Université Lille 1
École Doctorale Sciences Pour l'Ingénieur Lille 1 Nord-de-France
Laboratoire d'Informatique Fondamentale de Lille (UMR CNRS 8022)
Centre de Recherche Inria Lille Nord Europe

THÈSE

pour obtenir le grade de
Docteur de l'Université de Lille I
Discipline : Informatique
Numéro d'ordre : 41262 | Année : 2013

Classification sur données médicales à l'aide de méthodes d'optimisation et de datamining, appliquée au pré-screening dans les essais cliniques

par

Julie JACQUES

Date de soutenance : 2 décembre 2013

Jury

Directeurs :	Clarisse DHAENENS	Professeur des Universités, Université Lille I
	Lætitia JOURDAN	Professeur des Universités, Université Lille I
Rapporteurs :	Jean-Charles BILLAUT	Professeur des Universités, Université de Tours
	Nadia BRAUNER	Professeur des Universités, Université Grenoble I
Examinateurs :	Stéphane BONNEVAY	Maître de conférences, HDR, Université Lyon I
	Denis BOUYSSOU	Directeur de recherche, Université Paris Dauphine
	Sophie TISON	Professeur des Universités, Université Lille I
Invité :	David DELERUE	Gérant, Société Alicante

Abstract Medical data suffer from uncertainty and a lack of standardization, making them hard to use in medical software, especially for patient screening in clinical trials. In this PhD work, we propose to deal with these problems using supervised classification methods. We will focus on 3 properties of these data : imbalance, uncertainty and volumetry. We propose the MOCA-I algorithm to cope with this partial classification combinatorial problem, that uses a multi-objective local search algorithm. After having confirmed the benefits of multiobjectivization in this context, we calibrate MOCA-I and compare it to the best algorithms of the literature, on both real data sets and imbalanced data sets from literature. MOCA-I generates rule sets that are statistically better than models obtained by the best algorithms of the literature. Moreover, the models generated by MOCA-I are between 2 to 6 times shorter. Regarding balanced data, we propose the MOCA algorithm, statistically equivalent to best algorithms of literature. Then, we analyze both theoretically and experimentally the behaviors of MOCA and MOCA-I depending on imbalance. In order to help the decision maker to choose a solution and reduce over-fitting, we propose and evaluate different methods to handle all the Pareto solutions generated by MOCA-I. Finally, we show how this work can be integrated into a software application.

Keywords Imbalanced data, partial classification, combinatorial optimization, multi-objective optimization, machine learning, datamining.

Résumé Les données médicales souffrent de problèmes d'uniformisation ou d'incertitude, ce qui les rend difficilement utilisables directement par des logiciels médicaux, en particulier dans le cas du recrutement pour les essais cliniques. Dans cette thèse, nous proposons une approche permettant de pallier la mauvaise qualité de ces données à l'aide de méthodes de classification supervisée. Nous nous intéresserons en particulier à 3 caractéristiques de ces données : asymétrie, incertitude et volumétrie. Nous proposons l'algorithme MOCA-I qui aborde ce problème combinatoire de classification partielle sur données asymétriques sous la forme d'un problème de recherche locale multi-objectif. Après avoir confirmé les apports de la modélisation multi-objectif dans ce contexte, nous calibrons MOCA-I et le comparons aux meilleurs algorithmes de classification de la littérature, sur des jeux de données réels et asymétriques de la littérature. Les ensembles de règles obtenus par MOCA-I sont statistiquement plus performants que ceux de la littérature, et 2 à 6 fois plus compacts. Pour les données ne présentant pas d'asymétrie, nous proposons l'algorithme MOCA, statistiquement équivalent à ceux de la littérature. Nous analysons ensuite l'impact de l'asymétrie sur le comportement de MOCA et MOCA-I, de manière théorique et expérimentale. Puis, nous proposons et évaluons différentes méthodes pour traiter les nombreuses solutions Pareto générées par MOCA-I, afin d'assister l'utilisateur dans le choix de la solution finale et réduire le phénomène de sur-apprentissage. Enfin, nous montrons comment le travail réalisé peut s'intégrer dans une solution logicielle.

Mots clefs Classification sur données asymétriques, classification partielle, optimisation combinatoire, optimisation multi-objectif, apprentissage, fouille de données.

Remerciements

Je tiens tout d'abord à remercier Clarisse et Laetitia, mes directrices de thèse, avec qui j'ai eu la chance et le plaisir de travailler pendant ces 3 ans. Je les remercie toutes les deux pour leurs nombreux conseils, relectures, encouragements et leur disponibilité.

Je souhaite remercier les membres du jury pour l'intérêt qu'ils ont porté à mon travail et les remarques pertinentes qu'ils y ont apporté : Jean-Charles Billaut et Nadia Brauner, rapporteurs, Stéphane Bonnevay et Denis Bouyssou, examinateurs, et Sophie Tison, présidente du jury.

Je tiens à remercier les Professeurs Régis Bordet et Régis Beuscart d'avoir partagé leur expérience sur les essais cliniques pour la conception d'Opcyclin, ainsi que Virginie Deprez et Anne-Marie Bordet d'avoir partagé leur expérience d'attaché de recherche clinique.

Je remercie également mes collègues d'Alicante David, Julien, Muriel, Charles et Benjamin avec qui j'ai travaillé sur Opcyclin, ainsi que Cédric, Jacques, Fabienne, Samuel et Eve que j'ai cotoyés pendant ces 3 ans à Alicante.

Je tiens aussi à remercier les différents membres permanents de l'équipe Dolphin pour leur accueil : El-Ghazali Talbi, Nouredine Melab, Luce Brotcorne, Bilel Derbel, Dimo, François, Sébastien, Arnaud et Marie. Sans oublier les non-permanents : Thé Van, Sezin, Ines, Nadia, Moustapha, Bayrem, Tuan, Aline, Ekaterina, Sophie, Martin, Mathieu et en particulier Yacine et Julie avec qui j'ai pu partager la thèse au quotidien avec ses hauts et ses bas, et dont les nombreuses sessions "thé" ont boosté ma productivité.

Je remercie mon compagnon, André, de m'avoir soutenue et encouragée pendant ces 3 ans. Je tiens à remercier Floriane, ma petite soeur, pour son soutien, et ses conseils en anglais. Je remercie également mes parents, en particulier mon père pour m'avoir donné le goût des sciences et de l'innovation, et toujours contribué à développer ma curiosité scientifique, même s'il n'est plus là aujourd'hui.

Je remercie également mes amis pour leur soutien, et pour m'avoir rechargé les batteries quand j'en avais besoin : les amis d'Outre-Quiévrain : Florence, Michael et Astrid mais aussi les français : Yoann, Sandra, Virginie et Geoffrey.

Enfin, ce travail n'aurait pu avoir lieu sans le financement de la société Alicante. J'ai beaucoup apprécié de contribuer à certains des projets innovants d'Alicante, pour lesquels j'ai bénéficié de beaucoup de liberté et de confiance quant à la manière de les mener à bien.

Table des matières

TABLE DES MATIÈRES

Table des figures

TABLE DES FIGURES

Liste des tableaux

Introduction

D'après l'enquête du LEEM (Les Entreprises du Médicament) « La Place de la France dans la recherche clinique internationale » datant de 2012 [1], en matière d'essais cliniques la France dispose de taux de recrutement moyens plus faibles que les autres pays Européens. En moyenne, 4,8 patients seront inclus par centre pour un essai clinique donné, quand les autres pays ont une moyenne de 6 patients par centre. Cela prive de nombreux patients de la possibilité d'accéder à des techniques de soin innovantes, et d'un meilleur suivi par rapport à la prise en charge classique. Cela conduit même parfois à stopper des essais cliniques, par manque de patients, ce qui représente une perte financière pour le promoteur de l'essai clinique. Tout cela souligne le besoin d'améliorer le recrutement. Cela peut passer par la mise en place d'outils automatisés d'aide au recrutement, qui pourraient par exemple utiliser les données présentes à l'hôpital.

Ces données posent néanmoins quelques problèmes : elles sont souvent peu uniformisées, plusieurs codes ou désignations peuvent exister pour une même pathologie. Par exemple, le diabète de type I est également appelé diabète insulino-dépendant, ou encore diabète mellitus. Dans le codage médical des médicaments ATC (Anatomical Therapeutic Chemical), certaines molécules correspondent à plusieurs codes, par exemple l'aspirine dispose de 4 codes différents. Certaines données peuvent également être manquantes. Les données PMSI (Programme de Médicalisation des Systèmes d'Information) par exemple, majoritairement disponibles dans les hôpitaux, renseignent uniquement les diagnostics et actes qui ont impacté directement la facturation. On retrouve donc des projets, à l'instar du projet ASTEC, qui utilisent un système expert pour les essais cliniques en cancérologie, afin de pallier les différents problèmes de ces données et de les rendre ainsi exploitables pour le recrutement de patients. La réalisation d'un système expert consomme énormément de ressources humaines. À titre illustratif, sur le projet ANR AKENATON pour la reclassification d'alertes de pacemaker, la réalisation du système expert a pris plusieurs mois. D'autant plus que des experts différents devront être mobilisés lorsque les essais cliniques concernent d'autres pathologies, ce qui multiplie encore le temps de réalisation. L'enjeu de cette thèse est donc de proposer une méthode automatique de génération du système expert, à l'aide de méthodes de classification supervisée.

Cette thèse est financée par la société *Alicante*, PME spécialisée dans les progiciels pour les hôpitaux. Cette société intervient sur de nombreux domaines : identito-vigilance, décisionnel hospitalier, bibliométrie ou encore recherche clinique. Elle a également participé à de nombreux

1. http ://www.leem.org/attractivite-de-france-pour-recherche-clinique-internationale-resultats-de-lenquete-2012

projets de recherche, les derniers en date étant les projets AKENATON (ANR TECSAN 2007-2011) et FAROS (RNTL 2005-2008). Cette thèse est réalisée en collaboration avec l'équipe Dolphin (Discrete multiobjective Optimization for Large-scale Problems with Hybrid dIstributed techNiques) de Inria Lille Nord Europe et du laboratoire LIFL de l'Université Lille 1. L'équipe Dolphin est forte d'une expérience dans les méthodes d'optimisation et méta-heuristiques ainsi que dans les méthodes de fouille de données. L'équipe Dolphin a notamment réalisé des projets sur des applications de fouille de données en génomique, où l'on retrouve de larges volumes de données. Dans cette thèse, nous élaborons une méthode pour traiter les données médicales à l'aide de méthodes de classification supervisée (méthode de fouille de données) et d'optimisation. Cette méthode est appelée MOCA-I (Multi-Objective Classification Algorithm for Imbalanced data). Elle sera intégrée dans la suite logicielle Opcyclin, dédiée au recrutement dans les essais cliniques, qui est conçue par la société *Alicante* en parallèle de ces travaux de thèse.

La classification supervisée sur les données médicales présente plusieurs challenges. D'une part, ces données présentent souvent une asymétrie au niveau de la pathologie à prédire : dans l'ensemble des patients d'un hôpital, les patients atteints d'une pathologie donnée représentent de $<1\%$ à 10% des patients. Cette répartition pose des problèmes aux algorithmes usuels de classification supervisée. Un autre challenge réside dans l'incertitude amenée par certaines sources de données médicales : l'absence d'une donnée dans le dossier du patient peut avoir plusieurs significations. Il y a tout d'abord le cas idéal, où l'absence de la donnée dans le dossier correspond à la situation du patient : il n'a pas la pathologie x ou il n'a pas eu l'acte y ou encore on ne lui a pas prescrit le médicament z. Dans un autre cas, le patient peut avoir la pathologie sans qu'elle ait été diagnostiquée, ou avoir effectué l'acte ou la prescription de médicament dans un autre centre de soins : la donnée ne sera donc pas renseignée dans son dossier. Enfin, il peut s'agir également d'une absence par omission (par exemple dans les données PMSI, parce que le diagnostic n'a pas eu d'impact sur la facturation). L'incertitude amène à s'orienter vers des techniques de classification partielle, qui sont plus adaptées que la classification binaire. Le dernier challenge réside dans la volumétrie des données étudiées. Elles présentent une grande quantité d'informations disponibles, ce qui là aussi pose des problèmes aux algorithmes usuels de classification supervisée. On se trouve face à des problèmes d'explosion combinatoire, ce qui rend les méthodes d'optimisation combinatoire toutes indiquées pour ce type de problème. Dans la littérature, des approches ont été proposées pour traiter chacun de ces problèmes séparément. L'enjeu de cette thèse est ici de proposer une approche qui les traite simultanément.

Le premier chapitre de cette thèse détaille tout le contexte métier et applicatif. Nous commencerons en premier lieu par décrire la problématique métier du recrutement dans les essais cliniques. Pour cela, nous détaillerons tout d'abord le contexte des essais cliniques : déroulement d'un essai, acteurs et bien entendu fonctionnement du recrutement, en particulier du pré-screening, qui est la tâche que nous souhaitons automatiser. Nous identifierons les difficultés qui peuvent se poser lors du recrutement. Nous nous intéresserons ensuite aux données médicales disponibles dans les hôpitaux. En particulier, nous vérifierons si malgré leur problèmes de qualité, les données du PMSI peuvent être utilisées pour la fouille de données. Nous nous focaliserons ensuite sur la conception d'un logiciel d'aide au recrutement : quels sont les logiciels existants ? Quelles fonctionnalités offrent-ils, et avec quels résultats ? Nous présenterons ensuite

comment le logiciel Opcyclin se positionne par rapport aux logiciels existants. Puis, nous expliquerons comment la tâche de pré-screening peut être modélisée sous la forme d'un problème de classification partielle supervisée. Enfin, nous présenterons les différents verrous scientifiques amenés par les données hospitalières, et justifierons l'emploi des méthodes d'optimisation.

Dans le chapitre 2, nous nous intéresserons au contexte scientifique. Nous proposerons ainsi une introduction à la classification supervisée, présentant son fonctionnement et les méthodes pour évaluer sa qualité. Nous nous intéresserons ensuite à un des problèmes posés par les données médicales : la classification sur données asymétriques, lorsque la classe à prédire est peu présente dans les données. Nous présenterons les différentes solutions proposées dans la littérature pour traiter ce problème. Puis, nous présenterons les approches de classification supervisée de la littérature. Nous nous intéresserons ensuite à l'optimisation combinatoire utilisée pour la classification. Nous donnerons tout d'abord une introduction sur l'optimisation combinatoire, les méta-heuristiques et l'optimisation multi-objectif. Ensuite, nous présenterons les différentes approches qui ont été proposées pour la classification. Enfin, nous aborderons les concepts qui seront utilisés dans la majorité des protocoles expérimentaux de cette thèse. Ainsi, nous nous intéresserons aux bonnes pratiques en matière de comparaison statistique d'algorithmes, et présenterons les jeux de données utilisés pour les évaluations.

Dans le chapitre 3, nous montrerons comment le problème de classification partielle sur données asymétriques peut être modélisé sous la forme d'un problème de recherche locale multi-objectif. De cette réflexion, nous proposerons l'algorithme MOCA-I, fondé sur un modèle multi-objectif maximisant la *confiance* et la *sensibilité* et minimisant la taille des ensembles de règles obtenus. Nous étudierons ensuite les bénéfices de la multi-objectivisation : quels avantages apportent la modélisation multi-objectif par rapport à une modélisation mono-objectif, plus simple ? Ensuite, nous déterminerons le meilleur paramétrage à utiliser pour l'algorithme MOCA-I. Finalement, nous comparerons la version de MOCA-I ainsi paramétrée aux autres algorithmes de la littérature.

Dans le chapitre 4, nous étudierons si MOCA-I peut être appliqué à un problème de classification binaire standard, plus largement étudié dans la littérature. Celui-ci n'est plus concerné par les contraintes des données médicales (incertitude, asymétrie et volumétrie). Nous proposerons également deux modélisations alternatives $\text{MOCA}_{SeSpMDL}$ et MOCA_{ExaMDL}, fondées sur des critères classiques en classification binaire standard (*sensibilité*, *spécificité*, *exactitude*). Nous étudierons ensuite l'impact de l'asymétrie sur le choix d'une modélisation : à partir de quand MOCA-I est-il plus efficace que MOCA ? Comment guider ce choix ?

MOCA-I est un algorithme multi-objectif, ce qui implique qu'il génère un ensemble Pareto de solutions de compromis. Dans le chapitre 5, nous étudierons comment choisir efficacement une solution, afin de faciliter la tâche de l'utilisateur. Nous commencerons tout d'abord par montrer l'impact du choix de la solution finale sur les résultats de MOCA-I. Ensuite, nous présenterons deux types de méthodes : des méthodes fondées sur le choix d'une solution finale, ou des méthodes fondées sur la génération d'une nouvelle solution, à partir de toutes celles obtenues. Nous proposerons en particulier des méthodes tentant de contrer le phénomène de sur-apprentissage, ainsi que des méthodes fondées sur la courbe ROC, assez familière au monde

médical. Les différentes méthodes proposées seront évaluées.

Finalement, dans le chapitre 6 nous verrons comment tout ce qui a été présenté précédemment peut s'intégrer dans une suite logicielle : le logiciel Opcyclin. Nous détaillerons tout d'abord les différents modules composant Opcyclin, ainsi que leurs interactions. Chaque module sera ensuite présenté en détails. Nous analyserons la qualité des règles produites par MOCA-I. Nous expliquerons également comment il est possible de générer un score de pertinence pour un patient et un essai clinique, à partir des résultats issus de MOCA-I.

Le manuscrit se terminera par la présentation de perspectives et notamment la possibilité d'étendre les approches proposées à d'autres cas d'utilisations issus des hôpitaux. Par exemple, elles pourraient être étendues à l'analyse de parcours des patients ayant subi un AVC, afin de prédire divers événements comme la récidive ou la survenue de troubles secondaires. Le premier chapitre est précédé d'un glossaire qui définit toutes les abréviations et certains concepts importants qui apparaîtront dans tout le document.

Glossaire

AG Algorithme Génétique.

ARC Attaché de Recherche Clinique. Il s'assure du bon déroulement de l'essai clinique : respect du protocole mis en place pour l'essai clinique, contrôle de la qualité des données récoltées par les TEC.

AVC Accident Vasculaire Cérébral.

Bloat En optimisation, ce phénomène se produit sur les représentations de longueur variable, lorsqu'une solution se complexifie indéfiniment sans pour autant qu'il y ait une amélioration de sa ou ses fonctions objectif.

CHRU Centre Hospitalier Régional Universitaire.

CIC Centre d'Investigation Clinique. Il s'agit d'un centre de soins dédié exclusivement aux essais cliniques.

Confiance Elle est parfois également appelée Valeur Prédictive Positive (VPP). Elle permet de représenter la fiabilité d'une classification : il s'agit de la probabilité qu'une observation ait la prédiction, sachant que le classifieur l'a détectée comme positive. Elle est également utilisée pour évaluer la fiabilité d'un test diagnostic.

CRF Clinical Research Form. Il s'agit d'un formulaire qui collecte des données médicales ciblées d'un patient au cours d'un essai clinique.

CRO Contract Research Organization. Il s'agit d'une compagnie sous-traitant tout ou une partie de la gestion d'un essai clinique.

DMLS Dominance-based Multi-Objective Local Search.

EMR Electronic Medical Record. Il s'agit d'un dossier patient numérisé.

FAStroke File Active Stroke. Il s'agit d'une file active destinée à collecter l'ensemble des patients passés dans certains services neuro-vasculaires français en 2012. Ce fichier permet ensuite de réaliser des études de faisabilité pour des essais cliniques.

FEVG La Fraction d'Éjection du Ventricule Gauche est une estimation de la contraction du cœur, elle est utilisée notamment pour mesurer l'insuffisance cardiaque.

GHM Groupe Homogène de Malades. Á la fin de chaque séjour PMSI, l'ensemble des données PMSI du séjour (actes et diagnostics) est passé dans un logiciel de groupage, qui va classer chaque séjour dans un groupe de séjours (ex : AVC niveau 1)

GLOSSAIRE

HL7 Il s'agit d'un format d'échange de données médicales.

IHM Interface Homme-Machine.

Inclusion Il s'agit d'un patient participant à un essai clinique, ayant donné son consentement pour participer à l'essai clinique et respectant les critères d'éligibilité nécessaires pour y participer.

MDL Minimum Description Length principle. Il est parfois appelé principe de description minimale : lorsque deux hypothèses sont identiques, la plus simple doit être privilégiée.

NIHSS Le score NIHSS (NIH Stroke Scale) est une échelle qui permet de quantifier la gravité d'un AVC.

PMSI Programme de Médicalisation des Systèmes d'Information. Les données qui en sont issues sont utilisées par le gouvernement français dans le cadre de la *Tarification à l'activité (T2A)* depuis 2005. Elles correspondent à l'ensemble anonymisé des séjours et actes médicaux effectués dans les hôpitaux français.

PU-PH Professeur des Universités - Praticien Hospitalier.

RCP Réunion de Concertation Pluridisciplinaire. Il s'agit de réunions auxquelles participent des médecins de plusieurs spécialités afin de définir le traitement à suivre pour certains patients, souvent en cancérologie.

RSS Le Résumé de Sortie Standardisé regroupe toutes les informations d'un patient lors de son séjour hospitalier (y compris les RUM).

RUM Le Résumé d'Unité Médicale représente le passage du patient dans un des services de l'hôpital.

Sensibilité Elle permet de détecter la proportion des observations positives qui sont détectées par un classifieur. Elle est également utilisée pour évaluer la fiabilité d'un test diagnostic.

SIH Système Informatique Hospitalier. Il s'agit du système où est contenu l'ensemble des dossiers patient d'un hôpital.

Spécificité Elle permet de représenter la proportion d'observations négatives qui sont correctement classées négatives par le classifieur. Elle est également utilisée pour évaluer la fiabilité d'un test diagnostic.

T2A Tarification à l'acte.

TAL Le Traitement Automatique du Langage désigne un ensemble de techniques destinées à traiter automatiquement du texte non structuré.

TEC Technicien d'Étude Clinique. Il se charge de toutes les tâches liées à la mise en place d'un essai clinique dans un centre promoteur ou investigateur : tâches administratives, saisie et contrôle de la qualité des données de suivi patient, recrutement des patients, récupération des traitements, visites de suivi, envoi des échantillons et examens, dialogues avec le promoteur, etc.

TMS Thérapies par stimulation magnétique transcranienne.

UML Unified Modeling Language. Il s'agit d'un langage de modélisation informatique, utilisé pour la représentation de processus métiers ou logiciels.

VPN Valeur Prédictive Négative. Elle permet de représenter la fiabilité d'un classifieur lorsqu'il prédit qu'une observation est négative. Il s'agit de la probabilité que l'observation soit négative, sachant que le classifieur l'a classée comme négative. Elle est également utilisée pour évaluer la fiabilité d'un test diagnostic.

GLOSSAIRE

Chapitre 1

Contexte

Ce chapitre est dédié à la description et l'étude de la problématique métier dans laquelle s'inscrit le travail de cette thèse. Le vocabulaire spécifique aux essais cliniques sera présenté, ainsi que le déroulement d'un essai clinique et plus particulièrement de la phase de recrutement, avec les difficultés qui y sont rencontrées. Les données disponibles dans les hôpitaux français seront ensuite détaillées avec quelques indicateurs supplémentaires sur la qualité des données *PMSI* (données du Programme de Médicalisation des Systèmes d'Information, qui correspondent à une informatisation des séjours et des actes effectués dans les hôpitaux français). Un panorama des différents outils d'aide à la décision pour les essais cliniques sera exposé, ainsi qu'une description et présentation fonctionnelle du projet *Opcyclin*, dans lequel s'inscrit ce travail. Enfin, nous expliquerons comment le pré-screening dans les essais cliniques peut être modélisé sous forme d'un problème de classification et les verrous scientifiques sous-jacents, ce qui a fait l'objet d'une communication à la conférence ROADEF 2011 [58].

1.1 Enjeux

Les essais cliniques permettent d'améliorer la qualité et l'efficacité des soins. Ils préparent les techniques de soin de demain. Ils concernent à la fois les dispositifs médicaux (comme les pacemakers), les médicaments, les procédures de soin ou encore les connaissances médicales. Le plus souvent, les patients pris en charge dans un essai clinique bénéficient d'un meilleur suivi et d'un traitement plus efficace qu'avec la prise en charge habituelle. En 2009, la France comptabilisait 1685 essais cliniques en cours incluant 83 623 patients [1].
En pratique cependant, peu de patients "éligibles" à un essai clinique se verront proposer une participation : la France souffre d'un faible taux de recrutement de patients par centre. Elle s'éloigne de la moyenne européenne : 4,8 patients en moyenne sont inclus par centre en France contre 6,0 dans les autres pays [2]. D'une part ce problème est préjudiciable au patient, qui perd l'opportunité de bénéficier d'un traitement alternatif. D'autre part, certains essais cliniques peuvent être abandonnés faute de candidats. Par ailleurs, il y a une demande croissante des

1. source : AFSSAPS
2. source : Enquête du LEEM de 2012 : "La Place de la France dans la recherche clinique internationale"

organismes financeurs pour mesurer l'efficacité du recrutement en fin d'essai.
Un système d'aide à la décision peut répondre à ces problématiques sur plusieurs plans :

À la conception de l'essai : Prévoir le nombre de recrutements à venir : l'essai est-il réalisable ?

En cours d'essai : Aider les recruteurs à identifier les patients potentiels.

En fin d'essai : Détecter les patients non recrutés : sont-ils nombreux ? Pourquoi les a-t-on manqués ?

Un tel système permettrait notamment d'économiser le temps du personnel soignant affecté aux essais cliniques : en 2010, Lynne Penberthy *et al.* ont comparé les recrutements traditionnels et les recrutements effectués par un système automatisé [87]. Le temps nécessaire aux personnels de santé pour effectuer un recrutement traditionnel est 0.8 à 19.4 fois plus important que pour effectuer ce même recrutement avec un système automatisé. Le temps ainsi économisé pourrait être alloué aux soins et au suivi. À plus long terme, un tel système autoriserait la mise en place d'essais cliniques qui actuellement ne sont pas réalisés faute d'un potentiel de recrutement suffisant : un système créé par le Children's Hospital Informatics Program a montré une meilleure sensibilité que la méthode traditionnelle : 84% des patients éligibles ont été détectés par le système automatisé, contre 56% avec la méthode traditionnelle [18, 111].

1.2 Mise en place et déroulement d'un essai clinique

Afin de compléter au mieux nos connaissances du métier, plusieurs entretiens ont été réalisés avec des experts du métier. Nous avons interviewé Mme Bordet, actuellement attachée de recherche clinique (ARC) au service neuro-vasculaire du CHRU (Centre Hospitalier Régional Universitaire) de Lille. Nous avons également rencontré Mme Deprez, qui a travaillé en tant qu'ARC, à la fois pour le service public - au service de pneumologie du CHRU de Lille - et dans le privé - au sein d'une CRO de 100 lits (*Contract Research Organization*, compagnie sous-traitant tout ou une partie de la gestion d'un essai clinique). Nous avons également organisé des réunions avec le Professeur Beuscart, PU-PH (Professeur des Universités - praticien hospitalier) et le Professeur Bordet, PU-PH également, qui ont tous deux apporté beaucoup d'informations sur le déroulement des essais cliniques de part leur pratique médicale. Cette section résulte de ces divers entretiens.

1.2.1 Définitions et vocabulaire

Cette partie donne la définition d'un essai clinique et présente ensuite tous les acteurs qui interviennent au sein de l'essai clinique. Enfin, elle décrit le déroulement d'un essai clinique.

1.2.1.1 Essai Clinique

Une définition communément admise de l'essai clinique est : *« Toute approche scientifique visant à évaluer une technique ayant pour but la prévention, le diagnostic ou le traitement »* [72]. La durée d'un essai clinique varie entre 3, 5 et 7 ans, selon le type d'essai clinique. Dans le cas d'essais cliniques de suivi sur des pathologies aiguës, le patient est suivi pour une durée de 90 jours dès l'apparition de la maladie ; il n'est donc pas suivi sur toute la durée de l'essai clinique,

d'autres patients pourront être recrutés sur toute la durée de l'essai. Les essais cliniques de cohorte concernent les pathologies chroniques ou des populations particulières (ex : fumeurs et non-fumeurs), les patients sont suivis pendant 5 ans. Enfin pour certains essais concernant les médicaments, les patients peuvent être suivis pendant 7 ans.

1.2.1.2 Acteurs

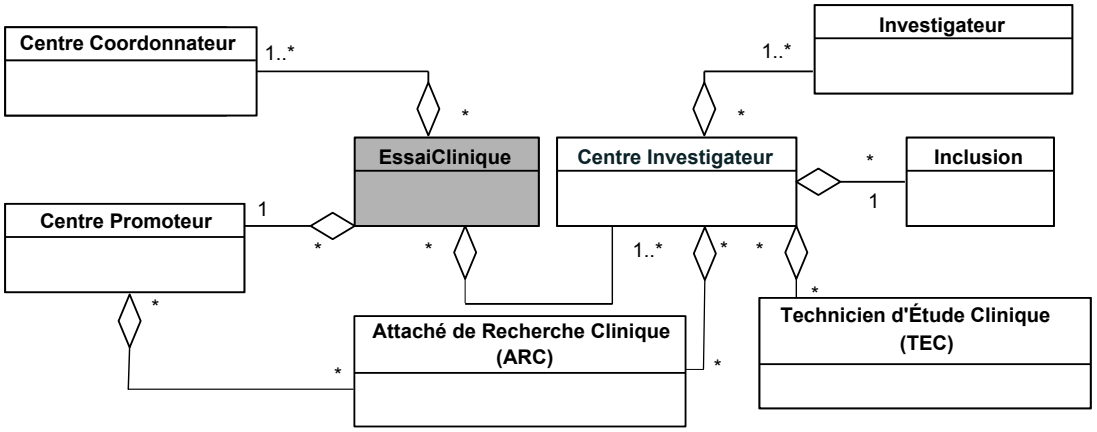


Figure 1.1: Acteurs et intervenants d'un essai clinique - Diagramme de classes UML

Nous proposons une modélisation des différents acteurs et intervenants d'un essai clinique au sein de la figure 1.1, sous la forme d'un diagramme de classes UML. Les différents acteurs sont détaillés ci-dessous :

Centre Promoteur Entité qui décide de mettre en place l'essai clinique : promoteur industriel (pour tester l'efficacité ou la tolérance d'un médicament ou d'un dispositif médical), centre hospitalier ou parfois association.

Centre Coordonnateur Entité qui s'assure du bon déroulement de l'essai, en communiquant avec les centres hospitaliers participant à l'essai : réunions de mise en place, élaboration des protocoles, contrôle et suivi des différents centres participants, gestion administrative, etc. Généralement le centre promoteur se charge du rôle de centre coordonnateur, mais celui-ci peut être délégué à un centre hospitalier ou une société privée spécialisée (CRO).

Centre Investigateur Lieu où se déroule l'essai clinique. Les centres investigateurs se chargent du recrutement et du suivi des patients. Les services des centres hospitaliers peuvent assurer ce rôle, mais également les CIC (Centre d'Investigation Clinique) qui sont des centres de santé dédiés aux essais cliniques, ou encore les centres privés (CRO). Un essai clinique peut comporter entre 1 et 100 centres investigateurs.

Technicien d'étude clinique (TEC) Personnel de santé dédié à la recherche clinique. Il se charge de toutes les tâches liées à la mise en place d'un essai clinique dans un centre promoteur ou investigateur : tâches administratives, saisie et contrôle de la qualité des données de suivi patient, recrutement des patients, récupération des traitements, visites de suivi, envoi des échantillons et examens, dialogues avec le promoteur, etc. Un TEC peut gérer jusqu'à une dizaine d'essais simultanément.

Attaché de Recherche Clinique (ARC) Personnel de santé dédié à la recherche clinique. Il s'assure du bon déroulement de l'essai clinique : respect du protocole mis en place pour l'essai clinique, contrôle de la qualité des données récoltées par les TEC.

Investigateur Personnel de santé d'un centre investigateur affecté à un essai clinique. Le médecin investigateur est responsable de l'essai clinique dans son centre investigateur.

Patient Inclus Patient participant à un essai clinique, ayant donné son consentement et respectant les critères d'éligibilité nécessaires pour y participer. Il ne peut pas participer à plus d'un essai clinique simultanément (hormis pour certains essais observationnels). Le nombre de patients inclus dans un centre investigateur dépend fortement de l'essai clinique concerné.

1.2.1.3 Phases d'un essai clinique

Un essai clinique comporte plusieurs phases dont nous détaillerons les plus importantes pour notre problématique. Nous avons modélisé les différentes phases d'un essai clinique sous la forme d'un diagramme d'états-transitions UML, qui est disponible dans la figure 1.2. L'essai clinique commence tout d'abord par une phase de conception du protocole, selon la finalité de l'essai clinique : il faut s'assurer que l'essai clinique aura une valeur scientifique. Le promoteur va ensuite réaliser une étude de faisabilité dans les centres investigateurs et choisira les centres selon leurs capacités. Le protocole doit ensuite être validé par des autorités compétentes (*Comité de protection des personnes* ou *AFSSAPS* pour les médicaments) pour s'assurer que le protocole respecte l'éthique. L'essai peut ensuite être mis en place dans les centres investigateurs : le promoteur va leur expliquer toutes les procédures à respecter. Ils vont ensuite commencer les recrutements et collecter les données des patients inclus. À la clôture de l'essai, les données seront analysées pour évaluer l'efficacité de l'objet de l'essai clinique.

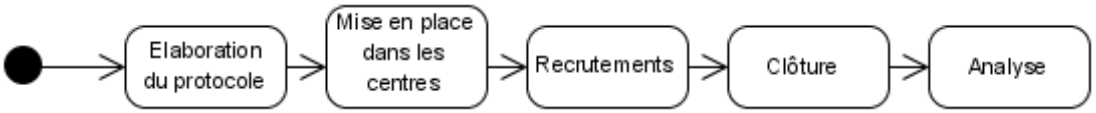


Figure 1.2: Phases d'un essai clinique - Diagramme d'états-transitions UML

1.2.2 Élaboration de l'essai clinique

Afin d'illustrer l'élaboration d'un essai clinique nous utiliserons l'essai clinique *tmstroke* issu du site clinicaltrials.org : *Use of Deep Transcranial Magnetic Stimulation After Stroke*. La conception d'un essai clinique est une phase très importante car elle détermine l'utilité et l'efficacité de l'essai clinique. Tout commence par une question. Par exemple : « Est-ce que les TMS (thérapies par stimulation magnétique transcranienne) peuvent améliorer la guérison d'un patient après un AVC ? (Accident Vasculaire Cérébral) ». Le promoteur va devoir élaborer un protocole médical qui permettra de répondre à la question. Il faut par exemple déterminer le critère de jugement : qu'est-ce qui détermine que la TMS a été efficace ? C'est ce qui va spécifier les données que les centres investigateurs devront collecter lors du suivi des patients. Le promoteur doit également déterminer si l'essai clinique est réalisable : si les patients potentiels sont trop rares l'essai clinique ne pourra pas donner de résultats.

1.2.2.1 Méthodologie

Il est également nécessaire de définir une méthodologie pour l'essai clinique : est-il nécessaire d'avoir un groupe témoin ? D'utiliser un placebo ? Dans l'exemple ci-dessus, il est important d'utiliser un groupe témoin. Des essais cliniques avec ou sans groupe témoin peuvent donner des résultats très différents. Dans un essai sans groupe témoin, si l'essai montre que 90% des patients voient leur état s'améliorer avec la TMS, cette dernière peut sembler être une très bonne méthode thérapeutique. Si l'essai est complété avec un groupe témoin dans lequel 100% des patients ont vu leur état s'améliorer sans la TMS, le résultat est à l'opposé : l'essai clinique met en évidence que la TMS est à éviter après un AVC. Le promoteur va également devoir définir le protocole de soin. Toujours dans notre essai clinique d'exemple, il s'agit de préciser le nombre de séances de TMS, leur durée, le paramétrage du matériel, etc. Il faut s'assurer que chaque centre investigateur va procéder de la même manière.

1.2.2.2 Critères d'éligibilité

Le promoteur doit ensuite élaborer la liste des critères d'inclusion et d'exclusion. Il s'agit des critères que le patient doit absolument respecter (inclusion) ou non (exclusion) pour faire partie de l'essai clinique. Certains essais en comportent jusqu'à 30. Il n'y a pas de possibilité de recruter un patient qui ne respecte pas un des critères. Ils sont de diverses natures : démographique, diagnostics, mesures biologiques, traitements médicamenteux, contre-indications et allergies, etc. Ils changent rarement en cours d'essai, sauf si le promoteur souhaite élargir ou préciser certains critères. Lorsqu'un critère n'est pas assez précis, l'investigateur doit toujours avoir l'aval du promoteur avant de recruter un patient pour lequel il a un doute. Voici quelques exemples de critères :

- $18 \leq$ âge ≤ 85,
- score NIHSS [1] ≤ 18,
- absence d'antécédent d'épilepsie,
- absence d'anévrisme crânien,
- absence de traitement anti-thrombotique,
- ...

Certains critères peuvent posséder une notion temporelle, ou une notion de précédence. Par exemple pour l'essai clinique NCT01144715 [2] :

- AVC de moins de 6 mois,
- perte d'indépendance avant l'AVC.

Enfin, il existe également des critères qui nécessitent un avis médical, comme par exemple *« espérance de vie inférieure à 3 mois »*.

1.2.2.3 Étude de faisabilité

Après la conception du protocole de l'essai, le promoteur va effectuer une étude de faisabilité. Il va présenter le protocole et ses divers besoins - techniques, matériels, qualifications, nombre de patients attendus - aux centres investigateurs. Chaque centre va constituer un dossier de

1. Le score NIHSS (NIH Stroke Scale) est une échelle qui permet de quantifier la gravité d'un AVC
2. ClinicalTrials.gov : http ://clinicaltrials.gov/

candidature, estimer son volume prévisionnel d'inclusions (ex : 3 patients par mois) et le communiquer au promoteur. Il est à noter que prévoir le volume prévisionnel d'inclusions est crucial mais peut être ardu, d'où le besoin d'un outil d'aide à la décision dédié à cette tâche. Si trois patients se présentent par mois avec le profil requis, le personnel qui effectue le recrutement pourra en recruter au mieux trois ; ajouter du personnel de recrutement n'augmentera pas le nombre de patients qui se présentent. Une fois les candidatures acceptées et le protocole validé par les autorités compétentes, le promoteur va présenter l'essai clinique dans chaque centre investigateur. Les centres se chargeront ensuite d'informer le personnel de santé de l'essai en cours (ARCs, TECs, médecins).

1.2.3 Recrutement : pré-screening, screening et inclusion

Le recrutement se déroule en trois phases : le pré-screening permet tout d'abord d'identifier des patients candidats à un essai clinique. Les dossiers de ces patients sont ensuite étudiés en détails dans la phase de screening pour finalement proposer ou non l'inclusion du patient dans l'essai clinique. Ces trois phases sont détaillées ci-dessous.

1.2.3.1 Pré-screening : identifier des patients potentiels

Le pré-screening est effectué régulièrement par les TECs. Il consiste à identifier des patients candidats pour un essai clinique. Sur les essais concernant des maladies aiguës, ils parcourent tous les matins les dossiers des patients entrés dans le service et vérifient s'ils respectent les critères d'inclusion d'un des essais en cours. À titre d'exemple, au service neuro-vasculaire de Lille, cette étape concerne environ 10 patients par jour et nécessite jusqu'à une heure de travail. Une partie des critères d'inclusion peut être commune à plusieurs essais, cependant le centre investigateur évite de participer à des essais qui entrent en concurrence au niveau des recrutements. En cancérologie, le pré-screening peut également s'effectuer durant les réunions de consortium ; il s'agit de réunions pluridisciplinaires où plusieurs médecins spécialistes discutent des soins et traitements de certains patients.
Concernant les maladies chroniques et essais cliniques de cohortes, le pré-screening s'effectue sur le planning des 15 jours de consultations à venir, ou à l'aide d'affiches en salle d'attente. Enfin, certains collègues d'un médecin investigateur peuvent parfois lui proposer des patients.

1.2.3.2 Screening : vérifier en détail les critères d'inclusion

Une fois un patient identifié par le pré-screening, il faut vérifier en détail si son profil correspond aux critères d'inclusion et de non inclusion de l'essai clinique. Il s'agit d'une étape assez fastidieuse : certains critères d'inclusion nécessitent de consulter les archives. D'autres critères nécessitent parfois la réalisation d'examens médicaux, qui seront effectués après un entretien avec le patient. Dans certains cas, les phases de pré-screening et de screening font en fait partie d'une même phase, ce qui est par exemple le cas de certains essais cliniques concernant des pathologies aiguës où le recrutement doit être effectué très rapidement (quelques heures). En moyenne, un à deux patients seront inclus par jour et par service, tous essais confondus. Malgré les efforts des centres investigateurs pour éviter de participer à plusieurs essais "concurrents", il peut arriver que le profil du patient corresponde à plusieurs essais cliniques. Dans ce cas, le médecin investigateur pourra privilégier l'essai qui donnera le plus de confort au patient ; il est

ainsi possible de privilégier un essai clinique avec moins de jours d'hospitalisation si le patient le souhaite. Dans le cas des essais de cohorte ou observationnels, le patient peut participer à plusieurs essais. Dans ce cas, la difficulté est d'identifier les essais dont les inclusions sont "compatibles".

1.2.3.3 Inclusion et consentement

Lorsqu'un patient est pré-senti pour un essai clinique, il va participer à une consultation au cours de laquelle le déroulement de l'essai clinique va être présenté : examens et visites de suivi nécessaires, risques et bénéfices. Le médecin ou l'ARC doivent s'assurer que leurs explications sont claires et que le patient participe en étant conscient de ce qu'implique l'essai clinique. Sa motivation sera également évaluée car certains essais cliniques peuvent être lourds en examens et procédures ou comporter de nombreuses visites de suivi. Dès que le patient donne son consentement, il est inclus dans l'étude.

1.2.3.4 Difficultés de recrutement

Les difficultés de recrutement sont nombreuses. Les plus courantes que nous avons identifiées ou relevées dans la littérature sont listées ci-dessous, pour davantage de détails ou d'exhaustivité nous invitons le lecteur à se référer à l'article de Campbell *et al.* [20].

Recrutement avant traitement Des pathologies aiguës comme l'AVC ou l'infarctus nécessitent une prise en charge très rapide. Cependant les essais cliniques associés nécessitent que le patient soit inclus dans l'essai avant tout traitement. L'investigateur dispose donc de très peu de temps entre l'apparition de la pathologie et l'inclusion. Ainsi, l'essai clinique *OPHELIE*[1] évalue l'efficacité d'un traitement administré au maximum 4h30 après l'apparition d'un AVC, ce qui ne permet pas de recruter les patients lors du screening. Dans ce cas, c'est souvent le médecin lors de la prise en charge de l'AVC qui va contacter le TEC ou le médecin investigateur pour lui signaler le patient. Dans certains essais le recrutement peut même être effectué dans l'ambulance. Ce recrutement est difficile à automatiser car il nécessite une saisie du dossier patient en temps réel, ce qui est rarement le cas.

Manque d'information, problème de faisabilité L'ensemble du personnel soignant en contact avec un patient potentiellement incluable n'est pas toujours au courant des essais en cours. La tâche est d'ailleurs difficile car un service peut participer à 15 essais simultanément. Dans certains services, le personnel soignant est trop nombreux et se relaie ce qui rend coûteux la mise en place de formations.

Manque de temps Le recrutement n'est pas toujours compatible avec les soins. Il nécessite du temps supplémentaire, qui n'est pas toujours disponible. Cela impacte le personnel de santé, qui n'a pas toujours le temps de mettre en place les démarches de recrutement. Cela impacte également les TECs qui n'ont pas le temps de parcourir de manière exhaustive les consultations planifiées. Allouer plus de temps aux TECs peut permettre d'augmenter les recrutements : au CHRU de Lille pour un essai clinique donné, un TEC avait été recruté pour parcourir l'ensemble des rendez-vous et consultations pendant 3 mois, ce qui avait

1. Outcome of Patients Treated by iv Rt-PA for Cerebral Ischaemia According to the Ratio Sc-tPA/Tc-tPA ; essai clinique promu par le CHRU de Lille qui évalue les effets du traitement de l'AVC par la molécule Rt-PA

augmenté les recrutements au prix d'une ressource dédiée. Cependant il n'est pas toujours possible de recruter un TEC supplémentaire.

Erreur de conception du protocole À la conception du protocole certains critères d'inclusion peuvent être mal choisis, ou trop restrictifs. Le nombre d'inclusions est alors impacté car les patients qui correspondent aux critères sont rares ou inexistants. Cependant dès lors que le ou les critères fautifs sont identifiés, il est parfois possible de les supprimer ou de les élargir à l'aide d'un amendement au protocole. L'essai peut ainsi améliorer ses recrutements.

Manque de motivation Les difficultés peuvent également provenir d'un manque de motivation. Elles s'expriment de manières diverses et variées : les recruteurs peuvent reporter la première inclusion, espérant avoir de meilleures inclusions dans le futur, ou encore se fixer des critères d'exclusion non présents dans le protocole ; un interne peut oublier de faire signer le consentement, ce qui invalide l'inclusion. Généralement un suivi de projet avec la diffusion de quelques indicateurs - comme un pourcentage des inclusions "ratées" ou la courbe d'inclusions - permet d'améliorer le recrutement. Cependant ces indicateurs ne sont pas faciles à collecter ou interpréter.

1.3 Données médicales et système d'information hospitalier

1.3.1 Données médicales

Les centres hospitaliers utilisent des systèmes informatiques pour gérer leurs dossiers patients. Pour certains, ces systèmes se limitent au système PMSI qui sera présenté dans la suite ; d'autres bénéficient de systèmes propriétaires gérant la totalité du dossier patient : traitements et prescriptions, résultats d'examens, comptes-rendus, courriers et diagnostics. Le système PMSI et les courriers médicaux présentent l'avantage d'être présents dans tous les hôpitaux français, ce qui n'est pas le cas des systèmes propriétaires. Dans la suite, nous présenterons uniquement les données présentes dans la majorité des hôpitaux français. Sur les données PMSI, une étude de qualité est réalisée afin de vérifier l'utilisabilité de ces données. Plus d'informations sur les formats de données utilisés sont données dans les annexes (voir section A).

1.3.1.1 Données de facturation PMSI

Ces données sont issues du *Programme de Médicalisation des Systèmes d'Information (PMSI)*. Elles sont utilisées par le gouvernement français dans le cadre de la *Tarification à l'activité (T2A)* depuis 2005. Chaque hôpital doit ainsi lui faire parvenir un récapitulatif contenant chacun des séjours effectués, anonymisés, contenant les actes médicaux réalisés durant le séjour et les diagnostics ayant affecté la prise en charge. Les séjours sont ensuite regroupés selon leur coût, ce qui permet de subventionner les hôpitaux selon leur activité. Ces données sont codées avec les nomenclatures *CIM10* et *CCAM*. Ces codages seront détaillés dans les annexes à la section A page 181.

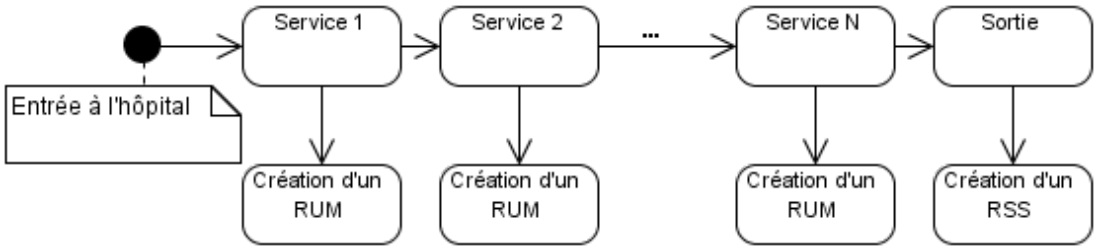


Figure 1.3: Informations PMSI liées à un séjour hospitalier - Diagramme d'état-transitions UML

Structure des données RUM et RSS Dans la figure 1.3 nous avons représenté le déroulement d'un séjour PMSI et la création des fichiers associés. Dans la figure 1.4 nous avons représenté la structure d'un fichier PMSI et les informations qui y sont contenues, sous la forme d'un diagramme de classes UML. Lorsqu'un patient quitte l'hôpital après un séjour, un *Résumé de Sortie Standardisé (RSS)* est généré. Il regroupe des *Résumés d'Unité Médicale (RUM)* qui représentent le passage du patient dans un des services de l'hôpital. Prenons l'exemple d'un patient venant à l'hôpital pour une crise d'appendicite. Il va tout d'abord se rendre aux urgences où il sera rapidement redirigé vers le service de gastro-entérologie ou de chirurgie, donnant lieu à la création de deux *RUM* : un pour le passage aux urgences et un second pour le service où il sera opéré. Dans chacun de ces *RUM* vont être renseignés les actes médicaux effectués : scanner, échographie, prise de sang, etc. Les dates d'entrée et de sortie dans les services vont permettre de dater les actes. Les diagnostics vont également être renseignés : diagnostic principal pour le diagnostic qui justifie le séjour (appendicite) ; diagnostics reliés lorsqu'il y a besoin de préciser le diagnostic principal avec par exemple une localisation ; diagnostics associés pour les diagnostics qui ont modifié le coût de la prise en charge, par exemple une hémorragie ou une complication.

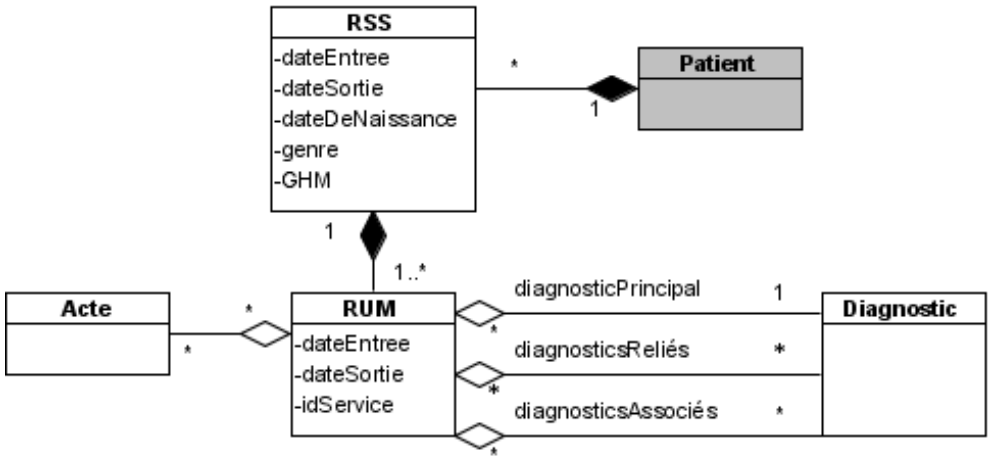


Figure 1.4: Structure des données PMSI - Diagramme de classes UML

Un dossier médical partiel Les données PMSI étant destinées à la T2A, seules les données présentées précédemment sont obligatoires. Cependant il est également possible de renseigner des diagnostics n'ayant pas eu d'effet sur la prise en charge, il s'agit des diagnostics à visée documentaire. Par exemple si le patient souffre de diabète mais que cela ne modifie pas sa prise en charge par rapport à un patient non diabétique, il n'y a pas obligation de renseigner le diagnostic de diabète dans le PMSI. Quelques rares centres hospitaliers le renseigneront cependant. Ainsi, les données PMSI représentent une vue partielle du dossier patient, certains diagnostics pouvant en être absents. Les RSS et RUM sont généralement complétés à la sortie de l'hôpital mais il existe également des cas, plus rares, où ces données sont complétées au fil de l'eau.

Qualité métrologique et valeur prédictive des données PMSI Les données PMSI ont l'avantage d'être présentes dans tous les hôpitaux français, mais l'inconvénient d'être biaisées car destinées à la facturation. Avant toute manipulation ou utilisation de ces données il est juste de se demander si le biais ne sera pas trop important. Ainsi, Bejot *et al.* ont démarré l'étude PMSI-AVC, pour évaluer la possibilité d'utiliser les données PMSI dans le cadre de la prise en charge de l'AVC [10]. Dans le cadre du projet AKENATON [1], nous avons eu l'occasion de comparer les données issues du PMSI à l'avis d'un expert médical. Cette évaluation a été réalisée sur 62 patients et permet de mettre en évidence la fiabilité des données PMSI par rapport à l'avis de l'expert sur le dossier.

Table 1.1: Qualités prédictives du PMSI (projet AKENATON). VPN : Valeur Prédictive Négative.

Critère	Formule	Hypertension	Diabète	Insuffisance cardiaque	Maladie artérielle périphérique
Confiance (VPP)	$P(M\|pmsi)$	1.00	1.00	0.93	0.71
Sensibilité	$P(pmsi\|M)$	0.90	0.79	0.70	0.25
Spécificité	$P(\neg pmsi\|\neg M)$	1.00	1.00	0.98	0.95
VPN	$P(\neg M\|\neg pmsi)$	0.83	0.94	0.87	0.71

Bonne sensibilité et confiance, mauvaise VPN La table 1.1 présente les résultats de l'évaluation que nous avons réalisée sur les données AKENATON. La première colonne contient les différentes mesures utilisées pour l'évaluation. Des informations détaillées pour la compréhension des différentes mesures utilisées (*Confiance*, *Sensibilité*, *Spécificité*, *VPN (Valeur Prédictive Négative)*) sont disponibles dans la section 2.1.2.3 page 42. La deuxième colonne donne l'expression de la mesure en terme de probabilités, sachant que M représente le fait que le médecin a détecté la pathologie dans le dossier patient ; $pmsi$ représente le fait que la pathologie est présente dans le dossier PMSI. Chaque colonne à partir de la troisième représente une pathologie. Pour chacune de ces pathologies, l'information donnée par le PMSI ($pmsi$) a été comparée à l'information donnée par l'expert médical (M). Ces données montrent que le

1. AKENATON était un projet ANR TecSan qui visait à reclasser les alertes issues d'un pacemaker en fonction du contexte patient (antécédents, traitements en cours, etc.), à l'aide d'un système expert sous forme de règles et d'ontologies. (http ://resmed.univ-rennes1.fr/akenaton/)

PMSI a une très bonne confiance pour l'*hypertension* et le *diabète*, et bonne pour les deux autres pathologies : lorsque l'information est indiquée dans le dossier PMSI, elle est juste car confirmée par le médecin. Elles montrent également une sensibilité élevée pour indiquer la présence de *l'hypertension*, du *diabète* et de *l'insuffisance cardiaque*. Ainsi, pour 90% des patients hypertendus qui ont été signalés par l'expert, l'hypertension est indiquée dans le dossier PMSI. Les scores VPN - valeur prédictive négative - sont moins bons : l'absence d'un diagnostic dans le PMSI ne correspond pas toujours à l'absence de l'information dans le dossier complet du patient. Ces informations nous permettent de quantifier l'absence d'exhaustivité dans le PMSI, qui pour les 4 pathologies de la table n'est pas trop importante et permet d'envisager l'utilisation d'algorithmes d'apprentissage.

Existence de problèmes d'expressivité La pathologie *Maladie artérielle périphérique* cependant est très mal renseignée dans le PMSI : lorsqu'elle est indiquée par le médecin, on la retrouve dans les données PMSI uniquement dans 25% des cas. L'explication réside dans une difficulté des experts à identifier la maladie. De plus, cette pathologie correspond à un regroupement de plusieurs autres pathologies, certaines n'existant pas dans le codage PMSI. Ainsi, il existe des pathologies qui ne pourront pas, ou difficilement, être exprimées avec le dossier PMSI.

1.3.1.2 Files actives

Certains services ou spécialités disposent parfois d'une file active : il s'agit d'un répertoire de patients qui ont été reçus au cours d'une période de référence (par exemple l'année courante). Concernant les pathologies neuro-vasculaires, la file active FAStroke recense tous les patients passés par le service neuro-vasculaire de certains hôpitaux français. Pour chaque patient, elle recense les informations susceptibles d'être utiles à un essai clinique sur ces thématiques (pathologie, examens effectués, délai depuis l'AVC...). Elle permet de réaliser des études de faisabilité et des analyses du recrutement a posteriori.

1.3.1.3 Comptes-rendus et courriers médicaux

Les dossiers patients hospitaliers contiennent des courriers médicaux et comptes-rendus. Ils peuvent se trouver sous différentes formes : HL7 pour les hôpitaux équipés d'un Système Informatique Hospitalier (SIH), documents Word ou fichiers texte. Il s'agit de données couramment disponibles dans les hôpitaux mais ayant un inconvénient majeur : elles sont non-structurées et correspondent à du langage naturel ce qui les rend difficiles à exploiter. À l'instar du projet *AKENATON*, les projets qui emploient ce type de données recourent à des outils de *Traitement Automatique du Langage (TAL)* pour extraire les données importantes sous forme structurée.

1.4 Aide à la décision et automatisation pour les essais cliniques : état de l'art

Plusieurs types d'outils sont possibles pour améliorer le déroulement d'un essai clinique. Il peut s'agir d'un outil qui apporte une aide lors de la conception de l'essai : conception des critères d'inclusion ou estimation des recrutements futurs. Durant l'essai clinique, les outils

peuvent aider au recrutement. Enfin, à la fin de l'essai clinique comme durant les recrutements il est possible d'évaluer son bon déroulement : recrute-t-on assez ? Assez vite ? Le recrutement est-il efficace ? Cette section présente une revue de littérature des différents outils existants, elle se focalise tout d'abord sur les approches destinées à aider le recrutement, classées selon la source des données qu'elles utilisent. Ensuite, les outils d'aide à la conception de l'essai seront présentés, suivis par les outils d'évaluation de l'essai. Enfin, un tableau fournira le récapitulatif de toutes les approches présentées.

1.4.1 Outils de pré-screening : aide au recrutement

Le plus souvent le recrutement de patients s'effectue manuellement. Un outil d'aide au recrutement peut intervenir de deux manières : permettre d'isoler une liste de potentiels d'inclusion en temps réel, en envoyant une alerte dès qu'un patient potentiel entre dans le service, ou à la demande en filtrant une liste de patients ; l'autre approche est de trouver quel(s) essais peu(ven)t être assigné(s) à un patient présent en consultation. En effet il peut y avoir jusqu'à 15 essais en cours dans un même service, ce qui rend difficile le choix de l'essai à privilégier pour un patient. Certains centres hospitaliers ont mis en place des outils spécifiques pour aider au recrutement afin d'augmenter les inclusions. La plupart de ces outils concerne des approches AD-HOC : le logiciel est adapté à un seul hôpital, une seule région voire à un logiciel de gestion des données patient spécifique. Dans tous les cas ces outils agissent uniquement comme une aide à la décision : l'investigateur retraitera le résultat obtenu.

1.4.1.1 Outils nécessitant un système EMR

Quelques établissements de santé disposent d'un système informatique hospitalier complet ou EMR (*Electronic Medical Record*), ce qui leur permet d'utiliser des outils d'aide au recrutement. Aux États-Unis, l'outil ASAP (*Advanced Screening for Active Protocols*) permet d'isoler une liste de profils de patients candidats à un essai à partir d'une liste de critères d'éligibilité, d'un entrepôt de données complet et de traitement automatique du langage (TAL) [33, 65] Toujours aux États-Unis, l'outil *IKnowChart* de la société *IKnowMed* se greffe sur leur suite logicielle de cancérologie. En s'appuyant sur les données récoltées par la suite logicielle, le module propose une liste de patients correspondant aux critères d'éligibilité [45].

Diabète Aux États-Unis deux expérimentations ont été réalisées sur des essais cliniques sur le diabète : un outil a permis de réduire le travail des recruteurs sur l'essai clinique *ACCORD* (Contrôle du risque cardio-vasculaire pour les patients atteints du diabète) pendant 2 semaines, en filtrant 193 patients, pour en obtenir 23 potentiels. Les 102 patients exclus par le système ont été confirmés non éligibles par un expert [105]. Des expérimentations ont également été réalisées par la même équipe sur l'essai *TECOS* [113]. Un autre outil se basant sur des données du logiciel propriétaire *EpicCare* détecte lors de la consultation si le patient correspond aux critères [37]. Les recrutements ont ainsi doublé et les investigateurs sont en mesure de proposer un essai clinique à un patient lors de la consultation 10 fois plus souvent. Une autre expérimentation a eu lieu au Royaume-Uni, en utilisant l'outil *SARMA* qui recrute des patients présents dans le système EMR et envoie une alerte durant les consultations des patients potentiels [107].

1.4.1.2 Outils génériques

Quelques outils ont été conçus pour fonctionner dans une majorité de centres hospitaliers. Ils utilisent des données d'activité Medicare (équivalent américain du PMSI), des résultats de laboratoire et du TAL sur des comptes-rendus médicaux. Un de ces outils, spécialisé en cancérologie, a été testé sur 5 essais cliniques. Les recrutements ont été effectués à la fois avec la méthode conventionnelle et avec le logiciel, ce qui a permis de comparer les performances. Des améliorations notables du recrutement sur 4 de ces essais ont été mises en évidence, diminuant fortement le temps passé par le personnel de santé pour le recrutement [87].

1.4.1.3 Recrutement via formulaires internet

Certains outils utilisent les données renseignées par les patients eux-mêmes pour participer à l'essai clinique. Le patient remplit lui-même ses informations cliniques et se voit proposer une liste d'essais candidats. L'avantage de cette technique est qu'elle propose uniquement des patients qui ont montré un intérêt pour l'essai clinique. Aux États-Unis, un logiciel s'appuie sur un annuaire de patients affectés par des maladies orphelines. Les patients s'enregistrent sur le site internet et sont prévenus dès qu'un essai peut correspondre à leur profil. L'évaluation de l'éligibilité s'effectue sur des critères d'âge, de genre, de maladie et de localisation géographique [93]. Les outils *TrialX* et *MindTrial* utilisent tous deux la technologie des ontologies : il s'agit de bases de connaissances stockées sous la forme de graphes, qui modélisent les relations entre différents concepts et peuvent être utilisées à la manière d'un système expert. Ces outils couplent les ontologies aux données patients saisies via un site internet afin d'effectuer le recrutement sur tous types d'essais [85, 86]. On retrouve également une approche spécialisée à la psychiatrie [73]. L'outil *TrialX* importe également les dossiers patients gérés sous *Google Health* ou *Microsoft Health Vault*.

1.4.1.4 Projets exploratoires et outils AD-HOC

Toujours aux États-Unis, le *Children's Hospital Informatics Program (CHIP)* a testé un système d'alertes en temps-réel, développé pour un essai clinique spécifique. Dès que le système d'information détecte l'admission d'un patient aux urgences avec une hypoglycémie, une alerte est envoyée au coordonnateur de l'essai clinique [18, 111]. Ce système est fondé sur les informations présentes dans le dossier des urgences : résultats de laboratoire et descriptif entré par le médecin. Des mots-clefs sont recherchés dans le descriptif du médecin, afin d'identifier les potentiels d'inclusion. Sur 10 mois, ce logiciel a lancé des alertes pour 84% des 49 patients potentiellement éligibles. Tandis qu'avec la méthode de référence, le personnel des urgences a identifié sur 11 mois 56% des 61 patients potentiellement éligibles.
Un autre projet à Harvard s'intéresse au cancer du sein. Les patients et praticiens peuvent remplir des fiches du profil patient et ainsi se voir proposer des essais correspondants. Lorsque trop d'essais sont proposés, l'application génère un formulaire dynamique contenant les critères à renseigner en priorité pour filtrer la liste d'essais. Chaque essai de la liste possède un score correspondant au nombre de critères restant à renseigner. Les données incomplètes sont gérées à partir de réseaux bayésiens. Les critères d'éligibilité sont codés dans une grammaire XML spécialement conçue pour cette problématique. Leur approche a été testée sur 85 essais [4, 80]

En France Le projet *ASTEC (Automatic Selection of clinical Trials based on Eligibility Criteria)* utilise les comptes-rendus de RCP (Réunions de Concertation Pluridisciplinaire) et la technologie des ontologies [12]. Il permet d'associer à un patient une liste d'essais cliniques de cancérologie auxquels le profil du patient peut correspondre, ainsi qu'un rapport d'éligibilité. À l'*AP-HP (Assistance Publique – Hôpitaux de Paris)*, l'outil *OncoDoc* d'aide à la décision pour le cancer du sein a été testé comme outil de recrutement [96]. Les données du patient sont saisies manuellement et les essais correspondants sont ensuite trouvés à l'aide d'arbres de décision. Utilisé pendant 4 mois à *l'Institut Gustave Roussy* (Villejuif), il a permis d'améliorer la connaissance du personnel médical des essais cliniques en cours. Le recrutement a ainsi été amélioré de 50%

1.4.2 Aide à la conception de l'essai

Estimation du nombre de patients Une première méthode pour aider à la conception d'un essai clinique est d'estimer le nombre de patients que l'essai clinique pourra recruter. S'il est trop faible, les critères d'inclusion sont à revoir. De nombreux outils ont été conçus dans ce but, Barnard *et al.* les ont recensés [9] ; beaucoup sont fondés sur des méthodes statistiques. L'outil de Taylor *et al.* par exemple, permet de prédire le nombre de patients attendus dans le cadre d'un essai sur les AVC [104].

Aide à l'élaboration des critères d'inclusion Une autre méthode est d'aider au choix de l'ensemble des critères d'éligibilité. Ainsi, Rubin *et al.* ont conçu un logiciel permettant d'afficher les critères d'éligibilité les plus courants pour une pathologie [95]. L'avantage est qu'il permet ainsi de normaliser les critères d'inclusion d'un essai à l'autre, en utilisant par exemple les mêmes seuils pour certaines mesures biologiques. Les risques d'oublier un critère sont également diminués. Gennari *et al.* ont également réalisé un plugin pour Protégé (logiciel de gestion d'ontologies) permettant d'élaborer les critères d'inclusion [46], qui communique avec l'outil *IKnowMed* précédemment présenté.

1.4.3 Évaluation du bon déroulement d'un essai

L'évaluation du bon déroulement d'un essai peut s'effectuer en cours d'essai - ce qui permet de rectifier rapidement un problème - ou à la fin de l'essai. Divers indicateurs sont disponibles dans ce but. Stanley *et al.* en décrivent certains qu'ils ont utilisés pour évaluer l'activité des centres investigateurs d'un réseau clinique de pédiatrie en 2007 [99]. Ils se sont servi des données issues de leur réseau de pédiatrie pour estimer le nombre de patients potentiellement incluables lors de l'essai : il est ainsi possible de mesurer l'activité de recrutement : quel pourcentage des patients atteints par la pathologie a-t-on recruté par rapport à la prévision ?

1.4.4 Conclusion

La table 1.2 présente un récapitulatif de toutes les approches évoquées précédemment. Pour chacune d'entre-elles est disponible le pays dans lequel est exploité le produit ou prototype, l'année de création, le nom de l'outil si disponible, les spécialités concernées et les références. Des cases à cocher renseignent les données utilisées par les logiciels : site internet lorsque les

Table 1.2: Outils d'aide au recrutement ou à la décision pour les essais cliniques.

	Pays	Année	Nom	Spécialités	Données					Réf.	Pré-screening	
					web	saisie	TAL	EMR	autre	particularités	patients	essais
Conception	USA	1999		cancérologie					x	physician data query database			[95]	
	USA	2001		cancérologie					x	suite logicielle de cancérologie	x		[45]	
	-	2010											[9]	
	USA	2010	MindTrial	psychiatrie	x						x	x	[73]	
Pré-screening	USA	2010		cancérologie		x	x	x			x		[87]	
	USA	2008	ASAP	toutes			x	x			x		[33, 65]	
	USA	2009		maladies orphelines	x						x	x	[93]	
	USA	2003		drépanocytose			x	x		résultat laboratoire + note d'entrée aux urgences	x		[18, 111]	
	USA	2001		cancer du sein	x	x						x	[4, 80]	
	FR (Paris)	2001	Oncodoc	cancer du sein		x						x	[96]	
	FR (Rennes)	2010	ASTEC	cancérologie		x						x	[12]	
	USA	2009		essais diabète					x	datawarehouse université de Columbia : diagnostics, actes et traitements avec terminologie ad-hoc (MED)	x		[105, 113]	
	USA	2005		essai diabète					x	EpicCare		x	[37]	
	UK	2010	VOTES	essais de cohorte									[100]	
	UK	2010	SARMÀ	essai diabète				x				x	[107]	
	USA	2009	trialX	toutes	x				x	Google health, Microsoft Health vault		x	[85, 86]	
Eval	USA	2007		pédiatrie					x	Réseau de recherche en pédiatrie (PECARN)			[104]	

données proviennent de la saisie des patients sur un site internet ; TAL pour les données extraites de courriers et documents écrits à l'aide de méthodes de traitement automatique du langage ; EMR pour les données provenant de dossiers patients informatisés disponibles dans la majorité des hôpitaux ; autre pour les données issues de logiciels propriétaires ou spécifiques à certains hôpitaux. Enfin, le périmètre d'utilisation de chaque outil est également documenté : la première colonne renseigne si l'outil traite de la *conception* (concerne les outils utilisables pour la conception de l'essai clinique) ; pré-screening et *évaluation* concerne les outils utilisés pour évaluer le bon déroulement de l'essai. La dernière colonne précise le type de pré-screening : *patients* pour les outils permettant d'extraire une liste de patients pour un essai clinique et *essais* pour les outils capables d'assigner des essais clinique à un patient présent en consultation.
Nous pouvons remarquer que la majorité des approches sont prévues pour l'aide au pré-screening, et quelques approches sont prévues pour l'aide à la conception de l'essai. Une seule approche a été proposée pour l'évaluation de l'essai, mais avec peu de modifications une approche permettant le pré-screening peut permettre également d'effectuer une évaluation. La majorité des approches proviennent des États-unis, on note cependant deux approches françaises et deux approches anglaises. La majorité des approches concernent la cancérologie, ou un essai clinique particulier. On trouve cependant deux approches généralisables à de nombreuses spécialités : *trialX* et *ASAP*. On note également que les données utilisées pour le pré-screening sont souvent issues de logiciels propriétaires ou d'approches ad-hoc, ce qui les rend difficilement exploitables sur d'autres systèmes d'information. En conclusion, il n'existe pas d'approche proposant à la fois de l'aide à la conception de l'essai, au pré-screening et à l'évaluation, généralisable à toutes les spécialités médicales et aux données médicales présentes dans les hôpitaux français. Dans la suite, nous détaillons comment Opcyclin s'insère dans l'offre existante.

1.4.5 Positionnement du logiciel Opcyclin

Aider à la décision : toutes pathologies confondues L'approche que nous proposons cible les essais cliniques toutes spécialités médicales confondues. Les approches déjà réalisées ou en cours en France sont OncoDoc et ASTEC, qui ciblent des domaines plus spécialisés comme la cancérologie ou le cancer du sein [12, 96]. Nous ciblons une approche générique, permettant de gérer tous types d'essais cliniques. Elle sera testée sur différentes spécialités : *MICI (maladies inflammatoires chroniques intestinales)*, *pathologies vasculaires cérébrales* (par l'intermédiaire du réseau *Strokavenir*) et *maladie d'Alzheimer*.

Système expert...sans expert À l'instar d'ASTEC qui utilise des connaissances métier pour étendre la recherche des patients, Opcyclin utilisera un ensemble de connaissances métier sous la forme de règles afin d'étendre les critères de recherche d'un patient. À la différence des systèmes experts classiques, les règles ne seront pas définies par un expert mais générées automatiquement à l'aide de techniques d'extraction de connaissances et de fouille de données. Notre approche est ainsi extensible à tous les domaines médicaux et n'a pas le besoin de solliciter un expert pour concevoir le système expert pour chaque nouveau critère d'inclusion.

Utiliser des données majoritairement disponibles Notre système de détection des patients éligibles pour les essais cliniques utilisera les données présentes dans la majorité des centres hospitaliers Français. Ces données limiteront le temps nécessaire au recrutement en diminuant ou supprimant une saisie manuelle du profil du patient. Les données d'activité PMSI sont candidates et permettent de gérer une partie des critères d'éligibilité. Le CHU de Bordeaux a utilisé ses données PMSI sur l'essai clinique Plasmacard pour vérifier l'exhaustivité des recrutements [16]. Les premiers résultats montrent une faisabilité de l'utilisation de ces données d'activité pour le recrutement de patients. Une autre étude est en cours au CHU de Dijon : le projet PMSI-AVC, pour mesurer la qualité métrologique du PMSI dans le cadre du repérage des AVC. Cependant d'autres études ont isolé des cas où les données PMSI ne seront pas suffisantes [74]. Cela se produit lorsque le critère d'éligibilité n'est pas exprimable avec le PMSI. Les critères suivants ne peuvent pas s'exprimer avec PMSI : « FEVG [1] $\geq 40\%$ », « anémie ($\geq$ 12,5 g/dl Hb $\leq$ 9 g/dl) », « absence de traitement anticoagulant », « score NIHSS < 10 ». Pour couvrir un maximum de critères d'éligibilité, nous proposons d'utiliser également des données hospitalières complémentaires aux données PMSI, comme les résultats de laboratoire. Une autre source de données candidate est le logiciel libre *OpenClinica* qui permet de gérer des CRF électroniques (Clinical Research Form). De plus, *Alicante* a élaboré conjointement avec le CHRU de Lille le logiciel *SIGREC* qui recense tous les essais cliniques promus en France. Ce qui laisse la possibilité de prévoir un interfaçage avec la base structurée SIGREC pour obtenir une liste d'essais candidats.
Le système informatique hospitalier ne sera pas toujours suffisant pour recruter dans certains essais. Certaines pathologies aiguës, comme l'AVC, doivent être traitées en temps réel, bien avant que le dossier informatisé du patient ne soit complété. Dans ce cas une saisie manuelle sera nécessaire, mais nous souhaitons la rendre moins contraignante à l'aide de formulaires personnalisés. Ceux-ci demanderont uniquement les informations nécessaires en priorisant les éléments à saisir selon le profil du patient et les essais en cours.

Améliorer l'ergonomie et les performances à l'aide de techniques d'apprentissage Il nous semble opportun dans notre approche d'allier les technologies issues de la fouille de données et de la recherche opérationnelle à des techniques d'apprentissage automatique. Ces techniques nous permettraient de recueillir plus de profils patients candidats qu'avec un système expert classique. Il ne s'agit pas uniquement de modéliser toute la connaissance médicale dans un système expert, mais d'utiliser de nouvelles relations et connaissances proposées par la fouille de données pour le recrutement. Si les utilisateurs le souhaitent, il sera cependant possible d'enrichir le système avec des connaissances médicales ou un feed-back afin d'améliorer encore les résultats. La volumétrie des données utilisées est importante : les données PMSI contenant à elles seules jusqu'à 31 789 informations potentielles par patient. Des méthodes d'optimisation adaptées aux problèmes de grande dimension seront utilisées : les *méta-heuristiques*. Par ses possibilités de déduction de règles d'association, la fouille de données offre la possibilité de détecter des

1. La fraction d'éjection du ventricule gauche (FEVG) est une estimation de la contraction du cœur, elle est utilisée notamment pour mesurer l'insuffisance cardiaque

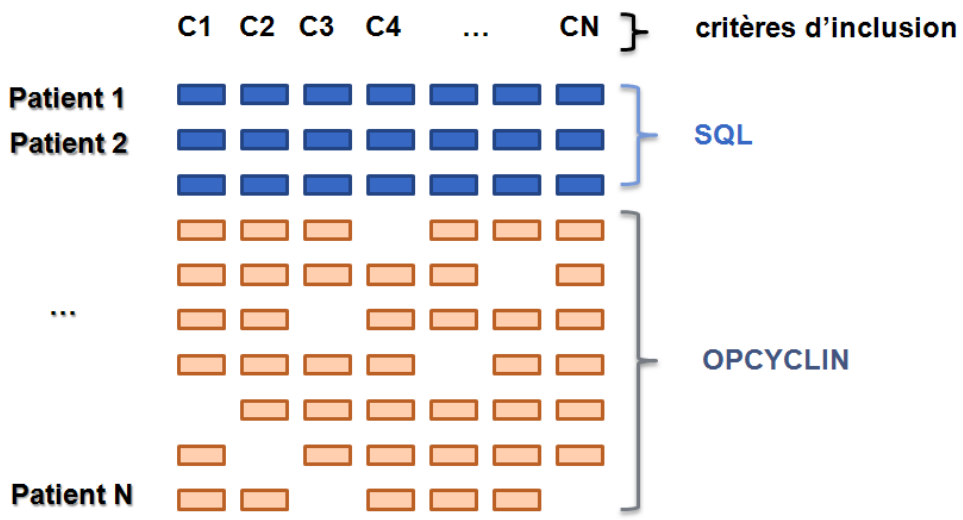


Figure 1.5: Différences SQL et Opcyclin - Populations ciblées

profils patients supplémentaires qui sortent à première vue des critères d'inclusion. Les erreurs de saisie ou les données manquantes du PMSI sont ainsi prises en charge, et la conception des critères d'inclusion est facilitée, contrairement à ce qui est permis par une approche classique par moteur de requêtes. Si l'on revient sur la section 1.3.1.1, une partie des 21% de patients diabétiques dont le diabète est non renseigné dans le PMSI pourraient ainsi être identifiés. Un moteur de requêtes comme SQL, lui, les raterait. Opcyclin permettra de rechercher parmi les patients dont un moins un des critères est mal renseigné, ce qui élargit l'espace de recherche des patients. Nous illustrons cette problématique dans la figure 1.5 ; elle montre des dossiers patients complets (en haut) et des dossiers pour lesquels des critères d'inclusion ne sont pas renseignés (en dessous). Une approche SQL sera uniquement capable de rechercher des patients dont tous les critères sont renseignés, ce qui finalement arrive peu souvent par rapport au nombre de dossiers incomplets où au moins un critère d'inclusion n'est pas renseigné. Opcyclin cible ces dossiers, en étant capable d'évaluer si une information manquante était importante ou non. Il fournira pour chaque critère non renseigné un score indiquant la probabilité de sa présence. Le système d'apprentissage permettra d'affiner au fil des utilisations les nouvelles règles d'association utilisées par la fouille de données pour la sélection de patients.
Plusieurs *scénarii* sont possibles et seront présentés dans la suite du document.

1.5 Conception d'un outil d'aide à la décision

Cette section détaille le périmètre d'utilisation du logiciel Opcyclin. Plusieurs *scenarii* sont étudiés, de la conception de l'essai clinique et de ses critères d'inclusion à l'analyse *a posteriori* de l'efficacité des recrutements, en passant par le recrutement de patients durant la validité de l'essai clinique.

1.5.1 A priori - aider à la conception de l'essai

Lors de la conception d'un essai clinique, l'élaboration des critères d'inclusion est une étape capitale car elle détermine directement la population à recruter. Il n'est pas rare de devoir arrêter un essai clinique faute de patients correspondant aux critères d'inclusion qui ont été définis, car même avec de bonnes capacités de recrutement il n'est pas possible de recruter plus de patients que ceux qui se présenteront.
Il est difficile de prédire combien de patients vont correspondre à un critère donné ; le médecin va les estimer selon sa pratique ou ses connaissances, ce qui amène parfois à changer des critères trop restrictifs en cours d'essai faute de recrutements. En fournissant des indicateurs au concepteur de l'essai, il va lui être possible de concevoir au plus juste les critères d'inclusion. Il est possible d'agir de plusieurs manières :

Outils statistiques De manière similaire à l'outil réalisé par Taylor *et al.* pour les AVC [104] il est possible d'estimer pour chaque critère d'inclusion le nombre de patients qui pourront être recrutés dans le futur, à l'aide de statistiques.

Méthodes de fouille de données À l'aide de la fouille de données, les critères d'inclusion peuvent être étendus. Si le praticien souhaite exclure les patients avec une *insuffisance hépatique (code K72)*, il est possible de lui proposer d'inclure / exclure également des codes similaires découverts par la fouille de données, comme *l'insuffisance hépatique alcoolique (code K70.4)*.

Tableaux de bord Des graphiques comme la courbe d'inclusion - qui permet de voir l'évolution du nombre de patients inclus au cours du temps - permettent de détecter un problème sur les critères d'inclusion. D'autres graphiques peuvent permettre de visualiser les patients exclus, pour essayer d'identifier les critères en cause.

D'autre part, ces outils pourront également être utiles aux centres investigateurs pour effectuer des études de faisabilité, afin d'éviter d'accepter un essai clinique pour lequel ils auront des difficultés à trouver des patients.

1.5.2 Au fil de l'eau - outil de pré-screening

Lors de l'affectation d'un patient à des essais cliniques, deux *scenarii* sont envisageables. Le premier s'applique en consultation, lorsque l'on souhaite proposer une liste d'essais cliniques adaptés à un patient. L'autre *scenario* est plus proche de l'approche du TEC, qui consiste à identifier parmi une liste de patients (par exemple les patients récemment arrivés dans le service), ceux qui pourraient être affectés à un essai clinique particulier.

1.5.2.1 Quels essais pour ce patient ?

Ce scénario est fondé sur le cas d'un patient en consultation. Le médecin souhaite vérifier si des essais cliniques peuvent lui convenir mais ne connaît pas forcément tous les essais en cours, ni leurs critères d'inclusion. La tâche est actuellement trop complexe pour être effectuée manuellement : certains services comprennent jusqu'à 15 essais en cours et chacun d'entre eux peut contenir une trentaine de critères d'inclusion. Ce cas d'utilisation est également semblable à celui du TEC qui vérifie les critères d'inclusion sur les patients récemment entrés dans le service.

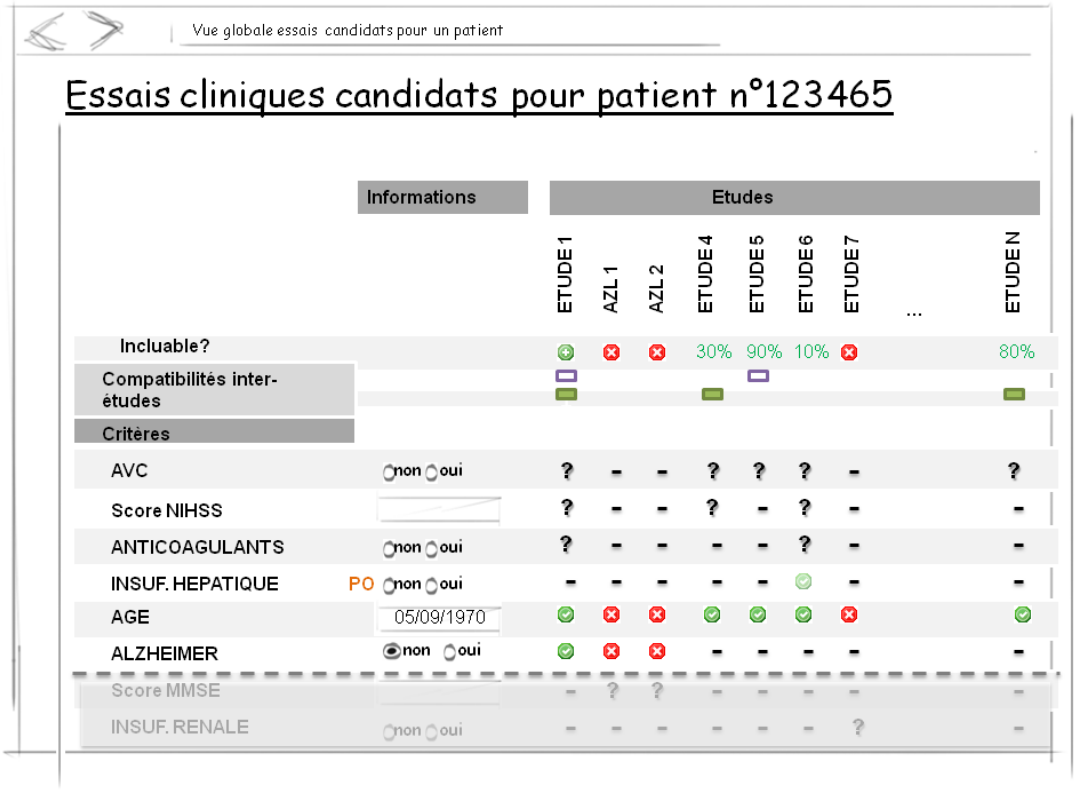


Figure 1.6: Identification d'essais candidats pour un patient en consultation - Maquette

Priorisation de la saisie Nous avons réalisé une maquette du logiciel pour ce cas d'utilisation, présentée dans la figure 1.6 propose une interface pour aider à l'identification d'essais candidats pour un patient. On y retrouve dans la colonne de gauche la liste des critères d'inclusion et de non inclusion disponibles pour toutes les études en cours. La colonne suivante indique les valeurs retrouvées dans le dossier du patient ou à remplir, pour chacun des critères. Enfin, les autres colonnes indiquent pour chaque étude si chacun des critères est validé ou non. Afin de minimiser la saisie demandée à l'utilisateur, les critères sont présentés du plus discriminant

au moins discriminant : le critère permettant de filtrer le plus d'études est présenté en premier (dans l'illustration, le critère « AVC »).Le critère *Insuffisance Hépatique* est proposé en dernier à la saisie, car il n'est utile qu'à une seule étude. Dans ce cas d'utilisation, le critère *Score MMSE* n'a pas à être renseigné car il concerne les études *AZL1* et *AZL2* qui ont été exclues de la recherche : le patient ne respecte pas le critère *Alzheimer*. Le long de la saisie, les indicateurs d'inclusion seront automatiquement mis à jour et les critères pourront être réordonnancés pour la saisie.

Pré-remplissage du dossier patient Pour chaque étude, un résumé est effectué, indiquant si le patient est éligible ou non. Dès qu'un critère d'inclusion n'est pas respecté, le patient est exclus de l'étude. Les lignes *compabilités inter-études* permettent de connaître les études auxquelles le patient peut participer simultanément s'il le souhaite - ce qui est parfois possible dans le cas d'études observationnelles. Au fil de la saisie les filtres seront de plus en plus restrictifs, permettant d'isoler la ou les études candidates.

1.5.2.2 Quels patients pour cet essai ?

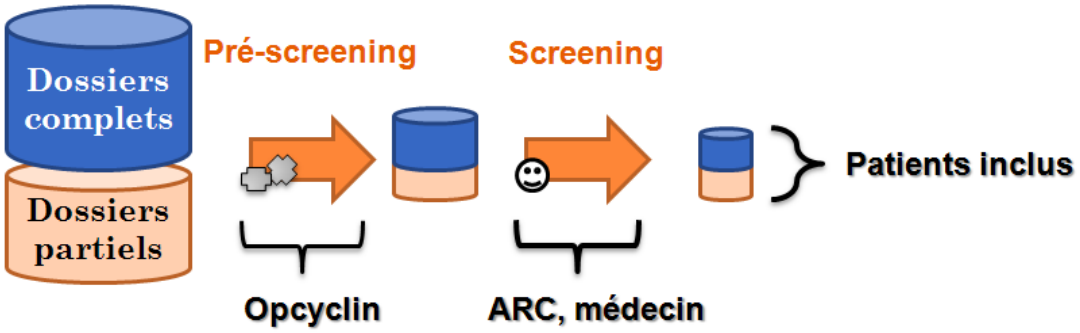


Figure 1.7: Processus de pré-screening et screening - Illustration du processus

Pré-screening Le TEC ou le médecin investigateur souhaitent augmenter les recrutements sur un essai. Actuellement les patients sont identifiés via le screening sur les entrées dans le service ou une partie des consultations. Parcourir manuellement l'ensemble des dossiers patients demanderait trop de temps, ce qui est cependant possible avec un outil informatique : ce dernier peut filtrer les dossiers, y compris ceux dont il manque des informations concernant un ou plusieurs critères d'inclusion, pour ne retenir qu'une liste - de taille raisonnable - de patients potentiels sur laquelle le TEC ou le médecin investigateur pourront effectuer le screening. Nous avons résumé ce fonctionnement dans la figure 1.7.

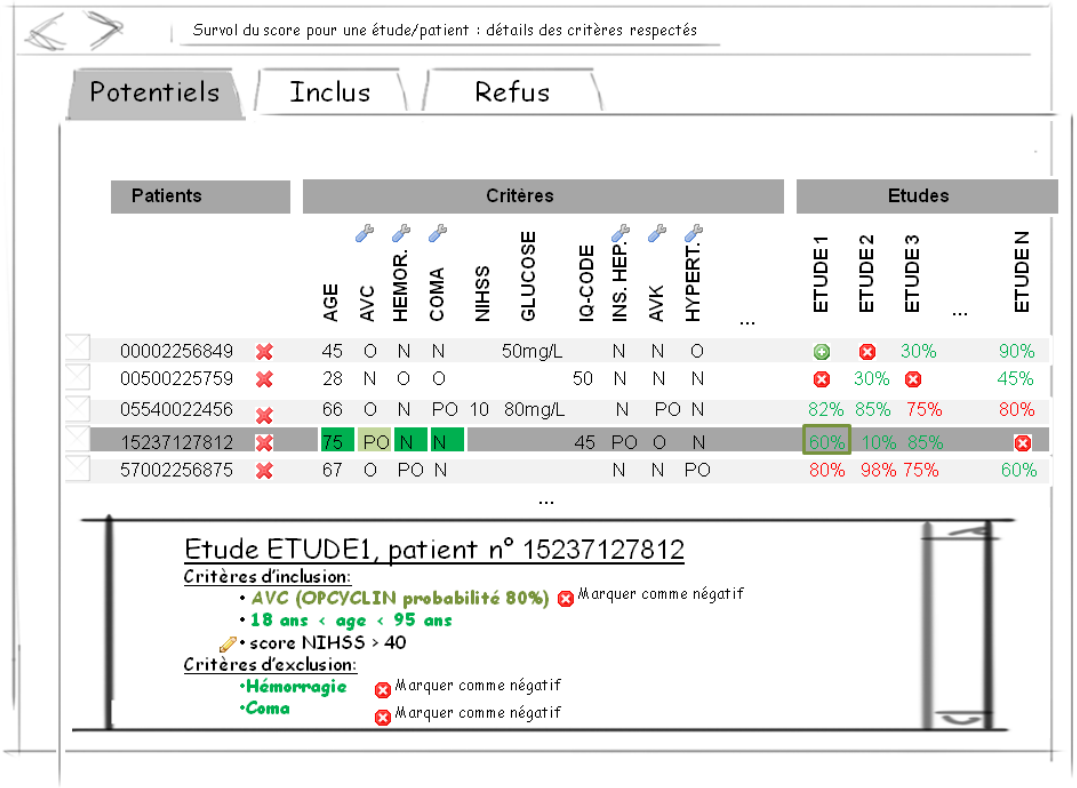


Figure 1.8: Identification de patients potentiels pour un ou des essais - Maquette

Dans la figure 1.8, nous proposons la maquette d'un écran de pré-screening pour une liste de patients. Chaque ligne correspond à un patient candidat pour un essai clinique, à ses informations concernant les critères d'inclusion, et à un score de correspondance pour chaque étude en cours dans le service. L'utilisateur peut ainsi voir pour chaque étude si des patients ont le profil désiré, comme c'est par exemple le cas pour le patient 00002256849 et l'étude 1. Il est possible d'avoir le détail des critères respectés ou non pour un patient, comme c'est le cas par exemple pour le patient de la quatrième ligne, avec la possibilité également de renseigner des critères (dans l'exemple, le score NIHSS). On observe pour ce patient que le critère « AVC » est indiqué comme POtentiellement présent (PO). Lorsqu'il manque des informations, le système fournit un pourcentage représentant la quantité d'information présentes pour évaluer l'éligibilité, parmi les informations requises. Dans un premier temps, l'application utilisera uniquement les informations PMSI et la saisie manuelle car il s'agit des informations présentes dans tous les hôpitaux. La généricité de l'outil permettra ensuite d'importer d'autres données, comme des comptes-rendus et résultats d'examens au format HL7 ou des traitements médicamenteux, selon les données disponibles dans l'hôpital.

Gestion des informations manquantes Les techniques de fouille de données permettront d'augmenter le score d'un patient lorsqu'elles détecteront une information susceptible de répondre au critère d'inclusion. Par exemple, un patient pour lequel le critère "AIC" n'est pas renseigné mais pour lequel on retrouve un code similaire (ex : *I64* au lieu de *I62*), ou un acte médical qui est souvent associé à ce critère, on pourra augmenter le score du patient. Dans ce cas, il peut s'agir d'une erreur de saisie ou d'une information manquante. En effet, dans la section 1.1 nous avons observé que 10 à 20% des diagnostics PMSI peuvent être non renseignés : sur l'ensemble des critères d'inclusion il risque de manquer des informations. Le remplissage de l'information via la fouille de données permet d'éviter de rater des patients, contrairement à un outil de requêtes classique, comme SQL.

Système d'apprentissage Afin d'améliorer les résultats au fil de l'utilisation, l'outil disposera d'un système d'apprentissage. L'utilisateur aura plusieurs moyens d'apprendre au système : lorsqu'il inclus ou refuse un patient de la liste de pré-screening il donne naturellement une information sur le respect des critères pour le patient concerné ; il peut également signaler au système lorsqu'un critère d'inclusion est mal renseigné par la fouille de données.

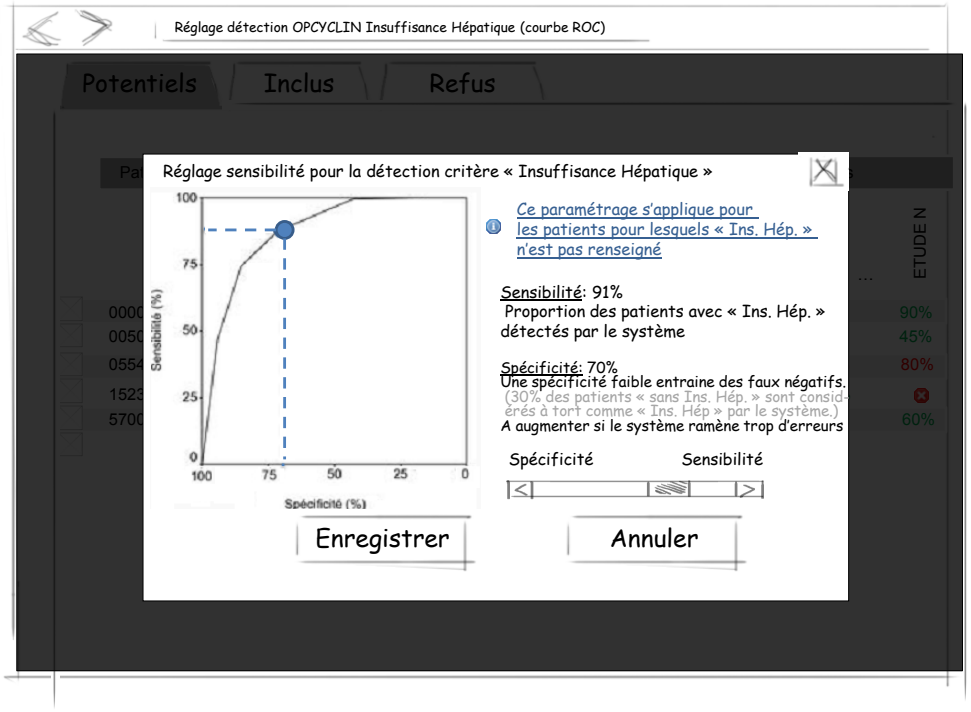


Figure 1.9: Réglage de la sensibilité du système - courbe ROC - Maquette

Calibrage de l'extension des critères d'inclusion Pour chaque critère d'inclusion éligible à la fouille de données, un outil de paramétrage sera disponible. Il permet pour chaque critère de

régler la sensibilité du système : doit-on élargir le critère d'inclusion - ce qui permet d'identifier plus de patients éligibles, au risque d'avoir beaucoup de faux positifs - ou le restreindre, au risque de rater des patients ? Si l'utilisateur dispose de peu de temps, il pourra ainsi restreindre le nombre de patients renvoyés par le système. Il pourra effectuer ce réglage par exemple à l'aide d'une courbe ROC. La maquette de la figure 1.9 illustre ce cas d'utilisation pour le critère de non inclusion "*insuffisance hépatique*", elle permet de régler le compromis entre la *Sensibilité* (*True Positives Rate*) – qui représente le nombre de patient détectés – et la *Spécificité* qui représente la proportion de patients sans le critère qui sont bien classés (*True Negatives Rate*).

1.5.3 A posteriori - évaluer la qualité du recrutement

Une fois l'essai terminé, il est intéressant d'évaluer la qualité du recrutement. Cela peut permettre de mettre en évidence des problèmes lors du recrutement (par exemple : manque de personnel) ou encore d'améliorer le recrutement lors des prochains essais. Deux approches sont proposées.

Proportion de patients recrutés parmi les potentiels d'inclusion L'outil mis en place pour affecter des patients à un essai (précédemment dans la section 1.5.2.2) peut également être réutilisé dans le cadre de l'évaluation du recrutement d'un essai clinique. En effet, il peut fournir la liste des patients qui auraient pu participer à l'essai clinique. Il est ensuite possible de comparer cette liste avec les patients réellement recrutés et analyser la population identifiée par le logiciel mais non recrutée.

Analyse des patients exclus Il est également intéressant d'analyser les patients exclus afin d'identifier les critères d'inclusion responsables. Afficher pour chaque critère le nombre d'exclus permettrait de mettre en évidence un critère qui bloque les recrutements.

1.6 Modélisation du pré-screening en un problème de classification

Le pré-screening peut se modéliser en un problème de classification : pour chaque critère d'inclusion et de non inclusion, le logiciel doit déterminer si un individu (le patient) appartient à une classe (respecte le critère d'inclusion). Le problème de classification identifié sera détaillé dans le chapitre 2. Prenons comme exemple un essai clinique fictif évaluant une technique pour réduire le risque de récidive de l'AVC. Les critères d'éligibilité peuvent être les suivants :

critères d'inclusion

- AVC de moins de 6 mois
- $18 \leq$ âge ≤ 65

critères de non inclusion

- insuffisance rénale
- insuffisance hépatique
- traitement anticoagulant

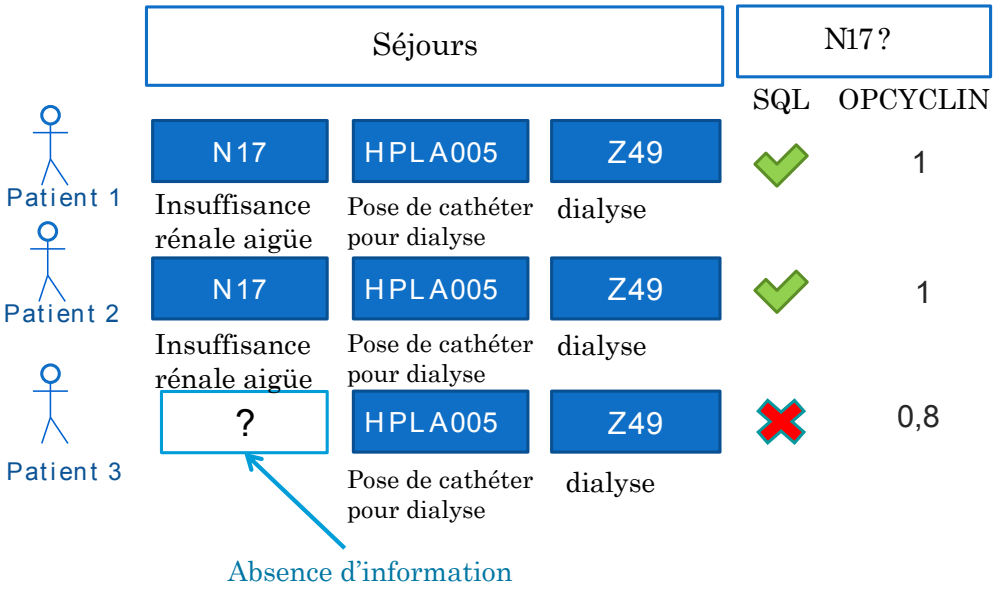


Figure 1.10: Principe Opcyclin - Identification de patients avec le code N17

Dans la suite nous présenterons le raisonnement à effectuer pour vérifier l'éligibilité à *un* des critères d'éligibilité. Ce raisonnement devra ensuite être répété pour tous les critères d'éligibilité afin de pouvoir déterminer si le patient peut participer à l'essai clinique. Si l'on se concentre sur le critère d'exclusion *« insuffisance rénale »* , codé *N17*, il s'agit de trouver des caractéristiques du patient qui vont permettre de déterminer si la présence du diagnostic « insuffisance rénale » est possible ou non. Dans la maquette d'écran de la figure 1.8, cela correspond à remplacer certains "N" par "PO". Dans la figure 1.10, nous présentons une illustration du raisonnement : on cherche à remplir le critère « insuffisance rénale » pour le patient 3. On dispose du dossier médical de plusieurs patients. Les informations des patients 1 et 2 peuvent donc être utilisées : on observe que les patients 1 et 2 ont une insuffisance rénale aiguë, suivie d'une pose de cathéter pour préparer les futures dialyses, et enfin une séance de dialyse. Dans le logiciel, ces patients seraient notés "O". Le patient 3 ne possède pas l'information sur l'insuffisance rénale : il peut s'agir d'une erreur dans le dossier ou d'un codage différent (par exemple *I12 : insuffisance rénale avec hypertension*). Des techniques de fouille de données comme l'extraction de connaissances peuvent permettre au logiciel Opcyclin de détecter ce patient. Il peut ainsi apprendre qu'une dialyse est très souvent accompagnée d'un diagnostic antérieur d'insuffisance rénale en regardant les patients 1 et 2, et utiliser l'information pour déduire l'insuffisance rénale chez le patient 3, qui sera alors annoté d'un score traduisant la probabilité (0,8 donné ici à titre d'exemple). Ces approches permettent de détecter des patients, comme le patient 3, qui n'auraient pas été détectés avec un moteur de requêtes classique comme SQL. Plusieurs de celles-ci seront présentées en détails par la suite.

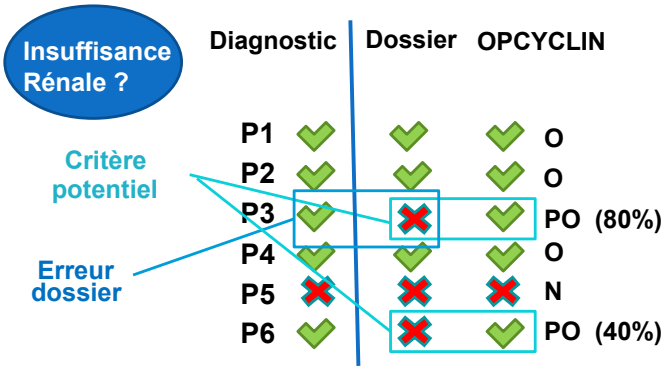


Figure 1.11: Opcyclin - détermination de la présence possible d'un diagnostic - Identification de patients avec suspicion d'insuffisance rénale

Détection de la présence possible d'un diagnostic L'extraction de connaissances va permettre d'extraire des règles qui donneront des causes et conséquences possibles de l'insuffisance rénale. Ces règles permettront d'estimer si le patient a un risque de présenter une insuffisance rénale ou non. Lorsque l'extraction détecte une présence possible du diagnostic en opposition avec l'information présente dans le dossier, le dossier de ce patient est marqué "PO" (pour "potentiel opcyclin"). Dans la figure 1.11 nous illustrons pour un critère de non inclusion donné – ici insuffisance rénale – les différents cas de figure qui peuvent se produire : diagnostic chez le patient, présence de l'information dans le dossier et détection par Opcyclin. L'objectif d'Opcyclin est de parvenir à détecter les omissions dans le dossier médical, comme cela se produit dans l'illustration sur le patient $P3$. Lorsque Opcyclin tente d'identifier ces omissions, il marque les patients suspects avec un "PO", associé à un score de confiance, dans l'illustration un score de 80% pour le patient $P3$ et un score de 40% pour le patient $P6$. Deux cas peuvent se produire. Dans le premier cas, il s'agit bien d'une erreur dans le dossier, comme c'est le cas dans l'exemple pour le patient $P3$, Opcyclin a donc efficacement détecté une information manquante. Dans le second cas, Opcyclin est trop permissif et propose le patient $P6$ à tort ; il est cependant associé à un faible score (40%), ce qui peut permettre d'exclure le dossier car il y a finalement peu de chance que le diagnostic soit effectivement présent. D'où l'importance du système de calibrage présenté précédemment (figure 1.9) qui doit permettre à l'utilisateur de limiter ou élargir les capacités de détection de l'outil selon ses besoins.

1.6.1 Verrous scientifiques

Nous avons vu précédemment l'objectif d'Opcyclin, qui est de détecter les critères d'inclusion et de non-inclusion lorsque les informations sont manquantes, à l'aide de techniques d'apprentissage et de fouille de données. Des données hospitalières devront être utilisées pour la phase d'apprentissage, cependant ces données présentent quelques particularités qui vont influencer les algorithmes utilisés et poser des verrous scientifiques.

1.6.1.1 Volumétrie et explosion combinatoire

À titre d'exemple, nous utilisons les données d'un hôpital entier pour effectuer le recrutement, ce qui représente sur 10 mois de temps 35 350 patients et 54 678 séjours. Les dossiers PMSI de ces patients contiennent 10 256 attributs différents : il s'agit des diagnostics et actes réalisés au moins une fois durant ces 10 mois sur au moins un patient. Pour chacun de ces attributs et patients la valeur peut être "oui" ou "non renseigné". Si l'on essaie de recenser toutes les règles de classification de 1 à 5 termes qui peuvent être créées à partir de ces attributs on se trouve face à une explosion combinatoire : il est possible d'en générer $9,44 \times 10^{17}$. Un algorithme naïf qui examinerait chacune de ces règles en 10ms aurait besoin d'environ 3 milliards d'années pour toutes les tester. Ce nombre peut encore augmenter si l'on souhaite complexifier la forme des règles que l'on souhaite obtenir : en ajoutant par exemple la temporalité, ou en ajoutant d'autres données patients comme les prescriptions médicamenteuses.
Cette contrainte rend appropriées les techniques de recherche opérationnelle et d'optimisation. Dans une revue de la littérature, Corne *et al.* expliquent comment ces techniques peuvent être utilisées pour la fouille de données [29]. Par ailleurs, de nombreux travaux utilisent l'optimisation combinatoire pour résoudre des problèmes de fouilles de données, comme l'atteste la revue de littérature de Srinivasan et Ramkrishnan [98]

1.6.1.2 Asymétrie dans les données

Un des verrous posé par les données médicales est l'asymétrie de l'information à prédire. Il se pose lorsque l'information a prédire est sous-représentée dans les données étudiées. Par exemple, l'*AVC* est une pathologie fréquente d'un point de vue médical, mais statistiquement elle représente 1% des séjours hospitaliers. Au mieux, une pathologie apparaîtra sur 20% des séjours (*Hypertension*). Cette asymétrie est problématique pour la plupart des algorithmes de fouille de données, qui utilisent des métriques qui ne sont pas adaptées à cette répartition et se focalisent sur la prédiction de l'information majoritaire (ici "non AVC"). Ces métriques et les raisons de leur inadaptation aux données asymétriques seront étudiées dans le Chapitre 2, ainsi que les solutions proposées pour gérer l'asymétrie.

1.6.1.3 Incertitude sur les données

Les diagnostics médicaux entrés dans le dossier patient peuvent prendre 3 valeurs : "oui" si le diagnostic est présent dans le dossier, ce qui signifie qu'il a été détecté par le médecin ; "non" si le médecin a effectué un test diagnostic et a exclu ce diagnostic (par exemple, absence de diabète car le taux de glucose dans le sang est correct) ; la valeur "inconnu" signifie qu'aucun test diagnostic n'a été effectué, le patient peut avoir la pathologie sans avoir été diagnostiqué, ou ne pas encore en montrer les signes. Certains systèmes d'informations comme le PMSI ne font pas la distinction entre "non" et "inconnu", ce qui rend hasardeux l'utilisation de cette information ; dans ce cas des techniques de classification partielle sont à privilégier, pour se focaliser uniquement sur l'information qui est présente – qui elle est plus sûre.
D'autre part les données médicales informatisées sont sujettes à des erreurs de saisie. Comme

nous l'avons vu précédemment les données PMSI représentent des données de facturation et bien souvent seuls les actes et diagnostics ayant influé sur la facturation vont être renseignés. D'autre part certains systèmes de codage proposent différents codes pour une même information. Ainsi, la classification ATC des médicaments (Anatomical Therapeutic Chemical Classification System) [1] permet de coder une même molécule de différentes manières, selon son indication thérapeutique. L'aspirine peut ainsi être codée de 4 manières différentes selon qu'elle est utilisée pour son effet anticoagulant, antipyrétique, anti-inflammatoire ou antalgique. Certains logiciels n'offrent pas cette souplesse de codage ; lors de la conversion vers la classification ATC un seul code par défaut va être utilisé, code qui peut changer selon le logiciel...

1.6.2 Caractéristiques souhaitées du logiciel

En plus des particularités des données, le choix des techniques à mettre en œuvre va être lié à des caractéristiques qui peuvent améliorer le logiciel et son succès.

Amélioration des résultats par apprentissage Comme présenté précédemment, il est souhaitable que le logiciel soit capable d'améliorer les résultats au cours du temps, en utilisant le retour des utilisateurs pour apprendre de ses erreurs.

Personnalisation et paramétrage Le logiciel présentera plus d'attractivité aux utilisateurs s'ils peuvent adapter leurs résultats en fonction des besoins. L'utilisation d'un seuil (via la courbe ROC par exemple) permet à l'utilisateur d'ajuster l'exhaustivité et le taux de faux-positifs selon le temps qu'il peut accorder au screening et les besoins de recrutement sur ses essais. Cependant le paramétrage doit rester simple et ne pas être nécessaire systématiquement : des travaux ont montré que bien souvent les utilisateurs n'ont pas les connaissances suffisantes en fouilles de données pour paramétrer les algorithmes [117].

Boîte blanche Les systèmes de type "boîte blanche" - pour lesquels il est possible d'expliquer la classification donnée par le logiciel - sont à privilégier car ils facilitent la maintenance.

1.7 Conclusion

Dans ce chapitre, nous avons tout d'abord présenté le contexte métier des essais cliniques, et identifié les différentes difficultés qui pouvaient être présentes lors de leur déroulement. Nous avons ensuite analysé les données médicales disponibles dans les hôpitaux, en particulier les données PMSI, afin d'envisager leur utilisation dans un logiciel d'aide à la décision. Après une revue des différentes approches qui ont été proposées dans la littérature en tant qu'aide à la décision pour les essais cliniques, nous avons détaillé l'approche proposée avec le logiciel Opcyclin, ainsi que ce qui la démarque des approches présentées précédemment. Ensuite, le périmètre d'utilisation d'Opcyclin a été détaillé, ainsi que les différents verrous scientifiques

1. Disponible sur le site de l'OMS : http ://www.whocc.no/atc_ddd_index/

posés notamment par les particularités des données médicales. Le prochain chapitre détaille les technologies qui vont être utilisées pour mettre en place le moteur d'Opcyclin, à savoir les techniques de classification fondées sur les règles et les méta-heuristiques. Il donne également quelques détails sur le protocole qui sera utilisé pour évaluer les performances du moteur dans les chapitres suivants : jeux de données et méthodes de comparaison statistique d'algorithmes.

Chapitre 2

Classification à base de règles et méthodes de comparaison statistique

Ce chapitre est dédié à la classification à base de règles et donnera également des concepts fondamentaux qui seront ensuite utilisés pour élaborer les protocoles d'évaluation (instances, méthodes statistiques de comparaison, etc.). Tout d'abord, une introduction à la classification détaillera ce qu'est la tâche de classification et les manières d'évaluer sa qualité. Nous détaillerons ensuite les difficultés posées par les données asymétriques, ce qui représente un des verrous posés par les données médicales. Ensuite, un état de l'art sera réalisé, à la fois des méthodes classiques de classification mais également des méthodes fondées sur les méta-heuristiques. Une rapide introduction sur les méta-heuristiques sera également inclue. Enfin, nous définirons les composants communs aux protocoles présents dans les prochains chapitres, à savoir les jeux de données qui seront utilisés pour évaluer les performances de classification ainsi que les méthodes de comparaison statistiques d'algorithmes qui sont recommandées en fouille de données.

2.1 Introduction à la classification

Avant de présenter les différents algorithmes d'extraction de connaissances il est important de présenter quelques notions sur la classification. Cette section détaillera ce qu'est la tâche de classification et comment évaluer ses performances.

2.1.1 Généralités sur la classification

La classification vise à prédire une *classe* (par exemple "'Grippe?"' oui/non) à partir d'une liste d'*observations* passées, pour lesquelles on dispose d'une liste d'*attributs* et dont on connaît la classe. On identifie deux natures d'attributs : qualitatif ou quantitatif. Un attribut quantitatif

est de forme numérique – par exemple l'âge, le poids,... Il peut être utilisé pour calculer divers indicateurs tels qu'une somme, une moyenne, un écart-type ; ce qui n'est pas possible avec un attribut qualitatif – par exemple la couleur. Certains attributs qualitatifs peuvent néanmoins présenter une notion d'ordre, on parle alors d'attribut ordinal (par exemple, la satisfaction d'un client).
Dans la figure 2.1, nous avons représenté un exemple de tâche de classification. On souhaite apprendre sur des anciens patients – ici les patients 1 à 6 – comment déterminer s'ils ont la grippe (la classe) à partir de leurs symptômes (les attributs). Il sera ensuite possible de déterminer si un nouveau patient a la grippe (par exemple sur le patientX). Pour y parvenir, les algorithmes de classification génèrent un modèle – un classifieur – qui effectue un ensemble de tests sur les attributs pour déterminer la classe. Les classifieurs sont ainsi formés de conjonctions ou de disjonctions de termes (tests sur attributs), ce qui leur permet de prendre de nombreuses formes : règles, arbres, ensembles de règles,... Pour la suite, nous nous intéressons aux classifieurs sous la forme d'une règle, mais le raisonnement reste applicable aux autres formes (arbres, ensembles de règles,...)

		ATTRIBUTS			CLASSE
	Nom	Fièvre	Douleur Musculaire	Toux	Céphalée	Grippe?
OBSERVATIONS (connues)	Patient1	O	O	O		O
	Patient2	O		O		
	Patient3				O	
	Patient4	O	O	O	O	O
	Patient5	O	O		O	O
	Patient6		O			
	PatientX	O	O			?

Figure 2.1: Illustration de la tâche de classification -

Voici un exemple de règle :

$$Fièvre \Rightarrow Grippe$$

Une règle se compose d'une *condition* et d'une *prédiction*, elle est de la forme *SI condition ALORS prédiction* que dans la suite nous noterons $C \Rightarrow P$. P se produit lorsque C est présent. Dans l'exemple, cela signifie qu'un patient a la *grippe* s'il a de la *fièvre*. Nous verrons plus loin comment évaluer si cette règle – très simple – est efficace ou non. C et P sont composés d'un ensemble de *termes*, qui représentent des tests à effectuer sur les caractéristiques (*attributs*) des observations. Les termes peuvent être de plusieurs formes : [98]

Partition binaire Ex : $température \geq 39°$ Celcius

Intervalle Le test consiste à vérifier un attribut quantitatif ou ordinal appartient à un intervalle de valeurs. Ex : $18 \leq âge \leq 85$

Valeur L'attribut doit avoir une valeur particulière. Par exemple : $Toux = oui$

Inégalité L'attribut ne doit pas être égal à une valeur particulière. Par exemple : $Toux \neq oui$

Sous-ensemble L'attribut qualitatif doit respecter une des valeurs proposées. Par exemple : $Département \in \{Nord, Pas\ de\ Calais\}$

Règles de classification Les règles de classification sont un cas particulier des règles d'associations dans lesquelles la prédiction P est fixe : on cherche toujours à prédire la même chose (ex : grippe ou absence de grippe). Lorsque l'on cherche à prédire partiellement une prédiction (ex : juste les cas positifs de grippe), il s'agit de *classification partielle*.

2.1.2 Évaluation de la qualité d'une règle

Lorsque l'on utilise des règles, il est important d'arriver à déterminer si elles sont vraies ou non, utiles ou encore représentatives : est-ce qu'elles se vérifient souvent ? Dans la littérature beaucoup de mesures existent pour évaluer la qualité d'une règle. Beaucoup d'entre-elles sont fondées sur la notion de matrice de confusion. Les plus courantes et pertinentes pour notre problématique seront présentées dans la suite. Pour plus d'exhaustivité, il est possible de se référer à l'article d'Ohsaki *et al.* qui ont étudié les différentes mesures utilisées dans les applications médicales [81]. Geng et Hamilton quant à eux ont réalisé une revue des mesures utilisées en fouille de données [44]. Khabzaoui *et al.* ont effectué une analyse statistique sur plusieurs critères [66]. Les scores présentés dans la suite sont issus d'une sélection parmi ceux présentés dans ces articles.

2.1.2.1 Matrice de confusion

La majorité des mesures de qualité d'une règle sont élaborées à partir de la matrice de confusion suivante :

Table 2.1: Matrice de confusion

	P	$\overline{P}$	
C	TP	FP	
$\overline{C}$	FN	TN	
			N

La table 2.1 représente la matrice de confusion pour une règle $C \rightarrow P$ donnée, évaluée sur un ensemble d'*observations* (ou *instances*) : il s'agit de comparer les résultats donnés par la règle à ce que l'on peut observer dans la population.

Vrais positifs (TP : True Positives) ($|C \wedge P|$) nombre d'observations qui vérifient la condition et la prédiction. Dans l'exemple, les patients qui ont à la fois de la fièvre et la grippe.

Faux positifs (FP : False Positives) ($|C \wedge \overline{P}|$) nombre d'observations qui vérifient la condition mais n'ont pas la prédiction. Dans l'exemple, les patients qui ont de la fièvre, mais n'ont pas la grippe ; il s'agit des patients que la règle classe à tort comme ayant la grippe.

Faux négatifs (FN : False Negatives) ($|\overline{C} \wedge P|$) nombre d'observations qui ont la prédiction sans vérifier la condition. Dans l'exemple, les patients qui ont la grippe alors qu'ils n'ont pas de fièvre. Il s'agit des malades de la grippe que la règle n'a pas réussi à identifier.

Vrais négatifs (TN : True Negatives) ($|\overline{C} \wedge \overline{P}|$) nombre d'observations sans la prédiction ni la condition . Il s'agit des patients qui n'ont pas la grippe, ni de fièvre.

C (TP + FP) nombre d'observations respectant la condition de la règle. Dans l'exemple précédent, nombre de patients ayant de la fièvre.

P (TP + FN) nombre d'observations respectant la prédiction de la règle (sans regarder la condition). Dans l'exemple précédent, nombre de patients ayant la grippe.

N nombre d'observations. Dans l'exemple précédent, la population de patients sur laquelle on évalue la règle.

2.1.2.2 Récapitulatif

La table 2.2 fournit un récapitulatif des mesures les plus fréquemment utilisées, avec leur formule de calcul et les noms utilisés dans la littérature. Leur interprétation sera expliquée dans la suite. Enfin, dans un chapitre ultérieur une analyse de corrélations entre ces différentes mesures sera réalisée.

2.1.2.3 Mesures communément utilisées en médecine : sensibilité, spécificité, VPN, VPP, prévalence

Certaines de ces mesures sont fréquemment utilisées en médecine pour évaluer les performances d'un test diagnostic. Nendaz et Perrier ont écrit un article illustratif de leur utilisation [79]. Prenons comme illustration la règle suivante :

$$Fièvre \Rightarrow Grippe$$

Sensibilité Permet de représenter la proportion de patients ayant P qui sont détectés par la règle. Si la règle présentée précédemment a une sensibilité de 0.9, cela signifie que 90% des patients avec la grippe ont également eu de la fièvre. Une valeur faible signifie que l'on rate des patients avec P, ici 10% des grippes. Il s'agit des patients qui ont eu la grippe sans avoir de fièvre.

Spécificité Représente la proportion de patients sains ($\overline{P}$) qui sont correctement classés par la règle. Une valeur faible signifie que certains patients sains sont diagnostiqués à tort (faux positifs). Si la règle présentée précédemment a une *spécificité* de 0.7, cela signifie que 70% des patients sans grippe n'ont pas eu de fièvre. Cependant il y a des patients qui ont de la fièvre sans avoir la grippe (par exemple les patients qui ont une infection ou un virus différent de la grippe). Il s'agit des 30% restant.

Nom	Formules		Autres noms
Confiance	$P(P\|C)$	$\frac{TP}{TP+FP}$	VPP [1], précision, consistance
Conviction	-	$\frac{(TP+FP)\times(FP+TN)}{N\times FP}$	
Cosinus	-	$\frac{TP}{\sqrt{(TP+FP)\times(TP+FN)}}$	
Exactitude	$P(C\wedge P)+P(\overline{C}\wedge\overline{P})$	$\frac{TP+TN}{N}$	*Accuracy rate*
F-Mesure	-	$\frac{(1+\beta^2)\times Confiance\times Sensibilité}{\beta^2\times Sensibilité+Confiance}$	
La Place	-	$\frac{TP+1}{TP+FP+2}$	
Lift	$\frac{P(P\|C)}{P(P)}$	$\frac{N\times TP}{(TP+FP)\times(TP+FN)}$	Intérêt
Moyenne géométrique de l'exactitude		$\sqrt{\frac{TP}{TP+FN}\frac{TN}{FP+TN}}$	*geometric mean of accuracy rate*
Négatifs découverts	$P(\overline{C}\wedge\overline{P})$	$\frac{TN}{N}$	*Uncovered negatives*
Piatetsky-Shapiro	-	$\frac{TP}{N}-\frac{TP+FP}{N}\times\frac{TP+FN}{N}$	
Prévalence	P(C)	$\frac{TP+FP}{N}$	
Sensibilité	$P(C\|P)$	$\frac{TP}{(TP+FN)}$	taux de vrais positifs, complétude, rappel
Spécificité	$P(\overline{C}\|\overline{P})$	$\frac{TN}{TN+FP}$	taux de vrais négatifs
Support	$P(C\wedge P)$	$\frac{TP}{N}$	Généralité, couverture
Surprise	-	$\frac{TP-FP}{TN+FP}$	
Valeur prédictive négative	$P(\overline{P}\|\overline{C})$	$\frac{TN}{TN+FN}$	VPN

TABLE 2.2: Quelques mesures de qualité d'une règle

Confiance - Valeur prédictive positive (VPP) Donne la probabilité que l'individu ait réellement P si C est présent ; permet d'évaluer si la règle est fiable. Dans l'exemple, si la règle a une *confiance* de 0.2, cela signifie que si un patient a la fièvre, il y a également la grippe dans 20% des cas. Cette mesure permet rapidement de se rendre compte que notre règle n'est pas très efficace pour prédire la grippe : la fièvre est présente dans beaucoup d'autres cas (infections, autres virus, etc.). Cette mesure est utilisée dans une majorité des algorithmes de classification. Utilisée seule la *confiance* est sujette au *sur-apprentissage* : les règles obtenues sont trop spécifiques aux données d'apprentissage et ne fonctionnent pas sur de nouvelles données.

Valeur prédictive négative (VPN) Donne la probabilité qu'un individu n'ait pas P si C n'est pas présent. Si la règle a une VPN de 0.90, cela signifie qu'un patient sans fièvre a 90% de chances de ne pas avoir la grippe.

Prévalence Cette mesure est utilisée en médecine et plus particulièrement en épidémiologie pour mesurer la proportion de la population affectée par une maladie. Ici elle donne la proportion de patients vérifiant C parmi toute la population. Il s'agit de l'ensemble des patients ayant eu de la fièvre.

Relations entre sensibilité et spécificité Les mesures de *sensibilité* et *spécificité* sont en concurrence : lorsque l'on essaye d'augmenter la *sensibilité* d'une règle la *spécificité* diminue. La courbe ROC permet dans certains cas d'ajuster la règle selon la sensibilité et la spécificité nécessaires [88]. Elle sera étudiée plus précisément dans le chapitre concernant l'aide à la décision.

2.1.2.4 Autres mesures utilisées en extraction de connaissances

Support Proportion des observations qui vérifient la règle (C et P). Ce score sera faible pour les règles rares. Dans l'exemple, il s'agit des observations avec de la fièvre et la grippe. Sur les données PMSI, selon les pathologies le support sera compris entre 0,5% et 10% car seule une petite proportion des consultations concernera en réalité une pathologie donnée. Le *support* est utilisé dans la plupart des algorithmes de classification.

Exactitude Évalue la proportion d'observations correctement classées par la règle : observations vérifiant C et P ou $\overline{C}$ et $\overline{P}$. Cette mesure est utilisé dans certains algorithmes de fouille de données, mais est très sensible à la répartition des données : les règles rares faussent le score. Sur une population de 1000 observations dont 100 avec la grippe et les performances de règles suivantes : TP = 10, FP = 40, FN = 90 et TN = 860, l'*exactitude* est de 0.87 qui semble à première vue un bon score. Cependant cette règle a une *confiance* de 20% et une *sensibilité* de 10% : elle détecte uniquement 10% des grippes et lorsque c'est le cas, elle se trompe dans 80% des cas [41].

Moyenne géométrique de l'exactitude L'*exactitude* (accuracy) pose problème lorsque les données sont mal réparties. Kubat *et al.* ont proposé d'utiliser plutôt la moyenne géométrique de l'exactitude positive et de l'exactitude négative, ce qui permet de dégrader le

score lorsque les cas négatifs ou positifs sont mal classés [70].

Négatifs découverts Proportion des observations $\overline{P}$ correctement classées par la règle. Dans l'exemple, permet de regarder la proportion de patients sans grippe correctement classés par la règle parmi tous les patients. Cette mesure est intéressante pour la recherche de règles médicales car elle donne de l'importance à la classification correcte de la population saine [81].

Cosinus Dérivé de la corrélation statistique, surtout intéressante pour les règles de faible support et fort intérêt.

Lift Mesure le degré de dépendance entre C et P. Si le lift vaut 1, C et P sont indépendants. Contrairement à l'*exactitude*, le *lift* n'est pas sensible à la répartition des données.

Conviction Similaire au *lift* mais tient compte de l'absence la prédiction ($\overline{P}$).

Surprise Permet de rechercher des règles étonnantes.

F-Mesure Permet de trouver des compromis entre la *confiance* et la *sensibilité* en faisant varier le coefficient β selon ce que l'on souhaite privilégier : *confiance* ou *sensibilité*. En général on utilise $\beta = 1$ [25].

2.1.2.5 Validation croisée

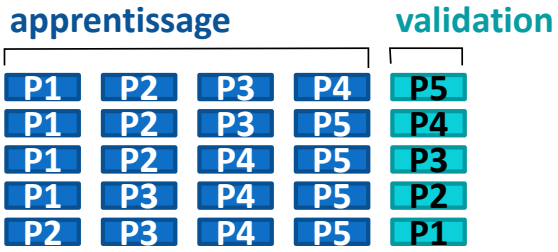


Figure 2.2: Illustration de la validation croisée (k=5) -

Une pratique courante en fouille de données consiste à évaluer la qualité des classifieurs sur des données qui n'ont pas été utilisées durant la phase d'apprentissage. Cela permet de détecter le *sur-apprentissage* qui se produit lorsqu'un classifieur se spécialise trop sur les données d'apprentissage et n'est plus capable d'effectuer des prédictions sur de nouvelles données. Cela peut être mis en place par la *validation croisée* : les données sont découpées en k partitions, l'apprentissage s'effectue sur un jeu d'apprentissage contenant les partitions $p_1...p_{k-1}$ et est évalué sur la partition p_k. Ensuite, l'apprentissage s'effectue sur les partitions $p_2...p_k$) et est évalué sur la partition p_1. Cette manipulation est répétée k fois (validation *5-fold*) pour permettre à chacune des partitions p_i d'être utilisée comme ensemble de validation. Dans la figure 2.2 nous proposons une illustration de la validation croisée avec $k = 5$; l'algorithme d'apprentissage sera donc exécuté 5 fois : une fois sur chacun des découpages proposés. Puis, une moyenne est effectuée sur l'ensemble des exécutions afin de lisser les résultats, certaines partitions pouvant être plus complexes que d'autres.

2.1.3 Classification sur données asymétriques

La tâche de classification peut rapidement s'avérer difficile lorsque la classe que l'on cherche à prédire – que nous appelons C_{pred} – est présente uniquement sur une faible proportion des observations. Sur les 35 350 patients évoqués dans le paragraphe précédent, 484 ont le diagnostic *AVC*, soient 2% de l'ensemble. En médecine, une prévalence de 2% caractérise une pathologie très courante. En fouille de données et statistiques elle est insuffisante pour permettre le raisonnement et revient à chercher une aiguille dans une botte de foin. Dans la matrice de confusion, cela se traduit par $|P| << |\overline{P}|$. Dans la suite, nous appelons degré d'asymétrie (d_{asy}) le pourcentage des observations du jeu de données qui ont la classe C_{pred} :

$$d_{asy} = \frac{|C_{pred}|}{N}.$$

Une valeur de $d_{asy} < 0.5$ indique une sous-représentation de la classe C_{pred} par rapport à son absence (classe $\overline{C_{pred}}$), tandis qu'une valeur de $d_{asy} > 0.5$ indique une sur-représentation de la classe C_{pred}. Nous nous intéressons ici au cas où $d_{asy} < 0.5$, ce qui signifie que la classe C_{pred} est sous-représentée.

Voyons maintenant l'impact de cette asymétrie sur les algorithmes de classification. Les algorithmes fondés sur l'*exactitude* ou le *taux d'erreur* donneront de mauvais résultats car ils estimeront faire peu d'erreur en « omettant » de classer les observations intéressantes. Comme nous l'avions vu précédemment avec l'*exactitude*, une règle qui agit de cette manière obtiendra tout de même une bonne *exactitude* car elle effectue en fait peu d'erreurs par rapport au nombre d'observations qu'elle aura réussi à bien classer (les négatifs) : si seulement 1% des observations sont intéressantes ($d_{asy} = 0.01$), elle considérera qu'elle en a bien classé 99%.
Certains travaux suggèrent que les difficultés liés à l'asymétrie ne sont pas liés uniquement au problème de répartition mais au *small disjunct problem* [63]. Dans le *small disjunct problem*, la classe à prédire est en fait explicable uniquement par des combinaisons d'attributs qui couvrent peu d'exemples. De nombreuses solutions ont été proposées pour résoudre le problème d'asymétrie, à la fois en transformant les données ou en agissant directement sur la conception de l'algorithme, ou sur le *small disjunct problem* sous-jacent. He *et al.* ont réalisé une revue de littérature qui donne la plupart des pistes empruntées pour résoudre le problème [50], qui sont exposées dans la suite.

2.1.3.1 Solutions fondées sur des transformations du jeu de données

Une première solution consiste à ré-échantillonner les jeux de données asymétriques, afin de les rendre artificiellement symétriques. Plusieurs approches sont disponibles.

Sous-échantillonnage Il s'agit de supprimer des observations négatives jusqu'à obtenir une répartition correcte. Le choix des observations à supprimer est assez complexe à réaliser car il y a un risque de supprimer des observations importantes pour la classification. Des approches ont été proposées pour conserver les observations qui ressemblent aux observations ayant la classe à prédire.

Sur-échantillonnage Il consiste à dupliquer les observations positives jusqu'à obtenir une répartition correcte. Cette technique a de fortes tendances au sur-apprentissage car une règle concernant uniquement une observation peut obtenir de très bons scores car elle sera vérifiée sur une forte proportion d'observations (qui sont en réalité ses clones).

Génération d'observations supplémentaires Il s'agit toujours de sur-échantillonnage, mais cette fois l'idée est de créer de nouvelles observations au lieu de dupliquer les existantes, afin de limiter le sur-apprentissage. L'algorithme SMOTE propose ainsi de générer des observations à partir d'observations existantes [26].

Un autre ensemble de techniques agit plus simplement sur la matrice de confusion, en donnant un poids plus important à P au lieu de $\overline{P}$. Ce qui permet ensuite de lancer des algorithmes qui ne sont pas forcément adaptés aux données asymétriques. Il s'agit des méthodes *cost-sensitive* qui seront détaillées dans la prochaine section. Ces techniques ont montré de meilleurs résultats que les techniques de ré-échantillonnage [50].

2.1.3.2 Solutions à la conception de l'algorithme

Les travaux de Jo et Japkowicz montrent qu'il est plus efficace d'adapter les algorithmes pour la recherche de règles rares (*small disjunct problem*), plutôt que de corriger un jeu de données déséquilibrées [63]. Cela peut par exemple passer par l'utilisation de mesures de qualité qui ne sont pas sensibles à l'asymétrie des données, comme la *moyenne géométrique de l'exactitude* et la *F-mesure* qui sont préconisées pour évaluer les performances en cas de répartition déséquilibrée. Une autre approche est de se concentrer sur les observations concernées par le *small disjunct problem*, comme proposé dans l'algorithme *DT_GA* qui sera abordé plus en détail dans la suite.

2.1.4 Algorithmes de classification

De nombreuses méthodes de génération de classifieurs ont été proposées ; dans cette partie nous nous intéresserons uniquement aux approches générant des modèles de type « boîte blanche » qui permettent à l'utilisateur de pouvoir interpréter le résultat. Nous aborderons tout d'abord les approches constructives, suivies par les méthodes utilisant des pondérations. Les approches fondées sur les méta-heuristiques seront détaillées dans la section suivante.

2.1.4.1 Les méthodes constructives

Les méthodes constructives partent d'un classifieur vide et vont ajouter à chaque itération le terme qui améliore le mieux le classifieur courant. L'algorithme le plus connu de ce type est *C4.5*, qui construit un arbre de décision en choisissant à chaque itération l'attribut qui maximise le gain d'information [90]. L'algorithme est de type "diviser pour régner" ; à chaque itération il évalue le gain d'information uniquement sur les données de la branche de l'arbre sur laquelle il se trouve. Ainsi, les observations pour lesquelles il existe déjà un arbre de décision sont supprimées de l'ensemble d'apprentissage. L'algorithme *CART (Classification And Regression Tree)* [15] suit le même principe mais maximise l'indice de Gini.

L'algorithme *IREP* [42] suit un peu le même principe, toujours sur une approche « diviser pour régner », mais traite un ensemble de règles au lieu d'un arbre de décision. Le jeu de données d'apprentissage est découpé en deux parties : deux tiers du jeu sont réservés à la génération de règles, le dernier tiers sert à élaguer les règles pour limiter le sur-apprentissage. Lors d'une itération, *IREP* construit une règle terme par terme, en choisissant le terme qui optimise le gain d'information. Elle est ensuite élaguée et ajoutée à l'ensemble de règles résultat. En fin d'itération, les observations classées par la nouvelle règle sont supprimées du jeu d'apprentissage. L'algorithme *RIPPER (Repeated Incremental Pruning to Produce Error Reduction)* est une amélioration de l'algorithme *IREP* [28]. Il apporte quelques améliorations à *IREP*, comme un opérateur d'élagage plus efficace et une nouvelle condition d'arrêt. Il ajoute une nouvelle phase à la fin d'*IREP* afin d'optimiser l'ensemble des règles obtenues : pour chaque règle de l'ensemble, il génère deux règles alternatives et gardera la règle qui parmi les 3 donne le meilleur résultat au sein de l'ensemble de règles.

2.1.4.2 Les méthodes de pondération

Certaines techniques améliorent les algorithmes proposés précédemment (le plus souvent *C4.5*) en pondérant les observations ou une sous-partie des observations – les faux positifs ou les faux négatifs – à l'aide de poids. Cela permet de focaliser l'algorithme sur les observations mal classées ; ou d'indiquer si un type d'erreur est plus important qu'un autre selon le contexte : est-il est plus gênant de laisser passer un faux positif ou un faux négatif ? Deux approches sont possibles, qui pondèrent soit directement les observations selon leur importance ; soit agissent sur la matrice de confusion.

Sur les observations Les techniques de *boosting* permettent d'améliorer les performances d'un classifieur simple (par exemple *C4.5*), en l'exécutant itérativement en le focalisant à chaque fois sur les observations mal classées. À chaque itération, on lance le classifieur courant c_i et on calcule son erreur de classification. Les pondérations des observations sont mises à jour : celles des observations mal classées sont augmentées. À la prochaine itération, le classifieur c_{i+1} va donc donner plus d'importance aux observations mal classées par c_i, c_{i-1},... car celles-ci auront un poids plus important. Lorsque le nombre d'itérations n est atteint, les classifieurs c_1,...,c_n obtenus à chaque itération sont combinés dans un classifieur $c_{final} = \alpha_1.c_1 + \alpha_2.c_2 + \ldots + \alpha_n.c_n$ où α_i est un poids donné à chaque classifieur c_i en fonction de ses performances. La classe d'une observation sera obtenue par vote des différents classifieurs, selon leur poids α_i. L'algorithme de *boosting* de référence est *AdaBoost*.

Sur la matrice de confusion Les méthodes *cost-sensitive* interviennent sur la matrice de confusion, en ajustant les poids (sur P et $\overline{P}$) pour obtenir artificiellement des données symétriques. Cela permet d'exécuter dans un contexte de données asymétriques des algorithmes qui ne sont pas prévus pour, par exemple s'ils sont fondés sur l'*exactitude*[1]. Différents algorithmes

1. accuracy

cost-sensitive ont été proposés, on retrouve notamment l'algorithme *C4.5-CS* qui est la version *cost-sensitive* de *C4.5*, mais également des algorithmes de *boosting* comme *AdaC2* [101] qui est une version *cost-sensitive* de *AdaBoost*. On retrouve également l'algorithme *DataBoost-IM* [47] qui en plus de mettre des poids sur les observations, utilise les observations mal classées pour générer des observations artificielles.

2.2 Optimisation combinatoire pour la classification

Les méthodes proposées précédemment ne sont pas la seule manière de régler le problème de classification. La tâche de classification peut être vue comme un problème combinatoire : une règle est une combinaison de termes, rechercher des règles présente donc un problème combinatoire. Comme nous l'avions vu précédemment, sur les données hospitalières cela consisterait à rechercher des règles intéressantes parmi $9,44 \times 10^{17}$ possibles. Les méta-heuristiques et méthodes d'optimisation sont particulièrement indiquées pour aborder ce type de problèmes. Cette section commence par une brève introduction aux méta-heuristiques et à l'optimisation multi-objectif, et détaille ensuite l'algorithme DMLS qui sera utilisé dans la suite de ces travaux. Enfin, elle présente les différentes applications à la tâche de classification proposées dans la littérature.

2.2.1 Généralités sur l'optimisation combinatoire

Les méta-heuristiques sont des méthodes génériques qui peuvent être utilisées dans de nombreux problèmes combinatoires : planification, ordonnancement, tournées de véhicules, placement d'antennes de téléphonie,... ou – ce qui nous intéresse ici – fouille de données. Elles nécessitent une phase de modélisation où seront définis des composants spécifiques au problème traité : ce qu'est une solution et la manière d'évaluer sa qualité. Selon le problème traité, une solution peut ainsi être un planning, un itinéraire, une règle, etc. Et la qualité peut représenter un nombre de conflits, un nombre de kilomètres, la *Confiance* d'une prédiction, etc.
Là où un algorithme naïf prendrait quelques millions d'années à tester toutes les solutions possibles, les méta-heuristiques permettent d'obtenir dans un temps raisonnable une solution d'une bonne qualité, sans garantie d'optimalité. Sur le problème du voyageur du commerce, dont l'objectif est de trouver l'itinéraire de plus court chemin pour le voyage d'un commercial qui doit visiter K villes, une approche naïve est capable de gérer jusqu'à 8 villes. Les méta-heuristiques, elles, parviennent à gérer sur certaines instances jusqu'à plusieurs dizaines de milliers de villes. Cependant, elles ne garantissent pas que la solution obtenue correspond à l'optimum global, c'est-à-dire la meilleure solution possible. Dans la suite nous présenterons uniquement une introduction aux méta-heuristiques, pour plus de détails le lecteur est invité à se référer au livre d'El-Ghazali Talbi [102].

2.2.2 Présentation des méta-heuristiques

Nous détaillerons tout d'abord la recherche locale fondée sur une seule solution au travers de l'algorithme du *Hill Climbing*. De nombreuses approches ont été proposées sur le même principe que le *Hill Climbing*, à la différence qu'elles font évoluer un ensemble de solutions – une *population* de solutions – au lieu d'une unique solution, qui seront détaillées dans la deuxième partie.

2.2.2.1 Recherche locale

L'algorithme de recherche locale le plus connu est l'algorithme de *Hill Climbing*. Il nécessite la définition d'un voisinage, qui détermine pour chaque solution un ensemble de solutions similaires, appelées voisins. Sur le problème du voyageur de commerce, un voisin peut consister à inverser deux villes ; sur de la recherche de règle il peut s'agir d'une règle contenant 1 terme de moins ou de plus. L'algorithme de *Hill Climbing* démarre d'une solution initiale et va visiter tous ses voisins jusqu'à trouver une solution de meilleure qualité. Lorsqu'une solution de meilleure qualité est trouvée, on recommence la recherche à partir de cette solution. Le *Hill Climbing* s'arrête naturellement dès qu'aucun voisin améliorant n'est trouvé.

De nombreuses variantes existent : le *recuit simulé* qui s'inspire de la métallurgie et autorise au début de la recherche à choisir des solutions qui dégradent le résultat ; la *recherche tabou* s'autorise également à dégrader le résultat mais dispose d'une liste tabou qui définit une liste de mouvements interdits lors de la visite d'autres solutions.

2.2.3 Méthodes à base de population

Ces méthodes sont particulièrement utilisées en optimisation multi-objectif, qui doivent gérer un ensemble de solutions de compromis entre les différents objectifs. Les algorithmes génétiques ont été proposés par John Holland [51] et s'inspirent de la théorie de l'évolution de Darwin. Ils font évoluer une population de solutions, qui sont autorisées à se croiser entre-elles afin de générer une nouvelle génération. À chaque génération, seules les solutions les plus adaptées sont conservées. Ils nécessitent la définition d'opérateurs de croisement qui permettent de générer une ou plusieurs solutions à partir de deux solutions, et d'opérateurs de mutation qui effectuent une légère modification sur une solution. Ces algorithmes offrent une grande souplesse quant au choix des opérateurs et leur probabilité d'application, mais également au niveau des différentes stratégies de fonctionnement : choix des parents pour la reproduction ou sélection des solutions pour la prochaine génération. Cette souplesse amène néanmoins une difficulté de mise en place de l'algorithme de part les nombreux paramètres qui peuvent agir sur son comportement.
On retrouve également les méthodes co-évolutionnaires, qui sont une forme plus aboutie des algorithmes génétiques. Elles font cohabiter plusieurs populations qui sont en compétition les unes avec les autres, à l'instar des proies et prédateurs dans la nature qui doivent constamment s'adapter l'un à l'autre : si les proies deviennent plus rapides, seuls les prédateurs qui sont encore assez rapides pour les attraper vont survivre. Les avantages développés par une population vont ainsi forcer l'autre population à évoluer également.

Enfin, d'autres approches inspirées par la nature ont été proposées, qui s'inspirent des colonies de fourmis, colonies d'abeilles ou encore du système immunitaire.

2.2.4 Optimisation multi-objectif

Précédemment nous avons vu que de nombreuses mesures permettaient de mesurer la performance d'un classifieur. Dès lors que l'on souhaite optimiser plus d'une mesure à la fois (par exemple, en cherchant des classifieurs de bonne *confiance* et de bonne *sensibilité*) il devient nécessaire d'utiliser des méthodes d'optimisation combinatoire dédiées à la gestion de plusieurs objectifs.

2.2.4.1 Définitions et concepts

L'optimisation combinatoire multi-objectif est appliquée dans des problèmes et domaines variés, comme les tournées de véhicules, la planification ou la fouille de données. La plupart des applications possibles sont recensées dans le livre de Coello *et al.* [27]. À la différence de l'optimisation mono-objectif, elle s'attache à trouver des solutions qui améliorent simultanément plusieurs critères (par exemple, trouver des règles qui maximisent à la fois la *Confiance* et la *Sensibilité*, ou un itinéraire qui minimise à la fois le coût et le temps). Un tel problème s'exprime de la manière suivante : soit une solution x, plusieurs fonctions d'évaluation $f_1(x), f_2(x) \dots f_n(x)$ qui caractérisent la qualité de x et des objectifs (maximiser f_1, minimiser f_2, etc.), l'optimisation multi-objectif va chercher les solutions x qui remplissent les objectifs. Pour illustrer, sur le problème du voyageur de commerce x serait un itinéraire, $f_1(x), f_2(x)$ représenteraient le coût (carburant+péages) et le temps passé, et les objectifs seraient de minimiser $f_1(x)$ et $f_2(x)$. Différentes approches sont disponibles pour gérer les différents objectifs, elles feront l'objet d'une étude comparative dans le chapitre suivant. Une première approche consiste à agréger les objectifs au sein d'une seule fonction objectif $f(x) = w_1 \times f_1(x) + w_2 \times f_2(x) \cdots + w_n \times f_n(x)$, en pondérant les objectifs à l'aide de poids w_i, ce qui revient alors à résoudre le problème sous une forme mono-objective, en utilisant par exemple le *Hill Climbing*. Une autre approche est fondée sur la dominance, où les objectifs sont considérés séparément. L'algorithme doit donc gérer une population de solutions de compromis, et l'utilisateur aura à choisir une solution parmi plusieurs (par exemple : un itinéraire plus lent mais qui évite les péages, ou un itinéraire plus rapide mais plus cher). La relation de dominance la plus courante est la dominance Pareto : on considère qu'une solution S_1 domine une solution S_2 sur les fonctions objectif à maximiser $f_1, ..., f_n$ si :

$$\forall i \in 1..n, f_i(S_1) \geq f_i(S_2)$$

et

$$\exists j : f_j(S_1) > f_j(S_2).$$

L'ensemble des solutions non dominées est appelé *Front Pareto*. Il contient les solutions de meilleur compromis. Dans la figure 2.3 nous illustrons la notion de dominance Pareto. On cherche à maximiser les fonctions f_1 et f_2. Le point S_1 domine le point S_2 car $f_1(S_1) > f_1(S_2)$

et $f_2(S_1) > f_2(S_2)$. Le point S_4 est dominé à la fois par S_1 et S_3 car $f_1(S_1) > f_1(S_4)$ et $f_1(S_3) > f_1(S_4)$. Le front Pareto est composé des points S_1 et S_3 car ils ne sont dominés par aucun point. Dans cette thèse, nous nous intéresserons aux méthodes de recherche locale car

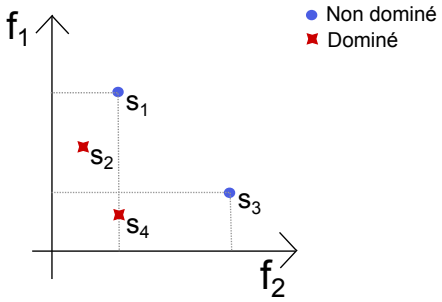


FIGURE 2.3: Dominance Pareto dans un contexte de maximisation

elles demandent moins de paramétrages que les algorithmes génétiques, tout en fournissant des résultats équivalents. Nous utiliseront l'algorithme *DMLS* – qui sera présenté dans la suite – qui a été prouvé aussi efficace que l'algorithme génétique de référence *NSGA-II* [75].

2.2.4.2 Dominance-based Multiobjective Local Search (DMLS)

DMLS (Dominance-based Multiobjective Local Search) est une adaptation des algorithmes de recherche locale mono-objectif – comme par exemple l'algorithme de *Hill Climbing* – à des problèmes multi-objectifs. De nombreuses approches fondées sur la dominance Pareto ont été proposées, incluant PAES (Pareto Archived Evolution Strategy) [67] ou encore PLS (Pareto Local Search) [83]. Liefooghe *et al.* ont proposé une unification de ces différentes méthodes au sein de l'algorithme DMLS.

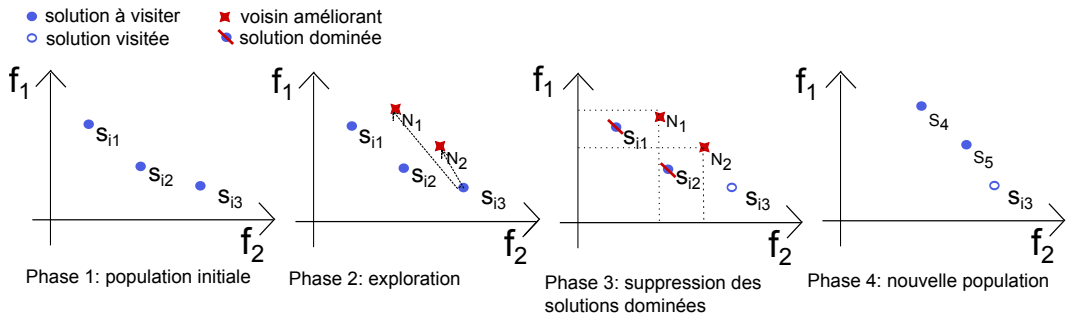


FIGURE 2.4: Illustration d'une itération de l'algorithme DMLS

Dans la figure 2.4 nous avons illustré une itération de l'algorithme DMLS. Il démarre d'une population initiale, composée d'un ensemble de solutions de compromis à améliorer (Phase 1). Lors de la première itération, toutes les solutions sont marquées comme « à visiter ». Selon

la stratégie d'exploration utilisée, DMLS va choisir une ou plusieurs solutions non visitées à explorer, et explorer leur voisinage. Dans l'illustration, la solution S_{i3} est choisie pour l'exploration et propose les voisins améliorants N_1 et N_2 (Phase 2). L'archive des solutions courantes est ensuite mise à jour avec les voisins découverts : dans l'illustration les solutions s_{i1} et s_{i2} sont supprimées car dominées par N_1 et N_2. N_1 et N_2 sont ajoutées à l'archive en tant que solutions à visiter (S_4 et S_5). Une nouvelle archive est obtenue, à partir de laquelle une nouvelle itération pourra être lancée. L'algorithme s'arrête naturellement lorsque l'archive ne contient plus de solutions à visiter.
Nous avons vu que plusieurs stratégies étaient possibles concernant le choix de la solution à explorer. C'est également le cas lors de l'exploration de voisinage. Voici les stratégies qui seront étudiées dans la suite de cette thèse.

Sélection de la (des) solution(s) à explorer (*1* or *)** Une stratégie simple consiste à sélectionner aléatoirement une solution à visiter (stratégie *1*), mais il est également possible de sélectionner l'ensemble des solutions à visiter (stratégie ***), ce qui est cependant plus coûteux.

Exploration du voisinage ($1_\succ$ or *)** Lors de la phase d'exploration de voisinage (Phase 2 sur la figure 2.4), il est possible d'arrêter l'exploration dès qu'un voisin améliorant est trouvé (stratégie $1_\succ$). Dans la figure 2.4, N_2 n'aurait donc pas été identifié lors de la première itération. L'autre stratégie étudiée consiste à visiter l'ensemble des voisins (stratégie ***), ce qui est plus coûteux en temps de calcul.

Dans la suite, chaque variante de DMLS sera nommée de la manière suivante : DMLS (*stratégie de sélection de la solution à explorer · stratégie d'exploration du voisinage*). Ainsi, DMLS ($1 \cdot 1_\succ$) correspond à un DMLS sélectionnant une solution à explorer à chaque itération (stratégie *1*), et arrêtant l'exploration du voisinage au premier voisin améliorant (stratégie $1_\succ$).

2.2.5 Application à la classification

De nombreux algorithmes fondés sur les méta-heuristiques ont été proposés pour la classification. Nous donnerons tout d'abord des informations sur les composants communs à plusieurs méta-heuristiques, comme la représentation d'une solution. Ensuite, nous présenterons les approches fondées sur des algorithmes génétiques, sur des algorithmes co-évolutionnaires et enfin sur des hybridations avec d'autres techniques, comme par exemple avec des méthodes d'apprentissage par renforcement. Finalement, nous donnons un récapitulatif de tous les algorithmes présentés.

2.2.5.1 Modélisation d'une solution

Dans la littérature, on retrouve plusieurs modélisations d'une solution d'un problème de classification. Une première solution est de modéliser une solution sous la forme d'un arbre de décision. Reynolds *et al.* ont par exemple utilisé cette représentation sur un problème de classification partielle [92], en laissant la possibilité d'avoir à la fois des conjonctions ou des disjonctions dans l'arbre, ce qui permet d'obtenir des arbres plus compacts que leur équivalent en

liste de règles. L'inconvénient réside dans les opérateurs de croisement qui sont plus complexes à mettre en place si l'on souhaite garder des arbres cohérents.
Du coté des modélisations à base de règles, on retrouve deux modélisations : Michigan et Pittsburgh. Dans la modélisation Michigan, chaque solution correspond à une règle, tandis que dans la modélisation Pittsburgh une solution correspond à un ensemble de règles. La modélisation Michigan est la plus répandue et la plus simple à mettre en place mais elle peut manquer des règles : une règle intéressante peut être perdue car une autre légèrement meilleure est trouvée. Ce problème ne se pose pas avec la modélisation Pittsburgh, qui gagne donc en qualité des prédictions. Cependant avec cette dernière il est nécessaire de définir comment les règles interagissent entre-elles, et comment la classe finale est déterminée (vote à la majorité, poids sur les règles, etc.). De plus, des incohérences peuvent apparaître entre les règles, Casillas *et al.* ont recensé les plus importantes [24]. À noter qu'il existe également des modélisations à base de règles floues, mais qui ne seront pas étudiées dans cette thèse. On peut ainsi noter les travaux d'Ishibuchi *et al.* [54], de Narukawa *et al.* [78] ou de Casillas *et al.* [24]. Les travaux de Wang *et al.* [109] couplent les méthodes multi-objectif à un système multi-agents.

On retrouve également des modélisations fondées sur les ensembles approximatifs. Il s'agit de trouver deux ensembles d'observations : un ensemble d'observations qui appartiennent de manière sûre à la classe à prédire, et un ensemble supérieur qui regroupe les observations qui appartiennent probablement à la classe à prédire. Slowinski *et al.* présentent en détail les ensembles approximatifs et leurs applications [97]. Des règles de décision peuvent être générées à partir des ensembles approximatifs obtenus. Une implémentation de ces méthodes est disponible dans le logiciel jMAF [14], qui utilise l'algorithme VC-DomLEM [13] pour extraire les règles de décision.

2.2.5.2 Algorithmes génétiques

L'algorithme DT_GA [23] utilise une représentation sous forme d'arbre au sein d'un algorithme génétique. Il est conçu pour gérer le problème des *small disjuncts*, ce qui doit le rendre particulièrement adapté aux jeux de données asymétriques. L'algorithme s'exécute en deux phases. Lors de la première phase, *C4.5* est utilisé pour générer des règles et identifier les *small disjuncts*. Lors de la deuxième phase, l'algorithme génétique (AG) va être lancé sur les observations concernées par les *small disjuncts* afin de générer des règles.
L'algorithme SIA [108] génère des règles une à une sur une méthode « diviser pour régner ». À partir d'une observation non classée une règle très spécifique est générée. Un AG est ensuite utilisé pour la généraliser au maximum. Toutes les observations concernées par cette règle sont ensuite supprimées et l'algorithme peut redémarrer à partir des observations restantes.
De nombreuses approches multi-objectives ont également été proposées, comme l'atteste une revue de littérature de Srinivasan *et al.* [98]. On y retrouve en plus des méthodes déjà citées, ou citées dans la suite, les approches de Iglesia *et al.* [91, 92] sur la classification partielle utilisant l'algorithme génétique de référence NSGA-II [31] et de Khabzaoui *et al.* sur la recherche de règles suivant 5 objectifs [76].

2.2.5.3 Algorithmes Co-évolutionnaires

L'algorithme *CORE* (COevolutionary Rule Extractor) [103] fait évoluer simultanément deux populations. L'une est composée de règles Michigan, l'autre est composée d'ensemble de règles (Pittsburgh). Ces deux populations sont à la fois en coopération et en concurrence. Un système de concurrence est mis en place grâce à un système de capture de jetons : chaque observation correspond à un jeton ; les classifieurs peuvent capturer les jetons pour lesquels ils donnent la bonne prédiction. Lorsque plusieurs classifieurs peuvent capturer un même jeton, on l'affecte à celui qui obtient le meilleur score sur la fonction objectif.
L'algorithme *OCEC* (Organisational Co-Evolutionary algorithm for Classification) [62] fait lui aussi évoluer simultanément deux populations. Cette fois-ci, une seule des populations contient des classifieurs. L'autre population est composée d'ensembles d'observations similaires (semblables à des clusters), qui seront utilisés par la première population pour bâtir des classifieurs efficaces. Cette fois, les deux populations doivent donc collaborer afin de fournir des règles utiles.

2.2.5.4 Programmation génétique

À la différence des autres approches présentées, Pappa et Freitas utilisent de la programmation génétique pour obtenir directement des algorithmes de classification [82]. Une solution ne représente donc plus une règle ou un ensemble de règles, mais un algorithme de classification qui devra être exécuté pour obtenir le classifieur final. C'est un algorithme génétique qui gère la population des algorithmes de classification, et va permettre de déterminer les caractéristiques les plus efficaces (objectifs, formes de classifieurs, etc.).

2.2.5.5 Colonies de fourmis

AntMiner [84] est un algorithme de recherche de règles fondé sur les colonies de fourmis. Les fourmis visitent les termes sur lesquels elles vont déposer des phéromones, avec une probabilité plus élevée de visiter des termes avec un bon gain d'information. L'évaluation du gain est différente de celle de *C4.5* : au lieu d'évaluer le gain d'un attribut comme c'est le cas dans *C4.5*, c'est le gain du couple attribut + valeur qui est évalué. AntMiner donne des modèles plus compacts et plus performants que les algorithmes auxquels il a été comparé.

2.2.5.6 Méthodes hybrides

Hybridation avec d'autres algorithmes L'algorithme *DT_Oblique* [21] s'inspire des *Support Vector Machine* (SVM) en cherchant des arbres « obliques » qui délimitent les classes en utilisant un hyperplan. Il utilise une hybridation entre un algorithme génétique et un *Hill Climbing*.
Une approche proposée par Yeon-Jin *et al.* [116] combine les réseaux de neurones et un algorithme génétique, les règles sont ensuite extraites du réseau de neurones.

Hybridation avec de l'apprentissage par renforcement Les techniques d'apprentissage par renforcement s'inspirent du mécanisme d'apprentissage présent chez les animaux et les êtres

humains. Elles sont par exemple utilisées dans certains robots et intelligences artificielles. Une famille de méthodes propose une hybridation entre l'apprentissage par renforcement, les algorithmes génétiques et la classification : les *Learning Classifier Systems* (LCS). Ils manipulent une population de classifieurs. À chaque itération, ils sont entraînés sur une observation : les classifieurs qui prédisent la bonne classe sont récompensés et ceux qui correspondent à l'observation courante sont sélectionnés par l'AG pour la reproduction. Ensuite, les classifieurs sont conservés selon leurs performances (récompenses reçues et *Exactitude*).
Les LCS les plus efficaces sont *XCS* [115], qui gère des solutions de type Michigan et *GAssist* [7] qui gère des solutions de type Pittsburgh. L'algorithme *BioHEL* succède à *GAssist* en lui ajoutant des possibilités de parallélisation [8].

2.2.6 Conclusion

La table 2.3 donne un récapitulatif des algorithmes présentés précédemment, à la fois fondés sur les méta-heuristiques ou non. Pour chacun d'entre eux est détaillé le type de classifieur généré, la ou les approches utilisées, les objectifs à optimiser et la référence. Les types de classifieurs générés sont assez variés, on retrouve à la fois des arbres de décision et des règles sous la forme Michigan ou Pittsburgh. Beaucoup d'algorithmes proposent une approche « diviser pour régner » ou une modélisation Michigan qui semblent *a priori* moins adaptées pour modéliser les facteurs de risque qui peuvent apparaître dans les données médicales, lorsque plusieurs règles peuvent expliquer la même pathologie (voir le prochain chapitre). On retrouve également beaucoup de méthodes fondées sur des algorithmes évolutionnaires. Les objectifs à optimiser sont assez variés. Plus d'un tiers des approches sont fondées sur le *Gain* et l'*Exactitude*, ce qui devrait poser des problèmes pour gérer les données asymétriques, mais pourra parfois être compensé par les techniques de *boosting* ou les méthodes *cost-sensitive*, voir par une adaptation de la fonction d'évaluation. Pour cette dernière solution, Pappa et Freitas calculent la différence entre l'*exactitude* obtenue avec l'*exactitude par défaut* [82] – qui est l'*exactitude* obtenue par un classifieur naïf qui classe toutes les observations dans la classe majoritaire. La majorité de ces algorithmes est implémentée dans le framework KEEL [3], ce qui permettra de comparer dans les prochains chapitres leurs performances à celles obtenues par notre approche. Dans le tableau récapitulatif, il s'agit des algorithmes situés dans le haut du tableau. La prochaine section détaille les bonnes pratiques en matière de comparaison des algorithmes, qui seront utilisées comme référence pour les protocoles de comparaison des prochains chapitres.

2.3 Protocole de comparaison des algorithmes

Dans les prochains chapitres nous serons amenés à comparer les performances de différents algorithmes ; à la fois pour évaluer les différences de performance entre plusieurs versions d'un même algorithme ou pour se comparer à d'autres méthodes de la littérature. Cette section évoque les bonnes pratiques décrites dans la littérature, en matière de comparaison statistique d'algorithmes. Elles s'appuient sur les recommandations de Demsar *et al.* [32] et les implé-

Table 2.3: État de l'art des algorithmes de classification

AD : arbre de décision ; AG : algorithme génétique ; LCS : learning classifier system ; CS : cost-sensitive ; DPR : Diviser pour régner ; MDL : minimum description length principle ; RN : réseaux de neurones ; PGG : programmation génétique à base de grammaire ; ADR-PGG algorithme de découverte de règles généré à partir de PGG.

Algorithme	Modèle	Approche(s)	Objectifs	Réf
AdaC2	AD	CS Boosting	Gain (C4.5)	[101]
BioHEL	Pittsburgh	AG, LCS	complexité, confiance, sensibilité	[8]
C4.5-CS	AD	construct. d'arbre CS	Gain (C4.5)	[106]
CORE	Pop1 : Pittsburgh	Co évolutionnaire	Sensibilité, Spécificité,	[103]
	Pop2 : Michigan		Compétition avec Jetons	
DataBoost-IM	AD	CS boosting, génération d'observations	Gain (C4.5)	[47]
DT-GA	AD	AG×AD	Sensibilité × Spécificité	[23]
DT-Oblique	Oblique AD	AG×Hill Climbing	Exactitude	[21]
GAssist	Pittsburgh	AG, LCS	Exactitude et complexité	[7]
Hider	Michigan	Michigan (Hiérarchique), DPR AG	N-FP,TP,Support	[2]
OCEC	Pop1 : Ens. d'observations	Co évolutionnaire	Complexité,	[62]
	Pop2 : Michigan		Importance des attributs	[62]
Ripper	Pittsburgh	DPR, élagage, post-traitement	Gain, taux d'erreur, MDL	[28]
SIA	Michigan	DPR×AG	Exactitude et Généralité	[108]
XCS	Michigan	AG, LCS	Exactitude relative	[115]
AntMiner	Michigan	Colonies de fourmis, DPR, élagage	Gain	[84]
jMAF	Ensembles approximatifs	Ensembles approximatifs	3 mesures de consistance	[14]
-	Michigan	AG	confiance, support, diversité	[91]
-	AD	AG	Taux d'erreur équilibré, complexité	[92]
-	ADR-PGG	PGG	Exactitude normalisée	[82]
-	Michigan	RN, AG	Confiance, Généralité	[116]
-	Michigan	AG	support, conviction, jmesure surprise, confiance	[76]

mentations correspondantes fournies par García *et al.* [43] dans le cadre de la comparaison de classifieurs et d'algorithmes d'apprentissage. Elles peuvent néanmoins être généralisées à la comparaison d'algorithmes. Tout d'abord nous détaillerons les différents jeux de données qui seront utilisés pour les expérimentations dans les prochains chapitres. Ensuite, les méthodes statistiques de comparaison d'algorithmes seront détaillées : comparaison de deux algorithmes, comparaison de plusieurs algorithmes, et comparaisons effectuées sur des instances différentes.

2.3.1 Jeux de données

Il est nécessaire de tester, calibrer et comparer les algorithmes sur des jeux de données variés, proches de la réalité. Cette partie présente tout d'abord les jeux de données réels sur lesquels nous travaillons. Afin de pouvoir se comparer à la littérature ou d'effectuer de nombreuses expérimentations, des jeux de données moins volumineux issus de la littérature sont nécessaires. Nous avons sélectionné 10 jeux de la littérature pour leur ressemblance à nos jeux de données réels, notamment au point de vue de l'asymétrie. Ils sont également détaillés dans cette partie. Pour finir, nous détaillons des jeux de données plus communs issus de la littérature, qui seront utilisés afin d'évaluer si notre approche peut être adaptée à des données plus génériques.

2.3.1.1 Données réelles

Nos jeux de données réelles sont issus de données PMSI, qui comme nous l'avions évoqué précédemment correspondent à un ensemble de patients, avec pour chacun d'entre eux l'ensemble des diagnostics et actes qui ont été réalisés durant leur(s) séjour(s) hospitalier. Chacun d'entre eux contient 10,000 patients, ce qui correspond approximativement au volume de pa-

tients qui passe dans un CHU de taille moyenne sur une durée de 2 mois et demi. Ces jeux sont présentés dans la table 2.4, qui détaille pour chacun d'entre eux son nombre de patients, le nombre d'attributs (dont attributs numériques), la répartition (degré d'asymétrie) et le libellé de la pathologie à prédire. Ainsi le jeu *avc* a un degré d'asymétrie de 2.44% ce qui signifie que 244 patients sur les 10,000 du jeu de données présentent un diagnostic d'AVC. Le jeu de données *hyp* représente la pathologie *hypertension*, qui est le diagnostic le plus souvent présent au sein de la base de données patient étudiée. Les deux premiers jeux correspondent à une extraction simple des données PMSI. Les deux derniers jeux ont été modifiés afin de permettre une comparaison avec les algorithmes de classification de la littérature, qui ont besoin d'un nombre plus restreint d'attributs. Seuls les attributs qui sont présents sur au moins un patient avec la pathologie à prédire ont été conservés. Ils sont donc post-fixés par un "f" signalant que les attributs ont été filtrés. Le jeu *tia-f* représente la prédiction de l'accident ischémique transitoire (AIC), qui est un AVC "léger" qui ne laisse pas de séquelles. Le jeu *s06-f* représente la prédiction d'une *commotion cérébrale*. Ces deux jeux présentent des degrés d'asymétrie assez importants, la classe à prédire est présente sur moins de 1% des observations.

Table 2.4: Détail des jeux de données réels

nom	#obs.	#att.	#num.	d_{asy}	Pathologie
avc	10,000	10,236	0	0.0244	AVC
hyp	10,000	10,255	0	0.2275	Hypertension
tia-f	10,000	699	0	0.0078	AIC - accident ischémique transitoire
s06-f	10,000	1,023	0	0.0099	Commotion cérébrale

2.3.1.2 Données standard de la littérature

Table 2.5: Détail des jeux de données de la littérature

nom	#obs.	#att.	#num.	source
adult	48,842	14	6	[40]
australian	690	14	6	[40]
breast	286	9	0	[40]
crx	125	15	6	[40]
heart	270	13	6	[40]
german	1,000	20	7	[40]
hepatisis	155	19	6	[40]
horsecolic	368	27	7	[40]
housevotes	435	16	0	[40]
mushrooms	8,124	112	0	[40],[89]

Nous avons sélectionné 10 jeux de données de la littérature comportant un maximum d'attributs qualitatifs. Ils ne présentent pas d'asymétrie des données, ce qui permettra de les utiliser pour évaluer l'algorithme proposé dans des conditions plus classiques de classification. La majorité de ces jeux de données proviennent de l'entrepôt de données de l'UCI[1]. Les jeux de données issues de partie de jeux de société – par exemple *tic tac toe* ou *chess* – ont été volontairement

1. http ://archive.ics.uci.edu/ml

exclus car ils ne permettent pas d'observer les capacités de généralisation des algorithmes. En effet, les observations (ici les parties) présentes dans les données utilisées pour l'apprentissage sont souvent présentes à l'identique dans les données de validation : un algorithme qui « colle » aux données d'apprentissage sans être capable de généraliser obtiendra de très bons scores. La table 2.5 détaille pour chacun des jeux de données sélectionné ses principales caractéristiques : nombre d'observations, nombre d'attributs (dont numériques) et la source. Un rapide descriptif de chacun des jeux de données est proposé dans les annexes (section B). Les attributs quantitatifs des jeux de données post-fixés par $_d$ seront discrétisés en 10 parts.

2.3.1.3 Données asymétriques de la littérature

De plus, afin de respecter une contrainte forte de nos données, nous avons sélectionné 10 jeux de données issus de la littérature, chacun présentant des degrés d'asymétrie d_{asy} différents, variant de 0.2742 à 0.0077. Cette particularité les rend plus ressemblants aux jeux de données médicales. La majorité de ces jeux de données provient de l'entrepôt de données de l'UCI [1]. Afin d'obtenir des jeux de données asymétriques, les jeux de données qui contiennent plusieurs classes à prédire ont été modifiés en jeux de classification binaires, comme proposé par Fernandez *et al.* [38], d'où proviennent donc ces jeux asymétriques. Peu de jeux de données binaires ou qualitatives sont disponibles dans la littérature, ce qui rend difficile de trouver des jeux de données correspondant à nos jeux réels. La plupart des jeux trouvés vont donc être discrétisés, afin d'obtenir des attributs non numériques, plus proches des données binaires. Les jeux de données sélectionnés sont détaillés dans la table 2.5. Elle détaille pour chacun des jeux de données leur nombre d'observations, d'attributs (dont numériques), le degré d'asymétrie (d_{asy} ; pourcentage des observations qui ont la classe à prédire) et la source. Les jeux de données *w1a* et *a1a* correspondent à des jeux des données de l'UCI qui ont été binarisés [89], ce qui les rend plus proches de nos jeux de données réels et permet d'obtenir un plus grand nombre d'attributs. Le jeu de données *lucap0* a été généré pour le Machine Learning Challenge [48]. Un rapide descriptif de chacun des jeux de données est proposé dans les annexes (section B).

Table 2.6: Détail des jeux de données asymétriques de la littérature

nom	#obs.	#att.	#num.	d_{asy}	source
haberman	306	3	3	0.2742	[40]
ecoli1	336	7	7	0.2292	[40]
ecoli2	336	7	7	0.1548	[40]
yeast3	1,484	8	8	0.1038	[40]
abalone9vs18	731	8	7	0.0565	[40]
yeast2vs8	482	8	8	0.0415	[40]
abalone19	4,174	8	7	0.0077	[40]
w1a [2]	2,477	300	0	0.0290	[89]
lucap0	2,000	144	0	0.2785	[48]
a1a [3]	1,605	123	0	0.2461	[89]

1. http ://archive.ics.uci.edu/ml

2.3.2 Comparaison statistique d'algorithmes

De nombreux tests statistiques sont disponibles et le choix du test le plus adapté à la problématique n'est pas toujours aisé. Après une courte introduction sur les tests statistiques, nous présentons les différents cas de figure qui peuvent se présenter et leurs combinaisons : comparaison sur une ou plusieurs instances, d'un ou plusieurs algorithmes. Pour chacun, nous détaillons les préconisations évoquées dans la littérature sur l'apprentissage automatique.

2.3.2.1 Généralités sur les tests statistiques

Un test statistique permet d'évaluer si une hypothèse H_0 – appelée hypothèse nulle – est vérifiée statistiquement. Chaque test fournit une probabilité – appelée la *p-value* – qui correspond à la probabilité de faire une erreur si l'on conclut que l'hypothèse H_0 est fausse. En général, on fixe un seuil $\alpha = 5\%$ ou $\alpha = 10\%$; si la *p-value* le dépasse on acceptera l'hypothèse H_0. On distingue deux types de tests : les paramétriques et non-paramétriques. Les premiers supposent que les données respectent certains pré-requis, par exemple qu'elles sont distribuées sous la forme d'une loi normale ou que l'homogénéité des variances est vérifiée. Lorsque les données ne correspondent pas aux pré-requis attendus les résultats de ces tests ne seront pas fiables. Les tests non-paramétriques quant à eux fonctionnent sans pré-requis sur la distribution des données. Dans le cadre de l'apprentissage automatique, Demsar *et al.* ont constaté expérimentalement que les données respectent rarement les pré-requis pour les tests paramétriques : ils recommandent de privilégier les tests non-paramétriques. Ainsi, par la suite nous présenterons uniquement des tests non-paramétriques.

2.3.2.2 Comparaison statistique sur une seule instance

Cette approche est utilisée lorsque l'on souhaite comparer l'efficacité de deux ou plusieurs algorithmes dans des conditions spécifiques, par exemple sur un jeu de données réel ou un problème particulier. Cela permet de déterminer quel algorithme est à privilégier sur un problème très particulier.

Comparaison statistique de deux algorithmes Dans cette partie, nous souhaitons comparer deux algorithmes A et B. Chacun d'entre eux a été exécuté 25 fois par instance ; nous obtenons donc 25 scores par algorithme. Le test de Mann-Whitney [77] permet de vérifier si l'ensemble des 25 scores d'un des deux algorithmes surpasse l'ensemble des 25 scores obtenus par l'autre. L'hypothèse nulle est que les scores obtenus par les deux algorithmes (soient ici les 50 scores) sont équivalents.

Comparaison statistique de plus de deux algorithmes Ce cas est similaire au cas précédent, à la différence près que l'on souhaite ici comparer plus de deux algorithmes sur un même jeu de données (ou instance). Ainsi, nous souhaitons comparer trois algorithmes A, B et C. Chacun d'entre eux a été exécuté 25 fois sur la même instance ; nous obtenons donc 25 scores par algorithme. Il est tentant d'effectuer la comparaison en effectuant plusieurs comparaison

deux à deux : A vs B, A vs C et B vs C avec le test de Mann-Whitney présenté précédemment. Habituellement, on rejette l'hypothèse nulle (H_0) si la p-valeur obtenue par le test est inférieure à 0.05 ; ce qui signifie que H_0 est rejetée avec une probabilité d'erreur de 5%. Cependant, si plusieurs tests sont effectués sur les mêmes données la probabilité de faire une erreur sur au moins un des tests va augmenter avec le nombre de tests : après 10 tests elle devient de 40% ! Une analogie est parfois faite avec un lancer de dé : un test statistique correspond à un lancer de dé ; plus on lance le dé, plus on a de chances d'obtenir un 6, dans le cas du test statistique, de faire une erreur...

Une solution est d'utiliser un test adapté à la comparaison statistique de plus de deux algorithmes, comme le test de Kruskal-Wallis [69]. Il permet de déterminer si les algorithmes sont semblables ou non, au regard des scores obtenus. Si non, des tests supplémentaires – appelés tests *post-hoc* – peuvent ensuite être réalisés pour déterminer lequel des algorithmes est le plus performant. Les algorithmes vont être comparés deux à deux avec le test de Mann-Whitney présenté précédemment, tout en ajustant la p-valeur (ou le seuil $\alpha = 0.05$ avec lequel on rejette $H0$) pour prendre en compte le risque augmenté d'erreur. Une manière simple consiste à diviser le seuil par le nombre de tests effectués ; il s'agit de la correction de Bonferroni [36]. Ainsi pour nos algorithmes A, B et C, 3 tests vont être effectués, la p-valeur seuil sera de $\alpha = 0.017$ au lieu de $\alpha = 0.05$. D'autres manières plus efficaces de procéder ont été proposées, comme la méthode de Holm [52]. La méthode de Bergmann-Hommel [11] est encore plus efficace, mais demande plus de temps de calcul, elle est donc à réserver aux cas où la méthode de Holm n'a pas permis de trancher.

2.3.2.3 Comparaison statistique sur plusieurs instances

Cette approche est utilisée lorsque l'on souhaite étudier le comportement de plusieurs algorithmes de manière générale, afin de détecter l'algorithme qui sera le plus robuste. Elle permet de détecter un algorithme qui fonctionne sur la majorité des instances et qui aura le plus de chances de donner de bons résultats sur une nouvelle instance similaire à celles étudiées. Dans de nombreux articles, une méthode pour évaluer les performances d'algorithmes sur plusieurs instances consiste à compter le nombre fois où chacun des algorithmes a été statistiquement plus efficace qu'un autre. Les méthodes présentées précédemment sont utilisées pour définir sur chaque instance, pour chaque algorithme le nombre d'algorithmes qu'il bat statistiquement, et par lesquels il est battu. Les nombres de victoires et défaites sont additionnés sur l'ensemble des instances ; l'algorithme qui cumule le plus grand nombre de victoires est considéré comme le plus efficace. Cependant cette méthode peut parfois masquer les résultats, en ne donnant pas de résultats significatifs. Si l'on compare un algorithme A et un algorithme B sur un grand nombre d'instances, A peut obtenir des résultats légèrement meilleurs que B sur chacune des instances, sans pour autant que la comparaison statistique soit significative sur chacune des instances prises séparément. Dans ce cas, cette méthode va conclure à tort que A et B sont équivalents. En effet, il est possible que A obtienne un meilleur score que B "par hasard" sur une des instances. Cependant si A est légèrement meilleur que B sur plusieurs instances, il est moins probable que cela soit "par hasard", mais les tests statistiques pris instance par instance

vont considérer la différence comme due au hasard. Ainsi, cette méthode de comparaison est trop stricte et peut supprimer des informations intéressantes, et ainsi masquer des différences entre deux algorithmes. Des tests statistiques plus efficaces, qui prennent en compte l'ensemble des instances, sont proposés dans la suite.

Deux algorithmes Comme auparavant, nous souhaitons comparer deux algorithmes A et B. Chacun d'entre eux a été exécuté 25 fois ; cette fois-ci sur 10 instances $\{I_1..I_{10}\}$ en validation croisée. Le test des rangs signés de Wilcoxon [114] va permettre de comparer les deux algorithmes A et B en fonction de la moyenne des résultats obtenus sur chacun des jeux de données. 10 paires de valeurs seront ainsi utilisées pour le test ; à chaque instance I_i sera associé une paire contenant la moyenne des résultats obtenus par l'algorithme A, et la moyenne de ceux obtenus par l'algorithme B. Le test va permettre de déterminer si un algorithme est plus performant sur l'ensemble des instances. L'hypothèse nulle est que les deux algorithmes sont équivalents sur les instances étudiées. Pour chaque instance, le test va calculer la différence de score entre les deux algorithmes. Dans notre exemple, on obtient 10 différences, auxquelles le test va associer un rang : de la plus faible différence à la plus grande. Il utilisera ensuite ces rangs pour évaluer si un algorithme est plus performant qu'un autre. Plus de détails sur le fonctionnement sont disponibles dans l'article de Demsar *et al.* [32].

Plus de deux algorithmes Nous souhaitons maintenant comparer trois algorithmes A, B et C sur 10 instances $\{I_1..I_{10}\}$. On dispose de la moyenne des scores obtenus par chacun des algorithmes sur chacune des instances, soient 3 scores par instance. À partir de ces données, le test de Friedman va assigner à chaque algorithme un rang sur chacune des instances. En cas d'*ex-æquo* sur une même instance, les algorithmes se voient assigner un rang moyen (par exemple si 2ème *ex-æquo*, les algorithmes auront un rang de 2.5). Le test calcule ensuite le rang moyen obtenu pour chaque algorithme, toutes instances confondues. L'hypothèse nulle est que les rangs moyens obtenus sont statistiquement équivalents. Lorsque les rangs sont statistiquement différents, des tests post-hoc peuvent être réalisés pour essayer de déterminer l'algorithme qui se démarque. Il s'agit des tests à une seule instance présentés précédemment qui peuvent être utilisés. D'autres part, les rangs moyens obtenus peuvent être également utilisés pour effectuer la comparaison. Le test d'Iman-Davenport est une amélioration du test de Friedman et peut également être utilisé.

2.3.2.4 Récapitulatif

Nous avons réalisé un récapitulatif des méthodes comparatives à utiliser selon le contexte d'utilisation, sous la forme d'une illustration dans la figure 2.5. Cette figure présente un récapitulatif des méthodes comparatives à utiliser selon le nombre d'instances et d'algorithmes étudiés. Le test de Mann-Whitney peut être utilisé seul pour comparer 2 algorithmes sur une même instance. Associé à des méthodes correctives comme Bonferroni ou Holm, il peut être exécuté plusieurs fois pour gérer plusieurs instances et plusieurs algorithmes. Cependant il sera moins efficace lorsque le nombre de tests augmente (nombre d'instances ou nombre d'algorithmes).

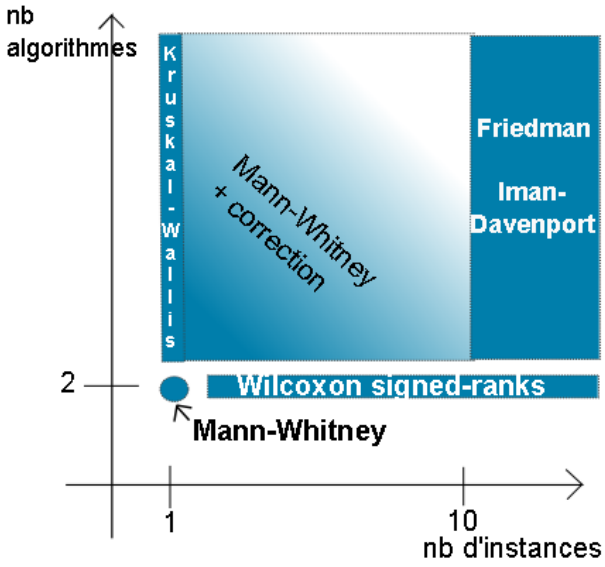


Figure 2.5: Méthodes statistiques selon leur champ d'application

Lorsque les algorithmes sont comparés sur une seule instance, les tests 2 à 2 de Mann-Whitney peuvent être précédés d'un test de Kruskal-Wallis, qui va déterminer si les algorithmes sont équivalents ou non. Lorsque les algorithmes sont comparés sur plusieurs instances, le test de Mann-Whitney peut être moins efficace que des approches qui évaluent les résultats sur toutes les instances confondues, car il traite les instances indépendamment. Ainsi, si un algorithme est « légèrement » meilleur sur un ensemble d'instances, la différence ne sera pas forcément visible en regardant uniquement les instances une à une, par rapport à des tests qui observent les résultats obtenus sur l'ensemble des instances. Ces derniers tests peuvent être réalisés à partir de 10 instances. Le test des rangs signés de Wilcoxon peut être utilisé pour comparer deux algorithmes, les tests de Friedman et Iman-Davenport seront préférés lorsque le nombre d'algorithmes est supérieur à 2.

2.4 Conclusion

Dans ce chapitre, nous avons défini le problème de classification, ainsi que les mesures et méthodes utilisées pour évaluer ses performances. Nous avons ensuite exposé rapidement les problèmes posés par les données asymétriques en classification et les solutions proposées dans la littérature, ainsi que quelques algorithmes qui présentent l'état de l'art. Nous nous sommes alors intéressés aux méta-heuristiques, en proposant une introduction à ces méthodes ainsi qu'à l'optimisation multi-objectif, et en examinant ensuite les différentes approches proposées dans la littérature pour la classification. Afin de pouvoir se comparer dans la suite à ces approches et évaluer correctement les performances, la fin du chapitre a été consacrée aux jeux de données

utilisés pour l'évaluation ainsi que les bonnes pratiques en matière de comparaison statistique d'algorithmes. Nous pouvons maintenant nous attaquer à la modélisation du problème de classification et l'évaluation de ses performances dans le chapitre suivant, en nous inspirant de ce qui a été présenté dans ce chapitre.

Chapitre 3

MOCA-I : Modélisation sous forme d'un problème multi-objectif et résolution à l'aide d'une recherche locale

Dans ce chapitre nous décrivons tout d'abord comment le problème de classification partielle sur données asymétriques peut être modélisé sous forme d'un problème multi-objectif et abordé à l'aide d'une recherche locale multi-objectif : l'algorithme MOCA-I (Multi-Objective Classification algorithm for Imbalanced data). Nous détaillerons ensuite notre modélisation, ainsi que les études statistiques qui ont été nécessaires pour l'élaborer. Ensuite, nous décrirons sa résolution à l'aide d'une recherche locale. Après cette phase de conception, nous étudierons le comportement de MOCA-I. Tout d'abord, nous démontrons expérimentalement l'apport de la multi-objectivisation, ce qui a fait l'objet d'une publication dans la conférence GECCO 2013 [59]. Enfin nous étudions l'impact de différentes implémentations et paramétrages de MOCA-I afin de déterminer l'implémentation la plus efficace. Les résultats obtenus par cette implémentation sont ensuite comparés aux meilleurs algorithmes de la littérature.

3.1 Modélisation sous forme d'un problème multi-objectif et résolution à l'aide d'une recherche locale

Cette section détaille la modélisation utilisée dans MOCA-I et sa résolution sous la forme d'une recherche locale. Cette section est divisée en deux parties, la première aborde la modélisation multi-objectif et une étude statistique est réalisée afin d'identifier les objectifs à utiliser comme fonctions objectif. La seconde partie se focalise sur la résolution à l'aide d'une recherche locale. Comme nous l'avons vu précédemment, la mise en place d'une recherche locale nécessite

la définition de plusieurs éléments : la représentation d'une solution, le voisinage et la (ou les) fonction(s) objectif(s). Nous présentons les composants nécessaires à l'implémentation de la recherche locale : la représentation d'une solution sous forme *Pittsburgh* et le voisinage associé.

3.1.1 Étude statistique des objectifs candidats

Comme nous l'avons vu dans le chapitre précédent, il existe de nombreuses mesures de qualité de classifieurs et donc tout autant d'objectifs candidats. On peut ainsi chercher des classifieurs qui optimisent l'*exactitude*, la *confiance*, le *support*, etc. Le choix des objectifs est important : Jensen *et al.* ont montré qu'utiliser trop d'objectifs n'améliorait pas les résultats [61], tout en augmentant la complexité de l'algorithme. De plus, ils suggèrent d'utiliser des objectifs complémentaires ou en conflit ; des objectifs trop similaires vont apporter des solutions identiques. Afin de faciliter le choix parmi ces objectifs candidats, nous avons réalisé une étude statistique sur les différents objectifs candidats. Cette étude est similaire à celle réalisée précédemment par Khabzaoui *et al.* [66] sur des règles d'association, qui a permis d'identifier des corrélations entre les objectifs et regrouper ceux-ci en 5 groupes. Les données utilisées dans notre étude seront différentes : il s'agit ici d'un problème de classification partielle et les données sont asymétriques. L'*analyse en composantes principales (ACP)* permet de mettre en évidence des corrélations à partir d'une liste d'individus et de leurs attributs. Dans notre cas, les individus seront des règles de classification et les attributs les mesures de qualité (*confiance*, *support*, etc.). Nous étudierons les règles de classification générées sur trois jeux de données de la littérature décrits précédemment, présentant des degrés d'asymétrie d_{asy} variables (de 0.0077 à 0.2742) : $haberman_d$, $yeast3_d$ et $abalone19_d$. Nous vérifierons ensuite que le comportement est identique sur les jeux des données réels *avc* et *hyp*.

3.1.1.1 Méthodologie

Les mesures suivantes ont été sélectionnées, en tenant compte de leurs capacités à gérer les données asymétriques : *confiance* (Cf), *conviction* (Conv), *cosinus* (cos), *laplace* (LP), *lift*, *piatetsky-shapiro* (PS), *sensibilité* (Se), *spécificité* (Sp), *négatifs découverts* (UN), *support* (S) et *surprise* (Sur). L'*exactitude* a été volontairement exclue de l'analyse car elle pose problème sur les données asymétriques. Diverses mesures combinées ont également été utilisées, qui permettent de limiter le sur-apprentissage ou gérer l'asymétrie des données. *CfRL* favorise les règles courtes : la *confiance* est divisée par le nombre de termes de la règle, ce qui limite ainsi le sur-apprentissage. Dans la littérature, la sensibilité peut être multipliée par la confiance (cfSe) ([112]) et par la spécificité (seSp) ([22]) pour gérer les règles rares. La *f-mesure* (FM) a été utilisée avec $\beta = 1$, qui est la valeur la plus communément utilisée sur les données asymétriques [50]. Les données nécessaires à l'ACP sont générées de la manière suivante : pour chacun des jeux de données, 1500 règles de classification sont générées et les 15 mesures étudiées sont calculées pour chacune d'entre-elles. Il est nécessaire de s'assurer d'obtenir des règles de classification qui permettent à chaque mesure d'obtenir un bon score. Ainsi, parmi les 1500 règles de classification, 750 règles sont générées aléatoirement. Les règles restantes sont générées à partir

de l'algorithme de recherche locale présenté plus tard dans ce chapitre. À partir de 50 règles aléatoires, chacune des 15 mesures est utilisée comme fonction objectif, ce qui permet d'obtenir 750 règles (50×15). Pour chacune des mesures nous avons donc la certitude d'obtenir au moins 50 règles sur lesquelles la mesure aura une bonne valeur. L'ACP a été réalisée avec le package *FactoMineR* du langage R [64].

3.1.1.2 Résultats sur jeux littérature

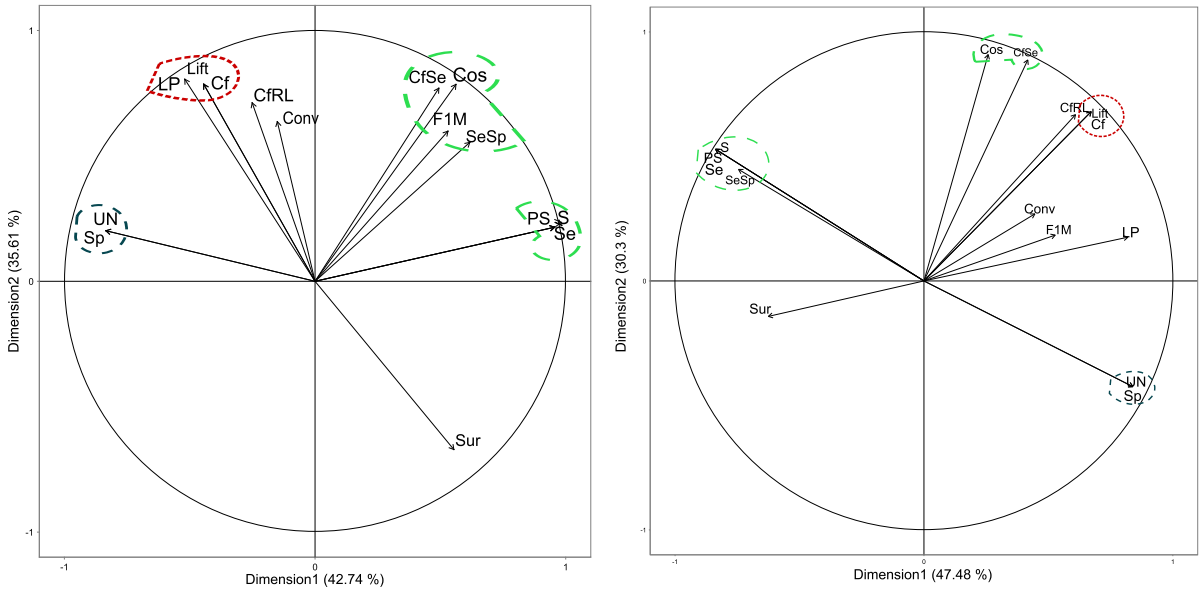


Figure 3.1: Cercle des corrélations - jeux $yeast3_d$ et $abalone19_d$

L'ACP produit 9 graphiques de corrélations : un par jeu de données, selon plusieurs axes d'analyse. Dans notre cas, 3 axes d'analyse sont significatifs. Les figures 3.1 et 3.2 présentent les projections les plus représentatives pour les 3 jeux de données $yeast3_d$, $abalone19_d$ et $haberman_d$, sur les axes 1 et 2 avec une inertie significative sur les 3 jeux (respectivement 84.59%, 78.35% et 77.78%). Les mesures les plus proches du cercle sont celles qui sont bien représentées sur les données étudiées. Lorsque les mesures sont éloignées du cercle, comme c'est le cas avec la *conviction*, les données ne permettent pas d'extraire une tendance ou une corrélation. Les mesures proches les unes des autres dans le cercle des corrélations sont corrélées. Ainsi, les différents cercles de corrélation mettent en évidence plusieurs groupes de mesures similaires :

- piatetsky-shapiro, sensibilité, sensibilité×spécificité, support et (f1-mesure) (axe 1)
- +(confiance×sensibilité, cosinus)
- spécificité, négatifs découverts et (surprise) (axes 1 et 3)
- confiance, (confiance / taille règle), (conviction), laplace, lift et (surprise) (axes 2 et 3)

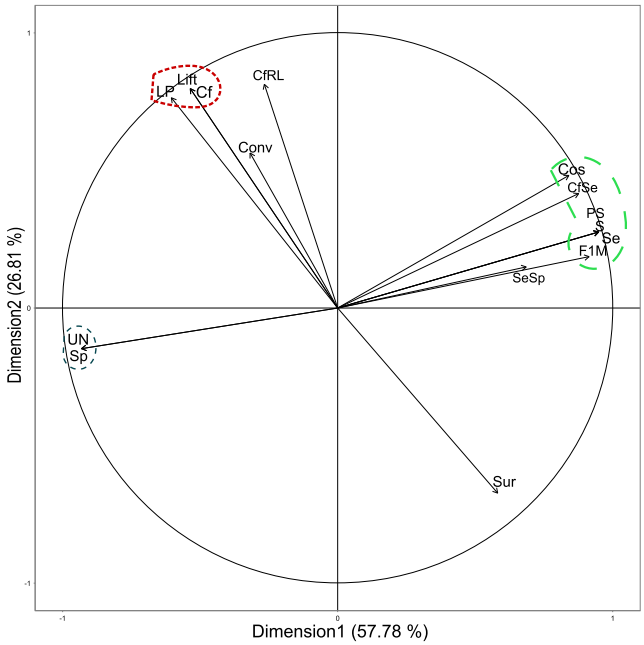


Figure 3.2: Cercle des corrélations - Règles de classification pour le jeux de données $haberman_d$

Les mesures indiquées entre parenthèses ne sont pas toujours bien représentées sur le cercle des corrélations, ou leur groupe peut changer selon le jeu de données. Ainsi, *laplace* est groupé avec le *lift* et la *confiance* sur les jeux $haberman_d$ et $yeast3_d$, mais ce n'est pas le cas sur le jeu $abalone19_d$. D'autre part, les deux premiers groupes sont fusionnés sur le jeu $haberman_d$ et sont séparés sur les deux autres jeux de données. Cette différence pourrait être expliquée par l'asymétrie des données qui est plus marquée sur les jeux $yeast3_d$ et $abalone19_d$.

3.1.1.3 Résultats sur jeux réels

L'ACP sur les jeux réels nous permet de confirmer les tendances observées précédemment sur les jeux de données de la littérature. Nous obtenons 6 cercles de corrélations, les plus significatifs sont proposés dans la figure 3.3 qui donne le cercle des corrélations avec la projection sur les axes 1 et 2 pour les jeux de données *hyp* et *avc* avec des inerties respectives de 76.51% et 79.07%. Les 3 groupes principaux observés précédemment sont retrouvés. Le groupe du *cosinus* et de la *sensibilité* est divisé en deux sur le jeu *avc* qui présente une asymétrie de la classe plus prononcée que le jeu *hyp*. De la même manière, la mesure de *laplace* n'est plus corrélée à la *confiance* sur le jeu *avc* - lorsque l'asymétrie des données est plus prononcée.

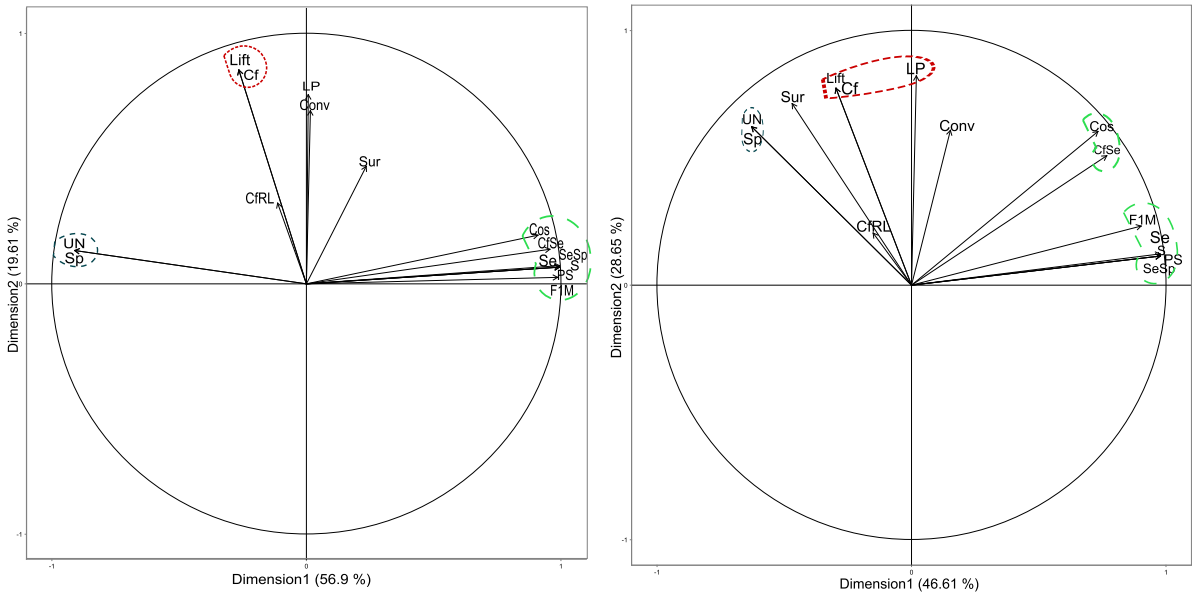


Figure 3.3: Cercle des corrélations - jeux réels *hyp* et *avc*

3.1.1.4 Discussion et interprétation

L'ACP a permis de regrouper les différentes mesures selon leur similarité. Trois grands groupes de mesures ont été identifiés et sont retrouvés sur chacun des jeux de données ; certaines corrélations peuvent être expliquées par l'asymétrie des données et sont détaillées dans la suite. Lorsque l'asymétrie des données augmente, certaines mesures quittent le groupe de mesures avec lesquelles elles étaient corrélées (comme *laplace* ou le *cosinus*). Ainsi, ces mesures risquent d'être moins efficaces si le jeu de données est assez équilibré. Dans l'objectif de choisir des mesures complémentaires, il est préférable de choisir des mesures qui sont dans des groupes différents quelque que soit le degré d'asymétrie d_{asy}.

Impacts de l'asymétrie des données (règles rares) L'asymétrie de nos données implique que la prédiction C_{pred} ou P est peu présente parmi les individus : $d_{asy} < 0.5$. Dans le tableau de contingence 2.1, cela implique pour l'AVC (2% des individus) que TP+FN ($|P|$) est 50 fois plus petit que FP+TN ($|\overline{P}|$). Cette répartition va avoir un impact sur certaines mesures qui deviennent proches, conformément à ce qu'indique l'ACP :

$$\textit{spécificité} = \frac{TN}{TN+FP} \approx \frac{TN}{TN+FP+(FN+TP)} = \textit{négatifs découverts}$$

Impacts des règles de classification L'ACP indique des similitudes entre *lift* et *confiance* et *support* et *sensibilité*. Celles-ci peuvent s'expliquer par la nature des règles : une règle de classification est une règle où la prédiction P est fixe. Les mesures qui utilisent TP+FN ($|P|$) ou FP+TN ($|\overline{P}|$) vont ainsi avoir un comportement similaire à des mesures qui utilisent N, car

elles seront proportionnelles :

$$lift = \frac{N \times TP}{(TP+FP)\times(TP+FN)} = \frac{N}{TP+FN} \times \frac{TP}{TP+FP} = \frac{N}{|P|} \times confiance$$

$$support = \frac{TP}{N} \sim sensibilité = \frac{TP}{|P|}$$

Différences avec la littérature Certaines différences sont à noter avec l'analyse précédemment effectuée par Khabzaoui et al. [66] sur des règles d'associations où : *cosinus* et *confiance* appartiennent au même groupe, *lift* (intérêt) et *confiance* à des groupes différents. D'autres divergences sont présentes avec les résultats de Abe et Tsumoto [1] sur des règles de classification : le *support* et la *confiance* sont corrélés, *lift* et *confiance* ne le sont pas, ce qui n'est pas le cas sur nos données. Cela peut s'expliquer par la nature des règles utilisées pour réaliser l'ACP : ils ont utilisé des règles de bonne qualité issues de l'algorithme PART, qui assurent une bonne confiance et un bon support. Ces différences mettent en évidence que les corrélations dépendent du jeu de données et du type de règles recherchées : il est préférable de les vérifier au préalable pour aider au choix des critères pour l'optimisation.

Conclusion Nous avons observé que, sur des règles de classification rares, certaines mesures comme le *lift* et la *confiance* ; *négatifs découverts* et *spécificité* ; *support* et *sensibilité* sont similaires ou proportionnelles. Cinq ACP sur 7500 règles de classification rares en provenance de cinq jeux de données, montrent que les mesures peuvent être séparées en trois groupes :

- confiance×sensibilité, cosinus, piatetsky-shapiro, sensibilité, sensibilité×spécificité, support et f-mesure (axe 1)
- spécificité, négatifs découverts et (surprise) (axes 1 et 3)
- confiance, (confiance / taille règle), (conviction), laplace, lift et (surprise) (axes 2 et 3)

Dans l'optique de la réalisation d'un algorithme de recherche de règles, il est préférable de choisir des mesures appartenant à des groupes séparés. Les mesures appartenant à un même groupe vont générer le même type de règles. Dans notre cas, nous choisissons d'utiliser la *confiance* et la *sensibilité* qui sont plus particulièrement utilisées en médecine. L'objectif sur la *spécificité* a été éliminé après les premières expérimentations car il s'est avéré qu'il générait le même type de règles que la *confiance*. Ce comportement est conforté par les formules de la *confiance* et de la *spécificité* (voir table 2.2 dans le chapitre 2) ; les maximiser revient à minimiser le nombre de faux positifs (FP).

Enfin, un objectif supplémentaire a été ajouté afin de limiter le phénomène de *bloat*. En optimisation, ce phénomène se produit sur les représentations de longueur variable, lorsqu'une solution se complexifie indéfiniment sans pour autant qu'il y ait une amélioration de sa ou ses fonctions objectif. Par exemple, la règle $âge = 62 \wedge toux = oui \wedge diabète = oui \wedge metformin = oui \wedge fièvre = oui \wedge courbatures = oui \rightarrow grippe = oui$ est le produit du *bloat* : elle correspond à un seul patient, ce qui présente un cas particulier et n'est pas très utile pour prédire la grippe. Une règle équivalente – présentant la même *Confiance* – sans *bloat* serait : $toux = oui \wedge fièvre = oui \wedge$

courbatures = oui → *grippe = oui*. Le principe de description de longueur minimale (*MDL : Minimum Description Length principle*) est communément mis en place pour pallier le *bloat*. Il a été introduit par Rissanen *et al.* [94] et reprend le principe du rasoir d'Occam : lorsque deux hypothèses sont identiques, la plus simple doit être privilégiée. Ce principe peut être implémenté de plusieurs manières : par l'intermédiaire d'un opérateur de voisinage ou de mutation (si l'on utilise un algorithme génétique) ou sous forme d'un objectif. L'opérateur de voisinage que nous présenterons dans la suite comporte quelques modifications pour contrer le *bloat*. Dans la littérature, une implémentation courante du *MDL* est d'ajouter un objectif qui favorise les solutions simples, dans notre cas minimiser la taille du classifieur. Cet objectif additionnel permet également de générer des solutions plus simples, ou d'autoriser des solutions complexes si les autres objectifs sont améliorés. Nous obtenons donc les trois objectifs suivants :

- maximiser la *confiance*
- maximiser la *sensibilité*
- minimiser le *nombre de termes*

3.1.2 Représentation d'une solution et voisinage

Nous venons de voir que notre algorithme d'optimisation devra trouver des classifieurs ayant une *confiance* maximale, *sensibilité* maximale et un *nombre de termes* minimal. Il est encore nécessaire de définir la représentation d'une solution – dans notre cas le type de classifieur que nous recherchons (arbre de décision, règles, ensemble de règles, etc.) – qui sera le plus adapté au besoin métier. Une fois cette représentation obtenue, nous proposerons un opérateur de voisinage, qui permet de passer d'une solution à une autre.

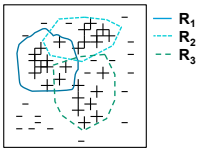
3.1.2.1 Représentation

Certaines pathologies que l'on souhaite prédire peuvent être générées par différents facteurs de risques. Par exemple, le risque thromboembolique est augmenté en cas de diabète, hypertension ou d'antécédent de thrombose. Lorsque plusieurs facteurs de risque sont présents chez un même patient, le risque est plus important que si un seul facteur de risque est présent. Ainsi, on peut s'attendre à ce que plusieurs règles puissent prédire une même pathologie, et que la prédiction soit renforcée si plusieurs règles sont déclenchées. Dans ce contexte, la représentation sous forme de règles semble plus pertinente que la représentation sous forme d'arbres de décision : il sera plus facile d'identifier les facteurs de risque s'ils sont chacun isolés dans une règle. De plus, les branches d'un arbre de décision sont souvent construites sur la base du « diviser pour régner » : les observations qui sont déjà gérées par une branche de l'arbre sont supprimées pour la suite de l'apprentissage, ce qui rend plus complexe la détection de facteurs de risque.

Il reste maintenant à affiner une représentation sous forme de règles. La figure 3.4 présente 3 règles R_1, R_2 et R_3 qui représentent chacune un facteur de risque. Les patients atteints de la pathologie sont représentés par $+$, les autres par $-$. La table de la figure 3.4 donne les scores de ces règles. Précédemment nous avions vu que deux représentations de règles sont possibles : à base de règles (Michigan) ou d'ensemble de règles (Pittsburgh). Si l'on regarde la table de la

Règle	Confiance	Sensibilité
R_1	1	0.42
R_2	0.9	0.35
R_3	0.83	0.38
$R_1 \cup R_2 \cup R_3$	0.88	1

(a) Performances des règles



(b) Règles Michigan et dominance

FIGURE 3.4: Couverture des règles avec la représentation Michigan

figure 3.4, un algorithme d'optimisation utilisant la représentation Michigan conserverait uniquement la règle ayant le meilleur score, soit R_1. En effet, les règles R_2 et R_3 sont dominées par R_1 et ne sont pas conservées alors qu'elles sont tout de même intéressantes d'un point de vue médical étant donné qu'elles ont une bonne confiance. La représentation Pittsburgh est plus complexe mais permettrait de représenter l'ensemble de règles $R_1 \cup R_2 \cup R_3$ qui a une meilleure sensibilité – c'est-à-dire couvre plus de patients – et conserve R_2 et R_3 qui auraient été perdus avec la représentation Michigan.
Ainsi, chaque solution sera représentée par un ensemble de règles de taille variable qui prédisent la même classe. Si au moins une des règles est déclenchée, on considère que la classe est présente. Les incohérences au sein d'un même ensemble de règles sont évitées car il ne peut pas exister au sein d'un ensemble deux règles avec des prédictions différentes [24]. Chaque ensemble de règles peut contenir un nombre variable de règles, et chaque règle est formée d'une conjonction de 1 à 9 termes. Chaque terme est composé d'un attribut, d'un opérateur et d'une valeur (facultative), permettant de gérer les types d'attributs suivants :

Attribut qualitatif le terme fait un test d'égalité sur une des valeurs possibles de l'attribut ; par exemple « genre = femme »

Attribut qualitatif avec notion d'ordre le terme fait un test d'égalité, supériorité ou infériorité sur une des valeurs possibles de l'attribut ; par exemple « age > [45-50] »

3.1.2.2 Voisinage

Le voisinage d'un ensemble de règles est représenté par tous les ensembles de règles ayant une différence d'un terme avec cet ensemble de règles ; il peut s'agir d'un terme en moins ou en plus, mais également d'un terme dont une valeur ou un opérateur a été modifié. Sur les attributs qualitatifs avec notion d'ordre nous proposons un voisinage simplifié qui permet de limiter le nombre de voisins d'un terme à 5, même lorsque l'attribut concerné a un grand nombre de valeurs possibles. Ce voisinage est présenté dans le tableau 3.1 qui indique la liste des voisins possibles pour chaque opérateur (=,<,>), en supposant que l'attribut peut prendre les valeurs v_i et que les valeurs sont ordonnées ($v_{i-1} < v_i < v_{i+1}$). Le symbole Ø signifie que le terme

est supprimé. Ce voisinage a été expérimenté sur le jeu de données *heart* – qui contient des listes ordonnées et classiques – avec une réduction du temps d'exécution de 14% en moyenne, sans pour autant dégrader les performances de la classification (moins de 1% de perte). Afin de réduire le temps d'exécution, nous supprimons également du voisinage certains voisins qui n'apporteront pas d'amélioration. Ainsi, il n'est pas nécessaire d'ajouter un terme à une règle qui a déjà une *Confiance* maximale, les termes ajoutés pourront uniquement réduire le nombre d'observations qui correspondent à la règle, et donc la *Sensibilité*. De la même manière, il n'est pas nécessaire d'ajouter un terme à une règle qui ne concerne aucune observation (nbC = 0). Ces simplifications du voisinage permettent également d'appliquer le principe MDL car elles privilégient les règles les plus simples.

Table 3.1: Voisinage simplifié d'un attribut qualitatif avec notion d'ordre

$a = v_i$	$a < v_i$	$a > v_i$
Ø	Ø	Ø
$a > v_{i-1}$	$a = v_{i-1}$	$a = v_{i+1}$
$a < v_{i+1}$	$a < v_{i-1}$	$a > v_{i+1}$
$a = v_{i-1}$	$a < v_{i+1}$	$a < v_{i+2}$
$a = v_{i+1}$	$a > v_{i-2}$	$a > v_{i-1}$

3.1.3 Implémentation DMLS

Nous avons vu les différents composants nécessaires à l'implémentation d'une recherche locale. Cette partie détaille l'implémentation de MOCA-I au sein de l'algorithme DMLS (présenté dans la section 2.2.4.2). L'algorithme 2 décrit le fonctionnement de MOCA-I. Il démarre d'une population initiale de 100 ensembles de règles, qui contiennent chacun deux règles générées à l'aide de l'algorithme 1. Cet algorithme assure d'obtenir des règles qui correspondent à au moins une observation, en créant chaque règle à partir d'une observation aléatoire et en lui ajoutant du bruit. Tous les ensembles de règles de la population initiale sont marqués comme non visités. À chaque itération, des ensembles de règles non visités sont sélectionnés ; plusieurs stratégies de sélection existent et seront étudiées dans la suite : une stratégie exhaustive qui visite toutes les solutions et une stratégie simple qui sélectionne une seule solution. Chaque solution sélectionnée est visitée : son voisinage – ou une sous-partie si l'on souhaite s'arrêter au premier voisin améliorant – est exploré. Tous les voisins non dominés par la solution courante sont ajoutés à un ensemble de travail. Lorsque tout le voisinage d'une solution a été visité, celle-ci est marquée comme visitée. Lorsque toutes les solutions sélectionnées ont été visitées, l'ensemble de travail est ajouté à l'archive : les solutions dominées sont supprimées de l'archive, les solutions non dominées y sont ajoutées et marquées comme non visitées. L'algorithme s'arrête naturellement lorsque l'archive des solutions courantes ne contient plus de solution non visitée.

3.1.3.1 Choix de la solution finale

Notre modélisation de MOCA-I est fondée sur une modélisation multi-objectif, ce qui implique que l'algorithme utilisé va générer un ensemble de solutions de compromis : le décideur peut ainsi obtenir des solutions avec une forte *confiance* et une *sensibilité* faible, des solutions de faible *confiance* avec une forte *sensibilité* ou encore des solutions avec un nombre de termes élevé. Il choisira la solution qui correspond le plus à ses besoins. Cette souplesse au niveau des solutions proposées peut cependant s'avérer être un problème lorsqu'une seule solution est nécessaire : le décideur peut disposer d'un temps limité, ce qui ne lui donne pas la possibilité de choisir une solution parmi celles proposées. Dans le contexte des essais cliniques, le décideur sera amené à choisir une solution pour chaque critère d'inclusion ; l'idéal est donc de simplifier cette étape en proposant une solution par défaut et en laissant la possibilité d'adapter le compromis selon les besoins. De nombreuses stratégies sont possibles pour le choix de la solution ou la génération d'une solution à partir de l'ensemble des solutions obtenues. Afin de pouvoir se comparer à la littérature, dans la suite de ce chapitre nous utiliserons une stratégie basique, qui consiste à choisir la solution qui propose la meilleure *f-mesure* sur le jeu d'apprentissage, la *f-mesure* étant utilisée comme critère d'évaluation des performances de la classification partielle. D'autres stratégies plus élaborées feront l'objet d'une étude approfondie dans le chapitre 5.

Algorithme 1 Initialisation d'une règle

/* créée une règle à partir d'une observation positive */
Sélectionne une observation aléatoire Obs_i qui vérifie la prédiction
Créée une règle vide r
for valeur $v_i \in Obs_i$ **do**
  r.ajouterTerme(recupererAttribut(i),v_i)
end for
/* réduit la taille de la règle ; minimum 2 termes et maximum 9 */
taille = random(2,9)
while $r.taille() <$ taille **do**
  $r.supprimeTermeAleatoire()$
end while
/* ajoute du bruit (plus de bruit sur les règles plus longues) */
for $i \in max(\frac{r.taille()}{3}, 1)$ **do**
  $r.ajouterTerme(recupererTermeAleatoire())$
end for
/* supprime des termes */
for $i \in max(\frac{r.taille()}{3}, 1)$ **do**
  $r.supprimeTermeAleatoire()$
end for

Algorithme 2 algorithme DMLS implémenté dans MOCA-I

```
archive ← genereEnsembleReglesAleatoires(100)
while archive.nbNonVisites() > 0 do
  ensembleReglesAVisiter ← selectionner(archive)
  resultats ← []
  /* Visite le voisinage des ensembles de règles sélectionnés */
  for RS_courant ∈ ensembleReglesAVisiter do
    voisins ← RS_courant.genereVoisins()
    for RS_voisin ∈ voisins do
      if RS_courant ⊁ RS_voisin then
        RS_voisin.visite ← false
        resultats.ajoute(RS_voisin)
      end if
    end for
    RS_courant.visite ← true
  end for
  /* Met à jour l'archive */
  for RS_resultat ∈ resultats do
    for RS_archive ∈ archive do
      if RS_resultat ≻ RS_archive then
        archive.enlever(RS_archive)
      end if
      if RS_archive ≻ RS_resultat then
        RS_resultat.estDomine ← true
      end if
    end for
    if !RS_resultat.estDomine then
      archive.ajouter(RS_resultat)
    end if
  end for
end while
```

3.1.3.2 Les différents composants de DMLS

Précédemment, nous avons vu que certaines parties de l'algorithme DMLS peuvent faire l'objet de stratégies différentes. Lors du choix des solutions à explorer, il est possible de choisir une seule solution à explorer (1), tout comme il est possible de sélectionner l'ensemble des solutions courantes pour une exploration exhaustive ($*$). Lors de l'exploration, toutes les solutions peuvent être explorées ($*$) mais il est également possible de s'arrêter à la première solution qui domine la solution courante ($1_{\succ}$). Ces différentes stratégies vont influer sur le comportement de DMLS : les stratégies 1 et $1 \succ$ seront sûrement plus rapides que les stratégies exhaustives $*$, et les solutions obtenues seront de qualité différente. Dans la suite, nous étudierons les combinaisons des 4 stratégies présentes dans la table 3.2. Pour chacun des composants (choix de la solution à explorer et exploration du voisinage) les stratégies disponibles sont énumérées, ainsi que leur abréviation qui sera utilisée dans la suite. Ces stratégies feront l'objet d'une comparaison poussée afin de déterminer la combinaison la plus efficace.

Table 3.2: Composants DMLS étudiés

Composant	Stratégie	Abr.
choix de la solution à explorer	1 aléatoire	1
	exhaustif	*
Exploration du voisinage	1er améliorant	$1_{\succ}$
	exhaustif	*

3.2 Apport de la multi-objectivisation

Nous avons vu que le problème de classification sur données asymétriques pouvait être modélisé sous forme d'un problème multi-objectif, en présentant MOCA-I. Cependant, la modélisation multi-objectif est plus complexe ; il est juste de s'interroger sur les avantages qu'elle apporte par rapport à une représentation mono-objectif. Dans cette section, nous comparons statistiquement MOCA-I à une approche mono-objective où les objectifs *confiance* et *sensibilité* sont agrégés au sein d'une même mesure : la *f-mesure*, qui permet donc de maximiser conjointement la *confiance* et la *sensibilité*.

3.2.1 Motivations

Dans le chapitre 2 de nombreuses approches ont été présentées pour résoudre le problème de classification. Ces approches sont variées : mono-objectives comme *C4.5*, *DT-Oblique*, etc. ou multi-objectives comme *DT-GA* ou *GAssist*. Parmi les approches multi-objectives, deux stratégies sont retrouvées pour gérer les objectifs : la dominance Pareto ou l'agrégation de tous les objectifs au sein d'une même fonction. Au sein d'une agrégation, des pondérations sont souvent utilisées pour prioriser certains objectifs. Cependant elles sont difficiles à déterminer, c'est

pourquoi l'algorithme *GAssist* propose une fonctionnalité d'autoréglage des pondérations [6]. Lorsqu'une agrégation est utilisée, l'algorithme réagit comme un algorithme mono-objectif et devient plus sensible aux *optima* locaux, entraînant de moins bons résultats.
Dans la littérature, de nombreux travaux ont étudié l'impact de l'agrégation, ou de la multi-objectivisation. Ainsi, Knowles *et al.* ont montré sur le problème du voyageur de commerce que la décomposition en plusieurs objectifs – la multi-objectivisation – améliorait les résultats [68]. Il n'est pas toujours possible de décomposer en plusieurs objectifs ; Jensen *et al.* ont proposé le concept d'objectif auxiliaire [1] [61] pour élargir le champ d'applications de la multi-objectivisation. Le principe est d'ajouter un objectif additionnel pour guider la recherche et éviter les *optima* locaux ; il est préférable de choisir un objectif complémentaire à l'objectif initial. L'approche a permis d'améliorer les résultats sur des problèmes de planification. D'autres auteurs ont ensuite étendu cette étude à d'autres problèmes : problème de sac-à-dos [57], de tournées de véhicules [110] ou encore de la prédiction de la structure des protéines [49] ; dans tous les cas la multi-objectivisation a amélioré les résultats. Deb *et al.* ont transformé un problème à 3 objectifs en un problème bi-objectif [30], en agrégeant deux objectifs en un seul. L'approche à 3 objectifs s'est avérée plus efficace que l'approche à 2 objectifs. Nous allons donc vérifier si ces travaux peuvent également s'appliquer à notre problème de classification, en comparant les résultats obtenus par MOCA-I et la recherche locale multi-objective (DMLS) aux résultats obtenus par une recherche locale mono-objectif (LS) fondée sur les mêmes composants que MOCA-I mais optimisant la *f-mesure*.

3.2.2 Algorithmes étudiés

Dans cette étude, MOCA-I – qui optimise la *confiance*, la *sensibilité* et le *nombre de termes* va être comparé à une version mono-objectif de MOCA-I fondée sur l'optimisation de la *f-mesure*. Les deux versions de MOCA-I partagent la même représentation de solution et le même voisinage présentés précédemment, sont toutes deux fondées sur des algorithmes de recherche locale et diffèrent uniquement au niveau de la fonction objectif. Dans la suite, la version multi-objectif de MOCA-I sera notée DMLS car elle est fondée sur l'algorithme DMLS. La version mono-objectif est fondée sur l'algorithme du *Hill Climbing* et sera notée LS (pour Local Search). L'algorithme 3 détaille son implémentation. Il démarre d'une règle initiale générée avec la procédure d'initialisation proposée précédemment (algorithme 1). Le voisinage de cette règle est ensuite généré ; dès qu'un voisin améliorant la solution courante est trouvé, il remplace la solution courante et on recommence l'exploration à partir de la solution courante. La ligne 5 présente une adaptation à l'algorithme usuel : nous autorisons à choisir un voisin avec un score équivalent si celui-ci diminue la taille de la règle. Cette adaptation s'est avérée nécessaire pour permettre à l'algorithme de progresser même lorsque la règle n'est vérifiée sur aucun individu, ce qui arrive fréquemment dans les données creuses, comme des données PMSI. L'algorithme stoppe naturellement lorsqu'aucun voisin améliorant n'a été trouvé.

1. *helper objective*

Algorithme 3 Hill Climbing (LS)

```
1: R_courante ← générer_règle()
2: évaluer_règle(R_courante)
3: while R_voisine ← générer_voisin_aléatoire(R_courante) do
4:    évaluer_règle(R_voisine)
5:    if R_voisine.score() >R_courante.score() or ( R_voisine.score() == R_courante.score() and
      R_voisine.taille() <R_courante.taille()) then
6:       R_courante ← R_voisine
7:    end if
8: end while
9: return R_courante
```

3.2.3 Protocole

DMLS et LS vont être comparés en utilisant les recommandations détaillées précédemment dans le chapitre 2 section 2.3. Les performances des deux algorithmes seront évaluées sur les 10 jeux de données asymétriques, présentant des degrés d'asymétrie variables. Les tests statistiques de Friedman et Iman-Davenport seront effectués pour déterminer si les algorithmes sont équivalents statistiquement du point de vue de la *f-mesure* sur l'ensemble des 10 jeux de données. Des tests post-hoc seront ensuite réalisés afin de déterminer lequel des algorithmes est le plus performant ; le test de *Wilcoxon* sera associé à la correction de *Bergmann et Hommel*. Une validation croisée (k=5) est utilisée pour évaluer les algorithmes et ils seront exécutés 5 fois par partition avec des populations initiales différentes ; chacun des algorithmes sera donc exécuté 25 fois par jeu de données. La *f-mesure* sur le jeu de test sera utilisée pour évaluer la performance de la classification partielle et sa capacité à généraliser sur des données inconnues. Nous avions vu précédemment que DMLS comportait plusieurs composants pouvant utiliser différentes stratégies, les 4 variantes de DMLS – $(1 \cdot 1_{\succ})$,$(* \cdot 1_{\succ})$,$(* \cdot *)$ et $(1 \cdot *)$ – seront utilisées pour la comparaison. DMLS génère un ensemble de solutions de compromis ; afin d'obtenir une seule solution nous sélectionnerons la solution de compromis qui obtient la meilleure *f-mesure* sur le jeu d'apprentissage. Tous les tests sont réalisés sur un ordinateur Xeon 3500 quad core disposant de 8 GB ram, sous Ubuntu 12, avec gcc 4.6.1. L'implémentation du Hill Climbing et du DMLS proviennent du framework ParadisEO, qui est un framework objet de type boîte blanche dédié à la conception de méta-heuristiques [19].
Le critère d'arrêt de chacun des algorithmes a été choisi afin de rester juste : certaines versions de DMLS sont plus complexes que d'autres, et DMLS est plus complexe que LS et nécessitera plus de temps avant d'atteindre l'optimum local. Le temps d'exécution de DMLS $(* \cdot *)$ – qui est la version la plus complexe car exhaustive – est utilisé comme critère d'arrêt de référence. Les algorithmes qui auront atteint l'optimum local avant d'avoir atteint le critère d'arrêt de référence seront redémarrés à partir d'une autre solution initiale, leur laissant une chance d'améliorer leurs résultats. Le critère d'arrêt temporel permet de pénaliser les DMLS qui passeraient trop de temps à gérer leur archive. Comme LS ne gère pas d'archive, son critère d'arrêt est

adapté : il est autorisé à effectuer le même nombre d'évaluations que l'algorithme de référence DMLS ($* \cdot *$). La table 3.3 montre le nombre moyen de redémarrages effectués par chacun des algorithmes (DMLS ou LS) sur leurs 25 exécutions. Nous pouvons observer que DMLS ($* \cdot *$) ne redémarre jamais tandis que les autres versions de DMLS tout en disposant du même temps arrivent à redémarrer plusieurs fois. Le nombre de redémarrages de LS est plus conséquent car celui-ci ne doit gérer qu'une seule solution, ce qui lui permet d'être plus rapide. La table 3.4 complète la table précédente, en précisant pour chacun des algorithmes le temps moyen d'une exécution sur chacun des 10 jeux de données. On observe que pour un même nombre d'évaluations, l'algorithme SO-LS est souvent plus rapide que les algorithmes issus de DMLS, à l'exception des jeux de données *ye*3 (*yeast*3*d*) et *luc* (*lucap*0) pour lesquels il a besoin de plus de temps pour atteindre l'optimum. Les DMLS ont tous des temps d'exécution différents, ce qui est cohérent avec le critère d'arrêt qui a été défini.

Table 3.3: Nombre moyen de redémarrages sur 25 exécutions.

	DMLS				SO-LS
	$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$	
hab	0.00	0.00	0.88	0.00	258.32
ec1	1.52	1.32	1.16	0.00	128.08
ec2	2.04	1.92	1.28	0.00	89.60
ye3	0.72	0.88	0.92	0.00	272.40
ab9	0.48	0.52	1.00	0.00	690.68
ye2	1.88	1.64	1.20	0.00	140.04
ab1	0.88	0.76	0.96	0.00	765.52
a1a	0.68	0.60	1.16	0.00	329.56
luc	0.76	0.68	1.24	0.00	245.20
w1a	1.04	1.00	1.12	0.00	293.56

3.2.4 Résultats et discussion

La table 3.5 donne la *f-mesure* moyenne et son écart-type obtenus par chacun des algorithmes pour chacun des jeux de données, à la fois sur l'apprentissage et sur le test, sur les 25 exécutions. Chaque paire de lignes représente un jeu de données ; la première ligne détaille le score obtenu sur le jeu d'apprentissage tandis que la seconde détaille les résultats sur le test. À titre informatif, la dernière colonne donne les résultats obtenus par *C4.5-CS*, qui est un des algorithmes de la littérature qui donne les meilleurs résultats sur ce type de données. Ce tableau permet tout d'abord de remarquer que LS est moins sujet au sur-apprentissage que DMLS : l'écart entre les résultats sur l'apprentissage et le test est moins important pour LS, montrant qu'il a moins tendance à coller aux données d'apprentissage. Cependant, on observe que LS n'obtient jamais les meilleurs résultats, qui sont toujours obtenus par une version de DMLS ou par *C4.5-CS*. En général DMLS semble obtenir une *f-mesure* 5 à 15% supérieure

Table 3.4: Temps moyen d'exécution (secondes) en 25 exécutions.

	DMLS				SO-LS
	$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$	
hab	23.51	23.54	23.77	23.22	12.59
ec1	12.27	12.27	12.40	12.22	12.11
ec2	5.59	5.60	5.72	5.57	4.99
ye3	130.57	130.37	131.65	130.02	176.21
ab9	125.12	124.95	125.68	124.50	73.23
ye2	5.90	5.90	6.07	5.86	5.10
ab1	532.81	532.86	537.69	531.90	501.90
a1a	319.78	320.16	333.06	318.69	278.25
luc	474.37	474.55	489.64	473.01	671.42
w1a	233.54	233.61	242.66	233.19	199.76

Table 3.5: F-mesure moyenne et écart-type (jeu d'apprentissage et test) sur 25 exécutions.

	DMLS				LS	C4.5- CS
	$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$		
hab	0.62 ± 0.02	0.62 ± 0.03	0.62 ± 0.02	**0.63 ± 0.03**	0.59 ± 0.07	0.43 ± 0.01
tst	**0.41 ± 0.11**	0.41 ± 0.12	0.40 ± 0.09	0.38 ± 0.11	0.39 ± 0.14	0.40 ± 0.03
ec1	**0.91 ± 0.01**	0.91 ± 0.01	0.90 ± 0.01	0.90 ± 0.01	0.77 ± 0.22	0.76 ± 0.05
tst	0.76 ± 0.06	0.77 ± 0.07	0.77 ± 0.04	**0.78 ± 0.05**	0.72 ± 0.15	0.77 ± 0.08
ec2	0.94 ± 0.02	**0.94 ± 0.02**	0.93 ± 0.02	0.93 ± 0.02	0.83 ± 0.22	0.53 ± 0.08
tst	0.82 ± 0.07	0.81 ± 0.06	**0.82 ± 0.06**	0.81 ± 0.09	0.78 ± 0.18	0.47 ± 0.06
ye3	0.81 ± 0.01	0.81 ± 0.01	**0.81 ± 0.01**	0.81 ± 0.01	0.52 ± 0.31	0.37 ± 0.03
tst	0.74 ± 0.05	0.73 ± 0.05	**0.74 ± 0.05**	0.73 ± 0.06	0.62 ± 0.27	0.34 ± 0.02
ab9	0.68 ± 0.03	0.67 ± 0.03	**0.70 ± 0.03**	0.69 ± 0.03	0.58 ± 0.13	0.59 ± 0.04
tst	0.31 ± 0.21	0.31 ± 0.21	**0.34 ± 0.22**	0.33 ± 0.20	0.31 ± 0.18	0.26 ± 0.24
ye2	0.89 ± 0.03	**0.90 ± 0.02**	0.88 ± 0.04	0.89 ± 0.03	0.47 ± 0.34	0.64 ± 0.06
tst	0.51 ± 0.18	0.49 ± 0.15	**0.52 ± 0.18**	0.51 ± 0.17	0.43 ± 0.28	0.35 ± 0.08
ab1	0.32 ± 0.02	**0.32 ± 0.04**	0.31 ± 0.03	0.30 ± 0.03	0.15 ± 0.13	0.26 ± 0.05
tst	0.02 ± 0.05	0.03 ± **0.07**	0.02 ± 0.05	0.03 ± 0.05	0.01 ± 0.04	0.03 ± 0.03
a1a	0.66 ± 0.01	0.66 ± 0.01	0.65 ± 0.01	0.65 ± 0.00	0.65 ± 0.02	**0.77 ± 0.02**
tst	0.62 ± 0.02	0.62 ± 0.02	0.63 ± 0.03	**0.63 ± 0.02**	0.60 ± 0.02	0.62 ± 0.01
luc	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.83 ± 0.04	**0.94 ± 0.01**
tst	0.82 ± 0.02	0.81 ± 0.02	0.81 ± 0.02	0.82 ± 0.02	0.82 ± 0.02	**0.83 ± 0.02**
w1a	0.69 ± 0.03	**0.69 ± 0.03**	0.69 ± 0.03	0.69 ± 0.03	0.60 ± 0.04	0.20 ± 0.07
tst	0.49 ± 0.20	**0.49 ± 0.20**	0.48 ± 0.19	0.47 ± 0.19	0.44 ± 0.19	0.13 ± 0.03

à celle obtenue par LS sur le jeu de test. Aucune version de DMLS ne semble se démarquer d'une autre sur l'ensemble des jeux de données. Sur les premiers jeux de données, l'écart-type plus élevé de LS semble indiquer qu'il est moins robuste d'une exécution à l'autre. Les jeux de données *ab9*, *ye2* et *w1a* semblent plus complexes à gérer car à la fois DMLS et LS obtiennent un écart-type élevé. La figure 3.5 permet d'obtenir un aperçu des résultats obtenus sur

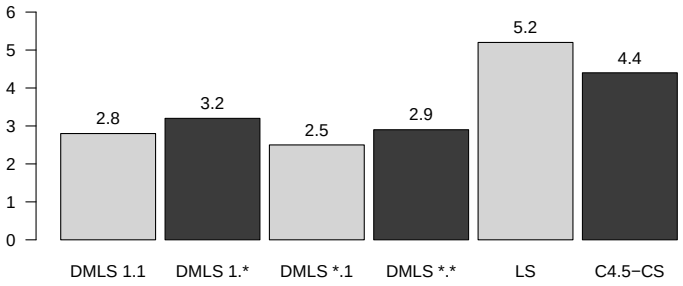


Figure 3.5: Rangs moyens sur les données de test.

le test sur l'ensemble des jeux de données. Pour chaque algorithme, elle donne le rang moyen obtenu sur l'ensemble des jeux de données. Par exemple, DMLS $(* \cdot 1_{\succ})$ a un rang moyen de (4+2+1+1+1+1+5+2+5+3)/10=2.5. On peut ainsi observer que LS obtient un rang moyen de 5.2 – largement supérieur à celui des différentes versions de DMLS – laissant supposer qu'il est souvent dépassé par DMLS sur l'ensemble des jeux de données. Aucune version de DMLS ne semble se démarquer ; chacune des versions obtient un rang moyen similaire.
Une étude plus poussée permet de valider statistiquement ces observations. Tout d'abord, les tests de Friedman et Iman-Davenport nous permettent de rejeter l'hypothèse nulle H_0 affirmant que les algorithmes sont identiques du point de vue des rangs moyens sur le test. La table 3.6

Table 3.6: Valeurs des tests de Friedman and Iman-Davenport avec α=0.05.

Test	Valeur critique	Valeur	H_0
Friedman	11.0704978	16.1142857	Rejetée
Iman-Davenport	2.42208546	4.27993255	Rejetée

donne les résultats obtenus par ces deux tests, ainsi que la valeur seuil à partir de laquelle nous pouvons rejeter H_0 avec $\alpha = 0.05$. Les deux valeurs obtenues sont toutes deux supérieures au seuil, ce qui nous permet de rejeter H_0 : les algorithmes ont des rangs moyens statistiquement différents. Nous pouvons maintenant réaliser une analyse post-hoc afin de déterminer précisément quels algorithmes sont statistiquement plus performants. La table 3.7 donne les résultats obtenus par le test de Wilcoxon associé à une correction de Bergmann-Hommel, pour la comparaison deux à deux de tous les algorithmes. Chaque colonne effectue la comparaison d'un algorithme donné à tous les autres, indiquant s'il est statistiquement meilleur ($\succ$), moins bon ($\prec$) ou équivalent ($\equiv$). On observe que LS est statistiquement moins performant que toutes les versions de DMLS. En revanche, les données expérimentales étudiées ne permettent pas

Table 3.7: Comparaison Post-hoc N x N avec test de Wilcoxon et correction de Bergmann-Hommel.

		DMLS				LS
		$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$	
DMLS	$1 \cdot 1_{\succ}$	$\times$	$\equiv$(0.4795)	$\equiv$(0.6714)	$\equiv$(0.8875)	$\prec$(0.0019)
	$1 \cdot *$	$\equiv$(0.4795)	$\times$	$\equiv$(0.2579)	$\equiv$(0.5716)	$\prec$(0.0162)
	$* \cdot 1_{\succ}$	$\equiv$(0.6714)	$\equiv$(0.2579)	$\times$	$\equiv$(0.5716)	$\prec$(0.0004)
	$* \cdot *$	$\equiv$(0.8875)	$\equiv$(0.5716)	$\equiv$(0.5716)	$\times$	$\prec$(0.0030)
LS		$\succ$(0.0019)	$\succ$(0.0162)	$\succ$(0.0004)	$\succ$(0.0030)	$\times$

de déterminer statistiquement si une version de DMLS est supérieure aux autres. De la même manière, les données n'ont pas permis de déterminer une relation entre DMLS et *C4.5-CS*, d'où son absence du tableau.

LS est statistiquement moins performant que DMLS, ce qui montre que la multi-objectivisation est plus efficace qu'une recherche mono-objectif, dans le cadre de la classification partielle sur données asymétriques. Ces résultats sont cohérents avec les résultats obtenus dans la littérature par la multi-objectivisation sur des problèmes différents. Une perspective de cette analyse pourrait consister à compléter cette étude par une analyse des différences en termes de diversité des solutions proposées par LS et DMLS. Une autre perspective serait de compléter ce travail par une analyse de la dynamique de LS et DMLS, en analysant leur comportement lorsque la recherche est stoppée à des temps d'exécution différents.

3.3 Étude de l'influence du paramétrage de DMLS sur les résultats

Précédemment nous avons vu que DMLS comportait un composant d'exploration du voisinage et un composant permettant de choisir une ou des solutions à explorer au sein de l'archive des solutions courantes. Ces deux composants peuvent faire l'objet de stratégies différentes, concernant par exemple le nombre de solutions à visiter ou des explorations partielles du voisinage. D'autre part, certains paramètres peuvent agir sur le comportement du DMLS, comme la taille de l'archive de départ ou la stratégie de gestion de l'archive. Dans cette section, nous étudions l'impact de ce paramétrage. Une fois un bon paramétrage déterminé, nous étudierons les différentes stratégies présentes dans les composants de DMLS.

3.3.1 Influence des paramètres

Du point de vue de la modélisation MOCA-I, deux paramètres peuvent avoir leur importance : la discrétisation appliquée pour préparer les données et le nombre de règles maximales autorisées dans un ensemble de règles. La discrétisation consiste à convertir un attribut quantitatif en un attribut qualitatif. Du point de vue du DMLS, deux autres paramètres sont étudiés : la taille de l'archive de départ (taille de la population initiale) et la taille maximale de l'archive. En effet, les premières expérimentations ont montré que, sur certains jeux de données, il est parfois nécessaire de limiter la taille de l'archive ; nous souhaitons ici déterminer si la limitation de la taille de l'archive a un impact sur les résultats. La table 3.8 donne un résumé des paramètres étudiés avec pour chacun les valeurs qui seront testées.

Table 3.8: Étude détaillée de MOCA-I - paramètres étudiés

Paramètres	Valeurs étudiées
Discrétisation (nombre de partitions)	5
	10
	20
Nombre de règles	5
	10
	20
Taille maximale de l'archive	100
	300
	500
Taille de la population initiale	50
	100
	200

3.3.1.1 Protocole expérimental

Chacun des 4 paramètres présentés précédemment va être étudié individuellement. Pour chaque étude, seules les valeurs du paramètre étudié vont varier. Les autres paramètres seront fixés à des valeurs par défaut utilisées dans une première version de MOCA-I [60] :

Nombre de règles maximum 10

Taille maximum de l'archive 500

Taille de l'archive initiale 100

De plus, l'étude sera réalisée pour chacune des versions de DMLS (DMLS $(1 \cdot 1_{\succ})$, DMLS $(1 \cdot *)$, DMLS $(* \cdot 1_{\succ})$ et DMLS $(* \cdot *)$) afin de vérifier si les paramètres se comportent de manière similaire sur chacune des versions.

Le protocole est similaire à celui présenté précédemment pour l'analyse de l'impact de la multi-objectivisation. Toutes les configurations – combinaison d'un paramètre et d'une version de DMLS – sont étudiées sur les 10 jeux de données asymétriques présentés précédemment dans la section 2.3.1.3 (chapitre 2). Chacune des configurations est exécutée 25 fois (5 graines aléatoires en validation croisée à 5 partitions) sur chacun des 10 jeux de données. Comme précédemment, nous cherchons la configuration la plus robuste, c'est-à-dire celle qui donne les meilleurs résultats sur l'ensemble des jeux de données. Les performances sont évaluées à l'aide de la *f-mesure* obtenue sur le jeu de test. Les comparaisons statistiques seront effectuées avec les tests de Friedman et Iman-Davenport, Holm, et Bergman et Hommel.

Chacune des configurations dispose à chaque exécution d'un nombre limité d'évaluations de voisinage qui est commun à toutes les configurations testées. Afin de proposer une comparaison équitable, les configurations qui s'arrêteront avant la limite auront droit à un ou plusieurs redémarrages à partir d'une autre population initiale, leur permettant d'améliorer leurs résultats : seuls les meilleurs résultats obtenus sur l'ensemble des redémarrages seront conservés. Ce nombre d'évaluations est déterminé de manière expérimentale, qui permet à un maximum de configurations d'atteindre l'optimum local, tout en pénalisant les configurations trop lentes à converger. Il est estimé de la manière suivante : chaque configuration est exécutée 25 fois (validation croisée 5 partitions $\times$ 5 graines aléatoires) et s'arrête dès qu'elle atteint l'optimum local ; le nombre moyen d'évaluations de voisinage nécessaires pour atteindre l'optimum local est collecté ; enfin le nombre moyen d'évaluations le plus élevé est choisi comme critère d'arrêt, ce qui permet à chaque configuration d'avoir une chance d'atteindre l'optimum local. Les jeux de données utilisés étant de complexités différentes, le critère d'arrêt sera différent pour chacun des jeux de données.

3.3.1.2 Résultats et discussion

Nous détaillons ici les résultats obtenus pour l'ensemble des paramètres étudiés. L'étude commence par analyser l'impact de la discrétisation. Nous étudions ensuite l'impact de la taille

des solutions, et finalement les paramètres relatifs à l'archive : taille maximale de l'archive et taille de l'archive initiale.

Influence de la discrétisation Parmi les 10 jeux de données asymétriques étudiés, 6 comportent des attributs quantitatifs (*haberman*, *ecoli*1, *ecoli*2, *yeast*3, *abalone*9*vs*18, *yeast*2*vs*8 et *abalone*19). Étant donné que MOCA-I prend en charge uniquement des attributs qualitatifs ou ordinaux (pour répondre aux besoins de l'application), une étape de préparation de ces jeux de données est nécessaire. La discrétisation permet de transformer des attributs quantitatifs en attributs qualitatifs, plusieurs techniques existent. Nous étudierons ici une technique simple qui consiste à diviser chaque attribut numérique en K intervalles de valeurs de tailles identiques. En discrétisant de cette manière un attribut quantitatif compris entre 20 et 50, on obtient un attribut qualitatif avec les valeurs suivantes ($K = 3$) : {"<30","30-40",">40"}. Trois valeurs différentes de K sont utilisées : {5, 10 et 20}, avec la discrétisation implémentée dans le logiciel Weka [1]

Table 3.9: Étude de l'influence de la discrétisation : F-mesure moyenne obtenue sur l'apprentissage et le test sur 25 exécutions, avec une discrétisation en {5, 10, 20} parts.

	k=5				k=10				k=20			
	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$
haberman	0.56	0.56	0.56	0.56	**0.63**	0.63	0.61	0.61	0.59	0.59	0.59	0.59
test	**0.48**	0.47	**0.48**	**0.48**	0.36	0.47	0.44	0.37	0.38	0.32	0.36	0.36
ecoli1	0.83	0.83	0.82	0.82	0.91	**0.91**	0.90	0.90	0.82	0.82	0.82	0.81
test	0.77	0.74	0.76	0.74	0.74	0.74	**0.79**	0.77	0.65	0.67	0.65	0.65
ecoli2	0.78	0.78	0.78	0.78	0.92	**0.93**	0.93	0.91	0.82	0.80	0.77	0.76
test	0.67	0.72	0.70	0.72	0.82	0.80	**0.82**	0.78	0.56	0.58	0.58	0.59
yeast3	0.61	0.63	0.60	0.61	0.81	0.81	**0.81**	0.81	0.70	0.71	0.70	0.70
test	0.56	0.55	0.55	0.54	0.74	0.73	0.74	**0.74**	0.63	0.66	0.65	0.64
abalone9vs18	0.61	0.60	0.59	0.60	**0.71**	0.71	0.70	0.69	0.66	0.68	0.66	0.66
test	0.26	0.25	0.24	0.25	0.43	0.45	**0.46**	0.39	0.30	0.30	0.26	0.31
yeast2vs8	0.77	0.77	0.77	0.77	0.90	0.88	0.87	0.86	0.92	**0.93**	0.92	0.93
test	0.49	0.49	0.41	0.55	0.48	0.56	0.49	0.55	0.56	**0.59**	0.54	0.52
abalone19	0.29	0.28	0.27	0.28	0.32	0.32	0.31	0.28	0.51	**0.52**	0.49	0.48
test	0.00	0.00	0.00	0.00	0.02	0.02	**0.03**	0.00	0.00	0.00	0.02	0.00

Les résultats sont disponibles dans la table 3.9 qui détaille pour chaque valeur de K et chaque DMLS la *f-mesure* moyenne obtenue sur les différents jeux de données, à la fois sur le jeu d'apprentissage (lignes préfixées par le nom du jeu) et de test (lignes préfixées par "test").

1. *weka.filters.unsupervised.attribute.Discretize ; bins=10, findNumBins=true* qui transforme un attribut quantitatif en un attribut qualitatif contenant au maximum k catégories, chacune représentant des intervalles de même valeur.

Sur les jeux de données *ecoli*1, *ecoli*2, *yeast*3 et *abalone*9*vs*18 la discrétisation avec $K = 10$ donne les meilleurs résultats. En particulier sur les jeux *yeast*3 et *abalone*9*vs*18 on remarque que la discrétisation avec $K = 10$ donne des résultats en terme de *f-mesure* 25% à 40% supérieurs à ceux obtenus par la discrétisation avec $K = 5$. Sur le jeu de données *haberman*, c'est la discrétisation $K = 5$ qui donne les meilleurs résultats. La figure 3.6 se concentre sur les perfor-

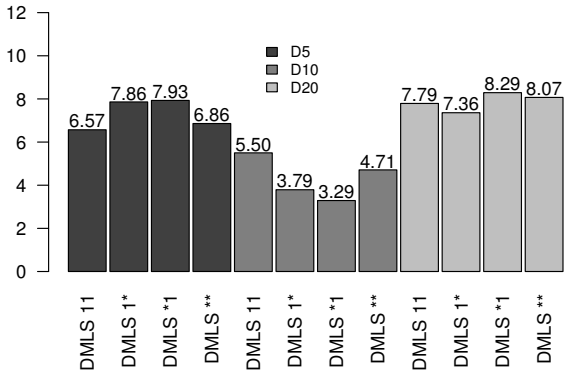


Figure 3.6: Étude de l'influence de la discrétisation : rangs moyens sur les données de test (f-mesure)

mances obtenues sur le test, ce qui – comme indiqué précédemment – montre la capacité d'une configuration à généraliser, c'est-à-dire gérer les données inconnues lors de la phase d'apprentissage. Elle donne les rangs moyens obtenus par chacune des configurations. La discrétisation $K = 10$ obtient un meilleur rang que $K = 5$ et $K = 20$, ce qui indique que cette version est plus performante sur l'ensemble des jeux de données. Ces résultats sont validés par les tests de Friedman et Iman-Davenport, qui indiquent que les configurations testées sont statistiquement différentes en termes de rangs avec $\alpha = 0.05$ car la valeur obtenue dépasse la valeur seuil. Le résultat du test autorise la mise en place de test post-hoc, mais ceux-ci n'ont pas permis d'identifier de différences significatives entre les configurations, sur les données étudiées. Par conséquent, nous choisissons le paramétrage qui donne les meilleurs résultats sur l'ensemble des jeux de données : $K = 10$. Dans la suite, nous travaillerons sur les jeux de données discrétisés avec $K = 10$ afin de s'approcher des caractéristiques des données médicales (données PMSI avec une majorité de données binaires). Les noms des jeux de données discrétisés seront post-fixés par $_d$ dans la suite, par exemple le jeu de données *habeman* discrétisé sera noté $haberman_d$

Influence de la taille maximum des ensembles de règles Chaque ensemble de règles peut contenir un nombre variable de règles. Nous avons comparé 3 limites de nombre de règles : 5 règles, 10 règles ou 20 règles. La table 3.10 donne les *F-mesures* moyennes obtenues par chacune des configurations de DMLS, pour chaque jeu de données. Cette table permet d'observer que la *f-mesure* moyenne varie peu d'un paramétrage à un autre, avec au mieux une différence de

Table 3.10: Étude de l'influence de la taille des ensembles de règles : f-mesure moyenne sur les données d'apprentissage et de test, sur 25 exécutions, en utilisant des ensembles de règles composés d'au maximum {5, 10, 20} règles.

	5 règles				10 règles				20 règles			
	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$
$haberman_d$	0.61	0.60	0.60	0.60	0.63	0.63	0.61	0.61	**0.63**	0.63	0.61	0.61
test	0.46	0.40	0.39	0.38	0.36	**0.47**	0.44	0.37	0.38	**0.47**	0.44	0.37
$ecoli1_d$	0.90	0.90	0.90	0.90	0.91	**0.91**	0.90	0.90	0.91	**0.91**	0.90	0.90
test	0.79	0.79	0.79	**0.81**	0.74	0.74	0.79	0.77	0.74	0.74	0.79	0.77
$ecoli2_d$	0.92	**0.93**	0.93	0.91	0.92	**0.93**	0.93	0.91	0.92	**0.93**	0.93	0.91
test	0.82	0.79	**0.82**	0.78	0.82	0.80	**0.82**	0.78	0.82	0.80	**0.82**	0.78
$yeast3_d$	0.81	0.81	0.81	0.81	0.81	0.81	**0.81**	0.81	0.81	0.81	**0.81**	0.81
test	0.72	0.73	0.74	0.74	0.74	0.73	**0.74**	0.74	0.74	0.73	0.74	**0.74**
$abalone9vs18_d$	0.55	0.55	0.55	0.54	0.71	0.71	0.70	0.69	**0.72**	0.71	0.70	0.70
test	0.32	0.29	0.35	0.31	0.43	0.45	**0.46**	0.39	0.42	0.44	0.41	0.36
$yeast2vs8_d$	0.89	0.89	0.88	0.86	**0.90**	0.88	0.87	0.86	**0.90**	0.88	0.87	0.86
test	0.49	0.50	0.49	**0.59**	0.48	0.56	0.49	0.55	0.48	0.56	0.49	0.55
$abalone19_d$	**0.32**	0.31	0.31	0.29	0.32	0.32	0.31	0.28	0.32	0.32	0.31	0.28
test	0.02	0.03	**0.06**	0.00	0.02	0.02	0.03	0.00	0.02	0.02	0.03	0.00
$a1a$	0.66	0.66	0.65	0.65	0.66	**0.67**	0.66	0.65	0.66	**0.67**	0.66	0.65
test	0.62	0.63	0.62	0.61	0.62	0.62	0.62	**0.63**	0.62	0.62	0.62	0.63
$lucap0$	0.84	0.84	0.83	0.83	0.84	**0.84**	0.83	0.83	0.84	**0.84**	0.83	0.83
test	**0.82**	0.82	0.81	0.82	**0.82**	0.82	0.81	0.82	**0.82**	0.82	0.81	0.82
$w1a$	0.54	0.54	0.54	0.54	0.70	0.69	0.69	0.69	0.76	0.77	**0.77**	0.74
test	0.44	0.43	0.36	0.44	0.48	**0.50**	0.50	0.47	0.46	0.42	0.49	0.47

l'ordre de 5%. Sur les jeux de données $ecoli2_d$, $yeast3_d$, $a1a$ et $lucap0$ la taille ne semble pas avoir d'influence sur les résultats. En revanche sur les jeux de données $abalone9vs18_d$, $haberman_d$ et $w1a$ limiter la taille à 5 semble moins avantageux que de la limiter à 10. Cela indique que la taille de l'ensemble de règles peut être dépendante de la complexité du jeu de données étudié : imposer une limite trop restrictive peut dégrader les performances sur ce type de jeux de données. À l'inverse, une limite trop large ne semble pas avoir d'impact sur la *F-mesure* sur les jeux de données plus simples ($ecoli2_d$, $yeast3_d$, $a1a$, $lucap0$). La figure 3.7 donne les rangs moyens obtenus sur les données de test, pour chaque version de DMLS et chaque limite : {5, 10, 20}. Les rangs moyens sont similaires, ce qui est confirmé par les tests de Friedman et Iman-Davenport qui indiquent que les rangs moyens sont statistiquement équivalents avec $\alpha = 0.05$. Nous choisissons donc le paramétrage qui obtient le meilleur rang moyen sur le test : la limite de 10 qui obtient un rang moyen légèrement meilleur que la limite de 20. Un autre argument est que le voisinage d'une solution pouvant comporter jusqu'à 20 règles est plus grand, ce qui va rendre l'algorithme de recherche plus long à converger. La limite de 5 est éliminée car elle s'est

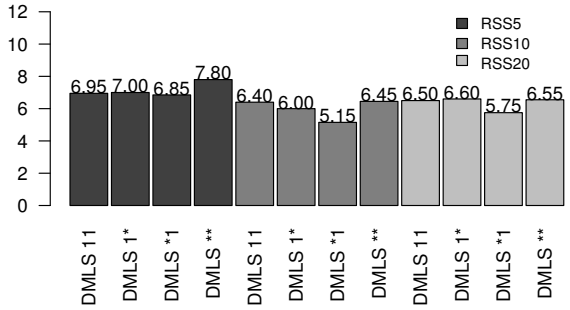


Figure 3.7: Étude de l'influence de la taille des ensembles de règles : rangs moyens obtenus à partir de la *f-mesure* sur les données de test

montrée plus faible sur certains jeux de données. Il est à noter que la taille limite de 10 règles s'est avérée la plus intéressante sur les 10 jeux de données étudiés mais il s'avère important de vérifier lors de l'utilisation d'autres jeux de données que leur complexité est compatible avec la limite étudiée.

Influence de la limitation de la taille de l'archive Dans cette étude, chaque version de DMLS est exécutée avec 3 limites d'archive différentes : {500, 300, 100}. Lorsque l'archive atteint le nombre de solutions spécifié, il devient uniquement possible d'ajouter des solutions qui dominent au moins une solution présente dans l'archive. Les nouvelles solutions de compromis ne sont pas ajoutées à l'archive car elles n'améliorent pas au moins une des solutions. La table 3.11 donne la *f-mesure* moyenne obtenue sur les données d'apprentissage et de test. Sur les jeux de données $ecoli2_d$, $yeast3_d$, $a1a$ et $w1a$ on remarque une légère diminution de la *f-mesure* – de l'ordre de 14% – lorsque la limite est basse. Sur le jeu de données $abalone9vs18_d$ cette baisse est plus importante (40%). Seul le jeu de données $haberman_d$ montre une amélioration des résultats lorsque la taille maximale autorisée de l'archive diminue. La figure 3.8 montre les rangs moyens obtenus sur les données de test, on observe que la limite la plus élevée (500 solutions) obtient un meilleur rang moyen. Les tests de Friedman et Iman-Davenport ne permettent pas d'affirmer que les configurations étudiées sont statistiquement différentes. Nous privilégions cependant la limite la plus élevée, qui offre le meilleur rang moyen sur les données de test.

Influence de la taille de la population initiale Pour chaque version de DMLS, 3 tailles de population initiale ont été testées : {50, 100, 200}. La table 3.12 détaille la *f-mesure* moyenne obtenue par chacune des versions de DMLS avec les 3 tailles de population initiale. Sur le jeu de données $haberman_d$ les résultats s'améliorent avec l'augmentation de la taille de la population initiale (jusqu'à 42%), à l'exception du DMLS $(* \cdot *)$. Sur les autres jeux de données les différences sont moins perceptibles. Les résultats semblent dépendre de la version de DMLS. Ainsi, sur le jeu de données $yeast2vs8_d$, DMLS $(1 \cdot *)$ obtient les meilleurs résultats avec une

Table 3.11: Étude de l'influence de la limitation de la taille de l'archive : *F-mesure* moyenne sur les données d'apprentissage et de test, sur 25 exécutions avec une limite de {500, 300, 100} ensembles de règles.

	500 ensembles				300 ensembles				100 ensembles			
	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$
$haberman_d$	**0.63**	0.63	0.61	0.61	0.61	0.61	0.60	0.59	0.59	0.58	0.56	0.57
test	0.36	0.47	0.44	0.37	0.43	0.40	0.45	0.46	0.44	**0.48**	0.46	0.47
$ecoli1_d$	0.91	**0.91**	0.90	0.90	0.91	**0.91**	0.90	0.90	0.88	0.88	0.88	0.88
test	0.74	0.74	0.79	0.77	0.74	0.74	0.79	0.77	0.78	0.78	0.77	**0.79**
$ecoli2_d$	0.92	**0.93**	0.93	0.91	0.92	**0.93**	0.93	0.91	0.92	0.93	0.92	0.91
test	0.82	0.80	0.82	0.78	0.82	0.80	0.82	0.78	0.82	0.80	**0.83**	0.78
$yeast3_d$	0.81	0.81	**0.81**	0.81	0.81	0.81	0.80	0.80	0.79	0.80	0.79	0.79
test	0.74	0.73	0.74	0.74	0.74	0.73	0.73	0.75	0.74	0.74	**0.75**	0.75
$abalone9vs18_d$	**0.71**	0.71	0.70	0.69	0.64	0.65	0.62	0.62	0.55	0.53	0.48	0.50
test	0.43	0.45	**0.46**	0.39	0.37	0.35	0.42	0.38	0.29	0.32	0.26	0.32
$yeast2vs8_d$	**0.90**	0.88	0.87	0.86	**0.90**	0.88	0.87	0.86	**0.90**	0.88	0.87	0.86
test	0.48	**0.56**	0.49	0.55	0.48	**0.56**	0.49	0.55	0.48	**0.56**	0.49	0.55
$abalone19_d$	0.32	**0.32**	0.31	0.28	0.29	0.29	0.26	0.27	0.23	0.24	0.23	0.23
test	0.02	0.02	0.03	0.00	**0.03**	0.00	0.00	0.00	0.00	0.00	0.02	0.00
$a1a$	0.66	**0.67**	0.66	0.65	0.66	0.66	0.65	0.65	0.64	0.64	0.63	0.62
test	0.62	0.62	0.62	0.63	0.62	0.62	0.62	**0.64**	0.62	0.62	0.61	0.60
$lucap0$	0.84	0.84	0.83	0.83	**0.84**	0.84	0.83	0.83	0.83	0.83	0.82	0.82
test	**0.82**	0.82	0.81	0.82	0.81	0.81	0.81	0.80	0.82	0.81	0.81	0.82
$w1a$	**0.70**	0.69	0.69	0.69	**0.70**	0.69	0.69	0.69	0.68	0.68	0.65	0.65
test	0.48	**0.50**	0.50	0.47	0.48	**0.50**	0.50	0.47	0.50	0.46	0.43	0.48

taille de la population initiale de 100 ; DMLS $(* \cdot 1_{\succ})$ en revanche obtient ses meilleurs résultats avec une taille de la population initiale de 50. La figure 3.9 donne les rangs moyens obtenus sur les données de test. Les tests de Friedman et Iman-Davenport ne permettent pas une nouvelle fois de déterminer qu'une configuration est statistiquement meilleure qu'une autre. Les rangs moyens nous permettent tout de même de choisir le meilleur paramétrage, qui est variable selon la version de DMLS : une taille de 50 pour DMLS $(* \cdot *)$; 200 for DMLS $(1 \cdot *)$; une taille par défaut de 100 pour les autres.

3.3.2 Etude de l'influence des composants de DMLS

Nous venons d'étudier l'influence des différents paramètres sur le comportement de MOCA-I, ce qui a permis de déterminer le meilleur paramétrage à utiliser. Nous allons maintenant déterminer la meilleure combinaison de composants de DMLS à utiliser. Précédemment nous avons vu que certains paramètres donnaient des résultats dépendants de la version de DMLS, le para-

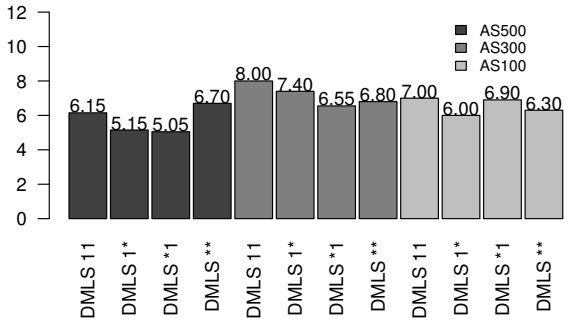


Figure 3.8: Étude de l'influence de la limitation de la taille de l'archive : rangs moyens sur les données d'apprentissage en utilisant la *f-mesure*

Table 3.12: Étude de l'influence de la taille de la population initiale : *F-mesure* moyenne obtenue sur les données d'apprentissage et de test sur 25 exécutions avec une taille de la population initiale de {50, 100, 200} ensembles de règles.

	pop 50				pop 100				pop 200			
	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1_{\succ})$	$(* \cdot *)$
$haberman_d$	0.63	0.62	0.62	0.62	0.63	0.63	0.61	0.61	0.77	**0.80**	0.78	0.61
test	0.33	0.45	0.39	0.35	0.36	0.47	0.44	0.37	0.66	0.64	**0.68**	0.35
$ecoli1_d$	0.91	**0.91**	0.90	0.90	0.91	0.91	0.90	0.90	0.91	0.91	0.91	0.90
test	0.78	0.77	0.74	0.78	0.74	0.74	0.79	0.77	0.77	**0.80**	0.75	0.79
$ecoli2_d$	0.93	**0.94**	0.92	0.90	0.92	0.93	0.93	0.91	0.93	0.93	0.92	0.90
test	**0.84**	0.79	0.82	0.80	0.82	0.80	0.82	0.78	0.84	0.83	0.84	0.83
$yeast3_d$	0.82	**0.82**	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.81
test	0.73	0.74	0.73	**0.75**	0.74	0.73	0.74	0.74	0.73	0.74	0.71	0.73
$abalone9vs18_d$	0.71	0.71	0.70	0.70	**0.71**	0.71	0.70	0.69	0.71	0.71	0.68	0.69
test	0.47	**0.48**	0.41	0.45	0.43	0.45	0.46	0.39	0.43	0.44	0.44	0.47
$yeast2vs8_d$	**0.90**	0.88	0.88	0.86	0.90	0.88	0.87	0.86	0.89	0.89	0.88	0.87
test	0.47	0.50	0.54	0.55	0.48	**0.56**	0.49	0.55	0.55	0.55	0.51	0.48
$abalone19_d$	0.32	0.31	0.31	0.29	0.32	**0.32**	0.31	0.28	0.31	0.31	0.30	0.29
test	0.00	0.03	0.03	**0.03**	0.02	0.02	0.03	0.00	0.02	0.02	0.02	0.03
$a1a$	0.67	0.66	0.66	0.65	0.66	**0.67**	0.66	0.65	0.66	0.66	0.65	0.65
test	0.63	0.63	0.63	0.62	0.62	0.62	0.62	**0.63**	0.62	0.63	0.63	0.62
$lucap0$	0.84	0.84	0.83	0.83	0.84	0.84	0.83	0.83	0.84	**0.84**	0.83	0.83
test	0.82	0.82	**0.82**	0.82	0.82	0.82	0.81	0.82	0.81	0.82	0.81	0.82
$w1a$	0.69	0.69	0.69	0.69	**0.70**	0.69	0.69	0.69	0.70	0.69	0.69	0.70
test	0.50	0.47	0.49	0.49	0.48	0.50	0.50	0.47	0.47	**0.51**	0.49	0.48

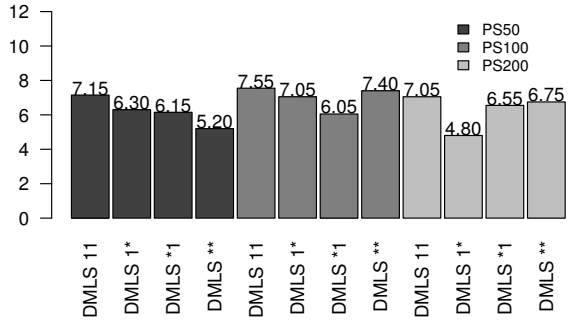


Figure 3.9: Étude de l'influence de la taille de la population initiale : rangs moyens obtenus sur les données de test (*F-mesure*)

métrage sera donc différent selon la version étudiée. Le récapitulatif des paramétrages utilisés et des versions comparées est disponible dans la table 3.13. Différents critères d'arrêt seront

Table 3.13: Paramètres des versions de DMLS étudiées

DMLS	Taille max. Archive	Taille pop. initiale	Nb règles par ensemble de règles
$(1 \cdot 1_{\succ})$	500	100	10
$(1 \cdot *)$	500	200	10
$(* \cdot 1_{\succ})$	500	100	10
$(* \cdot *)$	500	50	10

utilisés, qui vont stopper les versions étudiées avant qu'elles atteignent l'optimum local, ou au contraire leur permettre de redémarrer plusieurs fois. Pour chaque jeu de données, un critère d'arrêt de référence est déterminé en comptant le nombre d'évaluations moyen nécessaires à chacune des versions de DMLS pour atteindre l'optimum local. Le nombre le plus élevé sera utilisé comme référence, afin de permettre à la moins rapide des versions d'avoir une chance d'atteindre l'optimum local. Ce nombre maximum d'évaluations est noté t_{ne}. À partir de ce nombre, plusieurs critères d'arrêt sont définis : $t_{\frac{ne}{2}}$,t_{ne},t_{2ne},t_{4ne}. Le premier permet d'arrêter les versions après $\frac{t_{ne}}{2}$ évaluations, tandis que t_{2ne} et t_{4ne} arrêtent les versions respectivement après $t_{ne} \times 2$ et $t_{ne} \times 4$ évaluations.

La table 3.14 donne la *f-mesure* moyenne obtenue par chaque version de DMLS avec chacun des critères d'arrêt, à la fois sur les données d'apprentissage et de test. Le temps d'exécution va croissant de la gauche vers la droite. Sur tous les jeux de données, on remarque que la *f-mesure* sur les données d'apprentissage s'améliore avec l'augmentation de la durée d'exécution. Les jeux de données $yeast2vs8_d$, $abalone9vs18_d$ et $abalone19_d$ montrent les plus fortes progression de la *f-mesure* sur les données d'apprentissage (jusqu'à 30%), tandis que les autres jeux de données montrent des améliorations plus minimes (de l'ordre de 5%). En revanche, le comportement de

TABLE 3.14: Comparaison des versions de DMLS - F-mesure moyenne sur les données d'apprentissage et de test, pour les critères d'arrêt ($t_{\frac{ne}{2}}$,t_{ne},t_{2ne},t_{4ne})

		$t_{\frac{ne}{2}}$				t_{ne}				t_{2ne}				t_{4ne}			
		$(1 \cdot 1_{\succ})(1 \cdot *)(* \cdot 1_{\succ})(* \cdot *)$				$(1 \cdot 1_{\succ})(1 \cdot *)(* \cdot 1_{\succ})(* \cdot *)$				$(1 \cdot 1_{\succ})(1 \cdot *)(* \cdot 1_{\succ})(* \cdot *)$				$(1 \cdot 1_{\succ})(1 \cdot *)(* \cdot 1_{\succ})(* \cdot *)$			
$haberman_d$		0.61	0.61	0.60	0.59	0.63	0.63	0.61	0.62	0.63	0.63	0.62	0.62	0.64	**0.64**	0.63	0.63
	test	0.37	0.38	0.37	0.39	0.36	0.42	0.41	0.33	0.38	0.42	**0.42**	0.40	0.36	0.41	0.39	0.35
$ecoli1_d$		0.90	0.89	0.87	0.86	0.91	0.91	0.90	0.88	0.91	**0.92**	0.90	0.91	0.92	0.92	0.90	0.92
	test	0.77	0.79	**0.80**	0.78	0.74	0.77	0.80	0.78	0.74	0.76	0.77	0.78	0.76	0.76	0.75	0.76
$ecoli2_d$		0.91	0.90	0.88	0.86	0.92	0.92	0.92	0.89	0.94	0.93	0.93	0.93	0.94	**0.95**	0.94	0.94
	test	0.84	**0.84**	0.81	0.80	0.82	0.83	0.77	0.83	0.80	0.81	0.82	0.84	0.83	0.82	0.82	0.83
$yeast3_d$		0.81	0.81	0.80	0.79	0.81	0.81	0.81	0.80	0.81	0.81	0.81	0.81	**0.82**	0.82	0.82	0.82
	test	0.73	0.72	**0.74**	0.74	0.73	0.73	0.74	0.74	0.74	0.73	0.74	0.74	0.73	0.73	0.74	0.72
$abalone9vs18_d$		0.66	0.66	0.63	0.60	0.71	0.70	0.68	0.66	**0.71**	0.71	0.70	0.69	0.71	0.71	0.71	0.71
	test	0.41	0.40	0.41	0.35	0.41	0.46	0.45	0.40	0.41	0.43	0.48	0.43	0.43	0.43	0.46	**0.50**
$yeast2vs8_d$		0.86	0.86	0.82	0.80	0.88	0.90	0.86	0.84	0.90	0.90	0.88	0.87	0.90	**0.91**	0.90	0.89
	test	0.49	0.54	0.55	**0.60**	0.49	0.50	0.50	0.53	0.55	0.50	0.52	0.48	0.48	0.52	0.53	0.51
$abalone19_d$		0.29	0.28	0.26	0.24	0.31	0.32	0.29	0.28	0.33	0.33	0.31	0.31	**0.34**	0.34	0.32	0.32
	test	0.00	0.02	0.00	0.00	0.02	0.05	0.03	0.03	0.02	0.03	0.03	0.03	**0.11**	0.02	0.03	0.03
$a1a_i$		0.66	0.66	0.65	0.64	0.66	0.66	0.66	0.65	0.66	0.67	0.66	0.65	**0.67**	0.67	0.66	0.66
	test	0.63	0.62	0.61	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.61	0.63	0.63	0.63	**0.63**	0.63
$lucap0$		0.84	0.84	0.83	0.83	0.84	0.84	0.83	0.83	0.84	0.84	0.84	0.83	**0.85**	0.85	0.84	0.84
	test	0.82	0.80	**0.82**	0.81	0.82	0.80	0.82	0.82	0.81	0.81	0.82	0.82	0.81	0.81	0.81	0.81
$w1a_i$		0.68	0.69	0.66	0.62	0.69	0.69	0.69	0.67	**0.70**	0.69	0.69	0.69	0.67	0.70	0.69	0.69
	test	0.50	0.45	0.48	0.45	0.49	0.49	0.47	0.47	0.48	0.52	0.50	0.48	**0.67**	0.50	0.49	0.50

la *f-mesure* sur les données de test est différent ; elle peut se dégrader à mesure que le temps d'exécution augmente, à l'instar de ce qui se passe sur les jeux de données $ecoli1_d$ et $yeast2vs8_d$. Ce comportement peut être une manifestation du phénomène de sur-apprentissage, c'est-à-dire que l'algorithme se spécialise trop sur les données d'apprentissage, au détriment des données de test – qui correspondent à des données inconnues. Contrairement au comportement de la *f-mesure* sur les données d'apprentissage, sur les données de test une augmentation du temps d'exécution ne garantit pas une amélioration de la *f-mesure* : celle-ci peut rester identique, se dégrader ou s'améliorer.

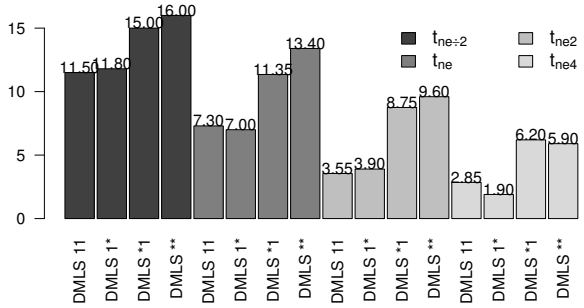


Figure 3.10: Comparaison des versions de DMLS - Rangs moyens obtenus sur les données d'apprentissage, en utilisant divers critères d'arrêt ($t_{\frac{ne}{2}}$,t_{ne},t_{2ne},t_{4ne})

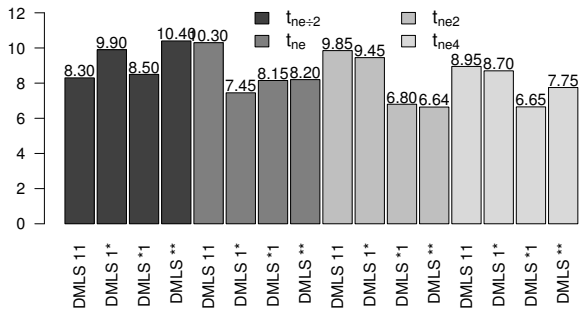


Figure 3.11: Comparaison des versions de DMLS - Rangs moyens obtenus sur les données de test, en utilisant divers critères d'arrêt ($t_{\frac{ne}{2}}$,t_{ne},t_{2ne},t_{4ne})

Les rangs moyens des figures 3.10 et 3.11 donnent un aperçu du comportement de chaque version de DMLS sur l'ensemble des jeux de données. Commençons tout d'abord par analyser les résultats obtenus sur les données d'apprentissage, ce qui permet d'observer la dynamique et la convergence de chaque algorithme. Sur la figure 3.10, on observe que le rang moyen sur

les données d'apprentissage s'améliore lorsque les algorithmes sont autorisés à effectuer plus d'évaluations. t_{4ne} donne de meilleurs résultats que t_{2ne}, qui donne de meilleurs résultats que t_{ne}, qui lui-même est meilleur que $t_{\frac{ne}{2}}$. Les algorithmes ont donc plus de chances de converger lorsqu'ils sont autorisés à effectuer le plus d'évaluations. Nous remarquons également que certaines versions de DMLS, à l'instar de $(* \cdot 1_{\succ})$ et $(* \cdot *)$, ont besoin de plus d'évaluations que les autres pour obtenir des résultats que les versions $(1 \cdot 1_{\succ})$ et $(1 \cdot *)$ ont obtenus plus tôt. C'est uniquement à partir du critère d'arrêt t_{4ne} qu'ils arrivent à obtenir une meilleure solution que celle qui a été obtenue après t_{ne} évaluations par $(1 \cdot 1_{\succ})$ et $(1 \cdot *)$. Ainsi, ils ont eu besoin de 4 fois plus évaluations pour obtenir les mêmes résultats que $(1 \cdot 1_{\succ})$ et $(1 \cdot *)$. Les tests de Friedman et Iman-Davenport confirment qu'il y a des différences statistiquement significatives entre les rangs ($\alpha = 0.05$). Nous pouvons donc approfondir l'étude statistique avec des tests post-hoc afin d'identifier les différences entre algorithmes.

Table 3.15: Comparaison des versions de DMLS - Tests post-hoc de Holm ($\alpha = 0.05$) à partir de la F-mesure sur les données d'apprentissage

	$t_{\frac{ne}{2}}$				t_{ne}				t_{2ne}				t_{4ne}			
	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1)$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1)$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1)$	$(* \cdot *)$	$(1 \cdot 1_{\succ})$	$(1 \cdot *)$	$(* \cdot 1)$	$(* \cdot *)$
$t_{\frac{ne}{2}}$ $(1 \cdot 1_{\succ})$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$
$(1 \cdot *)$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$
$(* \cdot 1)$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\succ$	$\succ$
$(* \cdot *)$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\succ$	$\succ$
t_{ne} $(1 \cdot 1_{\succ})$	$\equiv$	$\equiv$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$
$(1 \cdot *)$	$\equiv$	$\equiv$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$
$(* \cdot 1)$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\succ$	$\equiv$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$
$(* \cdot *)$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\equiv$	$\succ$	$\succ$	$\equiv$	$\succ$
t_{2ne} $(1 \cdot 1_{\succ})$	$\prec$	$\prec$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$
$(1 \cdot *)$	$\prec$	$\prec$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$
$(* \cdot 1)$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$
$(* \cdot *)$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\succ$	$\equiv$	$\equiv$
t_{4ne} $(1 \cdot 1_{\succ})$	$\prec$	$\prec$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$
$(1 \cdot *)$	$\prec$	$\prec$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\equiv$
$(* \cdot 1)$	$\equiv$	$\equiv$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$
$(* \cdot *)$	$\equiv$	$\equiv$	$\prec$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\prec$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$	$\equiv$

La table 3.15 donne les résultats de comparaisons deux à deux de chacun des algorithmes avec le test de Holm, avec $\alpha = 0.05$. Chaque colonne correspond à une version de DMLS et le critère d'arrêt associé, chaque ligne indique quelle version il surpasse ($\succ$) ou par laquelle il se fait surpasser ($\prec$). Elle confirme les observations précédentes effectuées sur les rangs : plus le critère d'arrêt autorise d'évaluations, plus les performances sont élevées. $t_{\frac{ne}{2}}$ est plus souvent surpassé que t_{ne}, et t_{4ne} n'est jamais surpassé. Nous n'observons pas de différence statistique entre les versions de DMLS pour un même critère d'arrêt. Nous pouvons uniquement supposer que $(1 \cdot 1_{\succ})$ et $(1 \cdot *)$ convergent plus rapidement que $(* \cdot 1_{\succ})$ et $(* \cdot *)$ car ils les surpassent plus fréquemment, et ils comment à dépasser les autres algorithmes dès t_{ne} évaluations. $(* \cdot 1_{\succ})$ et $(* \cdot *)$ ont besoin de t_{4ne} évaluations avant de dépasser les mêmes algorithmes.

Du point de vue des données de test, la figure 3.11 montre que les rangs moyens obtenus sur les données de test sont très similaires. Les tests de Friedman et Iman-Davenport confirment

cette affirmation car ils ne peuvent pas rejeter *H0*, ce qui indique que les rangs sont équivalents. Comme tous les algorithmes obtiennent des rangs similaires sur les données de test nous choisissons de privilégier DMLS $(1 \cdot *)$ qui, comme nous l'avons vu précédemment, est capable de converger rapidement. De plus, il obtient un rang moyen intéressant sur les données de test avec un critère d'arrêt raisonnable (t_{ne}). Il reste possible d'obtenir un rang légèrement meilleur en autorisant plus d'évaluations, mais il est nécessaire de les multiplier par deux ou quatre.

3.4 Comparaison à la littérature

Les sections précédentes se sont attachées à déterminer le meilleur paramétrage et la meilleure version de l'algorithme de DMLS à utiliser pour le problème de classification partielle sur données asymétriques. Nous avons ainsi déterminé les valeurs les plus robustes pour chacun des paramètres : une discrétisation en 10 parts, une taille limite de l'archive de 500 et un nombre de règles maximal de 10 par ensemble de règles. À l'issue des expérimentations, nous avons retenu DMLS $(1 \cdot *)$ pour ses bons résultats et sa convergence rapide. Dans cette section, nous allons confronter la version ainsi obtenue de MOCA-I aux autres algorithmes de la littérature. Les expérimentations seront effectuées tout d'abord sur les 10 jeux de la littérature utilisés précédemment, permettant de déterminer les algorithmes les plus performants. Ces derniers seront ensuite mis en conditions réelles sur deux jeux de données médicales, afin de tester leur tolérance à la volumétrie, l'asymétrie et l'incertitude des données. Nous détaillerons tout d'abord le protocole utilisé pour effectuer la comparaison, et les résultats obtenus.

3.4.1 Protocole

Nous comparons MOCA-I aux performances des algorithmes évoqués précédemment dans le chapitre 2 section 2.1.4. Seule une sélection des algorithmes les plus adaptés aux données asymétriques sera étudiée. Une première sélection est effectuée à partir des résultats des travaux de Fernandez *et al.* [38] qui ont comparé les performances de 22 algorithmes de classification sur des données asymétriques. Seuls les algorithmes les plus efficaces sur cette étude sont sélectionnés, il s'agit des algorithmes *CORE*, *DT_GA*, *GAssist*, *Hider*, *ObliqueDT*, *OCEC*, *Ripper*, *SIA* et *XCS*. Enfin, cette sélection est complétée par des algorithmes qui n'ont pas fait l'objet de cette étude de Fernandez *et al.* : *DataBoost-IM*, *AdaC2* et *C4.5-CS* qui sont issus des techniques de *boosting* ou *cost-sensitive*. Dans le cas où un algorithme plus récent a été proposé depuis l'étude, c'est la version la plus récente qui est utilisée, à l'instar de *BioHEL* qui est le successeur de l'algorithme *GAssist* [8]. Nous utilisons les implémentations de ces algorithmes fournies par le framework KEEL. Il s'agit d'un framework JAVA qui fournit un environnement expérimental pour la fouille de données, ainsi que de nombreuses implémentations d'algorithmes de la littérature [3]. Les paramètres de chacun de ces algorithmes qui seront utilisés pour la comparaison correspondent à ceux proposés par leurs auteurs.
Nous utilisons la configuration de MOCA-I déterminée précédemment, qui s'est avérée être la plus efficace : DMLS $(1 \cdot *)$ associé à une taille de la population initiale de 200 et une taille

limite de l'archive de 500. Une première comparaison est effectuée sur les 10 jeux de données asymétriques et discrétisés de la littérature présentés précédemment. Pour ces jeux, MOCA-I est limité à des ensembles de règles contenant au maximum 10 règles. Les meilleurs algorithmes seront ensuite évalués sur des jeux de données plus proches des données réelles : *tia-f* and *S06-f*. Ces deux jeux de données contiennent une volumétrie de données plus importante (10 000 observations, et jusqu'à 1000 attributs) et sont issus des données médicales PMSI. Leur nombre d'attributs a été réduit à moins de 1000 attributs afin de permettre au framework KEEL de les charger en mémoire et d'ainsi pouvoir effectuer la comparaison. Nous les avons réduits en conservant uniquement les attributs présents sur les observations positives, c'est-à-dire les observations qui vérifient la classe à prédire.
Le protocole de comparaison est similaire aux protocoles définis précédemment. L'objectif est de déterminer l'algorithme le plus efficace sur l'ensemble des jeux de données, sans adaptation du paramétrage d'un jeu de données sur l'autre. Lors de la comparaison sur les 10 jeux de la littérature, chaque algorithme est exécuté 25 fois : 5 fois en validation croisée à 5 parts. Les 2 jeux de données réels sont plus volumineux : seuls les algorithmes ayant donné les meilleurs résultats précédemment seront exécutés. D'autre part, le temps de calcul nécessaire est plus important, chaque algorithme sera donc cette fois-ci exécuté 15 fois : 5 fois en validation croisée à 3 parts. Dans les deux cas, la *f-mesure* obtenue sur les données de test est utilisée pour apprécier la qualité prédictive des solutions proposées par les différents algorithmes. Les méthodes de validation statistique adaptées seront ensuite utilisées, comme présentées dans le chapitre 2. Ces expérimentations sont réalisées sur un poste Z400 disposant de 8 Go de mémoire vive, Intel Xeon 2.66 GHz avec 4 cœurs, sous Ubuntu version 11.10 (oneiric). Lors des exécutions de KEEL framework, Java dispose d'une mémoire maximum de 5120 Mo, ce qui permet de charger les jeux de données volumineux en mémoire.

3.4.2 Résultats sur les jeux de données de la littérature

La table 3.16 donne les résultats obtenus par chacun des algorithmes sur les 10 jeux de données de la littérature. Pour chaque jeu de données, cette table donne la *f-mesure* moyenne et son écart-type, obtenus sur les données d'apprentissage ainsi que ceux obtenus sur les données de test.

Nous pouvons maintenant observer les performances des algorithmes sur l'ensemble des jeux de données, via la figure 3.12 qui donne le rang moyen obtenu par chacun des algorithmes sur les données de test. Nous observons que *MOCA-I* obtient le meilleur rang moyen, suivi par *Ripper* et *GAssist*. 5 algorithmes se démarquent par des rangs moyens élevés, les algorithmes restants obtiennent des rangs similaires. Les tests de Friedman et Iman-Davenport confirment que les rangs obtenus sont statistiquement différents, les valeurs obtenues étant supérieures aux valeurs seuil (respectivement 47.65 (valeur seuil 21.03) et 5.93 (valeur seuil 1.84)). Les rangs étant statistiquement différents, nous pouvons réaliser des tests post-hoc afin de déterminer quels algorithmes se démarquent. Pour cela, des comparaisons N $\times$ N avec Wilcoxon et Holm sont réalisées. Ces tests affirment que *MOCA-I* est statistiquement meilleur que *AdaC2*, *XCS*,

Table 3.16: Comparaison de *MOCA-I* à la littérature - *f-mesure* moyenne et écart-type sur les données d'apprentissage et de test

	AdaC2	BioHEL	C4.5-CS	DataBoost-IM	DT_GA	GAssist	Hider	OCEC	Ripper	SIA	XCS	CORE	Oblique DT	MOCA-I (1 . *)
hab_d	0.17 ± 0.21	0.70 ± 0.03	0.42 ± 0.00	0.69 ± 0.05	0.07 ± 0.14	0.55 ± 0.03	0.09 ± 0.07	0.65 ± 0.05	0.64 ± 0.03	0.65 ± 0.02	0.11 ± 0.14	0.32 ± 0.08	**0.73 ± 0.03**	0.63 ± 0.03
	0.17 ± 0.21	0.31 ± 0.09	**0.42 ± 0.01**	0.29 ± 0.12	0.02 ± 0.04	0.36 ± 0.11	0.06 ± 0.08	0.39 ± 0.11	0.41 ± 0.08	0.38 ± 0.08	0.05 ± 0.10	0.19 ± 0.12	0.37 ± 0.10	0.40 ± 0.12
$ec1_d$	0.68 ± 0.04	0.96 ± 0.02	0.77 ± 0.05	**0.97 ± 0.01**	0.76 ± 0.04	0.86 ± 0.05	0.78 ± 0.02	0.79 ± 0.05	0.83 ± 0.03	0.95 ± 0.03	0.66 ± 0.27	0.62 ± 0.10	**0.97 ± 0.01**	0.90 ± 0.01
	0.63 ± 0.05	0.65 ± 0.12	0.72 ± 0.05	0.72 ± 0.08	0.65 ± 0.06	**0.77 ± 0.07**	**0.77 ± 0.08**	0.71 ± 0.07	0.71 ± 0.07	0.65 ± 0.10	0.64 ± 0.28	0.57 ± 0.16	0.68 ± 0.08	0.76 ± 0.06
$ec2_d$	0.50 ± 0.12	0.97 ± 0.01	0.56 ± 0.09	**0.98 ± 0.01**	0.40 ± 0.14	0.87 ± 0.03	0.74 ± 0.05	0.80 ± 0.07	0.90 ± 0.03	0.96 ± 0.03	0.59 ± 0.35	0.59 ± 0.13	**0.98 ± 0.01**	0.93 ± 0.02
	0.47 ± 0.13	0.64 ± 0.11	0.48 ± 0.12	0.80 ± 0.12	0.38 ± 0.16	0.80 ± 0.08	0.64 ± 0.11	0.69 ± 0.11	0.63 ± 0.14	0.66 ± 0.09	0.56 ± 0.34	0.54 ± 0.18	0.72 ± 0.05	**0.83 ± 0.04**
$ye3_d$	0.27 ± 0.08	0.82 ± 0.02	0.36 ± 0.03	0.79 ± 0.02	0.64 ± 0.06	0.81 ± 0.02	0.62 ± 0.02	0.81 ± 0.05	0.85 ± 0.03	0.88 ± 0.02	0.30 ± 0.32	0.36 ± 0.21	**0.95 ± 0.01**	0.81 ± 0.01
	0.26 ± 0.07	0.68 ± 0.05	0.34 ± 0.02	0.70 ± 0.03	0.59 ± 0.03	0.72 ± 0.05	0.62 ± 0.07	0.61 ± 0.05	0.68 ± 0.05	0.64 ± 0.04	0.30 ± 0.31	0.35 ± 0.23	0.66 ± 0.05	**0.74 ± 0.05**
$ye2_d$	0.61 ± 0.04	0.98 ± 0.02	0.63 ± 0.06	0.94 ± 0.13	0.69 ± 0.03	0.70 ± 0.04	0.69 ± 0.03	0.73 ± 0.25	0.97 ± 0.02	**1.00 ± 0.01**	0.39 ± 0.36	0.69 ± 0.03	**1.00 ± 0.00**	0.88 ± 0.03
	0.21 ± 0.14	0.40 ± 0.20	0.38 ± 0.15	0.59 ± 0.16	0.66 ± 0.15	0.66 ± 0.15	**0.67 ± 0.15**	0.34 ± 0.13	0.59 ± 0.21	0.45 ± 0.15	0.33 ± 0.34	0.65 ± 0.15	0.39 ± 0.10	0.49 ± 0.15
$ab1_d$	0.09 ± 0.12	0.39 ± 0.06	0.25 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.01 ± 0.03	0.00 ± 0.00	0.18 ± 0.07	0.40 ± 0.04	0.25 ± 0.06	0.00 ± 0.00	0.01 ± 0.02	**0.48 ± 0.04**	0.32 ± 0.03
	0.01 ± 0.02	0.03 ± 0.07	**0.05 ± 0.03**	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.04 ± 0.03	0.03 ± 0.04	0.02 ± 0.04	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.02 ± 0.05
$a1a$	0.57 ± 0.06	0.82 ± 0.03	0.77 ± 0.02	0.97 ± 0.01	0.73 ± 0.02	0.66 ± 0.02	0.47 ± 0.04	0.79 ± 0.03	0.81 ± 0.02	**0.98 ± 0.01**	0.78 ± 0.03	0.00 ± 0.00	**0.98 ± 0.00**	0.66 ± 0.01
	0.52 ± 0.04	0.62 ± 0.03	0.62 ± 0.01	0.57 ± 0.03	0.61 ± 0.02	0.56 ± 0.03	0.42 ± 0.05	0.59 ± 0.03	0.61 ± 0.04	0.57 ± 0.03	0.56 ± 0.04	0.00 ± 0.00	0.57 ± 0.03	**0.63 ± 0.02**
luc	0.78 ± 0.08	0.97 ± 0.00	0.94 ± 0.01	0.82 ± 0.07	0.79 ± 0.06	0.90 ± 0.01	0.73 ± 0.05	0.92 ± 0.02	0.95 ± 0.01	**1.00 ± 0.00**	0.93 ± 0.01	0.00 ± 0.00	**1.00 ± 0.00**	0.84 ± 0.01
	0.72 ± 0.07	0.80 ± 0.03	0.83 ± 0.02	0.75 ± 0.09	0.74 ± 0.07	**0.84 ± 0.03**	0.72 ± 0.06	0.83 ± 0.02	0.79 ± 0.03	0.83 ± 0.03	0.37 ± 0.43	0.00 ± 0.00	0.77 ± 0.04	0.82 ± 0.01
$w1a$	0.04 ± 0.03	0.81 ± 0.03	0.20 ± 0.07	0.14 ± 0.02	0.07 ± 0.04	0.64 ± 0.05	0.08 ± 0.18	0.53 ± 0.20	0.80 ± 0.04	0.91 ± 0.02	0.64 ± 0.07	0.00 ± 0.00	**0.94 ± 0.01**	0.69 ± 0.03
	0.04 ± 0.03	0.39 ± 0.09	0.13 ± 0.03	0.10 ± 0.05	0.04 ± 0.03	0.37 ± 0.20	0.04 ± 0.11	0.24 ± 0.10	0.45 ± 0.12	0.33 ± 0.13	0.22 ± 0.19	0.00 ± 0.00	**0.49 ± 0.14**	0.48 ± 0.19
$ab9_d$	0.11 ± 0.00	0.86 ± 0.02	0.61 ± 0.04	**0.92 ± 0.01**	0.00 ± 0.00	0.27 ± 0.12	0.12 ± 0.05	0.70 ± 0.14	0.81 ± 0.04	0.85 ± 0.04	0.07 ± 0.10	0.18 ± 0.06	**0.92 ± 0.01**	0.70 ± 0.03
	0.11 ± 0.01	0.32 ± 0.28	0.33 ± 0.19	0.34 ± 0.31	0.00 ± 0.00	0.08 ± 0.10	0.12 ± 0.15	0.34 ± 0.17	0.40 ± 0.24	0.36 ± 0.25	0.03 ± 0.09	0.14 ± 0.16	0.43 ± 0.28	**0.45 ± 0.19**

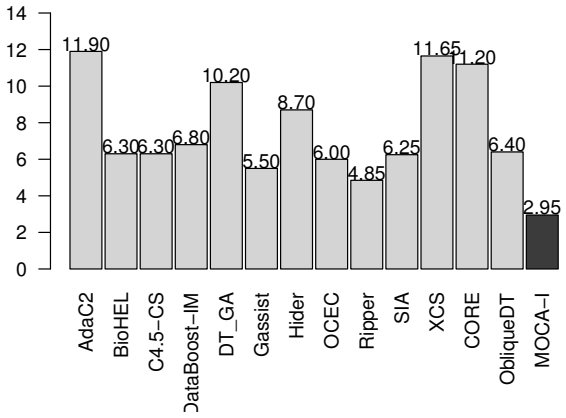


Figure 3.12: Comparaison de *MOCA-I* à la littérature - rangs moyens sur les données de test, sur les jeux de données de petite taille et taille moyenne, en utilisant la *F-mesure*

CORE, *DT_GA* et *Hider* avec $\alpha = 0.05$. *Ripper* est quant à lui statistiquement meilleur que *AdaC2* et *XCS*. Les données expérimentales ne permettent pas de donner de résultat sur les autres algorithmes, ce qui signifie que ceux-ci sont statistiquement équivalents.

3.4.3 Résultats sur les jeux de données réels

À partir des résultats précédents, nous avons sélectionné les algorithmes les plus performants : *MOCA-I*, *GAssist* et *Ripper*. À ces algorithmes, nous ajoutons un sous-ensemble des algorithmes qui sont considérés comme équivalents : *Oblique-DT* car il a été prouvé efficace par rapport à d'autres algorithmes sur des données asymétriques [38] ; *C4.5-CS* afin d'avoir un représentant d'une méthode *cost-sensitive* et *DataBoost-IM* comme représentant d'une approche de *boosting*. Nous préférons sélectionner *BioHEL* à laplace de *GAssist*, étant donné qu'il s'agit de son successeur. Finalement, l'étude est réalisée sur les algorithmes *BioHEL*, *C4.5-CS*, *DataBoost-IM*, *Ripper*, *ObliqueDT* et *MOCA-I*. Du point de vue de la complexité de ces jeux de données, nous avons expérimentalement vérifié si la limite de règles de *MOCA-I* était adaptée. Il s'est avéré que sur ces jeux de données une limite de 20 règles améliore les résultats de l'ordre de 1% à 30% (jeu de données *S06-f*) par rapport à la limite de 10 règles qui a été utilisée sur les jeux de données de la littérature. En présence de jeu de données complexes, cela souligne l'importance de vérifier que la limite du nombre de règles n'a pas d'impact sur les performances de classification, et d'éventuellement la ré-évaluer.

F-mesure, temps d'exécution et complexité des modèles La table 3.17 donne les résultats obtenus par chacun des algorithmes sur les deux jeux de données réels. Elle détaille à la fois les performances en prédiction avec la *f-mesure* moyenne et son écart-type, mais donne éga-

Table 3.17: Comparaison de *MOCA-I* à la littérature - Résultats moyens obtenus sur les jeux réels *tia-f* et *s06-f* : *F-mesure* sur données d'apprentissage et de test, temps d'exécution (en secondes) et nombre de termes dans les classifieurs générés (tests d'attributs (#TA))

Mesure	Jeu de données	BioHEL	C4.5-CS	DataBoost-IM	Ripper	ObliqueDT	MOCA-I
F-mesure	tia-f	0.88 ± 0.03	0.65 ± 0.06	0.00 ± 0.00	0.90 ± 0.03	**1.00 ± 0.00**	0.82 ± 0.02
	(tst)	0.16 ± 0.04	0.16 ± 0.04	0.00 ± 0.00	0.29 ± 0.07	0.30 ± 0.07	**0.36 ± 0.09**
	s06-f	0.85 ± 0.03	0.73 ± 0.04	-	0.94 ± 0.02	-	**0.96 ± 0.01**
	(tst)	0.36 ± 0.07	0.39 ± 0.04	-	0.64 ± 0.07	-	**0.70 ± 0.05**
#TA	tia-f	98.13 ± 9.21	112.67 ± 16.56	**11.00 ± 4.39**	61.80 ± 3.10	55.00 ± 0.85	25.60 ± 2.32
	s06-f	70.60 ± 10.01	104.67 ± 19.01	-	49.53 ± 10.57	-	**26.00 ± 2.62**
Temps	tia-f	11724 ± 676	**22 ± 0**	172 ± 9	32 ± 2	273 ± 14	24154 ± 5442
	s06-f	15769 ± 1637	**27 ± 0**	-	107 ± 234	-	15766 ± 10043

lement le temps d'exécution moyen en secondes, et le nombre moyens de termes présents dans les modèles générés. De la même manière que précédemment, *Oblique-DT* obtient la meilleure *F-mesure* sur les données d'apprentissage, sur le jeu de données *tia-f*. On observe que la table ne fournit pas de résultats pour les algorithmes *DataBoost-IM* et *ObliqueDT* pour le jeu de données *s06-f* : ces deux algorithmes n'ont pas été en mesure de traiter en mémoire ce jeu de données. Concernant la *f-mesure* sur les données de test, *MOCA-I* obtient les meilleurs résultats, suivi par *Ripper*. Les ensembles de règles fournis en résultat par *DataBoost-IM* sont les plus compacts, avec en moyenne 11 termes. Ils sont suivis par les ensembles de règles de *MOCA-I*, qui contiennent en moyenne 26 termes, ce qui est 2 à 6 fois plus compact que les modèles générés par les autres algorithmes. Du point de vue du temps d'exécution, les algorithmes les plus rapides sont *C4.5-CS* et *Ripper*.

Table 3.18: Comparaison de *MOCA-I* à la littérature - Comparaison de MOCA-I aux autres algorithmes à l'aide du test de Mann-Whitney, à partir de la *F-mesure* sur les données de test

Jeu de données		MOCA-I	p-valeur ajustée (Holm)
tia-f	BioHEL	$\succ$	8.7e-06
	C4.5-CS	$\succ$	8.7e-06
	Ripper	$\succ$	1.8e-02
S06-f	BioHEL	$\succ$	8.7e-06
	C4.5-CS	$\succ$	8.7e-06
	Ripper	$\succ$	1.8e-02

MOCA-I statistiquement plus performant La table 3.18 donne les résultats obtenus par les tests de Mann-whitney comparant *MOCA-I* à chacun des algorithmes, associé à une

correction de Holm. La comparaison est effectuée sur la base de la *f-mesure* sur les données de test. Pour chacun des algorithmes et des jeux de données, on indique si *MOCA-I* est meilleur ($\succ$) ou moins bon ($\prec$), ainsi que la *p-valeur* du test, ajustée avec la correction de Holm. Nous observons que *MOCA-I* est statistiquement meilleur que *Ripper*, *BioHEL* et *C4.5-CS*, sur les deux jeux de données réels.

Qualité des modèles et solutions de compromis Concernant la qualité des règles obtenues par *MOCA-I*, leur contenu est cohérent d'un point de vue médical par rapport aux pathologies prédites : on retrouve des règles contenant des actes médicaux ou pathologies couramment associées. En plus de proposer des modèles deux fois plus simples que les autres algorithmes, *MOCA-I* propose une souplesse dans le choix de la solution. De part ses solutions de compromis, il laisse la possibilité au décideur de choisir une solution plus adaptée au besoin. Plus précise si l'on souhaite détecter des patients avec une faible marge d'erreur, plus sensible si l'on souhaite détecter le plus de patients possible.

3.4.4 Conclusion

Cette section a évalué les performances de *MOCA-I* par rapport à plusieurs algorithmes de la littérature, à la fois sur des jeux de données de la littérature et des jeux de données réels. *MOCA-I* s'avère statistiquement plus performant que les autres algorithmes du point de vue de la qualité prédictive (*f-mesure*), sur tous les jeux de données étudiés. Il génère des classifieurs 2 à 6 fois plus compacts que les autres algorithmes, ce qui les rend plus interprétables. En revanche des améliorations peuvent être apportées au niveau du temps d'exécution, *MOCA-I* étant plus lent que les algorithmes auxquels il était comparé. Une perspective pourrait être d'initialiser *MOCA-I* avec les résultats d'un algorithme rapide comme *Ripper*. Les travaux de Dos Santos *et al.* [35] ouvrent une autre voie intéressante : ils ont montré qu'il était possible d'obtenir de meilleurs résultats sur les données de test en arrêtant la recherche de solution avant la convergence. Ainsi, arrêter l'algorithme *MOCA-I* avant d'atteindre l'optimum local pourrait permettre d'augmenter sa rapidité tout en améliorant les résultats.

3.5 Conclusion

Dans ce chapitre, nous nous sommes tout d'abord penchés sur la modélisation du problème de classification partielle sur données asymétriques sous la forme d'un problème multi-objectif. Pour cela, une étude statistique sur les différents objectifs candidats a été réalisée sous la forme d'une analyse en composantes principales (ACP), afin de déterminer des similarités entre des objectifs et guider le choix d'objectifs complémentaires. L'ACP nous a permis de mettre en évidence 2 objectifs candidats : la *confiance* et la *sensibilité* sur différents jeux de données asymétriques. Un troisième objectif est ajouté pour privilégier les solutions simples. Suite à cette étude, nous proposons la modélisation MOCA-I, qui modélise le problème de classification partielle sur données asymétriques sous la forme d'un problème multi-objectif.
Nous avons ensuite évalué les bénéfices apportés par la modélisation multi-objectif par rapport à une approche – plus simple – où les objectifs sont agrégés au sein d'un même objectif. Deux approches ont ainsi été comparées : une recherche locale mono-objectif (Hill-Climbing) et une recherche locale multi-objectif (algorithme DMLS). La modélisation multi-objectif (implémentée avec DMLS) s'est avérée statistiquement plus performante que la mono-objectif.
Par la suite, nous avons étudié l'impact de différents paramétrages de la recherche locale et de différentes variantes de DMLS, afin de déterminer la version la plus efficace. Ainsi, DMLS $(1 \cdot *)$ a été estimé comme étant la version la plus efficace.
Les performances de cette version ont ensuite été comparées à celles des meilleurs algorithmes de la littérature sur 10 jeux de données de la littérature et 2 jeux de données issus des données réelles. MOCA-I donne statistiquement de meilleurs résultats que les autres algorithmes de la littérature, sur l'ensemble des jeux de données étudiés. L'étude est ensuite affinée sur les 2 jeux de données réels, où l'on peut observer qu'en plus de donner de meilleurs résultats, les modèles proposés par MOCA-I sont 2 à 6 plus compacts et donc plus facilement interprétables. Cependant MOCA-I est plus lent que certains autres algorithmes, ce qui n'est pas gênant dans notre cas d'utilisation mais peut ouvrir des perspectives d'amélioration si l'on souhaite utiliser MOCA-I dans d'autres domaines d'application.
Ainsi, dans ce chapitre, nous avons vu comment MOCA-I pouvait traiter le problème de classification partielle sur données asymétriques. Dans le chapitre suivant, nous allons étudier MOCA – qui vise à traiter des problèmes de classification n'ayant pas forcément les mêmes caractéristiques (absence d'asymétrie, de données partielles ou de problème de volumétrie) – afin de vérifier si notre approche est généralisable à des problèmes de classification plus "standards".

3. MOCA-I : MODÉLISATION SOUS FORME D'UN PROBLÈME MULTI-OBJECTIF ET RÉSOLUTION À L'AIDE D'UNE RECHERCHE LOCALE

Chapitre 4

Impact de l'asymétrie sur les approches de classification

Dans le chapitre précédent, nous avons proposé l'algorithme MOCA-I, une méthode adaptée à la classification dans les données médicales. Ces dernières présentent de nombreuses particularités, les plus importantes étant l'asymétrie des données et la nécessité d'utiliser la classification partielle. Nous allons maintenant évaluer si MOCA-I peut être également utilisé efficacement sur des données plus classiques ; c'est-à-dire dans un cadre de classification binaire et non plus de classification partielle et sur des données ne présentant pas d'asymétrie sur la classe à prédire. Dans la suite, nous désignerons cette tâche par « classification binaire standard ». Nous étudierons ensuite des modélisations alternatives théoriquement plus adaptées à cette tâche, fondées sur d'autres mesures comme l'*exactitude* ou la *sensibilité* associée à la *spécificité*. Enfin, nous déterminerons théoriquement et expérimentalement quelles approches sont à préconiser selon le degré d'asymétrie des jeux de données étudiés.

4.1 Étude de la pertinence de MOCA-I en classification binaire standard

Dans cette partie, nous allons déterminer expérimentalement si MOCA-I est performant en classification sur les jeux de données plus « classiques » de la littérature, c'est-à-dire ne présentant pas d'asymétrie. Nous commencerons par un rapide état de l'art et détaillerons ensuite les adaptations nécessaires sur MOCA-I afin de gérer ce type de données. Ensuite, nous comparerons expérimentalement MOCA-I à une sélection d'algorithmes de la littérature sur des jeux de données utilisés couramment en classification binaire et qui ne présentent pas d'asymétrie.

4.1.1 État de l'art sur la classification binaire standard

Dans le chapitre 2 nous avons présenté un état de l'art sur les algorithmes de classification. Nous n'allons pas le reprendre ici. Notons cependant que nous avions remarqué que peu d'entre eux étaient prévus pour gérer des données asymétriques, et sont donc initialement destinés à des problèmes de classification binaire standard, où l'on ne rencontre pas de fortes différences de répartitions entre le nombre d'observations positives et négatives. Ces approches devraient donc donner de meilleurs résultats sur ce type de données, par rapport aux résultats qu'elles avaient obtenus dans le chapitre précédent, en classification sur données asymétriques. La mesure de référence couramment utilisée pour évaluer les performances en classification binaire standard est l'*exactitude*, qui représente la proportion d'observations bien classées parmi l'ensemble des observations.

4.1.2 Adaptations de MOCA-I

MOCA-I est prévu à la base pour la tâche de classification partielle, ce qui signifie que lorsqu'il est évalué, on mesure sa capacité à bien détecter les observations positives, sans s'intéresser aux observations négatives, à travers la *f-mesure*. Comme nous venons de le voir, en classification standard c'est plutôt l'*exactitude* qui est utilisée, qui se focalise sur la proportion d'observations (positives et négatives) correctement classées. Afin de ne pas pénaliser MOCA-I lors de l'évaluation, à l'issue de l'exécution de l'algorithme nous proposerons une légère modification lors du choix de la solution finale, qui consiste à choisir la solution de meilleure *exactitude* sur le jeu d'apprentissage. Cette version est nommée dans la suite MOCA-I (MaxExa). Nous comparerons également ses résultats à ceux d'une version non modifiée de MOCA-I, où la solution de meilleure *f-mesure* est sélectionnée.

4.1.3 Évaluation des performances de MOCA-I

Nous allons maintenant évaluer les performances de MOCA-I par rapport aux autres algorithmes de la littérature sur des jeux de données de classification binaire standard. Nous présenterons tout d'abord le protocole de comparaison, suivi par les résultats, la discussion associée et enfin la conclusion.

4.1.3.1 Protocole

MOCA-I et sa variante MOCA-I (MaxExa) vont être comparés à une sélection de 15 algorithmes de la littérature présentés précédemment : *AdaC2*, *BioHEL*, *C4.5*, *C4.5-CS*, *DataBoost-IM*, *DT_GA*, *GAssist*, *Hider*, *OCEC*, *Ripper*, *SIA*, *XCS*, *CORE*, *Oblique-DT* et *SMOTEBoost*. Il s'agit des mêmes algorithmes que ceux utilisés pour l'évaluation du chapitre 3, auxquels nous avons ajouté l'algorithme de référence *C4.5*. Ces algorithmes font également partie de l'état de l'art en classification standard. Nous utilisons les implémentations de ces algorithmes fournies avec le framework KEEL [3]. La configuration utilisée pour chacun d'entre eux correspond à celle recommandée par leurs auteurs. Nous utilisons le paramétrage de MOCA-I préconisé dans

le chapitre 3, associé à une taille limite des ensembles de règles de 10 règles. MOCA-I est arrêté dès qu'il atteint un optimum local. Le framework KEEL ne permet pas d'instaurer de limite de temps d'exécution, les autres algorithmes ont le temps nécessaire pour converger ou atteindre la limite de nombre d'itérations recommandée par leurs auteurs. Chacun d'entre eux dispose d'une mémoire vive maximum de 5 Go. Les algorithmes sont évalués sur les 10 jeux de données de classification présentés dans la table 2.6 (chapitre 2 sous-section 2.3.1.3). MOCA-I étant à la base destiné à de la classification sur attributs qualitatifs, les jeux de données contenant des attributs quantitatifs vont être transformés, comme effectué précédemment sur les jeux de données de classification asymétrique. Leurs attributs quantitatifs sont discrétisés en attributs qualitatifs à 10 intervalles réguliers [1]. Les noms des jeux de données ayant fait l'objet d'une discrétisation sont post-fixés par $_d$, par exemple $adult_d$. Chacun des algorithmes est exécuté 25 fois ; 5 fois en validation croisée à 5 partitions. L'*exactitude* moyenne sur les 25 exécutions est utilisée pour comparer les performances des algorithmes, et les validations statistiques sont effectuées conformément aux préconisations émises précédemment dans le chapitre 2.

4.1.3.2 Résultats et discussion

Nous avons récapitulé dans la table 4.1 l'*exactitude* moyenne obtenue par les différents algorithmes sur chacun des jeux de données. Les lignes impaires donnent le nom de chaque jeu de données et l'*exactitude* moyenne obtenue sur les données d'apprentissage. Les lignes paires indiquent l'*exactitude* moyenne obtenue sur les données de test. L'absence de résultat pour certains jeux de données s'explique d'une part par la présence de valeurs manquantes pour certains des attributs dans les jeux de données $horsecolic_d$ et $votes$, ce qui n'est pas géré par tous les algorithmes. D'autre part, le jeu de données $adult_d$ comprend un grand nombre d'observations (environ 50 000), ce qui a posé problème à certains algorithmes comme DT_GA, $CORE$ ou $ObliqueDT$ pour monter l'ensemble de données en mémoire. Sur les données d'apprentissage, l'algorithme *DataBoost-IM* obtient le plus souvent les meilleurs résultats. Concernant les données de test, il n'y a pas d'algorithme qui obtient la meilleure solution sur l'ensemble des jeux de données. Au mieux, XCS obtient les meilleurs résultats sur 2 des jeux de données.

Regardons maintenant les rangs moyens obtenus par chacun des algorithmes sur l'ensemble des jeux de données, afin de déterminer lequel(s) des algorithmes s'en sort(ent) le mieux sur l'ensemble des jeux de données étudiés. La figure 4.1 donne les rangs moyens obtenus par les algorithmes précédents sur l'ensemble des 10 jeux de données de la littérature. On observe que les algorithmes *C4.5*, *XCS*, *GAssist* et *Hider* donnent de meilleurs résultats que les deux versions de MOCA-I. On observe quelques changements avec les algorithmes qui étaient les plus efficaces sur les données asymétriques dans le chapitre 3. En effet, l'algorithme *Ripper* a un rang beaucoup plus élevé que la majorité des algorithmes, ce qui indique qu'il se classe le plus souvent après les autres algorithmes, alors qu'il était parmi les plus efficaces sur les données asymétriques. À l'inverse, l'algorithme *Hider* qui était parmi les moins efficaces sur

1. weka.filters.unsupervised.attribute.Discretize ; bins=10, findNumBins=true

TABLE 4.1: Comparaison de *MOCA-I* à la littérature - *Exactitude* moyenne sur les données d'apprentissage et de test, jeux de données de classification binaire standard

	AdaC2	BioHEL	C4.5	C4.5-CS-IM	DataBoost	DT_GA	Giassist	Hider	OCEC	Ripper	SIA	XCS	CORE DT	Oblique Boost	SMOTE	MOCA-I (MaxExa)	MOCA-I
$adult_d$	0.25	0.85	0.86	0.74	0.25	-	0.84	0.79	**0.93**	0.84	-	0.81	-	-	0.25	0.85	0.83
tst	0.25	0.83	0.84	0.73	0.25	-	**0.84**	0.79	0.77	0.82	-	0.81	-	-	0.25	0.84	0.82
$australian_d$	0.99	0.98	0.89	0.87	**0.99**	0.87	0.91	0.86	0.90	0.90	0.99	0.91	0.82	**0.99**	0.99	0.92	0.92
tst	0.83	0.81	0.85	0.85	0.83	0.84	0.85	0.85	**0.86**	0.82	0.83	0.85	0.81	0.79	0.83	0.83	0.83
$breast_d$	0.57	0.97	0.78	0.41	**0.98**	0.76	0.87	0.77	0.85	0.88	0.97	0.82	0.76	0.98	0.89	0.86	0.86
tst	0.50	0.70	**0.76**	0.35	0.70	0.75	0.73	0.75	0.65	0.65	0.71	0.74	0.75	0.68	0.66	0.72	0.71
crx_d	0.99	0.98	0.87	0.87	**0.99**	0.87	0.91	0.87	0.90	0.90	0.99	0.90	0.81	0.99	0.99	0.90	0.90
tst	0.82	0.82	0.86	**0.86**	0.81	0.86	0.86	0.86	0.86	0.83	0.82	0.86	0.82	0.81	0.84	0.85	0.85
$heart_d$	**1.00**	**1.00**	0.86	0.84	**1.00**	0.81	0.95	0.83	0.87	0.91	**1.00**	0.96	0.76	**1.00**	1.00	0.93	0.93
tst	0.76	0.75	0.78	0.77	0.77	0.75	0.78	0.76	0.79	0.75	0.78	**0.80**	0.72	0.76	0.78	0.75	0.74
$german_d$	0.85	0.97	0.82	0.75	**1.00**	0.76	0.84	0.73	0.87	0.88	**1.00**	0.92	0.69	**1.00**	1.00	0.82	0.82
tst	0.61	0.71	0.72	0.64	0.69	0.71	0.73	0.73	0.67	0.66	0.72	**0.73**	0.68	0.67	0.72	0.73	0.72
$hepatisis_d$	0.82	**1.00**	0.90	0.86	**1.00**	0.87	0.99	0.93	0.91	0.97	**1.00**	0.99	0.88	**1.00**	**1.00**	1.00	1.00
tst	0.68	0.87	0.84	0.76	**0.91**	0.84	0.81	0.87	0.80	0.84	0.89	0.85	0.82	0.76	0.86	0.86	0.86
$horsecolic_d$	0.00	1.00	0.86	0.85	0.00	-	0.99	**1.00**	0.89	0.82	0.96	0.93	0.68	-	-	0.92	0.92
tst	0.00	0.78	0.85	0.81	0.00	-	0.89	**1.00**	0.79	0.71	0.77	0.85	0.68	-	-	0.83	0.82
$votes$	**1.00**	1.00	0.97	0.96	**1.00**	-	0.99	0.97	0.96	0.93	0.97	0.97	0.93	-	**1.00**	0.98	0.98
tst	0.96	0.95	0.96	0.95	0.96	-	**0.97**	0.94	0.94	0.91	0.94	0.96	0.92	-	0.96	0.94	0.95
$mushrooms$	**1.00**	1.00	1.00	**1.00**	**1.00**	**1.00**	1.00	0.99	**1.00**	1.00	1.00	1.00	0.52	**1.00**	**1.00**	1.00	1.00
tst	**1.00**	1.00	1.00	**1.00**	**1.00**	**1.00**	1.00	0.98	**1.00**	1.00	1.00	1.00	0.52	**1.00**	**1.00**	1.00	1.00

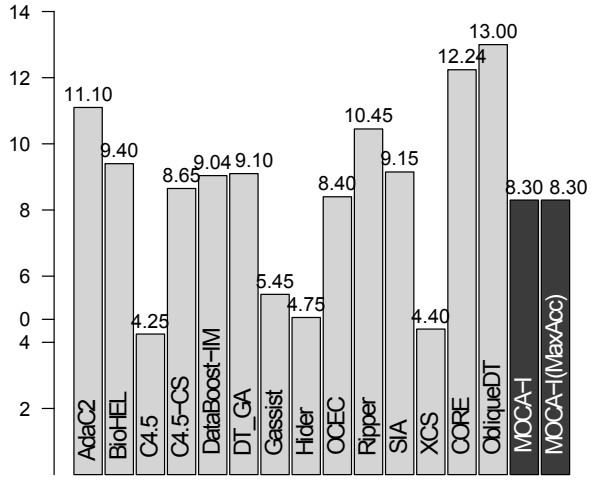


Figure 4.1: Rangs moyens obtenus par les algorithmes sur les jeux de données de classification binaire standard

les données asymétriques est maintenant parmi les meilleurs algorithmes. MOCA-I se retrouve avec des performances dans la moyenne des autres algorithmes, ne se trouvant ni parmi les meilleurs (*C4.5*, *GAssist*, *XCS* et *Hider*), ni parmi les moins bons (*AdaC2*, *Ripper*, *CORE* et *ObliqueDT*). Le test de Friedman montre que les rangs moyens sont statistiquement différents avec une *p-value* inférieure à 0.05. Nous pouvons donc effectuer des tests statistiques post-hoc afin de comparer les algorithmes 2 à 2.

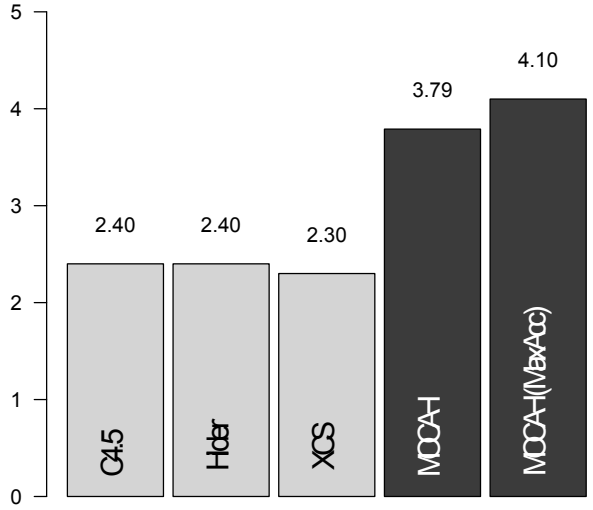


Figure 4.2: Rangs moyens obtenus par les algorithmes *C4.5*, *Hider*, *XCS* et *MOCA-I* sur les jeux de données de classification binaire standard

Afin de réduire le nombre de comparaisons statistiques et ainsi augmenter la significativité des résultats, nous avons comparé 2 à 2 les trois algorithmes de la littérature qui se sont

montrés les plus efficaces (*Hider*, *XCS* et *C4.5*) aux résultats de MOCA-I et de sa variante (MOCA-I (MaxExa)) sur les données de test. Nous avons regroupé les rangs moyens obtenus sur les données de test, dans la figure 4.2. Les observations restent conformes à celles réalisées précédemment : *Hider* et *XCS* obtiennent les meilleurs rangs, ce qui indique leur capacité à être les plus robustes sur l'ensemble des jeux de données. On retrouve ensuite l'algorithme de référence *C4.5*. Ces trois algorithmes donnent de meilleurs performances que MOCA-I et sa variante. Le test de Friedman indique que les rangs moyens sont statistiquement différents avec une *p-value* < 0.05, ce qui nous permet d'affiner avec des tests post-hoc. Nous avons regroupé dans la table 4.2 les résultats des comparaisons de l'algorithme le plus efficace, *XCS*, avec les algorithmes *Hider*, *C4.5* et *MOCA-I* et sa variante. Pour chaque ligne, il est indiqué si *XCS* est statistiquement plus performant ($\succ$), moins performant ($\prec$) ou équivalent ($\equiv$) à l'algorithme de la ligne. Le test de Wilcoxon + correction de Holm indique que *XCS* est statistiquement plus performant que la variante de MOCA-I (MOCA-I (MaxExa)). En revanche, les données expérimentales ne permettent pas de déterminer les relations entre les autres algorithmes, ni si MOCA-I est plus efficace que sa variante MOCA-I (MaxExa).

Table 4.2: Comparaison statistique post-hoc (Wilcoxon + Holm) de XCS à Hider, C4.5 et MOCA-I sur jeux de données de classification binaire standard

	XCS
MOCA-I (MaxExa)	$\succ$ (0.04)
MOCA-I	$\equiv$
Hider	$\equiv$
C4.5	$\equiv$

4.1.3.3 Conclusion

L'étude expérimentale réalisée sur 10 jeux de données de classification binaire standard a permis de montrer la variante de MOCA-I, (MOCA-I (MaxExa)), est statistiquement moins efficace que l'algorithme *XCS*, en donnant toutefois des résultats raisonnables comparativement aux autres algorithmes. Les tests statistiques n'ont pas permis de déterminer sur les données étudiées si MOCA-I est plus ou moins efficace que les algorithmes *XCS*, *Hider* ou *C4.5*. La modélisation utilisée dans MOCA-I a été prévue pour la classification partielle sur données asymétriques, nous allons maintenant voir si l'utilisation d'une modélisation plus adaptée à la classification standard, que nous appellerons MOCA, peut améliorer les résultats.

4.2 Étude de modélisations alternatives

Dans cette section, nous allons concevoir et évaluer des modélisations multi-objectif alternatives (MOCA : Multi-Objective Classification Algorithm) plus adaptées au problème de classification binaire standard. Nous commencerons tout d'abord par recenser les mesures utilisées

dans la littérature, pour ensuite réaliser une analyse par composantes principales (ACP) pour déterminer les objectifs candidats pour une modélisation multi-objectif. Plusieurs modélisations seront ensuite proposées et évaluées.

4.2.1 Mesures et objectifs utilisés dans la littérature

Si l'on revient sur le récapitulatif des algorithmes de classification présenté dans le chapitre 2 (table 2.3), on peut effectuer quelques observations sur les mesures et objectifs utilisés. Un tiers des approches utilisent le *gain* ou l'*exactitude* pour construire leur solution. On retrouve également des méthodes fondées sur le *taux d'erreur* ou sur la combinaison de plusieurs objectifs comme la *sensibilité*, *spécificité*, *confiance* ou le *support*. Lorsqu'il s'agit de mesurer les performances des algorithmes de classification, c'est souvent l'*exactitude* qui est utilisée. Ainsi, Demsar *et al.* ont analysé des articles acceptés à la conférence ICML (International Conference on Machine Learning). Selon eux, dans 70% à 80% des cas en fonction des années c'est l'*exactitude* qui est utilisée pour évaluer les performances [32].

4.2.2 Étude des corrélations entre objectifs candidats

Dans le chapitre 3, nous avons vu l'apport de la multi-objectivisation pour la classification partielle. Il est plus efficace d'optimiser la *f-mesure* en la découpant en plusieurs objectifs (*confiance* et *sensibilité*) plutôt que d'optimiser directement la *f-mesure*. Nous allons donc étudier comment l'*exactitude* peut être découpée en plusieurs objectifs. Rappelons tout d'abord la formule de l'*exactitude* :

$$Exactitude = \frac{TP + TN}{N}$$

Elle correspond à la proportion d'observations correctement classées (positives ou négatives), parmi l'ensemble des observations. Une première manière de mettre en place la multi-objectivisation pourrait consister à optimiser le nombre de TP (*vrais positifs*) et de TN (*vrais négatifs*). Nous allons tout d'abord vérifier si ces objectifs ne sont pas corrélés, et déterminer si d'autres objectifs ne sont pas plus intéressants. Cette étude va être réalisée à l'aide d'une analyse par composantes principales, comme réalisée précédemment dans le chapitre 3.

Protocole Nous allons étudier les corrélations entre les 15 mesures étudiées précédemment (*confiance* (Cf), $\frac{confiance}{RuleLength}$ (*CfRL*), *confiance*×*sensibilité* (CfSe),*conviction* (Conv), *cosinus* (cos), *f-mesure* (FM), *laplace* (LP), *lift*, *piatetsky-shapiro* (PS), *sensibilité* (Se), *sensibilité*×*specificité* (SeSp), *spécificité* (Sp), *négatifs découverts* (UN), *support* (S), *surprise* (Sur)) auxquelles nous ajoutons l'*exactitude* (Exa), le nombre de vrais positifs (TP) et le nombre de vrai négatifs (TN), soit un total de 18 mesures. Le protocole est identique à celui mis en place dans l'étude du chapitre 3 à l'exception des jeux de données utilisés. La nature des jeux de données utilisés – asymétriques ou non – peut avoir une influence. Aussi, dans le chapitre 3 l'ACP a été réalisée sur des jeux de données asymétriques. Maintenant, nous allons la réaliser sur 3 jeux de données plus standards du point de vue de l'asymétrie (d_{asy} proche de 0.5) : $australian_d$, $heart_d$ et *mushrooms*. Pour chacun des jeux de données, 900 règles de classification sont générées. La

moitié correspondent à des règles aléatoires, l'autre moitié correspond à des règles générées en utilisant 25 fois chacune des mesures étudiées comme objectif, à l'aide de la méthode de Hill Climbing présentée dans le chapitre 3. Pour chacune des règles générées, nous calculons ensuite la valeur prise par chacune des 18 mesures. Une analyse par composantes principales (ACP) est ensuite réalisée sur l'ensemble des valeurs obtenues (package R *factoMineR*).

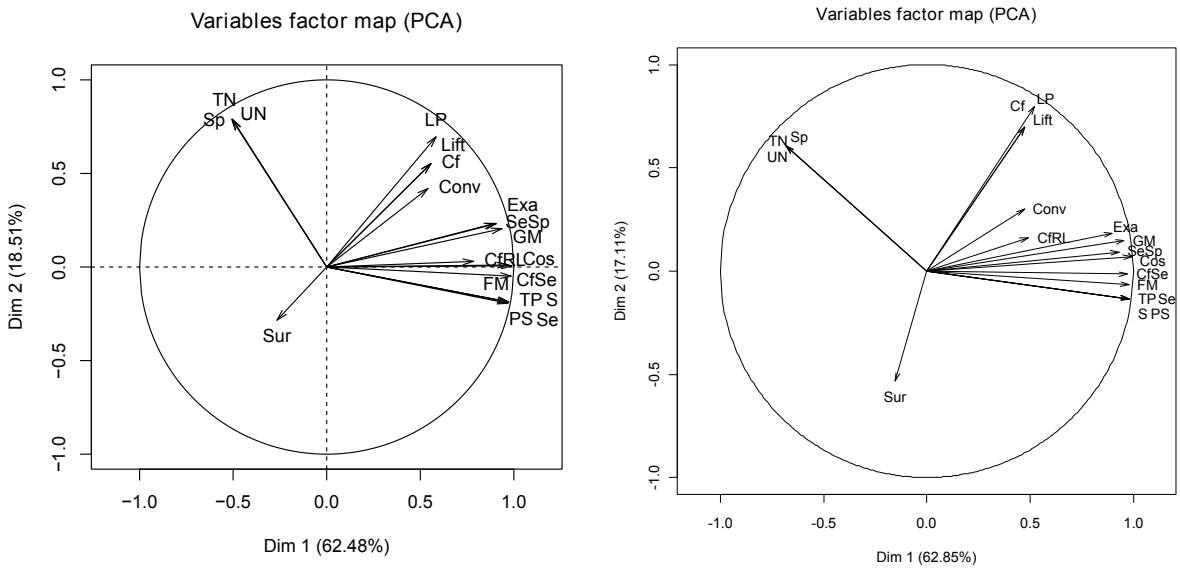


Figure 4.3: Cercle des corrélations - jeux $australian_d$ (à gauche) et $heart_d$ (à droite)

Résultats L'ACP génère pour chacun des jeux de données des cercles de corrélations. Les projections les plus représentatives sont présentées dans les figures 4.3 pour les jeux de données $australian_d$ (à gauche) et $heart_d$ (à droite), et la figure 4.4 pour le jeu de données *mushrooms*. Ces projections sur l'axe 1 et 2 ont une inertie de respectivement de 80.99%, 79.96% et 81.82%, ce qui indique une bonne représentation des données étudiées. Les mesures les plus proches du cercle sont les mieux représentées, les mesures plus éloignées comme la *surprise* (Sur) ou la *conviction* (Conv) ne permettent pas d'analyse. Les groupes obtenus sont assez similaires avec ceux obtenus dans le chapitre 3. On distingue 4 groupes :

- *spécificité* (Sp), *négatifs découverts* (UN), *vrais négatifs* (TN)
- *la place* (LP), *lift* et *confiance* (Cf)
- *exactitude* (Exa), *sensibilité*×*spécificité* (SeSp), *moyenne géométrique de l'exactitude* (GM)
- *cosinus* (Cos), *f-mesure* (FM), *vrais positifs* (TP), *sensibilité* (Se), *support* (S), *piatetsky-shapiro* (PS)

Sur les jeux de données $australian_d$ et $heart_d$ les deux derniers groupes sont confondus. Sur le jeu de données *mushrooms* l'*exactitude* (Exa) et la *moyenne géométrique de l'exactitude* (GM) se rapprochent plutôt du 2ème groupe. Ces deux mesures sont très proches, ce qui est cohérent

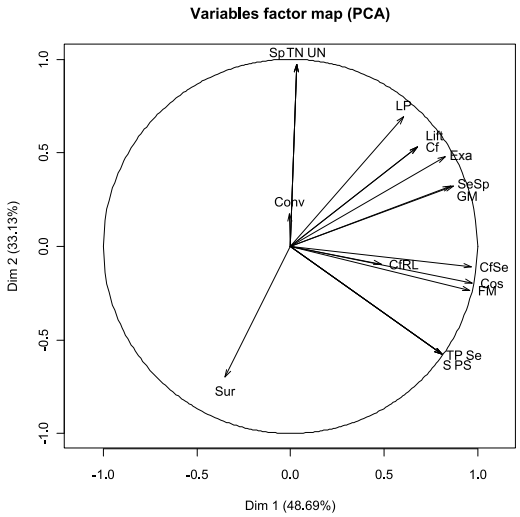


Figure 4.4: Cercle des corrélations - Règles de classification pour le jeux de données *mushrooms*

avec leur fonction, la *moyenne géométrique de l'exactitude* (GM) est en effet une version de l'*exactitude* adaptée pour gérer une éventuelle asymétrie entre les classes à prédire. Dans les jeux de données étudiés ici, il n'y a pas d'asymétrie, elles obtiennent donc les mêmes résultats. Si l'on revient sur les deux mesures candidates à la multi-objectivisation, on observe que les *vrais négatifs* (TN) sont confondus avec la *spécificité* et les *vrais positifs* (TP) avec la *sensibilité*. Ces deux mesures ne sont pas corrélées car elles appartiennent à des groupes différents. Elles devraient faire de bons objectifs pour la multi-objectivisation car cela signale qu'elles mesurent des effets différents. De plus, ces groupes ne sont pas à l'opposé l'un de l'autre, ce qui signale qu'ils ne sont pas anti-corrélés. Il est donc possible d'utiliser les *vrais positifs* (TP) et les *vrais négatifs* (TN) comme objectifs à la place de l'*exactitude*. Afin de se rapprocher des mesures utilisées dans la littérature, nous optimiserons plutôt la *sensibilité* et la *spécificité*, qui, comme nous venons de le voir, sont équivalentes respectivement aux *vrais positifs* (TP) et *vrais négatifs* (TN). Nous allons maintenant proposer 2 modèles fondés sur les observations réalisées : $MOCA_{SeSpMDL}$ et $MOCA_{ExaMDL}$ qui vont tous deux être présentés et évalués.

4.2.3 Modèle Multi-objectif $MOCA_{SeSpMDL}$ (sensibilité, spécificité, nombre de termes)

Nous venons de voir que l'*exactitude* pouvait être décomposée en deux objectifs : maximiser la *sensibilité* et maximiser la *spécificité*. À l'instar de ce qui a été réalisé pour *MOCA-I*, un troisième objectif est ajouté afin d'appliquer le principe de description de longueur minimale (MDL). Dans la suite, nous appellerons ce modèle $MOCA_{SeSpMDL}$. Il consiste donc à optimiser les trois objectifs suivants :

- Maximiser la *sensibilité*
- Maximiser la *spécificité*
- Minimiser le nombre de termes

La solution finale est choisie en sélectionnant la solution qui obtient la meilleure *exactitude* sur les données d'apprentissage. Le reste de la modélisation est strictement identique à celle qui a été présentée pour MOCA-I (voir section 3.1.2 et suivantes).

4.2.3.1 Évaluation du meilleur DMLS

Table 4.3: Modèle $MOCA_{SeSpMDL}$ - choix du meilleur DMLS - exactitude moyenne obtenue sur 25 exécutions

	$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$
$adult_d$	**0.838**	0.838	0.837	0.837
tst	0.837	**0.837**	0.837	0.837
$australian_d$	0.900	**0.900**	0.896	0.897
tst	0.839	0.840	**0.849**	0.841
$breast$	0.846	0.847	0.846	**0.847**
tst	0.720	**0.728**	0.711	0.726
crx_d	0.895	0.895	**0.895**	0.895
tst	0.865	0.865	**0.869**	0.868
$heart_d$	0.925	0.926	0.921	**0.926**
tst	0.719	0.747	0.743	**0.747**
$german_d$	**0.793**	0.792	0.776	0.780
tst	0.734	**0.736**	0.725	0.722
$hepatisis_d$	0.999	0.999	0.999	**0.999**
tst	0.840	**0.868**	0.848	0.848
$horsecolic_d$	0.923	0.923	0.922	**0.924**
tst	**0.828**	0.826	0.823	0.818
$votes$	0.983	**0.983**	0.982	0.982
tst	0.945	0.953	0.951	**0.954**
$mushrooms$	1.000	**1.000**	1.000	1.000
tst	1.000	**1.000**	1.000	1.000

De la même manière que l'étude détaillée de MOCA-I réalisée précédemment dans le chapitre 3, nous allons maintenant déterminer expérimentalement quelle version de DMLS est la plus efficace avec la modélisation $MOCA_{SeSpMDL}$. Quatre différentes variantes de DMLS (DMLS $1 \cdot 1_{\succ}$, DMLS $1 \cdot *$, DMLS $* \cdot 1_{\succ}$ et DMLS $* \cdot *$) vont être évaluées. L'objectif est de déterminer la variante de DMLS qui offre les meilleures performances sur la majorité des jeux de données, avec une bonne capacité de généralisation (évaluation sur les données de test). Chacun des DMLS est utilisé avec le paramétrage qui a été déterminé dans le chapitre 3. Le protocole

est similaire à celui utilisé dans le chapitre 3. DMLS $(* \cdot *)$, qui est la version la plus longue à atteindre un optimum local, est utilisé comme référence. Les autres DMLS auront donc droit au même temps d'exécution que celui nécessaire à DMLS $(* \cdot *)$ pour atteindre un optimum. Les 4 versions sont exécutées sur les 10 jeux de données de classification binaire standard, en validation croisée à 5 partitions, avec 5 exécutions par partition soient 25 exécutions par jeu de données.

La table 4.3 donne l'*exactitude* moyenne obtenue par chacune des variantes de DMLS avec le modèle $MOCA_{SeSpMDL}$ sur les 10 jeux de données de classification standard. L'*exactitude* est donnée à la fois sur les données d'apprentissage (1ère ligne de chaque jeu de données) et sur les données de test (2ème ligne de chaque jeu de données). On observe qu'aucune variante de DMLS ne semble efficace sur la majorité des jeux de données. Au mieux, DMLS $1 \cdot *$ donne les meilleurs résultats sur les données de test, sur la moitié des jeux de données. Les 4 variantes donnent cependant des résultats très similaires ; la différence la plus forte s'exprime sur le jeu de données $heart_d$, où DMLS $1 \cdot *$ améliore les résultats de DMLS $* \cdot 1_{\succ}$ de 2%. Le phénomène de sur-apprentissage semble moins présent que dans le chapitre précédent, mais reste tout de même du même ordre sur les jeux de données *breast*, $heart_d$, $hepatisis_d$ et $horsecolic_d$.

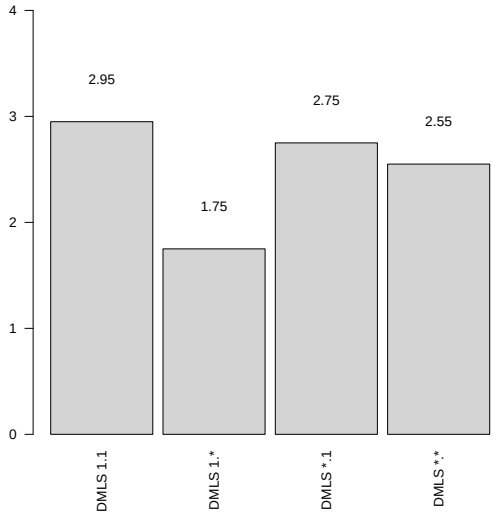


Figure 4.5: Rangs moyens obtenus par les versions de DMLS avec le modèle $MOCA_{SeSpMDL}$ sur les jeux de données de classification « classique »

Voyons maintenant les performances sur les données de test, tous jeux de données confondus. La figure 4.5 donne les rangs moyens obtenus par les différentes versions de DMLS avec la modélisation $MOCA_{SeSpMDL}$, sur chacun des jeux de données, en utilisant les données de test. On observe que DMLS $1 \cdot *$ obtient le plus faible rang moyen sur les données de test, ce qui signifie qu'il offre plus souvent de meilleures performances que les autres approches. Le test de Friedman indique que les différentes versions de DMLS sont statistiquement identiques du

point de vue des rangs moyens ($\alpha = 0.05$). Nous privilégions tout de même la version de DMLS qui obtient le meilleur rang moyen, DMLS $1 \cdot *$. Il s'agit de la même version de DMLS que celle qui s'est avérée précédemment être la plus performante pour la modélisation MOCA-I.

4.2.4 Modèle Multi-objectif $MOCA_{ExaMDL}$ (exactitude, nombre de termes)

Comme nous l'avions vu dans l'état de l'art sur la multi-objectivisation, une autre manière d'appliquer la multi-objectivisation sur l'*exactitude* est d'utiliser un objectif complémentaire (helper objective). Nous proposons donc cette fois d'ajouter le MDL comme objectif supplémentaire, ce qui donne les objectifs suivants :

- maximiser l'*exactitude*
- minimiser le nombre de termes

Dans la suite, ce modèle est appelé $MOCA_{ExaMDL}$. Le choix de la solution finale parmi l'archive des solutions est identique à ce qui a été proposé pour $MOCA_{SeSpMDL}$: la solution de meilleure *exactitude* sur les données d'apprentissage est sélectionnée.

4.2.4.1 Évaluation du meilleur DMLS

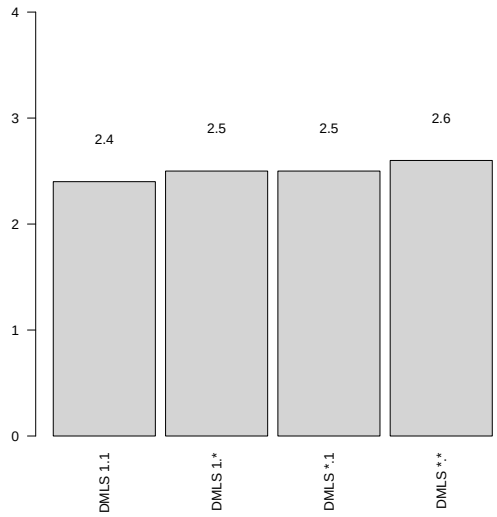


Figure 4.6: Rangs moyens obtenus sur les données de test par les versions de DMLS avec le modèle $MOCA_{ExaMDL}$ (jeux de données de classification standard)

De la même manière que pour le modèle $MOCA_{SeSpMDL}$, nous allons maintenant déterminer la meilleure version de DMLS à utiliser pour le modèle $MOCA_{ExaMDL}$. Par conséquent, le protocole expérimental est strictement identique à celui utilisé pour le modèle $MOCA_{SeSpMDL}$.

Nous avons récapitulé dans la table 4.4 l'*exactitude* moyenne obtenue par chacune des variantes de DMLS associées au modèle $MOCA_{ExaMDL}$ sur l'ensemble des jeux de données de

Table 4.4: Modèle $MOCA_{ExaMDL}$ - choix du meilleur DMLS - exactitude moyenne obtenue sur 25 exécutions

	$1 \cdot 1_{\succ}$	$1 \cdot *$	$* \cdot 1_{\succ}$	$* \cdot *$
$adult_d$	0.840	0.840	**0.840**	0.840
tst	0.839	0.839	**0.839**	0.839
$australian_d$	**0.875**	0.874	0.874	0.874
tst	0.847	0.848	**0.849**	0.843
$breast$	0.810	0.804	**0.814**	0.807
tst	0.737	**0.743**	0.738	0.730
crx_d	**0.871**	0.870	0.869	0.869
tst	**0.861**	0.858	0.858	0.859
$heart_d$	0.869	0.864	**0.871**	0.867
tst	0.727	0.736	**0.756**	0.742
$german_d$	0.773	0.770	**0.773**	0.773
tst	0.721	**0.724**	0.715	0.720
$hepatisis_d$	0.976	0.968	**0.978**	0.970
tst	**0.855**	0.838	0.850	0.840
$horsecolic_d$	0.901	0.897	**0.902**	0.899
tst	0.843	**0.843**	0.842	0.836
$votes$	**0.960**	0.958	0.958	0.957
tst	0.941	0.942	0.937	**0.944**
$mushrooms$	0.998	0.998	0.998	**0.998**
tst	0.998	0.998	0.998	**0.998**

classification standard. Chaque colonne indique les résultats obtenus par une variante de DMLS sur les 10 jeux de données, à la fois sur les données d'apprentissage et de test. Les différentes versions de DMLS obtiennent encore des résultats sur les données de test très similaires, avec au maximum une différence de l'ordre de 4% entre les résultats de DMLS $1 \cdot 1_{\succ}$ et de DMLS $* \cdot 1_{\succ}$ sur le jeu de données $heart_d$. Sur les données de test, aucune version de DMLS ne s'impose sur tous les jeux de données. Au mieux, DMLS $1 \cdot *$ et DMLS $* \cdot 1_{\succ}$ obtiennent chacun les meilleurs résultats sur 3 jeux de données parmi les 10.

Regardons maintenant les rangs moyens obtenus sur les données de test, afin de déterminer si une version de DMLS se démarque des autres sur l'ensemble des jeux de données. Ces rangs moyens sont représentés dans la figure 4.6. Les rangs moyens obtenus par les différentes versions de DMLS sont tous très similaires. Cette observation est confirmée par le test de Friedman qui accepte l'hypothèse H_0. Ce qui indique que les rangs obtenus sont statistiquement similaires (avec *p-value* > 0.05). Nous retenons la version DMLS $1 \cdot 1_{\succ}$ qui obtient le meilleur rang sur les données de test.

4.2.5 Comparaison à MOCA-I et à la littérature

Nous venons de déterminer pour chacun des deux modèles $MOCA_{SeSpMDL}$ et $MOCA_{ExaMDL}$ la version de DMLS la plus efficace, respectivement DMLS $1 \cdot *$ et DMLS $1 \cdot 1_{\succ}$. Nous allons maintenant comparer les performances de ces deux modèles à celles de MOCA-I et de sa variante *MOCA-I (MaxExa)*, et finalement aux résultats obtenus par les algorithmes de la littérature *C4.5*, *Hider* et *XCS*.

4.2.5.1 Comparaison à MOCA-I

Les performances des 4 modèles ($MOCA_{SeSpMDL}$, $MOCA_{ExaMDL}$, MOCA-I et MOCA-I (MaxExa)) sont évaluées sur 10 jeux de données de classification. Chaque modèle est exécuté 25 fois par jeu de données : 5 exécutions par partition, en validation croisée à 5 partitions. Pour chacun des modèles, le critère d'arrêt correspond au critère d'arrêt naturel de DMLS. L'exécution s'arrête donc dès que l'algorithme atteint un optimum local, ce qui signifie que l'archive des solutions ne peut plus être améliorée.

Table 4.5: Comparaison des performances des modèles $MOCA$ et MOCA-I - exactitude moyenne obtenue sur 10 jeux de données

	$MOCA_{SeSpMDL}$	$MOCA_{ExaMDL}$	MOCA-I	MOCA-I (MaxExa)
$adult_d$	0.838	0.838	0.825	**0.848**
tst	0.837	0.837	0.821	**0.838**
$australian_d$	0.900	0.900	0.916	**0.917**
tst	**0.840**	0.839	0.829	0.834
breast	0.847	0.846	0.864	**0.864**
tst	**0.728**	0.720	0.718	0.708
crx_d	0.895	0.895	0.902	**0.902**
tst	**0.865**	0.865	0.854	0.853
$heart_d$	0.926	0.925	0.932	**0.932**
tst	**0.747**	0.719	0.745	0.739
$german_d$	0.792	0.793	0.821	**0.824**
tst	**0.736**	0.734	0.725	0.721
$hepatisis_d$	**0.999**	**0.999**	**0.999**	**0.999**
tst	**0.868**	0.840	0.855	0.858
$horsecolic_d$	**0.923**	0.923	0.918	0.918
tst	0.826	**0.828**	0.826	0.822
votes	**0.983**	0.983	0.982	0.982
tst	**0.953**	0.945	0.943	0.946
mushrooms	**1.000**	1.000	1.000	1.000
tst	**1.000**	1.000	0.999	0.999

La table 4.5 donne l'*exactitude* moyenne obtenue sur les 10 jeux de données de classifica-

tion standard par les deux modélisations proposées pour MOCA et MOCA-I. La modélisation $MOCA_{SeSpMDL}$ obtient la meilleure *exactitude* sur les données de test, sur la majorité des jeux de données. La variante de MOCA-I (MOCA-I (MaxExa)) obtient la meilleure *exactitude* sur les données d'apprentissage, sur la majorité des jeux de données. Elle semble cependant plus sujette au sur-apprentissage et par conséquent n'obtient pas les meilleurs résultats sur les données de test. L'*exactitude* moyenne varie peu d'une modélisation à une autre, avec une différence au plus fort de 3%. Voyons maintenant si les meilleurs résultats de la modélisation $MOCA_{SeSpMDL}$ se confirment sur les rangs moyens. La figure 4.7 donne les rangs moyens obte-

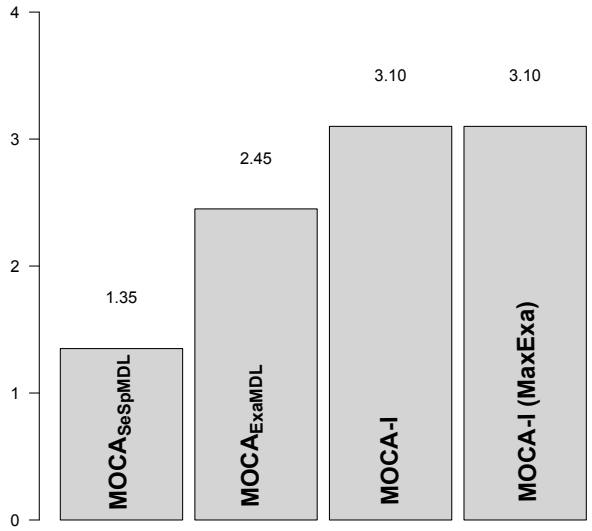


Figure 4.7: Rangs moyens obtenus sur les données de test par les différents modèles de MOCA et de MOCA-I (jeux de données de classification standard)

nus sur les données de test, pour les modélisations proposées de MOCA et MOCA-I. Les rangs moyens confirment l'observation réalisée précédemment, $MOCA_{SeSpMDL}$ offre les meilleurs résultats sur l'ensemble des jeux de données, ce qui se traduit par un rang moyen peu élevé. Il est suivi par le modèle $MOCA_{ExaMDL}$ qui semble donner de meilleurs résultats que la modélisation MOCA-I et sa variante. Les rangs moyens obtenus sont statistiquement différents selon le test de Friedman ($\alpha = 0.05$), nous pouvons donc affiner la comparaison avec des tests post-hoc. La table 4.6 donne les résultats des comparaisons statistiques de $MOCA_{SeSpMDL}$ – qui a obtenu les meilleurs résultats sur l'ensemble des jeux de données – aux autres variantes de MOCA et MOCA-I.

Dans cette table, on retrouve les résultats des comparaisons statistiques 2 à 2 de $MOCA_{SeSpMDL}$ aux autres modèles, à l'aide du test de Wilcoxon associé à une correction de Holm. On observe que $MOCA_{SeSpMDL}$ est statistiquement plus performant que MOCA-I et sa variante. En revanche, les données étudiées ne permettent pas de déterminer s'il est statistiquement plus performant que $MOCA_{ExaMDL}$. Nous allons maintenant comparer les résultats de $MOCA_{SeSpMDL}$ à ceux de la littérature.

Table 4.6: Comparaison statistique de $MOCA_{SeSpMDL}$ à MOCA-I et $MOCA_{ExaMDL}$ sur les données de test - tests de Wilcoxon + correction de Holm

	$MOCA_{SeSpMDL}$
$MOCA_{ExaMDL}$	$\equiv$ (0.058)
MOCA-I	$\succ$ (0.007)
MOCA-I($MaxExa$)	$\succ$ (0.007)

4.2.5.2 Comparaison à la littérature

$MOCA_{SeSpMDL}$ va être comparé à 3 algorithmes de la littérature : *Hider*, *C4.5* et *XCS*, qui ont été identifiés auparavant comme ceux donnant les meilleurs résultats. Comme précédemment, les algorithmes de la littérature sont exécutés dans le framework KEEL, avec la configuration recommandée par leurs auteurs. Le critère d'arrêt pour $MOCA_{SeSpMDL}$ correspond au temps nécessaire pour atteindre l'optimum local. Pour les autres algorithmes, le critère d'arrêt dépend directement de la configuration recommandée (nombre maximal d'itérations ou critère d'arrêt naturel). Chaque algorithme est exécuté 25 fois par jeu de données : 5 fois sur une validation croisée à 5 partitions.

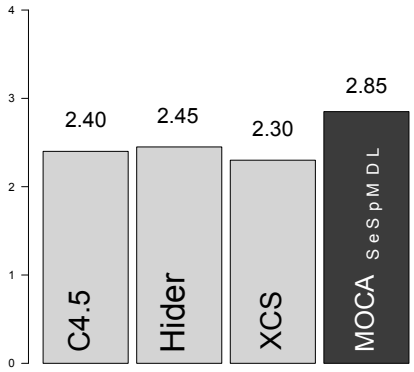


Figure 4.8: Rangs moyens obtenus sur les données de test par les différents modèles de MOCA et algorithmes de la littérature (jeux de données de classification « classique »)

La table 4.7 donne l'*exactitude* moyenne obtenue sur les jeux d'apprentissage et de test par $MOCA_{SeSpMDL}$ et les algorithmes de la littérature, sur les 10 jeux de données de classification classique. Sur la majorité des jeux de données, les algorithmes obtiennent des performances assez équivalentes sur les données de test. Sur certains jeux de données les différences sont un peu plus poussées, avec une différence de l'ordre de 3% sur *breast*, 6% sur $adult_d$ et 15% sur $horsecolic_d$. Aucun algorithme n'obtient les meilleurs résultats sur données de test sur la majorité des jeux

Table 4.7: Comparaison des résultats de $MOCA_{SeSpMDL}$ à ceux de la littérature (*Hider*, *C4.5* et *XCS*)

	$MOCA_{SeSpMDL}$	Hider	C4.5	XCS
$adult_d$	0.838	0.788	**0.856**	0.806
tst	0.837	0.788	**0.841**	0.806
$australian_d$	0.900	0.861	0.888	**0.907**
tst	0.840	0.849	0.851	**0.854**
breast	**0.847**	0.768	0.784	0.823
tst	0.728	0.754	**0.758**	0.744
crx_d	0.895	0.866	0.871	**0.897**
tst	**0.865**	0.859	0.857	0.857
$heart_d$	0.926	0.834	0.862	**0.964**
tst	0.747	0.764	0.781	**0.796**
$german_d$	0.792	0.735	0.824	**0.923**
tst	**0.736**	0.729	0.724	0.730
$hepatisis_d$	**0.999**	0.929	0.900	0.988
tst	**0.868**	0.868	0.838	0.850
$horsecolic_d$	0.923	**1.000**	0.864	0.928
tst	0.826	**1.000**	0.848	0.848
votes	**0.983**	0.970	0.970	0.970
tst	0.953	**0.969**	0.956	0.957
mushrooms	0.999	0.985	**1.000**	0.999
tst	0.999	0.984	**1.000**	0.999

de données, au mieux $MOCA_{SeSpMDL}$ et *C4.5* obtiennent les meilleurs résultats sur 3 des 10 jeux de données. Regardons maintenant les rangs moyens obtenus sur les données de test afin de déterminer si un algorithme s'en sort mieux que les autres sur l'ensemble des jeux de données.

La figure 4.8 donne les rangs moyens obtenus sur les données de test par $MOCA_{SeSpMDL}$ et les algorithmes *C4.5*, *Hider* et *XCS*. Les rangs moyens semblent équivalents, avec un rang moyen un peu moins intéressant pour $MOCA_{SeSpMDL}$. Le test de Friedman ne permet pas de rejeter l'hypothèse H_0 qui affirme que les rangs moyens sont équivalents ($\alpha = 0.05$) ; $MOCA_{SeSpMDL}$ est donc statistiquement équivalent à *C4.5*, *Hider* et *XCS*. La modélisation $MOCA_{SeSpMDL}$ (sensibilité, spécificité, nombre de termes) permet donc d'atteindre des performances similaires aux meilleurs algorithmes de la littérature.

4.3 Préconisations d'utilisation de MOCA-I

Dans la section précédente, nous avons vu que MOCA est plus efficace que MOCA-I pour gérer les jeux de données de classification standard, et donne des résultats comparables à ceux de la littérature. Une question se pose maintenant : à partir de quand est-il plus efficace d'utiliser MOCA-I plutôt que MOCA ? Ce qui revient à donner une définition précise de l'asymétrie entre les classes à prédire : à partir de quel degré d'asymétrie commence-t-on à rencontrer des problèmes ? Nous allons répondre à ces questions de deux manières. Tout d'abord d'une manière théorique en étudiant le comportement de la *f-mesure* et de l'*exactitude* en fonction du degré d'asymétrie. Ensuite, nous comparerons de manière expérimentale les résultats de *MOCA* et *MOCA-I* sur des jeux de données avec des degrés d'asymétrie variables. Enfin, nous donnerons les préconisations d'utilisation de MOCA et MOCA-I en fonction du degré d'asymétrie.

4.3.1 Analyse théorique de l'impact du degré d'asymétrie

Nous allons maintenant analyser le comportement de la *f-mesure* – utilisée dans *MOCA-I* – et de l'*exactitude* – utilisée dans $MOCA_{SeSpMdl}$ – en fonction de la répartition de la classe à prédire parmi les observations.

4.3.1.1 Définitions

Dans la suite, nous désignerons la classe à prédire par C_{pred}

Table 4.8: Matrice de confusion

		Classe réelle	
		C_{pred}	$\overline{C_{pred}}$
Classe prédite	C_{pred}	TP	FP
	$\overline{C_{pred}}$	FN	TN

Revenons tout d'abord sur les bases de la classification. Comme évoqué précédemment dans le chapitre 2, la qualité d'une classification est évaluée à l'aide de la matrice de confusion présentée dans la table 4.8. Elle est obtenue en comparant la prédiction trouvée par le classifieur (colonne de gauche) avec la classe réelle, sur un ensemble d'observations données. TP (vrais positifs) et TN (vrais négatifs) correspondent au nombre d'observations correctement classées, respectivement les *vrais positifs* et les *vrais négatifs*, ce qui se produit lorsque la classe prédite correspond à la classe réelle. Lorsque ce n'est pas le cas, on se trouve en face de FP (*faux positifs*) lorsque le classifieur identifie un C_{pred} à tort ; ou de FN (*faux négatifs*) lorsqu'il classe un C_{pred} comme $\overline{C_{pred}}$. Le nombre total d'observations est désigné par N, qui correspond également à la somme de toutes les classifications ($TP + FP + FN + TN$)

Comme défini précédemment, le degré d'asymétrie (d_{asy}) correspond au pourcentage des

observations du jeu de données qui ont la classe C_{pred} :

$$d_{asy} = \frac{|C_{pred}|}{N}.$$

Dans la suite, nous allons évaluer le comportement de la *f-mesure* ($\beta = 1$) et de l'*exactitude* qui se calculent de la manière suivante :

$$exactitude = \frac{TP + TN}{N}$$

$$\text{f-mesure} = \frac{2 \times \frac{TP}{TP+FP} \times \frac{TP}{TP+FN}}{\frac{TP}{TP+FP} + \frac{TP}{TP+FN}}$$

4.3.1.2 Analyse du comportement de la F-mesure et de l'Exactitude

Nous allons maintenant analyser le comportement de la *f-mesure* et de l'*exactitude* lorsque d_{asy} varie. Pour ce faire, nous avons généré 255 matrices de confusion. Pour construire chacune d'entre-elles, nous avons fixé $N = 4000$. Nous avons également fait varier le *taux de vrais positifs* (TPR) pour indiquer la proportion de C_{pred} détectés par le classifieur.

$$TPR = \frac{TP}{TP + FN}.$$

Enfin, nous avons fait varier un degré d'erreur d_{err} qui permet l'introduction d'erreurs dans la matrice de confusion (FP). Cela permet de simuler un classifieur qui ajoute des règles qui apportent à la fois de bonnes classifications (TP) et des mauvaises (FP).

$$d_{err} = \frac{FP}{TP}$$

Lorsque $d_{err} = 0$ le classifieur est très efficace et ajoute uniquement des règles qui détectent correctement de nouvelles observations C_{pred} sans se tromper. Lorsque d_{err} augmente, le classifieur apporte des règles qui font également des erreurs (ce qui est plus proche de la réalité, où l'algorithme peut trouver une règle qu'il va ensuite améliorer). Ainsi, à $d_{err} = 0.5$ lorsque le classifieur ajoute une règle qui classifie de nouvelles observations, pour x observations avec C_{pred} correctement détectées elle va également amener $\frac{x}{2}$ faux positifs.
Les valeurs des différents paramètres influant sur le calcul de la matrice de confusion sont les suivantes :

- $d_{asy} \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99, 0.995, 0.999\}$
- $TPR \in \{0, 0.25, 0.5, 0.75, 1\}$
- $d_{err} \in \{0, 0.25, 0.5\}$

Résultats Nous avons regroupé dans la figure 4.9 l'évolution de l'*exactitude* (graphiques de gauche) et de la *f-mesure* (graphiques de droite) en fonction du degré d'asymétrie d_{asy}. Pour les graphiques du haut, nous avons considéré un classifieur qui ajoute uniquement des règles qui améliorent strictement la classification de C_{pred}, sans apporter de *faux positifs*, ce qui se traduit par $d_{err} = 0$. Pour les graphiques du bas, on considère que l'on introduit des règles

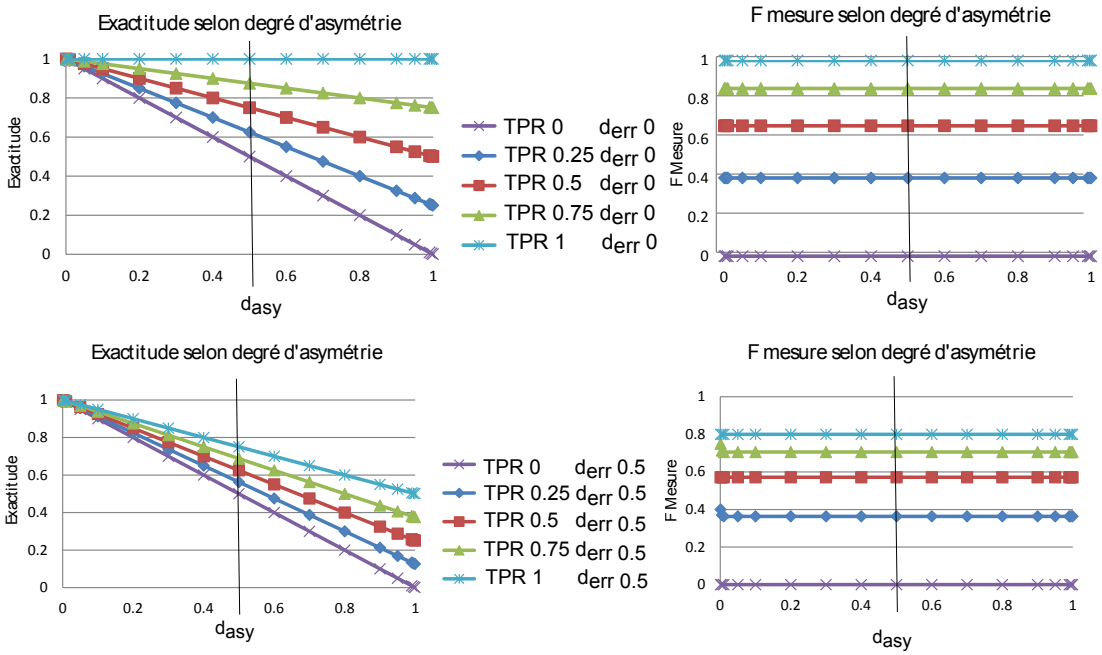


FIGURE 4.9: Étude de l'impact du degré d'asymétrie de la classe à prédire sur le comportement de l'*exactitude* (à gauche) et de la *f-mesure* (à droite) - règles améliorant strictement les TP (en haut) ou apportant également des FP (en bas)

qui contiennent un peu d'erreurs. Par exemple, nous avons considéré $d_{err} = 0.5$: à chaque fois que 2 *vrais positifs* (TP) sont découverts, nous introduisons également un *faux positif* (FP). Ensuite, pour chaque graphique nous représentons l'évolution de la *f-mesure* et de l'*exactitude* selon différents taux de *vrais positifs* (TPR).

Dans l'ensemble des graphiques, l'augmentation de la TPR (et donc du nombre de vrais positifs) augmente la valeur de la *f-mesure* et de l'*exactitude*. On remarque que l'*exactitude* est très sensible au degré d'asymétrie d_{asy} : plus il est faible, c'est-à-dire moins la classe C_{pred} est représentée, et moins l'*exactitude* récompense l'amélioration du *taux de vrais positifs* (TPR). La *f-mesure*, elle, n'apparaît pas sensible au degré d'asymétrie. Enfin, lorsque l'augmentation de la TPR amène également quelques erreurs (graphiques du bas), l'*exactitude* semble encore plus sensible au degré d'asymétrie.

Nous allons maintenant quantifier l'amélioration de l'*exactitude* et de la *f-mesure* en fonction du degré d'asymétrie. L'objectif est de déterminer à partir de quel degré d'asymétrie d_{asy} l'*exactitude* récompense le plus l'amélioration du TPR que la *f-mesure* et est donc plus pertinent à utiliser en tant que fonction objectif d'une méthode d'optimisation. Pour un degré d'erreur d_{err} donné, nous avons quantifié l'amélioration de la *f-mesure* et de l'*exactitude* lors du passage d'une matrice de confusion de 0 TPR à 0.25 TPR ; 0.25 à 0.5 TPR ; 0.5 à 0.75 TPR et 0.75 à 1 TPR. Cela permet de simuler l'amélioration des règles lors de la construction du classifieur :

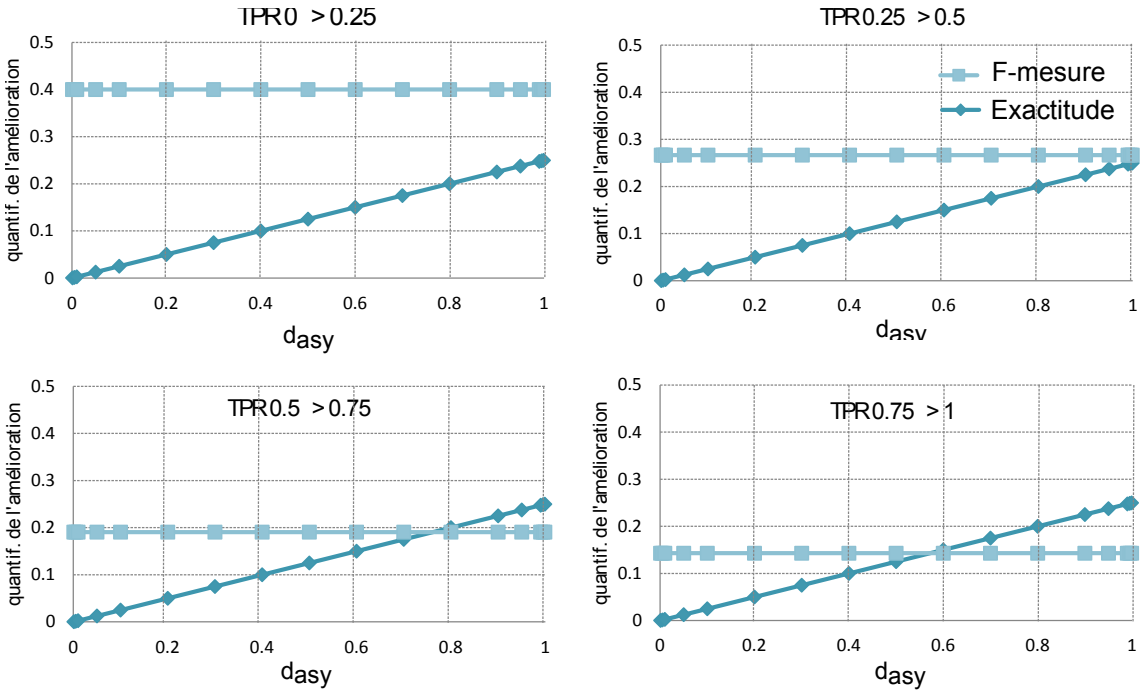


Figure 4.10: Quantification de l'amélioration de l'*exactitude* et de la *f-mesure* en fonction du degré d'asymétrie.

au début de la construction, on commence avec des règles peu efficaces ($TPR = 0$), lorsque l'on cherche de nouvelles règles, de faibles améliorations vont être proposées, ce qui obtiendra des règles avec un léger mieux au niveau du TPR (0.25). En fin de construction du classifieur, on part d'un classifieur déjà efficace $TPR = 0.75$ pour encore améliorer ses performances. La figure 4.10 présente la quantification de l'amélioration de l'*exactitude* et de la *f-mesure* en fonction du degré d'asymétrie, et du type de règles évoluées (passage d'une règle peu efficace de $TPR = 0$ à $TPR = 0.25$; jusqu'à passage d'une règle très efficace de $TPR = 0.75$ à $TPR = 1$). Elle est obtenue en calculant la différence entre la mesure obtenue deux valeurs de TPR, tout en normalisant avec la valeur obtenue pour une classification parfaite ($TPR = 1$). Nous fournissons uniquement les graphiques pour $d_{err} = 0$; les graphiques pour les autres valeurs de d_{err} ont un comportement identique.

On peut observer que la *f-mesure* récompense plus fortement les améliorations sur les règles faibles ($TPR\ 0 -> 0.25$) que sur les règles plus fortes ($TPR\ 0.25 -> 0.5$; $TPR\ 0.5 -> 0.75$ et $TPR\ 0.75 -> 1$). Plus les règles deviennent plus fortes et moins la *f-mesure* va récompenser l'amélioration. Encore une fois, elle n'est pas dépendante du degré d'asymétrie d_{asy}. L'*exactitude* offre elle des performances indépendantes quel que soit le TPR, mais reste toujours très sensible au degré d'asymétrie d_{asy} : lorsqu'il est faible, la classe à prédire C_{pred} est très peu représentée et l'amélioration de la classification est très peu récompensée. Lorsque les améliorations du TPR concernent déjà un classifieur de bon niveau ($TPR = 0.75$), l'*exactitude* récompense

mieux que la *f-mesure* au dessus de $d_{asy} = 0.6$. Lorsque le classifieur est un peu moins efficace (passage de $TPR = 0.5$ à $TPR = 0.75$), le seuil est plus important, avec $d_{asy} = 0.8$. En dessous, c'est la *f-mesure* qui récompense le plus. Difficile donc de définir un seuil de d_{asy} à partir duquel l'*exactitude* est la plus efficace, car cela dépend de l'efficacité du classifieur en cours de construction. Ainsi, la *f-mesure* devrait être plus efficace en début de construction, lorsque le classifieur n'est pas encore très efficace, et l'*exactitude* à la fin. Seule une analyse expérimentale va nous permettre de déterminer l'impact sur la globalité de la construction du classifieur. C'est l'objectif de la section suivante.

4.3.2 Analyse expérimentale de l'impact du degré d'asymétrie

Nous venons de voir qu'en théorie, l'influence du degré d'asymétrie d_{asy} dépend de la phase de recherche dans laquelle se trouve la construction du classifieur. Nous allons maintenant déterminer expérimentalement à l'échelle de toute la construction du classifieur l'impact du degré d'asymétrie d_{asy}. Pour cela, nous allons comparer les performances de $MOCA_{SeSpMDL}$ et de MOCA-I sur des jeux de données à d_{asy} variable. L'objectif est de déterminer à partir de quel seuil de d_{asy} il faut privilégier un modèle fondé sur l'*exactitude* ($MOCA_{SeSpMDL}$) ou un modèle fondé sur la *f-mesure* (MOCA-I).

Protocole $MOCA_{SeSpMDL}$ et MOCA-I vont être comparés sur 20 jeux de données de la littérature. Nous avons choisi 10 jeux de données comportant des degrés d'asymétrie variables ($0.0077 < d_{asy} < 0.44$). Nous avons ensuite obtenu 10 nouveaux jeux de données en inversant la classe à prédire C_{pred}, par exemple en prédisant la classe « negative » à la place de la classe « positive » pour le jeu $abalone19_d$. Les jeux ainsi obtenus sont préfixés par « inv- », ce qui donne dans l'exemple précédent $inv-abalone19_d$. Les deux algorithmes sont exécutés 25 fois par jeu de données : 5 fois par partition en validation croisée à 5 partitions. Le critère d'arrêt est le critère d'arrêt naturel des deux algorithmes : ils sont autorisés à s'exécuter jusqu'à ce qu'ils atteignent l'optimum local, ce qui signifie qu'ils ne peuvent plus améliorer l'archive des solutions courantes. Les performances seront évaluées à l'aide de la *moyenne géométrique de l'exactitude* (GM), qui est la mesure de performance recommandée en cas de classification binaire où une asymétrie peut être présente [50]. À l'instar de l'*exactitude* elle évalue les observations correctement classées (*vrais positifs* (TP) et *vrais négatifs* (TN)) mais elle n'est pas sensible à la répartition des classes. Elle permet de donner une importance égale aux deux prédictions C_{pred} et $\overline{C_{pred}}$. Nous n'avons pas retenu la *f-mesure* car celle-ci se focalise uniquement sur la bonne classification de la prédiction C_{pred} et n'accorde pas d'importance aux *vrais négatifs*, qui sont pourtant importants pour la classification binaire.

Résultats et discussion Nous avons rassemblé les résultats des exécutions de $MOCA_{SeSpMDL}$ et MOCA-I dans la table 4.9. Pour chaque jeu de données, nous détaillons la prédiction qui fait l'objet de la classification (C_{pred}), le degré d'asymétrie de la prédiction C_{pred} et enfin la moyenne et l'écart-type de la *moyenne géométrie de l'exactitude* obtenue par MOCA-I et

Table 4.9: Comparaison des performances de MOCA-I et $MOCA_{SeSpMDL}$ en fonction du degré d'asymétrie (d_{asy}) - Moyenne de la *moyenne géométrique de l'exactitude* sur les données d'apprentissage et de test.

Jeu	C_{pred}	d_{asy}	MOCA-I	$MOCA_{SeSpMDL}$
$abalone19_d$	positive	0.0077	**0.546 ± 0.044**	0.343 ± 0.054
tst			**0.061 ± 0.144**	0.000 ± 0.000
$w1a$	+1	0.029	**0.736 ± 0.020**	0.726 ± 0.031
tst			**0.544 ± 0.222**	0.527 ± 0.180
$yeast2vs8_d$	positive	0.0415	**0.929 ± 0.042**	0.912 ± 0.038
tst			0.617 ± 0.178	**0.667 ± 0.142**
$abalone9vs18_d$	positive	0.0565	**0.742 ± 0.024**	0.727 ± 0.031
tst			**0.531 ± 0.202**	0.508 ± 0.155
$yeast3_d$	positive	0.1038	**0.910 ± 0.012**	0.870 ± 0.035
tst			**0.865 ± 0.030**	0.818 ± 0.061
$ecoli2_d$	positive	0.1548	**0.963 ± 0.016**	0.960 ± 0.020
tst			**0.881 ± 0.046**	0.877 ± 0.036
$ecoli1_d$	positive	0.2292	**0.941 ± 0.014**	0.927 ± 0.019
tst			**0.831 ± 0.047**	0.826 ± 0.040
$haberman_d$	positive	0.2742	**0.755 ± 0.021**	0.656 ± 0.052
tst			**0.541 ± 0.099**	0.421 ± 0.092
$housevotes$	republican	0.38	**0.971 ± 0.005**	0.970 ± 0.004
tst			0.951 ± 0.017	**0.951 ± 0.017**
$australian_d$	1	0.44	**0.901 ± 0.010**	0.885 ± 0.012
tst			**0.837 ± 0.038**	0.823 ± 0.042
$inv - australian_d$	0	0.56	**0.915 ± 0.008**	0.900 ± 0.009
tst			0.824 ± 0.035	**0.842 ± 0.037**
$inv - housevotes$	democrat	0.62	0.980 ± 0.004	**0.982 ± 0.004**
tst			0.939 ± 0.025	**0.947 ± 0.025**
$inv - haberman_d$	negative	0.7258	**0.629 ± 0.036**	0.627 ± 0.044
tst			0.385 ± 0.087	**0.391 ± 0.115**
$inv - ecoli1_d$	negative	0.7708	0.907 ± 0.016	**0.912 ± 0.016**
tst			0.797 ± 0.029	**0.810 ± 0.035**
$inv - ecoli2_d$	negative	0.8452	0.952 ± 0.016	**0.957 ± 0.010**
tst			0.853 ± 0.078	**0.881 ± 0.061**
$inv - yeast3_d$	negative	0.8962	0.876 ± 0.016	**0.876 ± 0.019**
tst			0.812 ± 0.043	**0.824 ± 0.043**
$inv - abalone9vs18_d$	negative	0.9415	0.499 ± 0.051	**0.579 ± 0.059**
tst			0.447 ± 0.105	**0.461 ± 0.102**
$inv - yeast2vs8_d$	negative	0.9585	0.806 ± 0.034	**0.830 ± 0.040**
tst			0.528 ± 0.267	**0.625 ± 0.124**
$inv - w1a$	-1	0.971	0.000 ± 0.000	**0.000 ± 0.000**
tst			0.000 ± 0.000	**0.000 ± 0.000**
$inv - abalone19_d$	negative	0.9923	0.249 ± 0.058	**0.330 ± 0.043**
tst			0.000 ± 0.000	**0.000 ± 0.000**

$MOCA_{SeSpMDL}$. Pour chaque jeu de données, la première ligne correspond aux résultats obtenus sur les données d'apprentissage ; la seconde ligne à ceux obtenus sur les données de test. Nous observons que pour $d_{asy} < 0.5$, MOCA-I obtient les meilleurs résultats sur les données de test, sur la majorité des jeux de données. Pour $d_{asy} > 0.5$ la situation s'inverse, avec de meilleurs performances de $MOCA_{SeSpMDL}$. Les différences de performances entre $MOCA_{SeSpMDL}$ et MOCA-I sont souvent de l'ordre de 4%, mais on retrouve des différences plus importantes sur certains jeux de données, comme $abalone19_d$ où $MOCA_{SeSpMDL}$ ne parvient pas à classer C_{pred}, $yeast2vs8_d$ où la différence est de l'ordre de 7%, $yeast3_d$ avec 5%, $haberman_d$ avec 22% et $inv - yeast2vs8_d$ avec 15%. Sur un même jeu de données, avec le modèle MOCA-I, lorsque la classe C_{pred} est majoritaire il semble plus intéressant de prédire $\overline{C_{pred}}$ car MOCA-I semble plus efficace (en classification binaire standard l'important est juste d'arriver à distinguer C_{pred} et $\overline{C_{pred}}$). Sur le jeu de données $haberman_d$ on note une forte différence entre la prédiction de la classe minoritaire (GM de 0.541) et de la classe majoritaire (la GM chute alors à 0.385). Elle est également importante sur le jeu de données $ecoli1_d$ où la GM passe de 0.831 à 0.797 lorsque l'on prédit la classe majoritaire, $yeast3_d$ de 0.865 à 0.812, $yeast2vs8_d$ de 0.617 à 0.528, $w1a$ de 0.544 à 0. Sur les autres jeux de données, il reste toujours légèrement plus avantageux de prédire la classe minoritaire. Ceci n'est pas vérifié pour le modèle $MOCA_{SeSpMDL}$, où le choix de la prédiction ne semble pas avoir d'influence.

Nous allons maintenant tenter de déterminer plus précisément à partir de quel seuil de d_{asy}

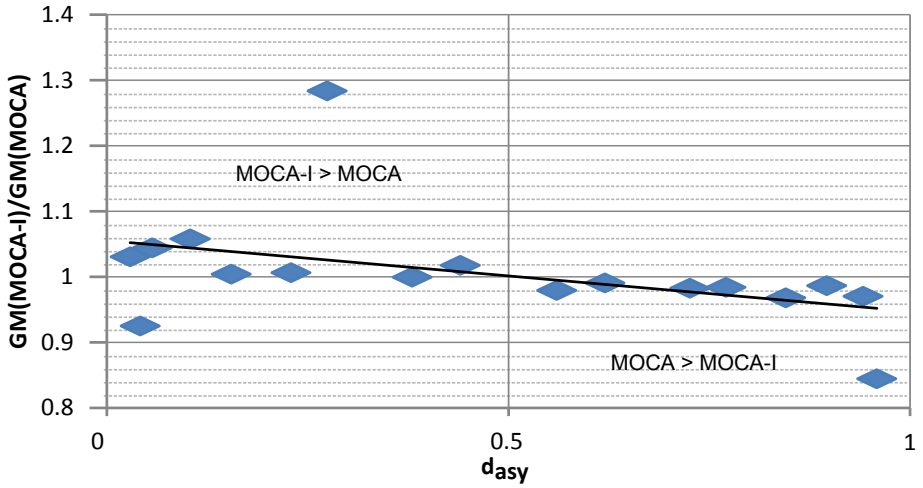


Figure 4.11: Étude de l'impact du degré d'asymétrie de la classe à prédire sur le comportement de MOCA et MOCA-I

$MOCA_{SeSpMDL}$ devient plus efficace que MOCA-I. Pour chaque jeu de données, nous avons fait le rapport entre la GM obtenue par MOCA-I et celle de $MOCA_{SeSpMDL}$. Le résultat est présenté dans la figure 4.11, où nous avons tracé chaque résultat avec la valeur associée de d_{asy} pour le jeu de données concerné. Nous avons également tracé la courbe d'interpolation afin de visualiser plus facilement la tendance. Cette courbe nous permet d'observer que lorsque $d_{asy} < 0.5$ MOCA-I est plus performant. À l'inverse, lorsque $d_{asy} > 0.5$ c'est $MOCA_{SeSpMDL}$

qui est à privilégier.

4.3.3 Conclusion

Dans cette section, nous avons étudié le comportement de MOCA-I et $MOCA_{SeSpMDL}$ en fonction du degré d'asymétrie de la classe à prédire. Cette étude a tout d'abord été réalisée de manière théorique, en étudiant le comportement de la *f-mesure* et de l'*exactitude* en fonction du degré d'asymétrie d_{asy}. Nous avons ainsi observé que la *f-mesure* n'est pas du tout sensible à d_{asy}, contrairement à l'*exactitude*. Nous avons également analysé à partir de quel d_{asy} il était plus intéressant d'utiliser la *f-mesure* ou l'*exactitude*. Cependant, ce seuil de d_{asy} est dépendant des performances du classifieur qui est amélioré. En début de construction du classifieur, lorsque celui-ci est le moins performant, la *f-mesure* récompense beaucoup plus l'amélioration du classifieur que l'*exactitude*. Lorsque le classifieur est plus performant, c'est l'*exactitude* qui récompense le mieux l'amélioration de la classification. Comme le seuil est dépendant de la progression de la construction du classifieur, nous avons réalisé une étude expérimentale sur le comportement de MOCA-I et $MOCA_{SeSpMDL}$ en fonction de d_{asy}. L'objectif est de déterminer le seuil à partir duquel il est préférable d'utiliser l'une ou l'autre approche. Les résultats ont montré qu'il est plus efficace d'utiliser MOCA-I lorsque $d_{asy} < 0.5$, et $MOCA_{SeSpMDL}$ dans l'autre cas. De plus, en classification binaire standard, lorsque $d_{asy} > 0.5$ il est plus efficace avec MOCA-I de prédire $\overline{c_{pred}}$.

4.4 Conclusion du chapitre

Dans ce chapitre, nous avons étudié si MOCA-I peut être utilisé dans un contexte de classification binaire plus classique, sur des données qui ne comportent pas de problème d'asymétrie. Les premières expérimentations ont montré que MOCA-I donne des résultats équivalents à ceux des algorithmes de la littérature, voir moins intéressants dans le cas de la variante MOCA-I (MaxExa). Dans la deuxième partie, dans l'optique d'améliorer les résultats, nous avons donc proposé des modélisations alternatives, $MOCA_{ExaMDL}$ et $MOCA_{SeSpMDL}$, plus adaptées à la classification binaire standard. Nous avons ensuite évalué les performances de celles-ci, à la fois par rapport à MOCA-I et la littérature. La modélisation $MOCA_{SeSpMDL}$ est plus efficace que MOCA-I et donne des résultats statistiquement équivalents à ceux de la littérature. Une question se pose donc : quand faut-il utiliser $MOCA_{SeSpMDL}$ plutôt que MOCA-I ? Dans la dernière partie, nous avons donc étudié le comportement des deux approches selon le degré d'asymétrie de la classe à prédire. MOCA-I est à privilégier pour un degré d'asymétrie $d_{asy} < 0.5$, ensuite c'est $MOCA_{SeSpMDL}$ qui est à privilégier. En classification binaire, MOCA-I est plus performant si on le pousse à classer la classe minoritaire. Une perspective à court terme est de mettre en place un système adaptatif qui détermine automatiquement le modèle à utiliser selon le degré d'asymétrie détecté dans le jeu de données. Une perspective supplémentaire serait d'adapter la fonction objectif au cours de la recherche : utiliser la *f-mesure* en début de recherche, là où elle est plus efficace que l'*exactitude*. Ensuite, lorsque le classifieur devient plus efficace,

la fonction objectif peut utiliser l'*exactitude*, qui devient à ce moment-là de la recherche plus efficace que la *f-mesure*. Une autre perspective serait de généraliser cette étude à des jeux de données contenant des attributs quantitatifs. MOCA-I et MOCA sont conçus initialement pour des jeux de données contenant des attributs qualitatifs uniquement. Aussi, il serait nécessaire d'implémenter au préalable une méthode de discrétisation dans MOCA-I et MOCA.

Chapitre 5

Aide à la décision

Ce chapitre se focalise sur la gestion efficace des nombreuses solutions de compromis qui sont générées par l'algorithme multi-objectif MOCA-I vu précédemment : quel ensemble de règles faut-il choisir parmi tous ceux obtenus ? Est-il plutôt intéressant de générer une nouvelle solution à partir de l'ensemble Pareto des solutions ? Nous commencerons par motiver notre approche en étudiant l'impact des méthodes de choix de la solution finale qui ont été présentées dans les chapitres précédents. Nous étudierons ensuite deux types de méthodes d'obtention d'une solution finale : le choix d'une solution et la génération d'une solution à partir de l'ensemble Pareto des solutions obtenues. Pour chacun des types de méthodes, nous proposerons plusieurs stratégies qui seront ensuite évaluées afin de déterminer les plus efficaces.

5.1 Motivations

Dans les chapitres précédents, nous avons vu que l'algorithme MOCA-I générait une liste de solutions de compromis (Pareto). Nous obtenons ainsi après chaque exécution de MOCA-I un ensemble d'ensembles de règles (ruleset) dont certains sont de bonne *confiance*, d'autres de bonne *sensibilité*, qui contiennent un nombre limité de termes ou encore des valeurs moyennes pour chacun des objectifs. La stratégie d'obtention d'une solution finale présentée dans le chapitre 3 consiste à choisir la solution qui obtient la meilleure *f-mesure* sur l'ensemble d'apprentissage, notée par la suite $f\text{-}mesure_{appr}$. Comme nous l'avions vu précédemment, l'objectif de MOCA-I est d'obtenir la solution qui obtient les meilleures performances en classification sur des données inconnues ; ce qui revient à trouver la solution qui obtient la meilleure *f-mesure* sur les données de test notée $f\text{-}mesure_{test}$. Nous allons maintenant vérifier si cette stratégie consistant à se fonder sur l'ensemble d'apprentissage est efficace : y a-t-il d'autres solutions plus performantes en terme de $f\text{-}mesure_{test}$ que celle choisie par cette stratégie ?

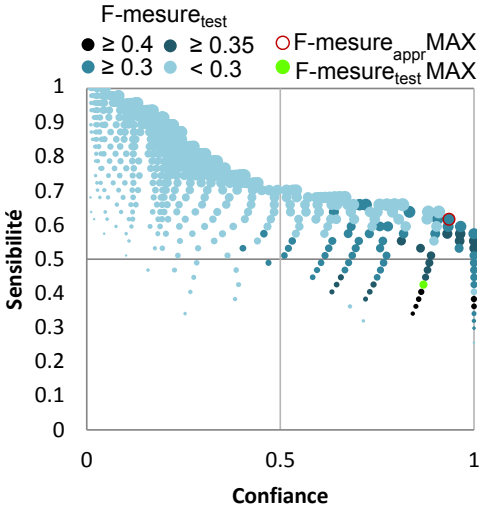


FIGURE 5.1: Projection des solutions obtenues par une exécution de MOCA-I sur le jeu de données *tia-f*

5.1.1 Illustration sur un cas réel

Dans la figure 5.1 nous avons projeté l'ensemble des solutions obtenues lors d'une exécution de MOCA-I sur un jeu de données défini (ici *tia-f*). Chaque solution est représentée par une bulle. Les différents objectifs optimisés par MOCA-I sont représentés : *confiance* en abscisse, *sensibilité* en ordonnée et la taille de chaque bulle est proportionnelle au nombre de termes que contient la solution (nombre de tests sur attributs). Nous avons représenté la $f\text{-}mesure_{test}$ obtenue par chacune des solutions à l'aide d'un code couleur. Des couleurs additionnelles sont utilisées pour représenter la solution qui a été choisie par MOCA-I (représentée par un cercle rouge sur la figure), nommée $f\text{-}mesure_{appr}MAX$ (solution avec la meilleure $f\text{-}mesure_{appr}$) ainsi que la solution idéale, qui parmi toutes les solutions obtient la meilleure $f\text{-}mesure_{test}$, intitulée f-mesure$_{test}MAX$ (représentée sous la forme d'un disque vert). On observe que les solutions obtenant les meilleurs scores sur le test se trouvent dans la région de forte *confiance* et faible nombre de termes ; elles sont représentées par une couleur foncée. La solution choisie par MOCA-I dans le chapitre 3 est choisie dans une région moins intéressante, avec une $f\text{-}mesure_{test}$ de 0.34 alors qu'il existe des solutions de meilleure $f\text{-}mesure_{test}$, comme par exemple la solution f-mesure$_{test}MAX$ qui obtient une $f\text{-}mesure_{test}$ de 0.4. Ce qui nous montre que cette stratégie de choix de solution n'est pas idéale car elle rate des solutions plus performantes, qui permettraient d'améliorer les résultats (ici de 0.34 à 0.4). Cette situation illustre bien le phénomène de sur-apprentissage : la meilleure solution sur le jeu d'apprentissage ($f\text{-}mesure_{appr}MAX$) n'est pas forcément la meilleure solution sur le jeu de test ($f\text{-}mesure_{test}MAX$). Autrement dit, la solution $f\text{-}mesure_{appr}MAX$ s'est trop spécialisée sur les données d'apprentissage et n'est plus capable de traiter des données différentes (ici les données de test). Diverses stratégies sont envisageables, comme essayer de choisir une meilleure solution, fusionner plusieurs solutions pour en obtenir une plus efficace, ou encore tenter de tenir compte du sur-apprentissage. Regardons

tout d'abord les approches utilisées dans la littérature pour exploiter les solutions générées par un ou plusieurs algorithmes.

5.1.2 Méthodes utilisées dans la littérature

On retrouve deux types de méthodes : celles qui, à l'instar de la méthode qu'utilise MOCA-I dans le chapitre 3, sont fondées sur le choix d'une ou plusieurs solutions dans l'archive des solutions obtenues, et celles qui visent à générer une solution à partir d'une fusion de toutes les solutions obtenues. Dans la suite, nous présenterons tout d'abord les travaux sur la sélection d'une solution, pour ensuite terminer sur les travaux de fusion.

Sélection d'une solution Les stratégies de sélection d'une solution les plus simples sont similaires à celle présentée dans le chapitre 3, où l'on choisit la solution qui obtient le meilleur score sur le jeu de données d'apprentissage (dans notre cas la *f-mesure*). Ainsi, dans un de leurs travaux sur les algorithmes de classification multi-objectif, Reynolds et Iglesia simulent le comportement du décideur en supposant qu'il aurait choisi la règle qui génère le moins d'erreur de classification sur le jeu d'apprentissage [92]. Dans d'autres travaux, Casillas *et al.* se focalisent sur l'ensemble de règles qui obtient la meilleure *exactitude* sur le jeu d'apprentissage, l'ensemble de règles comportant le plus petit nombre de termes ou encore l'ensemble de règles médian sur le front Pareto [24].
Des stratégies de sélection plus complexes existent, dans lesquelles la ou les solutions finales sont sélectionnées à l'aide d'un algorithme dédié. Par exemple, Ishibuchi *et al.* utilisent un algorithme multi-objectif optimisant le nombre de règles, l'*exactitude* et la taille totale de la solution finale pour sélectionner des règles parmi un ensemble de 700 000 [56]. Dans des travaux plus récents, Ishibuchi *et al.* proposent cette fois un algorithme génétique multi-objectif où les objectifs sont agrégés au sein d'une même fonction objectif à l'aide de pondérations [55]. Les travaux de Dos Santos *et al.* quant à eux [34, 35] portent sur la sélection de solutions en tenant compte du phénomène de sur-apprentissage. Dans le cadre de ces travaux, ils ont observé que dans leur algorithme génétique de recherche de règles, les règles issues des premières générations de l'algorithme présentaient moins de sur-apprentissage tout en donnant des résultats intéressants sur le jeu de test. Ils proposent également des stratégies où l'archive des solutions courante est conservée jusqu'à ce qu'un sur-apprentissage soit détecté.

Fusion de solutions Une autre approche possible est d'essayer de bénéficier de toutes les solutions obtenues, en les utilisant pour en générer une seule. Dès lors que la solution finale est un regroupement de beaucoup de règles, divers conflits vont apparaître au sein de la solution obtenue. Certaines règles vont par exemple se chevaucher ou obtenir des résultats contradictoires. Casillas *et al.* ont recensé un certain nombre des problèmes qui pouvant se produire [24]. Ainsi, de nombreux travaux utilisent les méthodes d'ensemble – il s'agit de méthodes fondées sur la combinaison de plusieurs classifieurs peu efficaces – pour résoudre ces problèmes. On retrouve à la fois des méthodes fondées sur des votes entre plusieurs classifieurs, mais également d'autres méthodes fondées sur l'ordonnancement des classifieurs. Par exemple, Ishibuchi et Nojima [53]

utilisent un vote majoritaire parmi toutes les solutions pour déterminer la classe finale d'une observation : la classe qui a été la plus prédite par les règles sera utilisée. Wang *et al.* ont eux comparé plusieurs techniques, comme un vote sur les K premières règles qui se déclenchent, le résultat de la première règle qui se déclenche, etc. Dans certaines des techniques d'ensemble, l'ordre a de l'importance. C'est pourquoi certains auteurs en tiennent compte lors de la génération de la solution finale. Ainsi, Ye *et al.* ont généré une solution de 534 règles ordonnées, à partir d'une solution initiale de 75 887 règles. Comme les règles sont ordonnées, la classe d'une observation est donnée par la première règle qui se déclenche.

5.2 Méthodes de choix d'une solution

Dans cette partie, nous proposerons plusieurs stratégies de choix d'une solution. Deux types de stratégies seront étudiées : stratégies fondées sur le jeu d'apprentissage ou tenant compte du sur-apprentissage. Ces dernières utilisent une sous-partie du jeu d'apprentissage comme jeu de validation. Enfin, les différentes stratégies proposées seront évaluées. Toutes les stratégies étudiées sont fondées sur la *f-mesure*, qui est le critère final à optimiser dans MOCA-I.

5.2.1 Stratégies fondées sur le jeu d'apprentissage

Les stratégies présentées dans la suite sont fondées uniquement sur le jeu d'apprentissage. Après avoir présenté des méthodes simples de choix de solution, nous effectuerons une étude du comportement de la *f-mesure* sur le front Pareto des solutions obtenues. Deux méthodes issues de cette étude seront ensuite proposées.

5.2.1.1 Meilleure f-mesure sur le jeu d'apprentissage (MaxFM)

Il s'agit de la stratégie utilisée dans le chapitre 3. Parmi l'ensemble des solutions de compromis $\{S_1, ...S_i, ...S_N\}$ on choisit la solution S_i qui obtient la meilleure *f-mesure* sur le jeu d'apprentissage.

5.2.1.2 F-mesure en fonction des valeurs de la confiance et sensibilité

Dans cette partie, notre objectif est d'observer l'évolution de la *f-mesure* en fonction des valeurs de *confiance* et *sensibilité* des solutions qui peuvent être obtenues en résultat de MOCA-I. Nous avons généré 100 couples de valeurs {*confiance*, *sensibilité*}, pour chacun d'entre eux nous avons calculé leur *f-mesure* et ainsi obtenu la figure 5.2. Dans cette figure, l'abscisse et l'ordonnée représentent la *confiance* et la *sensibilité* tandis que la taille des bulles représente la *f-mesure* obtenue. Dans l'illustration, le point de *confiance*= 1 et *sensibilité*= 1 obtient une *f-mesure* de 1. On observe que plus l'on se rapproche de ce point, plus la *f-mesure* est élevée. Voyons maintenant l'impact de cette figure sur le choix de la solution dans le front Pareto. Dans la figure 5.3 nous avons représenté un front Pareto représentatif de ceux issus de MOCA-I. Comme sur la figure précédente, la taille des bulles correspond à la *f-mesure* obtenue. Sur ce front Pareto, les

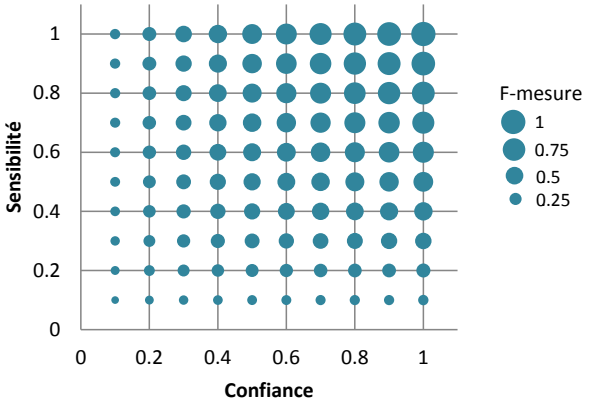


Figure 5.2: F-mesure en fonction des valeurs de la confiance et sensibilité

solutions qui vont donner la meilleure *f-mesure* se situent dans la zone de confiance élevée (qui correspond également à la zone où les *faux positifs* sont les moins nombreux). Les solutions qui sont situées dans la zone de forte *sensibilité* offrent une faible *confiance* et sont plus éloignées du point idéal (*confiance* = 1 et *sensibilité* = 1) où la *f-mesure* est maximale. Sur ce front Pareto, il faut donc se concentrer sur la zone de forte *confiance*, dans laquelle il faut cependant choisir une seule solution.

Nous avons effectué quelques tests et manipulations sur des fronts Pareto issus de plusieurs

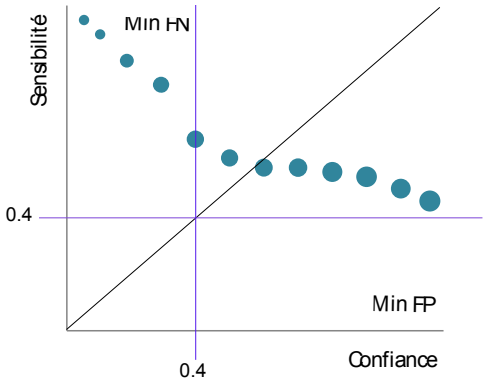


Figure 5.3: Choix d'une solution sur le front Pareto (confiance, sensibilité)

exécutions sur divers jeux de données. Nous avons ainsi remarqué que le phénomène de sur-apprentissage était moins présent dans la zone de *confiance* > 0.4 et *sensibilité* > 0.4. De plus, une fois ce seuil franchi, dans la zone de forte *confiance* les solutions maximisant la *confiance* donnent une meilleure *f-mesure* ; tandis que si le front se situe dans la zone de forte *sensibilité* la meilleure *f-mesure* est obtenue avec les solutions maximisant la *sensibilité*. Les deux méthodes proposées dans la suite sont directement inspirées de ces observations. Dans l'évaluation qui suivra, nous verrons si ces observations sont généralisables à tous les jeux de données.

5.2.1.3 Seuil sensibilité et confiance maximum (Se04MaxCf)

Il s'agit de l'opération à appliquer lorsque le front se situe dans la région de forte confiance. Tout d'abord les solutions de *sensibilité* > 0.4 sont sélectionnées. Ensuite, on conserve uniquement les solutions dont la *confiance* est maximum. À l'issue de cette sélection, s'il reste plus d'une solution on conserve uniquement celle de plus forte *f-mesure* sur le jeu d'apprentissage. Cependant, si le front est situé dans la région de forte *sensibilité* cette méthode ne sera peut-être pas aussi efficace, auquel cas la méthode qui suit est à privilégier.

5.2.1.4 Seuil confiance et sensibilité maximum (Cf04MaxSe)

S'il l'on regarde la figure 5.3 il s'agit juste de l'application de l'opération précédente lorsque le front est situé cette fois dans la région de forte sensibilité. On sélectionne tout d'abord toutes les solutions dont la *confiance* est supérieure à 0.4, parmi ces solutions on extrait la ou les solution(s) de meilleure *spécificité*. Si plusieurs solutions sont obtenues, la solution finale correspond à celle qui obtient la meilleure *f-mesure* sur le jeu d'apprentissage. Il est à noter que cette méthode peut être combinée à la méthode précédente si l'on ne sait pas dans quelle région se trouve le front : il suffit d'exécuter les deux méthodes et de prendre celle qui donne le meilleur résultat.

5.2.2 Stratégies de réduction du sur-apprentissage

Une autre approche est de tenter de maximiser la *f-mesure* sur le jeu de test. Une régression linéaire a été réalisée sur un ensemble de 400 solutions de compromis généré par MOCA-I. L'objectif était de déterminer des facteurs explicatifs de la *f-mesure* obtenue sur jeu de test ; chacun des facteurs explicatifs est généré à partir du jeu d'apprentissage. Ainsi, pour chacune des solutions de compromis nous avons fourni à l'algorithme de régression (Weka) l'ensemble des mesures obtenues sur le jeu d'apprentissage (nombre de faux positifs, vrais positifs, faux négatifs, vrai négatifs, confiance, sensibilité, f-mesure) ainsi que la f-mesure obtenue sur le jeu de test. Cependant l'algorithme de régression linéaire n'est pas parvenu à trouver un modèle fiable (taux d'erreur de 0.4). Il semble complexe de prédire un bon résultat sur le jeu de test à l'aide du jeu d'apprentissage. Dans la suite, nous allons réserver un sous-ensemble du jeu d'apprentissage au choix de la solution, ce qui permettra de tester les solutions sur des observations qui n'ont pas été utilisées pour l'apprentissage et évaluer la capacité de généralisation du modèle.

5.2.2.1 Partitionnement du jeu d'apprentissage

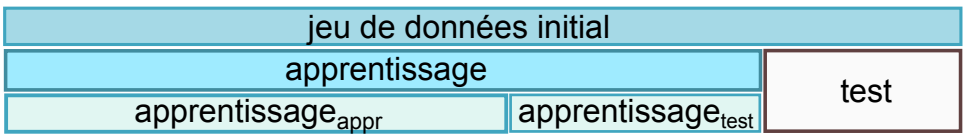


Figure 5.4: Partitionnement du jeu de données pour la détection du sur-apprentissage

Lors de l'utilisation du jeu de données d'apprentissage, nous allons réserver une partie des observations à la tâche de choix d'une solution. Dans la figure 5.4 nous proposons une illustration du partitionnement qui sera effectué sur un jeu de données déterminé. À l'instar de ce qui a été fait dans les chapitres précédents, nous utilisons une validation 5-fold, ce qui signifie que le jeu de données est séparé en 5 partitions de taille équivalente. Quatre de ces partitions sont réservées à l'algorithme pour l'apprentissage des règles et ensemble de règles. La dernière partition – dans l'illustration la partition "test" – est utilisée pour évaluer les performances de l'algorithme sur des observations inconnues. Cette répartition est illustrée dans la deuxième ligne de la figure 5.4.

Afin de pouvoir détecter le sur-apprentissage, nous allons réserver une partie des données d'apprentissage à cet effet. Le jeu d'apprentissage est séparé en 2 : 2/3 pour une partition d'apprentissage $apprentissage_{appr}$ sur lequel va être lancé MOCA-I, 1/3 dédié au choix de la solution $apprentissage_{test}$. Ce partitionnement est illustré dans la troisième ligne de la figure 5.4.

5.2.2.2 Meilleure f-mesure sur la partition $apprentissage_{test}$ (MaxFM(test))

Dans cette méthode, MOCA-I est exécuté sur la partition $apprentissage_{appr}$ du jeu d'apprentissage. Une fois l'exécution terminée, les solutions obtenues sont évaluées sur la partition $apprentissage_{test}$ du jeu d'apprentissage. La solution qui obtient le meilleur résultat est choisie comme solution finale. Comme elle a été choisie pour ses capacités à traiter des observations différentes de celles utilisées pour l'apprentissage, elle devrait donc obtenir un bon score sur d'autres observations (celles du jeu de test). Cependant le risque de sur-apprentissage n'est pas tout à fait écarté car il est possible de choisir une solution trop spécialisée sur la partition $apprentissage_{test}$ qui ne s'adaptera pas bien au jeu de test. La solution proposée dans la suite cherche à diminuer ce risque.

5.2.2.3 Meilleure f-mesure sur la partition $apprentissage_{test}$, toutes itérations confondues (MaxFM(global))

Cette méthode s'inspire des travaux de Dos Santos *et al.* [35]. Ils ont observé que dans leur algorithme génétique d'apprentissage, il était intéressant de sauvegarder l'archive de solutions obtenues avant que le phénomène de sur-apprentissage n'apparaisse. Dans notre cas, nous n'allons pas sauvegarder l'ensemble de l'archive mais uniquement la meilleure solution en terme de *f-mesure* de l'archive sur le jeu utilisé pour l'apprentissage (partition $apprentissage_{appr}$). À chaque itération, cette solution sera évaluée sur la partition d'apprentissage réservée au test (partition $apprentissage_{test}$). La solution qui donne les meilleurs résultats sur la partition $apprentissage_{test}$ toutes itérations confondues sera conservée.

5.2.3 Évaluation des stratégies

Dans cette partie nous évaluerons les performances de chacune des méthodes de sélection. Nous commencerons tout d'abord par détailler le protocole expérimental qui a été utilisé. Nous

fournirons ensuite les résultats obtenus, les interpréterons et conclurons.

5.2.3.1 Protocole

Le protocole expérimental est similaire à celui utilisé dans les expérimentations précédentes. Les sélecteurs de solution sont comparés sur les 10 jeux de données de classification asymétrique présentés en section 2.3.1.3. 5 sélecteurs sont évalués : *MaxFM*, qui est le sélecteur qui a été utilisé dans les chapitres précédents, et dans l'ordre de leur évocation dans ce chapitre : *Cf04MaxSe*, *Se04MaxCf*, *MaxFM(test)*, et *MaxFM(global)*. Ils sont exécutés sur le résultat des exécutions de MOCA-I avec la configuration qui a été identifiée dans le chapitre 3 : DMLS $(1 \cdot *)$ avec des ensembles de 10 règles maximum, une archive limitée à 500 solutions et une population initiale de 200 solutions. Le critère naturel d'arrêt de DMLS est utilisé ; MOCA-I s'arrête donc dès qu'un optimum est atteint, i.e. lorsque l'archive ne peut plus être améliorée par recherche locale. Chaque jeu de données est découpé en 5 partitions afin de réaliser une validation croisée à 5 partitions ; MOCA-I est exécuté 5 fois par partition, ce qui donne un total de 25 exécutions par jeu de données. Pour les sélecteurs qui utilisent une partition de validation (*MaxFM(test)* et *MaxFM(global)*), MOCA-I effectue l'apprentissage sur 2/3 du jeu d'apprentissage (partition $apprentissage_{appr}$), le tiers restant (partition $apprentissage_{test}$) est utilisé par le sélecteur de solution. Les solutions retenues par les sélecteurs sont ensuite évaluées en terme de *f-mesure* sur la partition d'apprentissage et la partition de test. Les résultats sont ensuite validés statistiquement à l'aide des tests de Friedman et de méthodes post-hoc (Wilcoxon + Holm) comme préconisé précédemment.

5.2.3.2 Résultats et discussion

Nous avons regroupé dans la table 5.1 les résultats obtenus par les différentes méthodes de sélection de solution, pour chacun des 10 jeux de données de classification asymétrique. Pour chacun des jeux de données, la première ligne donne la *f-mesure* moyenne et l'écart-type obtenus sur le jeu d'apprentissage (ou sur la partition $apprentissage_{appr}$ pour les 2 dernières méthodes). La seconde ligne donne les résultats obtenus sur le jeu de test, et rappelle le degré d'asymétrie du jeu de données (d_{asy}). On observe que la méthode *MaxFM* – la plus basique – donne les meilleurs résultats sur la majorité des jeux de données. Sur le jeu de données $haberman_d$ c'est la méthode *Cf04MaxSe* qui donne les meilleurs résultats ; il s'agit d'un des jeux de données sur lesquels les observations réalisées pour concevoir la méthode avaient été réalisées. Cependant, les observations qui avaient été réalisées sur ces jeux de données ne sont probablement pas généralisables à tous les autres, ce qui se traduit par de moins bons résultats. Sur le jeu de données $ecoli1_d$ c'est la méthode *MaxFM(global)* qui offre de légèrement meilleures performances. Sur le jeu de données $yeast2vs8_d$ elle offre des performances nettement supérieures à toutes les autres. Les bonnes performances de *MaxFM(global)* ne semblent pas liées au degré d'asymétrie (d_{asy}). Enfin, sur le jeu de données $abalone19_d$ c'est la méthode *MaxFM(test)* qui est la plus efficace, mais dépasse de peu les résultats des autres méthodes.
Sur les jeux de données où la méthode *MaxFM* n'est pas la plus performante sur les données

Table 5.1: F-mesure moyenne et écart-type sur jeu d'apprentissage et de test obtenus par les méthodes de sélection de solution

Jeu + d_{asy}	MaxFM	Cf04MaxSe	Se04MaxCf	MaxFM(test)	MaxFM(global)
$haberman_d$	**0.630 ± 0.029**	0.571 ± 0.008	0.543 ± 0.016	0.437 ± 0.060	0.523 ± 0.094
(27.42 %)	0.396 ± 0.117	**0.468 ± 0.054**	0.263 ± 0.105	0.367 ± 0.090	0.413 ± 0.069
$ecoli1_d$	**0.904 ± 0.013**	0.823 ± 0.024	0.787 ± 0.054	0.769 ± 0.057	0.815 ± 0.066
(22.92%)	0.758 ± 0.064	0.728 ± 0.073	0.663 ± 0.080	0.742 ± 0.069	**0.766 ± 0.046**
$ecoli2_d$	**0.930 ± 0.022**	0.857 ± 0.042	0.812 ± 0.065	0.704 ± 0.178	0.815 ± 0.187
(15.48%)	**0.828 ± 0.037**	0.736 ± 0.134	0.702 ± 0.082	0.748 ± 0.133	0.703 ± 0.190
$yeast3_d$	**0.811 ± 0.010**	0.623 ± 0.025	0.594 ± 0.026	0.749 ± 0.039	0.784 ± 0.031
(10.38%)	**0.738 ± 0.049**	0.589 ± 0.046	0.482 ± 0.081	0.715 ± 0.051	0.737 ± 0.041
$abalone9vs18_d$	**0.697 ± 0.028**	0.548 ± 0.025	0.667 ± 0.042	0.197 ± 0.075	0.547 ± 0.181
(5.65%)	**0.452 ± 0.189**	0.344 ± 0.167	0.381 ± 0.186	0.300 ± 0.170	0.284 ± 0.153
$yeast2vs8_d$	**0.884 ± 0.032**	0.731 ± 0.110	0.847 ± 0.046	0.506 ± 0.208	0.727 ± 0.124
(4.15%)	0.491 ± 0.153	0.403 ± 0.135	0.469 ± 0.158	0.459 ± 0.199	**0.599 ± 0.169**
$abalone19_d$	**0.316 ± 0.033**	0.312 ± 0.035	0.279 ± 0.035	0.068 ± 0.044	0.095 ± 0.129
(0.77%)	0.022 ± 0.046	0.022 ± 0.045	0.017 ± 0.036	**0.030 ± 0.047**	0.000 ± 0.000
$a1a$	**0.664 ± 0.009**	0.578 ± 0.006	0.550 ± 0.016	0.632 ± 0.030	0.662 ± 0.043
(24.61%)	**0.626 ± 0.021**	0.563 ± 0.013	0.479 ± 0.043	0.605 ± 0.032	0.607 ± 0.025
$lucap0$	**0.837 ± 0.006**	0.599 ± 0.023	0.580 ± 0.017	0.828 ± 0.017	0.834 ± 0.023
(27.85%)	**0.817 ± 0.013**	0.578 ± 0.020	0.545 ± 0.053	0.803 ± 0.031	0.811 ± 0.022
$w1a$	**0.693 ± 0.030**	0.543 ± 0.024	0.655 ± 0.029	0.450 ± 0.102	0.637 ± 0.116
(2.90%)	**0.484 ± 0.189**	0.415 ± 0.114	0.361 ± 0.181	0.340 ± 0.167	0.337 ± 0.148

de test ($haberman_d$, $yeast2vs8_d$, $abalone19_d$), on observe qu'elle était cependant la plus performante sur les données d'apprentissage, ce qui montre qu'elle est sujette au sur-apprentissage. Sur ces jeux, les autres méthodes ont un écart moins important avec le jeu d'apprentissage et de test et donc moins de sur-apprentissage. Cependant cet avantage ne leur permet pas d'être aussi efficaces sur les autres jeux de données.

Dans la figure 5.5 nous avons représenté les rangs moyens obtenus sur la *f-mesure* de test, ce qui permet d'avoir une vue globale du comportement des méthodes sur l'ensemble des jeux de données. Cette méthode confirme l'observation déjà réalisée précédemment : la méthode *MaxFM* semble significativement plus performante. La méthode *Se04MaxCf* est la moins efficace, et les méthodes restantes obtiennent des résultats similaires. Les différences entre les rangs moyens obtenus par les méthodes sont considérées statistiquement significatives par les tests de Friedman et Iman-Davenport ($\alpha = 0.05$). Cela nous permet donc d'approfondir les comparaisons statistiques via des tests *post-hoc*.

La table 5.2 complète les résultats précédents en effectuant la comparaison statistique post-hoc des performances du sélecteur *MaxFM* aux autres sélecteurs. Chaque colonne donne le résultat de la comparaison statistique d'un sélecteur particulier aux résultats du sélecteur *MaxFM*.

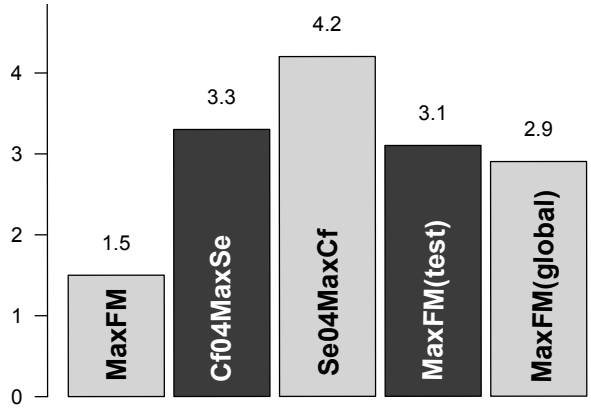


Figure 5.5: Rangs moyens sur la F-mesure de test obtenus par les méthodes de sélection d'une solution

Table 5.2: Comparaison des sélecteurs de solution - Tests post-hoc (Wilcoxon + Holm, $\alpha = 0.05$) à partir de la F-mesure sur les données de test

	Cf04MaxSe	Se04MaxCf	MaxFM(test)	MaxFM(global)
MaxFM	$\prec$ (0.044)	$\prec$ (5.37E-04)	$\prec$ 0.047	$\prec$ 0.048

Ainsi, le sélecteur *Cf04MaxSe* est statistiquement moins efficace que le sélecteur *MaxFM* avec une *p-value* de 0.044. On remarque que tous les sélecteurs proposés sont statistiquement moins performants que le sélecteur initial *MaxFM*. En revanche, les données expérimentales étudiées ne permettent pas de déterminer statistiquement le(s)quel(s) des autres sélecteurs sont les plus efficaces (en dehors de *MaxFM*). Les méthodes fondées sur la minimisation du sur-apprentissage ont probablement été victimes de l'asymétrie des données : la nécessité de découper plusieurs fois le jeu d'apprentissage laisse finalement très peu d'observations avec la classe dans chacune des partitions utilisées, ce qui rend difficile la création de classifieurs généralisables.

5.2.3.3 Conclusion

Dans cette partie, nous nous sommes attachés à concevoir des méthodes de sélection plus efficaces que la méthode basique (*MaxFM*) proposée avec MOCA-I dans les chapitres précédents. Plusieurs méthodes de sélection ont été proposées, fondées sur des observations sur le comportement de MOCA-I, tentant de maximiser la *f-mesure* ou encore de contrer le sur-apprentissage. Cependant, des expérimentations sur 10 jeux de données ont révélé que la méthode basique *MaxFM* donne statistiquement les meilleurs résultats, en dépit de l'aspect prometteur que présentaient les autres méthodes. Comme nous l'avions vu précédemment dans ce chapitre, cette méthode ne parvient toutefois pas toujours à choisir la meilleure solution sur les données de test. Nous allons maintenant étudier d'autres techniques fondées sur la combinaison de plusieurs solutions obtenues par MOCA-I.

5.3 Méthodes de fusion des solutions

Comme nous l'avions évoqué précédemment, certaines méthodes de classification – comme le boosting – sont fondées sur les méthodes d'ensembles de classifieurs. Ces méthodes définissent des stratégies pour combiner plusieurs classifieurs à l'aide de méthodes de vote pour déterminer quelle prédiction est conservée, ou d'ordonnancement pour décider de l'ordre d'application des classifieurs. Dans notre cas, nous allons essayer de combiner les classifieurs obtenus par MOCA-I. La représentation utilisée dans MOCA-I est fondée sur des ensembles de règles de classification partielle, ce qui implique que toutes les règles et ensembles de règles obtenus prédisent la même classe. Par conséquent les méthodes de vote seront inutiles dans notre cas, nous nous concentrerons donc sur les techniques d'ordonnancement des classifieurs. Comme l'ensemble des solutions obtenues par MOCA-I peuvent contenir des règles redondantes, nous étudierons également des stratégies pour sélectionner les règles à conserver. Des méthodes fondées sur la courbe ROC seront étudiées. Nous définirons tout d'abord la courbes ROC et son utilisation, avant de détailler les différentes méthodes et de les évaluer. Une partie de ce travail a fait l'objet d'une publication dans la conférence LION 7 [60].

5.3.1 Généralités sur la courbe ROC

Dans cette partie, nous verrons tout d'abord ce qu'est la courbe ROC et pour quoi elle peut être utilisée. Nous montrerons ensuite la méthode utilisée pour sa construction et l'interprétation de la représentation obtenue.

5.3.1.1 Définition et utilisation

La courbe ROC (Receiver Operating Characteristic) est utilisée pour évaluer les performances d'algorithmes de classification binaire. Elle peut également être utilisée pour déterminer un seuil dans les algorithmes qui proposent un score, ou définir efficacement un paramètre comme dans les travaux de Chawla *et al.* [26]. Enfin elle est également utilisée en médecine pour étalonner les tests diagnostics, comme l'illustrent par exemple Perneger *et al.* pour le diagnostic de l'embolie pulmonaire à l'aide d'un dosage sanguin [88] : « à partir de quel dosage peut-on exclure que le patient ait une embolie ? ».

5.3.1.2 Représentation

Nous allons maintenant représenter la courbe ROC d'un algorithme de classification qui donne un score compris entre 0 si l'observation n'a pas la classe, et 1 si l'observation a la classe. La courbe ROC correspond en fait à la représentation de la *sensibilité* et de l'*anti-spécificité* (1 - *spécificité*) pour plusieurs valeurs de score. Pour représenter la courbe ROC, nous allons choisir plusieurs valeurs de seuil, pour lesquelles nous allons calculer le nombre d'observations correctement / incorrectement classées. Ce qui va nous permettre d'obtenir pour chaque valeur de seuil une valeur de *sensibilité* et d'*anti-spécificité* qui pourront être reportées sur la courbe

Table 5.3: Illustration de la première étape de traçage d'une courbe ROC

Seuil	Anti-spécificité	Sensibilité	Point sur ROC
1	0.0	0.2	a
0.9	0.0	0.3	b
0.8	0.0	0.5	c
0.7	0.1	0.7	d
0.6	0.2	0.83	e
0.5	0.6	1	f

ROC. Nous obtenons ainsi la table 5.3. Dans cette table, on observe par exemple que lorsque l'on prend en compte uniquement les observations pour lesquelles l'algorithme a donné un score de 1 (première ligne), il n'y a pas de faux positifs (*anti-spécificité* = 0) et 20% des observations avec la classe sont détectées (*sensibilité* = 0.2). Lorsque l'on fixe le seuil à 0.7 – ce qui revient à considérer que toutes les observations dont le score est supérieur à 0.7 ont la classe – on retrouve 10% de faux positifs (*anti-spécificité = 0.1*) et on identifie 70% des observations (*sensibilité = 0.7*).
Nous avons projeté les différentes valeurs calculées dans la figure 5.6, qui représente la courbe ROC obtenue. On retrouve en abscisse l'*anti-spécificité* qui est aussi parfois appelée *taux de faux positifs* et en ordonnée la *sensibilité* qui est aussi appelée *taux de vrais positifs*. Chacun des points de la courbe est identifié par une lettre et est également présent dans la table 5.3 utilisée pour le traçage. Nous allons maintenant voir comment cette courbe s'interprète.

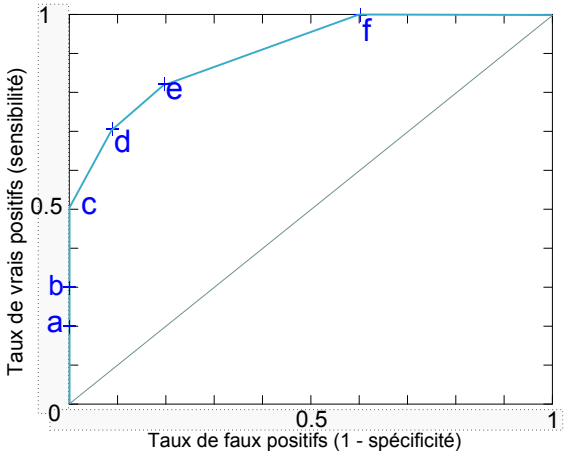


Figure 5.6: Illustration d'une courbe ROC

5.3.1.3 Interprétation

On évalue qu'un classifieur est efficace si sa courbe ROC se rapproche du point (0,1) qui correspond à un classifieur idéal qui identifie toutes les observations avec la classe (*sensibilité*

= 1) sans faire la moindre erreur (*anti-spécificité* = 0). Plus la courbe ROC d'un classifieur est proche de ce point, plus il est performant. En revanche, une courbe ROC qui se rapproche de la droite ((0,0) ;(1,1)) n'est pas du tout efficace. La courbe ROC est utilisée pour visualiser le compromis entre le *taux de vrais positifs* et le *taux de faux positifs*, ce qui permet de déterminer le seuil à utiliser selon les besoins. Dans l'illustration, si l'on souhaite que le classifieur ne fasse aucune erreur, il faudra choisir le seuil du point c car il s'agit du dernier point avant lequel le *taux de faux positifs* augmente. En revanche si l'on souhaite un classifieur qui identifie tous les cas positifs, il faudra plutôt choisir le seuil du point f.

Elle peut également être utile pour la comparaison d'algorithmes de classification, où deux manières de procéder sont possibles. Une première utilisation est sous la forme d'un indicateur – l'AUC (area under ROC Curve) – qui correspond à l'aire sous la courbe ROC et qui est parfois utilisée dans la littérature pour la comparaison d'algorithme, à la place de l'*exactitude* ou de la *f-mesure* [50]. L'inconvénient de cette méthode est qu'elle ne permet pas de comparer finement des classifieurs en regardant par exemple leurs performances dans une région particulière de l'espace ROC, comme par exemple la région de faible *taux de faux positifs*. La deuxième méthode est donc de comparer visuellement les courbes ROC selon leurs performances dans les différentes régions.

5.3.2 Courbe ROC d'un ensemble de règles (ROCCfMax)

Nous avons vu comment générer la courbe ROC d'un classifieur à scores, nous allons maintenant proposer une méthode pour construire la courbe ROC d'un ensemble de règles, ainsi que l'utilisation qui peut en être faite.

5.3.2.1 Génération de la courbe ROC

Algorithme 4 Génération de la courbe ROC d'un ensemble de règles ER

```
classer les règles ∈ ER par confidence DESC, sensibilité DESC
créer l'ensemble de règles vide ER_roc
for règle R_i ∈ ER {R_1, R_2, ..., R_n} do
  /* calculer la sensibilité et spécificité du sous-ensemble R_1, R_2, ..., R_i */
  ER_roc.ajouter(R_i)
  se ← ER_roc.calculerSensibilite()
  asp ← 1 - ER_roc.calculerSpecificite()
  tracer(se,asp)
end for
```

L'algorithme 4 détaille notre méthode de génération de la courbe ROC pour un ensemble de règles ER. Les règles sont tout d'abord classées par *confiance* décroissante ; lors d'un ex-æquo les règles avec la plus forte *sensibilité* sont privilégiées. Au lieu de faire varier un seuil comme précédemment, nous allons faire varier le nombre de règles de l'ensemble de règles ER.

Nous calculerons ainsi la *sensibilité* et l'*anti-spécificité* de l'ensemble des règles $\{R_1\}$, puis de l'ensemble de règles $\{R_1, R_2\}$, de l'ensemble de règles $\{R_1, \ldots, R_i\}$ jusqu'à l'ensemble de règles $\{R_1, R_2, \ldots, R_n\}$.

5.3.2.2 Aide à la décision et choix de la solution finale

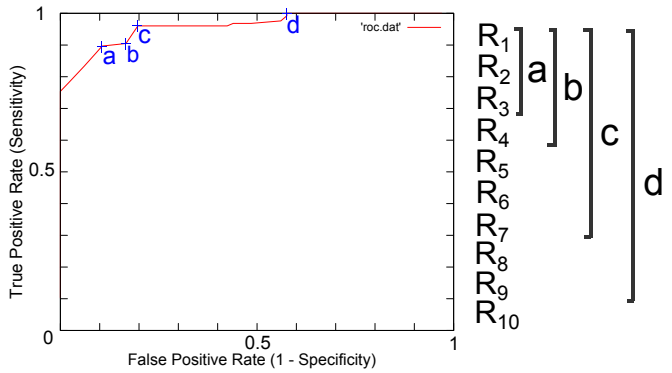


Figure 5.7: Exemple de la courbe ROC d'un ensemble de 10 règles issu de MOCA-I

Dans la figure 5.7 nous présentons la courbe ROC obtenue pour un ensemble de règles $\{R_1 \ldots R_{10}\}$ généré par MOCA-I. Sur la droite de la figure, nous avons représenté les différents sous-ensembles de règles utilisés pour la construction de la courbe ROC, par exemple l'ensemble de règles $\{R_1, R_2, R_3\}$ correspond au point a sur la courbe ROC. Ces différents points permettent de choisir l'ensemble de règles final à diffuser à l'utilisateur selon ses besoins. Si l'on reprend le cas du pré-screening pour les essais cliniques, pour chaque critère d'inclusion une courbe ROC va être disponible, qui correspondra aux performances de l'ensemble de règles obtenu. Si l'utilisateur dispose de très peu de temps pour examiner les dossiers, sur l'illustration, des points de coupe comme le point a seront à privilégier car ils apportent le moins de faux positifs. Les règles $\{R_1, R_2, R_3\}$ seront alors utilisées, avec la possibilité d'ajouter des règles supplémentaires (R_4, R_5,etc.) si l'on souhaite obtenir plus de patients. Si au contraire l'utilisateur dispose d'un peu plus de temps pour traiter les dossiers et que l'essai clinique manque de recrutements il est possible d'élargir le nombre de patients détectés. Cela peut s'effectuer en utilisant un point de coupe comme le point b ou le point c, qui correspondent aux règles $\{R_1, R_2, R_3, \ldots, R_7\}$. Étant donné que le nombre de critères d'inclusion peut être élevé (jusqu'à 30 par étude), il est préférable de proposer à l'utilisateur un point de coupe par défaut pour chacun des critères d'inclusion, qu'il pourra ajuster ensuite au besoin. Des méthodes de détermination automatique du point de coupe vont être proposées et étudiées dans la suite. Par ailleurs, si l'on souhaite fusionner toutes les règles obtenues par MOCA-I au sein d'un même ensemble de règles, des doublons et chevauchement entre les règles risquent d'apparaître. Dans la suite, nous allons donc proposer des méthodes de construction de la courbe ROC qui visent à limiter ce phénomène en filtrant les règles obtenues.

5.3.3 Méthodes de fusion

Dans cette section nous proposons deux méthodes visant à sélectionner et ordonner les règles d'un ensemble de règles.

5.3.3.1 Règles strictement améliorantes (ROCPlateauMin)

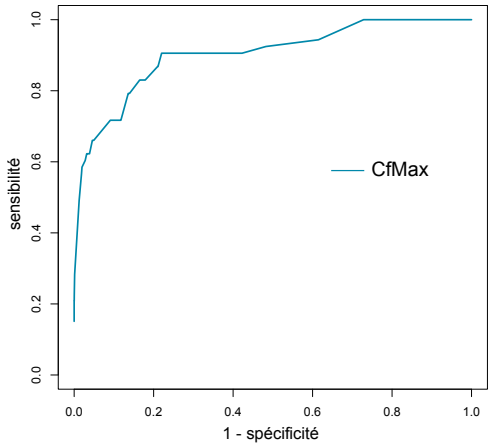


Figure 5.8: Exemple de la courbe ROC d'un ensemble de règles issu de MOCA-I (jeu *tia-f*), construit avec la méthode ROCCfMax

Observons une courbe roc générée par la méthode proposée précédemment (*ROCCfMax*), comme celle que nous proposons dans la figure 5.8. Cette courbe ROC a été réalisée en fusionnant toutes les solutions d'une exécution de MOCA-I dans un même ensemble de règles. On peut remarquer que l'on peut retrouver des plateaux à certains endroits de la courbe ROC, comme vers le point (0.15 ;0.7) ou encore à partir du point (0.23 ;0.9). Ces plateaux semblent correspondre à des règles qui ne permettent pas de classifier de nouvelles observations, peut-être à cause d'une redondance avec des règles précédentes. En plus de ne pas apporter de nouveaux cas à classifier, les règles présentes sur les plateaux dégradent l'*anti-spécificité*. La méthode que nous proposons (*ROCPlateauMin*) vise à minimiser ces plateaux, en supprimant les règles qui n'améliorent pas la *sensibilité*. Elle est similaire à la méthode présentée précédemment pour la construction de la courbe ROC : les règles sont classées par *confiance* décroissante et ajoutées progressivement à l'ensemble de règles résultat si et seulement si elles améliorent la *sensibilité*.

Dans la figure 5.9 nous montrons la courbe ROC obtenue à partir des mêmes règles, mais en appliquant cette fois la procédure proposée (*ROCPlateauMin*). Elle est accompagnée de la courbe ROC obtenue avec la méthode précédente, indiquée en pointillés, à des fins de comparaison. La courbe ROC obtenue avec la procédure *ROCPlateauMin* est supérieure à celle obtenue par la procédure proposée initialement (*ROCCfMax*). On observe cependant peu d'amélioration dans la région de faible *anti-spécificité*.

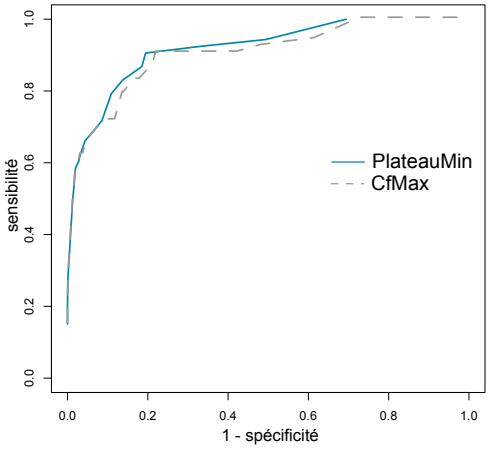


FIGURE 5.9: Exemple de la courbe ROC d'un ensemble de règles issu de MOCA-I (jeu *tia-f*), construit avec la méthode ROCPlateauMin

5.3.3.2 Meilleur ratio sensibilité / anti-spécificité (ROCRatioSeAsp)

Si l'on observe la courbe générée par la méthode précédente (*ROCPlateauMin*), on observe que la croissance de la courbe n'est pas constante. On peut ainsi retrouver des zones comme à partir du point (0.2,0.9) où la courbe croît fortement, ce qui semble indiquer l'ajout d'une règle plus efficace que les règles précédentes, qui contribuaient moins à la croissance de la courbe ROC. Les règles les plus intéressantes sont celles qui apportent une forte *sensibilité* sans amener (trop) de faux positifs, autrement dit des zones où la courbe ROC croît le plus fortement. La méthode que nous proposons cherche à placer les règles qui apportent le meilleur ratio $\frac{sensibilite}{anti-specificite}$ au début de l'ensemble de règle. L'ensemble de règles final ER_{roc} est construit de la manière suivante : il est tout d'abord initialisé avec la règle de plus forte *confiance* puis *sensibilité*. À chaque itération, on parcourt l'ensemble des règles R_i qui n'ont pas encore été ajoutées à ER_{roc}, et on calcule le ratio $\frac{sensibilite}{anti-specificite}$ qu'elles proposeraient si elles étaient ajoutées à ER_{roc}. Ce qui correspond pour chaque règle au ratio de l'ensemble de règles $ER_{roc} \cup R_i$. La règle R_i qui obtient le meilleur ratio est ajoutée à ER_{roc} et on recommence les itérations jusqu'à ce qu'il ne reste plus de règles, ou que la courbe ROC atteigne une $sensibilité = 1$. La courbe ROC obtenue est représentée dans la figure 5.10. Les courbes ROC obtenues par les méthodes précédentes y sont aussi représentées (*ROCCfMax* et *ROCRatioSeAsp*). Elle améliore la courbe ROC sur la majorité de la courbe, excepté sur la région d'*anti-spécificité* comprise entre 0.18 et 0.35.

5.3.4 Méthodes de coupe

Nous avons vu comment générer efficacement la courbe ROC d'un ensemble de règles, en filtrant certaines règles moins efficaces. Nous allons maintenant examiner plusieurs méthodes de choix du point de coupe, afin de simplifier le travail de l'utilisateur final. Avant cela, nous allons tout d'abord étudier le comportement de la *f-mesure* en fonction de la courbe ROC afin de guider la conception des différentes méthodes.

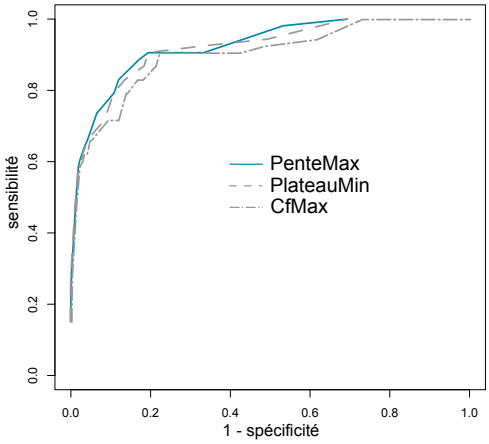


Figure 5.10: Exemple de la courbe ROC d'un ensemble de règles issu de MOCA-I (jeu *tia-f*), construit avec la méthode ROCRatioSeAsp

5.3.4.1 F-mesure en fonction de la courbe ROC

La figure 5.11 permet de visualiser l'impact de la courbe ROC sur la valeur de la *f-mesure*. Nous avons représenté dans les graphiques du dessus une courbe ROC sur le jeu d'apprentissage (à gauche) et sur le jeu de test (à droite). Les graphiques du bas quant à eux représentent la *f-mesure* sur le jeu d'apprentissage (à gauche) et sur le jeu de test (à droite). On observe toujours le phénomène de sur-apprentissage, qui diminue la courbe ROC sur le jeu de test, par rapport à celle qui avait été obtenue sur le jeu d'apprentissage. Sur le jeu d'apprentissage, la *f-mesure* se dégrade rapidement après avoir connu un pic, ce qui est moins le cas sur le jeu de test où plusieurs "pics" de *f-mesure* sont observés. Ces "pics" paraissent correspondre à ceux retrouvés sur la courbe ROC. Il semble opportun de tester des méthodes de choix du point de coupe qui se focalisent sur ces pics, nous en proposons 3.

coupe à la meilleure F-mesure (MaxFM) Cette méthode de choix du point de coupe consiste à choisir le sous-ensemble de règles $\{R_1, R_2, \ldots, R_i\}$ qui offre la meilleure *f-mesure* sur le jeu d'apprentissage. Sur l'exemple précédent, cela permet de choisir une solution qui propose une bonne *f-mesure* avant que la *f-mesure* commence à se dégrader.

coupe au point idéal (Ideal) Cette méthode est fondée sur le concept de classifieur idéal. Comme nous l'avions vu précédemment, celui-ci se situe au point (0,1). L'objectif est de choisir le sous-ensemble de règles $\{R_1, R_2, \ldots, R_i\}$ dont la projection sur la courbe ROC est située au plus proche du point idéal (0,1).

coupe au ratio sensibilité / anti-spécificité (RatioSeAsp) Cette méthode est uniquement disponible lorsque l'ensemble de règles dont est issue la courbe ROC est construit en choisissant les règles qui maximisent le ratio $\frac{sensibilite}{anti-specificite}$ (méthode ROCRatioSeAsp). Dans

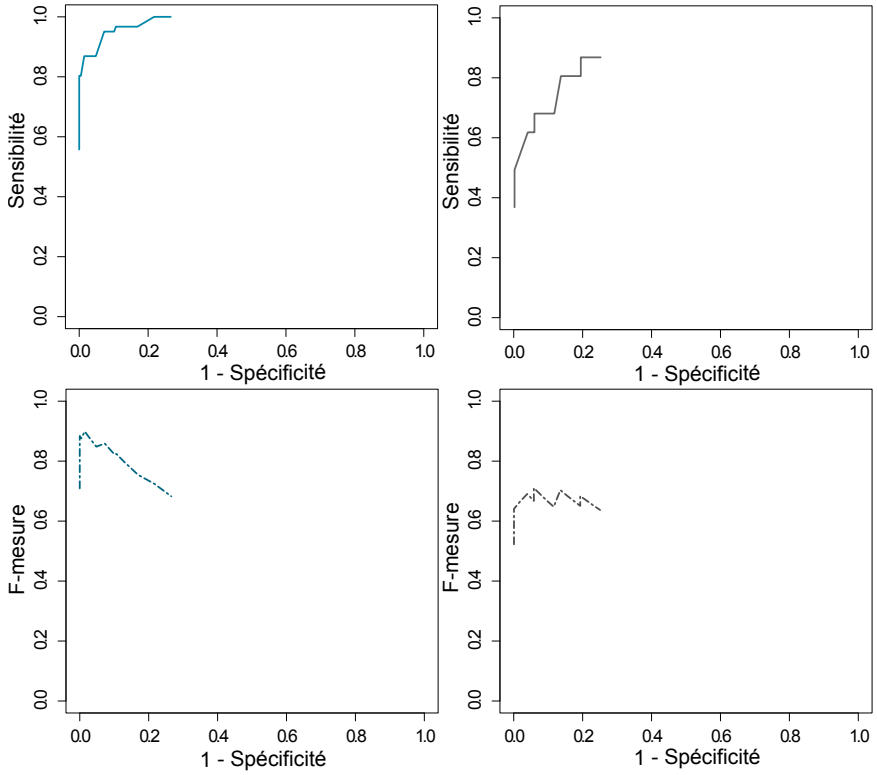


Figure 5.11: Courbe ROC sur jeu d'apprentissage et de test (figures du haut) et *f-mesure* associée (figures du bas)

ce cas, lorsque le nombre de règles augmente, le ratio $\frac{sensibilite}{anti-specificite}$ diminue. Un critère de coupe serait de ne plus sélectionner les règles lors que le ratio $\frac{sensibilite}{anti-specificite}$ est inférieur à 1, ce qui signifie que les nouvelles règles apportent plus de *faux positifs* que de *vrais positifs*.

5.3.5 Évaluation des stratégies fondées sur la courbe ROC

Dans cette partie, nous allons évaluer les différentes stratégies proposées précédemment (génération d'un ensemble de règles et méthodes de coupe), ainsi que les comparer aux résultats obtenus précédemment par la méthode de sélection *MaxFM* utilisée dans la première version de MOCA-I. Nous détaillerons tout d'abord le protocole utilisé pour les expérimentations. Ensuite, nous présenterons les résultats et l'interprétation associée, ainsi que les tests statistiques appropriés et la conclusion.

5.3.5.1 Protocole

Toutes les associations possibles entre une méthode de construction d'un ensemble de règles et une méthode de coupe vont être comparées. Les trois méthodes de construction *ROCCfMax*, *ROCPlateauMin* et *ROCRatioSeAsp* vont être combinées aux 2 méthodes de coupe *MaxFM* et

Ideal. La méthode de construction *ROCRatioSeAsp* sera également combinée à la méthode de coupe *RatioSeAsp*, ce qui donne au total 7 combinaisons étudiées. Nous donnerons également à des fins de comparaison les résultats obtenus par la méthode de sélection de solution présente initialement dans MOCA-I (*MaxFM*).
Le protocole est similaire à celui réalisé pour la comparaison des méthodes de sélection de solution. L'étude est réalisée sur les 10 jeux de données asymétriques de la littérature présentés précédemment. MOCA-I est exécuté 25 fois par jeu de données : une validation croisée à 5 partitions est utilisée, avec 5 exécutions par partition. Les 7 associations de méthodes de fusion/coupe sont ensuite exécutées sur les solutions générées par MOCA-I. La *f-mesure* obtenue sur les jeux d'apprentissage et de test est ensuite utilisée pour l'évaluation, qui est complétée par des tests statistiques comme préconisé dans un précédent chapitre.

5.3.5.2 Résultats et Discussion

Nous avons regroupé dans la table 5.4 les résultats (moyenne et écart-type de la *f-mesure*) obtenus par chacune des combinaisons d'une méthode de construction de la courbe ROC et de méthode de coupe. Les lignes impaires contiennent le nom de chaque jeu de données et les résultats obtenus sur le jeu d'apprentissage. Les lignes paires contiennent quant à elles un rappel du degré d'asymétrie du jeu de données (d_{asy}) et les résultats obtenus sur le jeu de test.
La combinaison *ROCPenteMax+MaxFM* offre les meilleurs résultats sur les données d'apprentissage sur la majorité des jeux de données. Cependant ce n'est pas le cas sur les données de test, pour lesquelles elle reste uniquement efficace sur les jeux de données ayant un $2\% < d_{asy} < 10\%$, soient les jeux de données fortement asymétriques. Sur les autres jeux de données, ce sont les combinaisons *ROCCfMax+Ideal* et *ROCPlateauMin+Ideal* qui offrent les meilleurs résultats. Sur les données de test, il n'y a donc pas de combinaison qui semble efficace sur l'ensemble des jeux de données étudiés. On observe également une forte différence entre les résultats obtenus sur les jeux d'apprentissage et de test, ce qui dénote une manifestation du *sur-apprentissage*.

En plus de la *f-mesure*, nous avons également collecté le nombre de termes moyens contenus dans les solutions générées par chaque méthode dans la table 5.5. La dernière colonne correspond aux valeurs obtenues par la méthode de sélection *MaxFM* présentée dans la section précédente, et utilisée par défaut dans MOCA-I. On observe que la combinaison des méthodes *ROCPlateauMin+MaxFM* donne les solutions les plus simples sur la majorité des jeux de données. Sur les jeux de données de volumétrie raisonnable, les solutions générées peuvent être jusqu'à 2 à 3 fois plus complexes que les solutions obtenues par la méthode de sélection d'une solution *MaxFM*. Sur les jeux de données plus complexes (*a1a* et *lucap*0) ils sont jusqu'à 7 fois plus complexes. Ces solutions trop complexes peuvent avoir des difficultés à généraliser et sont donc sujettes au sur-apprentissage.

Observons maintenant les résultats obtenus sur l'ensemble des jeux de données, à l'aide des rangs moyens obtenus sur les données de test, que nous avons représentés dans la figure 5.12. À titre informatif, lors du calcul des rangs nous avons intégré également les résultats de la méthode de sélection de solution *MaxFM* présentée précédemment. On observe que les rangs

TABLE 5.4: F-mesure moyenne et écart-type sur jeu d'apprentissage et de test obtenus par les méthodes de fusion de solutions fondées sur la courbe ROC

ROC	ROCCfMax		ROCPlateauMin		ROCRatioSeAsp		
Coupe	MaxFM	Ideal	MaxFM	Ideal	MaxFM	Ideal	RatioSeAsp
Jeu+d_{asy}							
$haberman_d$	0.629 ± 0.030	0.615 ± 0.027	0.644 ± 0.029	0.636 ± 0.028	**0.656 ± 0.027**	0.646 ± 0.025	0.646 ± 0.026
(27.42%)	0.388 ± 0.113	**0.441 ± 0.066**	0.406 ± 0.111	0.421 ± 0.091	0.370 ± 0.115	0.399 ± 0.083	0.398 ± 0.089
$ecoli1_d$	0.911 ± 0.017	0.895 ± 0.020	0.913 ± 0.017	0.898 ± 0.016	**0.916 ± 0.016**	0.901 ± 0.022	0.894 ± 0.026
(22.92%)	0.744 ± 0.054	**0.749 ± 0.064**	0.734 ± 0.058	0.739 ± 0.071	0.745 ± 0.058	0.743 ± 0.057	0.743 ± 0.054
$ecoli2_d$	0.942 ± 0.015	0.928 ± 0.025	**0.943 ± 0.015**	0.927 ± 0.027	0.943 ± 0.015	0.927 ± 0.026	0.932 ± 0.025
(15.48%)	0.801 ± 0.072	**0.805 ± 0.071**	0.803 ± 0.080	0.798 ± 0.077	0.795 ± 0.078	0.794 ± 0.073	0.805 ± 0.070
$yeast3_d$	0.816 ± 0.011	0.758 ± 0.023	0.819 ± 0.010	0.762 ± 0.025	**0.825 ± 0.009**	0.777 ± 0.021	0.765 ± 0.027
(10.38%)	0.708 ± 0.053	0.694 ± 0.046	0.709 ± 0.056	0.697 ± 0.048	**0.716 ± 0.044**	0.690 ± 0.053	0.681 ± 0.053
$abalone9vs18_d$	0.719 ± 0.041	0.514 ± 0.071	0.726 ± 0.042	0.534 ± 0.082	**0.727 ± 0.040**	0.532 ± 0.087	0.530 ± 0.088
(5.65%)	0.443 ± 0.227	0.377 ± 0.181	0.444 ± 0.230	0.383 ± 0.194	**0.446 ± 0.235**	0.388 ± 0.189	0.386 ± 0.188
$yeast2vs8_d$	0.896 ± 0.038	0.769 ± 0.115	0.896 ± 0.038	0.770 ± 0.115	**0.897 ± 0.036**	0.770 ± 0.115	0.858 ± 0.060
(4.15%)	0.438 ± 0.162	0.353 ± 0.139	0.438 ± 0.162	0.353 ± 0.139	**0.440 ± 0.162**	0.353 ± 0.139	0.428 ± 0.154
$abalone19_d$	0.377 ± 0.034	0.112 ± 0.024	0.380 ± 0.036	0.116 ± 0.022	**0.381 ± 0.036**	0.116 ± 0.025	0.118 ± 0.025
(0.77%)	0.022 ± 0.053	0.037 ± 0.021	0.022 ± 0.053	0.037 ± 0.023	0.022 ± 0.053	0.037 ± 0.022	**0.038 ± 0.021**
$a1a$	0.712 ± 0.014	0.702 ± 0.015	0.721 ± 0.013	0.709 ± 0.016	**0.738 ± 0.012**	0.724 ± 0.018	0.722 ± 0.019
(24.61%)	0.600 ± 0.030	0.616 ± 0.023	0.599 ± 0.029	**0.616 ± 0.025**	0.582 ± 0.031	0.611 ± 0.022	0.613 ± 0.020
$lucap0$	0.879 ± 0.012	0.875 ± 0.012	0.885 ± 0.012	0.882 ± 0.012	**0.891 ± 0.010**	0.887 ± 0.010	0.884 ± 0.011
(27.85%)	0.788 ± 0.026	0.791 ± 0.022	0.794 ± 0.026	**0.795 ± 0.024**	0.791 ± 0.031	0.792 ± 0.022	0.787 ± 0.026
$w1a$	0.714 ± 0.036	0.366 ± 0.074	0.718 ± 0.036	0.373 ± 0.074	**0.722 ± 0.036**	0.375 ± 0.071	0.456 ± 0.086
(2.90%)	0.468 ± 0.194	0.312 ± 0.108	0.468 ± 0.185	0.309 ± 0.109	**0.469 ± 0.197**	0.311 ± 0.111	0.363 ± 0.100

Table 5.5: Nombre moyens de termes (tests sur attributs) compris dans les solutions générées par les méthodes de fusion de solutions fondées sur la courbe ROC (et par la méthode de sélection de solution *MaxFM*

ROC	ROCCfMax		ROCPlateauMin		ROCRatioSeAsp			MaxFM
Coupe	MaxFM	Ideal	MaxFM	Ideal	MaxFM	Ideal	RatioSeAsp	
haberman	74.6	90.4	**52.4**	60.4	59.8	65.2	65.2	22.2
ecoli1	45.2	57.8	**42.6**	45.2	45	54.6	58.8	23.6
ecoli2	36.8	46.2	**34.2**	41.8	38.2	47.8	44.4	18.2
yeast3	112	230	**80.6**	131.4	103.6	148.8	161.2	31.4
abalone9vs18	63	104.4	**51.6**	81.4	65.6	94.2	94.2	33.6
yeast2vs8	21	36.4	**19.4**	34.2	24.2	35.4	29	19
abalone19	59.8	180.4	**47.8**	127.6	60.6	132.4	133.2	39
a1a	338.6	405.6	**217.8**	240.4	223.4	249.2	250.4	31.8
lucap0	498.6	570.4	313	334.6	**279.4**	309.8	313	48.2
w1a	33.6	45.8	**25.2**	33.6	34	42.6	42	15.8

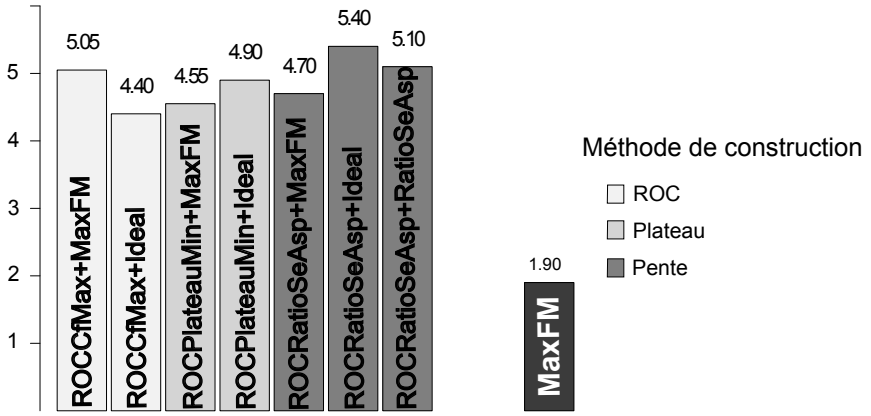


Figure 5.12: Rangs moyens sur la F-mesure de test obtenus par les méthodes de fusion de solution + rangs moyens de la méthode de sélection basique "MaxFM"

moyens obtenus par les méthodes de fusion sont clairement au dessus du rang moyen obtenu par la méthode de sélection *MaxFM* ; ce qui signifie que leurs performances en classification sont moins intéressantes que celles de *MaxFM*. Cela peut probablement s'expliquer par le sur-apprentissage observé sur ces méthodes. D'autre part, aucune de ces méthodes ne semble se démarquer des autres car elles obtiennent toutes des rangs moyens similaires.

Table 5.6: Comparaison des méthodes de génération d'ensemble de règles fondées sur la courbe ROC à la méthode de sélection de solution MaxFM - Tests post-hoc (Wilcoxon + Holm, $\alpha = 0.05$) à partir de la F-mesure sur les données de test

ROC	ROCCfMax		ROCPlateauMin		ROCRatioSeAsp		
Coupe	MaxFM	Ideal	MaxFM	Ideal	MaxFM	Ideal	RatioSeAsp
MaxFM	$\prec$ (0.02)	$\prec$ (0.03)	$\prec$ (0.03)	$\prec$ (0.02)	$\prec$ (0.03)	$\prec$ (0.01)	$\prec$ (0.02)

Le test de Friedman confirme que les rangs moyens obtenus sont statistiquement différents avec $\alpha = 0.05$, ce qui nous permet de réaliser des tests post-hoc pour effectuer les comparaisons des méthodes 2 à 2. La table 5.6 donne les résultats des comparaisons de chacune des combinaisons de méthodes de construction de la courbe ROC avec la méthode de sélection *MaxFM* définie précédemment. Chaque colonne précise pour une méthode si elle est plus efficace ou moins efficace que la méthode *MaxFM*, avec la p-value (ajustée avec correction) obtenue par le test. On observe que toutes les méthodes de génération de solution fondées sur la courbe ROC qui ont été proposées sont statistiquement moins performantes que la méthode *MaxFM*. Les données expérimentales ne sont pas suffisantes pour déterminer si parmi les méthodes restantes, une méthode est statistiquement plus efficace qu'une autre.

5.3.5.3 Conclusion

Nous avons présenté différentes méthodes de génération d'une solution à partir d'un ensemble de solutions fondées sur la courbe ROC. Nous avons tout d'abord présenté la courbe ROC et la manière de la générer et de l'utiliser. Nous avons ensuite proposé une approche pour générer la courbe ROC d'un ensemble de règles, ainsi que différentes méthodes pour en améliorer la construction. Nous avons également présenté plusieurs méthodes pour choisir le point de coupe le plus intéressant sur la courbe ROC à la place de l'utilisateur. Enfin, ces méthodes ont été évaluées et comparées à la méthode de sélection de solution *MaxFM* proposée précédemment avec MOCA-I. En dépit du fait qu'elles utilisent l'ensemble des solutions générées par MOCA-I pour bâtir une solution et non une seule comme dans la méthode *MaxFM*, elles se sont avérées moins performantes que la méthode *MaxFM*. Elles génèrent des solutions 2 à 3 fois plus complexes que la méthode *MaxFM*, voir jusqu'à 7 fois plus complexes sur les jeux de données avec un large nombre d'attributs. Elles laissent toutefois la possibilité à l'utilisateur de pouvoir adapter les performances du classifieur, en autorisant peu de faux positifs s'il dispose de peu de temps, ou au contraire en détectant plus de patients (au risque de présenter plus de faux positifs) s'il dispose de plus de temps ou si les besoins en recrutements sont plus importants sur un essai clinique.

5.4 Conclusion du chapitre

Dans les chapitres précédents, nous avons vu que MOCA-I générait un ensemble de solutions de compromis. Dans ce chapitre, nous avons proposé et évalué différentes méthodes pour choisir ou générer efficacement une solution à partir des solutions obtenues. Nous avons tout d'abord analysé les performances de la méthode de sélection mise en place par défaut dans MOCA-I (méthode *MaxFM*), qui ne choisit pas toujours la solution qui donne les meilleurs résultats sur les données de test. Nous avons ensuite proposé plusieurs méthodes d'amélioration : des méthodes fondées sur la sélection d'une solution parmi les solutions obtenues, et des méthodes axées sur la génération d'une solution, fondées sur l'utilisation de la courbe ROC. Pour chacun des types de méthodes, nous avons proposé différentes méthodes qui ont ensuite été évaluées et comparées. La méthode de sélection proposée initialement dans MOCA-I, qui consiste à choisir la solution qui obtient la meilleure *f-mesure* sur les données d'apprentissage, a été déterminée comme statistiquement plus efficace que les autres méthodes proposées. On note tout de même que les méthodes fondées sur la courbe ROC permettent plus de souplesse du coté de l'utilisateur, qui a la possibilité d'ajouter ou de supprimer des règles à l'ensemble de règles final pour obtenir plus ou moins d'observations vérifiant la classe (ce qui amène plus ou moins de faux positifs).

Une première perspective de ce travail pourrait être d'étudier plus précisément la courbe Précision-Rappel, qui est une courbe un peu similaire à la courbe ROC. Elle est différente au niveau des critères d'évaluation utilisés : elle utilise la *confiance* et la *sensibilité*, ce qui la rend plus proche de la tâche de classification partielle. Les premières expérimentations n'ont pas permis d'obtenir de meilleures performances que celles de la méthode de sélection par défaut de MOCA-I ; cependant il pourrait être intéressant de concevoir et évaluer de nouvelles méthodes de fusion à la manière de ce qui a été réalisé pour la courbe ROC. D'autre part, il pourrait être intéressant d'appliquer le principe MDL aux méthodes de fusion pour obtenir des solutions plus simples et donc plus généralisables aux données de test.

Une autre perspective pourrait être l'utilisation d'indicateurs sur la qualité d'un ensemble de règles pour guider le choix de la solution. Par exemple, Kuncheva *et al.* [71] ont proposé des indicateurs sur la diversité d'un ensemble de classifieurs. Ceux-ci pourraient être utilisés pour identifier si des ensembles de règles sont plus généralisables que d'autres, et ainsi obtenir de meilleurs résultats sur les données de test. Enfin, des méthodes fondées sur le bootstrapping semblent également être une piste intéressante pour améliorer les résultats. Le bootstrapping consiste à effectuer itérativement des ré-échantillonnages du jeu d'apprentissage. Dans les travaux de Fitzgerald *et al.* cette technique a permis d'améliorer les résultats obtenus par un algorithme de programmation génétique, en produisant des règles plus générales et moins sensibles au phénomène de bloat [39]. Il reste cependant à voir si ces travaux sont également applicables aux jeux de données asymétriques.

Chapitre 6

Le logiciel Opcyclin

Dans les chapitres précédents, nous avons vu comment réaliser efficacement de la classification sur les données médicales. Nous allons maintenant voir comment intégrer les différents éléments présentés précédemment au sein d'une suite logicielle : le logiciel Opcyclin. Nous détaillerons tout d'abord les différents modules utilisés dans Opcyclin : interface utilisateur, moteur d'apprentissage et système expert. Nous présenterons ensuite leurs rôles et leurs interactions, ainsi que les concepts généraux du logiciel Opcyclin : cas d'utilisation et base de données. Chaque module sera ensuite étudié en détail. Enfin, nous terminerons par des perspectives pour le logiciel Opcyclin.

6.1 Présentation générale

Nous commencerons tout d'abord par rappeler les différents *scenarii* d'utilisation présents dans le logiciel Opcyclin. Ensuite, nous présenterons les différents modules qui composent le logiciel Opcyclin, ainsi que leurs interactions.

6.1.1 Scenarii d'utilisation

Nous résumons ici les *scenarii* qui ont été détaillés dans le chapitre 1. Le logiciel Opcyclin s'articule autour de 2 fonctionnalités fondamentales :

- Quels patients pour cet essai ?
- Quels essais pour ce patient ?

La première consiste à déterminer pour un essai clinique donné une liste de patients dont le profil correspond à celui décrit dans les critères d'inclusion et de non inclusion de l'essai. L'utilisateur peut l'utiliser dans des cas d'utilisation différents :

Étude de faisabilité Il s'agit de « tester » les capacités de recrutement sur un essai clinique donné. À partir des dossiers de patients passés dans le service les années précédentes,

on estime le volume (théorique) d'inclusions que l'on peut espérer pour cette étude. Opcyclin retourne une liste de patients candidats à l'essai clinique, associés à un score de correspondance. Public visé : ARC ou promoteur.

Suivi au fil de l'eau L'objectif est de permettre à l'utilisateur d'identifier des patients potentiels, à partir des dossiers des patients passés récemment dans le service ou à l'hôpital, et d'une liste de critères d'inclusion. Opcyclin retourne une liste de patients candidats à l'essai clinique, associés à un score de correspondance. Public visé : investigateurs chargés du recrutement (TEC, médecins, etc.)

Évaluer la qualité du recrutement Une fois un essai clinique terminé, l'objectif est de comparer les patients potentiels identifiés par Opcyclin à ceux réellement recrutés, à partir des dossiers des patients passés dans le service sur la durée de l'essai clinique. Cela permet d'évaluer la qualité du recrutement, et d'identifier des problèmes de moyens ou des dysfonctionnements. Opcyclin retourne une liste de patients candidats à l'essai clinique, associés à un score de correspondance. Si on lui fournit la liste des patients inclus, Opcyclin peut également sortir uniquement les différences (patient proposé par Opcyclin mais non inclus ou patients inclus non détectés par Opcyclin). Public visé : ARC ou promoteur ; investigateurs chargés du recrutement (TEC, médecins, etc.) pour améliorer leurs pratiques de recrutement, ou appuyer une demande de moyens supplémentaires.

La deuxième fonctionnalité (« Quels essais pour ce patient ») correspond à un seul cas d'utilisation, destiné aux investigateurs chargés du recrutement (TEC, médecins, etc.). L'objectif est de déterminer les essais cliniques pour lesquels est éligible un patient, en minimisant la saisie, et donc le temps passé sur la fonctionnalité. Un formulaire dynamique est proposé à l'utilisateur, listant les critères d'inclusion du plus discriminant (= concerne le plus d'essais cliniques pour lesquels le patient est encore éligible) au moins discriminant. Lors d'une saisie utilisateur, l'ordre des critères d'inclusion du formulaire est recalculé dynamiquement et la liste des études éligibles est actualisée.

6.1.2 Modules

Le logiciel Opcyclin est composé de différents modules. Nous avons représenté ces modules dans la figure 6.1. Opcyclin se compose d'un module principal – Opcyclin-core – qui peut fonctionner de manière indépendante. Deux modules complémentaires, Opcyclin-DM et Opcyclin-Expert, lui permettent d'être capable de travailler également sur des dossiers de moins bonne qualité : données manquantes ou manque d'uniformisation des dossiers et/ou des critères d'inclusion (ex : 6 manières différentes de coder le diabète : type 1, type 2, gestationnel, etc. ; 4 manières différentes de coder l'aspirine ; différences de vocabulaire : diabète type 1 = diabète mellitus = diabète insulino-dépendant).

Opcyclin-Core Ce module regroupe le cœur des fonctionnalités d'Opcyclin : stockage des informations (patients, essais cliniques, critères, etc.), interfaces utilisateur, fonctionnalités de pré-screening et moteur d'import des données du système d'information hospitalier (SIH) dans la base de données.

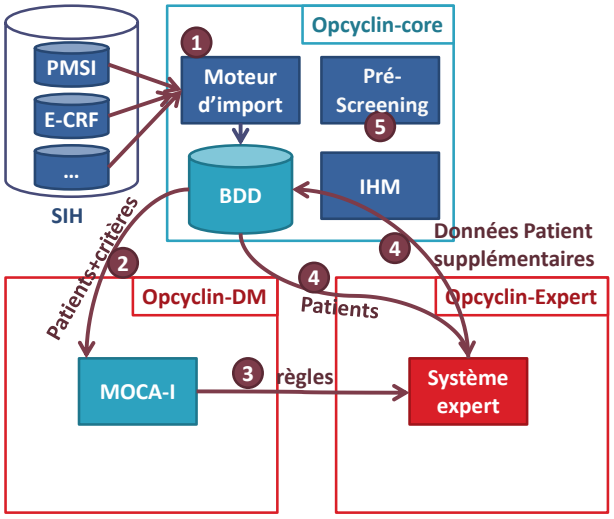


Figure 6.1: Illustration des différents modules présents dans le logiciel Opcyclin et leurs interactions

Opcyclin-DM À partir des critères d'inclusion et des données patients, ce module génère des critères d'inclusion alternatifs sous forme de règles. La majorité des travaux de cette thèse sont inclus dans ce module, on y retrouve donc tout naturellement l'algorithme MOCA-I.

Opcyclin-Expert Ce module permet d'exploiter les résultats fournis par le module Opcyclin-DM. Il correspond à un système expert, qui va utiliser les règles générées par Opcyclin-DM sur les données patients présentes dans Opcyclin-Core, afin de déclencher les critères d'inclusion sur les patients dont les dossiers contiennent des données manquantes ou non uniformisées.

6.1.3 Interactions entre les modules

Revenons sur la figure 6.1 qui présente les différents modules et leurs interactions. Cette figure illustre également les 5 étapes nécessaires pour effectuer un pré-screening sur des données manquantes ou mal uniformisées. Les étapes 2 à 4 sont celles qui vont permettre de pallier les problèmes sur les données (uniformisation ou informations manquantes). Par exemple, elles vont permettre de détecter des patients qui auraient été encodés différemment (ex : « diabète type 1 » au lieu de « diabète insulino-dépendant »). Elles vont être décrites en détail par la suite. L'étape 2 est effectuée une seule fois par critère d'inclusion, ce qui signifie qu'elle peut être réalisée à la création de l'essai, et ne sera plus nécessaire pour les recrutements en cours d'essai ou pour l'évaluation du recrutement. Les étapes 3 et 4 sont effectuées sur chaque patient, elles seront à effectuer automatiquement à chaque arrivée d'un nouveau patient.
Dans l'étape 1, les données patients sont importées du système d'information hospitalier (SIH) vers la base de données d'Opcyclin. Dès que l'import est effectué, l'utilisateur va également renseigner les essais cliniques en cours et leurs critères d'inclusion et de non inclusion. Une

fois les critères renseignés, l'étape 2 peut commencer : Opcyclin-DM va déterminer des critères d'inclusion « élargis » ou des synonymes des critères d'inclusion, sous forme de règles. À partir des dossiers de patients et de critères d'inclusion, l'algorithme *MOCA-I* va être lancé sur chacun des critères d'inclusion afin d'en déduire des règles. Dans l'étape 3, les règles obtenues sont transmises au module Opcyclin-Expert. Dans l'étape 4, le module Opcyclin-Expert va exécuter un système expert sur les règles obtenues et les dossiers de tous les patients. Le système expert va donc générer de nouvelles informations dans les dossiers patients, qui vont pouvoir être utilisées pour effectuer le pré-screening (étape 5).

6.1.4 Contributions

De nombreuses personnes sont intervenues sur la conception du logiciel Opcyclin. Nous avons réalisé les interviews métier et proposé des maquettes pour l'IHM auprès des ARCs et TECs. Le développement a été réalisé par la société *Alicante*. Pour ma part, je suis intervenue sur l'analyse métier, la conception de la base de données, le développement d'une partie du moteur de pré-screening (calcul du score essai/patient, gestions des horaires de présence du personnel de recrutement), de l'IHM (corrections mineures et tests fonctionnels) et du moteur expérimental de MOCA-I. D'autre part, j'ai encadré certains travaux sur l'IHM et le pré-screening. Julien Taillard (*Alicante*) est intervenu sur le moteur d'import, l'IHM, le pré-screening et également en encadrement. Benjamin Digeon (*Alicante*) est intervenu sur l'IHM, les tests unitaires et le pré-screening. Enfin, Charles Christiaens (*Alicante*) s'est chargé de l'IHM, d'une grande partie des correctifs et est également intervenu sur le pré-screening.

6.2 Opcyclin-Core - le coeur du logiciel Opcyclin

Le module Opcyclin-Core regroupe les fonctionnalités centrales d'Opcyclin, on y retrouve le moteur d'import des données, la base de données, la fonctionnalité de pré-screening et l'interface utilisateur. Nous allons maintenant détailler leur fonctionnement.

6.2.1 Base de données

La base de données est hébergée sur Postgres 9.2, un système de gestion de bases de données libre. Elle contient 28 tables. Le modèle de données est inspiré de ce qui a été proposé dans le projet ANR AKENATON [17], qui a été adapté afin de gérer également les essais cliniques et leurs critères d'inclusion, et le système expert. Le modèle de données a été conçu afin d'accueillir des données en provenance de différents systèmes d'information. Ainsi, il est totalement générique, ce qui le rend compatible nativement avec de nombreuses sources de données et simplifie les opérations d'import.

6.2.2 Moteur d'import

Le moteur d'import fait la liaison entre les données présentes dans le système d'information de l'hôpital (SIH) et la base de données d'Opcyclin, qui va servir à détecter des patients. À ce jour, des connecteurs sont disponibles pour les données du PMSI et pour l'E-CRF Openclinica.

6.2.3 Pré-screening : correspondance patients et essais

Ce module calcule les correspondances entre un patient et des critères d'inclusion (ce qui correspondra dans la suite à l'étape 2). Au préalable, l'utilisateur doit avoir renseigné les critères d'inclusion (dans la suite, étape 1). Le moteur s'en sert ensuite pour calculer les correspondances entre un patient et des essais cliniques (dans la suite, étape 3), qui seront détaillées dans l'interface.

Étape 1 : saisie des critères d'inclusion Dans Opcyclin, les critères d'inclusion peuvent prendre différentes formes :

Relation d'ordre Valeur inférieure ou supérieure à une borne (ex : score NIHSS < 4)

Intervalle Valeur comprise entre deux bornes (ex : $4 <$ score NIHSS < 10)

Présence Présence d'une information (ex : présence d'un diagnostic de diabète)

Délai Pour les essais cliniques sur pathologies aiguës : vérifie si la pathologie étudiée respecte un délai, sous la forme d'une borne ou d'un intervalle (ex : AVC de moins de 4h : le patient arrive dans le service avec un AVC qui date de moins de 4h).

Pour chaque critère d'inclusion appartenant à un même essai clinique, l'utilisateur a la possibilité de définir une pondération. Cela permet de donner plus ou moins de poids à certains critères d'inclusion, par exemple les critères qui ne sont pas nécessaires pour faire partie du groupe témoin de l'essai clinique. (ex : une étude qui teste un traitement pour les AVC de moins de 4h, pour faire partie du groupe témoin le patient peut avoir un AVC un peu moins récent.)
Pour certains critères d'inclusion, notamment les délais, il est possible d'ajouter une notion d'horaire du personnel de recrutement (ex : AVC de moins de 4h, le lundi et mardi de 9h à 14h). Si le critère n'est pas compatible avec la fenêtre horaire fournie, le critère n'est pas déclenché. Pour le critère « AVC de moins de 4h, le lundi et mardi de 9h à 14h », un patient qui montre des signes d'AVC à 0h00 et arrive dans le service à 2h ne déclenchera pas le critère. Son AVC date bien de moins de 4h à son arrivée dans le service, mais l'équipe de recrutement n'est pas disponible pour l'inclure dans l'essai clinique.

Étape 2 : évaluation des critères sur les données patients Pour chaque couple <critère d'inclusion> et <patient>, il s'agit d'évaluer si la valeur de la donnée patient est valide avec le critère d'inclusion (ex : score NIHSS < 4 et valeur de 5). Comme les critères d'inclusion peuvent s'exprimer sous de nombreuses formes, une procédure stockée passe sur l'ensemble des critères d'inclusion et des patients, et calcule un score de correspondance (1 si le critère est ok, 0 sinon). Les modules Opcyclin-DM et Opcyclin-Expert pourront également renseigner des scores de

correspondance, qui cette fois fourniront une valeur comprise entre 0 et 1 qui correspond à un indice de confiance. Nous détaillerons quelques pistes pour effectuer son calcul plus loin dans ce document.

Étape 3 : évaluation de la correspondance patient essais De l'étape précédente, on dispose pour chaque patient p et chaque critère d'inclusion c_i le score obtenu score(p,c_i). On connaît également la pondération associée à chaque critère c_k de l'étude E : poids(E,c_i). Le score est calculé de la manière suivante :

$$score(p, E) = \sum_{i=1}^{n} score(p, c_i) \times \frac{poids(E, c_i)}{\sum_{k=1}^{n} poids(E, c_k)}$$

Lorsqu'au moins un des critères n'est pas respecté, le score est affiché en rouge, ce qui permet d'exclure ces patients de la recherche.

6.2.4 Interface utilisateur

L'interface utilisateur est réalisée en Java 6, avec les frameworks Spring et JSF2, et les composants Primefaces. Nous présentons ici les différentes interfaces présentes dans le logiciel.

6.2.4.1 Pré-screening minute

Priorisation de la saisie La capture d'écran présente dans la figure 6.2 propose une interface pour aider à l'identification d'essais candidats pour un patient. Elle provient d'une version d'Opcyclin adaptée pour l'AVC (et les essais cliniques sur les pathologies aiguës), qui contient des champs pour préciser la date de l'AVC et de l'entrée. Sur la gauche, le système va demander de remplir les critères d'inclusion manquants, en commençant par le plus discriminant. Sur la droite, il propose les essais avec le score de correspondance associé. Un diagramme circulaire permet de visualiser pour chaque essai les critères respectés par le patient (en vert), non respectés par le patient (en rouge) et non renseignés (en gris). Les critères non renseignés qui sont les plus susceptibles de filtrer la liste d'essais sont demandés en priorité, afin de limiter la saisie et donc le temps nécessaire pour trouver des essais candidats : dans la capture d'écran c'est le champ « AIC » qui permet de sélectionner ou d'exclure le plus d'essais. Ensuite, parmi les essais restants (les essais 3 et 6 sont exclus car un des critères n'est pas respecté), le critère qui permet de trancher sur le plus d'essais va être sélectionné ; ici le critère NIHSS.

6.2.4.2 Pré-screening de masse

La capture d'écran de la figure 6.3 présente une liste de patients potentiels pour l'ensemble des essais en cours dans le service. Chaque ligne représente un dossier patient, et chaque colonne les essais ouverts à l'inclusion dans le centre. Le système récupère les dossiers patient pour élaborer la liste des patients les plus susceptibles de pouvoir participer à un essai, placés en tête du tableau (dans la capture d'écran, l'utilisateur a choisi de les trier pour l'essai 1). Pour

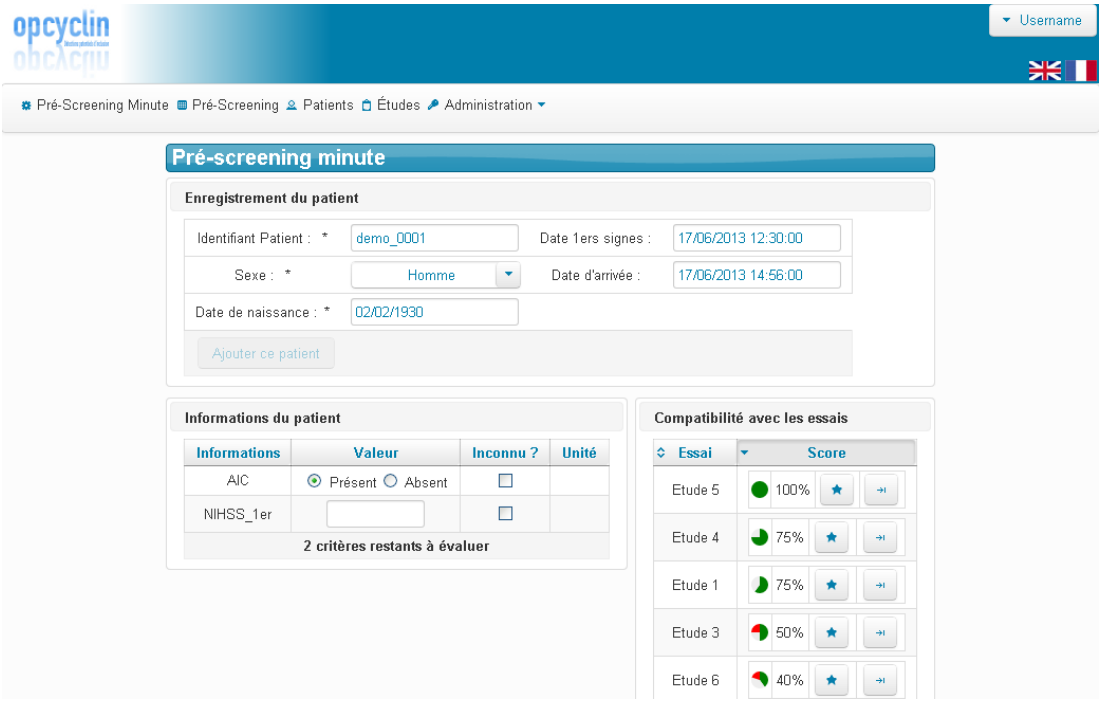


Figure 6.2: Identification d'essais candidats pour un patient en consultation - Capture d'écran

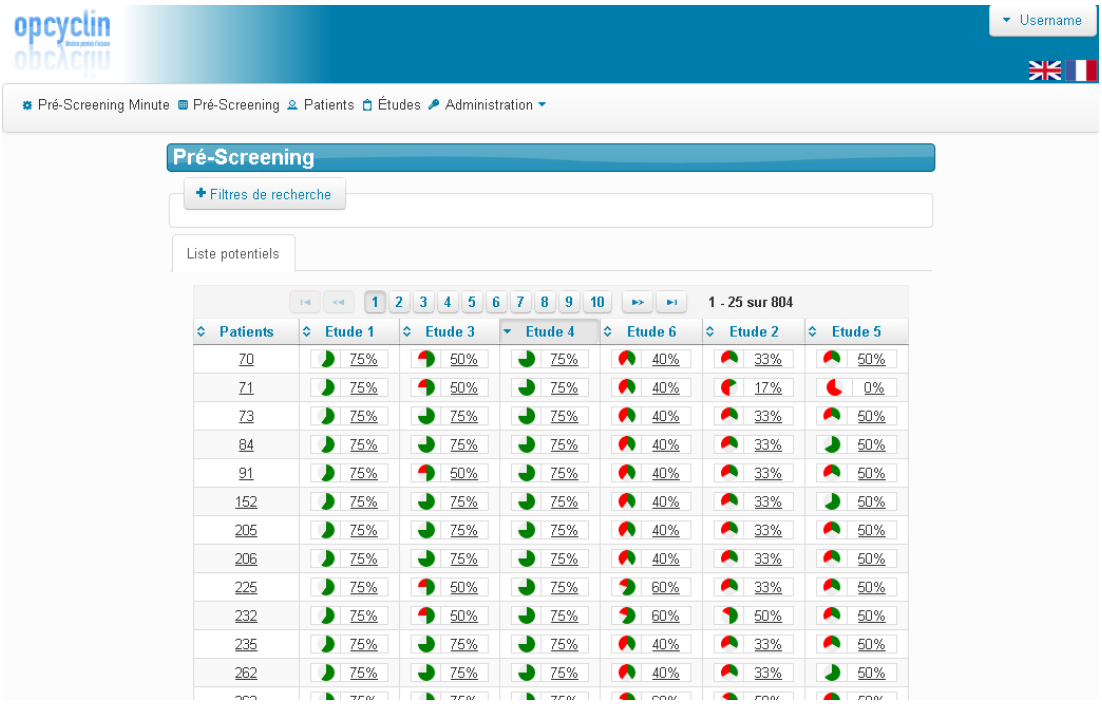


Figure 6.3: Identification de patients potentiels pour un ou des essais - Capture d'écran

chacun d'eux la possibilité de participer aux essais est évaluée : si tous les critères sont respectés l'utilisateur peut inclure le patient, si un critère est non respecté le patient est exclus. Lorsqu'il manque des informations, le système fournit un score de correspondance, dont la méthode de calcul est proposée dans la suite de ce document. Un diagramme circulaire indique pour chaque essai et chaque patient la proportion de critères respectés, non respectés et non renseignés. Lors du survol d'un essai et d'un patient, le système va afficher le détail des critères respectés. La capture d'écran de la figure 6.4 offre une vue plus détaillée pour un patient donné. Elle

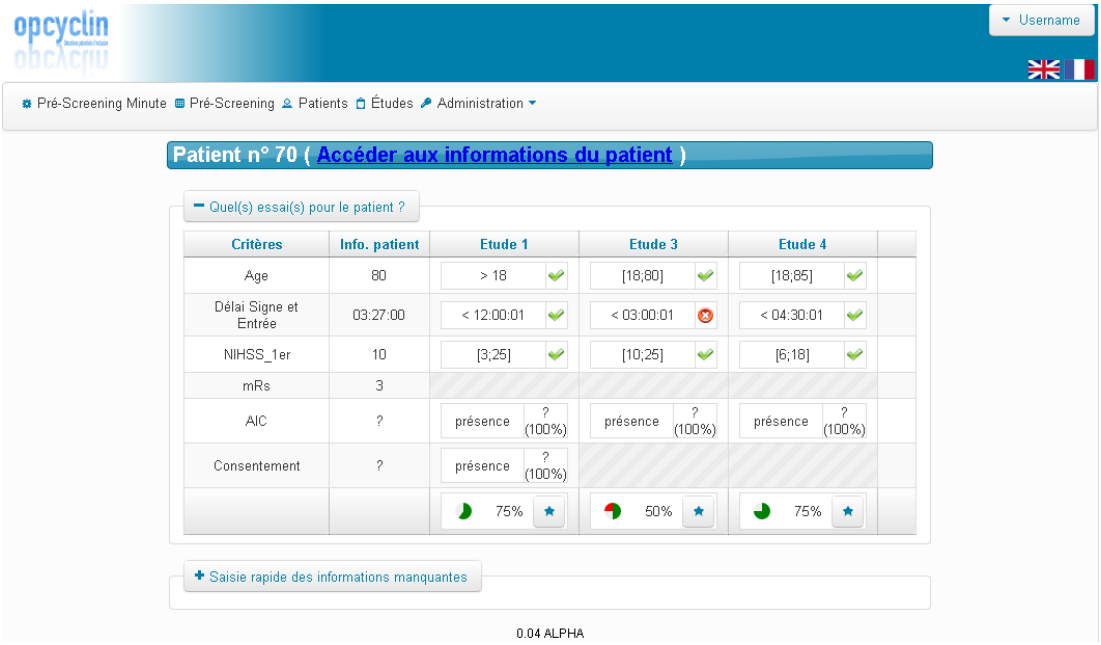


Figure 6.4: Identification d'essais candidats pour un patient en consultation - Capture d'écran

propose un tableau associant les différents critères, la valeur dans le dossier du patient, et les valeurs nécessaires pour chaque essai clinique, en indiquant si elles sont respectées ou non. De manière similaire à l'écran précédent, la dernière ligne indique pour chaque essai le score de correspondance du patient et la proportion de critères respectés/non respectés/non renseignés.

6.3 Opcyclin-DM : le moteur d'apprentissage (MOCA-I)

Comme nous l'avions vu précédemment, le rôle du moteur d'apprentissage est de générer des règles à partir des critères d'inclusion et des données patients. À la différence des systèmes experts où des experts vont élaborer pendant plusieurs mois une base de connaissances, elle est ici générée automatiquement à partir des données patients. Le moteur utilisé est l'algorithme MOCA-I qui est une des contributions principales de cette thèse. MOCA-I est réalisé en C++, la version expérimentale utilise le framework ParadisEO [19]. Dans la suite, nous allons évaluer

la qualité des règles générées par MOCA-I, à la fois de manière quantitative et qualitative.

6.3.1 Évaluation quantitative

Dans le chapitre 3, nous avions comparé les résultats de MOCA-I et de la littérature sur des jeux de données basées sur les données PMSI. MOCA-I obtient des ensembles de règles statistiquement plus performants que les classifieurs proposés par les autres algorithmes (*BioHEL*, *C4.5-CS* et *Ripper*). Les prédictions obtenues sont également plus simples, contenant jusqu'à 25 termes en moyenne, contre 50 à 100 pour les autres algorithmes. Ce qui facilite l'interprétation des règles, et va rendre le passage du système expert plus simple. Au niveau du temps d'exécution, il faut compter en moyenne 3 à 4h par critère d'inclusion, pour 10 000 patients et jusqu'à 10 055 attributs et un nombre de règles par ensemble fixé à 5 (exécutions sur un Intel Xeon 2.66 GHz à 8Go ram). Ce temps peut sembler important au premier abord, mais l'idée est ici d'éviter l'intervention d'un expert pour la réalisation du système expert, ce qui peut prendre plusieurs mois. Il correspond à une phase d'initialisation : il est uniquement réalisé à la création de l'essai clinique et de ses critères d'inclusion. De plus, une partie des critères d'inclusion sont souvent identiques d'un essai clinique à l'autre dans un même service : il ne sera pas nécessaire d'extraire à nouveau les règles pour des critères déjà connus.

6.3.2 Évaluation qualitative

Nous allons analyser la pertinence et la cohérence des prédictions générées par MOCA-I. Pour cela, MOCA-I est exécuté sur 10000 patients anonymisés issus du PMSI. L'objectif est de prédire la présence d'un *AVC* ou non dans le dossier ; ce qui correspond aux codes CIM10 de I61 à I64. Voici un des ensembles de règles obtenus :

- DZQM006 exists $\wedge$ G8100 exists
- 01M301 exists
- 01M302 exists
- EBQM001 exists $\wedge$ YYYY467 exists $\wedge$ G819 exists
- 01M304 exists

Cette règle a une confiance de 0.99 et une sensibilité de 0.48, ce qui signifie que lorsqu'elle détecte la présence du code I61 à I64 elle est juste dans 99% des cas, et elle identifie 48% des patients avec le code I61 à I64. Les règles trouvées sont cohérentes d'un point de vue métier :

- DZQM006 correspond à une échographie du cœur, G8100 à une hémiplégie. La majorité des hémiplégies sont causées par un AVC, il est donc cohérent d'y retrouver ce code.
- 01M301, 01M302 et 01M304 correspondent à des codes GHM (groupe homogène de malades). À la fin de chaque séjour PMSI, l'ensemble des données PMSI du séjour (actes et diagnostics) sont passées dans un logiciel de groupage, qui va classer le séjour dans un groupe de séjours. Les règles de groupage sont élaborées par des experts médicaux. Ici, ces codes correspondent aux GHM « Accidents vasculaires intracérébraux non transitoires, niveau 1, 2 et 4 ». Les codes obtenus sont également cohérents d'un point de vue métier.

De plus, MOCA-I a réussi à trouver automatiquement des « synonymes » des codes I61 à I64 sans devoir faire appel à un expert en codage PMSI.

– EBQM001, YYYY467 et G819. Ces codes correspondent respectivement à « Échographie-doppler des artères cervicocéphaliques extracrâniennes », « Supplément pour injection intraveineuse de produit de contraste » et « Hémiplégie, sans précision ». Cette règle est un peu similaire à la règle précédente sur l'hémiplégie. Le fait que l'hémiplégie soit associée à d'autres actes permet probablement à l'ensemble de règles de distinguer les hémiplégies non causées par un AVC.

Parmi les patients détectés par l'ensemble de règles signalé plus haut, il y a un patient pour lequel les codes I61 à I64 ne sont pas retrouvés. Il s'agit typiquement d'un patient cible pour Opcyclin-DM : un patient que l'on suspecte fortement d'avoir un critère d'inclusion (ensemble de règles de confiance 0.99), mais que l'on aurait raté en utilisant une approche classique (Opcyclin-Core utilisé seul) à cause d'un manque d'uniformisation des données ou d'une erreur de saisie. On peut imaginer par exemple que la saisie de son dossier a été effectué par une personne dont les habitudes de saisie sont différentes, ou dans l'urgence.

Dans d'autres ensembles de règles, on peut également retrouver la règle suivante, qui est également intéressante :

– ACQK001 exists $\wedge$ DEQP007 exists $\wedge$ E116 exists $\wedge$ I10 exists $\wedge$ genre = homme

Les codes ACQK001, DEQP007, E116 et I10 correspondent respectivement à :

– « Scanographie du crâne et de son contenu »
– « Surveillance continue de l'électrocardiogramme »
– « Diabète sucré non insulino-dépendant, avec autres complications précisées »
– « Hypertension essentielle (primitive) »

Cette règle est intéressante, car on y retrouve un examen qui est souvent réalisé pour les AVC, la scanographie du crâne. Cependant, une scanographie du crâne peut être réalisée dans de nombreux autres cas que l'AVC. Les autres diagnostics, diabète sucré et hypertension correspondent à des facteurs de risque de l'AVC, ce qui permet à la règle de « filtrer » les scanographies du crâne hors AVC. Ainsi, lorsqu'un patient avec des facteurs de risques de l'AVC se présente pour une scanographie du crâne, il y a en fait une plus forte probabilité que l'on se trouve dans le cas de l'AVC plutôt que des autres pathologies nécessitant une scanographie du crâne (accident, etc.).

6.4 Opcyclin-Expert : le système expert

Nous allons maintenant voir comment utiliser les règles générées précédemment. L'objectif du module Opcyclin-Expert est d'évaluer pour chaque patient et chaque critère d'inclusion, s'il existe des règles qui suspectent la présence du critère d'inclusion chez le patient. Le couple <patient, critère> se verra assigner une probabilité de vraisemblance, dont nous proposons des règles de calcul dans la suite. Les scores générés pourront compléter les scores déjà utilisés dans le processus de pré-screening pour évaluer la correspondance d'un patient à un essai. Nous donnerons également quelques pistes afin de permettre l'amélioration des résultats selon les

retours des utilisateurs (ex : patients inclus et identification de faux positifs).

6.4.1 Calcul du score final : quelques pistes

Le moteur de pré-screening présenté précédemment est fondé sur la notion de score entre un patient et un critère d'inclusion : 1 si le patient respecte le critère, et 0 sinon. Dans le cas où l'information n'est pas présente directement dans le dossier du patient mais est détectée par le système expert, il est souhaitable d'obtenir un score intermédiaire. Ce score refléterait la fiabilité de la règle, et aurait ensuite un impact sur le score de correspondance du patient pour l'essai. Cette méthode rend possible l'ordonnancement des patients selon leur score de correspondance. L'utilisateur pourra commencer par s'intéresser aux patients avec un score élevé, et s'il manque d'inclusions pour l'essai clinique, passer un peu plus de temps sur les dossiers potentiels. Dans la suite, nous commençons tout d'abord par présenter les propriétés souhaitables lors de la traduction des règles en score. Nous proposerons ensuite quelques solutions pour obtenir un score à partir d'une liste de règles déclenchées pour un patient. Il sera cependant nécessaire d'attendre les premières mises en place du logiciel Opcyclin avant de pouvoir les évaluer sur des cas réels d'utilisation.

6.4.1.1 Propriétés désirées du score

Impact de la confiance de la règle La *confiance* mesure le pourcentage des prédictions (positives) fournies par la règle qui s'avèrent justes. Une *confiance* proche de 1 indique que la règle réalise peu d'erreurs, et que sa prédiction est vraisemblable. Il est donc préférable qu'une règle avec une forte *confiance* donne un score élevé au patient qui la déclenche, pour le critère d'inclusion concerné.

Impact de la sensibilité de la règle La *sensibilité* mesure le pourcentage des patients ayant le critère d'inclusion qui sont détectés par la règle. Une forte *sensibilité* indique que la règle est vérifiée sur un grand nombre de patients. Cette mesure est intéressante si elle est combinée à la *confiance* : une règle de forte *confiance* est encore plus intéressante si elle est vérifiée sur un grand nombre de patients (forte *sensibilité*). À *confiance* égale, une règle qui donne une *sensibilité* plus élevée qu'une autre doit générer un meilleur score.

Chevauchement de plusieurs règles Les règles générées par MOCA-I peuvent se chevaucher, il peut donc arriver qu'un patient déclenche plusieurs règles pour un même critère d'inclusion. Il semble naturel de penser que lorsque plusieurs règles sont déclenchées, la probabilité que le patient ait le critère doit être encore plus forte. Le score va aussi dépendre des règles déclenchées. Supposons que deux patients p1 et p2 déclenchent les règles fournies dans la table 6.1. Les deux patients déclenchent tous deux la règle $R2$ ainsi qu'une seconde règle : $R1$ pour le patient $p1$ et $R3$ pour le patient $p2$. Les règles déclenchées par le patient $p1$ ont l'attribut $Att5$ en commun et sont donc assez similaires, tandis que le patient $p2$ déclenche des règles tout

Table 6.1: Illustration de règles déclenchées par des patients

Patient	Règle	Confiance	Sensibilité
p1	R1 **Att1** ∧ **Att5**	0.85	0.09
	R2 **Att2** ∧ **Att5**	0.9	0.12
p2	R2 **Att2** ∧ **Att5**	0.9	0.12
	R3 **Att3** ∧ **Att4**	0.85	0.09

à fait différentes. Dans ce cas, il est souhaitable que le patient $p2$ ait un score plus important que le patient $p1$ (ou un bonus de score) car des règles très différentes sont déclenchées.

6.4.1.2 Solutions proposées

Nous proposons maintenant des approches possibles pour l'élaboration du score, en respectant les contraintes présentées précédemment.

Obtention du score d'une règle Une première mesure qui peut être utilisée pour obtenir un score à partir de la *confiance* et la *sensibilité* d'une règle est la *f-mesure*. Elle dispose d'un paramètre β qui permet d'accorder plus ou moins d'importance à la *confiance* ou à la *sensibilité*. Ce paramètre sera à ajuster expérimentalement afin de se tenir aux contraintes fixées précédemment, avec des valeurs de $\beta < 1$, ce qui favorise la forte *confiance*.

Chevauchement de plusieurs règles Certains travaux de la littérature offrent des pistes intéressantes pour la gestion du score lorsque plusieurs règles se chevauchent. Zhang *et al.* ont effectué des travaux pour classer des entreprises selon leur risque de fraude, évalué à l'aide de règles de décision [118]. Même si ces travaux concernent du classement, on y retrouve des pistes intéressantes pour l'obtention d'un score. Ils ont en effet comparé différentes méthodes pour gérer les règles chevauchantes :

Maximum Le score final correspond au plus grand score parmi toutes les règles déclenchées

Moyenne Le score final correspond à la moyenne des scores de toutes les règles déclenchées

Somme probabiliste Pour un patient p et des règles $R_1, R_2, \ldots, R_n$.

$$\begin{aligned} score(p, \{R_1\}) &= score(p, R_1) \\ score(p, \{R_1, R_2\}) &= score(p, R_1) + score(p, R_2) \\ &- score(p, R_1) \times score(p, R_2) \\ score(p, \{R_1, \ldots, R_n\}) &= score(p, \{R_1, \ldots, R_{n-1}\}) + score(p, R_n) \\ &- score(p, \{R_1, \ldots, R_{n-1}\}) \times score(p, R_n) \end{aligned}$$

La méthode de la somme probabiliste s'est avérée la plus performante.
Il est également envisageable de pondérer le score lorsque plusieurs règles se déclenchent, selon

leur degré de ressemblance. Ainsi, Iglesia *et al.* proposent de mesurer la dissimilarité entre deux règles [5], ou encore d'utiliser la mesure de Jaccard. Une autre approche peut être également d'utiliser le *lift*, qui peut permettre de mesurer le degré d'indépendance entre deux règles.

6.4.2 Feedback utilisateur et amélioration des résultats

Afin d'améliorer les résultats d'Opcyclin, il est prévu de laisser l'opportunité aux utilisateurs d'améliorer le logiciel. Le retour des utilisateurs peut être effectué de manière passive ou active. De manière passive, l'inclusion d'un patient signifie que tous ses critères d'inclusion sont validés, ce qui est une indication supplémentaire pour l'apprentissage. De manière active, l'utilisateur peut indiquer les faux positifs ramenés par Opcyclin (patient avec un score élevé mais dont le dossier ne permet pas une inclusion).
Ensuite, le retour utilisateur peut être traité de deux manières :

- Désactiver les règles qui ramènent trop de faux positifs
- Relancer le moteur d'apprentissage MOCA-I avec le résultat des inclusions et les retours utilisateurs, afin d'affiner le modèle

6.5 Conclusion

Dans ce chapitre, nous avons vu comment les différents travaux présentés dans cette thèse peuvent s'intégrer au sein d'une suite logicielle : Opcyclin. Nous avons tout d'abord rappelé les différents *scenarii* et cas d'utilisation implémentés dans Opcyclin. Nous avons ensuite présenté les différents modules qui composent le logiciel Opcyclin, dont le module Opcyclin-DM qui contient le moteur MOCA-I développé dans les chapitres précédents. Nous avons ensuite détaillé les différentes interactions qui lient ces modules entre eux. Par la suite, nous avons décrit précisément chacun des modules. En particulier, nous avons expliqué comment il est possible de calculer le score de correspondance entre un patient et un essai clinique dans le module Opcyclin-Core, afin d'effectuer le pré-screening. Ensuite, nous avons présenté les différents écrans de l'interface principale. Puis, nous avons évalué les règles produites par le module Opcyclin-DM (algorithme MOCA-I) d'un point de vue qualitatif et quantitatif, par rapport au besoin métier. Des deux points de vue, le règles produites s'avèrent correspondre au besoin métier. Enfin, nous avons vu quelques pistes pour calculer un score < patient, critère > en fonction des règles déclenchées, à utiliser dans le module Opcyclin-Core pour le pré-screening.
Quelques perspectives se dessinent pour Opcyclin. Une première version d'Opcyclin adaptée à l'AVC, Opcyclin-Stroke, est en cours de test au sein du service Neurovasculaire du CHRU de Lille. Les données utilisées proviennent d'une file active de patients : il s'agit de l'ensemble des patients qui sont passés dans le service sur une période donnée. Elles ont la particularité d'être de très bonne qualité, les données collectées sont exhaustives et uniformisées. Elles permettront donc de tester les premiers mécanismes d'Opcyclin. Une autre expérimentation est prévue au CHU de Montpellier dans le cadre du projet ANR TECSAN Clinmine. D'une part pour la réalisation d'études de faisabilité d'essais cliniques, et d'autre part pour le pré-screening de

patients. Elle vise cette fois-ci des essais cliniques sur des pathologies chroniques. Le projet Clinmine est dédié à la prédiction sur des données médicales temporelles. Aussi, dans le cadre du projet Clinmine, une version 2.0 d'Opcyclin est envisagée, avec la possibilité de gérer la notion de temporalité dans MOCA-I.

Conclusion

Le travail effectué dans cette thèse a été financé par la société *Alicante*, PME spécialisée dans les progiciels pour les hôpitaux. Cette entreprise intervient dans de nombreux domaines : identito-vigilance, décisionnel hospitalier, bibliométrie ou encore recherche clinique. C'est dans ce dernier domaine que se positionne le travail issu de cette thèse, en particulier sur l'amélioration du recrutement dans les essais cliniques, en utilisant les données numériques disponibles à l'hôpital. Ce travail est réalisé en collaboration avec l'équipe Dolphin (Discrete multiobjective Optimization for Large-scale Problems with Hybrid dIstributed techNiques) de Inria Lille Nord Europe et du laboratoire LIFL de l'Université Lille 1. L'équipe Dolphin est forte d'une expérience dans les méthodes d'optimisation et méta-heuristiques et leur application en fouille de données.
L'automatisation du recrutement pour les essais cliniques n'est pas un problème simple. Les données présentes à l'hôpital montrent des problèmes d'uniformisation, de complexité du codage ou souffrent parfois de données manquantes. Par conséquent, certains outils de recrutement proposés dans la littérature utilisent un système expert pour remédier à ces problèmes de qualité des données. La réalisation d'un système expert est très coûteuse en ressources humaines, nous proposons donc dans cette thèse de générer les règles du système expert à l'aide de méthodes de fouilles de données, en particulier de classification. Le travail présenté dans cette thèse se focalise sur la réalisation d'un algorithme de classification adapté aux données médicales. Nous avons vu que celles-ci présentent trois particularités qui mettent à mal les algorithmes de classification : l'incertitude, l'asymétrie et la volumétrie. Nous rappelons ici les principales contributions de cette thèse.

La principale contribution de cette thèse est l'algorithme MOCA-I, qui propose une résolution du problème de classification partielle sur données asymétriques sous la forme d'un problème multi-objectif. Les critères utilisés pour trouver les règles visent à maximiser les performances en classification, tout en minimisant la taille des règles obtenues, permettant d'obtenir des modèles plus simples pour l'utilisateur. De plus, MOCA-I recherche des ensembles de règles (représentation Pittsburgh), qui comme nous l'avons montré vont permettre de trouver de règles qui n'auraient pas été trouvées avec une représentation basée sur les règles (approche Michigan). D'autre part, MOCA-I manipule des ensembles de règles partielles, ce qui permet d'obtenir des ensembles de règles dont les règles ne présentent pas d'incohérences. MOCA-I est basé sur une méthode de recherche locale multi-objectif (l'algorithme DMLS : dominance-based

multiobjective local search), ce qui permet de limiter le paramétrage par rapport à une approche plus classique en optimisation multi-objectif, basée sur un algorithme génétique. Nous avons vérifié les apports de la modélisation multi-objectif, qui s'avère statistiquement plus performante qu'une modélisation mono-objectif utilisant les mêmes objectifs, à l'aide d'une agrégation. Nous avons ensuite déterminé expérimentalement le paramétrage le plus robuste pour MOCA-I, nous permettant de comparer celui-ci aux meilleurs algorithmes de la littérature. Nous avons ainsi observé que MOCA-I est statistiquement plus efficace que les 14 algorithmes de la littérature auxquels il a été comparé, à la fois sur des jeux de données de la littérature présentant un taux d'asymétrie important (classe à prédire minoritaire), et sur deux jeux de données réels, issus de données hospitalières. Les modèles générés par MOCA-I sont également 2 à 6 fois plus compacts que ceux générés par les meilleurs algorithmes de la littérature comme *C4.5-CS*, *GAssist* ou *Ripper*. L'approche MOCA-I a donc montré ses performances pour traiter des données discrètes, sur des jeux de données dont la classe à prédire est minoritaire.

Nous avons ensuite étudié le comportement de MOCA-I dans un contexte de classification plus standard, où les données ne présentent pas d'incertitude, d'asymétrie ni de problème de volumétrie, ce qui est beaucoup plus étudié dans la littérature. Nous avons alors proposé une version plus adaptée à ce type de données en utilisant d'autres critères d'évaluation : MOCA$_{SeSpMDL}$ qui s'avère statistiquement plus efficace que MOCA-I sur la classification binaire standard. De plus, cette version est également statistiquement équivalente aux meilleurs algorithmes de classification de la littérature.

En vue de comprendre l'influence de l'asymétrie, nous avons ensuite étudié le comportement de MOCA-I et MOCA$_{SeSpMDL}$ en fonction de l'asymétrie des jeux de données étudiés. Les expérimentations ont confirmé qu'il est préférable d'utiliser MOCA-I lorsque la classe à prédire est minoritaire, en mettant en évidence une frontière à partir d'un degré d'asymétrie égal à 0.5. Dans l'autre cas, MOCA$_{SeSpMDL}$ est à préférer.

Puis, nous nous sommes intéressés aux solutions de compromis générées par MOCA-I : comment choisir une solution parmi toutes celles obtenues ? Nous avons tout d'abord montré que le choix de la solution qui obtient le meilleur score sur les données d'apprentissage n'est pas toujours efficace. Souvent, la solution choisie n'est pas celle qui donne les meilleurs résultats sur les données de test, ce qui signifie qu'elle ne dispose pas de bonnes capacités de généralisation. Nous avons proposé de nombreuses autres approches pour améliorer les résultats. Une partie de ces approches sont basées sur le choix d'une solution parmi celles générées, en essayant par exemple de choisir la solution la moins sensible au phénomène de sur-apprentissage. L'autre partie des approches proposées est basée sur la génération d'une solution à partir d'une fusion de l'ensemble des solutions générées par l'algorithme (population finale). Nous avons également étudié des solutions basées sur la construction de la courbe ROC, utilisée dans le domaine médical. Dans ce cadre, nous avons donc proposé une méthode de construction de la courbe ROC d'un ensemble de règles. Ces méthodes laissent la possibilité à l'utilisateur d'adapter la

solution générée à ses besoins : ratisser plus large au risque de provoquer des faux positifs lorsqu'il dispose de plus de temps, ou au contraire limiter les faux positifs s'il a besoin de résultats très précis. Cependant, toutes les approches proposées se sont avérées moins efficaces que la méthode simple, qui consiste à choisir la solution qui obtient le meilleur score sur les données d'apprentissage. Ce travail exploratoire souligne les difficultés provoquées par le phénomène de sur-apprentissage, qui produit des règles trop spécialisées et est probablement aggravé par la présence d'une asymétrie sur les données.

Finalement, nous avons montré comment l'algorithme MOCA-I peut s'intégrer dans le logiciel Opcyclin. Nous avons également proposé une méthode pour calculer le score de correspondance entre un patient et un essai clinique. Enfin, nous avons donné quelques pistes pour intégrer le résultat de MOCA-I dans le calcul de ce score.

Plusieurs perspectives sont envisagées à ce travail. Une partie d'entre-elles sont dans la continuité du travail de cette thèse et concernent le moteur MOCA-I. Ainsi, une perspective pourrait être de compléter l'étude sur le sur-apprentissage : d'autres techniques peuvent être utilisées pour le contrer, comme par exemple le bootstrapping. Il peut également être intéressant d'analyser le comportement du sur-apprentissage selon le degré d'asymétrie. Cela permettrait de concevoir de meilleures méthodes pour le contrer. Une autre perspective concerne les critères objectifs : nous avons montré qu'en théorie la *f-mesure* est plus efficace que l'*exactitude* en début de recherche, ce qui s'inverse lorsque le classifieur devient plus performant. Il serait donc intéressant de tester l'impact d'un critère objectif dynamique, où l'on optimise la *f-mesure* en début de recherche, pour ensuite optimiser l'*exactitude*.

D'autres perspectives à court terme concernent les expérimentations d'Opcyclin en utilisation réelle. Des installations sont prévues dans le service neurovasculaire du CHRU de Lille, et au CHU de Montpellier. Ces installations entrent dans le cadre du futur projet ANR Clinmine. Le projet ANR Clinmine s'inscrit dans la continuité des travaux qui ont été entrepris dans cette thèse sur l'exploitation des données médicales. Il part du constat que les établissements de soins français collectent de plus en plus de données sans avoir la possibilité de les exploiter pleinement. L'objectif de Clinmine est de fournir une plate-forme communautaire pour exploiter ces données, en y ajoutant la notion de trajectoire patient (succession de séjours). Il implique la société *Alicante*, les équipes DOLPHIN et MODAL de Inria, les CHRU de Lille et de Montpellier, ainsi que les cliniques du GHICL. Cependant le cadre est beaucoup plus large que celui de cette thèse, cette fois des méthodes de clustering (classification non supervisée) et de classification supervisée seront également utilisées pour analyser les données médicales. Le projet Clinmine s'articule autour de deux concepts innovants. Le premier est d'hybrider des méthodes informatiques et statistiques, afin de permettre de classifier de grands volumes de données, en bénéficiant des avantages apportés par les deux approches. Le second est de prendre en compte la dimension temporelle des parcours des patients, par le biais d'une modélisation statistique adaptée. Le projet Clinmine devra lever plusieurs verrous scientifiques. Un premier verrou est

posé par la prise en compte des variables temporelles, afin de pouvoir analyser les trajectoires patient. Comme le projet Clinmine traite lui aussi des données médicales, il aura lui aussi affaire aux différents verrous qui ont été levés dans cette thèse dans le contexte de la classification supervisée : asymétrie des données, incertitude et volumétrie.

Pour en revenir aux perspectives, dans le cadre du projet Clinmine, une perspective à moyen terme est de proposer une nouvelle version de MOCA-I adaptée à la gestion des attributs numériques : MOCA-I-Q. Toujours dans le cadre du projet Clinmine, une perspective à plus long terme est l'adaptation de MOCA-I pour la gestion de la temporalité : MOCA-I-QT. Le moteur ainsi généré pourra être utilisé dans un cadre plus large que les essais cliniques. Deux applications sont ainsi envisagées au sein du projet Clinmine. La première application concerne le suivi des patients ayant subi un AVC. Il s'agit d'analyser les parcours de ces patients afin d'identifier, dans un premier cas d'utilisation, les facteurs ou indicateurs de risque de récidive, dans un deuxième cas d'utilisation, les indicateurs de séquelles de l'AVC (épilepsie, dépression, trouble cognitif). Pour ces deux cas d'utilisation, l'objectif est d'identifier des profils types de trajectoire patient, afin de proposer lorsqu'un nouveau patient se présente, les évolutions possibles selon son profil type, ce qui permettra au personnel de santé de planifier les actions adaptées. La deuxième application concerne l'analyse des trajectoires patient au sein d'un hôpital. Elle s'intéresse aux séjours ambulatoires, l'objectif est d'identifier des anomalies (séjours à vocation ambulatoire traités en hospitalisation complète ou séjours en ambulatoire qui auraient dû faire l'objet d'une hospitalisation complète). Ainsi, tout nouveau séjour pourra faire l'objet de suggestions de prise en charge selon le profil du patient. Ce qui permettra d'orienter la prise en charge du patient si son profil se rapproche d'une situation à risque (ex : X% des patients présentant ce profil présentent une complication → préférer une hospitalisation complète).

Bibliographie

[1] H. Abe and S. Tsumoto. Analyzing correlation coefficients of objective rule evaluation indices on classification rules. In Guoyin Wang, Tianrui Li, Jerzy Grzymala-Busse, Duoqian Miao, Andrzej Skowron, and Yiyu Yao, editors, *Rough Sets and Knowledge Technology*, volume 5009 of *Lecture Notes in Computer Science*, pages 467–474. Springer Berlin / Heidelberg, 2008. ISBN 978-3-540-79720-3. 70

[2] J.S. Aguilar-Ruiz, J.C. Riquelme, and M. Toro. Evolutionary learning of hierarchical decision rules. *Transactions on Systems and Man and and Cybernetics - Part B : Cybernetics*, 33(2) : 324–331, 2003. 57

[3] J. Alcalá-Fdez, L. Sánchez, S. García, M. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas, J. Fernández, and F. Herrera. Keel : a software tool to assess evolutionary algorithms for data mining problems. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 13 :307–318, 2009. ISSN 1432-7643. 10.1007/s00500-008-0323-y. 56, 95, 104

[4] N. Ash, O. Ogunyemi, Q. Zeng, and L. Ohno-Machado. Finding appropriate clinical trials : evaluating encoded eligibility criteria with incomplete data. *AMIA*, 1067-5027, 2001. 21, 23

[5] A. Reynolds B. de la Iglesia and V. J. Rayward-Smith. Developments on a multi-objective metaheuristic (momh) algorithm for finding interesting sets of classification rules. *Lecture Notes in Computer Science*, 3410 :826–840, 2005. 165

[6] J. Bacardit. *Pittsburgh Genetic-Based Machine Learning in the Data Mining era : Representations, generalization, and run-time*. PhD thesis, Universitat Ramon Llull Barcelona, 2004. 77

[7] J. Bacardit and J.M. Garrell. Evolving multiple discretizations with adaptive intervals for a pittsburgh rule-based learning classifier system. In *Genetic and Evolutionary Computation Conference(GECCO'03)*, volume 2724 of *Lecture Notes on Computer Science*, pages 1818–1831. Lecture Notes on Computer Science, 2003. 56, 57

[8] J. Bacardit, E.K. Burke, and N. Krasnogor. Improving the scalability of rule-based evolutionary learning. *Memetic Computing*, 1 :55–67, 2009. ISSN 1865-9284. doi : 10.1007/s12293-008-0005-4. 56, 57, 95

[9] K.D. Barnard, L. Dent, and A. Cook. A systematic review of models to predict recruitment to multicentre clinical trials. *BMC Medical Research Methodology*, 10(1) :63, 2010. 22, 23

[10] Y. Bejot, J. Fauconnier, E. Benzenine, C. Quantin, M. Giroud, and M. Hommel. Evaluation de la qualité métrologique des données du pmsi concernant les avc et application à la caractérisation et à la modélisation de la prise en charge des avc en court séjour sur le plan national. présentation du phrc national pmsi-avc. In SFNV (Société Française Neuro-Vasculaire), editor, *JOURNÉES DE LA SOCIÉTÉ FRANÇAISE NEURO-VASCULAIRE*, number 15, 2010. 18

[11] G. Bergmann and G. Hommel. *Improvements of general multiple test procedures for redundant systems of hypotheses*, chapter Multiple Hypotheses Testing, page 100–115. Springer, Berlin, 1988. 61

[12] P. Besana, M. Cuggia, O. Zekri, A. Bourde, and A. Burgun. Using semantic web technologies for clinical trial recruitment. *Lecture Notes in Computer Science*, 6497/2010 :34–49, 2010. 22, 23, 24

[13] J. Blaszczyński, R. Slowiński, and M. Szelg. Sequential covering rule induction algorithm for variable consistency rough set approaches. *Inf. Sci.*, 181(5) :987–1002, March 2011. ISSN 0020-0255. 54

[14] J. Błaszczyński, S. Greco, B. Matarazzo, R. Słowiński, and M. Szeląg. *Rough Sets and Intelligent Systems - Professor Zdzisław Pawlak in Memoriam Rough Sets and Intelligent Systems - Professor Zdzisław Pawlak in Memoriam*, volume 1 of *Intelligent Systems Reference Library*, chapter jMAF - Dominance-Based Rough Set Data Analysis Framework, pages 185–209. Springer, 2012. 54, 57

[15] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, California, 1984. 47

[16] H. Bricout, V. Gilleron, C. Barat, A. Kostrzewa, P. Perez, C. Germain, J. Calderon, G. Janvier, and M. Puntous. Evaluation de l'exhaustivité du recrutement des patients dans une étude épidémiologique à partir de la base de données médicales du programme de médicalisation du système d'information (pmsi) : application à l'étude plasmacard. In *1ère conférence francophone d'épidémiologie clinique*, volume Livret I, page 31, 2007. 25

[17] A. Burgun, A. Rosier, L. Temal, J. Jacques, R. Messai, L. Duchemin, L. Deleger, C. Grouin, P. Van Hille, P. Zweigenbaum, R. Beuscart, D. Delerue, O. Dameron, P. Mabo, and C. Henry. Aide à la décision en télécardiologie par une approche basée ontologie et centrée patient. *{IRBM}*, 32(3) :191 – 194, 2011. ISSN 1959-0318. 156

[18] A.J. Butte, D.A. Weinstein, and I.S. Kohane. Enrolling patients into clinical trials faster using realtime recruiting. *AMIA*, 1067-5027, 2000. 10, 21, 23

[19] S. Cahon, N. Melab, and E.-G. Talbi. Paradiseo : A framework for the reusable design of parallel and distributed metaheuristics. *J. Heuristics*, 10(3) :357–380, 2004. 78, 160

[20] M.K. Campbell, C. Snowdon, D. Francis, D. Elbourne, A.M. McDonald, R. Knight, V. Entwistle, J. Garcia, I. Roberts, and A. Grant. Recruitment to randomised trials : strategies for trial enrolment and participation study. the steps study. *Health Technology Assessment*, 11 :48, 2007. 15

[21] E. Cantú-Paz and C. Kamath. Inducing oblique decision trees with evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 7(1) :54–68, 2003. 55, 57

[22] D.R. Carvalho and A.A. Freitas. A hybrid decision tree/genetic algorithm for coping with the problem of small disjuncts in data mining. 2000. 66

[23] D.R. Carvalho and A.A. Freitas. A hybrid decision tree/genetic algorithm method for data mining. *Information Sciences*, 163(1) :13–35, 2004. 54, 57

[24] J. Casillas, P. Martínez, and A. Benítez. Learning consistent, complete and compact sets of fuzzy rules in conjunctive normal form for regression problems. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 13 :451–465, 2009. ISSN 1432-7643. 10.1007/s00500-008-0361-5. 54, 72, 131

[25] N.V. Chawla. Data mining for imbalanced datasets : An overview. *Data Mining and Knowledge Discovery Handbook, 2nd ed.,*, 2010. 45

[26] N.V. Chawla, K.W. Bowyer, L.O. Hall, and W.P. Kegelmeyer. Smote : Synthetic minority oversampling technique. *Journal of Artificial Intelligence Research*, 16 :321–357, 2002. 47, 139

[27] C.A. Coello Coello, C. Dhaenens, and L. Jourdan. *Advances in Multi-Objective Nature Inspired Computing*. Studies in Computational Intelligence. Springer, 2010. 51

[28] W.W. Cohen. Fast effective rule induction. In *Machine Learning : Proceedings of the Twelfth International Conference*, pages 1–10, 1995. 48, 57

[29] D. Corne, C. Dhaenens, and L. Jourdan. Synergies between operations research and data mining : the emerging use of multi-objective approaches. *European Journal of Operational Research*, 221(3) (0) :469–479, 2012. ISSN 0377-2217. doi : 10.1016/j.ejor.2012.03.039. 35

[30] K. Deb and R.A. Raji. Reliable classification of two-class cancer data using evolutionary algorithms. *Biosystems.*, Nov 72(1-2) :111–29, 2003. 77

[31] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm : Nsga-ii, 2000. 54

[32] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7 :1–30, December 2006. ISSN 1532-4435. 56, 62, 109

[33] J. Ding, S. Erdal, T. Borlawsky, J. Liu, D. Golden-Kreutz, J. Kamal, and P.R. Payne. The design of a pre-encounter clinical trial screening tool : Asap. In *AMIA Annu Symp Proc.*, number 931 in 6, Nov 2008. 20, 23

[34] E. M. Dos Santos, R. Sabourin, and P. Maupin. A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. *Pattern Recognition*, 2008. 131

[35] E.M. Dos Santos, R. Sabourin, and P. Maupin. Overfitting cautious selection of classifier ensembles with genetic algorithms. *Inf. Fusion*, 10(2) :150–162, April 2009. ISSN 1566-2535. doi : 10.1016/j.inffus.2008.11.003. 100, 131, 135

[36] O. J. Dunn. Multiple comparisons among means. *Journal of the American Statistical Association*, 56 :52–64, 1961. 61

[37] P.J. Embi, A. Jain, J. Clark, S. Bizjack, R. Hornung, and C.M. Harris. Effect of a clinical trial alert system on physician participation in trial recruitment. *Archives of Internal Medicine*, 165 (19) :2272–2277, 2005. 20, 23

[38] A. Fernández, S. García, J. Luengo, E. Bernadó-Mansilla, and F. Herrera. Genetics-based machine learning for rule induction : Taxonomy, experimental study and state of the art. *IEEE Transactions on Evolutionary Computation*, 2010. 59, 95, 98

[39] J. Fitzgerald, R. M. A. Azad, and C. Ryan. Bootstrapping to reduce bloat and improve generalisation in genetic programming. In *Proceeding of the fifteenth annual conference companion on Genetic and evolutionary computation conference companion*, GECCO '13 Companion, pages 141–142, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1964-5. 151

[40] A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`. 58, 59

[41] A.A. Freitas. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer, 2002. 44

[42] J. Fürnkranz and G. Widmer. Incremental reduced error pruning, 1994. 48

[43] S. Garcìa, F. Herrera, and J. Shawe-taylor. An extension on -statistical comparisons of classifiers over multiple data sets- for all pairwise comparisons. *Journal of Machine Learning Research*, pages 2677–2694, 2008. 57

[44] L. Geng and H.J. Hamilton. Interestingness measures for data mining : A survey. *ACM Computing Surveys (CSUR)*, Volume 38 Issue 3, 2006. 41

[45] J.H. Gennari, D. Sklar, and J. Silva. Cross-tool communication : from protocol authoring to eligibility determination. In *Proc AMIA Symp.*, pages 199–203, 2001. 20, 23

[46] J.H. Gennari, D. Sklar, and J. Silva. Cross-tool communication : from protocol authoring to eligibility determination. *Proceedings of the AMIA Symposium*, pages 199–203, 2001. 22

[47] H. Guo and H.L. Viktor. Learning from imbalanced data sets with boosting and data generation : the databoost-im approach. *SIGKDD Explorations*, 6(1) :30–39, 2004. 49, 57

[48] I. Guyon, C.F. Aliferis, G.F. Cooper, A. Elisseeff, J.-P. Pellet, P. Spirtes, and A.R. Statnikov. Design and analysis of the causation and prediction challenge. *Journal of Machine Learning Research - Proceedings Track*, 3 :1–33, 2008. 59

[49] J. Handl, S.C. Lovell, and J. Knowles. Investigations into the effect of multiobjectivization in protein structure prediction. In *Proceedings of the 10th international conference on Parallel Problem Solving from Nature : PPSN X*, pages 702–711, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-87699-1. doi : 10.1007/978-3-540-87700-4_70. 77

[50] H. He and E.A. Garcia. Learning from imbalanced data. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 21, 2009. 46, 47, 66, 124, 141

[51] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975. 50

[52] S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6 :65–70, 1979. 61

[53] H. Ishibuchi and Y. Nojima. Comparison between fuzzy and interval partitions in evolutionary multiobjective design of rule-based classification systems. In *Fuzzy Systems, 2005. FUZZ '05. The 14th IEEE International Conference on*, pages 430–435, 2005. doi : 10.1109/FUZZY.2005.1452432. 131

[54] H. Ishibuchi and Y. Nojima. Evolutionary multiobjective optimization for the design of fuzzy rule-based ensemble classifiers. *Int. J. Hybrid Intell. Syst.*, 3(3) :129–145, August 2006. ISSN 1448-5869. 54

[55] H. Ishibuchi and T. Yamamoto. Comparison of heuristic criteria for fuzzy rule selection in classification problems. *Fuzzy Optimization and Decision Making*, 3 :119–139, 2004. ISSN 1568-4539. doi : 10.1023/B:FODM.0000022041.98349.12. 131

[56] H. Ishibuchi and T. Yamamoton. Fuzzy rule selection by data mining criteria and genetic algorithms. *Proc. of Genetic and Evolutionary*, 2002. 131

[57] H. Ishibuchi, Y. Nojima, and T. Doi. Comparison between single-objective and multi-objective genetic algorithms : Performance comparison and performance measures. In *Evolutionary Computation, 2006. IEEE Congress on*, pages 1143–1150, 2006. 77

[58] J. Jacques, D. Delerue, L. Jourdan, and C. Dhaenens. Extension des critères d'inclusions dans les essais cliniques à l'aide de méthodes d'optimisation. In *ROADEF 2011 : 12e congrès annuel de la Société française de Recherche Opérationnelle et d'Aide à la Décision*, Saint-Étienne, France, March 2011. 9

[59] J. Jacques, J. Taillard, D. Delerue, L. Jourdan, and C. Dhaenens. The benefits of using multi-objectivization for mining Pittsburgh partial classification rules in imbalanced and discrete data. In *Genetic and Evolutionary Computation Conference (GECCO 2013)*, page to appear, Amsterdam, Pays-Bas, 2013. 65

[60] J. Jacques, J. Taillard, D. Delerue, L. Jourdan, and C. Dhaenens. Moca-i : discovering rules and guiding decision maker in the context of partial classification in large and imbalanced datasets. *Learning and Intelligent OptimizatioN, Lecture Notes in Computer Science (LNCS)*, page (in press), 2013. 84, 139

[61] M. Jensen. Guiding single-objective optimization using multi-objective methods. In S. Cagnoni, C.G. Johnson, JJ.R. Cardalda, E. Marchiori, D.W. Corne, J.-A. Meyer, J. Gottlieb, M. Middendorf, A. Guillot, G.R. Raidl, and E. Hart, editors, *Applications of Evolutionary Computing*, volume 2611 of *Lecture Notes in Computer Science*, pages 268–279. Springer Berlin Heidelberg, 2003. ISBN 978-3-540-00976-4. doi : 10.1007/3-540-36605-9_25. 66, 77

[62] L. Jiao, J. Liu, and W. Zhong. An organizational coevolutionary algorithm for classification. *IEEE Transactions on Evolutionary Computation*, 10(12) :67–80, 2006. 55, 57

[63] T. Jo and N. Japkowicz. Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, 6(1) :40–49, 2004. 46, 47

[64] J. Josse. Factominer : An r package for multivariate analysis. *Journal Of Statistical Software*, 25 (1) :1–18, 2008. 67

[65] J. Kamal, K. Pasuparthi, P. Rogers, J. Buskirk, and H. Mekhjian. Using an information warehouse to screen patients for clinical trials : a prototype. In *AMIA Annu Symp Proc.*, volume 1004, 2005. 20, 23

[66] M. Khabzaoui, C. Dhaenens, A. N'Guessan, and E.-G. Talbi. Etude exploratoire des critères de qualité des règles d'association en datamining. *XXXVèmes journées de statistique*, 2003. 41, 66, 70

[67] J.D. Knowles and D.W. Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evol. Comput.*, 8(2) :149–172, June 2000. ISSN 1063-6560. doi : 10.1162/106365600568167. 52

[68] J.D. Knowles, R.A. Watson, and D. Corne. Reducing local optima in single-objective problems by multi-objectivization. In *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, EMO '01, pages 269–283, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-41745-1. 77

[69] W. H. Kruskal and W. A. Wallis. Use of Ranks in One-Criterion Variance Analysis. *Journal of the American Statistical Association*, 47(260) :583–621, 1952. ISSN 01621459. 61

[70] M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets : One-sided selection. In *In Proceedings of the Fourteenth International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann, 1997. 45

[71] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Mach. Learn.*, 51(2) :181–207, May 2003. ISSN 0885-6125. doi : 10.1023/A:1022859003006. 151

[72] J.-P. Le Floch L. Perlemuter. *Essais thérapeutiques et études cliniques*. 1995. 10

[73] Y. Lee, D. Dinakarpandian, N. Katakam, and D. Owens. Mindtrial : An intelligent system for clinical trials. *AMIA Annu Symp Proc.*, 2010 :442–446, 2010. 21, 23

[74] L. Li, H.S. Chase, C.O. Patel, C. Friedman, and C. Weng. Comparing icd9-encoded diagnoses and nlp-processed discharge summaries for clinical trials pre-screening : a case study. In *AMIA Annu Symp Proc*, page 404–408, 2008. 25

[75] A. Liefooghe, J. Humeau, S. Mesmoudi, L. Jourdan, and E.-G. Talbi. On dominance-based multiobjective local search : design, implementation and experimental analysis on scheduling and traveling salesman problems. *Journal of Heuristics*, pages 1–36, 2011. ISSN 1381-1231. 10.1007/s10732-011-9181-3. 52

[76] E.-G. Talbi M. Khabzaoui, C. Dhaenens. Combining evolutionary algorithms and exact approaches for multi-objective knowledge discovery. *RAIRO-Oper*, 42 :69–83, 2008. 54, 57

[77] H.B. Mann and D.R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, pages 50–60, 1947. 60

[78] K. Narukawa, Y. Nojima, and H. Ishibuchi. Modification of evolutionary multiobjective optimization algorithms for multiobjective design of fuzzy rule-based classification systems. In *Fuzzy Systems, 2005. FUZZ '05. The 14th IEEE International Conference on*, pages 809–814, 2005. doi : 10.1109/FUZZY.2005.1452498. 54

[79] M.R. Nendaz and A. Perrier. Sensibilité, spécificité, valeur prédictive positive et valeur prédictive négative d'un test diagnostique. *Revue des Maladies Respiratoires*, Vol 21, N 2 - avril 2004 :390–393, 2004. 42

[80] L. Ohno-Machado, S. J. Wang, P. Mar, and A. A. Boxwala. Decision support for clinical trial eligibility determination in breast cancer. In *Proc AMIA Symp.*, pages 340–4, 1999. 21, 23

[81] M. Ohsaki, H. Abe, S. Tsumoto, H. Yokoi, and T. Yamaguchi. Evaluation of rule interestingness measures in medical knowledge discovery in databases. *Artificial Intelligence in Medicine*, 41 : 177–196, 2007. 41, 45

[82] G.L. Pappa and A.A. Freitas. Automatically evolving rule induction algorithms tailored to the prediction of postsynaptic activity in proteins. *Intelligent Data Analysis*, 13 :243–259, 2009. 55, 56, 57

[83] L. Paquete, M. Chiarandini, and T. Stützle. Pareto local optimum sets in the biobjective traveling salesman problem : An experimental study. In *METAHEURISTICS FOR MULTIOBJECTIVE OPTIMIZATION, LECTURE*, pages 177–200. Springer Verlag, 2004. 52

[84] R.S. Parpinelli, H.S. Lopes, and A.A. Freitas. Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computation*, 6 :321–332, 2002. 55, 57

[85] C. Patel. Matching patient records to clinical trials using ontologies. *The Semantic Web ISWC 2007 ASWC 2007 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference*, 4825 :816–829, 2010. 21, 23

[86] C. Patel, S. Khan, and S. Khan. Trialx : Using semantic technologies to match patients to relevant clinical trials based on their personal health records. *Web Semantics Science Services and Agents on the World Wide Web*, 8(4) :342–347, 2010. 21, 23

[87] L. Penberthy, R. Brown, F. Puma, and B. Dahman. Automated matching software for clinical trials eligibility : Measuring efficiency and flexibility. *Contemporary Clinical Trials*, 31 :207–217, 2010. 10, 21, 23

[88] T. Perneger and A. Perrier. Analyse d'un test diagnostique : courbe roc, ou « receiver operating characteristic ». *Revue des Maladies Respiratoires*, Vol 21, N 2 - avril 2004 :398–401, 2004. 44, 139

[89] J.C. Platt. Advances in kernel methods. chapter Fast training of support vector machines using sequential minimal optimization, pages 185–208. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-19416-3. 58, 59

[90] J. R. Quinlan. *C4.5 : programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993. ISBN 1-55860-238-0. 47

BIBLIOGRAPHIE

[91] A. Reynolds and B. de la Iglesia. Rule induction using multi-objective metaheuristics : Encouraging rule diversity. *IEEE World Congress on Computational Intelligence*, pages 6375–6382, 2006. 54, 57

[92] A. Reynolds and B. De La Iglesia. Rule induction for classification using multi-objective genetic programming. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization*, volume 4403 of *Lecture Notes in Computer Science*, pages 516–530. Springer Berlin / Heidelberg, 2007. ISBN 978-3-540-70927-5. 53, 54, 57, 131

[93] R.L. Richesson, H.S. Lee, D. Cuthbertson, J. Lloyd, K. Young, and J.P. Krischer. An automated communication system in a contact registry for persons with rare diseases : Scalable tools for identifying and recruiting clinical research participants. *Contemp Clin Trials.*, 30(1) :55–62, January 2009. 21, 23

[94] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5) :465 – 471, 1978. ISSN 0005-1098. doi : 10.1016/0005-1098(78)90005-5. 71

[95] D.L. Rubin, J.H. Gennari, S. Srinivas, A. Yuen, H. Kaizer, M.A. Musen, and J.S. Silva. Tool support for authoring eligibility criteria for cancer trials. *Proceedings of the AMIA Symposium*, pages 369–373, 1999. 22, 23

[96] B. Seroussi, J. Bouaud, E.-C. Antoine, L. Zelek, and M. Spielmann. Using oncodoc as a computer-based eligibility screening system to improve accrual onto breast cancer clinical trials. *AIME 2001, LNAI 2101*, page 421–430, 2001. 22, 23, 24

[97] R. Slowinski, S. Greco, and B. Matarazzo. Rough sets in decision making. In *Encyclopedia of Complexity and Systems Science*, pages 7753–7787. 2009. 54

[98] S. Srinivasan and S. Ramakrishnan. Evolutionary multi objective optimization for rule mining : a review. *Artificial Intelligence Review*, pages 1–44. ISSN 0269-2821. 10.1007/s10462-011-9212-3. 35, 40, 54

[99] R. Stanley, K.A. Lillis, S.J. Zuspan, R. Lichenstein, R.M. Ruddy, M.J. Gerardi, J.M. Dean, and the Pediatric Emergency Care Applied Research Network (PECARN). Development and implementation of a performance measure tool in an academic pediatric research network. *Contemporary Clinical Trials*, 31 :429–437, 2010. 22

[100] A.J. Stell, R.O. Sinnott, and O. Ajayi. *Supporting the clinical trial recruitment process through the grid*. National e-Science Centre, 2010. 23

[101] Y. Sun, M.S. Kamel, A.K.C. Wong, and Y. Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12) :3358 – 3378, 2007. ISSN 0031-3203. 49, 57

[102] E.-G. Talbi. *Metaheuristics : From Design to Implementation*. Wiley, 2009. 49

[103] K.C. Tan, Q. Yu, and J.H. Ang. A coevolutionary algorithm for rules discovery in data mining. *International Journal of Systems Science*, 37(12) :835–864, 2006. 55, 57

[104] A. Taylor, A. Castle, J.G. Merino, A. Hsia, C.S. Kidwell, and S. Warach. Optimizing stroke clinical trial design : estimating the proportion of eligible patients. *Stroke*, 41(10) :2236–8, 2010. 22, 23, 27

[105] S.R. Thadani, C. Weng, J.T. Bigger, J.F. Ennever, and D. Wajngurt. Electronic screening improves efficiency in clinical trial recruitment. *Journal of the American Medical Informatics Association*, 16(6) :869–873, 2009. 20, 23

[106] K.M. Ting. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering*, 14(3) :659–665, 2002. 57

[107] S. Treweek, E. Pearson, N. Smith, R. Neville, P. Sargeant, B. Boswell, and F. Sullivan. Desktop software to identify patients eligible for recruitment into a clinical trial : using sarma to recruit to the road feasibility trial. *Informatics in Primary Care*, 18(1) :51–58, 2010. 20, 23

[108] G. Venturini. Sia : A supervised inductive algorithm with genetic search for learning attributes based concepts. In *6th European Conference on Machine Learning(ECML93)*, volume 667 of *Lecture Notes on Computer Science*, pages 280–296. Lecture Notes in Artificial Intelligence, 1993. 54, 57

[109] H. Wang, S. Kwong, Y. Jin, W. Wei, and K.-F. Man. Agent-based evolutionary approach for interpretable rule-based knowledge extraction. *Systems, Man, and Cybernetics, Part C : Applications and Reviews, IEEE Transactions on*, 35(2) :143–155, 2005. ISSN 1094-6977. doi : 10.1109/TSMCC.2004.841910. 54

[110] S. Watanabe and K. Sakakibara. A multiobjectivization approach for vehicle routing problems. In *Proceedings of the 4th international conference on Evolutionary multi-criterion optimization*, EMO'07, pages 660–672, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 978-3-540-70927-5. 77

[111] D.L. Weiner, A.J. Butte, P.L. Hibberd, and G.R. Fleisher. Computerized recruiting for clinical trials in real time. In *Ann Emerg Med*, volume 42(5), pages 712–3, 2003. 10, 21, 23

[112] G.M. Weiss. *Timeweaver : a Genetic Algorithm for Identifying Predictive Patterns in Sequences of Events*, volume 1, pages 718–725. Morgan Kaufmann, 1999. 66

[113] C. Weng, J.T. Bigger, L. Busacca, A. Wilcox, and A. Getaneh. Comparing the effectiveness of a clinical registry and a clinical data warehouse for supporting clinical trial recruitment : A case study. *AMIA Annu Symp Proc.*, 2010 :867–871, 2010. 20, 23

[114] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics*, 1 :80–83, 1945. 62

[115] S.W. Wilson. Classifier fitness based on accuracy. *Evol. Comput.*, 3(2) :149–175, June 1995. ISSN 1063-6560. doi : 10.1162/evco.1995.3.2.149. 56, 57

[116] C. Yeon-Jin, K. Hyeoncheol, and OH Heung-bum. Generating rules for predicting mhc class i binding peptide using ann and knowledge-based ga. *JDCTA : International Journal of Digital Content Technology and its Applications*, 3 :pp. 111–119, 2009. 55, 57

[117] I. Yoo, P. Alafaireet, M. Marinov, K. Pena-Hernandez, R. Gopidi, J.-F. Chang, and L. Hua. Data mining in healthcare and biomedicine : A survey of the literature. *Journal of Medical Systems*, 36 :2431–2448, 2012. ISSN 0148-5598. 36

[118] J. Zhang, J.W. Bala, A. Hadjarian, and B. Han. Learning to rank cases with classification rules. *Preference Learning*, pages 155–177, 2011. 164

Annexes

A Normes et formats des données médicales

A.1 Classification Commune des Actes médicaux - CCAM

La *classification commune des actes médicaux* ou *CCAM* est une classification française utilisée à la fois pour le calcul d'honoraires et pour la T2A par l'intermédiaire du PMSI. Elle recense l'ensemble des actes médicaux réalisables par le personnel de santé : radiographies, échographies, prise de sang, chirurgie,... Ce qui représente 12769 actes au total. Cette classification est hiérarchisée : chaque acte appartient à plusieurs chapitres plus ou moins spécialisés. Ainsi, l'appendicectomie est représentée sous la hiérarchie suivante :

7 APPAREIL DIGESTIF

7.3 ACTES THÉRAPEUTIQUES SUR LE TUBE DIGESTIF

7.3.5 Actes thérapeutiques sur l'appendice vermiforme [appendice]

HHFA001 Appendicectomie, par abord de la fosse iliaque

A.1.1 CIM10 / ICD-9

La *Classification internationale des maladies* est éditée par l'Organisation Mondiale de la Santé (OMS). Elle est actuellement dans sa version 10 et permet le codage des maladies et causes de recours aux services de santé, soit 19020 éléments. Elle est commune à plusieurs pays, elle est notamment utilisée aux États-Unis sous le nom d'ICD-10. Les éléments qu'elle contient sont codés sur 3 caractères, voire parfois un quatrième pour apporter des précisions. La CIM10 est divisée en 22 chapitres, l'ICD-10 possède une hiérarchisation fournie par l'OMS qui consiste en des ensembles de codes. En voici un exemple :

K00-K93 Diseases of the digestive system

K35-K38 Diseases of appendix

K35 Acute appendicitis

K35.9 Acute appendicitis, unspecified

A.2 Classification ATC des médicaments

Le *Système de Classification Anatomique, Thérapeutique et Chimique (ATC)* des médicaments est réalisé par l'OMS. Certains médicaments y sont codés plusieurs fois, l'indication de la prescription étant nécessaire pour connaitre le code à utiliser : selon que l'aspirine soit utilisée pour son effet anticoagulant ou anti-inflammatoire, elle aura un code différent. Il existe 5 manières différentes de la coder. Cela peut induire des incertitudes sur les données patient car les logiciels ne disposent pas toujours des informations nécessaires pour choisir le bon code ATC. Chaque code est composé de 7 caractères, chaque sous-ensemble de caractères correspond à une sous-famille. Par exemple, voici la hiérarchie pour une des représentations de l'aspirine :

N NERVOUS SYSTEM

N02 ANALGESICS

N02B OTHER ANALGESICS AND ANTIPYRETICS

N02BA Salicylic acid and derivatives

N02BA01 acetylsalicylic acid

A.3 Données HL7

Le format HL7 est utilisé pour l'interopérabilité entre logiciels informatiques de santé. Il encadre l'échange de données médicales comme les comptes-rendus, images et résultats d'examens ou mesures biologiques.

B Détail des jeux de données étudiés

B.1 Jeux de données classiques

adult A partir de données de recensement, l'objectif est de prédire si une personne gagne plus de 50k / an.

australian A partir de dossiers de demandes de cartes de crédit, déterminer si une personne peut ou non obtenir une carte.

breast A partir de données médicales, il s'agit de prédire si un cancer du sein va provoquer des métastases.

crx Similaire au jeu de données *australian* mais comprend un attribut supplémentaire

german L'objectif est de déterminer si une personne a un dossier compatible avec l'ouverture d'un crédit ou non

heart Il s'agit d'évaluer si un patient souffre d'une pathologie cardiaque.

hepatisis A partir de données médicales, il s'agit de prédire si un patient souffre d'hépatite ou non.

horse colic Contient 30% de valeurs manquantes. A partir des dossiers de chevaux souffrant de colique, il s'agit de déterminer s'il y a des lésions nécessitant une intervention chirurgicale.

housevotes A partir des avis des personnes sondées sur différents sujets, l'objectif est de prédire si elles vont voter démocrate ou républicain.

mushrooms L'objectif est de prédire si un champignon est vénéneux ou comestible, à partir de ses caractéristiques. Il s'agit du jeu UCI "'mushrooms"' qui a été binarisé.

B.2 Jeux de données asymétriques

a1a Il s'agit du jeu de données *adult* binarisé. L'objectif est de prédire le revenu d'une personne en fonction des données de recensement.

abalone19 L'abalone est un petit mollusque, l'objectif est de prédire sa catégorie d'âge en fonction de ses caractéristiques (ici âge = 19)

abalone9vs18 Idem, mais seules les données avec l'âge = 9 et âge = 18 ont été conservées. On souhaite prédire si l'abalone a un âge de 9

ecoli1 Localisation des sites de liaison des protéines, pour un type de protéine donné

ecoli2 Localisation des sites de liaison des protéines, pour un autre type de protéine

haberman Prédire la survie de patients après une opération chirurgicale du cancer.

lucap0

w1a

yeast2vs8 Localisation des sites de liaison des protéines, pour un type de protéine donné

yeast3 Localisation des sites de liaison des protéines, pour un autre type de protéine