ROYAUME DU MAROC
Université Mohammed V - RABAT -
جامعة محمد الخامس – الرباط –
Faculté des sciences
كلية العلوم
CENTRE D'ETUDES DOCTORALES - SCIENCES ET TECHNOLOGIES

N° d'ordre 3256

THESE

En vue de l'obtention du : ***DOCTORAT***

Structure de Recherche : Equipe Intelligent Processing and Security of Systems

Discipline : Informatique

Spécialité : Intelligence Artificielle

Présentée et soutenue le 29/10/2019 par :

REHIOUI Hajar

Conception et Développement de Nouveaux Algorithmes de Machine Learning pour une meilleure Classification des Données

JURY

HAYAR Aawatif	PES, École Nationale Supérieure d'Électricité et de Mécanique, Université Hassan II de Casablanca	Présidente
IDRISSI Abdellah	PH, Faculté des Sciences de Rabat, Université Mohammed V de Rabat	Directeur de thèse
BOULMAKOUL Azedine	PES, Faculté des Sciences et Techniques de Mohammedia, Université Hassan II de Casablanca	Rapporteur/Examinateur
BELOUADHA Fatima-Zahra	PES, Ecole Mohammadia d'Ingénieurs, Université Mohammed V de Rabat	Rapporteur/Examinateur
BENMILOUD Ibtissam	PES, École Nationale Supérieure des Mines de Rabat	Rapporteur/Examinateur

Année Universitaire : 2019-2020

Dédicace

Aux personnes chères à notre Cœur
qui nous ont quitté et qui ont laissé
un très grand vide.

À ma grand-mère bien aimée
À ma mère, mon père
À mon âme-sœur : Ayoub
À ma princesse : Hafssa

À mes beaux parents
À mon beau frère : Anass
À ma belle sœur : Hind
À ma petite amie : Rita

Remerciements

Cette thèse de doctorat a été menée pour l'obtention du grade de Docteur de la Faculté des Sciences à l'Université Mohammed V de Rabat.

Les travaux présentés dans ce mémoire ont été effectués au sein de l'équipe l'équipe Intelligent Processing and Security of Systems (IPSS) de la Faculté des Sciences de Rabat (FSR)-Université Mohammed V au Maroc sous la direction de M. Abdellah IDRISSI.

En premier lieu, je tiens à remercier mon directeur de thèse M. Abdellah IDRISSI, Professeur Habilité à la faculté des Sciences de Rabat. Avec son esprit bien veillant, il a bien guidé mes premiers pas dans le monde de la Recherche. Et je réitère mes remerciement pour sa grande disponibilité, son aide précieuse, ainsi que pour ses efforts prodigués pour l'accomplissement de ce travail de thèse.

Mes sincères remerciements vont également à notre directrice de structure de recherche Mme. Fouzia OMARY, Professeur d'Enseignement Supérieur à la faculté des Sciences de Rabat, de m'avoir accueilli au sein de son équipe et d'avoir été disponible pour résoudre le moindre problème. je suis très honoré d'avoir eu l'occasion de la connaitre.

Je tiens à remercier Mme. Aawatif HAYAR, Professeur d'Enseignement Supérieur à l'École Nationale Supérieure d'Électricité et de Mécanique de Casablanca et Présidente de l'Université Hassan II de Casablanca, d'avoir accepté de présider le jury de ma thèse, malgré ses occupations.

Je remercie M. Azedine BOULMAKOUL -Professeur d'Enseignement Supérieur à la Faculté des Sciences et Techniques de Mohammedia-, d'avoir accepté de juger la qualité de mon travail en tant que rapporteur et examinateur.

Mes remerciements vont également à Mme. Fatima-Zahra BELOUADHA -Professeur d'Enseignement Supérieur à l'École Mohammadia d'Ingénieurs- pour le temps accordé à la lecture de mon rapport afin de l'examiner et le rapporter.

Sans oublier de remercier Mme. Ibtissam BENMILOUD -Professeur d'Enseignement Supérieur à l'École Nationale Supérieure des Mines de Rabat- d'avoir accepté de rapporter et d'examiner ce travail.

Je remercie mes amis qui m'ont apporté leur soutien et leurs encouragements tout au long de cette aventure. Dans le désordre, Famille Terfaoui, Kaoutar, Sara, Safae, Houda, Fatima-Zahra, Nidal, Batoul, Latifa, Meryam, Aicha, Manar, Manal, Nisrine, Hanane.

Je garde le meilleur pour la fin, ma famille qui a supporté toutes les difficultés morales et matérielles pour me soutenir au terme de mes études. J'adresse ma profonde gratitude et mon immense reconnaissance à mes raisons d'être, ma mère et mon père, qui m'ont

éduqué et orienté. Merci de m'avoir encouragé et soutenu dans mes choix. Nul mot et nulles expressions refléteront le grand amour et la profonde gratitude que je porte pour vous. À toute la famille et particulièrement ma cousine Bahia et ma petite sœur de cœur Chaymae. À mes beaux parents qui m'ont soutenu tout au long de mon parcours de thésarde. Au plus beau cadeau qui m'a été offert par Allah, mon chère mari Ayoub. À ma raison de vivre, Hafssa.

Résumé

Actuellement, la science de données est un axe de recherche en plein essor grâce à la grande quantité de données générées quotidiennement par les différents moyens technologiques. Cet axe vise à extraire les informations pertinentes à partir des données brutes. Une description en amont de ces données est souvent indisponible y compris les classes des échantillons. Par conséquent, il est plus judicieux d'adopter des méthodes appropriées, en l'occurrence la classification non supervisée (dite Clustering en anglais) qui consiste à regrouper les données sous forme de classes homogènes appelées Clusters. Dans la présente thèse, nous nous sommes intéressés à l'amélioration de l'algorithme de clustering DENCLUE qui appartient à la famille de méthodes basées sur la densité. Cet algorithme a déjà prouvé sa robustesse surtout dans le cas des données bruitées multi-dimensionnelles. Cependant, il n'est pas assez performant en termes de temps d'exécution en particulier pour classifier une grande quantité de données. Pour remédier à cela, nous avons proposé trois nouvelles améliorations de DENCLUE qui ont montré leur performance à trouver un bon compromis entre le temps d'exécution et la qualité du clustering. En plus des améliorations considérables apportées, notre analyse de résultats nous a conduit à la détection d'un problème de chevauchement entre les clusters obtenus dans certains ensembles de données. Pour répondre à ce problème, nous avons proposé une mise en échelle des données en se basant sur leurs distributions de densités. Les résultats quantitatifs et visuels se sont avérés plus intéressants prouvant ainsi le grand intérêt de la méthode proposée. La deuxième partie de nos contributions s'est focalisée sur l'application de nos algorithmes tout en les adaptant à des domaines bien spécifiques, notamment la recherche et la sélection des services dans le Could Computing, l'analyse de sentiments dans le réseau social Twitter, et le cancer du nasopharynx (domaine médical).

Mots clés : Science de données, Machine Learning, Clustering, Algorithmes basés sur la Densité, Cloud Computing, Analyse de Sentiments.

Abstract

Today, the Data Science is considered as growing area of research due to the large amount of data generated daily by different current technologies. This discipline aims to extract the relevant information from the raw data. A beforehand description of these data is often unavailable, especially the labels of the samples. Therefore, it is wise to adopt appropriate methods, such as the unsupervised classification (also called clustering) which consists of grouping the data in homogeneous clusters. In this thesis, we are interested in improving the DENCLUE clustering algorithm that belongs to the family of density-based methods. This algorithm has already proved its robustness especially in the case of multi-dimensional noisy data. However, it is not efficient enough in terms of execution time especially for the classification of a large amount of data. To remedy this, we proposed three new improvements of DENCLUE that showed their performance to find a good trade-off between the execution time and the quality of the clustering. In addition to the considerable made improvements, our results analysis led us to detect the problem of overlapped clusters obtained in some datasets. To overcome this limit, we scaled the data based on their density distributions. The quantitative and visual results proved the efficiency and the great interest of the proposed method. The second part of our contributions focused on the application of our algorithms while adapting them to specific domains, including the research and selection of services in the Could Computing, the sentiment analysis in the social network Twitter , and the nasopharyngeal cancer (medical field).

Keywords : Data Science, Machine Learning, Clustering, Density-based Algorithms, Cloud Computing, Sentiment Analysis.

Table des matières

Liste des abreviations

BNL *Block-Nested-Loops*
CA *Cluster Accuracy*
CP *Compactness Index*
CSRSA *Cloud Services Research and Selection Agent*
CSRSS *Cloud Service Research and Selection System*
D And C *Divide And Conquer*
DBI *Davies-Bouldin Index*
DBSCAN *Density-Based Spatial Clustering of Application with Noise*
DENCLUE *DENsity-based CLUstEring*
DI *Dunn Index*
EISA *ELECTRE IS Agent*
ELECTRE *ELimination Et Choix Traduisant la Réalité*
EM *Expectation Maximization*
FNA *Fine Needle Aspiration*
FMM *Fuzzy Min-Max*
GA *Genetic Algorithm*
KNN *K-Nearest Neighbor*
NMI *Normalized Mutual Information*
NPC *NasoPharyngeal Carcinoma*
MARSAN *Méthode d'Analyse, de Recherche et de Sélection d'Activités Nouvelles*
MCDA *Multi Criteria Decision Aiding*
MDS *MultiDimensional Scaling*
OPTICS *Ordering Points To Identify the Clustering Structure*
PSPA *Pre-Skyline Processing Agent*
RN *Réseaux de Neurones*
SA *Simulated Annealing*
SDA *Scaled Data Algorithm*

SVM *Support Vector Machines*
UQPA *User's Query Processing Agent*
WDBC *Wisconsin Diagnostic Breast Cancer*
WFCS *Weighting Fuzzy Compactness and Separation*

Liste des figures

Liste des tableaux

Liste des algorithmes

Introduction générale

0.1 Contexte et problématique

Nous assistons aujourd'hui à une ère où la quantité de données générée connait une explosion. Ces données, qui peuvent être textuelles, sous format d'images, vidéos ou autres, sont généralement acquises à l'intermédiaire de smart phones, d'ordinateurs, de réseaux sociaux, etc. Le nombre important et la nature hétérogène de ces données ont suscité la curiosité de plusieurs chercheurs qui tentent de les exploiter pour en extraire l'information utile dans le but de servir divers domaines de la vie, notamment économique, politique ou autres. Ces recherches se regroupent dans le contexte général de la science de données (en anglais : Data Science).
Le Machine Learning est l'une des techniques de la science de données permettant une exploration et une extraction de l'information utile. Cette technique se compose de plusieurs disciplines, parmi elles on trouve la classification.

Le terme classification désigne un phénomène naturel fait instinctivement pour donner un sens d'organisation à notre vie. Le fait de regrouper les livres dans notre bibliothèque en fonction de leurs disciplines, de mettre des étiquettes sur nos sentiments (détecter le sentiment de joie, de tristesse ou autre), de se spécialiser dans la caractérisation des félins, d'oiseaux ou de plantes et les regrouper par espèces. Ce besoin de classifier les objets, les maladies, les thèmes ou les êtres vivants nous a permis une meilleur organisation qui a mené à une simplification de traitement dans plusieurs axes de vie quotidienne et professionnelle.
Les techniques de classification sont à leur tour classifiées et divisées en trois catégories : classification supervisée, semi-supervisée et non supervisée. La première et la deuxième catégorie se basent sur des connaissances préalables, tandis que la troisième catégorie ne se base sur aucune connaissance. C'est pour cette raison que nous avons choisi de travailler avec la classification non-supervisée autrement dit : Clustering.

Comme nous allons voir dans ce présent mémoire, plusieurs méthodes de clustering ont été définies dans la littérature, mais nous nous sommes particulièrement intéressés à DENCLUE, l'une des méthodes de clustering basées sur la densité. Notre choix de s'appuyer sur ce type d'algorithmes se justifie par sa capacité de trouver des clusters de formes

arbitraires et de supprimer les données bruitées afin d'offrir une meilleur qualité de clustering.
Malgré les avantages de DENCLUE, il souffre d'une lenteur d'exécution causée par son utilisation de l'algorithme de recherche local Hill Climbing. Cette algorithme utilisé dans l'étape de recherche des attracteurs de densités, pose aussi un problème de convergence à un optimum local (Hinneburg et Gabriel, 2007), surtout lorsque la quantité des données traitées augmente. La nature des données exploitées par DENCLUE pose aussi un problème lorsque les données sont chevauchées et ont une distribution de densité non unifiée. D'autre part, nous soulignons que l'algorithme DENCLUE est peu exploité dans la littérature, a fortiori dans la résolution de différentes problématiques liées à des applications bien précises. Tout cela nous a mené à bien creuser dans les détails de cet algorithme afin de combler ses lacunes citées ci-dessus.

0.2 Contributions et organisation de la thèse

Les contributions présentées sous cette thèse se divisent en deux volets : algorithmes de clustering généralisés et d'autres dédiés à des applications bien spécifiques. Les deux parties de nos contributions ont pour but de trouver un compromis entre la qualité du clsutering et le temps d'exécution.

Le premier chapitre a été consacré à la formulation de la problématique du clustering et à la définition et l'analyse de ses grandes familles d'algorithmes. Tandis que dans le deuxième chapitre, l'algorithme DENCLUE a été amélioré à trois reprises pour trouver l'approche qui répond le mieux aux besoins de la qualité et du temps d'exécution. La qualité des résultats a été améliorée davantage dans le chapitre 3, en traitant cette fois-ci les données et non les algorithmes. Ce traitement a été plus particulièrement consacré aux ensembles de données atteints du problème de chevauchement.

Les améliorations d'algorithmes et de données discutées dans la première partie ont été adaptées, puis exploitées dans la seconde partie de cette thèse. Pour ce faire, nous nous sommes basés dans le chapitre 4 sur DENCLUE-IM, une des améliorations proposées qui a donné ses fruits, en l'intégrant dans un système de recherche et de sélection des services Cloud. Dans le chapitre 5, des approches hybrides à bases de l'algorithme DENCLUE et K-means ont été crées et exploitées dans l'un des domaines d'actualités à savoir l'analyse des sentiment dans les réseaux sociaux, en accordant une attention particulière au réseau Twitter. Le chapitre 6, quant à lui touche au domaine humain en exploitant les algorithmes de clustering dans la prévention du cancer du nasopharynx.

0.3 Production scientifique

Revues internationales (3)

1. **Rehioui H.**, Idrissi A., Koukam A., Gihbid A., Tawfiq N., Khyatti M., "On the Use

of Clustering Algorithms in Medical Domain", International Journal of Artificial Intelligence (IJAI), Vol. 17, Issue 2, pp. 236-247.

2. **Rehioui H.**, Idrissi A., "New Clustering Algorithms for Twitter Sentiment Analysis", IEEE Systems Journal [IF : 4.463], doi : 10.1109/JSYST.2019.2912759
3. **Rehioui H.**, Idrissi A., "A fast clustering approach for large multidimensional data", International Journal of Business Intelligence and Data Mining, Vol. 15, Issue 3, pp. 349-369, doi : 10.1504/IJBIDM.2019.101946

Article soumis

— **Rehioui H.**, Idrissi A., "Scaling Approach for Discovering Overlapped Clusters", Soumis à IEEE Transactions on Knowledge and Data Engineering [IF : 3.857], (Under review).

Conférences internationales (7)

4. **Rehioui H.**, Idrissi A., Abourezq M., "The Research and Selection of Ideal Cloud Services using Clustering Techniques", Proceedings of the International Conference on Big Data and Advanced Wireless Technologies (BDAW'16), Blagoevgrad, Bulgaria, November 10 - 11, 2016, ACM.
5. Elhandri K., Idrissi A., **Rehioui H.** et Abourezq M., "Top-k and Skyline for Cloud Services Research and Selection System", Proceedings of the International Conference on Big Data and Advanced Wireless Technologies (BDAW'16), Blagoevgrad, Bulgaria, November 10 - 11, 2016, ACM.
6. Zegrari F., Idrissi A. et **Rehioui H.**, "Resource allocation with efficient load balancing in cloud environment", Proceedings of the International Conference on Big Data and Advanced Wireless Technologies (BDAW'16), Blagoevgrad, Bulgaria, November 10 - 11, 2016, ACM.
7. **Rehioui H.**, Idrissi A., Abourezq M. et Zegrari F., "DENCLUE-IM : A new approach for big data clustering", Procedia Computer Science, Vol. 83, p. 560-567, 2016.
8. Idrissi A.,**Rehioui H.**, Laghrissi A. et Retal S., "An improved denclue algorithm for data clustering", IEEE 2015 International Conference on Information and Communication Technology and Accessibility (ICTA'15), Marrakech, 21-23 December 2015.
9. Idrissi A., Retal S., **Rehioui H.** et Laghrissi A., "Gateway selection in vehicular ad-hoc network", IEEE 2015 International Conference on Information and Communication Technology and Accessibility (ICTA'15), Marrakech, 21-23 December 2015.
10. Idrissi A., Laghrissi A., Retal S., et **Rehioui H.**, "VANET congestion control approach using empathy", IEEE 2015 International Conference on Information

and Communication Technology and Accessibility (ICTA'15), Marrakech, 21-23 December 2015.

Chapitre

1 Clustering de données : généralités et état de l'art

1.1 Introduction

Dans cette dernière décennie, la quantité des informations a connu une évolution exponentielle à cause des nouvelles découvertes technologiques. La dispersion de ces informations nécessite un traitement spécial afin de mieux organiser, utiliser et exploiter l'information d'une bonne manière. Cela est effectué en faisant recours aux méthodes du Machine Learning qui ont pour objectif d'extraire des connaissances à partir des données brutes. Dans cette optique, l'une des techniques utilisées dans le Machine Learning est la classification. Cette dernière est utilisée pour regrouper les données en des classes homogènes, qui peuvent servir au traitement d'images, aidant ainsi au développement du domaine de l'intelligence artificielle afin de détecter les problème cardiovasculaire (Dey *et al.*, 2019), de mieux explorer les régions du cerveau (Huang *et al.*, 2019), ou pour regrouper les différentes empreintes de la main (Khan et Bianchi, 2019). La classification est aussi utilisée dans le domaine d'analyse des sentiments (Wang *et al.*, 2018; AL-Sharuee *et al.*, 2018) ou pour des fins de compression de données (Yang *et al.*, 2008) (qui travaillent avec les groupes plutôt que des éléments individuels), à l'identification des caractéristiques des sous-populations qui peuvent être ciblées à des fins spécifiques (par exemple, le marketing (Punj et Stewart, 1983; Hernandez *et al.*, 2019) visant une tranche spécifique de société), etc.
Dans ce chapitre, nous allons présenter généralement la notion de classification, en détaillant de plus en plus son aspect non supervisé, aussi appelé clustering. Un état de l'art sera aussi présenté en s'intéressant surtout aux algorithmes de clustering utilisés à des fins comparatives dans cette thèse.

1.2 Types de classifications

La classification est l'une des puissants outils d'exploration de données qui a pour objectif d'effectuer des regroupements d'un ensemble d'individus en classes homogènes. Ces regroupement sont effectués en augmentant la similarité entre les individus au sein d'une même classe (intra-classe), et en la diminuant entre les classes différentes (inter-classes). Dans la classification on distingue entre trois grandes familles : classification supervisée,

classification semi-supervisée et classification non supervisée aussi appelée clustering. Ce dernier terme sera adopté tout au long de ce mémoire.
Chacune de ces catégories de classification possède ses propres caractéristiques. La classification supervisée est un mode de classification qui nécessite une certaine connaissance des données. En fait, la classification supervisée se base sur deux étapes phares à savoir une phase d'apprentissage qui sert à bâtir le modèle d'apprentissage et une étape de validation dite aussi de test qui sert à évaluer le modèle puis le valider. Pour partitionner les données en deux segments, un d'apprentissage et un autre de validation, on fait recours en principe à la validation croisée (Kotsiantis *et al.*, 2007). En général, le segment d'apprentissage est plus grand que celui de validation.
Plusieurs méthodes de classification supervisée existent dans la littérature, parmi eux on trouve le K-plus proche voisin (KNN, K-Nearest Neighbor), RN (Réseaux de neurones), machine à vecteurs de support (SVM, Support Vector Machine), etc.
Malgré le grand succès des méthodes supervisées, le processus d'étiquetage, fait à la main, des données utilisées dans la phase d'apprentissage nécessite beaucoup de temps et d'efforts.

Le processus de la classification semi-supervisée, nécessite également des connaissances prédéfinies sous forme d'étiquettes (labels) ou de contraintes par paire. Ce type de classification a été exploité dans plusieurs domaines. Dans le domaine du text mining, Xing *et al.* (2010) ont proposé une méthode de classification semi-supervisée pour l'étiquetage des documents en se basant sur l'algorithme Naïve Bayes et l'algorithme Expectation-Maximization (EM). Une autre méthode a été proposée dans le domaine biologique afin de classifier les protéines d'une manière semi-supervisée (Weston *et al.*, 2004). Les méthodes de classification semi-supervisée ont été introduites aussi dans le domaine d'analyse des sentiments (Charalampakis *et al.*, 2016).
La difficulté de trouver des informations prédéfinies est également posée dans le cas de la classification semi-supervisée. Le processus de ce type de classification nécessite également des connaissances sous forme de labels ou de contraintes par paire, qui peuvent être difficiles à extraire dans certains ensemble de données.

À l'encontre des deux types de classification présentés ci-dessus, la classification non supervisée (clustering) a pour objectif de chercher les classes (appelée clusters) des observations sans aucune information sur la nature des données ni sur les clusters traités. Par contre le nombre de classe peut être déterminé à l'avance dans certains algorithmes (comme dans l'algorithme k-means).
Nos travaux se sont focalisés sur les algorithmes de clustering. La motivation majeure derrière ce choix réside dans la capacité de ce type d'algorithme de découvrir les clusters sans aucune information prédéfinie, tout en économisant le grand temps d'apprentissage nécessité par les algorithmes supervisés.

1.3 Formulation du problème

Le clustering permet de créer des groupes d'objets (clusters) de manière à ce que les objets présents dans chaque cluster soient distincts entre eux (clustering dur). Pour ce faire, les méthodes de clustering se basent sur différentes représentations de données en fonction de leur type (Gan *et al.*, 2007) : numérique, binaire, ordinal, etc.
Les données manipulées dans ce travail sont de type numérique, de ce fait la représentation la plus appropriée est celle dite description vectorielle des données (Gan *et al.*, 2007). Dans ce type de données, on doit classifier N objets (appelés aussi points ou observations) $X = \{x_1, x_2, ..., x_N\}$. Chaque objet x_i, avec $i \in \{1, 2, ..., N\}$, est représenté sous format vectorielle $x_i = (x_{i1}, x_{i2}, ..., x_{id})$ de d-dimensions, où chaque x_{ij} est un attribut caractéristique de l'objet x_i, avec $j \in \{1, 2, ..., d\}$. En général, un ensemble de données de ce type est représenté par la matrice D suivante :

$$D = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1d} \\ x_{21} & x_{22} & \dots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \dots & x_{Nd} \end{pmatrix} \tag{1.1}$$

Il faut noter que certaines données utilisées dans les expérimentations possèdent une autre colonne de plus désignant les étiquettes des objets, sans pour autant qu'elles ne soient utilisées dans le processus du clustering. Ces étiquettes n'ont pour but que de servir à des fins comparatives avec la vérité terrain (les étiquettes réelles).

1.4 Distances

Les distances et les similitudes jouent un rôle important dans l'analyse des clusters (Jain et Dubes, 1988, Anderberg, 1973). Dans la littérature, les mesures de similarité, les coefficients de similarité, les mesures de dissimilarité ou les distances sont utilisées pour décrire quantitativement la similarité ou la dissemblance de deux points de données ou de deux groupes.
Du fait que la nature des données traitées dans la présente thèse est numérique, nous avons opté pour l'utilisation des distances destinées à ce type de données. Nous présentons ainsi quelques unes qui sont utilisées fréquemment dans plusieurs travaux.

1.4.1 Distance Euclidienne

La distance euclidienne entre deux points de données x_i et $x_{i'}$ dans un espace de d-dimensions, est définie par l'équation suivante :

$$dist_{euc}(x_i, x_{i'}) = \sqrt{\sum_{j=1}^{d} (x_{i'j} - x_{ij})^2} \tag{1.2}$$

sachant que $i, i' \in \{1, ..., N\}$, et $j \in \{1, ..., d\}$.

1.4.2 Distance de Manahattan

La distance de Manhattan, aussi appelée "distance de ville", est définie comme étant la somme des distances de tous les attributs. C'est-à-dire que, pour deux points x_i et $x_{i'}$ dans un espace de d-dimensions, leur distance de Manhattan est définie comme suit :

$$dist_{man}(x_i, x_{i'}) = \sum_{j=1}^{d} |x_{i'j} - x_{ij}| \tag{1.3}$$

sachant que $i, i' \in \{1, ..., N\}$, et $j \in \{1, ..., d\}$.

1.4.3 Distance de Minkowski

La distance de Minkowski est considérée comme une généralisation des distances euclidienne et Manhattan. Elle est définie comme suit :

$$dist_{min}(x_i, x_{i'}) = \sqrt[r]{\sum_{j=1}^{d} |x_{i'j} - x_{ij}|^r}, \ r \geq 1 \tag{1.4}$$

r est appelé l'ordre de la distance de Minkowski. À noter : si $r = 2$ ou $r = 1$, nous obtenons la distance euclidienne ou celle de Manhattan, respectivement.

1.4.4 Clustering dur et clustering flou

En principe on distingue entre deux types de clustering : le clustering dur (appelé en anglais hard clustering) et le clustering flou (appelé en anglais fuzzy clustering). Dans les algorithmes de clustering dur, une étiquette de classe (label) l_{ii} avec $ii \in \{1, 2, ..., k\}$ est attribuée à chaque objet x_i pour identifier son cluster, où k est le nombre de clusters. En d'autres termes, dans un clustering dur, chaque objet est supposé appartenir à un et un seul cluster. Ce type de clustering implique les contraintes suivantes :

1. Chaque objet peut appartenir au plus à un seul cluster (0 ou un cluster).
2. Chaque cluster contient au moins un seul objet, (aucun cluster vide n'est autorisé).

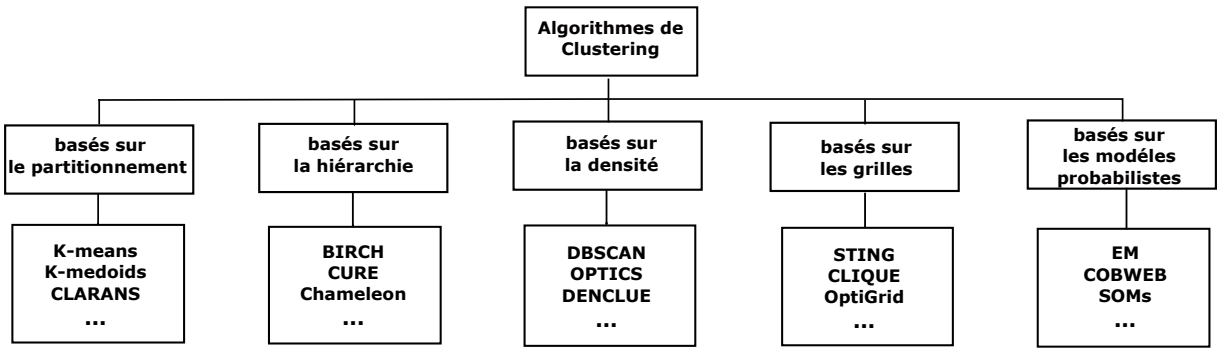
Dans le clustering flou, l'hypothèse devient plus souple de sorte que : un objet x_i peut appartenir à un ou plusieurs clusters sous certaines probabilités.

1.5 Taxonomie

Dans la littérature plusieurs types d'algorithmes de clustering ont été proposés. Selon certains critères (Berkhin, 2006; Xu *et al.*, 2005), ces algorithmes sont généralement

groupés en cinq grandes familles (Shah *et al.*, 2012; Fahad *et al.*, 2014), à savoir, les algorithmes de clustering basés sur le partitionnement, la hiérarchie, la densité, les grilles et les modèles probabilistes comme illustré sur la figure 1.1.

FIGURE 1.1 – La taxonomie des algorithmes de clustering (Fahad *et al.*, 2014)



1.5.1 Méthodes basées sur le partitionnement

Ce type de clustering a pour fonction de diviser les données en plusieurs clusters. Il est considéré comme l'un des plus simples types de clustering. Pour ce faire, des petits clusters initiaux sont formés et assemblés afin de trouver les clusters finaux.

1.5.2 Méthodes basées sur la hiérarchie

Dans le clustering hiérarchique, les objets sont regroupés sous format d'un arbre de clusters. Les algorithmes de ce type sont divisés en deux catégories, divisibles (de haut en bas) et agglomératifs (de bas en haut) (Ding et He, 2002). Les algorithmes divisibles mettent toutes les données dans un cluster, puis les divisent hiérarchiquement jusqu'à former les clusters finaux. Quant aux algorithmes agglomératifs, ils placent chaque objet de la base de données dans un cluster, par la suite, ils fusionnent récursivement ces clusters jusqu'à formation des résultats finaux.

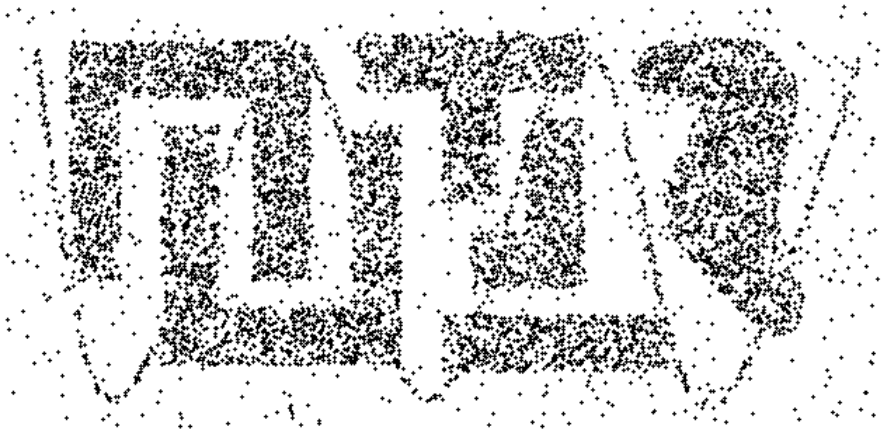
1.5.3 Méthodes basées sur la densité

Les algorithmes de clustering basés sur la densité classifient les objets en fonction de leurs régions de densité. Ce type d'algorithmes permet de découvrir des clusters de formes arbitraires (Karypis *et al.*, 1999) comme illustré sur la figure 1.2, et permet aussi d'omettre les objets bruyants.

1.5.4 Méthodes basées sur les grilles

Les algorithmes basés sur les grilles structurent les données dans une grille. Dans ce type, l'algorithme est appliqué sur la grille au lieu d'être appliqué directement sur la base de données.

FIGURE 1.2 – Données de formes arbitraires



1.5.5 Méthodes basées sur les modèles probabilistes

Le clustering basé sur les modèles probabilistes est fondé sur l'hypothèse selon laquelle les données sont générées par des distributions de probabilité. Ces méthodes visent à proposer une hypothèse de modèle pour chaque cluster, puis à trouver le meilleur ajustement des données au modèle.

1.6 Méthodes de clustering : état de l'art

La technique de clustering est riche par ses différentes méthodes citées dans la littérature. L'algorithme K-means (MacQueen *et al.*, 1967) reste l'un des algorithmes les plus utilisés jusqu'à nos jours (Jain, 2010). Son utilité est démontrée dans différents domaines, à l'instar du domaine médicale (Khanmohammadi *et al.*, 2017), du traitement d'images (Marwan *et al.*, 2018), du Cloud Computing (Jagli *et al.*, 2016), des réseaux sociaux (Yang *et al.*, 2016), etc.

Dans la famille du clustering basée sur les modèles probabilistes, l'algorithme Expectation Maximization (EM) (Dempster *et al.*, 1977) est l'un des algorithmes les plus connus au sein de cette famille. Son utilisation est répandue dans différentes applications, comme le domaine biomédicale (Tian *et al.*, 2018), l'analyse des sentiments (Shelke *et al.*, 2017), etc.

L'algorithme DBSCAN (pour le terme anglais : Density-Based Spatial Clustering of Applications with Noise) (Ester *et al.*, 1996), est connu pour sa qualité de classification des données spatiales (Wang *et al.*, 2015). En dépit de son application aux ensembles de données spatiales, DBSCAN est aussi utilisé dans d'autres domaines, comme : la segmentation d'images (Lee, 2015), l'analyse de sentiments (Hridoy *et al.*, 2015), et bien d'autres.

L'algorithme DENCLUE (Hinneburg et Keim, 1998) et son amélioration DENCLUE 2.0 (Hinneburg et Gabriel, 2007) sont considérés comme des algorithmes appartenant à la fois à la famille de clustering basée sur la densité et à celle basée sur les grilles. Ces deux algorithmes héritent des forces des deux familles. D'autres améliorations récentes ont été proposées dans ce sujet comme la proposition de l'algorithme DEBC-GM (Ramesh et Kumari, 2018) dédié au Big Data. Malgré l'avantage de DENCLUE est ses variantes, traités dans la littérature, ils restent mal exploités dans les domaines applicatifs. Il faut noter que tout au long de ce présent document, DENCLUE 1.0 fera référence à DENCLUE.

De ces faits, nous avons décidé dans cette présente thèse d'adopter ces algorithmes dans nos études comparatives. Plus de détails concernant ces algorithmes seront fournis dans ce qui suit.

1.6.1 K-means

L'algorithme K-means (MacQueen *et al.*, 1967) est l'un des algorithmes de partitionnent les plus utilisés. La force de K-means réside dans sa simplicité et sa capacité à classifier de grands ensembles de données. Cet algorithme se base sur la notion de distance entre objets en initialisant chacun des K-clusters par un point aléatoirement choisi de l'ensemble de données. Chaque cluster est représenté par un centroïde. Afin de remplir les clusters, tout point $x_i \in D$ est assigné au plus proche centroïde c_{K_k} appartenant au $cluster_k$ comme suit :

$$x_i \in cluster_k \iff dist(x_i, c_{K_j}) = arg\ min_{1 \leq ii \leq K}\ dist(x_i, c_{K_{ii}}) \tag{1.5}$$

En ajoutant à chaque itération les points, les centroïdes sont recalculés et mis à jour au fur et à mesure en prenant en compte chaque nouveau point ajouté.

1.6.2 EM : Expectation Maximization

L'algorithme EM (Dempster *et al.*, 1977) est un algorithme de clustering basé sur les modèles probabilistes. Il vise à trouver un maximum local pour l'estimation des paramètres du modèle. Cet algorithme est conçu pour estimer les paramètres de vraisemblance maximale d'un modèle statistique, en particulier dans le cas où les équations ne peuvent pas être résolues directement. L'algorithme EM se compose de deux étapes principales : une étape d'espérance, et une autre de maximisation.
Dans l'étape d'espérance (étape E), les objets sont affectés à chaque cluster en fonction de sa distribution postérieure, qui est évaluée à l'aide des valeurs de paramètre du modèle en cours comme le montre l'équation suivante :

$$Q(\theta, \theta^t) = E[log(p(x_i, x_{i'})|\theta)x_i, \theta^t] \tag{1.6}$$

θ est considéré comme le nouveau paramètre, θ^t désigne l'ancien paramètre ou celui en cours, E est le calcul de l'espérance et p représente la probabilité de deux points sachant

le paramètre θ.
Dans l'étape de maximisation (étape M), l'attribution des clusters est obtenue en estimant les paramètres du modèle avec les règles de vraisemblances maximales.

$$\theta^{t+1} = arg\ max_\theta\ Q(\theta, \theta^t) \tag{1.7}$$

En principe le but de l'algorithme EM est de trouver un paramétre θ tel que $p(x|\theta) > p(x|\theta^t)$

1.6.3 DBSCAN

DBSCAN (Ester *et al.*, 1996) est un algorithme de clustering basé sur la densité. Les paramètres d'entrée requis pour cet algorithme est le rayon maximum ϵ situant le voisinage et le nombre minimum de points ξ devant être présents dans un voisinage déterminé par ϵ. Pour construire les clusters, cet algorithme se base sur la notion du Eps-voisinage (N_ϵ) d'un objet x_i dans un ensemble de données D, définit comme suit :

$$N_\epsilon(x_i) = \{\forall x_{i'} \in D \mid x_i \neq x_{i'} \wedge dist(x_i, x_{i'}) \leq \epsilon\} \tag{1.8}$$

Le noyau est aussi l'un des principaux éléments. Il est considéré comme un objet ayant au moins un nombre de voisins ξ dans son N_ϵ.
En se basant sur la notion du noyau, on peut dire qu'un objet x_i est directement accessible par densité à partir d'un objet $x_{i'}$ si $x_i \in N_\epsilon(x_{i'})$ et $x_{i'}$ est un noyau.
Un objet x_i est accessible par densité à partir d'un objet $x_{i'}$ s'il existe un chemin d'objets $x_1, ..., x_n$ où $x_1 = x_{i'}$ et $x_n = x_i$, tel que x_{i+1} est directement accessible à partir de x_i.
On dit aussi q'un objet x_i est connecté par densité à un objet $x_{i'}$ s'il existe un objet o tel que x_i et $x_{i'}$ sont accessibles par densité à partir de l'objet o.
Ainsi, le cluster est considéré comme un ensemble maximal d'objets connectés par densité, et le bruit est considéré comme l'ensemble d'objets qui ne figurent dans aucun cluster.

1.6.4 DENCLUE 1.0

DENCLUE 1.0 (DENsity-based CLUstEring) est présenté dans (Hinneburg et Keim, 1998) comme un cas particulier de l'estimation de la densité par noyau (KDE pour le terme anglais Kernel Density Estimation) (Parzen, 1962; Rosenblatt *et al.*, 1956; Zaki et Meira Jr, 2014). La KDE est une technique d'estimation non paramétrique visant à trouver des régions denses de points.
A partir d'une variable donnée x_i ayant une distribution discrète, la fonction de densité associée $f(x_i)$ est définie dans l'équation (1.9).

$$\hat{f}(x_i) = \frac{1}{Nh} \sum_{t=1}^{N} Ker\left(\frac{x_i - x_t}{h}\right), \tag{1.9}$$

où Ker est le noyau, N est le cardinal de la base de données et h est un paramètre nommé fenêtre ayant comme centre de sa largeur le point x_i. La fenêtre h contrôle le degré d'homogénéité de l'estimation, aussi appelé paramètre de lissage. $Ker(z)$ est la densité de la fonction gaussienne standard définie dans l'équation (1.10).

$$Ker(z) = \frac{z^2}{2} \exp\left\{\frac{-z^2}{2}\right\}, \tag{1.10}$$

où $z = \dfrac{x_i - x_t}{h}$.
Dans ce contexte, le concept d'hypercube est introduit, permettant ainsi d'estimer la densité d'un point x_i de d-dimensions, tel que $x_i = (x_{i1}, x_{i2}, ..., x_{id})$. Cet hypercube est considéré comme une fenêtre d-dimensionnelle de taille h.
Le volume de l'hypercube est défini par l'équation (1.11).

$$vol(H_d(h)) = h^d, \tag{1.11}$$

où H_d est l'hypercube de d-dimensions. La densité est donc estimée comme suit :

$$\hat{f}(x_i) = \frac{1}{Nh^d} \sum_{t=1}^{N} Ker\left(\frac{x_i - x_t}{h}\right) \tag{1.12}$$

Et le noyau gaussien est défini comme dans l'équation (1.13).

$$Ker(z) = \frac{1}{(2\pi)^{\frac{d}{2}}} \exp\left\{\frac{-z^T z}{2}\right\}, \tag{1.13}$$

où $z = \dfrac{x_i - x_t}{h}$.

Les auteurs de DENCLUE 1.0 ont développé cet algorithme pour classer de grandes bases de données multimédias, sachant que ce type de bases de données contient beaucoup de bruit.

DENCLUE 1.0 fonctionne essentiellement en se basant sur deux étapes, l'étape du pré-clustering et l'étape du clustering comme illustré dans la figure 1.3. La première étape consiste à structurer les données dans un hyper-rectangle. Cette structure est utilisée pour accélérer les calculs de la fonction de densité. Quant à la seconde étape, elle identifie les clusters à partir de cubes très peuplés (les cubes dont le nombre de points dépasse un seuil ξ déterminé en paramètres), et leurs cubes peuplés voisins.
DENCLUE 1.0 est basé notamment sur le calcul de l'influence des points entre eux. La somme totale de ces fonctions d'influence représente la fonction de densité. Il existe de nombreuses fonctions d'influence, basées sur la distance entre deux points x_i et $x_{i'}$ avec $i, i' \in \{1, ..., N\}$ et $i \neq i'$. Nous signalons que dans ce travail, nous allons se baser sur la fonction gaussienne.

L'équation (1.14) montre la fonction d'influence entre deux points x_i et $x_{i'}$.

$$f_{Gauss}(x_i, x_{i'}) = \exp^{-\frac{dist(x_i, x_{i'})^2}{2\sigma^2}}, \tag{1.14}$$

où $dist(x_i, x_{i'})$ est la distance euclidienne entre x_i et $x_{i'}$, et σ représente le rayon du voisinage contenant x_i.
L'équation (1.15) représente la fonction de densité.

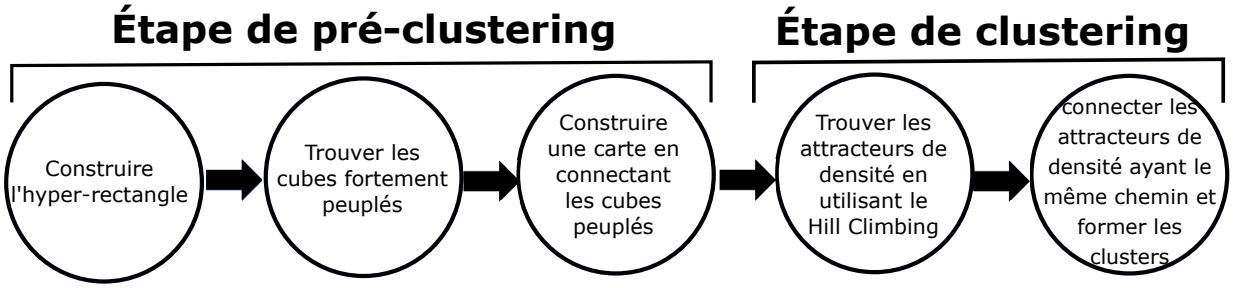
$$f_D(x_i) = \sum_{i'=1}^{N} f_{Gauss}(x_i, x_{i'}), \tag{1.15}$$

où D est l'ensemble des points de la base de données, et N son cardinal.
Afin de déterminer les clusters, DENCLUE 1.0 calcule un attracteur de densité pour chaque point de la base de données. Cet attracteur est considéré comme un maximum local de la fonction de densité. Ce maximum est identifié par l'algorithme Hill Climbing, basé sur l'approche ascendante du gradient (Zaki et Meira Jr, 2014) présentée dans l'équation (1.16).

$$x_i = x_i^0,\ x_i^{l'+1} = x_i^{l'} + \delta \frac{\nabla f_{Gauss}^D(x_i^{l'})}{\|\nabla f_{Gauss}^D(x_i^{l'})\|} \tag{1.16}$$

Le calcul se termine lorsque $f^D(x_i^{l'}) > f^D(x_i^{l'+1})$ avec $l' \in \mathbb{N}$, alors nous prenons $x_i^* = x_i^{l'}$ en tant qu'attracteur de densité.
Les points qui forment un chemin avec un attracteur de densité, sont appelés points attirés. Les clusters finaux sont trouvés en prenant en compte les attracteurs de densité et les points attirés.
L'efficacité de cet algorithme réside dans le choix de la structure avec laquelle les données sont structurées : c'est la notion d'hyper-rectangle. Un hyper-rectangle est constitué de nombreux hypercubes. Chaque hypercube est représenté par la dimension du vecteur caractéristique (c'est-à-dire le nombre de critères) et par une clé. Cette structure ne considère que les cubes peuplés, ce qui permet à DENCLUE 1.0 de gérer les données avec facilité.
L'hyper-rectangle est construit comme suit :
Étape 1. Répartir les données dans un hyper-rectangle dont chaque côté est de 2 σ en ne considérant que des cubes peuplés.
Étape 2. Calculer la moyenne dans chaque cube peuplé.
Étape 3. Repérer les cubes fortement peuplés dont le nombre de points dépasse un seuil ξ.
Étape 4. Déterminer la connexion entre chaque cube fortement peuplé et les autres cubes (cubes fortement ou simplement peuplés) selon la distance entre leurs moyennes. Si $dist(mean(c_1), mean(c_2)) < 4\sigma$, alors les deux cubes sont connectés.
Étape 5. Seuls les cubes fortement peuplés et ceux simplement peuplés qui sont connectés à un cube fortement peuplé sont pris en compte pour déterminer les clusters.

FIGURE 1.3 – Le processus de l'algorithme DENCLUE 1.0



1.6.5 DENCLUE 2.0

Après avoir présenté la première version de l'algorithme DENCLUE, les auteurs Hinneburg et Gabriel (2007) ont été motivés pour introduire une autre version appelée DENCLUE 2.0. Cette seconde version utilise une nouvelle méthode basée sur le Hill Climbing. Cette méthode, destinée aux noyaux gaussiens, a pour but d'éviter les étapes supplémentaires causées par la recherche du maximum locale. L'amélioration proposée, permet à l'algorithme de converger exactement vers un maximum local en le réduisant à un cas particulier de l'algorithme EM.
Dans l'algorithme DENCLUE 2.0, cette réduction est illustrée par la modification apportée dans l'approche ascendante du gradient (équation 1.11) utilisée par la méthode classique du Hill Climbing. La nouvelle formule est représentée dans l'équation (1.17).

$$x_i = \frac{\sum_{i\prime=1}^{N} Ker\left(\frac{x_i - x_{i\prime}}{\sigma}\right) x_{i\prime}}{\sum_{i\prime=1}^{N} Ker\left(\frac{x_i - x_{i\prime}}{\sigma}\right)}, \qquad x_i^{(l\prime+1)} = \frac{\sum_{i\prime=1}^{N} Ker\left(\frac{x_i^{(l\prime)} - x_{i\prime}}{\sigma}\right) x_{i\prime}}{\sum_{i\prime=1}^{N} Ker\left(\frac{x_i^{(l\prime)} - x_{i\prime}}{\sigma}\right)}, \tag{1.17}$$

sachant que Ker est le noyau gaussien définit comme suit :

$$Ker(x_i) = (2\pi)^{-\frac{d}{2}} . \exp\left[-\frac{x_i^2}{2}\right], \tag{1.18}$$

où d est la dimension de données. La formule (1.17) est réduite en deux étapes de l'algorithme EM comme indiqué dans les équations (1.19) et (1.20).
Étape E :

$$\theta_i = \frac{1/N.Ker\left(\frac{x_i^{(l\prime)} - x_{i\prime}}{\sigma}\right)}{f_D\left(x_i^{(l\prime)}\right)} \tag{1.19}$$

Étape M :

$$x_i^{(l\prime+1)} = \frac{\sum_{i\prime=1}^{N} \theta_{i\prime} x_{i\prime}}{\sum_{i\prime=1}^{N} \theta_{i\prime}} \tag{1.20}$$

1.7 Mesures d'évaluation

Pour mesurer la qualité des méthodes de clustering, plusieurs mesures d'évaluation ont été mentionnées dans la littérature (Fahad *et al.*, 2014; Cai *et al.*, 2013; Liu *et al.*, 2013). Elles sont divisées en deux catégories : externes et internes.

1.7.1 Mesures internes

Les mesures d'évaluation internes sont calculées sans aucune information a priori sur les clusters. Principalement, les informations externes telles que les étiquettes ne sont pas souvent disponibles dans les ensembles de données. Par conséquent, les mesures d'évaluation internes sont la seule option permettant d'évaluer la qualité des clusters lorsqu'aucune information externe n'est fournie. Ces mesures comprennent l'indice de Dunn (DI) (Dunn, 1974), l'indice de Davies-Bouldin (DBI) (Davies et Bouldin, 1979) et l'indice de compacité (CP) (Fahad *et al.*, 2014).

— Indice de Dunn (DI) : Cet indice évalue le degré de séparation entre les individus d'un même cluster, c'est-à-dire la similarité intra-cluster. Une valeur élevée indique un meilleur clustering.
— Indice de Davies-Bouldin (DBI) : Cet indice, similaire à DI, évalue également le degré de séparation, mais cette fois ci entre les clusters (dissimilarité inter-clusters). La plus petite valeur indique le meilleur clustering.
— Indice de Compacité (CP) : CP mesure la distance moyenne entre chaque paire dans le cluster, puis entre tous les clusters ; les membres de chaque cluster devraient être aussi proches les uns des autres que possible. La valeur la plus faible indique la meilleure qualité de clustering.

1.7.2 Mesures externes

Les mesures externes tirent leur appellation des informations externes utilisées par les calculs mais qui sont non incluses dans le processus de formation des clusters. Ce type de mesures est basé sur les données étiquetées. Parmi ces mesures, on trouve la précision (noté CA pour le terme anglais Cluster Accuracy) (Fahad *et al.*, 2014), l'indice d'informations mutuelles normalisées (noté NMI pour le terme anglais Normalized Mutual Information) (Fahad *et al.*, 2014) et l'entropie (Rendón *et al.*, 2011). Il faut noter que ce type d'indice ne peut pas être appliqué sur les ensembles de données non étiquetés.

— Précision (CA) : CA mesure le pourcentage d'objets correctement classifiés dans un cluster, en se basant sur des étiquettes prédéfinies. La valeur élevée indique la meilleure qualité de clustering.
— Indice d'informations mutuelles normalisées (NMI) : cet indice permet de mesurer les informations statistiques partagées par les points représentant les affectations des clusters et les affectations d'étiquettes prédéfinies des instances. La valeur de NMI varie entre 0 et 1. La valeur la plus élevée indique la meilleure qualité.

— Entropie : C'est le degré auquel chaque cluster est constitué d'objets appartenant à une seule classe (objets étiquetés par la même étiquette). La plus petite valeur désigne la meilleure performance.

1.8 Conclusion

Après avoir présenté les différents types de classification, et distingué entre la classification supervisée, semi supervisée et non supervisée, nous nous sommes basés sur le volet non supervisé, aussi appelé clustering. La formulation mathématique de la problématique du clustering a été ainsi proposée, tout en distinguant entre les deux types de clustering : dur et flou et en abordant les notions de distances et de similarités considérées comme notions de bases des techniques de clustering. Un état de l'art a été aussi présenté en se focalisant sur les algorithmes de clustering utilisés dans les études comparatives menées dans ce mémoire. Pour clôturer le chapitre, nous avons présenté les mesures d'évaluation servant à juger la qualité du clustering, et surtout permettant d'évaluer les performances de nos approches présentées dans les chapitres qui suivent.

Première partie

Améliorations générales

Chapitre

2 Contributions à l'amélioration de l'algorithme DENCLUE

Sommaire

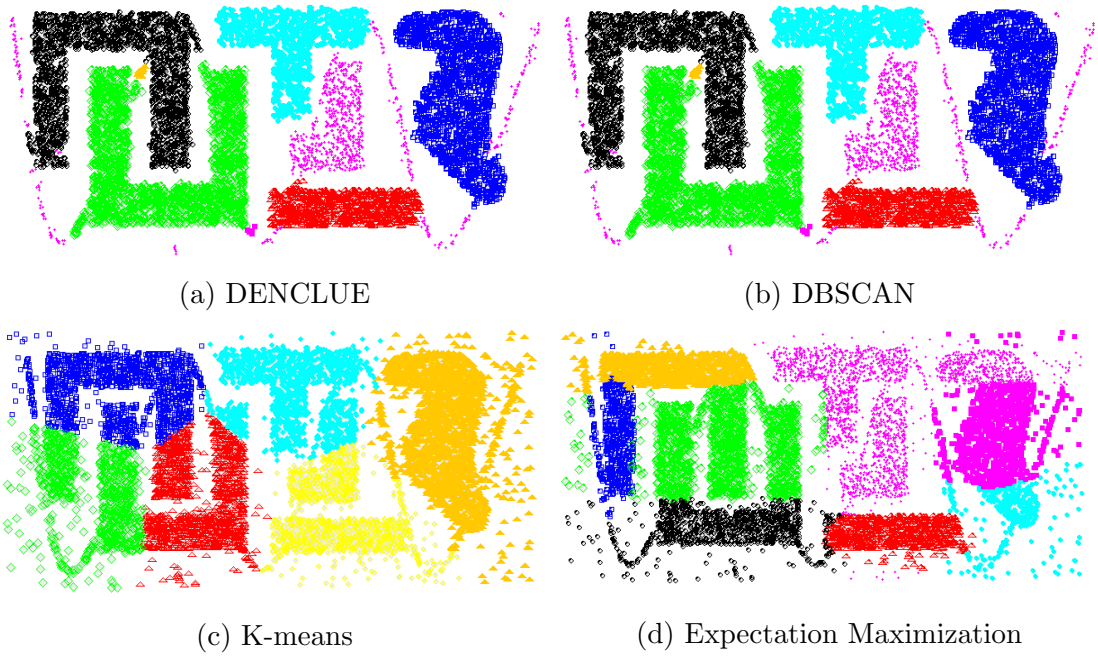
2.1 Introduction

Les algorithmes de clustering sont considérés comme l'une des techniques aidant dans le processus d'extraction des informations utiles.
Cette démarche nous a motivé à chercher et analyser les algorithmes de clustering, consistant à mener une classification non supervisée. Comme précisé auparavant, nous nous sommes focalisés sur la méthode DENCLUE 1.0 (cf. la sous-section 1.6.4 du chapitre 1) qui appartient à la famille des méthodes basées sur la densité. Les résultats prometteurs démontrés par cet algorithme (Fahad *et al.*, 2014; Hinneburg et Keim, 1998), nous ont poussé à s'en servir comme base pour l'implémentation de nos approches qui contribuent à son amélioration.

Pour mieux comprendre les approches proposées, ce chapitre est organisé comme suit : Dans la première section, un état de l'art portant sur les méthodes basées sur la densité est présenté, suivi d'un descriptif de nos méthodes proposées, ainsi que d'une étude comparative empirique qui met en valeur nos contributions.

2.2 Algorithmes de clustering basés sur la densité

La famille des méthodes de clustering basées sur la densité a montré sa suprématie dans différentes applications. L'intérêt porté à ce type de méthodes est justifié par sa capacité de trouver des clusters de formes différentes et de supprimer les données bruitées, comme illustré dans la figure **??**. Dans ce contexte, plusieurs algorithmes ont été développés. DBSCAN (Ester *et al.*, 1996), l'un des algorithmes de clustering basés sur la densité, est dédié aux ensembles de données spatiales. Différentes variantes de DBSCAN ont été



(a) DENCLUE (b) DBSCAN

(c) K-means (d) Expectation Maximization

FIGURE 2.1 – Comparaison basé sur un ensemble de données de formes arbitraires (Karypis *et al.*, 1999)

proposées, telles que OPTICS (Ankerst *et al.*, 1999), ST-DBSCAN (Birant et Kut, 2007), MR-DBSCAN (He *et al.*, 2011) et MDBSCAN (Schoier et Borruso, 2017). L'algorithme OPTICS (Ordering Points To Identify the Clustering Structure) a été proposé comme amélioration de DBSCAN. Cet algorithme est moins sensible au réglage des paramètres et tente de résoudre le problème de différence de densité. Dans (Birant et Kut, 2007), les auteurs ont mis au point ST-DBSCAN pour classifier les données spatio-temporelles et résoudre le problème de détection des données bruitées dans des clusters ayant des densités différentes. Quant à MR-DBSCAN (He *et al.*, 2011, 2014), c'est une combinaison de DBSCAN et de MapReduce. Cette combinaison a permis d'équilibrer la charge entre des tâches parallèles.

Malgré les avantages de DBSCAN et ses variantes présentées dans la littérature, leur efficacité reste restreinte sur les données spatiales. Pour pallier à ces limites, Hinneburg et Keim (1998) ont proposé l'algorithme DENCLUE 1.0. La puissance de cet algorithme réside dans son efficacité à classer des données dimensionnelles et de grandes tailles. Pour construire les clusters, DENCLUE 1.0 recherche un ensemble de points appelés attracteurs de densité à l'aide de l'algorithme Hill Climbing. L'un des inconvénients majeurs de DENCLUE 1.0 est l'utilisation de l'algorithme de recherche Hill Climbing, qui ne converge pas vers le maximum local réel. Pour résoudre ce problème, Les même auteurs (Hinneburg et Gabriel, 2007) de DENCLUE 1.0 ont proposé une autre version appelée DENCLUE 2.0 qui, à son rôle, présente certaines limites.

2.3 Méthodes proposées

2.3.1 DENCLUE-SA

DENCLUE-SA (Idrissi *et al.*, 2015) est notre première version améliorée de l'algorithme DENCLUE 1.0. Notre objectif était de réduire le temps d'exécution de DENCLUE 1.0. Nous avons proposé un ajustement de l'algorithme du recuit simulé, et l'avons introduit à la place de l'algorithme Hill Climbing, pour améliorer la recherche du maximum local. Le recuit simulé (noté SA pour le terme anglais "Simulated Annealing") est un algorithme utilisé pour la recherche des optimums. C'est une méta-heuristique inspirée de la physique des matériaux "métallurgie". Cet algorithme a été introduit pour la première fois par les expériences de Metropolis (Metropolis *et al.*, 1953) pour contrôler le processus de chauffage physique. Ce procédé est utilisé pour améliorer la qualité d'un solide en recherchant un état d'énergie minimum correspondant à une structure stable du métal. Cette méthode a été adoptée par (Kirkpatrick *et al.*, 1983), et (Černỳ, 1985), afin de résoudre les problèmes d'optimisation combinatoire pour éviter le piège des minima locaux, acceptant sous certaines conditions le déplacement d'une solution s vers une autre solution s' qui est inférieure à s.
Pour ce faire, notre algorithme s'est basé sur l'équation d'énergie $\exp\left(-\frac{\Delta_E}{temp}\right)$, avec $temp$ est une température qui s'initie en paramètre et qui change avec le temps. Quant à la variation d'énergie Δ_E, nous l'avons remplacé par la variation des fonctions de densités comme montré dans l'équation 2.1.

$$Pa = \frac{\exp(f^D(x^{t-1}) - f^D(x^t))}{temp} \tag{2.1}$$

Le calcul se termine lorsque $Pa < Tr$, alors nous prenons $x^* = x^{t-1}$ en tant qu'attracteur de densité comme procédé dans DENCLUE 1.0 (cf. la sous-section 1.6.4 du chapitre 1). Avec Pa est la probabilité d'acceptation d'une solution, et Tr est un taux de refroidissement qui change d'une manière aléatoire en prenant des valeurs entre 0 et 1. Toutes les étapes de l'algorithme DENCLUE-SA sont présentées dans l'algorithme 2.1, avec :
Hr : l'hyper-rectangle construit.
cube : l'hypercube peuplé.
A : l'ensemble des attracteurs de densité.
$Aed(x_{Hcube})$: l'ensemble des points attirés par un attracteur de densité x^* donné.

2.3.2 DENCLUE-GA

Notre seconde approche présentée dans (Idrissi *et al.*, 2015), est aussi une amélioration de DENCLUE 1.0 appelée DENCLUE-GA. Dans ce travail, l'algorithme DENCLUE 1.0 a été modifié, en remplaçant l'algorithme Hill Climbing par l'algorithme génétique.
Les algorithmes génétiques (Grefenstette, 2013; Idrissi et Zegrari, 2015; Pham et Karaboga, 2012) font partie des algorithmes de méta-heuristiques d'optimisation stochastique.

Algorithme 2.1 L'algorithme DENCLUE-SA

Procédure GetClusters $(Hr, \sigma, \xi, temp)$
$A = \varnothing$
$Aed = \varnothing$
$Cluster = \varnothing$
$Clusters = \varnothing$
pour chaque $cube \in Hr$ **faire**
    pour chaque $x \in cube$ **faire**
        $x^* = getDensityAttractor(x)$
        si $(f^D(x^*) \geqslant \xi)$
        {
            $A = A \cup \{x^*\}$
            $Aed(x^*) = Aed(x^*) \cup \{x\}$
        }
    fin pour
fin pour
pour chaque $x_i^* \in A$ **faire**
    pour chaque $x_{i'}^* \in A, i \neq i'$ **faire**
        si $(dist(x_i^*, x_{i'}^*) \leqslant \sigma)$
        {
            $Aed(x_i^*) = Aed(x_i^*) \cup Aed(x_{i'}^*)$
            retirer $x_{i'}^*$ de A
        }
        sinon {
            pour chaque $x_n \in Aed(x_i^*)$ **faire**
                pour chaque $x_m \in Aed(x_{i'}^*)$ **faire**
                    si $((dist(x_n, x_m) \leqslant \sigma)$
                        et $(f^D(x_n) \geqslant \xi)$
                        et $(f^D(x_m) \geqslant \xi))$
                    {
                        $Aed(x_i^*) = Aed(x_i^*) \cup Aed(x_{i'}^*)$
                        retirer $x_{i'}^*$ de A
                    }
                fin pour
            fin pour
        }
    fin pour
    $Cluster = Aed(x_i^*)$
    ajouter $Cluster$ à $Clusters$
fin pour
Retourner $Clusters$
Fin Procédure
Procédure getDensityAttractor (x)
$t = 0$
$x^0 = x$
répéter
    $x^{t+1} = x^t + \delta \dfrac{\nabla f^D_{Gauss}(x^t)}{\|\nabla f^D_{Gauss}(x^t)\|}$
    $t = t + 1$
    $temp = temp - 1$
jusqu'à $\exp\left(\dfrac{f^D(x^{t-1}) - f^D(x^t)}{temp}\right) <$ aléatoire()
Retourner x^{t-1}
Fin Procédure

Ce type d'algorithmes a pour objectif, l'obtention d'une solution optimale dans un délai acceptable. Ces algorithmes sont inspirés de la théorie de l'évolution des espèces. Les algorithmes génétiques ont été développés à l'origine par (Holland, 1975, 1992).
L'algorithme génétique est utilisé sur une population d'individus qui évoluent selon un processus de sélection naturelle. Ce processus est représenté par des opérateurs génétiques utilisés pour créer de nouveaux individus. Ces opérateurs incluent l'opérateur de sélection, l'opérateur de croisement et l'opérateur de mutation. Nous notons qu'un individu est considéré comme une solution du problème étudié.
Pour trouver des solutions, les trois opérateurs mentionnés ci-dessus sont appliqués successivement. L'algorithme génétique calcule également une valeur d'évaluation (appelée en anglais fitness). Cette valeur est mesurée par une fonction objective.
L'algorithme génétique est itéré afin d'obtenir une nouvelle génération. Cette génération est obtenue en créant de nouveaux individus et en détruisant d'autres (mécanisme de sélection naturelle) permettant le renouvellement de la population (toutes les solutions actuelles).

Toutes les étapes de l'algorithme DENCLUE-GA sont présentées dans l'algorithme 2.2, avec :
Hr : l'hyper-rectangle construit.
cube : l'hypercube peuplé.
A : l'ensemble des attracteurs de densité.
$Aed(x_{Hcube})$: l'ensemble des points attirés par un attracteur de densité x^* donné.
$initPop$: la population initiale de l'algorithme génétique, dont le nombre d'individus est : $init$.

2.3.3 DENCLUE-IM

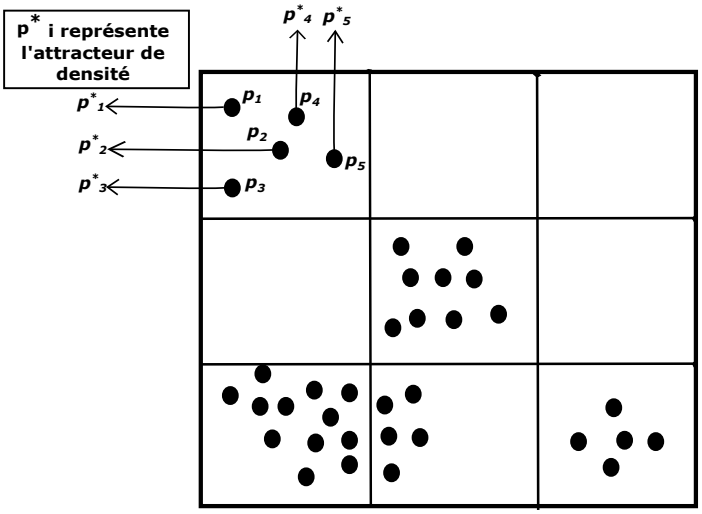
En analysant DENCLUE 1.0 et ses trois variantes, à savoir DENCLUE 2.0, DENCLUE-SA et DENCLUE-GA, nous avons constaté que l'étape visant à trouver des attracteurs de densité pose un problème lorsque la taille des données augmente. Pour résoudre ce problème, nous avons développé une version améliorée de l'algorithme DENCLUE 1.0, afin de l'ajuster efficacement aux données volumineuses.
Pour ce faire, nous avons contribué à modifier l'étape de recherche des attracteurs de densité, basée sur l'algorithme Hill Climbing. Cette étape, considérée comme déterminante dans l'algorithme DENCLUE 1.0, est basée sur des calculs de gradient qui sont faits pour chaque point afin de trouver son attracteur de densité comme illustré dans la figure 2.2. En effet le calcul des attracteurs pour chaque point est consommateur du temps d'exécution, surtout lorsqu'il s'agit du traitement de grandes bases de données.

Dans les mêmes circonstances, les autres versions de DENCLUE 1.0 effectuent un calcul basé sur chaque point de l'ensemble de données, leur temps d'exécution augmente en fonction de la taille des données.
Notre algorithme nommé DENCLUE-IM permet de trouver un élément équivalent à l'at-

Algorithme 2.2 L'algorithme DENCLUE-GA

Procédure GetClusters $(Hr, \sigma, \xi, init)$
$A = \varnothing$, $Aed = \varnothing$
$Cluster = \varnothing$, $Clusters = \varnothing$
$initPop = \varnothing$
pour chaque $cube \in Hr$ **faire**
    pour chaque $x \in cube$ **faire**
        $x^* = getDensityAttractor(x, init)$
        si $(f^D(x^*) \geqslant \xi)$
        {
            $A = A \cup \{x^*\}$
            $Aed(x^*) = Aed(x^*) \cup \{x\}$
        }
    fin pour
fin pour
pour chaque $x_i^* \in A$ **faire**
    pour chaque $x_{i'}^* \in A, i \neq i'$ **faire**
        si $(dist(x_i^*, x_{i'}^*) \leqslant \sigma)$
        {
            $Aed(x_i^*) = Aed(x_i^*) \cup Aed(x_{i'}^*)$
            retirer $x_{i'}^*$ de A
        }
        sinon {
            pour chaque $x_n \in Aed(x_i^*)$ **faire**
                pour chaque $x_m \in Aed(x_{i'}^*)$ **faire**
                    si $((dist(x_n, x_m) \leqslant \sigma)$
                        et $(f^D(x_n) \geqslant \xi)$
                        et $(f^D(x_m) \geqslant \xi))$
                    {
                        $Aed(x_i^*) = Aed(x_i^*) \cup Aed(x_{i'}^*)$
                        retirer $x_{i'}^*$ de A
                    }
                fin pour
            fin pour
        }
    fin pour
    $Cluster = Aed(x_i^*)$
    ajouter $Cluster$ à $Clusters$
fin pour
Retourner $Clusters$
Fin Procédure
Procédure getDensityAttractor (x)
$t = 0$
$x^0 = x$
répéter
    $x^{t+1} = x^t + \delta \dfrac{\nabla f_{Gauss}^D(x^t)}{\|\nabla f_{Gauss}^D(x^t)\|}$
    $t = t + 1$
    ajouter x^t à $initPop$
    $init = init - 1$
jusqu'à $init \leq 0$
$x^t = GeneticAlgorithm(initPop)$
Retourner x^t
Fin Procédure

FIGURE 2.2 – Les attracteurs de densité dans un hyper-rectangle de données bidimensionnelles

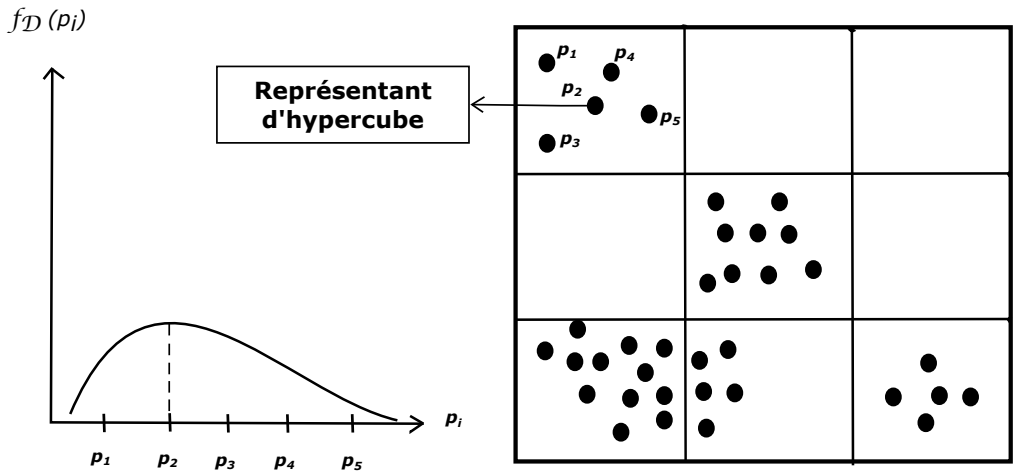


tracteur de densité, qui représentera tous les points contenus dans un hypercube, au lieu des calculs effectués pour chaque point de l'ensemble de données (cf. l'équation (1.11)). Ce représentant d'hypercube noté x_{Hcube}, sera considéré comme le point ayant la densité la plus élevée dans cet hypercube comme indiqué dans l'équation (2.2) et illustré dans la figure 2.3.

$$\forall x \in C_p \quad f_D(x) \leq f_D(x_{Hcube}), \tag{2.2}$$

où C_p est l'un des hypercubes peuplés dans l'hyper-rectangle construit.
Ainsi chaque hypercube constitue un cluster initial représenté par son x_{Hcube}. Ces clusters seront fusionnés si et seulement s'il existe un chemin entre leurs représentants.

FIGURE 2.3 – Le représentant d'un hypercube dans un hyper-rectangle de données bidimensionnelles



Plus de détails sur l'approche proposée sont donnés dans l'algorithme 2.3, en considérant :
Hr : l'hyper-rectangle construit.
cube : l'hypercube peuplé.
R : l'ensemble des représentants d'hypercubes.
$Red(x_{Hcube})$: l'ensemble des points représentés par un représentant d'hypercube x_{Hcube} donné.

Algorithme 2.3 L'algorithme DENCLUE-IM

Procédure GetClusters (Hr, σ, ξ)
$R = \varnothing$
$Red(null) = \varnothing$
$Cluster = \varnothing$
$Clusters = \varnothing$
pour chaque $cube \in Hr$ **faire**
    $x_{Hcube} = null$
    $f^D(x_{Hcube}) = 0$
    pour chaque $x \in cube$ **faire**
        si $(f^D(x) \geqslant f^D(x_{Hcube}))$
        {
            $x_{Hcube} = x$
        }
    fin pour
    si $(f^D(x_{Hcube}) \geqslant \xi)$
    {
        $R = R \cup \{x_{Hcube}\}$
        $Red(x^*) = Red(x_{Hcube}) \cup cube$
    }
fin pour
pour chaque $x_{Hcube_i} \in R$ **faire**
    pour chaque $x_{Hcube_{i'}} \in R, i \neq i'$ **faire**
        si $(dist(x_{Hcube_i}, x_{Hcube_{i'}}) \leqslant \sigma)$
        {
            $Red(x_{Hcube_i}) = Red(x_{Hcube_i}) \cup Red(x_{Hcube_{i'}})$
            retirer $x_{Hcube_{i'}}$ de R
        }
        sinon {
            pour chaque $x_n \in Red(x_{Hcube_i})$ **faire**
                pour chaque $x_m \in Red(x_{Hcube_{i'}})$ **faire**
                    si $(dist(x_n, x_m) \leqslant \sigma)$
                        et $(f^D(x_n) \geqslant \xi)$
                        et $(f^D(x_m) \geqslant \xi))$
                    {
                        $Red(x_{Hcube_i}) = Red(x_{Hcube_i}) \cup Red(x_{Hcube_{i'}})$
                        retirer $x_{Hcube_{i'}}$ de R
                    }
                fin pour
            fin pour
        }
    fin pour
    $Cluster = Red(x_{Hcube_i})$
    ajouter $Cluster$ à $Clusters$
fin pour
Retourner $Clusters$
Fin procédure

2.4 Résultats expérimentaux

2.4.1 Description des données

Pour évaluer l'efficacité de nos approches, nous avons utilisé huit ensembles de données :

— NPC : Cet ensemble de données fait parti d'un projet visant à évaluer les facteurs pronostiques du carcinome du nasopharynx (NPC : en anglais NasoPharyngeal Carcinoma) (il s'agit de la partie du pharynx située derrière la cavité nasale).

— Iris : Cet ensemble de données est aussi connu sous le nom de Iris de Fisher, présenté en 1936 par Ronald Fisher. L'ensemble Iris contient 50 échantillons de chacune des trois espèces de la fleur iris (Iris setosa, Iris virginica et Iris versicolor). Chaque entrée de l'ensemble comporte 4 attributs mesurés à partir de chaque échantillon : la longueur et la largeur des sépales et des pétales, en centimètres. L'ensemble Iris est extrait du référentiel d'apprentissage machine UCI [1].

— Stulong : Il s'agit d'une base de données concernant une étude des facteurs de risque de l'athérosclérose dans une population de 1419 personnes nées moyennement entre 1976 et 1999. Chaque entrée comporte cinq attributs. Les attributs concernent la taille, le poids, la pression artérielle systolique, la pression artérielle diastolique et le taux de cholestérol (Meger *et al.*, 2004). Cet ensemble de données est extrait du référentiel d'apprentissage KEEL [2].

— SpamBase : Cet ensemble de données, également extrait du référentiel d'apprentissage machine UCI [1], illustre les emails classifiés comme spam ou non-spam (Hopkins *et al.*, 1999).

— PageBlocks : Cet ensemble de données, extrait du référentiel d'apprentissage machine UCI [1], présente des blocs classés de la mise en page dans un document qui a été détecté par un processus de segmentation (Esposito *et al.*, 1994).

— PenDigits : Cet ensemble de données a été créé en recueillant 250 échantillons de 44 auteurs (Alimoglu *et al.*, 1996). PenDigits est également extrait du référentiel d'apprentissage machine UCI [1].

— CreditCard : Ce quatrième ensemble de données a été présenté dans (Yeh et Lien, 2009), dans le but de découvrir le crédit ou des clients non fiables. Nous avons extrait ces données du référentiel d'apprentissage machine UCI [1].

— CloudServices : Ces données sont constituées de 50 000 services cloud, chacun composé de 10 attributs (Abourezq et Idrissi, 2014a, 2015a).

La description des ensembles de données utilisés est présentée dans le tableau 2.1.

1. https ://archive.ics.uci.edu/ml/index.php
2. https ://sci2s.ugr.es/keel/datasets.php

Tableau 2.1 – Description des ensembles de données.

ensembles de données	# observations	# attributs	# classes
NPC	90	26	4
Iris	150	4	3
Stulong	1419	5	données non labellisées
SpamBase	4601	57	2
PageBlocks	5472	10	5
PenDigits	10992	16	10
CreditCard	30000	23	2
CloudServices	50000	10	données non labellisées

2.4.2 Évaluation des résultats

Nous rappelons qu'à travers ce chapitre, nous avons proposé trois améliorations de DENCLUE 1.0. Il s'agit de DENCLUE-SA, DENCLUE-GA et DENCLUE-IM. Pour montrer l'apport des méthodes proposées sur les performances de clustering, nous les avons testées sur les ensembles de données décrits dans la sous-section 2.4.1.
Pour ce faire, nous avons conduit une comparaison avec DENCLUE 1.0 et DENCLUE 2.0 en utilisant les mesures d'évaluation décrites dans la section 1.7 du chapitre 1.

Tableau 2.2 – Comparaison entre les cinq algorithmes en se basant sur les mesures d'évaluation internes.

Mesures	Algorithmes		NPC	Iris	Stulong	PageBlocks	SpamBase	PenDigits	CreditCard	Cloud Services
DI (à maximiser)	état de l'art	DENCLUE 1.0	0.489	**0.479**	0.598	0.721	0.789	0.844	0.845	0.898
		DENCLUE 2.0	0.512	**0.479**	0.606	**0.737**	0.795	**0.896**	**0.918**	**0.909**
	Proposés	DENCLUE-SA	0.547	**0.479**	0.466	0.721	0.821	0.828	0.848	0.898
		DENCLUE-GA	**0.562**	**0.479**	0.747	0.721	**0.835**	0.813	0.810	0.846
		DENCLUE-IM	0.518	**0.479**	**0.765**	0.693	0.831	0.864	0.811	0.899
DBI (à minimiser)	état de l'art	DENCLUE 1.0	**2.110**	**0.628**	1.619	0.563	0.867	2.837	2.149	**1.254**
		DENCLUE 2.0	2.166	**0.628**	2.287	2.304	0.953	**1.414**	2.212	3.388
	Proposés	DENCLUE-SA	3.062	**0.628**	1.201	0.474	**0.764**	2.880	2.166	1.714
		DENCLUE-GA	2.722	**0.628**	1.177	0.639	0.968	3.748	2.616	2.413
		DENCLUE-IM	2.506	**0.628**	**1.066**	**0.412**	1.041	1.950	**2.054**	1.262
CP (à minimiser)	état de l'art	DENCLUE 1.0	0.857	**0.086**	2.010	1.376	1.733	1.115	2.121	1.720
		DENCLUE 2.0	0.940	**0.086**	2.038	1.281	**1.019**	**0.535**	3.603	3.000
	Proposés	DENCLUE-SA	**0.781**	**0.086**	2.022	1.348	1.586	1.083	2.167	**1.381**
		DENCLUE-GA	0.802	**0.086**	2.221	1.014	1.206	1.316	2.590	1.958
		DENCLUE-IM	0.822	**0.086**	**1.333**	**0.663**	1.821	0.966	**1.450**	1.502

2.4.2.1 Résultats basés sur des mesures d'évaluation internes

D'après le tableau 2.2, nous pouvons remarquer qu'en appliquant les algorithmes sur l'ensemble de données Iris, les cinq algorithmes ont eu les mêmes résultats pour les trois mesures internes, à savoir, DI, DBI et CP.
Concernant l'ensemble Stulong, DENCLUE-IM a eu les meilleures valeurs des trois mesures internes.
En se basant sur les résultats de l'ensemble PageBlock, DENCLUE 2.0 a eu la meilleure

valeur de l'indice DI, la différence entre cette valeur obtenue de l'algorithme DENCLUE 2.0 et celle de DENCLUE-IM est de 0.044. Pour la deuxième et la troisième mesure interne, DENLUE-IM a eu les meilleures valeurs.
En ce qui concerne l'ensemble SpamBase, DENCLUE-GA a eu la meilleure valeur de l'indice DI, d'à peu prés 0.004 plus grande que celle de DENCLUE-IM, nous notons également que DENCLUE-IM a eu la deuxième meilleure valeur de l'indice DI. DENCLUE-SA a eu la meilleure valeur de l'indice DBI, 0.277 inférieure à celle de DENCLUE-IM. Alors que DENCLUE 2.0 a obtenu la meilleure valeur de l'indice CP, 0.802 inférieure à celle de l'approche proposée.
Pour l'ensemble de données PenDigits, l'algorithme DENCLUE 2.0 a eu la meilleure valeur pour les trois mesures d'évaluation, suivies par DENCLUE-IM d'une différence de valeur de 0.032 pour l'indice DI, 0.536 pour l'indice DBI et 0.431 pour l'indice PC.
Pour l'ensemble de données de CreditCard, DENCLUE 2.0 a obtenu la meilleure valeur de l'indice DI, 0.107 plus grande que celle de DENCLUE-IM. En ce qui concerne les indices DBI et CP, DENCLUE-IM a eu les meilleures valeurs.
En ce qui concerne l'ensemble Cloud Services, DENCLUE 2.0 a obtenu la meilleure valeur de l'indice DI, soit 0.010 plus grande que celle de DENCLUE-IM. Alors que DENCLUE 1.0 a eu la meilleure valeur de l'indice DBI, qui est 0.008 plus petit que celui de DENCLUE-IM. Enfin DENCLUE-SA a obtenu le meilleur indice CP, 0.121 plus petit que celui de DENCLUE-IM.
Pour le dernier ensemble de données, NPC, l'algorithme EM a eu la meilleur valeur de l'indice DI, tandis que DENCLUE a obtenu le meilleur indice DBI et DENCLUE-SA a eu le meilleur indice CP.

Tableau 2.3 – Comparaison entre les cinq algorithmes en se basant sur les mesures d'évaluation externes.

Mesures	Algorithmes		NPC	Iris	PageBlocks	SpamBase	PenDigits	CreditCard
CA (à maximiser)	état de l'art	DENCLUE 1.0	0.507	0.666	0.920	**0.805**	0.970	0.809
		DENCLUE 2.0	0.379	0.666	**0.943**	0.698	0.801	0.774
	Proposés	DENCLUE-SA	**0.592**	0.666	0.920	0.789	**0.973**	**0.812**
		DENCLUE-GA	0.526	0.666	0.916	0.718	0.961	0.801
		DENCLUE-IM	0.560	0.666	0.911	0.727	0.839	0.777
Entropy (à minimiser)	état de l'art	DENCLUE 1.0	1.283	0.666	0.352	**0.103**	0.101	0.502
		DENCLUE 2.0	1.817	0.666	**0.089**	0.278	0.173	0.647
	Proposés	DENCLUE-SA	**0.806**	0.666	0.335	0.106	**0.082**	**0.465**
		DENCLUE-GA	1.262	0.666	0.382	0.174	0.118	0.564
		DENCLUE-IM	1.124	0.666	0.232	0.144	0.335	0.701
NMI (à maximiser)	état de l'art	DENCLUE 1.0	0.088	0.733	0.085	**0.097**	0.592	0.036
		DENCLUE 2.0	0.012	0.733	0.122	0.016	0.235	0.009
	Proposés	DENCLUE-SA	0.155	0.733	0.105	0.075	0.565	0.041
		DENCLUE-GA	0.154	0.733	0.040	0.087	**0.599**	**0.048**
		DENCLUE-IM	**0.245**	0.733	**0.186**	0.050	0.505	0.017

Tableau 2.4 – Comparaison entre les cinq algorithmes en fonction de leur temps d'exécution (en seconde).

Algorithmes		NPC	Iris	Stulong	PageBlocks	SpamBase	PenDigits	CreditCard	Cloud Services
état de l'art	DENCLUE	0.058	0.060	1.430	71.028	1285.911	2582.538	188709.749	116202.129
	DENCLUE 2.0	0.100	0.071	1.728	25.88	185.661	2403.451	4685.922	12661.748
Proposés	DENCLUE-SA	0.092	0.051	1.676	107.852	1347.818	2684.75	188010.124	115906.865
	DENCLUE-GA	0.174	0.138	4.440	158.055	574.382	2501.988	77793.938	113650.162
	DENCLUE-IM	**0.026**	**0.031**	**0.355**	**5.749**	**27.540**	**616.406**	**1480.438**	**1675.508**

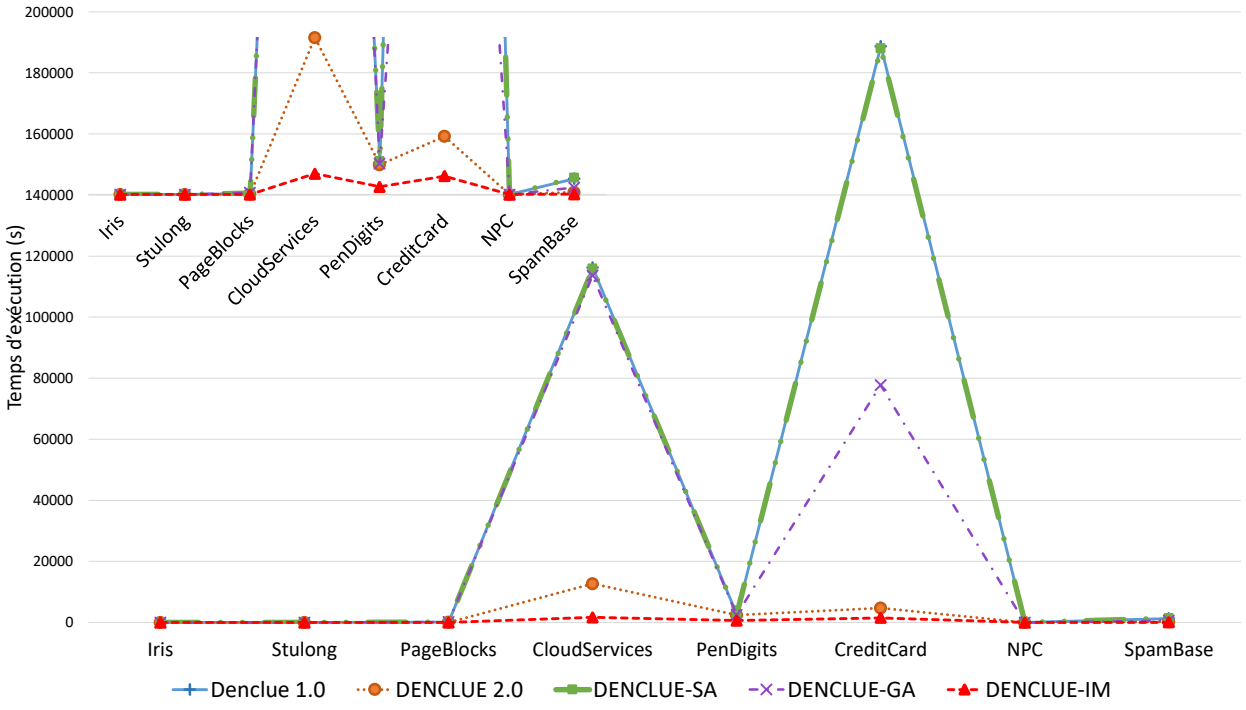
2.4.2.2 Résultats basés sur des mesures d'évaluation externes

En termes de mesures externes ; en appliquant les cinq algorithmes sur l'ensemble de données Iris, nous avons obtenus des résultats ex-æquo pour les trois mesures externes, à savoir, l'indice CA, l'entropie et le NMI.
Pour l'ensemble de données Stulong, les mesures externes n'ont pas été calculées car il s'agit d'un ensemble de données non étiqueté.
Pour le troisième ensemble de données, DENCLUE 2.0 a eu la meilleure valeur de l'indice CA, la différence entre le CA de DENCLUE 2.0 et celui de DENCLUE-IM est de 0.032. Pour la seconde mesure, DECNLUE-IM a eu la deuxième meilleure valeur, soit de 0.143 plus grande que la valeur d'entropie de DENCLUE 2.0. Concernant la mesure NMI, DENCLUE-IM a obtenu la meilleure valeur.
Concernant SpamBase, l'algorithme DENCLUE 1.0 a eu les meilleures valeurs des trois mesures. La valeur de l'indice CA obtenu par DENCLUE 1.0 est de 0.078 supérieure à celle de DENCLUE-IM, 0.041 plus petite que l'entropie de DENCLUE-IM, et 0.047 plus grande que la valeur de l'indice NMI de DENCLUE-IM.
Concernant l'ensemble PenDigits, l'algorithme DENCLUE-SA a obtenu les meilleures valeurs des indices CA et entropie. La meilleure valeur de CA est 0.134 plus grande que celle de DENCLUE-IM et 0.253 plus petite que son entropie. En ce qui concerne le NMI, DENCLUE-GA a eu la meilleure valeur, qu'est de 0.094 plus grande que DENCLUE-IM.
Pour l'ensemble de données CreditCard, DENCLUE-SA a obtenu les meilleures valeurs des deux mesures CA et entropie. La valeur de l'indice CA est de 0.035 supérieure à celle de DENCLUE-IM et de 0.236 inférieure à l'entropie de ce dernier. Pour l'indice NMI, DENCLUE-GA a eu la meilleure valeur, supérieur de 0.031 à celle de DENCLUE-IM.
Pour l'ensemble de données NPC, DENCLUE-SA a eu le meilleur indice CA et la meilleure entropie, suivi de DENCLUE-IM qui a obtenu les deuxièmes meilleures valeurs pour ces deux indices et qui a aussi eu le meilleur indice NMI.
En ce qui concerne l'ensemble de données Cloud Services, les mesures externes n'ont pas pu être calculées car il s'agit d'un ensemble de données non étiqueté.

2.4.2.3 Résultats basés sur le temps d'exécution

Nous étudions dans cette partie la performance des algorithmes en termes de temps d'exécution. les cinq algorithmes sont implémentés en utilisant une plateforme JAVA, sur

FIGURE 2.4 – Impact de la dimension des données sur la croissance du temps d'exécution



un PC Core 2 Duo (2,70 GHz) avec 4 Go de mémoire. Le tableau 2.4 enregistre le temps d'exécution de chaque méthode. Nous pouvons constater que DENCLUE-IM obtient toujours les temps d'exécutions les plus rapides.

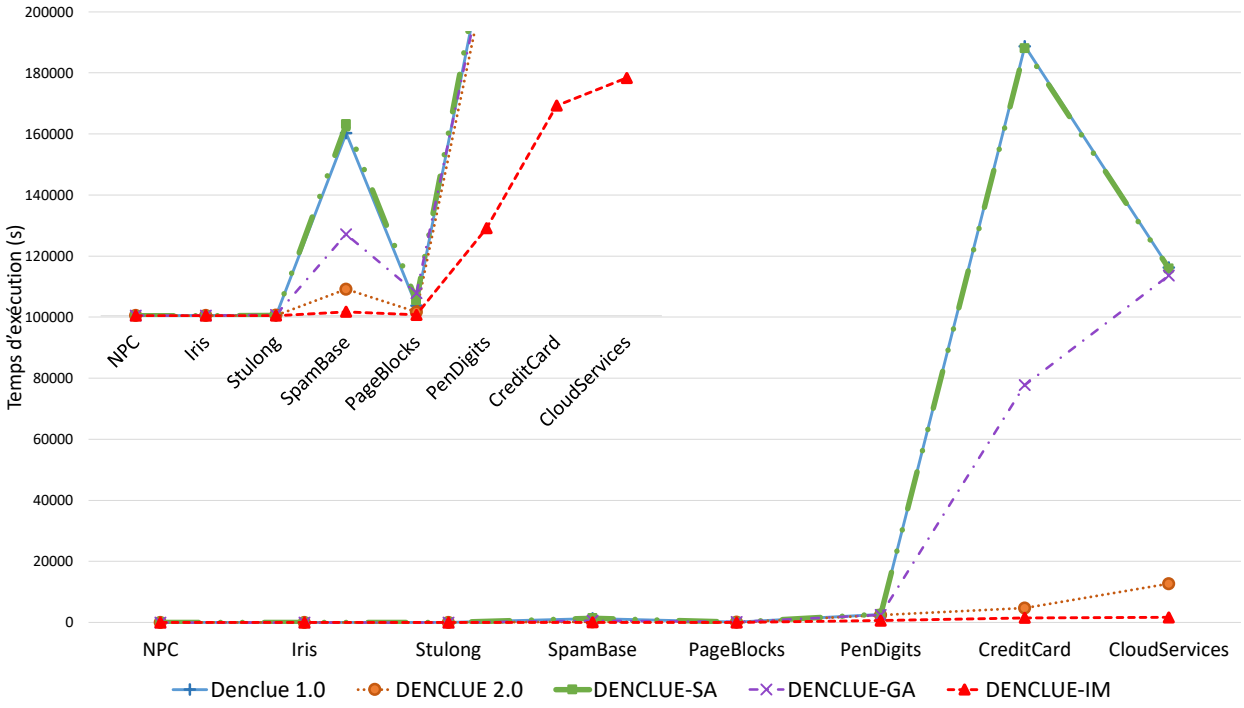
Pour le temps d'exécution de l'ensemble Iris, DENCLUE-IM a eu le meilleur temps, suivi de DENCLUE-SA par une différence de 0.02, de DENCLUE 1.0 par une différence de 0.029, de DENCLUE 2.0 par une différence de 0.04, enfin de DENCLUE-GA par une différence de 0.107.

Concernant l'ensemble Stulong, DENCLUE-IM est toujours en première place, suivi de DENCLUE 1.0 par une différence de 1.075, puis de DENCLUE-SA par une différence de 1.321, de DENCLUE 2.0 par une différence de 1.373, enfin DENCLUE-GA par une différence de 4.085.

Comme le montre le tableau 2.4, dans l'ensemble de données PageBlocks, le temps d'exécution de DENCLUE-IM est réduit de 5 fois par rapport à celui de DENCLUE 2.0 et de 12 fois par rapport à celui de DENCLUE 1.0. En ce qui concerne DENCLUE-SA et DENCLUE-GA, ils nécessitent un temps d'exécution multiplié approximativement par 19 et 27 respectivement, par rapport à DENCLUE-IM.

En ce qui concerne les temps d'exécution de l'ensemble SpamBase, celui de DENCLUE-IM est multiplié par environ 7 fois le temps obtenu par DENCLUE 2.0, suivi de celui de DENCLUE-GA, qui équivaut approximativement à 21 fois le temps d'exécution

FIGURE 2.5 – Impact de la taille des données sur la croissance du temps d'exécution



de DENCLUE-IM. DENCLUE 1.0 et DENCLUE-SA sont venu en quatrième et cinquième position, respectivement d'environ 47 et 49 fois plus que la durée d'exécution de DENCLUE-IM.

Dans le cinquième ensemble, PenDigits, le temps d'exécution de DENCLUE-IM est optimisé par environ 4 fois celui des autres algorithmes, à savoir DENCLUE 1.0, DENCLUE 2.0, DENCLUE-SA et DENCLUE-GA.

D'autre part, dans l'ensemble CreditCard, DENCLUE-IM n'a eu besoin que de 24 minutes pour accomplir le clustering, alors que DENCLUE 2.0 a terminé l'exécution après 1 heure 30 minutes, ce qui est environ 3 fois le temps d'exécution DENCLUE-IM, suivi du DENCLUE-GA, qui a nécessité 21 heures, environ 53 fois le temps de DENCLUE-IM. Et en dernier lieu, DENCLUE-SA et DENCLUE 1.0 qui ont nécessité environ 52 heures, un temps d'exécution qui est multiplié à peu près par 127 fois le temps obtenu par DENCLUE-IM.

Enfin, dans le dernier ensemble de données, à savoir Cloud Services, DENCLUE-IM a nécessité environ 28 minutes, suivi en deuxième position par DENCLUE 2.0, qui a nécessité environ 3 heures et 30 minutes ; les trois autres algorithmes ont nécessité approximativement 32 heures, ce qui est équivaut à un temps d'exécution multiplié par 68 par rapport à celui de DENCLUE-IM.

Pour mieux analyser les résultats obtenus, notamment en termes de temps d'exécution,

nous avons décidé de scinder les résultats en deux graphiques, comme illustré dans les figures 2.4 et 2.5. La première figure 2.4 et son zoom, illustré au fond de la figure elle même, montrent la croissance du temps d'exécution par rapport à celle de la dimension des données. Cette figure permet de percevoir que DENCLUE 2.0 et DENCLUE-IM sont moins sensibles à l'expansion de la dimension que les trois autres algorithmes.
Quant à la figure 2.5 et son zoom, qui est aussi illustré à l'intérieur de la figure, ils représentent la croissance du temps d'exécution en fonction de l'augmentation de la taille des données. À travers cette figure, nous constatons que l'algorithme DENCLUE-IM reste le meilleur choix pour le clustering des données volumineuses.

2.5 Conclusion

Dans ce chapitre, nous avons discuté l'impact de la croissance du volume et de la dimensionalité sur la qualité du clustering de données. Nous avons aussi proposé de nouvelles approches basées sur la densité pour améliorer l'algorithme DENCLUE 1.0 permettant ainsi d'assurer un compromis entre la qualité des résultats et le temps de réponse.
En se basant sur six mesures d'évaluation (internes et externes) nous avons pu démontrer que chacune des approches a sa particularité que ça soit en termes de qualité ou de temps de réponse. Cependant DENCLUE-IM reste un choix judicieux, vu qu'il a obtenu des résultats satisfaisants en termes de mesures d'évaluation ainsi qu'un meilleur temps de réponse pour les huit ensembles de données.
Malgré les résultats de clustering manifestés par nos approches, elles souffrent d'un problème de classification des données chevauchées et ayant des distributions de densités différentes. Des solutions pour cette problématique bien spécifique ont été proposées dans le chapitre suivant.

Chapitre

3

Contribution au clustering des données chevauchées

Sommaire

3.1 Introduction

Le rôle du clustering, comme expliqué précédemment dans les chapitres 1 et 2, est la découverte des clusters qui sont considérés comme groupes de données homogènes. Cependant, la répartition des données dans les clusters peux être erronée. Si les données de deux clusters sont chevauchées, il est fort probable qu'elles soient intégrées dans un même cluster au lieu de se répartir en deux. L'une des premières remarques qui avait attiré notre attention sur ce problèmes de chevauchements de données, ou autrement dit séparation de clusters, été la distribution des données dans l'ensemble Iris (Fisher, 1936). Cet ensemble de données qui comporte trois classes à savoir, iris-setosa, iris-virginica et iris-versicolor, souffre d'un chevauchement entre deux de ses classes qui sont iris-virginica et iris-versicolor. Ce chevauchement conduit à des résultats de clustering erronés. En effet, les algorithmes de clustering dans ce cas n'arrivent pas à distinguer entre les deux classes et les regroupent dans un même cluster, d'où les faibles résultats générés.
Après l'analyse minutieuse de cette problématique, nous avons proposé une méthode permettant de bien séparer les données des deux classes chevauchées toute en contribuant à l'amélioration des performances des résultats du clustering.

Ce chapitre décrit les étapes de notre approche, en commençant par une présentation de l'état de l'art suivi du descriptif de l'approche, ainsi que d'une étude expérimentale

démontrant son efficacité, avant de terminer par une conclusion.

3.2 Problème de chevauchement de données : état de l'art

La recherche de clusters à partir d'ensembles de données présentant un problème de chevauchement reste un défi pour les algorithmes de clustering. Dans ce contexte, de nombreux travaux ont été publiés dans l'objectif de relever ce défis.
L'une des approches proposées, dans ce sens, est celle basé sur les algorithmes de clustering flou. Ce type de méthodes permet à un objet d'appartenir à un ou plusieurs clusters à la fois, en respectant certaines conditions d'appartenance. Les auteurs de (Zhou *et al.*, 2015) ont proposé une méthode de classification (WFCS pour "weighting fuzzy compactness and separation") fondée sur la compacité et la séparation floue des vecteurs caractéristiques. Le but ultime de cette méthode est la capacité de trouver des clusters durs et flous afin de mieux gérer le problème de chevauchement. Dans (Mohammed et Lim, 2017), l'algorithme Fuzzy Min-Max (FMM) a été amélioré en le combinant avec les règles d'extension du K-plus-proche-hyperbox. Ainsi la création de nombreuses petites hyperboxes au cours de la phase d'apprentissage du FMM sera évitée. Le but de cette amélioration est de découvrir les clusters chevauchés avec une complexité réduite.

D'autres approches basées sur le clustering semi-supervisé se sont intéressées également à ce problème. Dans (Ding *et al.*, 2014), un algorithme de clustering spectrale semi-supervisé a été élaboré, afin de mieux regrouper les éléments qui se chevauchent, cet algorithme ajuste les différences de distances entre les objets en fonction des informations de contraintes par paire, en tenant compte de la connaissance des liaisons obligatoires (must-link) et celles impossibles (cannot-link) (Wagstaff *et al.*, 2001). Dans la même optique, les auteurs de (Xiao *et al.*, 2016), ont développé une version améliorée de l'algorithme CHAMELEON, l'un des algorithmes de classification hiérarchique, en utilisant des connaissances semi-supervisées, c'est-à-dire les étiquettes et les contraintes par paires. Dans d'autres travaux, les auteurs ont combiné entre le clustering flou et celui semi-supervisé, comme présenté dans (Abin et Beigy, 2015).

Cependant, dans certaines données provenant du monde réel, nous ne pouvons pas utiliser les clusters flous au lieu de ceux durs (cf. la sous section 1.4.4). Nous ne pouvons non plus toujours acquérir certaines connaissances des données comme les étiquettes ou les contraintes par paires.

Pour faire face à ses défis, notre approche, nommée SDA (Scaled Data Algorithm), se base d'une manière automatique sur les données eux même sans modifier les algorithmes de clustering. Les données souffrant du problème de chevauchement sont mises à l'échelle en fonction de leur distribution de densités.

3.3 Mise à l'échelle des données de clustering

Pour obtenir une meilleure qualité de clustering des données chevauchées, nous avons proposé une méthode de mise à l'échelle (SDA) permettent une meilleure séparation de données.
Pour ce faire, nous avons calculé la densité de chaque point en se basant sur le noyau gaussien. Ce dernier a pour effet d'intercepter les informations non linéaires dans l'ensemble de données afin de mesurer la distance ou la similarité entre les échantillons.
Notre nouvelle fonction de densité $fatt_D$ est présentée dans l'équation (3.1). Cette dernière renvoie la densité de chaque attribut composant le vecteur caractéristique d'un point donné $x_i = [x_{i1}, x_{i2}, ..., x_{id}]$ de d-dimension.

$$fatt_D(x_{ij}) = \sum_{i'=1}^{N} fatt_{Gauss}(x_{ij}, x_{i'j}), \tag{3.1}$$

où D représente l'ensemble de données, N son cardinal, $j \in \{1, 2, ..., d\}$, $i \in \{1, 2, ..., N\}$, et $fatt_{Gauss}$ est la fonction gaussienne entre deux attributs x_{ij} et $x_{i'j}$ comme représenté dans l'équation (3.2).

$$fatt_{Gauss}(x_{ij}, x_{i'j}) = \exp^{-\frac{dist(x_{ij}, x_{i'j})^2}{2\sigma_j^2}}. \tag{3.2}$$

L'écart type σ_j est calculé comme indiqué dans l'équation (3.3).

$$\sigma_j = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_{ij} - \mu_j)^2}, \ avec \ \mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij}. \tag{3.3}$$

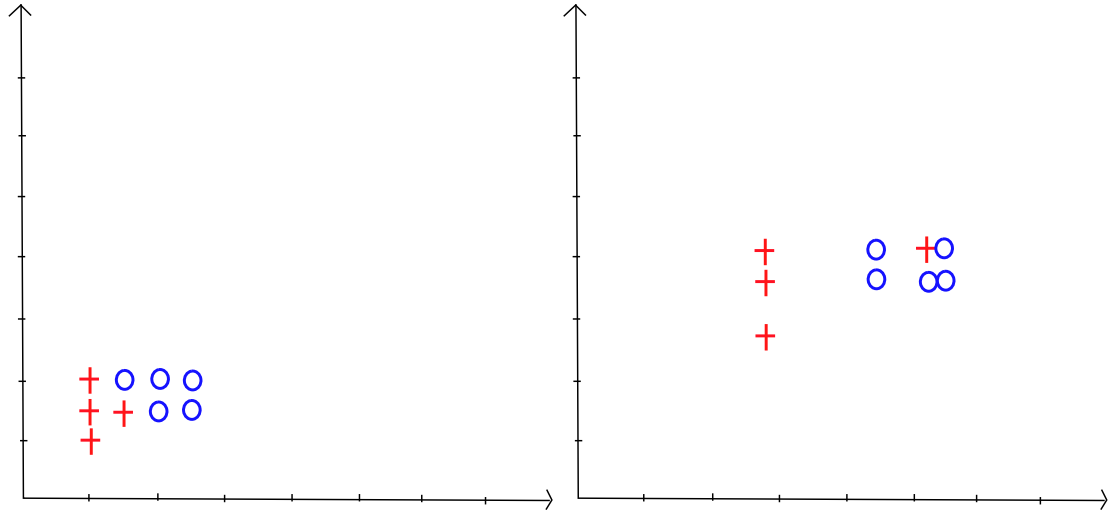
Ensuite, chaque attribut constituant un point suit une translation comme indiqué dans l'équation (3.4).

$$scaled(x_{ij}) = x_{ij} + fatt_D(x_{ij})^j. \tag{3.4}$$

L'algorithme global de notre méthode SDA est présenté dans Algorithme 3.1.

Algorithme 3.1 L'algorithme SDA

```
1: Procédure ScaleData (D)
2: ScaledD = ∅
3: pour chaque x ∈ D faire
4:     j = 1
5:     pour chaque attr ∈ x faire
6:         x_scaled.attr = attr + fatt_D(attr)^j
7:         j = j + 1
8:     fin pour
9: fin pour
10: Retourner ScaledD
11: Fin procédure
```



(a) Ensemble de données original (b) Ensemble de données mis à l'échelle

FIGURE 3.1 – Exemple de la mise à l'échelle obtenue par notre approche SDA

Pour mieux comprendre l'approche proposée, un exemple illustratif est représenté dans la figure 3.1. La sous-figure 3.1a, dont les coordonnées sont présentées ci-dessous à gauche, illustre les points représentant les données originales provenant de deux clusters chevauchées (cluster rouge et cluster bleu). Quant à la sous-figure 3.1b, elle illustre les données mises à l'échelle par notre approche. Afin de séparer au mieux les données, notre méthode a calculé, à partir des points originaux (cf. la figure 3.1), les nouvelles coordonnées présentées ci-dessous :

(1.0 , 1.5) $\longrightarrow$ (2.889 , 4.134)
(1.0 , 1.0) $\longrightarrow$ (2.889 , 2.803)
(1.5 , 1.5) $\longrightarrow$ (5.382 , 4.134)
(1.0 , 2.0) $\longrightarrow$ (2.889 , 3.715)
(1.5 , 2.0) $\longrightarrow$ (5.382 , 3.715)
(2.0 , 1.5) $\longrightarrow$ (5.418 , 4.134)
(2.5 , 1.5) $\longrightarrow$ (4.423 , 4.134)
(2.5 , 2.0) $\longrightarrow$ (4.423 , 3.715)
(2.0 , 2.0) $\longrightarrow$ (5.418 , 3.715)

Notons que les anciennes coordonnées sont à gauche et les nouvelles sont à droite.

3.4 Résultats expérimentaux

3.4.1 Description des données

Afin de prouver l'efficacité de notre approche, nous avons utilisé quatre ensembles de données réels provenant du référentiel d'apprentissage UCI (Lichman, 2013). Le tableau 3.1 présente les caractéristiques des ensembles de données utilisés.

Tableau 3.1 – Description des ensembles de données.

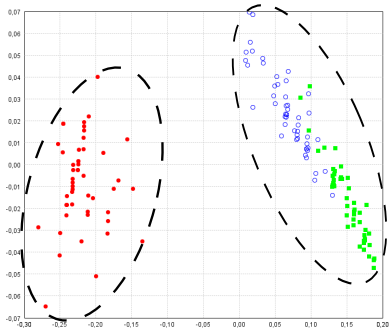
Ensembles de données	# observations	# attributs	# classes
Iris	150	4	3
Wine	178	13	3
WDBC	569	30	2
Seeds	210	7	3

— Iris : Cet ensemble de données est aussi connu sous le nom de Iris de Fisher, présenté en 1936 par Ronald Fisher. L'ensemble Iris contient 50 échantillons de chacune des trois espèces de la fleur iris (Iris setosa, Iris virginica et Iris versicolor). Chaque entrée de l'ensemble comporte 4 attributs mesurés à partir de chaque échantillon : la longueur et la largeur des sépales et des pétales, en centimètres. L'ensemble Iris est extrait du référentiel d'apprentissage machine UCI.
— Wine : Cet ensemble contient des données issues des résultats d'une analyse chimique de vins cultivés de trois cultivars différents dans une région en Italie. Cette analyse menée a déterminé l'existence de 13 composants trouvés dans chacun des trois types de vin. L'ensemble de données Wine est extrait du référentiel d'apprentissage machine UCI.
— WDBC : Cet ensemble représente des données du diagnostique du cancer du sein au Wisconsin. Les vecteurs caractéristiques de cet ensemble sont calculés à partir d'une image numérisée d'une aspiration à l'aiguille fine (FNA) d'une masse mammaire. Les caractéristiques sont une description des noyaux cellulaires présents dans l'image. Cet ensemble de données est extrait du référentiel d'apprentissage machine UCI.
— Seeds : L'ensemble de données Seeds se constitue de 70 échantillons de noyaux appartenant à chacune des trios variété de blé : Kama, Rosa et Canadian. Ces grains de blé récoltés avec des moissonneuses-batteuses proviennent des champs d'expérimentations de l'Institut d'agrophysique de l'Académie des sciences de Pologne à Lublin. Les vecteurs caractéristiques de cet ensemble sont calculés en se basant sur des images obtenues à l'aide d'une technique de rayons X mené sur la structure interne du noyau. L'ensemble de données Seeds est extrait du référentiel d'apprentissage machine UCI.

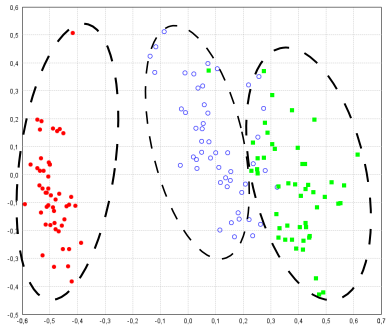
3.4.2 Évaluation des résultats

3.4.2.1 Évaluation visuelle

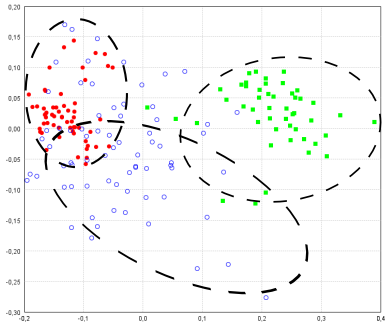
Pour mieux visualiser les changements qui ont affecté les données après application de notre méthode de mise à l'échelle SDA, nous avons utilisé la méthode de positionnement multidimensionnel MDS (Multidimensional scaling) (Hout *et al.*, 2013; Zhu *et al.*, 2016). Cette méthode est basée sur une matrice de dissimilarité (matrice de distance),



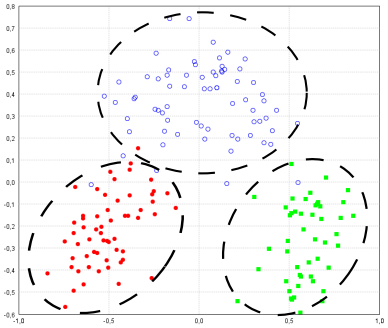
(a) MDS (Ensemble de données Iris original)



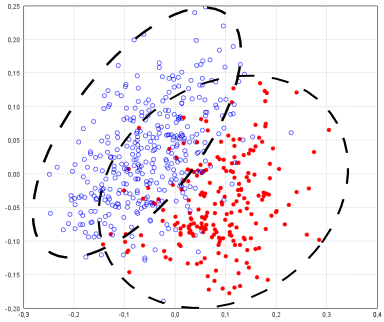
(b) MDS (Ensemble de données Iris mis à l'échelle)



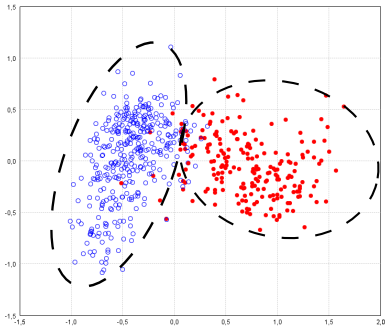
(c) MDS (Ensemble de données Wine original)



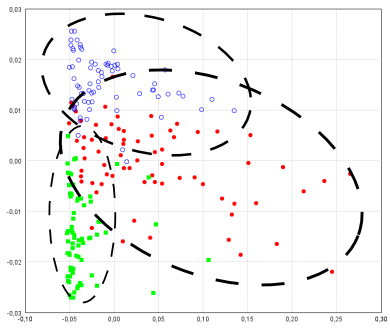
(d) MDS (Ensemble de données Wine mis à l'échelle)



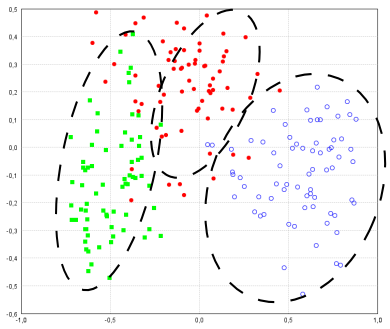
(e) MDS (Ensemble de données WDBC original)



(f) MDS (Ensemble de données WDBC mis à l'échelle)



(g) MDS (Ensemble de données Seeds original)



(h) MDS (Ensemble de données Seeds mis à l'échelle)

FIGURE 3.2 – Comparaisons visuelles démontrant l'intérêt de l'approche proposée SDA

utilisée pour quantifier le degrés de similarité entre les échantillons de données. Vu la grande complexité de la représentation visuelle des objets d'un ensemble de données multidimensionnelle (surtout ayant une dimension supérieure à trois), l'objectif de MDS est la réduction de cette grande dimensionnalité permettant ainsi une meilleure analyse visuelle. La figure 3.2 illustre une comparaison entre la distribution des classes des données d'origines et celle des données mises à l'échelle par notre approche dans les quatre ensembles de données utilisés. Le premier ensemble de données, Iris, présente un réel problème de chevauchement entre deux de ses classes. À partir de la sous-figure 3.2b, nous pouvons facilement distinguer les trois séparations des classes au lieu des deux séparations visuelles relevées sur la sous-figure 3.2a qui montre la distribution des données d'origines.
Sur les sous-figures 3.2c et 3.2g, qui représentent respectivement les ensembles de données Wine et Seeds, nous pouvons identifier un autre problème en plus des données chevauchées des clusters. Cela concerne aussi la densité de données qui varie soit dans le même cluster soit d'un cluster à l'autre. Notre méthode a pu résoudre ce problème. Comme indiqué sur les figures 3.2d et 3.2g, les densités variables des clusters se sont transformées en des densités uniformes et ne sont presque plus distinctes les unes des autres.
En ce qui concerne l'ensemble de données WDBC, la figure 3.2e représente les clusters originaux superposés, tandis que la figure 3.2f illustre les données traitées par notre méthode qui a permis une meilleur séparation et corrélation des clusters en fonction de leurs densités.

3.4.2.2 Évaluation quantitative

Les performances de notre approche SDA sont évaluées en comparant les résultats obtenus lors de l'application de l'algorithme DENCLUE et de ses variantes sur les données d'origines et celles mises à l'échelle par notre approche.

Tableau 3.2 – Évaluation de l'impact de l'approche SDA sur les données clusterisées en se basant sur les mesures internes.

Mesures	Algorithmes	Iris		Wine		WDBC		Seeds	
		Originale	proposée	Originale	proposée	Originale	proposée	Originale	proposée
DI (à maximiser)	DENCLUE1.0	0.471	**0.510**	**0.489**	0.483	**0.547**	0.541	0.538	**0.544**
	DENCLUE2.0	0.471	**0.562**	0.532	**0.573**	0.498	**0.619**	**0.794**	0.565
	DENCLUE-SA	0.471	**0.505**	**0.491**	0.488	**0.617**	0.537	0.529	**0.664**
	DENCLUE-GA	0.471	**0.672**	0.485	**0.496**	0.581	**0.633**	**0.554**	0.540
	DENCLUE-IM	0.471	**0.603**	**0.681**	0.659	**0.658**	0.638	0.753	**0.775**
DBI (à minimiser)	DENCLUE1.0	**0.597**	0.923	**1.324**	1.340	**1.480**	2.301	1.646	**1.216**
	DENCLUE2.0	**0.597**	1.914	1.439	**1.361**	**1.056**	2.595	1.949	**1.753**
	DENCLUE-SA	**0.597**	0.791	**1.264**	2.069	**1.742**	2.573	1.964	**1.345**
	DENCLUE-GA	**0.597**	1.276	1.383	**1.355**	2.589	**2.512**	1.916	**1.744**
	DENCLUE-IM	**0.597**	0.905	1.855	**1.484**	**2.311**	2.857	0.976	**0.901**
CP (à minimiser)	DENCLUE1.0	1.003	**0.815**	2.451	**2.450**	4.067	**3.989**	**1.275**	1.281
	DENCLUE2.0	1.003	**0.908**	2.226	**2.089**	3.798	**3.219**	**0.762**	1.550
	DENCLUE-SA	1.003	**0.583**	**2.342**	2.513	**3.801**	3.819	1.103	**0.974**
	DENCLUE-GA	1.003	**0.708**	2.509	**2.449**	3.863	**3.396**	1.421	**1.402**
	DENCLUE-IM	1.003	**0.590**	1.588	**1.470**	3.399	**3.198**	**0.659**	0.727

Le tableau 3.2 répertorie les mesures d'évaluation internes obtenues en appliquant DENCLUE 1.0 et ses variantes aux données originales et celles mises à l'échelle avec notre approche SDA.
Concernant la mesures DI, tous les algorithmes appliqués aux données mises à l'échelle de l'ensemble Iris ont eu des valeurs supérieures à ceux appliqués aux données d'origines. Dans les nouvelles données mises à l'échelle des ensembles Wine et WDBC, DENCLUE 2.0 et DENCLUE-GA ont obtenu les meilleures valeurs de l'indice DI, tandis que DENCLUE 1.0, DENCLUE-SA et DENCLUE-IM ont obtenu des valeurs inférieures de 0.006, 0.003 et 0.022, respectivement dans l'ensemble Wine et de 0.006, 0.08 et 0.02, respectivement dans l'ensemble WDBC. En ce qui concerne l'ensemble de données Seeds, trois algorithmes (DENCLUE 1.0, DENCLUE-SA et DENCLUE-IM) sur cinq ont obtenu les meilleures valeurs de l'indice DI résultant des données mises à l'échelle. DENCLUE 2.0 et DENCLUE-GA, quant à eux, ont eu des valeurs de l'indice DI plus faibles que ceux obtenu par les données originales de 0.229 et 0.014, respectivement.
En se basant sur les valeurs obtenues par l'indice DBI, les données d'origines de l'ensemble Iris ont obtenu les meilleures valeurs résultantes de tous les algorithmes. Dans l'ensemble Wine, les données mises à l'échelle surpassent celles d'origines dans trois algorithmes, à savoir DENCLUE 1.0, DENCLUE-SA et DENCLUE-IM. Pour DENCLUE 2.0 et DENCLUE-GA, les données d'origines ont obtenu les meilleures valeurs, inférieures à celles mises à l'échelle de 0.016 et 0.805, respectivement. Dans l'ensemble WDBC, seul DENCLUE-GA a obtenu une valeur de DBI meilleure en termes de données mises à l'échelle, tandis que dans l'ensemble Seeds, tous les algorithmes, appliqués aux données mises à l'échelle, ont obtenu les valeurs optimums.
Pour la dernière mesure interne, qui est représentée par l'indice CP, tous les algorithmes appliqués aux données mises à l'échelle ont eu les meilleures valeurs dans l'ensemble Iris. Pour les données mises à l'échelle des ensembles Wine et WDBC, tous les algorithmes ont obtenu les meilleures valeurs sauf un. Cependant que DENCLUE-SA a obtenu des valeurs plus grandes que celles eu par les données d'origines de 0.171 et 0.018 appliqué aux ensembles Wine et WDBC, respectivement. Dans le dernier ensemble de données, Seeds, les données mises à l'échelle ont obtenu les meilleures valeurs résultantes des algorithmes DENCLUE-SA et DENCLUE-GA. Quant aux algorithmes DENCLUE 1.0, DENCLUE 2.0 et DENCLUE-IM, ils ont eu des valeurs de l'indice CP plus grandes que celles obtenues par les données originales de 0.006, 0.788 et 0.068, respectivement.

Le tableau 3.3 répertorie les mesures d'évaluation externes obtenues par DENCLUE 1.0, DENCLUE 2.0, DENCLUE-SA, DENCLUE-GA et DENCLUE-IM, appliqués à des ensembles de données originaux et mis à l'échelle par notre approche SDA.
En ce basant sur l'indice CA, les cinq algorithmes ont obtenu les valeurs les plus élevées en les appliquant aux données mises à l'échelle dans presque tous les ensembles de données, à l'exception de l'ensemble Wine, où DENCLUE 2.0 a obtenu une valeur de l'approche inférieure à celle obtenue par les données d'origines par 0.016.
En termes d'entropie, les données mises à l'échelle d'Iris et WDBC ont obtenu les meilleures

Tableau 3.3 – Évaluation de l'impact de l'approche SDA sur les données clusterisées en se basant sur les mesures externes.

Mesures	Algorithmes	Iris		Wine		WDBC		Seeds	
		Originale	proposée	Originale	proposée	Originale	proposée	Originale	proposée
CA (à maximiser)	DENCLUE1.0	0.666	**0.934**	0.953	**0.959**	0.703	**0.905**	0.920	**0.921**
	DENCLUE2.0	0.666	**0.834**	**0.927**	0.911	0.715	**0.926**	0.706	**0.789**
	DENCLUE-SA	0.666	**0.906**	0.936	**0.953**	0.701	**0.935**	**0.915**	**0.915**
	DENCLUE-GA	0.666	**0.850**	0.914	**0.923**	0.716	**0.903**	0.814	**0.915**
	DENCLUE-IM	0.666	**0.942**	0.695	**0.704**	0.723	**0.766**	0.750	**0.756**
Entropy (à minimiser)	DENCLUE1.0	0.666	**0.232**	0.234	**0.201**	0.778	**0.393**	**0.397**	0.405
	DENCLUE2.0	0.666	**0.548**	**0.337**	0.359	0.738	**0.291**	**0.539**	0.641
	DENCLUE-SA	0.666	**0.259**	0.213	**0.190**	0.772	**0.270**	**0.285**	0.349
	DENCLUE-GA	0.666	**0.426**	0.343	**0.323**	0.738	**0.366**	0.576	**0.429**
	DENCLUE-IM	0.666	**0.217**	0.384	**0.312**	0.647	**0.381**	0.620	**0.594**
NMI (à maximiser)	DENCLUE1.0	0.733	**0.780**	0.819	**0.841**	0.121	**0.407**	**0.685**	0.681
	DENCLUE2.0	**0.733**	0.546	**0.691**	0.681	0.098	**0.372**	0.270	**0.437**
	DENCLUE-SA	**0.733**	0.681	0.582	**0.615**	0.080	**0.378**	0.489	**0.598**
	DENCLUE-GA	**0.733**	0.594	**0.767**	0.756	0.099	**0.412**	0.543	**0.660**
	DENCLUE-IM	**0.733**	0.720	0.193	**0.195**	0.042	**0.048**	0.324	**0.325**

valeurs par tous les algorithmes. Alors que dans le deuxième ensemble, seul DENCLUE 2.0 a obtenu une valeur médiocre des données mises à l'échelle, supérieure de 0.022 à celle des données d'origines. Ce qui est de l'ensemble Seeds, deux algorithmes (DENCLUE-GA et DENCLUE-IM) sur cinq, appliqués sur les données de notre approche, ont obtenu les meilleures entropies.
En ce qui concerne la mesure NMI, DENCLUE 1.0, appliqué aux données mises à l'échelle de l'ensemble Iris, a obtenu la meilleure valeur. Quant aux quatre autres algorithmes, appliqués aux données d'origines, ils ont obtenu chacun une valeur inférieure de 0.187, 0.052, 0.139, 0.013 dans DENLCUE 2.0, DENCLUE-SA, DENCLUE-GA et DENCLUE-IM, respectivement. Pour les données mises à l'échelle de l'ensemble Wine, trois algorithmes (DENCLUE 1.0, DENCLUE-SA et DENCLUE-GA) ont eu les meilleures valeurs. Tandis que DENCLUE 2.0 et DENCLUE-GA ont obtenu des valeurs inférieures de 0.1 et 0.011 appliqués aux données d'origines. Dans l'ensemble WDBC, les cinq algorithmes appliqués aux données mises à l'échelle ont eu les meilleures valeurs. Enfin, pour les données mises à l'échelle de l'ensemble Seeds, DENCLUE 2.0, DENCLUE-SA, DENCLUE-GA et DENCLUE-IM ont eu les meilleurs résultats. Seul DENCLUE 1.0 a obtenu une valeur inférieure de 0.004 à la valeur NMI des données d'origines.

Pour une description synthétique des résultats de clustering, les figures 3.3 et 3.4 illustrent le nombre d'ensembles de données ayant atteints les meilleures valeurs de mesures internes et externes pour chaque algorithme appliqué aux données originales et celles mises à l'échelle par notre méthode.

En se basant sur la figure 3.3 qui représente les mesures d'évaluation internes, DENCLUE-IM a obtenu les meilleures valeurs de l'indice DI dans deux ensembles de données en l'appliquant aux données mises à l'échelle par notre approche. En outre, DENCLUE 2.0 et DENCLUE-GA ont eu les meilleures valeurs dans trois ensembles de données. En

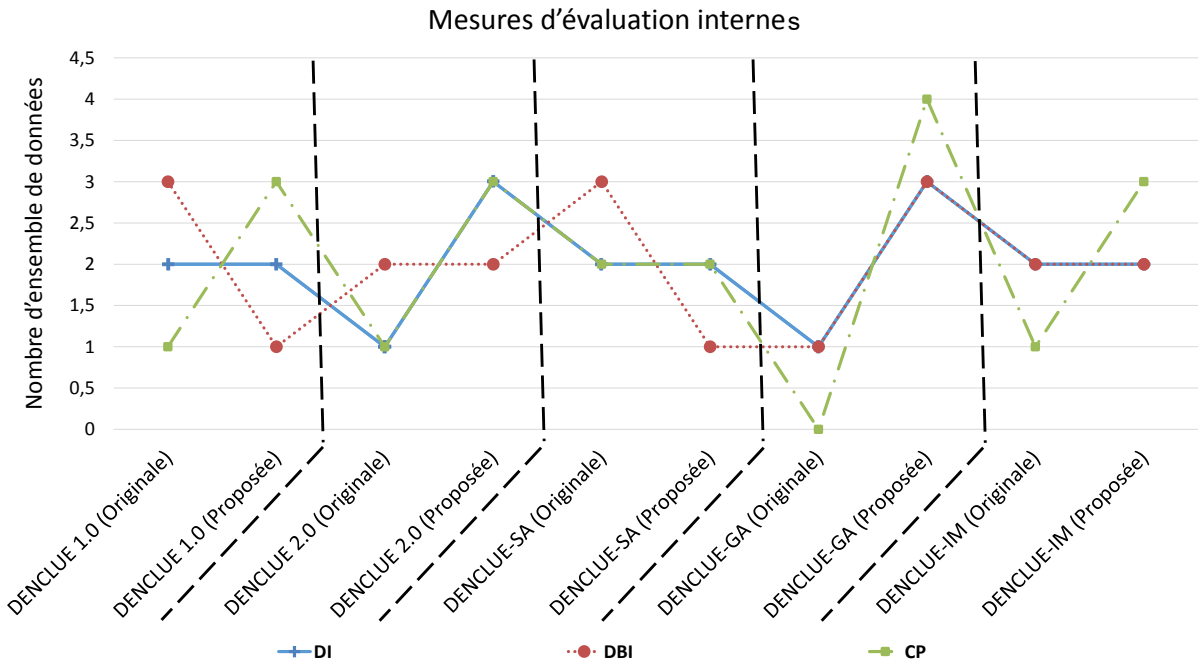


FIGURE 3.3 – Nombre de mesures externes satisfaites par les algorithmes appliqués aux données originales et celles mises à l'échelle par notre approche

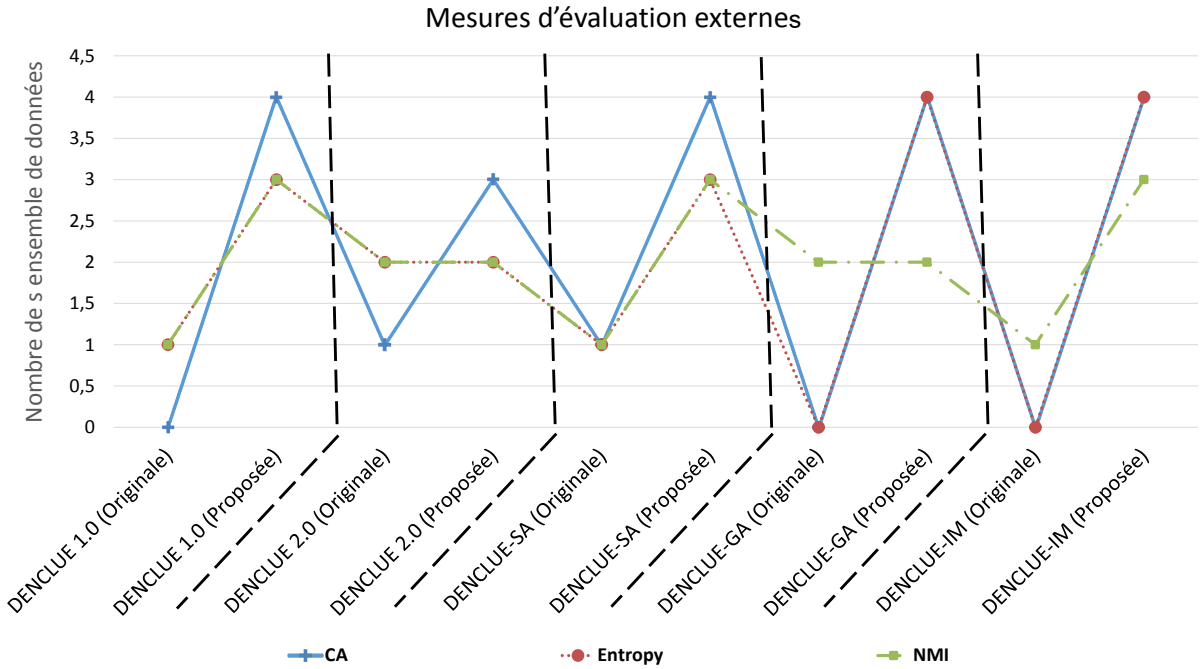


FIGURE 3.4 – Nombre de mesures internes satisfaites par les algorithmes appliqués aux données originales et celles mises à l'échelle par notre approche

ce qui concerne la mesure DBI, les meilleures valeurs ont été obtenues dans trois ensembles de données mises à l'échelle résultantes de l'algorithme DENCLUE-GA, tandis que DENCLUE 2.0 et DENCLUE-IM se sont dépassés dans deux ensembles de données et DENCLUE 1.0 et DENCLUE-SA dans un seul. Autrement dit, dans DENCLUE 1.0 et DENCLUE-SA, les données d'origines surpassent celles mises à l'échelle dans trois ensembles de données sur quatre. Pour la mesure CP, DENCLUE-GA, appliqué aux données mises à l'échelle, a obtenu les meilleures valeurs dans tous les ensembles de données, suivis de DENCLUE 1.0, DENCLUE 2.0 et DENCLUE-IM, puis DENCLUE 2.0, qui a prouvé sa performance dans deux ensembles parmi quatre.

En termes de mesures d'évaluation externes, comme montré sur la figure 3.4, les algorithmes DENCLUE 1.0, DENCLUE-SA, DENCLUE-GA et DENCLUE-IM, appliqués aux données mises à l'échelle, ont obtenu les meilleures valeurs de la mesure CA dans les quatre ensembles de données. Alors que DENCLUE 2.0, appliqué aux données mises à l'échelle, a excellé dans trois ensembles de données sur quatre. Concernant les valeurs de l'entropie des données mises à l'échelle, DENCLUE-GA et DENCLUE-IM ont excellé dans tous les ensembles, suivis de DENCLUE 1.0 et DENCLUE-SA qui ont obtenu les meilleures valeurs dans trois ensembles de données, puis DENCLUE 2.0 dans deux ensembles de données. Pour la troisième mesure, DENCLUE 1.0, DENCLUE-SA et DENCLUE-IM appliqués aux données mises à l'échelle se surpassent dans trois ensembles de données suivis de DENCLUE 2.0 et DENCLUE-GA qui ont obtenu les meilleures valeurs dans deux ensembles de données.
Les résultats discutés témoignent de l'efficacité de notre approche afin d'améliorer les performances des algorithmes du clustering appliqués à différents ensembles de données. De ce qui précède on peut remarquer qu'en termes de mesures d'évaluation internes, les algorithmes appliqués aux données d'origines ont eu dans certains cas les meilleures valeurs que ceux des données mises à l'échelle. Par exemple, DENCLUE 1.0 et DENCLUE-SA, appliqués aux données originales, ont obtenus les meilleures valeurs de l'indice DBI dans trois ensembles de données. Cette performance peut s'expliquer par le fait que les mesures internes sont basées sur des calculs de distances. Cela signifie que dans les données originales, les distances sont proches les unes des autres car les données se chevauchent entre eux plus que les données mises à l'échelle par notre approche.

3.5 Conclusion

Dans ce chapitre, nous avons proposé une méthode de mise à l'échelle des données chevauchées en utilisant une translation qui suit la distribution de densités des objets. Cette méthode nommé SDA a été évaluée par deux types de comparaisons, à savoir, visuelle et quantitative. En se basant sur les comparaisons visuelles, il a été constaté que notre approche permet de mieux séparer les clusters chevauchés et de traiter le problème de distributions des données de densités différentes. Pour valider les comparaisons quantitatives en fonction des mesures d'évaluation internes et externes, nous avons appliqué des algo-

rithmes de clustering basés sur la densité aux données mises à l'échelle par notre méthode. Les résultats obtenus indiquent que notre approche permet d'améliorer les performances du clustering ainsi que la qualité de séparation des clusters résultants.

Les améliorations proposées dans la première partie de notre thèse ont été exploitées dans la deuxième partie, en les adaptant à des applications bien précises, à savoir : la recherche et la sélection des services dans le Cloud Computing, la détection de sentiments dans le réseau social Twitter et la prévention du cancer du nasopharynx.

Deuxième partie

Améliorations applicatives

Chapitre

4 Contribution à la recherche et sélection des services Cloud idéaux à l'aide des techniques de clustering

Sommaire

4.1 Introduction

La sélection des services Cloud est devenu un besoin pressant suite à l'utilisation accrue du Cloud Computing. Les utilisateurs du Cloud commencent à vouloir trouver les services les mieux adaptés à leurs besoins parmi un grand nombre de services Cloud disponibles, ce qui nécessite un traitement spécial. Cela consiste principalement à comparer les services disponibles, plus particulièrement en termes de qualité et de coût, tâche qui est devenue difficile et qui a pour cause le grand nombre de services dans le Cloud.
Généralement, cela se fait à l'aide de diverses techniques, telles que la similarité, l'exploration de données, les systèmes de recommandation, les méthodes d'analyse décisionnelle multicritère (MCDA pour le terme anglais : Multi Criteria Decision Aiding), etc.
Le principal problème de ces techniques est, d'une part, le grand nombre de services retournés et, d'autre part, l'absence de communication directe entre l'utilisateur et le système. La plupart des méthodes sont basées sur les connaissances précédentes du système ou reposent sur les commentaires d'autres utilisateurs. Afin d'éviter ces problèmes, nous nous sommes basés, au premier lieu, sur un travail (Abourezq et Idrissi, 2014b) utilisant une forme de communication entre l'utilisateur et le système, ainsi notre contribution a pour fin de demander à l'utilisateur de saisir les valeurs de son service idéal, puis, selon les techniques de clustering, les services les plus proches du service idéal sont sélectionnés et renvoyés à l'utilisateur final.

Par le présent chapitre, nous survolons les différents travaux de recherches et de sélection de service Cloud proposés dans la littérature, tout en décrivant le principe de

notre approche et démontrant son efficacité en la comparant avec d'autres méthodes de recherche et de sélection.

4.2 Recherche et sélection des services Cloud : état de l'art

Le domaine de recherche et sélection des services Cloud a fait l'objet de nombreuses contributions au cours de la dernière décennie.

Dans le cadre de la sélection des services Cloud, Han *et al.* (2009) ont présenté un framework de sélection basé sur un système de recommandation (RS). Cette approche aide l'utilisateur à sélectionner les meilleurs services auprès de différents fournisseurs Cloud en se basant sur la qualité de service (QoS : Quality Of services) et des commentaires des utilisateurs. Dans le même contexte, Zain *et al.* (2014) ont développé une méthode de sélection permettant aux utilisateurs de spécifier leur perception des critères de qualité. Leur approche se base sur la technique de clustering K-means et sur les commentaires des utilisateurs qui sont transmis à un référentiel. Les clusters générés en fonction des commentaires se divisent en trois : services médiocres, bons et excellents. Dans (Soltani *et al.*, 2014), les auteurs ont développé un nouveau system de recommandation basé sur le concept de raisonnement par cas. Ce système, appelé QuARAMRecommender, a pour but de sélectionner le fournisseur du cloud et le type de machines virtuelles qui répond le mieux aux besoins de l'utilisateur. Les auteurs de (Abourezq et Idrissi, 2014b,c, 2015b; Idrissi et Abourezq, 2014), ont également présenté un système de recherche et de sélection de services Cloud (CSRSS) basé sur la technique Skyline et une méthode de surclassement pour répondre le mieux aux besoins des utilisateurs. Leurs méthodes ont montré des premiers résultats prometteurs. Dans le même contexte, Karim *et al.* (2016) ont construit un modèle qui exploite l'historique des valeurs QoS et les informations d'utilisateurs afin de prédire les valeurs QoS des services composites. La principale contribution de ce travail consiste à utiliser un système de surveillance et de collecte de services Cloud réels basé sur le QoS afin de mesurer les similitudes entre les services cloud composants (services atomiques faisant partie d'un service composite) et la qualité de service composite. Un nouveau framework décisionnel flou a été présenté dans (Sun *et al.*, 2016) afin d'aider les utilisateurs de services ordinaires à sélectionner les bons services Cloud. Les auteurs de ce system, appelé Cloud-FuSeR, expriment par flou les informations imprécises ou confuses données par les utilisateurs.

Tous les travaux cités ci-dessus ne demandent pas directement à l'utilisateur d'exprimer les besoins de son service idéal. La plupart des travaux sont simplement basés sur des connaissances antérieures, ainsi que sur des commentaires exprimés par des utilisateurs. Ainsi, les services sélectionnés peuvent être générés à partir d'un besoin collectif plutôt qu'individuel. Et si l'utilisateur avait juste le choix entre une dizaine de services qui se rapproche le plus à son service idéal. Il faut noter que le service idéal est un service dont l'utilisateur choisit les valeurs de ses critères. L'une des méthodes pouvant répondre à ce besoin est l'utilisation des techniques d'exploration de données telles que les méthodes de

clustering.

4.2.1 L'opérateur Skyline

L'opérateur Skyline est l'une des techniques qui traitent le problème du vecteur maximal (Kung *et al.*, 1975). Skyline permet la récupération des points pertinents dans un ensemble de données. Les points pertinents représentent les points qui optimisent certaines exigences tout en ayant de bonnes valeurs dans les autres critères. Ces points récupérés ne sont dominés par aucun autre point.
L'opérateur Skyline a pour but de trancher dans des problèmes ayant des objectifs complémentaires. L'exemple classique (Börzsönyi *et al.*, 2001) est celui de trouver un hôtel pas cher et proche de la plage. Ces deux objectifs sont complémentaires, car plus un hôtel est proche de la plage, plus son prix est élevé. La décision finale appartient aux utilisateurs : bien que l'utilisateur A puisse choisir un hôtel plus proche de la plage à un prix plus élevé, l'utilisateur B choisirait un hôtel bon marché et plus éloigné de la plage. La meilleure approche consiste à présenter aux utilisateurs tous les hôtels intéressants, à savoir les hôtels qui ne sont pas dégradé que tout autre hôtel en termes de distance et de prix. On dit alors que ces hôtels forment le Skyline. Le terme Skyline a été inventé en raison de la représentation graphique des points qui le composent et qui fait illusion à la ligne d'horizon (skyline en anglais), comme le montre la figure 4.1.
Plus de détails sur l'opérateur Skyline sont présentés dans l'annexe de ce présent mémoire.

4.2.2 ELECTRE IS

ELECTRE IS est considéré comme une généralisation d'ELECTRE I car il intègre l'utilisation de pseudo-critères (Nafi et Werey, 2009) au lieu de critères réels et introduit l'utilisation du seuil de veto v_{cr}, du seuil de préférence p_{cr} et du seuil d'indifférence q_{cr}. De plus, les conditions de concordance et de discordance changent et cette dernière est appelée la condition de non veto.
Un pseudo-critère est une fonction g_{cr} associée à deux fonctions de seuil : le seuil d'indifférence q_{cr} et le seuil de préférence p_{cr}.
Le seuil de préférence p_{cr} est une fonction à valeur réelle telle que, pour deux alternatives a et b dans Alt, $p_{cr}(g_{cr}(a))$ est la valeur minimale positive d'une différence de score de type $g_{cr}(a) - g_{cr}(b)$ cela pourrait être compatible avec la préférence de a sur b.
Le seuil d'indifférence q_{cr} est une fonction à valeurs réelles tel que, pour deux alternatives a et b dans Alt, $q_{cr}(g_{cr}(a))$ est la valeur maximale d'une différence de score de type $g_{cr}(b) - g_{cr}(a)$ qui pourrait être compatible avec l'indifférence entre a et b.
Le seuil de veto vcr est attribué au critère cr et permet de définir la valeur au-delà de laquelle la discordance relative à l'affirmation $a\ S\ b$ ne peut permettre la validation du surclassement. En d'autres termes, pour valider l'affirmation $a\ S\ b$, aucun critère parmi les critères discordants ne devrait mettre son veto.
Plus de détails sont proposés dans l'annexe de ce mémoire.

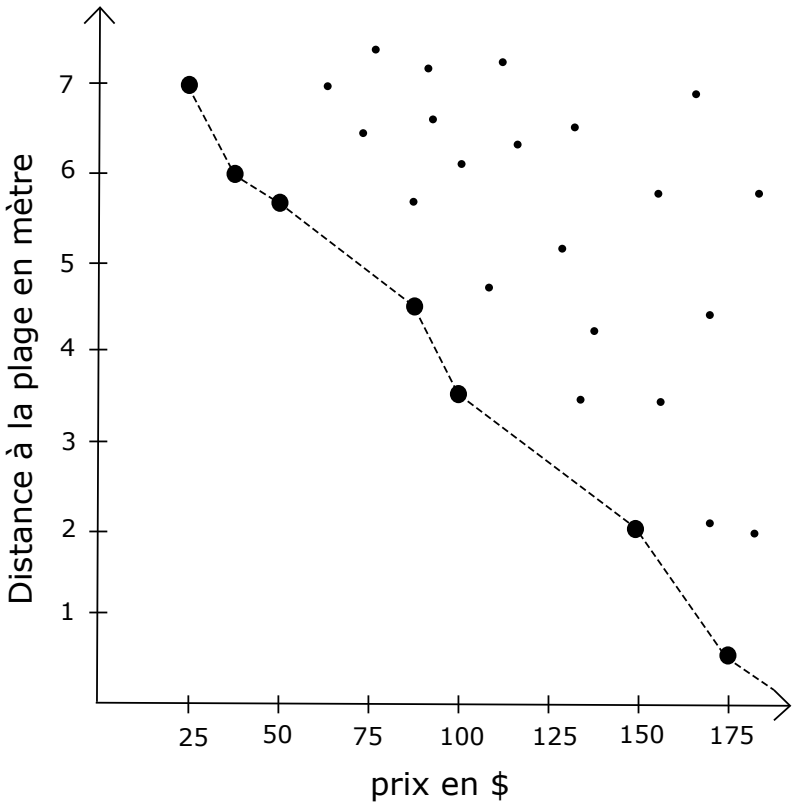


FIGURE 4.1 – Exemple de la représentation graphique du Skyline(Börzsönyi *et al.*, 2001)

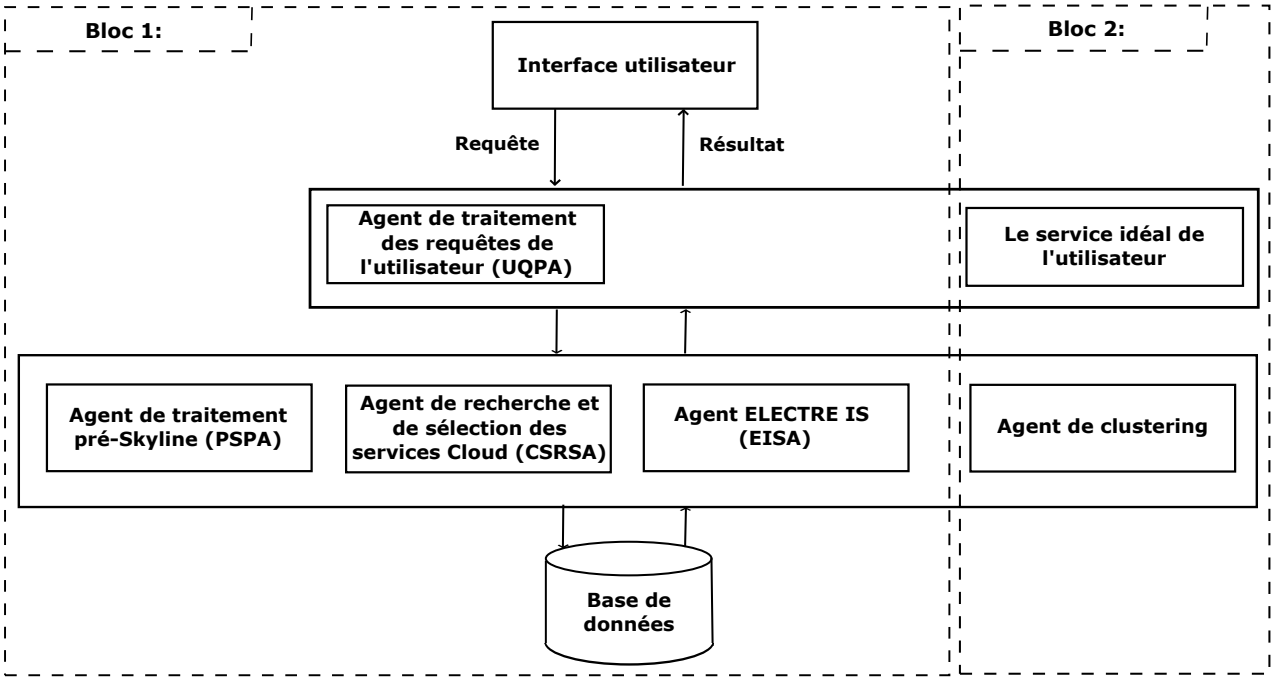
4.3 Méthode proposée

L'une des principales problématiques soulevées par l'utilisation du Cloud Computing est celle de trouver un service Cloud adapté aux critères des utilisateurs finaux. Notre approche s'est basée principalement sur le travail présenté dans (Idrissi et Abourezq, 2014). Dans ce travail un système de recherche et de sélection de services Cloud (CSRSS) a été développé en se basant sur Skyline et une des méthodes de surclassement qui est ELECTRE IS, afin de sélectionner les services Cloud qui répondent le mieux aux besoins des utilisateurs.

Comme présenté dans (Idrissi et Abourezq, 2014), le CSRSS a donné des résultats prometteurs. Il a permis de sélectionner 2528 services Cloud à partir de 50000 entrées. Ce nombre de services retournés est considéré comme important pour un utilisateur qui veut trouver ses services préférés. Mais pour plus d'affinement des résultats finaux, nous avons fais recours aux méthodes de classification non supervisées.

Comme expliqué précédemment, notre approche consiste à déléguer à l'utilisateur le choix des valeurs de son propre service idéal. Ce service ne figurera probablement pas dans les données, mais le système tentera de trouver les services les plus proches qui satisferont presque toutes les exigences de l'utilisateur. Ainsi, comme le montre la figure 4.2, les prototypes ont été divisés en deux blocs. Dans le premier bloc, nous exposons le système

FIGURE 4.2 – Le nouveau prototype de recherche et de sélection de services cloud



précédent tel que présenté dans (Idrissi et Abourezq, 2014), et dans le deuxième bloc, nous présentons les nouvelles étapes ajoutées à ce système :

- Premièrement, l'utilisateur se connecte au système via son interface et sélectionne les critères auxquels les services Cloud doivent répondre. L'utilisateur a également la possibilité de choisir les valeurs de ces critères, afin de construire son service idéal préféré ;
- Ensuite, l'agent chargé du traitement des requêtes de l'utilisateur (UQPA) extrait les exigences contenues dans la requête et les définit en deux types de critères : fixes (tels que le nom du fournisseur, le modèle de service, le système d'exploitation, ...) et variables (prix , bande passante, espace de stockage, ...) ; C'est dans cette étape que la requête basée sur le service idéal de l'utilisateur est construite ;
- Par la suite, l'agent de recherche et de sélection de services Cloud (CSRSA) se connecte à la base de données et exécute une requête SQL, dont les prédicats sont les exigences fixes spécifiées par l'utilisateur ;
- Après, l'agent de traitement pré-Skyline (PSPA) prépare les résultats extraits de la base de données par le CSRSA afin que l'opérateur Skyline soit exécuté. Les services Cloud retournés et leurs dimensions sont stockés sous forme de tuples. Les dimensions utilisées reflètent les critères variables que l'utilisateur avait choisi ;
- Ensuite, le CSRSA utilise l'algorithme Skyline sur l'ensemble des tuples renvoyés par le PSPA pour déterminer les services Cloud qui figurent dans le Skyline et qui répondent aux préférences de l'utilisateur.
- Après cela, l'agent ELECTRE IS (EISA) applique la méthode ELECTRE IS sur

l'ensemble des tuples renvoyés par le CSRSA afin de déterminer les services Cloud qui ne sont pas dominés par d'autres. Ces services sont donc considérés comme ceux offrant le meilleur compromis pour tous les critères définis par l'utilisateur ;

— Enfin, une fois que la requête de l'utilisateur est passée par toute les étapes décrites précédemment, l'agent de Clsutering, l'étape ajoutée pour définir notre approche, intervient pour affiner davantage les résultats renvoyés en se basant sur la requête du service Cloud idéal défini par l'utilisateur.

L'agent de clustering proposé est décrit dans l'algorithme 4.1. Il est composé de trois parties :

La première partie calcule les services Skyline en comparant les tuples et en déterminant ceux qui ne sont dominés par aucun autre. A chaque itération, un nouveau tuple est lu dans la liste d'entrée de tuples (L_P). Si le nouveau tuple est dominé par l'un des tuples existants dans la liste Skyline (L_S), il est éliminé. S'il domine un tuple dans L_S, le tuple dominé est éliminé et le nouveau tuple est ajouté à L_S pour être comparé aux futurs tuples. Si le nouveau tuple est incomparable, ce qui signifie qu'il n'est dominé par aucun autre tuple dans L_S, il est ajouté à L_S. À la fin de toutes les itérations, seuls les tuples qui ne sont dominés par aucun autre tuple sont conservés dans L_S et forment le Skyline.

La deuxième partie de l'algorithme consiste à appliquer la méthode ELECTRE IS à la liste Skyline L_S. Étant donné que Skyline considère que tous les critères ont la même importance, il ne permet pas d'arbitrer entre ces incomparables tuples. Pour surmonter cette limitation, les auteurs de (Abourezq et Idrissi, 2014b,c) ont pensé à utiliser une méthode de surclassement, ELECTRE IS, qui permet d'effectuer une comparaison par paire de tous les tuples contenus dans la liste Skyline et élimine ceux qui sont dominés. Ainsi, seuls les tuples qui répondent le mieux aux critères des utilisateurs, y compris leurs préférences, sont renvoyés.

Dans la troisième et dernière parties, notre approche, consistant à ajouter l'agent de clustering, est appliquée sur le résultat obtenu par l'agent ELECTRE IS, c'est-à-dire la liste L_{ES}, tout en ajoutant à cette liste le service idéal représenté par id. Le résultat renvoyé par l'agent de clustering représente le cluster idéal, c'est-à-dire le cluster qui comporte le service idéal choisi précédemment par l'utilisateur . L'algorithme 4.1, montrant les détails de notre approche, est basé sur les paramètres suivants :

p, q : tuples

c' : seuil de concordance

L_P : liste d'entrée des tuples sur lesquels on applique le Skyline

L_S : liste des tuples formant le Skyline

L_C : liste de critères en entrée avec leurs informations

L_{ES} : liste de sortie des tuples constituant la solution

id : le tuple contenant les valeurs idéales, appelé également service idéal.

L_{CS} : liste en sortie du cluster idéal constituant la solution.

Algorithme 4.1 L'algorithme IdealELECTREIsSkyline

```
1: Procédure CalculSolution (p, q, id, c', L_P, L_S, L_C)
2: pour chaque p ∈ L_P faire
3:     si (L_S = ∅)
4:     {
5:         L_S = {p}
6:      }
7:     sinon {
8:         pour chaque q ∈ L_S − {p} faire
9:             res = Comparer (p, q, L_C)
10:            si (res = compter (L_C))
11:            {
12:                L_S = L_S + {p} − {q}
13:                }
14:            sinon {
15:                si (res ≠ 0 et q est le dernier dans L_S)
16:                {
17:                    L_S = L_S + {p}
18:                      }
19:                sinon {
20:                    Goto (ligne 18)
21:                }
22:            }
23:        fin pour
24:    }
25: fin pour
26: L_ES = L_S
27: pour chaque p ∈ L_S faire
28:     pour chaque q ∈ L_S − {p} faire
29:         concordIndex = Concordance(p, q, L_C)
30:         vetoIndex = Veto(p, q, L_C)
31:         si (concordIndex ≥c' and vetoIndex = true)
32:         {
33:             L_ES = L_ES − {q}
34:         }
35:     fin pour
36: fin pour
37: L_ES = L_ES + {id}
38: L_CS= Clustering(L_ES)
39: Retourner L_CS
40: Fin procédure
```

4.4 Résultats expérimentaux

4.4.1 Description de données

Comme mentionné auparavant, pour affiner la sélection finale des services idéaux, nous avons proposé de combiner le CSRSS existant avec l'une des techniques d'exploration de données, à savoir le clustering. Pour cette raison, nous avons tout d'abord souligné l'efficacité de notre approche en comparant les résultats sélectionnés avant et après utilisation des techniques de clustering. Deuxièmement, nous avons comparé entre quatre algorithmes de clustering appliqués aux résultats sélectionnés en utilisant les trois mesures d'évaluation internes (vu que cet ensemble de données est non étiqueté comme montré dans la sous-section 1.7.1 du chapitre 1). Nos expériences ont porté sur 50000 services Cloud générés de manière aléatoire. Chaque service Cloud comporte 10 attributs, comme décrit dans le tableau 4.1.

Tableau 4.1 – La description des 10 attributs des services Cloud

Dimension	*Détail*	*Sens de comparaison*
Espace de stockage	—	**à maximiser**
Bande passante	—	**à maximiser**
Latence	—	**à minimiser**
Portabilité	**Nombre des systémes d'exploitation compatibles avec les services / Nombre des systémes d'exploitation requis par l'utilisateur**	**à maximiser**
Risque	**Nombre de certifications de gestion de risques obtenues par le fournisseur**	**à maximiser**
Perte de données	**Nombre d'incidents liés à la perte de données**	**à minimiser**
Coût d'acquisition	—	**à minimiser**
Coût permanent	—	**à minimiser**
Temps de réponse du service	**Temps de réponse moyen (ms) / Temps de réponse maximum défini dans le SLA (ms)**	**à minimiser**
Disponibilité	**Temps pendant lequel le service est indisponible (ms) / Durée totale d'utilisation (ms)**	**à minimiser**

4.4.2 Évaluation des résultats

Tous les résultats obtenus par les trois systèmes de sélection ont été enregistrés dans le tableau 4.2. Comme présenté dans ce tableau, les résultats de notre approche ne donnent que 0,02 % des 50000 de la liste des entrées, tandis que l'utilisation de Skyline en donne 14,72 % et la combinaison entre Skyline et ELECTRE IS en donne 5,05 %. Le nombre des services Cloud retournés varie entre 10 et 11 services les plus proches en termes de valeurs de la demande de l'utilisateur. Il faut noter que pour évaluer les performances de

notre approche, les algorithmes ont été implémentés dans un environnement JAVA, sur un PC Core i5 (2,70 GHz) avec 8 Go de mémoire.

Les résultats expérimentaux sont plus détaillés dans le tableau 4.3. Nous constatons que les meilleurs résultats sont ceux obtenus par EM, DENCLUE et DENCLUE-IM. Malgré la vitesse de calcul de l'algorithme K-means en k = 10, le nombre des services renvoyés est très grand. De plus, lors de la modification du paramètre k, non seulement le nombre des services reste assez grand, mais la durée d'exécution augmente également. Selon les mesures d'évaluation, l'algorithme EM obtient la meilleure valeur de l'indice DI. DENCLUE et DENCLUE-IM, quant à eux, obtiennent tous deux les meilleures valeurs des indices DBI et CP.
Après analyse des résultats, nous pouvons donc conclure que DENCLUE-IM est l'algorithme de clustering le plus approprié dans notre cas, en raison de son temps de réponse, de la qualité ses résultats de clustering et enfin en raison du nombre de services Cloud renvoyés.

Tableau 4.2 – Le nombre de services Cloud retournés en fonction des trois méthodes.

# Services Cloud	Algorithme Skyline	Algorithme ELECTREIsSkyline	Algorithme IdealELECTREIsSkyline
50000	7360	2528	11

Tableau 4.3 – Comparaison entre les quatre algorithmes de clustering en fonction des mesures d'évaluation.

Algorithmes / Mesures		DI	DBI	CP	Temps d'exécution (s)	Taille du cluster idéal
K-means	k=10	0.529	0.755	6.258	**0.336**	817
	k=50	0.612	0.928	13.460	2.826	314
	k=100	0.647	0.888	13.203	20.204	140
EM		**0.874**	2.775	7.338	440.265	**10**
DENCLUE		0.493	**0.340**	3.427	7.427	**11**
DENCLUE-IM		0.493	**0.340**	3.427	**0.997**	**11**

4.5 Conclusion

Dans le but de trouver une solution de sélection de services Cloud, nous avons développé une nouvelle approche, qui vise à combiner un système de recherche et de sélection de services Cloud existant avec les techniques de clustering. L'idée de notre approche est

d'épargner aux utilisateurs la recherche dans des milliers de services Cloud et de renvoyer un ensemble assez restreint de services adaptés le plus au service idéal choisi par l'utilisateur. Notre méthode nous a permis de passez de 50000 services sélectionnés par l'opérateur Skyline, à 2728 services sélectionnés par Skyline combiné à la technique ELECTRE IS, à enfin 11 services sélectionnés à l'aide du principe du service idéal et du clustering.
À cette fin, nous avons testé quatre algorithmes de clustering, appartenant à des familles différentes, afin de choisir la méthode la plus appropriée à notre cas. L'algorithme DENCLUE-IM a donné des résultats prometteurs. Il a permis de rendre 11 services pertinents en 0.997 secondes. Il a donc atteint un compromis entre la qualité du clustering, le temps de réponse et le nombre de services sélectionnés.

Chapitre

5 Contribution à l'analyse des sentiments dans Twitter

Sommaire

5.1 Introduction

De nos jours, les opinions exprimées sont devenues omniprésentes dans toutes les activités de vie. D'où la naissance de l'analyse des sentiments qui peut être appliquée dans presque tous les domaines des affaires et de société. Cette analyse de sentiments s'est révélée aussi très utile pour étudier les émotions, les expressions et les attitudes des utilisateurs des réseaux sociaux, en l'occurrence Twitter. Le réseau Twitter et l'un des réseaux sociaux considéré comme une riche source de signaux sentimentaux exprimés entre utilisateurs à l'intermédiaire des tweets. Malgré la taille réduite des tweets, ils peuvent contenir des significations riches qui servent à suivre l'humeur des internautes, leurs avis sur des produits, des services ou même des débats politiques. Ces avis peuvent être positifs, négatifs ou aussi neutres. Pour satisfaire cet objectif et depuis l'année 2009, les chercheurs ont commencé à s'intéresser à l'analyse des tweets (Go *et al.*, 2009) en utilisant diverses approches pour détecter la polarité des sentiments exprimés via ces tweets.
Le Machine Learning, en particulier la classification, et l'une des approches utilisées pour la détection de sentiments dans les tweets. Dans cette optique nous avons développé des approches basées sur le clustering (clasification non supervisée) en exploitant les deux algorithmes : DENCLUE et K-means.

Dans ce chapitre nous avons présenté un état de l'art pour pouvoir se situer par rapport à d'autres travaux. Nous avons par la suite décrit nos approches ainsi que le processus mené pour atteindre notre objectif. Les résultats expérimentaux sont enfin dressés pour mieux visualiser l'efficacité des algorithmes proposés, avant que les conclusions soit tirées.

5.2 Analyse des sentiments dans Twitter : État de l'art

Les sentiments classifiés des tweets permettent de découvrir les différents avis des utilisateurs de Twitter sur un produit ou un service. De nombreux travaux ont été publiés à cet effet concernant la classification supervisée, la classification semi-supervisée, ainsi que la classification non supervisée (clustering).

Dans l'un des travaux s'intéressant à la classification supervisée (Asghar *et al.*, 2018), un framework de classification des tweets a été unifié grâce à un système de classification hybride. Ce travail repose sur l'incorporation de quatre algorithmes de classification : un classificateur dédié à "l'argot", un autre dédié aux émoticônes, le classificateur SentiWord-Net et un autre dédié aux mots spécifique à un domaine. Dans un autre travail (Song *et al.*, 2017), deux méthodes ont été proposées, la première sert à pondérer des attributs et la seconde sert à extraire les vecteurs caractéristiques en se basant sur l'algorithme de classification Naïve Bayes. Ces deux approches visent à augmenter la précision des résultats issue de l'analyse des sentiments en adaptant le nombre d'attributs utilisés pour estimer le poids de chaque classe et en identifiant les mots significatifs permettant de prédire les classes.

Dans le domaine semi-supervisé, les auteurs de (Charalampakis *et al.*, 2016) ont introduit une approche détectant l'ironie dans les tweets concernant la politique grec. Leur travail est basé sur l'algorithme collective-tree. Le côté semi-supervisé de cette technique réside dans sa considération à la fois des données étiquetées et non étiquetées.

En ce qui concerne le clustering (classification non supervisée), les auteurs de (Pandey *et al.*, 2017) ont conçu une méthode considérée comme nouvelle version de l'algorithme cuckoo-search basé sur l'algorithme K-means. Leur méthode a été appliquée à l'analyse des sentiments dans Twitter. Dans (Riaz *et al.*, 2017), une nouvelle méthode d'analyse des sentiments a été réalisée afin de connaître les préférences des clients. Ensuite, l'algorithme K-means a été appliqué, en tant qu'algorithme de clustering, pour placer les mots en fonction de leur polarité dans les clusters adéquats. Dans (Mostafa, 2019), les auteurs ont mené une étude sur les tweets qui s'intéressent à la nourriture Halal. Ils ont utilisé l'algorithme de clustering PAM (pour le terme anglais : partitioning around medoids) pour grouper les avis des consommateurs sur ce type nourriture. Dans le même contexte (Kumari et Babu, 2016), les algorithmes K-means et DBSCAN ont été appliqués sur des données Twitter étiquetées afin d'évaluer la qualité du clustering des deux algorithmes. Les résultats obtenus ont démontré que DBSCAN a obtenu les meilleurs résultats. DBSCAN est aussi utilisé dans un autre travail (Stojanovski *et al.*, 2016) en le comparant avec d'autres algorithmes de clustering hiérarchiques, afin d'évaluer leur qualité de détection et d'identification de régions de New York dite chaudes (hotspot) à partir de données Twitter à l'aide de l'analyse de sentiments. Dans (Anumol Babu, 2016), l'analyse des sentiments a été faite à l'aide de la segmentation des tweets qui consiste à scinder les tweets en segments de signification sémantique. La segmentation est effectuée sur la base du score d'adhérence et le clustering des tweets est réalisé à l'aide de la méthode DBSCAN

avec le coefficient Jaccard comme mesure de similarité. D'autres algorithmes, à l'instar de l'algorithme expectation maximisation (EM), ont été utilisés dans le domaine d'analyse de sentiments dans Twitter. Un système explorant l'approche statistique de l'algorithme EM analysant les criques portées sur des produits via les tweets a été proposé (Shelke *et al.*, 2017).

Ce pendant contrairement aux techniques d'apprentissage supervisé ou semi-supervisé, les techniques basées sur le clustering produisent des résultats expérimentaux sans traitement d'étiquetage manuel, ni temps d'entrainement.

5.3 Schéma de l'approche proposée

L'approche adoptée est composée de trois étapes complémentaires : l'élimination des données bruitées en phase de pré-traitement (Riaz *et al.*, 2017), l'extraction des vecteurs caractéristiques et l'application des algorithmes de clustering proposés. Nous développerons ci-après ces trois concepts.

5.3.1 Pré-traitement

Les tweets sont autorisés à utiliser 140 caractères, par conséquent, cette limite pousse les internautes à utiliser de plus en plus des abréviations (acronymes), des expressions irrégulières, etc. Ce phénomène incrémente le niveau de données bruitées, affectant la reconnaissance du sentiment sur Twitter. Pour résoudre ce problème, différentes méthodes de pré-traitement sont largement utilisées avant d'extraire les fonctionnalités, d'abord pour éliminer le bruit et ensuite pour conserver la signification des tweets. Cette phase est traitée comme suit :

— Premièrement, afin d'omettre le bruit, les URL et les ponctuations incluant le "@" qui précède le nom des utilisateurs et le "#" qui précède les hash-tags sont supprimés (Pandey *et al.*, 2017). La séquence des espaces est également éliminée et remplacée par un espace unique. Concernant la suppression des ponctuations, il faut noter que les signaux exprimant les émoticônes sont conservés.

— Deuxièmement, pour donner plus de clarté au tweet avant d'être traité, tous les mots sont convertis en minuscules, puis comparés au dictionnaire des mots non significatifs (Stopwords, 2014) aussi appelés mots vides "a, the, of, and, etc." dans le but de les supprimer. Les tweets sont aussi comparés au dictionnaire des acronymes (Acronyms dictionary, 2015). Cette dernière comparaison permet de remplacer les acronymes par leur signification réelle, par exemple, (2nite → to night).

5.3.2 Extraction des caractéristiques

Pour caractériser les tweets, une méthode d'extraction des caractéristiques doit être appliquée. Il existe plusieurs techniques pour atteindre ces objectifs, parmi elles, la mé-

thode utilisée dans (Pandey *et al.*, 2017), où chaque tweet est converti en un vecteur de 11 caractéristiques. Cependant cette dimension reste importante surtout quand elle augmente proportionnellement avec la taille de l'ensemble de données. C'est pourquoi dans ce travail, nous proposons de réduire la dimension du vecteur caractéristique en passant de 11 à 5 caractéristiques : quatre sont primaires et la cinquième est facultative. Pour ce faire, nous nous sommes basés sur l'une des règles de fusions qui'est la somme (He *et al.*, 2010) comme démontré dans la Figure 5.1. En plus de la règle appliquée, une nouvelle caractéristique "valeur du thème" a été ajoutée et une autre "nombre de mots intenses" a été supprimé (cette fonctionnalité est particulièrement utilisée dans (Pandey et al., 2017) afin d'étudier le sarcasme dans les tweets ce qui n'est pas le cas dans ce présent travail). Ainsi, le vecteur caractéristique résultant sera défini comme suit :

- La première caractéristique concerne le nombre total de mots contenus dans un tweet après l'avoir prétraité (par exemple, si nous avons un tweet "love twitter ;)", sa première caractéristique sera 3).
- La seconde représente le nombre total de toutes les expressions positives, c'est-à-dire les émoticônes positives (Hogenboom *et al.*, 2015)) à l'instar de " :) " et " : D ", les exclamations positives (Exlamation, 2015), et les mots positifs (Liu *et al.*, 2005) (par exemple, si nous avons un tweet "love twitter ;)", sa deuxième caractéristique sera 2, parce que nous avons deux expressions positives "love" et l'émoticône " ;)").
- La troisième caractéristique est l'opposée de la seconde. Il représente toutes les expressions négatives, à savoir, les émoticônes négatives (Hogenboom *et al.*, 2015), les exclamations négatives (Exlamation, 2015), et les mots négatifs (Liu *et al.*, 2005) (par exemple, si nous avons un tweet "love twitter ;)", sa troisième caractéristique sera 0, car aucune opinion négative n'a été exprimée).
- La quatrième caractéristique concerne les expressions neutres. Il s'agit du nombre total des émoticônes neutres (Hogenboom *et al.*, 2015), des exclamations neutres (Exlamation, 2015), et des mots neutres (Neutral, 2015) (par exemple, si nous avons un tweet "twitter :-o", sa quatrième caractéristique sera 1, basée sur l'émoticône exprimée " :-o" qui sera considérée comme neutre. Vu que l'émoticône exprimée ne pourra être déterminée ni comme positive ni comme négative).
- La cinquième et dernière caractéristique prend aussi en considération les thèmes discutés dans les tweets. Par exemple, Si nous avons un ensemble de tweets concernant des opinions de produits ou services concurrents. Chaque produit ou service sera considéré comme un thème (à titre d'exemple, les quatre thèmes abordés dans Twitter-Sentiment-Corpus (Sanders, 2011)). Ainsi un numéro est attribué à chaque thème. Sinon, si nous avons un ensemble de données concernant les produits multivariés et non concurrents ou un ensemble de données concernant uniquement des opinions sur un seul thème, nous ne conservons que les quatre premières caractéristiques. De ce fait cette cinquième caractéristique est optionnelle et n'est ajoutée au vecteur caractéristique que si l'ensemble de tweets traités le permet.

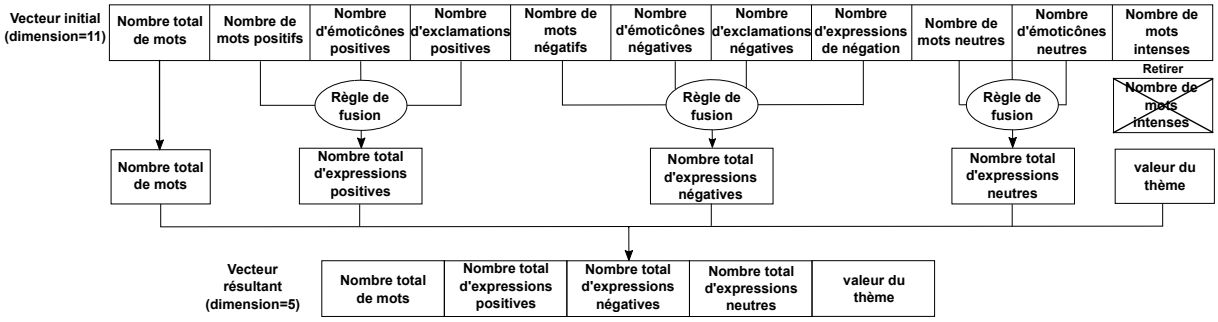


FIGURE 5.1 – La réduction du vecteur caractéristique en se basant sur une règle de fusion de type somme.

Après construction des vecteurs caractéristiques, nous les normalisons pour améliorer la précision et l'efficacité des algorithmes appliqués. Ce processus de normalisation est particulièrement utile pour les méthodes basées sur la distance, telles que les méthodes de clustering. Ça permet d'éviter l'obtention d'attributs avec des plages initialement importantes, en mettant leurs valeur à l'échelle. Cette dernière manœuvre normalise toutes les valeurs et les situent dans une plage spécifiée de petite taille, telle que 0.0 à 1.0 (Al Shalabi *et al.*, 2006). Dans ce travail, nous avons utilisé la méthode z-score (Al Shalabi *et al.*, 2006). Cette normalisation est utile lorsque le minimum et le maximum d'un attribut donné sont inconnus. Dans z-score, les attributs d'un vecteur donné x_i sont normalisés en fonction de la moyenne μ_{x_i} et de l'écart-type σ_{x_i} du point x_i. Un attribut x_{ij} est normalisé en $x_{ij}\prime$ comme indiqué dans l'équation (5.1).

$$x_{ij}\prime = \frac{x_{ij} - \mu_{x_i}}{\sigma_{x_i}}, \tag{5.1}$$

5.3.3 Algorithmes de clustering proposés

Comme indiqué auparavant, dans le domaine de l'analyse des sentiments, les points de vue exprimés sont généralement divisés en trois groupes : positif, négatif et neutre. Par conséquent, l'application des algorithmes de clustering ne permet pas souvent de connaître le nombre exact de clusters retournés, or que cette information est indispensable dans l'analyse des sentiments exprimés sur un service ou un produit donné. L'un des algorithmes le plus approprié pour cette problématique est l'algorithme K-means. Cet algorithme donne aux utilisateurs la possibilité de choisir le nombre de clusters retournés, ce qui aide à obtenir un nombre précis de clusters groupant les opinions exprimées. En dépit de son utilisation dans le domaine de l'analyse des sentiments de Twitter (Kumari et Babu, 2016; Pandey *et al.*, 2017; Riaz *et al.*, 2017), K-means reste moins efficace en termes de qualité de clustering. D'autre part les autres algorithmes de clustering tels que DENCLUE et ses variantes ont montré de bonnes performances dans différentes applications. Néanmoins, en classification des tweets, ils renvoient plus de trois clusters. Par exemple, dans les résultats expérimentaux, DENCLUE appliqué aux tweets obtient plus de trois clusters tandis que

K-means renvoie une précision moyenne aux alentours de 50%. Pour tirer profits des deux algorithmes, nous avons proposé des méthodes qui visent à avoir un compromis entre un nombre de clusters fixes, une bonne qualité de clusters et un temps de réponse raisonnable.
Plus précisément, nos algorithmes proposés, appelés K-DENCLUE, K-DENCLUE 2.0, K-DENCLUE-SA, K-DENCLUE-GA et K-DENCLUE-IM, sont une combinaison de K-means et DENCLUE avec ses variantes. Le processus général de nos algorithmes proposés est illustré dans la figure 5.2. Comme le montre cet organigramme, un hyper-rectangle est

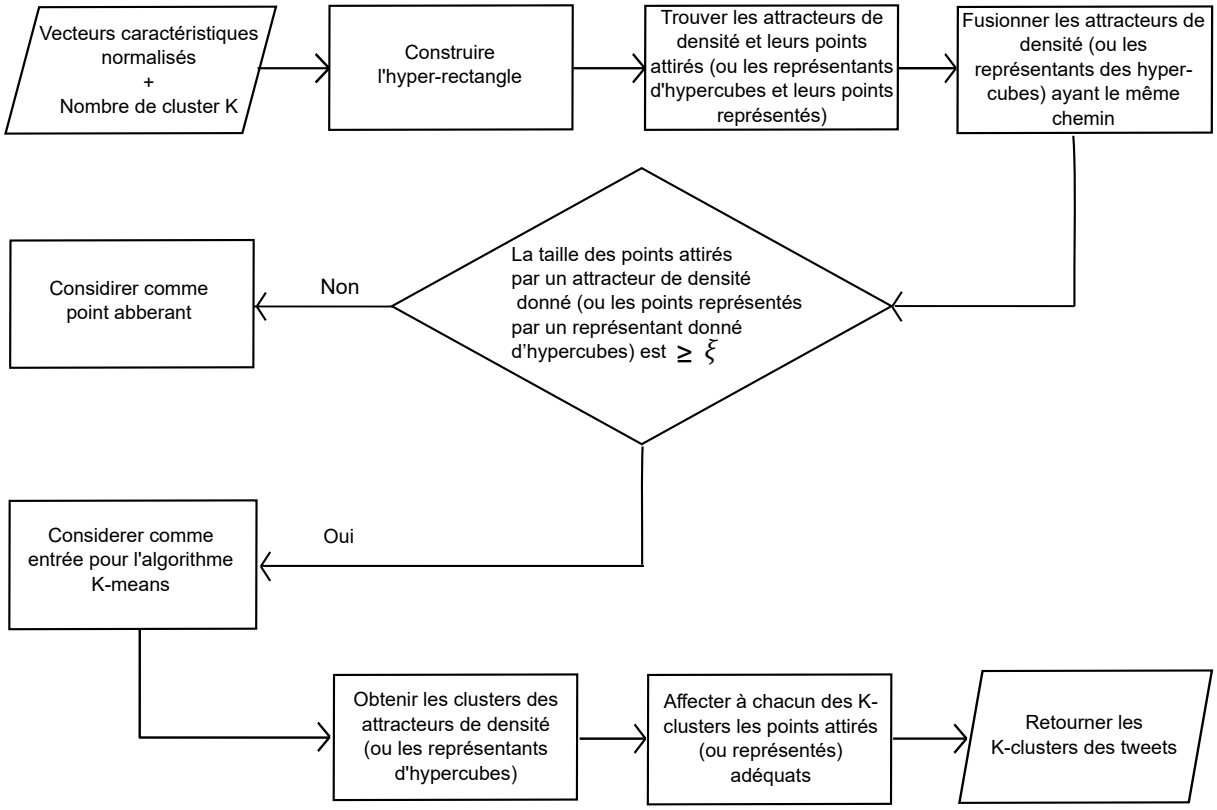


FIGURE 5.2 – L'organigramme général de K-DENCLUE et ses variantes.

construit en se basant sur les vecteurs caractéristiques extraits et normalisés. Cet hyper-rectangle est constitué d'hypercubes. Chaque hypercube est représenté par la dimension du vecteur caractéristique (c'est-à-dire le nombre d'attributs dans le vecteur) et par une clé. Cette structure permet à DENCLUE et à ses variantes de manipuler facilement les données, en utilisant les clés des cubes, et en ne considérant que les cubes peuplés.

Après la construction de l'hyper-rectangle, les attracteur de densité dans K-DENCLUE, K-DENCLUE 2.0, K-DENCLUE-SA et K-DENCLUE-GA, et les représentants d'hyper-cubes dans K-DENCLUE-IM sont extraits et fusionnés. Ensuite, les attracteurs de densité résultants et les représentants d'hypercubes, après fusion, sont considérés comme paramètres d'entrés de l'algorithme K-means. Dans ce cas, K-means ne sera pas appliqué sur l'ensemble des données mais juste sur quelques-unes, ce qui permet de réduire le temps

d'exécution. Enfin, lors de l'obtention des attracteurs de densité classifiés (ou des représentants des hyper-cubes dans le cas de K-DENCLUE-IM), chaque cluster est rempli par les points attirés (ou représentés) appropriés.
Nous présentons dans l'algorithme 5.1 le principe de K-DENCLUE. Nous notons que les autres variantes sont relativement proches de ce principe et notre approche est introduite dans les variantes de DENCLUE de la même manière décrite dans l'algorithme 5.1. Les notations utilisées par cet algorithme sont présentées ci-dessous :
Hr : l'hyper-rectangle construit.
cube : l'hypercube peuplé.
x : un point appartenant à un *cube* donné.
A : l'ensemble des attracteurs de densité.
$Aed(x_{Hcube})$: l'ensemble des points attirés par un ensemble donné d'attracteurs de densité x^*.
$Cluster$: un cluster construit.
$Clusters$: les clusters finaux retournés.

5.4 Résultats expérimentaux

5.4.1 Description des données

Pour évaluer la performance de distinction des différents sentiments des utilisateurs de Twitter, nous avons testé nos algorithmes sur cinq ensembles de données Twitter :

1. Twitter-Sentiment-Corpus-3 : C'est un corpus de 519 tweets positifs, de 572 tweets négatifs et de 2333 tweets neutres. Ces tweets recueillis abordent quatre thèmes à savoir Apple, Google, Facebook et Microsoft. Chaque tweet a été manuellement annoté en positif, négatif ou neutre (Sanders, 2011).
2. Twitter dataset : C'est un ensemble de données comportant des Tweets qui traitent des sujets nombreux tels que les sports, les images amusantes, etc. Ses tweets sont postés du 17 au 14 décembre 2014, étiquetés manuellement et divisés en deux classes à savoir une classe positive et une autre négative. Chacune des classes contient 1000 tweets, ceux positifs sont représentés par 1 et les négatifs par 0 (Twitter, 2014).
3. Testdata-manual-2009.06.14 : C'est un ensemble de données comportant des tweets basés sur divers sujets tels que Google, Obama, Kindle, Chine, Corée du Nord, Iran, San Francisco, dentistes, etc. Cet ensemble contient 497 tweets, chacun a été manuellement étiqueté soit avec une étiquette positive, négative ou neutre (Stanford, 2009).
4. Twitter-airline-sentiment : Il s'agit d'un ensemble de données concernant l'analyse de sentiments basée sur les tweets. Ces derniers concernent les grandes compagnies aériennes américaines. Ces données Twitter ont été recueillies en février 2015 et classées en tweets positifs, négatifs et neutres (Crowdflower, 2015).

Algorithme 5.1 L'algorithme K-DENCLUE

Procédure GetClusters (k, Hr, σ, ξ)
$A = \varnothing$, $Aed(null) = \varnothing$, $Cluster = \varnothing$, $Clusters = \varnothing$
pour chaque $cube \in Hr$ **faire**
    pour chaque $x \in cube$ **faire**
        $x^* = GETDENSITYATTRACTOR(x)$
        si $(f^D(x^*) \geqslant \xi)$
        {
            $A = A \cup \{x^*\}$
            $Aed(x^*) = Aed(x^*) \cup \{x\}$
        }
    fin pour
fin pour
pour chaque $x_i^* \in A$ **faire**
    pour chaque $x_j^* \in A, i \neq j$ **faire**
        si $(dist(x_i^*, x_j^*) \leqslant \sigma)$
        {
            $Aed(x_i^*) = Aed(x_i^*) \cup Aed(x_j^*)$
            retirer x_j^* de A
        }
        sinon {
            pour chaque $x_l \in Aed(x_i^*)$ **faire**
                pour chaque $x_m \in Aed(x_j^*)$ **faire**
                    si $((dist(x_l, x_m) \leqslant \sigma)$ et $(f^D(x_l) \geqslant \xi)$ et $(f^D(x_m) \geqslant \xi))$
                    {
                        $Aed(x_i^*) = Aed(x_i^*) \cup Aed(x_j^*)$
                        retirer x_j^* de A
                    }
                fin pour
            fin pour
        }
    fin pour
fin pour
$kClusters = kMeans(k, A)$
pour chaque $kCluster \in kClusters$ **faire**
    pour chaque $x^* \in A$ **faire**
        $Cluster = Cluster \cup Aed(x^*)$
    fin pour
    ajouter $Cluster$ à $Clusters$
fin pour
Retourner $Clusters$
Fin Procédure

Procédure getDensityAttractor (x)
$t = 0$
$x^0 = x$
répéter
    $x^{t+1} = x^t + \delta \dfrac{\nabla f^D_{Gauss}(x^t)}{\|\nabla f^D_{Gauss}(x^t)\|}$
    $t = t + 1$
jusqu'à $f^D(x^{t-1}) > f^D(x^t)$
Retourner x^{t-1}
Fin Procédure

Tableau 5.1 – Description des ensembles de données Twitter.

Ensembles de données	# Observations	# Classes
Twitter-Sentiment-Corpus-3	3424	3
Twitter dataset	2000	2
Testdata-manual-2009.06.14	497	3
Twitter-airline-sentiment	14640	3

Le tableau 5.1 donne une description synthétique des cinq ensembles de données abordés ci-dessus.

5.4.2 Évaluation des résultats

Pour une comparaison équitable entre les algorithmes, nous avons comparé nos approches avec d'autre algorithmes exploités dans le contexte d'analyse de sentiments dans Twitter décrits dans la section 5.2, tels que EM, K-means et DBSCAN. Mais aussi, nous avons décidé d'ajouter une comparaison avec l'algorithme DENCLUE vu qu'il représente la base de toutes les approches proposées. Cette étude comparative entre les différentes méthodes est élaborée à l'aide des mesures d'évaluation externes et internes décrites dans la section 1.7 du chapitre 1, du nombre de clusters retournés et du temps de réponse. Nous notons que pour cette étude comparative, tous les algorithmes ont été implémentés dans un environnement JAVA, sur un PC Core i5 (2,70 GHz) avec 8 Go de mémoire ; il faut noter aussi que chaque méthode a été exécutée dix fois et la moyenne de chaque mesure est retenue.
La qualité des clusters résultants et leur nombre ont été présentés dans les tableaux 5.2 et 5.3. Les meilleures valeurs obtenues sont mises en gras.

En fonction des mesures d'évaluation internes, présentées dans le tableau 5.2 et des valeurs moyennes calculées à partir des résultats des cinq ensembles de données présentés dans la dernière colonne, nous remarquons que DBSCAN a obtenu les meilleures valeurs des indices DI, DBI et CP dans deux ensembles de données, suivis de DENCLUE (les meilleures valeurs d'indices DI et CP dans un seul ensemble de données), puis de K-DENCLUE-SA (la meilleure valeur de DI dans un ensemble de données), de K-means (la meilleure valeur de DBI dans un seul ensemble de données), de K-DENCLUE (la meilleure valeur de DBI dans un seul ensemble) et K-DENCLUE 2.0 (le meilleur indice CP dans un ensemble de données). Tandis que pour les moyennes de tous les ensembles de données, DBSCAN a obtenu la meilleure valeur d'indice DI, K-DENCLUE-SA a obtenu la meilleure valeur de l'indice DBI et DENCLUE a la performance supérieure en se basant sur l'indice CP.
En termes de mesures externes, DBSCAN et DENCLUE ont obtenu chacun les meilleures valeurs de l'indice CA dans trois ensembles de données. En se basant sur l'entropie, DBSCAN et K-DENCLUE 2.0 ont surpassé les autres algorithmes dans deux ensembles de

Tableau 5.2 – Résultats du clustering des Tweets en fonction des mesures d'évaluation internes et externes.

Mesures		Algorithmes		Twitter-Sentiment -Corpus-3	Twitter dataset	Testdata-manual -2009.06.14	Twitter-airline -sentiment	Moyenne des ensembles
Mesures d'évaluation internes	DI (à maximiser)	État de l'art	EM	0.662	0.489	0.583	0.523	0.564
			K-means	0.505	0.486	0.528	0.452	0.493
			DBSCAN	0.771	**0.701**	0.549	**0.793**	**0.703**
			DENCLUE	**0.869**	0.650	0.544	0.718	0.695
		Proposés	K-DENCLUE	0.471	0.491	0.497	0.461	0.480
			K-DENCLUE2.0	0.567	0.543	0.574	0.521	0.551
			K-DENCLUE-SA	0.492	0.495	**0.727**	0.463	0.544
			K-DENCLUE-GA	0.498	0.498	0.494	0.584	0.518
			K-DENCLUE-IM	0.511	0.498	0.503	0.478	0.497
	DBI (à minimiser)	État de l'art	EM	1.320	1.572	1.431	1.125	1.362
			K-means	**1.146**	1.354	1.110	1.243	1.213
			DBSCAN	1.289	**1.053**	1.020	**0.868**	1.307
			DENCLUE	1.793	2.006	1.029	1.302	1.532
		Proposés	K-DENCLUE	1.237	1.678	**0.908**	1.148	1.242
			K-DENCLUE2.0	2.426	1.320	0.987	1.001	1.433
			K-DENCLUE-SA	1.156	1.276	0.919	0.899	**1.062**
			K-DENCLUE-GA	1.264	1.157	1.204	2.645	1.567
			K-DENCLUE-IM	1.795	1.222	0.969	1.164	1.287
	CP (à minimiser)	État de l'art	EM	1.491	2.081	1.853	1.167	1.648
			K-means	1.729	1.603	1.390	1.453	1.543
			DBSCAN	0.617	**0.554**	0.687	**0.497**	0.588
			DENCLUE	**0.484**	0.666	0.648	0.553	**0.587**
		Proposés	K-DENCLUE	1.220	1.067	0.629	1.311	1.056
			K-DENCLUE2.0	2.375	0.821	**0.489**	1.193	1.219
			K-DENCLUE-SA	1.109	1.171	0.727	1.326	1.083
			K-DENCLUE-GA	1.235	1.197	0.799	1.916	1.286
			K-DENCLUE-IM	1.329	1.245	0.705	1.366	1.161
Mesures d'évaluation externes	CA (à maximiser)	État de l'art	EM	0.681	0.534	0.476	0.641	0.583
			K-means	0.681	0.539	0.420	0.626	0.566
			DBSCAN	**0.726**	0.609	**0.620**	**0.646**	0.650
			DENCLUE	0.724	**0.616**	**0.620**	**0.646**	**0.651**
		Proposés	K-DENCLUE	0.710	0.603	0.578	0.626	0.629
			K-DENCLUE2.0	0.711	0.590	0.452	0.625	0.594
			K-DENCLUE-SA	0.708	0.565	0.542	0.626	0.610
			K-DENCLUE-GA	0.708	0.549	0.546	0.629	0.608
			K-DENCLUE-IM	0.714	0.571	0.596	0.628	0.627
	Entropy (à minimiser)	État de l'art	EM	1.169	0.992	1.462	1.197	1.205
			K-means	1.212	0.987	1.517	1.254	1.242
			DBSCAN	**0.845**	0.752	1.010	**1.116**	0.930
			DENCLUE	0.866	0.813	1.019	1.121	0.954
		Proposés	K-DENCLUE	0.938	0.707	0.894	1.209	0.937
			K-DENCLUE2.0	0.951	**0.586**	**0.752**	1.254	**0.885**
			K-DENCLUE-SA	0.976	0.823	1.058	1.285	1.035
			K-DENCLUE-GA	0.985	0.878	1.034	1.300	1.049
			K-DENCLUE-IM	0.927	0.885	0.986	1.226	1.006
	NMI (à maximiser)	État de l'art	EM	0.031	0.009	0.090	0.082	0.053
			K-means	0.007	0.013	0.053	0.056	0.032
			DBSCAN	**0.047**	0.029	0.130	0.088	0.073
			DENCLUE	0.046	0.031	0.129	**0.089**	**0.074**
		Proposés	K-DENCLUE	0.026	0.024	0.099	0.081	0.057
			K-DENCLUE2.0	0.013	0.016	0.032	0.030	0.023
			K-DENCLUE-SA	0.014	0.018	0.120	0.024	0.044
			K-DENCLUE-GA	0.020	0.013	0.102	0.014	0.037
			K-DENCLUE-IM	0.042	**0.035**	**0.133**	0.071	0.070

Tableau 5.3 – Comparaison entre les algorithmes en fonction du nombre de clusters retournés.

Algorithmes		Twitter-Sentiment -Corpus-3	Twitter dataset	Testdata-manual -2009.06.14	Twitter-airline -sentiment
État de l'art	EM	5	**2**	**3**	**3**
	K-means	**3**	**2**	**3**	**3**
	DBSCAN	17	8	4	27
	DENCLUE	24	7	4	30
Proposés	K-DENCLUE K-DENCLUE2.0 K-DENCLUE-SA K-DENCLUE-GA K-DENCLUE-IM	**3**	**2**	**3**	**3**

données. En ce qui concerne l'indice NMI, K-DENCLUE-IM a eu les meilleures valeurs dans deux ensembles, suivis de DBSCAN et de DENCLUE (un ensemble de données pour chaque algorithme). Concernant les moyennes calculées, DENCLUE a obtenu les meilleures valeurs des indices CA et NMI, alors que K-DENCLUE 2.0 présente la meilleure Entropie.

Tableau 5.4 – Comparaison entre les algorithmes en fonction de leur temps d'exécution (en seconde).

Algorithmes		Testdata-manual -2009.06.14(497)	Twitter dataset(2000)	Twitter-Sentiment -Corpus-3(3424)	Twitter-airline -sentiment(14640)
État de l'art	EM	1.503	8.230	140.146	7749.961
	K-means	0.15	6.900	48.781	5118.288
	DBSCAN	0.310	7.937	47.488	4695.724
	DENCLUE	0.187	7.934	44.864	2711.05
Proposés	K-DENCLUE	0.154	7.890	50.614	2607.274
	K-DENCLUE2.0	0.135	3.038	28.206	2594.624
	K-DENCLUE-SA	0.190	4.447	28.944	2519.562
	K-DENCLUE-GA	0.440	13.156	44.675	2612.374
	K-DENCLUE-IM	**0.038**	**0.720**	**2.588**	**125.193**

En général, nous avons remarqué que DBSCAN et DENCLUE ont eu les meilleures valeurs des mesures internes et externes que celles obtenues par les algorithmes EM et K-means, en particulier en termes de précision (indice CA). Le problème est que DBSCAN et DENCLUE renvoient un grand nombre de clusters comme remarqué dans le tableau 5.3. En ce qui concerne les algorithmes proposés, K-DENCLUE, K-DENCLUE 2.0, K-DENCLUE-SA, K-DENCLUE-GA et K-DENCLUE-IM ont pu trouver un équilibre entre la qualité des clusters et leur nombre.
D'après la figure 5.3, il est clairement observé que la différence entre les valeurs de précision (indice CA) obtenues par nos approches est faible par rapport aux meilleures valeurs.

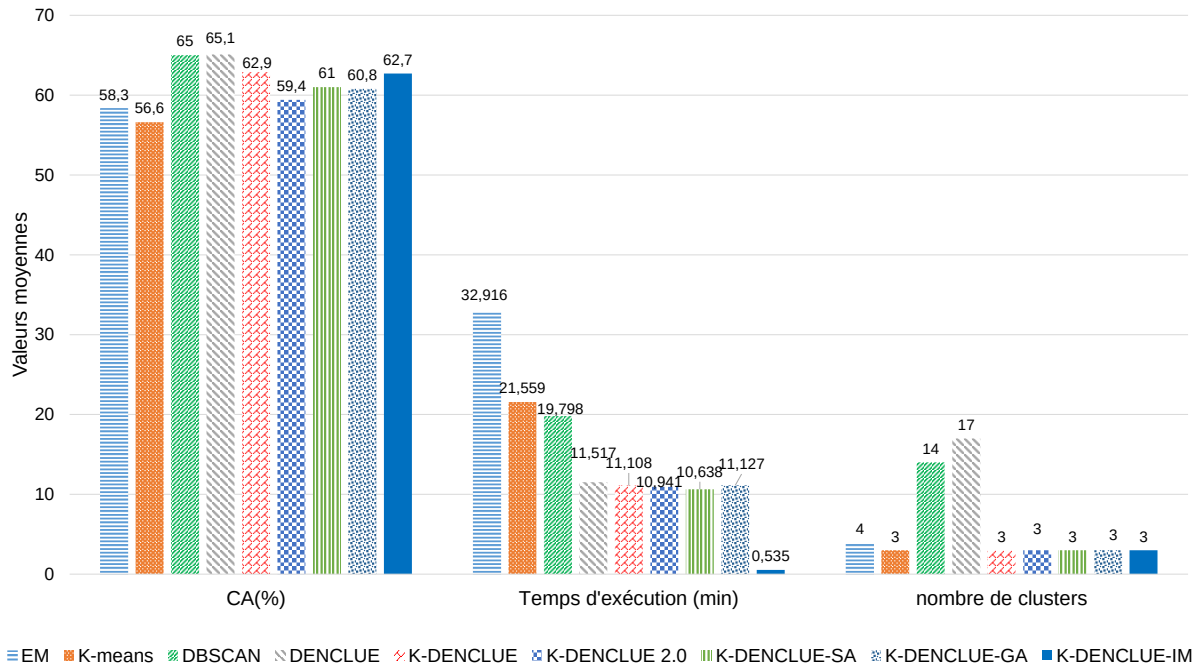


FIGURE 5.3 – Analyse comparative basée sur la moyennes de l'indice CA, le temps d'exécution et le nombre de clusters retournés

En termes de valeurs moyennes de la mesure CA, DENCLUE a obtenu la meilleure valeur (65,1%), suivi de DBSCAN (65%) et K-DENCLUE (62.9%), puis de K-DENCLUE-IM (62.7%), K-DENCLUE-SA (61%), K-DENCLUE-GA (60.8%) et K-DENCLUE 2.0 (59.4%). Tandis que EM et K-means ont obtenu respectivement 58.3% et 56.6%, ce qui est équivalent à plus que 8% de moins que la meilleure valeur.

En termes du temps d'exécution présenté dans le tableau 5.4 et des valeurs moyennes des cinq ensembles de données illustrées sur la figure 5.3, l'algorithme EM a obtenu le temps d'exécution le plus long, suivi de K-means, DBSCAN, DENCLUE, K-DENCLUE-GA, K-DENCLUE, K-DENCLUE-SA, K-DENCLUE 2.0 et enfin K-DENCLUE-IM. Le temps d'exécution est généralement affectée par l'augmentation de la taille des ensembles de données Twitter. Ces résultats démontrent que le compromis entre la qualité, le temps d'exécution et le nombre de clusters retournés a été atteint. Nous signalons que K-DENCLUE-IM reste un choix judicieux pour classifier les grands ensembles de tweets.

5.5 Conclusion

Dans ce travail, de nouveaux algorithmes de clustering ont été introduits afin d'analyser les sentiments des tweets. Les approches proposées, basées sur K-means et DENCLUE, ont été conçues pour réduire le nombre de clusters retournés, sachant que les sentiments exprimés par les utilisateurs de Twitter sont généralement divisés en trois clusters : positif, négatif et neutre. Les nouvelles méthodes ont été testées sur cinq ensembles de données

Twitter et comparées à quatre algorithmes d'état de l'art, à savoir EM, K-means, DBSCAN et DENCLUE. Les comparaisons basées sur six mesures d'évaluation, le nombre de clusters retournés et le temps d'exécution ont démontré l'efficacité de nos approches, en particulier dans le cas de l'algorithme K-DENCLUE-IM.

Chapitre

6 Prévention du cancer du nasopharynx à l'aide des algorithmes de clustering

Sommaire

6.1 Introduction

Dans cette dernière décennie, le domaine médical a connu une croissance exponentielle du nombre d'informations. Les spécialistes dans ce domaine, ne pouvant plus traiter ces informations manuellement, ont eu recours aux techniques du Maching Learning, parmi elles, les techniques de clustering. Dans ce contexte, les auteurs (Albayrak, 2003) ont mis en œuvre des méthodes de classification non supervisées pour regrouper les patients en utilisant des données en relation de la glande thyroide. Dans (Paul et Hoque, 2010), les auteurs ont proposé un algorithme de clustering, nommé k-Means-Mode dans le but de déterminer et réduire le risque d'une maladie, en particulier une maladie diabétique. Dans (Khanmohammadi *et al.*, 2017), une combinaison entre l'algorithme OKM (pour le terme anglais : Overlapping K-Means) et l'algorithme K-harmonic means a été proposée et appliquée à dix ensembles de données médicales.

Dans ce travail, nous avons proposé d'appliquer les techniques de clustering sur un ensemble de données qui traite le cancer du nasopharynx. Les algorithmes de clustering utilisés incluent l'algorithme K-means, l'algorithme EM, l'algorithme DENCLUE, l'algorithme DENCLUE 2.0, en plus de certains de nos algorithmes proposés, à savoir DENCLUE-SA, DENCLUE-GA et DENCLUE-IM.
Le but principal de ce travail est de montrer l'importance des techniques du Maching Learning et de l'intelligence artificielle dans le domaine médical, particulièrement la prévention de certaines maladies comme le cancer du nasopharynx.

Dans ce chapitre, nous avons commencé par décrire l'ensemble de données utilisé,

avant de fournir les résultats expérimentaux, les analyser et les discuter. Ainsi le chapitre est clôturé par une conclusion qui reprend les grandes lignes de ce qui a été réalisé.

6.2 Description des données

Cet ensemble de données fait partie d'un projet visant à évaluer les facteurs pronostiques du carcinome du nasopharynx (NPC : en anglais NasoPharyngeal Carcinoma) (il s'agit de la partie du pharynx située derrière la cavité nasale). Ces données ont été collectées par l'Institut Pasteur à Casablanca. Techniquement, cette base de données comprend 90 patients traités par le centre Mohammed IV pour le traitement des cancers à Casablanca, de décembre 2016 à février 2018. Chaque patient est caractérisé par 26 attributs (parmi eux : Les différents symptômes, les antécédents familiaux du cancer, la consommation d'alcool ou du tabac, etc.) Les patients sont classés selon les quatre stades d'avancement des tumeurs, à savoir :
1) tumeur limitée au nasopharynx,
2) tumeur étendue aux tissus mous de l'oropharynx ou de la cavité nasale,
3) tumeur envahissant les structures osseuses ou les cavités aériennes du visage,
4) tumeur avec extension intracrânienne, atteinte des nerfs crâniens, invasion de la fosse infratemporale, de l'orbite ou de l'hypopharynx.

6.3 Résultats expérimentaux

En fonction des attributs de l'ensemble de données et des quatre phases d'évolution de la tumeur, nous pourrons déduire plusieurs informations utiles aux praticiens dans le domaine qui traite du cancer du nasopharynx.

Tableau 6.1 – La qualité estimée par les mesures d'évaluation et le temps d'exécution

Mesures \ Algorithmes	k-means			EM	DENCLUE	DENCLUE 2.0	DENCLUE-SA	DENCLUE-GA	DENCLUE-IM
	k=3	k=4	k=5						
DI	0.514	0.510	0.488	**0.614**	0.489	0.512	0.547	0.562	0.518
DBI	2.254	2.329	2.323	2.540	**2.110**	2.166	3.062	2.722	2.506
CP	0.906	0.903	0.879	2.459	0.857	0.940	**0.781**	0.802	0.822
CA	0.433	0.455	0.455	0.522	0.507	0.379	**0.592**	0.526	0.56
Entropy	1.839	1.765	1.766	1.558	1.283	1.817	**0.806**	1.262	1.124
NMI	0.070	0.101	0.104	0.161	0.088	0.0123	0.155	0.154	**0.245**
Run time (ms)	0.035	0.034	0.038	1.503	0.058	0.100	0.092	0.174	**0.026**

Nous avons observé dans notre analyse que le nombre de patients détectés NPC, en troisième et quatrième stade de la maladie, augmente lorsque le patient a des antécédents

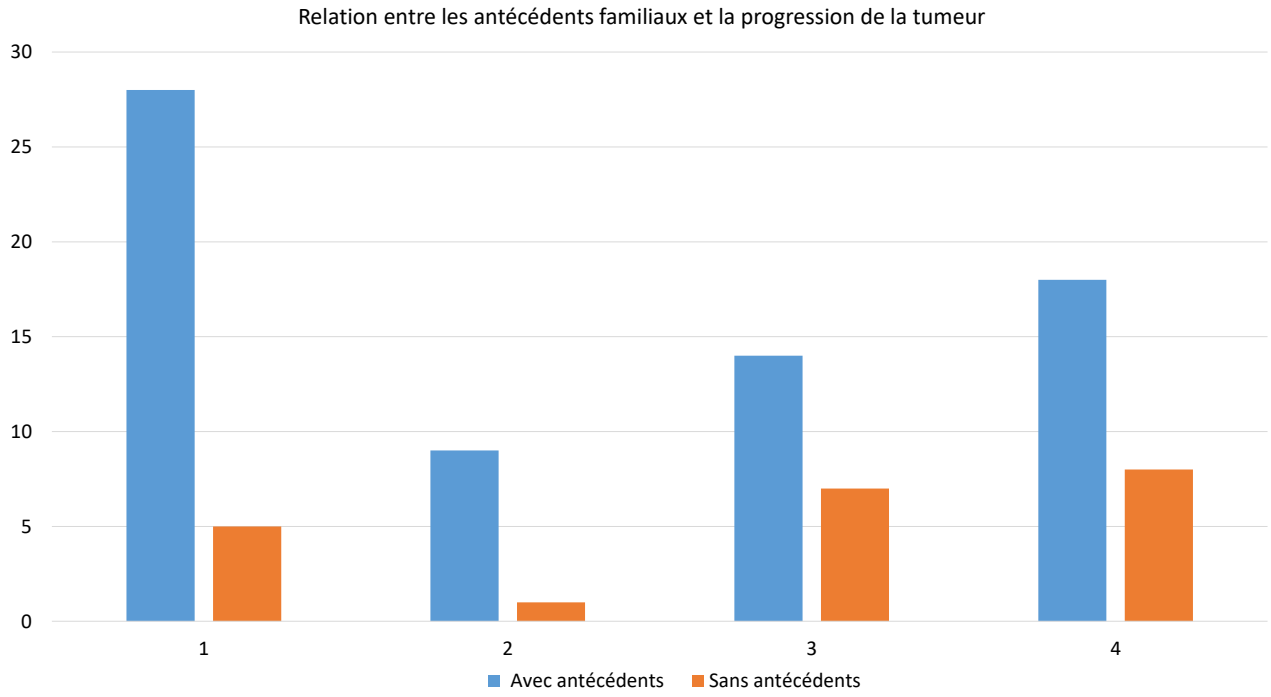


FIGURE 6.1 – Relation entre les antécédents familiaux du cancer et la progression de la tumeur.

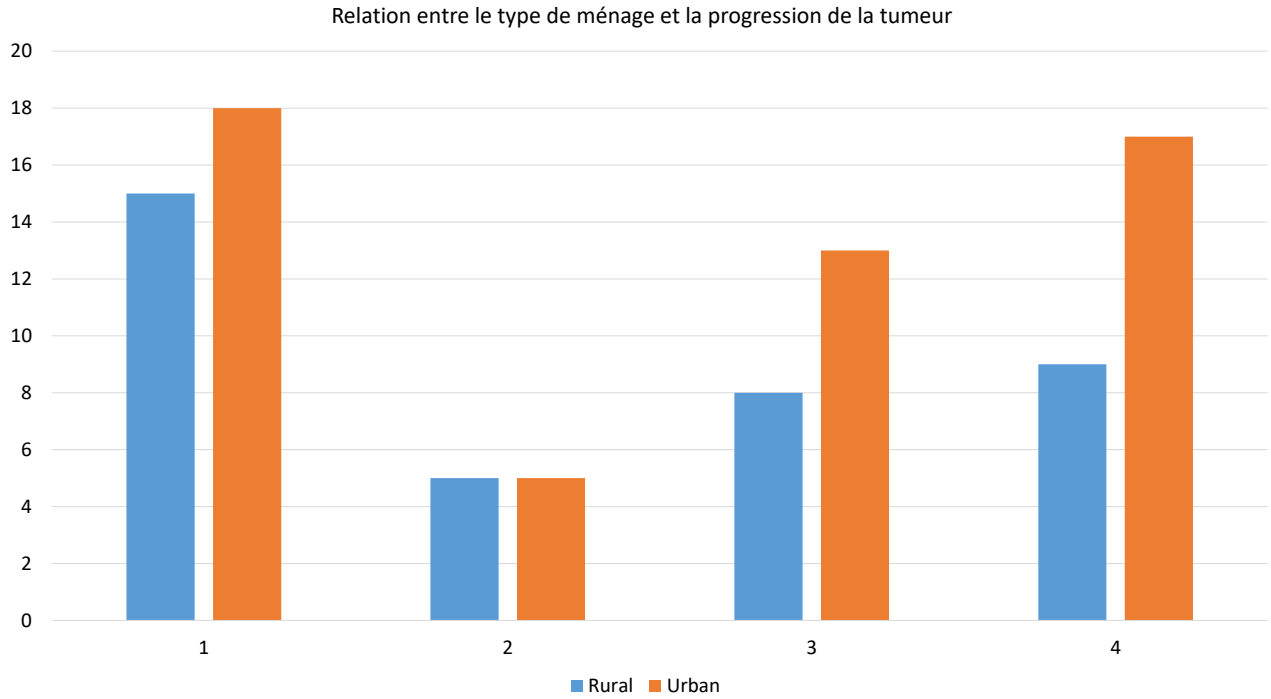


FIGURE 6.2 – Relation entre l'habitat d'enfance et la progression de la tumeur.

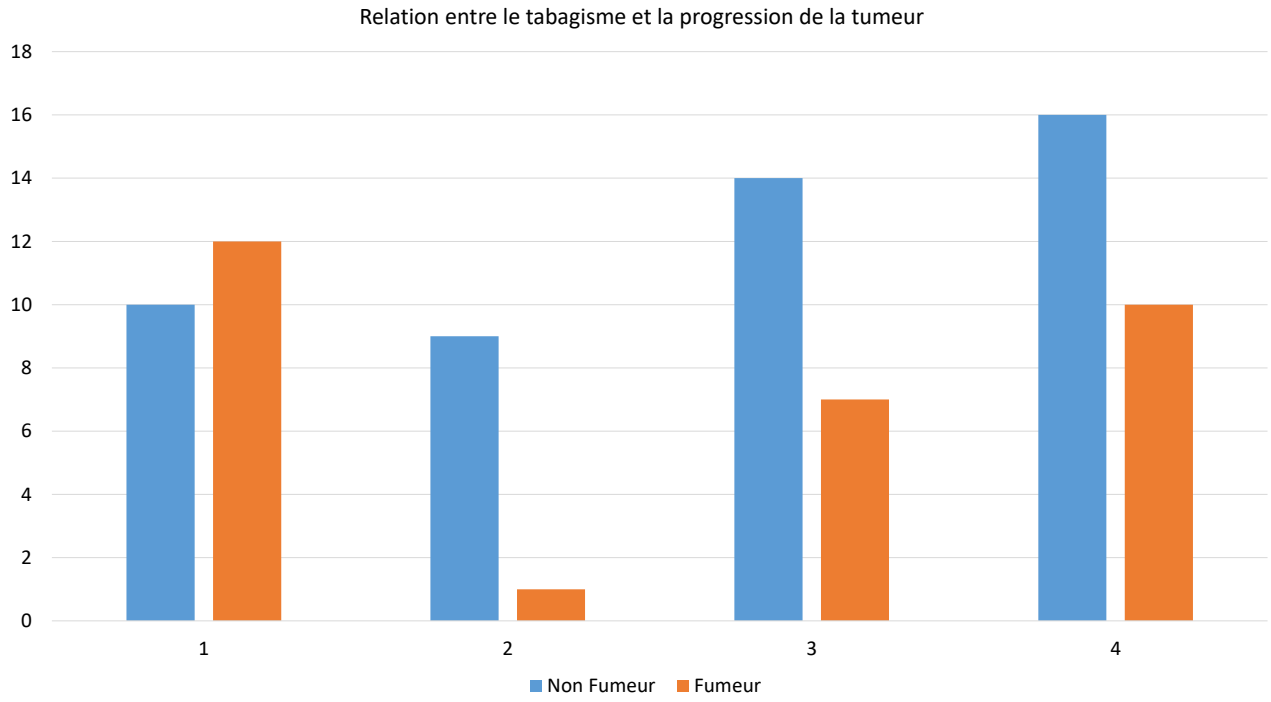


FIGURE 6.3 – Relation entre le tabagisme et la progression de la tumeur.

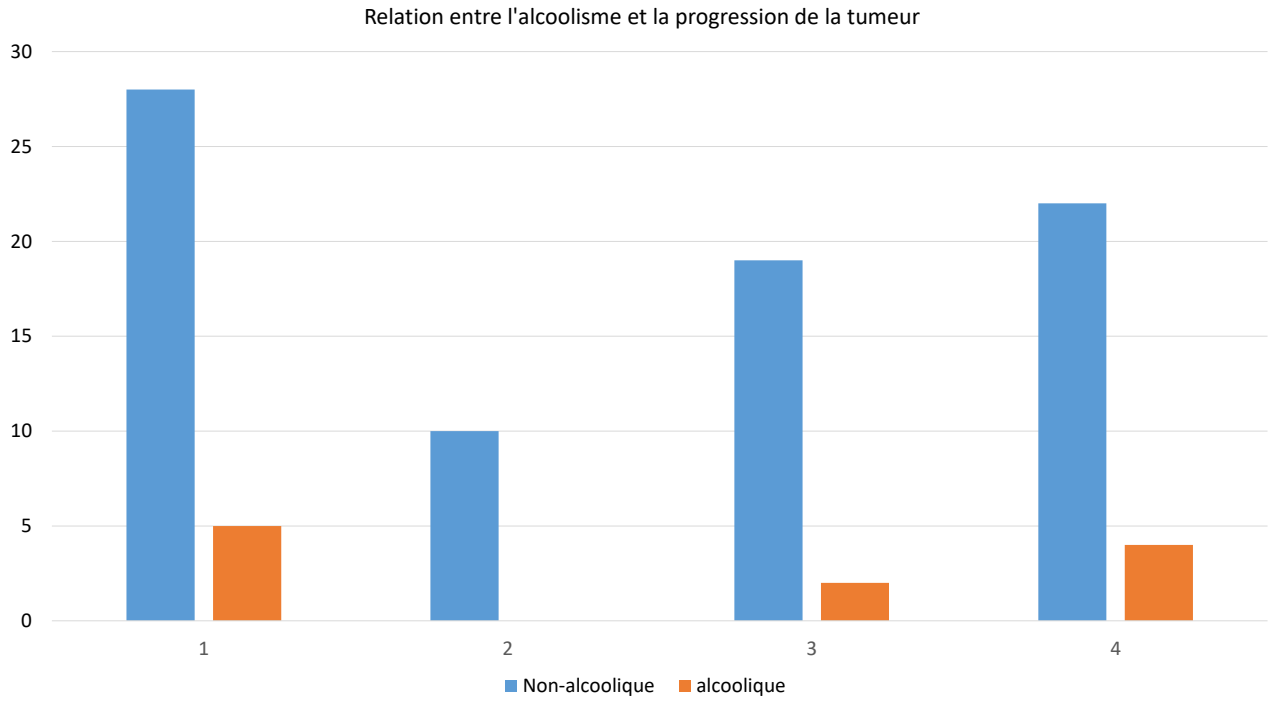


FIGURE 6.4 – Relation entre l'alcoolisme et la progression de la tumeur.

familiaux du cancer. Cette relation entre les antécédents familiaux des patients et la propagation de la tumeur dans le nasopharynx est illustrée à la figure 6.1. Les praticiens peuvent déduire que la génétique humaine pourrait avoir un impact sur ce type de cancer. Dans la figure 6.2, nous avons détecté que le type de ménage ou le patient avait passé son enfance (rural ou urbain) a aussi un impact sur le stade de la progression tumorale. En effet, nous notons que les personnes qui ont habiter dans des zones rurales pendant la période de l'enfance sont moins touchés que ceux des zones urbaines. Cela laisse à déduire que la pollution a son rôle dans la propagation de la tumeur. La relation entre la consommation de cigarettes et le degré de propagation de la tumeur dans le nasopharynx est également mise en évidence. Il est clair sur la figure 6.3 que les fumeurs atteints de cette maladie ont malheureusement plus de probabilité d'arriver à des stades avancés et sont plus affectés que les non-fumeurs. Dans la figure 6.4, nous avons étudié l'impact de l'alcoolisme sur la propagation de la tumeur. Les résultats obtenus ne peuvent pas détecter une vrai relation entre l'alcool et la propagation de cette maladie dans le corps humain.
Les résultats bien entendu relèvent d'une étude initiale et doivent d'avantage être renforcés par d'autres études. En termes de mesures d'évaluation décrites dans le tableau 6.1, nous démontrons que les algorithmes basés sur la densité obtiennent les meilleures résultats.

6.4 Conclusion

Dans ce chapitre, nous avons démontré l'utilité du Maching Learning, plus précisément les techniques de clustering, dans l'analyse préventive des maladies. Pour cette finalité, nous avons appliqué différents algorithmes de clustering, à savoir K-means, EM, DENCLUE, DENCLUE 2.0, DENCLUE-SA, DENCLUE-GA, DENCLUE-IM. Nous avons ensuite évalué les résultats obtenus selon trois mesures d'évaluation internes et trois mesures externes, à savoir DI, DBI, CP, CA, NMI et entropie. En se basant sur les résultats de cette étude, il a été conclu que les antécédents familiaux du cancer, le cadre de vie en milieu urbain ou rural et la consommation de tabac sont étroitement associés au stade avancé du NPC, laissant ainsi une probabilité d'associer ces facteurs de risques à la propagation de la tumeur initiale vers des stades plus avancés de la maladie en contribuant à la propagation des métastases. Les résultats obtenus font preuve de l'importance des techniques du Maching Learning dans le traitement des maladies, en particulier le cancer du nasopharynx.
Techniquement, pour cette base de données, les algorithmes basés sur la densité ont donné des résultats nettement supérieurs à ceux basés sur le partitionnement ou les modèles statistiques.
Comme perspectives, nous allons étendre notre étude sur un ensemble de données plus important regroupant des données en provenance du grand Maghreb. De plus nous sommes intéressés à étudier l'aspect dynamique des données dans le domaine médical en développant un algorithme de classification dynamique approprié basé sur la densité. Sachant que

les informations relatives aux patients sont ajoutées de manière dynamique dans la base de données et sont variables dans le temps.

Conclusion générale et perspectives

Dans ce mémoire, nous nous sommes intéressés au domaine de la science de données. Ce domaine qui a vu le jour suite à la prise de conscience de l'importance de l'information utile qu'on peut extraire d'une donnée. Le clustering, l'une des techniques du Machine Learning qui est en lui même une branche de la science de données, nous permet de partitionner un ensemble de données, sans connaissances a priori, afin de faciliter l'extraction des informations pertinentes. Notre attention s'est orientée sur les algorithmes de clustering basés sur la densité, plus précisément DENCLUE, vu sa force de trouver des clusters de formes arbitraires et de filtrer les données bruités, en plus de son utilisation des grilles, facilitant ainsi les calculs.

Avant de décrie nos contributions, nous avons commencé, dans le premier chapitre, par une formulation de la problématique du clustering, tout en présentant les notions de base, l'état de l'art et les mesures aidant à évaluer la qualité de nos approches.

Dans le deuxième chapitre, trois améliorations de l'algorithme DENCLUE ont été proposées. En fonction de six mesures d'évaluation nous avons pu déterminer la particularité de chacune des approches et sa performance que ça soit en termes de qualité ou de temps de réponse. L'une des approches : DENCLUE-IM, a démontrée sa force en ayant des résultats satisfaisants en termes de mesures d'évaluation tout en retournant les clusters dans les meilleurs des délais, et ce pour les huit ensembles de données expérimentés.

Malgré les performances réalisées par nos approches, nous avons remarqué que l'ensemble de données iris, quel que soit l'algorithme de clustering utilisé, se regroupe toujours en deux clusters au lieu de trois, ce qui influence directement la qualité du clustring. Pour cela nous avons analysé cet ensemble de données et découvert qu'il présente un problème de chevauchement de données entre deux de ses clusters. C'est ce qui nous a poussé à chercher un remède aux ensembles dont les données sont chevauchées afin d'améliorer davantage la qualité du clustering.

Dans le troisième chapitre, nous avons développé une méthode permettant une bonne séparation des données présentant un problème de chevauchement. Notre approche nommée SDA, nous a aidé à bien séparer les clusters et à améliorer la qualité du clsutering. Les performances de notre approche ont été démontrées visuellement et quantitativement à l'aide des mesures d'évaluation. En ce qui concerne les résultats visuels, nous avons

nettement constaté la différence entre les données d'origine et ceux remis à l'échelle par notre approche. Tandis que pour ceux quantitatives, les mesures d'évaluation ont obtenu de meilleurs scores surtout pour l'indice de précision CA.

Les améliorations proposées dans la première partie de ce mémoire, que ça soit du côté des algorithmes ou celui des données, ont été exploitées dans des domaines bien spécifiques tels que le Cloud Computing dans le chapitre 4 et l'analyse de sentiment dans twitter dans le chapitre 5.

Afin d'améliorer un système de recherche et de sélection de services Cloud, nous l'avons combiné avec les techniques de clustering. L'idée derrière est d'épargner aux utilisateurs la recherche dans des milliers de services Cloud et de retourner un ensemble assez restreint de services les plus proches au service idéal choisi par l'utilisateur et ainsi les plus adaptés à ses espérances. L'approche proposée dans le chapitre 4 a permis de passez de 50000 services sélectionnés par l'opérateur Skyline, à 2728 services sélectionnés par Skyline combiné à la technique ELECTRE IS, à enfin 11 services sélectionnés à l'aide de notre méthode qui se base sur le principe du service idéal et du clustering.
À cette fin, quatre algorithmes de clustering appartenant à des familles différentes ont été testés pour permettre le choix de la méthode la plus appropriée à notre cas. L'algorithme DENCLUE-IM a démontré des résultats prometteurs en permettant le renvoi de 11 services pertinents en 0.997 secondes. Par conséquent, cet algorithme a trouvé un compromis entre la qualité du clustering, le temps de réponse et le nombre de services sélectionnés et retournés à l'utilisateur final.

Dans le dernier chapitre, des approches basées sur les améliorations vues dans la première partie, ont été introduites afin d'analyser les sentiments dans Twitter. Les algorithmes de clustering proposés ont été basés sur K-means et DENCLUE. Ils ont été conçues pour réduire le nombre de clusters retournés, sachant que les sentiments exprimés par les utilisateurs de Twitter sont généralement divisés en trois clusters : positif, négatif et neutre. Les nouvelles méthodes, testées sur cinq ensembles de données Twitter, ont été comparées à quatre algorithmes d'état de l'art : EM, K-means, DBSCAN et DENCLUE. L'étude comparatif basée sur les six mesures d'évaluation, le nombre de clusters résultants et le temps d'exécution ont démontré l'efficacité de nos approches, en particulier dans le cas de l'algorithme K-DENCLUE-IM.

Comme perspectives, plusieurs voies sont envisageables. À cours terme, nous sommes en train d'élaborer un algorithme de clustering dynamique afin de répondre le plus aux besoins des données qui arrivent d'une manière dynamique et en temps réel. À moyen terme, nous allons investir nos approches dans le domaine du Big Data. À long terme, nous voulons analyser les techniques de l'apprentissage profond (dit en anglais : Deep Learning) dans le cadre de la classification non supervisée : Deep clutering.

Annexes

Annexe

A

Opérateur Skyline

L'opérateur Skyline est l'une des techniques les plus importantes qui permettent de résoudre le problème du vecteur maximal (Kung *et al.*, 1975). Il présente une approche permettant de récupérer les points pertinents dans un grand ensemble de données. La notion du point pertinent fait référence au point qui optimise certaines exigences tout en ayant de bonnes valeurs pour toutes les autres. Les points récupérés par l'opérateur Skyline ne sont dominés par aucun autre point.
Le Skyline est formé par les points qui ne sont dominés par aucun autre point. On dit qu'un point p domine le point q si p est égal ou supérieur à q pour toutes les dimensions et p est supérieur à q pour au moins une dimension. Si p est meilleur que q dans certaines dimensions et q est meilleur que p pour d'autres, alors p et q sont incomparables et peuvent tous les deux faire partie du Skyline s'ils ne sont dominés par aucun autre point.
Le Skyline a trois propriétés principales :

1. La relation de dominance est transitive. Si p domine q et q domine r, alors p domine r ;
2. Pour toute fonction de notation monotone $f : M \to \mathbb{R}$ où M est l'ensemble des points pour lesquels le Skyline doit être calculé, si $p \in M$ maximise f, alors p est dans l'horizon. Ainsi, les utilisateurs trouveront leur point favori dans Skyline, quelle que soit la pondération de leurs préférences pour les dimensions ;
3. Pour chaque point p dans le Skyline, il existe une fonction de notation monotone $f : M \to \mathbb{R}$ telle que p maximise f. Ainsi, chaque point contenu dans le Skyline est le favori d'au moins un utilisateur.

Il existe deux approches majeures pour calculer le Skyline. La première approche consiste à utiliser les bases de données existantes et à étendre la requête de sélection avec un nouvel opérateur logique, SKYLINE OF (Börzsönyi *et al.*, 2001). Quant à la seconde, elle se base sur l'utilisation des algorithmes.

A.1 Sykine : Requêtes SQL

L'extension des bases de données existantes consiste à utiliser les instructions SQL standard et à les étendre avec une nouvelle clause, le SKYLINE OF (Börzsönyi *et al.*,

2001). La structure d'une requête SQL étendue pour calculer le Skyline peux être exprimée comme suit :

— SELECT ... FROM ... WHERE ...
— GROUP BY ... HAVING ...
— SKYLINE OF [DISTINCT] $dimension_1$ [MIN, MAX, DIFF], ... , $dimension_n$ [MIN, MAX, DIFF]
— ORDER BY ...

Les n dimensions de Skyline, telles que le prix et la distance de la plage, sont désignées par $dimension_s$, avec $s \in [1, n]$. L'opérateur MIN indique que la dimension doit être minimisée tandis que l'opérateur MAX indique qu'elle doit être maximisée. L'opérateur $DIFF$ spécifie que la valeur de la dimension doit être différente. Si nous considérons l'exemple de l'hôtel, les dimensions prix et distance (c'est-à-dire la distance de la plage) doivent être minimisées. Une autre dimension, telle que le classement (c'est-à-dire le nombre d'étoiles), devrait être maximisé. Par exemple : SELECT nom, prix, distance, classement FROM hôtels
SKYLINE OF prix MIN, distance MIN, classement MAX

L'utilisation de requêtes SQL pour calculer Skyline est intuitive, mais elle présente l'inconvénient de se convertir en boucles, ce qui conduit à des requêtes très complexes lorsque le nombre de dimensions est supérieur à deux. Cette complexité engendre des performances médiocres et un coût de calcul supplémentaire.

A.2 Skyilne : Algorithmes

L'un des moyens les plus efficace pour calculer Skyline est utilisation des algorithmes, tels que l'algorithme BNL (Block-Nested-Loops) (Börzsönyi *et al.*, 2001), l'algorithme diviser pou reigner (D&C : Divide-And-Conquer) (Kung *et al.*, 1975; Preparata et Shamos, 1985) et l'utilisation des B-arbres (B-trees) (Comer, 1979). L'avantage qu'implique l'utilisation des algorithmes, c'est qu'ils peuvent être appliqués pour calculer le Syline, peu importe le nombre de dimensions dont les données disposent.

A.2.1 BNL : Block-Nested-Loops

L'algorithme BNL est basé sur l'algorithme Basic-Nested-Loops. Il calcule la liste de lignes dans Skyline en effectuant une comparaison par paires des n-uplets et en conservant les tuples incomparables dans une fenêtre de la mémoire principale. À chaque itération, un tuple p est lu à partir de l'ensemble de tuples d'entrée et comparé aux tuples incomparables de la fenêtre de la mémoire principale, ouvrant l'une des trois possibilités suivantes :

1. Si p s'avère être dominé par un tuple q dans la fenêtre (dénoté $p < q$), alors p est ignoré car il ne fait pas partie du Skyline. Ainsi, aucune autre comparaison n'est faite et un nouveau tuple est lu à partir de l'ensemble d'entrée ;

2. S'il s'avère que p domine un tuple q dans la fenêtre (dénoté $p > q$), le tuple q est supprimé de la fenêtre et p le remplace.
3. Si p s'avère incomparable avec tous les n-uplets de la fenêtre, ce qui signifie que p ne domine pas et qu'il n'est dominé par aucun autre tuple dans la fenêtre, alors p est ajouté à la fenêtre. Cependant, s'il existe une limite de mémoire, p est écrit dans un fichier temporaire (noté T). Ce fichier temporaire est stocké sur le disque et est pris en compte dans une itération lorsque le nouveau tuple à comparer est incomparable avec tous les n-uplets de la mémoire principale.

Afin de s'assurer que deux tuples ne sont pas comparés deux fois, un compteur est affecté à chaque tuple dans la fenêtre de la mémoire principale et dans le fichier temporaire du disque. Ce compteur permet de suivre l'ordre dans lequel les n-uplets ont été insérés dans la fenêtre ou le fichier temporaire et servira à déterminer si deux tuples ont déjà été comparés afin d'éviter de les comparer deux fois.
Comme l'algorithme BNL itère dans la liste Skyline, cela fonctionne mieux lorsque Skyline est petit. Sa complexité varie entre $O(n)$ dans le meilleur des cas et $O(n^2)$ dans le pire des cas, n étant le nombre de tuple dans l'ensemble d'entrées.
D'autres variantes du BNL ont également été présentées dans (Börzsönyi *et al.*, 2001). La première consiste à continuer à placer les nouveaux tuples dominants en haut de la fenêtre de la mémoire principale. Ainsi, ils seront d'abord comparés à tout nouveau tuple de la liste des entrées et auront une meilleure chance de l'éliminer. Cette approche s'avère plus efficace s'il y a des tuples dits tueurs dans l'ensemble des tuples, ou des tuples qui ont de si bonnes valeurs pour de nombreuses dimensions qu'ils sont capables d'éliminer un nouveau tuple sans avoir besoin de le comparer à tous les tuples de la fenêtre.
Une autre variante consiste à conserver les n-uplets les plus dominants dans la fenêtre de la mémoire principale plutôt que dans le fichier de disque temporaire. De cette façon, ils seront considérés en premier lorsqu'un nouveau tuple est lu dans la liste d'entrée.
Il convient de noter que ces approches étaient destinées aux environnements disposant de ressources de calcul et de stockage limitées, ce qui n'est plus très pertinent de nos jours car ces ressources deviennent de plus en plus abordables. Cela conduit à réexaminer ces approches, car elles peuvent accroître inutilement la complexité de l'algorithme BNL.

A.2.2 D & C : diviser pour régner

L'algorithme basique de D & C utilisé dans le calcule de Skyline (Kung *et al.*, 1975; Börzsönyi *et al.*, 2001), fonctionne en trois étapes :

1. La première étape consiste à calculer le point médian m_p de l'ensemble d'entrée de tuples P sur une dimension d_p à partir des dimensions utilisées pour calculer le Skyline. Ensuite, P est divisé en deux ensembles : P_1 contenant les n-uplets ayant une valeur de d_p meilleur que m_p et P_2 contenant les n-uplets ayant une valeur de d_p médiocre que m_p. Le sens de la comparaison (meilleur ou médiocre) dépend de la dimension. Par exemple, pour une dimension à minimiser, P_1 contiendra tous

les tuples pour lesquels la valeur de d_p est inférieure à m_p, et P_2 contiendra tous les tuples pour lesquels la valeur de d_p est supérieure à m_p ;

2. La deuxième étape consiste à calculer le Skyline de chaque partition en appliquant de manière récursive l'algorithme D & C jusqu'à ce qu'une partition ne puisse plus être partitionnée (lorsqu'elle est composée d'un seul tuple ou de n-uplets ayant la même valeur pour la dimension utilisée pour le partitionnement). Soit S_1 le Skyline de P_1 et S_2 le Skyline de P_2 ;
3. La dernière étape consiste à fusionner S_1 et S_2 en effectuant une comparaison par paires de leurs n-uplets. Comme S_1 contient des tuples qui sont meilleurs en ce qui concerne la dimension de partitionnement, ils ne peuvent pas être dominés par des tuples dans S_2. Ainsi, dans cette étape, les tuples de S_2 qui se révèlent être dominés par des tuples de S_1 sont éliminés, ne laissant que les tuples faisant partie du Skyline. Cette fusion est répétée jusqu'à ce que tous les ensembles soient fusionnés en un seul ensemble résultant le Skyline.

L'algorithme D & C a une complexité de $O(n \times log(n)^{d-2} + O(n \times log(n))$, où n est le nombre de n-uplets dans l'ensemble d'entrée et d le nombre de dimensions. Cette complexité est maintenu dans le meilleure et pire des cas. Ainsi, l'algorithme D & C offre de meilleures performances que l'algorithme BNL dans le pire des cas, et est pire dans le meilleur des cas.

A.2.3 B-arbres

Une autre implémentation de Skyline présentée dans (Börzsönyi *et al.*, 2001) utilise un B-arbre (B-tree). Dans ce cas, les tuples sont placés dans des index ordonnés, chacun contenant l'ensemble d'entrée de tuples ordonné en fonction de chaque dimension (par exemple, une liste de services cloud ordonnée par ordre croissant de coût d'acquisition, une autre de services cloud ordonnée par ordre décroissant de bande passante, etc.) Tous les index obtenus sont analysés simultanément et les n-uplets sont ajoutés à la liste Skyline. L'analyse s'arrête lorsqu'une correspondance est trouvée, ce qui signifie qu'un tuple p est atteint dans tous les index. Ce tuple p est sûr de dominer tout autre tuple à venir (qui est ordonné après) parce qu'il a une meilleure valeur pour toutes les dimensions. Une fois que cette liste de tuples initiale est obtenue, un algorithme (tel que le BNL ou le D & C) est exécuté pour calculer le Skyline.
Cette méthode peut être utilisée pour un calcul préalable à Skyline, afin de réduire la taille de l'ensemble des tuples en entrée en éliminant ainsi les tuples sûr à dominer. Il donne de meilleurs résultats lorsque le nombre de dimensions est inférieur à trois et que la condition d'arrêt est remplie à un stade précoce.

Après avoir vu les trois algorithmes utilisés dans le calcul du Syline, nous pourrons déduire que l'algorithme BNL est sans doute le plus simple et le plus facile à implémenté. En plus, il a prouvé être plus efficace pour éliminer les tuples que l'algorithme de D & C, car il est basé sur des maximaux globaux plutôt que locaux. En effet, à chaque passage

de l'algorithme BNL, de nouveaux tuples non dominés sont ajoutés à la liste Skyline. Ces tuples ont été comparés à tous les autres non dominés observés jusqu'à présent, ce qui en fait des maximaux globaux par rapport aux tuples examinés. Il est aussi plus efficace pour éliminer les tuples à venir que les maximaux locaux calculés récursivement dans l'algorithme D & C, comme expliqué dans (Godfrey *et al.*, 2005).

Pour cette raison, nous nous somme basé sur un algorithme Skyline calculé par le BNL dans le système de recherche et de sélection de services Cloud (CSRSS) utilisé dans le chapitre 3

Annexe

B

Méthodes ELECTRE

Les méthodes ELECTRE (pour le terme anglais : Elimination and Choice Translating Reality) sont les méthodes de surclassement les plus connues et les plus utilisées. Ils ont été introduits pour la première fois en 1965 (Figueira *et al.*, 2005), alors que les chercheurs travaillaient sur la sélection de nouvelles activités pour une compagnie. La première méthode construite était MARSAN (pour le terme anglais : Method of Analysis, Research and Selection of New Activities) est une méthode d'analyse, de recherche et de sélection de nouvelles activités (Laffy, 1966). Cependant, ce dernier a montré certaines limitations (Roy et Vanderpooten, 1996) et la méthode ELECTRE I a été construite (Benayoun *et al.*, 1966) pour les pallier.
Après cela, un nouveau type de problèmes de prise de décision est apparu : le classement des options du meilleur au pire. Cela a conduit à la construction d'une méthode dédiée à cet effet : la méthode ELECTRE II (Abgueguen, 1971). Mais pour être mieux adaptés aux problèmes de la vie réelle, il fallait tenir compte de l'incertitude, de l'imprécision et de l'indétermination. Cela a conduit à la naissance d'ELECTRE III (Roy, 1978) où des pseudo-critères, des seuils et des méthodes de surclassement binaire flou ont été introduits. Par la suite, la méthode ELECTRE IV a été construite pour traiter un nouveau problème lié au réseau de métro de Paris (Hugonnard et Roy, 1982). Sa principale particularité était de permettre le classement des actions sans l'utilisation de coefficients d'importance.
La version la plus récente des méthodes ELECTRE est la méthode ELECTRE TRI (Yu, 1992). Cependant, les méthodes ELECTRE continuent d'évoluer et font l'objet de nombreuses nouvelles contributions (Figueira *et al.*, 2013).
ELECTRE I a également évolué vers d'autres versions telles que ELECTRE Iv (Figueira *et al.*, 2005), qui prend en compte le seuil de veto, et ELECTRE IS (Roy et Skalka, 1985), méthode actuellement recommandée par ELECTRE pour résoudre les problèmes de choix. Ainsi, de nombreuses méthodes ELECTRE ont été développées, chacune pour traiter un problème spécifique.

Pour pouvoir utiliser les méthodes ELECTRE pour résoudre un problème multi-critères d'aide à la décision (MCDA pour Multi Criteria Decision Aiding), ce dernier doit inclure au moins trois critères et satisfaire à au moins l'une des caractéristiques suivantes (Roy, 1991) :

— les alternatives sont évaluées sur une échelle ordinale (Robert, 1979) ou à faible intervalle (Martel et Roy, 2006) pour au moins un critère ;
— Les évaluations des critères sont hétérogènes, ce qui rend difficile leur agrégation à une échelle unique ;
— les procédures d'agrégation utilisées ne doivent pas permettre la compensation entre critères ;
— L'utilisation, pour au moins un critère, de seuils de discrimination, traduisant le fait que de petites différences d'évaluation ne sont pas significatives de préférence, tandis que l'accumulation de nombreuses petites différences peut devenir significative de préférence.

L'objectif du travail de recherche et de sélection sur lequel nous nous sommes basé dans dans le chapitre 3 est de satisfaire des conditions, C'est pour ça que la méthode choisi dans le CSRSS (Abourezq et Idrissi, 2014c) qui a fait objet de notre étude comparative, est la méthode ELECTRE IS.
Comme mentionné ci-dessus, ELECTRE IS est une généralisation de ELECTRE I. Ainsi ; nous commençons par présenter ce dernier.

B.1 ELECTRE IS

ELECTRE I est la méthode ELECTRE la plus ancienne et la plus simple. Il est utilisé dans les problèmes de choix et exige que tous les critères soient exprimés en échelles numériques avec les mêmes plages. Considérant un ensemble Alt de m alternatives et un ensemble F de n critères, nous définissons pour chaque critère cr une fonction d'évaluation g_{cr} et un poids k_{cr} traduisant son importance dans le processus de prise de décision.
Pour affirmer qu'une alternative a surpassé une alternative b, notée $a\ S\ b$, nous devons d'abord calculer l'indice de concordance $C(a,b)$ et l'indice de discordance $D(a,b)$ (Tsoukias *et al.*, 2002). L'indice de concordance mesure la pertinence de l'affirmation aSb comme suit :

$$C(a,b) = \frac{1}{\sum_{cr=1}^{n} k_{cr}} \sum_{cr=1} k_{cr} \ \forall\ cr\ such\ as\ g_{cr}(a) \geqslant g_{cr}(b) \tag{B.1}$$

Cet indice est définit entre 0 et 1.
De même, l'indice de discordance mesure l'opposition à l'affirmation aSb comme suit :

$$Si\ \forall\ cr\ g_{cr}(a) \geqslant g_{cr}(b)\ alors\ D(a,b) = 0,$$
$$Sinon\ alors\ D(a,b) = \tfrac{1}{\delta} max_{cr}[g_{cr}(b) - g_{cr}(a)]$$

Où δ est la différence maximale concernant le même critère pour deux alternatives données.
La relation de surclassement S est construite en comparant les indices de concordance et de discordance à un seuil de concordance défini c' et à un seuil de discordance défini d'

comme suit :

$$a\ S\ b \Leftrightarrow C(a,b) \geqslant c'\ et\ D(a,b) \leqslant d' \tag{B.2}$$

Cependant, les problèmes du monde réel se compose souvent d'un ensemble hétérogène de critères qui peuvent être contradictoires et qui ne peuvent pas toujours être exprimés en échelles numériques. De plus, il faut tenir compte de l'imprécision, de l'incertitude et de la mauvaise détermination. Pour répondre à ces besoins, la méthode ELECTRE IS a été définie.

ELECTRE IS est une généralisation d'ELECTRE I car il intègre l'utilisation de pseudo-critères (Nafi et Werey, 2009) au lieu de critères réels et introduit l'utilisation du seuil de veto v_{cr}, du seuil de préférence p_{cr} et du seuil d'indifférence q_{cr}. De plus, les conditions de concordance et de discordance changent et cette dernière est appelée la condition de non veto.
Un pseudo-critère est une fonction g_{cr} associée à deux fonctions de seuil : le seuil d'indifférence q_{cr} et le seuil de préférence p_{cr}.
Le seuil de préférence p_{cr} est une fonction à valeur réelle telle que, pour deux alternatives a et b dans Alt, $p_{cr}(g_{cr}(a))$ est la valeur minimale positive d'une différence de score de type $g_{cr}(a) - g_{cr}(b)$ cela pourrait être compatible avec la préférence de a sur b.
Le seuil d'indifférence q_{cr} est une fonction à valeurs réelles telle que, pour deux alternatives a et b dans Alt, $q_{cr}(g_{cr}(a))$ est la valeur maximale d'une différence de score de type $g_{cr}(b) - g_{cr}(a)$ qui pourrait être compatible avec l'indifférence entre a et b.
Le seuil de veto vcr est attribué au critère cr et permet de définir la valeur au-delà de laquelle la discordance relative à l'affirmation $a\ S\ b$ ne peut permettre la validation du surclassement. En d'autres termes, pour valider l'affirmation $a\ S\ b$, aucun critère parmi les critères discordants ne devrait mettre son veto.
La principale différence entre le seuil de veto vcr et l'indice de discordance D réside dans le fait que vcr est corrélé aux différences de préférence entre $g_{cr}(a)$ et $g_{cr}(b)$ alors que D est corrélé à l'évaluation absolue d'un critère g_{cr} pour une action a de Alt. De plus, le seuil de veto n'est pas nécessairement associé à tous les critères.
Il est supposé que $\forall\ x\ \in\ \mathbb{R}\ q_{cr}(x) \leqslant p_{cr}(x) \leqslant v_{cr}(x)$.
L'indice de concordance est formalisé comme suit :

$$C(a,b) = \sum_{cr=1}^{n} w_{cr} c_{cr}(a,b) \tag{B.3}$$

Où, $w_{cr} = \dfrac{k_{cr}}{\sum_{cr=1}^{n} k_{cr}}$ et $c_{cr}(a,b)$ est définit comme suit :

$$\begin{aligned}
&Si\ g_{cr}(b) - g_{cr}(a) > p_{cr}(g_{cr}(a))\ alors\ c_{cr}(a,b) = 0,\\
&Sinon\ Si\ g_{cr}(b) - g_{cr}(a) \leqslant q_{cr}(g_{cr}(a))\ alors\ c_{cr}(a,b) = 1,\\
&Sinon\ c_{cr}(a,b) = \frac{p_{cr}(g_{cr}(a) + g_{cr}(b) - g_{cr}(a)}{p_{cr}(g_{cr}(a)) - q_{cr}(g_{cr}(a))}
\end{aligned}$$

Ainsi, pour valider l'affirmation $a\ S\ b$, il faut que $C(a,b) \geqslant c'$ et que :

$g_{cr}(a) + v_{cr}(g_{cr}(a)) \geqslant g_{cr}(b) + q_{cr}(g_{cr}(b))\eta_{cr}$.

Cette dernière est dite condition du non veto, où $\eta_{cr} = \dfrac{1 - C(a,b) - w_{cr}}{1 - c' - w_{cr}}$.

BIBLIOGRAPHIE

ABGUEGUEN, R. (1971). *La sélection des supports de presse*. R. Laffont.

ABIN, A. A. et BEIGY, H. (2015). Active constrained fuzzy clustering : A multiple kernels learning approach. *Pattern Recognition*, 48(3):953–967.

ABOUREZQ, M. et IDRISSI, A. (2014a). A cloud services research and selection system. *In International Conference on Multimedia Computing and Systems (ICMCS'14)*, pages 1195–1199. IEEE.

ABOUREZQ, M. et IDRISSI, A. (2014b). A cloud services research and selection system. *In Proceedings of the IEEE 4th International Conference on Multimedia Computing and Systems (ICMCS'14)*.

ABOUREZQ, M. et IDRISSI, A. (2014c). Introduction of an outranking method in the cloud computing research and selection system based on the skyline. *In Proceedings of the IEEE 8th International Conference on Research Challenges in Information Science*.

ABOUREZQ, M. et IDRISSI, A. (2015a). Integration of Qos aspects in the cloud service research and selection system. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 6(6).

ABOUREZQ, M. et IDRISSI, A. (2015b). Integration of Qos aspects in the cloud service research and selection system. *International Journal of Advanced Computer Science and Applications*, 6(6).

ACRONYMS DICTIONARY (2015). Acronyms dictionary. `http://www.netlingo.com/acronyms.php`. (accessed 23 october 2017).

AL SHALABI, L., SHAABAN, Z. et KASASBEH, B. (2006). Data mining : A preprocessing engine. *Journal of Computer Science*, 2(9):735–739.

AL-SHARUEE, M. T., LIU, F. et PRATAMA, M. (2018). Sentiment analysis : An automatic contextual analysis and ensemble clustering approach and comparison. *Data & Knowledge Engineering*, 115:194–213.

ALBAYRAK, S. (2003). Unsupervised clustering methods for medical data : an application to thyroid gland data. *In Artificial Neural Networks and Neural Information Processing ?ICANN/ICONIP 2003*, pages 695–701. Springer.

ALIMOGLU, F., DOC, D., ALPAYDIN, E. et DENIZHAN, Y. (1996). Combining multiple classifiers for pen-based handwritten digit recognition.

ANKERST, M., BREUNIG, M. M., KRIEGEL, H.-P. et SANDER, J. (1999). Optics : ordering points to identify the clustering structure. *In ACM Sigmod Record*, volume 28, pages 49–60. ACM.

ANUMOL BABU, R. V. P. (2016). Efficient density based clustering of tweets and sentimental analysis based on segmentation. *International Journal of Computer Techniques*, 3(3):53–57.

ASGHAR, M. Z., KUNDI, F. M., AHMAD, S., KHAN, A. et KHAN, F. (2018). T-saf : Twitter sentiment analysis framework using a hybrid classification scheme. *Expert Systems*, 35(1).

BENAYOUN, R., ROY, B. et SUSSMAN, B. (1966). Electre : Une méthode pour guider le choix en présence de points de vue multiples. *Note de travail*, 49.

BERKHIN, P. (2006). A survey of clustering data mining techniques. *In Grouping multidimensional data*, pages 25–71. Springer.

BIRANT, D. et KUT, A. (2007). St-dbscan : An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60(1):208–221.

BÖRZSÖNYI, S., KOSSMANN, D. et STOCKER, K. (2001). The skyline operator, intern. *In Conference on Data Engineering (ICDE), Heidelberg, Germany.*

CAI, X., NIE, F. et HUANG, H. (2013). Multi-view k-means clustering on big data. *In Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 2598–2604. AAAI Press.

ČERNỲ, V. (1985). Thermodynamical approach to the traveling salesman problem : An efficient simulation algorithm. *Journal of optimization theory and applications*, 45(1): 41–51.

CHARALAMPAKIS, B., SPATHIS, D., KOUSLIS, E. et KERMANIDIS, K. (2016). A comparison between semi-supervised and supervised text mining techniques on detecting irony in greek political tweets. *Engineering Applications of Artificial Intelligence*, 51:50–57.

COMER, D. (1979). Ubiquitous b-tree. *ACM Computing Surveys (CSUR)*, 11(2):121–137.

CROWDFLOWER (2015). airline-twitter-sentiment. `https://www.crowdflower.com/data/airline-twitter-sentiment/`. (accessed 29 march 2018).

DAVIES, D. L. et BOULDIN, D. W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2):224–227.

DEMPSTER, A. P., LAIRD, N. M. et RUBIN, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.

DEY, D., SLOMKA, P. J., LEESON, P., COMANICIU, D., SHRESTHA, S., SENGUPTA, P. P. et MARWICK, T. H. (2019). Artificial intelligence in cardiovascular imaging : Jacc state-of-the-art review. *Journal of the American College of Cardiology*, 73(11):1317–1335.

DING, C. et HE, X. (2002). Cluster merging and splitting in hierarchical clustering algorithms. *In IEEE International Conference on Data Mining*, pages 139–146.

DING, S., JIA, H., ZHANG, L. et JIN, F. (2014). Research of semi-supervised spectral clustering algorithm based on pairwise constraints. *Neural Computing and Applications*, 24(1):211–219.

DUNN, J. C. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104.

ESPOSITO, F., MALERBA, D. et SEMERARO, G. (1994). Multistrategy learning for document recognition. *International Journal of Applied Artificial Intelligence*, 8(1):33–84.

ESTER, M., KRIEGEL, H.-P., SANDER, J. et XU, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *In Kdd*, volume 96, pages 226–231.

EXLAMATION (2015). Exclamation dictionary. `http://www.vidarholen.net/contents/interjections/`. (accessed 23 october 2017).

FAHAD, A., ALSHATRI, N., TARI, Z., ALAMRI, A., KHALIL, I., ZOMAYA, A. Y., FOUFOU, S. et BOURAS, A. (2014). A survey of clustering algorithms for big data : Taxonomy and empirical analysis. *IEEE Transactions on Emerging Topics in Computing*, 2(3):267–279.

FIGUEIRA, J., MOUSSEAU, V. et ROY, B. (2005). Electre methods. *In Multiple criteria decision analysis : State of the art surveys*, pages 133–153. Springer.

FIGUEIRA, J. R., GRECO, S., ROY, B. et SŁOWIŃSKI, R. (2013). An overview of electre methods and their recent extensions. *Journal of Multi-Criteria Decision Analysis*, 20(1-2):61–85.

FISHER, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

GAN, G., MA, C. et WU, J. (2007). *Data clustering : theory, algorithms, and applications*, volume 20. Siam.

GO, A., BHAYANI, R. et HUANG, L. (2009). Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12).

GODFREY, P., SHIPLEY, R. et GRYZ, J. (2005). Maximal vector computation in large data sets. *In Proceedings of the 31st international conference on Very large data bases*, pages 229–240. VLDB Endowment.

GREFENSTETTE, J. J. (2013). *Genetic Algorithms and Their Applications : Proceedings of the Second International Conference on Genetic Algorithms*. Psychology Press.

HAN, S.-M., HASSAN, M. M., YOON, C.-W. et HUH, E.-N. (2009). Efficient service recommendation system for cloud computing market. *In Proceedings of the 2nd international conference on interaction sciences : information technology, culture and human*, pages 839–845. ACM.

HE, M., HORNG, S.-J., FAN, P., RUN, R.-S., CHEN, R.-J., LAI, J.-L., KHAN, M. K. et SENTOSA, K. O. (2010). Performance evaluation of score level fusion in multimodal biometric systems. *Pattern Recognition*, 43(5):1789–1800.

HE, Y., TAN, H., LUO, W., FENG, S. et FAN, J. (2014). Mr-dbscan : a scalable mapreduce-based dbscan algorithm for heavily skewed data. *Frontiers of Computer Science*, 8(1): 83–99.

HE, Y., TAN, H., LUO, W., MAO, H., MA, D., FENG, S. et FAN, J. (2011). Mr-dbscan : An efficient parallel density-based clustering algorithm using mapreduce. *In Parallel and Distributed Systems (ICPADS), 2011 IEEE 17th International Conference on*, pages 473–480. IEEE.

HERNANDEZ, A., VILA, N., KUSTER, I. et RODRIGUEZ, C. (2019). Clustering spanish alcoholic beverage shoppers to focus marketing strategies. *International Journal of Wine Business Research*, 31(3):362–384.

HINNEBURG, A. et GABRIEL, H. H. (2007). Denclue 2.0 : Fast clustering based on kernel density estimation. *In Advances in Intelligent Data Analysis VII*, pages 70–80. Springer.

HINNEBURG, A. et KEIM, D. A. (1998). An efficient approach to clustering in large multimedia databases with noise. *In KDD*, volume 98, pages 58–65.

HOGENBOOM, A., BAL, D., FRASINCAR, F., BAL, M., DE JONG, F. et KAYMAK, U. (2015). Exploiting emoticons in polarity classification of text. *Journal of Web Engineering*, 14(1&2):22–40. Available on-line at :https ://people.few.eur.nl/hogenboom/files/EmoticonSentimentLexicon.zip.

HOLLAND, J. H. (1975). Adaptation in natural and artificial system : an introduction with application to biology, control and artificial intelligence. *Ann Arbor, University of Michigan Press*.

HOLLAND, J. H. (1992). *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.

HOPKINS, M., REEBER, E., FORMAN, G. et SUERMONDT, J. (1999). Spam base dataset. *Hewlett-Packard Labs*.

HOUT, M. C., PAPESH, M. H. et GOLDINGER, S. D. (2013). Multidimensional scaling. *Wiley Interdisciplinary Reviews : Cognitive Science*, 4(1):93–103.

HRIDOY, S. A. A., EKRAM, M. T., ISLAM, M. S., AHMED, F. et RAHMAN, R. M. (2015). Localized twitter opinion mining using sentiment analysis. *Decision Analytics*, 2(1):8.

HUANG, H., MENG, F., ZHOU, S., JIANG, F. et MANOGARAN, G. (2019). Brain image segmentation based on fcm clustering algorithm and rough set. *IEEE Access*, 7:12386–12396.

HUGONNARD, J.-C. et ROY, B. (1982). Le plan d'extension du metro en banlieue parisienne, un cas typique d'application de l'analyse multicritere. *Cahiers Scientifiques de la Revue Transports*, (6).

IDRISSI, A. et ABOUREZQ, M. (2014). Skyline in cloud computing. *Journal of Theoretical and Applied Information Technology*, 60(3).

IDRISSI, A., REHIOUI, H., LAGHRISSI, A. et RETAL, S. (2015). An improved denclue algorithm for data clustering. *In IEEE 2015 International Conference on Information and Communication Technology and Accessibility (ICTA'15)*. IEEE.

IDRISSI, A. et ZEGRARI, F. (2015). A new approach for a better load balancing and a better distribution of resources in cloud computing. *International Journal of Advanced Computer Science and Applications*, 6(10).

JAGLI, D., PUROHIT, S. et NALLA, S. C. (2016). Implementation of k-means clustering for evaluating saas on the cloud computing environment. *In International Conference on ICT in Business Industry & Government (ICTBIG'16)*, pages 1–5. IEEE.

JAIN, A. K. (2010). Data clustering : 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666.

KARIM, R., DING, C., MIRI, A. et RAHMAN, M. S. (2016). Incorporating service and user information and latent features to predict qos for selecting and recommending cloud service compositions. *Cluster Computing*, pages 1–16.

KARYPIS, G., HAN, E.-H. S. et KUMAR, V. (1999). Chameleon : Hierarchical clustering using dynamic modeling. *Computer*, (8):68–75.

KHAN, S. et BIANCHI, T. (2019). Reduced complexity image clustering based on camera fingerprints. *In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019*, pages 2682–2688. IEEE.

KHANMOHAMMADI, S., ADIBEIG, N. et SHANEHBANDY, S. (2017). An improved overlapping k-means clustering method for medical applications. *Expert Systems with Applications*, 67:12–18.

KIRKPATRICK, S., GELATT, C. D., VECCHI, M. P. *et al.* (1983). Optimization by simulated annealing. *science*, 220(4598):671–680.

KOTSIANTIS, S. B., ZAHARAKIS, I. et PINTELAS, P. (2007). Supervised machine learning : A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24.

KUMARI, S. S. et BABU, G. A. (2016). Sentiment on social interactions using linear and non-linear clustering. *In 2nd International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*, pages 177–181. IEEE.

KUNG, H.-T., LUCCIO, F. et PREPARATA, F. P. (1975). On finding the maxima of a set of vectors. *Journal of the ACM (JACM)*, 22(4):469–476.

LAFFY, R. (1966). La méthode marsan pour la recherche de produits nouveaux. *In ESOMAR congress, Copenhagen*.

LEE, J. (2015). A method of color image segmentation based on dbscan (density based spatial clustering of applications with noise) using compactness of superpixels and texture information. *Journal of the Korea Society of Digital Industry and Information Management*, 11(4):89–97.

LICHMAN, M. (2013). UCI machine learning repository.

LIU, B., HU, M. et CHENG, J. (2005). Opinion observer : analyzing and comparing opinions on the web. *In Proceedings of the 14th international conference on World Wide Web*, pages 342–351. ACM. Available on-line at :https ://github.com/jeffreybreen/twitter-sentiment-analysis-tutorial-201107/tree/master/data/opinion-lexicon-English.

LIU, Y., LI, Z., XIONG, H., GAO, X., WU, J. et WU, S. (2013). Understanding and enhancement of internal clustering validation measures. *IEEE Transactions on Cybernetics*, 43(3):982–994.

MACQUEEN, J. *et al.* (1967). Some methods for classification and analysis of multivariate observations. *In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

MARTEL, J.-M. et ROY, B. (2006). Analyse de la signifiance de diverses procédures d'agrégation multicritère. *INFOR : Information Systems and Operational Research*, 44(3):191–215.

MARWAN, M., KARTIT, A. et OUAHMANE, H. (2018). Genetic k-means clustering algorithm for achieving security in medical image processing over cloud. *In International Conference on Advanced Intelligent Systems for Sustainable Development*, pages 140–145. Springer.

MEGER, N., LESCHI, C., LUCAS, N. et RIGOTTI, C. (2004). Mining episode rules in stulong dataset. *In Proceedings of PKDD'04 Discovery Challenge A Collaborative Effort in Knowledge Discovery, Pisa, Italy*, pages 1–12.

METROPOLIS, N., ROSENBLUTH, A. W., ROSENBLUTH, M. N., TELLER, A. H. et TELLER, E. (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

MOHAMMED, M. F. et LIM, C. P. (2017). Improving the fuzzy min-max neural network with a k-nearest hyperbox expansion rule for pattern classification. *Applied Soft Computing*, 52:135–145.

MOSTAFA, M. M. (2019). Clustering halal food consumers : A twitter sentiment analysis. *International Journal of Market Research*, 61(3):320–337.

NAFI, A. et WEREY, C. (2009). Aide à la décision multicritère : introduction aux méthodes d'analyse multicritère de type electre. *Module d'ingénierie financière, ENGEES*, 2010.

NEUTRAL (2015). Neutral dictionary. `https://quizlet.com/25024874/ap-literature-neutral-tone-attitude-words-flash-cards/`. (accessed 23 october 2017).

PANDEY, A. C., RAJPOOT, D. S. et SARASWAT, M. (2017). Twitter sentiment analysis using hybrid cuckoo search method. *Information Processing & Management*, 53(4):764–779.

PARZEN, E. (1962). On estimation of a probability density function and mode. *The annals of mathematical statistics*, pages 1065–1076.

PAUL, R. et HOQUE, A. S. M. L. (2010). Clustering medical data to predict the likelihood of diseases. *In Fifth International Conference on Digital Information Management 2010 (ICDIM'10)*, pages 44–49. IEEE.

PHAM, D. et KARABOGA, D. (2012). *Intelligent optimisation techniques : genetic algorithms, tabu search, simulated annealing and neural networks*. Springer Science & Business Media.

PREPARATA, F. P. et SHAMOS, M. I. (1985). Computational geometry. texts and monographs in computer science. *Berlin, Springer-Verlag*.

PUNJ, G. et STEWART, D. W. (1983). Cluster analysis in marketing research : Review and suggestions for application. *Journal of marketing research*, 20(2):134–148.

RAMESH, D. et KUMARI, K. (2018). Debc-gm : Denclue based gaussian mixture approach for big data clustering. *In Proceeding of 2018 IEEE International Conference on Current Trends towards Converging Technologies (ICCTCT)*, pages 1–8. IEEE.

RENDÓN, E., ABUNDEZ, I., ARIZMENDI, A. et QUIROZ, E. (2011). Internal versus external cluster validation indexes. *International Journal of computers and communications*, 5(1):27–34.

RIAZ, S., FATIMA, M., KAMRAN, M. et NISAR, M. W. (2017). Opinion mining on large scale data using sentiment analysis and k-means clustering. *Cluster Computing*, pages 1–16.

ROBERT, F. S. (1979). *Measurement theory with applications to decision-making, utility and the social sciences*.

ROSENBLATT, M. *et al.* (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837.

ROY, B. (1978). Electre iii : Un algorithme de classement fondé sur une représentation floue des préférences en présence de critères multiples. *Cahiers du CERO*, 20(1):3–24.

ROY, B. (1991). The outranking approach and the foundations of the electre methods theory and decision, 31.

ROY, B. et SKALKA, J. (1985). Electre is : Aspécts methodologiques et guide dutilization. document du lamsade. *Paris-Dauphine : Universite Pauris-Dauphine*, 30:1–125.

ROY, B. et VANDERPOOTEN, D. (1996). The european school of mcda : Emergence, basic features and current works. *Journal of Multi-Criteria Decision Analysis*, 5(1):22–38.

SANDERS (2011). Sanders dataset. `http://www.sananalytics.com/lab/twitter-sentiment/`. (accessed 23 october 2017).

SCHOIER, G. et BORRUSO, G. (2017). A methodology for dealing with spatial big data. *International Journal of Business Intelligence and Data Mining*, 12(1):1–13.

SHAH, G. H., BHENSDADIA, C. et GANATRA, A. P. (2012). An empirical evaluation of density-based clustering techniques. *International Journal of Soft Computing and Engineering (IJSCE)*, pages 2231–2307.

SHELKE, N. M., DESHPANDE, S. et THAKRE, V. (2017). Exploiting expectation maximization algorithm for sentiment analysis of product reviews. *In Inventive Communication and Computational Technologies (ICICCT), 2017 International Conference on*, pages 390–396. IEEE.

SOLTANI, S., MARTIN, P. et ELGAZZAR, K. (2014). Quaram recommender : Case-based reasoning for iaas service selection. *In Cloud and Autonomic Computing (ICCAC), 2014 International Conference on*, pages 220–226. IEEE.

SONG, J., KIM, K. T., LEE, B., KIM, S. et YOUN, H. Y. (2017). A novel classification approach based on naïve bayes for twitter sentiment analysis. *KSII Transactions on Internet and Information Systems (TIIS)*, 11(6):2996–3011.

STANFORD (2009). Testdata-manual-2009.06.14. `http://help.sentiment140.com/for-students/`. (accessed 23 october 2017).

STOJANOVSKI, D., CHORBEV, I., DIMITROVSKI, I. et MADJAROV, G. (2016). Social networks vgi : Twitter sentiment analysis of social hotspots. *European Handbook of Crowdsourced Geographic Information*, pages 223–235.

STOPWORDS (2014). Stopwords dictionary. `https://github.com/igorbrigadir/stopwords/blob/master/en/t101_minimal.txt`. (accessed 23 october 2017).

SUN, L., MA, J., ZHANG, Y., DONG, H. et HUSSAIN, F. K. (2016). Cloud-fuser : Fuzzy ontology and mcdm based cloud service selection. *Future Generation Computer Systems*, 57:42–55.

TIAN, G.-L., JU, D., CHUEN YUEN, K. et ZHANG, C. (2018). New expectation–maximization-type algorithms via stochastic representation for the analysis of truncated normal data with applications in biomedicine. *Statistical methods in medical research*, 27(8):2459–2477.

TSOUKIAS, A., PERNY, P. et VINCKE, P. (2002). From concordance/discordance to the modelling of positive and negative reasons in decision aiding. *In Aiding decisions with multiple criteria*, pages 147–174. Springer.

TWITTER (2014). Twitter dataset. `https://drive.google.com/file/d/0BwPSGZHAP_yoN2pZcVl1Qmp1OEU/view`. (accessed 23 october 2017).

WAGSTAFF, K., CARDIE, C., ROGERS, S., SCHRÖDL, S. *et al.* (2001). Constrained k-means clustering with background knowledge. *In ICML*, volume 1, pages 577–584.

WANG, W.-T., WU, Y.-L., TANG, C.-Y. et HOR, M.-K. (2015). Adaptive density-based spatial clustering of applications with noise (dbscan) according to data. *In 2015 International Conference on Machine Learning and Cybernetics (ICMLC)*, volume 1, pages 445–451. IEEE.

WANG, Y., KIM, K., LEE, B. et YOUN, H. Y. (2018). Word clustering based on pos feature for efficient twitter sentiment analysis. *Human-centric Computing and Information Sciences*, 8(17):1–25.

WESTON, J., ZHOU, D., ELISSEEFF, A., NOBLE, W. S. et LESLIE, C. S. (2004). Semi-supervised protein classification using cluster kernels. *In Advances in neural information processing systems*, pages 595–602.

XIAO, W., YANG, Y., WANG, H., LI, T. et XING, H. (2016). Semi-supervised hierarchical clustering ensemble and its application. *Neurocomputing*, 173:1362–1376.

XING, Z., PEI, J. et KEOGH, E. (2010). A brief survey on sequence classification. *ACM Sigkdd Explorations Newsletter*, 12(1):40–48.

XU, R., WUNSCH, D. *et al.* (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678.

YANG, A. Y., WRIGHT, J., MA, Y. et SASTRY, S. S. (2008). Unsupervised segmentation of natural images via lossy data compression. *Computer Vision and Image Understanding*, 110(2):212–225.

YANG, X., WANG, Y. et QIAO, W. (2016). Social network analysis on sina weibo based on k-means algorithm. *In IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA'16)*, pages 127–132. IEEE.

YEH, I.-C. et LIEN, C.-h. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2):2473–2480.

YU, W. (1992). Electre tri(aspects méthodologiques et manuel d'utilisation). *Document-Université de Paris-Dauphine, LAMSADE.*

ZAIN, T., ASLAM, M., IMRAN, M. et MARTINEZ-ENRIQUEZ, A. (2014). Cloud service recommender system using clustering. *In Electrical Engineering, Computing Science and Automatic Control (CCE), 2014 11th International Conference on*, pages 1–6. IEEE.

ZAKI, M. J. et MEIRA JR, W. (2014). *Data mining and analysis : fundamental concepts and algorithms.* Cambridge University Press.

Zhou, Y., Zuo, H.-f. et Feng, J. (2015). A clustering algorithm based on feature weighting fuzzy compactness and separation. *Algorithms*, 8(2):128–143.

Zhu, Y., Ting, K. M. et Carman, M. J. (2016). Density-ratio based clustering for discovering clusters with varying densities. *Pattern Recognition*, 60:983–997.

Résumé

Actuellement, la science de données est un axe de recherche en plein essor grâce à la grande quantité de données générées quotidiennement par les différents moyens technologiques. Cet axe vise à extraire les informations pertinentes à partir des données brutes. Une description en amont de ces données est souvent indisponible y compris les classes des échantillons. Par conséquent, il est plus judicieux d'adopter des méthodes appropriées, en l'occurrence la classification non supervisée (dite Clustering en anglais) qui consiste à regrouper les données sous forme de classes homogènes appelées Clusters. Dans la présente thèse, nous nous sommes intéressés à l'amélioration de l'algorithme de clustering DENCLUE qui appartient à la famille de méthodes basées sur la densité. Cet algorithme a déjà prouvé sa robustesse surtout dans le cas des données bruitées multi-dimensionnelles. Cependant, il n'est pas assez performant en termes de temps d'exécution en particulier pour classifier une grande quantité de données. Pour remédier à cela, nous avons proposé trois nouvelles améliorations de DENCLUE qui ont montré leur performance à trouver un bon compromis entre le temps d'exécution et la qualité du clustering. En plus des améliorations considérables apportées, notre analyse de résultats nous a conduit à la détection d'un problème de chevauchement entre les clusters obtenus dans certains ensembles de données. Pour répondre à ce problème, nous avons proposé une mise en échelle des données en se basant sur leurs distributions de densités. Les résultats quantitatifs et visuels se sont avérés plus intéressants prouvant ainsi le grand intérêt de la méthode proposée. La deuxième partie de nos contributions s'est focalisée sur l'application de nos algorithmes tout en les adaptant à des domaines bien spécifiques, notamment la recherche et la sélection des services dans le Could Computing, l'analyse de sentiments dans le réseau social Twitter, et le cancer du nasopharynx (domaine médical).

Mots clés (6)*: Science de Données, Machine Learning, Clustering, Algorithmes basés sur la Densité, Cloud Computing, Analyse de Sentiments.*

Abstract

Today, the Data Science is considered as growing area of research due to the large amount of data generated daily by different current technologies. This discipline aims to extract the relevant information from the raw data. A beforehand description of these data is often unavailable, especially the labels of the samples. Therefore, it is wise to adopt appropriate methods, such as the unsupervised classification (also called clustering) which consists of grouping the data in homogeneous clusters. In this thesis, we are interested in improving the DENCLUE clustering algorithm that belongs to the family of density-based methods. This algorithm has already proved its robustness especially in the case of multi-dimensional noisy data. However, it is not efficient enough in terms of execution time especially for the classification of a large amount of data. To remedy this, we proposed three new improvements of DENCLUE that showed their performance to find a good trade-off between the execution time and the quality of the clustering. In addition to the considerable made improvements, our results analysis led us to detect the problem of overlapped clusters obtained in some datasets. To overcome this limit, we scaled the data based on their density distributions. The quantitative and visual results proved the efficiency and the great interest of the proposed method. The second part of our contributions focused on the application of our algorithms while adapting them to specific domains, including the research and selection of services in the Could Computing, the sentiment analysis in the social network Twitter, and the nasopharyngeal cancer (medical field).

Keywords (6): Data *Science, Machine Learning, Clustering, Density based Algorithms, Cloud Computing, Sentiment Analysis.*