UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN MATHÉMATIQUES ET INFORMATIQUE
APPLIQUÉES

PAR
EMMANUEL TSHIBALA TSHITOKO

# PRÉDICTION DES EFFORTS DE TEST : UNE APPROCHE BASÉE SUR LES SEUILS DES MÉTRIQUES LOGICIELLES ET LES ALGORITHMES D'APPRENTISSAGE AUTOMATIQUE

Décembre 2019

# Résumé

De nos jours, les systèmes logiciels doivent être de bonne qualité, et dans certains cas, sans fautes. La prédiction des efforts de test est un domaine important du génie logiciel. Elle peut être utilisée par les développeurs et les testeurs pour prioriser les tests permettant ainsi une meilleure allocation des ressources en réduisant le temps et les coûts des tests. Cette allocation met l'accent sur les composants les plus critiques, et par conséquent permet d'améliorer le processus de test de manière significative et d'assurer un produit de qualité. Il est donc important de prédire, le plus tôt possible dans le processus de développement, l'effort requis pour tester les différents composants d'un logiciel.

Dans cette étude, nous explorons les différentes techniques de calcul de valeurs seuils qui peuvent être utilisées dans la prédiction des efforts de test : les courbes ROC et Alves Rankings. Nous avons aussi exploré l'approche basée sur les valeurs brutes des métriques logicielles. Nous avons comparé les performances de ces techniques pour savoir laquelle est la meilleure. Il était aussi question de savoir comment utiliser les algorithmes d'apprentissage automatique pour améliorer la performance des techniques de calcul des seuils.

Nous avons calculé les valeurs seuils pour un ensemble de huit systèmes logiciels différents. Nous avons construit des modèles de prédiction des efforts de test pour chaque technique et aussi pour les valeurs brutes des métriques. Pour les comparer, ces techniques ont été combinées aux différents algorithmes d'apprentissage automatiques retenus dans cette étude.

Les résultats montrent que l'approche basée sur les courbes ROC est celle qui donne les meilleures performances parmi les trois approches étudiées, suivie d'Alves Rankings. L'approche basée sur les valeurs brutes des métriques est relativement la moins performante. Par ailleurs, la combinaison des algorithmes d'apprentissage automatique aux techniques de calcul des seuils améliore significativement la performance de ces dernières.

En se basant sur nos résultats, on peut déduire que la méthode des courbes ROC est celle qui donne de meilleures performances. Cependant, Alves Rankings est aussi une bonne méthode pour le calcul de valeurs seuils. L'utilité de l'application d'Alves Rankings est à considérer parce que la méthode d'Alves Rankings est non supervisée, c'est-à-dire son point positif réside dans le fait qu'elle ne requiert pas des données relatives à l'effort de test.

# Abstract

Nowadays, software systems must be of good quality, and in some cases, without faults. The prediction of testing efforts is an important area of software engineering. It can be used by developers and testers to prioritize testing thereby allowing better allocation of resources by reducing testing time and costs. This allocation focuses on the most critical components, thereby improving the testing process significantly and ensuring a quality product. It is therefore important to predict, as early as possible in the development process, the effort required to test the various components of a software system.

In this study, we explore the different techniques for calculating threshold values that can be used in the prediction of the testing efforts : ROC curves and Alves Rankings. We also explored the approach based on the raw values of software metrics. We compared the performance of these techniques to find out which one is the best. It was also about how to use machine learning algorithms to improve the performance of threshold calculation techniques.

We calculated the threshold values for a set of eight different software systems. We built prediction models for testing efforts prediction for each technique and for the raw values of the metrics. To compare them, these techniques were combined with the different machine learning algorithms used in this study.

The results show that the ROC curve approach is the one that gives the best performance among the three approaches studied, followed by Alves Rankings. The approach based on the raw values of the metrics is relatively the least efficient. Besides, the combination of machine learning algorithms and threshold calculation techniques significantly improves the performance of these thresholds.

Based on our results, we can deduce that the ROC curve method is the one that gives better performance. However, Alves Rankings is also a good method for calculating threshold values. The usefulness of the Alves Rankings application is to be considered because the Alves Rankings method is unsupervised, that is, its positive point is that it does not require data related to the testing efforts.

# Remerciements

Je rends en premier lieu gloire à Dieu, maître des temps et des circonstances pour sa grâce dans ma vie.

À mes directeurs de recherche Mourad et Linda BADRI, qui m'ont fait confiance et surtout pour leur encadrement combien enrichissant et perspicace, trouver ici l'expression de mon profond respect et de ma profonde gratitude.

Je tiens aussi à rendre hommage à mes parents Stéphane TSHITOKO PEMBE (d'heureuse mémoire) et Astrid KASENGELA MUKENDI qui nous ont toujours encouragés à étudier et pour tous leurs bienfaits. Je remercie également les membres de ma famille pour leurs prières, leurs encouragements et leur soutien indéfectible. Particulièrement à mes grandes sœurs Ritha NCIEYI, Rachel MASHANGA et Stéphanie MENHONGO pour toute forme d'assistance combien louable et substantielle. À toi aussi, notre bien aimée Florence MBUYI qui nous a quitté très tôt, je te dédie ce travail.

J'adresse mes sincères remerciements à mes proches ici au CANADA, qui m'ont chaleureusement accueilli. À tous mes amis, merci pour les moments de joie et de bonheur. À tous mes collègues, merci pour les échanges fructueux.

En définitive, à toute personne m'ayant apporté son aide d'une manière ou d'une autre pour l'accomplissement du présent travail, qu'elle trouve ici l'expression de ma reconnaissance.

# Table Des Matières

# Liste des Tables

# Liste des Figures

*Chapitre. 1*

# *INTRODUCTION*

## 1.1 Introduction

En matière d'assurance qualité, les tests constituent une étape cruciale dans le processus de développement des logiciels [1]. L'évolution croissante et rapide des logiciels, leur complexité, et la gestion de leur qualité constituent une tâche aussi bien importante que coûteuse en termes de temps et de ressources [1]. Il s'avère important de prédire, le plus tôt possible dans le processus de développement, l'effort requis pour tester un logiciel. Ceci permettra une bonne planification des activités de test et une gestion optimale des ressources, tout en soulignant le fait que la qualité du processus détermine en partie la qualité du produit [2]. Malheureusement, très peu de travaux ont été effectués sur la prédiction des efforts de test, particulièrement en comparaison avec les nombreux travaux effectués sur la prédiction des efforts de développement.

Malheureusement, il n'y a que quelques études réalisées sur l'association entre les métriques orientées objet et l'effort de test. Aussi, et au mieux de nos connaissances, cette question cruciale n'a jamais été explorée sous la perspective de valeurs seuils de métriques et leur effet sur leurs capacités de prédiction de l'effort de test. Plusieurs des métriques logicielles, par exemple les métriques de Chidamber et Kemerer (CK) [3] qui sont les plus connues et utilisées, ont été validées (entre autres) comme indicateurs de classes fautives.

D'où la nécessité de savoir utiliser de manière efficace et efficiente ces métriques pour estimer l'effort de test des logiciels. Les techniques d'apprentissage automatique et des statistiques ont permis de démontrer l'étroite relation existant entre les métriques CK et les fautes [4–12]. Cette relation entre métriques logicielles et les fautes logicielles illustrent implicitement (pour nous) la relation métrique logicielle et effort de tests. Ainsi, en se basant sur ce résultat, il est possible d'ouvrir une brèche vers une approche basée sur les seuils (critiques) des métriques. L'apprentissage automatique et les techniques de statistique sont rigoureux, et les outils dédiés aux logiciels sont adéquats pour analyser cette relation. Identifier les classes sujettes aux fautes est nécessaire pour guider les

testeurs des logiciels afin d'améliorer leur performance et de diminuer les coûts des activités relatives aux tests et à la maintenance [5].

Pour évaluer la contribution des métriques et la mesure pour l'aide à la décision, il est essentiel de définir des *valeurs seuils* significatives [13]. Ces valeurs relatives aux différentes métriques retenues permettront de fixer des lignes directrices quant à la complexité de la conception ou à l'effort à fournir lors des tests. Cela étant fait dans le but ultime d'identifier les classes les plus critiques nécessitant le plus d'efforts de test.

Plusieurs questions de recherche restent posées dans ce domaine, incluant : comment déterminer de façon objective des seuils permettant de prédire l'effort de test ? Comment utiliser les techniques d'apprentissage pour améliorer leurs performances ? Par ailleurs, l'effort de test est une notion complexe qui peut être affectée par plusieurs facteurs. Dans ce travail, il s'agira d'appliquer quelques techniques non supervisées et supervisées de calcul des seuils. Il s'agira également d'explorer la combinaison de ces techniques avec plusieurs algorithmes d'apprentissage automatique.

## 1.2 Problématique

Le modèle basé sur l'effet des valeurs seuils des métriques estime les classes comme sujettes aux fautes quand les valeurs de leurs métriques dépassent certains seuils [14, 15]. Par ailleurs, il faut déterminer les niveaux des seuils pour les différentes métriques des différents systèmes [15]. Pour déterminer ces seuils, quelques techniques de calcul des seuils qui seront associées à certains algorithmes d'apprentissage automatique seront explorées [16–19]. Les buts étant de déterminer à partir de quelles techniques seront obtenues les meilleures performances de prédiction. Les modèles de prédiction non supervisés basés sur les valeurs seuils produisent-ils des performances similaires à ceux qui sont supervisés ? Lorsqu'ils sont associés à un modèle basé sur l'apprentissage automatique, les modèles basés sur les seuils atteignent-ils de meilleures performances ?

Plusieurs approches pour identifier les niveaux des seuils ont été proposées dans la littérature [20]. Pour l'identification des seuils relatifs aux métriques, Ferreira et *al.* [21] ont utilisé l'outil *EasyFit* [22] pour trouver la distribution la plus similaire à la distribution de la métrique à partir des coupes des percentiles. Les seuils ont été catégorisés comme bons, modérés et mauvais. Foucalt et *al.* [23] ainsi qu'Alves et *al.* [13] ont aussi utilisé les coupes des percentiles pour calculer les seuils. Foucault et *al.* [23] ont utilisé une analyse statistique appelée *bootstrap* [24] pour estimer l'intervalle de confiance du percentile.

La plupart de ces différentes expérimentations ont été effectuées pour la prédiction des fautes.

Il est aussi important de souligner que Shatnawi et *al.* [16] ont utilisé les courbes ROC (Receiver Operating Characteristic) d'une part pour associer les métriques orientées objet aux différentes catégories de sévérité de fautes dans les systèmes logiciels. D'autre

part, pour trouver des valeurs seuils qui donnent la meilleure justification pour ces catégories d'erreur [20].

Le but de notre travail est de préconiser une approche axée sur deux principales techniques de calcul de seuil. À savoir, la méthode des *courbes ROC* et *Alves Rankings* [15, 19] qui seront associées éventuellement aux algorithmes d'apprentissage automatique afin d'obtenir de meilleures performances dans la prédiction de l'effort de test.

## 1.3 Organisation du Mémoire

Ce travail est subdivisé en cinq principaux chapitres. Le premier chapitre est consacré à l'introduction générale portant sur la prédiction des efforts de test logiciel ainsi qu'à certains sujets connexes. Cette mise en contexte est suivie de l'exposé de nos motivations et objectifs poursuivis dans ce travail de recherche.

Le deuxième chapitre est un condensé de l'état de l'art en assurance qualité logicielle en relation avec ce que nous nous sommes fixé comme objectifs. Nous débutons par une brève introduction suivie d'une présentation des éléments essentiels en lien avec notre étude. Il s'agit précisément des métriques orientées objet. Cette étape est suivie de la présentation des solutions existantes en la matière, leurs apports et limitations, en passant par les orientations que nous avons fixées, basées sur les seuils des métriques. Nous terminons ce chapitre par une conclusion.

Le troisième chapitre est quant à lui consacré à la méthodologie de recherche adoptée pour la conduite de nos travaux de recherche. D'entrée de jeu, une introduction est présentée, par la suite des questions de recherche sont soulevées afin de conduire notre démarche. Nous poursuivons par la présentation des données qui sont au centre de notre étude. Enfin, nous présentons les algorithmes d'apprentissage automatique utilisés dans l'accomplissement de nos objectifs, quelques techniques de calcul de seuils, ainsi que les méthodes d'évaluation des performances.

Le quatrième chapitre met en exergue nos expérimentations et les résultats associés. Pour chaque objectif fixé dans nos questions de recherche, une expérimentation, des résultats ainsi qu'une discussion y sont associés. Il est important de souligner que la plupart des résultats sont présentés en annexe de ce mémoire.

Le cinquième et dernier chapitre de ce travail met en exergue les menaces de validité de notre étude, et s'achève par une conclusion et des perspectives.

*Chapitre. 2*

# *ÉTAT DE L'ART : un Aperçu*

## 2.1 Introduction

Dans ce chapitre, nous donnons un aperçu des différentes approches discutées dans la littérature ayant trait à la qualité logicielle, notamment à l'effort de test. Le nombre des travaux effectués dans ce domaine est considérable, mais il est important de souligner les progrès apportés jusqu'alors. Ces progrès peuvent être en matière d'approches, c'est-à-dire les différentes techniques utilisées pour le calcul de seuils. On peut aussi mentionner des progrès en matière de disponibilité des données avec une grande accessibilité, et des outils d'extraction des métriques logicielles.

Les métriques logicielles sont utilisées comme un instrument de mesure et de contrôle dans le processus de développement et de maintenance des logiciels [20].

Par exemple, Chidamber et Kemerer [3] ont proposé un ensemble de métriques orientées objet largement utilisées par les chercheurs. Ces métriques sont également utilisées dans notre étude et sont présentées dans la section suivante.

## 2.2 Métriques Orientées Objet

La gestion de la qualité est un ensemble d'activités coordonnées pour orienter et contrôler un organisme en matière de qualité et objectif qualité [2]. Pour y parvenir, il est utile de se servir de tout type de mesure concernant un logiciel. Ce type de mesure permet de quantifier le logiciel ou le processus, de prédire les caractéristiques d'un produit ou d'un processus logiciel, ou bien encore de faire des prédictions générales ou d'identifier des composants sujets à des comportements anormaux.

Les métriques orientées objet renferment certains principaux objectifs, notamment en matière de qualité de produit. Elles servent à sa compréhension, son évaluation, ainsi qu'à sa maîtrise. Elles contribuent également à l'évaluation de l'efficacité du processus et à l'amélioration de la qualité du travail à différents niveaux. Il est utile de souligner

l'usage important des métriques orientées objet dans la littérature, en particulier dans le contexte du calcul de seuils. Plusieurs approches ont été abordées dans ce sens [13, 16, 17, 21]. Ces métriques orientées objet, proposées par Chidamber et Kemerer [3], ont un grand mérite d'avoir fait l'objet de plusieurs expérimentations au cours desquelles elles ont été validées d'une manière empirique [8, 12, 16, 25, 26]. Une bonne partie de ces expérimentations a porté sur la prédiction des fautes. C'est dans cet ordre d'idées que nous présentons ces métriques qui nous serviront de support tout au long de ce travail. Elles sont subdivisées comme suit :

### 2.2.1 Métriques de complexité

*WMC (Weighted Methods per Class) :* Méthodes pondérées par classe, se réfère à la somme des complexités (cyclomatique) des méthodes d'une classe donnée. Plus la valeur est élevée, plus la classe est complexe [20].

### 2.2.2 Métriques d'héritage

*DIT (Depth of Inheritance Hierarchy) :* Profondeur de l'arbre d'héritage, cette métrique calcule la profondeur de l'arbre d'héritage (c'est-à-dire la distance entre une classe et sa classe racine). La métrique DIT indique que plus une classe est profonde dans une hiérarchie de classes, plus il y a des méthodes qui sont héritées. Ainsi, la classe devient plus complexe et peut présenter un risque important en termes d'erreurs [20]. Il est utile de disposer d'une mesure pour mieux quantifier l'importance de cette classe dans la hiérarchie afin que la classe puisse être bien conçue avec la réutilisation des méthodes héritées [27].

*NOC (Number of Child Classes) :* Nombre d'enfants, se réfère au nombre des sous-classes immédiates subordonnées à une classe dans la hiérarchie de classes [27].

Ces informations donnent une idée de l'importance et de l'impact de la classe pour le projet. Une valeur élevée pour cette métrique peut correspondre à des abstractions inappropriées ou à des erreurs liées aux concepts d'héritage [20].

### 2.2.3 Métriques de taille

*LOC (Lines Of Code) :* Nombre des lignes de code. Cette métrique compte le nombre de lignes d'instructions dans la classe mesurée. LOC évalue la taille d'une classe. La taille est fortement liée à la complexité. Une classe de trop grande taille est souvent synonyme de responsabilités disparates et de forte probabilité de présence des fautes, ce qui suggère que ces classes doivent être restructurées [14].

### 2.2.4 Métriques de couplage

*CBO (Coupling Between Objects) :* Couplage entre objets des classes. Le couplage d'une classe est caractérisé par le nombre de relations que possède une classe. Un couplage élevé est synonyme d'une faible possibilité de réutilisation et une augmentation de la sensibilité aux changements [20]. En plus de cette description, nous pouvons aussi ajouter ces détails liés à cette métrique :

- CBO pour une classe est un décompte du nombre de couples non liés à l'héritage avec d'autres classes [27] ;
- Deux classes s'allient si et seulement si au moins une d'entre elles «agit» sur l'autre [27] ;
- Toute preuve d'une méthode d'un objet utilisant des méthodes ou des variables d'instance d'un autre objet constitue un couplage [27].

### 2.2.5 Métriques de classe

*RFC (Response For Class) :* Réponse pour une classe, fait référence au nombre de méthodes pouvant être exécutées par une instance de classe en réponse à un événement ou à un message reçu. Plus la valeur est élevée, plus la complexité de ses tests et de sa maintenance est grande [20].

C'est une métrique de complexité et de couplage à la fois. Elle est définie par le nombre de méthodes qui peuvent être appelées par une instance d'une classe [28].

### 2.2.6 Métriques de cohésion

*LCOM (Lack of Cohesion in Methods) :* Manque de cohésion dans les méthodes, fait référence au nombre de paires de méthodes d'une classe particulière dans laquelle la similarité est égale à zéro moins le nombre de paires de méthodes dans lesquelles la similarité est non nulle. La similarité est calculée par l'utilisation commune des variables d'une instance de classe. Ainsi, une valeur élevée signifie que la classe n'est pas cohésive [20].

## 2.3 Relation entre métriques de code et effort de test

Les quelques travaux publiés sur le sujet dans la littérature démontrent le lien entre les métriques logicielles et le test logiciel [7, 29–32].

Les métriques logicielles existent depuis l'aube du génie logiciel. Les métriques sont utilisées comme instrument de contrôle dans le processus de développement et de

maintenance du logiciel. Par exemple, les métriques ont été proposées pour identifier les emplacements problématiques dans le code source pour mieux effectuer la maintenance [13]. La qualité du produit dépend (en partie) du processus de développement, il est donc évident que les tests logiciels sont une étape importante dans la réalisation des logiciels de grande envergure. Plus le logiciel est grand, plus les efforts à consacrer pour les tests sont importants. De ce fait, la gestion d'allocation des ressources s'avère une tâche aussi lourde qu'importante.

Comme mentionné dans les paragraphes précédents, plusieurs travaux ont été réalisés sur la prédiction des fautes logicielles et la prédiction des efforts de test. Mais, au mieux de nos connaissances, l'aspect des valeurs seuils des métriques et leur effet sur leurs capacités de prédiction de l'effort de test n'ont pas été explorés. L'effort à fournir pour tester un logiciel implique d'une manière ou d'une autre l'effort à fournir pour rendre les composants d'un logiciel de bonne qualité.

Cette comparaison nous pousse à penser qu'il existe une relation entre les métriques logicielles et l'effort de test.

Dans ce contexte, nous présentons certains travaux effectués dans ce sens, du moins dans le sens de la qualité logicielle.

- **Satwinder Singh et K. S. Kahlon :** Selon ces auteurs, les métriques logicielles jouent un rôle important dans la maintenance et les tests des logiciels. Plusieurs recherches ont démontré que les modèles basés sur les métriques logicielles sont efficaces pour l'analyse de la qualité des logiciels.

  Pour ces auteurs, la seule valeur d'une métrique n'est pas suffisante, il faut aussi tenir compte de sa valeur seuil [33].

- **Ruchika Malhotra et Ankita Jain Bansal :** Pour ces auteurs, les métriques logicielles ont été utilisées pour construire des modèles de prédiction afin d'identifier les composants des logiciels ayant une forte probabilité d'occurrence de fautes. Ils ont considéré l'effet des valeurs seuils des métriques orientées objet sur la propagation des fautes et ont construit des modèles de prédiction basés sur les valeurs seuils des métriques utilisées [26].

- **Raed Shatnawi, WeiLi, James Swain et Tim Newman :** Ils ont fait une étude empirique basée sur une relation entre les métriques orientées objet et les catégories de sévérité des erreurs. Ils se sont focalisés sur l'identification des valeurs seuils des métriques logicielles en utilisant les courbes ROC [16].

- **Tiago L. Alves, Chrsitian Ypma et Joost Visser :** Ils ont souligné le fait que l'usage effectif des métriques orientées objet est entravé par le manque de seuils significatifs. Selon eux, les valeurs seuils ont été proposées en se basant sur l'opinion d'experts et un petit nombre d'observations. D'où la conception d'une méthode permettant de déterminer empiriquement les seuils de mesure à partir de

données de mesure. Les données pour la mesure des différents systèmes logiciels sont regroupées et agrégées. Par la suite, des seuils sont sélectionnés qui (i) font apparaître la variabilité de la métrique entre les systèmes et (ii) permettent de se concentrer sur un pourcentage raisonnable du volume de code source. Leur méthode respecte les distributions et les échelles des métriques de code source et résiste aux valeurs aberrantes en ce qui concerne les métriques ou la taille du système [13, 19, 34].

En définitive, tout au long de ce travail, nous tenterons de démontrer le lien effectif entre les métriques orientées objet et l'effort de test et mettre en évidence leur apport dans ce contexte.

## 2.4 Solutions existantes

Selon Ian Sommerville [2], la répartition de l'effort de maintenance est comme suit : 65% pour l'ajout et modification des fonctionnalités, 18% pour l'adaptation environnementale et 17% pour la réparation des fautes. Cet état des choses illustre la complexité liée à la maintenance des logiciels en perpétuelle évolution. La maintenance pèse énormément sur les entreprises sur le plan des coûts, qu'ils soient financiers, humains ou temporels. Face à cette difficulté, diverses solutions sont proposées, notamment la dérivation des valeurs seuils sur des données de référence, la détermination des métriques appropriées selon l'opinion des experts et sur un nombre insignifiant d'observations.

Ci-dessous, nous donnons quelques éléments utiles traités dans la littérature :

### 2.4.1 Seuils des métriques

- Raed Shatnawi a mené une étude pour trouver les valeurs seuils des métriques en utilisant les courbes ROC. Cette étude empirique consistait à faire le lien entre les métriques orientées objet et les catégories de sévérité d'erreurs [16]. Le but étant d'identifier les valeurs seuils des métriques logicielles de trois versions du projet Eclipse. Il a trouvé les valeurs seuils de certaines métriques (CBO, RFC, CTM, WMC et NOO) qui séparent les classes qui n'ont pas d'erreurs des classes comportant des erreurs à fort impact. Bien que ces seuils ne prédisent pas si une classe comportera certainement des erreurs à l'avenir, ils permettent cependant de suivre une méthode plus scientifique pour évaluer les classes sujettes aux fautes et peuvent être utilisés facilement par les ingénieurs.

- A. Boucher et M. Badri ont quant à eux travaillé sur l'utilisation des valeurs seuils pour prédire les classes sujettes aux fautes [18]. Plusieurs techniques de calcul des seuils des métriques existent. Pour les deux auteurs, le choix a porté sur deux techniques spécifiques, à savoir les courbes ROC et Alves Rankings. Ces deux techniques fournissent les seuils des métriques qui sont pratiques pour la mesure

de la qualité du code ou même pour la prédiction des fautes. Dans cette étude, la technique d'Alves Rankings n'a pas été validée comme un bon choix de prédiction des fautes, mais la technique des courbes ROC en revanche l'a été partiellement sur un petit nombre de jeux de données.

Leur objectif était de comparer de façon empirique des méthodes sélectionnées pour le calcul des seuils, utilisées comme faisant partie des techniques de prédiction des fautes. Pour se faire, les seuils des différentes métriques orientées objet de quatre différents jeux de données ont été calculés. Ils ont utilisé les jeux de données du répertoire PROMISE et du projet Eclipse [18, 19, 34].

- Raed Shatnawi a une nouvelle fois fait une étude sur l'application de l'analyse des courbes ROC dans l'identification des seuils, les données débalancées et la sélection des métriques pour la prédiction des fautes logicielles [15]. Dans cette étude, l'auteur met en exergue le fait que toutes les métriques ne sont pas fortement corrélées avec les fautes. De ce fait, il a fixé trois objectifs. Le premier, l'analyse des courbes ROC est utilisée pour trouver la valeur seuil afin d'identifier les modules à risques. Le deuxième consiste à démontrer que la méthode d'analyse des courbes ROC est adéquate pour les données débalancées. Finalement, le troisième point consiste à valider l'utilisation de l'analyse des courbes ROC dans la sélection des métriques associées de manière significative à la prédiction des fautes dans les modules. Comme conclusion, cette étude a validé l'utilisation des courbes ROC pour identifier les seuils des métriques (WMC, CBO, RFC et LCOM). Les résultats des courbes ROC, après échantillonnage des données, ne sont pas significativement différents des résultats avant échantillonnage. Les courbes ROC sélectionnent les mêmes métriques (WMC, CBO et RFC) dans la plupart des jeux de données, en revanche, d'autres techniques d'échantillonnage telles que le suréchantillonnage avec SMOTE (Synthetic Minority Over-sampling Technique) et le sous-échantillonnage, ont une grande variation dans la sélection des métriques [15].

### 2.4.2 Effort de test

L'ISO définit la testabilité comme « des attributs de logiciel qui impliquent l'effort nécessaire pour valider le produit logiciel » [35]. À cette quête de la qualité sont liés des efforts impliquant des coûts. Cependant, la taille et la complexité importantes de la plupart des logiciels ne permettent pas un même effort pour l'ensemble des composants de ces derniers. D'ailleurs, pour Roger S. Pressman [36], l'objectif du test est de trouver un grand nombre d'erreurs possibles avec une quantité d'effort gérable appliquée sur une période réaliste.

C'est dans l'optique de trouver un équilibre entre l'effort de test et une période réaliste que plusieurs études ont été faites. Pour prédire la qualité des logiciels et leurs testabilités, ces études sont axées pour la plupart sur des approches basées sur les métriques. Nous présentons quelques-unes d'entre elles :

Melis Dagpinar et Jens H. Weber [37] ont étudié la prédiction de la maintenabilité mettant en corrélation les métriques et les efforts de maintenance estimés. Dans le même ordre d'idées, Magiel Bruntink et Arie van Deursen [38] ont étudié les facteurs de testabilité des systèmes orientés objet, en vue d'obtenir un modèle initial de testabilité et les métriques orientées objet existantes liées à la testabilité. Le but étant bien sûr, d'augmenter la compréhension des composants difficiles à tester. Ils ont pu démontrer une corrélation significative entre les métriques de classes (FANOUT, LOCC et RFC) et les métriques de test (dLOCC et dNOTC). De plus, ils ont discuté en détail de la façon dont diverses métriques peuvent contribuer à la testabilité. Pour ce faire, ils ont utilisé un logiciel libre et des systèmes Java commerciaux comme études de cas.

Par ailleurs, Binder [38, 39] a effectué une analyse des divers facteurs contribuant à la testabilité d'un système qu'il visualise à l'aide d'un diagramme en arêtes de poisson 2.1. Les principaux facteurs déterminants l'effort de test que Binder distingue comprennent le critère de test requis, l'utilité de la documentation, la qualité de la mise en œuvre, la réutilisabilité et la structure de la suite de tests, et enfin l'adéquation des outils de test utilisés, ainsi que les capacités du processus.

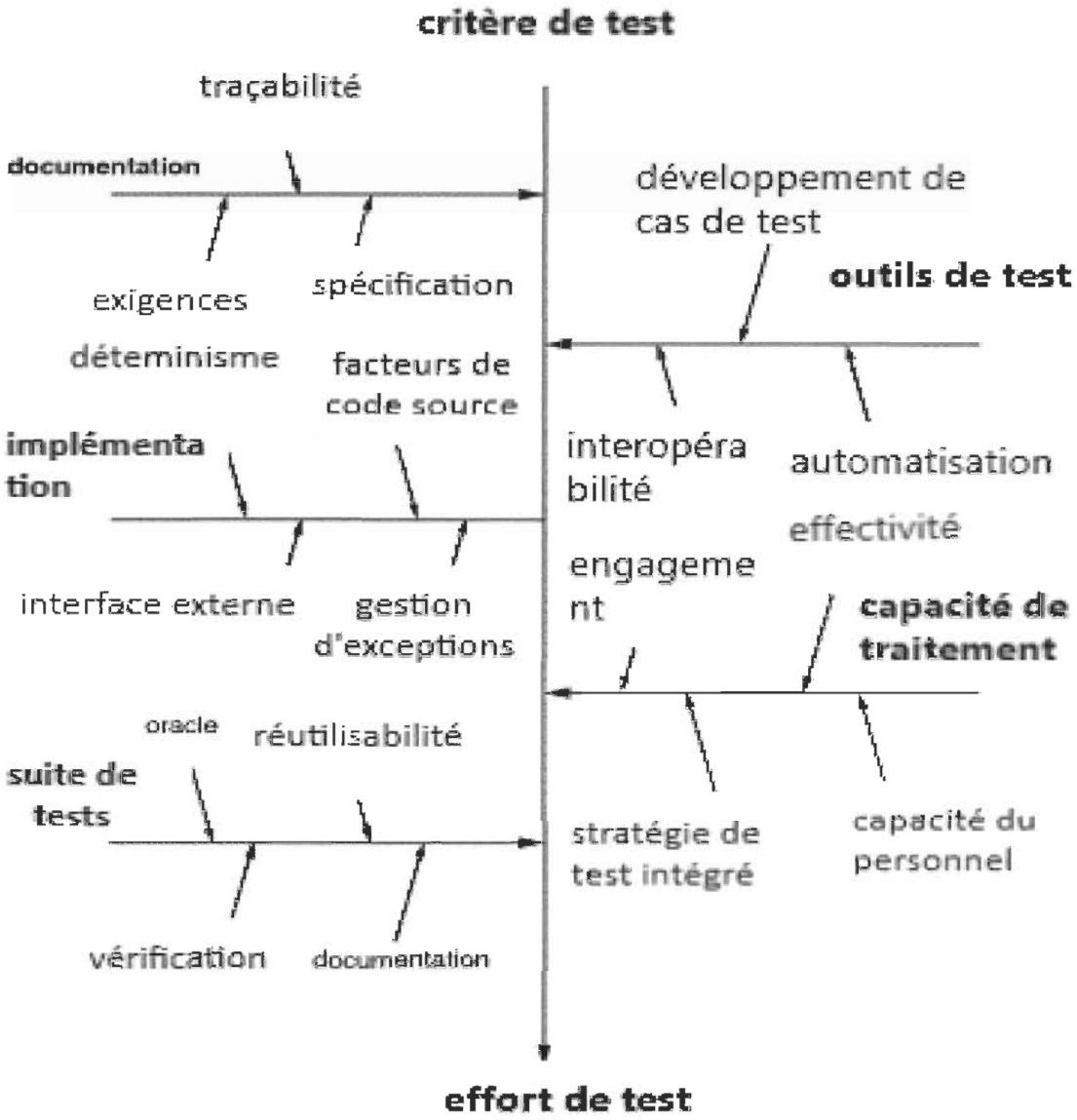


FIGURE 2.1 – La testabilité en arêtes de poisson [38]

Yogesh Singh, Arvinder Kaur et Ruchika Malhotra [40], dans leur étude pour la prédiction de l'effort de test, ont utilisé des techniques de Réseaux de Neurones Artificiels (RNA). Ils les présentent comme des outils intéressants qu'utilisent plusieurs chercheurs motivés par la construction des modèles de prédiction des attributs de qualité.

Le but étant d'examiner l'application des RNA pour la prédiction de la qualité logicielle en utilisant les métriques orientées objet. Les métriques CK considérées comme variables indépendantes ont été utilisées par les méthodes des RNA. Pour valider leurs travaux, les données publiques de la NASA ont été utilisées pour trouver la relation entre les métriques orientées objet et l'effort de test. Le modèle a estimé l'effort de test à

plus de 35% de l'effort réel dans plus de 72,54% des classes et avec une ERAM (Erreur Relative Absolue Moyenne) de 0,25.

Mourad Badri et Fadel Touré [41] ont étudié de manière empirique la relation entre les métriques de conception orientée objet et la testabilité des classes du point de vue de l'effort de test unitaire. Afin d'aboutir aux résultats escomptés, ils ont utilisé trois systèmes open source Java pour lesquels les cas des tests JUnit existaient. Pour capter cet effort de test, ils ont utilisé des métriques pour quantifier les cas des tests JUnit correspondants (leur code). Les classes ont été classifiées selon l'effort de test requis, en deux catégories : haut et bas. Ils ont, par ailleurs, utilisé la régression logistique univariée pour évaluer l'effet individuel de chaque métrique sur l'effort de test unitaire des classes. La régression logistique multivariée a été aussi utilisée pour explorer l'effet des métriques combinées. La performance des modèles de prédiction a été évaluée en utilisant les courbes ROC. Les résultats indiquent que la complexité, la taille, la cohésion et dans une certaine mesure le couplage sont de bons prédicteurs de l'effort de test unitaire des classes, et que les modèles de régression multivariée basés sur les métriques de conception orientée objet sont capables de prédire avec précision l'effort de test unitaire des classes.

### 2.4.3 Classification binaire

Comme l'ont souligné Yogesh Singh, Arvinder Kaur et Ruchika Malhotra [40], l'importance de la qualité logicielle est devenue une force de motivation pour le développement des techniques d'apprentissage automatique utilisées notamment pour la conception des modèles de prédiction.

C'est dans le cas des données non classifiées ou des données débalancées que l'apprentissage semi-supervisé peut être très utile. Cet avantage est dû au fait que cette technique d'apprentissage automatique utilise un ensemble de données étiquetées et non étiquetées.

C. Catal et B. Diri [42] ont travaillé sur un ensemble d'activités d'assurance qualité, notamment le test logiciel et la prédiction des fautes logicielles. Ils ont abordé, entre autres, l'effet de la taille des jeux de données et la taille de l'ensemble des métriques. Ils se sont focalisés sur la haute performance de prédiction des fautes basée sur l'apprentissage automatique tel que la Forêt aléatoire[42]. Le jeu de données publiques de la NASA du répertoire PROMISE (`http://promise.site.uottawa.ca/SERepository/datasets-page.html`) a été utilisé pour concevoir des modèles de prédiction reproductibles, réfutables et vérifiables. Afin de répondre aux questions de leur étude, sept groupes de tests ont été définis ainsi que neuf classificateurs ont été examinés pour chacun des cinq ensembles de données publiques de la NASA. Selon cette étude, la Forêt aléatoire produit les meilleures performances de prédiction pour de larges ensembles de données, en revanche le Bayésien naïf est un bon algorithme de prédiction pour les petits ensembles de données en fonction de l'AUC des courbes ROC comme paramètre d'évaluation.

## 2.5 Limitations

Les méthodes préconisées par les chercheurs et utilisées dans les différentes études présentent certaines limitations notamment concernant leur généralisation sur l'ensemble des logiciels qui sont souvent de différentes natures. Cette limite peut se justifier par différentes interprétations relatives aux métriques logicielles, c'est-à-dire que la définition de la valeur seuil ne peut être valide et applicable pour toutes les métriques.

Le choix du pourcentage (30% dans l'étude faite par A. Boucher et M. Badri [18]) de la distribution d'Alves Rankings pour trouver la valeur seuil constitue aussi une limitation. Cette dernière limitation montre qu'il est possible de procéder à un éventail de choix des pourcentages afin de déterminer d'une manière objective la valeur seuil. L'autre limitation, loin d'être la moindre, concerne l'utilisation de la matrice de confusion et des courbes ROC pour une évaluation des modules des logiciels sujets aux fautes ou aux efforts de test. Elles sont appliquées à tous les problèmes de classification et ne sont pas clairement et directement liées à la performance des modèles de prédiction des fautes et à l'effort de test [4].

## 2.6 Pistes potentielles

Comme nous l'avons mentionné dans la section précédente, les limites relatives aux méthodes évoquées ressortent des études réalisées en utilisant les techniques supervisées (courbes ROC) et non supervisées (Alves Rankings). Dans ce contexte, notre approche consiste en une application des techniques supervisées et non supervisées ainsi que la combinaison de ces techniques avec plusieurs algorithmes d'apprentissage automatique.

Raed Shatnawi a présenté un modèle basé sur l'analyse des courbes ROC pour la détermination des valeurs seuils [15]. Ce modèle est déterminé par l'utilisation de deux variables. Il s'agit d'une variable binaire qui consiste en l'existence d'une faute ou non. Dans notre cas, il s'agit de l'existence d'un effort important de test ou non. L'autre variable est continue, c'est celle des métriques CK. Cela consiste donc à déterminer l'aire sous la courbe (AUC) qui donne une valeur unique permettant ainsi d'évaluer le pouvoir de discrimination dans la courbe entre les classes contenant les fautes et celles qui n'en contiennent pas. Dans notre cas, il s'agit des classes qui demandent un effort important de test et celles qui n'en demandent pas. La valeur de l'AUC capable de déterminer la valeur seuil optimale des métriques doit être supérieure à 70%. Mais dans nos expérimentations, nous avons situé la valeur seuil optimale à 60% afin d'étendre le nombre des métriques ayant des valeurs seuils valides. Shatnawi dans ses travaux a utilisé la distance euclidienne à partir du point optimal des courbes ROC pour trouver les valeurs seuil optimales. Le modèle de Shatnawi possède également un avantage intéressant qui consiste à utiliser des données débalancées.

L'autre piste nous conduit à la méthode proposée par Alves et al. pour la dérivation des valeurs seuil [13]. Cette méthode a été utilisée et nommée *Alves Rankings* dans les

travaux de A. Boucher et M. Badri [19]. Elle a été appliquée pour déterminer les seuils des métriques pour la prédiction des fautes logicielles [19]. En revanche, Alves et al. l'ont initialement utilisée pour la détermination des niveaux de seuil afin de décrire la qualité des classes et de les catégoriser comme bonnes ou moins bonnes. Cette méthode comporte six étapes dans son application initiale, comme l'indique la figure 2.2 [13]. Mais selon les études de cas, certaines étapes peuvent être écartées, c'est le cas de A. Boucher et M. Badri [19] qui ont utilisées l'étape 1,2,3 et 6.

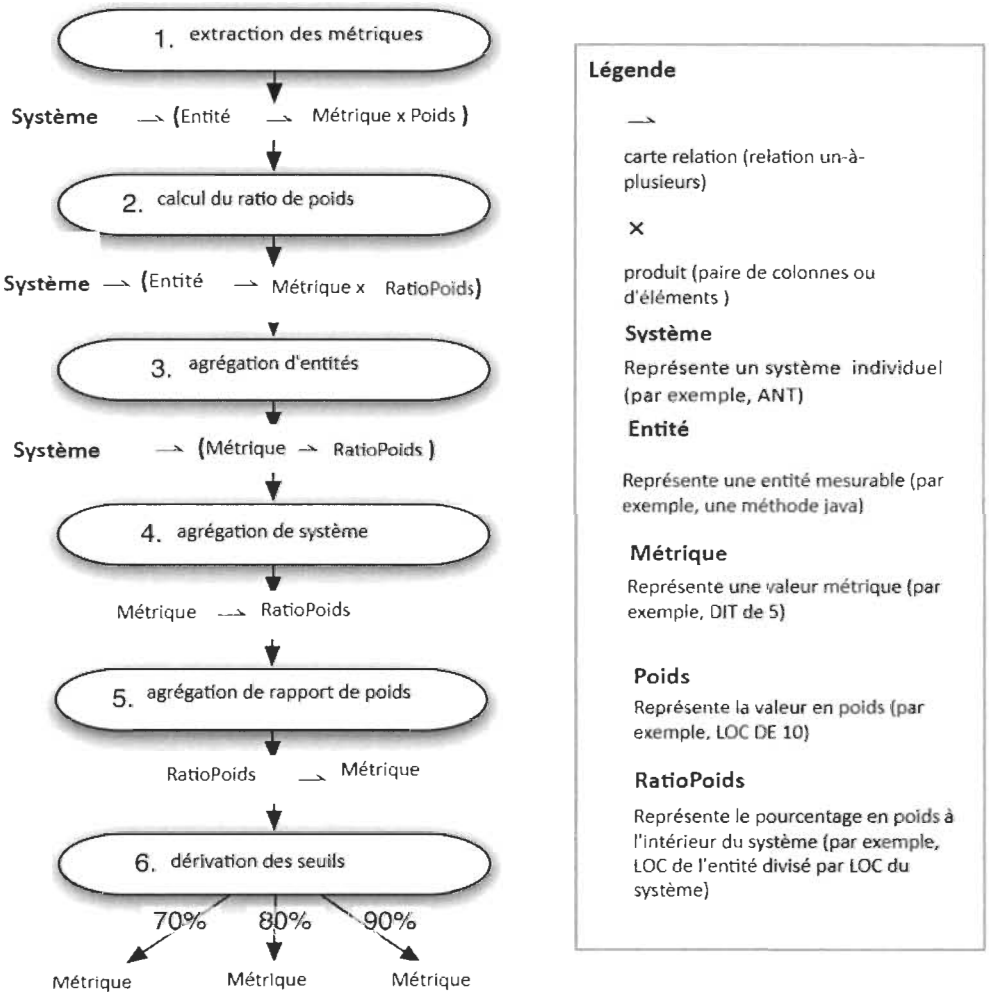


FIGURE 2.2 – Résumé des étapes de la méthode d'Alves et al. (*Alves Rankings*) [13]

La méthode Alves Rankings donne le choix de plusieurs niveaux de pourcentage, notamment 70%, 80% et 90%. Dans notre étude, nous explorerons ces trois niveaux de pourcentage, et nous ajouterons la dérivation des seuils à 30%. Il est évident que cette méthode a été appliquée sur la prédiction des fautes, mais nous l'appliquerons sur la prédiction de l'effort de test.

## 2.7 Conclusion

Après l'analyse de plusieurs travaux de recherche, il est utile de souligner l'apport et l'usage importants de la suite de métriques orientées objet CK dans le domaine de l'assurance qualité des logiciels [3, 43–45]. Cette suite a été aussi utilisée pour la validation empirique de plusieurs études [18, 46, 47] qui se sont révélées très utiles dans la détermination des seuils pour prédire les classes sujettes aux fautes. Les métriques ont également été utilisées pour la prédiction de l'effort de test des différents tests unitaires, mais sans toutefois aborder le point lié à la détermination du niveau de seuil. C'est dans cette optique que nous tenterons d'appliquer les différentes techniques de calcul de seuils, pour améliorer la qualité de leur validité, mais aussi pour déterminer quelles techniques donnent les meilleures performances une fois ces techniques combinées aux algorithmes d'apprentissage automatique.

*Chapitre. 3*

# *MÉTHODOLOGIE DE RECHERCHE*

## 3.1 Introduction

La prédiction des efforts de test a fait l'objet de plusieurs travaux, dont le nombre reste de loin inférieur aux travaux consacrés à la prédiction des efforts de développement. Par ailleurs, le sujet au mieux de nos connaissances n'a jamais été abordé sous la perspective de valeurs seuils. C'est dans ce contexte que nous avons appliqué, éventuellement adapté quelques techniques, en particulier celles qui sont non supervisées pour la détermination des seuils et la prédiction des efforts de test. Nous avons exploré aussi la combinaison de ces techniques avec plusieurs algorithmes d'apprentissage automatique. Nous avons utilisé des ensembles de données de 8 différents systèmes : ANT, IO, IVY, JFC, JODA, LUCENE, MATH et POI pour construire les différents modèles de prédiction.

Cette démarche se fait en 3 grandes étapes. La première étape permet la classification binaire des métriques à différents niveaux de seuils. La deuxième étape consiste en la prédiction des efforts de test. Enfin, la troisième étape consiste à évaluer les performances des modèles.

Dans ce qui suit, nous allons présenter nos questions de recherche ainsi que la démarche adoptée pour notre méthodologie de recherche.

## 3.2 Questions de recherche

Nous nous sommes proposé de répondre aux questions de recherche suivantes dans notre étude :

**QR1 :** Comment déterminer de façon objective des seuils permettant de prédire l'effort de test ? Au lieu de déterminer des valeurs seuils d'une manière empirique ou selon l'opinion des experts, nous avons appliqué différentes techniques pour calculer objectivement ces valeurs.

**QR2 :** Comment utiliser les techniques d'apprentissage automatique pour améliorer les performances de ces techniques ?

Dans l'optique d'une optimisation des performances, nous avons étudié la façon d'utiliser ces techniques afin d'accroitre la performance des modèles.

**QR3 :** L'approche avec valeurs seuils des métriques est-elle plus efficace que l'approche avec valeurs des métriques brutes ?

Nous avons évalué si l'application d'une approche basée sur les seuils des métriques donne de meilleures performances que celle basée sur les métriques brutes. Les valeurs seuils sont définies à plusieurs niveaux de pourcentage suivant l'application des techniques de calcul.

**QR4 :** Quelle technique parmi les courbes ROC et Alves Rankings donne les meilleures performances pour la prédiction de l'effort de test ?

Ces deux techniques ont été utilisées notamment pour la prédiction de fautes logicielles. Nous les avons également utilisées pour savoir si, appliquées dans notre étude, elles donnent de bonnes performances pour la prédiction des efforts de test.

## 3.3 Présentation des données

Pour les besoins de notre étude, nous avons sous format Excel des données de plusieurs systèmes orientés objet et codes ouverts écrits en Java. Ces systèmes ont été développés par différentes équipes et leurs domaines d'application diffèrent. Les données sont représentées avec l'effort de test réellement investi par les développeurs/testeurs (capturé à l'aide de plusieurs métriques). Nous présentons, ainsi, dans les lignes qui suivent, tous les systèmes par ordre décroissant du nombre de classes logicielles.

### 3.3.1 Les systèmes logiciels

- POI : Apache POI (`http://poi.apache.org/`) est une API (Application Programming Interface) Java permettant de manipuler différents formats de fichiers sur les normes Office Open XML (OOXML) et OLE 2, format qu'utilisent les documents de la suite office de Microsoft (Excel, Word et PowerPoint) [14]. Le système contient 57 734 lignes de code réparties sur 351 classes logicielles.

- JFC : JFreeChart (`http://www.jfree.org/jfreechart/`) est une bibliothèque Java graphique qui facilite, pour les développeurs, l'intégration de graphique de qualité professionnelle dans leurs applications. Elle comprend une API cohérente et documentée, soutenant un large éventail de types de graphiques, une conception flexible qui est facile à étendre, et des applications qui ciblent à la fois le côté serveur et le côté client. JFreeChart supporte aussi différents types de sorties, y compris les composants Swing et JavaFX, et exporte les graphiques dans les formats

variés d'images et de graphiques vectoriels [14]. Le système contient 53 115 lignes de code réparties dans 226 classes logicielles.

- LUCENE : Apache Lucene (`http://lucene.apache.org/`) est une bibliothèque de recherche d'information textuelle complète de haute performance, entièrement écrite en Java. C'est une technologie appropriée pour presque toutes les applications qui nécessitent la recherche textuelle avancée, en particulier pour les systèmes multiplateformes [14]. Le système renferme 22 098 lignes de code contenues dans 114 classes.

- ANT : Apache Ant (`http://www.apache.org/`) est une bibliothèque et une ligne de commande utilitaire écrite en Java dont le rôle est de piloter le lancement de processus décrits dans des fichiers XML comme des objectifs et des points d'extensions interdépendants. L'utilisation principale connue de Ant est la construction d'applications Java à la manière des *makefiles* pour le langage C. Ant fournit un certain nombre de tâches intégrées permettant de compiler, assembler, tester et exécuter des applications Java. La librairie peut également piloter tout type de processus pour décrire les objectifs et les tâches [14]. Nous avons, pour notre étude, utilisé le système contenant 8 730 lignes de code réparties dans 111 classes logicielles.

- IVY : Apache Ivy (`http://ant.apache.org/ivy/`) est une bibliothèque de gestion de dépendances intégrées à ANT, et orientées vers la gestion de dépendances Java. Elle peut, cependant, être utilisée pour gérer les dépendances de toute sorte dans un projet logiciel (Gestionnaire de dépendances agile) [14]. Le système contient 18 047 lignes de code réparties dans 95 classes.

- JODA : La librairie Joda-Time (`http://joda-time.sourceforge.net/`) regroupe un ensemble de fonctionnalités de manipulation des dates écrites en Java. Elle est conçue comme une alternative complète aux classes Date et Timestamp de JDK (Java Development Kit) et dispose d'une interface API intégrable pour divers systèmes de calendrier. Le déficit de fonctionnalités de manipulation de dates dans les versions de Java antérieures à Java8 a rendu Joda-Time de facto le standard de manipulation de date pour ces anciennes versions de JDK [14]. Le système renferme près de 17 624 lignes de code contenues dans 76 classes logicielles.

- IO : La librairie Commons IO (`http://commons.apache.org/io/`) est développée par Apache Software Foundation (ASF). Il s'agit d'une bibliothèque d'utilitaires standards permettant de manipuler les entrées/sorties des différents systèmes d'exploitation [14]. Le système utilisé dans notre étude dispose de 6 333 lignes de code contenues dans 67 classes logicielles.

- MATH : Cette librairie (`http://commons.apache.org/proper/commons-math/`) est développée elle aussi par l'ASF. Elle est simple et contient des fonctionnalités mathématiques et statistiques les plus communes qui ne sont pas disponibles dans la

bibliothèque Math de base du langage Java. Tous les algorithmes sont documentés et suivent les meilleures pratiques généralement acceptées. Cette bibliothèque n'a pas des dépendances externes au-delà du langage Java [14]. Le système utile pour nos expérimentations renferme 5 406 lignes de code contenues dans 59 classes logicielles.

### 3.3.2 Métriques Orientées objet

Les métriques logicielles orientées objet mesurent les attributs des classes logicielles. La majorité de ces métriques (du moins celles qui sont les plus utilisées) font partie de la suite de métriques OO de Chidamber et Kemerer (CK) [3]. Ces métriques capturent différentes caractéristiques des classes telles que le couplage, la cohésion, la complexité, l'héritage et la taille. Ces métriques que nous avons suffisamment abordées dans les lignes précédentes sont : CBO, DIT, LCOM, LOC, NOC, RFC et WMC.

## 3.4 Apprentissage automatique

Dans la présente section, nous présentons une brève description des algorithmes d'apprentissage automatique que nous avons utilisés au moyen de l'outil WEKA (outil d'exploration de données et de classification) [48].

Selon **Tom Mitchell** l'apprentissage automatique est : *Un programme informatique est réputé apprendre de l'expérience E en ce qui concerne certaines classes de tâches T et de la mesure de performance P, si sa performance aux tâches en T, mesurée par P, s'améliore avec l'expérience E* [49]. Telles que :

- Tâche T : Ce que la machine cherche à améliorer. Ça pourrait être : la prédiction, la classification, le regroupement (clustering) etc.
- Expérience E : Ça peut être les données d'entraînement ou les données d'entrée à travers lesquelles la machine essaie d'apprendre.
- Performance P : Ça peut être un facteur comme la précision des améliorations ou de nouvelles compétences que la machine ignorait auparavant, etc.

Ceci peut être illustré par le diagramme sémantique d'un système d'apprentissage automatique [49] ci-dessous :

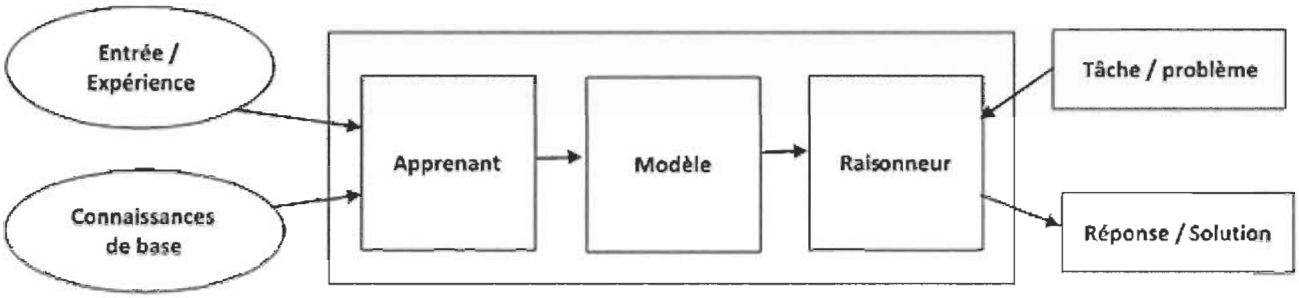


FIGURE 3.1 – Diagramme Sémantique d'un système d'apprentissage automatique [49]

Selon ce diagramme, un système d'apprentissage automatique comprend deux principaux composants, à savoir : Un *Apprenant* et un *Raisonneur*. Ci-dessous, nous décrivons chacun des composants de ce diagramme :

- *Entrée/Expérience* est fourni à *l'Apprenant* qui apprend de nouvelles compétences.
- *Connaissances de base* peut aussi être fourni à *l'Apprenant* pour un meilleur apprentissage.
- Avec l'aide de *l'Entrée* et *Connaissances de base*, *l'Apprenant* génère le modèle.
- *Modèle* contient l'information à propos de comment il a appris de *l'Entrée* et de *l'Expérience*.
- *Tâche/problème* est fourni au *Raisonneur*, il peut s'agir d'une prédiction, Classification etc.
- Avec l'aide du modèle entrainé, le *Raisonneur* essaie de générer la solution.
- *Réponse/Solution* peut être amélioré par l'addition d'un *Entrée/Expérience* et de *Connaissances de base* additionnels.

Les algorithmes d'apprentissage automatique sont utilisés dans la prédiction des fautes logicielles afin de comprendre les relations entre les ensembles de données des métriques de code source et les erreurs. Dans notre contexte, ces algorithmes ont été entrainés sur les données des métriques et d'efforts de test réels pour y trouver des modèles. Il s'agissait par la suite d'utiliser ces modèles pour effectuer des prédictions sur de nouvelles données. Cinq algorithmes ont été utilisés dans notre étude pour prédire les efforts de test. Après cette brève introduction à l'apprentissage automatique, nous allons présenter les différents algorithmes utilisés dans notre étude. Il s'agit d'algorithmes suivants : Bayésien naïf, Forêt d'arbres décisionnels, Machine à vecteurs de support, K plus proches voisins et Perceptron multicouche.

### 3.4.1 Bayésien naïf

Le Bayes naïf est un style de classification bayésienne probabiliste basé sur le théorème de Bayes fondé sur une forte hypothèse sur les descriptions (conditionnellement indépendants) d'où le qualificatif naïf [14]. Il présente un avantage qui est la simplicité de programmation, la facilité d'estimation des paramètres et sa rapidité même sur de très grandes quantités de données. Sur un ensemble d'apprentissages, l'approche consiste à : (1) déterminer les probabilités a priori de chaque classe (analyse de fréquence), (2) à appliquer la règle de Bayes pour déterminer les probabilités a posteriori des classes, et (3) à choisir la classe la plus probable.

Cet algorithme a certains avantages, notamment :

- Simple et facile à implémenter ;
- Demande moins de données d'entraînement ;
- Fonctionne à la fois sur des données discrètes et réelles ;
- Très rapide, permet une classification en temps réel.

Cet algorithme a été fortement utilisé dans le domaine de prédiction des fautes [12, 18, 50, 51].

### 3.4.2 Forêt d'arbres décisionnels

L'algorithme de Forêt d'arbres décisionnels a été développé par Leo Breinma, statisticien à l'Université de Californie Berkeley [52]. Il s'agit d'un méta-apprenant composé de nombreux arbres individuels conçu pour fonctionner rapidement sur de grands ensembles de données et surtout pour être diversifié en utilisant des échantillons aléatoires pour construire chaque arbre. Les données d'apprentissage sont échantillonnées pour créer une partie des données extraites de l'échantillon lors d'une réplication appelée *in-bag portion*, afin de construire l'arbre. Une plus petite partie non extraite de l'échantillon qu'on appelle *out-bag portion* consiste à tester l'arbre terminé pour évaluer ses performances. Cette mesure de performance est connue sous le nom d'erreur *out-of-bag* ou d'estimation de l'erreur hors du sac [53]. Cet algorithme a été utilisé notamment dans la prédiction des fautes logicielles [12, 18, 52, 54, 55]. Ci –dessous sont présentés quelques avantages de cet algorithme :

- Peut gérer des ensembles de données volumineux comportant un grand nombre d'attributs qui seront répartis entre les arbres ;
- Il peut modéliser l'importance des attributs ; par conséquent, il est également utilisé pour la réduction de la dimensionnalité ;
- Il maintient la précision, même lorsque des données sont manquantes ;

- Fonctionne avec des données non étiquetées (apprentissage non supervisé) pour la formation des grappes (clusters), la visualisation des données et la détection des valeurs aberrantes.

### 3.4.3 Machine à vecteurs de support

Les machines à vecteurs de support sont basées sur le concept des plans de décision de la théorie de l'apprentissage statistique qui définit les limites de décision. Un plan de décision sépare idéalement les objets appartenant à des classes différentes [56].

Cependant, en raison de sa faible complexité, il ne peut classer correctement que les données séparables linéairement. D'autre part, une limite de décision plus complexe peut correctement classifier des données générales qui ne peuvent pas être séparables linéairement. Cependant, un tel classificateur peut être beaucoup plus difficile à former. Une machine à vecteurs de support (SVM) combine le meilleur des deux mondes. En d'autres termes, elle utilise un algorithme d'entraînement efficace tout en représentant des limites de décisions complexes [57]. Elle est aussi utilisée dans la prédiction des fautes logicielles [57], dans la prédiction de test [58] et dans la détermination des valeurs seuils pour prédire les fautes [18, 19, 55].

### 3.4.4 K plus proches voisins

L'algorithme des K plus proches voisins est une méthode de classification et de régression non paramétrique permettant de reconnaitre des patrons dans les observations. Il consiste à estimer la classe d'une nouvelle entrée E en prenant en compte, avec le même poids, les k échantillons d'entraînement les plus "proches" de la nouvelle entrée E. La proximité est déterminée par une distance. Cet algorithme a fait l'objet d'étude pour la prédiction des fautes logicielles [59–63].

### 3.4.5 Perceptron multicouche

Le Perceptron multicouche est un exemple de réseau neuronal artificiel. Il est utilisé pour résoudre différents problèmes, par exemple, la reconnaissance de formes, l'interpolation, etc. Il constitue une avancée dans le modèle de réseau neuronal perceptron. Avec une ou deux couches cachées, ils peuvent résoudre presque tous les problèmes. Ce sont des réseaux neuronaux *feedforward* basés sur l'algorithme de propagation arrière. L'apprentissage par propagation arrière sur erreur se compose de deux passages : un passage en avant et un passage en arrière. Lors du passage en avant, une entrée est présentée au réseau de neurones et son effet se propage couche par couche dans le réseau. Lors du passage en avant, les poids du réseau sont fixes. Et lors du passage en arrière, les poids sont tous mis à jour et ajustés en fonction de l'erreur calculée. Une erreur est composée de la différence entre la réponse souhaitée et la sortie du système. Cette information d'erreur est renvoyée au système permettant ainsi d'ajuster les paramètres du système de manière systématique (règle d'apprentissage) [55]. Le

processus est répété jusqu'à ce que la performance soit acceptable [64]. Évidemment, cet algorithme a été fortement utilisé dans diverses études pour la prédiction de la qualité logicielle [18, 55, 64].

## 3.5 Techniques de calcul de seuils

Dans cette section, nous présentons deux techniques de calcul de seuils pour la prédiction des efforts de test. Ces techniques seront évaluées et comparées entre elles pour savoir celle produisant de meilleures performances une fois combinées aux algorithmes d'apprentissage automatique. Il s'agit de : courbes ROC et Alves Rankings.

### 3.5.1 Courbes ROC

L'analyse des courbes ROC est un test de précision diagnostique [65]. La méthode ROC peut être utilisée pour évaluer la qualité des informations fournies par la classification des classes dans une catégorie binaire à l'aide d'une métrique. Pour tracer la courbe ROC, nous devons définir deux variables : une binaire (c'est-à-dire 0 ou 1) et une autre continue [16]. La valeur continue est celle de la métrique utilisée dans cette étude. Après avoir défini une plage de valeurs de seuil pour les métriques, un tableau de classification (matrice de confusion) est créé pour chaque valeur de seuil. Chaque table produit un point dans la courbe ROC [16]. Chaque point est une paire de valeurs de *sensibilité* et de *1- spécificité*. Nous devons rassembler autant de paires que possible pour compléter le graphique ROC, c'est-à-dire trouver la paire pour chaque valeur métrique du minimum au maximum. Chaque paire représente les performances de classification de la valeur seuil qui produit cette paire [16]. La paire qui produit les meilleures performances de classification est la meilleure valeur seuil à utiliser dans la pratique [16].

La table 3.1 illustre comment les performances de ces paires sont calculées dans notre étude. Ces valeurs seuils divisent l'ensemble de données en deux sous-ensembles : les ensembles avec effort de test important et les ensembles avec effort de test relativement faible, selon que les valeurs de métriques sont inférieures ou non à la valeur seuil.

La sensibilité et la spécificité sont calculées à partir de la matrice de confusion comme suit :

| | Actuel | |
|---|---|---|
| Classifié | Effort de test | Pas d'effort de test |
| Métrique $\geq$ seuil | Vrais-positifs | Faux-positifs |
| | (TP ou alarmes correctes) | (FP ou fausses alarmes) |
| Métrique $<$ seuil | Faux-Négatifs | Vrais-Positifs |
| | (FN ou alarmes manquées) | (TP ou pas d'alarmes) |
| Totaux | P=TP+FN | N=FP+TN |

TABLE 3.1 – La matrice de confusion basée sur une valeur seuil

### 3.5.2 Alves Rankings

La méthode d'Alves et al. pour calculer les seuils n'avait pas de nom dans le papier original [13, 19]. C'est A. Boucher et M. Badri qui l'ont intitulé *Alves Rankings* dans leurs travaux [19]. Alves et al. ont défini leur méthode pour trouver des seuils qui décrivent la qualité des classes et enfin les catégoriser. Pour ce faire, ils ont procédé en 6 étapes pour le calcul des seuils [13]. En ce qui nous concerne, nous n'utiliserons que les étapes 1, 2, 3 et 6, jugées pertinentes pour notre étude.

La première étape, appelée *extraction de métriques*, consiste à extraire les métriques du système [13, 19]. Les métriques de code de chaque classe sont calculées à cette étape, mais également le poids de chaque classe [19]. Le poids d'une classe est défini par la métrique SLOC (Source Lines Of Code) dans le document d'Alves et al.. Pour notre étude, la première étape a consisté à rechercher les ensembles de données que nous avons décidé d'utiliser.

La deuxième étape, appelée *calcul du rapport de pondération* ou *ratio de poids*, consiste à calculer le rapport de pondération de chaque classe [13]. Ce ratio est calculé simplement en divisant le poids d'une classe (LOC) par la somme des poids de toutes les classes. Le rapport pondéral représente simplement la taille relative de chaque classe du système. Par exemple, si une classe a un rapport de pondération de 0,01, cela signifie que le code de la classe représente 1% du code total du système [19].

La troisième étape, appelée *agrégation d'entités*, consiste à agréger le poids de toutes les entités (qui sont ici des classes) par des valeurs métriques [13]. Le résultat de cette étape est semblable à un histogramme pondéré, donnant le pourcentage de code du système représenté par chaque valeur métrique. Par exemple, après cette étape, nous pourrions dire que 1% du LOC d'un système est représenté par une métrique CBO de 6 [19].

Les quatrième et cinquième étapes de la méthode ne sont pas utilisées dans notre étude. Cela est justifié par le fait que Alves et al. ont calculé les seuils utilisant une centaine de systèmes logiciels différents [13]. Dans notre cas, nous voulions calculer pour chaque système une valeur seuil pour chaque mesure de code. Les quatrième et

cinquième étapes de la méthode Alves Rankings ont consisté à normaliser les poids de chaque système évalué [19]. Par la suite, ils ont agrégé les valeurs métriques pour ces systèmes, obtenant ainsi un résultat similaire à celui de la troisième étape. La différence est que le pourcentage de chaque métrique représente le pourcentage de code sur tous les systèmes, pas un seul comme dans la troisième étape [19].

La sixième étape de cette méthode, appelée *dérivation de seuils*, consiste à calculer les valeurs seuils pour chaque classe [19]. Pour ce faire, nous avons défini un pourcentage de code que nous représentons avec nos valeurs seuils. Par exemple, choisir 80% du code global pourrait générer une valeur seuil de 30 pour la métrique CBO [19]. Cela signifierait que 20% du code de mauvaise qualité selon la métrique CBO seraient ciblés par la valeur seuil de 30 [19]. Ces différentes étapes sont décrites à la figure 2.2.

Nous avons utilisé des valeurs seuils définies à 30%, 70%, 80% et 90% des distributions de mesure pour nos modèles de prédiction.

Toutefois, cette technique a été étudiée pour trouver les seuils des métriques pour la prédiction des fautes [19], mais ne l'a pas été pour la prédiction des efforts de test, moins encore dans une approche basée sur les seuils des métriques et les algorithmes d'apprentissage automatique.

## 3.6 Méthodes d'évaluation de performance

Pour évaluer la performance de la prédiction de chaque technique de calcul de valeurs seuils et des modèles d'apprentissage automatique, nous avons utilisé les métriques suivantes : FPR (Taux de vrais positifs), FNR (Taux de faux négatifs), l'AUC (l'aire sous la courbe) ainsi que g-mean (moyenne géométrique). Elles peuvent être facilement calculées en utilisant la matrice de confusion décrite ci-dessous, résultant de la classification. Outre ces métriques, nous avons appliqué la validation croisée 10-Fold pour les algorithmes d'apprentissage automatique que nous avons utilisés dans nos expérimentations pour construire les modèles de prédiction.

| **Classifié** | **Actuel** | |
|---|---|---|
| | Fautive | Non-Fautive |
| Fautive | Vrais positifs (TP) | Faux positifs (FP) |
| Non-Fautive | Faux négatifs (FN) | Vrais négatifs (TN) |

TABLE 3.2 – Matrice de Confusion

### 3.6.1 10-Fold Cross Validation

Pour calculer avec précision les résultats de prédiction, une validation croisée de 10-Fold Cross Validation (CV) est utilisée. Cette méthode de validation croisée divise

les jeux de données étudiés en 10 parties égales (plis). 9 plis sur 10 sont ensuite utilisés comme données d'apprentissage et le pli restant est utilisé pour tester la prédiction. Cette démarche est effectuée 10 fois, chaque fois en utilisant un pli différent pour les tests. Chaque tableau de classification est résumé pour donner un tableau final, donnant la performance globale [19].

Cette méthode a des caractéristiques qui la rendent efficace, notamment :

- Utilise l'intégralité du jeu de données pour l'entraînement et pour la validation ;
- Un autre aspect de cette technique est la stratification qui est automatiquement intégrée dans WEKA (l'outil utilisé dans le présent travail) quand on sélectionne l'option **Cross-Validation** ;
- La stratification permet de créer les *k plis* de sorte qu'ils contiennent à peu près les mêmes proportions d'exemples de chaque classe que le jeu de données complet, on évite ainsi d'affecter négativement la performance du modèle ;
- Une procédure de validation plus robuste, car plusieurs exécutions de validation sont effectuées au lieu d'une seule ;
- La technique de validation croisée est utilisée pour remédier au problème du surapprentissage.

### 3.6.2 G-mean

Les métriques FPR et FNR sont utilisées dans notre approche pour évaluer la performance des modèles de prédiction des efforts de test [66, 67]. Dans ce contexte, nous avons adapté ces modèles appliqués par A. Boucher et M. Badri [19].

L'autre aspect motivant l'utilisation des métriques FPR et FNR est qu'elles sont plus importantes en matière d'évaluation du pourcentage de classes mal classifiées. Ces métriques décrivent mieux la classification, spécialement lorsque les données sont débalancées (différence importante entre le nombre de classes qui demandent un effort de test important et celui des classes qui en demande moins).

La métrique g-mean a été définie spécialement pour décrire la classification des données débalancées [26]. Elle a aussi été utilisée dans diverses études pour la prédiction des fautes logicielles [26, 68]. Ci-dessous les équations permettant de calculer les métriques FPR et FNR :

$$FPR = \frac{FP}{FP+TN}$$
$$FNR = \frac{FN}{FN+TP}$$

La métrique FPR donne le pourcentage de faux positifs parmi toutes les valeurs négatives réelles, tandis que la métrique FNR donne le pourcentage de faux négatifs parmi toutes les valeurs positives réelles. Plus chaque mesure est basse, meilleure est la classification.

La métrique g-mean utilise deux précisions différentes, qui sont l'exactitude des positifs (TPR) et l'exactitude des négatifs (TNR) [26] (qui sont l'opposées respectivement des métriques FPR et FNR dans leurs interprétations). Contrairement à FPR et FNR où plus les valeurs sont basses, meilleures elles sont, les métriques TPR et TNR et g-mean indiquent une bonne performance quand leurs valeurs sont élevées. La métrique g-mean sera acceptable si les métriques TPR et TNR sont bonnes, sinon ce ne sera pas le cas. On trouve ci-dessous les équations utilisées pour le calcul des métriques TPR (Taux des vrais positifs), TNR (Taux des vrais négatifs) et g-mean :

$$TPR = 1 - FNR = \frac{TP}{TP+FN}$$
$$TNR = 1 - FPR = \frac{TN}{TN+FP}$$
$$g - mean = \sqrt{TPR \times TNR}$$

Nous avons considéré les niveaux suivants pour décrire les valeurs de la moyenne géométrique (g-mean) obtenues, afin de représenter les valeurs de g-mean de manière textuelle et simplifier l'analyse des résultats [19, 33] :

- $g - mean < 0.5$ signifie mauvaise classification.
- $0.5 \leq g - mean < 0.6$ signifie faible classification.
- $0.6 \leq g - mean < 0.7$ signifie classification acceptable.
- $0.7 \leq g - mean < 0.8$ signifie bonne classification.
- $0.8 \leq g - mean < 0.9$ signifie très bonne classification.
- $g - mean \geq 0.9$ signifie excellente classification.

### 3.6.3 L'aire sous la courbe

L'aire sous la courbe (AUC) de la fonction d'efficacité du récepteur (ROC) est utilisée dans l'évaluation des classificateurs. La courbe ROC est tracée au moyen de deux variables : une variable binaire et une variable continue. La variable binaire est l'occurrence d'une classe sujette à nécessiter un effort de test important ou non. La variable continue sera la valeur respective de chaque métrique. Le modèle est évalué pour décider si une classe demande un effort important de test ($\geq$ seuil) ou non ($<$ seuil). Pour chaque seuil potentiel, une matrice de confusion est construite. Chaque matrice peut être utilisée pour calculer deux mesures importantes, à savoir : la sensibilité et la spécificité, respectivement TPR et TNR comme mentionné ci-dessus.

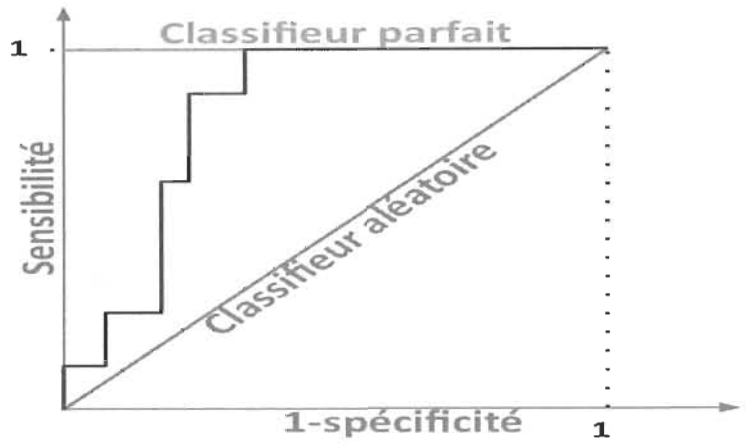


FIGURE 3.2 – Courbe ROC

L'AUC (Area Under the Curve) représentant l'aire sous la courbe ROC (Receiver Operating Characteristics) a une valeur entre 0 et 1. Une valeur proche de 1 signifie que le modèle s'ajuste parfaitement avec les données alors qu'une valeur proche de 0.5 indique que le modèle ne fait pas bonne figure et donne une classification complètement aléatoire. L'AUC permet de quantifier les performances d'un modèle, et de les comparer à celles d'autres modèles. Shatnawi et al. ont présenté différents niveaux de valeurs de l'AUC pour classifier les bonnes ou mauvaises métriques de classification, ces niveaux sont les suivants [16] :

- AUC = 0.5 signifie mauvaise classification, assimilée à une classification aléatoire ;
- 0.5 < AUC < 0.6 signifie mauvaise classification ;
- $0.6 \leq AUC < 0.7$ signifie classification juste ;
- $0.7 \leq AUC < 0.8$ signifie classification acceptable ;
- $0.8 \leq AUC < 0.9$ signifie une excellente classification ;
- $AUC \geq 0.9$ signifie une classification exceptionnelle.

## 3.7 Méthodes de comparaison de performance

Afin de déterminer si un modèle présente une meilleure performance qu'un autre, nous avons besoin d'une méthodologie de comparaison objective. Une telle méthodologie a été suggérée par Demšar avec pour objectif de comparer les performances de différents modèles ou classificateurs sur plusieurs jeux de données [69].

Cette méthodologie consiste à utiliser le test statistique de Friedman en conjonction avec un test post-hoc appelé Nemenyi [19]. Le test de Friedman [19, 70, 71] ainsi que le test de Nemenyi [19] ont également été utilisés dans d'autres études sur la prédiction des fautes pour comparer les résultats de différents modèles.

### 3.7.1 Test de Friedman

Le test de Friedman (Friedman, 1937, 1940) est un équivalent non paramétrique de l'ANOVA (l'analyse de variance qui permet de tester la différence entre les moyennes de plusieurs sous-groupes d'une variable) à mesures répétées. Il classe les algorithmes pour chaque ensemble de données séparément. L'algorithme le plus performant a le rang 1, le deuxième meilleur a le rang 2, etc. [69].

Le test de Friedman est intéressant dans ce cas, car il s'agit d'un test non paramétrique, ce qui signifie que les variables ne suivent pas une distribution particulière. Il n'évalue pas la performance de la distribution. Il compare uniquement les performances des différentes distributions. Pour ce faire, il compare les classements moyens des différents modèles sur les différents jeux de données. La statistique de Friedman est donc calculée comme suit : k est le nombre des modèles, N le nombre des jeux de données et Rj le rang moyen du modèle j sur tous les jeux de données.

$$X_F^2 = \frac{12N}{k(k+1)} \left( \sum_j R_j^2 - \frac{k(k+1)^2}{4} \right)$$

La statistique $X_F^2$ est ensuite comparée à sa valeur critique pour vérifier si l'hypothèse nulle est rejetée ou non. L'hypothèse nulle du test indique qu'il n'y a pas de différence significative entre les modèles. Si l'hypothèse nulle est rejetée, il existe une différence significative entre au moins deux des modèles.

### 3.7.2 Test de Nemenyi

Demšar recommande de faire le test post-hoc de Nemenyi pour comparer les performances de chaque paire de modèles [19, 69]. Selon le test de Nemenyi, il existe une performance significative différente entre deux modèles si le rang moyen CD diffère au moins de la différence critique (disponible dans [69]).

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}$$

Dans l'équation ci-dessus, $q_\alpha$ est basé sur les valeurs critiques de la statistique de la plage Studentisée, divisées par 2, selon [19, 69].

Dans notre étude, nous avons donc décidé d'utiliser le test de Friedman et le test post-hoc de Nemenyi pour comparer statistiquement les performances des modèles. Nous avons effectué le test de Friedman en utilisant la métrique de performance g-mean, qui décrit bien la performance.

Les tests statistiques sont effectués à l'aide des outils IBM SPSS Statistics V25 [72] pour le test de Friedman et la librairie PMCMR de R [73] pour le test de Nemenyi en utilisant 5% comme niveau de confiance.

*Chapitre. 4*

# *EXPÉRIMENTATIONS ET RÉSULTATS*

## 4.1 Introduction

Comme nous l'avons mentionné précédemment, plusieurs travaux ont été effectués sur la prédiction des efforts de test, et à notre connaissance l'aspect valeurs seuils n'a pas été abordé. C'est dans le but d'étudier cet aspect que nous avons effectué des expérimentations sur les jeux de données présentés au chapitre précédent. Dans cette section, les différents expérimentations et résultats sont présentés et discutés. Nous commençons par la section consacrée à la classification binaire pour finir par celle consacrée à la prédiction des efforts de test dans une approche basée sur les valeurs seuils des métriques logicielles.

## 4.2 Classification binaire

Shatnawi et al [16] ont effectué deux expériences de classification sur trois versions du projet Eclipse en utilisant leur méthodologie (dont le but était la prédiction des fautes) : une binaire et une autre ordinale. La classification binaire consiste à prédire si les classes sont ou non sujettes aux erreurs, tandis que l'ordinal essaie de prédire si une classe présente un risque élevé, moyen, faible ou nul [19] en termes d'erreur. Nous nous limitons à la classification binaire en ce qui nous concerne ; à savoir prédire si les classes demandent ou non des efforts de test importants.

L'étude initiale menée par Shatnawi et al. [16] reposant sur la méthodologie binaire n'a pas permis de trouver les valeurs seuils des métriques pour différencier les classes avec erreurs des classes sans erreurs. Nous avons donc voulu vérifier si leur conclusion était également valable pour la prédiction des efforts de test, avec des systèmes logiciels et ensembles de données différents.

### 4.2.1 Description de l'expérimentation

Le but poursuivi dans cette expérimentation est de trouver les valeurs seuils pouvant être utilisées pour la classification des classes qui demandent des efforts de test importants dans les systèmes logiciels orientés objet. À partir des techniques qui permettent de calculer ces valeurs seuils, à savoir les courbes ROC et Alves Rankings, ont été déterminées les métriques valides pour la suite des expérimentations. Le choix de valeurs seuils des métriques à utiliser se fait pour une AUC égale ou supérieure à 60% pour la méthode des courbes ROC et à différents niveaux de pourcentage d'extraction, 30%, 70%, 80% et 90% pour la méthode Alves Rankings.

Cette expérimentation permet d'identifier les valeurs seuils des métriques par analyse d'association entre métriques et effort de test à partir des données des systèmes présentés dans la section 3.3 du troisième chapitre.

Avant de procéder à l'analyse des expérimentations, nous présentons quelques éléments importants en relation avec les efforts de test. Il s'agit notamment des métriques de cas de tests unitaires, de la catégorisation des classes en fonction de l'effort de test unitaire impliqué ainsi que la classification binaire.

#### Métriques de cas de tests unitaires

JUnit (`http://www.junit.org/`) est un cadre logiciel simple permettant d'écrire et d'exécuter des tests unitaires automatisés pour les classes Java.

Pour capturer l'effort de tests unitaires des classes, quatre métriques qui quantifient différentes caractéristiques liées au code des cas de tests unitaires ont été utilisées. L'effort de tests unitaires des classes est abordé du point de vue de construction de cas de tests unitaires [74].

Dans les lignes qui suivent nous présentons ces métriques de cas de tests unitaires.

Deux métriques proposées par Bruntink et Deursen :

- TLOC : Cette métrique donne le nombre de lignes de code d'une classe de test [32, 38]. Elle est utilisée pour indiquer une perspective de la taille de la classe de test [74].

- TASSERT : Cette métrique indique le nombre d'appels de méthodes d'assertions JUnit qui se produisent dans le code d'une classe de test [32, 38]. Les appels d'assertions JUnit sont en fait utilisés par les testeurs pour comparer le comportement attendu de la classe sous test à son comportement actuel. Cette métrique est utilisée pour indiquer une autre perspective de la taille d'une classe de test. Il est directement lié à la construction de cas de test.

Cependant, ces deux métriques ne capturent pas certaines caractéristiques liées à l'interaction entre les classes et/ou à la création d'objets [74].

Les auteurs ont voulu démontrer que ces métriques de cas de tests unitaires reflètent les différents facteurs de code source [74]. Ces facteurs sont :

- Ceux qui influencent le nombre des cas de tests unitaires ;
- Ceux qui influencent l'effort nécessaire au développement de chaque cas de test unitaire.

Ces deux catégories ont été désignées sous le nom de facteurs de génération de tests et de construction de tests [74].

Mourad Badri, Fadel Touré et Luc Lamontagne [74] ont aussi utilisé deux métriques de tests unitaires qui quantifient les différentes caractéristiques relatives au code des cas de tests JUnit [75, 76]. Il s'agit de :

- TINVOK : Cette métrique compte le nombre d'appels de méthodes dans une classe de test. Elle capture les dépendances nécessaires à l'exécution de la classe de test [74].
- TDATA : Cette métrique indique le nombre de nouveaux objets Java créés dans une classe de test. Ces données sont nécessaires pour initialiser le test [74].

Ces deux métriques capturent certaines dimensions non prises en compte par TLOC et TASSERT, qui affectent selon l'opinion des auteurs [74], l'effort nécessaire au développement de chaque cas de test individuel.

Ces quatre métriques ont été utilisées pour collecter les données dans les huit systèmes logiciels Java open source [74]. L'analyse par composant principal (ACP), qui est une technique largement utilisée en génie logiciel a été appliquée afin d'identifier les dimensions sous-jacentes importantes capturées par la suite de métriques de cas de tests unitaires [76].

### Catégorisation des classes en fonction de l'effort de test unitaire impliqué

Différents moyens ont été étudiés pour déterminer les efforts requis pour les tests unitaires par les classes logicielles. Pour ce faire, les classes logicielles ont été catégoriser en cinq groupes selon l'effort de test requis (du bas niveau d'effort au plus élevé) [74]. Cette catégorisation se fait comme suit :

- Les classes qui ont nécessité un effort de test unitaire (relativement) très faible ;
- Les classes qui ont nécessité un effort de test unitaire (relativement) faible ;
- Les classes qui ont nécessité un effort de test unitaire (relativement) moyen ;
- Les classes qui ont nécessité un effort de test unitaire (relativement) important ;
- et enfin les classes qui ont nécessité un effort de test unitaire (relativement) très élevé.

Ces catégories ont été obtenues à l'aide d'approches de classification fondées sur les moyennes et les K-moyennes [74]. Ci-dessous, nous présentons ces deux approches.

- **Catégorisation basée sur les moyennes** : L'approche basée sur les moyennes consiste pour chaque projet, à calculer les différentes valeurs moyennes des métriques de test unitaire et à les regrouper suivant cinq catégories comme suit [74] :
  - Catégorie 5 : La valeur 5 (niveau très élevé) est assignée aux classes correspondantes. Cette catégorie comprend les cas de test JUnit pour lesquels les quatre conditions suivantes sont remplies : (1) grand nombre de lignes de code (TLOC correspondant $\geq$ valeur moyenne de TLOC) et (2) grand nombre d'instructions d'assertions (TASSERT $\geq$ valeur moyenne de TASSERT), et (3) grand nombre de créations de données (TDATA $\geq$ valeur moyenne de TDATA) et (4) grand nombre d'appels (TINVOK $\geq$ valeur moyenne de TINVOK).
  - Catégorie 4 : La valeur 4 (niveau élevé) est assignée aux classes dont les tests JUnit vérifient seulement trois des conditions énumérées dans la catégorie 5.
  - Catégorie 3 : La valeur 3 (niveau moyen) est assignée aux classes dont les tests JUnit vérifient seulement deux des conditions énumérées dans la catégorie 5.
  - Catégorie 2 : La valeur 2 (niveau bas) est assignée aux classes dont les tests JUnit vérifient seulement une des conditions énumérées dans la catégorie 5.
  - Catégorie 1 : La valeur 1(niveau très bas) est assignée aux classes dont les tests JUnit ne vérifient aucune des conditions énumérées dans la catégorie 5.
- **Catégorisation basée sur les K-moyennes** : La technique de regroupement basée sur les K-moyennes a été utilisée pour classer les classes testées en cinq catégories sur la base de quatre métriques de cas de test (prises ensemble) [74]. K-moyennes est une méthode de regroupement qui vise à partitionner $n$ observations (classes de logiciels dans notre cas) en $k$ grappes ($k$=5 dans notre cas) dans lesquelles chaque observation appartient au groupe dont la moyenne est la plus proche [74].

  L'approche de K-moyennes offre en effet un moyen naturel d'identifier des grappes d'objets liés en fonction de leur similarité. Dans notre cas, la similarité est basée sur les valeurs des métriques de suite de tests [74]. Les clusters ou regroupements résultants doivent être construits de manière que les classes testées au sein de chaque cluster soient plus étroitement liées que les classes testées affectées à différents clusters [74].

### Classification binaire BIN-MEAN et BIN-KM5

Dans cette section nous expliquons comment les différentes classifications BIN-MEAN et BIN-KM5 sont obtenues.

Mourad Badri, Linda Badri et Fadel Touré ont présenté une nouvelle métrique capturant de manière unifiée plusieurs aspects de la qualité des systèmes orientés objet [77]. Elle capture entre autres de manière intégrée, la complexité et le couplage (interaction entre classes) [77]. La métrique qu'ils ont appelée *Indicateur de Qualité (Qi)* utilise des chemins de flux de contrôle et des probabilités [77]. La métrique *Qi* est basée sur les graphes d'appels de contrôle, qui sont une forme réduite des graphes de flux de contrôle traditionnels [77]. Un graphe d'appels de contrôle est un graphe de flux de contrôle duquel sont supprimés les nœuds représentant des instructions, ou des blocs élémentaires d'instructions séquentielles, ne contenant pas d'appel à une méthode [77]. La métrique *Qi* est normalisée et donne des valeurs dans l'intervalle [0, 1] [77].

Cette métrique a été utilisée dans plusieurs études sur les tests logiciels [77–79], et en particulier elle a été utilisée dans l'évaluation de l'effet des métriques de code source sur l'effort de test unitaire des classes [74]. Selon Gupta et *al.* [80], aucune des métriques orientées objet ne suffit à elle seule pour donner une idée globale de la testabilité du logiciel [77]. C'est dans ce contexte que la métrique *Qi* a fait partie avec les métriques de code source, à l'évaluation de la testabilité unitaire des classes logicielles en termes d'effort impliqué dans l'écriture du code des cas de tests unitaires. Afin de compléter cette évaluation, une étude empirique a été conduite en utilisant trois méthodes [74] : la méthode de régression logistique univariée, la méthode de régression logistique linéaire univariée ainsi que la méthode de régression logistique multinomiale.

- **Classification avec la régression logistique univariée**

  Il est question ici d'explorer la capacité des métriques logicielles à ne prédire que deux niveaux d'effort de test unitaire : élevé et faible [74]. Les cinq catégories identifiées dans la section précédentes ont été utilisées, selon les deux approches (basée sur les moyennes et les K-moyennes), pour déterminer la valeur de l'effort de test unitaire. Les catégories 5 et 4 ont été réunies en une seule catégorie. À cette nouvelle catégorie a été assigné la valeur 1 (correspondant aux classes nécessitant un effort de test unitaire relativement élevé). Les catégories 3, 2 et 1 ont été réunies en seule autre catégorie, à laquelle la valeur 0 (correspondant aux classes nécessitant un effort de test unitaire relativement faible) a été assignée.

  L'analyse de régression univariée est un cas particulier de l'analyse de régression multivariée dans laquelle il n'y a qu'une seule variable indépendante (une métrique de code source). La p-valeur, liée à l'hypothèse statistique, est la probabilité que le coefficient soit différent de zéro par chance. C'est aussi un indicateur de la précision de l'estimation du coefficient [74]. Pour décider si une métrique est un prédicteur statistiquement significatif de l'effort de test unitaire, la p-valeur a été évaluée

et comparée au niveau de signification $\alpha = 5\%$ [74]. L'évaluation des modèles de prédiction a été faite à l'aide de l'analyse des courbes ROC, en utilisant précisément l'AUC [74].

Cette méthode a été utilisée pour explorer de manière empirique la relation entre les métriques de code source (comme variables indépendantes) et la testabilité unitaire des classes en termes d'effort de test unitaire (comme variables dépendantes) [74]. Il est ici question d'explorer la faculté des métriques logicielles à prédire seulement deux niveaux d'effort de test : élevé (effort de test relativement important) et faible (effort de test relativement faible). Les cinq catégories identifiées dans la section précédente ont été utilisées, selon les deux approches (moyennes et K-moyennes), pour déterminer la valeur de la variable dépendante.

- **Classification avec la régression logistique linéaire univariée**

  La régression linéaire est une technique statistique couramment utilisée pour modéliser la relation entre une variable dépendante et une ou plusieurs variables explicatives [74]. Elle est utilisée pour étudier la relation entre les métriques de code source (et de test) [74]. Les catégories basées sur le regroupement de K-moyennes sont ordonnées, comparées aux catégories basées sur l'approche de moyennes. En effet, les valeurs moyennes de toutes les métriques augmentent de la catégorie 1 à la catégorie 5, ce qui rend la technique de régression linéaire appropriée pour ces données [74].

  Les cinq catégories identifiées lors de la classification K-moyennes ont été utilisées pour déterminer la valeur de la variable dépendante (niveau d'effort de test unitaire) [74] : de 1 à 5. Aux classes de la catégorie 1 la valeur 1 a été affectée, aux classes de la catégorie 2 la valeur 2, et ainsi de suite.

- **Classification avec régression logistique multinomiale**

  Pour le regroupement basé sur les moyennes, la première et dernière catégorie peuvent être clairement ordonnées. Cependant, les catégories intermédiaires (2, 3 et 4) ne sont pas ordonnées. Cette remarque rend la régression logistique multinomiale appropriée pour les observations basées sur les moyennes [74]. Cette technique est utilisée pour explorer de manière empirique la relation entre les métriques de code source (en tant que variables indépendantes) et les niveaux de testabilité unitaire des classes (en tant que variable dépendante) [74].

  La régression logistique multinomiale permet de comparer chaque catégorie d'une variable de réponse non ordonnée à une catégorie de référence, fournissant ainsi un certain nombre de modèles de régression logistique [74]. Cette procédure génère un certain nombre de modèles de régression logistique permettant des comparaisons spécifiques des catégories de réponses.

Il convient de rappeler ici qu'avec la méthode de régression logistique univariée, l'étude a testé diverses hypothèses reliant les métriques de code source à l'effort de test unitaire (relativement important et relativement faible). Cependant, les deux autres méthodes ont testé diverses hypothèses liées à la relation entre les métriques de code source et le niveau d'effort de test unitaire (ce qui ne fait pas l'objet de notre présente étude). Cette méthode a été largement appliquée dans des études empiriques en génie logiciel pour la prédiction de classes sujettes aux fautes [8, 12, 74, 81–84].

Eu égard à ce qui précède, on peut donc constater que les notations BIN-MEAN et BIN-KM5 utilisées dans le présent travail font référence respectivement à la classification avec les approches basées sur les moyennes et sur les K-moyennes. Ces deux métriques d'effort de test (variables dépendantes) sont utilisées dans la construction de nos modèles de prédiction ensemble avec les métriques de code source (variables indépendantes) selon les deux approches définies à la section 3.5.

Ces deux métriques d'effort de test BIN-KM5 et BIN-MEAN ont été déjà calculées et fournies dans les données utilisées dans notre étude.

### 4.2.2 L'analyse des courbes ROC et identification des valeurs seuils

Pour identifier pour chaque métrique les possibles valeurs seuils, nous avons effectué le test t bilatéral pour trouver la signification de la différence entre une courbe et la courbe aléatoire (AUC = 0,5) à un niveau de signification de 95% [15]. Toute métrique ayant un seuil d'erreur supérieur à 0.05 ne sera pas considérée comme valide pour la construction des modèles de prédiction, car l'intervalle de confiance est fixé à 95%. Pour les courbes différant de manière significative des estimations aléatoires, nous avons identifié des valeurs seuils basées à la fois sur la sensibilité (c'est-à-dire les avantages) et sur la spécificité (c'est-à-dire les coûts) [15]. L'AUC est ainsi calculée et la p-valeur indique s'il existe une différence significative par rapport à un classificateur aléatoire (AUC = 0,5).

Shatnawi [15] a posé une hypothèse de base dans la détermination des valeurs seuils valides :

- $H_{o1}$ : *L'AUC pour une métrique particulière est égale à 0.5. L'AUC doit être sensiblement supérieure à la diagonale pour que la courbe puisse être utilisée pour identifier un seuil.*

Si l'AUC est significativement différente de la prédiction aléatoire (c'est-à-dire, rejet de $H_{o1}$), alors les courbes ROC sont utilisables et nous continuons la recherche de valeurs seuils pratiques, sinon, l'identification du seuil pour la métrique s'arrête à ce stade. Si la p-valeur est inférieure au seuil de signification ($\alpha$= 0.05), nous rejetons l'hypothèse nulle [15].

Pour identifier une valeur seuil pratique, la performance des valeurs de l'AUC doit être supérieure à 0.60.

### 4.2.3 Rappel des questions de recherche

Les résultats obtenus dans ces expérimentations nous permettent de répondre à la question de recherche 1, à savoir :

**QR1** : Comment déterminer de façon objective des seuils permettant de prédire l'effort de test ?

Le but poursuivi étant d'éviter de déterminer des valeurs seuils d'une manière empirique ou selon l'opinion des experts, mais plutôt d'appliquer différentes techniques pour calculer objectivement ces valeurs. De plus, des seuils sont dérivés dans ce travail à partir de la relation avec l'effort de test requis pour les systèmes en étude.

### 4.2.4 Résultats et discussion

Dans les tables 4.1, 4.2 et 4.3 nous présentons les valeurs seuils pour toutes les métriques. Ces valeurs sont obtenues par l'application de la méthode des courbes ROC proposée par Shatnawi [15]. Dans la suite de nos expérimentations, nous utiliserons les valeurs seuils sélectionnées pour les différentes classifications.

#### Résultats de valeurs seuils dérivées de la métrique d'effort BIN-KM5

| | ANT | | | POI | | | IO | | |
|---|---|---|---|---|---|---|---|---|---|
| Métriques | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur |
| CBO | **0.714** | **6** | **<0.001** | **0.658** | **8** | **0.003** | **0.842** | **4** | **<0.001** |
| DIT | 0.525 | 2 | 0.679 | **0.653** | **≤2** | **<0.001** | **0.747** | **≤ 1** | **<0.001** |
| LCOM | **0.778** | **58** | **<0.001** | 0.560 | 178 | 0.56 | **0.846** | **11** | **<0.001** |
| LOC | **0.828** | **68** | **<0.001** | **0.74** | **91** | **<0.001** | **0.916** | **59** | **<0.001** |
| NOC | 0.52 | 1 | 0.607 | 0.521 | 0 | 0.387 | 0.57 | 0 | 0.228 |
| RFC | **0.698** | **51** | **<0.001** | **0.633** | **56** | **0.017** | **0.768** | **38** | **<0.001** |
| WMC | **0.828** | **15** | **<0.001** | **0.735** | **49** | **<0.001** | **0.935** | **11** | **<0.001** |

TABLE 4.1 – Valeurs seuils calculées en utilisant la méthode des courbes ROC pour les systèmes ANT, POI et IO. (Métrique d'effort BIN-KM5)

| | IVY | | | JFC | | | JODA | | |
|---|---|---|---|---|---|---|---|---|---|
| Métriques | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur |
| CBO | **0.633** | **12** | **0.03** | **0.662** | **11** | **<0.001** | **0.806** | **6** | **<0.001** |
| DIT | **0.719** | **3** | **<0.001** | 0.529 | 1 | 0.477 | 0.568 | 2 | 0.358 |
| LCOM | **0.664** | **64** | **0.017** | **0.699** | **46** | **<0.001** | **0.925** | **178** | **<0.001** |
| LOC | **0.672** | **228** | **0.009** | **0.782** | **110** | **<0.001** | **0.912** | **246** | **<0.001** |
| NOC | 0.517 | 1 | 0.641 | 0.562 | 1 | 0.057 | 0.564 | **≤ 0** | 0.057 |
| RFC | **0.779** | **104** | **<0.001** | **0.62** | **50** | **0.002** | **0.763** | **87** | **<0.001** |
| WMC | **0.662** | **41** | **0.012** | **0.786** | **53** | **<0.001** | **0.945** | **56** | **<0.001** |

TABLE 4.2 – Valeurs seuils calculées en utilisant la méthode des courbes ROC pour les systèmes IVY, JFC et JODA. (Métrique d'effort BIN-KM5)

| | MATH | | | LUCENE | | |
|---|---|---|---|---|---|---|
| Métriques | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur |
| CBO | **0.752** | **2** | **<0.001** | 0.604 | 9 | 0.157 |
| DIT | **0.763** | **≤2** | **<0.001** | 0.541 | 2 | 0.565 |
| LCOM | 0.634 | 19 | 0.101 | 0.597 | 27 | 0.185 |
| LOC | **0.864** | **75** | **<0.001** | **0.676** | **309** | **0.018** |
| NOC | 0.577 | ≤ 0 | 0.056 | 0.576 | 2 | 0.245 |
| RFC | **0.817** | **45** | **<0.001** | **0.679** | **55** | **0.006** |
| WMC | **0.854** | **14** | **<0.001** | **0.646** | **30** | **0.04** |

TABLE 4.3 – Valeurs seuils calculées en utilisant la méthode des courbes ROC pour les systèmes MATH et LUCENE. (Métrique d'effort BIN-KM5)

Les différents résultats présentés dans les tables 4.1, 4.2 et 4.3 nous aident à déterminer pour les métriques de chaque système les valeurs seuils qui permettent d'avoir une précision, qui permet de prédire les efforts de test. On peut donc constater que les métriques LOC et WMC du système IO et la métrique WMC du système JODA présentent des classifications exceptionnelles. Il est à noter aussi les mauvaises classifications de la métrique NOC pour tous les systèmes. Dans les lignes qui suivent, nous discutons en détail chaque métrique.

- Métrique CBO

Les résultats de l'analyse des courbes ROC pour la métrique CBO présentées dans les tables 4.1, 4.2 et 4.3 montrent une valeur de l'AUC supérieur à 0.60 pour l'ensemble des systèmes. La p-valeur est inférieure au seuil de signification dans la plupart des systèmes, à l'exception du système LUCENE où sa valeur est supérieure à 0.05. De ce fait, nous rejetons l'hypothèse nulle. Bien que la valeur de l'AUC pour la métrique CBO soit supérieure à 0.60 cette métrique ne peut fournir une valeur seuil valide à cause de sa p-valeur qui produit un AUC aléatoire.

Les valeurs seuils trouvées pour cette métrique devraient nous permettre de scinder les classes en deux niveaux, les classes avec un couplage moins élevé et les classes avec un couplage plus élevé. Les classes ayant un couplage plus élevé nécessiteront un effort de test important et inversement dans le cas contraire.

Pour la prédiction des fautes logicielles, McCabe a suggéré un seuil de CBO = 6 [15, 85]. Autrement dit, un module couplé à plus de six autres modules est identifié comme sujet aux fautes. Shatnawi et al. [15, 16] ont trouvé un seuil de CB = 13 en utilisant l'analyse des courbes ROC. Shatnawi [15] a trouvé les valeurs seuils de CBO dans l'intervalle de 6 à 11. Rosenberg a préféré un seuil de CBO = 5 [15, 86]. Dans nos expérimentations, nous avons trouvé les valeurs seuils dans un intervalle de 2 à 12. Dans notre contexte, les modules dont la métrique CBO a un seuil supérieur à celui défini dans cet intervalle nécessiteront des efforts de test importants. Les seuils suggérés dans les travaux de McCabe et Rosenberg tombent dans l'intervalle des seuils présentés dans le présent travail.

- Métrique DIT

Pour la métrique DIT, la valeur de l'AUC est significative pour les systèmes POI, IO, IVY et MATH. Cependant, les systèmes ANT, JFC, JODA et LUCENE présentent une valeur de l'AUC inférieure à 0.6 comme présentée dans les tables 4.1, 4.2 et 4.3. 4.3. Cette métrique ne peut donc pas fournir des valeurs seuils valides pour ces systèmes.

La p-valeur de DIT dans les systèmes ANT, JFC, JODA et LUCENE est supérieure au seuil de signification, nous devrions accepter l'hypothèse nulle $H_{o1}$ et rejeter l'hypothèse alternative $H_{a1}$. Donc, les courbes ROC produites à partir de la métrique DIT pour ces derniers systèmes ne peuvent être utilisées pour classifier les logiciels en termes d'effort de test. Aussi, les valeurs seuils de ces systèmes ne peuvent être retenues pour la métrique DIT.

Rosenberg et al. [15, 86] ont défini les seuils pour la métrique DIT de cette manière : $2 < \text{DIT} < 5$. Les classes ayant la valeur de $\text{DIT} < 2$ peuvent représenter une mauvaise exploitation de l'héritage, alors que les classes ayant une valeur de $\text{DIT} > 5$ ont une large complexité. McCabe [15, 85] a défini les seuils de cette manière : $2 < \text{DIT} < 6$, c'est-à-dire, $\text{DIT} > 6$ nécessite un effort de test important et $\text{DIT} < 2$ indique une pauvre exploitation de l'héritage.

Nos résultats indiquent que $2 \leq \text{DIT} < 3$, c'est-à-dire $\text{DIT} \leq 2$ indique une pauvre exploitation de l'héritage, et $\text{DIT} > 3$ indique des efforts de test.

- Métrique LCOM

Les résultats de l'analyse des courbes ROC pour la métrique LCOM présentées dans les tables 4.1, 4.2 et 4.3 montrent que les valeurs de l'AUC et de p-valeur sont significatives pour les systèmes ANT, IO, IVY, JFC et JODA. Dans ce cas, on rejette l'hypothèse nulle. Cependant, les systèmes POI, MATH et LUCENE présentent une AUC inférieure à 0.6 et une p-valeur supérieure à 0.05. Donc, nous devrions accepter l'hypothèse nulle pour ces derniers systèmes.

Shatnawi et al. [15, 16] n'ont pas trouvé une valeur seuil pour la métrique LCOM dans leur étude en utilisant l'analyse des courbes ROC. Rosenberg n'a pas non plus fourni des valeurs seuils pour cette métrique [15, 86]. McCabe a quant à lui, suggéré une valeur seuil de LCOM = 75% [15, 85] mais pour une différente définition de la métrique LCOM.

Dans notre étude, les valeurs seuils trouvées sont dans un intervalle de 11 à 178.

- Métrique LOC

La métrique LOC présente de meilleurs résultats de l'AUC et p-valeur qui sont significatifs pour tous les systèmes. Voir les tables 4.1, 4.2 et 4.3. De ce fait, nous rejetons l'hypothèse nulle.

Nos résultats montrent que les valeurs seuils trouvées pour cette métrique se situent entre 59 et 309. Cela dit, des modules ayant une valeur seuil de LOC supérieure aux valeurs seuils identifiées pour ces différents systèmes demandent des efforts importants de test et inversement dans le cas contraire ils nécessitent (dans la relativité) un faible effort de test.

- Métrique NOC

Les résultats de l'analyse des courbes ROC pour la métrique NOC sont illustrés dans les tables 4.1, 4.2 et 4.3. On peut constater que les p-valeurs sont supérieures au seuil de signification. L'hypothèse nulle $H_{o1}$ est acceptée et l'hypothèse alternative $H_{a1}$ est rejetée. L'étude de McCabe et Rosenberg n'a suggéré aucune valeur seuil pour NOC [15, 85, 86]. Shatnawi et al. n'ont pas trouvé non plus un seuil pour la métrique NOC en utilisant l'analyse des courbes ROC [15, 16].

Donc, toutes les courbes ROC produites par NOC ne peuvent être utilisées pour classifier les modules des systèmes en termes d'efforts de test, et les valeurs seuils ne peuvent être déterminées pour cette métrique.

- Métrique RFC

Les résultats de la métrique RFC présentés dans les tables 4.1, 4.2 et 4.3 montrent une p-valeur inférieure au seuil de signification $\alpha = 0.05$. Dans ce cas, on rejette l'hypothèse nulle $H_{o1}$. Les valeurs de l'AUC sont significativement différentes du classifieur aléatoire pour tous les systèmes. Les valeurs seuils calculées pour la métrique RFC sont dans un large intervalle de 38 à 104.

Ces différents seuils permettent de décrire les modules des systèmes à deux niveaux : les modules à faibles responsabilités qui demandent des efforts de test moins importants et les modules à hautes responsabilités qui demandent des efforts de test importants. McCabe a suggéré une valeur seuil de RFC = 40 [15, 85]. Shatnawi et al. [15, 16] ont trouvé un seuil de RFC = 44 pour le système Eclipse en utilisant l'analyse des courbes ROC. Rosenberg a quant à lui préféré un seuil de RFC = 50 [15, 86]. Ces différentes valeurs seuils tombent dans l'intervalle des valeurs seuils trouvées dans notre étude.

- Métrique WMC

Les résultats de l'analyse des courbes ROC pour la métrique WMC telles qu'illustrés dans les tables 4.1, 4.2 et 4.3 montrent les courbes qui sont significativement différentes du classifieur aléatoire. Qu'il s'agisse de l'AUC calculée pour chaque système ou de la p-valeur, les résultats montrent une différence significative par rapport au classifieur aléatoire (AUC = 0.5). Avec une p-valeur inférieure au seuil de signification ($\alpha = 0.05$), on rejette l'hypothèse nulle $H_{o1}$. La performance des valeurs de l'AUC étant supérieure à 0.6, les valeurs seuils valides ont été identifiées.

Les valeurs seuils calculées dans ce présent travail sont comprises dans un large intervalle de 11 à 56. Les seuils peuvent permettre de décrire les modules en deux catégories : les modules à faible risque et les modules à haut risque. Les modules à faible risque ne nécessitent pas d'efforts de test importants. Cependant, les modules à haut risque nécessitent des efforts de test importants.

McCabe a suggéré un seuil de WMC = 10 [15, 85]. Shatnawi et al [15, 16] ont trouvé une valeur seuil de WMC = 9 pour un grand système *open-source* en utilisant l'analyse des courbes ROC. Rosenberg a préféré une autre valeur seuil possible à WMC = 20 ou WMC = 40 qui peut être acceptable aussi [15, 86], bien qu'ils aient trouvé que la plupart des classes ont des valeurs inférieures à 20.

### Résultats de valeurs seuils basées sur la métrique d'effort BIN-MEAN

| | ANT | | | POI | | | IO | | |
|---|---|---|---|---|---|---|---|---|---|
| Métriques | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur |
| CBO | **0.681** | 6 | **0.001** | **0.726** | **9** | **<0.001** | **0.89** | **3** | **<0.001** |
| DIT | 0.579 | 2 | 0.2 | **0.619** | **<=2** | **<0.001** | **0.764** | **<=1** | **<0.001** |
| LCOM | **0.71** | **112** | **<0.001** | 0.53 | 172 | 0.456 | 0.906 | 11 | <0.001 |
| LOC | **0.783** | **68** | **<0.001** | **0.725** | **101** | **<0.001** | **0.923** | **84** | **<0.001** |
| NOC | 0.514 | 3 | 0.729 | 0.518 | 0 | 0.299 | 0.542 | 2 | <0.001 |
| RFC | **0.69** | **51** | **<0.001** | **0.709** | **54** | **<0.001** | **0.814** | **40** | **<0.001** |
| WMC | **0.71** | **112** | **<0.001** | **0.745** | **18** | **<0.001** | **0.92** | **21** | **<0.001** |

TABLE 4.4 – Valeurs seuils calculées en utilisant la méthode des courbes ROC pour les systèmes ANT, POI et IO. (Métrique d'effort BIN-MEAN)

| | IVY | | | JFC | | | JODA | | |
|---|---|---|---|---|---|---|---|---|---|
| Métriques | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur |
| CBO | **0.681** | **15** | **0.005** | **0.678** | **27** | **<0.001** | **0.786** | **13** | **<0.001** |
| DIT | **0.753** | **3** | **<0.001** | 0.557 | 1 | 0.148 | **0.745** | **2** | **<0.001** |
| LCOM | **0.695** | **64** | **0.013** | **0.747** | **42** | **<0.001** | **0.931** | **178** | **<0.001** |
| LOC | **0.738** | **228** | **<0.001** | **0.786** | **238** | **<0.001** | **0.841** | **246** | **<0.001** |
| NOC | 0.54 | 1 | 0.386 | 0.56 | 1 | 0.075 | 0.585 | <=0 | 0.001 |
| RFC | **0.86** | **115** | **<0.001** | **0.64** | **53** | **<0.001** | **0.804** | **94** | **<0.001** |
| WMC | **0.718** | **41** | **<0.001** | **0.809** | **41** | **<0.001** | **0.896** | **56** | **<0.001** |

TABLE 4.5 – Valeurs seuils calculées en utilisant la méthode des courbes ROC pour les systèmes IVY, JFC et JODA. (Métrique d'effort BIN-MEAN)

| | MATH | | | LUCENE | | |
|---|---|---|---|---|---|---|
| Métriques | AUC | Seuil | p-valeur | AUC | Seuil | p-valeur |
| CBO | **0.722** | **0** | **0.002** | **0.662** | **9** | **0.015** |
| DIT | **0.773** | **<=2** | **<0.001** | 0.508 | 2 | 0.905 |
| LCOM | **0.685** | **19** | **0.035** | 0.575 | 47 | 0.317 |
| LOC | **0.895** | **75** | **<0.001** | **0.656** | **139** | **0.026** |
| NOC | 0.565 | <=0 | 0.117 | 0.528 | 10 | 0.528 |
| RFC | **0.858** | **45** | **<0.001** | **0.717** | **35** | **<0.001** |
| WMC | **0.902** | **14** | **<0.001** | **0.686** | **31** | **0.005** |

TABLE 4.6 – Valeurs seuils calculées en utilisant la méthode des courbes ROC pour les systèmes MATH et LUCENE. (Métrique d'effort BIN-MEAN)

- Métrique CBO

Les résultats de la métrique CBO sont repris dans les tables 4.4, 4.5 et 4.6. On rejette l'hypothèse nulle puisque la p-valeur pour l'ensemble des systèmes est inférieure à 0.05. Également, l'analyse des courbes ROC a permis de trouver les valeurs seuils valides pour CBO avec une valeur d'AUC supérieure à 0.6 pour l'ensemble des systèmes. Les valeurs seuils sont comprises dans l'intervalle de 0 à 15. Comparativement, les valeurs seuils trouvées avec la métrique d'effort BIN-KM5 pour cette métrique sont comprises dans cet intervalle (2 à 12).

- Métrique DIT

La métrique DIT telle qu'illustrée dans les tables 4.4, 4.5 et 4.6, donne des valeurs seuils valides pour la plupart des systèmes. À l'exception des systèmes ANT, JFC et LUCENE pour lesquels les valeurs respectives de l'AUC sont inférieures à 0.6 et les p-valeurs respectives sont supérieures au seuil de signification ($\alpha = 0.05$). Nos résultats indiquent que $1 \leq \text{DIT} \leq 3$, comparativement aux seuils obtenus à partir de la métrique BIN-KM5, qui se situent à $2 \leq \text{DIT} < 3$.

- Métrique LCOM

Les résultats de la métrique LCOM sont repris dans les tables 4.4, 4.5 et 4.6. On y constate que seuls L'AUC et la p-valeur des systèmes POI et LUCENE qui ne sont pas significatifs. LCOM a un intervalle de seuils allant de 11 à 178, équivalent à celui trouvé en utilisant la métrique d'effort BIN-KM5.

- Métrique LOC

Les tables 4.4, 4.5 et 4.6 reprennent les résultats de la métrique LOC. Les résultats sont significatifs pour l'ensemble des systèmes. L'hypothèse nulle est rejetée et les valeurs seuils valides sont dégagées. Les valeurs seuils pour la métrique logicielle LOC sont dans un intervalle de 68 à 246 qui est compris dans l'intervalle de valeurs obtenues en utilisant la métrique d'effort BIN-KM5 (59 à 309).

- Métrique NOC

La métrique NOC ne fournit aucune valeur seuil valide, étant donné que l'AUC est inférieur à 0.6 pour l'ensemble des systèmes et la p-valeur est supérieur à 0.05 pour la plupart des systèmes, à l'exception de IO. Cela entraine l'acceptation de l'hypothèse nulle et le rejet de l'hypothèse alternative pour les sept autres systèmes. Les tables 4.4, 4.5 et 4.6 montrent ces résultats.

- Métrique RFC

Les tables 4.4, 4.5 et 4.6 montrent les résultats de la métrique RFC pour l'ensemble des systèmes. Les valeurs de l'AUC et p-valeur sont significatives pour tous les systèmes. Les valeurs seuils trouvées sont comprises dans l'intervalle de 35 à 115 ; un intervalle plus large que celui de RFC obtenu avec la métrique d'effort BIN-KM5 (38 à 104).

- Métrique WMC

Comme illustré dans les tables 4.4, 4.5 et 4.6, les courbes sont significativement différentes du classifieur aléatoire. Avec une AUC supérieure à 0.6 et une p-valeur inférieure au seuil de signification, les valeurs seuils valides ont été identifiées pour la métrique WMC. Les valeurs seuils trouvées dans le présent travail sont comprises dans un large intervalle de 14 à 112 comparativement aux seuls de WMC avec la métrique d'effort BIN-KM5 qui a un intervalle de seuils compris entre 11 à 56.

### 4.2.5 Alves Rankings et identification des valeurs seuils

Afin d'identifier pour chaque métrique les valeurs seuils valides, nous recourons à la méthode d'Alves Rankings qui se résume à quatre principales étapes décrites dans la section 3.5.2.

#### Résultats de valeurs seuils avec la méthode Alves Rankings

Les résultats obtenus dans cette expérimentation nous permettent de répondre à la question de recherche **QR1** : comment déterminer de façon objective des seuils permettant de prédire l'effort de test. Les tables ci-après illustrent les différentes valeurs seuils extraites à différents niveaux de pourcentage : 30%, 70%, 80% et 90%.

| | ANT | | | | POI | | | |
|---|---|---|---|---|---|---|---|---|
| Métriques | 30% | 70% | 80% | 90% | 30% | 70% | 80% | 90% |
| CBO | 7 | 13 | 15 | 16 | 7 | 44 | 66 | - |
| DIT | - | 2 | - | - | - | 3 | - | - |
| LCOM | 7 | 25 | 35 | 97 | 20 | 605 | 1936 | - |
| LOC | 86 | 244 | 277 | 279 | 137 | 651 | 1134 | - |
| NOC | - | - | - | - | - | - | 1 | - |
| RFC | 56 | 90 | 95 | 98 | 56 | 166 | 324 | - |
| WMC | 16 | 41 | 45 | 51 | 24 | 96 | 187 | - |

TABLE 4.7 – Valeurs seuils calculées en utilisant la méthode Alves Rankings pour les systèmes ANT et POI.

| | IO | | | | IVY | | | |
|---|---|---|---|---|---|---|---|---|
| Métriques | 30% | 70% | 80% | 90% | 30% | 70% | 80% | 90% |
| CBO | 4 | 16 | 24 | - | 17 | 38 | 50 | 65 |
| DIT | - | - | 2 | 3 | - | 3 | - | 4 |
| LCOM | 11 | 621 | 2201 | - | 28 | 467 | 1050 | 2934 |
| LOC | 89 | 490 | 512 | - | 209 | 523 | 724 | 1019 |
| NOC | - | 0 | 1 | 2 | - | - | - | 0 |
| RFC | 43 | 84 | 127 | - | 113 | 187 | 259 | 283 |
| WMC | 1 | 136 | 145 | - | 33 | 82 | 104 | 180 |

TABLE 4.8 – Valeurs seuils calculées en utilisant la méthode Alves Rankings pour les systèmes IO et IVY.

| | JFC | | | | JODA | | | |
|---|---|---|---|---|---|---|---|---|
| Métriques | 30% | 70% | 80% | 90% | 30% | 70% | 80% | 90% |
| CBO | 16 | 40 | 49 | 59 | 40 | 19 | 22 | 23 |
| DIT | - | 3 | - | 4 | 3 | - | 3 | 4 |
| LCOM | 30 | 1037 | 1450 | 4504 | 1037 | 2556 | 2755 | 4470 |
| LOC | 232 | 681 | 853 | 1251 | 681 | 621 | 917 | 1155 |
| NOC | - | 1 | 2 | 8 | 1 | - | 0 | 1 |
| RFC | 110 | 371 | 409 | 432 | 371 | - | 231 | 244 |
| WMC | 44 | 127 | 149 | 216 | 127 | 110 | 119 | 149 |

TABLE 4.9 – Valeurs seuils calculées en utilisant la méthode Alves Rankings pour les systèmes JFC et JODA.

| | MATH | | | | LUCENE | | | |
|---|---|---|---|---|---|---|---|---|
| Métriques | 30% | 70% | 80% | 90% | 30% | 70% | 80% | 90% |
| CBO | 2 | 4 | 6 | 8 | 8 | 33 | 48 | 4 |
| DIT | - | - | 2 | 3 | - | - | 2 | - |
| LCOM | 0 | 219 | 366 | 501 | 1 | 206 | 1246 | 219 |
| LOC | 86 | 238 | 533 | - | 189 | 945 | 1955 | 238 |
| NOC | - | - | - | 0 | 0 | 2 | 7 | - |
| RFC | 43 | 68 | 92 | - | 52 | 151 | 284 | 68 |
| WMC | 17 | 56 | 139 | - | 28 | 104 | 141 | 56 |

TABLE 4.10 – Valeurs seuils calculées en utilisant la méthode Alves Rankings pour les systèmes MATH et LUCENE.

Les tables 4.7, 4.8, 4.9 et 4.10 montrent les différentes valeurs seuils obtenues pour chacun des systèmes. Au niveau d'extraction de 90%, aucune valeur seuil n'a pu être obtenue pour toutes les métriques logicielles pour le système POI. Cependant, la métrique NOC dans la table 4.7 ne fournit aucune valeur pour le système ANT. Pour le système POI elle ne fournit qu'une valeur à 80%. Dans la table 4.8 elle ne fournit qu'une valeur à 90% pour le système IVY. Dans la table 4.10 elle ne fournit qu'une valeur à 90% pour le système MATH. La métrique DIT dans la table 4.7 ne fournit qu'une valeur à 70% pour les systèmes ANT et POI. Et dans la table 4.10 elle ne fournit qu'une valeur à 80% pour le système LUCENE.

Nous allons à présent voir en détail les résultats relatifs aux différentes métriques.

- Métrique CBO

À 30%, on peut s'apercevoir que la métrique CBO fournit des valeurs seuils pour tous les systèmes. Avec une valeur de 40, le système JODA donne la valeur seuil la plus élevée contrairement au système MATH avec une valeur seuil de 2. À 70% la métrique CBO donne des valeurs seuils comprises entre 4 pour le système MATH et 44 pour le

système POI. À 80%, la métrique CBO donne des valeurs seuils comprises entre 6 pour le système MATH et 66 pour le système POI. Et à 90% elle donne des valeurs seuils dans l'intervalle de 4 pour le système LUCENE et 65 pour le système IVY. Cependant, les systèmes POI et IO ne fournissent pas des valeurs seuils à 90%.

- Métrique DIT

À l'exception du système JODA avec une valeur seuil de 3, la métrique ne fournit aucune autre valeur seuil valide pour le reste des systèmes à 30%. À 70%, le système ANT a une valeur seuil de 2. POI, IVY et JFC ont une valeur seuil de 3. Cependant, les systèmes IO, JODA, MATH et LUCENE ne fournissent pas des valeurs seuils. À 80% les systèmes IO, MATH, LUCENE donnent une valeur seuil de 2 et JODA donne une valeur seuil de 3 ; les autres systèmes n'en donnent pas. À 90%, les systèmes IO et MATH donnent des valeurs seuils égalent à 3. Les systèmes IVY, JFC et JODA donnent des valeurs seuils égales à 4.

- Métrique LCOM

À 30%, on peut constater l'écart important entre les valeurs seuils des différents systèmes. Le système MATH par exemple a une valeur seuil de 0 tandis que le système JODA a une valeur seuil de 1037. À 70%, le même constat est fait avec des valeurs seuils comprises dans l'intervalle de 25 pour le système ANT et 2556 pour JODA. À 80%, la métrique LCOM donne des valeurs seuils comprises entre 35 pour ANT et 2755 pour JODA. Enfin à 90% les valeurs seuils sont comprises entre 97 pour ANT et 4504 pour JFC. Cependant, les systèmes POI et IO ne fournissent pas des valeurs seuils à ce niveau de pourcentage.

- Métrique LOC

À 30%, la métrique LOC présente des valeurs seuils pour tous les systèmes. Ces valeurs sont comprises entre 86 pour les systèmes MATH et ANT et 681 pour JODA. À 70%, les valeurs seuils sont comprises entre 238 pour MATH et 945 pour LUCENE. À 80% le système LUCENE a la valeur seuil la plus grande égale à 1955 et ANT la valeur la plus petite égale à 277. Tous les autres ont des valeurs seuils comprises dans cet intervalle. À 90% avec une valeur seuil de 238, le système LUCENE a la plus petite valeur seuil et la plus grande valeur seuil pour JFC égale à 1251. Les autres systèmes ont des valeurs seuils comprises dans cet intervalle. Cependant, les systèmes MATH, POI et IO ne fournissent aucune valeur seuil à ce niveau de pourcentage d'extraction.

- Métrique NOC

La métrique NOC ne donne aucune valeur seuil pour le système ANT à tous les niveaux d'extraction. À 30% seuls les systèmes JODA et LUCENE donnent des valeurs seuils de 1 et 0 respectivement. À 70% seuls les systèmes IO, JFC et LUCENE donnent des valeurs seuils comprises entre 0 et 2. À 80%, le système LUCENE est celui qui donne la

plus grande valeur seuil égale à 7, tandis que le système JODA donne la valeur seuil égale à 0. Les systèmes POI et IO donnent tous une valeur seuil égale à 1 et enfin JFC donne une valeur seuil égale à 2. À 90%, les systèmes ANT, POI et LUCENE ne donnent aucune valeur seuil à ce niveau d'extraction. Cependant, les autres systèmes donnent des valeurs seuils comprises entre 0 pour IVY et MATH ainsi que 8 pour JFC.

- Métrique RFC

À 30%, tous les systèmes ont donné des valeurs seuils. Les systèmes MATH et IO ont tous la plus petite valeur qui est de 43 et le système JODA la plus grande valeur qui est de 371. Les autres systèmes ont des valeurs seuils comprises dans cet intervalle. À 70%, seul le système JODA qui ne donne aucune valeur seuil. Les autres systèmes donnent des valeurs seuils allant de 68 pour le système MATH à 371 pour le système JFC. À 80% les valeurs seuils sont dans un intervalle de 92 pour le système MATH et 409 pour le système JFC. À 90%, les valeurs seuils sont comprises entre 68 pour le système LUCENE et 432 pour le système JF. Les systèmes POI, IO et MATH ne donnent aucune valeur seuil.

- Métrique WMC

À 30%, tous les systèmes ont donné des valeurs seuils pour la métrique WMC. Ces valeurs seuils sont reparties sur un intervalle allant de 1 pour le système IO à 127 pour le système JODA. À 70% le système IO a la plus grande valeur seuil égale à 136 et ANT la plus petite valeur seuil égale à 41. Les autres systèmes donnent des valeurs seuils comprises dans cet intervalle. À 80% également tous les systèmes donnent des valeurs seuils. Ces valeurs sont comprises entre 45 pour le système ANT et 187 pour le système POI. Enfin à 90% les valeurs seuils sont comprises entre 51 pour le système ANT et 216 pour le système JFC. Cependant, les systèmes POI, IO et MATH ne donnent pas de valeurs seuils.

## 4.3 Prédiction des efforts de test : approche basée sur les seuils des métriques

### 4.3.1 Introduction

Dans cette section, les résultats des différentes expérimentations sont présentés et discutés. Nous traitons principalement de la classification des modèles avec différents algorithmes d'apprentissage automatique, après la binarisation effectuée sur la base des valeurs seuils valides des métriques logicielles. Ces valeurs ont été obtenues en appliquant les techniques de calcul des seuils présentées précédemment.

Nous commençons par un rappel des questions de recherche, suivi : (1) des résultats obtenus selon l'approche supervisée et non supervisée, (2) des résultats obtenus à partir des valeurs brutes des métriques, et (3) d'un condensé des meilleurs modèles.

### 4.3.2 Rappel des questions de recherche

Nos expérimentations nous permettront de répondre aux questions de recherche suivantes :

**QR2 :** Comment utiliser les techniques d'apprentissage pour améliorer les performances des techniques de calcul de seuils ?

**QR3 :** L'approche avec valeurs seuils des métriques est-elle plus efficace que l'approche avec valeurs des métriques brutes ?

**QR4 :** Quelle technique parmi les courbes ROC et Alves Rankings donne de meilleures performances pour la prédiction de l'effort de test ?

### 4.3.3 Résultats basés sur l'approche supervisée

L'analyse des courbes ROC, une méthodologie utilisée par Shatnawi et al. permettant de déterminer les valeurs seuils appliquées à la prédiction binaire (0 pour l'effort de test relativement faible et 1 pour un effort de test relativement important) des efforts de test, ces valeurs ont permis d'obtenir les résultats présentés dans les tables A.1, A.2, A.3, A.4, A.5 pour la métrique d'effort de test BIN-KM5 et B.1, B.2, B.3, B.4 et B.5 pour la métrique d'effort de test BIN-MEAN.

#### Résultats d'algorithmes d'apprentissage automatique pour BIN-KM5

Nos résultats montrent de meilleures performances pour l'algorithme de Forêt aléatoire avec une moyenne de g-mean égale à 0.996 (table A.3), suivis de K plus proches voisins avec une moyenne de g-mean égale à 0.994 (table A.5), de Perceptron multicouche avec une moyenne de g-mean égale à 0.964 (table A.4), de Bayésien naïf avec une moyenne de g-mean égale à 0.941 (table A.1) et enfin l'algorithme de Machine à vecteurs de support avec une moyenne de g-mean acceptable, égale à 0.57 (table A.2). Cette classification acceptable pour l'algorithme de Machine à vecteurs de support se traduit par la valeur élevée de la métrique FNR pour l'ensemble des systèmes. Cela pose un risque, car les concepteurs de logiciels vont consacrer peu d'effort de test aux composants potentiellement sujets aux fautes. Le fait que l'algorithme de Forêt aléatoire donne de meilleures performances, s'explique par le détail que la métrique FPR pour la plupart de systèmes donne des valeurs proches ou égales à 0. Cela permettra d'investir des efforts de test importants sur des classes qui en requièrent probablement. Ce constat peut être fait pour les algorithmes de K plus proches voisins, de Perceptron multicouche et de Bayésien naïf pour lesquels les valeurs g-mean sont élevées et les valeurs FNR faibles de manière générale.

### Résultats d'algorithmes d'apprentissage automatique pour BIN-MEAN

Les résultats obtenus pour les modèles classifiés à l'aide de la métrique d'effort de test BIN-MEAN sont similaires à ceux obtenus pour la métrique BIN-KM5 en ce qui concerne les performances des classifications des différents algorithmes d'apprentissage automatique. Comme on peut l'apercevoir dans la table B.3, la Forêt aléatoire produit la meilleure performance avec une valeur moyenne de g-mean égale à 0.989 et la Machine à vecteurs de support produit une classification faible avec une moyenne de g-mean égale à 0.549 (table B.2).

Les meilleures performances produites par l'algorithme de Forêt aléatoire, se traduisent notamment par la qualité de classifications obtenues pour tous les systèmes. Par exemple, le système MATH pour toutes les métriques ayant des valeurs seuils valides, les valeurs g-mean sont égales à 1. Mais aussi les valeurs de métriques FPR et FNR sont toutes égales à 0. Ceci traduit d'excellentes classifications. Les autres systèmes suivent la même tendance avec des valeurs des g-mean proches de 1 excepté la métrique WMC de ANT qui a une valeur g-mean égale à 0.707. Les autres algorithmes à l'exception de Machine à vecteurs de support donnent presque les mêmes tendances. L'algorithme de Bayésien naïf donne des valeurs des g-mean en moyennes comprises entre 0.779 pour le système ANT et 0.985 pour le système JODA. Pour les K plus proches voisins les valeurs g-mean sont dans l'intervalle de 0.937 pour le système ANT et 1 pour le système POI. L'algorithme de Perceptron multicouche quant à lui a des valeurs moyennes de g-mean comprises entre 0.924 pour le système ANT et 0.987 pour le système JODA. Pour répondre à la question de recherche 2, on peut affirmer que l'application des techniques de calcul de seuils combinées aux algorithmes d'apprentissage automatique a permis d'améliorer considérablement les performances des modèles. Cette amélioration consiste à construire des modèles de prédiction sur base de techniques appropriées (courbes ROC et Alves Rankings). Ce qui nous permet de déterminer de manière objective les valeurs seuils valides qui sont indispensables dans la construction de nos modèles de prédiction. Au bénéfice de cette amélioration liée à l'usage de ces deux techniques de calcul de valeurs seuils, on peut aussi souligner le fait de ne pas recourir (et se fier) à l'opinion des experts dans la manière de définir les différentes valeurs seuils. Il est aussi important de souligner l'apport majeur de ces techniques dans la performance des modèles comparée aux performances produites avec les modèles construits sur base des métriques brutes (sans l'application d'aucune technique de calcul de valeurs seuils). Seul l'algorithme de Machine à vecteurs de support donne des classifications acceptables, en considérant les valeurs g-mean qu'il produit pour l'ensemble des systèmes. Cependant, les autres algorithmes donnent d'excellentes classifications avec des valeurs de g-mean supérieures à 0.9 pour l'ensemble des systèmes.

### 4.3.4 Résultats basés sur l'approche non supervisée

Les résultats obtenus en utilisant la méthode Alves Rankings combinée aux différents algorithmes d'apprentissage automatique sont présentés dans les tables comme suit :

- Pour Le Bayésien naïf : C.1 à 30%, C.2 à 70%, C.3 à 80% et C.4 à 90%.
- Pour La Machine à vecteurs de support : C.5 à 30%, C.6 à 70%, C.3 à 80% et C.8 à 90%.
- Pour la Forêt aléatoire : C.9 à 30%, C.10 à 70%, C.11 à 80% et C.12 à 90%.
- Pour le Perceptron multicouche : C.13 à 30%, C.14 à 70%, C.15 à 80%, C.8 à 90%.
- Pour les K plus proches voisins : C.17 à 30%, C.18 à 70%, C.19 à 80% et C.20 à 90%.

Les conclusions que l'on peut tirer de ces résultats sont largement positives.

Pour le Bayésien naïf, les performances issues des métriques extraites à 30% sont les meilleures avec une moyenne de la valeur g-mean pour l'ensemble des systèmes. Cette moyenne est égale à 0.887 par rapport à celles des métriques extraites à 70% avec une moyenne g-mean de 0.87. Les performances à 30% sont aussi meilleures que celles obtenues à 90% avec une moyenne de g-mean égale à 0.76 et celles obtenues à 80% avec une moyenne g-mean de 0.741.

Le même constat est fait pour l'algorithme de K plus proches voisins avec une moyenne g-mean de 0.996 à 30%, 0.934 à 70%, 0.874 à 90% et 0.811 à 80%. La Forêt aléatoire à 30% donne une moyenne de g-mean égale à 0.996, 0.938 à 70%, 0.791 à 80% et 0.655 à 90%. Cependant, pour la Machine à vecteurs de support les meilleures performances sont obtenues avec des métriques extraites à 90% avec une moyenne de g-mean de 0.815, suivi de 80% avec 0.751, plus loin à 30% avec 0.523 pour finir à 70% avec une mauvaise performance de 0.448 .

Le Perceptron multicouche présente de meilleures performances à 70% avec une moyenne de g-mean de 0.926, suivi de 30% avec 0.911, ensuite de 90% avec 0.815 pour finir par 80% avec 0.786. Les conclusions précédentes concernent les résultats obtenus à partir de la métrique d'effort de test BIN-KM5.

Concernant la métrique d'effort BIN-MEAN, les conclusions que l'on peut tirer sont les suivantes : pour le Bayésien naïf, nous avons de meilleures performances à 30% avec une moyenne de g-mean égale à 0.886, à 70% avec 0.87, à 80% avec 0.713 et à 90% avec 0.704. Pour la Forêt aléatoire à 30% la moyenne de g-mean est de 0.966, à 70% la moyenne est de 0.938, à 90% elle est de 0.843 et à 80% elle est de 0.792.

Les K plus proches voisins donnent aussi de meilleures performances à différents niveaux de pourcentage d'extraction des métriques. À 30% on a une moyenne de g-mean de 0.996, à 70% une moyenne de 0.937, à 90% une moyenne de 0.843, ainsi qu'à 80% une moyenne de 0.811.

Le perceptron multicouche présente aussi de meilleures performances, à 70% nous avons une moyenne de g-mean de 0.923, à 30% nous avons 0.911, à 90% la moyenne est de 0.819 et à 80% elle est de 0.804.

Pour l'algorithme de Machine à vecteurs de support, nous avons une seule performance acceptable à 30% de 0.545, suivie de mauvaises performances à 70% de 0.427, à 90% de 0.308 et à 80% de 0.265.

Pour la métrique d'effort BIN-KM5 (obtenue par la classification basée sur l'approche des K-moyennes), les algorithmes de Forêt aléatoire et de K plus proches voisins donnent les meilleures performances à 30%, l'algorithme de Forêt aléatoire donne les meilleures performances à 70% et 80% et à 90% les K plus proches voisins sont ceux qui donne les meilleures performances.

Pour la métrique d'effort BIN-MEAN (obtenue par la classification basée sur l'approche des moyennes), les algorithmes de Forêt aléatoire et de K plus proches voisins donnent les meilleures performances à 30%, l'algorithme de Forêt aléatoire donne les meilleures performances à 70%, l'algorithme de K plus proches voisins donne les meilleures performances à 80%. Les algorithmes de Forêt aléatoire et de K plus proches voisins donnent les meilleures performances à 90%.

### 4.3.5 Résultats basés sur les valeurs brutes des métriques

Les résultats des prédictions de l'effort de test avec les valeurs brutes des métriques pour les métriques d'effort de test BIN-KM5 et BIN-MEAN sont présentés dans les tables suivantes pour chaque algorithme d'apprentissage automatique :

- Bayésien naïf à la table D.1 pour BIN-KM5 et E.1 pour BIN-MEAN
- Machine à vecteurs de support à la table D.2 pour BIN-KM5 et E.2 pour BIN-MEAN
- Forêt aléatoire à la table D.3 pour BIN-KM5 et E.3 pour BIN-MEAN
- Perceptron multicouche à la table D.4 pour BIN-KM5 et E.4 pour BIN-MEAN
- K plus proches voisins à la table D.5 pour BIN-KM5 et E.5 pour BIN-MEAN

Au regard de ces résultats, nous pouvons constater que pour la métrique d'effort de test BIN-KM5, tous les algorithmes d'apprentissage automatique appliqués sur l'ensemble des systèmes présentent de mauvaises performances. La Machine à vecteurs de support affiche une moyenne de g-mean égale à 0.098, la Forêt aléatoire donne une moyenne g-mean de 0.342, et le Perceptron multicouche 0.365. Le Bayésien naïf 0.370, et l'algorithme de K plus proches voisins est celui qui donne une moyenne g-mean assez intéressante soit 0.435.

Pour la métrique BIN-MEAN, les résultats sont quasi similaires à ceux de la métrique BIN-KM5 avec une moyenne de g-mean égal à 0.09 pour la Machine à vecteurs de support, 0.307 pour le Bayésien naïf, 0.31 pour le Perceptron multicouche, 0.398 pour les K plus proches voisins et 0.412 pour la Forêt aléatoire.

Ces moyennes sont toutes inférieures à 0.5, ce qui signifie que tous les algorithmes ont donné de mauvaises classifications en considérant la moyenne des valeurs g-mean de tous les systèmes.

Cependant, certains systèmes considérés isolément donnent de bonnes classifications pour certains algorithmes. Il s'agit entre autres pour la métrique d'effort de test BIN-KM5, des systèmes IO et MATH pour l'algorithme Bayésien naïf ; IO, JODA et MATH pour la Forêt aléatoire ; JODA, IO et MATH pour le Perceptron multicouche ; IO et JODA pour les K plus proches voisins. Cependant, aucun système ne donne une moyenne g-mean supérieure ou égale à 0.5 pour la Machine à vecteurs de support, seule la métrique WMC du système de JODA donne une classification acceptable.

Pour la métrique BIN-MEAN les systèmes IO et JODA donnent de bonnes performances pour l'algorithme des K plus proches voisins ; IO, JODA et MATH pour la Forêt aléatoire ; IO pour le Perceptron multicouche. Le Bayésien naïf et la Machine à vecteurs de support ne donnent aucun système ayant une bonne performance.

### Résumé des résultats

Les différents résultats nous permettent de répondre aux questions de recherche suivantes :

**QR2** : Comment utiliser les techniques d'apprentissage pour améliorer les performances de ces techniques ?

**QR3** : L'approche avec valeurs seuils des métriques est-elle plus efficace que l'approche avec valeurs des métriques brutes ?

Pour répondre à la question de recherche 2, nous pouvons constater que l'utilisation des techniques d'apprentissage automatique a permis d'améliorer significativement les performances des techniques de calcul des seuils. Ces techniques d'apprentissage ont été appliquées sur la base des valeurs seuils déterminées à partir des méthodes des courbes ROC et d'Alves Rankings. Ces valeurs seuils nous ont permis alors de binariser les différentes valeurs des métriques et poursuivre avec des constructions des modèles de prédiction. Dans les résultats présentés précédemment, les algorithmes de Bayésien naïf, Forêt aléatoire, Perceptron multicouche et K plus proches voisins ont donné de meilleures performances contrairement à l'algorithme de Machine à vecteurs de support.

À la question de recherche 3, nous disons que l'approche basée sur les techniques de calcul de valeurs seuils, à savoir la méthode des courbes ROC et Alves Rankings, est plus efficace que l'approche basée sur les valeurs brutes des métriques. Cette approche nous a permis d'obtenir de meilleures performances pour la quasi-totalité d'algorithmes utilisés, excepté l'algorithme de Machine à vecteurs de support. Ceci pour les modèles construits sur base de l'application de deux techniques de calcul de valeurs seuils. L'efficacité de cette approche réside surtout dans le fait de déterminer de manière méthodique des valeurs seuils valides.

En fait, pour la méthode des courbes ROC, nous avons considéré les métriques ayant l'AUC supérieur à 0.6 comme celles dont les valeurs seuils valides seront déterminées comme l'illustrent les tables 4.1, 4.2, 4.3, 4.4, 4.5 et 4.6. On peut aussi constater que les métriques produisant des valeurs seuils valides ont les p-valeurs pour la plupart inférieures au seuil de signification (5%). Par exemple les systèmes ANT, IO, JODA et MATH pour ne citer qu'eux, ont tous des p-valeurs inférieures à 0.001 ; ce qui traduit une très forte présomption contre l'hypothèse nulle et affecte positivement les performances.

La méthode d'Alves Rankings quant à elle, a permis de déterminer des valeurs seuils à différents niveaux d'extraction et cela de manière non supervisée. Excepté le niveau d'extraction de 90% pour le système POI, pour lequel aucune métrique n'a fourni des valeurs seuils. Cette technique a permis de déterminer des valeurs seuils pour tous les systèmes et à différents niveaux pour une grande partie des métriques. Ce qui a considérablement impacté les performances des modèles avec des valeurs g-mean élevées comparées à celles obtenues avec l'approche basée sur les valeurs de métriques brutes.

### 4.3.6 Condensé des meilleurs modèles

Pour répondre à la question de recherche 4, à savoir : quelle technique parmi les courbes ROC et Alves Rankings donne de meilleures performances pour la prédiction de l'effort de test ?

Nous présentons au moyen des méthodes de comparaison de performances, à savoir les tests de Friedman et de Nemenyi, les meilleures performances des techniques de calcul de valeurs seuils retenues dans le présent travail. Ces techniques ont été combinées aux différents algorithmes d'apprentissage automatique retenus pour notre étude. À ces techniques, nous ajoutons aussi l'application des algorithmes d'apprentissage automatique aux valeurs brutes des métriques avec pour objectif d'identifier l'approche donnant les meilleures performances.

Le post-test selon Nemenyi (1963) peut être utilisé si la condition relative au test de Friedman indique une différence significative soit $\alpha < 0.05$.

#### Résultats basés sur la métrique d'effort de test BIN-KM5

Les résultats des tests de Friedman et Nemenyi sont présentés dans les tableaux F.1 à F.20. Nous présentons les résultats issus des méthodes de comparaison de performances des métriques extraites à 30%, 70%, 80% et 90% pour la méthode Alves Rankings. Nous présentons également les valeurs g-mean obtenues par la méthode des courbes ROC ainsi que les valeurs g-mean obtenues à partir des valeurs brutes des métriques. Ces résultats sont présentés comme suit :

- Alves Rankings à 30%

  - Les résultats pour le Bayésien naïf sont repris dans la table F.1 : plusieurs systèmes montrent que les approches basées sur les courbes ROC et Alves Rankings sont plus efficaces que l'approche basée sur les valeurs brutes des métriques. À titre d'exemple, pour le système POI, qui donne la valeur de chi-carré la plus élevée égale à 9.579 avec une p-valeur de 0.008 (la plus faible), ce qui traduit une différence statistiquement significative en faveur des courbes ROC et Alves Rankings. Cependant, les systèmes POI et IO avec une valeur chi-carré de 7.111 et une p-valeur de 0.029 ainsi que JODA avec une valeur chi-carré de 8.316 et p-valeur de 0.016 n'indiquent pas des différences significatives entre la méthode Alves Rankings et les valeurs brutes des métriques, mais plutôt entre la méthode des courbes ROC et les valeurs brutes des métriques. Avec trois systèmes de plus que la méthode d'Alves Rankings et avec des p-valeurs plus significatives que celles de la méthode d'Alves Rankings, nous pouvons conclure que pour cet algorithme, l'approche basée sur les courbes ROC est la plus efficace.
  - Les résultats pour la Machine à vecteurs de support sont repris dans la table F.2 : les systèmes POI qui a un chi-carré de 7.895 et p-valeur de 0.019, IVY avec un chi-carré égale à 7.6 et une p-valeur de 0.022 ainsi que JODA, indiquent des performances efficaces pour les approches basées sur Alves Rankings et les courbes ROC au détriment de l'approche basée sur les valeurs brutes des métriques. Cependant, le système LUCENE avec un chi-carré de 6 et p-valeur de 0.05 indique uniquement une différence significative entre l'approche basée sur les courbes ROC et l'approche basée sur les valeurs brutes des métriques. Il est aussi important de signaler que les tests de Friedman pour les systèmes ANT avec une p-valeur de 0.143, IO avec une p-valeur de 0.513, JFC avec une p-valeur de 0.165 et MATH avec une p-valeur de 0.156, ne sont pas significatifs. Par conséquent, il n'est pas utile de procéder à de multiples comparaisons afin d'identifier les différences entre les différentes approches. Il en ressort, donc, de cette analyse que l'approche basée sur les courbes ROC est celle qui est efficace pour cet algorithme.
  - Les résultats pour la Forêt aléatoire sont repris dans la table F.3 : pour l'ensemble des systèmes, l'approche basée sur les courbes ROC est la plus efficace. On peut noter les bonnes performances pour le système IVY avec une valeur de chi-carré égale à 9.333 et une p-valeur de 0.009, POI avec un chi-carré de 9.294 et une p-valeur de 0.01. Cependant, le système MATH donne une p-valeur de 0.135, ce qui ne permet pas de faire de multiples comparaisons entre approches. L'approche basée sur les courbes ROC présente des p-valeurs plus significatives par rapport à l'approche basée sur les valeurs brutes des métriques, contrairement à l'approche avec Alves Rankings sur les valeurs brutes des métriques.

- Les résultats pour le Perceptron multicouche sont repris dans la table F.4 : une fois de plus, l'approche basée sur les courbes ROC est la plus efficace. À titre illustratif on peut constater que le système POI qui a un chi-carré de 8.444 et une p-valeur de 0.015. Le même constat est fait avec le système ANT qui donne un chi-carré égale à 8.316 et une p-valeur de 0.016. L'approche basée sur les courbes ROC est la plus performante pour six systèmes, sur un ensemble de huit. Les système IO avec une p-valeur égale à .056 et MATH avec une p-valeur égale à 0.156 quant à eux ne permettent pas de procéder à de multiples comparaisons en raison d'un test de Friedman qui est non significatif.
- Les résultats pour les K plus proches voisins sont repris dans la table F.5 : sur les huit systèmes, trois systèmes indiquent de meilleures performances pour l'approche basée sur les courbes ROC. Il s'agit notamment des systèmes IO et JFC avec un chi-carré de 8.824 et une p-valeur de 0.012, ainsi que JODA avec un chi-carré de 8.444 et une p-valeur de 0.015. Quatre autres systèmes présentent de meilleures performances pour Alves Rankings. Il s'agit des systèmes ANT et POI ayant les mêmes valeurs de chi-carré, soit 8.444 et de p-valeur, soit 0.015. Nous pouvons citer aussi les systèmes IVY avec un chi-carré de 8.316 et une p-valeur de 0.016, ainsi que le système LUCENE avec un chi-carré de 7.6 et une p-valeur de 0.022. Seul le système MATH avec une p-valeur de 0.156 ne permet pas de procéder à des multiples comparaisons entre les différentes approches.

- Alves Rankings à 70%
  - Les résultats pour le Bayésien naïf sont repris dans la table F.6 : les résultats indiquent que l'approche basée sur Alves Rankings est plus efficace que celle basée sur les courbes ROC. Avec un chi-carré égal à 10 et une p-valeur de 0.007, le système JFC est celui qui donne la meilleure performance en termes de comparaisons entre approches ; cela se traduit par une différence significative évalué à 0.004 pour l'approche d'Alves Rankings par rapport à l'approche basée sur les valeurs brutes de métriques. On peut aussi citer le système IVY qui donne l'une de meilleure performance parmi tant d'autres ; avec un chi-carré égal à 9.478 et une p-valeur de 0.009. Les systèmes ANT et IO montrent aussi cette performance avec des valeurs identiques de chi-carré de 8.4 et p-valeur de 0.015. Seul le système MATH indique une égalité entre les deux approches à savoir une différence statistique évaluée à 0.014 comme l'indique le test de Nemenyi.
  - Les résultats pour la Machine à vecteurs de support sont repris dans la table F.7 : la méthode des courbes ROC est la plus efficace. Les tests de Friedman pour les systèmes IO avec une p-valeur de 0.854 (largement supérieure à 0.05), JFC avec une p-valeur de 0.074, MATH avec une p-valeur de 0.247 et LUCENE avec une p-valeur de 0.085, indiquent un niveau de signification supérieur à 0.05.

Il n'est donc pas utile de procéder à plusieurs comparaisons afin d'identifier les différences entre les différentes approches pour ces systèmes.

- Les résultats pour la Forêt aléatoire sont repris dans la table F.8 : pour l'ensemble des systèmes, l'approche basée sur les courbes ROC est la plus efficace. Nous pouvons évoquer les meilleures performances affichées par les systèmes POI et IVY avec un chi-carré de 10.182 et une p-valeur de 0.006. Le système IO aussi affiche des résultats intéressants avec un chi-carré égal à 10 et une p-valeur égale à 0.007.
- Les résultats pour le Perceptron multicouche sont repris dans la table F.9 : l'approche basée sur les courbes ROC est celle qui est efficace pour cinq systèmes. Parmi ces systèmes on peut citer POI et IVY pour lesquels le chi-carré est égale à 9.478 et la p-valeur égale à 0.009. Notons que le système POI donne une même différence significative évaluée à 0.014 pour l'approche basée sur courbes ROC et l'approche d'Alves Rankings par rapport à l'approche basée sur les valeurs brutes des métriques. ANT et LUCENE montrent aussi des résultats intéressants avec un chi-carré égale à 8.316 et une p-valeur égale à 0.016. Le système MATH avec un chi-carré équivalant à 3.714 et une p-valeur égale à 0.156, ne montre pas l'utilité de procéder à plusieurs comparaisons afin d'identifier les différences entre les différentes approches.
- Les résultats pour les K plus proches voisins sont repris dans la table F.10 : l'approche basée sur les courbes ROC est la plus efficace. Comme l'illustre ces résultats, pour les systèmes IVY avec un chi-carré de 10.182 et une p-valeur de 0.006, POI avec un chi-carré de 9.818 et une p-valeur de 0.007. L'ensemble des systèmes indiquent un niveau de signification supérieur à 0.05 ce qui a permis d'aboutir à ces résultats.

- Alves Rankings à 80%

  - Les résultats pour le Bayésien naïf sont repris dans la table F.11 : nous constatons que pour cet algorithme trois systèmes sont en faveur de l'approche basée sur les courbes ROC. Il s'agit de IVY avec un chi-carré égale à 7.6 et une p-valeur de 0.022, JODA avec une valeur chi-carré de 6.4 et une p-valeur de 0.041. Trois autres systèmes en faveur de l'approche basée sur Alves Rankings notamment le système JFC avec un chi-carré égale à 9.579 et une p-valeur de 0.008. Toutefois, il est important de souligner le fait que le système JODA montre que les courbes ROC diffèrent très significativement ($p<0.01$) de l'approche basée sur les valeurs brutes des métriques contrairement à Alves Rankings ($p>0.05$). Le test de Friedman pour les systèmes MATH et LUCENE indique un niveau de signification supérieur à 0.05. Par conséquent, il n'est pas utile de procéder à plusieurs comparaisons afin d'identifier les différences entre les différentes approches pour ces systèmes.
  - Les résultats pour la Machine à vecteurs de support sont repris dans la table F.12 : l'approche basée sur Alves Rankings est la plus efficace pour cet al-

gorithme. Nous pouvons noter de meilleures performances produites par le système ANT, POI et IVY avec des valeurs similaires de chi-carré égale à 10 et de p-valeur égale à 0.007. Cependant, les tests de Friedman des systèmes MATH et LUCENE indiquent des niveaux de signification supérieurs à 0.05.

- Les résultats pour la Forêt aléatoire sont repris dans la table F.13 : avec de meilleures performances pour six systèmes, l'approche basée sur les courbes ROC est la plus efficace. La méthode d'Alves Rankings n'a produit de meilleure performance que pour le système ANT qui a un chi-carré de 8.444 et une p-valeur de 0.015. Pour le système ANT Alves Rankings diffère de manière très significative par rapport à l'approche des valeurs brutes des métriques. Cependant, le test de Friedman pour le système JODA indique un niveau de signification supérieur à 0.05.
- Les résultats pour le Perceptron multicouche sont repris dans la table F.14 : la méthode des courbes ROC présente de meilleures performances pour un large nombre de systèmes. Notamment, les systèmes POI et IO avec des résultats très intéressants de chi-carré égal à 10 et de p-valeur égale 0.007. Cependant, plusieurs comparaisons ne peuvent être conduites en raison des tests de Friedman liés aux systèmes JODA et MATH, qui ne sont pas significatifs.
- Les résultats pour les K plus proches voisins sont repris dans la table F.15 : la méthode des courbes ROC vient largement en tête. Parmi ces systèmes, on peut nommer IO avec un chi-carré de 10 et une p-valeur de 0.007. Il faut noter que les deux méthodes, courbes ROC et Alves Rankings ont des performances égales pour les systèmes ANT et POI. Ces deux approches diffèrent significativement ($p<0.05$) de l'approche basée sur les valeurs des brutes des métriques. Cependant, les systèmes JODA et MATH indiquent des niveaux de signification supérieurs à 0.05.

- Alves Rankings à 90%

  - Les résultats pour le Bayésien naïf sont repris dans la table F.16 : les meilleures performances sont en faveur de la technique des courbes ROC pour la plupart des systèmes. Notamment avec les systèmes POI, IO et LUCENE qui donnent des chi-carré compris entre 4 et 6. Ce qui se traduit par des différences significatives (test de Nemenyi) de 0.008 pour les trois systèmes exclusivement pour l'approche des courbes ROC par rapport à l'approche des valeurs brutes des métriques. Tandis que le système JODA avec une p-valeur de 0.074 et le système MATH avec une p-valeur de 0.202 indiquent des tests de Friedman non significatifs. Par conséquent, nous ne pouvons pas déterminer laquelle de ces approches est la plus efficace.
  - Les résultats pour la Machine à vecteurs de support sont repris dans la table F.17 : la technique des courbes ROC présente de meilleures performances pour trois systèmes à l'instar du système IVY avec un chi-carré équivalent à 8.4 et

une p-valeur de 0.015. Le test de Nemenyi confirme cette tendance, puisque l'approche des courbes ROC diffère significativement (p=0.01) de l'approche basée sur valeurs brutes des métriques. La technique d'Alves Rankings quant à elle indique de bons résultats pour deux systèmes. Notamment le système ANT pour lequel le test de Nemenyi montre une différence très significative pour la technique d'Alves Rankings. Les tests de Friedman pour les systèmes IO, JODA et MATH donnent des p-valeurs de 0.655, 0.91 et 0.307 respectivement. Les tests de Friedman pour les systèmes IO, JODA et MATH donnent des niveaux de signification supérieurs à 0.05, il n' y a donc pas lieu de procéder à plusieurs comparaisons afin d'identifier les différences entre les approches.

- Les résultats pour la Forêt aléatoire sont repris dans la table F.18 : les meilleures performances sont en faveur de la technique des courbes ROC. En notant les performances intéressantes des systèmes ANT et JFC avec des valeurs chi-carré et p-valeur identiques de 8.824 et 0.012 respectivement. Les systèmes POI, IO et LUCENE affichent des résultats intéressants pour les courbes ROC exclusivement ; soit une différence très significative évaluée à 0.008 (p<0.01). Les tests de Friedman pour les systèmes JODA et MATH ne sont pas significatifs.
- Les résultats pour le Perceptron multicouche sont repris dans la table F.19 : la majorité des systèmes donne de meilleures performances pour la technique des courbes ROC sauf pour les systèmes JODA et MATH qui n'ont pas des tests de Friedman significatifs. Les tests de Nemenyi pour les systèmes ANT, POI, IO, JFC et LUCENE affichent des différences très significatives (p<0.01) en faveur de l'approche de courbes ROC.
- Les résultats pour les K plus proches voisins sont repris dans la table F.20 : Ici on remarque que les résultats de performances sont similaires à ceux de l'algorithme de Perceptron multicouche.

### Résultats basés sur la métrique d'effort de test BIN-MEAN

Les résultats des tests de Friedman et Nemenyi sont présentés dans les tableaux F.21 à F.40.

Les résultats issus des méthodes de comparaison de performances entre les valeurs g-mean des métriques extraites à 30%, 70%, 80% et 90% pour la méthode Alves Rankings, la méthode des courbes ROC ainsi que les valeurs brutes des métriques sont présentés comme suit :

- Alves Rankings à 30%

  - Les résultats pour le Bayésien naïf sont repris dans la table F.21 : l'approche basée sur les courbes ROC donne de meilleures performances. Les systèmes POI, IVY, JFC, JODA, MATH et LUCENE donnent tous des p-valeurs significatives (p<0.05) pour les tests de Friedman. Cette tendance est confirmée par les tests

de Nemenyi en faveur des courbes ROC. On peut noter à titre illustratif les systèmes IVY, JODA, LUCENE qui ont tous une p-valeur de 0.009, JFC avec une p-valeur de 0.004 ; ce qui montre des différences très significatives ($p<0.01$) entre les courbes ROC et l'approche basée sur valeurs brutes des métriques. Cependant, les systèmes ANT et IO ne sont pas significatifs pour les tests de Friedman ; nul besoin donc de procéder à plusieurs comparaisons d'approches.

- Les résultats pour la Machine à vecteurs de support sont repris dans la table F.22 : les systèmes ANT, IO, JFC, MATH et LUCENE indiquent des tests de Friedman non significatifs puisqu'ayant tous des p-valeurs supérieurs à 0.05. Il n'est donc pas utile de procéder à plusieurs comparaisons afin d'identifier les différences entre les approches pour ces systèmes. Cependant, les systèmes POI et JODA avec des p-valeur égalent à 0.009 donnent de meilleures performances pour la méthode des courbes ROC. Le système IVY avec une p-valeur de 0.014 (test de Nemenyi) indique quant à lui une égalité de performance entre la méthode des courbes ROC et la méthode d'Alves Rankings.
- Les résultats pour la Forêt aléatoire sont repris dans la table F.23 : sur un total de huit systèmes, cinq indiquent de meilleures performances pour l'approche basée sur la méthode d'Alves Rankings. Les tests de Nemenyi montrent des performances intéressantes : ANT et POI avec une p-valeur de 0.014, IO et IVY avec une p-valeur de 0.009 et JODA avec une p-valeur de 0.006. Trois systèmes donnent de meilleures performances pour la méthode des courbes ROC comme l'indique ces résultats qui sont tous très significatifs ($p<0.01$) : JFC et LUCENE ont une p-valeur égale à 0.009, MATH a une p-valeur égale à 0.004.
- Les résultats pour le Perceptron multicouche sont repris dans la table F.24 : la méthode des courbes ROC donne de meilleures performances. Les tests de Nemenyi montrent que l'approche basée sur les courbes ROC diffère très significativement de l'approche basée sur les valeurs brutes des métriques contrairement à l'approche d'Alves Rankings. Par exemple les systèmes IVY a une p-valeur de 0.004 et les systèmes ANT, JFC, MATH et LUCENE ont des p-valeur de 0.006. Seul le système JODA donne des p-valeur ($p= 0.014$) identiques pour l'approche des courbes ROC et d'Alves Rankings.
- Les résultats pour les K plus proches voisins sont repris dans la table F.25 : La méthode d'Alves Rankings donne de meilleures performances pour cet algorithme. Nous pouvons citer le système JFC avec un chi-carré de 9.333 et une p-valeur de 0.009 le système ANT et JODA avec des chi-carré de 8.316 et des p-valeur de 0.016 pour le test de Friedman ainsi que le système IO avec un chi-carré de 8.444 et une p-valeur de 0.015. Ces résultats sont confirmés par les tests de Nemenyi pour JFC et IO avec une p-valeur de 0.009, JODA avec une p-valeur de 0.006 et ANT avec une p-valeur de 0.014.

- Alves Rankings à 70%
  - Les résultats pour le Bayésien naïf sont repris dans la table F.26 : la méthode d'Alves Rankings avec cinq systèmes sur huit, dont le système POI avec une p-valeur égale à 0.006 et IVY avec une p-valeur de 0.03, donne de meilleures performances.
  - Les résultats pour la Machine à vecteurs de support sont repris dans la table F.27 : avec seulement quatre systèmes sur lesquels ont été appliqués le test de Nemenyi, la méthode de courbes ROC est celle qui présente les meilleures performances. Avec des p-valeur égalent à 0.002 les systèmes POI et IVY donnent des différences statistiques très significatives ($p<0.01$). Les quatre autres systèmes à savoir IO, JFC, MATH et LUCENE ne donnent pas des tests de Friedman significatifs.
  - Les résultats pour la Forêt aléatoire sont repris dans la table F.28 : la méthode des courbes ROC est majoritairement la meilleure en ce qui concerne les performances pour l'ensemble des systèmes. Les tests de Nemenyi montrent que le système LUCENE a une p-valeur de 0.001, JFC a une p-valeur de 0.002 ; les systèmes ANT, POI, IVY et MATH ont des p-valeurs de 0.004.
  - Les résultats pour le Perceptron multicouche sont repris dans la table F.29 : la méthode des courbes ROC est largement performante pour sept systèmes, alors que la méthode d'Alves n'est performante que pour le système POI. Les tests de Nemenyi pour cet algorithme offrent des résultats intéressants pour certains systèmes tels que LUCENE avec une p-valeur de 0.001, les systèmes ANT, IO, IVY, JFC et JFC donnent des p-valeur égalent à 0.006.
  - Les résultats pour les K plus proches voisins sont repris dans la table F.30 : pour l'ensemble des systèmes, la méthode des courbes ROC présente de meilleures performances. Les tests de Nemenyi pour certains systèmes comme JFC et LUCENE ($p=0.001$), POI ($p=0.002$), IVY (0.004) et MATH ($p=0.006$) montrent que l'approche des courbes ROC diffère très significativement de l'approche basée sur les valeurs brutes des métriques.

- Alves Rankings à 80%
  - Les résultats pour le Bayésien naïf sont repris dans la table F.31 : quatre systèmes donnent de meilleures performances en faveur de la méthode des courbes ROC, deux systèmes pour la méthode d'Alves Rankings. Pour les courbes ROC, les systèmes JODA ($p=0.006$), IVY et JFC avec une même p-valeur de 0.009, donnent des performances intéressantes. Les systèmes IO et MATH ne donnent pas des tests de Friedman significatifs.
  - Les résultats pour la Machine à vecteurs de support sont repris dans la table F.32 : les tests de Nemenyi montrent que cinq systèmes, à savoir ANT ($p=0.009$), POI et LUCENE ($p=0.014$), IVY ($p=0.021$) et JODA ($p=0.03$) indiquent de

meilleures performances pour l'approche basée sur les courbes ROC. Cependant, les systèmes IO, JFC et MATH ne donnent pas des tests de Friedman significatifs, on n'a donc pas pu procéder à plusieurs comparaisons afin d'identifier les différences entre les approches pour identifier la plus performante.

- Les résultats pour la Forêt aléatoire sont repris dans la table F.33 : la méthode des courbes ROC est largement performante. Des performances intéressantes sont données par les tests de Nemenyi. Comme pour les systèmes JFC et LUCENE ont une p-valeur de 0.002, POI et IVY ont une p-valeur de 0.006 et JODA a une p-valeur de 0.009. La méthode d'Alves Rankings ne donne de bonne performance que pour le système ANT avec une p-valeur de 0.01.
- Les résultats pour le Perceptron multicouche sont repris dans la table F.34 : les méthodes des courbes ROC et d'Alves Rankings donnent de meilleures performances tel que l'illustrent les résultats des tests de Nemenyi pour ANT (p=0.014), POI (p=0.004), JODA (p=0.014) et LUCENE (p=0.004) pour la méthode d'Alves Rankings IVY et JFC (p=0.006) ainsi que MATH (p=0.043) pour la méthode des courbes ROC. Cependant le système IO avec un chi-carré de 5.304 et une p-valeur de 0.07 ne permet de procéder à des multiples comparaisons de différentes approches.
- Les résultats pour les K plus proches voisins sont repris dans la table F.35 : la méthode des courbes ROC est largement performante à l'exception du système MATH avec un chi-carré de 5.429 et une p-valeur de 0.066 ne permet de procéder à des multiples comparaisons de différentes approches. Avec les tests de Nemenyi certains systèmes montrent des différences statistiques très significatives pour POI, JFC et LUCENE avec des p-valeurs de 0.002, IVY et JODA avec des p-valeur égalent à 0.006 ainsi que ANT avec une p-valeur de 0.009. Seul le système IO (p=0.009) donne de résultats intéressants pour l'approche d'Alves Rankings.

- Alves Rankings à 90%

  - Les résultats pour le Bayésien naïf sont repris dans la table F.36 : la méthode des courbes ROC présente de meilleures performances. Deux systèmes ANT et IVY avec une même p-valeur de 0.021 sont en faveur de la méthode d'Alves Rankings. Pour les courbes ROC les résultats de Nemenyi montrent par exemple que les systèmes POI, JFC et LUCENE ont des p-valeurs inférieures à 0.01. En revanche, le système IO n'indique pas de test de Friedman significatif.
  - Les résultats pour la Machine à vecteurs de support sont repris dans la table F.37 : la méthode des courbes ROC est performante pour quatre systèmes, ANT (p=0.004), IVY (p=0.006), POI (p=0.008) et JODA (p=0.021). Les quatre autres systèmes, à savoir IO, JFC, MATH et LUCENE ne donnent pas des tests de Friedman significatifs.

- Les résultats pour la Forêt aléatoire sont repris dans la table F.38 : la méthode des courbes ROC est performante pour un grand nombre de systèmes. Comme l'indique les tests de Nemenyi pour les systèmes JFC (p=0.006), POI, IO et LUCENE partagent tous une p-valeur égale à 0.008. La méthode d'Alves Rankings ne donne pas des résultats intéressants. Cependant, le système MATH n'indique pas de test de Friedman significatif.
- Les résultats pour le Perceptron multicouche sont repris dans la table F.39 : la méthode d'Alves Rankings n'est performante que pour les systèmes JODA (p=0.043) et IVY (p=0.021). La méthode des courbes ROC donne de meilleures performances pour un large nombre de systèmes. Parmi ces systèmes, on peut citer ANT (p=0.004), JFC (p=0.006), POI et IO avec une p-valeur de 0.008 qui ont des résultats intéressants, comme l'illustrent les tests de Nemenyi. Les systèmes MATH et LUCENE indiquent quant à eux des tests de Friedman non significatifs.
- Les résultats pour les K plus proches voisins sont repris dans la table F.40 : à l'exception du système MATH pour lequel le test de Friedman n'est pas significatif, les autres systèmes indiquent de meilleures performances pour la méthode des courbes ROC. Les tests de Nemenyi montrent que les systèmes suivants ont une p-valeur inférieure à 0.01 : ANT (p=0.006) ; POI, IO, LUCENE qui ont une p-valeur similaire de 0.008 ainsi que JFC (p=0.009). Ces résultats démontrent l'aspect prépondérant de l'approche des courbes sur l'approche d'Alves Rankings et celle basée sur les valeurs brutes des métriques.

*Chapitre. 5*

# *DISCUSSION ET CONCLUSION*

## 5.1 Menaces de validité

Notre étude au regard de bien d'autres études en génie logiciel est confrontée à des menaces qui peuvent biaiser la validité de certains résultats. Premièrement, notre étude couvre un ensemble de huit différents systèmes logiciels, cela sous-entend que les conclusions liées à ces différents systèmes ne peuvent être généralisées sur l'ensemble d'autres systèmes qui n'ont pas fait l'objet de notre étude. Ce genre d'études doivent être reproduites sur un grand nombre de systèmes pour pouvoir tirer des conclusions généralisables. L'ensemble de nos systèmes a été écrit en Java, ce qui signifie que les caractéristiques d'extraction inhérentes aux métriques logicielles de ces systèmes peuvent différer des autres langages de programmation, quoique le langage Java est un bon représentant des langages de programmation orientés objet. Ces faits impacteraient sur la possibilité de généraliser nos conclusions à d'autres systèmes écrits dans des langages autres que Java.

L'autre menace de validité est liée aux différents niveaux d'extraction 30%, 70%, 80% et 90% choisis pour la détermination de valeurs seuils pour la méthode d'Alves Rankings. L'utilisation des niveaux de pourcentage d'extraction autres que ceux utilisés dans notre étude pourraient donner des résultats légèrement différents.

L'autre menace à laquelle on fait face concerne la configuration utilisée lors de l'application des algorithmes d'apprentissage automatique avec les paramètres par défaut de l'outil WEKA. En effet, ces différents algorithmes ont été appliqués pour les trois différentes approches afin d'en dégager la plus performante d'entre elles. Ceci peut donc affecter l'évaluation des performances de ces dernières.

Une autre menace consiste dans le fait que si on applique la régression logistique multinomiale ou la régression logistique linéaire univariée dans nos modèles, cela aboutira à des classifications prenant en compte des niveaux (1 à 5) d'efforts de test différents.

Donc, l'application d'une classification ordinale va impacter d'une manière ou d'une autre les résultats et les conclusions tirés de la présente étude.

## 5.2 Conclusion et perspectives

Dans cette étude, nous avons voulu comparer différentes méthodes de calcul de seuil ainsi que les valeurs brutes des métriques logicielles afin de réaliser la prédiction des efforts de test. Nous avons donc étudié ces différentes approches pour construire des modèles de prédiction des efforts de test. Ces modèles offrent une aide importante aux testeurs (développeurs) en leur permettant de focaliser leur attention sur les composants critiques des systèmes nécessitant des efforts (relativement) importants de test. Dans cette optique, nous offrons aux concepteurs de logiciels et testeurs des outils adéquats pour garantir la qualité des systèmes logiciels. En considérant la prédiction des efforts de test basée sur les valeurs seuils, nous avons calculé les seuils des métriques en utilisant deux différentes techniques ; à savoir les courbes ROC et Alves Rankings. À ces différentes techniques, nous avons joint les valeurs brutes des métriques et essayé de prédire les efforts de test pour 8 ensembles de données (provenant de 8 systèmes logiciels différents). Ces différentes méthodologies ont été comparées en utilisant les valeurs seuils obtenues. Nous avons calculé le g-mean et l'AUC pour évaluer la qualité de prédiction de nos modèles avec une validation croisée 10-plis.

Pour construire nos modèles, nous avons utilisé cinq algorithmes d'apprentissage automatique qui sont : le Bayésien naïf, la Machine à vecteurs de support, la Forêt aléatoire, le Perceptron multicouche et les K plus proches voisins.

L'utilisation de deux techniques de calcul de seuils a permis d'obtenir des valeurs seuils valides pour tous les systèmes, à l'exception du système POI qui n'a fourni aucune valeur seuil pour la méthode d'Alves Rankings à son niveau d'extraction de 90%. Ces techniques nous ont permis de déterminer de manière objective les valeurs seuils des métriques et nous ont permis de répondre positivement à la question de recherche 1.

La technique d'Alves Rankings a été étudiée pour savoir si elle donnerait des résultats acceptables quand elle est combinée aux algorithmes d'apprentissage automatique pour la prédiction des efforts de test. Nos expérimentations indiquent d'excellents résultats pour la prédiction des efforts de test. Ces résultats montrent que pour la métrique d'effort de test BIN-KM5 (classification avec l'approche basée sur les K-moyennes), les algorithmes Forêt aléatoire et K plus proches voisins sont ceux les plus performants à 30%, 70% et 80%. L'algorithme de Forêt aléatoire est le plus performant et celui de K plus proches voisins l'est à 90%. Pour la métrique d'effort BIN-MEAN (classification avec l'approche basée sur les moyennes), les résultats sont quasi similaires, mais à 80% où l'algorithme de K plus proches voisins est le plus performant et à 90% ce sont les algorithmes de Forêt aléatoire et les K plus proches voisins qui le sont. La technique des courbes ROC donne des résultats similaires pour les mêmes algorithmes en termes de performance. Il se dégage que la Forêt aléatoire est l'algorithme le plus

performant suivi de K plus proches voisins. Pour les deux métriques représentant (de deux manières différentes) l'effort de test BIN-KM5 et BIN-MEAN, on constate que ces deux algorithmes donnent de meilleures performances pour les deux techniques de calcul de seuils. Il est fort important de souligner que les autres algorithmes donnent de bons résultats aussi, à l'exception de la Machine à vecteurs de support qui donne des résultats plus ou moins acceptables. Nous pouvons donc répondre à la question de recherche 2 positivement, c'est-à-dire que les techniques d'apprentissage automatique ont permis d'améliorer significativement les performances des techniques de calcul de valeurs seuils.

Les résultats obtenus nous permettent aussi de répondre positivement à la question de recherche 3. Les performances produites par l'approche basée sur les valeurs brutes des métriques sont mauvaises pour tous les algorithmes pour lesquels des modèles de prédiction ont été construits.

Afin de déterminer laquelle de ces deux techniques est la plus performante pour la prédiction des efforts de test et pour répondre à la question de recherche 4, nous avons utilisé les tests de Friedman et Nemenyi. Nos résultats montrent que la méthode des courbes ROC est celle qui donne de meilleures performances. Cependant, nous concluons aussi qu'Alves Rankings est une bonne méthode pour le calcul de valeurs seuils, bien que les courbes ROC puissent avoir de meilleures performances en se basant sur nos tests statistiques. En tant que technique non supervisée, Alves Rankings est facile à appliquer sans au préalable avoir des données sur l'effort de test. L'utilité de son application est à considérer ; son point fort réside dans le fait qu'elle ne requiert pas des données relatives à l'effort de test (technique non supervisée contrairement à celle basée sur les courbes ROC qui elle nécessite des données d'apprentissage).

Comme perspectives, nous pourrions étendre notre étude en appliquant les méthodes de courbes ROC et d'Alves Rankings sur un ensemble d'autres systèmes pour valider leur utilité dans l'ensemble. On peut aussi les appliquer sur plusieurs versions des systèmes utilisés dans notre étude, pour valider nos modèles de prédiction des efforts de test. Il est aussi utile de trouver une méthodologie pour définir les différents niveaux de pourcentage d'extraction des valeurs seuils pour la méthode d'Alves Rankings. Par ailleurs, notre étude pourrait être approfondie en considérant des niveaux d'efforts de test suivant une certaine échelle de valeurs.

# *BIBLIOGRAPHIE*


[1] Karim O Elish and Mahmoud O Elish. Predicting defect-prone software modules using support vector machines. *Journal of Systems and Software*, 81(5) :649–660, 2008.

[2] Ian Sommerville. Software engineering 9th edition. *ISBN-10*, 137035152, 2011.

[3] Shyam R Chidamber and Chris F Kemerer. A metrics suite for object oriented design. *IEEE Transactions on software engineering*, 20(6) :476–493, 1994.

[4] Erik Arisholm, Lionel C Briand, and Eivind B Johannessen. A systematic and comprehensive investigation of methods to build and evaluate fault prediction models. *Journal of Systems and Software*, 83(1) :2–17, 2010.

[5] Khaled El Emam, Saïda Benlarbi, Nishith Goel, and Shesh N. Rai. The confounding effect of class size on the validity of object-oriented metrics. *IEEE Transactions on Software Engineering*, 27(7) :630–650, 2001.

[6] Khaled El Emam, Walcelio Melo, and Javam C Machado. The prediction of faulty classes using object-oriented design metrics. *Journal of Systems and Software*, 56 (1) :63–75, 2001.

[7] Victor R Basili, Lionel C. Briand, and Walcélio L Melo. A validation of object-oriented design metrics as quality indicators. *IEEE Transactions on software engineering*, 22(10) :751–761, 1996.

[8] Tibor Gyimothy, Rudolf Ferenc, and Istvan Siket. Empirical validation of object-oriented metrics on open source software for fault prediction. *IEEE Transactions on Software engineering*, 31(10) :897–910, 2005.

[9] Taghi M Khoshgoftaar and Naeem Seliya. Comparative assessment of software quality classification techniques : An empirical case study. *Empirical Software Engineering*, 9(3) :229–257, 2004.

[10] Hector M Olague, Letha H Etzkorn, Sampson Gholston, and Stephen Quattlebaum. Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes. *IEEE Transactions on software Engineering*, 33(6) :402–419, 2007.

[11] Raed Shatnawi and Wei Li. The effectiveness of software metrics in identifying error-prone classes in post-release software evolution process. *Journal of systems and software*, 81(11) :1868–1882, 2008.

[12] Yuming Zhou and Hareton Leung. Empirical analysis of object-oriented design metrics for predicting high and low severity faults. *IEEE Transactions on software engineering*, 32(10) :771–789, 2006.

[13] Tiago L Alves, Christiaan Ypma, and Joost Visser. Deriving metric thresholds from benchmark data. In *2010 IEEE International Conference on Software Maintenance*, pages 1–10. IEEE, 2010.

[14] Fadel Toure. Orientation de l'effort des tests unitaires dans les systèmes orientés objet : une approche basée sur les métriques logicielles. page 40, 2016.

[15] Raed Shatnawi. The application of roc analysis in threshold identification, data imbalance and metrics selection for software fault prediction. *Innovations in Systems and Software Engineering*, 13(2-3) :201–217, 2017.

[16] Raed Shatnawi, Wei Li, James Swain, and Tim Newman. Finding software metrics threshold values using roc curves. *Journal of software maintenance and evolution : Research and practice*, 22(1) :1–16, 2010.

[17] Raed Shatnawi. Deriving metrics thresholds using log transformation. *Journal of Software : Evolution and Process*, 27(2) :95–113, 2015.

[18] Alexandre Boucher and Mourad Badri. Using software metrics thresholds to predict fault-prone classes in object-oriented software. In *2016 4th Intl Conf on Applied Computing and Information Technology/3rd Intl Conf on Computational Science/Intelligence and Applied Informatics/1st Intl Conf on Big Data, Cloud Computing, Data Science & Engineering (ACIT-CSII-BCD)*, pages 169–176. IEEE, 2016.

[19] Alexandre Boucher and Mourad Badri. Software metrics thresholds calculation techniques to predict fault-proneness : An empirical comparison. *Information and Software Technology*, 96 :38–67, 2018.

[20] Leonardo Da Costa Santos, Renata M Saraiva, Mirko Perkusich, Hyggo O Almeida, and Angelo Perkusich. An empirical study on the influence of context in computing thresholds for chidamber and kemerer metrics. In *SEKE*, pages 357–362, 2017.

[21] Kecia AM Ferreira, Mariza AS Bigonha, Roberto S Bigonha, Luiz FO Mendes, and Heitor C Almeida. Identifying thresholds for object-oriented software metrics. *Journal of Systems and Software*, 85(2) :244–257, 2012.

[22] Robert Martin. Oo design quality metrics. *An analysis of dependencies*, 12 :151–170, 1994.

[23] Matthieu Foucault, Marc Palyart, Jean-Rémy Falleri, and Xavier Blanc. Computing contextual metric thresholds. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, pages 1120–1125. ACM, 2014.

[24] B Efron. Bootstrap methods : another look at the jackknife annals of statistics 7 : 1–26. *View Article PubMed/NCBI Google Scholar*, 1979.

[25] Bassey Isong and Ekabua Obeten. A systematic review of the empirical validation of object-oriented metrics towards fault-proneness prediction. *International Journal of Software Engineering and Knowledge Engineering*, 23(10) :1513- 1540, 2013.

[26] Ruchika Malhotra and Ankita Jain Bansal. Fault prediction considering threshold effects of object-oriented metrics. *Expert Systems*, 32(2) :203–219, 2015.

[27] Ewan Tempero. Compsci 702 :software measurementthe "ck" metrics. page 19. https ://www.cs.auckland.ac.nz/ ewan/, 2008.

[28] Lionel C Briand, Jurgen Wust, and Hakim Lounis. Using coupling measurement for impact analysis in object-oriented systems. In *Proceedings IEEE International Conference on Software Maintenance-1999 (ICSM'99).'Software Maintenance for Business Change'(Cat. No. 99CB36360)*, pages 475–482. IEEE, 1999.

[29] Nachiappan Nagappan. Toward a software testing and reliability early warning metric suite. In *Proceedings of the 26th international conference on software engineering*, pages 60–62. IEEE Computer Society, 2004.

[30] David P Darcy and Chris F Kemerer. Oo metrics in practice. *IEEE Software*, 22 (6) :17–19, 2005.

[31] Yasutaka Kamei, Shinsuke Matsumoto, Akito Monden, Ken-ichi Matsumoto, Bram Adams, and Ahmed E Hassan. Revisiting common bug prediction findings using effort-aware models. In *2010 IEEE International Conference on Software Maintenance*, pages 1–10. IEEE, 2010.

[32] Magiel Bruntink and Arie Van Deursen. Predicting class testability using object-oriented metrics. In *Source Code Analysis and Manipulation, Fourth IEEE International Workshop on*, pages 136–145. IEEE, 2004.

[33] Satwinder Singh and KS Kahlon. Object oriented software metrics threshold values at quantitative acceptable risk level. *CSI transactions on ICT*, 2(3) :191–205, 2014.

[34] Alexandre Boucher and Mourad Badri. An unsupervised fault-proneness prediction model using multiple risk levels for object-oriented software systems : An empirical study. *MÉMOIRE PRÉSENTÉ À L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES*, 2018.

[35] Iso Iso and IEC Std. 9126 software product evaluation–quality characteristics and guidelines for their use. *Geneva, International Organization for Standardization*, 1991.

[36] S Roger. Pressman. software engineering : A practitioner's approach, 2001.

[37] Melis Dagpinar and Jens H Jahnke. Predicting maintainability with object-oriented metrics-an empirical comparison. In *null*, page 155. IEEE, 2003.

[38] Magiel Bruntink and Arie van Deursen. An empirical study into class testability. *Journal of systems and software*, 79(9) :1219–1232, 2006.

[39] Robert V Binder. Design for testability in object-oriented systems. *Communications of the ACM*, 37(9) :87–102, 1994.

[40] Yogesh Singh, Arvinder Kaur, and Ruchika Malhotra. Predicting testing effort using artificial neural network. In *Proceedings of the World Congress on Engineering and Computer Science (WCECS 2008) San Francisco, USA. Newswood Limited*, pages 1012–1017, 2008.

[41] Mourad Badri and Fadel Toure. Empirical analysis of object-oriented design metrics for predicting unit testing effort of classes. *Journal of Software Engineering and Applications*, 5(7) :513, 2012.

[42] Cagatay Catal and Banu Diri. Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem. *Information Sciences*, 179(8) :1040–1058, 2009.

[43] Danijel Radjenović, Marjan Heričko, Richard Torkar, and Aleš Živkovič. Software fault prediction metrics : A systematic literature review. *Information and software technology*, 55(8) :1397–1418, 2013.

[44] Ruchika Malhotra. *Empirical research in software engineering : concepts, analysis, and applications*. Chapman and Hall/CRC, 2016.

[45] Ramanath Subramanyam and Mayuram S. Krishnan. Empirical analysis of ck metrics for object-oriented design complexity : Implications for software defects. *IEEE Transactions on software engineering*, 29(4) :297–310, 2003.

[46] Mahmoud O Elish, Ali H Al-Yafei, and Muhammed Al-Mulhem. Empirical comparison of three metrics suites for fault prediction in packages of object-oriented systems : A case study of eclipse. *Advances in Engineering Software*, 42(10) : 852–859, 2011.

[47] Segla Kpodjedo, Filippo Ricca, Philippe Galinier, Yann-Gaël Guéhéneuc, and Giuliano Antoniol. Design evolution metrics for defect prediction in object oriented systems. *Empirical Software Engineering*, 16(1) :141–175, 2011.

[48] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software : an update. *ACM SIGKDD explorations newsletter*, 11(1) :10–18, 2009.

[49] introduction-to-machine-learning. `https://blog.knoldus.com/introduction-to-machine-learning-2/`. Visité le 16 mars 2019.

[50] Yuming Zhou, Baowen Xu, and Hareton Leung. On the ability of complexity metrics to predict fault-prone classes in object-oriented systems. *Journal of Systems and Software*, 83(4) :660–674, 2010.

[51] Pooja Kapoor, Deepak Arora, and Ashwani Kumar. Effects of mean metric value over ck metrics distribution towards improved software fault predictions. In *Advances in Computer and Computational Sciences*, pages 57–71. Springer, 2017.

[52] Leo Breiman. Random forests. *Machine learning*, 45(1) :5–32, 2001.

[53] N Gayatri, S Nickolas, AV Reddy, and R Chitra. Performance analysis of datamining algorithms for software quality prediction. In *2009 International Conference on Advances in Recent Technologies in Communication and Computing*, pages 393–395. IEEE, 2009.

[54] Arvinder Kaur and Ruchika Malhotra. Application of random forest in predicting fault-prone classes. In *2008 International Conference on Advanced Computer Theory and Engineering*, pages 37–43. IEEE, 2008.

[55] Ruchika Malhotra and Ankita Jain. Fault prediction using statistical and machine learning methods for improving software quality. *Journal of Information Processing Systems*, 8(2) :241–262, 2012.

[56] Robert Nisbet, John Elder, and Gary Miner. *Handbook of statistical analysis and data mining applications*. Academic Press, 2009.

[57] Iker Gondra. Applying machine learning to software fault-proneness prediction. *Journal of Systems and Software*, 81(2) :186–195, 2008.

[58] Lionel C Briand. Novel applications of machine learning in software testing. In *2008 The Eighth International Conference on Quality Software*, pages 3–10. IEEE, 2008.

[59] Raed Shatnawi. Improving software fault-prediction for imbalanced data. In *2012 international conference on innovations in information technology (IIT)*, pages 54–59. IEEE, 2012.

[60] Bhekisipho Twala. Software faults prediction using multiple classifiers. In *2011 3rd International Conference on Computer Research and Development*, volume 4, pages 504–510. IEEE, 2011.

[61] Rinkaj Goyal, Pravin Chandra, and Yogesh Singh. Suitability of knn regression in the development of interaction based software fault prediction models. *Ieri Procedia*, 6 :15–21, 2014.

[62] Romi Satria Wahono, Nanna Suryana Herman, and Sabrina Ahmad. A comparison framework of classification models for software defect prediction. *Advanced Science Letters*, 20(10-11) :1945–1950, 2014.

[63] Chubato Wondaferaw Yohannese and Tianrui Li. A combined-learning based framework for improved software fault prediction. *International Journal of Computational Intelligence Systems*, 10 :647–662, 2017. ISSN 1875-6883. doi : https://doi.org/10.2991/ijcis.2017.10.1.43. URL `https://doi.org/10.2991/ijcis.2017.10.1.43`.

[64] Ruchika Malhotra and Ankita Jain. Software effort prediction using statistical and machine learning methods. *International Journal of Advanced Computer Science and Applications*, 2(1) :145–152, 2011.

[65] Mark H Zweig and Gregory Campbell. Receiver-operating characteristic (roc) plots : a fundamental evaluation tool in clinical medicine. *Clinical chemistry*, 39 (4) :561–577, 1993.

[66] Cagatay Catal, Ugur Sevim, and Banu Diri. Clustering and metrics thresholds based software fault prediction of unlabeled program modules. In *2009 Sixth International Conference on Information Technology : New Generations*, pages 199–204. IEEE, 2009.

[67] Golnoush Abaei, Ali Selamat, and Hamido Fujita. An empirical study based on semi-supervised hybrid self-organizing map for software fault prediction. *Knowledge-Based Systems*, 74 :28–39, 2015.

[68] Raed Shatnawi. A quantitative investigation of the acceptable risk levels of object-oriented metrics in open-source systems. *IEEE Transactions on software engineering*, 36(2) :216–225, 2010.

[69] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan) :1–30, 2006.

[70] Karel Dejaeger, Thomas Verbraken, and Bart Baesens. Toward comprehensible software fault prediction models using bayesian network classifiers. *IEEE Transactions on Software Engineering*, 39(2) :237–257, 2013.

[71] Thilo Mende and Rainer Koschke. Effort-aware defect prediction models. In *2010 14th European Conference on Software Maintenance and Reengineering*, pages 107–116. IEEE, 2010.

[72] Ibm spss statistics v25. `https://www.ibm.com/support/knowledgecenter/en/SSLVMB_25.0.0/statistics_kc_ddita/spss/product_landing.html`.

[73] posthoc.friedman.nemenyi.test. `https://rdrr.io/cran/PMCMR/man/posthoc.friedman.nemenyi.test.html`.

[74] Fadel Toure, Mourad Badri, and Luc Lamontagne. Predicting different levels of the unit testing effort of classes using source code metrics : a multiple case study on open-source software. *Innovations in Systems and Software Engineering*, 14(1) : 15–46, 2018.

[75] Fadel Toure, Mourad Badri, and Luc Lamontagne. Towards a unified metrics suite for junit test cases. In *SEKE*, pages 115–120, 2014.

[76] Fadel Toure, Mourad Badri, and Luc Lamontagne. A metrics suite for junit test code : a multiple case study on open source software. *Journal of Software Engineering Research and Development*, 2(1) :14, 2014.

[77] Mourad Badri and Fadel Toure. Empirical analysis for investigating the effect of control flow dependencies on testability of classes. In *SEKE*, pages 475–480, 2011.

[78] Mourad Badri, Linda Badri, and Fadel Toure. Empirical analysis of object-oriented design metrics : Towards a new metric using control flow paths and probabilities. *Journal of Object Technology*, 8(6) :123–142, 2009.

[79] Mourad Badri and Fadel Toure. Evaluating the effect of control flow on the unit testing effort of classes : An empirical analysis. *Advances in Software Engineering*, 2012 :5, 2012.

[80] Vandana Gupta, KK Aggarwal, and Y Singh. A fuzzy approach for integrated measure of object-oriented software testability. *Journal of Computer Science*, 1(2) : 276–282, 2005.

[81] Andrian Marcus, Denys Poshyvanyk, and Rudolf Ferenc. Using the conceptual cohesion of classes for fault prediction in object-oriented systems. *IEEE Transactions on Software Engineering*, 34(2) :287–300, 2008.

[82] Yogesh Singh, Arvinder Kaur, and Ruchika Malhotra. Empirical validation of object-oriented metrics for predicting fault proneness models. *Software quality journal*, 18(1) :3, 2010.

[83] Lionel C Briand, John W Daly, and Jürgen Wüst. A unified framework for cohesion measurement in object-oriented systems. *Empirical Software Engineering*, 3(1) : 65–117, 1998.

[84] Lionel C Briand, Jürgen Wüst, John W Daly, and D Victor Porter. Exploring the relationships between design measures and software quality in object-oriented systems. *Journal of systems and software*, 51(3) :245–273, 2000.

[85] Mccabe software (2012) using code quality metrics in management of outsourced development and maintenance, white paper. `http://www.mccabe.com/pdf/McCabeCodeQualityMetrics-OutsourcedDev.pdf`. Visité le 02 mai 2019.

[86] Linda H Rosenberg, Ruth Stapko, and Albert Gallo. Risk-based object oriented testing. *24th SWE*, 1999.

# *ANNEXE. A*

## *Prédiction de l'effort de test : Approche supervisée (BIN-KM5)*

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.045 | 0.977 | 0 | 0.037 | 0.981 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.025 | 0 | 0.987 | 0.026 | 0.053 | 0.96 |
| LOC | 0 | 0.063 | 0.969 | 0 | 0.119 | 0.939 |
| NOC | 0 | 0.6 | 0.632 | - | - | - |
| RFC | 0 | 0.092 | 0.953 | 0 | 0.244 | 0.869 |
| WMC | 0 | 0.103 | 0.947 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.1 | 0.949 | 0.049 | 0 | 0.975 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.609 | 0.625 | 0 | 0.25 | 0.866 |
| LOC | 0.029 | 0.091 | 0.939 | 0.015 | 0.074 | 0.955 |
| NOC | - | - | - | - | - | - |
| RFC | 0.029 | 0.061 | 0.955 | 0.019 | 0 | 0.99 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.056 | 0.972 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.177 | 0.907 | 0 | 0.03 | 0.985 |
| LOC | 0.011 | 0.037 | 0.976 | 0.019 | 0 | 0.99 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.114 | 0.941 | 0 | 0 | 1 |
| WMC | 0 | 0.119 | 0.939 | 0.019 | 0 | 0.99 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.095 | 0.951 |
| DIT | 0 | 0.048 | 0.976 | - | - | - |
| LCOM | 0 | 0.143 | 0.926 | - | - | - |
| LOC | 0 | 0.05 | 0.975 | 0 | 0.063 | 0.969 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.15 | 0.922 | 0 | 0.15 | 0.922 |
| WMC | 0 | 0.261 | 0.86 | 0 | 0.333 | 0.817 |

TABLE A.1 – Résultats de la classification avec Bayésien naïf et la métrique BIN-KM5

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.182 | 0.904 | 0 | 0.578 | 0.65 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.935 | 0.255 | 0 | 0.974 | 0.161 |
| LOC | 0 | 0.328 | 0.82 | 0 | 0.571 | 0.655 |
| NOC | 0 | 0.9 | 0.316 | - | - | - |
| RFC | 0.63 | 0 | 0.608 | 0 | 0.764 | 0.486 |
| WMC | 0 | 0.293 | 0.841 | 0 | 0.755 | 0.495 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.8 | 0.447 | 0 | 0.278 | 0.85 |
| DIT | - | - | - | 0.016 | 0 | 0.992 |
| LCOM | 0 | 0.957 | 0.207 | 0 | 0.964 | 0.19 |
| LOC | 0 | 0.848 | 0.39 | 0 | 0.63 | 0.608 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.758 | 0.492 | 0 | 0.116 | 0.94 |
| WMC | 0.121 | 0.853 | 0.36 | 0 | 0.731 | 0.519 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.14 | 0.927 | 0 | 0.152 | 0.921 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.962 | 0.195 | 0 | 0.697 | 0.55 |
| LOC | 0.944 | 0 | 0.237 | 0 | 0.792 | 0.456 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.394 | 0.778 | 0 | 0.147 | 0.923 |
| WMC | 0 | 0.712 | 0.537 | 0 | 0.348 | 0.807 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.407 | 0.77 | 0 | 0.619 | 0.617 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 0.429 | 0.756 | - | - | - |
| LOC | 0 | 0.9 | 0.316 | 0 | 0.875 | 0.354 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.55 | 0.671 | 0 | 0.875 | 0.354 |
| WMC | 0 | 0.87 | 0.361 | 0 | 0.944 | 0.237 |

TABLE A.2 – Résultats de la classification avec Machine à vecteurs de support et la métrique BIN-KM5

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0 | 0.026 | 0.987 |
| LOC | 0 | 0.016 | 0.992 | 0 | 0 | 1 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0.019 | 0 | 0.99 | 0.003 | 0 | 0.998 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0.015 | 0 | 0.992 |
| LOC | 0.029 | 0 | 0.985 | 0.015 | 0 | 0.992 |
| NOC | - | - | - | - | - | - |
| RFC | 0.088 | 0 | 0.955 | 0.019 | 0.023 | 0.979 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0.03 | 0.985 |
| LOC | 0 | 0 | 1 | 0.019 | 0 | 0.99 |
| NOC | - | - | - | - | - | - |
| RFC | 0.011 | 0.008 | 0.99 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0.019 | 0 | 0.99 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 0 | 1 | - | - | - |
| LOC | 0 | 0 | 1 | 0 | 0 | 1 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0.014 | 0 | 0.993 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |

TABLE A.3 – Résultats de la classification avec la Forêt aléatoire et la métrique BIN-KM5

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.028 | 0.044 | 0.964 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.032 | 0.984 | 0 | 0.211 | 0.888 |
| LOC | 0.085 | 0.016 | 0.949 | 0.011 | 0.034 | 0.977 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 0 | 1 | 0.031 | 0.065 | 0.952 |
| WMC | 0 | 0 | 1 | 0 | 0.02 | 0.99 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.056 | 0.972 |
| DIT | - | - | | 0 | 0 | 1 |
| LCOM | 0.023 | 0.478 | 0.714 | 0 | 0.357 | 0.802 |
| LOC | 0.029 | 0 | 0.985 | 0 | 0.037 | 0.981 |
| NOC | - | - | - | - | - | - |
| RFC | 0.088 | 0 | 0.955 | 0 | 0.023 | 0.988 |
| WMC | 0.061 | 0.029 | 0.955 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.008 | 0 | 0.996 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0.014 | 0.329 | 0.813 | 0 | 0.061 | 0.969 |
| LOC | 0.011 | 0.044 | 0.972 | 0.019 | 0 | 0.99 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.053 | 0.973 | 0 | 0.029 | 0.985 |
| WMC | 0.006 | 0.085 | 0.954 | 0.019 | 0 | 0.99 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 0.143 | 0.926 | - | - | - |
| LOC | 0 | 0 | 1 | 0 | 0 | 1 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0.075 | 0.962 |
| WMC | 0.028 | 0.043 | 0.964 | 0 | 0.139 | 0.928 |

TABLE A.4 – Résultats de la classification avec Perceptron multicouche et la métrique BIN-KM5

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0 | 0.026 | 0.987 |
| LOC | 0 | 0.016 | 0.992 | 0 | 0.006 | 0.997 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0.019 | 0.017 | 0.982 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0.015 | 0 | 0.992 |
| LOC | 0.029 | 0 | 0.985 | 0.015 | 0.037 | 0.974 |
| NOC | - | - | - | - | - | - |
| RFC | 0.029 | 0 | 0.985 | 0.019 | 0.023 | 0.979 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0.03 | 0.985 |
| LOC | 0 | 0 | 1 | 0.019 | 0 | 0.99 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.008 | 0.996 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0.019 | 0 | 0.99 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 0.071 | 0.964 | - | - | - |
| LOC | 0 | 0 | 1 | 0.01 | 0.063 | 0.964 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0.014 | 0 | 0.993 |
| WMC | 0 | 0.043 | 0.978 | 0 | 0.028 | 0.986 |

TABLE A.5 – Résultats de la classification avec K plus proches voisins et la métrique BIN-KM5

# *ANNEXE. B*

## *Prédiction de l'effort de test : Approche supervisée (BIN-MEAN)*

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.045 | 0.977 | 0 | 0.175 | 0.908 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.012 | 0 | 0.994 | - | - | - |
| LOC | 0 | 0.063 | 0.969 | 0 | 0.155 | 0.919 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.092 | 0.953 | 0 | 0.256 | 0.863 |
| WMC | 0 | 1 | 0 | 0 | 0.148 | 0.923 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.143 | 0.926 | 0 | 0.156 | 0.919 |
| DIT | 0.053 | 0 | 0.973 | 0 | 0 | 1 |
| LCOM | 0 | 0.609 | 0.625 | 0 | 0.25 | 0.866 |
| LOC | 0 | 0.174 | 0.909 | 0.015 | 0.074 | 0.955 |
| NOC | - | - | - | - | - | - |
| RFC | 0.026 | 0.103 | 0.935 | 0.018 | 0 | 0.991 |
| WMC | 0 | 0.474 | 0.725 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 0.994 | 0 | 0.042 | 0.979 |
| DIT | - | - | - | 0 | 0.034 | 0.983 |
| LCOM | 0.232 | 0.232 | 0.876 | 0 | 0.03 | 0.985 |
| LOC | 0.088 | 0.088 | 0.955 | 0.019 | 0 | 0.99 |
| NOC | - | - | - | - | - | - |
| RFC | 0.071 | 0.071 | 0.964 | 0 | 0.03 | 0.985 |
| WMC | 0.064 | 0.064 | 0.967 | 0.019 | 0 | 0.99 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.095 | 0.951 |
| DIT | 0 | 0.048 | 0.976 | - | - | - |
| LCOM | 0 | 0.143 | 0.926 | - | - | - |
| LOC | 0 | 0.05 | 0.975 | 0 | 0.238 | 0.873 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.15 | 0.922 | 0 | 0.058 | 0.971 |
| WMC | 0 | 0.261 | 0.86 | 0 | 0.4 | 0.775 |

TABLE B.1 – Résultats de la classification avec Bayésien naïf et la métrique BIN-MEAN

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.182 | 0.904 | 0 | 0.6 | 0.632 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.929 | 0.266 | - | - | - |
| LOC | 0 | 0.328 | 0.82 | 0 | 0.571 | 0.655 |
| NOC | - | - | - | - | - | - |
| RFC | 0.63 | 0 | 0.608 | 0 | 0.767 | 0.483 |
| WMC | 0 | 1 | 0 | 0 | 0.692 | 0.555 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.821 | 0.423 | 0 | 0.4 | 0.775 |
| DIT | 0.789 | 0 | 0.459 | 0.016 | 0 | 0.992 |
| LCOM | 0 | 0.957 | 0.207 | 0 | 0.964 | 0.19 |
| LOC | 0 | 0.913 | 0.295 | 0 | 0.63 | 0.608 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.897 | 0.321 | 0 | 0.026 | 0.987 |
| WMC | 0 | 0.842 | 0.397 | 0 | 0.731 | 0.519 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.213 | 0.887 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0.345 | 0.809 |
| LCOM | 0 | 0.963 | 0.192 | 0 | 0.697 | 0.55 |
| LOC | 0 | 0.559 | 0.664 | 0 | 0.792 | 0.456 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.373 | 0.792 | 0 | 0.152 | 0.921 |
| WMC | 0 | 0.654 | 0.588 | 0 | 0.348 | 0.807 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 1 | 0 | 0 | 0 | 0.619 | 0.617 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 0.429 | 0.756 | - | - | - |
| LOC | 0 | 0.9 | 0.316 | 0 | 0.929 | 0.266 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.55 | 0.671 | 1 | 0 | 0 |
| WMC | 0 | 0.87 | 0.361 | 0 | 0.943 | 0.239 |

TABLE B.2 – Résultats de la classification avec Machine à vecteurs de support et la métrique BIN-MEAN

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.012 | 0 | 0.994 | - | - | - |
| LOC | 0 | 0.016 | 0.992 | 0.005 | 0 | 0.997 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0 | 0.5 | 0.707 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0.015 | 0 | 0.992 |
| LOC | 0.045 | 0 | 0.977 | 0.015 | 0 | 0.992 |
| NOC | - | - | - | - | - | - |
| RFC | 0.026 | 0.034 | 0.97 | 0 | 0 | 1 |
| WMC | 0.021 | 0 | 0.989 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.007 | 0 | 0.996 | 0 | 0.03 | 0.985 |
| LOC | 0 | 0 | 1 | 0.019 | 0 | 0.99 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0.023 | 0 | 0.988 |
| WMC | 0.007 | 0 | 0.996 | 0.019 | 0 | 0.99 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 0 | 1 | - | - | - |
| LOC | 0 | 0 | 1 | 0.014 | 0 | 0.993 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0.013 | 0 | 0.993 |

TABLE B.3 – Résultats de la classification avec Forêt aléatoire et la métrique BIN-MEAN

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0.025 | 0.983 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.071 | 0.964 | - | - | - |
| LOC | 0.085 | 0.016 | 0.949 | 0.021 | 0.031 | 0.974 |
| NOC | - | - | - | 0.009 | 0.068 | 0.961 |
| RFC | 0 | 0 | 1 | - | - | - |
| WMC | 0 | 0.5 | 0.707 | 0.038 | 0.047 | 0.957 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.044 | 0.978 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0.023 | 0.478 | 0.714 | 0 | 0.357 | 0.802 |
| LOC | 0.068 | 0 | 0.965 | 0 | 0.037 | 0.981 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.034 | 0.983 | 0 | 0 | 1 |
| WMC | 0.042 | 0.053 | 0.952 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.028 | 0.256 | 0.85 | 0 | 0.061 | 0.969 |
| LOC | 0.019 | 0.044 | 0.968 | 0.019 | 0 | 0.99 |
| NOC | - | - | - | - | - | - |
| RFC | 0.03 | 0.016 | 0.977 | 0.023 | 0.03 | 0.973 |
| WMC | 0.027 | 0.038 | 0.967 | 0.019 | 0 | 0.99 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | 0 | 1 | - | - | - |
| LCOM | 0 | 0.143 | 0.926 | - | - | - |
| LOC | 0 | 0 | 1 | 0.069 | 0.095 | 0.918 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0.029 | 0.985 |
| WMC | 0.028 | 0.043 | 0.964 | 0.025 | 0.086 | 0.944 |

TABLE B.4 – Résultats de la classification avec Perceptron multicouche et la métrique BIN-MEAN

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.012 | 0 | 0.994 | - | - | - |
| LOC | 0 | 0.016 | 0.992 | 0 | 0 | 1 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0 | 0.5 | 0.707 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0.015 | 0 | 0.992 |
| LOC | 0 | 0 | 1 | 0.015 | 0.037 | 0.974 |
| NOC | - | - | - | - | - | - |
| RFC | 0.026 | 0.034 | 0.97 | 0 | 0 | 1 |
| WMC | 0.021 | 0 | 0.989 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.007 | 0 | 0.996 | 0 | 0.03 | 0.985 |
| LOC | 0 | 0.015 | 0.992 | 0.019 | 0 | 0.99 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0.023 | 0 | 0.988 |
| WMC | 0 | 0.013 | 0.993 | 0.019 | 0 | 0.99 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 0.071 | 0.964 | - | - | - |
| LOC | 0 | 0 | 1 | 0.014 | 0 | 0.993 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0 | 0.043 | 0.978 | 0 | 0.029 | 0.985 |

TABLE B.5 – Résultats de la classification avec K plus proches voisins et la métrique BIN-MEAN

# *ANNEXE. C*

## *Prédiction de l'effort de test : Approche non supervisée (BIN-KM5 & BIN-MEAN)*

# Classification avec Bayésien naïf

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.01 | 0.995 | 0 | 0.124 | 0.936 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.45 | 0.742 | 0 | 0.363 | 0.798 |
| LOC | 0 | 0.058 | 0.971 | 0 | 0.181 | 0.905 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.086 | 0.956 | 0 | 0.244 | 0.869 |
| WMC | 0 | 0.088 | 0.955 | 0 | 0.148 | 0.923 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.1 | 0.949 | 0 | 0.139 | 0.928 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.609 | 0.625 | 0 | 0.425 | 0.758 |
| LOC | 0 | 0.211 | 0.888 | 0.031 | 0 | 0.984 |
| NOC | - | - | - | - | - | - |
| RFC | 0.049 | 0.115 | 0.917 | 0.036 | 0.026 | 0.969 |
| WMC | 1 | 0.031 | 0 | 0.016 | 0.059 | 0.962 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.007 | 0 | 0.996 | 0.047 | 0 | 0.976 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.261 | 0.86 | 0 | 0.207 | 0.891 |
| LOC | 0 | 0.099 | 0.949 | 0.038 | 0 | 0.981 |
| NOC | - | - | - | - | - | - |
| RFC | 0.022 | 0 | 0.989 | 0 | 0 | 1 |
| WMC | 0 | 0.08 | 0.959 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.047 | 0 | 0.976 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.345 | 0.809 | 0 | 0.77 | 0.48 |
| LOC | 0 | 0.105 | 0.946 | 0 | 0.133 | 0.931 |
| NOC | - | - | - | 0 | 0.325 | 0.822 |
| RFC | 0 | 0.042 | 0.979 | 0 | 0.182 | 0.904 |
| WMC | 0 | 0.056 | 0.972 | 0 | 0.286 | 0.845 |

TABLE C.1 – Résultats de la classification avec Bayésien naïf et les métriques BIN-KM5 & BIN-MEAN à 30%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.018 | 0 | 0.991 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0.014 | 0.146 | 0.918 | 0.021 | 0.063 | 0.958 |
| LOC | 0.05 | 0 | 0.975 | 0.015 | 0 | 0.992 |
| NOC | - | - | - | - | - | - |
| RFC | 0.13 | 0 | 0.993 | 0.015 | 0 | 0.992 |
| WMC | 0.038 | 0 | 0.981 | 0.021 | 0 | 0.989 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.016 | 0.333 | 0.81 | 0.035 | 0 | 0.982 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.015 | 1 | 0 | 0.024 | 0.3 | 0.827 |
| LOC | 0.015 | 1 | 0 | 0.022 | 0 | 0.989 |
| NOC | 0 | 0.538 | 0.68 | - | - | - |
| RFC | 0 | 1 | 0 | 0 | 0.143 | 0.926 |
| WMC | 0 | 1 | 0 | 0.011 | 0 | 0.994 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.005 | 0 | 0.997 | 0 | 0.5 | 0.707 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0.024 | 0 | 0.988 | 0.057 | 0 | 0.971 |
| LOC | 0.028 | 0 | 0.986 | 0.028 | 0 | 0.986 |
| NOC | 0 | 0.13 | 0.933 | - | - | - |
| RFC | 0.074 | 0.028 | 0.949 | - | - | - |
| WMC | 0.019 | 0 | 0.99 | 0.029 | 0 | 0.985 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0 | 0.995 |
| DIT | - | - | - | - | - | - |
| LCOM | 0.057 | 0 | 0.971 | 0 | 0.167 | 0.913 |
| LOC | 0.018 | 0.333 | 0.809 | 0.018 | 0 | 0.991 |
| NOC | - | - | - | 0.044 | 0 | 0.978 |
| RFC | 0 | 0.25 | 0.866 | 0.019 | 0 | 0.99 |
| WMC | 0 | 0 | 1 | 0.019 | 0 | 0.99 |

TABLE C.2 – Résultats de la classification avec Bayésien naïf et les métriques BIN-KM5 & BIN-MEAN à 70%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.025 | 0 | 0.987 | 0.02 | 0 | 0.99 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0.015 | 0.143 | 0.919 |
| LOC | 0.06 | 0 | 0.97 | 0.009 | 0 | 0.995 |
| NOC | - | - | - | 0 | 0.188 | 0.902 |
| RFC | 0.013 | 0.125 | 0.929 | 0.017 | 0 | 0.991 |
| WMC | 0.012 | 0 | 0.994 | 0.012 | 0.25 | 0.861 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0.011 | 0.25 | 0.861 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 1 | 0 | 0.022 | 0.2 | 0.885 |
| LOC | 0 | 1 | 0 | 0 | 0 | 1 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 1 | 0 | 0 | 0.25 | 0.866 |
| WMC | 0 | 1 | 0 | 0.011 | 0 | 0.994 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.014 | 0 | 0.993 | 0 | 0.333 | 0.817 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.041 | 0 | 0.979 | 0.042 | 0 | 0.979 |
| LOC | 0.014 | 0 | 0.993 | 0.014 | 1 | 0 |
| NOC | 0 | 0.143 | 0.926 | 0 | 0.3 | 0.837 |
| RFC | 0.063 | 0.056 | 0.941 | 0.044 | 0.125 | 0.915 |
| WMC | 0.009 | 0 | 0.995 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 1 | 0 |
| DIT | 0 | 0.048 | 0.976 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0.009 | 0.333 | 0.813 |
| LOC | 0 | 1 | 0 | 0 | 1 | 0 |
| NOC | - | - | - | 0.04 | 0 | 0.98 |
| RFC | 0 | 1 | 0 | 0.009 | 1 | 0 |
| WMC | 0 | 1 | 0 | 0.018 | 1 | 0.991 |

TABLE C.3 – Résultats de la classification avec Bayésien naïf et les métriques BIN-KM5 & BIN-MEAN à 80%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | - | - | - |
| DIT | - | - | - | - | - | - |
| LCOM | 0.049 | 0 | 0.975 | - | - | - |
| LOC | 0.057 | 0 | 0.971 | - | - | - |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.071 | 0.964 | - | - | - |
| WMC | 0.024 | 0 | 0.988 | - | - | - |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | - | - | - | 0.011 | 1 | 0 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | - | - | - | 0 | 1 | 0 |
| LOC | - | - | - | 0.011 | 1 | 0.994 |
| NOC | 0.031 | 0.333 | 0.804 | 0 | 0.444 | 0.746 |
| RFC | 0 | - | - | 0 | 1 | 0 |
| WMC | 0 | - | - | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.333 | 0.817 | 0 | 0.333 | 0.817 |
| DIT | 0 | 0 | 1 | 0 | 0.2 | 0.894 |
| LCOM | 0.009 | 0.333 | 0.813 | 0.014 | 1 | 0.993 |
| LOC | 0.013 | 0 | 0.993 | 0 | 1 | 0 |
| NOC | 0.009 | 0 | 0.995 | 0.014 | 0 | 0.993 |
| RFC | 0 | 1 | 0 | 0 | 0.333 | 0.817 |
| WMC | 0.004 | 0 | 0.998 | 0 | 1 | 0 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | - | - | - |
| DIT | 0 | 0.048 | 0.976 | 0 | 0 | 1 |
| LCOM | 0.017 | 1 | 0 | - | - | - |
| LOC | - | - | - | - | - | - |
| NOC | 0 | 0.778 | 0.471 | 0.049 | 0 | 0.975 |
| RFC | - | - | - | - | - | - |
| WMC | - | - | - | - | - | - |

TABLE C.4 – Résultats de la classification avec Bayésien naïf et les métriques BIN-KM5 & BIN-MEAN à 90%

# Classification avec Machine à vecteurs de support

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 1 | 0 | 0 | 0 | 0.586 | 0.643 |
| DIT | - | - | - | - | - | - |
| LCOM | 1 | 0 | 0 | 0 | 0.973 | 0.164 |
| LOC | 0 | 0.288 | 0.844 | 0 | 0.638 | 0.602 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.379 | 0.788 | 0 | 0.764 | 0.486 |
| WMC | 0 | 0.281 | 0.848 | 0 | 0.704 | 0.544 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.8 | 0.447 | 0 | 0.417 | 0.764 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.957 | 0.207 | 0 | 0.975 | 0.158 |
| LOC | 0 | 0.895 | 0.324 | 0 | 0.5 | 0.707 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.953 | 0.277 | 0 | 0.026 | 0.987 |
| WMC | 1 | 0 | 0 | 0 | 0.618 | 0.618 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.069 | 0.965 | 0 | 0 | 0.969 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.966 | 0.184 | 0 | 0.655 | 0.587 |
| LOC | 0 | 0.577 | 0.65 | 0 | 0.783 | 0.466 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.189 | 0.901 | 0 | 0.323 | 0.823 |
| WMC | 0 | 0.667 | 0.577 | 0 | 0.375 | 0.791 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.407 | 0.77 | 0 | 0.46 | 0.735 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.69 | 0.557 | 1 | 0 | 0 |
| LOC | 0 | 0.895 | 0.324 | 0 | 0.9 | 0.316 |
| NOC | - | - | - | 0 | 0.85 | 0.387 |
| RFC | 0 | 0.333 | 0.817 | 0 | 0.818 | 0.427 |
| WMC | 0 | 0.833 | 0.409 | 0 | 0.952 | 0.219 |

TABLE C.5 – Résultats de la classification avec Machine à vecteurs de support et les métriques BIN-KM5 & BIN-MEAN à 30%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.206 | 0.891 | 0 | 0.733 | 0.517 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0.951 | 0.221 | 0 | 0.938 | 0.251 |
| LOC | 0 | 0.129 | 0.933 | 0 | 0.455 | 0.738 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.6 | 0.632 | 0 | 0.824 | 0.42 |
| WMC | 0 | 0.323 | 0.823 | 0 | 0.786 | 0.463 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0 | 0.9 | 0.316 |
| DIT | - | - | - | 0.016 | 0 | 0.992 |
| LCOM | 0 | 0.5 | 0.707 | 0 | 0.9 | 0.316 |
| LOC | 0 | 1 | 0 | 0 | 0.667 | 0.577 |
| NOC | 0 | 0.923 | 0.277 | - | - | - |
| RFC | 0 | 1 | 0 | 0 | 0.857 | 0.378 |
| WMC | 0 | 0 | 1 | 0 | 0.875 | 0.354 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.526 | 0.688 | 0 | 1 | 0 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 0.824 | 0.42 | 0 | 0.833 | 0.409 |
| LOC | 0 | 0.733 | 0.517 | 0 | 0.8 | 0.447 |
| NOC | 0 | 0.913 | 0.295 | - | - | - |
| RFC | 0.074 | 0 | 0.962 | - | - | - |
| WMC | 0 | 0.786 | 0.463 | 0 | 1 | 0 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.667 | 0.577 | 0 | 0.833 | 0.409 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.333 | 0.817 | 0 | 0.833 | 0.409 |
| LOC | 0 | 1 | 0 | 0 | 1 | 0 |
| NOC | - | - | - | 0 | 0.826 | 0.417 |
| RFC | 0 | 1 | 0 | 0 | 0.857 | 0.378 |
| WMC | 0 | 1 | 0 | 0 | 0.833 | 0.409 |

TABLE C.6 – Résultats de la classification avec Machine à vecteurs de support et les métriques BIN-KM5 & BIN-MEAN à 70%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.003 | 0 | 0.998 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.061 | 0.969 | 0 | 0 | 1 |
| LOC | 0.012 | 0 | 0.994 | 0.003 | 0 | 0.998 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0.013 | 0.094 | 0.946 | 0.003 | 0 | 0.998 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0.011 | 0.25 | 0.861 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 1 | 0 | 0.022 | 0.4 | 0.766 |
| LOC | 0 | 1 | 0 | 0 | 0 | 1 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 1 | 0 | 0 | 0 | 1 |
| WMC | 0 | 1 | 0 | 0 | 0.25 | 0.866 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.333 | 0.816 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.111 | 0.943 | 0 | 0 | 1 |
| LOC | 0.005 | 0.125 | 0.933 | 0.014 | 0.5 | 0.702 |
| NOC | 0 | 0 | 1 | 0 | 0 | 1 |
| RFC | 0 | 0.111 | 0.943 | 0.029 | 0 | 0.985 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0.5 | 0.704 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 1 | 0 | 0.009 | 1 | 0 |
| NOC | - | - | - | 0 | 0.077 | 0.961 |
| RFC | 0 | 1 | 0 | 0.009 | 0.5 | 0.704 |
| WMC | 0 | 1 | 0 | 0 | 0.5 | 0.707 |

TABLE C.7 – Résultats de la classification avec Machine à vecteurs de support et les métriques BIN-KM5 & BIN-MEAN à 80%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.04 | 0.98 | - | - | - |
| DIT | - | - | - | - | - | - |
| LCOM | 0.012 | 0.069 | 0.959 | - | - | - |
| LOC | 0 | 0.042 | 0.979 | - | - | - |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.036 | 0.982 | - | - | - |
| WMC | 0.012 | 0 | 0.994 | - | - | - |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | - | - | - | 0.011 | 0.5 | 0.703 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | - | - | - | 0 | 1 | 0 |
| LOC | - | - | - | 0 | 1 | 0 |
| NOC | 0 | 0.333 | 0.817 | 0 | 0.111 | 0.943 |
| RFC | - | - | - | 0 | 1 | 0 |
| WMC | - | - | - | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.333 | 0.817 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0.014 | 0 | 0.993 |
| LOC | 0.004 | 0.333 | 0.815 | 0 | 1 | 0 |
| NOC | 0.005 | 0.125 | 0.933 | 0 | 0 | 1 |
| RFC | 0 | 0.8 | 0.447 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.2 | 0.894 | - | - | - |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0.017 | 1 | 0 | - | - | - |
| LOC | - | - | - | - | - | - |
| NOC | 0 | 0 | 1 | 0 | 0 | 1 |
| RFC | - | - | - | - | - | - |
| WMC | - | - | - | - | - | - |

TABLE C.8 – Résultats de la classification avec Machine à vecteurs de support et les métriques BIN-KM5 & BIN-MEAN à 90%

# Classification avec Forêt aléatoire

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0 | 1 | 0 | 0 | 1 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.017 | 0.991 | 0 | 0 | 1 |
| WMC | 0.019 | 0 | 0.99 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0.053 | 0.973 | 0 | 0 | 1 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0.026 | 0.987 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0 | 1 | 0.019 | 0.043 | 0.969 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0.007 | 0 | 0.996 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0.025 | 0 | 0.987 | 0 | 0 | 1 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0.029 | 0 | 0.985 | 0 | 0 | 1 |
| WMC | 0 | 0.056 | 0.972 | 0 | 0 | 1 |

TABLE C.9 – Résultats de la classification avec Forêt aléatoire et les métriques BIN-KM5 & BIN-MEAN à 30%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.003 | 0 | 0.998 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0.014 | 0 | 0.993 | 0.003 | 0 | 0.998 |
| LOC | 0.013 | 0.032 | 0.978 | 0.003 | 0 | 0.998 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0.003 | 0.071 | 0.962 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.016 | 0.333 | 0.81 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.5 | 0.707 | 0.012 | 0 | 0.994 |
| LOC | 0.015 | 0.5 | 0.702 | 0 | 0.167 | 0.913 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 0.5 | 0.707 | 0 | 0.071 | 0.964 |
| WMC | 0 | 0.5 | 0.707 | 0 | 0.125 | 0.935 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0.005 | 0.059 | 0.968 | 0 | 0.167 | 0.913 |
| LOC | 0.005 | 0.067 | 0.964 | 0 | 0.2 | 0.894 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 0 | 1 | - | - | - |
| WMC | 0 | 0.071 | 0.964 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0.167 | 0.909 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0.333 | 0.817 | 0 | 0.333 | 0.817 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0 | 0.25 | 0.866 | 0.009 | 0 | 0.995 |
| WMC | 0 | 0.333 | 0.817 | 0.009 | 0.167 | 0.909 |

TABLE C.10 – Résultats de la classification avec Forêt aléatoire et les métriques BIN-KM5 & BIN-MEAN à 70%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.003 | 0 | 0.994 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.03 | 0.985 | 0 | 0 | 1 |
| LOC | 0.012 | 0 | 0.994 | 0.003 | 0 | 0.994 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0 | 0 | 1 | 0.003 | 0 | 0.994 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0.011 | 0.25 | 0.861 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 1 | 0 | 0.022 | 0.2 | 0.885 |
| LOC | 0 | 1 | 0 | 0 | 0 | 1 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 1 | 0 | 0 | 0 | 1 |
| WMC | 0 | 1 | 0 | 0 | 0.25 | 0.866 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.005 | 0 | 0.997 | 0.014 | 0.333 | 0.811 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.005 | 0.111 | 0.941 | 0 | 0 | 1 |
| LOC | 0.005 | 0.125 | 0.933 | 0 | 0.5 | 0.707 |
| NOC | 0 | 0 | 1 | 0 | 0 | 1 |
| RFC | 0 | 0.056 | 0.972 | 0 | 0.125 | 0.935 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0.5 | 0.704 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 1 | 0 | 0.009 | 1 | 0 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0 | 1 | 0 | 0.009 | 0.5 | 0.704 |
| WMC | 0 | 1 | 0 | 0 | 0.5 | 0.707 |

TABLE C.11 – Résultats de la classification avec Forêt aléatoire et les métriques BIN-KM5 & BIN-MEAN à 80%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | - | - | - |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | - | - | - |
| LOC | 0 | 0.042 | 0.979 | - | - | - |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | - | - | - |
| WMC | 0.012 | 0 | 0.994 | - | - | - |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | - | - | - | 0.011 | 0.5 | 0.703 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | - | - | - | 0 | 1 | 0 |
| LOC | - | - | - | 0.011 | 1 | 0.994 |
| NOC | 0 | 0.333 | 0.817 | 0 | 0 | 1 |
| RFC | - | - | - | 0 | 0.5 | 0.707 |
| WMC | - | - | - | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.014 | 0.333 | 0.811 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0 | 0.5 | 0.707 |
| LOC | 0.004 | 0.333 | 0.815 | 0 | 1 | 0 |
| NOC | 0.005 | 0 | 0.997 | 0 | 0 | 1 |
| RFC | 0.005 | 0 | 0.997 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.019 | 0.2 | 0.886 | - | - | - |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0.017 | 1 | 0 | - | - | - |
| LOC | - | - | - | - | - | - |
| NOC | 0 | 0 | 1 | 0 | 0 | 1 |
| RFC | - | - | - | - | - | - |
| WMC | - | - | - | - | - | - |

TABLE C.12 – Résultats de la classification avec Forêt aléatoire et les métriques BIN-KM5 & BIN-MEAN à 90%

# Classification avec Perceptron multicouche

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.039 | 0.007 | 0.977 |
| DIT | - | - | - | - | - | - |
| LCOM | 0.196 | 0.3 | 0.75 | 0.132 | 0.288 | 0.786 |
| LOC | 0 | 0.019 | 0.99 | 0.008 | 0.038 | 0.977 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.017 | 0.991 | 0.031 | 0.065 | 0.952 |
| WMC | 0 | 0.018 | 0.991 | 0 | 0.043 | 0.978 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.017 | 0.083 | 0.949 |
| DIT | - | - | - | - | - | - |
| LCOM | 0.023 | 0.478 | 0.714 | 0 | 0.4 | 0.775 |
| LOC | 0.021 | 0.105 | 0.936 | 0 | 0.033 | 0.983 |
| NOC | - | - | - | - | - | - |
| RFC | 0.049 | 0.038 | 0.956 | 0 | 0.026 | 0.987 |
| WMC | 1 | 0 | 0 | 0.033 | 0.029 | 0.969 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0.058 | 0.307 | 0.808 | 0.021 | 0.103 | 0.937 |
| LOC | 0 | 0.085 | 0.957 | 0.038 | 0 | 0.981 |
| NOC | - | - | - | - | - | - |
| RFC | 0.007 | 0 | 0.996 | 0 | 0 | 1 |
| WMC | 0 | 0.053 | 0.973 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0.1 | 0.241 | 0.826 | 0.811 | 0.164 | 0.397 |
| LOC | 0.025 | 0.053 | 0.961 | 0.024 | 0.067 | 0.954 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0.029 | 0 | 0.985 | 0 | 0.114 | 0.941 |
| WMC | 0 | 0.056 | 0.972 | 0.042 | 0.024 | 0.967 |

TABLE C.13 – Résultats de la classification avec Perceptron multicouche et les métriques BIN-KM5 & BIN-MEAN à 30%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0.195 | 0.897 | 0 | 0.063 | 0.969 |
| LOC | 0.013 | 0.032 | 0.978 | 0.003 | 0.091 | 0.952 |
| NOC | - | - | - | - | - | - |
| RFC | 0.013 | 0 | 0.993 | 0.003 | 0 | 0.998 |
| WMC | 0 | 0.032 | 0.984 | 0 | 0.071 | 0.964 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.016 | 0.333 | 0.81 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.5 | 0.707 | 0 | 0.3 | 0.837 |
| LOC | 0.015 | 0.5 | 0.702 | 0 | 0.167 | 0.913 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 0.5 | 0.707 | 0 | 0.071 | 0.964 |
| WMC | 0 | 0.5 | 0.707 | 0 | 0.125 | 0.935 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.125 | 0.935 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0.01 | 0 | 0.995 | 0 | 0.167 | 0.913 |
| LOC | 0.005 | 0.067 | 0.964 | 0.014 | 0.2 | 0.993 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 0 | 1 | - | - | - |
| WMC | 0.005 | 0.143 | 0.923 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0.167 | 0.909 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0.333 | 0.817 |
| LOC | 0 | 0.333 | 0.817 | 0 | 0.333 | 0.817 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0 | 0.25 | 0.866 | 0.009 | 0.143 | 0.922 |
| WMC | 0 | 0.333 | 0.817 | 0.009 | 0.167 | 0.909 |

TABLE C.14 – Résultats de la classification avec Perceptron multicouche et les métriques BIN-KM5 & BIN-MEAN à 70%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.003 | 0 | 0.998 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.061 | 0.969 | 0 | 0 | 1 |
| LOC | 0.012 | 0 | 0.994 | 0.003 | 0 | 0.998 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0.013 | 0.094 | 0.946 | 0.003 | 0 | 0.998 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0.011 | 0.25 | 0.861 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 1 | 0 | 0.022 | 0.4 | 0.766 |
| LOC | 0 | 1 | 0 | 0 | 0 | 1 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 1 | 0 | 0 | 0 | 1 |
| WMC | 0 | 1 | 0 | 0 | 0.25 | 0.866 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.333 | 0.817 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.111 | 0.943 | 0 | 0 | 1 |
| LOC | 0.005 | 0.125 | 0.933 | 0.014 | 0.5 | 0.702 |
| NOC | 0 | 0 | 1 | 0 | 0 | 1 |
| RFC | 0 | 0.111 | 0.943 | 0.029 | 0 | 0.985 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0.05 | 0.704 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 1 | 0 | 0.009 | 1 | 0 |
| NOC | - | - | - | 0 | 0.077 | 0.961 |
| RFC | 0 | 1 | 0 | 0.009 | 0.5 | 0.704 |
| WMC | 0 | 1 | 0 | 0 | 0.5 | 0.707 |

TABLE C.15 – Résultats de la classification avec Perceptron multicouche et les métriques BIN-KM5 & BIN-MEAN à 80%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.04 | 0.98 | - | - | - |
| DIT | - | - | - | - | - | - |
| LCOM | 0.012 | 0.069 | 0.959 | - | - | - |
| LOC | 0 | 0.042 | 0.979 | - | - | - |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.036 | 0.982 | - | - | - |
| WMC | 0.012 | 0 | 0.994 | - | - | - |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | - | - | - | 0.011 | 0.5 | 0.703 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | - | - | - | 0 | 1 | 0 |
| LOC | - | - | - | 0 | 1 | 0 |
| NOC | 0 | 0.333 | 0.817 | 0 | 0.111 | 0.943 |
| RFC | - | - | - | 0 | 1 | 0 |
| WMC | - | - | - | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.333 | 0.817 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0.014 | 0 | 0.993 |
| LOC | 0.004 | 0.333 | 0.815 | 0 | 1 | 0 |
| NOC | 0.005 | 0.125 | 0.933 | 0 | 0 | 1 |
| RFC | 0 | 0.8 | 0.447 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.2 | 0.894 | - | - | - |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0.017 | 1 | 0 | - | - | - |
| LOC | - | - | - | - | - | - |
| NOC | 0 | 0 | 1 | 0 | 0 | 1 |
| RFC | - | - | - | - | - | - |
| WMC | - | - | - | - | - | - |

TABLE C.16 – Résultats de la classification avec Perceptron multicouche et les métriques BIN-KM5 & BIN-MEAN à 90%

# Classification avec K plus proches voisins

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0 | 1 | 0 | 0 | 1 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0.017 | 0.991 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0 | 0.017 | 0.991 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0.053 | 0.973 | 0 | 0 | 1 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0.026 | 0.987 |
| WMC | 0 | 0 | 1 | 0 | 0.029 | 0.985 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0 | 1 | 0.019 | 0.043 | 0.969 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0.007 | 0 | 0.996 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0 | 1 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0.025 | 0 | 0.987 | 0 | 0 | 1 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0.029 | 0 | 0.985 | 0 | 0 | 1 |
| WMC | 0 | 0.056 | 0.972 | 0 | 0 | 1 |

TABLE C.17 – Résultats de la classification avec K plus proches voisins et les métriques BIN-KM5 & BIN-MEAN à 30%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.067 | 0.966 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0.014 | 0 | 0.993 | 0.003 | 0 | 0.998 |
| LOC | 0.013 | 0.032 | 0.978 | 0.003 | 0 | 0.998 |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0.003 | 0.071 | 0.962 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.016 | 0.333 | 0.81 | 0 | 0 | 1 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0 | 0.5 | 0.707 | 0.012 | 0 | 0.994 |
| LOC | 0.015 | 0.5 | 0.702 | 0 | 0.167 | 0.913 |
| NOC | 0 | 0.231 | 0.877 | - | - | - |
| RFC | 0 | 0.5 | 0.707 | 0 | 0.071 | 0.964 |
| WMC | 0 | 0.5 | 0.707 | 0 | 0.125 | 0.935 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.053 | 0.973 | 0 | 0 | 1 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0.005 | 0.059 | 0.968 | 0 | 0.167 | 0.913 |
| LOC | 0.005 | 0.067 | 0.964 | 0 | 0.2 | 0.894 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 0 | 1 | - | - | - |
| WMC | 0 | 0.071 | 0.964 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0.167 | 0.909 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0.333 | 0.817 | 0 | 0.333 | 0.817 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0 | 0.25 | 0.866 | 0.009 | 0 | 0.995 |
| WMC | 0 | 0.333 | 0.817 | 0.009 | 0.167 | 0.909 |

TABLE C.18 – Résultats de la classification avec K plus proches voisins et les métriques BIN-KM5 & BIN-MEAN à 70%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.003 | 0 | 0.998 |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0.03 | 0.985 | 0 | 0 | 1 |
| LOC | 0.012 | 0 | 0.994 | 0.003 | 0 | 0.998 |
| NOC | - | - | - | 0 | 0 | 1 |
| RFC | 0 | 0 | 1 | 0.003 | 0 | 0.998 |
| WMC | 0 | 0.036 | 0.982 | 0 | 0 | 1 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0.011 | 0.25 | 0.861 |
| DIT | 0 | 0 | 1 | - | - | - |
| LCOM | 0 | 1 | 0 | 0.022 | 0.2 | 0.885 |
| LOC | 0 | 1 | 0 | 0 | 0 | 1 |
| NOC | 0 | 0 | 1 | - | - | - |
| RFC | 0 | 1 | 0 | 0 | 0 | 1 |
| WMC | 0 | 1 | 0 | 0 | 0.25 | 0.866 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.005 | 0 | 0.997 | 0 | 0.333 | 0.817 |
| DIT | - | - | - | 0 | 0 | 1 |
| LCOM | 0.005 | 0.111 | 0.941 | 0 | 0 | 1 |
| LOC | 0.005 | 0.125 | 0.933 | 0 | 0.5 | 0.707 |
| NOC | 0 | 0 | 1 | 0 | 0 | 1 |
| RFC | 0 | 0.056 | 0.972 | 0 | 0.125 | 0.935 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0.009 | 0.5 | 0.704 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0 | 0 | 1 |
| LOC | 0 | 0 | 1 | 0.009 | 1 | 0 |
| NOC | - | - | - | 0 | 0.077 | 0.961 |
| RFC | 0 | 1 | 0 | 0.009 | 0.5 | 0.704 |
| WMC | 0 | 1 | 0 | 0 | 0.5 | 0.707 |

TABLE C.19 – Résultats de la classification avec K plus proches voisins et les métriques BIN-KM5 & BIN-MEAN à 80%

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.04 | 0.98 | - | - | - |
| DIT | - | - | - | - | - | - |
| LCOM | 0 | 0 | 1 | - | - | - |
| LOC | 0 | 0.042 | 0.979 | - | - | - |
| NOC | - | - | - | - | - | - |
| RFC | 0 | 0 | 1 | - | - | - |
| WMC | 0.012 | 0 | 0.994 | - | - | - |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | - | - | - | 0.011 | 0.5 | 0.703 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | - | - | - | 0 | 1 | 0 |
| LOC | - | - | - | 0.011 | 1 | 0 |
| NOC | 0 | 0.333 | 0.817 | 0 | 0 | 1 |
| RFC | - | - | - | 0 | 0.5 | 0.707 |
| WMC | - | - | - | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0 | 1 | 0 | 0.333 | 0.817 |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0 | 0 | 1 | 0 | 0.5 | 0.707 |
| LOC | 0.004 | 0.333 | 0.815 | 0 | 1 | 0 |
| NOC | 0.005 | 0 | 0.997 | 0 | 0 | 1 |
| RFC | 0 | 0 | 1 | 0 | 0 | 1 |
| WMC | 0 | 0 | 1 | 0 | 0 | 1 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.2 | 0.894 | - | - | - |
| DIT | 0 | 0 | 1 | 0 | 0 | 1 |
| LCOM | 0.017 | 1 | 0.991 | - | - | - |
| LOC | - | - | - | - | - | - |
| NOC | 0 | 0 | 1 | 0 | 0 | 1 |
| RFC | - | - | - | - | - | - |
| WMC | - | - | - | - | - | - |

TABLE C.20 – Résultats de la classification avec K plus proches voisins et les métriques BIN-KM5 & BIN-MEAN à 90%

# *ANNEXE. D*

## *Prédiction de l'effort de test : Valeurs brutes des métriques (BIN-KM5)*

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.1 | 0.75 | 0.474 | 0.048 | 0.85 | 0.378 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.038 | 0.844 | 0.387 | 0.016 | 0.85 | 0.384 |
| LOC | 0.114 | 0.594 | 0.6 | 0.023 | 0.775 | 0.469 |
| NOC | 0.013 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0.025 | 0.969 | 0.174 | 0.035 | 0.825 | 0.411 |
| WMC | 0.101 | 0.656 | 0.556 | 0.042 | 0.7 | 0.536 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.065 | 0.619 | 0.597 | 0 | 1 | 0 |
| DIT | 0.152 | 0.429 | 0.696 | 0.1 | 0.72 | 0.502 |
| LCOM | 0.022 | 0.667 | 0.571 | 0.871 | 0.2 | 0.321 |
| LOC | 0.022 | 0.667 | 0.571 | 0.043 | 0.96 | 0.196 |
| NOC | 0.065 | 0.952 | 0.212 | 0 | 1 | 0 |
| RFC | 0.043 | 0.714 | 0.523 | 0.1 | 0.76 | 0.465 |
| WMC | 0.022 | 0.571 | 0.648 | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.084 | 0.847 | 0.374 | 0.17 | 0.478 | 0.658 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.03 | 0.831 | 0.405 | 0.057 | 0.522 | 0.671 |
| LOC | 0.066 | 0.729 | 0.503 | 0.038 | 0.696 | 0.541 |
| NOC | 0.024 | 0.983 | 0.129 | 0.906 | 0.13 | 0.286 |
| RFC | 0 | 1 | 0 | 0.113 | 0.87 | 0.34 |
| WMC | 0.054 | 0.729 | 0.506 | 0.038 | 0.391 | 0.765 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.1 | 0.737 | 0.765 | 0.053 | 0.842 | 0.387 |
| DIT | 0.325 | 0.316 | 0.679 | 0 | 1 | 0 |
| LCOM | 0.075 | 0.632 | 0.583 | 0.21 | 0.895 | 0.321 |
| LOC | 0.05 | 0.632 | 0.591 | 0.042 | 0.842 | 0.389 |
| NOC | 0.05 | 0.895 | 0.316 | 0 | 1 | 0 |
| RFC | 0.125 | 0.526 | 0.644 | 0.042 | 0.947 | 0.225 |
| WMC | 0.075 | 0.632 | 0.583 | 0.063 | 0.842 | 0.385 |

TABLE D.1 – Résultats de la classification avec Bayésien naïf et la métrique BIN-KM5

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0 | 1 | 0 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0 | 0.969 | 0.176 | 0 | 1 | 0 |
| LOC | 0.025 | 0.938 | 0.248 | 0 | 1 | 0 |
| NOC | 0.013 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0 | 1 | 0 |
| WMC | 0.013 | 0.938 | 0.249 | 0 | 1 | 0 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.022 | 0.905 | 0.305 | 0 | 1 | 0 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0 | 0.905 | 0.308 | 0 | 1 | 0 |
| LOC | 0 | 0.905 | 0.308 | 0 | 1 | 0 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0 | 1 | 0 |
| WMC | 0 | 0.857 | 0.378 | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0 | 1 | 0 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0 | 0.949 | 0.256 | 0.019 | 0.783 | 0.461 |
| LOC | 0 | 1 | 0 | 0.019 | 0.87 | 0.357 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0 | 1 | 0 |
| WMC | 0 | 0.949 | 0.226 | 0.038 | 0.565 | 0.647 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.025 | 1 | 0 | 0 | 1 | 0 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.025 | 0.789 | 0.454 | 0 | 0.947 | 0.23 |
| LOC | 0 | 0.895 | 0.324 | 0 | 1 | 0 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 0.947 | 0.23 | 0 | 1 | 0 |
| WMC | 0 | 0.895 | 0.324 | 0 | 1 | 0 |

TABLE D.2 – Résultats de la classification avec Machine à vecteurs de support BIN-KM5

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.152 | 0.531 | 0.631 | 0.016 | 0.9 | 0.313 |
| DIT | 0.063 | 0.938 | 0.243 | 0 | 1 | 0 |
| LCOM | 0.266 | 0.688 | 0.479 | 0.029 | 0.825 | 0.412 |
| LOC | 0.253 | 0.5 | 0.611 | 0.09 | 0.85 | 0.369 |
| NOC | 0.051 | 0.938 | 0.245 | 0.003 | 1 | 0 |
| RFC | 0.266 | 0.719 | 0.454 | 0.068 | 0.8 | 0.432 |
| WMC | 0.165 | 0.469 | 0.666 | 0.048 | 0.675 | 0.556 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.333 | 0.817 | 0.086 | 0.68 | 0.541 |
| DIT | 0.152 | 0.429 | 0.696 | 0.071 | 0.84 | 0.386 |
| LCOM | 0.087 | 0.381 | 0.752 | 0.157 | 0.56 | 0.609 |
| LOC | 0.152 | 0.19 | 0.829 | 0.214 | 0.76 | 0.434 |
| NOC | 0.087 | 1 | 0 | 0.057 | 0.96 | 0.194 |
| RFC | 0.087 | 0.429 | 0.722 | 0.143 | 0.76 | 0.454 |
| WMC | 0.152 | 0.333 | 0.752 | 0.2 | 0.72 | 0.473 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.132 | 0.831 | 0.383 | 0.189 | 0.478 | 0.651 |
| DIT | 0 | 1 | 0 | 0.038 | 0.478 | 0.709 |
| LCOM | 0.12 | 0.729 | 0.488 | 0.151 | 0.348 | 0.744 |
| LOC | 0.246 | 0.576 | 0.565 | 0.113 | 0.261 | 0.81 |
| NOC | 0.042 | 0.915 | 0.285 | 0.019 | 1 | 0 |
| RFC | 0.228 | 0.712 | 0.472 | 0.189 | 0.478 | 0.651 |
| WMC | 0.132 | 0.61 | 0.582 | 0.113 | 0.261 | 0.81 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.2 | 0.632 | 0.543 | 0.053 | 0.947 | 0.224 |
| DIT | 0.3 | 0.632 | 0.508 | 0 | 1 | 0 |
| LCOM | 0.15 | 0.684 | 0.518 | 0.063 | 0.684 | 0.544 |
| LOC | 0.15 | 0.474 | 0.669 | 0.084 | 0.737 | 0.491 |
| NOC | 0 | 1 | 0 | 0.053 | 0.842 | 0.387 |
| RFC | 0.275 | 0.526 | 0.586 | 0.084 | 0.789 | 0.44 |
| WMC | 0.15 | 0.421 | 0.702 | 0.126 | 0.895 | 0.303 |

TABLE D.3 – Résultats de la classification avec Forêt aléatoire et la métrique BIN-KM5

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.038 | 0.969 | 0.173 | 0.003 | 0.875 | 0.353 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.127 | 0.656 | 0.548 | 0.016 | 0.825 | 0.415 |
| LOC | 0.241 | 0.406 | 0.671 | 0.01 | 0.85 | 0.385 |
| NOC | 0.025 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0.006 | 0.875 | 0.352 |
| WMC | 0.253 | 0.438 | 0.649 | 0.01 | 0.8 | 0.445 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.13 | 0.381 | 0.734 | 0 | 1 | 0 |
| DIT | 0.152 | 0.429 | 0.696 | 0.1 | 0.8 | 0.424 |
| LCOM | 0.043 | 0.81 | 0.426 | 0.029 | 0.96 | 0.197 |
| LOC | 0.13 | 0.333 | 0.762 | 0.1 | 0.92 | 0.268 |
| NOC | 0.043 | 1 | 0 | 0.014 | 1 | 0 |
| RFC | 0.087 | 0.619 | 0.59 | 0.129 | 0.64 | 0.56 |
| WMC | 0.065 | 0.286 | 0.817 | 0.043 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.036 | 1 | 0 | 0.151 | 0.522 | 0.637 |
| DIT | 0 | 1 | 0 | 0.189 | 0.739 | 0.46 |
| LCOM | 0.042 | 0.763 | 0.476 | 0.057 | 0.261 | 0.835 |
| LOC | 0.108 | 0.61 | 0.59 | 0.094 | 0.13 | 0.888 |
| NOC | 0.006 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0.283 | 0.478 | 0.612 |
| WMC | 0.096 | 0.627 | 0.581 | 0.075 | 0.13 | 0.897 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.225 | 0.474 | 0.638 | 0.021 | 1 | 0 |
| DIT | 0.375 | 0.526 | 0.544 | 0 | 1 | 0 |
| LCOM | 0.075 | 0.684 | 0.541 | 0 | 0.842 | 0.397 |
| LOC | 0.125 | 0.368 | 0.744 | 0.021 | 0.895 | 0.324 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0.15 | 0.368 | 0.733 | 0.021 | 1 | 0 |
| WMC | 0.1 | 0.368 | 0.754 | 0 | 0.895 | 0.324 |

TABLE D.4 – Résultats de la classification avec Perceptron multicouche BIN-KM5

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.139 | 0.625 | 0.568 | 0.016 | 0.9 | 0.314 |
| DIT | 0.025 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.215 | 0.813 | 0.384 | 0.029 | 0.825 | 0.4122 |
| LOC | 0.228 | 0.531 | 0.602 | 0.077 | 0.875 | 0.34 |
| NOC | 0.063 | 0.969 | 0.17 | 0.003 | 1 | 0 |
| RFC | 0.266 | 0.781 | 0.401 | 0.061 | 0.8 | 0.433 |
| WMC | 0.139 | 0.531 | 0.635 | 0.029 | 0.725 | 0.517 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.333 | 0.817 | 0.071 | 0.76 | 0.472 |
| DIT | 0.152 | 0.429 | 0.696 | 0.057 | 0.84 | 0.388 |
| LCOM | 0.087 | 0.381 | 0.752 | 0.143 | 0.56 | 0.614 |
| LOC | 0.152 | 0.19 | 0.829 | 0.229 | 0.76 | 0.43 |
| NOC | 0.022 | 1 | 0 | 0.043 | 1 | 0 |
| RFC | 0.087 | 0.429 | 0.722 | 0.143 | 0.76 | 0.454 |
| WMC | 0.152 | 0.429 | 0.696 | 0.143 | 0.84 | 0.37 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.09 | 0.898 | 0.305 | 0.151 | 0.522 | 0.637 |
| DIT | 0 | 1 | 0 | 0.038 | 0.478 | 0.709 |
| LCOM | 0.096 | 0.746 | 0.479 | 0.151 | 0.348 | 0.744 |
| LOC | 0.198 | 0.627 | 0.547 | 0.113 | 0.261 | 0.81 |
| NOC | 0.042 | 0.915 | 0.285 | 0.019 | 1 | 0 |
| RFC | 0.21 | 0.712 | 0.477 | 0.17 | 0.478 | 0.658 |
| WMC | 0.108 | 0.678 | 0.536 | 0.094 | 0.304 | 0.794 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.125 | 0.737 | 0.48 | 0.053 | 0.947 | 0.224 |
| DIT | 0.175 | 0.947 | 0.21 | 0 | 1 | 0 |
| LCOM | 0.15 | 0.684 | 0.518 | 0.053 | 0.684 | 0.547 |
| LOC | 0.175 | 0.526 | 0.625 | 0.074 | 0.737 | 0.493 |
| NOC | 0 | 1 | 0 | 0.042 | 0.842 | 0.389 |
| RFC | 0.275 | 0.579 | 0.552 | 0.074 | 0.789 | 0.442 |
| WMC | 0.15 | 0.474 | 0.669 | 0.116 | 0.947 | 0.216 |

TABLE D.5 – Résultats de la classification avec K plus proches voisins BIN-KM5

# *ANNEXE. E*

## *Prédiction de l'effort de test : Valeurs brutes des métriques (BIN-MEAN)*

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.035 | 1 | 0 | 0.051 | 0.848 | 0.38 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.047 | 0.808 | 0.428 | 0.015 | 0.911 | 0.296 |
| LOC | 0.059 | 0.808 | 0.425 | 0.033 | 0.835 | 0.399 |
| NOC | 0.012 | 0.962 | 0.194 | 0 | 1 | 0 |
| RFC | 0.012 | 1 | 0 | 0.044 | 0.823 | 0.411 |
| WMC | 0.071 | 0.808 | 0.422 | 0.044 | 0.709 | 0.527 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.078 | 0.563 | 0.635 | 0 | 1 | 0 |
| DIT | 0.02 | 1 | 0 | 0.066 | 0.789 | 0.444 |
| LCOM | 0 | 0.5 | 0.707 | 0.079 | 0.895 | 0.311 |
| LOC | 0.02 | 0.563 | 0.655 | 0.026 | 0.947 | 0.227 |
| NOC | 0.02 | 0.938 | 0.248 | 0.013 | 1 | 0 |
| RFC | 0.039 | 0.75 | 0.49 | 0.105 | 0.895 | 0.307 |
| WMC | 0.039 | 0.563 | 0.649 | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.071 | 0.786 | 0.446 | 0.085 | 0.882 | 0.329 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.024 | 0.786 | 0.457 | 0.034 | 0.412 | 0.754 |
| LOC | 0.053 | 0.661 | 0.791 | 0.034 | 1 | 0 |
| NOC | 0.018 | 0.982 | 0.133 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0.119 | 0.706 | 0.509 |
| WMC | 0.041 | 0.679 | 0.555 | 0.068 | 0.529 | 0.663 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.07 | 1 | 0 | 0.033 | 0.864 | 0.363 |
| DIT | 0.302 | 0.313 | 0.693 | 0 | 1 | 0 |
| LCOM | 0.07 | 0.563 | 0.638 | 0.043 | 0.864 | 0.361 |
| LOC | 0.047 | 0.625 | 0.598 | 0.033 | 1 | 0 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0.07 | 0.563 | 0.638 | 0.043 | 0.955 | 0.208 |
| WMC | 0.07 | 0.625 | 0.591 | 0.043 | 0.909 | 0.295 |

TABLE E.1 – Résultats de la classification avec Bayésien naïf et la métrique BIN-MEAN

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0 | 1 | 0 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0 | 1 | 0 | 0 | 0.987 | 0.114 |
| LOC | 0 | 1 | 0 | 0.004 | 0.987 | 0.114 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0 | 1 | 0 |
| WMC | 0 | 1 | 0 | 0 | 0.949 | 0.226 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.938 | 0.251 | 0 | 1 | 0 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0 | 0.938 | 0.251 | 0 | 1 | 0 |
| LOC | 0 | 0.938 | 0.251 | 0 | 1 | 0 |
| NOC | 0 | 1 | 0 | 0.013 | 1 | 0 |
| RFC | 0 | 0.938 | 0.251 | 0 | 1 | 0 |
| WMC | 0 | 0.938 | 0.251 | 0 | 1 | 0 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0 | 1 | 0 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0 | 0.946 | 0.232 | 0.017 | 0.706 | 0.538 |
| LOC | 0.006 | 0.946 | 0.232 | 0.034 | 1 | 0 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0 | 1 | 0 |
| WMC | 0 | 0.946 | 0.232 | 0.034 | 0.824 | 0.412 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0 | 1 | 0 |
| DIT | 0 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.023 | 0.75 | 0.494 | 0 | 0.995 | 0.212 |
| LOC | 0 | 0.875 | 0.354 | 0 | 1 | 0 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 0.938 | 0.251 | 0 | 1 | 0 |
| WMC | 0 | 0.875 | 0.354 | 0 | 1 | 0 |

TABLE E.2 – Résultats de la classification avec Machine à vecteurs de support BIN-MEAN

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.153 | 0.846 | 0.361 | 0.092 | 0.873 | 0.34 |
| DIT | 0.035 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.2 | 0.808 | 0.392 | 0.066 | 0.861 | 0.36 |
| LOC | 0.247 | 0.654 | 0.51 | 0.169 | 0.823 | 0.384 |
| NOC | 0.047 | 1 | 0 | 0.018 | 1 | 0 |
| RFC | 0.212 | 0.808 | 0.389 | 0.096 | 0.722 | 0.501 |
| WMC | 0.2 | 0.769 | 0.43 | 0.092 | 0.785 | 0.442 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.078 | 0.375 | 0.759 | 0.145 | 0.842 | 0.368 |
| DIT | 0.157 | 0.313 | 0.762 | 0.039 | 1 | 0 |
| LCOM | 0.078 | 0.438 | 0.72 | 0.132 | 0.684 | 0.524 |
| LOC | 0.157 | 0.375 | 0.726 | 0.184 | 0.789 | 0.415 |
| NOC | 0 | 1 | 0 | 0.039 | 1 | 0 |
| RFC | 0.078 | 0.5 | 0.679 | 0.105 | 0.632 | 0.574 |
| WMC | 0.118 | 0.313 | 0.607 | 0.105 | 0.632 | 0.574 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.118 | 0.75 | 0.47 | 0.085 | 0.765 | 0.464 |
| DIT | 0 | 1 | 0 | 0.034 | 0.294 | 0.826 |
| LCOM | 0.076 | 0.643 | 0.574 | 0.068 | 0.588 | 0.62 |
| LOC | 0.147 | 0.607 | 0.579 | 0.237 | 0.647 | 0.519 |
| NOC | 0.035 | 0.875 | 0.347 | 0 | 1 | 0 |
| RFC | 0.188 | 0.607 | 0.565 | 0.169 | 0.647 | 0.542 |
| WMC | 0.118 | 0.554 | 0.627 | 0.153 | 0.529 | 0.632 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.233 | 0.75 | 0.438 | 0.076 | 0.818 | 0.41 |
| DIT | 0.163 | 0.875 | 0.323 | 0 | 1 | 0 |
| LCOM | 0.116 | 0.625 | 0.576 | 0.087 | 0.864 | 0.352 |
| LOC | 0.093 | 0.313 | 0.79 | 0.12 | 0.773 | 0.447 |
| NOC | 0 | 1 | 0 | 0.054 | 0.955 | 0.206 |
| RFC | 0.186 | 0.438 | 0.677 | 0.141 | 0.636 | 0.559 |
| WMC | 0.14 | 0.438 | 0.696 | 0.185 | 1 | 0 |

TABLE E.3 – Résultats de la classification avec Forêt aléatoire et la métrique BIN-MEAN

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 1 | 0 | 0.044 | 0.848 | 0.381 |
| DIT | 0.024 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.024 | 0.923 | 0.274 | 0.015 | 0.924 | 0.274 |
| LOC | 0.024 | 1 | 0 | 0.048 | 0.785 | 0.452 |
| NOC | 0.024 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0.059 | 0.747 | 0.488 |
| WMC | 0 | 1 | 0 | 0.04 | 0.722 | 0.517 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.098 | 0.313 | 0.788 | 0 | 1 | 0 |
| DIT | 0.157 | 0.313 | 0.762 | 0.039 | 1 | 0 |
| LCOM | 0 | 0.5 | 0.707 | 0.013 | 0.947 | 0.229 |
| LOC | 0.078 | 0.5 | 0.679 | 0.092 | 0.842 | 0.379 |
| NOC | 0 | 1 | 0 | 0.026 | 1 | 0 |
| RFC | 0.039 | 0.5 | 0.693 | 0.132 | 0.737 | 0.478 |
| WMC | 0.059 | 0.438 | 0.728 | 0.066 | 0.947 | 0.222 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.047 | 0.821 | 0.413 | 0.102 | 0.882 | 0.326 |
| DIT | 0 | 1 | 0 | 0.203 | 0.294 | 0.75 |
| LCOM | 0.053 | 0.661 | 0.567 | 0.051 | 0.412 | 0.747 |
| LOC | 0.071 | 0.607 | 0.604 | 0.153 | 0.529 | 0.632 |
| NOC | 0.018 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0 | 1 | 0 | 0.068 | 0.941 | 0.234 |
| WMC | 0.076 | 0.536 | 0.655 | 0.136 | 0.412 | 0.713 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.07 | 1 | 0 | 0.033 | 1 | 0 |
| DIT | 0.233 | 0.813 | 0.38 | 0 | 1 | 0 |
| LCOM | 0.07 | 0.625 | 0.591 | 0 | 0.864 | 0.369 |
| LOC | 0.14 | 0.313 | 0.769 | 0.033 | 1 | 0 |
| NOC | 0 | 1 | 0 | 0 | 1 | 0 |
| RFC | 0.14 | 0.375 | 0.733 | 0.022 | 1 | 0 |
| WMC | 0.093 | 0.25 | 0.825 | 0.022 | 1 | 0 |

TABLE E.4 – Résultats de la classification avec Perceptron multicouche BIN-MEAN

| Systèmes | ANT | | | POI | | |
|---|---|---|---|---|---|---|
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.094 | 0.885 | 0.323 | 0.04 | 0.937 | 0.246 |
| DIT | 0.035 | 1 | 0 | 0 | 1 | 0 |
| LCOM | 0.165 | 0.846 | 0.359 | 0.059 | 0.861 | 0.362 |
| LOC | 0.224 | 0.654 | 0.518 | 0.147 | 0.848 | 0.36 |
| NOC | 0.012 | 1 | 0 | 0.015 | 1 | 0 |
| RFC | 0.2 | 0.846 | 0.351 | 0.066 | 0.734 | 0.498 |
| WMC | 0.129 | 0.846 | 0.366 | 0.063 | 0.81 | 0.422 |
| Systèmes | IO | | | IVY | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0 | 0.375 | 0.791 | 0.132 | 0.895 | 0.302 |
| DIT | 0.157 | 0.313 | 0.762 | 0.039 | 1 | 0 |
| LCOM | 0.059 | 0.438 | 0.728 | 0.132 | 0.684 | 0.524 |
| LOC | 0.157 | 0.375 | 0.726 | 0.197 | 0.789 | 0.412 |
| NOC | 0 | 1 | 0 | 0.026 | 1 | 0 |
| RFC | 0.098 | 0.5 | 0.672 | 0.092 | 0.684 | 0.536 |
| WMC | 0.098 | 0.375 | 0.751 | 0.079 | 0.632 | 0.582 |
| Systèmes | JFC | | | JODA | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.088 | 0.875 | 0.338 | 0.085 | 0.706 | 0.519 |
| DIT | 0 | 1 | 0 | 0.034 | 0.294 | 0.826 |
| LCOM | 0.076 | 0.661 | 0.56 | 0.068 | 0.588 | 0.62 |
| LOC | 0.135 | 0.679 | 0.527 | 0.237 | 0.647 | 0.519 |
| NOC | 0.029 | 0.911 | 0.294 | 0 | 1 | 0 |
| RFC | 0.141 | 0.679 | 0.525 | 0.169 | 0.647 | 0.542 |
| WMC | 0.088 | 0.589 | 0.612 | 0.136 | 0.529 | 0.638 |
| Systèmes | MATH | | | LUCENE | | |
| Métriques | FPR | FNR | g-mean | FPR | FNR | g-mean |
| CBO | 0.116 | 0.75 | 0.47 | 0.076 | 0.818 | 0.41 |
| DIT | 0.093 | 0.938 | 0.239 | 0 | 1 | 0 |
| LCOM | 0.093 | 0.625 | 0.583 | 0.076 | 0.864 | 0.354 |
| LOC | 0.093 | 0.375 | 0.753 | 0.12 | 0.773 | 0.447 |
| NOC | 0 | 1 | 0 | 0.043 | 1 | 0 |
| RFC | 0.186 | 0.5 | 0.638 | 0.12 | 0.636 | 0.566 |
| WMC | 0.093 | 0.438 | 0.715 | 0.141 | 1 | 0 |

TABLE E.5 – Résultats de la classification avec K plus proches voisins BIN-MEAN

# *ANNEXE. F*

## *Méthodes de comparaison de performance : Tests de Friedman et de Nemenyi*

# Tests de Friedman et de Nemenyi avec g-mean d'Alves Rankings(30%), Courbes ROC et des valeurs brutes des métriques. (BIN-KM5)

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.80 | 8.4 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.20 | | | ROC | .702 | - |
| ANT_RAW | 1 | | | RAW | .004 | .043 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.10 | 9.579 | .008 | Méthode | ALVES | ROC |
| POI_ROC | 2.90 | | | ROC | .533 | - |
| POI_RAW | 1.00 | | | RAW | .06 | .002 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 2.00 | 7.111 | .029 | Méthode | ALVES | ROC |
| IO_ROC | 2.80 | | | ROC | .244 | - |
| IO_RAW | 1.20 | | | RAW | .244 | .004 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.20 | 8.400 | .015 | Méthode | ALVES | ROC |
| IVY_ROC | 2.80 | | | ROC | .702 | - |
| IVY_RAW | 1.0 | | | RAW | .043 | .004 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.60 | 7.600 | .022 | Méthode | ALVES | ROC |
| JFC_ROC | 2.40 | | | ROC | .961 | - |
| JFC_RAW | 1.0 | | | RAW | .021 | .009 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.30 | 8.316 | .016 | Méthode | ALVES | ROC |
| JODA_ROC | 2.70 | | | ROC | .533 | - |
| JODA_RAW | 1.00 | | | RAW | .06 | .002 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.25 | 3.714 | .156 | Méthode | ALVES | ROC |
| MATH_ROC | 2.75 | | | ROC | - | - |
| MATH_RAW | 1.00 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.25 | 6.00 | .05 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.50 | | | ROC | .702 | - |
| LUCENE_RAW | 1.0 | | | RAW | .043 | .004 |

TABLE F.1 – Résultats des tests de Friedman et Nemenyi pour Bayésien naïf avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.3 | 3.895 | .143 | Méthode | ALVES | ROC |
| ANT_ROC | 2.4 | | | ROC | - | - |
| ANT_RAW | 1.3 | | | RAW | - | - |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| POI_ROC | 2.5 | | | ROC | 1 | - |
| POI_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.8 | 1.333 | .513 | Méthode | ALVES | ROC |
| IO_ROC | 2.4 | | | ROC | - | - |
| IO_RAW | 1.8 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| IVY_ROC | 2.4 | | | ROC | .961 | - |
| IVY_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.6 | 3.6 | .165 | Méthode | Alves | ROC |
| JFC_ROC | 2 | | | ROC | - | - |
| JFC_RAW | 1.4 | | | RAW | - | - |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| JODA_ROC | 2.4 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.25 | 3.714 | .156 | Méthode | ALVES | ROC |
| MATH_ROC | 2.75 | | | ROC | - | - |
| MATH_RAW | 1 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.5 | 6 | .05 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.5 | | | ROC | .533 | - |
| LUCENE_RAW | 1 | | | RAW | .147 | .009 |

TABLE F.2 – Résultats des tests de Friedman et Nemenyi pour Machine à vecteurs de support avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| ANT_ROC | 2.5 | | | ROC | 1 | - |
| ANT_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.7 | 9.294 | .01 | Méthode | ALVES | ROC |
| POI_ROC | 2.3 | | | ROC | .854 | - |
| POI_RAW | 1 | | | RAW | .006 | .03 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| IO_ROC | 2.5 | | | ROC | .854 | - |
| IO_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.8 | 9.333 | .009 | Méthode | ALVES | ROC |
| IVY_ROC | 2.2 | | | ROC | .702 | - |
| IVY_RAW | 1 | | | RAW | .004 | .043 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| JODA_ROC | 2.4 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .021 | .01 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.5 | 4 | .135 | Méthode | ALVES | ROC |
| MATH_ROC | 2.5 | | | ROC | - | - |
| MATH_RAW | 1 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.63 | 7.538 | .023 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.38 | | | ROC | .854 | - |
| LUCENE_RAW | 1 | | | RAW | .03 | .006 |

TABLE F.3 – Résultats des tests de Friedman et Nemenyi pour Forêt aléatoire avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| ANT_ROC | 2.7 | | | ROC | .854 | - |
| ANT_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.4 | 8.444 | .015 | Méthode | ALVES | ROC |
| POI_ROC | 2.6 | | | ROC | .961 | - |
| POI_RAW | 1 | | | RAW | .021 | .01 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 2.2 | 5.778 | .056 | Méthode | ALVES | ROC |
| IO_ROC | 2.6 | | | ROC | - | - |
| IO_RAW | 1.2 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.2 | 8.4 | .015 | Méthode | ALVES | ROC |
| IVY_ROC | 2.8 | | | ROC | .702 | - |
| IVY_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | .961 | - |
| JFC_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| JODA_ROC | 2.5 | | | ROC | .854 | - |
| JODA_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.25 | 3.714 | .156 | Méthode | ALVES | ROC |
| MATH_ROC | 2.75 | | | ROC | - | - |
| MATH_RAW | 1 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.63 | | | ROC | .452 | - |
| LUCENE_RAW | 1 | | | RAW | .111 | .004 |

TABLE F.4 – Résultats des tests de Friedman et Nemenyi pour Perceptron multicouche avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.4 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| POI_ROC | 2.4 | | | ROC | .961 | - |
| POI_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| IO_ROC | 2.5 | | | ROC | .854 | - |
| IO_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.7 | 8.316 | .016 | Méthode | ALVES | ROC |
| IVY_ROC | 2.3 | | | ROC | .854 | - |
| IVY_RAW | 1 | | | RAW | .006 | .03 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| JODA_ROC | 2.4 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.75 | 3.714 | .156 | Méthode | ALVES | ROC |
| MATH_ROC | 2.25 | | | ROC | - | - |
| MATH_RAW | 1 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.88 | 7.6 | .022 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.13 | | | ROC | .533 | - |
| LUCENE_RAW | 1 | | | RAW | .002 | .06 |

TABLE F.5 – Résultats des tests de Friedman et Nemenyi pour K plus proches voisins avec BIN-KM5

# Tests de Friedman et de Nemenyi avec g-mean d'Alves Rankings(70%), Courbes ROC et des valeurs brutes des métriques. (BIN-KM5)

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.8 | 8.4 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.2 | | | ROC | .702 | - |
| ANT_RAW | 1 | | | RAW | .004 | .043 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.58 | 9.478 | .009 | Méthode | ALVES | ROC |
| POI_ROC | 2.42 | | | ROC | 1 | - |
| POI_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.2 | 8.4 | .015 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .009 | - |
| IO_RAW | 1.8 | | | RAW | .961 | .021 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.42 | 9.478 | .009 | Méthode | ALVES | ROC |
| IVY_ROC | 2.58 | | | ROC | .854 | - |
| IVY_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 3 | 10 | .007 | Méthode | ALVES | ROC |
| JFC_ROC | 2 | | | ROC | .702 | - |
| JFC_RAW | 1 | | | RAW | .004 | .043 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2 | 8 | .018 | Méthode | ALVES | ROC |
| JODA_ROC | 3 | | | ROC | .375 | - |
| JODA_RAW | 1 | | | RAW | .082 | .002 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| MATH_ROC | 2.5 | | | ROC | 1 | - |
| MATH_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALES | 3 | 8 | .018 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | .961 | - |
| LUCENE_RAW | 1 | | | RAW | .009 | .021 |

TABLE F.6 – Résultats des tests de Friedman et Nemenyi pour Bayésien naïf avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| ANT_ROC | 2.6 | | | ROC | .0961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.42 | 9.478 | .009 | Méthode | ALVES | ROC |
| POI_ROC | 2.58 | | | ROC | .854 | - |
| POI_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.9 | .316 | .854 | Méthode | ALVES | ROC |
| IO_ROC | 2.2 | | | ROC | - | - |
| IO_RAW | 1.9 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.25 | 10.174 | .006 | Méthode | ALVES | ROC |
| IVY_ROC | 2.75 | | | ROC | .533 | - |
| IVY_RAW | 1 | | | RAW | .06 | .002 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_AL | 2.6 | 5.2 | .074 | Méthode | ALVES | ROC |
| JFC_ROC | 2.2 | | | ROC | - | - |
| JFC_RAW | 1.2 | | | RAW | - | - |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_AL | 1.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| JODA_ROC | 3 | | | ROC | .111 | - |
| JODA_RAW | 1.63 | | | RAW | .702 | .014 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_AL | 1.6 | 2.8 | .247 | Méthode | ALVES | ROC |
| MATH_ROC | 2.6 | | | ROC | - | - |
| MATH_RAW | 1.8 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | 2.38 | 4.933 | .085 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.5 | | | ROC | - | - |
| LUCENE_RAW | 1.13 | | | RAW | - | - |

TABLE F.7 – Résultats des tests de Friedman et Nemenyi pour Machine à vecteurs de support avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.4 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.6 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.33 | 10.182 | .006 | Méthode | ALVES | ROC |
| POI_ROC | 2.67 | | | ROC | .702 | - |
| POI_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1 | 10 | .007 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .004 | - |
| IO_RAW | 2 | | | RAW | .702 | .0.43 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.33 | 10.182 | .006 | Méthode | ALVES | ROC |
| IVY_ROC | 2.67 | | | ROC | .702 | - |
| IVY_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| JFC_ROC | 2.7 | | | ROC | .533 | - |
| JFC_RAW | 1 | | | RAW | .06 | .002 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| JODA_ROC | 2.63 | | | ROC | 854 | - |
| JODA_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.2 | 9.333 | .009 | Méthode | ALVES | ROC |
| MATH_ROC | 2.8 | | | ROC | .702 | - |
| MATH_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.25 | 6.5 | .039 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.75 | | | ROC | .375 | - |
| LUCENE_RAW | 1 | | | RAW | .082 | .002 |

TABLE F.8 – Résultats des tests de Friedman et Nemenyi pour Forêt aléatoire avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| ANT_ROC | 2.7 | | | ROC | .854 | - |
| ANT_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.58 | 9.478 | .009 | Méthode | ALVES | ROC |
| POI_ROC | 2.42 | | | ROC | 1 | - |
| POI_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .043 | - |
| IO_RAW | 1.4 | | | RAW | .702 | .003 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.42 | 9.478 | .009 | Méthode | ALVES | ROC |
| IVY_ROC | 2.58 | | | ROC | .854 | - |
| IVY_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | .961 | - |
| JFC_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.5 | 6 | .05 | Méthode | ALVES | ROC |
| JODA_ROC | 2.5 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.75 | 3.714 | .156 | Méthode | ALVES | ROC |
| MATH_ROC | 2.25 | | | ROC | - | - |
| MATH_RAW | 1 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2 | 8 | .018 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .021 | - |
| LUCENE_RAW | 1 | | | RAW | .854 | .082 |

TABLE F.9 – Résultats des tests de Friedman et Nemenyi pour Perceptron multicouche avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.4 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.6 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.5 | 9.818 | .007 | Méthode | ALVES | ROC |
| POI_ROC | 2.5 | | | ROC | .961 | - |
| POI_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.2 | 8.4 | .015 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .009 | - |
| IO_RAW | 1.8 | | | RAW | .961 | .021 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.33 | 10.182 | .006 | Méthode | ALVES | ROC |
| IVY_ROC | 2.67 | | | ROC | .702 | - |
| IVY_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.2 | 8.4 | .015 | Méthode | ALVES | ROC |
| JFC_ROC | 2.8 | | | ROC | .375 | - |
| JFC_RAW | 1 | | | RAW | .082 | .002 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| JODA_ROC | 2.63 | | | ROC | .854 | - |
| JODA_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| MATH_ROC | 2.7 | | | ROC | .854 | - |
| MATH_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.25 | 6.5 | .039 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.75 | | | ROC | .375 | - |
| LUCENE_RAW | 1 | | | RAW | .082 | .002 |

TABLE F.10 – Résultats des tests de Friedman et Nemenyi pour K plus proches voisins avec BIN-KM5

# Tests de Friedman et de Nemenyi avec g-mean d'Alves Rankings(80%), Courbes ROC et des valeurs brutes des métriques. (BIN-KM5)

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.8 | 8.4 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.2 | | | ROC | .702 | - |
| ANT_RAW | 1 | | | RAW | .004 | .043 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| POI_ROC | 2.4 | | | ROC | .961 | - |
| POI_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1 | 10 | .007 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .004 | - |
| IO_RAW | 2 | | | RAW | .702 | .043 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| IVY_ROC | 2.6 | | | ROC | .961 | - |
| IVY_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.9 | 9.579 | .008 | Méthode | ALVES | ROC |
| JFC_ROC | 2.1 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .006 | .03 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2 | 6.4 | .041 | Méthode | ALVES | ROC |
| JODA_ROC | 2.8 | | | ROC | .244 | - |
| JODA_RAW | 1.2 | | | RAW | .244 | .004 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.83 | 4.727 | .094 | Méthode | ALVES | ROC |
| MATH_ROC | 2.67 | | | ROC | - | - |
| MATH_RAW | 1.5 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2 | 2 | .368 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.5 | | | ROC | - | - |
| LUCENE_RAW | 1.5 | | | RAW | - | - |

TABLE F.11 – Résultats des tests de Friedman et Nemenyi pour Bayésien naïf avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 3 | 10 | .007 | Méthode | ALVES | ROC |
| ANT_ROC | 2 | | | ROC | .375 | - |
| ANT_RAW | 1 | | | RAW | .002 | .082 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 3 | 10 | .007 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | .375 | - |
| POI_RAW | 1 | | | RAW | .002 | .082 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.1 | 6.632 | .036 | Méthode | ALVES | ROC |
| IO_ROC | 2.6 | | | ROC | .03 | - |
| IO_RAW | 2.3 | | | RAW | .53 | .31 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 3 | 10 | .007 | Méthode | ALVES | ROC |
| IVY_ROC | 2 | | | ROC | .375 | - |
| IVY_RAW | 1 | | | RAW | .002 | .082 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 3 | 8.4 | .015 | Méthode | ALVES | ROC |
| JFC_ROC | 1.8 | | | ROC | .533 | - |
| JFC_RAW | 1.2 | | | RAW | .009 | .147 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.8 | 8.4 | .015 | Méthode | ALVES | ROC |
| JODA_ROC | 2.2 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.92 | 1.826 | .401 | Méthode | ALVES | ROC |
| MATH_ROC | 2.42 | | | ROC | - | - |
| MATH_RAW | 1.67 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.63 | 5.2 | .074 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.25 | | | ROC | - | - |
| LUCENE_RAW | 1.13 | | | RAW | - | - |

TABLE F.12 – Résultats des tests de Friedman et Nemenyi pour Machine à vecteurs de support avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.4 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| POI_ROC | 2.6 | | | ROC | .961 | - |
| POI_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1 | 10 | .007 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .004 | - |
| IO_RAW | 2 | | | RAW | .702 | .043 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.4 | 7.6 | .22 | Méthode | ALVES | ROC |
| IVY_ROC | 2.6 | | | ROC | .961 | - |
| IVY_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.1 | 9.579 | .008 | Méthode | ALVES | ROC |
| JFC_ROC | 2.9 | | | ROC | .244 | - |
| JFC_RAW | 1 | | | RAW | .111 | .001 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.2 | 5.2 | .074 | Méthode | ALVES | ROC |
| JODA_ROC | 2.6 | | | ROC | - | - |
| JODA_RAW | 1.2 | | | RAW | - | - |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.75 | 6 | .05 | Méthode | ALVES | ROC |
| MATH_ROC | 2.75 | | | ROC | .15 | - |
| MATH_RAW | 1.5 | | | RAW | .078 | .03 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 1.75 | 6.5 | .039 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .082 | - |
| LUCENE_RAW | 1.25 | | | RAW | .375 | .002 |

TABLE F.13 – Résultats des tests de Friedman et Nemenyi pour Forêt aléatoire avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.4 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.6 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 3 | 10 | .007 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | .375 | - |
| POI_RAW | 1 | | | RAW | .002 | .08 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1 | 10 | .007 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .004 | - |
| IO_RAW | 2 | | | RAW | .702 | .043 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| IVY_ROC | 2.6 | | | ROC | .961 | - |
| IVY_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | .961 | - |
| JFC_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.3 | 5.158 | .076 | Méthode | ALVES | ROC |
| JODA_ROC | 2.5 | | | ROC | - | - |
| JODA_RAW | 1.2 | | | RAW | - | - |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.83 | 4.727 | .094 | Méthode | ALVES | ROC |
| MATH_ROC | 2.67 | | | ROC | - | - |
| MATH_RAW | 1.5 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 1.75 | 6.5 | .039 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .082 | - |
| LUCENE_RAW | 1.25 | | | RAW | .375 | 2 |

TABLE F.14 – Résultats des tests de Friedman et Nemenyi pour Perceptron multicouche avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| ANT_ROC | 2.5 | | | ROC | 1 | - |
| ANT_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| POI_ROC | 2.5 | | | ROC | 1 | - |
| POI_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1 | 10 | .007 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .004 | - |
| IO_RAW | 2 | | | RAW | .702 | .043 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| IVY_ROC | 2.6 | | | ROC | .961 | - |
| IVY_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.1 | 9.579 | .008 | Méthode | ALVES | ROC |
| JFC_ROC | 2.9 | | | ROC | .244 | - |
| JFC_RAW | 1 | | | RAW | .111 | .001 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.2 | 5.2 | .074 | Méthode | ALVES | ROC |
| JODA_ROC | 2.6 | | | ROC | - | - |
| JODA_RAW | 1.2 | | | RAW | - | - |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.08 | 5.429 | .066 | Méthode | ALVES | ROC |
| MATH_ROC | 2.58 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | 1.75 | 6.5 | .039 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .082 | - |
| LUCENE_RAW | 1.25 | | | RAW | .375 | .002 |

TABLE F.15 – Résultats des tests de Friedman et Nemenyi pour K plus proches voisins avec BIN-KM5

# Tests de Friedman et de Nemenyi avec g-mean d'Alves Rankings(90%), Courbes ROC et des valeurs brutes des métriques. (BIN-KM5)

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.8 | 8.4 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.2 | | | ROC | .702 | - |
| ANT_RAW | 1 | | | RAW | .004 | .043 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 6 | .014 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | - | 5 | .025 | Méthode | ALVES | ROC |
| IO_ROC | 2 | | | ROC | - | - |
| IO_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.75 | 6 | .05 | Méthode | ALVES | ROC |
| IVY_ROC | 2.75 | | | ROC | .15 | - |
| IVY_RAW | 1.5 | | | RAW | .78 | .03 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.3 | 6.633 | .036 | Méthode | ALVES | ROC |
| JFC_ROC | 2.6 | | | ROC | .618 | - |
| JFC_RAW | 1.1 | | | RAW | .082 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 1.9 | 5.2 | .074 | Méthode | ALVES | ROC |
| JODA_ROC | 2.8 | | | ROC | - | - |
| JODA_RAW | 1.4 | | | RAW | - | - |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2 | 3.2 | .202 | Méthode | ALVES | ROC |
| MATH_ROC | 2.67 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | 4 | .046 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | - | - |
| LUCENE_RAW | 1 | | | RAW | - | .008 |

TABLE F.16 – Résultats des tests de Friedman et Nemenyi pour Bayésien naïf avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 3 | 10 | .007 | Méthode | ALVES | ROC |
| ANT_ROC | 2 | | | ROC | .375 | - |
| ANT_RAW | 1 | | | RAW | .002 | .082 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 6 | .014 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | - | .2 | .655 | Méthode | ALVES | ROC |
| IO_ROC | 1.6 | | | ROC | - | - |
| IO_RAW | 1.4 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.83 | 8.4 | .015 | Méthode | ALVES | ROC |
| IVY_ROC | 2.83 | | | ROC | .147 | - |
| IVY_RAW | 1.33 | | | RAW | .533 | .01 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.8 | 6.4 | .041 | Méthode | ALVES | ROC |
| JFC_ROC | 2 | | | ROC | .854 | - |
| JFC_RAW | 1.2 | | | RAW | .021 | .082 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.4 | 4.8 | .091 | Méthode | ALVES | ROC |
| JODA_ROC | 2.4 | | | ROC | - | - |
| JODA_RAW | 1.2 | | | RAW | - | - |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.17 | 2.364 | .307 | Méthode | ALVES | ROC |
| MATH_ROC | 2.5 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | 4 | .046 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | - | - |
| LUCENE_RAW | 1 | | | RAW | - | .008 |

TABLE F.17 – Résultats des tests de Friedman et Nemenyi pour Machine à vecteurs de support avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| ANT_ROC | 2.5 | | | ROC | 1 | - |
| ANT_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | | 6 | .014 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | | 5 | .025 | Méthode | ALVES | ROC |
| IO_ROC | 2 | | | ROC | - | - |
| IO_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.92 | 6.348 | .042 | Méthode | ALVES | ROC |
| IVY_ROC | 2.75 | | | ROC | .244 | - |
| IVY_RAW | 1.33 | | | RAW | .452 | .014 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 1.9 | 4.526 | .104 | Méthode | ALVES | ROC |
| JODA_ROC | 2.7 | | | ROC | - | - |
| JODA_RAW | 1.4 | | | RAW | - | - |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.83 | 3.818 | .148 | Méthode | ALVES | ROC |
| MATH_ROC | 2.83 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | 4 | .046 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | - | - |
| LUCENE_RAW | 1 | | | RAW | - | .008 |

TABLE F.18 – Résultats des tests de Friedman et Nemenyi pour Forêt aléatoire avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.2 | 8.4 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.8 | | | ROC | .702 | - |
| ANT_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 6 | .014 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | - | 5 | .025 | Méthode | ALVES | ROC |
| IO_ROC | 2 | | | ROC | - | - |
| IO_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.5 | 8.273 | .016 | Méthode | ALVES | ROC |
| IVY_ROC | 2.92 | | | ROC | .03 | - |
| IVY_RAW | 1.58 | | | RAW | .99 | .021 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | .961 | - |
| JFC_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.4 | 4.8 | .091 | Méthode | ALVES | ROC |
| JODA_ROC | 2.4 | | | ROC | - | - |
| JODA_RAW | 1.2 | | | RAW | - | - |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.83 | 3.818 | .148 | Méthode | ALVES | ROC |
| MATH_ROC | 2.83 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | 4 | .046 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | - | - |
| LUCENE_RAW | 1 | | | RAW | - | .008 |

TABLE F.19 – Résultats des tests de Friedman et Nemenyi pour Perceptron multicouche avec BIN-KM5

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.4 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.6 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 6 | .014 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | - | 5 | .025 | Méthode | ALVES | ROC |
| IO_ROC | 2 | | | ROC | - | - |
| IO_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.58 | 7.913 | .019 | Méthode | ALVES | ROC |
| IVY_ROC | 2.92 | | | ROC | .043 | - |
| IVY_RAW | 1.5 | | | RAW | .915 | .014 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 8.824 | .012 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_AL | 1.9 | 4.526 | .104 | Méthode | ALVES | ROC |
| JODA_ROC | 2.7 | | | ROC | - | - |
| JODA_RAW | 1.4 | | | RAW | - | - |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_AL | 2.5 | 4.909 | .086 | Méthode | ALVES | ROC |
| MATH_ROC | 2.5 | | | ROC | - | - |
| MATH_RAW | 1 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | - | 4 | .046 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | - | - |
| LUCENE_RAW | 1 | | | RAW | - | .008 |

TABLE F.20 – Résultats des tests de Friedman et Nemenyi pour K plus proches voisins avec BIN-KM5

# Tests de Friedman et de Nemenyi avec g-mean d'Alves Rankings(30%), Courbes ROC et des valeurs brutes des métriques. (BIN-MEAN)

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.6 | 5.2 | .074 | Méthode | ALVES | ROC |
| ANT_ROC | 2.2 | | | ROC | - | - |
| ANT_RAW | 1.2 | | | RAW | - | - |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.63 | 6.533 | .038 | Méthode | ALVES | ROC |
| POI_ROC | 2.38 | | | ROC | 1 | - |
| POI_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.9 | 2.211 | .331 | Méthode | ALVES | ROC |
| IO_ROC | 2.5 | | | ROC | - | - |
| IO_RAW | 1.6 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| IVY_ROC | 2.6 | | | ROC | .961 | - |
| IVY_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| JFC_ROC | 2.6 | | | ROC | .702 | - |
| JFC_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| JODA_ROC | 2.6 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| MATH_ROC | 2.5 | | | ROC | 1 | - |
| MATH_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.75 | 6.5 | .039 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.25 | | | ROC | .961 | - |
| LUCENE_RAW | 1 | | | RAW | .021 | .009 |

TABLE F.21 – Résultats des tests de Friedman et Nemenyi pour Bayésien naïf avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.5 | 5.444 | .066 | Méthode | ALVES | ROC |
| ANT_ROC | 2.3 | | | ROC | - | - |
| ANT_RAW | 1.2 | | | RAW | - | - |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.5 | 6 | .05 | Méthode | ALVES | ROC |
| POI_ROC | 2.5 | | | ROC | .961 | - |
| POI_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| OI_ALVES | 2.1 | 1.368 | .504 | Méthode | ALVES | ROC |
| OI_ROC | 2.3 | | | ROC | - | - |
| OI_RAW | 1.6 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| IVY_ROC | 2.5 | | | ROC | 1 | - |
| IVY_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_AL | 2.2 | 2.8 | .247 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | - | - |
| JFC_RAW | 1.4 | | | RAW | - | - |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_AL | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| JODA_ROC | 2.6 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_AL | 2.6 | 3.263 | .196 | Méthode | ALVES | ROC |
| MATH_ROC | 1.9 | | | ROC | - | - |
| MATH_RAW | 1.5 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | 2.75 | 5.733 | .057 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.13 | | | ROC | - | - |
| LUCENE_RAW | 1.13 | | | RAW | - | - |

TABLE F.22 – Résultats des tests de Friedman et Nemenyi pour Machine à vecteurs de support avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.7 | 8.316 | .016 | Méthode | ALVES | ROC |
| ANT_ROC | 2.3 | | | ROC | 1 | - |
| ANT_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.63 | 7.539 | .023 | Méthode | ALVES | ROC |
| POI_ROC | 2.38 | | | ROC | 1 | - |
| POI_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| IO_ROC | 2.4 | | | ROC | .961 | - |
| IO_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| IVY_ROC | 2.4 | | | ROC | .961 | - |
| IVY_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.6 | 9.5 | .009 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | .961 | - |
| JFC_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.7 | 8.316 | .016 | Méthode | ALVES | ROC |
| JODA_ROC | 2.3 | | | ROC | .854 | - |
| JODA_RAW | 1 | | | RAW | .006 | .03 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.2 | 9.333 | .009 | Méthode | ALVES | ROC |
| MATH_ROC | 2.8 | | | ROC | .702 | - |
| MATH_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.75 | 7.429 | .024 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.25 | | | ROC | .961 | - |
| LUCENE_RAW | 1 | | | RAW | .021 | .009 |

TABLE F.23 – Résultats des tests de Friedman et Nemenyi pour Forêt aléatoire avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| ANT_ROC | 2.5 | | | ROC | .854 | - |
| ANT_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.67 | 4.667 | .097 | Méthode | ALVES | ROC |
| POI_ROC | 2.33 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | - |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 2 | 7.111 | .029 | Méthode | ALVES | ROC |
| IO_ROC | 2.8 | | | ROC | .533 | - |
| IO_RAW | 1.2 | | | RAW | .147 | .009 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.2 | 8.4 | .015 | Méthode | ALVES | ROC |
| IVY_ROC | 2.8 | | | ROC | .702 | - |
| IVY_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 7.985 | .019 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| JODA_ROC | 2.5 | | | ROC | 1 | - |
| JODA_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_AL | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| MATH_ROC | 2.7 | | | ROC | .854 | - |
| MATH_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | 2.63 | 6.533 | .038 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.38 | | | ROC | .854 | - |
| LUCENE_RAW | 1 | | | RAW | .03 | .006 |

TABLE F.24 – Résultats des tests de Friedman et Nemenyi pour Perceptron multicouche avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.7 | 8.316 | .016 | Méthode | ALVES | ROC |
| ANT_ROC | 2.3 | | | ROC | 1 | - |
| ANT_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.38 | 7.539 | .023 | Méthode | ALVES | ROC |
| POI_ROC | 2.63 | | | ROC | .854 | - |
| POI_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| IO_ROC | 2.4 | | | ROC | .961 | - |
| IO_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| IVY_ROC | 2.5 | | | ROC | 1 | - |
| IVY_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_AL | 2.8 | 9.333 | .009 | Méthode | ALVES | ROC |
| JFC_ROC | 2.2 | | | ROC | .964 | - |
| JFC_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_AL | 2.7 | 8.316 | .016 | Méthode | ALVES | ROC |
| JODA_ROC | 2.3 | | | ROC | .854 | - |
| JODA_RAW | 1 | | | RAW | .006 | .03 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_AL | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| MATH_ROC | 2.7 | | | ROC | .854 | - |
| MATH_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | 2.75 | 7.429 | .024 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.25 | | | ROC | .961 | - |
| LUCENE_RAW | 1 | | | RAW | .021 | .009 |

TABLE F.25 – Résultats des tests de Friedman et Nemenyi pour K plus proches voisins avec BIN-MEAN

# Tests de Friedman et de Nemenyi avec g-mean d'Alves Rankings(70%), Courbes ROC et des valeurs brutes des métriques. (BIN-MEAN)

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.8 | 6.4 | .041 | Méthode | ALVES | ROC |
| ANT_ROC | 2 | | | ROC | .854 | - |
| ANT_RAW | 1.2 | | | RAW | .021 | .082 |
| **Système** | POI | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.9 | 9.579 | .008 | Méthode | ALVES | ROC |
| POI_ROC | 2.1 | | | ROC | .854 | - |
| POI_RAW | 1 | | | RAW | .006 | .03 |
| **Système** | IO | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.2 | 6.4 | .041 | Méthode | ALVES | ROC |
| IO_ROC | 2.8 | | | ROC | .082 | - |
| IO_RAW | 2 | | | RAW | .961 | .147 |
| **Système** | IVY | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.42 | 9.478 | .009 | Méthode | ALVES | ROC |
| IVY_ROC | 2.58 | | | ROC | .854 | - |
| IVY_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | JFC | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.8 | 8.4 | .015 | Méthode | ALVES | ROC |
| JFC_ROC | 2.2 | | | ROC | .961 | - |
| JFC_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | JODA | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2 | 8 | .018 | Méthode | ALVES | ROC |
| JODA_ROC | 3 | | | ROC | .702 | - |
| JODA_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | MATH | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| MATH_ROC | 2.5 | | | ROC | 1 | - |
| MATH_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | LUCENE | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 3 | 8 | .018 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | .961 | - |
| LUCENE_RAW | 1 | | | RAW | .009 | .021 |

TABLE F.26 – Résultats des tests de Friedman et Nemenyi pour Bayésien naïf avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.6 | 6.632 | .036 | Méthode | ALVES | ROC |
| ANT_ROC | 2.3 | | | ROC | .99 | - |
| ANT_RAW | 1.1 | | | RAW | .03 | .021 |
| **Système** | POI | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| POI_ROC | 2.7 | | | ROC | .533 | - |
| POI_RAW | 1 | | | RAW | .059 | .002 |
| **Système** | IO | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.4 | 3.6 | .165 | Méthode | ALVES | ROC |
| IO_ROC | 2.6 | | | ROC | - | - |
| IO_RAW | 2 | | | RAW | - | - |
| **Système** | IVY | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.25 | 10.174 | .006 | Méthode | ALVES | ROC |
| IVY_ROC | 2.75 | | | ROC | .533 | - |
| IVY_RAW | 1 | | | RAW | .06 | .002 |
| **Système** | JFC | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.4 | 4.8 | .091 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | - | - |
| JFC_RAW | 1.2 | | | RAW | - | - |
| **Système** | JODA | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 1.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| JODA_ROC | 3 | | | ROC | .31 | - |
| JODA_RAW | 1.63 | | | RAW | .53 | .03 |
| **Système** | MATH | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.8 | .737 | .692 | Méthode | ALVES | ROC |
| MATH_ROC | 2.3 | | | ROC | - | - |
| MATH_RAW | 1.9 | | | RAW | - | - |
| **Système** | LUCENE | | | Test de Nemenyi | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2.38 | 3.857 | .145 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.38 | | | ROC | - | - |
| LUCENE_RAW | 1.25 | | | RAW | - | - |

TABLE F.27 – Résultats des tests de Friedman et Nemenyi pour Machine à vecteurs de support avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.4 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.6 | | | ROC | .702 | - |
| ANT_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.4 | 8.444 | .015 | Méthode | ALVES | ROC |
| POI_ROC | 2.6 | | | ROC | .702 | - |
| POI_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ ALVES | 1.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| IO_ ROC | 3 | | | ROC | .142 | - |
| IO_ RAW | 1.4 | | | RAW | .533 | .009 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.33 | 10.182 | .006 | Méthode | ALVES | ROC |
| IVY_ROC | 2.67 | | | ROC | .702 | - |
| IVY_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_AL | 2.2 | 9.333 | .009 | Méthode | ALVES | ROC |
| JFC_ROC | 2.8 | | | ROC | .375 | - |
| JFC_RAW | 1 | | | RAW | .082 | .002 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_AL | 2.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| JODA_ROC | 2.63 | | | ROC | 1 | - |
| JODA_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_AL | 2.2 | 9.333 | .009 | Méthode | ALVES | ROC |
| MATH_ROC | 2.8 | | | ROC | .702 | - |
| MATH_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | 2 | 8 | .018 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .147 | - |
| LUCENE_RAW | 1 | | | RAW | .147 | .001 |

TABLE F.28 – Résultats des tests de Friedman et Nemenyi pour Forêt aléatoire avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| ANT_ROC | 2.5 | | | ROC | .854 | - |
| ANT_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.63 | 6.533 | .038 | Méthode | ALVES | ROC |
| POI_ROC | 2.38 | | | ROC | 1 | - |
| POI_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.7 | 8.316 | .016 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .191 | - |
| IO_RAW | 1.3 | | | RAW | .375 | .006 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.42 | 9.478 | .009 | Méthode | ALVES | ROC |
| IVY_ROC | 2.58 | | | ROC | .854 | - |
| IVY_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.25 | 6.5 | .039 | Méthode | ALVES | ROC |
| JODA_ROC | 2.75 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.1 | 6 | .05 | Méthode | ALVES | ROC |
| MATH_ROC | 2.7 | | | ROC | .702 | - |
| MATH_RAW | 1.2 | | | RAW | .111 | .014 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2 | 8 | .018 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .147 | - |
| LUCENE_RAW | 1 | | | RAW | .147 | .001 |

TABLE F.29 – Résultats des tests de Friedman et Nemenyi pour Perceptron multicouche avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.4 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.2 | 9.333 | .009 | Méthode | ALVES | ROC |
| POI_ROC | 2.8 | | | ROC | .375 | - |
| POI_RAW | 1 | | | RAW | .082 | .002 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| OI_ALVES | 1.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| IO_ROC | 3 | | | ROC | .082 | - |
| IO_RAW | 1.6 | | | RAW | .854 | .021 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.33 | 10.182 | .006 | Méthode | ALVES | ROC |
| IVY_ROC | 2.67 | | | ROC | .702 | - |
| IVY_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.1 | 9.579 | .008 | Méthode | ALVES | ROC |
| JFC_ROC | 2.9 | | | ROC | .244 | - |
| JFC_RAW | 1 | | | RAW | .111 | .001 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| JODA_ROC | 2.63 | | | ROC | 1 | - |
| JODA_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| MATH_ROC | 2.7 | | | ROC | .854 | - |
| MATH_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 2 | 8 | .018 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .142 | - |
| LUCENE_RAW | 1 | | | RAW | .142 | .001 |

TABLE F.30 – Résultats des tests de Friedman et Nemenyi pour K plus proches voisins avec BIN-KM5

# Tests de Friedman et de Nemenyi avec g-mean d'Alves Rankings(80%), Courbes ROC et des valeurs brutes des métriques. (BIN-MEAN)

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.8 | 6.4 | .041 | Méthode | ALVES | ROC |
| ANT_ROC | 2 | | | ROC | .854 | - |
| ANT_RAW | 1.2 | | | RAW | .021 | .082 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.75 | 6.5 | .039 | Méthode | ALVES | ROC |
| POI_ROC | 2.25 | | | ROC | .961 | - |
| POI_RAW | 1 | | | RAW | .009 | .021 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| OI_ALVES | 1.33 | 5.3 | .069 | Méthode | ALVES | ROC |
| OI_ROC | 2.67 | | | ROC | - | - |
| OI_RAW | 2 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| IVY_ROC | 2.6 | | | ROC | .961 | - |
| IVY_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_AL | 2.6 | 7.6 | .022 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | .961 | - |
| JFC_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.1 | 7.7 | .021 | Méthode | ALVES | ROC |
| JODA_ROC | 2.8 | | | ROC | .618 | - |
| JODA_RAW | 1.1 | | | RAW | .082 | .006 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_AL | 1.83 | 4.7 | .094 | Méthode | ALVES | ROC |
| MATH_ROC | 2.67 | | | ROC | - | - |
| MATH_RAW | 1.5 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | 1.13 | 7.6 | .022 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .014 | - |
| LUCENE_RAW | 1.88 | | | RAW | 1 | .014 |

TABLE F.31 – Résultats des tests de Friedman et Nemenyi pour Bayésien naïf avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.4 | 6.421 | .04 | Méthode | ALVES | ROC |
| ANT_ROC | 2.5 | | | ROC | .782 | - |
| ANT_RAW | 1.1 | | | RAW | .06 | .009 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 1.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| POI_ROC | 3 | | | ROC | .111 | - |
| POI_RAW | 1.63 | | | RAW | .702 | .014 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.57 | 2.571 | .276 | Méthode | ALVES | ROC |
| IO_ROC | 2.43 | | | ROC | - | - |
| IO_RAW | 2 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.8 | 7.6 | .039 | Méthode | ALVES | ROC |
| IVY_ROC | 2.8 | | | ROC | .375 | - |
| IVY_RAW | 1.4 | | | RAW | .375 | .021 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2 | 4 | .135 | Méthode | ALVES | ROC |
| JFC_ROC | 2.6 | | | ROC | - | - |
| JFC_RAW | 1.4 | | | RAW | - | - |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 1.58 | 7.143 | .028 | Méthode | ALVES | ROC |
| JODA_ROC | 2.83 | | | ROC | .06 | - |
| JODA_RAW | 1.58 | | | RAW | .96 | .03 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.58 | 2.818 | .244 | Méthode | ALVES | ROC |
| MATH_ROC | 2.5 | | | ROC | - | - |
| MATH_RAW | 1.92 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_AL | 1.63 | 6 | .05 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2.75 | | | ROC | .111 | - |
| LUCENE_RAW | 1.63 | | | RAW | .702 | .014 |

TABLE F.32 – Résultats des tests de Friedman et Nemenyi pour Machine à vecteurs de support avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.4 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .01 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.38 | 6.533 | .038 | Méthode | ALVES | ROC |
| POI_ROC | 2.63 | | | ROC | .854 | - |
| POI_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.25 | 8.957 | .011 | Méthode | ALVES | ROC |
| IO_ROC | 2.92 | | | ROC | .009 | - |
| IO_RAW | 1.83 | | | RAW | .782 | .06 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| IVY_ROC | 2.7 | | | ROC | .854 | - |
| IVY_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_AL | 2.2 | 8.4 | .015 | Méthode | ALVES | ROC |
| JFC_ROC | 2.8 | | | ROC | .375 | - |
| JFC_RAW | 1 | | | RAW | .082 | .002 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_AL | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| JODA_ROC | 2.6 | | | ROC | .961 | - |
| JODA_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.75 | 6 | .05 | Méthode | ALVES | ROC |
| MATH_ROC | 2.75 | | | ROC | .15 | - |
| MATH_RAW | 1.5 | | | RAW | .78 | .03 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 1.75 | 6.5 | .039 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .082 | - |
| LUCENE_RAW | 1.25 | | | RAW | .375 | .002 |

TABLE F.33 – Résultats des tests de Friedman et Nemenyi pour Forêt aléatoire avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.7 | 8.316 | .016 | Méthode | ALVES | ROC |
| ANT_ROC | 2.3 | | | ROC | 1 | - |
| ANT_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.9 | 9.579 | .008 | Méthode | ALVES | ROC |
| POI_ROC | 2.1 | | | ROC | .702 | - |
| POI_RAW | 1 | | | RAW | .004 | .043 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.58 | 5.304 | .07 | Méthode | ALVES | ROC |
| IO_ROC | 2.75 | | | ROC | - | - |
| IO_RAW | 1.67 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| IVY_ROC | 2.7 | | | ROC | .854 | - |
| IVY_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.58 | 9.478 | .009 | Méthode | ALVES | ROC |
| JODA_ROC | 2.42 | | | ROC | 1 | - |
| JODA_RAW | 1 | | | RAW | .014 | .014 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.67 | 5.6 | .061 | Méthode | ALVES | ROC |
| MATH_ROC | 2.33 | | | ROC | .244 | - |
| MATH_RAW | 1 | | | RAW | .702 | .043 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 1.88 | 7.6 | .022 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .043 | - |
| LUCENE_RAW | 1.13 | | | RAW | .004 | .702 |

TABLE F.34 – Résultats des tests de Friedman et Nemenyi pour Perceptron multicouche avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.4 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | 2.13 | 7.6 | .022 | Méthode | ALVES | ROC |
| POI_ROC | 2.88 | | | ROC | .533 | - |
| POI_RAW | 1 | | | RAW | .06 | .002 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | 1.25 | 8.957 | .011 | Méthode | ALVES | ROC |
| IO_ROC | 2.92 | | | ROC | .009 | - |
| IO_RAW | 1.83 | | | RAW | .782 | .06 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 2.3 | 8.316 | .016 | Méthode | ALVES | ROC |
| IVY_ROC | 2.7 | | | ROC | .854 | - |
| IVY_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.2 | 8.4 | .015 | Méthode | ALVES | ROC |
| JFC_ROC | 2.8 | | | ROC | .375 | - |
| JFC_RAW | 1 | | | RAW | .082 | .002 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.42 | 9.478 | .009 | Méthode | ALVES | ROC |
| JODA_ROC | 2.58 | | | ROC | .854 | - |
| JODA_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.08 | 5.429 | .066 | Méthode | ALVES | ROC |
| MATH_ROC | 2.58 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | 1.75 | 6.5 | .039 | Méthode | ALVES | ROC |
| LUCENE_ROC | 3 | | | ROC | .082 | - |
| LUCENE_RAW | 1.25 | | | RAW | .375 | .002 |

TABLE F.35 – Résultats des tests de Friedman et Nemenyi pour K plus proches voisins avec BIN-KM5

# Tests de Friedman et de Nemenyi avec g-mean d'Alves Rankings(90%), Courbes ROC et des valeurs brutes des métriques. (BIN-MEAN)

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.8 | 6.4 | .041 | Méthode | ALVES | ROC |
| ANT_ROC | 2 | | | ROC | .854 | - |
| ANT_RAW | 1.2 | | | RAW | .021 | .082 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 5 | .025 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | - | 2.667 | .103 | Méthode | ALVES | ROC |
| IO_ROC | 1.83 | | | ROC | - | - |
| IO_RAW | 1.17 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.42 | 8.857 | .012 | Méthode | ALVES | ROC |
| IVY_ROC | 2.92 | | | ROC | .021 | - |
| IVY_RAW | 1.67 | | | RAW | .99 | .03 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_AL | 2.3 | 6.632 | .036 | Méthode | ALVES | ROC |
| JFC_ROC | 2.6 | | | ROC | .618 | - |
| JFC_RAW | 1.1 | | | RAW | .082 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALES | 1.42 | 9.478 | .009 | Méthode | ALVES | ROC |
| JODA_ROC | 3 | | | ROC | .014 | - |
| JODA_RAW | 1.58 | | | RAW | 1 | .014 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 2.17 | 2.364 | .307 | Méthode | ALVES | ROC |
| MATH_ROC | 2.5 | | | ROC | .96 | - |
| MATH_RAW | 1.33 | | | RAW | .03 | .06 |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | 4 | .046 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | - | - |
| LUCENE_RAW | 1 | | | RAW | - | .008 |

TABLE F.36 – Résultats des tests de Friedman et Nemenyi pour Bayésien naïf avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.2 | 7.053 | .029 | Méthode | ALVES | ROC |
| ANT_ROC | 2.7 | | | ROC | .045 | - |
| ANT_RAW | 1.1 | | | RAW | .111 | .004 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 5 | .025 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| OI_ALVES | - | 2.667 | .102 | Méthode | ALVES | ROC |
| OI_ROC | 1.83 | | | ROC | - | - |
| OI_RAW | 1.17 | | | RAW | - | - |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.58 | 11.474 | .003 | Méthode | ALVES | ROC |
| IVY_ROC | 3 | | | ROC | .03 | - |
| IVY_RAW | 1.42 | | | RAW | .854 | .006 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 1.6 | 3.111 | .211 | Méthode | ALVES | ROC |
| JFC_ROC | 2.6 | | | ROC | - | - |
| JFC_RAW | 1.8 | | | RAW | - | - |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 1.33 | 11.2 | .004 | Méthode | ALVES | ROC |
| JODA_ROC | 3 | | | ROC | .01 | - |
| JODA_RAW | 1.67 | | | RAW | .961 | .021 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.67 | 3 | .223 | Méthode | ALVES | ROC |
| MATH_ROC | 2.67 | | | ROC | - | - |
| MATH_RAW | 1.67 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | 3 | .083 | Méthode | ALVES | ROC |
| LUCENE_ROC | 1.88 | | | ROC | - | - |
| LUCENE_RAW | 1.13 | | | RAW | - | - |

TABLE F.37 – Résultats des tests de Friedman et Nemenyi pour Machine à vecteurs de support avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| ANT_ROC | 2.4 | | | ROC | .961 | - |
| ANT_RAW | 1 | | | RAW | .021 | .01 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 5 | .025 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | - | 6 | .014 | Méthode | ALVES | ROC |
| IO_ROC | 2 | | | ROC | - | - |
| IO_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.58 | 7.913 | .019 | Méthode | ALVES | ROC |
| IVY_ROC | 2.92 | | | ROC | .043 | - |
| IVY_RAW | 1.5 | | | RAW | .915 | .014 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_AL | 2.33 | 6.909 | .032 | Méthode | ALVES | ROC |
| JODA_ROC | 2.5 | | | ROC | .854 | - |
| JODA_RAW | 1.17 | | | RAW | .082 | .021 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.83 | 3.818 | .148 | Méthode | ALVES | ROC |
| MATH_ROC | 2.83 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | 4 | .046 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | - | - |
| LUCENE_RAW | 1 | | | RAW | - | .008 |

TABLE F.38 – Résultats des tests de Friedman et Nemenyi pour Forêt aléatoire avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.4 | 7.6 | .022 | Méthode | ALVES | ROC |
| ANT_ROC | 2.6 | | | ROC | .702 | - |
| ANT_RAW | 1 | | | RAW | .043 | .004 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 5 | .025 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | - | 6 | .014 | Méthode | ALVES | ROC |
| IO_ROC | 2 | | | ROC | - | - |
| IO_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.42 | 8.087 | .018 | Méthode | ALVES | ROC |
| IVY_ROC | 2.92 | | | ROC | .021 | - |
| IVY_RAW | 1.67 | | | RAW | .99 | .03 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| JFC_ROC | 2.5 | | | ROC | .854 | - |
| JFC_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.5 | 6.909 | .032 | Méthode | ALVES | ROC |
| JODA_ROC | 2.33 | | | ROC | 1 | - |
| JODA_RAW | 1.17 | | | RAW | .043 | .043 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.83 | 3.818 | .148 | Méthode | ALVES | ROC |
| MATH_ROC | 2.83 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | .143 | .705 | Méthode | ALVES | ROC |
| LUCENE_ROC | 1.57 | | | ROC | - | - |
| LUCENE_RAW | 1.43 | | | RAW | - | - |

TABLE F.39 – Résultats des tests de Friedman et Nemenyi pour Perceptron multicouche avec BIN-MEAN

| Système | ANT | | | Test de Nemenyi | | |
|---|---|---|---|---|---|---|
| | Rang Moyen | Chi-carré | p | | | |
| ANT_ALVES | 2.5 | 7.895 | .019 | Méthode | ALVES | ROC |
| ANT_ROC | 2.5 | | | ROC | .854 | - |
| ANT_RAW | 1 | | | RAW | .03 | .006 |
| **Système** | **POI** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| POI_ALVES | - | 5 | .025 | Méthode | ALVES | ROC |
| POI_ROC | 2 | | | ROC | - | - |
| POI_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IO** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IO_ALVES | - | 6 | .014 | Méthode | ALVES | ROC |
| IO_ROC | 2 | | | ROC | - | - |
| IO_RAW | 1 | | | RAW | - | .008 |
| **Système** | **IVY** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| IVY_ALVES | 1.58 | 7.913 | .019 | Méthode | ALVES | ROC |
| IVY_ROC | 2.92 | | | ROC | .043 | - |
| IVY_RAW | 1.5 | | | RAW | .915 | .014 |
| **Système** | **JFC** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JFC_ALVES | 2.6 | 8.444 | .015 | Méthode | ALVES | ROC |
| JFC_ROC | 2.4 | | | ROC | .964 | - |
| JFC_RAW | 1 | | | RAW | .021 | .009 |
| **Système** | **JODA** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| JODA_ALVES | 2.33 | 6.909 | .032 | Méthode | ALVES | ROC |
| JODA_ROC | 2.5 | | | ROC | .854 | - |
| JODA_RAW | 1.17 | | | RAW | .082 | .021 |
| **Système** | **MATH** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| MATH_ALVES | 1.83 | 3.818 | .148 | Méthode | ALVES | ROC |
| MATH_ROC | 2.83 | | | ROC | - | - |
| MATH_RAW | 1.33 | | | RAW | - | - |
| **Système** | **LUCENE** | | | **Test de Nemenyi** | | |
| | Rang Moyen | Chi-carré | p | | | |
| LUCENE_ALVES | - | 4 | .046 | Méthode | ALVES | ROC |
| LUCENE_ROC | 2 | | | ROC | - | - |
| LUCENE_RAW | 1 | | | RAW | - | .008 |

TABLE F.40 – Résultats des tests de Friedman et Nemenyi pour K plus proches voisins avec BIN-KM5