الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جامعة عمار ثليجي بالاغواط
Université Ammar Telidji Laghouat
كلية العلوم
Faculté des Sciences
قسم الإعلام الآلي
Département d'Informatique

Mémoire de fin d'étude
pour l'obtention de diplôme de Master en informatique

Domaine : Mathématique et Informatique
Filière : Informatique
Option : Systèmes d'Information et de Décision

Réalisé par:

Boughoufala Imane

THÈME

La méthode KNN pour le Big Data

Soutenu le xx/09/2020 , devant le jury composé de :

Dr. CHELLAMA Laaradj	Président
Dr. MAICHA Mohamed El Habib	Examinateur
Pr. OUINTEN Youcef	Encadreur

Nº d'ordre ..... Année universitaire 2019/2020

Remerciements

«Et lorsque votre Seigneur proclama : "Si vous êtes reconnaissants, très certainement J'augmenterai [Mes bienfaits] pour vous»

[Coran S14.V7]

Avant tout, je remercie Allah le très haut qui m'a donné le courage et la volonté de réaliser ce modeste travail.

«(CELUI QUI NE REMERCIE PAS LES GENS, NE REMERCIE PAS ALLAH.)»

[Authentique Hadith]

Que professeur Youcef Ouinten , Ses encouragements, et surtout ses critiques, Sa sensibilisation, ont largement contribué à l'accomplissement de mes travaux. Je le remercie infiniment de m'avoir toujours poussé vers l'avant. Je tiens également à remercier « TOUS » les Messieurs et dames, mes professeurs qui m'ont enseigné durant deux ans de formation master en Informatique, pour leurs précieux conseils et ses orientations.

Mes remerciements vont également aux membres du jury d'avoir accepté d'évaluer mon travail. Sans oublier de remercier mes amis et mes collègues (de l'université ou dans le monde Virtuel « internet ») qui, tous d'une manière différente, ont contribué à ce que je puisse aboutir à la réalisation de ce mémoire.

Enfin, merci à ma famille pour le soutien et l'encouragement qu'ils m'ont apporté tout au long de mon travail.

Dédicaces

Avant tout, je remercie ALLAH le tout puissant de m'avoir donné le courage et la patience pour réaliser ce travail malgré toutes les difficultés rencontrées.
Tous d'abord Je dédie ce modeste travail
À mes très cher parents qui m'ont
encouragés par tous les moyens à avoir la confiance et l'espoir
au cours de la rédaction de ce travail.
Je le dédie aussi À ma très cher sœur *RYMA* ,*KHADIDJA*,ET *WISSAM*.
À mon très cher frère *MOHAMMED LARBI*.
Et à toute la famille.
À mes meilleurs cousines *NADJET* et *MARIA* .
À mes meilleurs amies *IMANE* et *NADJET* .
À tous les étudiants de la faculté Informatique surtout les étudiants
de la 2 ème année master promotion 2020.

IMANE

ملخص

االمصنف kNN (مصنف K-Nearest Neighbor) هو طريقة معروفة على نطاق واسع تستخدم في استخراج البيانات. لكن لهذه الطريقة عيباًا كبيراً يكمن في الكمية الكبيرة من الحسابات التي تولدها ومقدار الذاكرة التي تتطلبها. لذلك فإن هذه الطريقة ليست مناسبة على الإطلاق لتطبيقات تحليل البيانات الضخمة. للتجاوز عيوب أساليب التنقيب عن البيانات المعروفة ، تم اقتراح العديد من البدائل البيئية الموزعة. تشمل هذه البدائل نظام Hadoop MapReduceالموزع بالإضافة إلى sparkالتي تجذب اهتماماً كبيراً بشكل متزايد. ستكون مهمتنا جرد العمل على طريقة Knn للبيانات الضخمة ، للتعرف على نظام Hadoop البيئي أو نظام Apache Spark البيئي. الهدف هو تنفيذ إحدى الطرق المدرجة واختبارها على مقياس الأداء.

Abstract

The KNN classifier (K-Nearest Neighbor classifier) is a widely known method used in data mining. But this method has a major drawback which lies in the large amount of computation it generates and the amount of memory it requires. So this method is not at all suitable for big data analysis applications. To circumvent the drawbacks of known data mining methods, several distributed environment alternatives have been proposed. These alternatives include the Hadoop MapReduce distributed ecosystem as well as spark, which is increasingly attracting considerable attention. our task will be to identify work on the kNN method for Big Data, to learn about the Hadoop ecosystem or the Apache Spark ecosystem. The goal is to implement one of the methods identified and to test it on a Benchmark.

Keywords :K-Nearest Neighbor,Big data, classification , Hadoop , Mapreduce

Résumé

Le classifieur KNN (K-Nearest Neighbor classifier) est une méthode largement connue et utilisée dans la fouille de données. Mais cette méthode présente un inconvénient majeur qui réside dans la grande quantité de calcul qu'elle génère et la quantité de mémoire qu'elle nécessite. Ce qui fait que cette méthode n'est pas du tout adaptée aux applications d'analyses de Big Data. Pour contourner les inconvénients des méthodes connues de fouille de données, plusieurs alternatives d'environnement distribuées ont été proposées. Parmi ces alternatives on peut citer l'écosystème distribué Hadoop MapReduce ainsi que spark qui attire de plus en plus une attention considérable. Nous aurons pour tache de recenser les travaux sur la méthode kNN pour le Big Data, s'initier à l'écosystème Hadoop ou l'écosystème d'Apache Spark. Le but étant d'implémenter une des méthodes recensées et de la tester sur un Benchmark.

Mot clés : K-Le plus proche voisins,Big data, classification , Hadoop , Mapreduce

Table des matières

Table des figures

Abréviations

K-NN K Nearest Neighbor
K-PPV Les k Plus Proches Voisins
RC Random Clustering kNN
ACP (Analyse linéaire en Composantes Principales
LSC Landmark-based Spectral Clustering
WDBC The Wisconsin Diagnostic Breast Cancer
HDFS Hadoop Distributed File System
LIBSVM Library for Support Vector Machines
BSD Berkeley Software Distribution

Introduction Générale

A l'ère de la quatrième révolution industrielle, la prise de décision dans entreprises repose largement sur les données récupérées par les appareils connectés à Internet, capables de collecter et de traiter des quantités d'informations toujours plus importantes.

Pour obtenir des prévisions précises, les données provenant de différents environnements (par exemple, différents outils de médias sociaux, entrepôts de données, stockages dans le cloud, etc.) doivent être analysées intelligemment par les entreprises.

L'analyse des données extraites de nombreuses sources de données permet de traiter de grandes quantités de données brutes non structurées, qui sont importantes en termes de volume, de variété et de vitesse d'acquisition. Le processus d'analyse de ces données est récemment un domaine de recherche populaire, connu sous le nom d'analyse de Big Data.

L'une des tâches les plus importantes de l'exploration de données est la classification. La classification, qui consiste à attribuer des objets à l'une des nombreuses catégories prédéfinies, est un problème omniprésent qui englobe de nombreuses applications diverses.

Pour classer les données à l'ère du Big Data, les techniques centralisées présentent les faibles performances de retard de classification, ce qui est vital pour faire face aux flux de données à grande vitesse du Big Data.

Pour faire face à cette exigence de temps de la classification moderne des données, plusieurs écosystèmes distribués ont été testés et utilisés par différentes recherches. L'un de ces écosystèmes distribués est communément appelé Apache Hadoop et MapReduce Framework de Google.

La classification des K les plus proches voisins (K-NN) a été l'un des algorithmes de classification les plus populaires . L'algorithme classique K-NN est basé sur le calcul des distances entre l'instance de données de test à classer et toutes les instances de l'ensemble de données d'apprentissage et sur la recherche du nombre K d'instances d'apprentissage le plus proche. Après avoir détecté le nombre K d'instances d'apprentissage les plus proches, l'algorithme K-NN applique le vote à la majorité qui est le processus de détection de la classe de données avec le nombre maximum d'instances parmi les K instances sélectionnées.

Puisque l'algorithme K-NN classique est entièrement basé sur des proximités d'instances individuelles, il souffre fortement de coûts de calcul élevés. De plus, étant donné que la stratégie de prise de décision de l'algorithme repose sur les proximités des instances individuelles plutôt que sur des représentations de classe plus fortes, la précision de classification de l'algorithme n'est pas non plus adéquate pour l'analyse de Big Data moderne qui nécessite des résultats de classification rapides et précis.

D'un autre côté, la stratégie de distance des instances individuelles de K-NN fait de K-NN un candidat idéal pour la classification des données distribuées, qui est la base pour obtenir des délais de classification suffisamment bas tout en classant le Big Data. Compte tenu de l'adéquation de K-NN aux environnements distribués, de nombreuses études basées sur K-NN qui tentent d'améliorer les performances des algorithmes K-NN et travaillant sur l'environnement Hadoop et MapReduce ont été récemment proposées dans la littérature.
Pour cela on réparti notre travail en trois chapitres, comme suite :
Le premier chapitre débute par une introduction général sur les big data et data mining et les définitions et le langage commun des méthodes de la classification, ainsi que nous parlerons d'une de ses grandes approches « le classement, discrimination ».
Le deuxième chapitre consacré à la deuxième grande approche des méthodes classificatoires, qui est l'approche de notre méthode cible, il s'agit de discussions sur l'un des méthodes supervisées,"KNN" et quelques expérimentations et résultats.
Le chapitre trois quand a lui présente les différents outils qui vont servir a l'implémentation de notre projet, et les discussions sur la réalisation.

Chapitre 1

Généralités

1 Introduction :

" Le seul moyen de faire une méthode instructive et naturelle, est de mettre ensemble les choses qui se ressemblent et de séparer celles qui différent les unes des autres."

M. Georges Buffon, Histoire naturelle,1749

En partant de ce principe, nous présenterons dans ce chapitre tout d'abord quelques concepts de Big data et Data mining et ce que c'est la classification, ses méthodes, techniques, ses grandes approches, domaines d'applications, . . .etc. et on détaillera à la fin une de ses grandes approches en étudiant et analysant deux de ses algorithmes.

2 Big data

1.2.1 Définition

Big data est un terme utilisé pour désigner une collection d'ensembles de données volumineux et complexes, difficiles à stocker et à traiter à l'aide d'outils de gestion de base de données classiques (SGBD) ou d'applications de traitement de données traditionnelles. [16]

Une définition très répondue de big data est celle dite des "5V" du Gartner group
Et s'appuie sur cinq notions :[1]

1. La nature de données (variety).
2. La vitesse (Velocity) à laquelle des données sont produites et évoluent dans le temps.
3. Les volumes (volumes) de données sont collecter, stocker et traiter.
4. incertitude incomplétude des données (veracity).
5. La valeur des données (value).

1.2.2 Type de données de Big data

1. **Structurés :** qui sont stockées dans des bases de données relationnelles ou des datawarehouses d'entreprises.
2. **Semi-structurés :** qui provient de la vitesse d'arrivée des données, de leur volume, rendant impossible la structuration.
3. **Non-structurés :** sont des fichiers plats. C'est la forme abstraite la plus simple, une suite d'octets. Appartiennent à cette catégorie, les données de réseaux sociaux et les e-mails, images, vidéos, fichiers texte etc.

1.2.3 Les défis liés au Big Data

Si le Big Data ouvre des perspectives intéressantes, il n'en présente pas moins certains écueils.
Premièrement, le Big Data est volumineux. Même si de nouvelles technologies ont été mises au point pour le stockage des données, les volumes de données doublent environ tous les deux ans. Les entreprises éprouvent toujours des difficultés à maîtriser leur croissance et à trouver des moyens de les stocker efficacement.

Mais il ne suffit pas de stocker les données. Pour être utiles, celles-ci doivent être exploitées et, en amont, organisées. Des données propres, ou des données pertinentes pour le client et organisées de manière à permettre une analyse significative, nécessitent beaucoup de travail. Les spécialistes des données passent 50 à 80% de leur temps à organiser et à préparer les données avant leur utilisation.
Enfin, la technologie du Big Data évolue rapidement. Il y a quelques années, Apache Hadoop était la technologie la plus utilisée pour traiter le Big Data. Puis, Apache Spark fit son apparition en 2014. Actuellement, l'association des deux infrastructures semble constituer la meilleure approche. Maîtriser la technologie du Big Data est un enjeu continu. [1]

1.2.4 Fonctionnement du Big Data

Le Big Data offre de nouvelles perspectives, qui ouvrent de nouvelles opportunités et favorisent de nouveaux business models. Son adoption implique trois actions principales :

1. **Intégrer** Le Big Data rassemble des données provenant de sources et d'applications disparates. Les mécanismes traditionnels d'intégration de données, tels que ETL (extract, transform, and load - extraire, transformer et charger) ne sont généralement pas à la hauteur de la tâche. Pour analyser des jeux de Big Data à l'échelle de téraoctets, voire de pétaoctets, il est nécessaire d'adopter de nouvelles stratégies et technologies.
   Lors de l'intégration, vous devez importer les données, les traiter et vous assurer qu'elles sont formatées et disponibles sous une forme accessible à vos analystes.
2. **Gérer** Le Big Data nécessite du stockage. Votre solution de stockage peut se trouver dans le cloud, sur site, ou les deux à la fois. Vous pouvez stocker vos données sous la forme de votre choix et imposer à ces jeux de données vos exigences de traitement, ainsi que les moteurs de traitement nécessaires, à la demande. Nombreux sont ceux qui choisissent leur solution de stockage en fonction de l'endroit où sont hébergées leurs données. Le cloud est de plus en plus adopté, car il prend en charge vos besoins informatiques actuels et laisse la possibilité d'augmenter les ressources en fonction des besoins.

3. **Analyser** Votre investissement dans le Big Data porte ses fruits dès lors que vous êtes en mesure d'analyser vos données et d'agir à partir de l'analyse. Forgez-vous un nouveau point de vue grâce à une analyse visuelle de vos divers jeux de données. Explorez davantage les données afin de faire de nouvelles découvertes. Partagez vos conclusions avec d'autres utilisateurs. Créez des modèles de données avec l'apprentissage automatique et l'intelligence artificielle. Exploitez vos données.[1]

3 Data mining

1.3.1 Historique

Le *"Data Mining"* que l'on peut traduire par "fouille de données" apparaît au milieu des années 1990 aux États-Unis comme une nouvelle discipline à l'interface de la statistique et des technologies de l'information : bases de données, intelligence artificielle, apprentissage automatique *"machine learning "*.
« La naissance du data mining est essentiellement due à la conjonction des deux facteurs suivants :

— l'accroissement exponentiel dans les entreprises, de données liés à leur activité (données sur la clientèle, les stocks, la fabrication, la comptabilité .. .) qu'il serait dommage de jeter car elles contiennent des informations-clé sur leur fonctionnement (. . .) stratégiques pour la prise de décision.
— Les progrès très rapides des matériels et des logiciels (...) L'objectif poursuivi par le Data Mining est donc celui de la valorisation des données contenues dans les systèmes d'information des entreprises.

1.3.2 Définition 1

Plusieurs définitions ont été proposées dans [32], le Data Mining serait :

— " la découverte de nouvelles corrélations, tendances et modèles par le tamisage d'un grand nombre de données " ;
— " un processus d'aide à la décision où les utilisateurs cherchent des modèles d'interprétation dans les données" ;
— " l'extraction d'informations originales, auparavant inconnues, potentiellement utiles à partir des données " .

1.3.3 Définition 2

Le Data Mining, ou la fouille de données est l'ensemble des méthodes et techniques destinées à l'exploration et l'analyse de grandes bases de données informatiques, de façon automatique ou semi-automatique, en vue de détecter dans ces données des règles des associations, des tendances inconnues ou cachées, des structures particulières restituant l'essentiel de l'information utile tout en réduisant la quantité de données permettant d'étayer les prises de décision . En bref, le Data Mining est l'art d'extraire des informations (ou même des connaissances) à partir des données.

1.3.4 Type d'applications en data mining

1. **Association :** chercher des patterns au sein desquelles un événement est lié à un autre événement.

2. **Analyse de séquence :** – chercher des patterns au sein desquelles un événement mène à un autre événement plus tardif.

3. **Classification :** chercher de nouvelles patterns, quitte à changer la façon dont les données sont organisées.

4. **Clustering :** – trouver et documenter visuellement des groupes de faits précédemment inconnus.

5. **Prédiction :** découvrir des patterns de données pouvant mener à des prédictions raisonnables sur le futur. Ce type de data mining est aussi connu sous le nom d'analyse prédictive.

4 Classification :

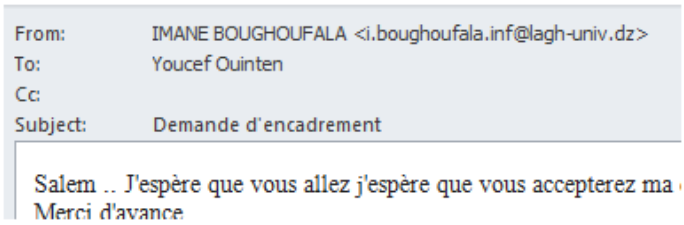
1.4.1 Définition

La classification est une méthode d'analyse des données qui vise à regrouper en classes homogènes un ensemble d'observations. Ces dernières années, les besoins d'analyse de données et en particulier de classification ont augmenté significativement. En effet, de plus en plus de domaines scientifiques nécessitent de catégoriser leurs données dans un but descriptif ou décisionnel.
D'un autre part, la classification est la tâche la plus connus du data mining. Elle consiste à étudier les caractéristiques d'un nouvel objet pour lui attribuer une classe prédéfinie. Les objets à classifier sont généralement des enregistrements d'une base de données. La tâche de classification est caractérisée par une définition des classes bien précises et un ensemble d'exemples classés auparavant.[7]

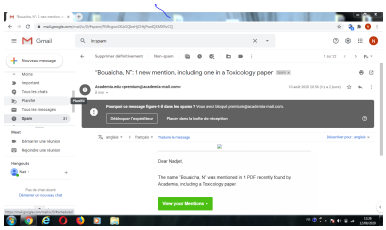
1.4.2 Exemples de problèmes de classification :

Ce sont les domaines que la classification pourrait viser et en même temps ils représentent les différents types de données d'entrées des techniques de classification, ce qui est nécessaire d'en présenter Avant d'aborder les méthodes classificatoires.[6]

1. **Prédiction e-mail / Spam :** Comme le fait de différencier un E-mail valide d'un SPAM , d'ailleurs la classification est fort utile en ce qui concerne la catégorisation des documents sur internet quel que soit la nature du document (image , fichier , son . . .etc) et elle peut même être utilisée pour classifier les document selon leur sens (le web sémantique et les moteurs de recherche où on associe des sens pour les termes et pour les classifier il faut développer un langage de traitement/classification sémantique par exemple à base d'ontologie.) ou tout simplement pour classifier les ouvrages dans le monde des bibliothèques et des archives(le système de Classification de la Bibliothèque du Congrès **LCC**



E-mail valide



E-mail non valide

2. **Reconnaissance de formes :** Généralement c'est une question qui vise à reconnaitre ou identifier certain motifs à partir de données brutes afin de prendre une décision dépendant de la catégorie attribuée à ce motif [11], ces motifs (formes) peuvent s'agir d'une image (visage, empreinte digitale, rayon X, EEG,...) ou sonore (reconnaissance de parole), et bien d'autres. Comme la reconnaissance des caractères manuscrits :

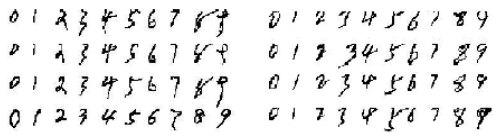


FIGURE 1.1: Reconnaissance des caractère manuscrits

Ou d'empreintes digitales : C'est l'exemple où on cherche à identifier une personne grâce à son empreinte ,

FIGURE 1.2: Reconnaissance d'empreintes digitales

1.4.3 Les étapes d'une classification :

1. Choix des données.
2. Calcul des similarités entre les n individus à partir des données initiales.
3. Choix d'un algorithme de classification et exécution.
4. L'interprétation des résultats :
   — évaluation de la qualité de la classification,
   — description des classes obtenues.

1.4.4 Techniques des datamining :

Les techniques de data mining peuvent être classifiés selon l'information à priori :les techniques supervisées et celles dites non supervisées.

Classification non supervisé

est un regroupement en classes homogènes consistent à représenter un nuage des points d'un espace quelconque en un ensemble de groupes appelé **Cluster**.
C'est un traitement sur un ensemble d'objets qui n'ont pas été étiquetés par un superviseur.

Généralement lié au domaine de l'analyse des données comme ACP (analyse linéaire en composantes principales),ce type de méthodes vise à répondre au problème de : diminution de la dimension de l'espace d'entrée, ou pour le groupement des objets en plusieurs catégories (clusters) non définies à l'avance. Parmi les méthodes qu'on peut trouver dans ce type de classification : les cartes auto-organisatrices de kohonen [17], GMM . . .etc
Un «Cluster» est donc une collection d'objets qui sont «similaires» entre eux et qui sont «dissemblables » par rapport aux objets appartenant à d'autres groupes. On peut voir cette définition clairement graphiquement dans l'exemple suivant :

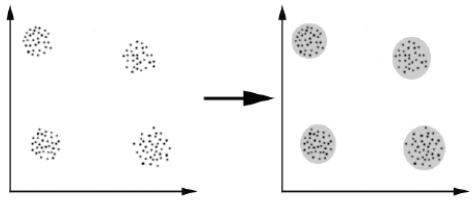


FIGURE 1.3: llustration de regroupement en clusters

Dans ce cas, il est très facile pour une personne d'identifier 4 Clusters dans lesquels les données (nuage des points) peuvent êtres divisées, le critère de similarité est la distance : deux ou plusieurs objets appartiennent au même cluster s'ils sont «proches», bien sur cela dépend d'une distance donnée (dans ce cas la distance géométrique).
Un autre type de regroupement est le clustering conceptuel : deux ou plusieurs objets appartiennent au même cluster si celui-ci définit un concept commun à tous les objets. En d'autres termes, les objets sont regroupés en fonction de leur adéquation aux concepts descriptifs, et non pas en fonction de mesures de similarité simple.[22]

Exemple

On utilise souvent ce type de classification en traitement d'images pour fixer les divers objets qu'elles contiennent (segmentation) : routes, villes, rues , des organes humaines (pour les images médicales) . . .

Type de clustering

Il existe deux grands types du clustering :

- **A/**le clustering hiérarchique :**d'agglomération («bottom-up»)**
- **B/**le clustering non-hiérarchique :**de division («top-down»)**

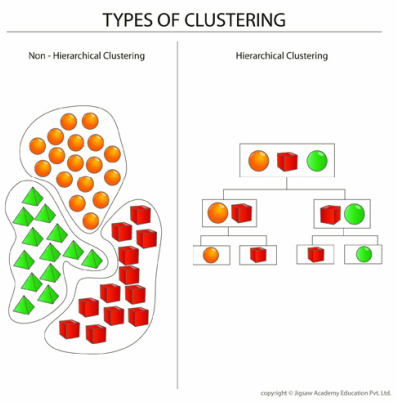


FIGURE 1.4: les deux types de clustering non-hiérarchique/hiérachique[© jigsaw academy education pvt. ltd]

METHODE SUPERVISEE «Classement» ou« DISCRIMINATION» :

Le «classement» est une méthode supervisée qui consiste à définir une fonction qui attribue une ou plusieurs classes à chaque donnée. Dans cette approche on suppose qu'un expert fournit auparavant les étiquettes pour chaque donnée, les étiquettes sont des classes d'appartenance.
Selon [5] : «(la classification supervisée (appelée aussi classement ou classification inductive) a pour objectif « d'apprendre » par l'exemple. Elle cherche à expliquer et a prédire l'appartenance de documents à des classes connues a priori. Ainsi c'est l'ensemble des techniques qui visent à deviner l'appartenance d'un individu à une classe en s'aidant uniquement des valeurs qu'il prend)»
Nous allons présenter dans le chapitre suivant une technique classique de classification supervisée : l'algorithme étudié c'est l'algorithme « k plus proches voisins ».

5 Conclusion

Nous avons vu une généralité sur quelques concepts du big data, data maining et la classification et ses méthodes, techniques, ses grandes approches, domaines d'applications. Dans le chapitre suivant nous nous intéresserons à une technique de classification supervisée c'est la méthode K-NN .

Chapitre 2

La méthode KNN

1 Introduction :

Comme nous avons pu le voir dans le premier chapitre qu'il y a deux grandes approches en classification : la classification automatique (clustering)et la discrimination (classement), dans ce chapitre nous détaillerons l'un des méthodes du deuxième type « classement » qui est une des techniques statistiques largement utilisées dans la Fouille de Données. il est dans un cadre d'apprentissage supervisé,c'est la méthode KNN, nous définirons la méthode k-ppv et présenterons certains des travaux de littérature révisés liés à cette étude.

2 Les K plus proches voisins (K-Nearest Neighbors) :

2.2.1 Définition 1

C'est une méthode de classification qui propose une analyse de similitude entre les données en utilisant la distance entre elles. L'algorithme fait un calcul de distance entre tous les individus et chaque objet est classé dans le groupe ou se trouvent ses K plus proche voisins. Quand on parle de voisin cela implique la notion de distance ou dissimilarité. [3]

l'algorithme K-NN (K-nearest neighbors) est une méthode d'apprentissage supervisé. Il peut être utilisé aussi bien pour la régression que pour la classification. Son fonctionnement peut être assimilé à l'analogie suivante "dis moi qui sont tes voisins, je te dirais qui tu es…". Pour effectuer une prédiction, l'algorithme K-NN ne va pas calculer un modèle prédictif à partir d'un Training Set comme c'est le cas pour la régression logistique ou la régression linéaire. En effet, K-NN n'a pas besoin de construire un modèle prédictif. Ainsi, pour K-NN il n'existe pas de phase d'apprentissage proprement dite. C'est pour cela qu'on le catégorise parfois dans le Lazy Learning. Pour pouvoir effectuer une prédiction, K-NN se base sur le jeu de données pour produire un résultat.

- La distance la plus populaire est **la distance euclidienne :** distance qui calcule la racine carrée de la somme des différences carrées entre les coordonnées de deux points :

  $D(x_1,x_2,...,x_p),(u_1,u_2,...,u_p) = \sqrt{(x_1-u_1)^2+(x_2-u_2)^2+...+(x_p-u_p)^2}$
- **La distance Manhattan :** calcule la somme des valeurs absolues des différences entre les coordonnées de deux points :
  $D(x_1,x_2,...,x_p),(u1,u2,...,up) = |(x_1-u_1)^2+(x_2-u_2)^2+...+(x_p-u_p)^2|$
- **La distance Hamming :** la distance entre deux points donnés est la différence maximale entre leurs coordonnées sur une dimension :
  $D(x_1,x_2,...,x_p),(u1,u2,...,up) = |(x_1-u_1)^2+(x_2-u_2)^2+...+(x_p-u_p)^2|$

2.2.2 Algorithme

1. initialisation , choix de :
   - Nombre de classes, Valeur de k, exemples initiaux, mesure de similarité.
2. pour chaque vecteur d'objet à classer :
   - mesurer la distance du vecteur avec tous les autres déjà classés
   - déterminer la liste des k vecteurs les plus proches de lui (k-ppv)
   - déterminer la classe la plus représentée dans la liste des k-ppv et affecter notre vecteur à cette classe.

2.2.3 choix du K

La valeur *K* est un des paramètres à déterminer lors de l'utilisation de ce type de méthode.La valeur que l'on choisit pour *k* va être plus critique,plus déterminante en rapport avec la performance de classificateur 2.1. On peut se permettre de considérer un plus grand nombre de voisins , sachant que plus ils différent du document à classer, moins ils ont d'impact sur la prise de décision .Cependant, il demeure nécessaire de limiter le nombre de voisins pour s'en tenir à un temps de calcul raisonnable . L'emploie de *k* voisins ,au lieu d'un seul , assure une plus grande robustesse à la prédiction . Classiquement, dans le cas où la variable à prédire comporte deux étiquettes , ce paramètre *k* est une valeur impaire afin d'avoir une majorité plus facilement décidable . [7]

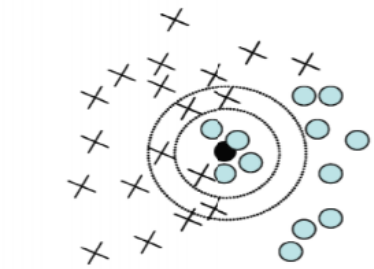


FIGURE 2.1: Le choix de "k" influence la décision : pour k=5 , la décision est de classer l'objet "noir" dans la classe "rond". pour k=9, la décision est de classer en tant que "croix"

3 Travaux connexes :

2.3.1 Algorithme de classification kNN efficace pour le Big Data

À l'ère du big data, il est très important d'apprendre efficacement à grande échelle dans toutes sortes d'applications réelles, telles que la classification et le clustering. Par conséquent, il est très évident de mettre à l'échelle les algorithmes de classification traditionnels, tels que les arbres de décision, Naive Bayes, le réseau neutre et les k plus proches voisins (KNN), afin que ces méthodes puissent être facilement utilisées dans le Big Data. En raison de la simplicité, de la facilité de compréhension et des performances relativement élevées de kNN, se concentrons sur la mise à l'échelle de la classification kNN dans l'application du Big Data [29]

Méthode

Dans cette section, nous introduisons deux processus dans notre algorithme, à savoir le processus de formation(the training process) et le processus de test. Le processus de formation est conçu pour sélectionner un cluster le plus proche pour chaque échantillon de test en tant que nouvel ensemble de données de formation, et le processus de test est utilisé pour classer chaque échantillon de test par un algorithme kNN dans son cluster le plus proche.

— **Le processus de formation (Training process) :**
Le clustering est l'une des techniques fondamentales de data mining car je peux être utilisé pour la segmentation de bases de données et la compression de données et peut être utilisé pour le prétraitement des données de data mining.
Le clustering est conçu pour regrouper un ensemble d'échantillons dans le même groupe (cluster)

qui sont plus similaires les uns aux autres qu'à ceux des autres groupes. En d'autres termes, les échantillons montrent une grqnde similarité au sein d'une cluster et une faible similarité entre les clusters.
Les méthodes de clustering récentes peuvent être réparties dans les catégories suivantes :
clustering basé sur la densité, clustering basé sur la grille, clustering de partitionnement et clustering hiérarchique, respectivement. Même si les méthodes de clustering précédentes ont montré de bonnes performances, mais ils sont limités dans leur applicabilité aux big data en raison de leur grande complexité de calcul.

Pour résoudre ce problème, on considére d'employer un algorithme de clustering satisfaisant les deux avantages suivants : faible complexité ; échelles linéairement [23] [12], respectivement.

À cette fin, on a utilisé le clustering spectral basé sur le Landmark (LSC),La raison d'être de LSC est de sélectionner (P inférieur de n) échantillon représentatif comme points de repère et représente les échantillons originaux sous forme de combinaisons linéaires de ces points de repère. Différent de cette méthode traditionnelle de regroupement spectral d'utiliser les échantillons entiers pour représenter chaque échantillon, le LSC réduit significativement la complexité de la matrice d'affinité. Dans le même temps, la complexité de la résolution de la valeur propre à des échelles linéaires [43].

LSC essaie de compresser les échantillons originaux en trouvant un ensemble de vecteurs de base et la représentation par rapport aux bases pour chaque échantillon, c'est-à-dire en recherchant p échantillons représentatifs. De cette façon, nous avons deux méthodes simples et efficaces pour sélectionner un échantillon de repère à partir de l'échantillon original, telles que l'échantillonnage aléatoire et la méthode basée sur les k-moyennes. L'échantillonnage aléatoire sélectionne au hasard des échantillons comme échantillons de repère tandis que la méthode basée sur les k moyennes effectue d'abord plusieurs fois le regroupement de tous les échantillons (pas besoin de convergence), puis utilise les centres de cluster comme échantillons de repère. Dans cet article, nous répétons l'algorithme k-means 10 fois, puis nous utilisons les centres de cluster comme échantillons de repère.

Tout d'abord, nous traitons chaque échantillon comme un vecteur de base pour construire la matrice de repère Z. Notez que LSC utilise p repères pour représenter chaque échantillon $X = [x_1, x_2, .., x_n] \in R^{m*n}$ nous devons trouver la matrice W qui est la matrice de projection de X à la matrice de repère Z [10]. La fonction de projection peut être définie comme suit :

$$W_{i,j} = \frac{K_i j(X_i, Z_j)}{\sum_{j \ldots Z_{<i>}} K_h(X_i, Z_j)}, j \in Z_{<i>} \ldots . (1)$$

Où : Z_i est la j ème colonne du vecteur Z ; et $Z_{<i>}$ désigne une sous-matrice de Z composée de r références les plus proches de X_i.
Le noyau gaussien $K_i = (X_i, Z_j) \exp(-\frac{\|X_i - Z_j\|^2}{2h^2})$ est l'un des plus couramment utilisés.
Ensuite, nous effectuons une analyse spectrale sur un graphique basé sur des points de repère et calculons la matrice du graphique comme suit :

$$GW\wr^T W\wr. (2)$$

Dans cette méthode, nous choisissons : $W\wr = D^{\frac{1}{2}} W$.
où D est la somme des lignes de W. Notez que chaque colonne de W totalise jusqu'à 1 et donc la

matrice de degrés de G est I.
Soit la décomposition en valeur singulière (SVD) de W est la suivante :
Algorithm 1 : The pseudo of LSC :
Input : n data points $x_1, x_2, .., x_n \in R^m$; Cluster number k ;
Output : k clusters ;

1. Produire P Landmark en utilisant la méthode des k-means.
2. Construire une matrice de Landmark Z entre les points de données et les échantillons de landmark, l'affinité étant calculée selon l'équation (1) ;
3. Calculer les k premiers vecteurs de WW^T, notés $U = [u_1, u_2, ..., u_k]$;
4. Calculez $V = [v_1, v_2, ..., v_k]$ selon l'équation (4) ;
5. Chaque ligne de V est un point de données et applique des k-means pour obtenir les clusters.

— **Le processus de test (Testing process) :**

En supposant que nous produisions déjà k clusters et centres de cluster par l'algorithme LSC, nous trouvons alors le centre de cluster le plus proche pour chaque échantillon de test et utilisons le cluster correspondant comme nouvel ensemble de données d'apprentissage pour chaque échantillon de test.
En raison de la réalisation des clusters avec une forte similitude au sein d'un cluster, nous appliquons kNN pour classer les échantillons de test dans le nouvel ensemble de données d'apprentissage. De cette manière, l'algorithme proposé garantit toujours une précision de classification relativement élevée. Nous listons le pseudo de notre méthode proposée dans l'algorithme 2.

Algorithm 2 : The pseudo of LC-kNN algorithm :
Input : Training dataset, test samples Y ;
Output :Class label ; Output :

1. Produire m centre de cluster avec l'algorithme LSC.
2. Calculer la distance $D(y, C_i)$ entre échantillon y et centre du cluster $D(y, C_i), i = 1, 2, ..., m$;
3. Calculer le centre de cluster le plus proche $C_i = minD(y, C_i)$
4. Faire l'appele de l'algo kNN pour prédire y dans les données de taining process.

2.3.2 Expérimentations

Comme mentionné dans les sections précédentes, notre algorithme proposé est une extension de kNN. Ainsi, afin de montrer l'efficacité de l'algorithme LC-kNN, nous avons pris le kNN comme référence et avons fait une comparaison entre kNN, LC-kNN et RC-kNN (random clustering kNN), et plusieurs jeux de données réels de l'UCI [2] , ensembles de données d'imagerie médicale [14] [13] et ensembles de données LIBSVM [8].

Résultats expérimentaux de LC-kNN et RC-kNN avec différentes valeurs de m :

La valeur de m est très importante pour l'algorithme LC-kNN, car elle affecte directement les performances finales et les applications réelles.

Ainsi, afin de sélectionner m approprié pour apporter un grand effet à LC-kNN, un groupe d'expériences ont été menées sur des ensembles de données en choisissant différentes valeurs de m pour LC-kNN et RC-kNN. Plus précisément, les expériences LC-kNN et RC-kNN dans ce groupe ont été réalisées sur les ensembles de données avec m = 10, 15, 20, 25 et 30, respectivement. La comparaison des performances de classification et du coût en temps de deux algorithmes (c'est-à-dire RC-kNN et LC-kNN, respectivement) avec différentes valeurs de *m* est présentée dans le tableau $1 \sim 9$. La liste des tablaux se trouve dans l'annexe A (voir l'annexe).

Résultats :

1. A partir du tableau 1 jusqu'à 9, nous avons constaté que les deux algorithmes proposés nécessitaient moins de temps avec le plus grand nombre de clusters *m*, tandis que plus de temps coûtait pour le plus petit nombre de *m*.
2. Les résultats expérimentaux ont montré que la classification kNN proposée fonctionnait bien en termes de précision et d'efficacité, et qu'il était approprié de traiter les big data

2.3.3 Un classificateur de KNN distribué pour le Big Data

L'algorithme proposé "Z-KNN "

Dans cette section, l'algorithme Z-KNN proposé et son application MapReduce seront expliqués.
L'algorithme classique K-NN est basé sur l'idée simple de calculer les distances entre une donnée de test à classer et la totalité du nombre m d'instances de données dans l'ensemble d'apprentissage. Après avoir calculé toutes les distances, le K-NN classique trie les distances mesurées et utilise le premier nombre K de voisins d'entraînement des données testées.
La décision de classification est ensuite donnée en déterminant quelle catégorie de données a le plus de cas parmi les K voisins les plus proches, ce que l'on appelle le vote à la majorité.

1. toute la décision est basée sur les distances des instances individuelles entre tous les tsv_i et trv_j.

2. Le calcul des distances d'instance étant une tâche indépendante, la possibilité de répartir les calculs de distance entre plusieurs processus rend la stratégie K-NN adaptée aux environnements distribués.

3. D'un autre côté, en particulier lorsqu'un ensemble de données avec un nombre élevé d'instances et un nombre élevé de fonctionnalités par instance doit être classé, les performances de précision de classification de l'algorithme K-NN classique deviennent inférieures à celles de ses autres concurrents bien connus, comme K-Means. classification [37].

4. Par conséquent, on peut en déduire que pour devenir un algorithme de classification adapté aux besoins modernes d'analyse des données, les performances de précision de classification du K-NN devraient être améliorées dans un environnement distribué.

L'algorithme Z-KNN proposé est un algorithme de classification basé sur K-NN distribué, conçu pour fonctionner dans l'environnement MapReduce. Hadoop MapReduce est un framework logiciel permettant d'écrire facilement des applications qui traitent de grandes quantités de données en parallèle sur de grands clusters de matériel de base de manière fiable et tolérante aux pannes [4].

1. a mapreduce job :
2. Dans le framework MapReduce, toute tâche distribuée est conçue comme une combinaison d'au moins trois fonctions : Les fonctions Driver, Mapper et Reducer, héritées des classes MapReduce correspondantes [4].

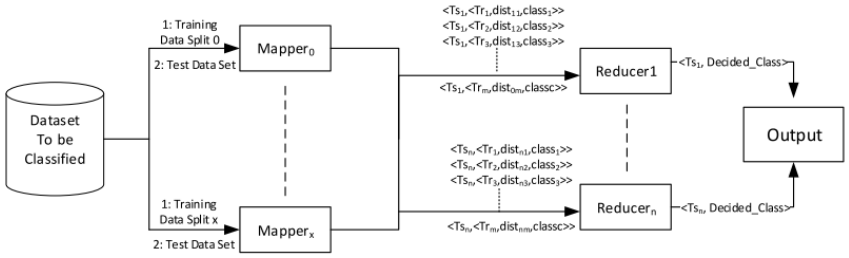


FIGURE 2.2: MapReduce flux de l'algorithme Z-KNN [28]

Le framwork MapReduce de l'algorithme Z-KNN pour m nombre d'instances dans l'ensemble de données d'apprentissage et n nombre d'instances dans l'ensemble de données de test est présenté dans la figure 2.2.
Comme on peut le voir sur la figure 2.2, les sous-tâches de l'algorithme Z-KNN sont composées des fonctions Mapper et Reducer du bien connu MapReduce Framework.
Le travail Mapper est responsable de recevoir les fractionnements de l'ensemble de données d'entraînement du pilote MapReduce, qui est la fonction principale de configuration de l'environnement et de gestion du traitement distribué s'exécutant au-dessus du framework Hadoop, et de calcul des distances entre l'instance de test à traiter et les instances d'apprentissage trouvées dans la division des données d'apprentissage reçues.

Selon le Framework MapReduce, la sortie de la fonction Mapper doit être un vecteur <Key, Value> [4]. Dans l'algorithme Z-KNN, la clé de sortie du mappeur est l'ID de l'instance de test et la valeur est un objet, qui contient l'instance d'entraînement utilisée, la distance entre les instances de test et d'entraînement et l'ID de classe auquel appartient l'instance d'entraînement.

Algorithm 1: Z-KNN Mapper Function()

Input: <key, value>
key : the record id of the training instance
value: the set of feature values of the training instance

Output to MapReduce Env. : <key1, value1>
key1: the record id of the test instance
value1: a vector containing the training instance id, distance and the class id of the training instance

1: class_tr = readClassId(value)
2: for i=1 to n
3: //the loop to iterate each test instance
4: $dist_{ij}$ = DistanceFunction(tr_j, ts_i)
5: Context.write(i, object <tr_j , $dist_{ij}$, class_tr>)
6: end for
7: return

FIGURE 2.3: La fonction Mapper de l'algorithme Z-KNN [28]

La fonction Mapper : Cette méthode permet de deviser la Dataset en de lignes, ou en de vecteurs de données, et de faire appel à la méthode euclideandist pour calculer la distance entre chaque vecteur de données et le vecteur de données d'entrée qui on cherche sa classe. Ensuite, elle stocke le résultat dans une liste.
Comme un exemple de calcul de distance, si une seule instance de test ts_i et une seule instance d'apprentissage (tr_j) sont considérées,
la distance entre ces deux instances est calculée par l'équation suivante.

$$dist_{ij} = \sqrt{\sum_{k=1}^{p}(tsi_{feat_k} - trj_{feat_k})^2}$$

FIGURE 2.4

Si nous supposons que l'id de classe de l'instance d'apprentissage tr_j est la classe A, alors la sortie d'un mappeur Z-KNN devient ; $< ts_i, < tr_j, dist_{ij}$, classe A ».

La fonction Reducer :
La fonction reducer contient la phase de décision de classification pour les instances de test, où la contribution principale de l'algorithme Z-KNN proposé peut être vue.

2.3.4 Expérimentations et résultats

Les fonctions MapReduce du Z-KNN sont codées dans Sun JAVA JDK 1.8 [15]. Les expériences de classification Z-KNN sont menées sur un petit cluster de stations de travail HP installées avec Ubuntu Linux 16.04 et Apache Hadoop 2.7.4. Pour valider le schéma de classification, pour chaque ensemble de données utilisé dans les expériences, une validation croisée de 10 fois est utilisée, où chaque test est répété 10 fois et les moyennes des 10 tests sont prises en compte afin que des résultats fiables puissent être obtenus. Dans les expériences, des ensembles de données réels téléchargés depuis l'UCI Le référentiel d'apprentissage automatique [27] est utilisé. Les 5 ensembles de données réels utilisés dans les expériences sont résumés dans le tableau 2.3.4.

Dataset	**Instant**	**Features**	**Classes**
ionosphere	351	34	2
wdbc	569	32	2
wine	178	13	3
seeds	210	36	7
satimage	6435	36	7
pendigits	10992	16	10

The real datasets used in the experiments

1. Ionosphere : L'ensemble de données de l'ionosphère est constitué des données issues de la classification des retours radar de l'ionosphère. L'ensemble de données contient 351 instances appartenant à 2 classes. Chaque instance contient des valeurs appartenant à 34 entités. Cet ensemble de données est également utilisé dans [33]

2. WDBC :Le WDBC (diagnostic du cancer du sein du Wisconsin) a été utilisé pour la première fois dans [21]. L'ensemble de données contient 569 instances appartenant à 2 classes. Chaque instance contient des valeurs appartenant à 32 entités. L'ensemble de données WDBC est également utilisé dans [33]

3. Wine :L'ensemble de données sur le vin contient des données d'analyse chimique pour déterminer l'origine des vins.

4. Seeds : les mesures des propriétés géométriques des grains appartenant à trois variétés différentes de blé.

Dans cette section, les résultats acquis après de nombreuses expériences sont présentés. Les performances de l'algorithme Z-KNN sont mesurées en termes de précision de classification, qui représente le rapport entre le nombre de classifications correctes et le nombre de toutes les classifications. Les résultats de la précision de la classification sont donnés dans le figure 2.5.

	Z			
K	3	5	7	9
5	**0.9507**	0.9443	0.9474	0.9443
7	0.9474	0.9443	0.9474	0.9443
9	0.9474	0.9443	0.9474	0.9443
11	0.9474	0.9478	0.9474	0.9443

(a) Wdbc Dataset

	Z			
K	3	5	7	9
5	0.9524	**0.9714**	0.9714	0.9714
7	0.9524	0.9714	0.9714	0.9714
9	0.9524	0.9714	0.9714	0.9714
11	0.9524	0.9714	0.9714	0.9714

(b) Seeds Dataset

	Z			
K	3	5	7	9
5	0.9199	0.9141	0.9147	0.9147
7	0.9196	0.9144	**0.9203**	0.9144
9	0.9196	0.9144	0.9203	0.9144
11	0.9196	0.9144	0.9203	0.9144

(c) Ionosphere Dataset

	Z			
K	3	5	7	9
5	0.9797	0.9803	0.9803	0.9803
7	0.9803	0.9808	0.9808	0.9808
9	0.9803	0.9808	0.9811	0.9811
11	0.9806	0.9811	**0.9814**	0.9814

(d) Pendigits Dataset

	Z			
K	3	5	7	9
5	0.9220	0.9245	0.9245	0.9220
7	0.9235	0.9265	0.9270	0.9245
9	0.9250	0.9280	**0.9285**	0.9255
11	0.9240	0.9275	0.9285	0.9255

(e) Satimage Dataset

	Z			
K	3	5	7	9
5	0.7993	0.7974	**0.8320**	0.8203
7	0.7917	0.7974	0.8209	0.8092
9	0.7882	0.7973	0.8209	0.8092
11	0.7271	0.7379	0.7611	0.7608

(f) Wine Dataset

FIGURE 2.5: Les résultats de précision de classification pour (a) Wdbc, (b) Seeds, (c) Ionosphere, (d) Pendigits , (e) Satimage and (f) Wine datasets

En tant que performance de précision de classification globale, on peut voir sur la figure 2 que le Z-KNN a réussi à détecter correctement la classe de plus de 92% des données testées dans tous les ensembles de données.
En regardant les performances de précision du Z-KNN, on peut voir que, pour la majorité des ensembles de données, l'algorithme Z-KNN parvient à détecter la classe correcte des instances de test avec K valeurs 5 ou 7, sans avoir besoin de analyser plus de nombre de voisins les plus proches et donc atteindre un coût de calcul raisonnable.
En ce qui concerne le paramètre Z, on constate que l'algorithme Z-KNN parvient à atteindre une précision de classification élevée avec 5 à 7 voisins les plus proches dans la représentation de classe, ce qui montre également que l'ajout du paramètre Z n'augmente pas le coût de calcul. significativement.
Surtout sur les datasets Pendigits et Satimage, qui contiennent un nombre plus élevé d'instances, d'entités et de classes par rapport à d'autres datasets,
Il convient de mentionner qu'en atteignant des valeurs Z inférieures ou égales à 7, Z-KNN montre une applicabilité réaliste également à de vraies applications Big Data. Les performances de précision de l'algorithme Z-KNN proposé et sa comparaison avec la précision classique de K-NN sont données dans le tableau 2.3.4

Dataset	**Classical K-NN**	**Z-KNN**
ionosphere	0.8295	0.8320
wdbc	0.6548	0.9507
wine	0.8424	0.9714
seeds	0.6286	0.9203
satimage	0.978	0.9814
pendigits	0.9065	0.9285

K-NN classique vs Z-KNN

Comme on peut le voir dans le tableau 2.3.4, le Z-KNN améliore considérablement les performances de précision de l'algorithme classique K-NN dans tous les ensembles de données. De plus, les performances de

l'algorithme Z-KNN sont comparées à deux algorithmes récemment proposés dans [10] [33]. Les résultats comparatifs sont présentés dans le tableau 2.3.4.

Dataset	**LC-KNN**	**SR-KNN**	**Z-KNN**
ionosphere	-	0.9707	0.8320
wdbc	-	0.965	0.9507
wine	-	0.9019	0.9714
seeds	-	0.8971	0.9203
satimage	0.8883	0.8806	0.9285
pendigits	0.9721	0.9452	0.9285

Comparaisons de performances

Comme on peut le voir dans les comparaisons de performances, Z-KNN fonctionne mieux dans presque tous les datasets par rapport aux autres propositions basées sur KNN, ce qui démontre que la représentation d'instance Z proposée améliore considérablement les performances de précision du K-NN classique et de certains de ses variations.[28]

2.3.5 Conclusion

Dans ce chapitre nous avons vu la méthode KNN, et KNN distribué "Z-KNN " et quelques expérimentations et résultats qui ont montré que la classification kNN proposée fonctionnait bien en termes de précision et d'efficacité, et qu'il était approprié de traiter les big data.

Chapitre 3

Application

1 Introduction :

Dans ce dernier chapitre et après l'aperçu théorique des chapitres précédents, nous présentons le côté pratique de notre travail. Nous avons pour objectif de réaliser l'implémentation de l'algorithme KNN dans une première étape dans un environnement séquentiel et dans une deuxième étape dans un environnement distribué, Hadoop Mapreduce, avec un seul nœud et avec plusieurs nœuds. le but est de faire une comparaison entre les résultats obtenue , en terme de performance, dans les trois différents configurations. Dans ce qui suit nous allons décrire l'environnement de travail choisi et les étapes fondamentales de la réalisation de notre travail.

2 L'environnement de travail

En ce qui concerne le matériel utilisé, nous avions fait deux tentative une première où nous avions installé Ubuntu 20.04 LTS et utilisé le système de virtualisation VirtualBox 5.2 pour la création du cluster de machines virtuelles. A cause du confinement nous avions perdu l'accès physique à cette machine. Nous avion fait une deuxième tentative dans laquelle nous avion tenté d'utiliser un serveur qui a été mis à notre disposition au niveau de la cellule réseau avec un accès à distance via un vpn. Ce serveur est un Dell Inc. PowerEdgeR710 équipé d'un processeur Intel(R) Xeon(R) E5506 @2.13Ghz avec 16Go de Ram et une capacité de stockage de 130 Go. Le serveur fonctionnait sous le système de virtualisation VMware ESXi version 5.1.0 dans lequel les machines virtuel qui constituent le cluster ont été crées sous le système d'exploitation Ubuntu 20.04 LTS (GNU/Linux 5.4.0-26-generic x86-64). Ce système d'exploitation a été choisi pour son adaptation avec l'environnement Hadoop Mapreduce.

Apache^{TM}Hadoop

Cet écosystème est un framework open-source pour stocker et traiter les données volumineuses sur un cluster. Il est utilisé par un grand nombre de contributeurs et utilisateurs. Son système de fichiers distribué favorise un taux élevé de transfert de données entre les nœuds et permet un fonctionnement ininterrompu du système en cas de défaillance d'un des noeuds du cluster. Cette approche diminue le risque de panne majeure, même lorsqu'un nombre important de nœuds deviennent inopérants.

FIGURE 3.1: HADOOP

Hadoop s'inspire de Google MapReduce, un modèle logicielle qui consiste à fragmenter une application en de nombreux petits composants. Chacun de ces composants (appelé fragment ou bloc) peut s'exécuter sur n'importe quel nœud du cluster, Il a une licence Apache 2.0.
L'écosystème Apache^{TM}Hadoop se compose du noyau Hadoop, de MapReduce, du système de fichiers distribué (HDFS) Hadoop et d'un certain nombre de projets associés, notamment Apache Hive, HBase et Zookeeper.[25]

L'infrastructure Hadoop est utilisée par des acteurs majeurs - dont Google, Yahoo, etc. - pour des applications faisant intervenir des moteurs de recherche, de la publicité en ligne ou tout autre gestion des Big Data.

Les systèmes d'exploitation privilégiés sont Windows Server et Linux, mais Hadoop fonctionne également avec BSD [1].

Hadoop est aujourd'hui en version 3.3.0. Hadoop 3.1 apporte des nouveautés pour le traitement par flux (et plus simplement par batch).

Hadoop est composé principalement de deux parties. La première partie s'appelle hdfs qui est le systéme de gestion des fichiers distribués qui sert au stockage des données massives (Big Data). donc hdfs comme son nom l'indique c'est bien un système de fichiers et non pas une base de données.C'est-à-dire qu'il est composé d'un ensemble de nœuds qui peuvent être des machines physiques ou virtuelles. Un noeuds est appelé Namenode et c'est lui orchestre toutes les opérations qui vont se faire et c'est lui qui va gérer la distribution de données sur l'ensemble des autres noeuds qui sont appelés datanodes. Le namenode joue le rôle de maitre (Master node).

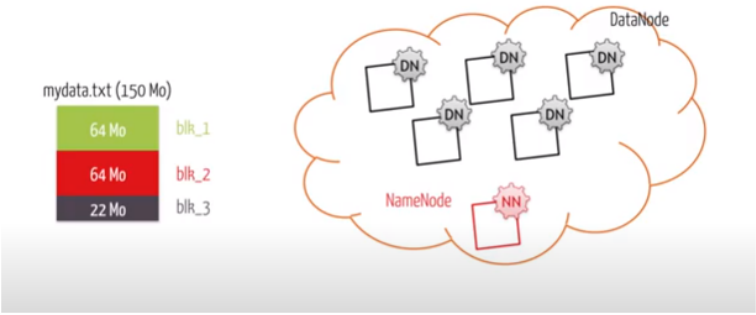


FIGURE 3.2: Le principe de HDFS

Donc en plus de hdfs, Hadoop dispose également d'un framwork qui s'appelle Mapreduce, qui est est un framwork d'exécution et de traitement de données et en plus également c'est un paradigme de traitement qui va permettre d'exécuter des données de façon parallèle pour réaliser une certaine action .[25]

3.2.1 Le langage de codage

Pour réaliser notre implémentation, nous avons choisi le langage Java a travers sur la plateforme IntelliJ IDEA Community Edition 2020. Néanmoins, il est possible de choisir le langage Python qui est en train de s'imposer de plus en plus.

1. (à l'origine, Berkeley Software Distribution) désigne la version spécifique du système d'exploitation UNIX...

FIGURE 3.3: IntelliJ IDEA

l'approche suivie dans l'implémentation :

(1) Création d'un projet Maven dans IntelliJ IDEA , en définissant les valeurs suivantes pour le projet :

- **GroupID :** org.example
- **ArtifactID :** KNN.master
- **Version :** 1

(2) Ouvreture du fichier **pom.xml**, et on ajout des dépendances pour Hadoop, HDFS et Map Reduce)figure 3.4)

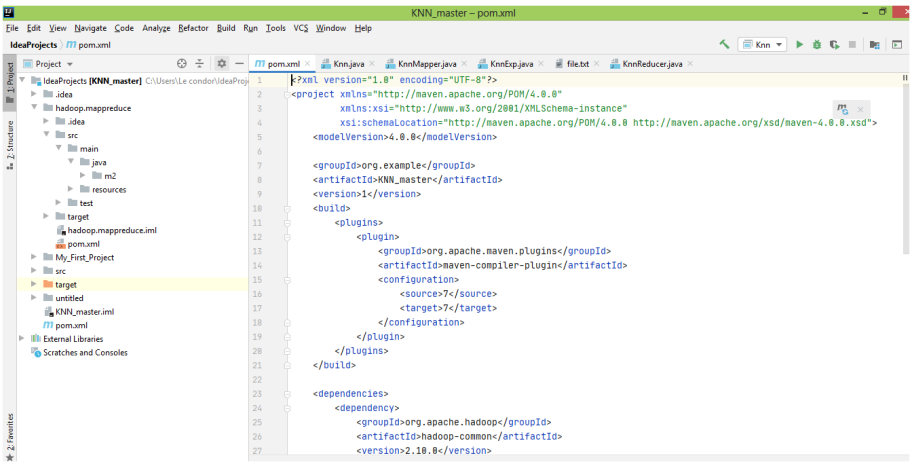


FIGURE 3.4: Le fichier pom.xml

(3) Création de la classe **KnnMapper** sous le package **KNN.master** qui implémente quatre méthodes qui sont :

1. **euclideandist :** Cette méthode permet de calcul la distance euclidienne entre l'élément qu'on cherche sa classe et les autres éléments existants dans le Dataset. Autrement, cette méthode calcule la distance entre la ligne ou le vecteur dans le fichier de input et chaque ligne de Dataset.

2. **setup :** Cette méthode s'exécute avant la fonction map et elle permet de lire de paramètres à patir de l'objet context et cela permet de personnaliser la logique de traitement. Dans notre cas, à l'aide de cette méthode, on a pu initialiser le variable numoffeatures qui représente le nombre totale des attributs et aussi le tableau feature qui représente la ligne de données au niveau de la fichier d'input.
3. **map :** Cette méthode permet de deviser la Dataset en de lignes, ou en de vecteurs de données, et de faire appel à la méthode euclideandist pour calculer la distance entre chaque vecteur de données et le vecteur de données d'entrée qui on cherche sa classe. Ensuite, elle stocke le résultat dans une liste.
4. **cleanup :** Cette méthode s'exécute après la fonction map, elle permet de faire une sorte de clean up de ressources qui on a utilisé. Dans notre cas, cette méthode permet de trier la liste retourner par la fonction map contenant le résultat de comparaison.

(4) Création de la classe **KnnReducer** sous le package**KNN.master (5)** Enfin, Création de la classe **KNN** sous le package **KNN.master**

qui contient la fonction main du programme et qui va permettre de :

1. Récupérer la configuration générale du cluster.
2. Créer un job avec le nom **Find K-Nearest Neighbour**.
3. Préciser quelles sont les classes Map et Reduce du programme en utilisant les setters **setMapperClass()** et **setReducerClass()**.
4. Préciser les types de clés et de valeur correspondant à notre problème à l'aide de méthodes **setOutputKeyClass()**, **setOutputValueClass()**.
5. Indiquer où sont les données d'entrée et de sortie dans HDFS, à l'aide de **setInputPaths()** et **setOutputPath()**.
6. Définir le type de clé et de valeur pour notre problème à l'aide de**setOutputKeyClass()** et **setOutputValueClass ()**.
7. Lancer l'exécution de la tâche.

Conclusion Générale

Dans ce travail de master, nous avons tenté d'étudier l'extension de l'algorithme kNN à un environnement distribué pour pouvoir l'utiliser pour l'analyse de données dans le cas du big data. Nous avons ciblé deux problématiques l'efficacité en terme de temps d'exécution et la performance en terme de classification. Le problème d'efficacité se pose en raison d'un besoin élevé en matière de stockage et de temps d'exécution dans le cas des données massives.

Nous avons trouvé dans la recherche bibliographique des travaux de recherche qui ont bien montré que l'algorithme kNN est adapté à un environnement parallèle et qu'il est prometteur en termes de précision et d'efficacité, et qu'il était bien approprié pour le traitement du big data.

Nous avons exposé une variante du classificateur kNN distribué pour les big data nommé Z-KNN dont les auteurs ont montré qu'il réalisait des résultats de performance encourageant après des expériences approfondies menées sur l'environnement Hadoop MapReduce. Les auteurs ont aussi observé que l'algorithme Z-kNN qu'ils ont proposé s'est avèré être un concurrent sérieux avec sa précision de classification élevée obtenue pour plusieurs ensembles de données réels et différents.

Pour notre part nous avions pour objectif de mener une expérimentations avec notre propre implémentation d'une version distribué de l'algorithme kNN et avec d'autres ensembles de donnée dans le but de mesurer le gain en terme de d'efficacité et de performance. Mais nous n'avons pas pu mener à terme notre réalisation à cause de la pandémie du Covid19 qui a mené à la fermeture de l'université. Ceci nous a privé de l'accès à la machine sur laquelle nous avions commencé notre travail au niveau du Laboratoire LIM à l'université Amar Telidji. La deuxième tentative qui consistait à utiliser un serveur à distance a coupé court à cause de problème technique d'accès à distance via internet. Les machines personnelles n'étant pas assez puissantes pour supporter la charge de traitement que nécessite un environnement parallèle, nous n'avons pas été en mesure de mener à terme notre travail dans les délais impartis. A la fin nous ne pouvons qu'espérer pouvoir reprendre la partie application dès que les circonstances le permettent et ainsi compléter ce travail.

Annexes

A Annexe A

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.9027±1.6498e-005	0.9355±7.1306e-006
	Time	3.5589±0.0107	3.7605±0.0242
15	Accuracy	0.8964±5.1803e-005	0.9338±4.1625e-006
	time	2.4857±0.0032	2.7260±0.0077
20	Accuracy	0.8770±7.4889e-005	0.9300±4.9238e-006
	time	2.3202±0.0010	2.5157±0.01928
25	Accuracy	0.8793±4.9917e-005	0.9284±1.0637e-005
	time	1.8586±0.0008	1.9971±0.0042
30	Accuracy	0.8607±4.6629e-005	0.9275±1.1596e-005
	time	1.6441±0.0002	1.9249±0.0023

FIGURE 0.5: Classification accuracy (meanstandard deviation) and Time cost (seconds : mean standard deviation) on USPS dataset at different values of m

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.7221±4.8878e-005	0.8389±3.1656e-005
	time	2.9369±0.0508	3.5504±0.0927
15	Accuracy	0.6840±2.3333e-004	0.8364±2.3136e-005
	time	2.8905±0.0456	3.1222±0.01397
20	Accuracy	0.6657±2.4739e-004	0.8353±3.3233e-005
	time	2.0564±0.0011	2.1490±0.0065
25	Accuracy	0.6478±2.2689e-004	0.8338±8.7844e-005
	time	1.8240±0.0020	2.1148±0.0094
30	Accuracy	0.6396±6.9156e-005	0.8313±3.8678e-005
	time	1.5457±0.0002	1.7274±0.0011

FIGURE 0.6: Classification accuracy (meanstandard deviation) and Time cost (seconds : mean standard deviation) on MNIST dataset at different values of m.

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.9311±5.0989e-005	0.9526±1.4511e-005
	time	23.3933±0.9677	28.5940±3.2405
15	Accuracy	0.9252±1.0573e-004	0.9494±1.3378e-005
	time	18.0106±0.2434	23.1904±1.0894
20	Accuracy	0.9166±2.8267e-005	0.9411±5.4699e-004
	time	12.7685±0.0966	16.2759±0.8880
25	Accuracy	0.9150±7.0000e-005	0.9321±6.4810e-004
	time	9.9201±0.3696	13.8645±1.5093
30	Accuracy	0.9079±1.0366e-004	0.9192±5.3796e-004
	time	8.4064±0.0784	11.3922±0.0658

FIGURE 0.7: Classification accuracy (meanstandard deviation) and Time cost (seconds : mean standard deviation) on GISETTE dataset at different values of m.

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.7892±3.8822e-005	0.9495±1.0760e-006
	time	3.2391±0.0015	3.2994±0.0010
15	Accuracy	0.7932±3.7106e-005	0.9469±5.5751e-006
	time	3.3808±0.0435	3.4334±0.0585
20	Accuracy	0.6815±1.3812e-004	0.9451±1.9756e-006
	time	3.0938±5.8392e-004	3.1285±3.1243e-004
25	Accuracy	0.7279±5.6480e-005	0.9423±5.2818e-006
	time	3.3950±0.0018	3.4813±0.0054
30	Accuracy	0.6214±9.8480e-005	0.9403±3.9204e-006
	time	3.0889±1.3000e-003	3.1168±3.8514e-004

FIGURE 0.8: Classification accuracy (meanstandard deviation) and Time cost (seconds : mean standard deviation) on LETTER dataset at different values of m.

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.9452±3.5382e-005	0.9721±4.7991e-006
	time	2.3380±0.0041	2.4056±0.0101
15	Accuracy	0.9316±1.0341e-004	0.9711±6.0196e-006
	time	2.5451±0.0011	2.5709±0.0089
20	Accuracy	0.9163±1.5515e-004	0.9700±2.5390e-006
	time	2.2233±6.4795e-005	2.2554±2.1569e-004
25	Accuracy	0.9216±1.5677e-004	0.9687±3.5642e-006
	time	2.5270±0.0056	2.5468±0.0083
30	Accuracy	0.9088±1.8409e-004	0.9683±1.5809e-006
	time	2.1805±7.4785e-005	2.2022±8.9611e-005

FIGURE 0.9: Classification accuracy (meanstandard deviation) and Time cost (seconds : mean standard deviation) on PENDIGITS dataset at different values of m

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.8603±8.9122e-005	0.8883±8.1139e-006
	time	1.2868±6.5495e-005	1.3027±1.1429e-004
15	Accuracy	0.7917±3.1680e-005	0.9468±3.7244e-006
	time	3.8583±0.0332	3.9337±0.0152
20	Accuracy	0.8418±8.8847e-005	0.8884±6.8028e-006
	time	1.2292±3.2277e-005	1.2463±4.3126e-005
25	Accuracy	0.7283±1.1039e-004	0.9421±8.8449e-006
	time	3.5062±0.0061	3.6287±0.0052
30	Accuracy	0.8312±3.5146e-004	0.8878±4.9556e-006
	time	1.2225±1.8176e-005	1.2396±1.7711e-005

FIGURE 0.10: Classification accuracy (meanstandard deviation) and Time (seconds : mean standard deviation) on SATIMAGE dataset at different values of m.

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.7024±0.0010	0.7667±0.0019
	time	0.0310±2.7390e-004	0.0333±1.9201e-005
15	Accuracy	0.6857±0.0014	0.7500±0.0013
	time	0.0308±7.9360e-006	0.0325±3.3635e-005
20	Accuracy	0.6619±0.0029	0.7143±0.0028
	time	0.0295±3.0063e-006	0.0313±6.8908e-006
25	Accuracy	0.6548±0.0039	0.7071±0.0038
	time	0.0281±4.8219e-007	0.0286±7.0881e-007
30	Accuracy	0.6619±0.0015	0.7190±0.0031
	time	0.0306±1.2641e-005	0.0326±5.3013e-006

FIGURE 0.11: Classification accuracy (meanstandard deviation) and Time cost (seconds : mean standard deviation) on ADNC dataset at different values of m.

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.4792±0.0082	0.5833±0.0089
	time	0.0168±4.5147e-007	0.0171±1.4130e-006
15	Accuracy	0.5125±0.0019	0.6042±0.0040
	time	0.0167±9.4536e-007	0.0173±5.6957e-007
20	Accuracy	0.5458±0.0060	0.6500±0.0035
	time	0.0170±1.2416e-006	0.0188±1.1793e-005
25	Accuracy	0.5333±0.0053	0.6417±0.0035
	time	0.0163±4.9822e-007	0.0286±7.0881e-007
30	Accuracy	0.5000±0.0042	0.6125±0.0019
	time	0.0166±8.6385e-007	0.0250±2.2840e-004

FIGURE 0.12: Classification accuracy (meanstandard deviation) and Time cost (seconds : mean standard deviation) on psMCI dataset at different values of m.

m	Criterion	RC-kNN	LC-kNN
10	Accuracy	0.5476±0.0049	0.6159±0.0045
	time	0.0468±2.8577e-005	0.0506±5.9543e-005
15	Accuracy	0.5397±0.0032	0.5633±0.0336
	time	0.0505±2.7585e-005	0.0538±4.6465e-005
20	Accuracy	0.5458±0.0060	0.6500±0.0035
	time	0.0452±4.9222e-006	0.0480±1.1608e-005
25	Accuracy	0.5222±0.0015	0.5746±0.0008
	time	0.0489±1.5349e-005	0.0540±4.5819e-005
30	Accuracy	0.5016±0.0245	0.5984±0.0021
	time	0.0490±2.6160e-005	0.0536±4.5625e-005

FIGURE 0.13: Classification accuracy (meanstandard deviation) and Time cost (seconds : mean standard deviation) on MCINC dataset at different values of m.

Dataset	RC-kNN		LC-kNN		kNN	
	Accuracy	time	Accuracy	time	Accuracy	time
USPS	0.9027	3.5589	0.9355	3.7605	0.9482	32.8764
MNIST	0.7221	2.9369	0.8389	3.5504	0.8635	24.1575
GISETTE	0.9311	23.3933	0.9526	28.594	0.9660	217.3327
LETTER	0.7892	3.2391	0.9495	3.2994	0.9518	19.8246
PENDIGITS	0.9452	2.3380	0.9721	2.4056	0.9780	7.2982
SATIMAGE	0.8603	1.2868	0.8883	1.3027	0.9065	3.5525
ADNC	0.7024	0.0310	0.7667	0.0333	0.7857	0.0355
psMCI	0.4792	0.0168	0.5833	0.0171	0.6042	0.0176
MCINC	0.5476	0.0468	0.6159	0.0506	0.6413	0.0575

FIGURE 0.14: Classification Accuracy and Time Cost of three algorithm on nine dataset

Bibliographie

[1] http ://www.lebigdata.fr.

[2] Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

[3] Timothy L Bailey and Charles Elkan. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine learning*, 21(1-2) :51–80, 1995.

[4] Milind Bhandare, Vikas Nagare, et al. Generic log analyzer using hadoop mapreduce framework. *International Journal of Emerging Technology and Advanced Engineering (IJETAE)*, 3(9) :603–607, 2013.

[5] Christophe Biernacki, Gilles Celeux, and Gérard Govaert. Choosing starting values for the em algorithm for getting the highest likelihood in multivariate gaussian mixture models. *Computational Statistics & Data Analysis*, 41(3-4) :561–575, 2003.

[6] Gilles Bisson. La similarité : une notion symbolique/numérique. *Apprentissage symbolique-numérique*, 2 :169–201, 2000.

[7] Gilles Celeux, Edwin Diday, Gérard Govaert, Yves Lechevallier, and Henri Ralambondrainy. *Classification automatique des données :[environnement statistique et informatique]*. Bordas Paris, 1989.

[8] Chih-Chung Chang and Chih-Jen Lin. Libsvm : A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3) :1–27, 2011.

[9] Xinlei Chen and Deng Cai. Large scale spectral clustering with landmark-based representation. In *Twenty-fifth AAAI conference on artificial intelligence*. Citeseer, 2011.

[10] Zhenyun Deng, Xiaoshu Zhu, Debo Cheng, Ming Zong, and Shichao Zhang. Efficient knn classification algorithm for big data. *Neurocomputing*, 195 :143–148, 2016.

[11] Richard O Duda. Hart. peter e., stork, david g.,". *Pattern Recognition" 2nd Edition p. cm."A Wiley-Interscience Publication." Partial Contents : Part*, 1, 2001.

[12] Maurizio Filippone, Francesco Camastra, Francesco Masulli, and Stefano Rovetta. A survey of kernel and spectral methods for clustering. *Pattern recognition*, 41(1) :176–190, 2008.

[13] Yue Gao, Meng Wang, Dacheng Tao, Rongrong Ji, and Qionghai Dai. 3-d object retrieval and recognition with hypergraph analysis. *IEEE Transactions on Image Processing*, 21(9) :4290–4303, 2012.

[14] Yue Gao, Meng Wang, Zheng-Jun Zha, Jialie Shen, Xuelong Li, and Xindong Wu. Visual-textual joint relevance learning for tag-based social image search. *IEEE Transactions on Image Processing*, 22(1) :363–376, 2012.

[15] James Gosling, Bill Joy, Guy L Steele, Gilad Bracha, and Alex Buckley. The java language specification, java se 8 edition. 1st, 2014.

[16] Laurent Jolia-Ferrier. *Big Data : concepts et mise en oeuvre de Hadoop*. Éditions ENI, 2014.

[17] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1) :59–69, 1982.

[18] Upmanu Lall and Ashish Sharma. A nearest neighbor bootstrap for resampling hydrologic time series. *Water Resources Research*, 32(3) :679–693, 1996.

[19] Wei Liu, Junfeng He, and Shih-Fu Chang. Large graph construction for scalable semi-supervised learning. In *ICML*, 2010.

[20] LI Rong Lu and HU Yun Fa. A density-based method for reducing the amount of training data in knn text classification [j]. *Journal of Computer Research and Development*, 4(003), 2004.

[21] Olvi L Mangasarian, W Nick Street, and William H Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4) :570–577, 1995.

[22] Mohamed Nadif. *Classification automatique et données manquantes*. PhD thesis, Université Paul Verlaine-Metz, 1991.

[23] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering : Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

[24] Klaus Schwab. The fourth industrial revolution : Crown business. *New York*, 192, 2017.

[25] Lilia Sfaxi and Aissa. *Data & Knowledge Engineering*, 2020.

[26] Dilpreet Singh and Chandan K Reddy. A survey on platforms for big data analytics. *Journal of big data*, 2(1) :8, 2015.

[27] Tarigoppula VS Sriram, M Venkateswara Rao, GV Satya Narayana, DSVGK Kaladhar, and T Pandu Ranga Vital. Intelligent parkinson disease prediction using machine learning algorithms. *International Journal of Engineering and Innovative Technology (IJEIT)*, 3(3) :1568–1572, 2013.

[28] Tamer Tulgar, Ali Haydar, and İbrahim Erşan. A distributed k nearest neighbor classifier for big data. *Balkan Journal of Electrical and Computer Engineering*, 6(2) :105–111, 2018.

[29] Xindong Wu, Chengqi Zhang, and Shichao Zhang. Efficient mining of both positive and negative association rules. *ACM Transactions on Information Systems (TOIS)*, 22(3) :381–405, 2004.

[30] Xindong Wu, Chengqi Zhang, and Shichao Zhang. Database classification for multi-database mining. *Information Systems*, 30(1) :71–88, 2005.

[31] Xindong Wu and Shichao Zhang. Synthesizing high-frequency rules from different data sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(2) :353–367, 2003.

[32] Lei Xu, Chunxiao Jiang, Jian Wang, Jian Yuan, and Yong Ren. Information security in big data : privacy and data mining. *Ieee Access*, 2 :1149–1176, 2014.

[33] Shichao Zhang, Debo Cheng, Ming Zong, and Lianli Gao. Self-representation nearest neighbor search for classification. *Neurocomputing*, 195 :137–142, 2016.

[34] Shizhao Zhang. Knn-cf approach : Incorporating certainty factor to knn classification. *IEEE Intell. Informatics Bull.*, 11(1) :24–33, 2010.

[35] Dequn Zhao, Weiwei Zou, and GuangMin Sun. A fast image classification algorithm using support vector machine. In *2010 2nd International Conference on Computer Technology and Development*, pages 385–388. IEEE, 2010.

[36] Yanchang Zhao and Shichao Zhang. Generalized dimension-reduction framework for recent-biased time series analysis. *IEEE Transactions on Knowledge and Data Engineering*, 18(2) :231–244, 2005.

[37] Xiaofeng Zhu, Zi Huang, Hong Cheng, Jiangtao Cui, and Heng Tao Shen. Sparse hashing for fast multimedia search. *ACM Transactions on Information Systems (TOIS)*, 31(2) :1–24, 2013.

[38] Xiaofeng Zhu, Zi Huang, Jiangtao Cui, and Heng Tao Shen. Video-to-shot tag propagation by graph sparse group lasso. *IEEE Transactions on Multimedia*, 15(3) :633–646, 2012.

[39] Xiaofeng Zhu, Zi Huang, Heng Tao Shen, Jian Cheng, and Changsheng Xu. Dimensionality reduction by mixed kernel canonical correlation analysis. *Pattern Recognition*, 45(8) :3003–3016, 2012.

[40] Xiaofeng Zhu, Zi Huang, Heng Tao Shen, and Xin Zhao. Linear cross-modal hashing for efficient multimedia search. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 143–152, 2013.

[41] Xiaofeng Zhu, Zi Huang, Yang Yang, Heng Tao Shen, Changsheng Xu, and Jiebo Luo. Self-taught dimensionality reduction on the high-dimensional small-sized data. *Pattern Recognition*, 46(1) :215–229, 2013.

[42] Xiaofeng Zhu, Xuelong Li, and Shichao Zhang. Block-row sparse multiview multilabel learning for image classification. *IEEE transactions on cybernetics*, 46(2) :450–461, 2015.

[43] Xiaofeng Zhu, Heung-Il Suk, and Dinggang Shen. A novel matrix-similarity based loss function for joint regression and classification in ad diagnosis. *NeuroImage*, 100 :91–105, 2014.

[44] Xiaofeng Zhu, Lei Zhang, and Zi Huang. A sparse embedding and least variance encoding approach to hashing. *IEEE transactions on image processing*, 23(9) :3737–3750, 2014.

[45] Xiaofeng Zhu, Shichao Zhang, Zhi Jin, Zili Zhang, and Zhuoming Xu. Missing value estimation for mixed-attribute data sets. *IEEE Transactions on Knowledge and Data Engineering*, 23(1) :110–121, 2010.