.
- [Alizadeh *et al.*, 2000] cité p. 1, 12, 21
A. Alizadeh, M.B. Eisen, R.E. Davis, C.Ma, I.S. Lossos, A. Rosenwald, J.C. Boldrick, H. Sabet, T. Tran, X. Yu, J.I. Powell, L. Yang, G.E. Marti, T. Moore, J.J. Hudson, L. Lu, D.B. Lewis, R. Tibshirani, G. Sherlock, W.C. Chan, T.C. Greiner, D.D. Weisenburger, J.O. Armitage, R. Warnke, R. Levy, W. Wilson, M.R. Grever, J.C. Byrd, D. Botstein, P.O. Brown, and L.M. Staudt. Distinct types of diffuse large (b)-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, February 2000.
- [Alon *et al.*, 1999] cité p. 11
U. Alon, N. Barkai, and D. Notterman et al. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Nat. Acad. Sci. USA.*, 96:6745–6750., 1999.
- [Ambroise et McLachlan, 2002] cité p. 37
C. Ambroise and G. McLachlan. Selection bias in gene extraction on the basis of microarray gene-expression data. *Proc. Nat. Acad. Sci. USA*, 99(10):6562–6566, 2002.
- [Baldi et Brunak, 2001] cité p. 7
P. Baldi and S. Brunak. *Bioinformatics : The machine learning approach (second edition)*. A Bradford Book. The MIT Press, 2001.
- [Belhumeur *et al.*, 1997] cité p. 93
P. N. Belhumeur, J. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:711–720, 1997.
- [Ben-Israel et Greville, 2003] cité p. 93, 94, 94
A. Ben-Israel and N. Greville. *Generalized Inverses: Theory and Applications*. Springer-Verlag New York Inc, 2nd edition, 2003.
- [Bishop, 2006] cité p. 18
C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.
- [Blum et Langley, 1997] cité p. 1, 26, 26, 30
A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artif. Intell.*, 97:245–271, 1997.
- [Boser *et al.*, 1992] cité p. 16
B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *COLT*, pages 144–152, 1992.

- [Breiman, 1996] cité p. 20
L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.
- [Breimann *et al.*, 1984] cité p. 19, 19, 36
L. Breimann, J. Friedman, R. Olsen, and C. Stone. *Classification and regression trees*. California: Wadworth International, 1984.
- [Burges, 1998] cité p. 17
C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.
- [C-T.Lin et Lee, 1996] cité p. 40, 41, 41, 54, 57
C-T.Lin and C. S. George Lee. *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*. Prentice-Hall, 1996.
- [Cevikalp *et al.*, 2005] cité p. 93
H. Cevikalp, M. Neamtu, M. Wilkes, and A. Barkana. Discriminative common vectors for face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(1):4–13, 2005.
- [Chambers, 2001] cité p. 70, 74, 89
I. L. Chambers. *The practical handbook of genetic algorithms, applications*. Chapman & Hall/CRC, 2001.
- [Cho et Won, 2007] cité p. 88, 108
S.-B. Cho and H.-H. Won. Cancer classification using ensemble of neural networks with multiple significant gene subsets. *Applied Intelligence*, 26(3):243–250, 2007.
- [Chu *et al.*, 2005] cité p. 35
W. Chu, Z. Ghahramani, F. Falciani, and D. Wild. Biomarker discovery in microarray gene expression data with gaussian processes. *Bioinformatics*, 21(16):3385–3393, 2005.
- [Cios *et al.*, 2007] cité p. 1
K. J. Cios, W. Pedrycz, R. W. Swiniarski, and L. A. Kurgan. *Data mining: A knowledge discovery approach*. Springer, 2007.
- [Cornuéjols et Miclet, 2002] cité p. 14, 17
A. Cornuéjols and L. Miclet. *Apprentissage artificielle: Concepts et algorithmes*. Eyrolles, 2002.
- [Cortes et Vapnik, 1995] cité p. 16
C. Cortes and V. Vapnik. Support vector network. *Learning machine*, 20:1–25, 1995.
- [Cotta et Moscato, 2003] cité p. 30
C. Cotta and P. Moscato. The k-feature set problem is w[2]-complete. *Journal of computer and system sciences*, 67:686–690, 2003.
- [Cristianini et Shawe-Taylor, 2000] cité p. 14, 17
N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines*. Cambridge University Press, Cambridge, UK, 2000.
- [Cunningham, 2007] cité p. 1
Pádraig Cunningham. Dimension reduction. Technical report, University College Dublin, 2007.
- [Dai *et al.*, 2006] cité p. 26, 26
J. J. Dai, L. Lieu, and D. Rocke. Dimension reduction for classification with gene expression microarray data. *Statistical Applications in Genetics and Molecular Biology*, 5(1):1–19, 2006.
- [Dash et Liu, 1997] cité p. 26, 26, 31
M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1:131–156, 1997.

Références bibliographiques

- [Dash et Liu, 2006] cité p. 29
M. Dash and H. Liu. Hybrid search of feature subsets. In *PRICA198*. Springer, 2006.
- [Davies et Russell, 1994] cité p. 30
S. Davies and S. Russell. Np-completeness of searches for smallest possible feature sets. In *Proceedings of the 1994 AAAI Fall Symposium on Relevance*, 1994.
- [Deng et al., 2004] cité p. 35
L. Deng, J. Pei, J. Ma, and D.L. Lee. A rank sum test method for informative gene discovery. In W. Kim, R. Kohavi, J. Gehrke, and W. DuMouche, editors, *KDD*, 2004.
- [Ding et Peng, 2005] cité p. 88, 108
C. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *Bioinformatics and Computational Biology*, 3(2):185–206, 2005.
- [Dorigo et al., 2002] cité p. 114
M. Dorigo, L. M. Gambardella, M. Middendorf, and T. Stutzle. Guest editorial: special section on ant colony optimization. *IEEE Transactions on Evolutionary computation*, 6(4):317–319, 2002.
- [Draminski et al., 2008] cité p. 110, 110, 110
M. Draminski, A. Rada-Iglesias, S. Enroth, C. Wadelius, J. Koronacki, and H. J. Komorowski. Monte carlo feature selection for supervised classification (a statistical supplement). *Bioinformatics*, 24(1):100–117, 2008.
- [Dubitzky et al., 2003] cité p. 7, 7, 9
W. Dubitzky, M. Granzow, S. Downes, and D. Berrar. *A practical approach to microarray data analysis*, chapter Introduction to microarray data analysis, pages 1–46. Kluwer Academic Publishers, 2003.
- [Duch, 2006] cité p. 1, 26
W. Duch. *Feature extraction, foundations and applications*, chapter Filter Methods, pages 89–118. Series Studies in Fuzziness and Soft Computing, Physica-Verlag, Springer, 2006.
- [Duda et al., 2000] cité p. 18, 35, 36
R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley, Inc., New York, 2nd edition, 2000.
- [Dudoit et al., 2002] cité p. 7, 8, 9, 11, 34, 35, 60, 62, 62
S. Dudoit, J. Fridlyand, and T. Speed. Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association*, 97:77–87, 2002.
- [Fu et al., 2006] cité p. 21
X. Fu, F. Tan, H. Wang, Y. Zhang, and R. W. Harrison. Feature similarity based redundancy reduction for gene selection. In *DMIN*, pages 357–360, 2006.
- [Fukunaga, 1990] cité p. 26
K. Fukunaga. *Statistical pattern recognition : Second edition*. Morgan Kaufmann, 1990.
- [Furey et al., 2000] cité p. 87, 87
T. Furey, N. Cristianini, N. Duffy, D. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.
- [Furlanello et al., 2003] cité p. 110, 110, 110, 110, 110
C. Furlanello, M. Serafini, S. Merler, and G. Jurman. Entropy-based gene ranking without selection bias for the predictive classification of microarray data. *BMC Bioinformatics*, 4:1–20, 2003.

- [Gacôgne, 1997] cité p. 40
L. Gacôgne. *Eléments de logique floue*. Hermes Sciences Publicat., 1997.
- [Godoy, 2001] cité p. 7
W. W. Godoy. *DNA Arrays: methods and protocols*, chapter Ethical ramifications of genetic analysis using DNA arrays, pages 53–69. Humana Press, 2001.
- [Goldberg, 1989] cité p. 68, 69, 70, 70, 72, 73
D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*, volume 3. Reading, MA: Addison-Wesley., 1989.
- [Golub *et al.*, 1999] cité p. 1, 10, 11, 21, 34, 35, 87, 110
T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
- [Golub et Loan, 1996] cité p. 94, 94, 94
G.H. Golub and C.F.V. Loan. *Matrix Computations (3rd Edition ed.)*. The John Hopkins University Press, Baltimore, Maryland 21218-4319, 1996.
- [Golub et Reinsch, 1970] cité p. 94, 94, 94, 94
G. Golub and C. Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14(5):403–420, April 1970.
- [Gordon *et al.*, 2002] cité p. 12
G.J. Gordon, R.V. Jensen, L.L. Hsiao, S.R. Gullans, J.E. Blumenstock, S. Ramaswamy, W.G. Richards, D.J. Sugarbaker, and R. Bueno. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Research*, 17(62):4963–4967, 2002.
- [Grabmeier et Rudolph, 2002] cité p. 21
J. Grabmeier and A. Rudolph. Techniques of cluster algorithms in data mining. *Data Min. Knowl. Discov.*, 6(4):303–360, 2002.
- [Grody, 2001] cité p. 7
W. W. Grody. *DNA Arrays : methods and protocols*, chapter Ethical ramifications of genetic analysis using DNA arrays, pages 53–69. Humana Press, 2001.
- [Guyon *et al.*, 2002] cité p. 26, 33, 36, 37, 87, 87, 87, 110
I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.
- [Guyon et Elisseeff, 2003] cité p. 26, 26, 29
I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.
- [Haupt et Haupt, 2004] cité p. 70, 70, 70, 89
R.L Haupt and S.E. Haupt. *Practical genetic algorithms*. John Wiley & Sons, Inc. Hoboken, New Jersey, second edition edition, 2004.
- [Hernandez *et al.*, 2007] cité p. 88, 108
J. C. Hernandez, B. Duval, and J-K. Hao. A genetic embedded approach for gene selection and classification. In *EVOBIO*, pages 90–101, 2007.
- [Holland, 1975] cité p. 68, 71
J.H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI: University of Michigan Press, 1975.

Références bibliographiques

- [Hu *et al.*, 2006] cité p. 88, 108
Y. Hu, S. Pang, and I. Havukkala. A novel microarray gene selection method based on consistency. In *HIS 06: Proceedings of the Sixth International Conference on Hybrid Intelligent Systems*, pages 14–17, 2006.
- [Huerta *et al.*, 2006] cité p. 88, 108
E. Bonilla Huerta, B. Duval, and J-K. Hao. A hybrid ga/svm approach for gene selection and classification of microarray data. In *Lecture Notes in Computer Science. Applications of Evolutionary Computing. (4th European Workshop on Evolutionary BIOinformatics).*, pages 34–44. Springer, 2006.
- [Jaeger *et al.*, 2003] cité p. 21, 29, 62
J. Jaeger, R. Sengupta, and W. Ruzzo. Improved gene selection for classification of microarrays. In *Pacific Symposium on Biocomputing*, pages 53–64, 2003.
- [Jain *et al.*, 1999] cité p. 21
A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering : a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.
- [Jain et Zongker, 1997] cité p. 26, 26
A. Jain and D. Zongker. Feature selection : Evaluation, application, and small sample performance. *IEEE Trans Pattern Anal. Mach. Intell.*, 19(2):153–157, 1997.
- [Jang *et al.*, 1997] cité p. 40, 41, 42, 42, 45
J-S. R. Jang, C-T. Sun, and E. Mizutani. *Neuro-fuzzy and soft computing*. Prentice Hall, 1997.
- [Jensen et Shen, 2007] cité p. 114
R. Jensen and Q. Shen. Fuzzy-rough sets assisted attribute selection. *IEEE T. Fuzzy Systems*, 15(1):73–89, 2007.
- [Jensen, 2006] cité p. 114
R. Jensen. Performing feature selection with aco. In *Swarm Intelligence in Data Mining*, pages 45–73. Springer, 2006.
- [Jiang *et al.*, 2008] cité p. 9
N. Jiang, L. J Leach, X. Hu, E. Potokina, T. Jia, A. Druka, R. Waugh, M. J Kearsey, and Z. W Luo. Methods for evaluating gene expression from affymetrix microarray datasets. *BMC Bioinformatics*, 9(284):1–10, 2008.
- [John *et al.*, 1994] cité p. 29
G. John, R. Kohavi, and K. Pflieger. Irrelevant features and the subset selection problem. In *Proc. ICML*, pages 121–129, 1994.
- [Kaufmann, 1973] cité p. 40
A. Kaufmann. *Introduction à la théorie des sous-ensembles flous*. Masson et Cie, 1973.
- [Kennedy et Eberhart, 2001] cité p. 114
J. Kennedy and R. C. Eberhart. *Swarm intelligence*. The Morgan Kaufmann, 2001.
- [Kim et Cho, 2004] cité p. 110, 110
K-J. Kim and S-B. Cho. Prediction of colon cancer using an evolutionary neural network. *Neurocomputing*, 61:361–379, 2004.
- [Knudsen, 2004] cité p. 7
S. Knudsen. *Guide to analysis of DNA microarray data (second edition)*. Wiley-Liss, 2004.
- [Kohavi et John, 1997] cité p. 1, 26, 26, 27, 35
R. Kohavi and G.H. John. Wrappers for feature subset selection. *Artif. Intell.*, 97(1-2):273–324, 1997.

- [Kohonen, 2001] cité p. 23
T. Kohonen. *Self-organizing maps : Third edition*. Springer, 2001.
- [Koller et Sahami, 1996] cité p. 28, 28, 28, 32
D. Koller and M. Sahami. Toward optimal feature selection. *13th International conference on machines learning*, pages 1–15, 1996.
- [Lander, 1999] cité p. 7, 7
E. Lander. Array of hope. *Nature Genetics*, 21:3–4, 1999.
- [Li et al., 2001] cité p. 110
L. Li, C. Weinberg, T. Darden, and L. Pedersen. Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method. *Bioinformatics*, 17(12):1131–1142, 2001.
- [Li et al., 2007] cité p. 88, 89, 108, 109
G-Z. Li, X-Q. Zeng, J.Y. Yang, and M.Q. Yang. Partial least squares based dimension reduction with gene selection for tumor classification. In *Proceedings of IEEE 7th International Symposium on Bioinformatics and Bioengineering*, pages 1439–1444, 2007.
- [Li et al., 2008a] cité p. 88, 107, 108, 109
S. Li, X. Wu, and X. Hu. Gene selection using genetic algorithm and support vectors machines. *Soft Comput.*, 12(7):693–698, 2008.
- [Li et al., 2008b] cité p. 110, 110, 110
S. Li, X. Wu, and M. Tan. Gene selection using hybrid particle swarm optimization and genetic algorithm. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 12:1039–1048, 2008.
- [Lin et Lee, 1996] cité p. 73
C.T. Lin and C.S.G. Lee. *Neural Fuzzy Systems: A neuro-fuzzy synergism to intelligent systems*. Upper Saddle River, NJ 07458: Prentice Hall, 1996.
- [Liu et al., 2004] cité p. 21, 88, 108, 108, 109, 109, 109
B. Liu, Q. Cui, T. Jiang, and S. Ma. A combinational feature selection and ensemble neural network method for classification of gene expression data. *BMC Bioinformatics*, 5(138):1–12, 2004.
- [Liu et Motoda, 2008a] cité p. 26
H. Liu and H. Motoda. *Computational methods of feature selection*. Taylor & Francis group, 2008.
- [Liu et Motoda, 2008b] cité p. 29
H. Liu and H. Motoda, editors. *Feature selection for genomic data analysis*, chapter 17, pages 337–353. Chapman & Hall/CRC, 2008.
- [Liu et Yu, 2005] cité p. 27, 27, 32
H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. on Knowledge and Data Engineering*, 17(4):491–502, 2005.
- [Mao et Tang, 2007] cité p. 29
K. Mao and W. Tang. Correlation-based relevancy and redundancy measures for efficient gene selection. In *Pattern Recognition in Bioinformatics*, volume 4774 of *Lecture Notes in Computer Science*, pages 230–241. Springer, 2007.
- [McLachlan et al., 2008] cité p. 37
G. J. McLachlan, J. Chevelu, and J. Zhu. Correcting for selection bias via cross-validation in the classification of microarray data. In *Beyond Pramatetics in Interdisciplinary research*, pages 364–376. IMS Collections, 2008.

Références bibliographiques

- [Mitchell, 1997] cité p. 17
T.M. Mitchell. *Machine Learning*. Mc-Graw Hill, 1997.
- [Mitchell, 1999] cité p. 70, 70, 70, 72, 73, 89
M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1999.
- [Molina *et al.*, 2002a] cité p. 26, 30
L. C. Molina, L. Belanche, and A. Nebot. Evaluating feature selection algorithms. *CCIA-LNCS*, 2504:216–227, 2002.
- [Molina *et al.*, 2002b] cité p. 26
L. C. Molina, L. Belanche, and A. Nebot. Feature selection algorithms: A survey and experimental evaluation. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM)*, pages 306–313, 2002.
- [Mundra et Rajapaks, 2007] cité p. 29
P.A. Mundra and J.C. Rajapaks. SVM-RFE with relevancy and redundancy criteria for gene selection. In *Pattern Recognition in Bioinformatics*, volume 4774 of *Lecture Notes in Computer Science*, pages 242–252. Springer, 2007.
- [Navin *et al.*, 2006] cité p. 36
T. Navin, O. Chapelle, J. Weston, and A. Elissef. *Feature extraction, foundations and applications*, chapter Embedded Methods, pages 139–167. Series Studies in Fuzziness and Soft Computing, Physica-Verlag, Springer, 2006.
- [Nguyen et Rocke, 2002] cité p. 34, 62
D. Nguyen and D. Rocke. Tumor classification by partial least squares using microarray gene expression data. *Bioinformatics*, 18(1):39–50, 2002.
- [Pang *et al.*, 2007] cité p. 88, 108, 109
S. Pang, I. Havukkala, Y. Hu, and N. Kasabov. Classification consistency analysis for bootstrapping gene selection. *Neural Computing and Applications*, 16:527,539, 2007.
- [Parthalain *et al.*, 2007] cité p. 114
N. M. Parthalain, Q. Shen, and R. Jensen. Distance measure assisted rough set feature selection. In *FUZZ-IEEE*, pages 1–6, 2007.
- [Peng *et al.*, 2003] cité p. 87, 87, 87
S. Peng, Q. Xu, X.B. Ling, X. Peng, W. Du, and L.Chen. Molecular classification of cancer types from microarray data using the combination of genetic algorithms and support vector machines. *FEBS Letters*, 555(2):358–362, 2003.
- [Peng *et al.*, 2006] cité p. 88, 108, 108, 108
Y. Peng, W. Li, and Y. Liu. A hybrid approach for biomarker discovery from microarray gene expression data. *Cancer Informatics*, 2:301–311, 2006.
- [Petricoin *et al.*, 2002] cité p. 13
E. F. Petricoin, A. M. Ardekani, B. A. Hitt, P. J. Levine, V. A. Fusaro, S. M. Steinberg, G. B. Mills, C. Simone, D. A. Fishman, E. C. Kohn, and L. A. Liotta. Use of proteomic patterns in serum to identify ovarian cancer. *Lancet*, 359:572–577, 2002.
- [Pomeroy *et al.*, 2002] cité p. 12
S. L. Pomeroy, P. Tamayo, M. Gaasenbeek, L.M. Sturia, M. Angelo, M.E. McLaughlin, J.Y.H. Kim, L.C. Goumnerova, P.M. Black, C. Lau, J.C. Allen, D. Zagzag, M.M. Olson, T. Curran, C. Wetmore, J.A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D.N. Louis, J.P.E.S. Mesirov, Lander, and T.R. Golub. Prediction of central nervous system embryonal tumour outcome based on gene expression. *Nature*, 415:436–442, 2002.

- [Quinlan, 1993] cité p. 19, 36
 J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc, 1993.
- [Rakotomamonjy, 2003] cité p. 12, 36
 A. Rakotomamonjy. Variable selection using svm-based criteria. *Machine Learning Research*, 3:1357–1370, 2003.
- [Reddy et Deb, 2003] cité p. 87, 87, 87, 87, 87
 A. R. Reddy and K. Deb. Classification of two-class cancer data reliably using evolutionary algorithms. Technical report, Kanpur Genetic Algorithms Laboratory (KanGAL), Indian Institute of Technology Kanpur, 2003.
- [Reeves, 2003] cité p. 75
 C. Reeves. *Genetic Algorithms*, chapter 3, pages 55–82. Handbook of Metaheuristics. Kluwer Academic Publishers, 2003.
- [Ross, 2005] cité p. 40, 41, 45, 49, 54, 55, 55, 57
 T. Ross. *Fuzzy Logic with Engineering Applications (Second Edition)*. Wiley, 2005.
- [Roth, 2002] cité p. 110, 110
 Volker Roth. *The generalized LASSO: a wrapper approach to gene selection for microarray data*. University of Bonn, Dep. Computer Science III, Tech. Report IAI-TR-2002-8., 2002.
- [Ruschhaupt *et al.*, 2004] cité p. 37
 M. Ruschhaupt, W. Huber, A. Poustka, and U. Mansmann. A compendium to ensure computational reproducibility in high dimensional classification tasks. In *PARA*, volume 3, pages 939–948, 2004.
- [Saeys *et al.*, 2007] cité p. 36
 Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.
- [Schapire, 1990] cité p. 20
 R.E. Schapire. The strength of weak learning. *Machine Learning*, 5(2):197–227, 1990.
- [Schlogl *et al.*, 2002] cité p. 61
 A. Schlogl, C. Neuper, and G. Pfurtscheller. Estimating the mutual information of an eeg-based brain-computer-interface. Technical report, Department of Medical Informatics, Institute of Biomedical Engineering, University of Technology, Graz, 2002.
- [Schoemaker et Lin, 2005] cité p. 7
 J. S. Schoemaker and S. M. Lin, editors. *Methods of microarray data analysis IV*. Springer, 2005.
- [Scholkopf et Smola, 2001] cité p. 17
 B. Scholkopf and A. Smola. *Learning with kernels*. MIT Press, 2001.
- [Shena *et al.*, 2007] cité p. 110
 Q. Shena, W.-Min Shia, W. Konga, and B.-X. Yea. A combination of modified particle swarm optimization algorithm and support vector machine for gene selection and tumor classification. *Talanta*, 71:1679–1683, 2007.
- [Shi et Chen, 2005] cité p. 87, 87
 C. Shi and L. Chen. Feature dimension reduction for microarray data analysis using locally linear embedding. In *APBC*, pages 211–217, 2005.
- [Singh *et al.*, 2002a] cité p. 13
 D. Singh, P. Febbo, K. Ross, D. Jackson, J. Manola, C. Ladd, P. Tamayo, A. Renshaw, A. D’Amico, and J. Richie. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1:203–209, 2002.

Références bibliographiques

- [Singh *et al.*, 2002b] cité p. 26
S. Singh, M. Singh, and M. Markou. Feature selection for face recognition based on data partitioning. *ICPR*, 1:680–683, 2002.
- [Soularue et Gidrol, 2002] cité p. 7, 7
P. Soularue and X. Gidrol. Puces à adn. Recherche, 06 2002.
- [Southern, 2001a] cité p. 7, 8, 9
E. M. Southern. *DNA Arrays methods and protocols*, chapter DNA Microarrays, pages 1–15. Humana Press, 2001.
- [Southern, 2001b] cité p. 7
E. M. Southern. *DNA Arrays: Methods and Protocols*, chapter DNA Microarrays, pages 1–15. Humana Press, 2001.
- [Speed, 2000] cité p. 7
T. Speed, editor. *Statistical analyse of gene expression microarray data*. Chapman & Hall/CRS, 2000.
- [Stafford, 2008] cité p. 9
P. Stafford, editor. *Methods in microarray normalization*. CRC Press, 2008.
- [Stekel, 2003] cité p. 7, 7, 9
D. Stekel. *Microarray bioinformatics*. Cambridge University Press, 2003.
- [Tan et Gilbert, 2003] cité p. 88, 108, 109
A. C. Tan and D. Gilbert. Ensemble machine learning on gene expression data for cancer classification. *Applied Bioinformatics*, 2(2):75–83, 2003.
- [Theodoridis et Koutroumbas, 1999] cité p. 21, 22, 22
S. Theodoridis and K. Koutroumbas. *Pattern recognition*. Academic Press, 1999.
- [Tong-Tong, 1995] cité p. 40, 40, 41, 41, 44, 45, 45, 54, 57
J-R. Tong-Tong. *La Logique Floue*. Hermes, 1995.
- [Troyanskaya *et al.*, 2001] cité p. 12, 110
O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. Altman. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17(6):520–525, 2001.
- [Vafaie et Jong, 1993] cité p. 70
H. Vafaie and K. Jong. Robust feature selection algorithms. In *Proc. 5th Intl. Conf. on Tools with Artificial Intelligence*, pages 356–363, 1993.
- [Vapnik, 1998] cité p. 16
V. Vapnik. *Statistical learning theory*. John Wiley, 1998.
- [Veer *et al.*, 2002] cité p. 13
L.J. Van't Veer, H. Dai, M.J. Van de Vijver, Y.D. He, A.A.M. Hart, M. Mao, H.L. Peterse, K. Van der Kooy, M.J. Marton, A.T. Witteveen, G.J. Schreiber, R.M. Kerkhoven, C. Roberts, P.S. Linsley, R. Bernards, and S.H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415:530–536, 2002.
- [Wang *et al.*, 2005] cité p. 87, 87, 87
Y. Wang, F. Makedon, J. C. Ford, and J. D. Pearlman. Hykgene: a hybrid approach for selecting marker genes for phenotype classification using microarray gene expression data. *Bioinformatics*, 21(8):1530–1537, 2005.
- [Wang *et al.*, 2006] cité p. 88, 88, 108, 108
Z. Wang, V. Palade, and Y. Xu. Neuro-fuzzy ensemble approach for microarray cancer gene expression data analysis. In *Proc. Evolving Fuzzy Systems.*, pages 241–246, 2006.

- [Wang *et al.*, 2007] cité p. 88, 108
S. Wang, H. Chen, S. Li, and D. Zhang. Feature extraction from tumor gene expression profiles using dct and dft. In *EPIA Workshops*, pages 485–496, 2007.
- [Weston *et al.*, 2002] cité p. 12, 36
J. Weston, A. Elisseeff, B. Scholkopf, and M. Tipping. The use of zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2002.
- [Wu *et al.*, 2008] cité p. 17, 18
X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z-H Zhou, M. Steinbach, D. J. Hand, and D. Steinberg. Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 1:1–37, 2008.
- [Xiu et Wunsch, 2005] cité p. 21
R. Xiu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.
- [Yang *et al.*, 2002] cité p. 9
Y.H. Yang, S. Dudoit, P. Luu, D. M. Lin, V. Peng, J. Ngai, and T.P. Speed. Normalization for cdna microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res*, 30:1–12, 2002.
- [Yang *et al.*, 2006] cité p. 88, 108
W-H. Yang, D-Q. Dai, and H. Yan. Generalized discriminant analysis for tumor classification with gene expression data. *Machine Learning and Cybernetics.*, 1:4322–4327, 2006.
- [Yang et Honovar, 1998] cité p. 26
J. Yang and V. Honovar. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13:44–49, 1998.
- [Yang et Zhang, 2007] cité p. 110
P. Yang and Z. Zhang. Hybrid methods to select informative gene sets in microarray data classification. In *Australian Conference on Artificial Intelligence*, pages 810–814, 2007.
- [Ye *et al.*, 2004] cité p. 88, 93, 93, 93, 108
J. Ye, T. Li, T. Xiong, and R. Janardan. Using uncorrelated discriminant analysis for tissue classification with gene expression data. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 1(4):181–190, 2004.
- [Ye et Li, 2004] cité p. 93
J. Ye and Q. Li. Lda/qr: an efficient and effective dimension reduction algorithm and its theoretical foundation. *Pattern Recognition*, 37(4):851–854, 2004.
- [Ye, 2005] cité p. 93, 93, 93, 93
J. Ye. Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems. *Journal of Machine Learning Research*, 6:483–502, 2005.
- [Yu et Liu, 2004a] cité p. 26, 28, 28, 29, 29
L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.
- [Yu et Liu, 2004b] cité p. 26, 29
L. Yu and H. Liu. Redundancy based feature selection for microarray data. In *KDD*, pages 737–742, 2004.
- [Yue *et al.*, 2007] cité p. 88, 93, 108
F. Yue, K. Wang, and W. Zuo. Informative gene selection and tumor classification by null space lda for microarray data. In *ESCAPE*, pages 435–446, 2007.

Références bibliographiques

- [Zadeh, 1965] cité p. 40, 40, 40, 41
L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.
- [Zadeh, 1975] cité p. 41
L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning - i. *Inf. Sci*, 8:199–249, 1975.
- [Zhang *et al.*, 2007] cité p. 88, 108, 108
L. Zhang, Z. Li, and H. Chen. An effective gene selection method based on relevance analysis and discernibility matrix. In *PAKDD*, pages 1088–1095, 2007.
- [Zhu et Hastie, 2004] cité p. 36
J. Zhu and T. Hastie. Classification of gene microarrays by penalized logistic regression. *Bio-statistics*, 5(3):427–443, 2004.

Liste des publications personnelles

Revue internationale

1. E. Bonilla-Huerta, B. Duval and J. K. Hao. Fuzzy logic for elimination of redundant information of microarray data. *Genomics Proteomics and Bioinformatics*, Elsevier, Octobre 2008, Vol.6, No. 2.

Conférences internationales avec comité de sélection

1. E. Bonilla-Huerta, B. Duval and J. K. Hao. A hybrid GA/SVM approach for gene selection and classification of Microarray data. *Lecture Notes in Computer Science*, volume 3907 dans *Lecture Notes in Computer Science*, pages 34-44, Budapest, Hungary. April 2006. Springer.
2. E. Bonilla-Huerta, B. Duval and J. K. Hao. Gene selection for Microarray data by a LDA-based genetic algorithm. *Third IAPR International Conference on Pattern Recognition in Bioinformatics (PRIB'08)*, dans *Lecture Notes in Bioinformatics*, Springer-Verlag, 2008, pages 250-261 Melbourne, Australia. Octobre 2008. Springer.

Conférences francophones avec actes de résumés étendus

1. E. Bonilla-Huerta, B. Duval and J. K. Hao. Une méthode de pré-traitement et de pré-sélection par la logique floue pour les données de puces ADN . *Actes du Septième Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'06)*, Lille, France, Février 2006.
2. E. Bonilla-Huerta, B. Duval and J. K. Hao. Algorithme génétique et SVM pour la sélection et la classification de données des puces. *Actes du Septième Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'06)*, Lille, France, Février 2006.

LOGIQUE FLOUE ET ALGORITHMES GÉNÉTIQUES POUR LE PRÉ-TRAITEMENT DE DONNÉES DE BIOPUCES ET LA SÉLECTION DE GÈNES

Résumé

Dans le domaine de la biologie moléculaire, les technologies d'analyse d'expression génique comme les biopuces suscitent un intérêt très grand. Une des applications de ces technologies est le diagnostic et la classification de différents types de tumeurs. Une des particularités des données issues des biopuces est qu'elles sont décrites par un très grand nombre d'attributs (gènes) alors que peu d'échantillons analysés sont disponibles. Cela empêche la compréhension des données et réduit de manière considérable la performance des algorithmes de classification.

Dans cette thèse, nous proposons des méthodes innovantes pour réduire la taille initiale des données et pour sélectionner des ensembles de gènes pertinents pour une classification supervisée. Nous proposons tout d'abord une méthode de pré-traitement des données et de réduction de dimension basée sur la logique floue. Le problème de la sélection d'attributs est ensuite traité par deux types d'approche. Dans la première, nous proposons une méthode enveloppe qui grâce à une double exploration génétique sélectionne un ensemble de gènes pertinents pour un classifieur SVM. Dans la deuxième, nous proposons une méthode intégrée où les informations fournies par un classifieur linéaire (ADL) guident le processus de recherche vers un sous-ensemble de petite taille et performant pour la classification.

Les différentes expérimentations que nous avons menées montrent de très bonnes performances, surtout pour la méthode intégrée.

Mots-clés : sélection d'attributs, sélection de gènes, logique floue, algorithmes génétiques, puces à ADN.

FUZZY LOGIC AND GENETIC ALGORITHMS FOR THE PRE-PROCESSING OF MICROARRAY DATA AND GENE SELECTION

Abstract

In molecular biology, technologies for gene expression analyses, as DNA microarrays, give rise to a lot of research. One possible application of such technologies is diagnosis and recognition of different kinds of tumours. One particularity of microarray data is their great number of attributes (genes) whereas very few samples are available. This dimensionality problem makes the data difficult to understand and reduces the efficiency of classification algorithms.

This thesis proposes new methods to reduce the initial dimension of data and to select relevant gene subsets for classification. First, we propose a method for data pre-processing and dimension reduction based on fuzzy logic. Then the problem of gene selection is treated by two kinds of approaches. In the first one, we propose a wrapper method that selects a relevant gene subset for a SVM classifier by a double genetic exploration. In the second approach, we propose an embedded method where LDA classifier provides information about gene relevancy to guide the genetic algorithm.

The different experimentations that we have performed give very good results, especially for the embedded method.

Keywords: feature selection, gene selection, fuzzy logic, genetic algorithm, microarray data.