Université Assane Seck de Ziguinchor

UFR SCIENCES ET TECHNOLOGIES

DÉPARTEMENT INFORMATIQUE

Mémoire de fin d'étude

Pour l'obtention du diplôme de master

DOMAINE :	SCIENCES ET TECHNOLOGIES
MENTION :	INFORMATIQUE
SPÉCIALITÉ :	GÉNIE LOGICIEL

THÈME :

APPLICATION DES ALGORITHMES DE MACHINE LEARNING SUR UNE ARCHITECTURE BIG DATA

Présenté par :

Yéro NIANG

Sous la direction de :

Dr **Gorgoumack SAMBE**

Pr **Khadim DRAME**

Soutenu publiquement le 3 Août 2024 devant le jury composé de :

Youssou FAYE	Professeur Assimilé	Président et Examinateur	Université Assane Seck
Serigne DIAGNE	Maître de conférence Titulaire	Rapporteur	Université Assane Seck
Gorgoumack SAMBE	Maître de conférence Titulaire	Encadrant	Université Assane Seck
Khadim DRAME	Professeur Assimilé	Encadrant	Université Assane Seck

Année universitaire 2022-2023

✠ Résumé ✠

Ces dernières années, les données numériques échangées à travers Internet ont connu une augmentation exponentielle et une diversité sans précédent, donnant naissance au concept de Big Data. Ce phénomène, en plein essor, désigne aujourd'hui un vaste volume de données structurées et non structurées, souvent difficiles à traiter et à analyser en utilisant les technologies traditionnelles. Ces immenses volumes de données regorgent de précieuses informations qui peuvent être extraites à l'aide des algorithmes de machine learning. Cependant, l'application de ces algorithmes soulève plusieurs défis importants.

L'objectif de ce mémoire est, dans un premier temps, de réaliser une étude globale sur le machine learning en passant en revue les différents types d'apprentissage ainsi que les diverses méthodes d'analyse. Cette étude inclura notamment l'analyse de classification, où l'algorithme des K-plus proches voisins (KNN) et celui des arbres de décision (ID3) feront l'objet d'une analyse détaillée. Nous aborderons également les méthodes de régression et d'analyse par graphe.

Dans un second temps, ce mémoire explorera les concepts fondamentaux du Big Data en discutant de l'origine des 5V, en examinant les défis associés au traitement des données massives, et en présentant les technologies utilisées dans ce domaine. Une attention particulière sera portée à une étude détaillée de Hadoop, une des technologies phares du Big Data.

Enfin, une étude comparative des performances des algorithmes KNN et ID3 sera réalisée. Cette comparaison se fera en testant les algorithmes sur une machine simple, puis sur un cluster Hadoop constitué de trois nœuds. L'objectif de cette étude est de mettre en évidence les différences de performances entre une exécution sur une seule machine et une exécution sur un environnement distribué Hadoop.

Mots-clés : Big data, systèmes de fichiers, Hadoop, HDFS, mapreduce, Machine Learning, K-plus proches voisins, KNN, Arbre de décision, ID3

✠ Abstract ✠

In recent years, digital data exchanged over the Internet has experienced exponential growth and unprecedented diversity, giving rise to the concept of Big Data. This burgeoning phenomenon now refers to a vast volume of structured and unstructured data, often difficult to process and analyze using traditional technologies. These immense volumes of data contain valuable information that can be extracted using machine learning algorithms. However, the application of these algorithms presents several significant challenges.

The objective of this thesis is, first, to conduct a comprehensive study on machine learning by reviewing the different types of learning as well as various analysis methods. This study will include, notably, classification analysis, where the k-nearest neighbors (KNN) algorithm and the ID3 decision tree algorithm will be analyzed in detail. We will also address regression methods and graph analysis.

Secondly, this thesis will explore the fundamental concepts of Big Data by discussing the origin of the 5Vs, examining the challenges associated with processing massive data, and presenting the technologies used in this field. Special attention will be given to a detailed study of Hadoop, one of the key technologies in Big Data.

Finally, a comparative study of the execution times of the KNN and ID3 algorithms will be conducted. This comparison will test the algorithms on a single machine and then on a three-node Hadoop cluster. The objective of this study is to highlight the performance differences between execution on a single machine and execution in a distributed Hadoop environment.

Keywords : Big Data, file systems, Hadoop, HDFS, MapReduce, Machine Learning, k-nearest neighbors, KNN, Decision Tree, ID3

✠ Dédicaces ✠

Je dédie ce mémoire

À mon père **Moustapha** et ma mère **Banna BALDE** pour leur amour inestimable, leurs sacrifices, leur confiance, leur soutien et toutes les valeurs qu'ils ont su m'inculquer.

À mon tuteur **Maodo NIANG** pour m'avoir mis dans de bonnes conditions.

À mes frères **Bouna**, **Moro**, **Babacar** et **Maodo** sans oublier mes soeurs **Adama**, **Ndèye Marie** et **Ndèye Marceline** pour leur soutien.

À mes oncles **Mamadou SAGNA**, **Abdoul Aziz NIANG** et **Barou DRAME** pour leurs encouragements.

À mes tantes **Maimouna CISSE** et **Ndèye Marceline NIANG**.

À mes amis **Moussa SAGNA**, **Alassane KA**, **Mamadou DIALLO**, **Mamadou Yéro DIALLO**, **Nfally BADJI**, **Baciré DIAO**, **El Hadj Omar FAYE**, **Mame Cheikh Ibrahima DIAGNE**, **Seydou THIENE** et **Marame Balla BASSE** pour les bons moments passés avec vous.

À tous mes camarades de classe.

✠ Remerciements ✠

*En préambule à ce mémoire, nous exprimons notre profonde gratitude envers **Allah le Tout-Puissant** pour nous avoir donné la force et l'audace de surmonter toutes les difficultés et de mener ce travail à terme.*
*Je tiens à remercier tout particulièrement mes encadrants, Dr **Gorgoumack SAMBE** et Pr **Khadim DRAME**, pour m'avoir encouragé à choisir ce sujet de recherche, et sans qui ce mémoire n'aurait jamais vu le jour. Je vous suis également très reconnaissant de m'avoir apporté l'expertise et l'approche nécessaire à la réalisation de mon travail, mais aussi pour leur patience et le temps qu'ils ont bien voulu me consacrer.*
*Monsieur le Président et examinateur **Youssou FAYE**, Monsieur le Rapporteur **Sérigne DIAGNE**,*
Je tiens à exprimer mes sincères remerciements pour votre présence et votre participation active en tant que membres de mon jury de mémoire. Je suis particulièrement reconnaissant de l'attention que vous avez porté à mon projet et du temps que vous avez consacré à l'évaluation de mon mémoire.
Chers professeurs du département informatique,
Je tenais à vous exprimer ma profonde gratitude pour votre soutien et dévouement tout au long de mon parcours .Votre patience, votre expertise, et votre amour pour l'enseignement ont été une source de motivation pour moi. Grâce à vous, j'ai acquis des connaissances et des compétences qui me seront précieuses dans ma vie future. Je suis vraiment reconnaissant pour tout ce que vous avez fait pour moi.
Chère famille et chers amis,
Je voulais également profiter de cette occasion pour vous remercier du fond du cœur. Vos encouragements, votre amour et votre soutient ont été importants dans ma réussite. Votre présence à mes côtés m'a donné la force de persévérer dans les moments difficiles. Votre soutien et aide ne sera jamais oublié. Cela m'a profondément touché et m'a rappelé à quel point il est important d'avoir des personnes formidables comme vous dans notre entourage.
Je m'engage à utiliser cette aide de manière responsable et à poursuivre mes efforts pour atteindre mes objectifs. Votre confiance en moi renforce ma détermination à réussir et à rendre votre générosité encore plus significative en accomplissant des choses positives.

Yéro NIANG

✠ Table des matières ✠

✠ Table des figures ✠

✠ Liste des tableaux ✠

✠ Liste des Algorithmes ✠

✠ Liste des acronymes ✠

ANN Artificial Neural Network

API Application Programming Interface

CART Classification and Regression Trees

CDH Cloudera Distribution of Hadoop

CPU Central Processing Unit

DBMS Database Management System

ETL Extract, Transform, Load

HDP HortonWorks Data Platform

HDFS Hadoop Distributed File System

IA Intelligence Artificielle

IBM International Business Machines Corporation

ID3 Iterative Dichotomizer 3

IoT Internet of Things

k-NN k-Nearest Neighbors

M3 MapR

NLP Natural Language Processing

PCA Principal Component Analysis

SQL Structured Query Language

SVM Support Vector Machine

UE Union Européenne

YARN Yet Another Resource Negotiator

✠ Introduction générale ✠

Le Machine Learning, également connu sous le nom d'apprentissage automatique, constitue une composante de l'intelligence artificielle (IA) qui permet aux systèmes informatiques d'acquérir des connaissances et de s'améliorer grâce à l'expérience, sans nécessiter de programmation explicite pour chaque tâche. Cette discipline s'appuie sur des algorithmes et des modèles capables d'analyser des données pour identifier des motifs, réaliser des prédictions et prendre des décisions de manière autonome. Généralement, ces algorithmes constituent le cœur d'une application web ou mobile, représentant souvent la partie la plus « compliquée » de l'application. Pensez à la complexité de l'assistant vocal Siri d'Apple ou aux sections de recommandations de produits d'Amazon. Tous ces outils sont des applications du Machine Learning. Ces avancées en Machine Learning ont été rendues possibles par l'augmentation exponentielle de la quantité de données disponibles, ce phénomène facilite l'avènement du Big data.

En effet, le 21e siècle a été témoin d'une explosion de données, alimentée en grande partie par la numérisation croissante de l'information dans presque tous les aspects de la vie quotidienne et des activités personnelles. Les sites web, les réseaux sociaux, les applications mobiles et les transactions génèrent en permanence d'énormes volumes de données. En outre, l'avènement des capteurs et des appareils connectés contribue aussi à cette croissance exponentielle du volume de données. Cette abondance de données a mis à l'épreuve les capacités des infrastructures traditionnelles de gestion des données, conduisant à l'émergence de nouvelles approches et technologies, notamment le Big data. C'est dans ce contexte que le Big data a vu le jour. Ce dernier se réfère à des ensembles de données dont la taille dépasse la capacité de gestion des bases de données relationnelles traditionnelles pour capturer, stocker, gérer et analyser [1] ; il offre un terrain fertile pour l'application des techniques de Machine Learning. Cependant, les techniques classiques présentent plusieurs limites lorsqu'elles sont appliquées dans un environnement Big Data, car elles ne sont pas conçues pour gérer de grands volumes de données variées. C'est pourquoi il est nécessaire de les optimiser pour ces architectures Big Data.

Ce mariage entre le Big Data et le Machine Learning marque un domaine de recherche et d'application d'une importance cruciale. Les algorithmes du Machine Learning deviennent des acteurs clés dans la transformation des données massives en informations significatives, ouvrant ainsi la voie à des avancées majeures dans des domaines variés tels que la médecine, la finance, la logistique, et bien d'autres. Cependant, cette convergence soulève également des défis uniques en matière de traitement, de stockage, de qualité des données et de confidentialité, nécessitant une expertise technique et des infrastructures adéquates. Le présent mémoire a pour objectif d'explorer en profondeur les concepts de base du Big Data et du Machine Learning, ainsi que de comparer deux algorithmes d'apprentissage exécutés sur une machine simple et dans une architecture Big Data, plus particulièrement dans l'environnement Hadoop. Ce mémoire se décompose en 3 chapitres :

Dans le premier chapitre, nous aborderons d'abord le Machine Learning en discutant des approches d'apprentissage et des algorithmes d'apprentissage, avec une étude détaillée des algorithmes des K-plus proches voisins (KNN) et des arbres de décision.

Ensuite, le deuxième chapitre sera consacré à l'étude du Big Data. Nous commencerons par définir le Big Data, puis nous aborderons les origines des 5V et les technologies associées au Big Data.

Enfin, dans le troisième chapitre, nous effectuerons une étude comparative des algorithmes des K-plus proches voisins (KNN) et de ID3 des arbres de décision, en les appliquant à la fois sur une machine simple et dans un environnement Big Data tel que Hadoop.

Chapitre I

MACHINE LEARNING

1.1 INTRODUCTION

L'homme, par nature, a toujours cherché à rendre l'exécution de ses tâches plus facile en créant des outils. La créativité du cerveau humain a conduit à l'invention de différentes machines. Ces machines ont facilité la vie humaine en permettant aux gens de répondre à divers besoins de la vie, notamment les voyages, les industries et l'informatique. L'apprentissage automatique en fait partie. Le Machine Learning est le processus par lequel les systèmes informatiques apprennent à partir de données pour améliorer leur performance sur une tâche spécifique sans être explicitement programmés [2]. Il représente une révolution dans le domaine de l'informatique et de l'intelligence artificielle, offrant aux ordinateurs la capacité d'apprendre à partir de données et d'améliorer leurs performances avec l'expérience. Cette discipline s'appuie sur des algorithmes et des modèles informatiques pour découvrir des schémas et des relations dans les données, permettant ainsi de prendre des décisions autonomes et de réaliser des prédictions précises.

Ainsi, dans la suite de ce chapitre, nous allons d'abord aborder les approches d'apprentissage, puis les modèles et algorithmes, ainsi que les méthodes d'évaluation des performances des modèles. Enfin, nous examinerons quelques avancées récentes dans le domaine en analysant les applications du Machine Learning dans des domaines variés tels que la vision par ordinateur, la recommandation de produits et la prédiction de données.

1.2 Les approches d'apprentissage

Comment Google fait-il pour comprendre aussi bien et toujours mieux les requêtes complexes des utilisateurs ? Comment réussit-il à nous proposer avec toujours plus de pertinence la suite de notre recherche quand nous avons nous-même du mal à formuler notre besoin ? Ou encore, comment ChatGPT se débrouille-t-il pour comprendre notre question et y apporter la

réponse appropriée ? La réponse à toutes ces questions réside dans le Machine Learning (ML). Ce dernier est une discipline scientifique qui explore la construction et l'étude d'algorithmes permettant aux ordinateurs d'apprendre à partir de données [3]. Il existe plusieurs approches d'apprentissage ; dans la suite, nous verrons les plus fondamentaux dans le Machine Learning :

1.2.1 Apprentissage supervisé

L'apprentissage supervisé, où le modèle est entraîné sur des données étiquetées, consiste généralement à apprendre une fonction qui mappe une entrée à une sortie basée sur des exemples de paires entrée-sortie [4]. Il utilise des données d'entraînement étiquetées et une collection d'exemples d'entraînement pour déduire une fonction. L'apprentissage supervisé est effectué lorsque certains objectifs sont identifiés comme devant être atteints à partir d'un certain ensemble d'entrées [5], c'est-à-dire une approche axée sur les tâches. Les algorithmes d'apprentissage supervisé sont ceux qui nécessitent une assistance externe [6]. L'ensemble de données d'entrée est divisé en ensemble de données d'entraînement et de test. L'ensemble de données d'entraînement a une variable de sortie qui doit être prédite ou classée. Les tâches supervisées les plus courantes sont la « classification », qui sépare les données, et la « régression », qui ajuste les données. Par exemple, prédire l'étiquette de classe ou le sentiment d'un morceau de texte, comme un tweet ou une critique de produit, c'est-à-dire la classification du texte, est un exemple d'apprentissage supervisé. les domaines d'application cette approche sont : le traitement du langage naturel (NLP), les Prévisions financières, la reconnaissance vocale, etc.

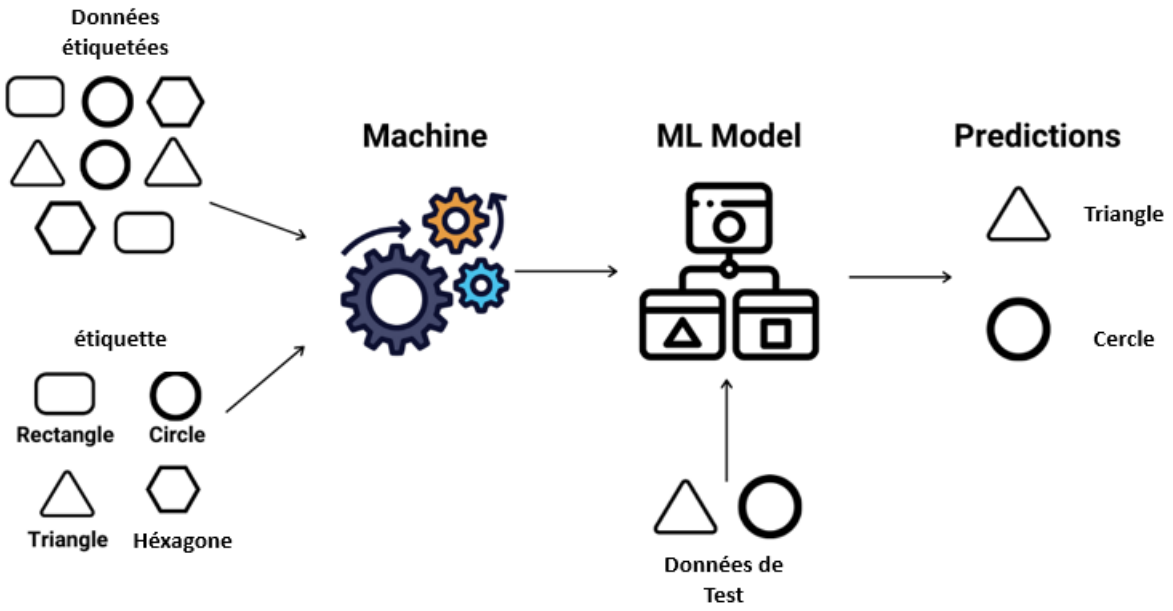

FIGURE 1.2.1 – Flux de travail d'apprentissage supervisé

Les techniques courantes de l'apprentissage supervisé sont :

— **La régression** : Elle est appliquée lorsqu'on attend une sortie continue, permettant de prédire des valeurs quantitatives comme le prix d'une maison. Le principe de la régression repose sur la recherche d'une relation entre les variables d'entrée (ou caractéristiques) et la variable de sortie (ou cible). Par exemple, pour prédire le prix d'une maison, les variables d'entrée peuvent inclure sa taille, le nombre de chambres, l'emplacement et l'âge, tandis que le prix constitue la variable de sortie. À partir de ces variables d'entrée, la régression crée un modèle mathématique capable de prédire la variable de sortie en se basant sur des données d'entraînement. Des modèles tels que la régression linéaire ou polynomiale apprennent des exemples d'entraînement pour anticiper la variable de sortie pour de nouvelles données.

— **La classification** : Elle est employée lorsqu'on attend en sortie une catégorie ou une classe, comme dans la détection de spam dans les emails en se basant sur diverses caractéristiques telles que les mots utilisés, la fréquence de certains termes ou l'adresse de l'expéditeur. En utilisant des données d'entraînement préalablement fournies au modèle, celui-ci peut prédire la catégorie de sortie pour de nouvelles données d'entrée. Divers modèles de classification sont disponibles, tels que la régression logistique, les arbres de décision, les forêts aléatoires et les machines à vecteurs de support (SVM).

1.2.2 Apprentissage non supervisé

L'apprentissage non supervisé analyse des ensembles de données non étiquetés sans nécessiter d'interférence humaine, c'est-à-dire un processus basé sur les données [4]. Les algorithmes sont laissés à eux-mêmes pour découvrir et présenter la structure intéressante des données. Cette méthode est fréquemment employée afin d'extraire des caractéristiques génératives, détecter des tendances et des structures importantes, regrouper les résultats, ainsi que pour des analyses exploratoires. Parmi les tâches d'apprentissage non supervisé les plus répandues, on retrouve la segmentation (clustering), l'estimation de la densité, l'apprentissage des caractéristiques, la réduction de la dimensionnalité et la recherche de règles d'association. la Segmentation d'images, la surveillance des systèmes), etc sont des domaines d'application de l'apprentissage non supervisé.

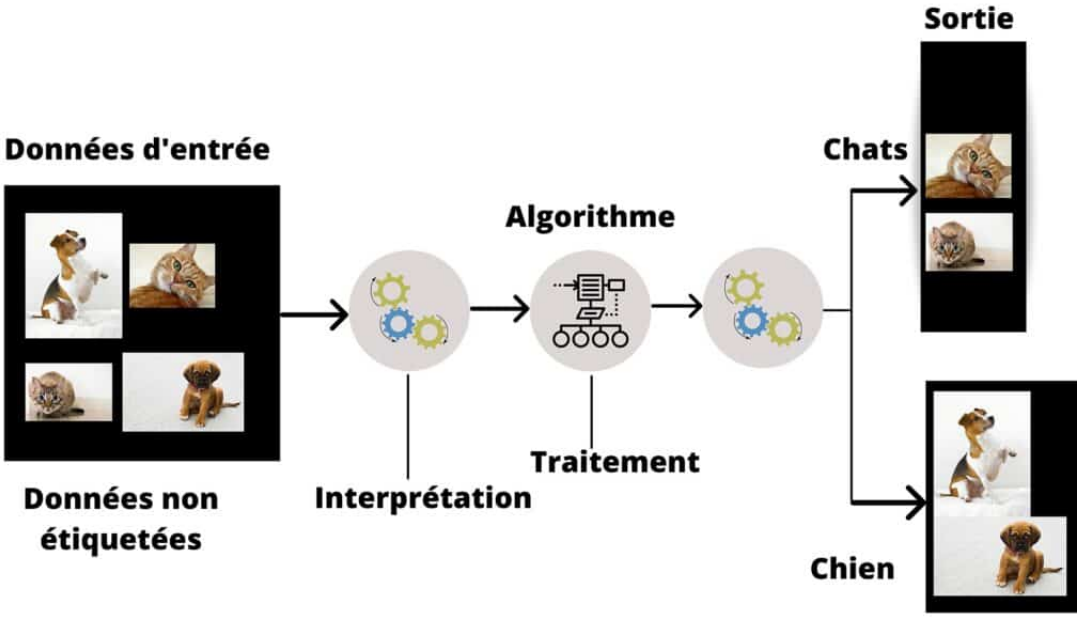

FIGURE 1.2.2 – Apprentissage non supervise [7]

1.2.3 Apprentissage semi-supervisé

L'apprentissage semi-supervisé peut être défini comme une hybridation des méthodes supervisées et non supervisées mentionnées ci-dessus, car il fonctionne à la fois sur des données étiquetées et non étiquetées [4, 5]. Il se situe ainsi entre l'apprentissage « sans supervision » et l'apprentissage « avec supervision ». Dans le monde réel, les données étiquetées pourraient être rares dans plusieurs contextes, et les données non étiquetées sont nombreuses, où l'apprentissage semi-supervisé est utile [8]. Le but ultime d'un modèle d'apprentissage semi-supervisé est

de fournir un meilleur résultat de prédiction que celui produit en utilisant uniquement les données étiquetées du modèle. Certains domaines d'application où l'apprentissage semi-supervisé est utilisé incluent la traduction automatique, la détection de fraude, l'étiquetage des données et la classification de textes.

1.2.4 Apprentissage par renforcement

L'apprentissage par renforcement est une approche d'algorithme d'apprentissage automatique qui permet aux agents logiciels et aux machines d'évaluer automatiquement le comportement optimal dans un contexte ou un environnement particulier pour améliorer son efficacité [9], c'est-à-dire une approche axée sur l'environnement. Cette approche d'apprentissage est basée sur la récompense ou la pénalité, et son objectif ultime est d'utiliser les informations obtenues auprès des militants environnementaux pour prendre des mesures visant à augmenter la récompense ou à minimiser le risque [8]. C'est un outil puissant pour développer des modèles d'intelligence artificielle qui peuvent contribuer à automatiser davantage ou à optimiser l'efficacité opérationnelle de systèmes complexes comme la robotique, les opérations de conduite autonome, et la gestion logistique de la fabrication et de la chaîne d'approvisionnement.

1.3 Les algorithmes d'apprentissage automatique

Il existe une multitude d'algorithmes d'apprentissage automatique, chacun étant spécifiquement conçu pour résoudre des types particuliers de problèmes ou s'adapter à diverses configurations de données. Nous allons examiner quelques types de tâches ainsi que les algorithmes qui les effectuent.

1.3.1 Algorithmes de classification

La classification est considérée comme une méthode d'apprentissage supervisé en apprentissage automatique, faisant référence à un problème de modélisation prédictive où une étiquette de classe est prédite pour un exemple donné [4]. En d'autres termes, l'objectif est de prédire la classe d'un nouvel exemple en se basant sur les informations apprises à partir d'exemples déjà étiquetés dans l'ensemble de données d'entraînement. C'est une tâche cruciale dans de nombreux domaines tels que la reconnaissance d'images, le traitement du langage naturel et la détection de spam. Mathématiquement, cela consiste à mapper une fonction f des variables d'entrée (x) aux variables de sortie y comme cible, étiquette ou catégorie. Cette prédiction de

classe peut être effectuée sur des données structurées ou non structurées [6]. Nous présenterons de manière exhaustive les algorithmes des k plus proches voisins (section 1.3.1.3) et les arbres de décision (section 1.3.1.4) sur lesquelles portent notre cadre d'application. Nous traiterons assez brièvement d'autres algorithmes tel que l'algorithme de Naive Bayes, Régression logistique, etc.

La tâche de classification se décompose en trois sous familles selon le nombre de classes/étiquettes à prédire :

1. **Classification binaire** : Elle désigne les tâches de classification ayant deux étiquettes de classe telles que "vrai et faux" ou "oui et non" [4]. Dans de telles tâches, une classe peut représenter l'état normal, tandis que l'autre représente l'état anormal. Par exemple, dans un test médical, "cancer non détecté" pourrait être l'état normal, et "cancer détecté" l'état anormal. De même, "spam" et "non-spam" dans les filtres de messagerie sont une forme de classification binaire [6].
2. **Classification multiclasse** : C'est une tâche d'apprentissage supervisée dans laquelle les données sont classées parmi plusieurs classes possibles. Contrairement à la classification binaire, il n'y a pas de distinction entre résultats normaux et anormaux. Au lieu de cela, les exemples sont classés parmi plusieurs classes spécifiques. Par exemple, la classification des types d'attaques réseau dans l'ensemble de données NSL-KDD [10] pourrait comporter des classes comme DoS (Denial of Service Attack), U2R (User to Root Attack), R2L (Root to Local Attack) et Probing Attack [4, 6].
3. **Classification multi-étiquettes** : En apprentissage automatique, il s'agit d'une généralisation de la classification multiclasse où chaque exemple peut être associé à plusieurs classes ou étiquettes. Ce concept est particulièrement pertinent dans des domaines comme la classification de texte multi-niveaux, où les classes peuvent être hiérarchisées et non exclusives les unes des autres. Les algorithmes avancés d'apprentissage automatique soutiennent cette capacité à prédire plusieurs étiquettes pour un même exemple [11].

Ce qui suit résume les méthodes les plus couramment utilisées et populaires dans divers domaines d'application.

1.3.1.1 Naive Bayes (NB)

Naïve Bayes est un algorithme d'apprentissage simple qui utilise la règle de Bayes en faisant l'hypothèse forte que les attributs sont conditionnellement indépendants étant donné la classe. Bien que cette hypothèse d'indépendance soit souvent violée en pratique, Naïve Bayes offre

une précision de classification compétitive. Couplé à son efficacité de calcul et à de nombreuses autres caractéristiques souhaitables, cela a conduit à une large application de Naïve Bayes dans la pratique [12]. Pour classer efficacement les instances bruitées dans les données et construire un modèle de prédiction robuste, le classificateur Naïve Bayes peut être utilisé [13]. Son principal avantage est que, comparé à des approches plus sophistiquées, il nécessite une petite quantité de données d'entraînement pour estimer rapidement les paramètres nécessaires [11]. Cependant, ses performances peuvent être affectées en raison de ses hypothèses fortes sur l'indépendance des caractéristiques. Les variantes courantes du classificateur Naïve Bayes incluent Gaussien, Multinomial, Complément, Bernoulli et Catégorique [11].

1.3.1.2 Régression logistique (LR)

Un autre modèle statistique probabiliste couramment utilisé pour résoudre les problèmes de classification en apprentissage automatique est la régression logistique (LR) [14]. En général, la régression logistique utilise une fonction logistique pour estimer les probabilités, également connue sous le nom de fonction sigmoïde, définie mathématiquement par l'équation 1.3.1.2. Elle peut présenter un surajustement avec des ensembles de données de grande dimension, mais elle fonctionne efficacement lorsque l'ensemble de données peut être séparé linéairement. La régression logistique est l'une des méthodes d'analyse statistique multivariée les plus utilisées en épidémiologie, aux côtés de la régression linéaire multiple et du modèle de Cox [15]. Elle est également appliquée dans des domaines tels que la finance, le marketing et la bio-informatique en raison de sa capacité à modéliser des relations complexes entre les variables.

$$g(z) = \frac{1}{1+e^{-z}} \quad (1)$$

1.3.1.3 K-plus proches voisins (KNN)

1. **Présentation de l'algorithme K -Nearest Neighbour**

   K-Nearest-Neighbours ou KNN est un algorithme de classification simple proposé pour la première fois par Evelyn Fix et Joseph Hodges [16] en 1951 et développé par Thomas Cover [17] en 1967. Cet algorithme stocke toutes les données d'entrée avec leurs étiquettes correspondantes et classe une nouvelle observation basée sur la similarité [18]. Cette classification se fait à partir des étiquettes de ses plus proches voisins. L'algorithme KNN a été utilisé sur plusieurs applications-cadres de l'Industrie 4.0, telles que la cybersécurité [19], la prédiction de la durée de vie utile des avions [20], la classification des pannes [21], la

prédiction de la néphropathie chez les enfants [22], les systèmes de détection d'intrusion [23]. Le KNN représente l'un des algorithmes de classification statistique les plus couramment utilisés, visant à classifier les objets en se basant sur les exemples d'apprentissage les plus proches dans l'espace des caractéristiques. C'est un algorithme d'apprentissage paresseux, ce qui signifie que la fonction KNN est évaluée localement et que tous les calculs sont différés jusqu'à la phase de classification. Pendant la phase d'entraînement, aucun modèle ou apprentissage effectif n'est réalisé ; cependant, un ensemble de données d'entraînement est nécessaire pour remplir un échantillon de l'espace de recherche avec des instances dont la classe est connue, d'où le nom d'« apprentissage paresseux » de cet algorithme. Cela implique que les points de données d'entraînement ne contribuent pas à la généralisation, et ils sont tous requis pendant la phase de test. Lorsqu'une instance à classifier est présentée, l'algorithme calcule ses K voisins les plus proches, et la classe est déterminée par un vote parmi ces voisins. Dans l'algorithme KNN, la phase d'entraînement est très rapide mais la phase de test est coûteuse en termes de temps et de mémoire [24]. Afin de mieux comprendre le fonctionnement de KNN, prenons un bref exemple :
Des données sur la taille et le poids des hommes et des femmes ont été recueillies et sont présentées dans la figure 1.3.3a. Les points bleus représentent les données étiquetées comme hommes et les rouges comme femmes. Supposons ensuite que nous ajoutions un nouvel individu à l'ensemble de données. On nous dit que ce nouvel individu est un homme ; cependant, nous introduisons uniquement leur taille et leur poids dans l'algorithme pour voir s'il est capable de les classer correctement. Ceci est illustré sur la figure 1.3.3b , où le nouvel individu est représenté par une étoile.

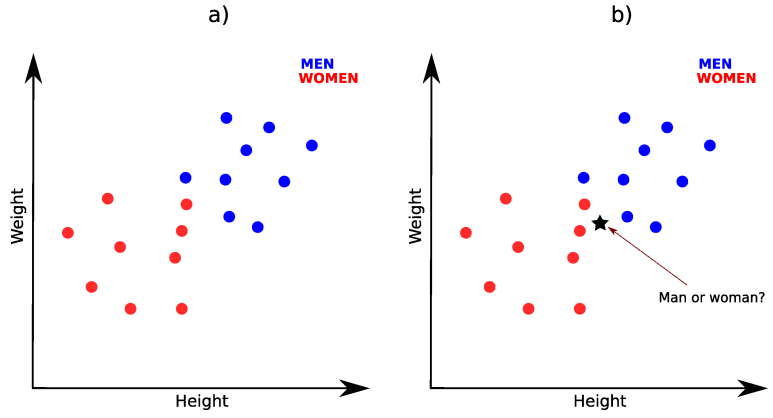

FIGURE 1.3.3 – (a) Répartition des hommes et des femmes selon la taille et le poids. (b) Nouvelle observation ajoutée à l'ensemble de données.

2. **Paramètre K et métrique de distance**

L'un des atouts de la méthode KNN pour la classification des objets réside dans sa faible dépendance à l'égard des paramètres à ajuster, principalement K et la métrique de distance, pour atteindre un niveau de précision de classification satisfaisant. Ainsi, lors des implémentations basées sur KNN, choisir judicieusement K et la métrique de distance est crucial pour calculer la distance la plus proche.

En général, des valeurs plus élevées de K réduisent l'impact du bruit sur la classification, mais peuvent rendre les frontières entre les classes moins nettes. Lorsque K est égal à 1, ce qui correspond au cas où la classe est définie comme celle de l'échantillon d'apprentissage le plus proche, cela est désigné sous le nom d'algorithme du plus proche voisin. Dans le cadre des problèmes de classification binaire, il est recommandé de choisir un K impair pour éviter les situations où les votes sont à égalité. Ainsi, la valeur de K est définie de manière à produire le taux de classification correcte le plus élevé possible [24].

3. **Mesures de distance** Il existe plusieurs types de fonctions qui peuvent être utilisées pour obtenir la distance entre les points, telles que la mesure de similarité cosinus, Minkowsky, la corrélation, le chi carré et la distance euclidienne [25]. Étant donné deux points $P = (p_1, p_2, ..., p_n)$ et $Q = (q_1, q_2, ..., q-n)$. Les différentes distances sont définies comme suit :

   — **Métrique de distance euclidienne (UE)** : C'est la manière la plus courante de calculer une distance entre deux objets. Elle calcule la racine des différences carrées entre les coordonnées d'une paire d'objets et est défini à l'aide de l'équation suivante [26] :

$$d(P,Q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \quad (2)$$

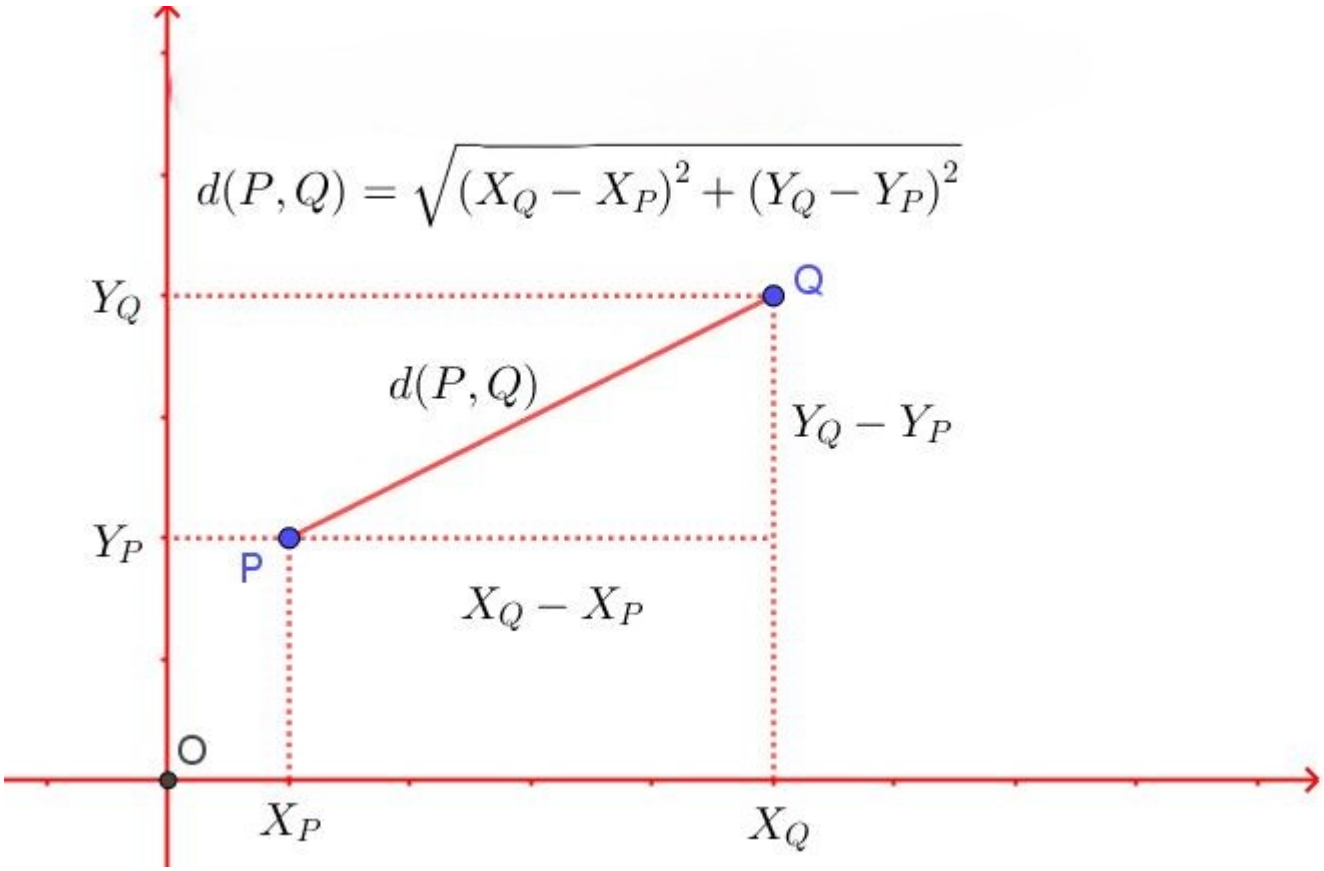

FIGURE 1.3.4 – distance euclidienne (UE)

— **Métrique de distance du pâté de maisons (CB)** : Elle est basée sur la géométrie Taxicab, considérée pour la première fois par Hermann Minkowski au 19ème siècle, est une forme de géométrie dans laquelle la métrique habituelle de la géométrie euclidienne est remplacée par une nouvelle métrique dans laquelle la distance entre deux points est la somme des différences absolues. de leurs coordonnées définies à l'aide de l'équation suivante :

$$d_{P,Q} = \sum_{i=1}^{n} |p_i - q_i| \quad (3)$$

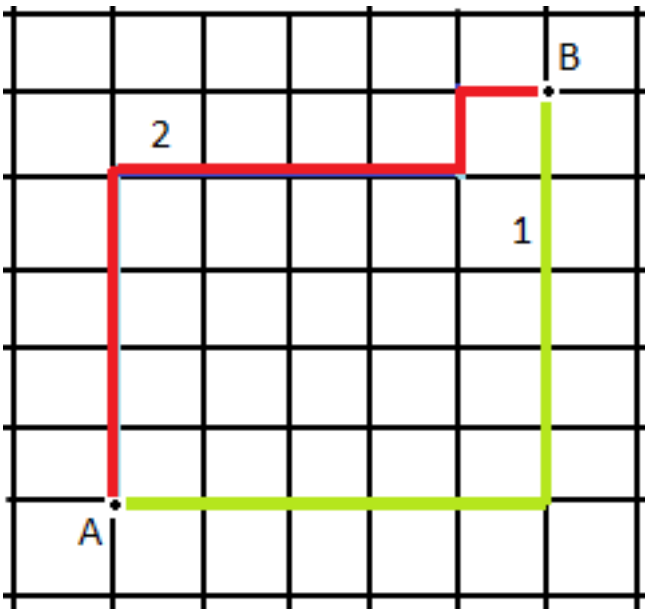

FIGURE 1.3.5 – distance de manathan

Dans l'image ci-dessus, imaginez chaque cellule comme un bâtiment et les lignes de la grille comme des routes. Maintenant, si nous voulons voyager du point A au point B marqué dans l'image et suivre le chemin rouge ou jaune. On voit que le chemin n'est pas droit et qu'il y a des virages. Dans ce cas, nous utilisons la métrique de distance de Manhattan pour calculer la distance parcourue.

Il existe d'autres métriques, comme la métrique de distance de corrélation (CO), qui mesure la similarité en soustrayant la corrélation entre les points (traités comme des séquences de valeurs).

4. **L'algorithme kNN**

Algorithm 1 Algorithme kNN

```
Variables :
D : liste données
x : exemple à classifier
C, L, F : collection de données
k : entier
Debut
Initialisation(k)
for x_i in D do
    d_i = distance(x_i,x)
    C = Add( d_i, Label(d_i))
end for
C = short(C)
L = Sélect(k, C)
F = Count(label, L)
if régression ; return avarage(L étiquettes)
if classification ; return F.best(Label) =0
```

Une fois que nous avons obtenu la distance entre la nouvelle observation et les autres points, nous pouvons choisir les voisins les plus proches pour procéder à la classification. Cependant, une question importante se pose ici : combien de voisins les plus proches devons-nous prendre en compte ? Prenons la figure 1.3.6 comme exemple.

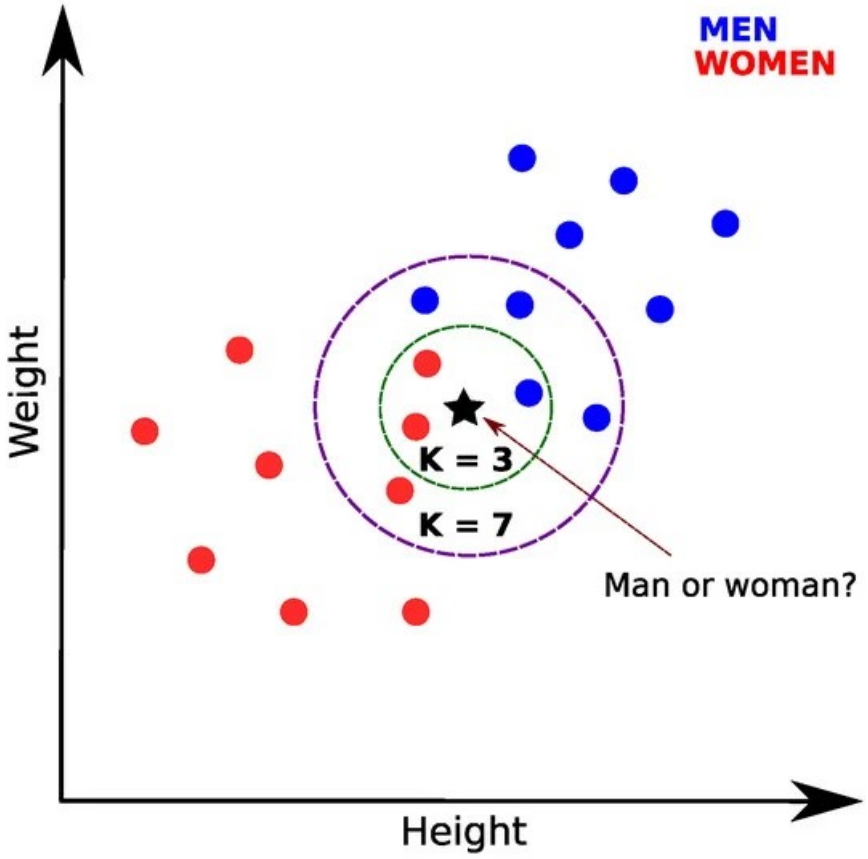

FIGURE 1.3.6 – Classification basée sur différentes valeurs de K

Pour K=3, les voisins les plus proches sont ceux à l'intérieur du cercle vert, et pour K=7, les voisins les plus proches se trouvent à l'intérieur du cercle violet. On observe que lorsque K=3, deux de ces trois voisins sont des femmes, donc la nouvelle observation serait classée comme femme. Cependant, lorsque K=7, la plupart des voisins les plus proches sont des hommes, et le nouvel individu serait alors classé comme un homme. Ainsi, pour le classement de cet individu, K=7 est meilleur que K=3 car il le classe correctement comme un homme, correspondant à son véritable sexe. Néanmoins, cela ne signifie pas que K=7 est la meilleure valeur globale pour l'ensemble de données ; il est nécessaire de classer davantage de nouvelles observations avec différentes valeurs de K pour déterminer celle offrant les meilleures performances.

Comme nous pouvons le voir, la classification résultante de l'algorithme KNN dépend fortement du nombre choisi de voisins les plus proches. Ainsi, l'algorithme doit être testé avec

différentes valeurs de K jusqu'à ce que nous trouvions la valeur la plus optimale pour l'ensemble de données avec lequel nous travaillons. C'est généralement la tâche la plus exigeante en termes de temps lorsqu'on parle de l'algorithme KNN. Une fois la valeur optimale de K trouvée, le temps de formation est généralement faible, ce qui constitue l'un des principaux avantages par rapport aux autres méthodes de classification.

Forces et faiblesses de KNN

Étant l'un des algorithmes les plus utilisés dans le domaine, KNN présente à la fois des avantages et des inconvénients.

1. **Les avantages de la méthode des k plus proches voisins**
   - L'algorithme KNN est sensible envers des données bruitées [27].
   - La méthode des k plus proches voisins est efficace si les données sont larges et incomplètes [28].
   - Cette méthode est l'une des plus simples de tous les algorithmes d'apprentissage automatique [29].
2. **Inconvénients de la méthode des k plus proches voisins**
   - Le besoin de déterminer la valeur du nombre des plus proches voisins (le paramètre k).
   - . Le temps de prédiction est très long puisqu'on doit calculer la distance de tous les exemples [27].
   - Cette méthode est gourmande en espace mémoire car elle utilise une grande capacité de stockage pour le traitement des corpus [30].

1.3.1.4 Arbre de décision

1. **Présentation**

   L'exploration de données consiste à extraire des informations d'un ensemble de données et à les transformer en une structure compréhensible [31]. Les arbres de décision font partie des algorithmes d'exploration de données les plus utilisés. Un arbre de décision est un organigramme élaboré autour d'une idée principale qui se subdivise ensuite en fonction des conséquences de vos décisions. Ce sont des structures de données arborescentes où chaque nœud interne représente un test sur une caractéristique, chaque branche représente le résultat de ce test, et chaque feuille représente une classe de décision [32]. Les arbres de décision peuvent à la fois réaliser des tâches de classification et de régression. Un arbre de

décision est un classificateur sous la forme d'une structure arborescente où chaque nœud est soit :

— **Nœud racine** est le point de départ de la construction de l'arbre et représente la caractéristique la plus importante pour diviser les données d'entraînement lors de la construction de l'arbre

— **Nœud de décision** spécifie tous les tests possibles sur une seule valeur d'attribut, avec une branche et un sous-arbre pour chaque résultat possible du test [33].

— **Nœud feuille** est un indicateur de la valeur de attribut cible (classe) d'exemples

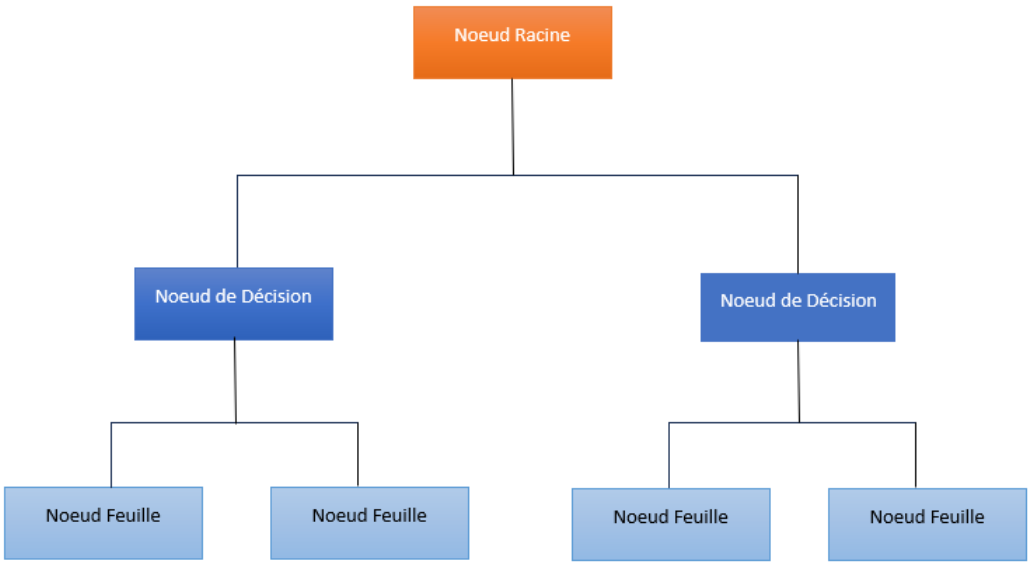

FIGURE 1.3.7 – Structure d'un arbre de décision

Comme vous pouvez le voir sur la figure 1.3.7, Un arbre de décision commence par un nœud racine qui ne possède pas de branches entrantes. À partir de ce nœud racine, des branches se déploient vers les nœuds internes, également appelés nœuds de décision. Ces nœuds, basés sur les caractéristiques disponibles, réalisent des évaluations sur des sous-ensembles homogènes, identifiés comme nœuds feuilles ou terminaux. Les nœuds feuilles décrivent tous les résultats possibles à l'intérieur de la structure de l'arbre

2. **Sélection du meilleur attribut**

   Il existe plusieurs approches pour choisir le meilleur attribut à chaque nœud, mais deux méthodes principales, le gain d'information et l'indice de Gini, servent de critères de fractionnement dans les modèles d'arbres de décision. Ils permettent d'évaluer la qualité de chaque test conditionnel et sa capacité à classer les échantillons dans une classe.

— **Entropie**

Il est complexe d'aborder le concept de gain d'information sans d'abord discuter de l'entropie. L'entropie, issue de la théorie de l'information, quantifie l'incertitude ou l'impureté des valeurs échantillonnées. Soit une variable X avec un ensemble de valeurs possibles $x_1, x_2, ..., x_n$ et P_i la probabilité de x_i, l'entropie est définie comme :

$$Entropie(X) = -\sum_{i=1}^{n} P(x_i) log_2 P(x_i) \quad (4)$$

Les valeurs d'entropie sont généralement comprises entre 0 et 1. Une entropie de 0 signifie que tous les échantillons du jeu de données, noté S, appartiennent à une seule classe, ce qui indique une pureté maximale. À l'inverse, une entropie de 1 se produit lorsque les échantillons sont répartis uniformément entre différentes classes, ce qui signifie une impureté maximale. Pour choisir la meilleure fonction de fractionnement et construire un arbre de décision optimal, on cherche l'attribut qui minimise le total de l'entropie. Le gain d'informations mesure la réduction d'entropie obtenue après un fractionnement sur un attribut spécifique. Un attribut avec un gain d'informations élevé est préféré car il permet de mieux organiser les données selon leur classe cible.

— **Impureté Gini**

L'impureté de Gini est une mesure couramment utilisée pour construire des arbres de décision en classification. Elle offre une vision détaillée de la répartition des données au niveau de chaque nœud de l'arbre, ce qui la distingue de la simple précision de classification qui évalue la précision globale de l'arbre.

Pour calculer l'impureté d'un nœud dans un arbre de classification, on examine le nombre d'occurrences de chaque classe cible parmi tous les enregistrements associés à ce nœud. L'impureté de Gini totale est déterminée en calculant la somme des carrés des proportions de chaque classe cible dans le nœud, puis en soustrayant ce résultat de 1. Le total obtenu est ensuite multiplié par le nombre total d'enregistrements dans le nœud. Pour un nœud donné avec k classes, l'impureté de Gini est défini comme :

$$GiniImpurity = 1 - \sum_{i} (Pi)^2 \quad (5)$$

P_i est la proportion d'éléments appartenant à la classe i dans le noeud.

3. **Les types d'arbres de décision**

L'algorithme de Hunt, développé dans les années 1960 pour modéliser l'apprentissage hu-

main en psychologie, constitue la fondation de nombreux algorithmes d'arbre de décision bien connus, notamment :

— **ID3 :**

ID3 est un algorithme développé par Ross Quinlan utilisé pour générer un arbre de décision à partir d'un ensemble de données [34]. ID3 est l'abréviation de « ***Iterative Dichotomiser 3*** ». Cet algorithme utilise l'entropie et le gain d'information comme mesures pour évaluer les fractionnements candidats. Pour construire un arbre de décision, ID3 utilise une recherche gourmande de haut en bas à travers les ensembles de données, où chaque attribut à chaque nœud de l'arbre est testé pour sélectionner celui qui convient le mieux à la classification de l'ensemble donné [35].

— **C4.5 :**

Cet algorithme est considéré comme une évolution ultérieure d'ID3, également développée par Quinlan [36]. Il peut utiliser des mesures comme le gain d'information ou le ratio de gain pour évaluer les points de fractionnement au sein des arbres de décision. C4.5 génère des arbres de décision utilisés principalement pour la classification, ce qui lui vaut souvent le titre de classificateur statistique. Il est supérieur à l'algorithme ID3 car il gère à la fois les attributs continus et discrets ainsi que les valeurs manquantes, et effectue un élagage des arbres après leur construction.

— **CART :**

Le terme CART est une abréviation pour "Classification and Regression Trees" et a été introduit par Leo Breiman [37]. Cet algorithme utilise typiquement l'indice de Gini pour identifier l'attribut idéal pour effectuer le fractionnement. L'indice de Gini mesure la fréquence à laquelle un attribut choisi au hasard est mal classé. Lors de l'évaluation à l'aide de l'indice de Gini, une valeur inférieure est plus souhaitable.

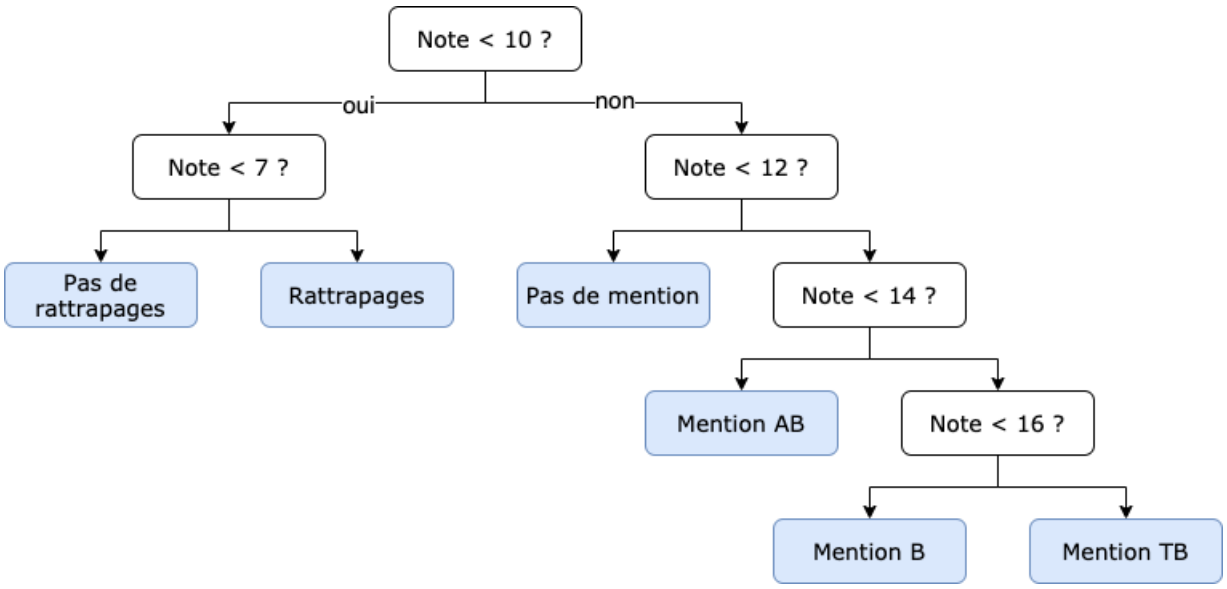

FIGURE 1.3.8 – classification des notes avec un arbre

Algorithm 2 Algorithme d'un arbre de décision

Entrée : exemples ← liste d'exemples étiquetés
Entrée : attributs ← liste des attributs non utilisés
Sortie : nœud de l'arbre de décision
Déclaration : *attribut* ← premier attribut de la liste *attributs*
Déclaration : n ← nœud courant
Déclaration : e ← sous-ensemble d'exemples
Déclaration : *valeur* ← valeur possible de *attribut*
if *exemples* = ∅ **then**
    return nœud vide
end if
if classe(*exemples*).unique() **then**
    return feuille(*classe*)
end if
if *attributs* = ∅ **then**
    return feuille(classe-majoritaire(*exemples*))
end if
attribut ← premier-élément(*attributs*)
n ← nœud(*attribut*)
for each *valeur* in *attribut* **do**
    e ← sous-ensemble(*exemples*, *attribut* = *valeur*)
    add-fils(n : appel-récursif(arbre(e, *attributs* \ {*attribut*}))
end for
return n =0

Forces et faiblesses des arbre de décision

Les agorithmes des arbre de décision ont des avantages et des limites cités ci-dessous :

4. **Avantages des arbres de décision :**

— **Facile à interpréter :** La simplicité de la logique booléenne et la visualisation graphique des arbres de décision en font des outils plus accessibles et plus conviviaux. Leur structure hiérarchique permet de déterminer rapidement les attributs les plus in-

fluents, un aspect parfois moins évident dans d'autres méthodes telles que les réseaux neuronaux.

— **Peu ou pas de préparation de données nécessaire :** Les arbres de décision se distinguent par leur flexibilité, offrant la possibilité de traiter divers types de données, qu'elles soient discrètes ou continues. Ils peuvent aussi convertir des valeurs continues en catégorielles en utilisant des seuils. De plus, leur capacité à gérer les valeurs manquantes les rend adaptés à des situations où d'autres classificateurs, comme Naïve Bayes, pourraient rencontrer des difficultés.

— **Plus flexible :** Les arbres de décision offrent une polyvalence remarquable, car ils peuvent être utilisés à la fois pour la classification et la régression, les rendant ainsi plus souples que certains autres algorithmes. De plus, ils sont insensibles aux relations sous-jacentes entre les attributs ; ainsi, en cas de forte corrélation entre deux variables, l'algorithme ne sélectionne qu'une seule des caractéristiques pour le fractionnement.

5. **Inconvénient des arbres de décision**

— **Sujet au surajustement :** Les arbres de décision complexes ont tendance à souffrir de surajustement et ne généralisent pas bien aux nouvelles données. Pour éviter ce problème, des techniques de pré-élagage et de post-élagage sont souvent utilisées. Le pré-élagage interrompt la croissance de l'arbre lorsqu'il n'y a pas suffisamment de données, tandis que le post-élagage consiste à supprimer les sous-arbres avec des données insuffisantes après la construction de l'arbre.

— **Estimateurs à variance élevée :**De légères variations dans les données peuvent entraîner des arbres de décision très différents. Le bagging, qui consiste à moyenner les estimations, peut être utilisé pour réduire la variance des arbres de décision. Cependant, cette méthode a ses limites car elle peut conduire à des prédicteurs fortement corrélés.

— **Plus coûteux :** Comme les arbres de décision adoptent une approche de recherche gloutonne lors de leur construction, ils peuvent être plus coûteux à entraîner par rapport à d'autres algorithmes.

1.3.2 Algorithmes de régression

L'analyse de régression comprend plusieurs méthodes d'apprentissage automatique qui permettent de prédire une variable de résultat continue y en fonction de la valeur d'une ou plusieurs

variables prédictives x [4]. La distinction la plus significative entre la classification et la régression est que la classification prédit des étiquettes de classe distinctes, tandis que la régression facilite la prédiction d'une quantité continue [6]. La figure 1.3.9 illustre la distinction entre la classification et les modèles de régression, avec parfois des zones de chevauchement entre ces deux types d'algorithmes d'apprentissage automatique. Les modèles de régression sont largement utilisés dans divers domaines tels que les prévisions financières, l'estimation des coûts, l'analyse des tendances, le marketing, la modélisation de séries chronologiques, la réponse aux médicaments, etc. Parmi les types courants d'algorithmes de régression, on trouve la régression linéaire, polynomiale, les arbres de décisions comme nous l'avons souligné plus haut et beaucoup d'autres algorithmes .Nous nous limiterons ici à présenter la regression linéaire et la regression polynomiale

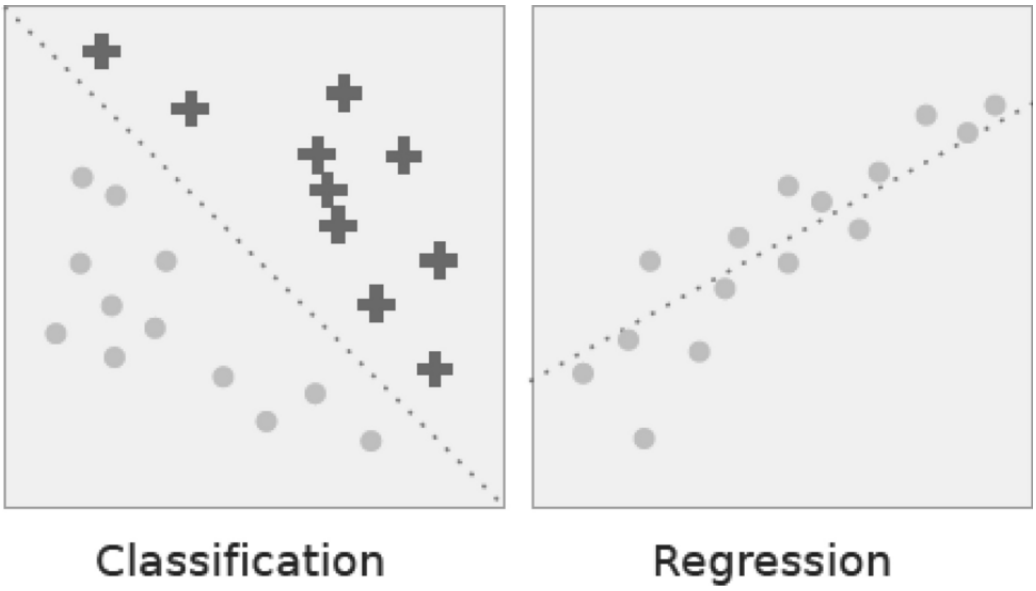

FIGURE 1.3.9 – Classification vs régression

1.3.2.1 Régression linéaire simple et multiple

C'est l'une des méthodes de modélisation les plus utilisées en apprentissage automatique, ainsi qu'une technique de régression largement reconnue. Elle implique une variable dépendante continue, des variables indépendantes pouvant être continues ou discrètes, et une forme de droite de régression linéaire. La régression linéaire crée une relation entre la variable dépendante (Y) et une ou plusieurs variables indépendantes (X) en utilisant la ligne droite qui s'ajuste le mieux [4]. Elle est défini par les équations suivantes :

$$y = a + bx + e \quad (6)$$

$$y = a + b_1x_1 + b_2x_2 + ... + b_nx_n + e \quad (7)$$

Dans l'équations 6, a représente l'ordonnée à l'origine, b désigne la pente de la droite, et e représente le terme d'erreur. Cette formule est employée pour estimer la valeur de la variable cible en fonction des variables prédictives fournies. La régression linéaire multiple est une extension de la régression linéaire simple qui permet à deux ou plusieurs variables prédictives de modéliser une variable de réponse, y, en tant que fonction linéaire [4] définie dans l'équation 7 , alors que la régression linéaire simple n'a qu'une seule variable indépendante, définie dans l'équation 6.

1.3.2.2 Régression polynomial

La régression polynomiale est une forme d'analyse de régression dans laquelle la relation entre la variable indépendante x et la variable dépendante y n'est pas linéaire, mais correspond au degré polynomial de x [11]. La formulation de la régression polynomiale découle également de l'équation de régression linéaire (qui correspond à une régression polynomiale de degré 1), laquelle est définie comme suit :

$$y = b_0 + b_1x + b_2x^2 + b_3x^3 + ... + b_nx^n + e \quad (8)$$

Dans cette équation, y représente la sortie prédite ou cible, b sont les coefficients de régression, et x est une variable indépendante ou d'entrée. En d'autres termes, lorsque les données ne suivent pas une distribution linéaire mais plutôt un certain degré de polynôme, la régression polynomiale est utilisée pour obtenir le résultat désiré.

1.3.3 Algorithmes de grappe

La méthode de regroupement, aussi appelée clustering, est une approche d'apprentissage automatique non supervisée visant à détecter et à regrouper des points de données similaires au sein de vastes ensembles de données sans viser un résultat spécifique prédéfini. Il regroupe une collection d'objets de telle manière que les objets de la même catégorie, appelés cluster, soient en quelque sorte plus similaires les uns aux autres que les objets des autres groupes [4]. Souvent employée comme méthode d'analyse de données, elle vise à découvrir des tendances ou des motifs significatifs dans les données, comme par exemple la segmentation de groupes de consommateurs en fonction de leurs comportements. Elle trouve des applications diverses dans des domaines tels que la cybersécurité, le commerce électronique, le traitement des données

mobiles, l'analyse de la santé, la modélisation des utilisateurs et l'analyse comportementale. Il existe différents types de méthodes de clustering qui sont :

— **Méthodes de partitionnement** : En se basant sur les caractéristiques et les similarités des données, cette méthode de clustering classe les données en différents groupes ou clusters. Les spécialistes des données ou les analystes définissent généralement le nombre de clusters de manière dynamique ou statique, en fonction des besoins spécifiques des applications visées.

— **Méthodes basées sur la densité** : Pour détecter des groupes ou des clusters distincts, cette méthode se base sur le principe selon lequel un cluster dans l'espace de données est une zone continue avec une densité élevée de points, séparée des autres clusters de ce type par des zones avec une densité de points plus faible. Les points qui ne sont pas inclus dans un cluster sont considérés comme du bruit.

— **Méthodes basées sur la hiérarchie** : Le clustering hiérarchique vise généralement à construire une structure arborescente de clusters, appelée hiérarchie. Les stratégies de clustering hiérarchique sont généralement divisées en deux types : (i) Agglomératif : une approche "ascendante" où chaque observation démarre dans son propre cluster et les paires de clusters sont combinées progressivement pour former des clusters plus grands dans la hiérarchie, et (ii) Divisif : une approche "descendante" où toutes les observations débutent dans un seul cluster et les divisions sont réalisées de manière récursive pour créer des sous-clusters, descendant dans la hiérarchie comme le montre la figure 1.3.10.

— **Méthodes basées sur une grille** : Pour gérer de vastes ensembles de données, le clustering basé sur une grille est spécialement adapté. Cette méthode consiste d'abord à résumer les données avec une grille, puis à combiner les cellules de cette grille pour former les clusters.

— **Méthodes basées sur un modèle** : Il existe principalement deux types d'algorithmes de clustering basés sur un modèle : l'un qui utilise l'apprentissage statistique et l'autre basé sur une méthode d'apprentissage des réseaux neuronaux [38].

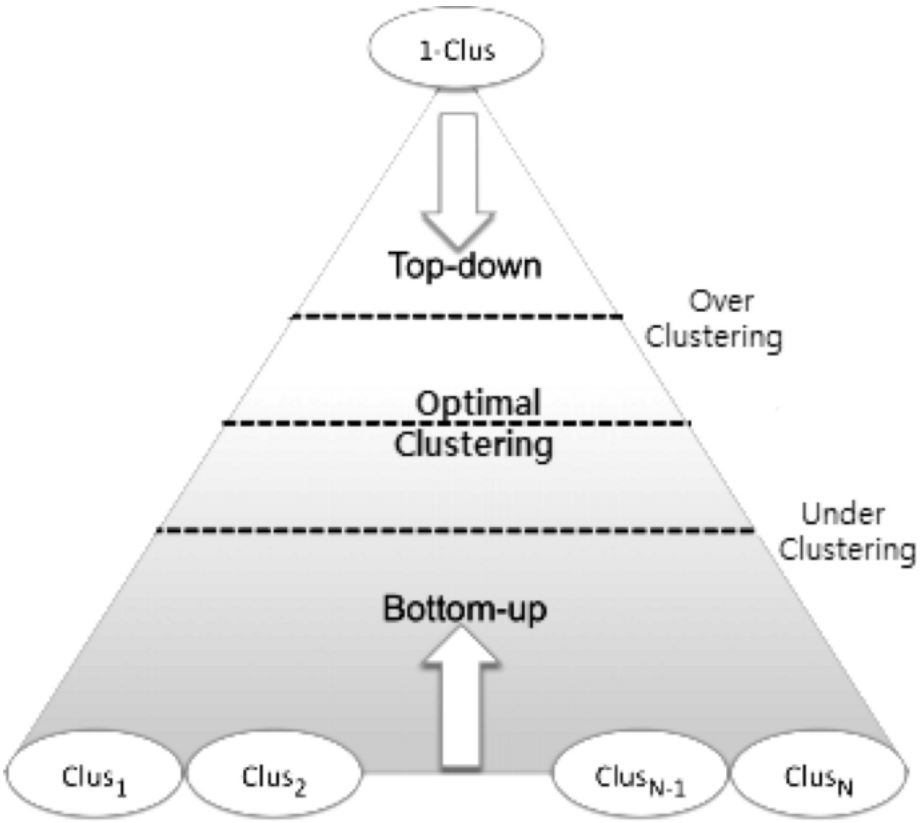

FIGURE 1.3.10 – clustering hiérarchique

De nombreux algorithmes de clustering ont été développés pour regrouper des données dans les domaines de l'apprentissage automatique. Dans ce qui suit, nous présentons une synthèse des méthodes populaires largement utilisées dans divers domaines d'application.

1.3.3.1 Clustering K-means

L'algorithme de clustering K-means est réputé pour sa rapidité, sa robustesse et sa simplicité, produisant des résultats fiables lorsque les ensembles de données sont distinctement séparés. Il assigne les points de données à des clusters de manière à minimiser la somme des carrés des distances entre les points et les centroïdes. En d'autres termes, K-means identifie d'abord k centroïdes, puis assigne chaque point au cluster dont le centroïde est le plus proche, tout en maintenant les centroïdes aussi compacts que possible. En raison de son initialisation aléatoire, les résultats peuvent varier, et sa sensibilité aux valeurs extrêmes peut entraîner des résultats inconséquents. Le clustering K-medoids [39] est une variante de K-means qui est plus robuste aux bruits et aux valeurs aberrantes.

Algorithm 3 Algorithme de Clustering K-means

Entrée : $K \leftarrow$ nombre de clusters
Entrée : $P \leftarrow \{p_1, p_2, \ldots, p_n\}$ ensemble de points
Sortie : clusters $C = \{c_1, c_2, \ldots, c_K\}$
Choisir-centres $(c_1, c_2, \ldots, c_K)$
while clusters-changent() **do**
  for $i = 1$ à n **do**
    dist(p_i, c_j)
    Affecter(p_i, plus-proche(c_j))
  end for
  for $j = 1$ à K **do**
    $n_j \leftarrow$ nombre-points(c_j)
    $c_j \leftarrow \frac{1}{n_j} \sum_{p_i \in c_j} p_i$
  end for
end while
return clusters C =0

1.3.3.2 Clustering par décalage moyen

Le clustering par décalage moyen est une méthode non paramétrique qui n'exige pas de connaître à l'avance le nombre de clusters ni de contraintes sur leur forme. Cette méthode vise à découvrir des "blobs" dans une distribution ou une densité lisse d'échantillons [11]. C'est un algorithme centré sur les centroïdes qui procède en ajustant les candidats centroïdes pour qu'ils représentent la moyenne des points dans une région donnée. Pour obtenir l'ensemble final de centroïdes, ces candidats sont filtrés lors d'une étape de post-traitement pour éliminer les doublons quasi-identiques. Des domaines d'application tels que l'analyse de clusters en vision par ordinateur et en traitement d'images en bénéficient. Cependant, Mean Shift présente l'inconvénient d'être intensif en calcul et de ne pas fonctionner efficacement dans les cas de grande dimension où le nombre de clusters change brusquement.

Algorithm 4 Clustering par Décalage Moyen

Entrée : $X = \{x_1, x_2, \ldots, x_n\}$ ensemble de données, h rayon
Sortie : clusters
for each x in X **do**
  $x_i \leftarrow x$
  repeat
    avarage(x_i in h)
    Déplacer(x_i)
  until stabilisation(x_i)
  Assign(x_i)
end for
return clusters =0

1.3.3.3 Clustering hiérarchique aggloméré

La méthode de clustering la plus fréquemment utilisée pour regrouper des objets en clusters en fonction de leur similarité est le clustering agglomératif. Cette approche utilise une méthode ascendante, où chaque objet est initialement considéré comme un cluster singleton par l'algorithme. Ensuite, les paires de clusters sont fusionnées successivement jusqu'à ce que tous les objets appartiennent à un seul grand cluster. Le résultat est un dendrogramme, une représentation arborescente des éléments.

1.4 CONCLUSION

En conclusion, le Machine Learning représente une avancée transformative dans notre capacité à extraire des informations significatives à partir d'un ensembles de données. À mesure que nous continuons à exploiter les potentiels du Machine Learning, il devient clair que son rôle dans la prise de décisions éclairées et l'innovation continue de croître, ouvrant ainsi de nouvelles perspectives passionnantes pour l'avenir de l'analyse des données à grande échelle.

Chapitre II

BIG DATA

2.1 INTRODUCTION

Depuis les années 2000, on assiste à une augmentation exponentielle des données et à l'émergence du big data. Il y a plus de smartphones en circulation dans le monde que d'humains ; près de 8 milliards de téléphones sont utilisés, avec environ 30 milliards d'éléments de contenu partagés sur Facebook par mois. À titre d'exemple, lors du débat entre le président Obama et le gouverneur Mitt Romney le 4 octobre 2012, plus de 10 millions de tweets ont été déclenchés dans les deux heures qui ont suivi [40]. Le big data est un terme pratique utilisé pour décrire un ensemble de concepts et de technologies qui traitent efficacement les données à une très grande échelle [41].

2.2 Définition du Big Data

Même s'il est aujourd'hui omniprésent, le concept de « big data » est naissant et ses origines sont incertaines. Diebold affirme que le terme « big data... est probablement né de conversations autour d'un déjeuner chez Silicon Graphics Inc. (SGI) au milieu des années 1990, dans lesquelles John Mashey figurait en bonne place » [42]. La figure 2.6.22 montre que le terme s'est répandu seulement en 2011. Le big data constitue l'une des frontières actuelles et futures de la recherche. Par exemple, Gartner a répertorié les « 10 principales tendances technologiques stratégiques pour 2013 » et les « 10 principales tendances technologiques critiques pour les cinq prochaines années », avec le big data figurant dans les deux listes [43].

Pour définir le Big Data, il existe différentes explications allant des 3 V aux 5 V. Doug Laney a introduit le volume, la vitesse et la variété, connus sous le nom de 3 V [44]. Selon les besoins individuels, certaines personnes ajoutent parfois un quatrième V, qui peut être la valeur, la variabilité ou le virtuel. Plus couramment, le Big Data désigne une collection d'ensembles

de données très volumineux avec une grande diversité de types, rendant difficile leur traitement avec des approches traditionnelles de gestion et de traitement des données [43]. Cette définition met en lumière l'ampleur et la diversité des données générées à une vitesse exponentielle à partir de diverses sources telles que les réseaux sociaux, les appareils connectés et les systèmes d'information. Le Big Data représente donc un défi majeur pour les entreprises et les organisations, nécessitant des technologies et des stratégies spécifiques pour leur traitement et leur exploitation. Dans ce contexte, les capacités de stockage, de traitement et d'analyse des données deviennent essentielles pour extraire des informations utiles et prendre des décisions éclairées. Ainsi, la définition du Big Data évolue en fonction des avancées technologiques et des besoins des organisations dans un environnement numérique en constante évolution.

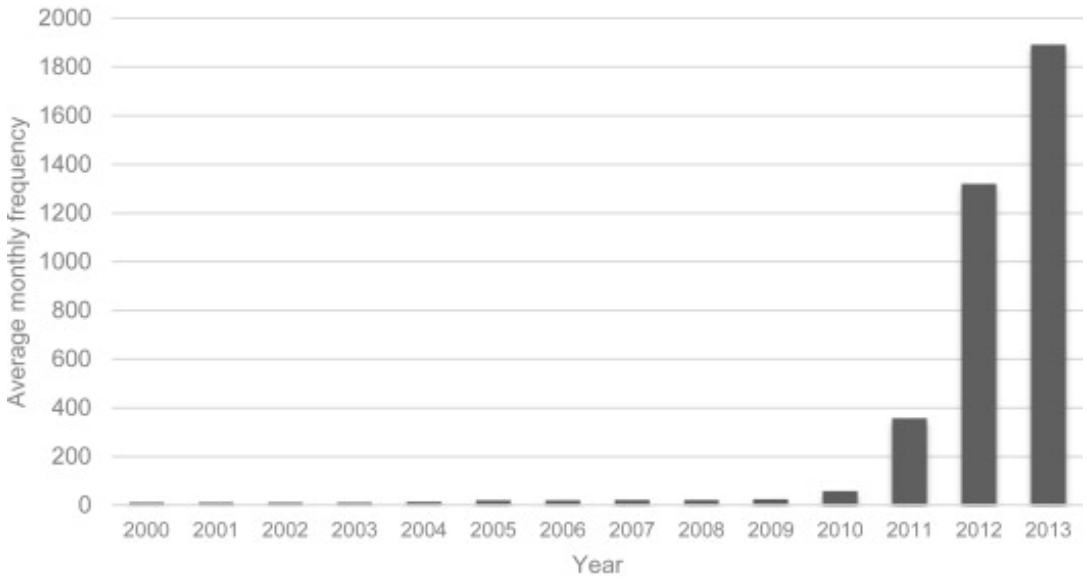

FIGURE 2.2.11 – Distribution de fréquence des documents contenant le terme « big data » dans la bibliothèque de recherche ProQuest

2.3 Les 5V du big data

Les 5 V du Big Data constituent un cadre conceptuel essentiel pour comprendre la complexité et la diversité des données massives. En 2001, Doug Laney a souligné l'importance de trois dimensions clés du Big Data : Volume, Variété et Vélocité. Par la suite, cette liste a été étendue pour inclure deux autres dimensions : Variabilité et Valeur. Ces 5 V sont devenus une référence dans le domaine du Big Data.

Tout d'abord, nous avons :

— Le **Volume** fait référence à la quantité massive de données générées à une échelle sans précédent. Cette dimension souligne l'explosion des données dans le monde numérique

moderne, où les entreprises et les organisations doivent gérer des pétaoctets (10^{15}) voire des exaoctets (10^{18}) de données provenant de multiples sources. Les définitions du volume de Big Data sont relatives et varient en fonction de facteurs tels que le temps et le type de données. Ce qui peut être considéré aujourd'hui comme du Big Data pourrait ne pas atteindre le seuil à l'avenir, car les capacités de stockage augmenteront, permettant ainsi de capturer des ensembles de données encore plus volumineux [42].

— La **Variété** met en avant la diversité des types de données, qu'elles soient structurées, semi-structurées ou non structurées comme le montre la figure 2.3.13. Cela inclut les textes, les images, les vidéos, les sons, les données géospatiales et bien d'autres. Cette variété croissante des données pose des défis pour leur traitement et leur analyse, nécessitant des outils et des techniques adaptés.

— La **Vélocité** se réfère à la vitesse à laquelle les données sont générées, collectées et traitées. Avec la montée en puissance des capteurs IoT, des réseaux sociaux et des transactions en ligne, les données sont générées à un rythme effréné, exigeant des systèmes et des infrastructures capables de traiter et d'analyser ces flux de données en temps réel.

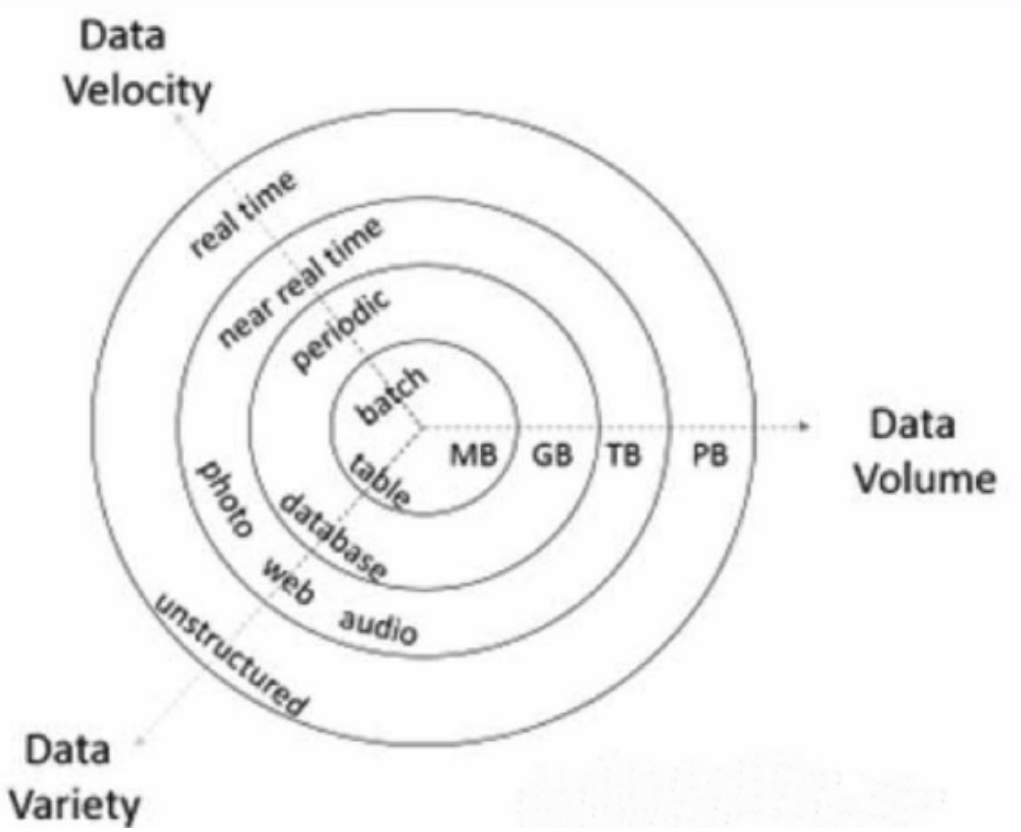

FIGURE 2.3.12 – les 3V du big data

— La **Véracité** met en évidence la nature changeante et imprévisible des données. Cela inclut

les fluctuations saisonnières, les tendances temporaires, les anomalies et les incertitudes qui peuvent affecter les données. Cette variabilité rend l'analyse des données plus complexe et nécessite des méthodes robustes pour en tirer des informations précieuses.

— La **Valeur** souligne l'importance et la pertinence des données pour les organisations. Les données doivent être transformées en informations exploitables qui peuvent conduire à des décisions stratégiques, des innovations et des avantages concurrentiels. Cependant, la valeur des données dépend de leur qualité, de leur intégrité et de leur capacité à répondre aux besoins spécifiques des utilisateurs.

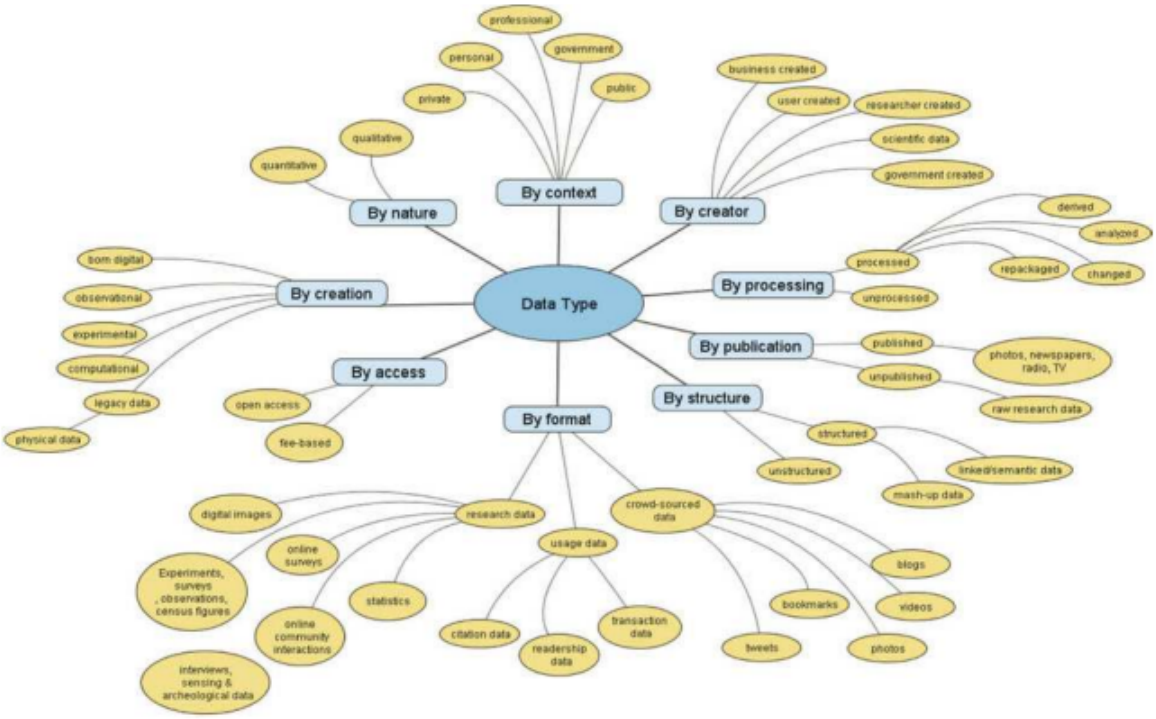

FIGURE 2.3.13 – Différents types de données

Dans l'ensemble, les 5V du Big Data offrent un cadre analytique puissant pour comprendre les défis et les opportunités associés à la gestion et à l'analyse des données massives dans un environnement numérique en constante évolution. Ainsi, la relativité des volumes de Big Data évoquée précédemment s'applique à toutes les dimensions. Il n'existe pas de références universelles pour le volume, la variété et la vélocité qui définissent le Big Data.[42].

2.4 Principales sources de données massives

Les principales sources de données massives englobent une diversité de flux d'information provenant de multiples origines et canaux. Parmi ces sources, on retrouve notamment :

— Les réseaux sociaux tels que Facebook, Twitter, et LinkedIn, où les utilisateurs génèrent une quantité considérable de données à travers leurs interactions, publications, et partages.

— Les appareils connectés à l'Internet des Objets (IoT) constituent une source majeure de données massives, avec des capteurs intégrés dans divers dispositifs tels que les smartphones, les voitures intelligentes, et les appareils domestiques intelligents.

— Les transactions en ligne effectuées sur les plateformes de commerce électronique telles que Amazon et Alibaba représentent également une source importante de données, fournissant des informations sur les préférences des consommateurs, les habitudes d'achat, et les tendances du marché.

— Les données générées par les systèmes d'information et les applications d'entreprise, tels que les bases de données transactionnelles et les logs de serveur, constituent une autre source significative de données massives.

— Les données géospatiales provenant de systèmes de positionnement global (GPS), de satellites, et de drones offrent des informations précieuses sur les déplacements, les territoires, et les ressources naturelles.

— Les données scientifiques et académiques provenant de domaines tels que la recherche médicale, la météorologie, et la génomique fournissent des ensembles de données massives utilisés pour des analyses et des découvertes scientifiques.

Dans l'ensemble, ces principales sources de données massives reflètent la diversité et la complexité des flux d'information dans le monde numérique moderne, et soulignent l'importance de leur gestion et leur analyse pour générer des informations précieuses et des avantages concurrentiels.

2.5 Les défis liés au traitement du Big Data

Le déluge de données en forte augmentation a l'ère du big data entraine d'énorme défis en matière de collecte, de stockage, de gestion, d'analyse et de visualisation des données [45]. Les principaux défis résident dans les 3V :

— **Volume** : Il représente le volet le plus saillant et assurément le plus remarquable du Big Data. Internet, les smartphones, les capteurs, ainsi que l'ouverture des données publiques, contribuent à une augmentation exponentielle de la quantité de données à gérer, nécessitant ainsi une réévaluation des architectures informatiques aptes à traiter cette abondance d'informations vue que les architectures traditionnelles ne sont plus en mesure de les gérer.

— **Variété** : La Variété évoque la diversité des sources de données primaires, telles que les puces RFID pour le suivi des colis, les caméras de surveillance, les réseaux sociaux, la musique, ou encore les capteurs de pollution. Ces différentes sources génèrent des données non structurées, rendant ainsi leur traitement complexe pour les bases de données traditionnelles qui exigent une structure de données fixe.

— **Vitesse** : La génération de données massives à un rythme sans précèdent constitue un défi pour le big data car nos technologies traditionnelles sont incapables d'en tirer de la valeur en temps réel. Le défi consiste à pouvoir traiter et analyser en temps réel le flux incessant de données au moment de leur arrivée dans votre système d'information. Car la valeur des données dépend de leur qualité, de leur intégrité et de leur pertinence, ce qui soulève des défis en termes de gestion de la qualité des données et de gouvernance de l'information [46].

Enfin, des préoccupations liées à la confidentialité, à la sécurité et à l'éthique des données sont soulevées, notamment en ce qui concerne la collecte, le stockage et l'utilisation des données personnelles. Ces défis nécessitent des solutions innovantes et des approches multidisciplinaires pour aborder efficacement et de manière responsable le traitement du Big Data.

2.6 Les technologies du Big data

Au début du nouveau millénaire, les entreprises étaient confrontées à des difficultés pour gérer le stockage de leurs données massives. Heureusement, des progrès significatifs dans les capacités de stockage et de traitement, à moindre coût, ont permis de résoudre ces problèmes. L'émergence de l'architecture extensible s'est révélée particulièrement cruciale, permettant le déploiement en parallèle de centaines voire de milliers de serveurs peu coûteux. Chacun de ces serveurs est équipé de multiples processeurs et de vastes caches mémoire partagées, favorisant ainsi la répartition des données et le traitement simultané. Cette approche facilite grandement le stockage et l'analyse efficace de volumes considérables de données. Si davantage de capacités de stockage ou de traitement sont requises, il est possible d'ajouter des serveurs supplémentaires

à cette architecture, qui repose sur le principe du traitement massivement parallèle (MPP). Les infrastructures dédiées aux données massives sont ainsi basées sur cette architecture extensible.

Les avancées technologiques telles que les disques à mémoire vive et les disques à semi-conducteurs représentent des progrès significatifs. Elles permettent d'améliorer les temps de réponse en stockant les données en mémoire plutôt que sur des disques durs. Le principal facteur limitant les performances, à savoir le temps nécessaire pour accéder aux données et les récupérer depuis les disques durs, est considérablement réduit grâce à ces nouvelles technologies.

Ainsi, la nécessité de stocker et d'analyser des mégadonnées a donné naissance à une variété de technologies, d'approches et de plateformes. Beaucoup d'entre elles sont complémentaires. Une attention particulière sera accordée à Hadoop/MapReduce, qui fera l'objet d'une étude dans la suite de ce chapitre, en raison de l'attention considérable qu'il reçoit et de son importance potentielle [47].

2.6.1 Moteurs de streaming et de traitement d'événements complexes (CEP)

Nous entrons dans l'ère de l'« Internet des objets », où des dispositifs tels que les voitures et les compteurs de services publics transmettent automatiquement des données en ligne. Pour les entreprises, il est précieux de recevoir ces données, de les traiter en temps réel et d'agir rapidement en conséquence. Les applications phares comprennent la négociation automatique d'actions, la détection de fraudes par carte de crédit, la gestion de la chaîne d'approvisionnement et la surveillance des équipements.

Les moteurs de streaming et de traitement d'événements complexes (CEP) comme Tibco StreamBase et BusinessEvents fournissent une intelligence continue en ingérant de grandes quantités de données en temps réel. Ils permettent d'accéder aux données historiques à partir de bases de données, d'appliances et d'entrepôts de données haute performance, d'effectuer des calculs et des corrélations, de détecter des modèles et des anomalies, d'appliquer des règles métier aux flux de données entrants, de fournir des informations aux utilisateurs et d'automatiser la prise de décision [47].

2.6.2 Base de données NoSQL

NoSQL, également connu sous le nom de « not only SQL » et « non-SQL », représente une approche dans la conception de bases de données permettant le stockage et l'interrogation de données en dehors des schémas traditionnels des bases de données relationnelles. Alors qu'elle peut encore stocker des données similaires à celles présentes dans les systèmes de gestion de

bases de données relationnelles (SGBDR), elle le fait simplement d'une manière différente de celle d'un SGBDR.

2.6.3 Architecture Big data : cas de Hadoop

Dans cette section nous nous focalisons sur l'architecture de Hadoop car elle reste la plus utilisée dans le marché. Hadoop est un framework open-source qui permet le traitement distribué de grands ensembles de données sur des clusters de serveurs. Hadoop a été développé par Doug Cutting et Mike Cafarella. Il a été conçu pour résoudre les problèmes de stockage et de traitement de données, le plus souvent non structurées, à grande échelle, en utilisant des ressources matérielles peu coûteuses et en fournissant une haute disponibilité et une tolérance aux pannes. Ainsi, plusieurs fournisseurs ont construit des distributions prêtes à l'emploi pour gérer le Big Data, à savoir HortonWorks, MapR, Cloudera, IBM Infosphere BigInsights, Pivotal HD, Microsoft HD Insight [48].

2.6.3.1 Architecture de cluster Hadoop

Dans le but de stocker et d'analyser d'énormes quantités de données non structurées dans un environnement informatique distribué, Hadoop utilise le concept de cluster qui est un ensemble de machines connectées qui fonctionnent ensemble comme un système unique. Dans cette architecture deux termes principaux apparaissent :

— **Cluster** : Qui est une collection de nœuds

— **Nœud** : Qui est un point d'intersection / connexion au sein d'un réseau c'est-à-dire un serveur.

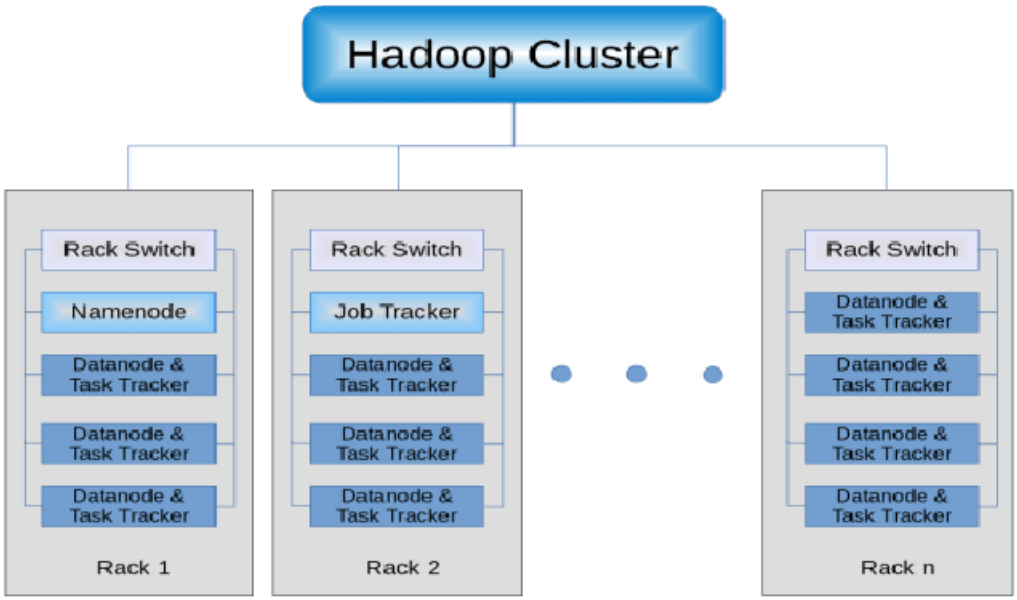

FIGURE 2.6.14 – Architecture hadoop

L'architecture de Hadoop se compose de 4 modules principaux [49] :

— **Hadoop Distributed File System (HDFS)**

— **Hadoop MapReduce**

— **Hadoop YARN**

— **Hadoop Common**

2.6.3.2 Hadoop Distributed File System (HDFS)

HDFS est le système de fichiers distribué de Hadoop, conçu pour stocker de grandes quantités de données de manière fiable et scalable. Il est hautement tolérant aux pannes et utilise du matériel à coût minimal et il se compose d'un cluster de machines et les fichiers sont stockés sur elles [49]. HDFS divise les fichiers en blocs et les réplique sur plusieurs nœuds du cluster pour assurer la redondance et la disponibilité des données. L'architecture de HDFS comprend les éléments suivants :

— Maître : HDFS NameNode, YARN ResourceManager.

— Esclaves : HDFS DataNodes, YARN NodeManagers.

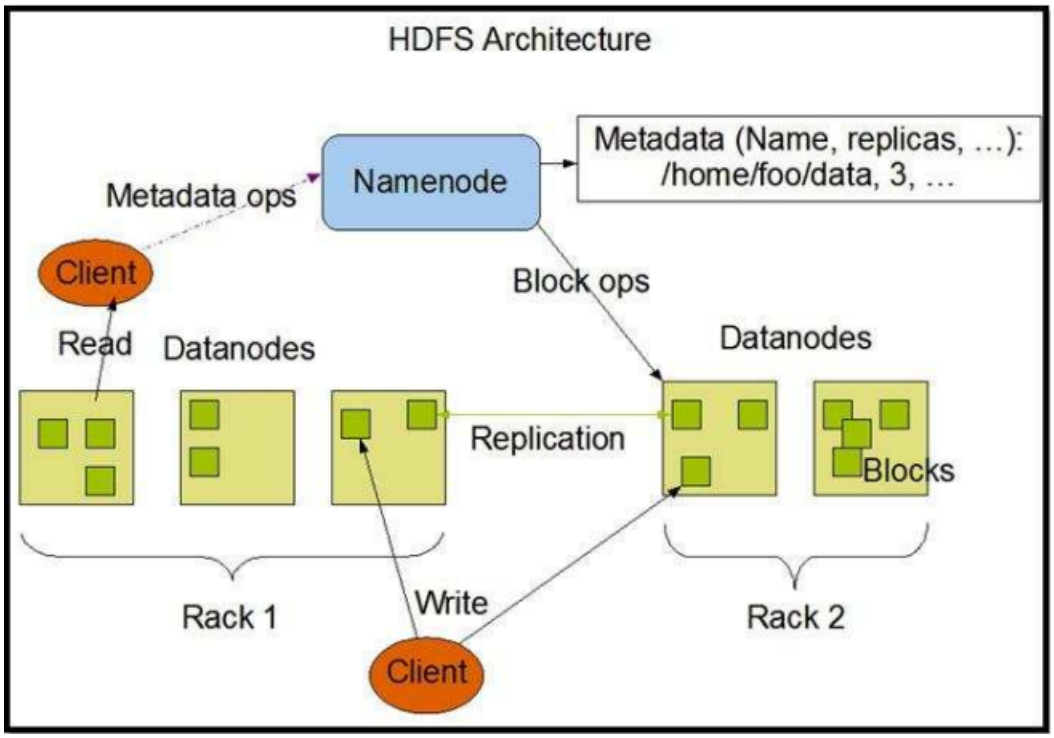

FIGURE 2.6.15 – HDFS Architecture

1. **HDFS NameNode**

Le NameNode aussi appelé Maitre, il s'agit du composant central de HDFS. Le NameNode gère le namespace et les méta-données du système de fichiers. Il garde une trace de l'emplacement des blocs de données sur les DataNodes. Les rôles du namenode sont les suivants :

— Mappage des blocs sur leurs nœuds de données [49].

— Gestion de l'espace de noms du système de fichiers [49].

— Exécuter des opérations sur le système de fichiers : ouverture, fermeture et renommer les fichiers [49].

2. **HDFS DataNodes**

Le HDFS est constitué de plusieurs nœuds de données appelés aussi datanodes. Ces nœuds de données stockent les blocs de fichiers qui sont gérés par le nœud maître. Les nœuds de données sont responsables de l'exécution des opérations de lecture et d'écriture sur le système de fichiers en réponse aux requêtes des clients. Ils gèrent également la création et la réplication des blocs de données.

Un bloc est la plus petite unité de données que le système peut lire ou écrire. Par défaut, la taille d'un bloc dans HDFS est de 128 Mo, mais cette valeur n'est pas fixe et peut être modifiée pour répondre aux besoins spécifiques d'une application ou d'un système.[49].

2.6.3.3 YARN

YARN est le framework de gestion des ressources de Hadoop. L'idée fondamentale de YARN est de diviser les fonctionnalités de gestion des ressources et de planification/surveillance des tâches en démons distincts dans un cluster Hadoop. YARN permet l'exécution de divers frameworks de traitement de données, tels que MapReduce, Spark, Hive, et bien d'autres, sur la même infrastructure. Il est composé de :

1. **ResourceManager**

   Le ResourceManager est le composant principal de YARN. Il est localisé dans le NameNode et agit en tant que planificateur centralisé qui alloue les ressources du cluster aux applications en fonction de leurs besoins. Voici les rôles clés du ResourceManager :

   — **Allocation des ressources** : Le ResourceManager gère les ressources disponibles dans le cluster, telles que la capacité de stockage, la mémoire et la puissance de calcul. Il alloue ces ressources aux applications en fonction de leurs demandes.

   — **Planification des tâches** : Le ResourceManager attribue les ressources aux différentes applications ou tâches en fonction de politiques de planification spécifiées. Il assure également une utilisation efficace des ressources en tenant compte des priorités et des contraintes de capacité.

   — **Suivi des nœuds de travail** : Le ResourceManager surveille l'état des nœuds de travail (NodeManagers) dans le cluster. Il vérifie leur disponibilité, leur charge et signale toute défaillance. Cela permet une détection rapide des pannes et une redistribution des tâches si nécessaire.

   — **Gestion des files d'attente** : Le ResourceManager prend en charge la création et la gestion de files d'attente pour les différentes applications. Cela permet de prioriser les tâches et de contrôler l'accès aux ressources en fonction des politiques définies.

2. **NodeManager**

   Chaque nœud de travail (NodeManager) dans le cluster Hadoop est responsable de l'exécution des tâches et de la gestion des ressources locales. Voici les principales responsabilités du NodeManager :

— **Allocation des ressources locales** : Le NodeManager alloue les ressources locales, telles que la mémoire et la puissance de calcul, aux différentes tâches exécutées sur le nœud.

— **Suivi des tâches** : Le NodeManager surveille l'exécution des tâches sur le nœud, signale leur progression au ResourceManager et gère les ressources allouées.

— **Gestion des conteneurs** : Le NodeManager crée et gère des conteneurs pour exécuter les tâches. Un conteneur est une unité d'isolation qui encapsule les ressources allouées à une application ou à une tâche spécifique

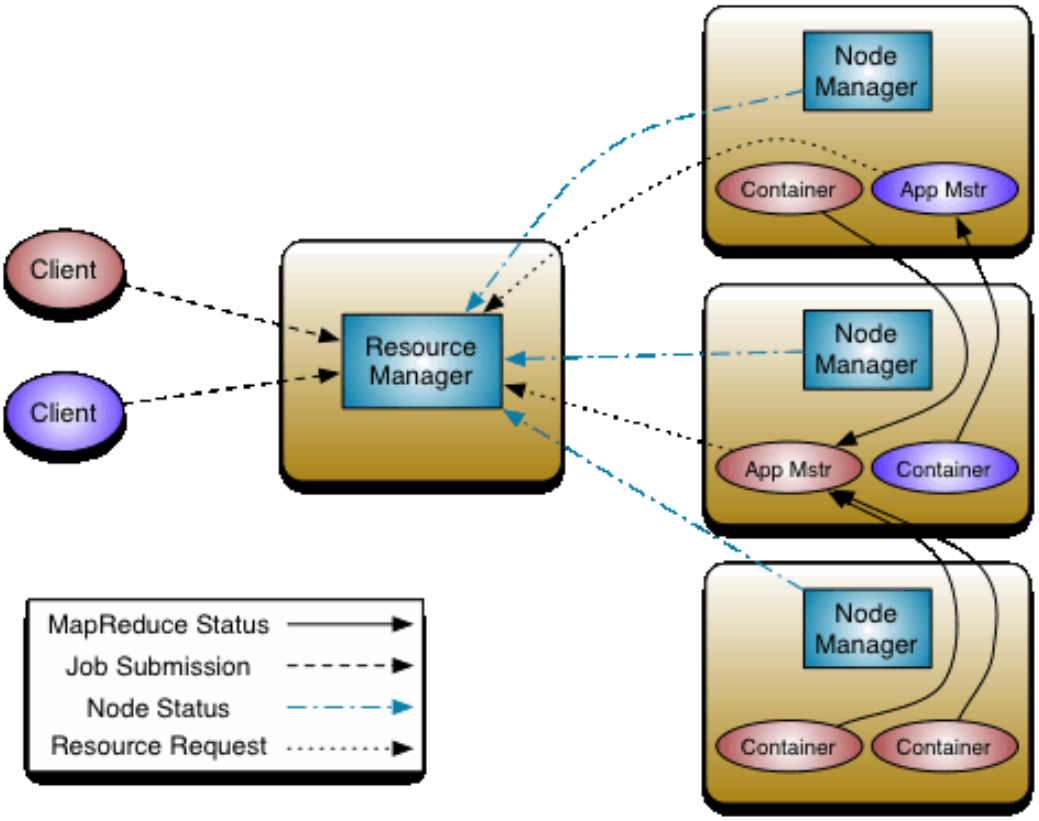

FIGURE 2.6.16 – hadoop yarn

2.6.3.4 Hadoop MapReduce

MapReduce est un modèle de programmation et d'exécution utilisé pour le traitement parallèle de grands ensembles de données. Il est basé sur le principe de diviser et conquérir. C'est un modèle de programmation distribuée basé sur plusieurs langages de programmation comme le Java, le python et la scala. Ce paradigme divise le travail en deux étapes principales : **Map** et **Reduce** [49]. L'architecture de Hadoop MapReduce comprend les éléments suivants :

— **JobTracker** : Il est responsable de la gestion des tâches MapReduce dans le cluster. Il répartit les tâches sur les nœuds de données disponibles et assure le suivi de l'exécution.

— **TaskTracker** : Chaque nœud de données dispose d'un TaskTracker qui exécute les tâches MapReduce. Il communique avec le JobTracker pour obtenir de nouvelles tâches et signaler l'état d'avancement.

— **MapReduce Jobs** : Les jobs MapReduce sont les programmes qui utilisent le modèle MapReduce pour effectuer des opérations de traitement distribué. Chaque job est divisé en tâches map et reduce, qui sont ensuite réparties sur les nœuds de données.

— **Map et Reduce** : Les fonctions Map et Reduce sont les blocs de construction fondamentaux de MapReduce. La fonction Map prend en entrée des paires clé/valeur et génère des paires intermédiaires clé/valeur. La fonction Reduce fusionne les paires intermédiaires associées à une même clé et produit des résultats finaux.

En somme l'architecture de hadoop cluster s'illustre très bien avec l'image suivante :

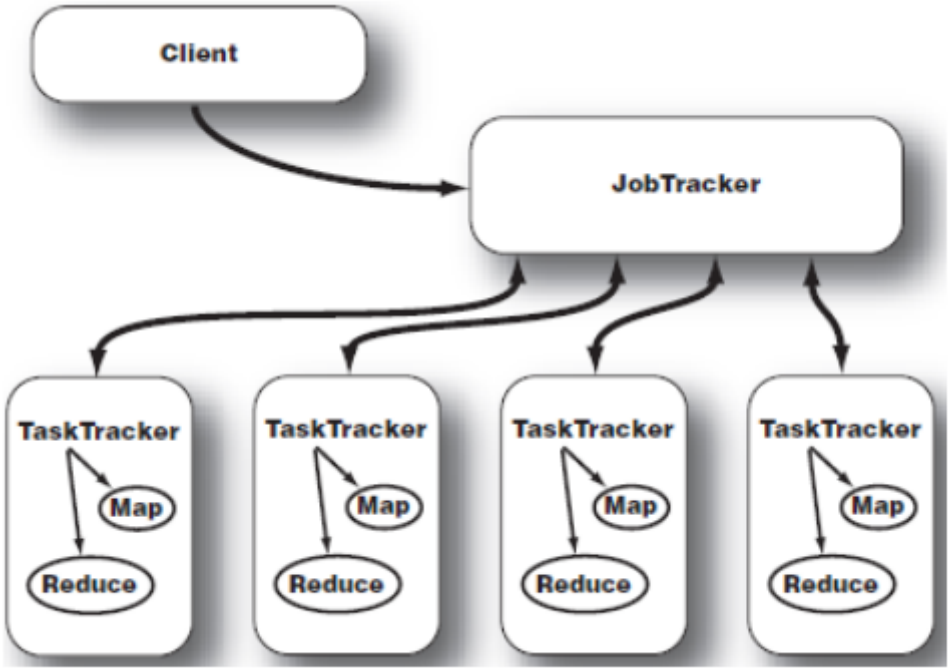

FIGURE 2.6.17 – job Tracker and Task Tracker Fonctionnement

2.6.3.5 Fonctionnalités et opérations sur HDFS

Le Hadoop Distributed File System (HDFS) offre un ensemble de fonctionnalités et d'opérations essentielles pour le stockage et la gestion des données distribuées. Voici un tableau récapitulatif de ces fonctionnalités et opérations :

Fonctionnalité/Opération	Description
Stockage de fichiers distribués	HDFS stocke les fichiers en les divisant en blocs de taille fixe et en les répartissant sur plusieurs nœuds dans le cluster pour assurer la redondance et la tolérance aux pannes.
Répartition et réplication	Les blocs de fichiers sont répartis sur plusieurs nœuds, avec une réplication configurable (par défaut, trois copies) pour garantir la disponibilité et la résilience.
Tolérance aux pannes	HDFS détecte et récupère automatiquement des pannes de nœuds et de disques. Les blocs de données manquants sont automatiquement recréés à partir des répliques.
Scalabilité	HDFS peut évoluer horizontalement en ajoutant simplement de nouveaux nœuds au cluster, augmentant ainsi la capacité de stockage et la puissance de traitement.
Accès aux données	Fournit une API Java et des interfaces de commande (comme 'hadoop fs') pour interagir avec le système de fichiers, permettant la lecture, l'écriture et la gestion des fichiers.
Lecture optimisée	Les données sont lues en parallèle à partir de plusieurs nœuds, améliorant ainsi les performances de lecture pour les grandes quantités de données.
Sécurité	HDFS prend en charge l'authentification via Kerberos, les listes de contrôle d'accès (ACL) et le chiffrement des données au repos et en transit.
Quota de stockage	Les quotas peuvent être définis pour limiter l'espace de stockage utilisé par les utilisateurs ou les groupes, assurant une gestion efficace des ressources.
Snapshots	Permet de capturer l'état du système de fichiers à un moment donné, facilitant ainsi la sauvegarde et la récupération des données.
Contrôle des versions	Les fonctionnalités de snapshots et de gestion des versions permettent de restaurer les données à un état précédent en cas de suppression ou de corruption.

Fonctionnalité/Opération	Description
Équilibrage des données	L'outil 'balancer' permet de rééquilibrer la distribution des blocs de données entre les nœuds pour une utilisation optimale des ressources du cluster.
Garbage collection	HDFS gère automatiquement la suppression des blocs orphelins et non référencés pour libérer de l'espace de stockage.
Support des fichiers de grande taille	Conçu pour stocker et traiter des fichiers de plusieurs gigaoctets ou téraoctets, permettant des analyses de données à grande échelle.
Accès par ligne de commande	L'outil 'hdfs dfs' fournit des commandes pour manipuler les fichiers et répertoires dans HDFS (par exemple, 'put', 'get', 'ls', 'rm').
Intégration avec Hadoop	Étroitement intégré avec les autres composants de l'écosystème Hadoop, comme YARN, MapReduce, et Hive, pour une gestion et un traitement efficaces des données.

TABLE 2.6.1 – Fonctionnalités et opérations sur HDFS

2.6.3.6 Les distributions Hadoop

Le défi le plus important du big data est celui qui représente la croissante sans fin et de façon abusive des données. Compte tenu de ce défi, plusieurs distributions émergent dans le domaine du traitement de données massives. Ainsi nous allons voir les plus connues et les plus utilisées.

— **Distribution HortonWorks** : En 2011, l'équipe de Yahoo a lancé Hortonworks, une distribution entièrement open source de Hadoop sous licence Apache. L'objectif principal de cette distribution est de simplifier l'adoption de la plateforme Apache Hadoop. Hortonworks est un contributeur majeur au projet Hadoop, et les fournisseurs Hadoop adoptent souvent un modèle commercial où ils vendent des services de support technique et de formation en plus des licences. La plateforme Hadoop d'Apache et la solution HortonWorks sont compatibles entre elles [48].

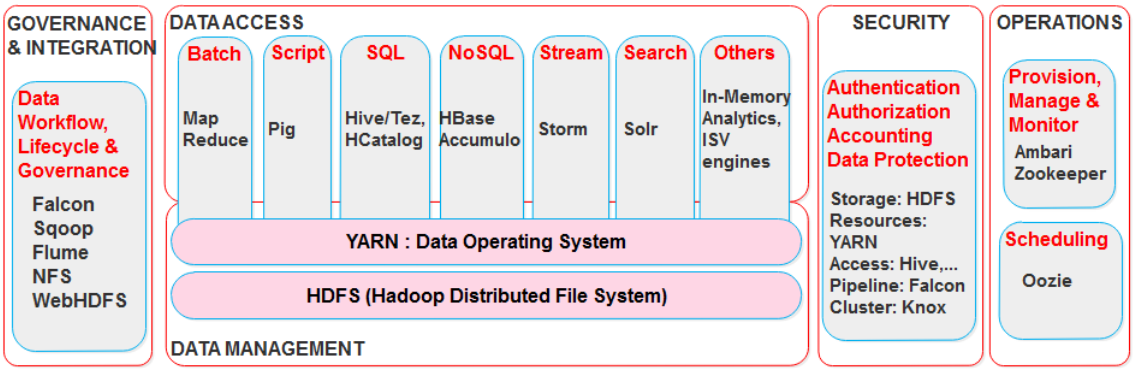

FIGURE 2.6.18 – HortonWorks Hadoop Platform (HDP) [50]

— **Distribution Cloudera** : Les experts Hadoop de Facebook, Google, Oracle et Yahoo réussissent à trouver Cloudera. Cette distribution inclut les composants d'Apache Hadoop et réussit à développer efficacement des composants maison pour la gestion des clusters. L'objectif du modèle économique de Cloudera n'est pas seulement de vendre des licences aux clients, mais également de leur vendre des services de formation et d'assistance. Cloudera fournit une version entièrement open source de sa plateforme (licence Apache 2.0) [48].

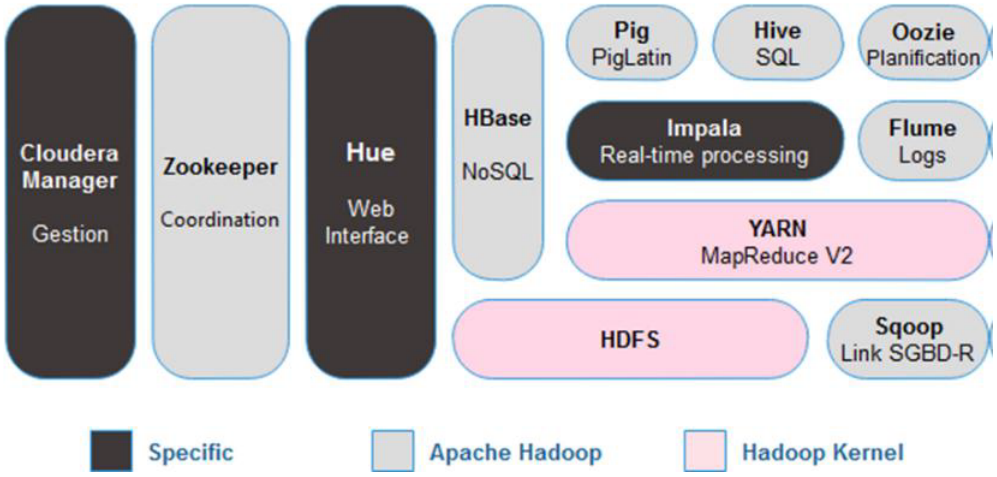

FIGURE 2.6.19 – Cloudera Distribution of Hadoop Platform (CDH) [15]

— **MapR distribution** : En 2009, d'anciens membres de Google développent la distribution MapR. Ils contribuent principalement aux projets Apache Hadoop comme HBase, Hive, Pig, Zookeeper et surtout Drill [50]. Ils proposent avec succès leur propre version de MapReduce et système de fichiers distribués : MapR FS et MapR MR [48]. MR [50]. Ainsi, trois versions de leur solution sont disponibles mais la version M3 est Open source.

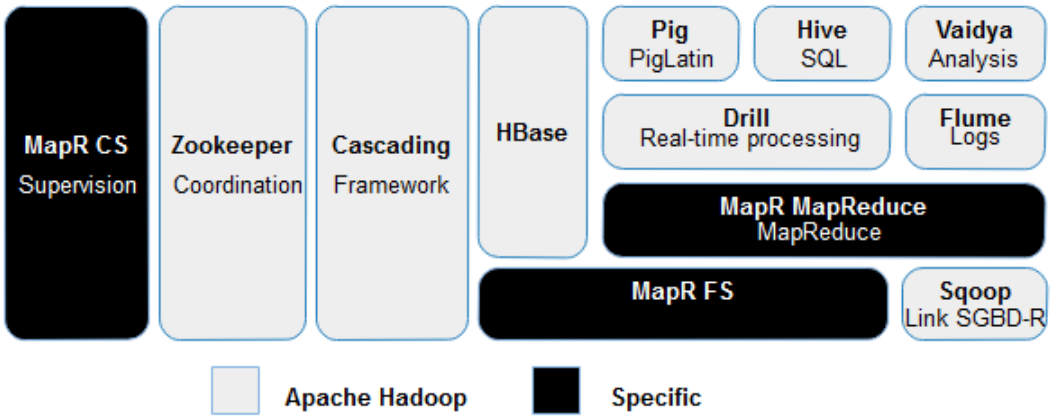

FIGURE 2.6.20 – MapR (M3) [51]

— **Distribution IBM InfoSphere BigInsights** : En 2011, les professionnels d'IBM développent InfoSphere BigInsights pour Hadoop en deux versions : l'Entreprise Edition et la version de base, qui était un téléchargement gratuit d'Apache Hadoop, fourni avec une console de gestion Web. En juin 2013, IBM a lancé Infosphere BigInsights Quick Start Edition. Cette nouvelle édition propose des capacités massives d'analyse de données sur une plate-forme centrée sur l'entreprise [52]. Il combine à la fois la solution Open Source d'Apache Hadoop et les performances de l'entreprise et laisse ainsi place à une analyse à grande échelle, marquée par les défauts de tolérance et résilience. En bref, cette distribution prend en charge les données structurées, non structurées et semi-structurées et offre une flexibilité maximale [48].

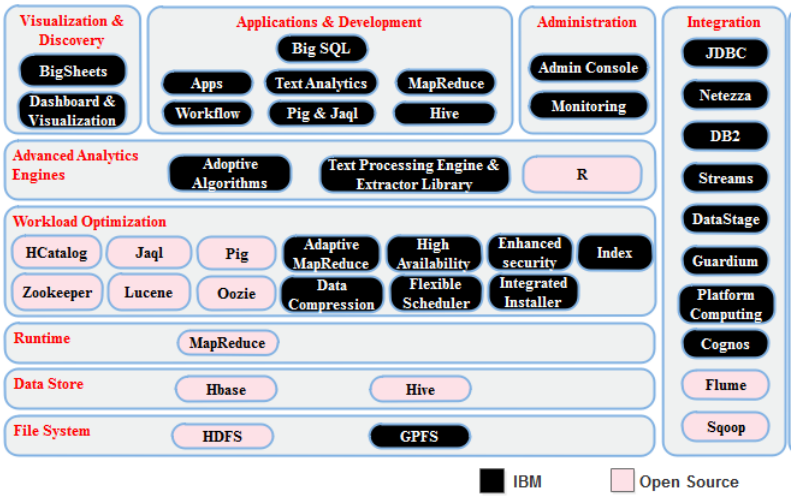

FIGURE 2.6.21 – IBM InfoSphere BigInsights Enterprise Edition [53]

— **Distribution HD pivot** : Pivotal Software Inc est une société de logiciels dont le siège est à San Francisco, en Californie. Ses bureaux principaux impliquent Pivotal Lab, le groupe de développement Pivotal Cloud Foundry et un groupe de développement de produits pour

le marché du Big Data. Il a fallu attendre 2013 pour que les développeurs mettent en place la distribution Apache Hadoop appelée Pivotal HD. En bref, Pivotal HD Enterprise est une distribution commerciale d'Apache Hadoop [51]. La figure ci-dessous montre comment chaque composant Apache et Pivotal s'intègre dans l'architecture globale de Pivotal HD Enterprise [48].

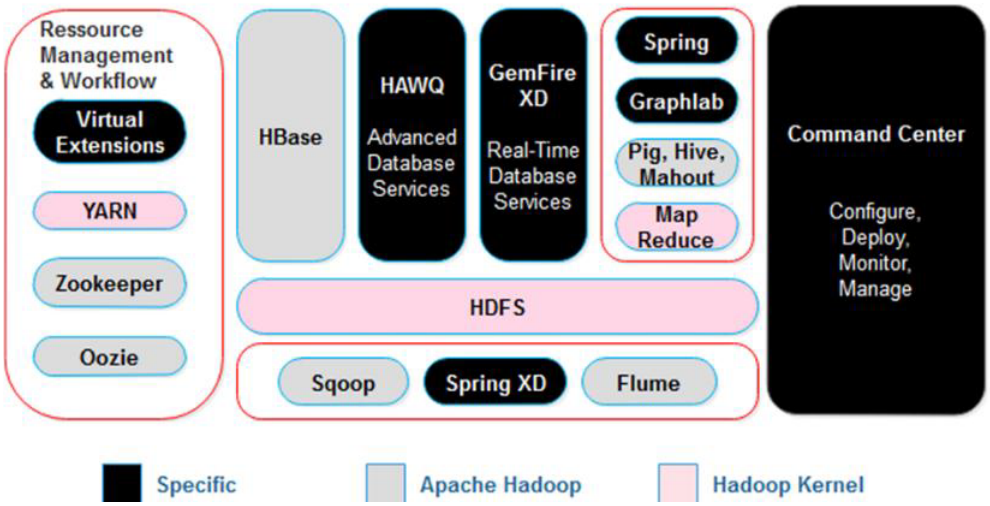

FIGURE 2.6.22 – Pivotal HD Entreprise

Cette tendance à la collecte et au traitement du Big Data a donné naissance à de nouvelles distributions conçues pour gérer un système Big Data. L'objectif de ces distributions est d'ouvrir la voie à l'adoption de la plateforme Hadoop d'Apache et de gérer les clusters principalement Cloudera, HortonWorks, MapR, IBM Infosphere BigInsights, Microsoft HD Insight, Pivotal HD [48].

2.6.3.7 Avantages d'un cluster Hadoop

Voici quelques avantages du cluster hadoop :

— **Scalabilité** : Hadoop permet d'ajouter facilement de nouveaux nœuds au cluster, ce qui augmente sa capacité de traitement et de stockage sans nécessiter de modifications majeures à l'architecture existante. Il permet également de gérer des pétaoctets de données, ce qui le rend idéal pour les entreprises qui traitent de grandes quantités de données.

— **Tolérance aux Pannes** : Les données dans HDFS sont répliquées sur plusieurs nœuds (par défaut trois copies), ce qui garantit que les données restent accessibles même en cas de défaillance d'un ou plusieurs nœuds. Hadoop détecte automatiquement les défaillances des nœuds et redirige les tâches vers d'autres nœuds opérationnels, assurant ainsi la continuité des opérations.

— **Coût-Efficacité** : Hadoop est conçu pour fonctionner sur du matériel standard, ce qui réduit les coûts d'infrastructure comparé aux solutions de stockage et de traitement de données traditionnelles. Il est open source, ce qui signifie qu'il n'y a pas de frais de licence, et les entreprises peuvent personnaliser le logiciel en fonction de leurs besoins spécifiques.

— **Flexibilité et Adaptabilité** : Hadoop peut traiter et stocker des données structurées, semi-structurées et non structurées, ce qui le rend adapté à une variété d'applications, y compris l'analyse de données, le traitement de logs, et les systèmes de recommandation. En outre, Il s'intègre bien avec un large éventail d'outils et de frameworks (comme Apache Hive, Apache Pig, Apache HBase, Apache Spark, etc.), offrant une flexibilité pour répondre à divers besoins analytiques.

— **Haute Performance** : Hadoop divise les tâches en sous-tâches et les exécute en parallèle sur différents nœuds, ce qui accélère le traitement des grandes quantités de données. Il Hadoop minimise les déplacements de données en traitant les données là où elles sont stockées, réduisant ainsi la latence et améliorant les performances.

— **Sécurité** : Hadoop offre des listes de contrôle d'accès pour gérer les permissions sur les fichiers et les répertoires dans HDFS. Mais également, il supporte Kerberos pour l'authentification sécurisée des utilisateurs et des services.

Ces avantages font de Hadoop une solution puissante et flexible pour le traitement et le stockage de grandes quantités de données dans divers secteurs et applications. Cependant, il présente des limites.

2.6.3.8 Limites d'un cluster Hadoop

Un cluster hadoop comporte également des limites qui sont :

— **Problème avec les petits fichiers** : Le principal problème rencontré avec HDFS concerne les petits fichiers. Un petit fichier est considérablement plus petit que la taille du bloc HDFS, qui est généralement de 128 Mo par défaut. Lorsque nous stockons un grand nombre de petits fichiers, HDFS n'est pas en mesure de les gérer efficacement, car il est optimisé pour traiter un petit nombre de fichiers volumineux qui contiennent de grandes quantités de données. En cas de présence excessive de petits fichiers, le NameNode, qui stocke l'espace de noms de HDFS, sera surchargé.

— **Vitesse de traitement lente** : Dans le cadre de Hadoop, l'algorithme parallèle et distribué appelé MapReduce est utilisé pour traiter de vastes ensembles de données. Cependant,

l'utilisation de MapReduce implique l'exécution de deux tâches essentielles : la fonction de mappage (mapping) et la fonction de réduction (reducing), ce qui peut entraîner une augmentation de la latence et un temps de traitement plus long.

— **Pas de traitement de données en temps réel** : Il est important de noter que Hadoop n'est pas adapté au traitement de données en temps réel. En raison de sa nature basée sur le traitement par lots, il n'est pas conçu pour fournir des résultats instantanés. Si vous avez besoin de résultats en temps réel ou d'un traitement rapide des données, d'autres technologies plus adaptées, telles que les systèmes de traitement de flux (Stream processing), peuvent être plus appropriées.

— **Pas facile à utiliser** : Dans Hadoop, les développeurs utilisant MapReduce doivent fournir du code spécifique pour chaque opération qu'ils souhaitent effectuer, ce qui peut rendre leur travail complexe. De plus, MapReduce ne dispose pas d'un mode interactif natif, ce qui peut rendre le processus de développement et de test plus difficile.

— **Vulnérable par nature** : Hadoop est effectivement principalement écrit en Java, un langage de programmation largement utilisé dans le développement de nombreux systèmes et applications. En raison de sa popularité, Java a été ciblé par des cybercriminels, ce qui a conduit à la découverte de certaines failles de sécurité dans les applications Java, y compris celles basées sur Hadoop.

2.7 CONCLUSION

Nous avons souligné l'énorme potentiel des données massives pour générer des connaissances, prendre des décisions éclairées et stimuler l'innovation dans divers domaines. En examinant les différentes sources de données massives telles que les médias sociaux, les capteurs IoT et les données transactionnelles, nous avons constaté la diversité et la richesse des informations disponibles pour l'analyse. L'analyse des données est essentielle pour générer des insights, optimiser les processus et prendre des décisions éclairées. Les algorithmes de machine learning (ML) sont déjà performants pour traiter des volumes de données modérés, accomplissant des tâches comme la classification, la régression et le clustering. Cependant, l'essor du Big Data révèle les limites des infrastructures traditionnelles. Les architectures Big Data, comme Apache Hadoop et Apache Spark, sont cruciales pour gérer les volumes, vitesses et variétés de données modernes. Les défis incluent la gestion des grands volumes de données en temps réel, le prétraitement des données hétérogènes et l'adaptation des algorithmes pour un traitement dis-

tribué. Bien que cette transition pose des défis, elle offre des opportunités considérables pour transformer des données massives en connaissances exploitables et avantages concurrentiels. Exploiter pleinement le potentiel des analyses à grande échelle nécessite donc l'intégration des algorithmes de ML dans des architectures Big Data. Ainsi, dans le chapitre suivant, nous entreprendrons une étude comparative entre les algorithmes de machine learning appliqués sur une seule machine et ceux déployés sous Hadoop, en adaptant ces algorithmes traditionnels pour un traitement distribué.

COMPARAISON DES ALGORITHMES DE MACHINE LEARNING ENTRE LES ARCHITECTURES CLASSIQUE ET LES ARCHITECTURES BIG DATA

3.1 INRODUCTION

Les techniques d'apprentissage automatique (ML) ont eu un impact considérable sur de nombreuses applications, telles que la vision par ordinateur, le traitement de la parole, la compréhension du langage naturel, les neurosciences, la santé et l'Internet des objets. L'ère du Big Data a suscité un immense intérêt pour l'apprentissage automatique. Les algorithmes d'apprentissage automatique n'ont jamais été aussi prometteurs, tout en étant mis au défi par le Big Data pour extraire de nouvelles informations sur diverses applications commerciales et comportements humains. Le Big Data offre une richesse d'informations inédite pour que les algorithmes d'apprentissage automatique puissent identifier des modèles et construire des modèles prédictifs. Cependant, les algorithmes d'apprentissage automatique traditionnels doivent surmonter des défis critiques, tels que l'évolutivité, pour exploiter pleinement le potentiel du Big Data. Avec l'expansion continue du Big Data, le ML doit évoluer pour transformer ces données en intelligence exploitable.

Cependant, à l'heure du Big Data, la collecte d'ensembles de données est si vaste et si complexe qu'elle est difficile à gérer à l'aide de méthodes d'apprentissage traditionnelles, car le processus établit l'apprentissage à partir d'ensembles de données conventionnels n'a pas été conçu pour fonctionner correctement et ne fonctionnera pas correctement avec de gros volumes de données. Par exemple, la plupart des algorithmes d'apprentissage automatique traditionnels

sont conçus pour des données qui seraient entièrement chargées en mémoire [54], ce qui n'est plus valable dans le contexte du big data. Par conséquent, même si l'apprentissage de ces nombreuses données devrait apporter des avancées scientifiques et techniques significatives ainsi que des améliorations de la qualité de notre vie, cela pose en même temps d'énormes défis.

Dans ce contexte, ce chapitre explore en profondeur les défis liés à l'application des algorithmes de machine learning sur le Big Data en apportant des solutions. Ensuite, une étude comparative des performances de deux algorithmes (arbre de décision ID3 et K-plus proches voisins) dans un environnement traditionnel et dans un environnement Big Data (Apache Hadoop) est réalisée. Enfin, nous analyserons les résultats obtenus et les comparerons aux travaux déjà faits.

3.2 Les defis liés à l'application des algorithmes du machine learning sur le big data

Malgré les avancées récentes en apprentissage automatique évoquées dans le chapitre 1, l'émergence du Big Data continue de présenter de nombreux défis significatifs qui doivent encore être relevés. Les cinq aspects principaux du big data incluent le volume, la variété, la vélocité, la véracité et la valeur. Chacune de ces caractéristiques pose des défis distincts pour les techniques d'apprentissage automatique. Dans cette section, nous abordons les principaux défis des techniques d'apprentissage automatique appliquées au big data, examinés sous cinq angles distincts, comme illustré à la figure 3.2.23. Ces perspectives incluent : l'apprentissage sur des données à grande échelle, l'apprentissage sur divers types de données, l'apprentissage sur des données en streaming à haute vitesse, l'apprentissage sur des données incertaines et incomplètes, ainsi que l'apprentissage visant à extraire des informations précieuses à partir de volumes massifs de données. Nous parlerons des solutions potentielles pour surmonter les obstacles identifiés dans les recherches récentes.

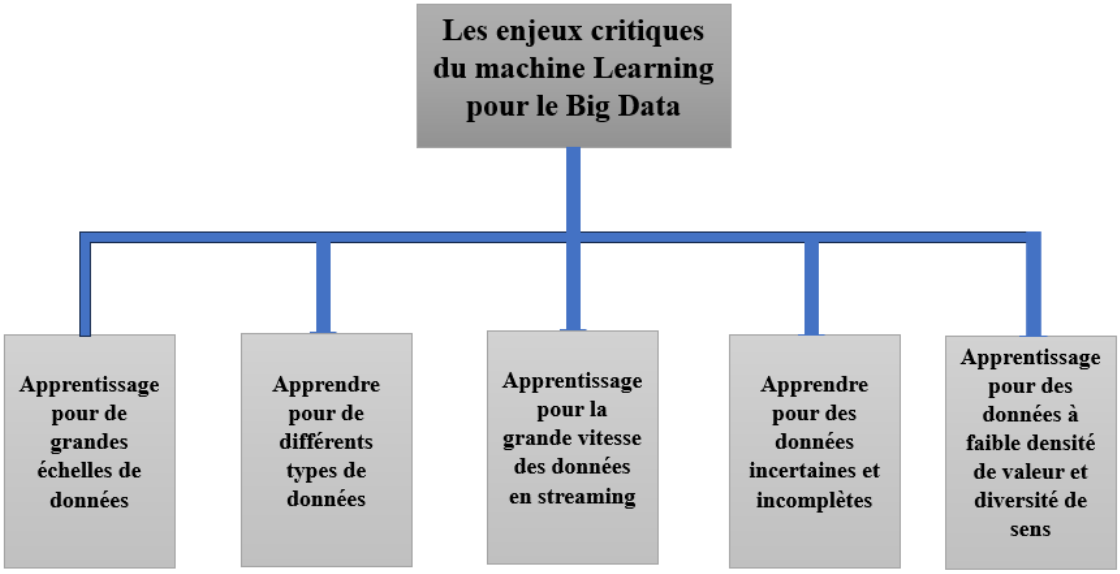

FIGURE 3.2.23 – Les enjeux critiques du machine learning pour le big data

3.2.1 Apprendre pour des données à grande échelle

— **Problème critique**

Il est clair que le volume de données constitue l'attribut principal du Big Data, posant un défi significatif pour l'apprentissage automatique. En prenant uniquement les données numériques comme exemple, chaque jour, Google doit traiter à lui seul environ 24 pétaoctets (pétaoctet = $2^{10} \times 2^{10} \times 2^{10} \times 2^{10} \times 2^{10}$ octets) de données [55]. En outre, en intégrant davantage de sources de données, l'échelle des données augmentera considérablement. Dans le cadre des tendances de développement actuelles, les données stockées et analysées par les grandes organisations atteindront sans aucun doute bientôt une magnitude allant du pétaoctet à l'exaoctet (exa octet = 2^{10} pétaoctets) [56].

— **Solutions possibles**

Il est indéniable que nous sommes désormais submergés par une quantité croissante de données, trop volumineuses pour entraîner un algorithme d'apprentissage automatique avec un processeur central et un stockage conventionnels. Ainsi, les frameworks distribués avec calcul parallèle sont préférés. La méthode des multiplicateurs à direction alternée (Alternating direction method of multipliers ADMM) [57, 58], en tant que cadre informatique prometteur pour développer des algorithmes d'optimisation convexes en ligne, distribués et évolutifs, est particulièrement adaptée pour le traitement de données parallèles et distribuées à grande échelle. Les principaux atouts de l'ADMM résident dans sa capacité à

diviser ou découpler plusieurs variables dans les problèmes d'optimisation, facilitant ainsi la résolution d'un problème global à grande échelle en coordonnant les solutions de sous-problèmes plus petits. En général, l'ADMM converge pour les optimisations convexes, mais il manque de garanties de convergence et de performance théoriques pour les optimisations non convexes. Cependant, de vastes preuves expérimentales dans la littérature soutiennent la convergence empirique et les bonnes performances de l'ADMM [59]. Une grande variété d'applications d'ADMM aux problèmes d'apprentissage automatique pour des ensembles de données à grande échelle ont été discutées dans [59].

Outre le cadre théorique distribué pour l'apprentissage automatique, visant à atténuer les défis posés par les volumes de données élevés, certaines méthodes pratiques de programmation parallèle sont également proposées et appliquées aux algorithmes d'apprentissage pour traiter des ensembles de données à grande échelle. MapReduce [60, 61], un cadre de programmation puissant, permet la mise en parallèle et la distribution automatique des applications de calcul sur de grands clusters de machines ordinaires. De plus, MapReduce offre une grande tolérance aux pannes, ce qui est essentiel pour gérer de vastes ensembles de données. Le principe central de MapReduce consiste à diviser d'abord les données massives en petits morceaux, puis à traiter ces morceaux de manière parallèle et distribuée pour générer des résultats intermédiaires. En agrégeant tous les résultats intermédiaires, on obtient le résultat final. C'est cette methode que nous allons utiliser dans la suite de ce chapitre. Un moyen général de programmation d'algorithmes d'apprentissage automatique sur multicœur avec l'avantage de MapReduce a été étudié dans 2.6.3.4. L'apprentissage assisté par le cloud computing représente une autre avancée remarquable, permettant aux systèmes de données de relever le défi posé par le volume du Big Data. Le cloud computing [62, 63] a déjà montré une remarquable élasticité, ouvrant la voie à la réalisation de l'évolutivité nécessaire pour les algorithmes d'apprentissage automatique. Il peut renforcer les capacités de calcul et de stockage grâce à l'infrastructure cloud. Dans ce contexte, GraphLab distribué , un cadre pour l'apprentissage automatique dans le cloud, a été proposé [64].

3.2.2 Apprendre pour différents types de données

— **Problème critique**

La vaste diversité des données constitue la deuxième dimension qui rend le Big Data à

la fois fascinant et stimulant. Cette diversité découle du fait que les données proviennent généralement de multiples sources et sont de types variés. Les sources de données peuvent être structurées, semi-structurées, voire complètement non structurées, ce qui engendre la création de données hétérogènes, de grande dimension et non linéaires, présentant différentes formes de représentation. Apprendre à partir d'un tel ensemble de données expose à un défi considérable, la complexité étant souvent difficile à imaginer avant de plonger profondément dans l'analyse.

— **Solutions possibles**

En ce qui concerne les données hétérogènes, l'intégration de données, qui consiste à combiner des données provenant de diverses sources pour offrir à l'utilisateur une vue unifiée, représente une méthode essentielle. Une solution efficace pour résoudre le problème d'intégration des données consiste à apprendre de bonnes représentations de données à partir de chaque source de données individuelle, puis à intégrer les fonctionnalités apprises à différents niveaux [54]. Ainsi, l'apprentissage par représentation est privilégié dans ce contexte. Dans [65], les auteurs ont introduit une théorie de fusion de données basée sur l'apprentissage statistique pour les données hétérogènes à spectre bidimensionnel. De plus, les méthodes d'apprentissage profondes se sont également avérées très efficaces pour intégrer des données provenant de différentes sources. Par exemple, Srivastava et Salakhutdinov [66] ont développé une nouvelle application des algorithmes d'apprentissage profond pour apprendre une représentation unifiée en intégrant des données d'images denses à valeurs réelles et des données textuelles.

Un autre défi associé à la grande variété de données est que celles-ci sont souvent de haute dimension et non linéaires, comme les modèles climatiques mondiaux, les spectres stellaires et la distribution des gènes humains. Pour traiter ces données de haute dimension, la réduction de dimensionnalité est une solution efficace en identifiant des structures significatives de basse dimension cachées dans les observations de haute dimension. Les approches courantes incluent la sélection ou l'extraction de fonctionnalités pour réduire les dimensions des données. Par exemple, Sun et al. [67] ont proposé un algorithme de sélection de fonctionnalités basé sur l'apprentissage local pour l'analyse de données de grande dimension. Les algorithmes d'apprentissage automatique typiques existants pour la réduction de la dimensionnalité des données comprennent l'analyse en composantes principales (ACP), l'analyse discriminante linéaire (LDA), l'intégration localement linéaire (LLE) et les cartes propres laplaciennes [68]. Plus récemment, les matrices de faible rang jouent un

rôle de plus en plus central dans l'analyse de données à grande échelle et la réduction de dimensionnalité [69, 70]. La récupération d'une matrice de faible rang est un problème fondamental avec des applications en apprentissage automatique [71].

3.2.3 Apprendre à utiliser des données générées à grande vitesse

— **Problème critique**

Pour les big data, la vitesse ou vélocité est un facteur déterminant, posant un autre défi émergent pour l'apprentissage. Dans de nombreuses applications du monde réel, il est crucial de terminer une tâche dans un certain délai, sinon les résultats du traitement deviennent moins utiles, voire sans valeur, comme dans le cas des prévisions de tremblements de terre, des prévisions boursières et des systèmes autonomes d'échange (achat/vente) basés sur des agents. Dans ces situations où le temps est un facteur critique, la valeur potentielle des données dépend de leur fraîcheur, nécessitant un traitement en temps réel.

— **Solutions possibles**

Une solution prometteuse pour apprendre à partir de données à une vitesse élevée est l'apprentissage en ligne. L'apprentissage en ligne [72, 73] est un paradigme d'apprentissage bien établi, dont la stratégie consiste à apprendre une instance à la fois, contrairement à l'apprentissage hors ligne ou par lots, qui nécessite de collecter toutes les données de formation. Ce mécanisme d'apprentissage séquentiel est bien adapté aux big data, car les machines actuelles ne peuvent pas stocker l'ensemble des données en mémoire. Pour accélérer l'apprentissage, un nouvel algorithme a été proposé : la machine d'apprentissage extrême (ELM) [74], conçue pour les réseaux neuronaux à une seule couche cachée (SLFN). Par rapport à d'autres algorithmes d'apprentissage traditionnels, ELM offre une vitesse d'apprentissage extrêmement plus rapide, de meilleures performances de généralisation et une intervention humaine minimale [75].

3.2.4 Apprentissage à partir de données incertaines et incomplètes

— **Problème critique**

Autrefois, les algorithmes d'apprentissage automatique étaient généralement alimentés par des données relativement précises provenant de sources connues et limitées, ce qui rendait les résultats d'apprentissage également fiables. Ainsi, la véracité des données n'était pas un problème majeur. Cependant, avec l'énorme quantité de données disponibles aujourd'hui, la précision et la fiabilité des sources de données deviennent rapidement problématiques,

car ces sources proviennent souvent d'origines diverses et la qualité des données n'est pas toujours vérifiable. Par conséquent, nous incluons la véracité comme le quatrième problème critique pour l'apprentissage avec le big data, soulignant l'importance de traiter et de gérer l'incertitude et l'incomplétude des données.

— **Solutions possibles**

Les données incertaines représentent une réalité particulière où les lectures et collectes de données ne sont plus déterministes, mais soumises à des distributions aléatoires ou probabilistes. L'incertitude des données est courante dans de nombreuses applications. Par exemple, dans les réseaux sans fil, certaines données spectrales sont intrinsèquement incertaines en raison du bruit, de l'évanouissement et de l'ombre omniprésents. De plus, les limitations technologiques des équipements de capteur GPS restreignent également la précision des données à certains niveaux. Pour les données incertaines, le principal défi est que les caractéristiques ou attributs des données ne sont pas capturés par une valeur ponctuelle unique, mais sont représentés sous forme de distributions d'échantillons [40]. Pour gérer l'incertitude des données, on peut appliquer des statistiques récapitulatives comme les moyennes et les variances aux échantillons. Une autre méthode est d'utiliser les distributions de probabilité complètes pour construire un arbre de décision, appelée approche basée sur la distribution. Une autre approche consiste à utiliser les informations complètes véhiculées par les distributions de probabilité pour construire un arbre de décision, appelée approche basée sur la distribution abordée dans [76]. Les auteurs de [76] ont analysé les sources d'incertitude des données, illustré par des exemples concrets, et proposé un algorithme pour construire des arbres de décision à partir de données incertaines selon l'approche basée sur la distribution. Ils ont ensuite établi une base théorique permettant de dériver des techniques d'élagage visant à améliorer considérablement l'efficacité computationnelle des algorithmes associés aux données incertaines.

Le problème des données incomplètes est répandu dans divers domaines avec l'essor du Big Data, souvent dû à des défaillances des appareils de collecte. Apprendre à partir de telles données représente un défi majeur, car la plupart des algorithmes d'apprentissage automatique ne peuvent pas être directement appliqués. Par exemple, dans l'apprentissage de classifieurs, le traitement des données incomplètes est crucial car leur absence peut altérer l'interprétation des données et réduire l'efficacité des modèles prédictifs. Pour aborder ces défis, Chen et Lin [54] ont exploré l'application de méthodes avancées d'apprentissage en profondeur pour gérer les données bruitées et tolérer une certaine irrégularité.

3.2.5 Apprentissage pour des données à faible densité de valeurs et une diversité de sens

— **Problème critique**

En utilisant diverses méthodes d'apprentissage pour analyser de vastes ensembles de données, l'objectif ultime est d'extraire des informations précieuses sous forme de connaissances approfondies ou d'avantages commerciaux à partir de volumes massifs de données. Par conséquent, la valeur est également caractérisée comme une caractéristique saillante du big data [77, 56]. Cependant, il est difficile d'extraire une valeur significative à partir de volumes élevés de données ayant une faible densité de valeurs. Par exemple, la police doit souvent examiner de nombreuses vidéos de surveillance pour traiter des affaires pénales, où quelques images clés sont souvent enfouies parmi une grande quantité de sources vidéo.

— **Solutions possibles**

Les technologies d'exploration de données et la découverte de connaissances dans les bases de données (KDD) jouent un rôle essentiel pour relever le défi des données incomplètes. Ces approches, discutées dans [78, 40], explorent l'utilisation de techniques telles que le regroupement, la classification et les modèles fréquents pour extraire des informations précieuses à partir de vastes ensembles de données, notamment dans le domaine de l'IoT. Wu et ses collègues, dans [40], ont également caractérisé les caractéristiques de la révolution du Big Data et proposé des méthodes pour traiter ces données à l'aide d'algorithmes d'apprentissage automatique et d'exploration de données.

Un autre défi complexe lié à la valeur du big data est la diversité de sens des données : la valeur économique de différentes données peut varier considérablement, même si les mêmes données peuvent avoir une valeur différente selon les perspectives ou les contextes considérés. La diversité des significations des données dans le Big Data nécessite des technologies d'apprentissage cognitif comme Watson d'IBM [79], pour une flexibilité et une intelligence accrues, adaptées aux différents contextes et perspectives.

En résumé, les cinq aspects principaux du big data incluent le volume, la variété, la vélocité, la véracité et la valeur posants des défis distincts pour les techniques d'apprentissage automatique. Pour relever ces défis, l'apprentissage automatique dans le contexte du Big Data diffère considérablement des méthodes traditionnelles. Ainsi, des approches évolutives, multi-domaines, parallèles, flexibles et intelligentes sont privilégiées pour surmonter ces obstacles.

3.3 Travaux connexes

La recherche sur le machine learning appliqué au big data est un domaine en pleine expansion, attirant un grand intérêt pour ses nombreuses applications pratiques et ses défis liés à la scalabilité et à l'efficacité. Dans ce contexte, notre étude se concentre sur l'analyse comparative des algorithmes K-Nearest Neighbors (KNN) et des arbres de décision, adaptés aux environnements de calcul distribués.

La recherche sur la méthode kNN est devenue un sujet très prisé dans l'exploration de données et l'apprentissage automatique depuis que l'algorithme a été proposé en 1967. Pour appliquer la méthode kNN traditionnelle aux big data, les travaux précédents peuvent souvent être classés en deux parties : la recherche rapide des échantillons les plus proches [80] et la sélection d'échantillons représentatifs (ou la suppression de certains échantillons) pour réduire le calcul du kNN [81]. Par exemple, Zhang a proposé une mesure du Facteur Certain (CF) pour traiter l'inadéquation de la distribution des classes biaisées dans les méthodes kNN [82]. Li et al. ont proposé une méthode basée sur la densité pour réduire la quantité de données d'entraînement [83]. Zhao et al. ont proposé un nouvel algorithme basé sur l'utilisation d'échantillons étiquetés et ont ajouté une condition de processus de filtrage, réduisant de manière significative la complexité temporelle du nouvel algorithme, sans effet significatif sur les résultats de l'algorithme [84]. Ces méthodes étaient principalement appliquées pour la recherche rapide [85, 86], la réduction de dimension [87, 88] et l'amélioration de l'efficacité des algorithmes. Kaushik Roy et Prajesh Anchalia ont fait une étude comparative de la méthode des k plus proches voisins dans un environnement informatique distribué utilisant Apache Hadoop qui applique le paradigme MapReduce pour traiter de gros volumes de données et dans une machine simple. Leur étude a montré que l'algorithme sur Hadoop est plus efficace que celui lancé sur une machine simple [89].

Dai, Wei et Ji ont transformé l'algorithme traditionnel C4.5 des arbres de décision en une série de procédures Map et Reduce, et leurs résultats indiquent que leur algorithme présente à la fois une efficacité temporelle et une évolutivité [90].

Ainsi, dans la suite, nous avons réalisé une étude comparative des performances en temps d'éxécution des algorithmes de plus proches voisins (KNN) et d'ID3 pour les arbres de décision, déployés sur une machine simple ainsi que dans un environnement Hadoop. Pour ce faire, nous allons parler des conditions et des configurations expérimentales.

3.4 Configurations expérimentale

3.4.1 Architecture matérielle, réseau et système

Pour la réalisation de nos expérimentations nous avons configuré un cluster hadoop à trois (3) noeuds dans un réseau local. Un noeud a été utilisé comme noeud maitre (namenode) et les deux autres noeuds comme noeuds esclaves (datanode). Nous avons choisi d'utiliser Hadoop pour sa capacité à gérer de grandes quantités de données de manière scalable, fiable et rentable, tout en offrant une flexibilité et un écosystème riche pour répondre à divers besoins en traitement de données [91]. Vous trouverez dans le tableau 3.4.1 toutes les informations concernant les machines utilisées.

TABLE 3.4.1 – Spécifications des nœuds dans le cluster Hadoop

Composant	Nœud Maître	DataNodes
Processeur	Intel(R) à 2,40 GHz	Intel(R) i3
Mémoire RAM	8 Go	4 Go
Système d'exploitation	Ubuntu 20.04.6 LTS.	Ubuntu 20.04.6 LTS.
Version Hadoop	3.3.6	3.3.6

Les configurations du cluster ont été effectuées conformément au guide trouvé dans [92].

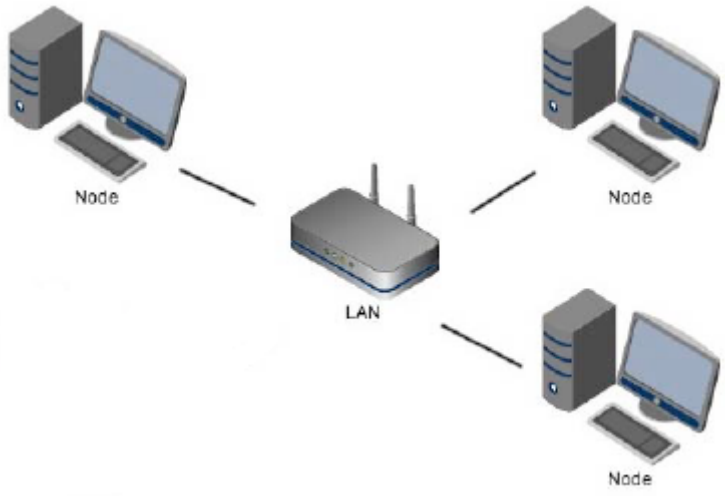

FIGURE 3.4.24 – cluster

3.4.2 Jeux de données

Le langage de progarammation utilisé pour l'implémentation de tous les algorithmes est python car il est très adapté pour l'exploration des données [93].

Nous avons utilisé, pour tout nos algorithmes, un générateur de données aléatoire qui génère, dans des fichiers CSV, des données continues à trois classes dont le nombre d'attribut vaut 4. Nous avons fait varier le nombre de données générées entre **1000** à **800000** données. Ainsi la taille des fichiers CSV générés varie entre **31,4 Ko** à **245 Mo** qui correspondent respectivement à 1000 et 800000 données générées.

3.5 Implémentation mapreduce des algorithmes et expérimentation

La classification des big data devient une tâche essentielle dans divers domaines tels que la biomédecine, les médias sociaux, le marketing, etc. Les progrès récents en matière de collecte de données dans bon nombre de ces domaines ont entraîné une augmentation inexorable du volume de données à gérer. Le volume, la diversité et la complexité qu'apporte le Big Data peuvent entraver les processus d'analyse et d'extraction de connaissances. Dans ce contexte, les modèles standard d'exploration de données doivent être repensés ou adaptés pour traiter ces données. Nous avons choisi d'utiliser l'algorithmes **ID3** qui offre une interprétabilité claire grâce à ses arbres de décision basés sur le gain d'information, tandis que **KNN** excelle par sa simplicité, sa flexibilité et sa capacité à capturer des structures locales dans les données sans nécessiter de phase d'entraînement complexe. Ainsi nous allons utiliser le paradigme **MapReduce** présenté dans le chapitre 1 pour l'implémentation de nos deux algorithmes qui sont :

— **KNN**

— **ID3 des arbres de décision**

Dans ce chapitre, nous utiliserons l'implémentation Hadoop en raison de ses performances, de sa nature open source, de ses fonctionnalités d'installation et de son système de fichiers distribué (Hadoop Distributed File System, HDFS).

3.5.1 Implémentation KNN mapreduce

L'algorithme k-NN est une méthode non paramétrique qui peut être utilisée pour des tâches de classification et de régression. Malgré qu'il soit l'un des algorithmes les plus utiliés et ses résultats prometteurs montrés dans une grande variété de problèmes, k-NN manque d'évolutivité pour traiter de grands ensembles de données. Les principaux problèmes rencontrés pour traiter des données à grande échelle sont :

— **Durée d'exécution** : la complexité de l'algorithme k-NN traditionnel est O((n · D)), où n est le nombre d'instances et D le nombre de fonctionnalités

— **Consommation mémoire** : Pour un calcul rapide des distances, le modèle kNN peut normalement nécessiter de stocker les données d'entraînement en mémoire. Lorsque les données sont trop volumineuses, il peut facilement dépasser la mémoire RAM disponible.

Ces inconvénients motivent l'utilisation des technologies Big Data pour répartir le traitement des k-NN sur un cluster et des nœuds et d'utiliser le paradigme mapreduce pour l'implémentation des algorithmes. L'idée est de diviser le problème en deux étapes principales :

Algorithm 5 Mapper pour k-NN

Entrée : *train_data* (données d'entraînement), *train_labels* (étiquettes d'entraînement), *test_data* (données de test)
Sortie : Paires $(test\_instance, (distance, label\_r))$
for each t in *test_data* **do**
  for each r in *train_data* **do**
    dist(t, r)
    Émettre$(t, (distance, label_r))$
  end for
end for
return =0

Algorithm 6 Reducer pour k-NN

Entrée : Paires $(test\_instance, (distance, label))$, k (nombre de voisins)
Sortie : Étiquettes-prédites
for each s i t **do**
  Collecter(paires$(distance, label)$)
  short(paires)
  Sélect$(k - premires - paire)$
  Prédire(vote-majoritaire(t))
  Émettre$(t, predicted_label)$
end for
return =0

Nous rappelons que l'algorithme kNN est un algorithme de classification simple qui attribue à un exemple de test la classe majoritaire parmi ses k voisins les plus proches dans les données d'entraînement. Lorsqu'il est implémenté en utilisant le framework MapReduce, l'algorithme est divisé en deux phases pour paralléliser le calcul sur un grand ensemble de données.
Dans la phase Mapper, Pour chaque exemple de test dans Test_data, l'algorithme calcule la distance entre cet exemple de test et chaque exemple dans Train_data. Ensuite chaque calcul de distance produit une paire clé-valeur, où la clé est l'exemple de test et la valeur est un tuple contenant la distance calculée et l'étiquette de l'exemple d'entraînement. Ces paires sont émises pour être traitées dans les phases suivantes.

Enfin dans la phase Reducer, Pour chaque groupe d'exemples de test (chaque clé), l'algorithme sélectionne les k paires avec les plus petites distances et applique un vote majoritaire

(pour un problème de classification) ou une moyenne (pour un problème de régression) pour déterminer l'étiquette prédite pour l'exemple de test. La paire résultante, contenant l'exemple de test et l'étiquette prédite, est émise via un fichier.

3.5.2 Comparatif des temps d'éxécution de kNN sur une machine simple et sur cluster

L'application des kNN sur une machine simple et sur cluster hadoop sur différentes tailles de données générées aléatoirement nous a permis de récupérer les temps d'éxécutions que nous avons renseignés dans le tableau suivant.

TABLE 3.5.2 – Temps d'éxécution (10^{-2}*secondes*) de kNN simple et kNN mapreduce pour différentes quantités de données

Nombre de données (10^3)	knn simple($10^{-2}s$)	knn-MapRed($10^{-2}s$)
1	1.56259	15.5706
3	2.60007	15.8252
5	4.68413	16.0995
8	6.25419	18.2557
10	7.81264	23.131
15	12.4969	39.3353
20	17.1880	42.5626
25	20.312	63.465
30	23.441	77.625
35	26.562	87.184
40	31.256	95.500
50	40.657	95.544
60	48.483	107.367
70	54.005	121.429
80	62.497	145.923
90	73.448	208.645
100	79.703	213.363
110	87.500	215.169
120	95.388	251.012
150	117.285	281.651
200	181.282	282.451

Nombre de données (10^3)	knn simple	knn-MapRed
300	234.429	286.96
500	384.419	296.837
800	622.221	328.067

Pour mieux appréhender et visualiser ces résultats nous avons tracer la courbe des temps d'éxecution en fonction du nombre de données pour mieux observer la variation des temps des différents algorithmes.

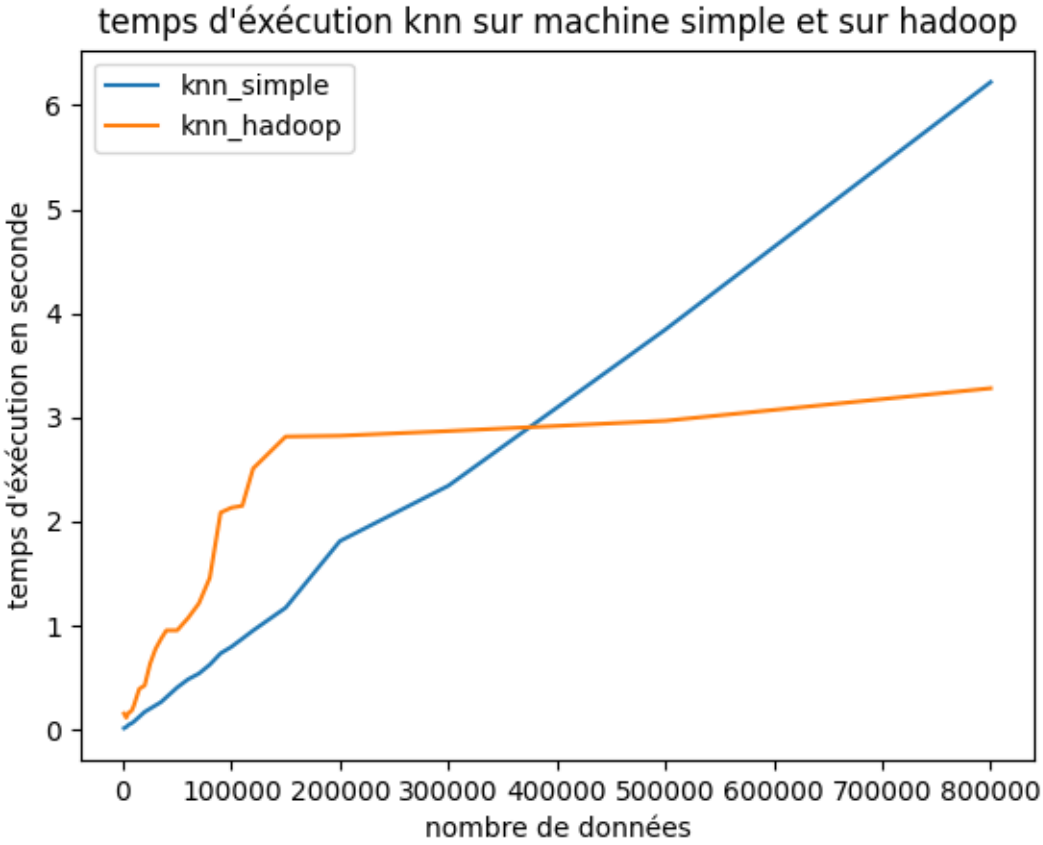

FIGURE 3.5.25 – Courbe des temps d'éxécution en seconde des algorithmes knn simple et knn mapreduce

3.5.3 Implémentation de ID3 arbre de décision

ID3 (Iterative Dichotomiser 3) est un algorithme d'apprentissage supervisé utilisé pour la classification. Il construit des arbres de décision en utilisant une approche de top-down, greedy search à partir de l'ensemble des attributs pour tester chaque attribut à chaque nœud de l'arbre. Bien qu'efficace pour des ensembles de données de taille modérée, ID3 présente plusieurs inconvénients lorsqu'il est appliqué à de grands volumes de données. Parmi eux nous avons :

— **Complexité Computationnelle** : La complexité temporelle de l'algorithme ID3 augmente exponentiellement avec le nombre de caractéristiques et de classes, le rendant peu pratique pour les grands ensembles de données. Chaque division nécessite le calcul de l'entropie et du gain d'information pour chaque attribut, ce qui peut être très coûteux en terme de calcul.

— **Consommation Mémoire** : La construction d'un arbre de décision nécessite de maintenir en mémoire de grandes quantités de données. Pour des ensembles de données volumineux, cela peut entraîner des problèmes de mémoire, en particulier si l'algorithme est implémenté de manière non optimale.

Comme nous avons vu leur fonctionnement dans le chapitre 2, nous allons aborder leur implémentation en adoptant le paradigme mapreduce.

Algorithm 7 Mapper pour ID3

```
Input : Data, Attributes, Label
Output : (attribute, (value, label))
for chaque data_example dans Data do
    for each attribute in Attributes do
        Calcul-valeur(attribute, data_example)
        Émettre(attribute, (value, label))
    end for
end for
return =0
```

Algorithm 8 Reducer pour ID3

```
Input : (attribute, (value, label)), Attributes, Label
Output : (attribute, information-gain)
for each attribute in attribute do
    Calcul-entropie(partitions(attribute))
    Calcul- gain-information(partitions(attribute))
    Émettre(attribute, information_gain)
end for
Sélect(attribut_max_gain)
Divise-données(sélection_attribut)
for each s in sous-ensemble do
    if sous-ensemble pur then
        Creer-feuille(label)
    else
        ID3(sous-ensemble, attributs_restants)
    end if
end for
Assembler(sous-arbres)
return =0
```

L'algorithme ID3 (Iterative Dichotomiser 3) est utilisé pour construire des arbres de décision, principalement dans le cadre de la classification. Lorsqu'il est implémenté avec le framework MapReduce, il est divisé en deux phases pour paralléliser le calcul sur de grands ensembles de

données.

Dans la Phase de mappage, pour chaque exemple de données dans Data, l'algorithme calcule les valeurs pour chaque attribut. Et chaque calcul produit une paire clé-valeur où la clé est l'attribut et la valeur est un tuple contenant la valeur de l'attribut et l'étiquette de l'exemple.

Dans la phase de réduction, Pour chaque attribut regroupé, l'algorithme calcule l'entropie de chaque partition créée par les valeurs de l'attribut puis calcule le gain d'information pour chaque partition en utilisant la formule du gain d'information basée sur l'entropie et émet une paire (attribut, gain d'information) pour chaque attribut. L'attribut avec le gain d'information maximal est sélectionné comme racine de l'arbre de décision. Les données sont divisées en sous-ensembles basés sur les valeurs de l'attribut sélectionné. Et pour chaque sous-ensemble résultant de la division :

Si le sous-ensemble est pur (toutes les instances ont la même étiquette), une feuille de décision est créée avec l'étiquette correspondante. Sinon, l'algorithme ID3 est répété pour le sous-ensemble en utilisant les attributs restants.

Les sous-arbres résultants de l'étape récursive sont collectés et assemblés pour former l'arbre de décision final. Enfin, l'arbre de décision final est structuré de manière à ce que chaque nœud interne représente un test sur un attribut et chaque feuille représente une classe (étiquette).

3.5.4 Comparatif des temps d'éxécution de ID3 sur une machine simple et sur cluster

En appliquant les algorithmes 2 et 7 &8 qui sont respectivement lancés sur une machine simple et dans un cluster sur des données de différentes tailles générées aléatoirement ; nous avons obtenu les résultats communiqués dans dans le tableau suivant. Ces résultats représentent le temps d'éxécution de ces deux algorithmes sur les différents environnements.

TABLE 3.5.3 – Temps d'exécution (10^{-2}*secondes*) de ID3 simple et ID3 mapreduce pour différentes quantités de données

Nombre de données (10^3)	ID3 simple($10^{-2}s$)	ID3-MapRed($10^{-2}s$)
1	3.47	19.5162
3	6.25	40.5344
5	9.38	74.00014
8	18.75	108.68
10	21.88	114.657
15	35.94	160.663

Nombre de données (10^3)	ID3 simple	ID3-MapRed
20	51.57	166.7125
25	67.19	167.997
30	87.57	204.873
35	101.58	236.34
40	118.84	261.29
50	159.53	265.001
60	206.55	276.551
70	222.20	318.685
80	261.08	349.128
90	299.22	369.472
100	339.58	380.806
110	363.52	392.210
120	406.40	392.721
150	504.16	414.797
200	690.87	429.839
300	1033.92	453.487
500	1793.46	520.720
800	3190.45	541.980

Avec ces résultats, nous avons tracé la courbe des temps d'éxecution en fonction du nombre de données pour mieux observer la variation des temps des différents algorithmes.

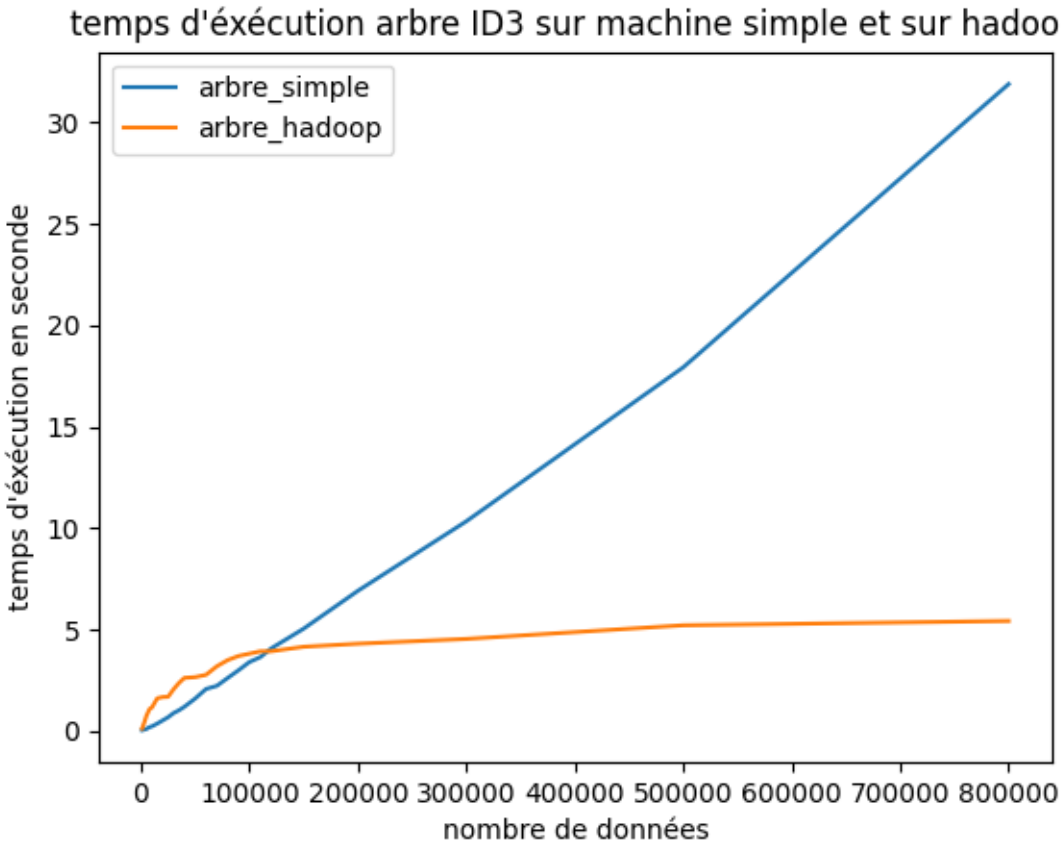

FIGURE 3.5.26 – Courbe des temps d'éxécution en seconde des algorithmes ID3 simple et ID3 mapreduce

3.6 Discussion

Au vu des résultats nous constatons que les temps d'éxécution des algorithmes **knn** et **ID3** sur une machine simple augmentent de façon exponentielle en fonction du nombre de données. Autrement dit, plus les données sont importantes plus le temps d'éxecution augmente. tandis que les temps d'éxécution des algorithmes **knn** et **ID3** sur hadoop augmente moins rapidement en fonction des données puis se stabilise en un moment donnée. La figure 3.5.26 montre qu'avec moins de 150000 données, **ID3** sur une simple machine est plus efficace que **ID3** sur hadoop. Nous notons cela aussi sur la figure 3.5.25 car moins de 400000 données **knn** sur une simple machine est plus performant que le **knn** sur hadoop. Sachant que dans un cluster hadoop les noeuds communiquent en réseau. Cette communication entre les noeuds a un coût en temps. Ce coût dépend de plusieurs facteurs, notamment : la **Latence**, la **bande passante**, le **protocole de communication** et de la **topologie du réseau**. De ce fait avec de petites données les algorithmes lancés sur une machine simple sont plus efficace en terme de temps d'éxécution que les algorithmes lancés dans un cluster avec ces mêmes petites données. Cette observation vient

conforter les resultats de [89].

Cependant, les figures 3.5.26 et 3.5.25 montrent respectivement qu'à partir de 150000 et de 400000 les algorithmes **ID3** et **knn** sur hadoop sont moins couteux en terme de temps d'éxécution que les algorithmes **ID3** et **knn** sur une machine simple. Autrement dit les résultats montrent que pour de grandes masses de données, les algorithmes lancés sous un cluster hadoop prennent moins de temps que les algorithmes lancés sur une machine simple. Ces résultats sont en parfaite corrélation avec ceux de [89] qui dit qu'avec des données de grande taille knn mapreduce hadoop est plus efficace que le knn simple.

Enfin, nous avons remarqué, qu'il a fallu atteindre 400000 pour que knn lancé sur un cluster soit plus performent que le **knn** simple alors qu'il n'a fallu que 150000 données pour que **ID3** hadoop soit plus efficace que le **ID3** simple. Cela s'explique par le fait que l'algorithme **knn** a beaucoup moins de calcul que celui de **ID3**. En effet, dans l'algorithme de **ID3** il y a le calcul de l'entropie , le calcul du gain, la recherche du meilleur attribut etc à faire lors de l'éxecution du programme alors que **knn** ne calcule que la distance entre les données. C'est pour cela que **knn** prend moins de temps pour executer le même nombre de données que **ID3**.

En somme, Les algorithmes de machine learning exécutés sur Hadoop démontrent une efficacité supérieure comparée à leur exécution sur une simple machine, principalement en raison de la capacité d'Hadoop à paralléliser les tâches et à distribuer le traitement des données sur de multiples nœuds. Cette architecture permet de traiter des volumes de données massifs de manière plus rapide et plus efficace. Toutefois, il est important de noter que l'efficacité des algorithmes ne dépend pas uniquement de l'architecture Hadoop. Le choix du matériel physique, tel que la puissance des processeurs, la quantité de mémoire RAM et la vitesse du réseau, ainsi que la configuration logicielle, y compris les paramètres de réglage de Hadoop et des systèmes de gestion des ressources comme YARN, jouent également un rôle crucial dans l'optimisation des performances. En combinant une architecture distribuée performante avec un matériel et des logiciels bien choisis et configurés, il est possible d'atteindre des niveaux de performance et d'efficacité exceptionnels dans le traitement des données massives.

3.7 CONCLUSION

L'application des algorithmes du machine learning sur le big data a beaucoup de défis à surmonter sur le plan du volume, de la variété, de la vitesse, etc. Cependant des solutions ont été

proposées pour palier à ces problèmes. En explorant divers algorithmes appliqués tant sur des machines simples que sous l'infrastructure distribuée de Hadoop, nous avons vu comment ces techniques peuvent être adaptées pour répondre aux défis de traitement des données massives. Au vu de nos expériences et des résultats obtenus, nous avons relevé que pour de grandes masses de données, il est préférable d'utiliser le paradigme mapreduce pour l'implémentation des algorithmes et de les lancer dans un cluster hadoop.

✠ Conclusion générale ✠

Dans un contexte où les volumes de données numériques échangées à travers Internet augmentent de manière exponentielle, le concept de Big Data émerge comme une nécessité incontournable. Cette nouvelle ère de données massives pose des défis considérables en termes de traitement et d'analyse, nécessitant des technologies avancées et des approches innovantes. Les algorithmes de machine learning se révèlent être des outils puissants pour extraire des informations précieuses de ces vastes ensembles de données. Cependant, leur application dans un environnement de Big Data soulève de nombreuses questions sur les performances, l'efficacité et les défis techniques.

Le premier chapitre de ce mémoire est consacré à une étude approfondie du machine learning. Nous avons exploré les différentes approches d'apprentissage, notamment supervisé, non supervisé, semi-supervisé et par renforcement. Ensuite, nous avons détaillé plusieurs algorithmes d'apprentissage automatique, en mettant l'accent sur l'analyse de classification avec les algorithmes KNN (K-plus proches voisins) et ID3 (arbres de décision). Les forces et les faiblesses de ces algorithmes ont été discutées, ainsi que d'autres méthodes telles que la régression et l'analyse par grappes.

Le deuxième chapitre s'est penché sur les concepts fondamentaux du Big Data. Nous avons défini ce qu'est le Big Data, en mettant en avant les 5V (Volume, Vélocité, Variété, Véracité, Valeur) qui caractérisent ce phénomène. Les principales sources de données massives ont été identifiées, ainsi que les défis associés à leur traitement. Nous avons ensuite présenté les technologies clés du Big Data, avec une attention particulière à Hadoop, en décrivant son architecture de cluster, le système de fichiers distribué HDFS, et le framework MapReduce. Les avantages et les limites de l'utilisation de clusters Hadoop ont également été discutés, soulignant leur capacité à gérer des volumes de données massifs de manière efficace et rentable.

L'expérimentation décrite dans le chapitre 3 a permis de comparer les performances des algorithmes KNN et ID3 sur une machine simple et sur un cluster Hadoop constitué de trois nœuds.

Les résultats obtenus montrent clairement les avantages de l'exécution sur un environnement distribué Hadoop. Les performances des algorithmes s'améliorent significativement sur le cluster, démontrant l'efficacité de Hadoop pour traiter des volumes de données massives de manière plus rapide et plus efficace. Cette étude comparative met en lumière l'importance de choisir des infrastructures adaptées pour les applications de machine learning dans un contexte de Big Data. En particulier, l'expérimentation a révélé que l'architecture distribuée de Hadoop permet de surmonter les limitations de performance observées sur une machine simple, en distribuant les tâches de calcul et de stockage de manière optimale.

Les résultats obtenus dans ce mémoire offrent plusieurs perspectives intéressantes pour des recherches futures. Premièrement, il serait pertinent d'explorer l'application d'autres algorithmes de machine learning, tels que les réseaux neuronaux profonds et les méthodes d'ensemble, dans un environnement Hadoop. Ces algorithmes, connus pour leur capacité à gérer des données complexes et non structurées, pourraient bénéficier de l'architecture distribuée de Hadoop pour améliorer leurs performances.

Deuxièmement, l'intégration de technologies complémentaires à Hadoop, comme Apache Spark, pourrait être étudiée pour évaluer leur impact sur les performances des algorithmes de machine learning. Spark, avec ses capacités de traitement en mémoire, pourrait offrir des avantages supplémentaires en termes de vitesse et d'efficacité.

Enfin, une autre perspective de recherche pourrait consister à examiner les aspects de sécurité et de confidentialité des données dans un environnement Hadoop. Avec l'augmentation des volumes de données sensibles traitées, il est crucial de développer des méthodes pour assurer la protection des informations personnelles tout en maintenant des performances élevées.

En définitive, ce mémoire a permis de souligner l'importance de l'intégration des technologies Big Data et des algorithmes de machine learning pour exploiter pleinement le potentiel des données massives. Les résultats de l'expérimentation confirment que l'utilisation de clusters Hadoop offre des gains de performance notables, rendant les traitements de données massives plus efficients. Ces travaux ouvrent la voie à de futures recherches visant à améliorer et à optimiser l'analyse des Big Data. Les progrès dans ce domaine auront des implications significatives pour une variété de secteurs, allant de la santé et la finance à la science des données et l'intelligence artificielle.

✠ Bibliographie ✠

[1] James MANYIKA et al. “Big data : The next frontier for innovation, competition, and productivity”. In : (2011).

[2] Pedro DOMINGOS. “A few useful things to know about machine learning”. In : *Communications of the ACM* 55.10 (2012), p. 78-87.

[3] Tom M MITCHELL. “Artificial neural networks”. In : *Machine learning* 45.81 (1997), p. 127.

[4] Jiawei HAN, Jian PEI et Hanghang TONG. *Data mining : concepts and techniques*. Morgan kaufmann, 2022.

[5] Iqbal H SARKER et al. “Cybersecurity data science : an overview from machine learning perspective”. In : *Journal of Big data* 7 (2020), p. 1-29.

[6] Iqbal H SARKER. “Machine learning : Algorithms, real-world applications and research directions”. In : *SN computer science* 2.3 (2021), p. 160.

[7] Juvénal JVC. *Introduction au Machine Learning avec Python*. Consulté le 22 juillet 2024. URL : https://www.data-transitionnumerique.com/machine-learning-python/.

[8] Mohssen MOHAMMED, Muhammad Badruddin KHAN et Eihab Bashier Mohammed BASHIER. *Machine learning : algorithms and applications*. Crc Press, 2016.

[9] Leslie Pack KAELBLING, Michael L LITTMAN et Andrew W MOORE. “Reinforcement learning : A survey”. In : *Journal of artificial intelligence research* 4 (1996), p. 237-285.

[10] Mahbod TAVALLAEE et al. “A detailed analysis of the KDD CUP 99 data set”. In : *2009 IEEE symposium on computational intelligence for security and defense applications*. Ieee. 2009, p. 1-6.

[11] Fabian PEDREGOSA et al. “Scikit-learn : Machine learning in Python”. In : *the Journal of machine Learning research* 12 (2011), p. 2825-2830.

[12] Geoffrey I WEBB, Eamonn KEOGH et Risto MIIKKULAINEN. "Naıve Bayes." In : *Encyclopedia of machine learning* 15.1 (2010), p. 713-714.

[13] Iqbal H SARKER. "A machine learning based robust prediction model for real-life mobile phone data". In : *Internet of Things* 5 (2019), p. 180-193.

[14] S le CESSIE et JC Van HOUWELINGEN. "Ridge estimators in logistic regression". In : *Journal of the Royal Statistical Society Series C : Applied Statistics* 41.1 (1992), p. 191-201.

[15] Mohamed EL SANHARAWI et Florian NAUDET. "Comprendre la régression logistique". In : *Journal français d'ophtalmologie* 36.8 (2013), p. 710-715.

[16] JL HODGES et E FIX. "Discriminatory analysis, nonparametric discrimination : Consistency properties". In : *USA. FSchool of Aviation Medicine, Randolph Field, Texas* (1951).

[17] Thomas COVER et Peter HART. "Nearest neighbor pattern classification". In : *IEEE transactions on information theory* 13.1 (1967), p. 21-27.

[18] Shiliang SUN et Rongqing HUANG. "An adaptive k-nearest neighbor algorithm". In : *2010 seventh international conference on fuzzy systems and knowledge discovery*. T. 1. IEEE. 2010, p. 91-94.

[19] Luca ROMEO et al. "Machine learning-based design support system for the prediction of heterogeneous machine parameters in industry 4.0". In : *Expert Systems with Applications* 140 (2020), p. 112869.

[20] Hussein A TAHA, Ahmed H SAKR et Soumaya YACOUT. "Aircraft engine remaining useful life prediction framework for industry 4.0". In : *Proceedings of the 4th North America conference on Industrial Engineering and Operations Management, Toronto, ON, Canada*. 2019, p. 23-25.

[21] Chongyu ZHOU et Chen-Khong THAM. "Graphel : A graph-based ensemble learning method for distributed diagnostics and prognostics in the industrial internet of things". In : *2018 IEEE 24th international conference on parallel and distributed systems (ICPADS)*. IEEE. 2018, p. 903-909.

[22] Ping ZHANG, Rongqin WANG et Nianfeng SHI. "IgA nephropathy prediction in children with machine learning algorithms". In : *Future Internet* 12.12 (2020), p. 230.

[23] Niraj THAPA et al. "Comparison of machine learning and deep learning models for network intrusion detection systems". In : *Future Internet* 12.10 (2020), p. 167.

[24] Saravanan THIRUMURUGANATHAN. “A detailed introduction to K-nearest neighbor (KNN) algorithm”. In : *Retrieved March* 20 (2010), p. 2012.

[25] Li-Yu HU et al. “The distance function effect on k-nearest neighbor classification for medical datasets”. In : *SpringerPlus* 5 (2016), p. 1-9.

[26] Saeed KARIMIFARD et al. “Morphological heart arrhythmia detection using hermitian basis functions and kNN classifier”. In : *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE. 2006, p. 1367-1370.

[27] *3 Apprentissage automatique et réduction du nombre de dimensions*. URL : https://webia.lip6.fr/~tollaris/ARTICLES/THESE/node7.html (visité le 22/07/2024).

[28] p LAVANYA DEVI n. et thirumurugan. “classification du cancer du col de l’utérus à partir d’images de frottis de pap à l’aide de moyennes c floues modifiées, pca et knn. journal de recherche de l’iete”. Thèse de doct. 2021.

[29] Ali LABIAD. “Sélection des mots clés basée sur la classification et l’extraction des règles d’association”. Thèse de doct. Université du Québec à Trois-Rivières, 2017.

[30] Hassane HILALI. “Application de la classification textuelle pour l’extraction des règles d’association maximales”. Thèse de doct. Université du Québec à Trois-Rivières, 2009.

[31] Gopalan KESAVARAJ et Sreekumar SUKUMARAN. “A study on classification techniques in data mining”. In : *2013 fourth international conference on computing, communications and networking technologies (ICCCNT)*. IEEE. 2013, p. 1-7.

[32] Tom M MITCHELL et Machine LEARNING. “Mcgraw-hill science”. In : *Engineering/Math* 1 (1997), p. 27.

[33] Hemlata CHAHAL. “ID3 modification and implementation in data mining”. In : *International Journal of Computer Applications* 975 (2013), p. 8887.

[34] J. Ross QUINLAN. “Induction of decision trees”. In : *Machine learning* 1 (1986), p. 81-106.

[35] T Miranda LAKSHMI et al. “An analysis on performance of decision tree algorithms using student’s qualitative data”. In : *International journal of modern education and computer science* 5.5 (2013), p. 18-27.

[36] Paul E UTGOFF. “ID5 : an incremental ID3”. In : *Machine Learning Proceedings 1988*. Elsevier, 1988, p. 107-120.

[37] Leo BREIMAN et al. “Cart”. In : *Classification and regression trees* (1984).

[38] Dongkuan XU et Yingjie TIAN. “A comprehensive survey of clustering algorithms”. In : *Annals of data science* 2 (2015), p. 165-193.

[39] Lior ROKACH. “A survey of clustering algorithms”. In : *Data mining and knowledge discovery handbook* (2010), p. 269-298.

[40] X WU et al. “Semana 07-data mining with big data”. In : *Knowl Data Eng IEEE Trans* 26.1 (2014), p. 97-107.

[41] Andrew MCAFEE et al. “Big data : the management revolution”. In : *Harvard business review* 90.10 (2012), p. 60-68.

[42] Amir GANDOMI et Murtaza HAIDER. “Beyond the hype : Big data concepts, methods, and analytics”. In : *International journal of information management* 35.2 (2015), p. 137-144.

[43] CL Philip CHEN et Chun-Yang ZHANG. “Data-intensive applications, challenges, techniques and technologies : A survey on Big Data”. In : *Information sciences* 275 (2014), p. 314-347.

[44] Doug LANEY et al. “3D data management : Controlling data volume, velocity and variety”. In : *META group research note* 6.70 (2001), p. 1.

[45] Min CHEN, Shiwen MAO et Yunhao LIU. “Big data : A survey”. In : *Mobile networks and applications* 19 (2014), p. 171-209.

[46] Andrew MCAFEE et al. “Big data : the management revolution”. In : *Harvard business review* 90.10 (2012), p. 60-68.

[47] Hugh J WATSON. “Tutorial : Big data analytics : Concepts, technologies, and applications”. In : *Communications of the Association for Information Systems* 34.1 (2014), p. 65.

[48] Manavalan RADHAKRISHNAN et Thangavel KUTTIANNAN. “IJCSI International Journal of Computer Science Issues”. In : *Comparative Analysis of Feature Extraction Methods for the Classification of Prostate Cancer from TRUS Medical Images* 9.1 (2012), p. 1694-0814.

[49] Gayathri RAVICHANDRAN. “Big Data processing with Hadoop : a review”. In : *Int. Res. J. Eng. Technol* 4 (2017), p. 448-451.

[50] HDB HORTONWORKS. “Hortonworks data platform”. In : (2015).

[51] Ted DUNNING et Ellen FRIEDMAN. *Real-world hadoop*. " O’Reilly Media, Inc.", 2015.

[52] Rohit MENON. *Cloudera administration handbook*. Packt Publishing Ltd, 2014.

[53] Dino QUINTERO et al. *Implementing an IBM InfoSphere BigInsights Cluster using Linux on Power*. IBM Redbooks, 2015.

[54] Xue-Wen CHEN et Xiaotong LIN. “Big data deep learning : challenges and perspectives”. In : *IEEE access* 2 (2014), p. 514-525.

[55] TH DAVENPORT, PFP BARTH et R BEAN. “How ‘Big Data’is Different FALL 2012”. In : (2012).

[56] Han HU et al. “Toward scalable systems for big data analytics : A technology tutorial”. In : *IEEE access* 2 (2014), p. 652-687.

[57] Fredrik ANDERSSON et al. “A new frequency estimation method for equally and unequally spaced data”. In : *IEEE Transactions on Signal Processing* 62.21 (2014), p. 5761-5774.

[58] Fu LIN, Makan FARDAD et Mihailo R JOVANOVIĆ. “Design of optimal sparse feedback gains via the alternating direction method of multipliers”. In : *IEEE Transactions on Automatic Control* 58.9 (2013), p. 2426-2431.

[59] Stephen BOYD et al. “Distributed optimization and statistical learning via the alternating direction method of multipliers”. In : *Foundations and Trends® in Machine learning* 3.1 (2011), p. 1-122.

[60] Jeffrey DEAN et Sanjay GHEMAWAT. “MapReduce : simplified data processing on large clusters”. In : *Communications of the ACM* 51.1 (2008), p. 107-113.

[61] Jeffrey DEAN et Sanjay GHEMAWAT. “MapReduce : a flexible data processing tool”. In : *Communications of the ACM* 53.1 (2010), p. 72-77.

[62] Michael ARMBRUST et al. “A view of cloud computing”. In : *Communications of the ACM* 53.4 (2010), p. 50-58.

[63] Marios D DIKAIAKOS et al. “Cloud computing : Distributed internet computing for IT and scientific research”. In : *IEEE Internet computing* 13.5 (2009), p. 10-13.

[64] Yucheng LOW et al. “Distributed graphlab : A framework for machine learning in the cloud”. In : *arXiv preprint arXiv :1204.6078* (2012).

[65] Qihui WU et al. “Spatial-temporal opportunity detection for spectrum-heterogeneous cognitive radio networks : Two-dimensional sensing”. In : *IEEE Transactions on Wireless Communications* 12.2 (2013), p. 516-526.

[66] Nitish SRIVASTAVA et Russ R SALAKHUTDINOV. “Multimodal learning with deep boltzmann machines”. In : *Advances in neural information processing systems* 25 (2012).

[67] Yijun SUN, Sinisa TODOROVIC et Steve GOODISON. “Local-learning-based feature selection for high-dimensional data analysis”. In : *IEEE transactions on pattern analysis and machine intelligence* 32.9 (2009), p. 1610-1626.

[68] Laurens VAN DER MAATEN, Eric POSTMA, Jaap VAN DEN HERIK et al. “Dimensionality reduction : a comparative”. In : *J Mach Learn Res* 10.66-71 (2009).

[69] Qihui WU et al. “Cognitive internet of things : a new paradigm beyond connection”. In : *IEEE Internet of Things journal* 1.2 (2014), p. 129-143.

[70] Morteza MARDANI, Gonzalo MATEOS et Georgios B GIANNAKIS. “Subspace learning and imputation for streaming big data matrices and tensors”. In : *IEEE Transactions on Signal Processing* 63.10 (2015), p. 2663-2677.

[71] Emmanuel J CANDÈS et al. “Robust principal component analysis ?” In : *Journal of the ACM (JACM)* 58.3 (2011), p. 1-37.

[72] Shai SHALEV-SHWARTZ et al. “Online learning and online convex optimization”. In : *Foundations and Trends® in Machine Learning* 4.2 (2012), p. 107-194.

[73] Mikhail BILENKO, S BASIL et Mehran SAHAMI. “Adaptive product normalization : Using online learning for record linkage in comparison shopping”. In : *Fifth IEEE International Conference on Data Mining (ICDM’05)*. IEEE. 2005, 8-pp.

[74] Guang-Bin HUANG, Qin-Yu ZHU et Chee-Kheong SIEW. “Extreme learning machine : theory and applications”. In : *Neurocomputing* 70.1-3 (2006), p. 489-501.

[75] Shifei DING, Xinzheng XU et Ru NIE. “Extreme learning machine and its applications”. In : *Neural Computing and Applications* 25 (2014), p. 549-556.

[76] Smith TSANG et al. “Decision trees for uncertain data”. In : *IEEE transactions on knowledge and data engineering* 23.1 (2009), p. 64-78.

[77] John GANTZ. “Extracting Value from Chaos. The IDC iView”. In : *http ://idcdocserv.com/1142* (2011).

[78] Chun-Wei TSAI et al. “Data mining for internet of things : A survey”. In : *IEEE Communications Surveys & Tutorials* 16.1 (2013), p. 77-97.

[79] John E KELLY et Steve HAMM. *Smart machines : IBM’s Watson and the era of cognitive computing*. Columbia University Press, 2013.

[80] Xiaofeng ZHU, Lei ZHANG et Zi HUANG. “A sparse embedding and least variance encoding approach to hashing”. In : *IEEE transactions on image processing* 23.9 (2014), p. 3737-3750.

[81] Xiaofeng ZHU et al. “Missing value estimation for mixed-attribute data sets”. In : *IEEE Transactions on Knowledge and Data Engineering* 23.1 (2010), p. 110-121.

[82] Shizhao ZHANG. “KNN-CF approach : Incorporating certainty factor to knn classification.” In : *IEEE Intell. Informatics Bull.* 11.1 (2010), p. 24-33.

[83] Rong-Lu LI et YunFa HU. “A density-based method for reducing the amount of training data in kNN text classification”. In : *Journal of computer research and development* 41.4 (2004), p. 539-545.

[84] Dequn ZHAO, Weiwei ZOU et GuangMin SUN. “A fast image classification algorithm using support vector machine”. In : *2010 2nd International Conference on Computer Technology and Development*. IEEE. 2010, p. 385-388.

[85] Xiaofeng ZHU et al. “Sparse hashing for fast multimedia search”. In : *ACM Transactions on Information Systems (TOIS)* 31.2 (2013), p. 1-24.

[86] Xiaofeng ZHU et al. “Linear cross-modal hashing for efficient multimedia search”. In : *Proceedings of the 21st ACM international conference on Multimedia*. 2013, p. 143-152.

[87] Xiaofeng ZHU et al. “Dimensionality reduction by mixed kernel canonical correlation analysis”. In : *Pattern Recognition* 45.8 (2012), p. 3003-3016.

[88] Xiaofeng ZHU et al. “Self-taught dimensionality reduction on the high-dimensional small-sized data”. In : *Pattern Recognition* 46.1 (2013), p. 215-229.

[89] Prajesh P ANCHALIA et Kaushik ROY. “The k-nearest neighbor algorithm using MapReduce paradigm”. In : *2014 5th International conference on intelligent systems, modelling and simulation*. IEEE. 2014, p. 513-518.

[90] Wei DAI et Wei JI. “A mapreduce implementation of C4. 5 decision tree algorithm”. In : *International journal of database theory and application* 7.1 (2014), p. 49-60.

[91] Sifat IBTISUM et al. “A comparative analysis of big data processing paradigms : Mapreduce vs. apache spark”. In : *World Journal of Advanced Research and Reviews* 20.1 (2023), p. 1089-1098.

[92] Anas BOULBALI. *Comment configurer un cluster multi-nœuds Hadoop 3.2.1 sur Ubuntu 18.04 (2 nœuds)*. Consulté le 02 juin 2024. 2020. URL : https://anasblp.medium.com/realisation-b189bc95b0f0.

[93] Fernando PEREZ, Brian E GRANGER et John D HUNTER. “Python : an ecosystem for scientific computing”. In : *Computing in Science & Engineering* 13.2 (2010), p. 13-21.