Centre d'Etudes Doctorales : Sciences et Techniques et Sciences Médicales

N° d'ordre 17/2022

THESE DE DOCTORAT

Présentée par

Mr : Abderrahmane ED-DAOUDY

Spécialité : Informatique

Sujet de la thèse : Machine learning pour l'aide au diagnostic médical en temps réel dans un environnement big data.

Thèse présentée et soutenue le samedi 26 février 2022 à 09h à l'Ecole Nationale des Sciences Appliquées devant le jury composé de :

Nom Prénom	Titre	Etablissement	
Hamid TAIRI	PES	Faculté des Sciences Dhar El Mehraz de Fès	Président
Brahim OUHBI	PES	Ecole Nationale Supérieure d'Arts et Métiers de Meknès	Rapporteur
Mohammed OUANAN	PES	Faculté des Sciences de Meknès	Rapporteur
Jamal RIFFI	PH	Faculté des Sciences Dhar El Mehraz de Fès	Rapporteur
Ahmed ZINEDINE	PES	Faculté des Sciences Dhar El Mehraz de Fès	Examinateur
Aziza EL OUAAZIZI	PH	Faculté Polydisciplinaire de Taza	Examinateur
Youness KHAMLICHI	PH	Ecole Nationale des Sciences Appliquées de Fès	Examinateur
Khalil MAALMI	PES	Ecole Supérieure de Technologie de Fès	Directeur de thèse

Laboratoire d'accueil : Intelligence Artificielle, Sciences de données et Systèmes Émergents

Etablissement : École Nationale des Sciences Appliquées de Fès

Résumé

Dans cette thèse, nous présentons un certain nombre de solutions dédiées au diagnostic médical en termes d'architectures, de méthodologies et d'applications autour de deux grands paradigmes, à savoir le Data Mining (DM) avec Machine Learning (ML) et les technologies big data. Le premier volet de ce travail consiste à améliorer le taux de classification en intégrant des méthodes de sélection des attributs les plus significatifs. Pour ce faire, la méthode des règles d'associations (Association Rules) a été utilisée pour éliminer les attributs non significatifs, puis un ensemble de modèles de classification les plus connus ont été appliqués pour distinguer l'état de santé. La méthode proposée a été validée sur le problème de classification des tumeurs. Le deuxième volet est consacré à la proposition d'une architecture du système de diagnostic et de suivi médicale introduisant à la fois les technologies émergentes de l'écosystème big data et ML parallèle et distribué. Il s'agit d'incorporer les nouvelles technologies du big data, distribuées et parallèle à la fois afin d'apporter des solutions robustes, rapides et évolutives aux exigences de la collection, du calcul intensif, de l'analyse prédictive, du stockage et de la visualisation des données médicales. Pour ce faire, la démarche méthodologique et la réalisation du système sont présentées, ainsi que le déploiement d'une architecture de traitement parallèle et distribuée sur un cluster de nœuds combinant les technologies récentes du big data.

Pour valider notre approche, nous avons testé le système sur la maladie cardiaque et le diabète. Tout d'abord, nous transformons les algorithmes traditionnels en des algorithmes parallèles, distribués, évolutifs et rapides. Deuxièmement, en utilisant Spark streaming, le modèle le plus performant est appliqué à la diffusion de données en continu provenant de sources distribuées pour prédire l'état de santé en temps réel. Le résultat ainsi que le flux de données seront stockés dans une base de données distribuée en vue de mettre en œuvre le reporting et la surveillance en temps réel. Dans un premier lieu, pour tester notre système en termes de sources et du flux de données, nous avons opté pour une application de simulation agissante comme flux de données générés par différentes technologies cliniques (IoT). Dans

un deuxième temps, nous intégrons le Twitter qui sert du canal de communication gratuit. Les utilisateurs tweetent leurs attributs liés à la santé. Le système reçoit tous les attributs souhaités, puis le modèle de ML est appliqué sur les données de flux pour prédire l'état de santé et envoyer un message approprié indiquant l'absence ou la présence de maladie. Les résultats montrent de meilleures performances et ont le potentiel d'ouvrir de nouvelles opportunités et perspectives en ML parallèle et distribué dans un environnement big data.

Mots-clés: Data mining, Big data, Machine learning parallèle et distribué, Temps réel, Classification, Apache Spark, Soins de santé, Traitement de tweet, Traitement de flux, Internet des objets.

Abstract

In this thesis, we present a number of solutions dedicated to medical diagnosis in terms of architectures, methodologies and applications around two main paradigms, namely Data Mining (DM) with Machine Learning (ML) and big data technologies. The first part of this work consists in improving the classification rate by integrating methods for selecting the most significant features. To do so, the Association Rules method was used to eliminate non-significant features, and then a set of the best known classifiers were applied to distinguish the health status. The proposed method was validated on the tumor classification problem. The second part is dedicated to the proposal of a system architecture introducing both the emerging technologies of the big data ecosystem and parallel and distributed ML. The aim is to incorporate new technologies of big data, both distributed and parallel, in order to provide robust, fast and scalable solutions to the requirements of collection, high performance computing, predictive analysis, storage and visualization of medical data. To do so, the methodological approach and the realization of the system are presented, as well as the deployment of a parallel and distributed processing architecture on a cluster of nodes combining recent big data technologies.

To validate our approach, we tested the monitoring system on heart and diabetes diseases. First, we transform traditional algorithms into parallel, distributed, scalable and fast algorithms. Second, using Spark streaming, the best performing model is applied to streaming data from distributed sources to predict health status in real time. The result as well as the data stream will be stored in a distributed database to implement real-time reporting and monitoring. In a first step, to test our system in terms of data sources and data flow, we have opted for a simulation application acting as a data flow generated by different clinical technologies (IoT). In a second step, we integrate the Twitter that serves as the free communication channel. Users tweet their health-related attributes. The system receives all the desired attributes and then the ML model is applied on the stream data to predict the health status and send an appropriate message indicating the absence or presence of disease. The results show better performance and have the potential to open up new opportunities and perspectives in parallel

and distributed ML in a big data environment.

Keywords: Data mining, Big data, Parallel and distributed machine learning, Real-time, Classification, Apache Spark, Healthcare, Tweet processing, Stream processing, Internet of Things.

Dédicaces

A mes très chers parents :

Aucune dédicace, aucun mot ne pourrait exprimer à leur juste valeur la gratitude et l'amour que je vous porte.

Je mets entre vos mains, le fruit de longues années d'études, de longs mois de distance de votre amour de votre tendresse, de longs jours d'apprentissage.

Loin de vous, votre soutien et votre encouragement m'ont toujours donné de la force pour persévérer et pour prospérer dans la vie.

Chaque ligne de cette thèse chaque mot et chaque lettre vous exprime la reconnaissance, le respect et l'estime et le merci d'être mes parents.

A mes chers frères et mes chères sœurs

A tous mes amis

A tous ceux et celles qui me sont chers

Je dédie ce modeste travail.

Remerciements

Au terme de ce travail, je tiens vivement à remercier mon directeur de thèse et professeur monsieur **Khalil MAALMI**, d'abord pour l'opportunité qu'il m'a offerte de faire partie de son équipe de travail. Ensuite, je le remercie pour tous ses précieux conseils et directives, sa sympathie, sa patience, sa compréhension et sa disponibilité durant toute cette période.

Je remercie monsieur **Hamid TAIRI**, professeur de la Faculté des Sciences Dhar El Mehraz de Fès. Je le remercie de m'avoir fait l'honneur de présider le jury de ma thèse, malgré ses nombreuses occupations.

Mes vifs remerciements s'adressent à monsieur **Brahim OUHBI**, professeur de l'Ecole Nationale Supérieure d'Arts et Métiers de Meknès, à monsieur **Mohammed OUANAN**, professeur de la Faculté des Sciences de Meknès, et monsieur **Jamal RIFFI**, professeur de la Faculté des Sciences Dhar El Mehraz de Fès, qui ont bien voulu accepter d'évaluer le présent travail et de rapporter ma thèse malgré les nombreuses activités dont ils ont la charge. Je leur suis reconnaissant pour leur lecture très attentive du manuscrit. Par leurs questions et remarques constructives, ils m'ont permis d'améliorer et d'enrichir certaines parties de mon manuscrit.

Mes remerciements vont aussi à monsieur **Ahmed ZINEDINE**, professeur de la Faculté des Sciences Dhar El Mehraz de Fès, à madame **Aziza EL OUAAZIZI**, professeur de la Faculté Polydisciplinaire de Taza, à monsieur **Youness KHAMLICHI**, professeur de l'Ecole Nationale des Sciences Appliquées de Fès, d'avoir consacré de leur précieux temps à la lecture de cette thèse.

Mes sincères remerciements adressés à tout le corps enseignant et administratif de l'École Supérieure de Technologie de Fès, de l'École Nationale des Sciences Appliquées de Fès et de la Faculté des Sciences et Techniques de Fès. J'exprime ma gratitude et mon amitié à tous mes collègues et mes amis pour leurs disponibilités et leurs encouragements au cours de cette thèse ainsi que pour leurs aides, leurs gentillesses et précieux conseils.

Enfin, je ne pourrais clôturer ces remerciements sans me tourner vers ma famille, mes parents, mes sœurs, mes frères et tout mon entourage proche. Merci du fond du cœur pour votre amour, vos encouragements et pour le soutien de tous les instants. Je vous dédie cette thèse.

Abréviations

ACP Analyse en Composante Principale
AR Règles d'Association
BDSC Big Data Streaming Computing
BN Naïve Bayes
CONF Confiance
DM Data Mining
DT Arbre de Décision
ECD Extraction de Connaissance à partir de Données
ECG Electrocardiogramme
EHR Dossiers de Santé Electroniques
HDFS Système de Fichiers Distribués de Hadoop
IoT Internet des Objets
KDD Knowledge Discovery in Databases
LA Architecture Lambda
LG Régression Logistique
ML Machine Learning
MLlib Machine Learning Library
MLP Perceptron Multicouche
NoSQL Not SQL
RDD Données Distribuées et Résilientes
RF Forêt Aléatoire
RFRS ReliefF et Rough Set
RNA Réseaux de Neurones Artificiels
SDT Spark DT
SGBDR Système de Gestion de Bases de Données Relationnelles
SIH Systèmes d'Information Hospitaliers

SPRF Spark RF
SUPP Support
SVM Machines à Vecteurs de Support
WBCD Wisconsin Breast Cancer Database
YARN Yet Another Resource Negotiator

Table des Matières

Résumé **i**

Abstract **iii**

Dédicaces **v**

Remerciements **vi**

Abréviations **vii**

Introduction générale **xvi**

1 Contexte et motivation de la thèse xvi
2 Objectifs xix
3 Contributions xix
4 Schéma de la thèse xx
5 Publications et communications xxi

I ECD et Machine Learning : application au diagnostic médical **1**

Chapitre 1 : Le processus d'extraction de connaissances à partir de données et machine learning **2**

1 Introduction 2
2 Data mining 2
2.1 Définition 2
2.2 Historique 3
3 Le processus d'extraction de connaissances à partir de données 4
3.1 Le domaine et l'objectif d'extraction 5
3.2 Nettoyage et intégration de données 6
3.3 Transformation de données 6
3.4 La prédiction et la description 6
3.5 La sélection de l'algorithme de DM 7
3.6 L'utilisation de l'algorithme de DM 7

3.7 L'évaluation . . . 7
3.8 L'utilisation des connaissances extraites . . . 8
4 Machine learning . . . 8
4.1 Définition . . . 8
4.2 Machine learning supervisé . . . 8
4.3 Machine learning non supervisé . . . 11
4.4 Mesures de performance des modèles de ML . . . 12
4.5 Tâches de machine learning . . . 13
4.5.1 La classification . . . 13
4.5.2 La prédiction . . . 13
4.5.3 L'estimation . . . 14
4.5.4 Le regroupement (Clustering) . . . 14
4.5.5 Les règles d'association . . . 14
4.5.6 Visualisation . . . 15
5 Machine learning et domaine de la santé : applications et défis . . . 15
5.1 Vue historique . . . 16
5.2 Approches traditionnelles . . . 16
5.3 Approches basées sur les technologies big data . . . 18
5.4 Défis face au domaine de la santé . . . 20
5.4.1 Intégration de données . . . 20
5.4.2 Stockage de données . . . 21
5.4.3 Traitement de données . . . 22
5.4.4 Techniques de machine learning . . . 23
6 Conclusion . . . 24

Chapitre 2 : Amélioration des performances du diagnostic médical par sélection des attributs pertinents 25

1 Introduction . . . 25
2 Algorithmes utilisés de machine learning . . . 25
2.1 Algorithmes de classification . . . 26
2.1.1 Régression logistique . . . 26
2.1.2 Arbres de décision . . . 27
2.1.3 Forêts aléatoires (Random Forest) . . . 28
2.1.4 Réseaux de neurones artificiels . . . 28
2.1.5 Naïve bayes . . . 30
2.1.6 Machines à vecteurs de support . . . 31
2.2 Les règles d'association . . . 32
2.2.1 Algorithme Apriori . . . 34

3 Étude de cas : classification du cancer du sein . . . 36
3.1 Revue de la littérature . . . 37
3.2 La base de données (WBCD) . . . 39
4 Expérimentation . . . 40
4.1 Application des règles d'association à la base de données WBCD . . . 41
4.2 Résultats sans utiliser les règles d'association . . . 43
4.3 Résultats en utilisant les règles d'association . . . 44
4.4 Conclusion . . . 48

II Le diagnostic médical en temps réel dans un environnement big data 49

Chapitre 3 : Principes fondamentaux du big data 50
1 Introduction . . . 50
2 Big data : définition . . . 50
3 MapReduce : les concepts de base . . . 52
4 Les outils d'analyse du big data . . . 54
4.1 Les plateformes distribuées . . . 54
4.1.1 Hadoop . . . 54
4.1.2 Apache Spark . . . 56
4.1.3 Apache Spark Vs Hadoop MapReduce . . . 59
4.2 Les plateformes de streaming . . . 60
4.2.1 Spark streaming . . . 63
4.2.2 Apache Storm . . . 64
4.2.3 Apache Flink . . . 65
4.2.4 Apache Samza . . . 65
5 La chaine de valeur big data . . . 67
5.1 Ingestion . . . 68
5.1.1 La collection . . . 68
5.1.2 Le pré-traitement . . . 69
5.1.3 Le stockage . . . 70
5.2 Analyse . . . 71
5.3 Visualisation . . . 71
6 Applications . . . 72
6.1 Banque, finances et sécurité . . . 72
6.2 Réseaux sociaux . . . 73
6.3 Villes intelligentes . . . 73
7 Conclusion . . . 74

Chapitre 4 : Nouvelle architecture pour le suivi médical en temps réel **76**
1 Introduction . . . 76
2 Revue de la littérature . . . 76
3 L'architecture proposée pour le système de suivi et d'analyse de l'état de santé en temps réel . . . 79
3.1 Collection . . . 80
3.1.1 Source de données . . . 80
3.1.2 Ingestion de données en temps réel avec Kafka . . . 82
3.2 Analyse . . . 83
3.2.1 Spark streaming . . . 83
3.2.2 Architecture du cluster Spark . . . 84
3.2.3 Étude de cas : description de données . . . 84
3.2.4 Implémentation du modèle de classification évolutif . . . 86
3.3 Stockage et visualisation . . . 87
3.3.1 Stockage distribué . . . 87
3.3.2 La visualisation . . . 89
4 Expérimentation et résultats . . . 90
4.1 Implémentation . . . 90
4.2 Évaluation de modèle d'apprentissage . . . 91
4.3 Les performances de Apache Spark vs Weka . . . 93
4.4 Evolutivité du DT à l'aide de Spark . . . 94
4.5 Le débit du cluster Spark . . . 96
4.6 Débit des requêtes . . . 98
4.7 Comparaison des performances de DT avec d'autres ML de MLlib . . . 98
5 Conclusion . . . 102

Conclusion générale et perspectives **104**
1 Conclusion générale . . . 104
2 Perspectives . . . 106

Références **107**

Liste des figures

1.1 ECD à la confluence de nombreux domaines (Kodratoff, 1998). 4
1.2 Processus d'extraction de connaissances à partir de données (Fayyad, Piatetsky-Shapiro, and Smyth, 1996). 5
1.3 Extraction du modèle. 9
1.4 Processus de ML supervisé. 10
1.5 Partitionnement de données. 10
1.6 Partitionnement de la méthode de validation croisée. 11
1.7 Partitionnement de données ou clustering. 11

2.1 Graphe de la fonction segmoid. 27
2.2 Exemple d'arbre de décision. 28
2.3 Construction de RF. 29
2.4 Modèle de réseaux de neurones artificiels. 30
2.5 Réseaux de neurones multi-couche. 31
2.6 Hyperplan de séparation avec les vecteurs de support. 32
2.7 Exemple d'application de l'algorithme Apriori. 36
2.8 Modèle de base impliqué dans cette étude. 41
2.9 Comparaison d'accuracy sans utiliser AR. 44
2.10 Comparaisons d'accuracy. 46
2.11 Temps d'apprentissage. 47

3.1 Les 5V du big data. 51
3.2 Architecture de MapReduce. 53
3.3 Le processus global de MapReduce appliqué à WordCount. 54
3.4 La lecture et l'écriture vers et à partir du HDFS dans les algorithmes itératifs de ML. 55
3.5 Architecture de Apache Hadoop. 55
3.6 Architecture de Apache Spark. 58
3.7 La lecture et l'écriture vers et à partir de la RAM dans Spark. 59
3.8 Processus de Hadoop MapReduce. 60
3.9 Processus de Spark. 60
3.10 Temps d'exécution de KMeans sur les deux frameworks (Zaharia et al., 2012b). . 61

3.11 Temps d'exécution de WordCount. 61
3.12 Architecture de Spark streaming. 63
3.13 Flux de données de Spark streaming. 63
3.14 DStream. 64
3.15 Tendances de recherche de Hadoop, Spark, Storm, Flink et Samza. 66
3.16 La chaine de valeur big data. 67
3.17 Batch Vs stream Vs Micro-batch. 69

4.1 Architecture générale du système proposé. 79
4.2 Architecture du système proposé avec Twitter. 81
4.3 Diagramme opérationnel du système proposé. 82
4.4 Système de messagerie de Kafka. 83
4.5 Architecture de cluster Spark. 84
4.6 Le processus de ML avec Spark. 87
4.7 Le processus de ML. 88
4.8 Temps d'exécution du DT en utilisant Spark MLlib et Weka, temps d'apprentissage de modèle. 93
4.9 Temps d'exécution du DT en utilisant Spark MLlib et Weka, temps de test de modèle. 94
4.10 Temps d'exécution du DT en utilisant Spark pour différents nœuds, temps d'apprentissage de modèle. 95
4.11 Temps d'exécution du DT en utilisant Spark pour différents nœuds, temps de test de modèle. 95
4.12 Temps d'exécution du DT en utilisant Spark et Weka pour différents nœuds, temps d'apprentissage de modèle. 96
4.13 Temps d'exécution du DT en utilisant Spark et Weka pour différents nœuds, temps de test de modèle. 96
4.14 Simulation de flux de données. 97
4.15 Temps de traitement du débit par rapport à l'augmentation de la taille de données pour différents nœuds et différents intervalles. 98
4.16 Comparaison de temps d'exécution des requêtes entre Spark SQL et MySQL. . . 99
4.17 Comparaison de l'accuracy. 99

Liste des tableaux

1.1 Matrice de confusion 12

2.1 Exemple de base de transaction 34
2.2 Règles d'association générées 37
2.3 Description des attributs de WBCD 40
2.4 Exemple d'attributs de WBCD 40
2.5 Résultats d'analyse de la base WBCD sans utiliser AR 43
2.6 Evaluation des performances sans utiliser AR (%) 43
2.7 Résultats d'analyse de la base WBCD en utilisant AR 45
2.8 Évaluation des performances en utilisant AR (%) 45
2.9 La comparaison des performances de AR avec le SVM 45
2.10 Matrices de confusions 46
2.11 Comparaison de notre approche avec d'autres approches 46

3.1 Avantages et inconvénients de Hadoop MapReduce 57
3.2 Avantages et inconvénients d'Apache Spark 59
3.3 Spark Vs Hadoop MapReduce 62
3.4 Spark Vs Strom vs Flink vs Samza 66

4.1 La description de la base de données de diabète 85
4.2 Échantillon de la base de données de la maladie de diabète 85
4.3 La description de la base de données de la maladie cardiaque 86
4.4 Échantillon de la base de données de la maladie cardiaque 86
4.5 Liste des requêtes 90
4.6 Caractéristiques de cluster 91
4.7 Le taux de classification de DT (MLlib) 93
4.8 Les expériences menées 97
4.9 Les mesures d'évaluation des performances avec d'autres modèles (MLlib) 100
4.10 Comparaison de différentes architectures big data avec l'architecture proposée . 100

Introduction générale

1 Contexte et motivation de la thèse

Machine learning (ML) peut être défini comme le processus consistant à trouver des modèles et des tendances inconnus auparavant dans les bases de données et les utilisées pour construire des modèles prédictifs. ML n'est pas nouvel, il a été utilisé de manière intensive et largement dans les différentes domaines et par de nombreuses organisations. Dans le domaine des soins de santé, ML est de plus en plus populaire, voir essentiel. Les applications de ML peuvent grandement profiter à toutes les parties impliquées dans le secteur de la santé. Par exemple, ML peut aider les assureurs des soins de santé à détecter les fraudes et les abus, les organismes de santé à prendre des décisions en matière de gestion de la relation client, les médecins à identifier les traitements efficaces et les meilleures pratiques, les patients à recevoir des services de santé de meilleure qualité et plus abordables. L'énorme volume de données généré par les transactions des soins de santé est trop complexe et volumineux pour être traité et analysé par les méthodes traditionnelles. ML fournit la méthodologie et la technologie permettant de transformer ce grand nombre de données en informations utiles pour la prise de décision. L'une des tâches les plus importantes de ML est de sélectionner la bonne technique de ML. La technique de ML doit être choisie en fonction du problème confronté. Une approche généralisée doit être utilisée pour améliorer la précision et la rentabilité de l'utilisation de ML. Les techniques de ML, considérées comme étant plus fréquemment utilisées sont abordées dans cette thèse. Pour mettre en œuvre les techniques de ML, deux composantes seront utilisées : la première est la base de données et la seconde est la technique de ML. La base de données offre des techniques de gestion des données tandis que ML offre des techniques automatiques d'analyse des données.

ML est essentiellement le domaine de l'informatique grâce auquel les systèmes informatiques peuvent donner un sens aux données de la même manière que les êtres humains. En

terme simple, il est un type d'intelligence artificielle qui extrait des modèles à partir de données brutes en utilisant un algorithme ou une méthode de DM. L'objectif principal du ML est de permettre aux systèmes informatiques d'apprendre à partir des expériences sans avoir été programmés explicitement ou sans intervention humaine. Les techniques de ML, par exemple, appliquées aux sujets de la science médicale, se développent rapidement en raison de leur potentiel à donner une grande précision dans la classification et la prévision des résultats, à promouvoir la santé des patients, à améliorer la valeur et la qualité des soins de santé en prenant des décisions en temps réel. Cela permettra de réduire les complications de plusieurs tests et d'économiser de l'argent. Cependant, avec la quantité toujours croissante de données qui peuvent être décrites avec du big data, le besoin des frameworks pour traiter ces données devient de plus en plus important. Le big data n'est pas seulement une énorme quantité de données, mais aussi un nouveau paradigme et un ensemble de méthodes qui peuvent analyser et traiter ces données. Dans ce domaine, un ensemble de frameworks avancés et open source axés sur le traitement du big data ont émergés.

La conception d'un système distribué et parallèle pour traiter du big data de santé fait face à trois principaux défis : premièrement, en raison de l'hétérogénéité et de l'énorme volume de données, il est difficile de collecter des données à partir des endroits répartis. Deuxièmement, le stockage est le principal problème pour les bases de données hétérogènes et volumineuses. Un système big data doit stocker tout en offrant la garantie des performances. Le dernier défi est lié à l'analyse de big data, plus précisément à l'exploration des données massives en temps réel qui comprennent la modélisation, la visualisation, la classification, la prédiction et l'optimisation. Ces défis exigent un nouveau paradigme de traitement car les systèmes de gestion de données actuels ne sont pas efficaces pour traiter les données de nature hétérogène ou en temps réel. Cependant, les systèmes traditionnels de gestion de bases de données relationnelles (SGBDR) tels que MySQL sont principalement utilisés pour la gestion de données structurées. Ces systèmes traditionnels n'offrent aucun support pour les données non structurées ou semi-structurées. Du point de vue de l'extensibilité, lorsque la taille de données augmente, il y a de nombreuses défaillances du SGBDR standard dans l'extensibilité pour la gestion parallèle du matériel et la tolérance aux pannes, ce qui n'est pas adapté à la gestion de données croissantes. Pour faire face aux problèmes liés au stockage de données massives et hétérogènes, de nombreux travaux de recherche ont été proposés par la communauté scientifique, tels que les systèmes de gestion de bases de données NoSQL (Not SQL).

MapReduce avec son système de fichier HDFS est également très tolérant aux pannes mais il ne supporte que le traitement par lots, ce qui est limité pour le traitement en temps réel. En revanche, le stream computing implique une entrée et une sortie continue de données et met l'accent sur la vitesse des données. Big data streaming computing (BDSC) fournit un haut débit, des messages distribués, un calcul en temps réel et un traitement à faible latence. Avec ses architectures de traitement massivement parallèles, le BDSC est un bon choix pour acquérir

des connaissances utiles à partir de big data, ce qui est une exigence clé de l'analyse de big data dans les soins de santé.

L'expansion rapide d'analyse de données à grande échelle a commencé à jouer un rôle essentiel dans le développement des pratiques et de la recherche dans les soins de santé. Elle a fourni des outils pour la collecte, la gestion, l'analyse et l'absorption de grandes quantités de données, structurées et non structurées, produites par les systèmes de santé existants. Aujourd'hui, le BDSC joue un rôle important dans l'analyse du big data pour obtenir la valeur cachée du big data en temps réel en soins de santé. Cependant, en raison de la répartition des sources de données sur les soins de santé (les données proviennent de différentes sources), telles que les dispositifs médicales (IoT). L'application de ML à ce flux du big data constitue un défi, car les systèmes de ML traditionnels ne sont pas adaptés pour traiter un volume aussi important ou avec une vitesse aussi variable. Un autre problème est lié au traitement analytique de données.

L'enrichissement de traitement analytique de données implique une intégration efficace de données entre les systèmes. La plupart des travaux de recherche font appel au ML, mais dans le cas de ML en temps réel appliqué au flux de big data n'est pas pris en charge. D'autre part, la plupart des solutions analytiques dans le domaine de la santé sont principalement axées sur Hadoop, qui est un système de traitement par lots. Récemment, le nombre de personnes âgées et de citoyens souffrant des maladies chroniques augment rapidement, les inconvénients des services de santé conventionnels deviennent de plus en plus importants. En outre, l'utilisation de l'IoT médicale se développe pour un suivi continu afin de pouvoir agir en temps réel en cas d'urgence, notamment pour les maladies cardiaques. C'est pourquoi des millions de capteurs génèrent un volume massif de données. Le traitement de ces données et la réalisation d'actions en temps réel dans des situations critiques est une tâche difficile. En générale, il faut consulter un professionnel de santé, mais les valeurs des signes vitaux sont disponibles auprès de la personne pour s'informer de son état de santé, ce qui n'est pas une option envisageable à faire fréquemment. Grâce aux techniques avancées de ML appliquées aux données de santé déjà disponibles, les systèmes informatiques actuels peuvent prévoir l'état de santé avec une précision raisonnable.

L'industrie des soins de santé génère aujourd'hui une grande quantité de données qui peut être décrite par les caractéristiques 5V du big data, principalement le volume, la variété, la vitesse, la véracité et la valeur. Le volume fait référence aux données de santé à collecter et à analyser qui sont considérables et en constante augmentation, la variété fait référence aux données de santé collectées à partir de sources multiples. Les données sur les soins de santé et la connaissance du domaine de la santé doivent être à jour, notamment en ce qui concerne la vitesse. La véracité fait référence à la fiabilité des données sur les soins de santé. Enfin, des informations précieuses pourraient être trouvées en analysant soigneusement les données massives sur les soins de santé. Dans cette thèse, nous nous sommes concentrés sur les cinq

grands défis du big data, tels que l'intégration et le traitement de données, les techniques de DM et ML, le stockage et la visualisation des données.

2 Objectifs

Les objectifs de cette thèse sont d'aborder les défis du diagnostic médical :

- Proposer une méthode pour réduire le nombre des attributs de données en utilisant une technique de DM, puis un système de diagnostic médical en utilisant les algorithmes de ML pour prédire l'état de santé des patients. Cette méthode permet de gagner du temps pendant la phase l'apprentissage menant à une meilleure précision et à des systèmes de classification automatique et rapides.
- Elaborer une étude comparative des outils de traitement du big data.
- Proposer et développer une nouvelle architecture pour un système de suivi et d'analyse de l'état de santé en temps réel en utilisant les technologies big data, ML parallèle et distribué pour surmonter les défis liés au traitement du big IoT data de santé en temps réel.

3 Contributions

Nos travaux de recherches sont couronnés par quatre principales contributions :

1. La première contribution présente une approche combinant la réduction de dimension de données. L'approche proposée comporte deux étapes. Dans la première, la méthode des règles d'association (AR) a été utilisée pour éliminer les attributs non significatifs. Dans la seconde, plusieurs modèles de classifications sont appliqués pour différencier les entrées. Pour mettre en évidence la performance de l'approche proposée, elle a été testée sur la base de données académique de cancer du sein connue par sa complexité. Les résultats montrent que l'approche proposée est plus performante pour réduire le nombre des attributs, gagner du temps pendant la phase l'apprentissage ainsi que pour le traitement des patients menant à une meilleure précision et à des systèmes de classification automatique plus rapide.
2. La seconde contribution porte sur le concept du big data et les outils utilisés pour le traitement du big data. Nous identifions selon différentes perspectives, les outils et les méthodes existantes. Cette contribution nous a permis de choisir les outils adéquats pour proposer et mettre en œuvre une nouvelle architecture de système de suivi et d'analyse de l'état de santé en temps réel.

3. La troisième contribution consiste à une étude comparative des algorithmes de ML en se basant sur le framework de traitement du big data, Apache Spark. Cette étude permet de choisir un modèle de classification plus efficace pour la mise en œuvre d'une solution efficace et évolutive qui permet de prédire les maladies dans un délai court. À ce stade, les performances de six algorithmes de classification bien connus, à savoir les Machines à Vecteurs du Support (SVM), Arbre de Décision (DT), Forêts Aléatoire (RF) et Régression Logistique (LG), Multilayer Perceptron (MLP), Naïve Bayes (NB) ont été évaluées pour la prédiction des maladies. La comparaison des performances globales a été évaluée en termes de la précision de classification, de temps d'apprentissage et de teste.

4. Dans la quatrième contribution, afin de surmonter les problèmes liés au traitement du big data de santé en temps réel, nous avons proposé et développé une nouvelle architecture pour un système de suivi et d'analyse de l'état de santé en temps réel en utilisant les technologies big data. Le système est basé sur l'application d'un modèle d'apprentissage distribué, parallèle et évolutif sur les attributs liés aux données de santé provenant des sources en continue et distribuées de différentes maladies. Les algorithmes standards sont transformés en des algorithmes parallèles, distribués, évolutifs et rapides en utilisant Spark au lieu de Hadoop MapReduce qui devient limité pour les calculs en temps réel. Sur la base de plusieurs données, le système prédit l'état de santé, envoie un message d'alerte aux prestataires des soins de santé et aux utilisateurs, puis il stocke les détails dans une base de données distribuée pour effectuer l'analyse et le reporting de données. Notre approche a été testée sur la maladie cardiaque et le diabète.

4 Schéma de la thèse

Par ailleurs, la présente thèse s'articule autour de deux parties et quatre chapitres; la première concerne le ML et ses applications dans le domaine de la santé tandis que la deuxième partie concerne le big data et le traitement en temps réel.

Le premier chapitre présente les concepts fondamentaux du DM et ML, où sont décrites les différentes étapes d'un processus d'extraction de connaissances à partir des données. Parmi ces étapes, nous détaillons la phase de data mining ainsi que les différentes approches de sa mise en œuvre. Ainsi, nous présentons les principales tâches et méthodes de data mining ainsi que leurs applications et défis en santé.

Dans le deuxième chapitre, nous nous effectuons un tour d'horizon sur les algorithmes de ML les plus connus dans la littérature et nous proposons une approche hybride combinant la réduction de données grâce à la sélection des attributs significatives et un algorithme de ML pour améliorer les performances du diagnostic, le cancer du sein a été testé comme étude de cas. Vers la fin de ce chapitre, plusieurs résultats sont exposés.

Le troisième chapitre porte sur le big data, ça définition, son historique et sa chaine de valeur depuis la collection jusqu'à la visualisation en passant par le traitement. Il propose également un tour d'horizon des outils de traitement du big data en comparant leurs caractéristiques et leurs performances. Ainsi, nous mettons l'accent sur le choix de l'outil approprié pour le big data streaming et en particulier les données de santé.

Le dernier chapitre traite une nouvelle approche pour le traitement du big data en santé, il s'agit d'une nouvelle architecture pour le suivi médical qui comporte la collection, le traitement (classification) et la visualisation de données de l'état de santé en temps réel. Les résultats expérimentaux sont menés afin de mesurer l'évolutivité, la robustesse et la performance de l'architecture proposée. Nous clôturons ce mémoire de thèse par une conclusion générale, où nous suggérons les éventuelles perspectives.

5 Publications et communications

Publications

- ED-DAOUDY, Abderrahmane et MAALMI, Khalil. Real-time heart disease detection and monitoring system based on fast machine learning using Spark. **Health and Technology, Springer**, 2020, vol. 10, no 5, p. 1145-1154.
- ED-DAOUDY, Abderrahmane et MAALMI, Khalil. Breast cancer classification with reduced feature set using association rules and support vector machine. **Network Modeling Analysis in Health Informatics and Bioinformatics, Springer**, 2020, vol. 9, p. 1-10.
- ED-DAOUDY, Abderrahmane et MAALMI, Khalil. A new Internet of Things architecture for real-time prediction of various diseases using machine learning on big data environment. **Journal of Big Data, Springer**, 2019, vol. 6, no 1, p. 1-25.
- ED-DAOUDY, Abderrahmane et MAALMI, Khalil. A scalable and real-time big data predictive analytics system for disease prediction using distributed machine learning. **Multimedia Tools and Applications, Springer**, 2021. (Under review).

Communications

- ED-DAOUDY, Abderrahmane et MAALMI, Khalil. "Performance evaluation of machine learning based big data processing framework for prediction of heart disease." 2019 International Conference on Intelligent Systems and Advanced Computing Sciences (ISACS). FP Taza 26-27 December 2019. **IEEE**.
- ED-DAOUDY, Abderrahmane et MAALMI, Khalil. "Real-time machine learning for early detection of heart disease using big data approach." 2019 International Conference on

Wireless Technologies, Embedded and Intelligent Systems (WITS). ENSA Fès 3-4 April 2019. **IEEE**.

- ED-DAOUDY, Abderrahmane et MAALMI, Khalil. "Application of machine learning model on streaming health data event in real-time to predict health status using spark." 2018 International Symposium on Advanced Electrical and Communication Technologies (ISAECT). ENSA Kénitra 21-23 November 2018. **IEEE**.
- ED-DAOUDY, Abderrahmane et MAALMI, Khalil. "Breast cancer classification using Apache Spark machine learning and GLCM." 2017 International Symposium on Computer Science and Application(ISCSA). FST Errachidia 26-28 October 2018.

Partie I

ECD et Machine Learning : application au diagnostic médical

1 Le processus d'extraction de connaissances à partir de données et machine learning

1 Introduction

La tendance de ces dernières décennies à informatiser le processus de traitement des maladies assure une accumulation plus rapide des informations médicales. Les technologies de l'information sont activement utilisées dans le secteur de la santé. Les institutions des soins de santé mettent en place et déploient des systèmes d'information hospitaliers (SIH), de contrôle, d'archivage et autres. Ces systèmes accumulent de données médicales structurées du patient incluant des attributs classés, les résultats des tests et les données non structurées, telles que les fichiers images et vidéos. L'analyse et l'exploration de ces données sont stratégiquement importantes pour le secteur de la santé et pour chaque patient. Une analyse profonde de données accumulées offre de nouveaux instruments pour le diagnostic plus rapide du patient, sélection du traitement optimal, prévision de la durée de traitement et son issue et l'optimisation des coûts des institutions de santé. Par rapport à d'autres disciplines scientifiques et techniques, l'extraction de connaissances à partir de données médicales en utilisant les méthodes de DM et les techniques de ML est à ses débuts.

Dans ce chapitre, une définition, le processus d'extraction de connaissances à partir de données, les tâches et les méthodes de ML sont décrites, avec un accent sur l'applicabilité dans le domaine de la santé. C'est donc dans ce contexte que nous nous intéressons et particulièrement aux travaux relatifs à l'analyse de données de soin de santé.

2 Data mining

2.1 Définition

Malgré le fait que les méthodes de DM sont appliquées depuis les années 60, l'expression data mining est apparue vers le début des années 1960 et avait, à cette époque, un sens péjoratif. Elle

existe de nombreuses définitions de DM. U. Fayyad et al. ont défini le DM comme une tentative de trouver de modèles utiles à partir de données (Fayyad, Piatetsky-Shapiro, and Smyth, 1996) et complète la notion d'extraction de connaissance à partir de données (knowledge discovery in databases (KDD), Extraction de connaissance à partir de données (ECD)). Depuis les années 90, ces termes sont utilisés de manière interchangeable par des statisticiens, des analystes de données et des experts en systèmes d'information. Le DM peut être définit comme le processus d'extraction des modèles et relations intéressantes et utiles à partir de grands volumes de données avec les sous-domaines de la modélisation prédictive et descriptive, de l'extraction par modèle et l'exploration des anomalies.

2.2 Historique

Vers la fin des années 1980, afin de valoriser les bases de données, Rakesh Agrawal (Agrawal, Imieliński, and Swami, 1993), ont commencé à travailler sur l'exploration du contenu des bases de données volumineuses comme par exemple celles des tickets de caisses des grandes surfaces, convaincus de pouvoir valoriser ces masses de données dormantes. En mars 1989, Shapiro Piatetski (Piatetsky-Shapiro, 1991) proposa le terme knowledge discovery (Extraction de connaissance) à l'occasion d'un atelier sur l'ECD. La première conférence internationale sur le thème DM a été organisée par ACM aux États-Unis en 1995 à la suite de nombreux ateliers sur le ECD entre 1989 et 1994. D'où la première revue du domaine intitulée "Data mining and knowledge discovery journal" publiée par Kluwers a été lancée en 1997.

La notion de DM utilisée dans cette thèse fait partie de processus d'extraction de connaissances qui utilise des méthodes d'analyse de données telles que les statistiques, ML et l'intelligence artificielle pour atteindre de nouvelles connaissances non triviales, par exemple des valeurs de prédiction, des modèles cachés et des dépendances, qui peuvent continuer à être utilisés pour la prise de décision.

L'application des techniques liées au DM dans le domaine médical a d'abord connu un rythme de croissance lent. Toutefois, on observe depuis peu une quantité exponentielle de publications sur le sujet. Les informations médicales peuvent être exprimées sous forme d'informations statiques, en enregistrant l'état instantané d'un patient, par exemple les résultats des tests etc. L'exploration de ces multiples données nécessite l'adaptation des méthodes et les outils standards spécialisés qui assurent l'interopérabilité, l'entrepôt de données et plus généralement une méthodologie d'application détaillée.

3 Le processus d'extraction de connaissances à partir de données

Le terme ECD, fait référence à la procédure générale de découverte de connaissances dans les données et met l'accent sur les applications de haut niveau des techniques spécifiques de DM. C'est un domaine qui intéresse les chercheurs dans divers domaines, notamment l'intelligence artificielle, ML, la reconnaissance des formes, les bases de données, les statistiques, l'acquisition de connaissances pour les systèmes experts et la visualisation de données (Figure 1.1). L'objectif principal du processus d'ECD est d'extraire des informations à partir de

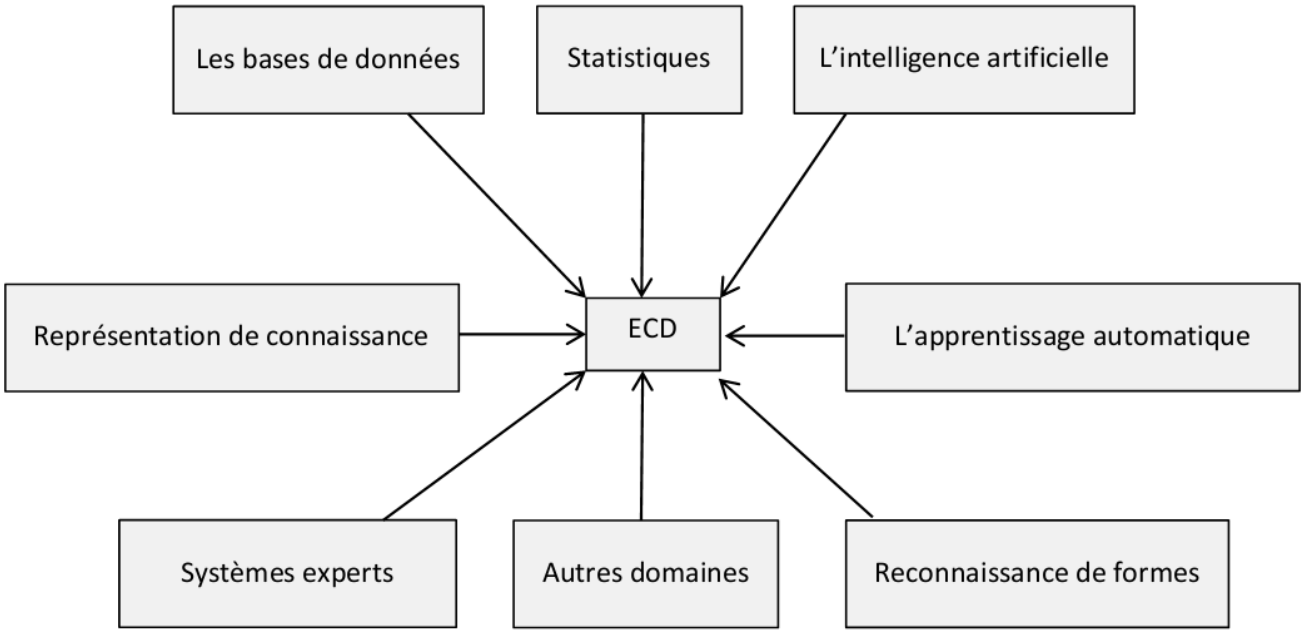

FIGURE 1.1: ECD à la confluence de nombreux domaines (Kodratoff, 1998).

données dans le contexte de grandes bases de données. Pour ce faire, il utilise des algorithmes de DM pour identifier ce qui est considérée comme des connaissances.

Le processus d'extraction de connaissances (illustré dans la figure 1.2) est itératif et interactif. Le processus est itératif à chaque étape, ce qui implique qu'il peut être nécessaire de revenir aux actions précédentes. Le processus a de nombreux aspects créatifs dans le sens où l'on ne peut pas présenter une formule ou faire une catégorisation scientifique complète pour les décisions correctes pour chaque étape et type de demande. Il est donc nécessaire de comprendre le processus et les différentes exigences et possibilités à chaque étape.

Le processus d'ECD comprend quatre grandes étapes, à savoir la sélection de données, la transformation de données, les techniques de DM et enfin l'interprétation des résultats (Gouda and Zaki, 2001). Ces étapes peuvent être plus détaillées en huit étapes.

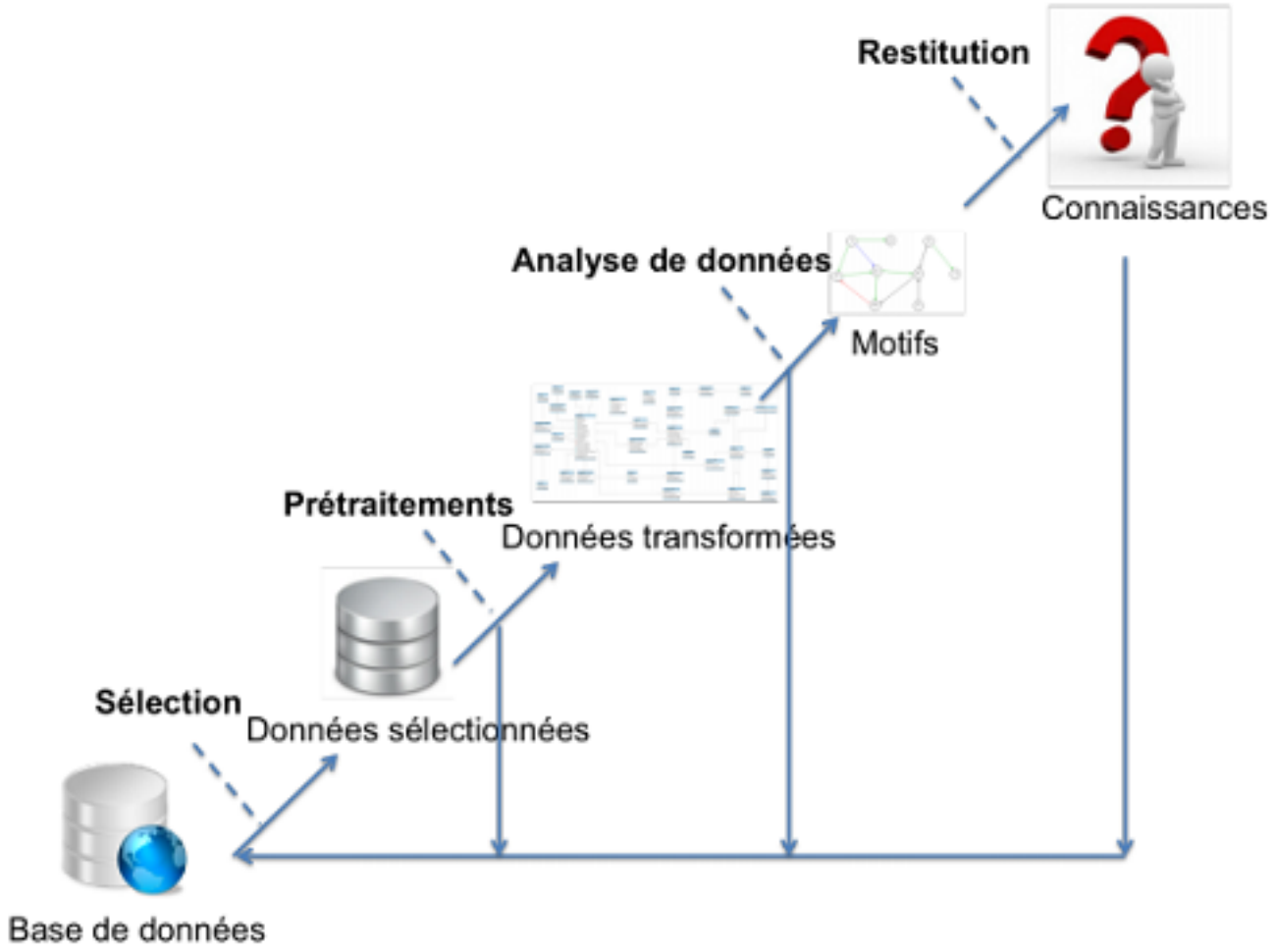

FIGURE 1.2: Processus d'extraction de connaissances à partir de données (Fayyad, Piatetsky-Shapiro, and Smyth, 1996).

3.1 Le domaine et l'objectif d'extraction

Il s'agit d'une étape préliminaire essentielle. Elle développe la scène pour comprendre ce qu'il faut faire avec les différentes décisions comme la transformation, les algorithmes, la représentation, etc. Les personnes qui sont en charge de l'ECD doivent comprendre et caractériser les objectifs de l'utilisateur final et l'environnement dans lequel le processus d'extraction de connaissances se déroulera (implique des connaissances préalables et pertinentes). Une fois les objectifs définis, il convient de déterminer les données qui seront utilisées pour le processus d'extraction de connaissances. Cela implique de découvrir quelles données sont accessibles, d'obtenir de données pertinentes, et ensuite, d'intégrer toutes les données pour construire une seule base de données qui sera prise en compte pour le processus ECD. Ce processus est important car le DM apprend et découvre à partir de données accessibles. Il s'agit de la base de preuves pour la construction des modèles. Si certains attributs significatifs manquent, à ce stade, alors l'étude entière peut être infructueuse de ce point de vue, d'autant plus d'attributs

sont pris en compte. D'autre part, l'organisation, la collecte et l'exploration des entrepôts de données avancés sont coûteuses, et il existe un arrangement permettant de comprendre au mieux les phénomènes. Cet arrangement se réfère à un aspect interactif et itératif de l'ECD. Cela commence avec les données disponibles et se développe ensuite on observant l'impact en termes d'extraction de connaissances et de modélisation.

3.2 Nettoyage et intégration de données

Dans cette étape, la fiabilité de données est améliorée. Elle comprend l'épuration de données, par exemple, le traitement de quantités manquantes et la suppression du bruit ou des valeurs aberrantes. Elle peut inclure des techniques statistiques complexes ou utiliser un algorithme de DM dans ce contexte. Par exemple, lorsque l'on soupçonne qu'un attribut spécifique manque de fiabilité ou qu'il manque de nombreuses données, à ce stade, cet attribut pourrait devenir l'objectif d'un algorithme supervisé de DM. Un modèle de prédiction pour ces attributs sera créé et après cela, les données manquantes pourront être prédites.

3.3 Transformation de données

Au cours de cette étape, la création de données appropriées pour le DM est préparée et développée. Les techniques utilisées ici comprennent la réduction de dimension, la sélection et l'extraction des attributs et l'échantillonnage des enregistrements, ainsi que la transformation des attributs à savoir la discrétisation des attributs numériques. Cette étape peut être essentielle pour la réussite de l'ensemble du projet ECD, et elle est généralement très spécifique au projet. Par exemple, dans les évaluations médicales, le quotient des attributs peut souvent être le facteur le plus significatif et non pas chacun d'entre eux en lui-même. Dans le monde des affaires, il peut être nécessaire de réfléchir à des impacts qui échappent à notre contrôle, ainsi qu'à des efforts et à des questions transitoires. Par exemple, étudier l'impact de l'accumulation de la publicité. Cependant, si nous n'utilisons pas la bonne transformation au départ, nous pouvons alors acquérir un effet étonnant qui nous renseigne sur la transformation nécessaire à la prochaine itération. Ainsi, le processus ECD se poursuit et permet de comprendre la transformation requise.

3.4 La prédiction et la description

Dans cette étape on va décider du quel type de DM à utiliser, par exemple, la classification, la régression, le regroupement, etc. Cela repose principalement sur les objectifs de l'ECD, mais aussi sur les étapes précédentes. Il y a deux objectifs importants dans le DM, le premier est une prédiction, et le second est la description. La prédiction est généralement appelée DM supervisé, tandis que le DM descriptif intègre les aspects non supervisés et de visualisation de

DM. La plupart des techniques de DM dépendent de l'apprentissage inductif, où un modèle est construit explicitement ou implicitement en généralisant à partir d'un nombre adéquat de modèles de préparation. L'hypothèse fondamentale de l'approche inductive est que le modèle préparé s'applique à des cas futurs.

3.5 La sélection de l'algorithme de DM

Ayant la technique, nous décidons maintenant des stratégies. Cette étape comprend le choix d'une technique particulière à utiliser pour la recherche de modèle comprenant plusieurs inducteurs. Par exemple, si l'on considère la précision par rapport à la compréhensibilité, la première est meilleure avec les réseaux de neurones, tandis que la seconde est meilleure avec les arbres de décision. Pour chaque système d'apprentissage, il existe plusieurs possibilités de réussite. L'apprentissage se concentre sur la clarification de ce qui fait qu'un algorithme de DM est fructueux ou non dans un domaine spécifique. Ainsi, cette méthodologie tente de comprendre la situation dans laquelle un algorithme de DM est le plus approprié. Chaque algorithme a des paramètres et des stratégies d'apprentissage, comme la validation croisée (folds cross-validation) ou une autre division pour l'apprentissage et le test.

3.6 L'utilisation de l'algorithme de DM

Enfin, l'implémentation de l'algorithme de DM est atteinte. À ce stade, il se peut que nous devions utiliser l'algorithme plusieurs fois jusqu'à ce qu'un résultat satisfaisant soit obtenu. Par exemple, en tournant les paramètres de contrôle de l'algorithme, tels que le nombre minimum d'instances dans une seule feuille d'un arbre de décision.

3.7 L'évaluation

Dans cette étape, nous évaluons et interprétons les modèles exploités, les règles et la fiabilité par rapport à l'objectif caractérisé dans la première étape. Nous considérons ici les étapes de pré-traitement quant à leur impact sur les résultats de l'algorithme de DM. Par exemple, inclure une caractéristique dans l'étape 4 et répéter la procédure à partir de là. Cette étape se concentre sur la compréhensibilité et l'utilité du modèle induit. Au cours de cette étape, les connaissances identifiées sont également enregistrées en vue d'une utilisation ultérieure. La dernière étape est l'utilisation et les résultats globaux de rétroaction et de l'extraction acquis par le DM.

3.8 L'utilisation des connaissances extraites

Dans cette étape les connaissances sont prêtes à être inclus dans un autre système pour une activité ultérieure. La connaissance devient effective dans le sens où nous pouvons apporter des changements au système et en mesurer les impacts. L'accomplissement de cette étape décide de l'efficacité de l'ensemble du processus de l'ECD. Cette étape comporte de nombreux défis, tels que la perte des conditions de travail. Par exemple, la connaissance a été découverte à partir d'une certaine représentation statique, il s'agit généralement d'un ensemble de données, mais maintenant les données deviennent dynamiques. Les structures de données peuvent être changées, certaines quantités qui deviennent indisponibles, et le domaine de données peut être modifié, comme un attribut qui peut avoir une valeur qui n'était pas attendue auparavant.

4 Machine learning

4.1 Définition

ML, inventé par Arthur Samuel (Samuel, 1959), est une technologie d'intelligence artificielle permettant aux ordinateurs d'apprendre à partir de données sans avoir été programmés explicitement à cet effet. Il s'agit d'une science moderne permettant de découvrir des modèles et d'effectuer des prédictions à partir de données en se basant sur des statistiques et les analyses prédictives. Autrement dit, ML c'est l'application des méthodes de DM par la machine, ça c'est la définition historique. En fait, le ML consiste à développer un modèle mathématique à partir de données expérimentales (Figure 1.3), pour ça, il existe trois techniques: ML supervisé, ML non supervisé et ML par renforcement. Dans notre thèse on va se concentrer sur le ML supervisé et ML non supervisé. La première catégorie, autrement dit ML supervisé comprend les tâches de classification et de prédiction. La deuxième catégorie, autrement dit ML non supervisé est attribuée aux tâches de regroupement et d'extraction de règles d'association.

4.2 Machine learning supervisé

Soit les données suivantes écrites sous forme matricielle:

$$X = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{pmatrix} \quad et \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

Etant donnée un ensemble d'entraînement $\mathbb{E}$, on cherche à déterminer $f : \mathbb{X} \to \mathbb{Y}$ une fonction modélisant la relation entre les X décrits (vecteur d'attributs) dans l'espace de représentation

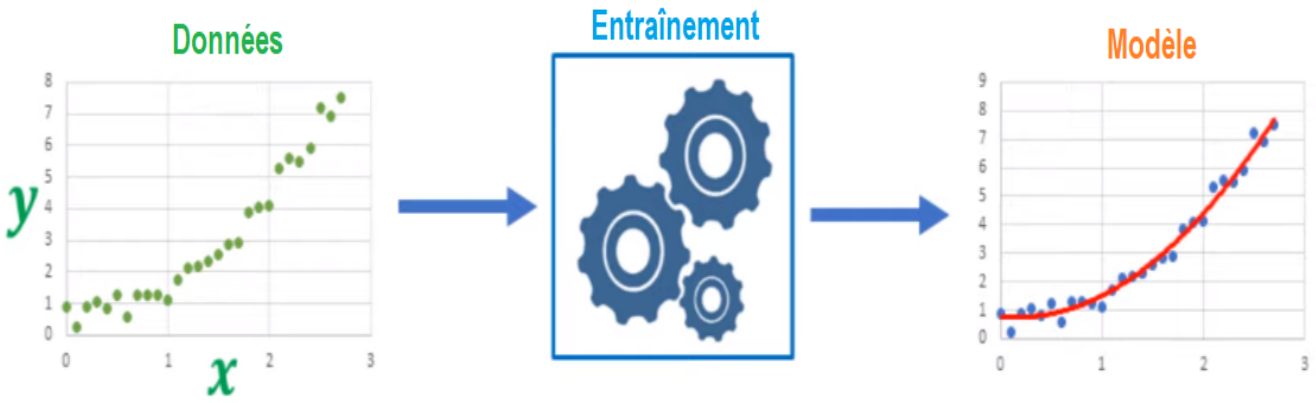

FIGURE 1.3: Extraction du modèle.

$\mathbb{X}$ et la variable cible Y :

$$f(X) = Y \tag{1.1}$$

Par contre, la vraie nature de la relation entre X et Y et les données observées en $\{X^1, \ldots, X^p\}$ étant soit incomplètes, soit bruitées, c'est pourquoi une erreur (ϵ) souvent s'ajoute à la fonction, alors la fonction est de la forme suivante :

$$f(X) = Y + \epsilon \tag{1.2}$$

La fonction finale doit approximer f en commettant le moins d'erreurs possibles sur $\mathbb{E}$ tout en faisant de bonnes prédictions pour des valeurs de $\mathbb{X}$ non encore observées.

Dans le ML supervisé, la machine reçoit de données qui sont caractérisées par des variables X et annotées d'une variable Y. Dans le vocabulaire de ML, on appelle les variables X des caractéristiques ou attributs (Features en anglais), la variable Y quant à elle, est nommée la classe (Label en anglais) ce qui veut dire étiquette. Dans ML, le but c'est que la machine apprenne à prédire la valeur Y en fonction des attributs X qu'on lui donne. Autrement dit, chercher la relation existante entre les attributs et la classe. Pour réussir cette démarche, on commence par donner tout plein de données à la machine, on constitue alors un jeu de données (Dataset en anglais). Ensuite, on spécifie le type de modèle que la machine doit apprendre, est ce que il s'agit d'un modèle linéaire, d'un modèle polynomiale, d'un réseau de neurones ou d'un arbre de décision etc. Il existe tout plein de modèles. On spécifie également les hyper-paramètres pour notre modèle, par exemple, combien de branches il doit y avoir dans un arbre de décision, combien de neurones il doit y avoir dans le réseau de neurones. A cette étape, la machine va commencer à travailler sur les données en utilisant un algorithme d'optimisation pour trouver qu'ils sont les paramètres du modèle qui nous donne les meilleures performances pour notre jeu de données. Cette étape s'appelle la phase d'entraînement ou d'apprentissage (training). Après cette phase, notre modèle de ML est prêt à être utilisé. Lorsque la machine reçoit de

nouvelles données sans étiquettes cet fois-ci, elle va utiliser le modèle pour prédire quelle est la valeur de y (Voir exemple, figure). La figure 1.4 illustre le processus de construction et de l'utilisation de ML supervisé.

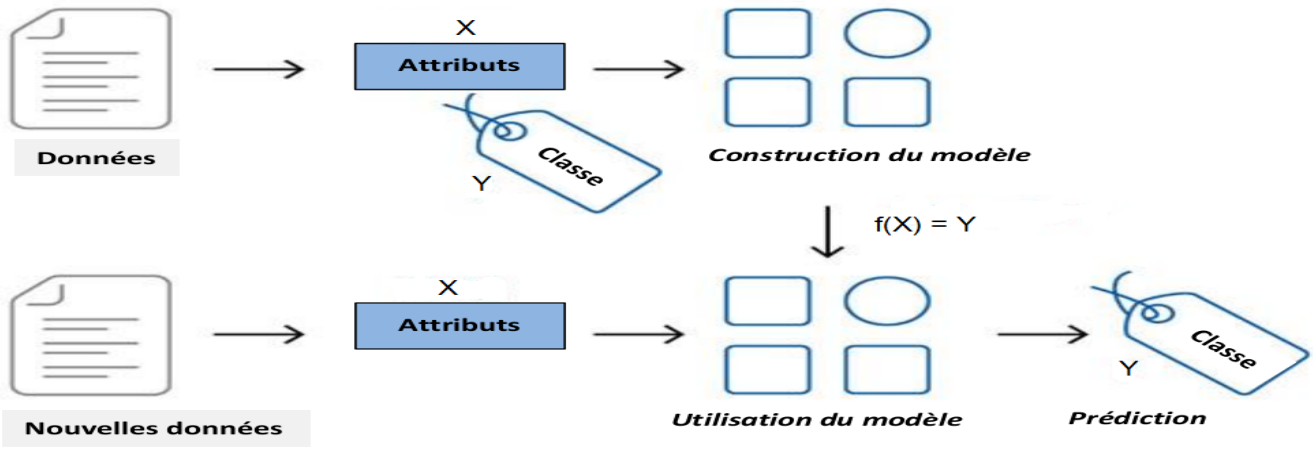

FIGURE 1.4: Processus de ML supervisé.

Pour tester un modèle de ML, on divise le jeu de données en deux parties, une partie pour l'entraînement du modèle et l'autre pour tester le modèle, le pourcentage de partitionnement se diffère selon la quantité de données et le domaine d'étude. Souvent on utilise 70% ou 80% pour l'entraînement du modèle et 30% ou 20% pour tester le modèle. La figure 1.5 présente le partitionnement de données. La validation croisée (Cross-Validation) est une autre méth-

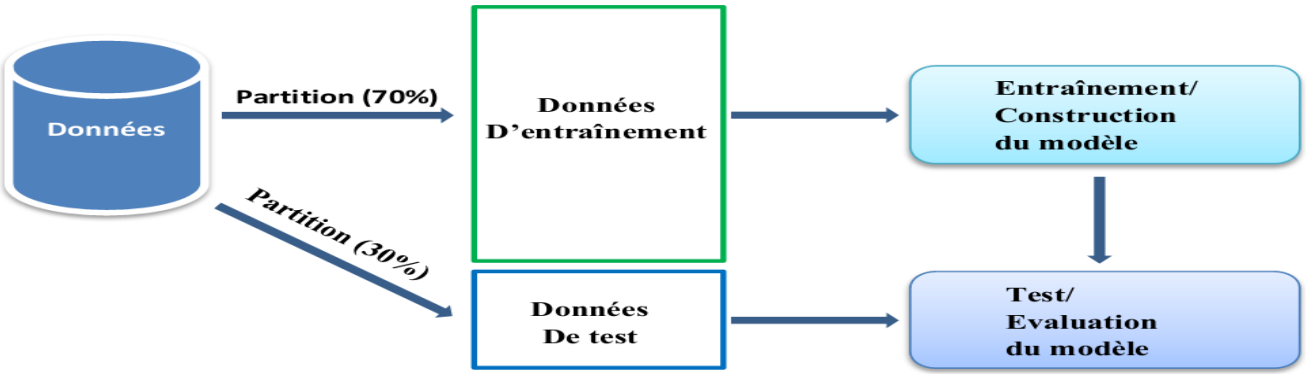

FIGURE 1.5: Partitionnement de données.

ode plus efficace pour évaluer la fiabilité d'un modèle d'apprentissage. Dans cette méthode, les données sont divisées de manière aléatoire en k sous-ensembles mutuellement exclusifs, chacun ayant la même taille. Parmi ceux-ci, l'un est conservé pour les tests et les autres sont utilisés pour entraîner le modèle. La figure 1.6 illustre le déroulement de cette méthode.

Ce type de ML est utilisé pour réponde à deux questions: classification "quelle classe" et régression "combien".

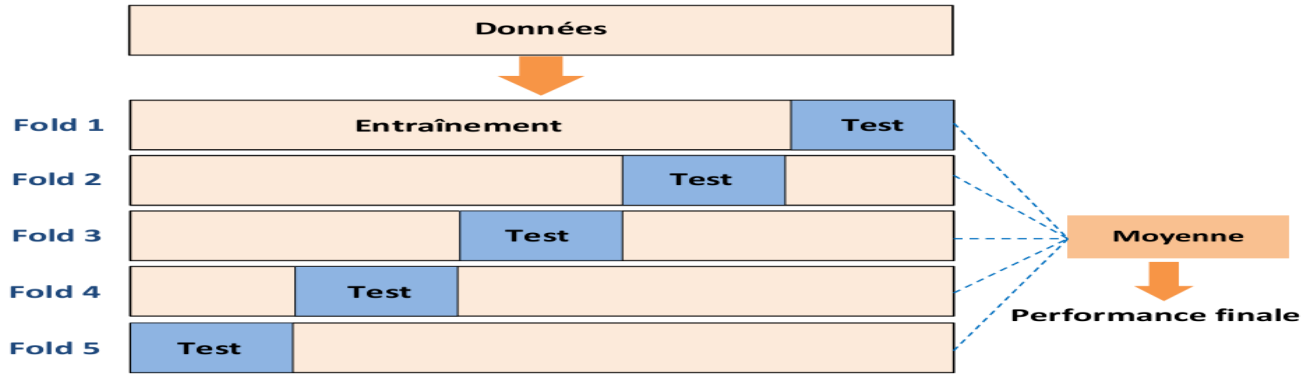

FIGURE 1.6: Partitionnement de la méthode de validation croisée.

4.3 Machine learning non supervisé

Soit les données suivantes écrites sous forme matricielle:

$$X = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{pmatrix}$$

ML non supervisé est la deuxième branche très connu de ML, il est utilisé lorsque les informations utilisées pour la l'apprentissage ne sont ni classifiées ni étiquetées. L'apprentissage non supervisé étudie comment les systèmes peuvent déduire une fonction pour décrire une structure cachée à partir de données non étiquetées. Le but de ML pouvant être formalisée comme la recherche d'un groupe $\bigcup_{k=1}^{K} C_k$ des n observations x_i , i = 1, . . ., n. Ce groupe doit être pertinent au vu d'un ou plusieurs critères à préciser. ML non supervisé peut être illustré par le schéma suivant (Figure 1.7) :

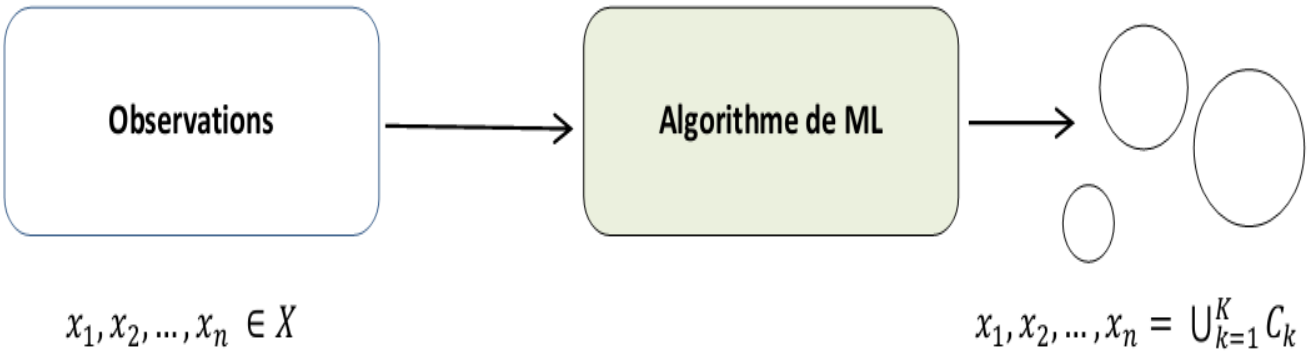

FIGURE 1.7: Partitionnement de données ou clustering.

4.4 Mesures de performance des modèles de ML

Il existe plusieurs façons d'évaluer les performances des modèles. La matrice de confusion permet de conserver les résultats de classification corrects et incorrects pour mesurer la qualité des modèles. Ces éléments sont organisés dans le tableau 1.1.

TABLEAU 1.1: Matrice de confusion

	Prédit comme positif	Prédit comme négatif
Positif actuel	Vrai positif (TP)	Faux négatif (FN)
Négatif actuel	Faux positif (FP)	Vrai négatif (TN)

- Vrai positif ou true positive (TP): le nombre d'élément positifs correctement classés.
- Faux positif ou false positive (FP): le nombre d'élément positifs mal classés.
- Faux négatif ou false negative (FN): le nombre d'élément négatifs mal classés.
- Vrai négatif ou true negative (TN): le nombre d'élément négatifs correctement classés.

Accuracy

La mesure empirique la plus courante pour évaluer l'exactitude (Accuracy) des modèles est donnée par l'équation 1.3. Elle représente le rapport entre les cas correctement classés et le nombre total des cas.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.3}$$

La précision

La précision représente les instances positives correctement classées par rapport au nombre total d'instances positives prédites. Elle est donnée par l'équation 1.4.

$$Précision = \frac{TP}{TP + FP} \tag{1.4}$$

La spécificité

Elle représente les instances négatives correctement classées par rapport au nombre total d'instances négatives. Elle est donnée par l'équation 1.5.

$$Spécificité = \frac{TN}{TN + FP} \tag{1.5}$$

Sensibilité

Elle représente les instances positives correctement classées par rapport au nombre total d'instances positives. Elle est donnée par l'équation 1.6.

$$Sensibilité(Rappel) = \frac{TP}{TP + FN} \tag{1.6}$$

F-measure

F-measure est une mesure de la précision d'un test. Elle prend en compte à la fois la précision et le rappel à calculer. Elle se calcule par l'équation 1.7.

$$F - measure = \frac{2 * Précision * Rappel}{Précision + Rappel} \tag{1.7}$$

4.5 Tâches de machine learning

4.5.1 La classification

La classification consiste à examiner les caractéristiques d'un objet et lui attribuer une classe prédéterminée. Le rôle de la classe est joué par un attribut sélectionné dans l'ensemble de données déterminant un objet. En statistique, l'attribut est appelé la variable dépendante. Tout en classifiant les objets, l'algorithme créé un modèle de classification, qui peut être adapté à de nouvelles données. Par exemple, le modèle de diagnostic du cancer du sein développé au moyen de l'algorithme de classification, peut continuer à être appliqué au système d'aide à la décision aux fins du diagnostic d'un patient dont les données n'ont pas été utilisées pour créer le modèle de prédiction. La classification est un processus en deux étapes comprenant la phase d'apprentissage et la phase de test. Au cours de la phase d'apprentissage, l'algorithme analyse les données destinées à l'apprentissage et crée un modèle de classification. Lors de la phase de test, la précision du modèle est vérifiée à l'aide d'un autre jeu de données. Les méthodes de classification les plus populaires sont DT, BN, RNA, LG, RF , SVM etc.

4.5.2 La prédiction

En termes d'intelligence artificielle, la prédiction est le processus consistant à prendre de données du passé et à utiliser les méthodes de ML pour entraîner un ordinateur à prédire avec précision les événements futurs en fonction de ces données. Cela consiste à estimer une valeur future. Elle est devenue très populaire ces dernières années et peut être plus précise que les méthodes de prévision actuellement utilisées. Dans ce cas-là, un modèle sera construit qui prédit une fonction à valeur continue ou une valeur ordonnée. L'analyse de régression est une méthodologie statistique qui est le plus souvent utilisée pour la prédiction numérique.

4.5.3 L'estimation

Alors que la classification porte sur des résultats discrets tels que oui ou non, prêt au logement ou financement d'un véhicule, l'estimation traite en permanence des résultats valorisés. Si certaines données d'entrées sont disponibles, l'estimation peut être utilisée pour trouver une variable continue inconnue comme le revenu ou la hauteur. Dans l'estimation, on veut trouver une valeur probable ou un intervalle de valeurs probables pour les paramètres inconnus d'un système. La classification et l'estimation sont souvent utilisées ensemble, comme lorsque l'exploration de données est utilisée ou pour prédire qui est susceptible de répondre à une offre de transfert de solde de carte de crédit et également pour estimer la taille du solde à transférer.

4.5.4 Le regroupement (Clustering)

Le clustering est défini comme une technique d'apprentissage non supervisée qui pourrait être le processus d'organisation des objets en groupes (cluster) dont les membres sont similaires d'une certaine manière. Le clustering est crucial car il détermine le regroupement intrinsèque parmi les données actuelles non étiquetées. Ils font certaines hypothèses sur les points de données pour constituer leur similarité. Chaque hypothèse construira des regroupements différents mais tout aussi valables. En appliquant des techniques heuristiques, l'algorithme de clustering classe les objets dans un nombre de groupes par défaut selon la similarité de données. La mesure de la similarité peut être sélectionnée en fonction des attributs caractérisant l'objet. Afin d'évaluer la similarité des objets, des mesures de distance sont souvent utilisées : Euclidienne, Manhattan, Jacquard, etc. Les méthodes de clustering les plus courantes sont : regroupement hiérarchique et partiel.

4.5.5 Les règles d'association

Comme son nom l'indique, les règles d'association (AR) est de simple énoncé "si/alors" qui aide à découvrir des relations entre des bases de données relationnelles apparemment indépendantes ou d'autres entrepôts de données. Il a été proposé par Piatetsky-Shapiro en 1991 (Piatetsky-Shapiro, 1991). Parfois appelée méthode d'analyse du panier de la ménagère, il permet de trouver des modèles non triviaux à partir de données. AR défini les relations entre les éléments de données. Un exemple typique : lors de l'analyse de panier du marché, un modèle a été établi montrant que les acheteurs qui ont acheté du pain et le beurre ont également acheté du lait. L'algorithme Apriori le plus courant fournit deux paramètres d'entrées : Support et Confiance. La confiance dans les règles d'association est la proportion de l'ensemble de données à laquelle cette règle s'applique. Par exemple, une règle de confiance de 80 % signifierait que 80 % des clients qui ont acheté du pain et du beurre ont également acheté du lait. Le support dans les règles d'association est la proportion de l'ensemble de données qui constitue la

condition de la règle. Par exemple, un support de 20 % de la règle signifierait qu'un total de 20 % des clients a acheté du pain et du beurre.

4.5.6 Visualisation

La visualisation de données est la représentation graphique de l'information et de données. En utilisant des éléments visuels tels que les diagrammes, les graphiques et les cartes, les outils de visualisation de données offrent un moyen accessible de voir et de comprendre les tendances, les valeurs aberrantes et les modèles dans les données. Dans le monde du big data, les outils et technologies de visualisation de données sont essentiels pour analyser des quantités massives d'informations et prendre des décisions fondées sur les données. Lorsque nous voyons un graphique, nous percevons rapidement les tendances et les valeurs aberrantes.

Avec l'ère du big data, la visualisation est devenue de plus en plus importante pour l'interprétation et la compréhension de grandes quantités d'informations. Les entreprises utilisent couramment les données pour prendre des décisions stratégiques plus éclairées. Les spécialistes de la visualisation de données sont essentiels pour garantir que ces données sont présentées de la manière la plus efficace possible afin de permettre aux parties prenantes de prendre des bonnes décisions.

5 Machine learning et domaine de la santé : applications et défis

Le ML est l'un des domaines de recherche les plus motivants, qui devient de plus en plus populaire. Le ML joue un rôle important dans l'extraction de nouvelles tendances pour les organisations des soins de santé qui, à leur tour, sont utiles pour toutes les parties associées à ce domaine.

Les progrès technologiques de stockage, de puissance informatique et de numérisation progressive de données des soins de santé dans ces dix dernières années, ont conduit à une nouvelle ère d'analyse et de modélisation. Ce qui se faisait autrefois manuellement et était limité par la capacité humaine peut maintenant être assisté par des ordinateurs à une échelle qu'on aurait difficilement pu imaginer dans le passé. Compte tenu de l'énorme quantité de données disponibles sur les soins de santé, la question urgente est de savoir comment concevoir des outils plus intelligents pour explorer, analyser, utiliser ces données. D'une manière générale, le domaine de recherche est nommé l'informatique de santé, qui est une discipline combinant les sciences de l'information et l'informatique dans le domaine des soins de santé dont l'objectif est l'amélioration de la qualité des soins (Herland, Khoshgoftaar, and Wald, 2014a). Selon l'étude de Herland et al., l'informatique de santé peut être classée en cinq sous-domaines, à savoir la bio-informatique, la neuro-informatique, l'informatique clinique, l'informatique de

santé publique et la bio-informatique transitionnelle, basée sur le niveau d'existence humaine sur lequel ils se concentrent. Dans ces sous-domaines, l'exploration de données est un outil essentiel pour l'analyse du big data. En fait, un nombre croissant d'applications a été développé pour améliorer les services de santé au fil des ans (Dua, Acharya, and Dua, 2014).

Dans cette thèse, nous ciblons les problèmes de classification dans les applications des soins de santé, en particulier les problèmes de prédiction de santé basés sur la classification. Nous passerons d'abord en revue les approches traditionnelles de l'analyse de données. Ensuite, nous passerons en revue sur les études d'analyses de données des soins de santé, en particulier en ce qui concerne la classification. Nous examinerons les travaux existants sur la prédiction des risques basée sur la classification. Enfin, nous examinerons les travaux existants et les approches basées sur les technologies big data. Cette revue souligne également les applications, les défis et les enjeux futurs de ML dans le domaine de la santé.

5.1 Vue historique

La collecte d'informations médicales et leur systématique d'analyse statistique a été réalisée depuis le moyen âge. La première publication médicale connue analysant les statistiques médicales a été publiée en 1662 à Londres (Graunt, 1939). En 1863, F. Nightingale (Nightingale, 1863), pionnier des soins médicaux modernes, avait s'est plainte dans ses notes de l'absence de dossiers médicales et de leur stockage décousue dans les hôpitaux, qui entravait auparavant l'analyse du coût et l'efficacité des traitements. En 1977, le Congrès des États-Unis a publié une étude "Implications politiques des systèmes d'information médicale" (Office of Technology Assesment. Congrès des États-Unis., 1977). L'étude proposait que les systèmes d'information médicale pourraient être utilisés à des fins éducatives pour aider les professionnels de soins de santé, dans la prestation de services de santé et d'accroître l'efficacité des traitements et l'activité des organisations du fournisseur de soins de santé. Les auteurs de l'étude ont déclaré que ces systèmes fourniraient des informations et des connaissances qui étaient auparavant inaccessibles aux chercheurs et aux dirigeantes du secteur de santé. Depuis l'an 2000, des systèmes de dossiers médicales électroniques, qui visent à collecter tous les dossiers médicales pertinents des patients ont été activement déployés dans le monde entier.

5.2 Approches traditionnelles

Le ML est un pratique qui permet d'extraire des informations utiles et intéressantes à partir du big data. Vu la quantité croissante de données sur les soins de santé stockées numériquement au fil des ans, de plus en plus les chercheurs de la communauté de ML se sont consacrés au développement d'applications visant à améliorer les services de santé. Selon la classification d'Esfandiary et al. dans leur étude exhaustive (Esfandiari et al., 2014), les applications de ML en médecine peuvent être considérées en termes de six activités clés, à savoir, le dépistage, le

diagnostic, le traitement, le pronostic, le suivi et la gestion. Dans cette section, nous allons se concentrer nos discussions sur la classification.

Dans la littérature de ML appliqué à la santé, l'accent est mis sur la prévision des maladies spécifiques. Le cancer a été la maladie la plus étudiée de toutes (Wei, Wang, and Yuan, 2009; Zemmal et al., 2016; Pan, Wu, and Zhu, 2015; Esfandiari et al., 2014). Des études sur d'autres maladies ont été également signalées, comme par exemple le diabète (Neuvirth et al., 2011; Simon et al., 2013), insuffisance cardiaque (Qian et al., 2015), problèmes de santé mentale comme la dépression ou le suicide (Tran et al., 2013; Tran et al., 2015), ou maladies particulières (Nguyen, Valizadegan, and Hauskrecht, 2014; Mohktar et al., 2015). Une autre souche se concentre sur la prédiction de la mortalité/survie, par exemple, sur des patients de l'unité des soins intensifs ou les patients atteints du cancer (Kim and Shin, 2013). Selon la revue systématique de la littérature la plus récente et la plus solide, quatre principaux domaines d'application de ML dans la médecine peuvent être définies :

1. Augmenter l'efficacité et l'élimination du facteur humain : traite des tâches de diagnostic de certaines maladies pour lesquelles la précision est essentielle.

2. Réduction du temps et des coûts : applicable lorsque les méthodes de diagnostic conventionnelles prennent beaucoup de temps ou sont très coûteuses.

3. Système d'aide à la décision médicale : utilise une automatisation multi-processus, par exemple des modèles de prédiction et des systèmes experts, appliqués comme aide pour le personnel médical moins expérimenté ou moins qualifié.

4. Extraction de connaissances : utilisé pour l'extraction de nouvelles connaissances ou des hypothèses.

Les études d'application de ML visant à améliorer l'efficacité, l'extraction de connaissances cachées et l'aide à la décision médicale sont tout aussi populaires.

Récemment, la tendance dans le domaine des soins de santé s'est éloignée des soins à la prévention en raison de l'augmentation rapide de données. Les chercheurs se sont attachés à améliorer la fiabilité et l'efficacité des systèmes des soins de santé afin de minimiser le coût de la guérison dans les soins de santé et de fournir de meilleurs médicaments aux patients. Les hôpitaux et les systèmes des soins de santé sont de bons entrepôts du big data comme les antécédents des patients, les rapports de tests, les images médicales (Hermon and Williams, 2014). En fait, il est difficile de faire face à une grande quantité de données structurées et non structurées et de valeurs manquantes. L'amélioration des méthodes existantes de ML peut aider au développement des soins plus personnalisés qui peuvent guérir et prévenir les maladies (Raghupathi and Raghupathi, 2014). Les algorithmes traditionnels de ML existants fonctionnent sur des bases de données centralisées, ce qui nécessite beaucoup de temps pour entraîner et analyser une grande quantité de données. De même, il n'est pas possible de stocker et de

traiter de grandes quantités de données sur une seule machine. Par conséquent, il est nécessaire de paralléliser les approches existantes et de les modifier avec des approches hybrides qui ont suffisamment de capacités pour surmonter le défi du stockage et du traitement du big data dans un environnement distribué (Suthaharan, 2014).

5.3 Approches basées sur les technologies big data

Vu le volume et la variété de données en constante augmentation, les approches et les outils traditionnels de traitement de données sont devenus inadaptés au contexte du big data. Cela a poussé à la création des outils de traitement spécifiques bien adaptés aux nouveaux besoins. Cependant, il est souvent difficile de choisir la solution adéquate car une liste plus vaste des outils disponibles est en constante évolution.

Big data dans le domaine des soins de santé impliquent des bases de données numériques liées à la santé. Le traitement et la gestion de ces données est compliquée (Garapati and Garapati, 2018). Big data en santé offrent de nombreux avantages, tels que la détection et la prévision des maladies (Chen et al., 2017), la réduction des erreurs médicales, les bons soins au bon moment et de meilleurs résultats. Big data a un grand potentiel pour améliorer la médecine et guider les cliniciens dans la prestation des soins fondés sur la valeur. Les données relatives à la santé devraient être multipliées par 50 d'ici 2030, notamment grâce aux développements de la génomique, aux équipements médicaux connectés (IoT), à la numérisation des dossiers des patients et à l'utilisation des applications mobiles et les capteurs. Des tentatives de recherche plus récentes préconisent divers frameworks de soins de santé pour traiter un grand volume de données diversifiées provenant de différentes sources afin de dégager des modèles et des tendances significatifs. Cette section passe en revue ces frameworks de big data et souligne leur contribution dans le domaine des soins de santé.

Dans le but de faire des prévisions de pronostic et de diagnostic pour les patients : dans (Thommandram et al., 2013), les auteurs ont développé un système qui détecte si un patient est en période de crise cardiorespiratoire. Dans (Nair, Shetty, and Shetty, 2018) les chercheurs ont proposé un système en se basant sur le framework Spark, ce système exploite les attributs de la santé dans les tweets des utilisateurs. Ensuite, le système effectue des traitements pour prédire leur état de santé. Un framework appliqué pour le système de soins de santé utilisant l'analyse du big data a été proposé par Raghupathi et Raghupathi (Raghupathi and Raghupathi, 2014). Le framework se compose de plusieurs couches telles que la couche source de données, la couche de transformation, la couche de la plateforme big data et la couche analytique. La couche de source de données se concentre principalement sur les sources de données internes et externes de soins de santé qui se trouvent à plusieurs endroits sous différents formats. La couche de transformation est responsable des opérations telles que l'extraction, la transformation et le chargement de données dans la plateforme big data par le biais de diverses techniques

de mise en scène de données telles que les intergiciels et les opérations d'entrepôt de données. La couche de la plateforme big data comprend divers outils de l'écosystème Hadoop pour effectuer des opérations spécifiques sur le système de fichiers distribués de Hadoop (HDFS) en utilisant le modèle de programmation MapReduce. La couche analytique effectue des opérations telles que des requêtes, des rapports, des traitements analytiques en ligne et des techniques d'exploration de données.

Kim et al. (Kim et al., 2014) ont tenté de mettre en place un autre framework d'analyse du big data qui utilise un système de soins de santé omniprésent. Ce framework analyse les signes vitaux extraits des accéléromètres pour fournir des services de soins de santé. Les signes vitaux sont de données chronologiques continues qui sont de nature non structurée et qui ne peuvent pas être stockées dans des bases de données traditionnelles. Les signaux de l'électrocardiogramme (ECG), la respiration et les données de mouvement ont été pris en compte dans les signes vitaux. Le framework a utilisé une plateforme standard open source pour soutenir l'interopérabilité entre les données et les différents dispositifs médicaux. La plateforme Hadoop a été étendue par l'ajout des algorithmes permettant d'extraire les attributs significatifs à partir de données brutes de signes vitaux et de les stocker pour une analyse en temps réel.

De nombreux systèmes de prédiction et de recommandation ont été étudiés dans le domaine de soins de santé. Dans (Masethe and Masethe, 2014a), une expérience a été réalisée pour la prédiction des crises cardiaques. Une comparaison a été menée afin de trouver la meilleure méthode de prédiction. La classification du cancer du sein est effectuée en utilisant un modèle de réseau de neurone génétiquement optimisé (Bhardwaj and Tiwari, 2015a). D'autres techniques d'extraction de données et de recherche d'informations ont été proposées dans (Tomar and Agarwal, 2013a; Herland, Khoshgoftaar, and Wald, 2014b). Un nouveau système de classification basé sur les méthodes ReliefF et Rough Set (RFRS) a été proposé par Liu et al. (Liu et al., 2017) pour le diagnostic des maladies cardiaques. La principale nouveauté de cet article réside dans l'approche proposée : la combinaison des méthodes ReliefF et RS pour classer les problèmes de maladies cardiaques dans un système efficace et rapide. Le système de classification RFRS se compose de deux sous-systèmes : le sous-système de sélection des caractéristiques RFRS et le sous-système de classification. Sang Hun Han et al. (Han et al., 2017b) ont conçu un framework permettant de collecter et de stocker différents domaines sur les causes des maladies cardiovasculaires, et ont construit une grande base de données intégrée. Le framework était composé de la couche de données, de vitesse, d'analyse et de service, toutes stockées sur des dispositifs de stockage distribués. Enfin, ils ont proposé un framework pour un système de prévision des maladies cardiovasculaires basée sur l'architecture lambda pour résoudre les problèmes liés à l'analyse en temps réel du big data.

Un système évolutif de prévision de l'état de santé en temps réel a été construit autour de framework open source de traitement big data, Apache Spark, testé et déployé sur le cloud,

dans lequel le modèle de ML est appliqué sur le le flux de données en continu (Nair, Shetty, and Shetty, 2018). Le système reçoit les données pertinentes liées à la santé de l'utilisateur par le biais de flux de tweet et prédit l'état de santé en appliquant le modèle d'arbre de décision sur les données et envoie un message direct à l'utilisateur concernant son état de santé. Ainsi, avec un seul tweet, un utilisateur reçoit des informations sur son état de santé de manière instantanée et privée, sur la base desquelles il peut décider si de faire appel ou non à un expert de soins de santé. Le travail offre aux utilisateurs une surveillance de la santé à distance en temps réel et sans frais supplémentaires. Les auteurs dans (Ta, Liu, and Nkabinde, 2016) ont proposé une architecture générique pour l'analyse de données médicales en utilisant des outils big data, notamment Hadoop, Apache Storm, Kafka et NoSQL Cassandra. La combinaison d'une messagerie à haut débit de type publication-abonnement pour les flux, d'un système de calcul distribué en temps réel et d'un système de stockage distribué permet d'analyser efficacement une énorme quantité de données de santé arrivant à un rythme rapide.

5.4 Défis face au domaine de la santé

Vue le big data en santé, l'industrie des soins de santé doit trouver des méthodes pour analyser correctement ces données générées et stockées dans le monde entier chaque seconde afin de découvrir des associations, de comprendre les modèles et les tendances qui fourniront des possibilités importantes pour le suivi en temps réel des maladies, de prédire les épidémies, d'améliorer les soins, de sauver des vies et avec des coûts moins élevés.

L'extraction, l'intégration et l'analyse de données hétérogènes, énormes et les données complexes de soins de santé capturées à partir de différentes sources électroniques de soins de santé sont un défi majeur. Les défis majeurs du big data peuvent être résumés en dix défis (Mathew and Pillai, 2016) : l'intégration de données, le stockage de données, traitement de données, les techniques de ML, transmission de données, la visualisation de données, l'évolutivité, la sécurité de données, la qualité de données et manque de normalisation. Dans cette partie, nous examinons les principaux défis.

5.4.1 Intégration de données

L'intégration de données fait référence à l'acte de réunir les données provenant de nombreuses sources en un ensemble unifié (Osman, 2019). Au cours du processus d'intégration de données, les données sont nettoyées et transformées pour garantir une analyse précise. Grâce à l'intégration de données, les différentes sources de données font l'objet d'un examen commun pour fournir une décision inestimable. Les données peuvent provenir d'une multitude de sources, ce qui représente un énorme défi pour les prestataires de soins de santé. En voici un exemple : un patient se rend dans trois cliniques de sa région pour obtenir des avis sur une question médicale. Chacune de ces cliniques dispose d'un système de EHR différent qui stocke

les données recueillies dans chaque visite. Après la dernière visite, le patient va chercher une ordonnance à la pharmacie, ce qui présente un quatrième point de données. Toute donnée supplémentaire sur la santé, telle que les enregistrements de la fréquence cardiaque recueillis à partir d'un appareil portable ou les visites virtuelles chez le médecin, créera de nouveaux points de données stockés dans des bases de données distinctes. La connexion de points de données provenant de plusieurs bases de données est lourde, mais la consolidation de données serait extrêmement bénéfique, même dans ce cas simple. C'est là que l'intégration de données entre en jeu.

Tous les organismes de santé utilisent aujourd'hui un système de gestion de données numériques propres aux patients. Les données relatives aux patients doivent être rendues facilement et accessibles non seulement aux médecins des soins dans un hôpital, une clinique ou un cabinet médical, mais aussi aux diagnosticiens, aux pharmaciens et aux autres spécialistes médicales. Toutefois, les professionnels de la santé sont également confrontés à des défis uniques en matière de gestion de données relatives aux soins de santé : manque d'intégration entre les systèmes cliniques et administratifs, manque de talents en matière d'analyse et de partage de données relatives aux patients.

5.4.2 Stockage de données

La domination des bases de données relationnelle diminue lentement avec l'apparition de nouveaux types de bases de données. La base de données relationnelle qui utilise une approche basée sur un schéma strict ne peut pas stocker efficacement de données non structurées et semi-structurées. L'augmentation constante de volume de données entraîne une baisse des performances de la technologie des bases de données relationnelles, car elle ne fournit pas de solution évolutive pour le traitement du big data. Le traitement des bases de données massives nécessite un grand espace de stockage. Deux approches pour y faire face, qui sont la compression et l'échantillonnage.

En utilisant les techniques de compression, le temps nécessaire est généralement plus long mais l'espace utilisé est moindre. En utilisant l'échantillonnage, l'information est perdue. Feldman et al. (Feldman, Schmidt, and Sohler, 2013) dans son travail utilise des bases de données de base qui sont des petites bases qui se rapprochent aux données originales pour un problème donné afin de réduire la complexité du big data avec une approche de fusion et de réduction pour des problèmes tels que le K-means, ACP et le clustering. Le cloud computing peut être utilisé comme une aide puissante pour stocker et traiter du big data sur les soins de santé. L'une des solutions les plus couramment utilisées pour traiter ce problème de données volumineuses est la réduction de la dimension.

5.4.3 Traitement de données

Les dispositifs médicaux portables dotés d'un capteur génèrent en permanence d'énormes quantité de données en temps réel, ce que l'on appelle souvent le flux de données qui se compose de données structurées et non structurées. En raison de la complexité de données, il est difficile de traiter et d'analyser ces données volumineuses pour extraire des informations précieuses qui peuvent être utiles à la prise de décision. Le traitement en temps réel du big data a fait l'objet d'une attention croissante, notamment en raison de l'augmentation du volume et de la complexité de données.

Vu la quantité et la vitesse de données générées par les dispositifs médicales, les systèmes de soins sont alimenté en temps réel par de très nombreuses informations. Le traitement de ces informations en temps réel est la nécessité de l'heure actuelle. Alors, un défi est de concevoir un mécanisme optimal pour la réalisation d'actions en temps réel dans des situations critiques et une analyse s'adaptant aux contraintes imposées par ces flux de données. Les technologies de traitement et de calcul en mémoire peuvent être adoptées par les organisations pour améliorer la vitesse de traitement. De nombreuses organisations tirent parti du traitement en mémoire, en fait qu'il est plus rapide que les solutions déjà disponibles.

La solution fournie par Hadoop est largement utilisée pour faire face aux problèmes liés au big data, mais Hadoop étant un framework de traitement par lots (batch), il ne répond pas aux besoins d'analyse en temps réel. Pour répondre à ces besoins, Nathan Marz a mis au point le paradigme de l'architecture Lambda (LA), qui est une architecture évolutive et tolérante aux pannes pour le traitement de données. L'architecture Lambda consiste en une couche de traitement par lots, une couche de service et une couche de vitesse qui permet de faire des analyses en temps réel pour surmonter la lenteur de la couche de traitement par lots, c'est une approche qui traite à la fois les analyses par lots et en temps réel (Hausenblas and Bijnens, 2015).

En comparaison avec d'autres frameworks de traitement du big data, Spark a la capacité de traiter les données par lots et en temps réel et offre des performances élevées grâce au calcul en mémoire. Il s'agit d'un framework de calcul distribué qui garantit des performances jusqu'à 100 fois plus rapide, ce qui le rend particulièrement adapté au ML et au traitement de données en temps réel. Il offre une plateforme complète et unifiée pour gérer les besoins de traitement du big data avec de données de types différents, ainsi que le traitement de flux de données. Comme Spark, Storm est tolérant aux pannes, trait les calculs distribués et il est capable de gérer des calculs complexes (Chintapalli et al., 2016).

Selon une étude bibliographique approfondie que nous avons également mené, nous avons observé que le framework Hadoop MapReduce restait un choix dominant pour le calcul distribué dans le domaine de l'informatique de santé et probablement dans d'autres disciplines scientifiques par rapport aux nouvelles versions plus orientées vers le traitement de flux de données à savoir Apache Spark et Apache Flink.

5.4.4 Techniques de machine learning

De nombreuses techniques de ML ne sont pas capables d'effectuer du ML distribué. Les données sont en constante évolution et les techniques de ML doivent donc être capables de s'adapter aux nouveaux besoins. Des algorithmes parallèles et des techniques d'implémentation efficaces peuvent améliorer considérablement et répondre aux exigences de performance et d'évolutivité. De nombreuses techniques de ML souffrent de problèmes d'extensibilité, tant au niveau de l'utilisation de la mémoire que de temps de calcul. Pour surmonter ce problème, des algorithmes parallèles basés sur MapReduce itératif ont été proposés pour la classification et bien d'autres. L'utilisation du paradigme MapReduce a permis de réduire considérablement le temps d'apprentissage. Toutefois, pour l'exploration intensive de données, MapReduce dans le cloud computing s'avère plus efficace que MapReduce dans Hadoop (Xu et al., 2014).

D'autre part, les algorithmes d'extraction de AR sont souvent utilisés pour trouver des associations entre les éléments d'une base de données. L'algorithme Apriori utilise une approche itérative dans laquelle les résultats des itérations précédentes sont utilisés pour trouver l'ensemble d'items fréquents pour l'itération en cours. L'inconvénient principal de cette approche est que l'ensemble de candidats deviennent volumineux si la base de données est trop importante et que de multiples scans de données sont nécessaires. Ainsi, pour traiter les problèmes liés au big data, des algorithmes parallèles sont introduits. Par conséquent, afin d'assurer la mise à l'échelle sur plusieurs machines, un framework tolérant aux pannes pour les applications distribuées telles que MapReduce est utilisé. L'implémentation de l'algorithme Apriori en se basant sur le framework Hadoop MapReduce, traite de données massives dans le HDFS. Pour chaque itération, le résultat sera enregistré dans le HDFS, cela nécessite un temps d'entrée/sortie élevé, ce qui réduit les performances. L'implémentation de l'algorithme Apriori sur la plateforme Spark est plus rapide jusqu'à 18 fois par rapport aux différents frameworks (Bhargava, Goyal, and Keswani, 2019). Par contre, les résultats générés sur les données de santé du monde réel sont beaucoup plus rapides par rapport au framework MapReduce.

Les frameworks traditionnels de ML tels que Weka et Rapidminer sont limités, ils ne peuvent pas faire face au traitement du big data, alors que Apache Mahout est un framework open source et distribué pour faire face aux problèmes liés au big data. Mahout est une bibliothèque évolutive de ML utilisée et développée au-dessus de la plateforme Hadoop. Actuellement, Mahout et Spark MLlib sont deux approches pour exécuter les algorithmes de ML distribués. Dans le cas de Mahout, tous les résultats d'itération sont écrits et lus sur le disque, tandis que pour Spark MLlib, toutes les itérations peuvent être stockées en mémoire. Comme le traitement de données peut être effectué directement à partir de la mémoire, les performances sont nettement meilleures avec Spark MLlib qu'avec Mahout (Aziz, Zaidouni, and Bellafkih, 2018).

6 Conclusion

Dans ce chapitre nous avons présenté les éléments de base de processus d'ECD dans le cadre général puis le principe de ML est décrit, par la suite nous nous sommes intéressés par les méthodes appliquées dans le domaine de la santé. Par conséquent, l'analyse de la littérature a montré que le ML est utilisé pour soutenir les objectifs et d'améliorer la qualité du traitement des patients et l'optimisation de la gestion des soins de santé. Diverses méthodes de ML, dont la classification, le regroupement, les règles d'association, la visualisation et l'analyse de liens, ont été utilisées avec succès dans différentes spécialités cliniques. En outre, grâce à la disponibilité des différentes technologies cliniques, plusieurs caractéristiques des maladies ont été recueillies pour la classification. Par conséquent, la sélection de la méthode de ML la plus précise et les attributs les plus pertinents sont devenues une tâche difficile et longue.

Dans le chapitre suivant nous examinons les algorithmes de ML les plus connus dans la littérature et nous proposons une approche combinant la réduction de données médicales et une méthode efficace pour la classification des maladies.

2 Amélioration des performances du diagnostic médical par sélection des attributs pertinents

1 Introduction

Afin d'améliorer la précision de la classification des maladies, qu'elle soit positive ou négative et gérer les caractéristiques de données et de l'information en forte augmentation, les chercheurs se sont tournés vers les technologies d'exploration de données et les approches de ML pour prédire les maladies. Dans la littérature, les techniques de ML sont largement utilisées en se basant sur les attributs de données. Vu le nombre des attributs descriptifs augmente, le temps de calcul augment aussi rapidement. En ML, l'extraction des attributs est la clé de la reconnaissance des modèles et de classification. Le filtrage et l'extraction des attributs qui contient l'information utile à partir du vecteur original avec une haute précision est devenue un défi et une nouvelle question. Le meilleur modèle sera peu performant si les attributs ne sont pas bien choisis. Comme les attributs des maladies peuvent être décrites de manière aussi détaillée que possible, les informations peuvent être redondantes mais sans aucune contribution significative au modèle final, ce qui entraîne un temps de calcul plus long pour les calculs fastidieux. A ce stade, les maladies nécessitent une meilleure précision et un diagnostic automatique et rapide.

Dans ce chapitre nous examinons les algorithmes de ML les plus connus dans la littérature et nous proposons une approche combinant la réduction de la dimension de données médicales et une méthode efficace pour la classification de cancer du sein comme étude de cas.

2 Algorithmes utilisés de machine learning

Un algorithme en DM ou ML est un ensemble d'heuristiques et de calculs qui construit un modèle à partir de données. Pour créer un modèle, l'algorithme analyse d'abord les données que vous fournissez, en recherchant des types de modèles ou de tendances spécifiques.

L'algorithme utilise les résultats de cette analyse au cours de nombreuses itérations pour trouver les paramètres optimaux pour la création du modèle d'exploration. Ces paramètres sont ensuite appliqués à l'ensemble de données afin d'extraire des modèles exploitables et des statistiques détaillées. Les algorithmes de DM sont au cœur du processus d'ECD. Ces algorithmes déterminent comment les cas sont traités et fournissent ainsi les capacités de prise de décision nécessaires pour classer, segmenter, associer et analyser les données pour le traitement. Dans cette partie on s'intéresse aux algorithmes de classification et les règles d'association.

2.1 Algorithmes de classification

Il existe de nombreux algorithmes de classification, nous limiterons dans cette partie aux algorithmes de classification et de la recherche d'associations les plus connus dans la littérature.

2.1.1 Régression logistique

La régression logistique (LG) est l'une des méthodes d'analyse multi-variée les plus courantes pour prédire une réponse binaire. C'est une technique de classification discriminante (Hosmer Jr, Lemeshow, and Sturdivant, 2013) qui fonctionne sur un vecteur d'entrée à valeur réelle. Dans le cas d'un résultat binaire, la LG binomiale est utilisée tandis que la LG multinomiale est utilisée pour prédire un résultat multi-classe. L'idée principale de la régression logistique est d'estimer la probabilité d'une réponse binaire basée sur une ou plusieurs variables indépendantes.

L'algorithme LG utilise également une équation (2.1) linéaire avec des prédicteurs indépendants pour prédire une valeur. La valeur prédite peut se situer n'importe où entre l'infini négatif et l'infini positif. Nous avons besoin que la sortie de l'algorithme soit une variable de classe, c'est-à-dire 0-non, 1-oui. Par conséquent, nous écrasons la sortie de l'équation linéaire dans une plage de [0,1]. Pour écraser la valeur prédite entre 0 et 1, nous utilisons la fonction sigmoïde (2.2).

$$z = \theta_0 + \theta_1.x_1 + \theta_2.x_2 + \theta_3.x_3 + ... + \theta_n.x_n \tag{2.1}$$

x_i : est une variable prédictive qui servira dans le calcul du modèle prédictif.
θ_i : est un poids/paramètre de la fonction hypothèse. Ce sont ces θ_i : qu'on cherche à calculer pour obtenir notre fonction de prédiction.
θ_0 : est une constante nommée le bias (biais).

$$g(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

$$h = g(z) = \frac{1}{1 + e^{-z}} \tag{2.3}$$

Nous prenons la sortie (z) de l'équation linéaire et la donnons à la fonction g(x) qui retourne une valeur écrasée h (2.3), la valeur h se situera dans la plage de 0 à 1. Pour comprendre comment la fonction sigmoïde écrase les valeurs dans la plage, visualisons le graphique de la fonction sigmoïde (Figure 2.1). Comme vous pouvez le voir sur le graphique, la fonction

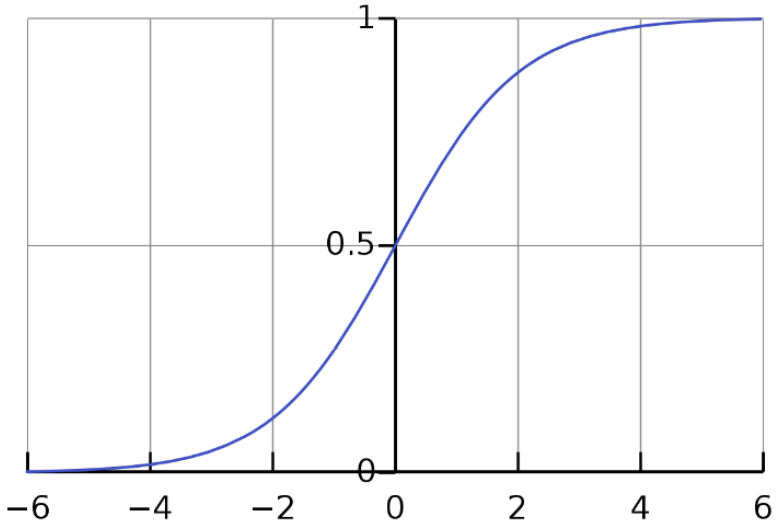

FIGURE 2.1: Graphe de la fonction segmoid.

sigmoïde devient asymptote à y=1 pour les valeurs positives de x et devient asymptote à y=0 pour les valeurs négatives de x.

2.1.2 Arbres de décision

Un arbre de décision (DT) est un modèle de ML qui divise les données en sous-ensembles (Breiman et al., 1984). Le processus de partitionnement commence par une division binaire et se poursuit jusqu'à ce qu'aucune autre division ne puisse être effectuée. Le partitionnement récursif est le processus d'étape par étape avec lequel un DT est construit en divisant ou en ne divisant pas chaque nœud. Chaque partition est sélectionnée en trouvant la meilleure parmi toutes les divisions possibles. La division est basée sur un critère particulier, tel que l'impureté de Gini et l'Entropie. La mesure de l'homogénéité de l'étiquette au niveau du nœud est basée sur l'impureté du nœud. Le représentant le plus populaire de DT est le C4.5 développé par J. Ross Quinlan (Quinlan, 2014). Il peut être utilisé pour des tâches de classification et de régression. Les deux principales entités d'un arbre sont les nœuds de décision, où les données sont divisées et les feuilles où l'on obtient le résultat. Les terminologies importantes de DT sont :

-Nœud racine : il représente l'ensemble de la population ou de l'échantillon et se divise ensuite en deux ou plusieurs ensembles homogènes.

-Séparation : elle s'agit d'un processus de division d'un nœud en deux ou plusieurs sous-nœuds.

-Nœud de décision : lorsqu'un sous-nœud se divise en d'autres sous-nœuds, il est appelé nœud de décision.
-Feuille : les nœuds qui ne se divisent pas sont appelés nœud feuille.
-Branche/Sous-arbre : une sous-section de l'arbre entier est appelée branche ou sous-arbre.
-Nœud parent et enfant : un nœud qui est divisé en sous-nœuds, est appelé nœud parent de sous-nœuds alors que les sous-nœuds sont les enfants d'un nœud parent. L'exemple d'un arbre permettant de prédire la maladie d'une personne, fournissant diverses informations comme la douleur, la fièvre et la toux est donné ci-dessous (Figure 2.2). DT peut être utilisé comme

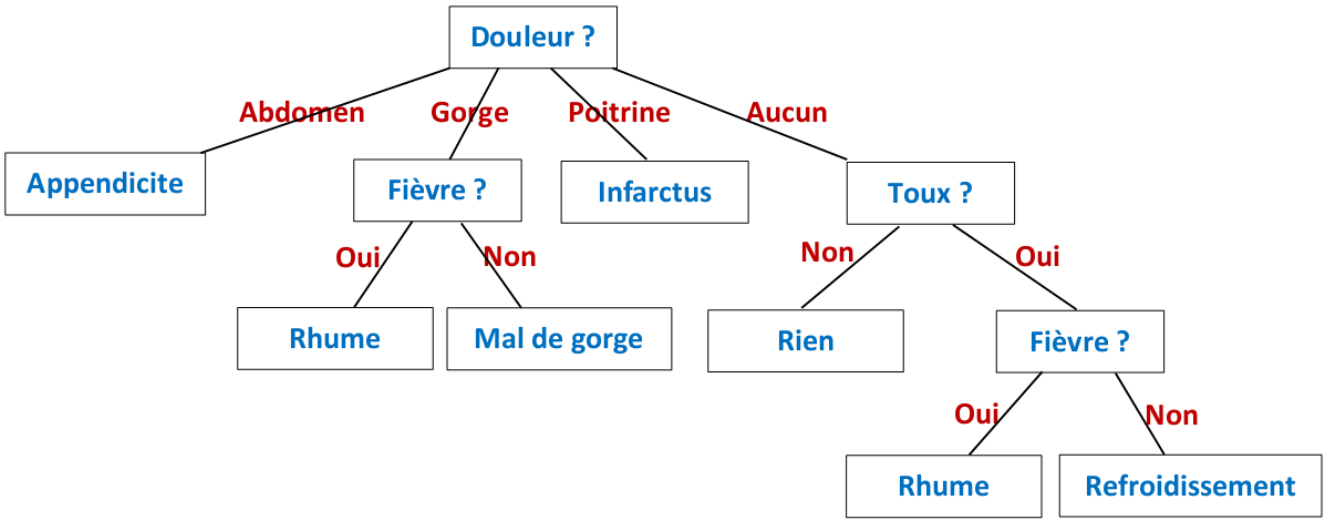

FIGURE 2.2: Exemple d'arbre de décision.

un moyen de pré-traitement pour sélectionner les variables pertinentes qui seront ensuite les entrées d'une autre méthode de classification.

2.1.3 Forêts aléatoires (Random Forest)

Les forêts aléatoires (RF) est une méthode d'apprentissage ensembliste et l'un des algorithmes de ML supervisé les plus performants et les plus puissants, capable d'effectuer à la fois des tâches de classification et de régression. Comme son nom l'indique, RF construit la forêt à partir de plusieurs DT (Breiman, 2001). Chaque arbre donne un vote pour classer un nouvel élément, la classe qui reçoit le plus de votes est choisie par le RF (Figure 2.3).

2.1.4 Réseaux de neurones artificiels

Les réseaux de neurones artificiels (RNA) est un algorithme qui s'inspire de la biologie et plus particulièrement du fonctionnement du cerveau humain. Il s'agit de neurones en activités, ces neurones sont reliés entre eux par des liens de connexion et ont un poids. Ils se multiplient avec le signal transmis dans le réseau. La sortie de chaque neurone est déterminée en utilisant une fonction d'activation telle que sigmoïde et softmax. Habituellement, des fonctions d'activation

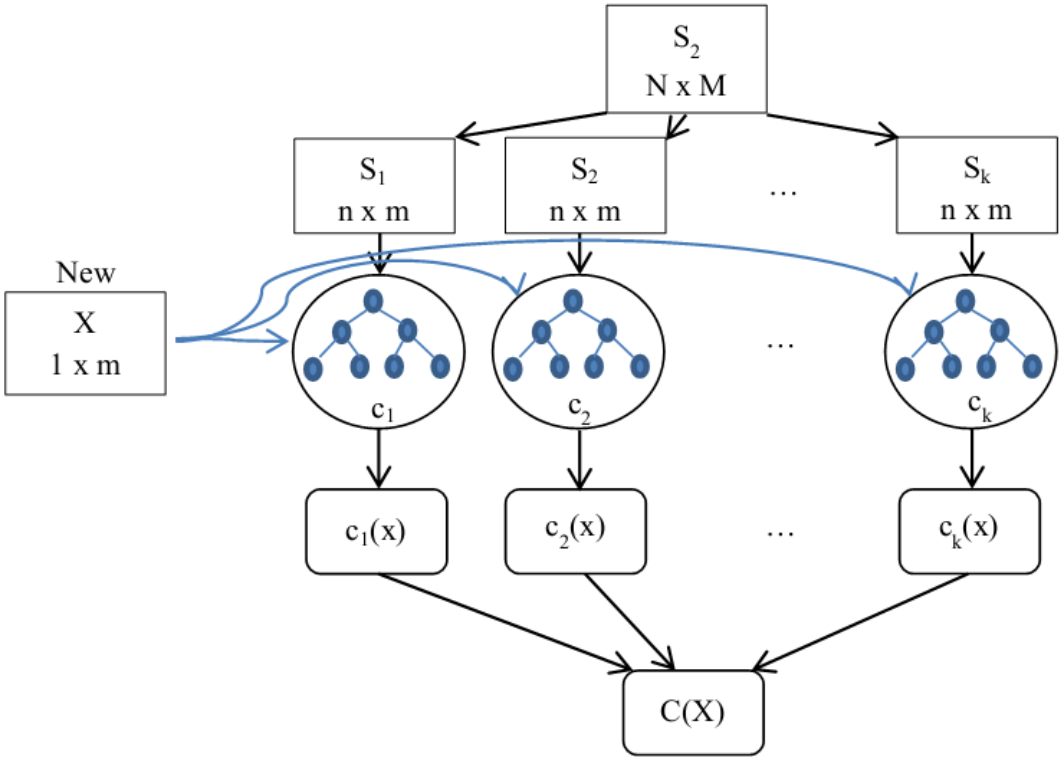

FIGURE 2.3: Construction de RF.

non linéaires sont utilisées $F()$. Le résultat ou la valeur de sortie (notée y) est encore appelée niveau d'activation. Les neurones sont formés par l'expérience, lorsqu'on leur applique une entrée inconnue dans le réseau, ils peuvent généraliser à partir des expériences passées et produire un nouveau résultat ((Haykin, 1994)). La sortie du réseau de neurones est déterminée par l'équation (2.4). Un modèle simple de neurones artificiels est illustré dans la figure 2.4.

$$y = F(\sum_{i=1}^{n} w_i x_i + w_0) = F(\sum_{i=0}^{n} w_i x_i) = F(W^T X) \tag{2.4}$$

où $X = [1, x_1, x_2, ..., x_n]$, T est le vecteur d'entrée augmenté avec $x_0 = 1$
Les poids w_i sont des paramètres adaptatifs, dont la valeur est à déterminer en fonction du problème via un algorithme d'apprentissage.

Il existe de nombreux algorithmes différents utilisés pour entraîner les RNA avec trop de variantes. Un RNA comporte trois partie, appelées couches :

- Couche d'entrée : la couche d'entrée d'un RNA est composée de neurones artificiels d'entrées, elle apporte les données initiales dans le système pour un traitement ultérieur par les couches suivantes. La couche d'entrée transmet les données directement à la première couche cachée, où les données sont multipliées par les poids de la première couche cachée.

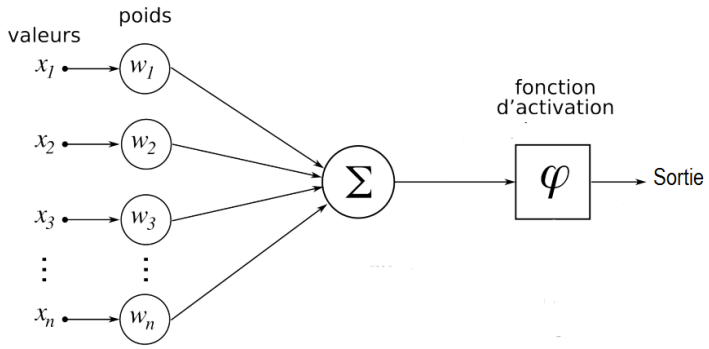

FIGURE 2.4: Modèle de réseaux de neurones artificiels.

- Couche cachée : dans les RNA, la couche cachée est située entre l'entrée et la sortie de l'algorithme, dans laquelle la fonction applique des poids aux entrées et les dirige à travers une fonction d'activation en tant que sortie. En bref, les couches cachées effectuent des transformations non linéaires des entrées dans le réseau. Les couches cachées varient en fonction de la fonction du réseau de neurone, et de même, les couches peuvent varier en fonction des poids qui leur sont associés.

- Couche de sortie : cette couche est la dernière couche du réseau et reçoit les données de la dernière couche cachée. Avec cette couche, nous pouvons obtenir le nombre de valeurs désiré (classification) et dans une plage souhaitée.

On distingue deux architectures principales de RNA, le RNA mono-couche et le RNA mutli-couche. Le RNA mono-couche se compose d'une seule couche d'entrée et une seule couche neuronale, qui est également la couche de sortie. Il est composé de n entrées et de m sorties. Les RNA mutli-couches se composent de plusieurs couches cachées dont le nombre de neurones peuvent être différent, connectées via le biais de liaisons pondérées unidirectionnelles. Les entrées de la couche n sont les sorties de la couche $n-1$. La figure 2.5 illustre un exemple de RNA multi-couche.

2.1.5 Naïve bayes

Il s'agit d'une technique de classification basée sur le théorème de Bayes avec une hypothèse d'indépendance entre l'ensemble de données (Lewis, 1998). En termes simples, un modèle de classification Naïf de Bayes (BN) suppose que la présence d'un attribut particulier dans une classe n'est pas liée à la présence d'un autre attribut. Le modèle BN est facile à construire et particulièrement utile pour traiter du big data. En plus de sa simplicité, le modèle BN est connu pour être plus performant que les méthodes de classification, même les plus sophistiquées. Le théorème de Bayes fournit un moyen de calculer la probabilité à postériori $P(c|x)$ à partir de

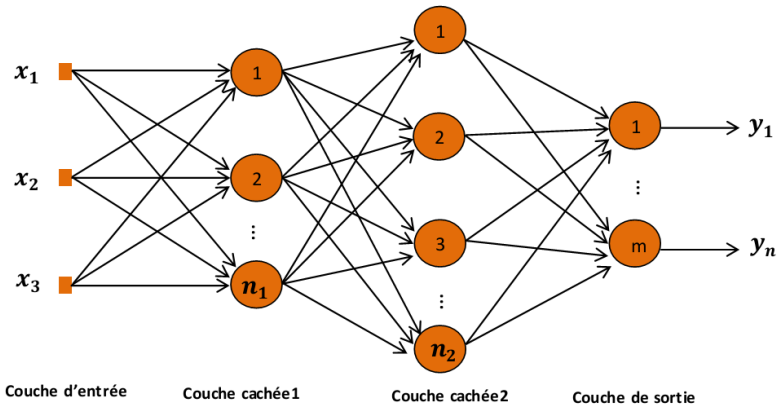

FIGURE 2.5: Réseaux de neurones multi-couche.

$P(c)$, $P(x)$ et $P(x|c)$ selon l'équation suivante (2.5).

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \tag{2.5}$$

$P(c|x)$ est la probabilité à postériori de la classe (c) étant donné l'ensemble de données(x, attributs).
$P(c)$ est la probabilité antérieure de la classe.
$P(x|c)$ est la probabilité du l'ensemble de données étant donné la classe.
$P(x)$ est la probabilité à priori du l'ensemble de données.

2.1.6 Machines à vecteurs de support

Les machines à vecteurs de support (SVM) proposé par Vapnik (Vapnik, 2013) est un algorithme de ML supervisé qui peut être utilisé pour les défis de classification et de régression. Il s'agit d'un modèle linéaire discriminant qui tente de maximiser la marge entre les classes durant l'apprentissage. Cependant, il est surtout utilisé dans les problèmes de classification en raison de son avantage de précision. Les vecteurs de support sont les points situés sur les marges indiqués dans la figure 2.6. L'équation de l'hyperplan de séparation est définie par l'équation linéaire suivante (2.6) :

$$g(x) = w^T x + b \tag{2.6}$$

où x décrit les points de données, y est un vecteur de coefficient, et b (biais) indique le décalage par rapport à l'origine. Dans le cas d'un SVM linéaire, $g(x) \geq 0$ pour le point le plus proche sur celui de la classe, $g(x) \leq 0$ pour le point le plus proche appartient à une autre classe. Le modèle SVM généralisé est révisé pour ce problème pour rechercher le modèle comme suit

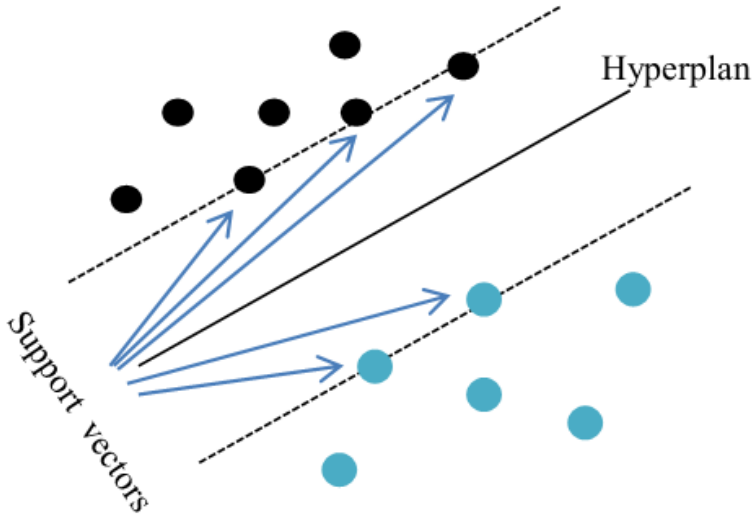

FIGURE 2.6: Hyperplan de séparation avec les vecteurs de support.

(Cortes and Vapnik, 1995) :

$$Maximiser \ [\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(X_i, Y_j)] \tag{2.7}$$

$$Avec \ \sum_{i=1}^{n} \alpha_i y_j = 0, \ \ 0 \leq \forall \alpha_i \leq L \tag{2.8}$$

où X est le vecteur d'apprentissage, Y est l'étiquette associée aux vecteurs d'apprentissage, a est le vecteur du paramètres de l'hyperplan de modèle, $K(X_i, Y_j)$ est la fonction noyau égale à $\{\Phi(X_i), \Phi(Y_i)\}$ où $\Phi(X)$ représente le mappage des vecteurs d'entrées sur l'espace du noyau X pour mesurer la distance entre le vecteur d'apprentissage X_i et Y_j. L est un paramètre de pénalité pour contrôler le nombre d'erreurs de classification. Dans notre étude de cas, la dimension de l'espace des attributs a été réduite et la base de données avec de nouveau attributs a été reconstruite, puis les différentes techniques de ML sont appliquées.

2.2 Les règles d'association

Les règles d'association (Association rules, AR) est une technique qui permet à l'utilisateur de découvrir la corrélation entre différents objets dans une base de données. Une AR a la forme logique générale suivante : **Si** *Conditions* **alors** *Résultats* et présenté sous forme d'antécédent et de conséquence $X_1 \wedge X_2 \ldots \rightarrow X_n$. Une AR est une relation d'implication X à Y entre deux ensembles d'attributs X et Y. Cette règle indique que les transactions qui contiennent les attributs (appelé **Items**) de l'ensemble X ont tendance à contenir les attributs de l'ensemble Y.

AR est utilisé pour trouver les groupes d'article ou attributs (appelés **Itemsets**) fréquents, puis générer des règles significatives qui expliquent la présence de certains attributs en fonction de la présence des autres dans la base de données en satisfaisant certains paramètres comme le seuil minimal de support et la confiance minimale. Elle a été initiée par Agrawal et al. (Agrawal, Imieliński, and Swami, 1993) pour la première fois, afin d'analyser les bases de données transactionnelles. C'est une déclaration de la forme $A \rightarrow B$, où $A, B \subset I$ tel que, $A \neq \emptyset$, $B \neq \emptyset$ et $A \cap B = \emptyset$. L'ensemble A est appelé antécédent de la règle, l'ensemble B est appelé conséquence de la règle et I est un itemset.
Soit $L = \{i_1, i_2, \cdots i_m\}$ un ensemble d'attributs et D une base de données de transaction, une transaction T est un ensemble d'attributs tel que $T \subseteq L$, soit X et Y deux ensembles d'attributs, la règle d'association, $X \Longrightarrow Y$ avec $X \subseteq T$, $Y \subseteq T$, $X \cap Y = \emptyset$ est au moins défini par deux mesures de qualité : **support** et **confiance** définis par les équations 2.9 et 2.10 respectivement. Le support est défini comme la proportion de transaction dans la base de données, qui contient les items A, tandis que la confiance détermine la fréquence à laquelle les éléments de B apparaissent dans la transaction qui contient A.

$$Supp(A \rightarrow B) = Supp(A \cup B) = \frac{|t(A \cup B)|}{t(A)} \tag{2.9}$$

$$Conf(A \rightarrow B) = \frac{Supp(A \cup B)}{Supp(A)} \tag{2.10}$$

Soit D une base de données, I un itemset et S_{min} un support minimal alors I est considéré comme fréquent si et seulement si $Support(I) \geq S_{min}$.

Exemple 2.1

Dans le but d'étudier les achats des clients dans un supermarché, les encaisseurs ont enregistré les achats des clients dans une base de données. Un identifiant est affecté pour chaque client, ainsi que les films qu'il a acheté. Les produits concernés sont données dans le tableau I. Le tableau D (Tableau 2.1) représente l'ensemble des transactions ou chaque transaction est représentée par un ensemble d'attributs ou articles. Par exemple, le Tid =100 de D concerne un client ayant l'identifiant 100 et ayant acheté les produits '1', '3' et '4'.
En se basant sur la base des transactions nous avons:
-**{1}, {2}, {3}, {4}, {5}** sont des items dans la base D relative aux achats.
-La transaction 4 est un 2-itemsets constitué de deux items **{2}, {5}** qui correspondants aux produits produit1 et produit2.
-Supp ({3}) = 4/5 = 0, 8 (80%) vu que l'itemset {3} apparait dans quatre transactions parmi 5 de D.
-Conf (1 $\rightarrow$ 3) =3/3= 1 (100%)
AR vise à découvrir les modèles de co-occurrence des attributs dans une base de données. Par exemple, une AR dans une base de données sur le cancer du sein peut être : dans 47% des

TABLEAU 2.1: Exemple de base de transaction

I	
Item	Titre
1	Produit1
2	Produit2
3	Produit3
4	Produit4
5	Produit5

D	
Tid	Transaction
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5
500	1 3 5

transactions, 99% des patients qui ont la valeur de premier et le 6ème attributs est égale à 1 ont également la valeur du 9ème attribut d'entrée est égale à 1.

2.2.1 Algorithme Apriori

La génération des AR est généralement divisée en deux étapes distinctes. Tout d'abord, un support minimal est appliqué pour trouver tous les itemsets fréquents dans une base de données. Deuxièmement, ces itemsets fréquents et la contrainte de confiance minimale sont utilisées pour former des règles. En exploitant cette propriété, l'algorithme Apriori peut trouver tous les itemsets fréquents.

L'algorithme Apriori est un algorithme de DM conçu en 1994 par Agrawal et al.(Agrawal, Imieliński, and Swami, 1993). Il est utilisé pour reconnaître les propriétés qui se produisent fréquemment dans une base de données et d'en déduire une catégorisation. C'est l'algorithme clé pour l'extraction des AR car il constitue la base des algorithmes majoritaires qui sont conçus pour extraire les AR. Il fonctionne de manière itérative, il génère d'abord des itemsets, il calcule leurs fréquence, il conserve ceux dont la fréquence dépasse un certain seuil (support) et génère des règles à partir de ces itemsets, enfin il conserve ceux dont la fiabilité dépasse un certain seuil (confidence). L'algorithme 1 montre l'algorithme Apriori présenté dans Agrawal et al. (Agrawal, Srikant, et al., 1994).

Soit D une base de données de transactions. Premièrement il faut trouver les itemsets fréquents en utilisant l'algorithme Apriori. Ensuite, AR seront générées en utilisant un support minimal et une confidence minimale. Au cours de cette étape, l'algorithme Apriori recherche dans D le nombre de chaque candidat $C1$ et il compare le support des candidats avec le support minimal pour générer un modèle fréquent $1-itemsets$. L'ensemble des $1-itemsets$ fréquents est constitué des $1-itemsets$ candidats satisfaisant à un support minimal. Générer des candidats $C2$ à partir de $L1$ pour générer des $2-itemsets$ et ainsi de suite. L'algorithme utilise une recherche par niveau, où les $k-itemsets$ sont utilisés pour explorer $(k+1)-itemsets$. Apriori, détermine les itemsets fréquents qui peuvent être utilisés pour déterminer les AR et qui mettent en évidence les tendances générales dans la base de données.

Algorithme 1 Le pseudo-code de l'algorithme Apriori

procedure Apriori (T, minSupport) {
//T is the database
// minSupport is the minimum support
C_k : Candidate itemset of size k
L_k : frequent itemset of size k
L_1= {frequent items};
for (k= 2; L_{k-1} !=∅; k++) {
C_k= candidates_generated from L_{k-1}
for each transaction t in database do{
L_k = candidates in C_k with minSupport
}//end for each
}//end for
return $\cup L_k$;
}

Remarque 2.1

Les sous-itemsets d'un k-itemset fréquent sont obligatoirement fréquents. Autrement dit, un sous ensemble d'articles fréquents doit être un sous ensemble d'articles fréquents. {1,2},{1,3},{2,3} sont des sous-ensembles de l'ensemble {1,2,3}, sachant que l'item {1,2} n'est pas fréquent, alors l'itemset {1,2,3} ne sera pas pris en considération dans L_3.

L'algorithme de génération du candidat (Algorithme 2) est donné comme suit (Karabatak and Ince, 2009).

Algorithme 2 Algorithme de génération du candidat

candidates_generated(L_{k-1})
C_k=∅
for all itemsets $X \in L_{k-1}$ and $Y \in L_{k-1}$ do
if $X_1 = Y_1 \wedge ... \wedge X_{k-2} = Y_{k-2} \wedge X_{k-1} < Y_{k-1}$ then
begin
C= $X_1 X_2 ... X_{k-1} Y_{k-1}$
add C to C_k
end
delete candidate itemsets in C_k whose any subset is not in L_{K-1}.

Exemple 2.2

Pour un seuil minimal de support minSupp= 40%, premièrement, chaque item I de transaction T est un $1-itemset$ de C_1. Un premier scan de D permet de trouver le support de chaque item ($1-itemset$), seuls les 1-itemset fréquents (support $\geq$ support minimum) sont gardés dans L_1. Par la suite, pour construire les 2-itemsets, une jointure de L_1xL_1 est effectuée pour trouver l'ensemble C_2. De même un deuxième scan et même opérations se répètent pour trouver le L_2. L'ensemble C_5 sera vide car L_4 ne contient pas d'élément. Alors les itemsets fréquents sont les

éléments de L_3. La figure 2.7 représente les étapes de l'algorithme Apriori appliqué à la base de données D.

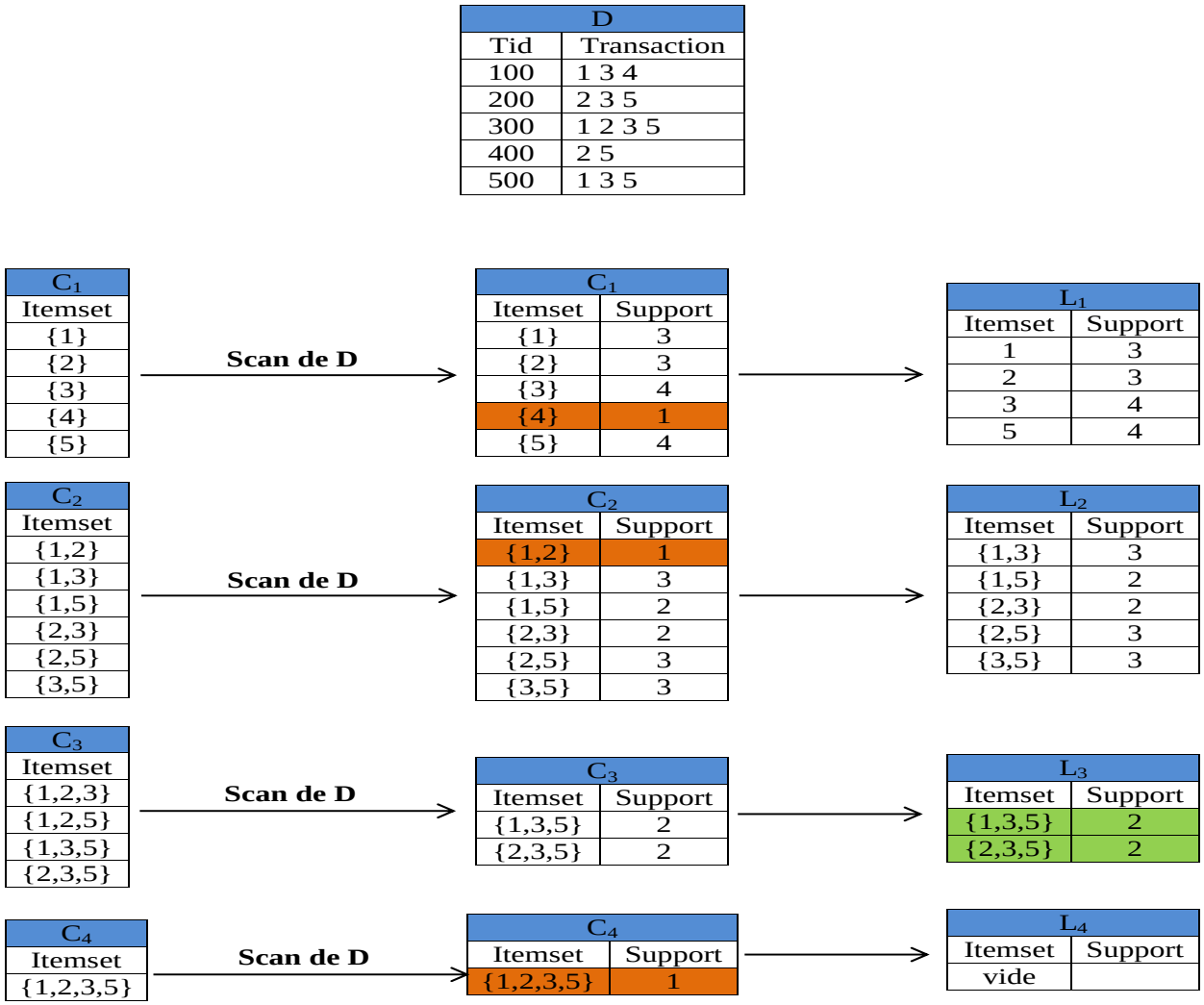

FIGURE 2.7: Exemple d'application de l'algorithme Apriori.

Pour générer les AR, nous limitons à l'itemset {1, 3, 5}. Nous générons tous les sous ensembles non vides pour chaque itemset fréquent. Pour {1, 3, 5} les sous-ensembles non vides sont {1,3}, {1,5}, {3,5}, {1}, {3}, {5}. Pour un seuil minimal de minconf = 60%, la génération des AR à partir de l'ensemble L_3 est présentée dans le tableau 2.2. Les deux dernières seront supprimées car leurs confiance est inférieure à 60%.

3 Étude de cas : classification du cancer du sein

Le cancer du sein est devenu une maladie courante autour du monde et il est le cancer le plus mortel et le plus fréquent chez les femmes. Il apparaît chez les femmes sous forme de tumeurs dans le sein. Il peut être diagnostiqué et détecté par des tests physiques ou une

TABLEAU 2.2: Règles d'association générées

Nom de la règle	Règle (A→B)	Supp(A∪B)/Supp(A)	Confiance	Sélectionnée ?
R1	1,3→ 5	2/3	66.66%	oui
R2	1,5→ 3	2/2	100.00%	oui
R3	3,5→ 1	2/3	66.66%	oui
R4	1→ 3, 5	2/3	66.66%	oui
R5	3→ 1, 5	2/4	50.00%	non
R6	5→ 1, 3	2/4	50.00%	non

analyse d'images. La détection précoce du stade du cancer augmente considérablement les chances de réussite, ce qui pourrait conduire à un taux de survie élevé. La mammographie est actuellement la modalité d'imagerie la plus efficace pour le dépistage du cancer du sein. Toutefois, l'approche traditionnelle de diagnostic du cancer dépend fortement de l'expérience des médecins et leurs inspections visuelles qui sont limitées en raison des erreurs humaines.

Les systèmes experts développés par des techniques de ML sont des outils précieux qui ont été utilisés avec succès pour le diagnostic de la maladie. Des outils de diagnostic assistés par ordinateur sont destinés à aider les médecins afin d'améliorer la précision du diagnostic (Tartar, Kilic, and Akan, 2013). Une étude a été réalisée pour démontrer que le ML peut améliorer la précision du diagnostic. Dans les travaux de Brauses, les résultats montrent que le médecin le plus expérimenté diagnostique avec une précision de 79,97 %, tandis que le diagnostic correct obtenu grâce au ML est de 91,1 %. Cette étude a démontré que le ML pouvait améliorer la précision de diagnostic (Brause, 2001).

Les soins de santé font souvent l'objet de la recherches et des médecins, tant pour le diagnostic que pour le traitement. Malheureusement, en raison d'absence des symptômes de la maladie au début, la détection précoce du cancer est souvent difficile. La mammographie est un processus très complexe qui nécessite des ressources humaines et plus de ressources matérielles qui n'ont pas été efficaces. En se basant sur les données disponibles sur les tests médicaux, la détection précoce peut être simplifiée en exploitant les cas passés pour prévoir les situations actuelles grâce aux techniques de ML.

3.1 Revue de la littérature

Au cours des dernières décennies, des techniques et des méthodes du diagnostic assisté par ordinateur ont été proposées pour améliorer la précision de la classification du cancer du sein afin d'aider les médecins à prendre des décisions de diagnostic. Récemment, ML est la science moderne d'extraction des modèles et de la prévision à partir du big data basées sur les statistiques, le forage de données, la reconnaissance des motifs et l'analyse prédictive. Par

conséquent, à ce jour, plusieurs systèmes experts ont été développés qui sont utilisés dans le domaine médical pour la classification du cancer du sein, le diagnostic est fait en utilisant différentes méthodologies telles que les statistiques, SVM, etc (Arya and Tiwari, 2016).

Il existe de nombreuses techniques de prévision et de classification du cancer du sein. Karabatak et Ince ont combiné AR avec les RNA. AR est utilisé pour réduire la dimension de la base de données sur le cancer du sein et les RNA sont utilisés pour la classification intelligente, ils ont proposé deux techniques différentes pour éliminer les entrées, nommés respectivement AR1 et AR2 (Karabatak and Ince, 2009). AR1 supprime les attributs s'ils dépendent des autres avec une valeur seuil de confiance et de support. AR2 utilise tous les paramètres d'entrée mais pas tous leurs enregistrements. Pauline et Santha kumaran (Paulin and Santhakumaran, 2011) ont utilisé les RNA Feed Forward et l'algorithme de rétropropagation pour entraîner le réseau. La base WBCD est utilisée pour évaluer les performances du réseau. Zheng et al. ont proposé une méthode hybride (K-means +SVM) en utilisant une validation croisée. Les auteurs ci-dessus ont utilisé des méthodes de classification très complexes sur le plan informatique, ils ont utilisé l'algorithme de K-means pour extraire des informations utiles et le SVM pour diagnostiquer la tumeur (Zheng, Yoon, and Lam, 2014). Un modèle basé sur le Rough Set (RS) et le modèle SVM (RSSVM) a été développé par Chen et al. pour le diagnostic du cancer du sein. Ils ont utilisé le RS pour sélectionner les attributs pertinents de la base de données. Une amélioration supplémentaire de la précision du système de diagnostic a été obtenue par le SVM (Chen, 2014).

Dans (Nahato, Harichandran, and Arputharaj, 2015), les auteurs ont utilisé une méthode de relation d'indiscernabilité approximative avec Back Propagation RN (RS-BPNN). Ce modèle fonctionne en deux étapes. La première étape traite les valeurs manquantes pour obtenir une base de données lisse et pour sélectionner les attributs appropriés à partir de la base de données cliniques par la méthode de relation d'indiscernabilité. Dans la deuxième étape, la classification est faite à l'aide d'un réseau de neurones de rétro-propagation. Mert, Kl, Bilgili et Akan ont étudié les propriétés de réduction des attributs de l'analyse en composantes indépendantes (ACI) sur le système d'aide à la décision du cancer du sein. Ils ont prouvé qu'un vecteur caractéristique unidimensionnel obtenu à partir de (ACI) permet au modèle Radial Bases Function Neural Network (RBFNN) de mieux se distinguer avec une précision plus élevée (Mert et al., 2015).

Un nouveau système de classification des maladies liées au cancer du sein basé sur la connaissance et utilisant des techniques de ML a été développé dans (Nilashi et al., 2017). L'espérance-maximisation (Expectation maximization algorithm) et l'analyse en composantes principales (ACP) ont été utilisées respectivement pour regrouper et traiter le problème de la multi-colinéarité dans les bases de données. Ensuite, des arbres de classification et de régression (CART) ont été utilisés pour générer automatiquement les règles floues qui seront utilisées dans le système de connaissances pour la classification des maladies. Le système basé sur la

connaissance a été évalué sur deux bases de données du monde réel, WBCD et les images mammographies. Les résultats de l'expérience ont indiqué que la méthode proposée, qui utilisait des techniques de regroupement, d'élimination du bruit et des règles floues, permettait d'obtenir une bonne précision de prédiction pour le cancer du sein.

Une étude proposée par (Sun, Wang, and Li, 2018) qui s'appuie sur un réseau de neurones profond multimodal en intégrant de données multidimensionnelles pour la prévision du pronostic du cancer du sein. La nouveauté de la méthode réside dans la conception de l'architecture et la fusion de données multidimensionnelles. Les résultats complets de l'évaluation des performances montrent que la méthode proposée est plus performante que les méthodes de prédiction utilisant de données unidimensionnelles et d'autres approches existantes. L'étude proposée dans (Sangaiah and Kumar, 2019) utilise une réduction d'attributs avec un algorithme génétique basé sur l'entropie pour la détection du cancer du sein. La combinaison hybride de ces techniques est utilisée pour traiter l'ensemble de données avec une dimension et des incertitudes élevées. Les données sont obtenues à partir de la base de données WBCD, ces données ont été classées en fonction de différentes propriétés. La performance de la méthode proposée est évaluée et les résultats sont comparés avec d'autres méthodes de sélection des attributs bien connues. Le résultat obtenu montre que la méthode proposée a une capacité remarquable à générer des sous-ensembles de taille réduite des attributs bruités tout en offrant une précision de classification significative pour les grandes bases de données.

La plupart de ces travaux utilisent la classification pour le diagnostic du cancer du sein soit avec la totalité des attributs de la base de données soit avec des méthodes plus complexes et moins précises. Vu l'augmentation de données, l'extraction et la sélection des caractéristiques est devenue un nouveau défis tant pour la précision des résultats que pour la redondance des informations. C'est la clé pour simplifier la phase d'apprentissage du processus de ML et améliorer les performances sans modifier le corps principal des algorithmes. Dans cette partie, une approche combinant AR et le SVM a été proposée pour résoudre le problème de classification du cancer du sein. La méthode proposée se compose de deux parties. Premièrement, l'espace des attributs de la base de données est réduite en utilisant la méthode de AR, qui consiste à éliminer les données inutiles et redondantes. Deuxièmement, le modèle SVM a été appliqué à une nouvelle base de données réduite pour la classification du cancer du sein.

3.2 La base de données (WBCD)

Dans cette étude, la base de données du Wisconsin du cancer du sein (WBCD) a été utilisée et analysée, elle a été utilisée pour entraîner et tester les algorithmes de ML pour classifier le cancer du sein. Elle a été collectée par Wolberg et Mangasarian (1990) à l'université du Wisconsin-Madison (WIlliam H. Wolberg, 2017), elle a été utilisée dans de nombreux travaux

de recherche de ML. Pour chaque observation du cancer, nous disposent des informations suivantes, détaillées dans le tableau 2.3. Nous avons construit une base de données étiquetée avec

TABLEAU 2.3: Description des attributs de WBCD

Numéro d'attribut	La description des attributs	Valeurs des attributs	Moyenne	Standard deviation
1	Clump thickness	1-10	3.42	2.82
2	Uniformity of cell size	1-10	3.13	3.05
3	Uniformity of cell shape	1-10	3.20	2.97
4	Marginal adhesion	1-10	2.80	2.86
5	Single epithelial cell size	1-10	3.21	2.21
6	Bare nuclei	1-10	3.46	3.64
7	Bland chromatin	1-10	3.43	2.44
8	Normal nucleoli	1-10	2.87	3.05
9	Mitoses	1-10	1.59	1.71

des attributs, où l'attribut de l'étiquette de classe est étiqueté avec deux classes, bénigne et maligne. Les valeurs de l'attribut classe ont été modifiées par 0 et 1, où la valeur 1 indique la classe maligne et la valeur 0 pour indiquer la classe bénigne, en la transformant en une base de données binaire, les autres attributs étant détaillées dans le tableau 2.3. La base de données se compose de 699 enregistrements, 241 (34,5 %) maligne et 458 (65,5 %) bénigne. Chaque enregistrement de la base de données possède neuf attributs. Les neuf attributs sont classés sur un intervalle allant d'un état normal à un état anormal de 1 à 10, 10 étant l'état le plus anormal. Le tableau 2.4 présente un échantillon de la base de données WBCD. Dans cette étude, les neuf attributs originaux de la base de données WBCD ont été réduits en utilisant AR, puis ils sont déployés pour évaluer les performances des modèles pour la décision relative au cancer du sein. Ainsi, le modèle proposé est illustré dans la figure 2.8.

TABLEAU 2.4: Exemple d'attributs de WBCD

Clump thickness	Uniformity of cell size	Uniformity of cell shape	Marginal adhesion	Single epithelial cell size	Bare nuclei	Bland chromatin	Normal nucleoli	Mitoses	Classe
5	1	1	1	2	1	3	1	1	0
5	4	4	5	7	10	3	2	1	0
8	10	10	8	7	10	9	7	1	1
5	3	3	3	2	3	4	4	1	1

4 Expérimentation

Le travail proposé dans cette partie est effectué en deux étapes, tout d'abord, un pré-traitement de WBCD qui consiste à l'extraction des attributs pertinents à l'aide de AR, et ensuite, en

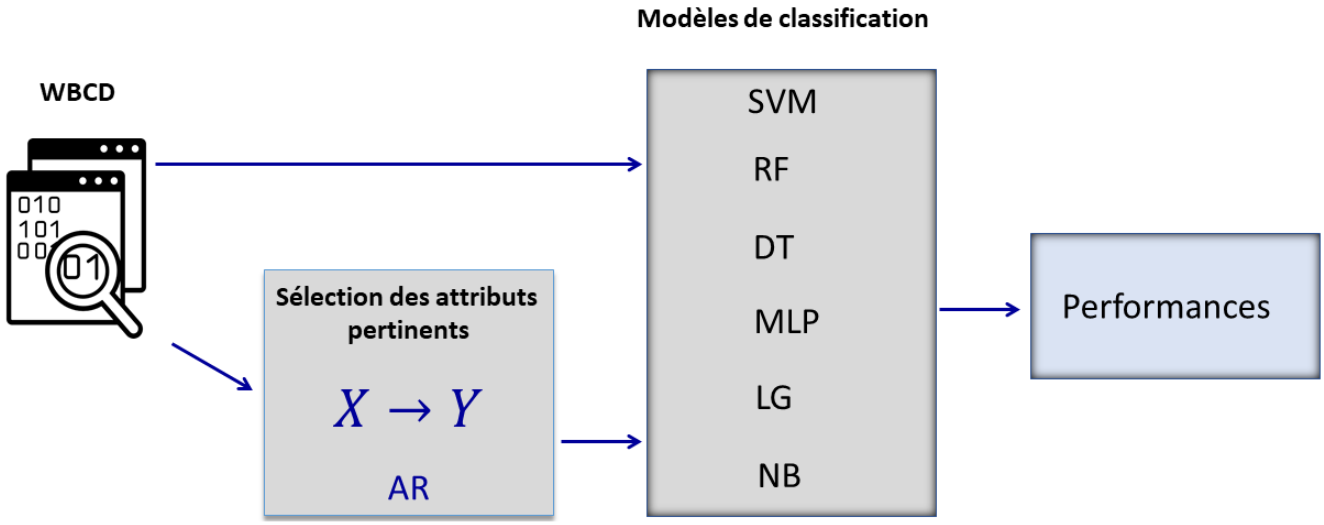

FIGURE 2.8: Modèle de base impliqué dans cette étude.

appliquant les algorithmes de ML, en particulier le SVM. En utilisant le framework de ML disponible publiquement, appelé Weka (Waikato environment for knowledge analysis) avec l'algorithme Apriori, les AR ont été extraites, puis l'espace des attributs a été réduit de neuf à huit et à quatre attributs basés sur les AR extraites. Ensuite, les différents algorithmes sont testés sur l'ensemble de données réduits en utilisant la méthode de validation croisée à trois niveaux, les valeurs moyennes ont été calculées. La mise en œuvre a été effectuée en utilisant python. SVM a été entraîné pour différentes valeurs de C et différentes fonction noyau jusqu'à ce que le résultat soit meilleur. Le meilleur résultat a été obtenu pour C = 1 avec la fonctions poly-noyau dans la procédure de test.

4.1 Application des règles d'association à la base de données WBCD

Le choix des valeurs appropriées pour le support et la confiance dépend de données et les préférences de l'utilisateur. Différentes bases de données peuvent avoir des valeurs différentes. En général, nous fixons un support minimal avec une valeur élevée, puis le diminuer progressivement jusqu'à ce que vous trouvez suffisamment de règles. Le support minimal n'est pas fixé. L'algorithme Apriori commence par une limite supérieure minimale (par défaut 1,0 = 100%), puis il la diminuer itérativement par delta (par défaut 0,05 = 5%). L'algorithme s'arrête lorsque la limite supérieure minimale du support est atteinte, ou lorsque le nombre requis de règles a été généré. En ce qui concerne la confiance minimale, cela est évident car elle représente la confiance que nous voulons avoir dans les règles. La valeur minimale de la confiance dépend également du domaine de connaissances et de données. Donc, généralement, nous choisissons des valeurs supérieures à 60% parce que nous ne sommes pas intéressés par une règle qui est en réalité inférieure à 60% de fiabilité. En général, plus la valeur de la confiance minimale est grande, plus les règles deviennent significatives. En revanche, avec une

petite valeur de confiance, l'algorithme peut ne pas générer des AR. Le choix de ces paramètres dépend également du nombre de règles que vous souhaitez.

Dans notre étude de cas, puisque la base de données n'est pas assez large, seulement 699 enregistrements, nous avons fixé le seuil minimal de support et de la confiance minimale à 50% et 80% respectivement, ce qui peut potentiellement expliquer des règles pertinentes. Malgré les seuils minimaux, l'algorithme s'arrête à un support et une confiance minimum de 50% et 99% respectivement, car le nombre de règles est fixé à 4. Nous avons essayé d'extraire 4 règles plus pertinentes avec une confiance de 100%. L'ajustement des paramètres de AR génère d'autres règles qui seront soit significatives ou non significatives, selon les valeurs des seuils correspondantes. Dans notre étude de cas nous avons essayé de générer des règles qui maximisent la confiance. En fait, en appliquant l'algorithme Apriori à la base de données WBCD, nous avons extrait les meilleures règles suivantes :

1. Attribut 8 =1 , Attribut 9 =1 $\Rightarrow$ Attribut 2 =1 (350 enregistrements, conf =100%)
2. Attribut 2 =1 , Attribut 8 =1 $\Rightarrow$ Classe = 0 (355 enregistrements, conf=100%, 78% (BD))
3. Attribut 2 =1 , Attribut 8 =1 , Attribut 9 =1 $\Rightarrow$ Classe = 0 (349 enregistrements, conf=100%, 76% (BD))
4. Attribut 6 =1 $\Rightarrow$ Classe = 0 (401 enregistrements, 88% (BD))

Nous disons que lorsque la valeur de 8ème et 9ème attributs est égale à 1, la valeur du 2ème attribut est égale à 1, avec une confiance de 100%. Comme résultat de cette règle, le 2ème attribut dépend des autres, 8ème et 9ème attributs. Nous pouvons donc l'éliminer de la base de données. Nous disons aussi que lorsque la valeur de 2ème et 8ème attributs est égale à 1, la classe est bénigne, avec une confiance de 100%. Cette règle présente un grand itemsets pour la classe bénigne (78% de toute la base de données), et lorsque la valeur de 2ème, 8ème et 9ème attributs est égale à 1, la classe est bénigne, avec une confiance de 100%, cette règle présente un grand itemsets pour la classe bénigne (76% de toute la base de données). Nous disons aussi que lorsque la valeur de 6ème attribut est égale à 1, la classe est bénigne, cette règle présente un grand itemsets pour la classe bénigne (88% de toute la base de données). De même, lorsque la valeur de 6ème attribut est égale à 10, la classe est maligne, cette règle présente un grand itemsets pour la classe maligne (54% de toute la base de données). Selon ces règles, nous pouvons dire que le 2ème, 8ème, 9ème attributs peuvent déjà définir la classe bénigne et le 6ème attribut peut définir la classe maligne. Nous pouvons donc réduire la base de données WBCD de neuf à huit attributs (tous les attributs sauf le deuxième) qui sont :

-Clump thickness
-Uniformity of cell shape
-Marginal adhesion
-Single epithelial cell size

-Bare nuclei
-Bland chromatin
-Normal nucleoli
-Mitoses
D'autre part, nous pouvons réduire la base de données WBCD de neuf à quatre attributs (2ème, 6ème, 8ème et 9ème attributs) qui sont :
-Uniformity of cell size
-Bare nuclei
-Normal nucleoli
-Mitoses

4.2 Résultats sans utiliser les règles d'association

Les résultats expérimentaux obtenus pour la base de données WBCD sont présentés dans le tableau 2.5. On utilisant la méthode de validation croisée avec k=3 (threefold cross-validation), nous obtenons un accuracy moyen de 94,85% pour le DT (J48), 96,42% pour le RF, 96,85% pour le LG, 97,42% pour le BN, 97,00% pour le SVM, 95,57% pour le MLP. Les mesures d'évaluation

TABLEAU 2.5: Résultats d'analyse de la base WBCD sans utiliser AR

Modèles	Temps d'apprentissage de modèle (Secondes)	Cas bien classés	Cas mal classés	TP	TN	FP	FN
BN	0,05	681	18	443	238	3	15
SVM	0,02	678	21	446	232	9	12
DT(J48)	0,01	663	36	436	227	14	22
LG	0,16	669	22	447	230	11	11
RF	0,14	674	25	442	232	9	16
MLP	0,73	668	31	442	226	15	16

des performances des autres algorithmes et le modèle SVM, telles que la précision, le rappel, F-measure, la spécificité et l'accuracy sont donnés dans le tableau 2.6 et la figure 2.9.

TABLEAU 2.6: Evaluation des performances sans utiliser AR (%)

Modèles	Précision	Rappel	F-measure	Spécificité	Accuracy
BN	99,33	96,72	98,01	98,75	97,42
SVM	98,02	97,38	97,70	96,26	97,00
DT(J48)	96,89	95,20	96,04	94,191	94,85
LG	97,60	97,60	97,60	95,43	96,85
RF	98,00	96,51	97,25	96,26	96,42
MLP	96,72	96,51	96,61	93,77	95,57

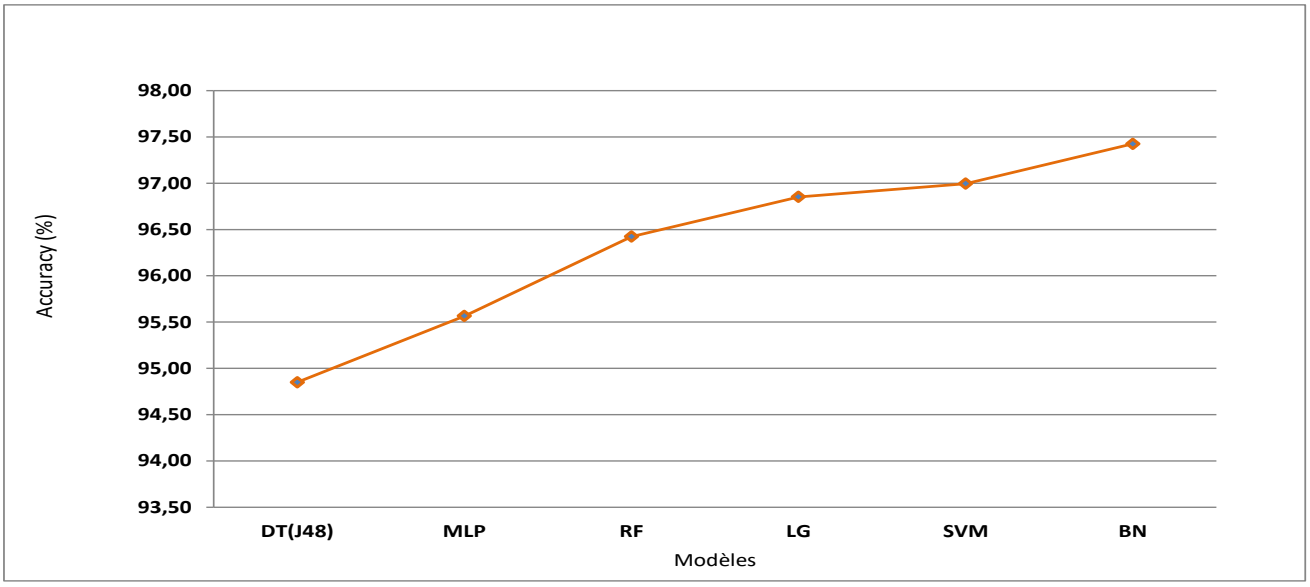

FIGURE 2.9: Comparaison d'accuracy sans utiliser AR.

4.3 Résultats en utilisant les règles d'association

Dans le deuxième test, nous avons d'abord utilisé la sélection des attributs en se basant sur AR pour sélectionner les meilleurs attributs, puis nous avons utilisé les mêmes techniques de ML que dans la section précédente. Les résultats expérimentaux ont montrés que les performances de classification les plus élevées sont atteintes lorsque le SVM est utilisé comme modèle de classification. Par conséquent, l'approche proposée dans cette étude est une approche où AR est utilisé pour la réduction des attributs et le SVM est utilisé pour la classification, comme c'est indiqué dans la figure 2.8.

Les résultats expérimentaux obtenus pour la base de données WBCD sont indiqués dans le tableau 2.7, le tableau 2.8 et la figure 2.10. Avec 8 attributs, nous obtenons un accuracy de 95,42% pour le DT (J48), 96,28% pour le RF, 97,00% pour le LG, 97,14% pour le BN, 98,00% pour le SVM et 95,42% pour le MLP. Avec 4 attributs, nous avons obtenu un Accuracy de 94,99% pour le DT (J48), 95,57% pour le RF, 95,57% pour le LG, 96,14% pour le BN, 96,14% pour le SVM et 96,14% pour le MLP. D'après le tableau 2.8, les résultats obtenus avec AR sont meilleurs que ceux obtenus par les autres modèles sans utiliser AR. Comme le montre le tableau 2.8, les résultats les plus performants sont accomplis en utilisant AR avec le SVM dans le cas de huit attributs avec un accuracy de classification de 98,00%. Les mesures de performance des autres modèles et le SVM tels que le temps d'apprentissage, les cas correctement classés, les cas mal classés, TP, TN, FP, FN sont données dans le tableau 2.7 pour comparer l'effet de AR sur la classification. Les mesures de performance des autres algorithmes et le SVM telles que la précision, le rappel, F-measure, la spécificité et l'accuracy sont données dans le tableau 2.8 et la figure 2.10. Le tableau 2.9 présente les performances de AR avec le SVM.

TABLEAU 2.7: Résultats d'analyse de la base WBCD en utilisant AR

Modèles	Dimension	Temps d'apprentissage (Secondes)	Cas bien classés	Cas mal classés	TP	TN	FP	FN
BN	8	0,02	679	20	442	237	4	16
	4	0,01	672	27	441	231	10	17
SVM	8	0,02	685	14	448	237	5	9
	4	0,01	672	27	446	226	15	12
DT(J48)	8	0,01	667	32	442	225	16	16
	4	0,00	664	35	436	228	13	22
LG	8	0,12	672	21	447	231	10	11
	4	0,10	667	31	445	223	18	13
RF	8	0,10	673	26	444	229	12	14
	4	0,09	668	31	440	228	13	18
MLP	8	0,67	667	32	441	226	15	17
	4	0,38	672	27	439	233	8	19

TABLEAU 2.8: Évaluation des performances en utilisant AR (%)

Modèles	Dimension	Précision	Rappel	F-measure	Spécificité	Accuracy
BN	8	99,10	96,51	97,79	98,340	97,14
	4	97,78	96,29	97,03	95,85	96,14
SVM	**8**	**98,90**	**98,03**	**98,46**	**97,93**	**98,00**
	4	96,75	97,38	97,06	93,776	96,14
DT(J48)	8	96,51	96,51	96,51	93,361	95,42
	4	97,10	95,20	96,14	94,606	94,99
LG	8	97,81	97,60	97,70	95,85	97,00
	4	96,11	97,16	96,63	92,53	95,57
RF	8	97,37	96,94	97,16	95,02	96,28
	4	97,13	96,07	96,60	94,60	95,57
MLP	8	96,71	96,29	96,50	93,77	95,42
	4	98,21	95,85	97,02	96,680	96,14

TABLEAU 2.9: La comparaison des performances de AR avec le SVM

	SVM	AR+SVM	
Dimension de l'espace	9	8	4
Accuracy (%)	96.99	98.00	96.14

Le tableau 2.10 présent les matrices de confusion indiquant les résultats de classification des différents modèles mis en œuvre pour la détection du cancer du sein. A partir de ces matrices, on peut savoir la fréquence avec laquelle un enregistrement est mal classé comme un autre. Comme le montre le tableau 2.11, la méthode proposée fournie une qualité de prédiction stable et élevée par rapport aux résultats expérimentaux précédents, notamment ceux obtenus

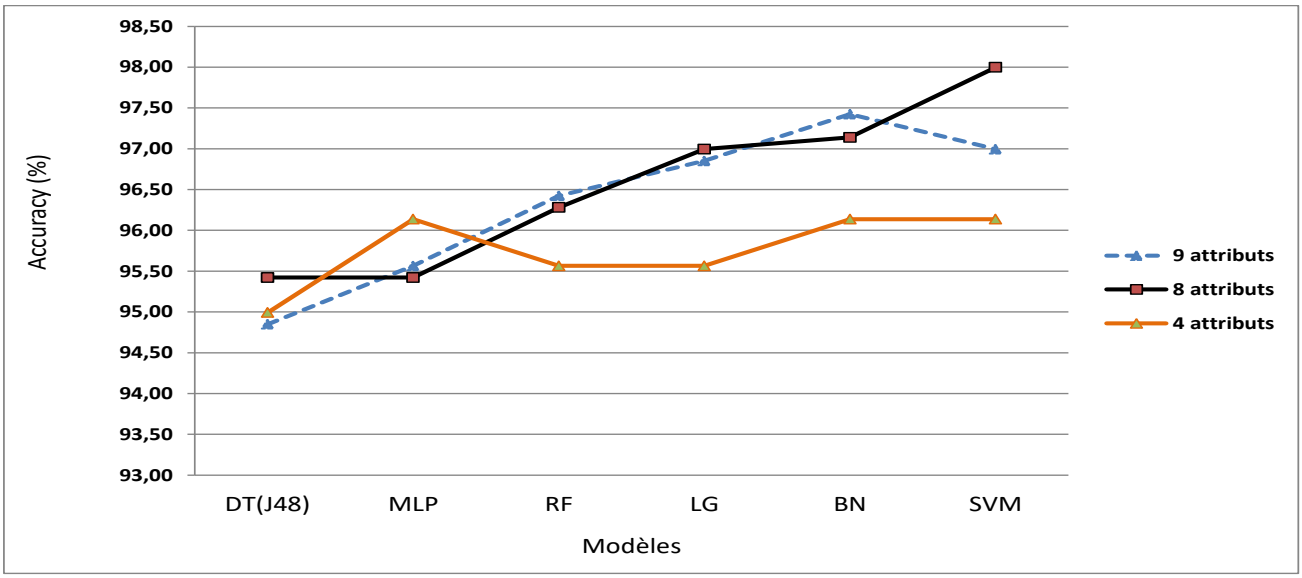

FIGURE 2.10: Comparaisons d'accuracy.

TABLEAU 2.10: Matrices de confusions

Modèles	Dimension	Résultat désiré	Résultat de sortie	
			Bénigne	Maligne
SVM	9	Bénigne	446	12
		Maligne	9	232
AR+SVM	8	Bénigne	448	9
		Maligne	5	237
AR+SVM	4	Bénigne	446	12
		Maligne	15	226

par Karabatak and Ince, 2009. Les performances démontrées par l'ensemble des techniques de ML pour le diagnostic du cancer du sein résident dans le choix des variables d'entrées et le choix de la méthode de classification. Les attributs les plus appropriés pour le diagnostic du

TABLEAU 2.11: Comparaison de notre approche avec d'autres approches

Références	Dimension	Modèle	Accuracy (%)
Karabatak and Ince, 2009	8 (wbcd)	AR-NN	97.40
Zheng, Yoon, and Lam, 2014	6 (wdbc)	K-SVM	97.38
Stoean and Stoean, 2013	—	SVM-evoalgo.	97.23
Prasad and Biswas, 2010	17 (wdbc)	PSO-SVM	97.37
Sweilam, Tharwat, and Moniem, 2010	30 (wdbc)	QPSO + SVM	93.06
Notre étude	**8** (wbcd)	**AR-SVM**	**98.00**
	4 (wbcd)	**AR-SVM**	**96.14**

cancer du sein doivent être utilisés comme les entrées du modèle. Pour cette raison, AR est

approprié pour la sélection des attributs de la base de données WBCD. Dans le deuxième test, où AR est appliqué, la précision la plus élevée obtenue est de 98,00% avec le même modèle utilisé précédemment. Pour voir l'utilité de la réduction de la base de données et tester le temps d'apprentissage avec des bases de données assez larges, nous avons dupliqué le nombre des enregistrements. Le temps d'apprentissage est calculé pour chaque cas. La figure 2.11 illustre les résultats. On peut voir que le temps d'exécution augmente avec l'implication de plus d'attributs dans la base de données.

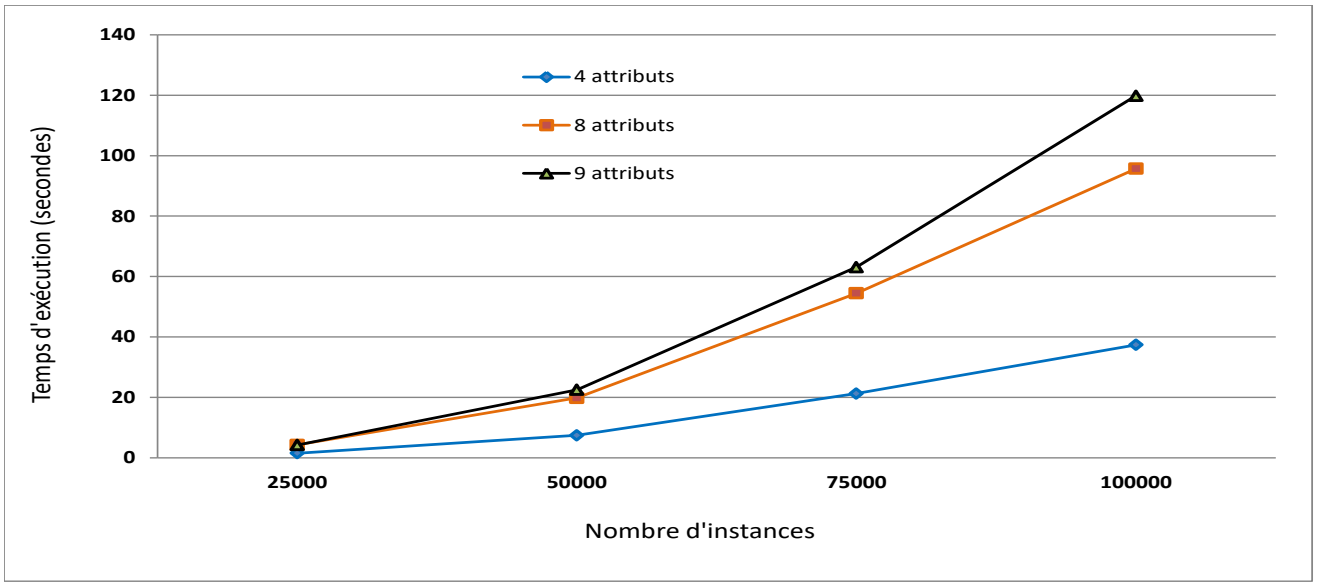

FIGURE 2.11: Temps d'apprentissage.

Dans la littérature, plusieurs méthodes ont été proposées pour le diagnostic du cancer du sein basées sur la classification et le regroupement. Cependant, ces dernières années, la quantité de données disponibles a considérablement augmentée. Les méthodologies traditionnelles montrent leurs inconvénients sur les données à grande échelle. Bien que l'utilisation de ces méthodes de classification nécessite un temps de calcul élevé pour les différents apprentissages. Comme la classification est basée sur la relation entre les variables d'entrées et la variable cible, la sélection des variables d'entrées qui ont la plus forte relation avec la variable cible améliorera significativement l'accuracy de classification. D'autre part, AR s'est principalement concentré sur l'exploration et la recherche des associations et les relations à travers de grands ensembles d'attributs. Il permet d'extraire les variables qui contribuent de manière significative à la variable cible. Par conséquent, les résultats obtenus montrent que l'utilisation de AR avec un modèle efficace comme le SVM est encore plus significatif que d'autres méthodes.

En se basant sur les résultats obtenus, l'approche proposée pour le diagnostic du cancer du sein offre non seulement un meilleur accuracy par rapport aux résultats expérimentaux précédents avec un ensemble minimal d'attributs, mais elle montre également la réduction des attributs de manière simple, ce qui permet de gagner du temps lors de la phase d'apprentissage,

en particulier dans le contexte du big data qui est devenu un grand défi. D'autre part, les médecins peuvent également tirer parti des attributs représentatifs de la tumeur extraits en comprenant la relation entre les différents attributs.

4.4 Conclusion

Les techniques de ML sont largement utilisées pour découvrir des modèles à partir de données, qui peuvent ensuite être utilisés à des fins diverses. La réduction des attributs est une étape importante dans le processus de prétraitement de données. En effet, pour les données appartenant à un grand espace, certains attributs ne fournissent pas des informations, d'autres sont redondantes ou corrélées. Cela rend les algorithmes de décision, inefficaces et difficile à interpréter. Ainsi, l'extraction et la réduction des attributs amélioreront considérablement les performances des algorithmes d'apprentissage. Dans cette partie, nous avons examiné les algorithmes de ML les plus connus dans la littérature, par la suite, une approche pour détecter le cancer du sein basée sur AR et le SVM a été proposée. Les performances de l'approche proposée sont évaluées et comparées avec le modèle SVM et les travaux antérieurs dans ce contexte.

Avec l'augmentation de données dans de nombreux domaines et en particulier celui des soins de santé, ces données nécessitent une gestion et une analyse appropriée afin de dériver des informations significatives. En autre, il y a plusieurs défis à relever associées à chaque étape du traitement de données volumineuses qui ne peuvent pas être dépassées qu'en utilisant des solutions informatiques dédiées spécifiquement à l'analyse de données volumineuses et big data. Dans le chapitre suivant nous abordant une vue d'ensemble du big data, définition caractéristiques, les technologies récentes développées pour le big data et ses applications.

Partie II

Le diagnostic médical en temps réel dans un environnement big data

3 Principes fondamentaux du big data

1 Introduction

Ces dernières années, le développement rapide de l'internet, de l'internet des objets et du cloud computing a entraîné une croissance explosive de données dans presque tous les secteurs industriels et commerciaux, appelées big data (Manogaran and Lopez, 2018). Les données volumineuses sont rapidement devenues un sujet brûlant qui attire l'attention des universités, de l'industrie et des gouvernements du monde entier. Ce volume vertigineux de données ne peut plus être collecté, traité, stocké et exploité par des solutions informatiques traditionnelles combinant des infrastructures physiques et des bases de données relationnelles. Pour cela, de nouvelles technologies ont émergées pour nous aider à relever ces défis. Dans ce chapitre, nous commençons par présenter brièvement le concept du big data, notamment sa définition, ces caractéristiques et sa chaine de valeur. Nous identifions ensuite, selon différentes perspectives, les outils et les méthodes existantes. Le développement des nouvelles architectures des systèmes pour l'analyse et la prédiction du big data représente un grand défi. C'est dans ce contexte que s'inscrit la dernière partie de ce chapitre.

2 Big data : définition

Big data est un terme qui désigne le grand volume de données (parfois appelé méga-données) structurées et non structurées que les outils classiques de traitement de données sont inadéquats de les traiter. Les défis du big data comprennent le stockage, l'analyse, la recherche, le partage, le transfert, la visualisation, la mise à jour et la confidentialité de l'information. Le terme big data se réfère souvent et simplement à l'utilisation de l'analyse prédictive, l'analyse du comportement de l'utilisateur ou certaines autres méthodes avancées d'analyse de données qui extraient la valeur de données et rarement à une taille particulière de l'ensemble de données. Même si un certain nombre de définitions du big data existent, la définition la plus populaire et la mieux acceptée a été donnée par Douglas Laney. Laney a observé que les données (big

data) se développaient dans trois dimensions différentes, à savoir, le volume, la vitesse et la variété connus sous le nom de 3 V (Laney, 2001). Ces trois V sont devenus la définition standard du big data. Bien que d'autres chercheurs aient ajouté plusieurs autres V à cette définition (De Mauro, Greco, and Grimaldi, 2016), le 4ème V le plus accepté reste la véracité. Par ailleurs, d'autres chercheurs (Mohammed, Humbe, and Chowhan, 2016) ont définis le big data par 5V. La figure 3.1 représente le résumé des 5V du big data :

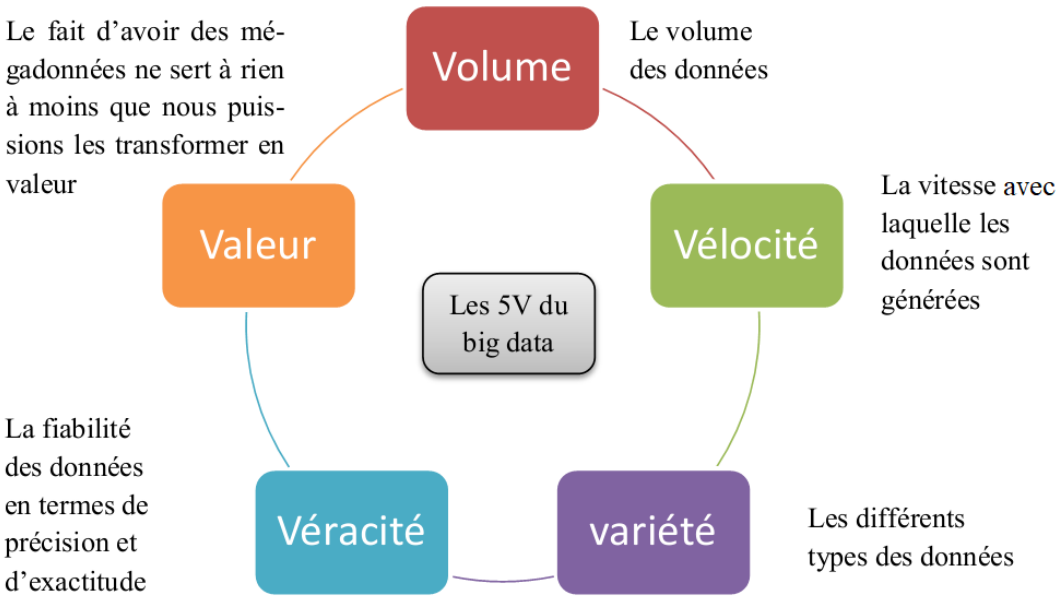

FIGURE 3.1: Les 5V du big data.

- **Volume** : big data implique d'énormes volumes de données. C'étaient des employés qui créaient des données. Actuellement les données sont générées par différentes technologies cliniques, des machines, des réseaux et des interactions humaines sur des systèmes comme les réseaux sociaux. Le volume de données à analyser est massif, pourtant, le volume de données n'est pas autant le problème que les autres V.

- **Vélocité** (Rapidité) : la vélocité du big data traite les données au rythme auquel le flux de données proviennent de source différentes. Le flux de données est massif et continu. Dans le domaine de la santé, IoT médical et bien d'autres génères des données massives en temps réel pour la prévision et le suivi médical.

- **Variété** : la variété fait référence aux nombreuses sources et types de données à la fois structurées et non structurées. Les données se présentent sous forme de texte, d'emails, de photos, de vidéos, de fichiers PDF, d'audio, etc. Cette variété de données non structurées crée des problèmes de stockage, d'extraction et d'analyse. Le big data n'est pas

seulement grande, mais aussi variée et en croissance rapide. De nombreuses technologies et techniques analytiques sont nécessaires afin de tenter d'extraire des informations pertinentes.

- **Véracité** : lorsque nous traitons un volume, une vélocité et une variété de données élevés, il n'est pas possible que toutes les données soient correctes à 100%, il y aura de données sales. La qualité de données capturées peut varier considérablement. La précision de l'analyse de données dépend de la véracité de source de données. Nous disposons de toutes les données, mais se pourrait-il qu'il nous manque quelque chose ? Les données sont-elles propres et exactes ? Ont-elles vraiment quelque chose à offrir ?.

- **Valeur** : la valeur est l'aspect le plus important du big data. Cependant, la valeur potentielle du big data est énorme. C'est bien d'avoir accès au big data, mais à moins que nous ne puissions les transformer en valeur, cela devient inutile.

3 MapReduce : les concepts de base

MapReduce (Dean and Ghemawat, 2008) est un framework proposé par Dean et Ghemawat à Google pour traiter des problèmes parallèles à travers des ensembles de données géants utilisant un grand nombre d'ordinateurs (nœuds). Le calcul est effectué sur les nœuds est indépendant de l'emplacement physique des nœuds, c'est-à-dire, si tous les nœuds sont sur le même réseau, utilisent du matériel similaire (appelé cluster) ou les nœuds sont partagés entre les systèmes répartis géographiquement, administrativement et utilisent du matériels plus hétérogène (appelé grille). Il est également indépendant de l'emplacement des données. Les données peuvent être stockées dans un cluster à n'importe quel nœud. Le traitement informatique peut se faire sur des données stockées dans un système de fichiers (non structuré/HDFS) ou dans une base de données (structurée/HBase). Il se compose de deux fonctions, Map et reduce:

• **Map**: le nœud maître (master) prend l'entrée, la divise en sous-problèmes plus petits et la distribue aux nœuds esclaves (workers). Un nœud esclave peut le faire à nouveau à tour de rôle, menant à une structure arborescente multiniveau. Le nœud esclave traite le problème plus petit et transmet la réponse à son nœud maître.

• **Reduce** : le nœud maître collecte ensuite les réponses de tous les sous-problèmes et les combine d'une façon à former la sortie de la réponse au problème. Une tâche MapReduce divise généralement l'ensemble de données d'entrée en blocs indépendants qui seront traités par la fonction Map d'une manière complètement parallèle. La fonction Map trie les sorties des Maps, qui seront ensuite des entrées dans les tâches de réduction. Généralement, l'entrée et la sortie de la tâche sont stockées dans un système de fichiers. La fonction Map prend en charge les tâches de planification, les surveille et ré-exécute les tâches ayant échoué.

Le framework MapReduce (Lee et al., 2012) fonctionne exclusivement sur des paires <clé, valeur>, c'est-à-dire que la structure considère l'entrée de la tâche (job) comme un ensemble de paires <clé, valeur> et produite un ensemble de paires <Clé, valeur>, concevable de différents types. Le framework MapReduce suit cette architecture (Figure 3.2) (Bhosale and Gadekar, 2014):

Map (k1, v1) —> list (k2, v2)

Reduce (k2, list (v2)) —> list (k3, v3)

Exemple (WordCount)

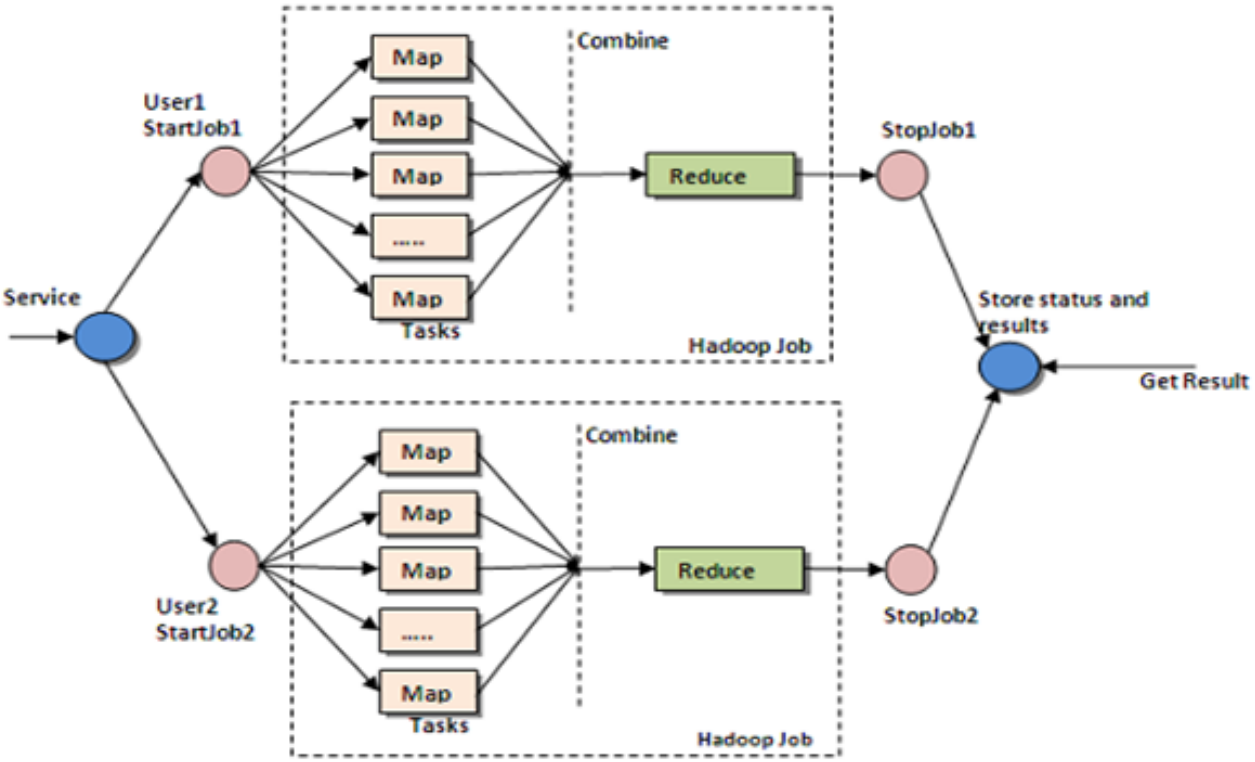

FIGURE 3.2: Architecture de MapReduce.

Considérons le texte suivant comme l'entré de de programme MapReduce "Deer Bear River Car Car River Deer Car Bear", le processus de MapReduce appliqué à cet exemple est illustré dans la figure 3.3.

- **Limitations de MapReduce**

L'un des principaux inconvénients de MapReduce est son inefficacité dans l'exécution des algorithmes itératifs. MapReduce n'est pas conçu pour les processus itératifs. Les mappeurs lisent les mêmes données encore à partir du disque. Par conséquent, après chaque itération, les résultats doivent être écrits sur le disque pour les passer à la prochaine itération. Cela rend l'accès au disque un goulot d'étranglement majeur, ce qui dégrade considérablement la performance. Pour chaque itération, un nouveau mappeur et réducteur doit être initialisé. Parfois, les

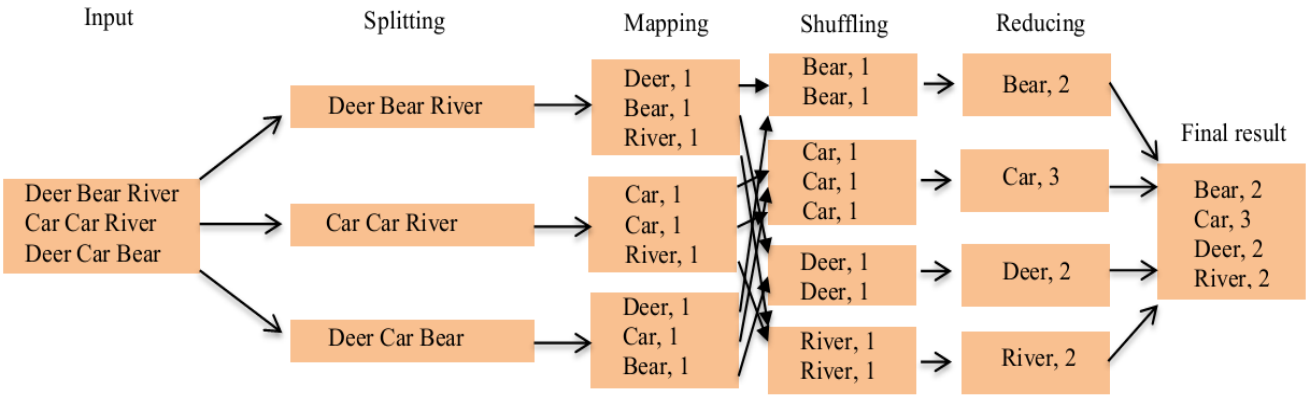

FIGURE 3.3: Le processus global de MapReduce appliqué à WordCount.

tâches MapReduce sont de courte durée, auquel la surcharge de l'initialisation de cette tâche devient un surcoût significatif pour la tâche elle-même.

4 Les outils d'analyse du big data

Le big data est connu comme le processus dans lequel nous collectons et analysons les données qui aide à extraire des modèles cachés utiles et d'autres informations telles que les choix des clients, les tendances du marché, la prise de décision et la prédiction. Par conséquent, big data est un domaine qui regroupe plusieurs disciplines principalement ML, DM et le parallélisme. Dans la présente section, nous allons présenter et discuter les plateformes existantes pour le traitement du big data et pour chaque phase de la chaine de valeur big data. Nous présentons également une étude comparative des plateformes les plus populaires, en mettant l'accent sur l'écosystème Hadoop.

4.1 Les plateformes distribuées

4.1.1 Hadoop

Apache Hadoop[1] est l'une des plateformes logicielles les mieux établies qui permettent le traitement distribué du big data sur des clusters d'ordinateur en utilisant des modèles de programmation simples. Hadoop met en œuvre le paradigme MapReduce. Il est conçu pour passer de serveurs uniques à des milliers de machines, chacune d'entre elles assurant le calcul et le stockage local. D'une manière générale, le framework Hadoop utilise le système de fichier distribué HDFS pour le stockage et la récupération de données massives d'une manière distribuée. Par conséquent, les performances d'entrées/sorties d'une tâche Hadoop MapReduce dépend fortement de HDFS. En effet, le HDFS a un temps d'accès non négligeable; la lecture et

[1] Voir http://hadoop.apache.org/

l'écritures de données sont suffisamment longues en raison de l'utilisation du disque dur pour sauvegarder les données intermédiaires comme le montre la figure 3.4. Le processus de ML qui fait généralement de nombreuses itérations sur les données soit inefficace. Le framework est conçue pour détecter et gérer les pannes au niveau de la couche application, fournissant ainsi un service hautement disponible au-dessus d'un cluster d'ordinateurs, qui peuvent être sujets à des pannes. La figure 3.5 illustre l'architecture de base de Apache Hadoop.

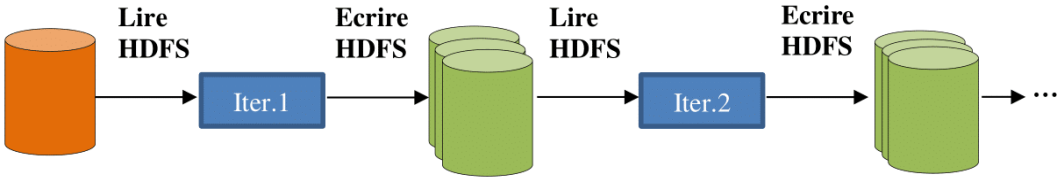

FIGURE 3.4: La lecture et l'écriture vers et à partir du HDFS dans les algorithmes itératifs de ML.

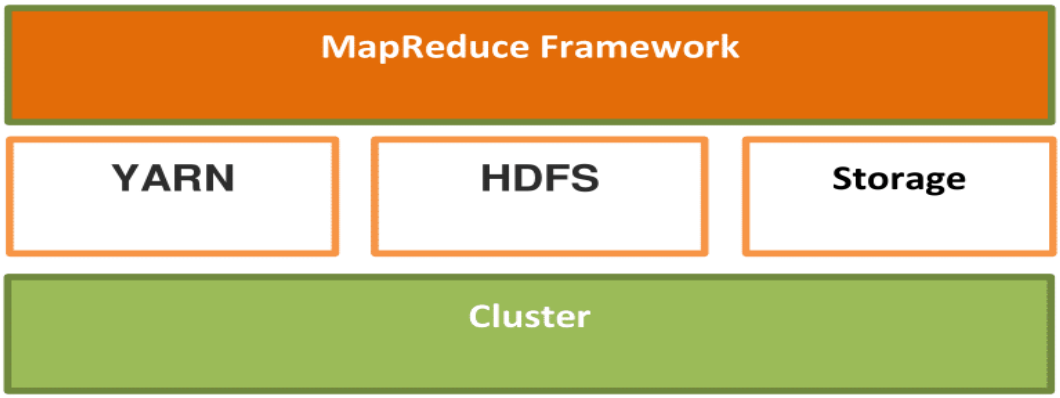

FIGURE 3.5: Architecture de Apache Hadoop.

- **Cluster**

C'est l'ensemble des machines (nœuds). Il s'agit de la partie matérielle de l'infrastructure.

- **YARN**

(Yet Another Resource Negotiator) (Vavilapalli et al., 2013) un mécanisme d'ordonnancement de programmes de type MapReduce il se compose de :
-Resource Manager (Un par cluster) est le master. Il sait où se trouvent les Workers et combien ont-ils de ressources.
-Node Manager (Plusieurs par cluster) est le worker de l'infrastructure, lorsqu'il démarre, il s'annonce au gestionnaire de ressources. Chaque gestionnaire de nœud propose certaines ressources au cluster, sa capacité de ressource, la quantité de mémoire et le nombre de processeur.

- **HDFS**

(Hadoop Distributed File System) (Borthakur et al., 2008) est un système de fichier qui répartit les données sur de nombreuses machines. HDFS est le framework chargé de fournir un stockage permanent, fiable et distribué. Il est généralement utilisé pour stocker les entrées et les sorties.

- **Framework MapReduce**

C'est la couche logicielle implémentant le paradigme MapReduce.

- **L'aspect techniques de Hadoop**

Hadoop possède quelques fichiers de configuration importants qui doivent être prisent en compte lors de la configuration du Hadoop. Ci-dessous les fichiers de configuration les plus importants avec une petite description :
-hadoop-env.sh/cmd: variables d'environnement qui utilise un script pour exécuter les tâches de Hadoop.
-Core-site.xml: contient les paramètres de configuration pour Hadoop Core, tels que les paramètres d'E/S qui sont MapReduce et HDFS.
-Hdfs-site.xml: paramètres de configuration de HDFS Daemons: le namenode, le namenode secondaire, et les nœuds de données (datanodes).
-Mapred-site.xml: paramètres de configuration pour Daemons MapReduce: le jobtracker et le Tasktrackers.
-Masters: liste des machines qui exécutent chacune un programme secondaire Namenode.
-Slaves: liste des machines qui exécutent chacune un datanode et un tasktracker. Le tableau suivant (Tableau 3.1) présente quelques avantages et inconvénients de MapReduce.

4.1.2 Apache Spark

Récemment, le monde du big data a connu un changement dynamique avec l'introduction d'une version stable d'Apache Spark [2], qui fournit une interface de programmation conviviale pour réduire les efforts de codage et une meilleure performance dans la majorité des problèmes liés au big data. Spark ne fournit pas seulement une alternative au paradigme MapReduce, mais a également des options comme les requête SQL, et une bibliothèque d'apprentissage automatique, MLlib. Les performances et le fonctionnement de Spark sont considérablement différent de celui de MapReduce, et également dépend des contraintes de parallélisme, les types de problèmes et les ressources disponibles. Apache Spark (Zaharia et al., 2012a) a été commencé comme un projet de recherche à UC Berkeley dans l'AMPLab, Spark a été lancé dans le but de concevoir un modèle de programmation qui supporte des types d'applications

[2] voir https://spark.apache.org

TABLEAU 3.1: Avantages et inconvénients de Hadoop MapReduce

Avantages	Inconvénients
-Distribue les données et les calculs, le calcul local empêche la surcharge du réseau. -Les tâches sont indépendantes, donc nous pouvons facilement supporter une défaillance partielle. Ici, les nœuds entiers peuvent échouer et redémarrer. -Modèle de programmation simple. Le programmeur écrit seulement des tâches de map et reduce. -Évolutivité linéaire. -HDFS stocke une grande quantité d'informations de différents types, avec un accès rapide. -Tolérance de défaut en détectant les défauts et en appliquant une récupération rapide et automatique. -Portabilité à travers de matériels et de systèmes d'exploitation hétérogènes. -Efficacité en répartissant les données et la logique pour les traiter en parallèle sur les nœuds où se trouvent les données. -Fiabilité en maintenant automatiquement plusieurs copies de données et en redéployant automatiquement la logique de traitement en cas de défaillance. -Aptitude à traiter rapidement de grandes quantités de données en parallèle	-Hadoop est écrit en Java, qui est plus lent que les autres frameworks en C ou C ++. -Le système de fichier Hadoop/MapReduce aborde les données entre les étapes de réduction, ce qui constitue une énorme contrainte de performance. -Le système de fichier Hadoop gère centralement un index nécessaire pour mapper les tâches vers les données distribuées dans les nœuds. -Les opérations nécessitent une co-implantation de données pour calculer le résultat (jointures, agrégations, sortes, déduplications, etc.) qui seront s'exécutées d'une manière non efficace lorsque la distribution de données ont de différente indices. -Hadoop n'est pas un bon choix lorsqu'il s'agit du traitement en temps réel. -Le démarrage des tâches est lent, ce qui peut constituer une lourde pénalité pour les performances, en particulier pour des petits traitements. -Il n'est pas toujours facile de mettre en œuvre le paradigme MapReduce à tous les problèmes. -Le paradigme MapReduce n'est pas un bon choix lorsque vos processus intermédiaires ont besoin de parler les uns aux autres (les taches s'exécutent isolement).

que MapReduce, tout en maintenant ses fonctions de tolérance aux pannes. Spark offre une abstraction appelée Données Distribuées et Résilientes (Resilient distributed Datasets) (RDDs) pour supporter efficacement ces applications. Les RDDs peuvent être stockés en mémoire entre les requêtes sans réplication. Ils reconstruisent les données perdues en cas d'échec : chaque RDD se souvient de la façon dont il a été construit à partir de l'ensemble de données (par transformations comme map, join ou groupBy). Les RDDs (Zaharia et al., 2010) permettent à Spark d'augmenter la performance Jusqu'à 100x dans les analyses itératives. Les RDDs peuvent prendre en charge une grande variété d'algorithmes itératifs. La figure 3.6 illustre l'architecture de base de Apache Spark.

- **Les composantes de Spark**

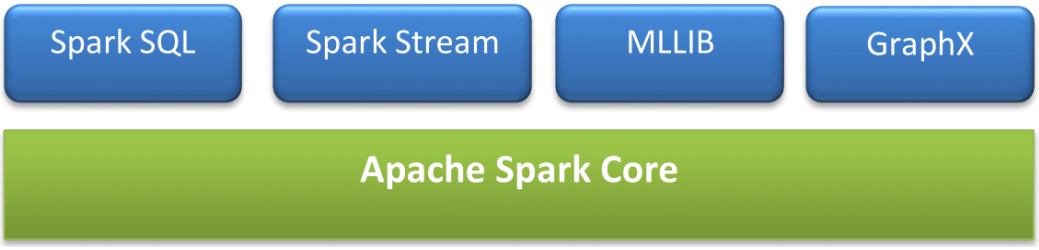

FIGURE 3.6: Architecture de Apache Spark.

-Apache Spark Core : c'est le cœur de Spark, il est responsable des fonctions de gestion, telles que l'ordonnancement des tâches. Spark Core implémente et dépend d'une abstraction de programmation connue sous le nom RDDs.
-Spark SQL : il s'agit du module Spark pour le traitement des données structurées et il est conçu pour supporter les charges de travail qui combinent des requêtes de base de données SQL familières et des analyses algorithmiques plus compliquées. Spark SQL prend également en charge les connexions JDBC et ODBC, permettant un degré d'intégration avec les bases de données existantes, les entrepôts de données et les outils de business intelligence.
-Spark Stream : Spark permet de traiter des données qui sont figées à un instant T. Grâce au module Spark Streaming, il est possible de traiter des flux de données qui arrivent en continu, et donc de traiter ces données au fur et à mesure de leurs arrivées.
-MLlib : il s'agit de la bibliothèque de ML évolutive de Spark, qui met en œuvre un ensemble d'algorithmes de statistiques et de ML couramment utilisés. Il s'agit de corrélations et de test d'hypothèses, de classification et de régression, de regroupement et d'analyse en composantes principales...
-GraphX : ce module a commencé la vie comme un projet de recherche séparé de UC Berkeley, qui a été par la suite donné au projet Apache Spark. GraphX prend en charge l'analyse et le calcul sur des graphes de données et prend en charge une version de l'API Pregel du traitement du graphe. GraphX inclut un certain nombre d'algorithmes de graphes largement utilisés, y compris le PageRank.
-Resilient Distributed Datasets (RDDs) : données distribuées et résilientes est un concept au cœur de Spark. Il est conçu pour prendre en charge le stockage de données en mémoire (RAM), réparti sur un cluster d'une manière qui est à la fois tolérante aux pannes et efficace. La tolérance aux pannes est obtenue par le suivi des transformations appliquées aux ensembles de données. L'efficacité est obtenue par la parallélisation du traitement entre plusieurs nœuds du cluster et la minimisation de la réplication des données entre ces nœuds. Une fois les données chargées dans un RDD, deux types de fonctionnement de base peuvent être effectués :
• ***Transformations***, qui créent un nouveau RDD en modifiant l'original par des processus tels que le mapping, le filtrage et plus encore.
• ***Les actions***, telles que les comptes, qui mesurent mais ne changent pas les données originales.

Spark est spécifiquement conçu pour être facile à utiliser et permettre aux développeurs d'écrire leurs applications rapidement en Java, Scala, Python ou R. La figure 3.7 montre qu'à partir de la deuxième itération, les données intermédiaires sont enregistrées en mémoire (RAM) où Spark peut récupérer les données en accédant à la RAM plutôt qu'au disque dur, ce qui rend l'exécution beaucoup plus rapide. Le tableau suivant (Tableau 3.2) présente quelques avantages et inconvénients de Spark.

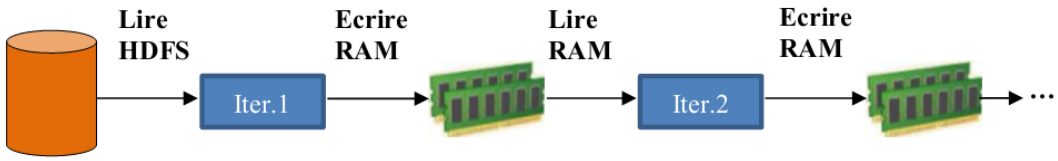

FIGURE 3.7: La lecture et l'écriture vers et à partir de la RAM dans Spark.

TABLEAU 3.2: Avantages et inconvénients d'Apache Spark

Avantages	Inconvénients
-Fournit un calcul en mémoire rapide. -Très efficace au niveau des requêtes interactives et les algorithmes itératifs. -Intégré la bibliothèque d'apprentissage automatique. -Fournit une plateforme de Spark streaming. -Très efficace dans l'analyse en temps réel en utilisant Spark streaming et Spark SQL. -Compatibilité avec n'importe quelle api Java, Scala, Python, R qui facilitent la programmation. -Traitement des problèmes liés aux graphes.	- Il consomme beaucoup de mémoire, et les problèmes liés à la consommation de mémoire ne sont pas gérés de manière conviviale. -N'a pas son système de gestion de fichiers, donc il nécessite une intégration avec Hadoop, ou autre plateforme de données.

4.1.3 Apache Spark Vs Hadoop MapReduce

Hadoop et Spark sont des projets populaires de Apache dans l'écosystème big data. Apache Spark est une amélioration sur le composant Hadoop MapReduce original de l'écosystème de Hadoop. Il y a une grande excitation autour d'Apache Spark car il offre un avantage réel dans l'interrogation interactive des données en mémoire et également dans les algorithmes itératifs d'apprentissage multi-passage. Hadoop est un framework de traitement parallèle des données qui a été traditionnellement utilisé pour exécuter des tâches de type Map/Reduce. Ce sont des opérations par lots de longue durée qui prennent des minutes ou des heures à compléter. Spark est une alternative au modèle traditionnel MapReduce qui peut être utilisé pour le traitement des données en temps réel et des requêtes interactives rapides qui se terminent en quelques secondes. Il est également basé sur le framework Hadoop. Les figures 3.8, 3.9 représentent le processus de Hadoop MapReduce et Spark. Le tableau 3.3 fournit une com-

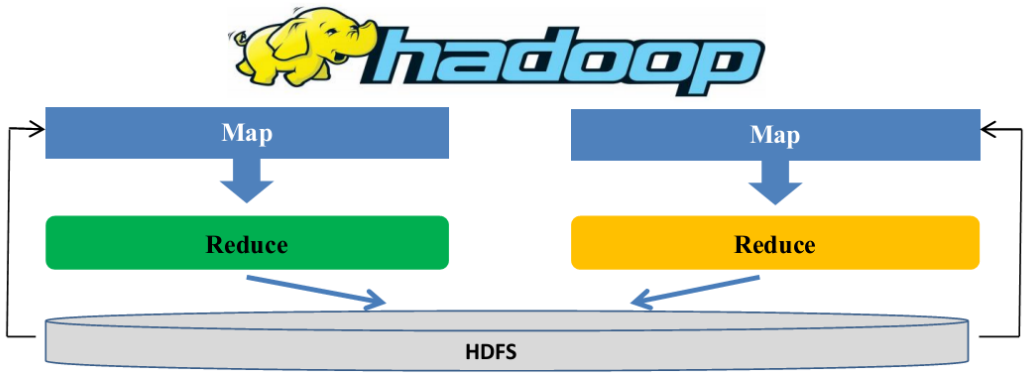

FIGURE 3.8: Processus de Hadoop MapReduce.

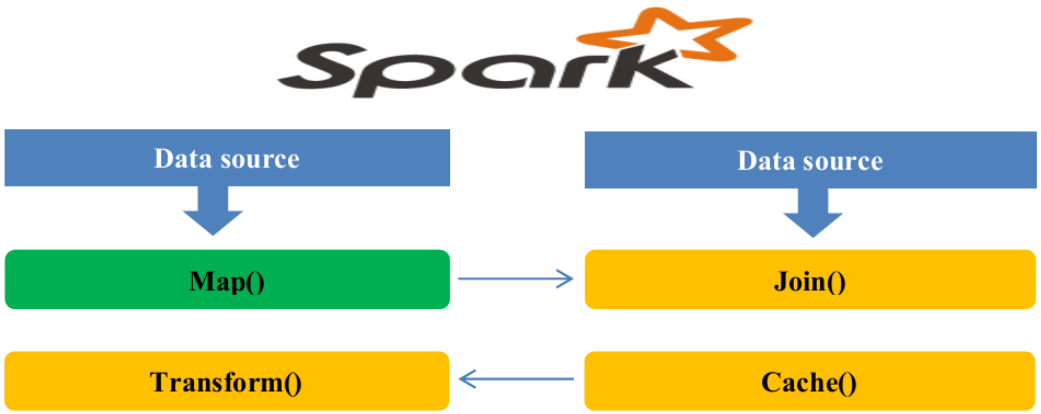

FIGURE 3.9: Processus de Spark.

paraison entre Apache Spark et Hadoop MapReduce. Dans ce tableau, nous allons couvrir la différence entre Spark et Hadoop MapReduce, comme Spark est 100x plus rapide que MapReduce. À titre d'exemple, la figure 3.10 compare Hadoop MapReduce et Spark en termes de temps d'exécution pour l'agorithme de KMeans et la régression logistique. Un autre exemple est celui de WordCount implémenté sur les deux frameworks en prenant différentes taille de fichier d'entré (Figure 3.11).

4.2 Les plateformes de streaming

Le streaming désigne le traitement en temps réel de données générées à grande vitesse à partir de sources multiples. C'est un nouveau paradigme nécessaire en raison des nouvelles scénarios de sources de génération de données (Liu et al., 2014). Il peut être appliqué au flux de données provenant des différentes sources en temps réel à savoir IoT, les capteurs, les données du marché, les applications mobiles etc.
En général, il existe deux types de systèmes de traitement de streaming :

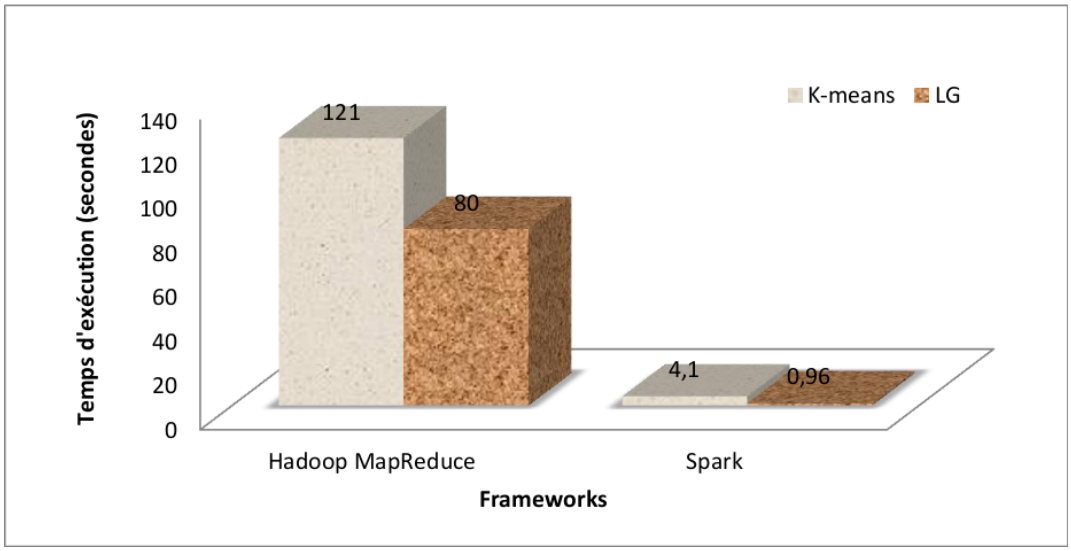

FIGURE 3.10: Temps d'exécution de KMeans sur les deux frameworks (Zaharia et al., 2012b).

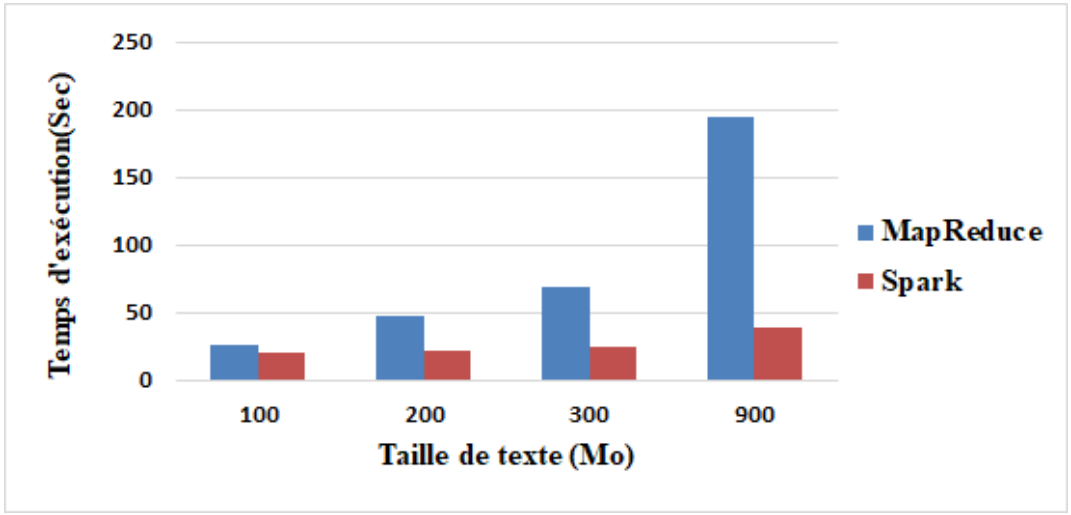

FIGURE 3.11: Temps d'exécution de WordCount.

- **les plateformes traditionnelle de streaming**, dont lesquelles nous pouvons implémenter les algorithmes de streaming en utilisant les langages de programmation traditionnelles d'une manière séquentielles.

- **les plateformes de streaming distribuées**, dont lesquelles les données sont distribuées à travers un cluster de machines, dans ce cas-là, le modèle de traitement de données se fait en utilisant le paradigme MapReduce.

TABLEAU 3.3: Spark Vs Hadoop MapReduce

	Apache Spark	Hadoop MapReduce
Définition	Un framework open source du big data. Il fournit un moteur de traitement de données plus rapide et plus général. Il est essentiellement conçu pour le calcul rapide.	Un framework open source qui trait de données structurées et non structurées qui sont stockées dans le HDFS, Hadoop MapReduce est conçu d'une manière à traiter un grand volume de données sur un cluster.
Vitesse	Spark exécute des applications dans des clusters Hadoop jusqu'à 100x plus rapide en mémoire et 10x plus rapide sur le disque. En réduisant le nombre de cycles de lecture/écriture sur le disque.	MapReduce lit et écrit les données à partir du disque ce qui ralentit la vitesse du traitement.
Difficulté	Spark facilite la programmation car il constitue un ensemble d'opérateurs de haut niveau avec les RDDs.	Dans le MapReduce les développeurs ont besoin de coder à la main chaque opération.
Analyse en temps réel	Il peut traiter de données en temps réel, c'est-à-dire des données provenant de flux d'événements en temps réel, à raison de millions d'événements par seconde. Données de Twitter par exemple ou Facebook, partage/affichage. La force de Spark réside dans sa capacité à traiter efficacement le flux en temps réel.	MapReduce échoue quand il s'agit du traitement de données en temps réel car il a été conçu pour effectuer le traitement par lots sur des quantités volumineuses de données.
Utilisation	Spark est plus facile à utiliser, son abstraction (RDD) permet à l'utilisateur de traiter des données en utilisant des opérateurs de haut niveau. Il fournit des API riches en Java, Scala, Python et R.	MapReduce est complexe; nous devons gérer les API de bas niveau pour traiter les données, ce qui nécessite beaucoup du codage manuel.
Langage de développement	Spark est développé en Scala.	Hadoop MapReduce est développé en Java.
Mise en cache	Spark peut mettre en mémoire les données pour d'autres itérations, ce qui améliora ses performances.	MapReduce ne peut pas mettre en mémoire les données pour des besoins futurs.

4.2.1 Spark streaming

Spark Streaming (Zaharia et al., 2013) est une extension du l'API Spark. Il s'agit d'un framework de traitement de flux qui permet un traitement à haut débit, évolutif et tolérant aux pannes pour le flux de données en temps réel. Pour ingérer des données dans le système, de nombreuses sources peuvent être utilisées, à savoir Kafka, Twitter, Flume, Kinesis ou TCP sockets. Ces données peuvent ensuite être traitées à l'aide des algorithmes et des fonctions de haut niveau, comme Map, Reduce et autres. D'autres extensions de Spark comme Spark GraphX ou Spark MLlib peuvent être appliquées sur ces flux de données. Une fois les données traitées, elles peuvent être envoyées vers des systèmes de fichiers, des tableaux de bord et des bases de données en temps réel (Figure 3.12)[3].



FIGURE 3.12: Architecture de Spark streaming.

Concrètement, Spark streaming ingère le flux continu de données d'entrée divisés en micro-batchs. Les transformations à base de RDD sont appliquées aux bachts par "Spark engine" pour générer le flux de résultat final (Figure 3.13).

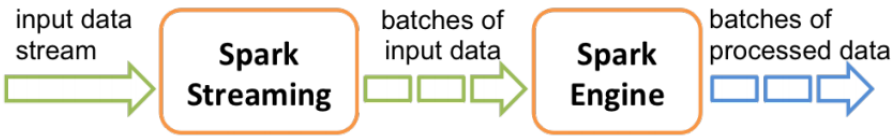

FIGURE 3.13: Flux de données de Spark streaming.

Cette structure permet d'utiliser le même code pour le traitement des données hors ligne et l'analyse d'un flux continu de données. Spark streaming est basé sur le concept de **DStreams** (Discretized Streams), qui est exprimé par une séquence de RDDs.

- **Discretized stream (le flux discret)**

Le flux discrets, appelés également Dstreams, est une abstraction d'un flux continu de données fourni par Spark Streaming. Les DStreams sont en fait des séquences de RDDs, chacun contenant de données à partir d'un intervalle du temps spécifique (Figure 3.14).

[3]https://spark.apache.org/docs/latest/streaming-programming-guide.html

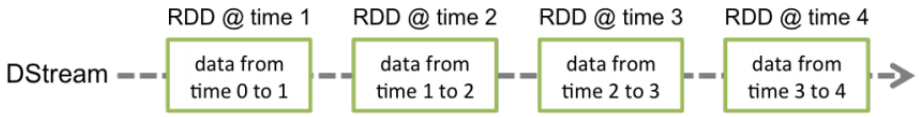

FIGURE 3.14: DStream.

Les opérations (actions et transformations) sur un DStream sont appliquées concrètement à tous les RDDs qui composent le flux. Spark Streaming intègre deux catégories de sources de streaming :
• **Sources de base**: sources directement disponibles dans l'API StreamingContext, comme les sockets et systèmes de fichiers (HDFS/S3...).
• **Sources avancées**: sources disponibles via des bibliothèques externes, comme Flume, Kafka, Twitter et Kinesis. Il est également possible d'implémenter une source de données sur mesure en étendant la classe Receiver.

- **Déploiement du cluster**

Pour tirer pleinement parti du potentiel d'Apache Spark, il faut l'exécuter dans un cluster. Pour ce faire, un gestionnaire de cluster et un système de stockages distribué sont nécessaires. Spark peut utiliser une grande variété de systèmes de stockage distribués, y compris HDFS, MapR File System (MapR-FS), Cassandra, Amazon S3,... etc. Il permet également de mettre en œuvre une solution personnalisée. Spark peut se connecter à plusieurs types de gestionnaires de clusters. Le gestionnaire de cluster est responsable de l'allocation des ressources pour les applications à travers le cluster. Après la mise en service du cluster, Spark acquiert des exécuteurs sur les nœuds du cluster, qui sont des processus qui exécutent des calculs et stockent des données. Lorsque les exécuteurs sont prêts, il leur envoie le code. Enfin, SparkContext envoie différentes tâches aux exécuteurs.
La dernière version de Spark (version 2.1.0) prend en charge trois gestionnaires de clusters différents.

1. **Standalone** : un simple gestionnaire de cluster intégré dans Spark.
2. **Apache Mesos** : un gestionnaire de cluster open-source développé par l'UC, Berkeley.
3. **Hadoop YARN** : le gestionnaire de ressources dans Hadoop 2.

4.2.2 Apache Storm

Apache Storm[4] est un framework open source pour le traitement de données structurés et non structurés en temps réel. Storm est un framework tolérant aux pannes qui convient à

[4]https://storm.apache.org/

l'analyse de données en temps réel, ML, le calcul séquentiel et itératif. Un programme Storm est représenté par un graphe orienté acyclique (Directed acyclic graph) (DAG). Les sommets du programme DAG représentent le transfert de données. Les nœuds du DAG sont divisés en deux types : « spouts » et « bolts ». Les spouts (ou points d'entrée) d'un programme Storm représentent les sources de données (Mazumder, 2016). Les bolts représentent les fonctions à exécuter sur les données. Noter bien que Storm répartit les bolts sur plusieurs nœuds pour traiter les données en parallèle. Storm est basé sur deux démons appelés Nimbus (dans le nœud maître) et un superviseur pour chaque nœud esclave. Nimbus supervise les nœuds esclaves et leur assignent des tâches. S'il détecte une défaillance de nœud dans le cluster, il réassigne la tâche à un autre nœud. Chaque superviseur contrôle l'exécution de ses tâches (affectées par le Nimbus). Il peut arrêter ou démarrer les spots en suivant les instructions du Nimbus. Chaque topologie soumise au cluster Storm est divisée en plusieurs tâches.

4.2.3 Apache Flink

Flink[5] est un framework open source pour le traitement des données en temps réel et en mode batch, il offre plusieurs avantages, tels que la tolérance aux pannes et le calcul à grande échelle. Le modèle de programmation de Flink est similaire à celui de MapReduce. En revanche, Flink fournit des fonctions supplémentaires de haut niveau telles que la jointure (join), le filtrage (filter) et l'agrégation de données. Flink permet un traitement itératif et un calcul en temps réel sur les données collectées en continu par différents outils comme Flume et Kafka. Il offre plusieurs API à un niveau plus abstrait permettant à l'utilisateur de lancer des calculs distribués d'une manière transparente et facile.

4.2.4 Apache Samza

Apache Samza[6] est un framework open source de traitement distribué créé par Linkedin pour résoudre divers types et les exigences de traitement de flux tels que le suivi, l'enregistrement, et l'ingestion de données pour les services en temps réel. Depuis, il a été adopté et déployé dans plusieurs projets. Samza est conçu pour traiter et assurer la persistance du système de fichiers pour les messages de grande taille. Il utilise Apache Kafka comme un agent distribué pour la messagerie et Hadoop YARN pour l'allocation et la planification des ressources distribuées. Le démon de gestion des ressources YARN est adopté par Samza pour assurer la tolérance aux pannes, l'isolation du processeur, la sécurité et la gestion des ressources dans le cluster.

Les frameworks présentés ci-dessus sont comparés selon plusieurs critères (voir le tableau 3.4). La figure 3.15 présente les dernières tendances de ces frameworks selon google trends.

[5]https://flink.apache.org/
[6]http://samza.apache.org/

TABLEAU 3.4: Spark Vs Strom vs Flink vs Samza

	Spark	Strom	Flink	Samza
Modèle de streaming	Micro-Batching	Native	Native hybride	Native streaming
Latence	Moyen (en fonction de la taille du lot)	Très bas	Bas	Bas
Partitionnement de données	Oui	Non	Non	Oui
Maturité	Élevée	Élevée	Bas	Moyen
Auto-évolutif	Oui	Non	Non	Non
Cluster manager	Hadoop YARN, Apache Mesos	Zookeeper	Hadoop YARN, Apache Mesos	YARN
Modèle de programmation	Transformation et action	Bolts	Actions de fonctions (map,groupby,..)	Mapreduce
Requête de Streaming	SparkSQL	Non	Non	Oui (Samza-SQL API)
Tolérance aux pannes	RDD	ACKs	Check pointing	Log-based
Outils d'apprentissage automatique associés	MLib, Mahout, H2O	Samoa	Flink-ML, Samoa, Mahout	Samoa
Programming languages	Java, Scala, Python and R	Java	Java	Java

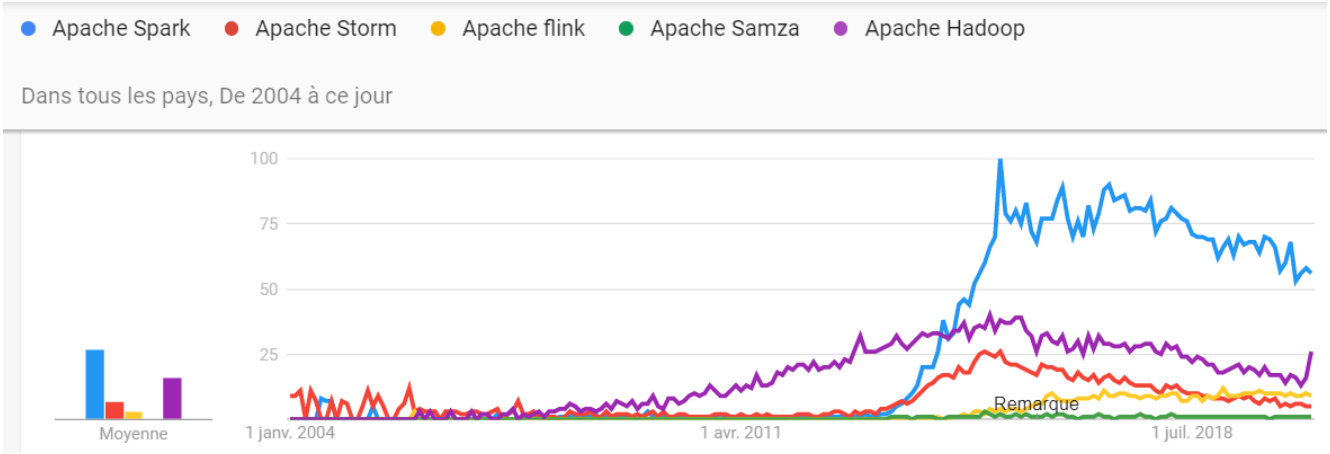

FIGURE 3.15: Tendances de recherche de Hadoop, Spark, Storm, Flink et Samza.

5 La chaine de valeur big data

Au cours des deux dernières décennies, notre ère peut être décrite comme une ère de méga-données où les données numériques prennent de plus en plus d'importance dans de nombreux domaines comme les soins de santé, la science, la technologie et la société. Une grande quantité de données a été capturée et générée à partir de plusieurs domaines, de plusieurs sources telles que des machines de streaming, des instruments à haut débit, des réseaux de capteurs, des applications mobiles et dans chaque domaine en particulier dans les soins de santé, ce volume de données élevé représente le big data. La collection, le stockage, le traitement, la visualisation et l'extraction de connaissances à partir de ces données volumineuses devenus un défi en utilisant des méthodes traditionnelles. L'un des défis technologiques le plus importants de l'analyse du big data consiste à explorer les moyens disponibles pour obtenir des informations précieuses et efficaces. Les Vs du big data défient les approches techniques existantes et nécessitent de nouvelles formes de traitement de données pour permettre une prise de décision, l'extraction de connaissance et une amélioration et l'optimisation des processus. Ainsi, ces données nécessitent une gestion et une analyse appropriée afin d'obtenir des informations significatives. La figure 3.16 représente un schéma de la chaine de valeur big data divisée en trois phase principales : l'ingestion, l'analyse et la visualisation de données (Curry, Freitas, et al., 2010).

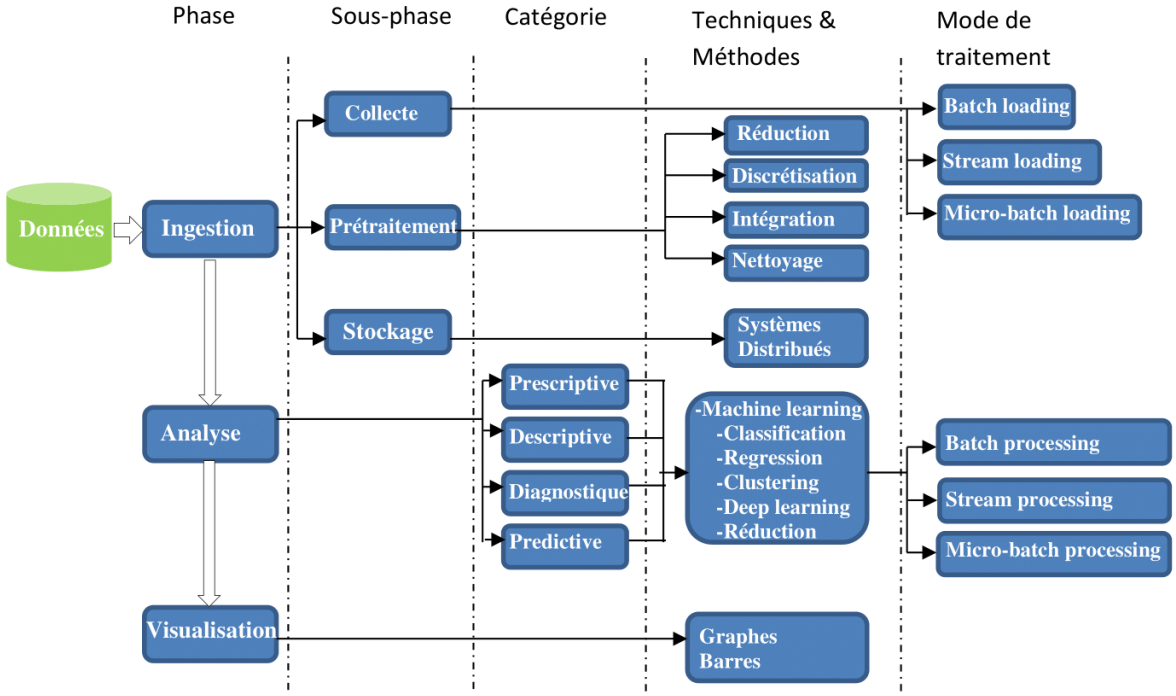

FIGURE 3.16: La chaine de valeur big data.

5.1 Ingestion

L'ingestion de données est définie comme le processus d'importation, de transfert, de chargement et de traitement de données pour une utilisation ou un stockage ultérieur dans une base de données. Elle implique la connexion à diverses sources de données, l'extraction de données et la détection des changements dans les données. Les sous-systèmes d'ingestion de données doivent aller chercher des données à partir de diverses sources, telles que les SGBDR, les journaux Web, les journaux d'application, les données en streaming, les sites de médias sociaux, etc. Cette phase elle-même se compose de trois sous phases: la collection, le pré-traitement et le stockage.

5.1.1 La collection

Les données sont ingérées selon trois méthodes différentes : **batch** (par lots), **micro-batch** (par micro lots), **streaming** (en temps réel).

a. **Batch** : est l'exécution d'une série de tâches dans un programme sur un ordinateur sans intervention manuelle (non interactif). Il s'agit d'un mode de traitement : l'exécution d'une série de programmes chacun sur un ensemble ou "lot" d'entrées, plutôt qu'une seule entrée (qui serait plutôt une tâche personnalisée). Le chargement de données par lots est un moyen efficace pour traiter les données lorsqu'un grand nombre de transactions sont collectées au fil du temps et ne nécessitent pas des traitements en temps réel. Les données sont collectées, saisies et traitées, puis les résultats des lots sont produits avec des outils comme Hadoop et les technologies d'extraction ETL (Extract-Transform-Load).

b. **Streaming** : avec le streaming de données, les données sont immédiatement traitées à partir de flux de données entrants. Ce type d'ingestion est souvent utilisé pour l'analyse prédictive. Les outils les plus connus utilisés dans cette phase sont : **Flume**[7] et **Chukwa**[8].

c. **Micro-batch** : les données en temps réel, par contre, exigent une entrée, un traitement et une sortie de données continus. Les données doivent être traitées dans un temps ou une période réduite (ou presque en temps réel). Il s'agit du processus consistant à transférer les données vers de grands systèmes de données au fur et à mesure de leur arrivée. C'est l'intersection du mode stream et le mode batch (Figure 3.17). les outils les plus connus utilisés dans cette phase sont : **Spark** [9] et **Kafka**[10].

[7] https://flume.apache.org/
[8] http://chukwa.apache.org/
[9] https://spark.apache.org/
[10] https://kafka.apache.org/

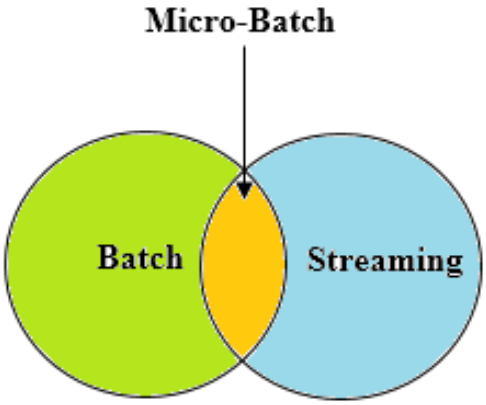

FIGURE 3.17: Batch Vs stream Vs Micro-batch.

5.1.2 Le pré-traitement

Lorsque nous parlons de données, nous pensons généralement à de grands ensembles de données avec un nombre énorme de lignes et de colonnes. Bien que ce soit un scénario probable, ce n'est pas toujours le cas, les données peuvent se présenter sous des formes très différentes. Le pré-traitement de données est l'étape au cours de laquelle les données sont transformées, ou codées, pour les amener à un état tel que la machine puisse les analyser facilement (cas de ML). En d'autres termes, les caractéristiques de données peuvent maintenant être facilement interprétées par les algorithmes. Le pré-traitement est une étape majeure dans le processus de l'analyse de données et en particulier ML. Les outils utilisés dans cette phase sont presque les mêmes que celle utilisés dans la section 5.1. La phase de pré-traitement consiste à effectuer le nettoyage, la qualification, et l'amélioration de la fiabilité de données. Pour cela, de multiples méthodes et techniques sont disponibles.

- **Nettoyage** : le nettoyage de données est le processus de préparation de données en vue de leur analyse en supprimant ou en modifiant les données qui sont incorrectes, incomplètes, non pertinentes, dupliquées ou mal formatées. Ces données ne sont généralement pas nécessaires ou utiles lorsqu'il s'agit d'analyser des données car elles peuvent entraver le processus ou fournir des résultats incorrects. Cependant, cette méthode n'est pas efficace lorsque le pourcentage des valeurs manquantes est important (Rehman et al., 2016).

- **Réduction** : la réduction de données est le processus de réduction de la quantité de capacité nécessaire pour stocker des données. Les techniques de réduction de données peuvent être appliquées pour obtenir une représentation réduite de l'ensemble de données qui est moins volumineuse mais qui contient toujours des informations pertinentes. La réduction de données peut être réalisée de plusieurs façons. Les principales sont la réduction du big data (Santhanam and Padmavathi, 2014), la compression et décompression de données (Zhai, Ong, and Tsang, 2014).

- **Transformation** : la transformation de données est le processus qui consiste à apporter quelques ou plusieurs modifications aux données pour les rendre utiles pour une analyse plus approfondie en passant par une série d'étapes spécifiques (García et al., 2016).
- **Intégration** : L'intégration de données est la collecte de données provenant de différentes sources et la combinaison de ces données en un ensemble unifié (Ahamed, Ramkumar, and Hariharan, 2014). Elle peut facilement être utilisée pour tirer des informations significatives et précieuses de ces données.
- **Discrétisation** : la discrétisation de données consiste à diviser la plage d'attributs continus en sous intervalles (García et al., 2016).

5.1.3 Le stockage

Elle consiste à persister une énorme quantité de données qui sont collectées auprès de sources multiples. Il est important de mentionner que le système de stockage a un impact considérable sur l'extensibilité et les performances de la chaine de valeur big data. Ainsi, cette étape doit fournir un espace de stockage fiable et permettre un accès puissant pour l'analyse de données. En général, les systèmes qui fournissent un stockage du big data repose autour de quatre piliers (Saggi and Jain, 2018) :

- **Modèles de stockage** : il est le pilier de tout système s'appuyant sur le big data. Comme tout système de stockage peut prendre en charge trois modes de stockage, à savoir le stockage en mode blocs, le stockage en mode fichiers ou le stockage en mode objet. L'architecture big data repose sur le mode en blocs.
- **Modèles de données** : comme indiqué précédemment, les bases de données traditionnelles ne peuvent pas répondre aux critères du big data (le stockage et le traitement d'énormes quantités de données non structurées). Pour cela, le stockage distribué et les bases de données NoSQL sont souvent adoptées pour manipuler les données persistantes et des bases de données hétérogènes. Les bases de données NoSQL distribuées (Storey and Song, 2017) sont regroupées en quatre familles : clé/valeurs, orientées colonne, orientées document et les bases de données orientées graphe.
- **Infrastructure de stockage** : l'infrastructure de stockage matérielle pour les systèmes big data peut être considérée sous différents aspects tels que les dispositifs de stockage, l'infrastructure de stockage réseau et la virtualisation du stockage.
- **Infrastructure de traitement distribuée** : en informatique, les systèmes distribués consistent à partager les données et les tâches (calcul et traitement) sur plusieurs nœuds (Dutta, 2017). Alors que l'utilisateur perçoit le système comme une seule unité. Il est important de noter que tous les aspects ci-dessus permettent de faciliter la qualité de données et les requêtes de données en faveur de la phase d'analyse.

5.2 Analyse

Il s'agit de l'étape la plus importante de la chaîne de valeur big data. Elle permet d'extraire des informations précieuses à partir de données stockées pour soutenir un large éventail de décisions de l'entreprise allant des décisions opérationnelles aux décisions stratégiques. L'élément le plus connu de l'analyse de données est le business intelligence ou informatique décisionnelle, qui fournit des vues historiques, actuelles et prédictives des opérations commerciales, telles que le reporting et la gestion des performances de l'entreprise.

Les différents formes d'analyse big data peuvent être divisé en quatre catégorie (Saggi and Jain, 2018) :

1. **Analyse descriptive** : la principale caractéristique de l'analyse descriptive est qu'il s'agit d'une analyse basée sur des données (historiques) passées. Dans la pratique, elle est généralement communiquée sous forme de graphiques et de tableaux de bord.

2. **Analyse prédictive** : l'analyse prédictive est le domaine spécial de l'analyse qui examine les données de passé et tente de prédire ce qui va se passer à l'avenir, et c'est là que se trouvent à la fois les opportunités et les problèmes.

3. **Analyse prescriptives** : ce type d'analyse révèle les décisions à prendre. C'est le type d'analyse le plus précieux et il conduit généralement à des règles et des recommandations pour les étapes suivantes.

4. **Analyse diagnostique** : elle permet une identification des principales causes menant à la réalisation d'une performance en examinant le passé.

Les différentes formes d'analyse qui prouvent leurs intérêts, nous permet d'obtenir de nombreuses valeurs. L'extraction de connaissance à partir de ces valeurs peut se faire à l'aide de différentes méthodes à savoir ML.

Après cette étape, les données sont prêtes d'être visualisées au moyen de tableaux de bord significatifs où à être partagées.

5.3 Visualisation

La visualisation de données est la représentation d'informations sous forme de tableau, de diagramme, d'image, etc. Ceux-ci sont créés comme la représentation visuelle de l'information. Elle est très utile aux décideurs de voir les analyses présentées visuellement, afin qu'ils puissent travailler en conséquence et saisir le concept difficile ou identifier de nouveaux schémas (Chen, 2019).

6 Applications

Dans cette sous-section, nous examinons l'utilisation des frameworks étudiés dans cinq applications du monde réel, notamment les banque, finances et sécurité, l'analyse des réseaux sociaux et les villes intelligentes (smart cites).

6.1 Banque, finances et sécurité

Big data est un grande source d'information pour les analystes de risques. Afin d'évaluer les risques et pour obtenir un plan d'affaires efficace, il est nécessaire d'avoir une vision complète du marché. Les algorithmes de ML avec l'écosystème Hadoop peuvent donner aux banques et aux différentes institutions financières la possibilité d'obtenir des informations douteuses sur les données stockées.

Pour tester si le comportement des transactions boursières peut être détecté en utilisant une technique de traitement de données sur un indice synthétique de flux de capitaux, les auteurs dans (Yu et al., 2018) ont proposé une architecture logique et globale et un cadre fonctionnel du système d'analyse. En outre, un prototype de système développé a été testé, la manière dont les principales composantes sont mises en œuvre est également illustrée. Les expériences montrent que le prototype du système est capable de réaliser un meilleur compromis en ce qui concerne la précision, la disponibilité et la flexibilité. Zhihao (Peng, 2019) a proposé une approche solide basée sur Cloudera et Hadoop pour effectuer des analyses pour toute échelle et tout type de données. Les stocks américains sont sélectionnés et sont analysés pour prévoir les gains quotidiens basé sur les données en temps réel de Yahoo Finance. Hadoop est fourni pour traiter les données volumineuses par le stockage et le traitement distribués. Les stocks du marché boursier américain sont choisis et leurs données de gains quotidiens sont divisées en un ensemble de données d'apprentissage et de test pour prévoir les stocks avec des gains quotidiens élevés grâce au module de ML de Spark.

L'un des problèmes les plus débattus est l'intrusion dans les réseaux. Cette intrusion peut provoquer une fuite de données classifiées qui peut poser un énorme problème. Shan Suthaharan a fourni une analyse approfondie du problème à l'aide de données du big data et les techniques de ML pour la classification des réseaux (Suthaharan, 2014). L'tilisation du SVM et d'autres algorithmes de ML peut fournir un environnement sécurisé pour l'informatique générale sur le big data a été effectué dans (Xu et al., 2015). Ce système distribué peut présenter une faille dans les données, ce qui peut entraîner des problèmes de confidentialité. ML offre plusieurs approches pour trouver des solutions à ces problèmes. Apache Mahout supporte principalement deux paradigmes de ML qui sont l'apprentissage supervisé et l'apprentissage non supervisé. Nayak et ses collaborateurs (Nayak, Pai, and Pai, 2016) ont utilisé les réseaux de neurones pour prédire le prix de la bourse à l'aide de Hadoop et de MapReduce en construisant deux modèles : l'un pour la prévision quotidienne basée sur la combinaison entre le prix

historique et l'analyse du sentiment des tweets, la précision a atteint jusqu'à 70 %. Le second modèle constate que la corrélation de la tendance boursière entre deux mois et le résultat de la corrélation était très faible. Mukesh et Rohini (Rohini, 2016) ont utilisé les réseaux de neurones et le Hadoop-HDFS pour comparer deux algorithmes et ont prouvé que l'algorithme des moindres carrés était meilleur que l'algorithme sigmoïde.

6.2 Réseaux sociaux

Avec l'émergence des technologies et des smartphones, nous sommes arrivés à un point où nous voyons que chacun d'entre nous est connecté à l'une des plateformes de médias sociaux, il peut s'agir de Facebook, Twitter, Instagram, des blogs Web, etc. Les médias sociaux sont un domaine où des pétaoctets de données sont générés quotidiennement. L'analyse de données peut être effectuée sur toutes les données collectées sur les réseaux de médias sociaux. La plupart des chercheurs préfèrent collecter des données sur Twitter car il dispose de données provenant de tous les domaines. Plusieurs travaux d'analyse des sentiments ont été réalisés en utilisant des techniques de ML.

Cunha et al. (Cunha, Silva, and Antunes, 2015) ont proposé une architecture fonctionnelle et générique en se basant sur le framework Hadoop et Mahout pour la manipulation de ML. Le stockage et l'analyse du big data peuvent être utilisées dans différents scénarios. Pour démontrer sa valeur, ils ont montré ses caractéristiques, ses avantages et des applications sur les données sanitaires de Twitter. Ils ont conclu que le big social data sur la santé peuvent générer des informations importantes et précieuses à la fois pour les praticiens que pour les utilisateurs. Le modèle de ML BN a été utilisé pour classer les sentiments de données de Twitter par Das et al (Das et al., 2017), ils ont classé les tweets en tweets négatifs, tweets neutres et tweets positifs. Comme l'analyse et le traitement du big social data sont difficiles avec les méthodes traditionnelles, Sehgal et Agarwal (Sehgal and Agarwal, 2018) ont utilisé le framework Hadoop pour que le processus de traitement soit simple et rapide. Ils ont pris en considération les critiques de films sur Twitter. De plus, l'expérience du client est l'une des préoccupations majeures des compagnies aériennes. Twitter est l'une des plateformes de médias sociaux les plus populaires où les voyageurs de vol partagent leurs réactions envers les vols sous forme des tweets. Dans (Kumar and Zymbler, 2019) les auteurs ont proposé une approche de ML pour analyser les tweets et améliorer l'expérience du client.

6.3 Villes intelligentes

L'émergence des villes intelligentes vise à atténuer les défis posés par le développement continue de l'urbanisation et l'augmentation de la densité de population dans les villes. Pour faire face à ces défis, les gouvernements et les décideurs prennent des projets de villes intelligentes visant une croissance économique durable et une meilleure qualité de vie pour les habitants et

les visiteurs. Les technologies de l'information et de la communication (TIC) sont une technologie clé pour rendre les villes plus intelligentes. De nombreuses nouvelles applications pour les villes intelligentes ont été proposées.

Les recherches au niveau des villes intelligentes utilisant l'analyse du big data devient de plus en plus important. Rahman et al. ont proposé un système de prévision permettant de prévoir la quantité d'électricité requis par les villes à un taux proche de la consommation d'électricité dans les États-Unis (Rahman, Esmailpour, and Zhao, 2016). Le document de Arfat et al. (Arfat et al., 2017) propose un système informatique mobile qui permet de rendre les villes plus intelligentes en améliorant les informations sur la mobilité grâce aux technologies big data, fog et le cloud. Le système comprend une application mobile, un système d'analyse du big data basé sur le cloud et une plateforme middleware. L'architecture du système et les technologies qui le composent sont également décrites dans le cas de l'utilisation d'une application mobile.

Le travail de (Khan et al., 2015) a démontré un prototype utilisant Hadoop et Spark pour analyser et comparer les données libres de Bristol, qui sont des données relatives à la qualité de vie dans les villes, principalement sur l'économie, la sécurité, la criminalité et l'emploi. Un autre prototype de démonstration a été proposé par (Silipo and Winters, 2013) pour prévoir la consommation d'énergie pour des groupes spécifiques de compteurs. Le prototype a utilisé l'analyse prédictive de séries temporelles pour créer des modèles prédictifs. Un modèle de prédiction du comportement de l'utilisateur basé sur la programmation parallèle MapReduce et les réseaux de neurones à rétropropagation a été proposé par Xu et al. (Xu et al., 2016), ce qui a permis l'intellectualisation de la maison intelligente traditionnel, ainsi l'amélioration des performances interactives et l'expérience de l'utilisateur du système de maison intelligente de manière significative.

7 Conclusion

L'analyse du big data comprend le traitement de données volumineuses et variés. L'extraction de connaissance à partir de données vastes offre des possibilités importantes aux différents domaines. Les méthodes traditionnelles ne sont pas assez efficaces et évolutives en termes de calcul lorsqu'il s'agit d'analyser les données à grandes échelles. Dans ce chapitre, nous avons rappelé la définition du big data et les concepts associés. Nous avons également présenté les technologies et les plateformes big data existantes et les plus utilisé. Les trois phases de la chaine de valeurs big data ainsi que les outils permettant d'accompagner chaque phase ont été explorés. Il a été souligné que l'environnement informatique distribué est une plateforme appropriée pour l'analyse de données volumineuses qui utilisent des méthodes très efficaces pour le traitement et la gestion de données tout en réduisant les coûts. Finalement, nous passons en revue sur les cas d'utilisations des outils big data dans les différents domaines à savoir

la santé, la banque, la finance, la sécurité, les réseaux sociaux et les villes intelligente. Pour faire face aux défis du big data, beaucoup de travaux ont été réalisés. En conséquence, différents types de distributions et de technologies ont été développés. Le choix de la technologie convenable reste un défi. Dans le chapitre suivant nous présentant une nouvelle architecture pour la détection des maladies en temps réel en se basant sur les nouvelles technologies du big data, en particulier Spark streaming.

4 Nouvelle architecture pour le suivi médical en temps réel

1 Introduction

Au cours des deux dernières décennies, notre époque peut être décrite comme celle de données massives, où les données numériques deviennent de plus en plus importantes dans de nombreux domaines comme les soins de santé, la science, la technologie et la société. Une grande quantité de données a été saisie et générée à partir de multiples domaines, de multiples sources à savoir les machines de diffusion en continu, les instruments à haut débit, les réseaux de capteurs, les applications mobiles et à partir de chaque domaine, en particulier dans le domaine de la santé, ce grand volume de données représente big data. Le stockage, le traitement, la visualisation et l'extraction de connaissances à travers ces types de données volumineuses et variées est devenu un défi en utilisant des outils technologiques traditionnelles inadéquats. L'un des défis technologiques le plus important de l'analyse du big data consiste à explorer les moyens d'obtenir efficacement des informations précieuses pour différents types d'utilisateurs. Actuellement, les différentes formes de sources de données sur les soins de santé sont collectées dans des environnements cliniques et non cliniques, où les dossiers numériques de données médicales d'un patient sont les données les plus importantes.

Dans ce chapitre nous présentons une nouvelle architecture pour le suivi de l'état de santé en temps réel en utilisant les technologies big data, ML et IoT.

2 Revue de la littérature

De nos jours, l'analyse du big data, en particulier l'analyse de données des soins de santé, est devenue un défi important pour un grand nombre de domaines de recherche tels que DM, ML avec l'énorme quantité de données sur les soins de santé de plus en plus nombreuses, ainsi que les informations potentielles qu'elles contiennent. L'évolution de la science et de la

technologie dans le domaine des soins de santé a fait une percée significative au niveau de la collection de données. Dans le secteur des soins de santé, les données sont collectées par trois principaux types de données numériques telles que les dossiers cliniques, les dossiers de recherche en santé et les dossiers d'exploration des organisations. Le rapport de (Sun and Reddy, 2013) donne un bref aperçu des sources de données sur les soins de santé. L'extraction de connaissances à partir de cette quantité de données distribuées, importantes et variées est devenue une tâche difficile en utilisant les techniques traditionnelles de DM qui aident à traiter les données et à les transformer en informations utiles. De nombreux systèmes de prédiction et de recommandation ont été étudiés dans le domaine des soins de santé. Dans (Masethe and Masethe, 2014b), une expérience a été réalisée pour la prédiction de crise cardiaque avec une comparaison afin de trouver la meilleure méthode de prédiction. Une classification du cancer du sein est effectuée en utilisant un modèle de réseau de neurones génétiquement optimisé dans (Bhardwaj and Tiwari, 2015b). D'autres techniques de DM et de recherche d'informations ont été proposées dans (Tomar and Agarwal, 2013b; Herland, Khoshgoftaar, and Wald, 2014a).

L'analyse des soins de santé a été étudiée dans de nombreux systèmes tels que la prévision et la prévention des épidémies, les systèmes de recommandation sanitaire, la prise de décision médicale afin d'améliorer la qualité des soins, la prise en charge, la réduction des coûts et l'augmentation de l'efficacité. K-means clustering basé sur le cloud et le modèle MapReduce a été proposé dans (Rallapalli, Gondkar, and Rao, 2016), leur approche utilise les données de santé sur le cloud pour le clustering. Un framework de gestion des dossiers médicaux électroniques de santé et des dossiers personnels distribués sur le web a été proposé dans (Sarkar et al., 2016), ce framework est basé sur Hadoop et HBase. Dans (Sampath et al., 2017a), la prédiction du diabète sucré et du type de traitement à adopter est réalisée en utilisant l'algorithme d'analyse prédictive et l'environnement Hadoop MapReduce. Un système de soin intelligent basé sur Hadoop a été proposé dans (Rathore et al., 2017) qui illustre le partage contextuel de données volumineuses sur l'internet des objets (IoT) entre tous les appareils d'un système de santé. Le système proposé adopte une architecture de réseau avec des capacités de traitement améliorées pour la collecte de données générées par différents appareils connectés. Les données collectées sont transmises à un bâtiment intelligent. Une analyse en temps réel axée sur les dossiers médicaux électroniques produits par de nombreuses sources telles que les appareils médicaux et les applications mobiles est décrite dans (Basco and Senthilkumar, 2017). Le framework proposé combinait Hadoop, MongoDB et une technique de traitement improvisée destinée à améliorer les résultats du traitement des dossiers des patients.

La plupart des solutions d'analyse des soins de santé se sont principalement concentrées sur Hadoop (Yadranjiaghdam, Pool, and Tabrizi, 2016), il peut traiter un volume important et avec des sources de données diverses dans le cas d'un calcul par lots. Hadoop serait limité pour le calcul en temps réel, alors que Spark est plus rapide que Hadoop et il est plus performant, en particulier dans les problèmes de ML itératif (Hazarika, Ram, and Jain, 2017). Hadoop et

Spark sont tous deux des projets Apache et les outils les plus populaires dans l'écosystème big data, avec une grande excitation autour de Spark.

D'autre part, un certain nombre d'algorithmes de ML évolutifs sont développés pour surmonter les différents problèmes liés à l'analyse du big data. Dans (Rallapalli and Suryakanthi, 2016) un modèle prédictif lié au risque du diabète est réalisé à l'aide de l'algorithme RF évolutif. L'utilisation de LG en ligne pour la détection des URL de phishing est abordée dans (Feroz and Mengel, 2014), où Hadoop est utilisé pour le traitement de données et Mahout pour le ML. Une méthode automatisée capable de détecter des modèles anormaux pour les personnes âgées vivant seules, entrant et sortant de comportements recueillis à partir de simples capteurs équipés à domicile est décrite dans (Zhao et al., 2014), la méthode est basée sur le modèle de chaîne de Markov. Un système d'intervention d'urgence médicale en temps réel qui implique des capteurs médicaux basés sur l'IoT et déployés sur le corps humain est décrit dans (Rathore et al., 2016). Un aperçu des architectures du big data et des algorithmes de ML dans le domaine des soins de santé est fourni dans (Manogaran and Lopez, 2017).

Le ML est utilisé dans différents travaux de recherches, mais le streaming de données n'est pas traité que dans quelques travaux. Divers travaux de recherche ont été réalisés afin de dégager des informations utiles pour l'analyse de données des médias sociaux, en particulier celles de Twitter et d'autres sources pour un soin de santé efficaces. Par exemple, un système de surveillance en temps réel de la grippe et du cancer par l'exploitation de données de Twitter est décrit dans (Lee, Agrawal, and Choudhary, 2013). Un modèle d'analyse en temps réel du big médicales data est proposé dans (Akhtar, Khattak, and Lee, 2016). L'approche est illustrée par le Spark streaming et Apache Kafka qui utilisent le traitement de flux du big médicales data.

La plupart de ces travaux considèrent une source de données spécifique sur les soins de santé ou se concentrent uniquement sur l'informatique orientée vers le traitement par lots. Mais en réalité, les sources de données sur les soins de santé sont diverses et génèrent en permanence de données variées à un rythme élevé. En outre, soit qu'ils considèrent des outils puissants pour l'analyse de données tels que ML et le DM, soit qu'ils se concentrent uniquement sur le stockage et la visualisation de données. Ainsi, l'analyse en temps réel des soins de santé incluent le flux de données, la collection, le traitement en temps réel et les outils puissants de ML, de stockage de données distribuées et de l'analyse en temps réel sont nécessaires pour mettre en place un système efficace de traitement de flux de données distribuées de la santé.

Au cours des dernières décennies, la maladie cardiaque et le diabète sont la cause la plus fréquente de décès dans le monde. La détection précoce de ces maladies et un suivi continu peuvent donc réduire le taux de mortalité. En outre, la disponibilité des moniteurs de santé portables, la technologie IoT médicale adoptée dans le système des soins de santé et vu le nombre croissant de maladies des patients, ont déclenchés l'idée de tirer parti des technologies big data pour prédire l'état de santé en temps réel. La prédiction en temps réel peut réduire

le temps de présence des médecins, aider les médecins et les patients à réagir à l'avance à une maladie probable. Une autre caractéristique importante est une fois que la maladie du patient n'est pas normale, le service d'urgence doit être immédiatement averti par une technologie d'alerte pour effectuer des actions en temps réel en cas d'urgence.

3 L'architecture proposée pour le système de suivi et d'analyse de l'état de santé en temps réel

L'objectif de cette partie est de développer un système de traitement et de suivi de données en temps réel, combinant IoT médical, Kafka streaming et Spark streaming. Elle se compose d'une architecture à trois niveaux. Le premier niveau se concentre sur la collection de données. Le deuxième niveau utilise Spark MLlib pour développer le modèle de prédiction des maladies. Le troisième niveau utilise Apache Cassandra pour stocker l'énorme volume de données. Dans un premier lieu, nous avons développé une architecture générale (Figure 4.1). Dans un deux-

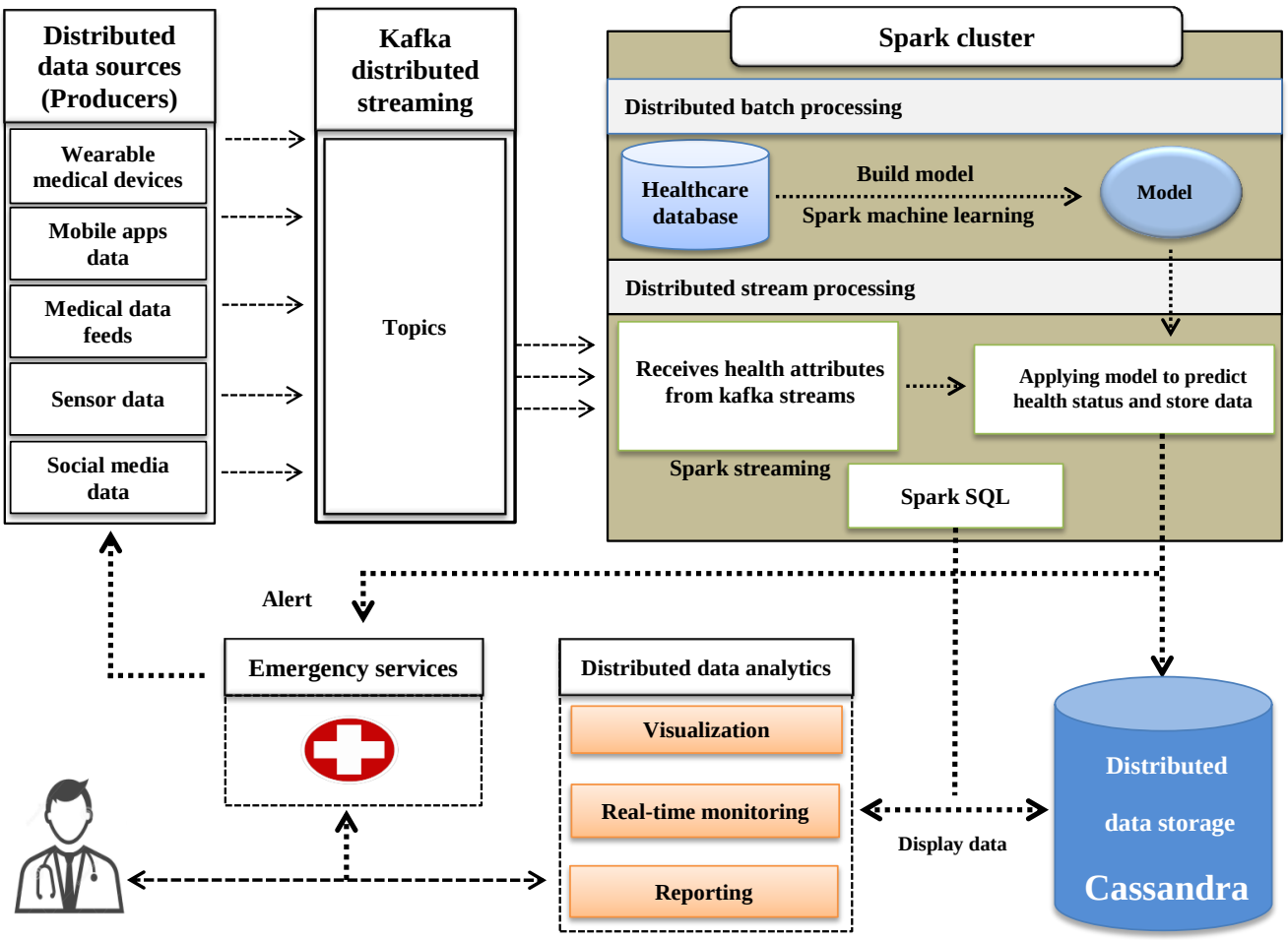

FIGURE 4.1: Architecture générale du système proposé.

ième lieu, nous avons intégré le Twitter dans notre système agissant comme les données de IoT.

La figure 4.2 montre l'architecture du système proposé avec Twitter. Tout d'abord, les attributs liés à la santé sont envoyés à l'application de Spark streaming par le biais du Kafka streaming, où le traitement est effectué. Spark streaming reçoit des attributs de santé de Kafka avec le nom de l'utilisateur de Twitter et appliquer le modèle de ML pour prédire l'état de santé. Ensuite, un message approprié est renvoyé à l'utilisateur en se basant sur le nom de l'utilisateur de Twitter, les résultats ainsi que le flux de données seront stockés dans une base de données distribuée pour une analyse historiques de données et le suivi en temps réel. Ce système a certaines caractéristiques qui le distinguent des autres systèmes traditionnels. L'idée principale ici est qu'il est nécessaire de disposer de méthodes pour gérer et analyser des milliers de messages provenant de sources différentes chaque seconde dans une courte période de temps. En outre, le système doit être indépendant du volume de données importé.

Nous remarquons que la plupart des approches proposées dans la littérature sont utiles pour prévoir l'état de santé, mais la gestion de flux de données en temps réel, le stockage de données, les interactions avec les utilisateurs et la visualisation de données ne sont pas couverts. Avec les populations élevées, couvrir tous les patients par les médecins disponibles est un défi. D'autre part, l'un des défis technologiques le plus important de l'analyse du big data est l'exploration des moyens pour obtenir efficacement des informations précieuses pour différents types d'utilisateurs et l'extraction de connaissances à partir du big data générées par tous les utilisateurs. En se basant sur l'architecture du système proposé, les données des moniteurs (IoT) seront analysées en temps réel et envoyer une alerte aux prestataires des soins afin qu'ils connaissent instantanément les changements de l'état d'un patient. Les données seront actualisées automatiquement en fixant des intervalles de temps. Les sous-sections suivantes donnent des informations détaillées de cette architecture.

3.1 Collection

3.1.1 Source de données

L'Internet des objets (Internet of Things, IoT) est un réseau d'appareils physiques et d'autres éléments, intégrant des appareils électroniques, des vêtements intelligents, des logiciels et des applications intelligentes, des capteurs et une connectivité réseau afin qu'ils puissent collecter et échanger des données entre eux ou avec les systèmes de centres de données. Avec la disponibilité de moniteurs de santé portables dans de nombreux endroits, les données générées par ces appareils sont volumineuses et de nature aléatoire. Elles doivent être analysées à l'aide d'un système d'analyse du big data afin de comprendre les modèles comportementaux des utilisateurs ou d'extraire les informations cruciales. 40 % des technologies liées à l'IoT seront liées à la santé plus que tout autre domaine (Bauer H, Patel M, Veira J, 2017). La convergence de la médecine et les technologies de l'information tel que l'informatique médicale transformera les soins de santé tel que nous les connaissons, en réduisant les inefficacités, en limitant les

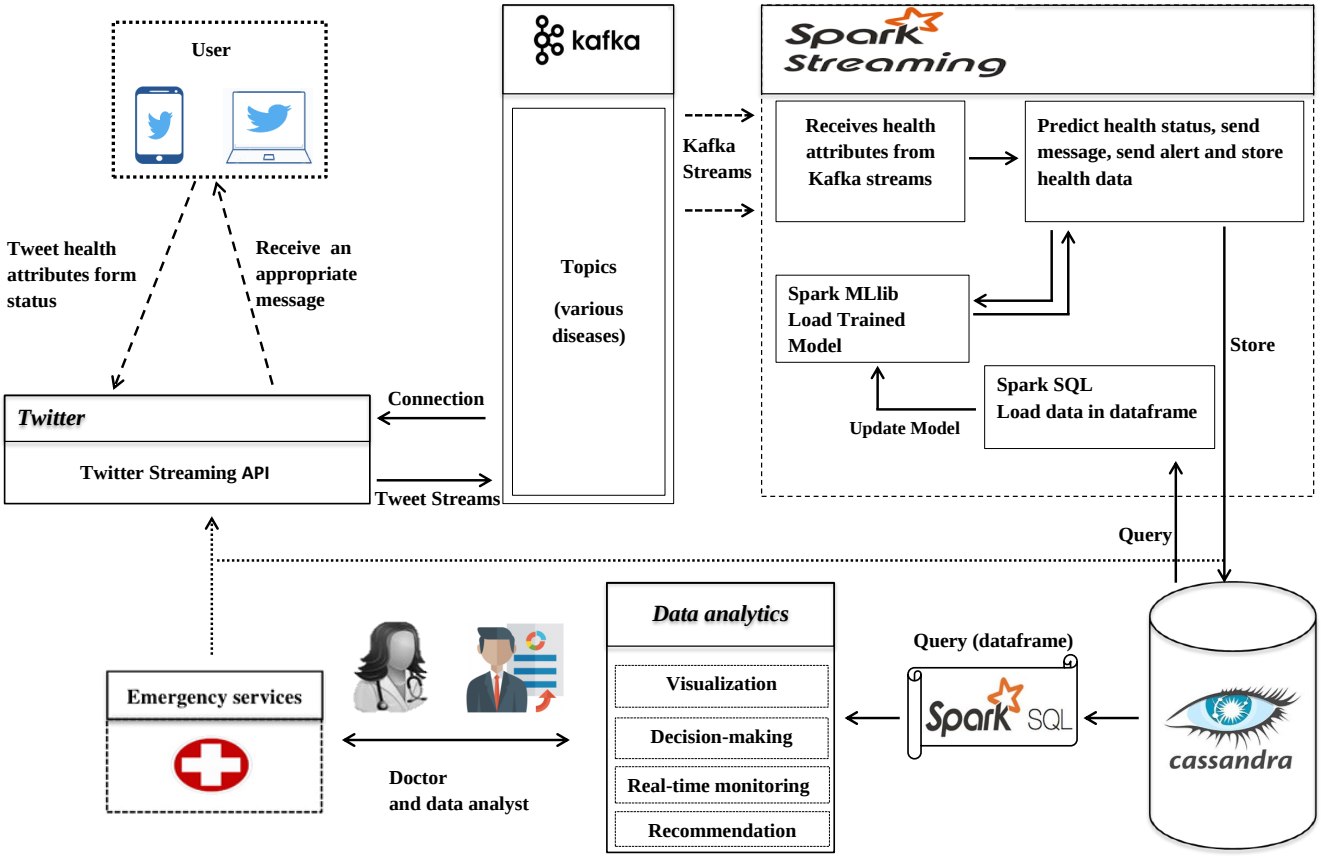

FIGURE 4.2: Architecture du système proposé avec Twitter.

coûts et en sauvant des vies. La surveillance en temps réel via l'IoT peut sauver des vies en cas d'urgence médicale comme la maladie cardiaques, le diabète et de nombreuses autres maladies chroniques. De nombreuses sources liées à la santé sont désormais disponibles et permettent de suivre en permanence les indicateurs de santé. La figure 4.3 montre l'organigramme opérationnel du système proposé avec différentes sources de données.

L'évolution de la technologie a permis d'accroître la disponibilité et l'utilisation des smartphones, des ordinateurs personnels et bien d'autres encore. En raison de la croissance rapide des réseaux sociaux et leur rôle dans la vie quotidienne de millions de personnes à travers le monde, le nombre et la quantité de données générées dans ces médias augmente de manière exponentielle. Twitter est l'un des réseaux sociaux les plus populaires, c'est un site de microblogging et de mises à jour en 140 caractères par le biais de messages ultra courts appelés tweets dans lesquels les personnes peuvent partager leurs opinions, en envoyant et en recevant des messages. Twitter devient le moyen le plus rapide d'obtenir des informations en temps réel dans le monde entier, il offre un accès gratuit et public à des flux de tweets. Comme il est pris en charge par les smartphones et le Spark streaming, Twitter peut être utilisé comme un moyen de communication efficace. Au lieu d'utiliser Twitter, il est possible de générer des données volumineuses en continu à partir d'autres sources de données, comme les dispositifs IoT médicale.

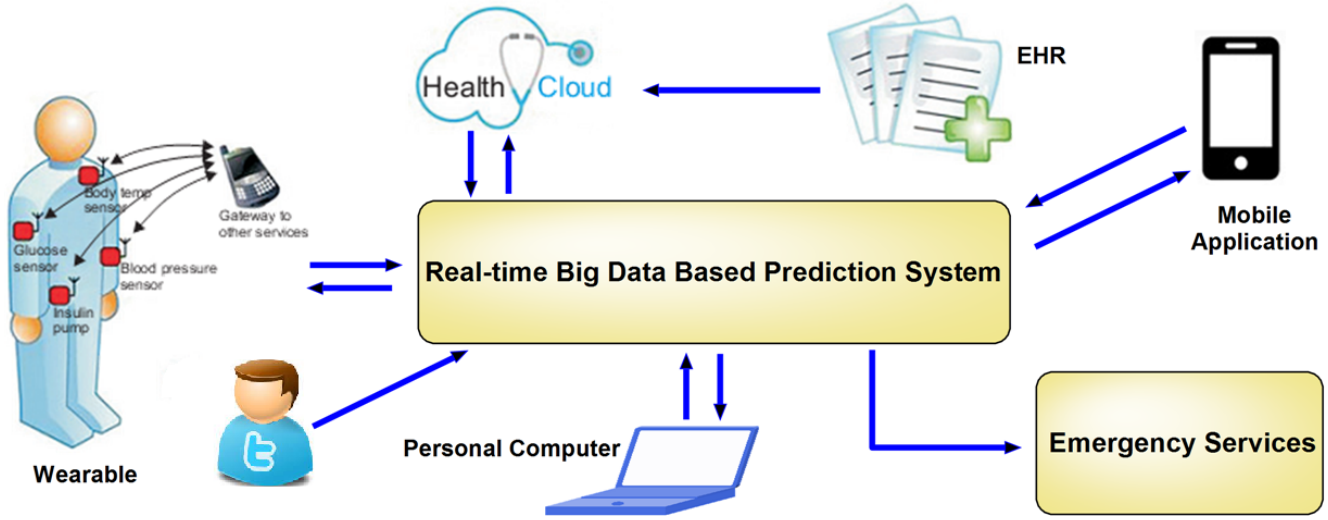

FIGURE 4.3: Diagramme opérationnel du système proposé.

Dans la deuxième architecture, nous capturons le flux de tweets à l'aide d'un mot-clé spécifique lié à la maladie (cardiaque par exemple) suivie des attributs dans le même format que le vecteur caractéristiques dans la base de test séparés par un espace (#rtbigdhdsparkt 63 1 1 145 233 1 2 150 0 2.3 3 0 6), il s'agit du premier module de ce système. Pour une demande d'authentification sur Twitter, un compte Twitter et une application Twitter doivent être créés et définir le niveau d'accès en lecture et en écriture. De même, nous avons besoin du 'Consumer Key' et 'Secrete key'. Nous avons également besoin du 'Access token' et du 'Access Token Secret' pour effectuer des demandes d'API de streaming sur vos propres comptes sans avoir à partager le mot de passe. Ensuite, le flux Twitter est ingéré dans Spark par l'intermédiaire de Kafka. Pour équilibrer la charge de flux entrant dans notre système, nous utilisons Kafka, qui est plus approprié au traitement de données de diffusion en temps réel. Dans ce module, les données en temps réel sont diffusées en continu à partir de Twitter par le producteur Kafka.

3.1.2 Ingestion de données en temps réel avec Kafka

Comme les données générées dans le domaine de la santé évoluent à un rythme exponentiel, la gestion de ces données avec Spark il même devient une tâche difficile, alors que Kafka est conçu spécifiquement pour la gestion de données en continu. C'est pourquoi il a été intégré dans notre système. Dans l'architecture proposée, le bloc de collection de données est utilisé pour recueillir les données liées à la santé de l'individu à partir de sources distribuées et de différentes maladies en utilisant différents dispositifs intégrés à la télémédecine et à la télésanté. Ce bloc collecte, filtre et gère les données cliniques du patient de manière continue. Il nous permet de classer les données du flux continu en fonction du sujet (topic) correspondant (type de maladie) dans lequel les attributs sont publiés.

Apache Kafka (Apache kafka, 2017) est un système de diffusion en continu distribué qui utilise la messagerie de publication et d'abonnement et il est conçu comme un service distribué, partitionné et répliqué. Les données en temps réel sont diffusées en continu à partir des dispositifs de surveillance de santé par les producteurs Kafka. Les serveurs Kafka stockent tous les messages entrants de la part des producteurs pendant un certain temps et les publient dans un flux de données appelé Topic, qui est un nom de catégorie dans lequel les enregistrements sont publiés. Les Topics constituant le cœur que Kafka fournit pour un flux d'enregistrements. Chacun de ces Topics est divisé en plusieurs partitions, chacun stockant une ou plusieurs de ces partitions avec la possibilité d'accepter plusieurs formats. Du côté des consommateurs, les consommateurs de Kafka s'abonnent à un ou plusieurs Topics et reçoivent les données au fur et à mesure de leur publication. Un flux ou un Topic peut avoir plusieurs consommateurs différents comme le consommateur en temps réel, tous ayant leur propre position dans le flux maintenu. La figure 4.4 montre le système de messagerie de Kafka. La coordination et la facilitation du système distribué sont effectuées à l'aide de Zookeeper (Hunt et al., 2010). Dans un premier temps, les producteurs de données sont deux applications de simulation pour les dispositifs connectés et utilisent Apache Kafka pour générer les événements de données. Dans un deuxième temps, la source du flux de données pour notre étude de cas est remplacée par le flux de Twitter.

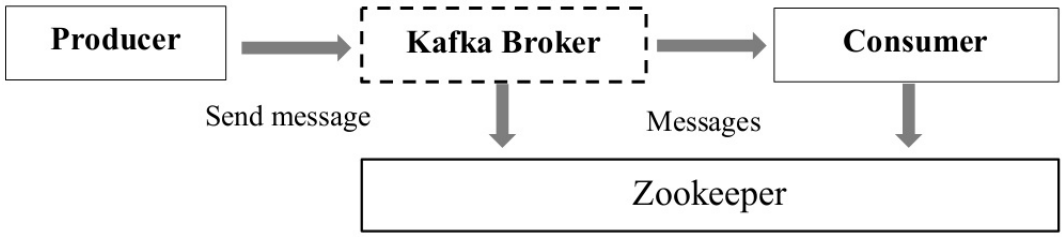

FIGURE 4.4: Système de messagerie de Kafka.

3.2 Analyse

3.2.1 Spark streaming

Spark fournit une bibliothèque d'apprentissage automatique, MLlib, qui consiste en des algorithmes populaires d'apprentissage tels que la classification, la régression, le regroupement, etc. Spark streaming est une extension de l'API Spark pour le traitement en temps réel de données provenant de diverses sources comme Twitter et Kafka. Le flux de données entrant est regroupé en intervalle de lots inférieur à une seconde et traité avec le traitement par lots de Spark qui intègre des fonctionnalités puissantes de traitement en temps réel. Spark met en œuvre une extension par le biais du module de Spark streaming fournissant une abstraction de haut niveau appelée 'discretized stream' ou DStream qui est une séquence de mini lots où

chaque mini lot est représenté comme un Spark RDD. Dans ce travail, le traitement de flux de données utilise Spark où Spark streaming gère le flux de données envoyé par Kafka en utilisant la bibliothèque Spark streaming, tandis que l'implémentation des modèles de classification est effectuée en utilisant la bibliothèque d'apprentissage automatique de Spark, MLlib.

3.2.2 Architecture du cluster Spark

Spark est un moteur de traitement distribué et il suit l'architecture maître-esclave (master-workers), ainsi, pour chaque application Spark, il créera un processus master et plusieurs workers. Dans la terminologie de Spark, le master est le pilote (driver) et les workers sont les exécuteurs (executors). Comme le master est le driver, il est le responsable de l'analyse, de la distribution, de la programmation et du suivi des tâches des exécuteurs. Le master est également le responsable de la maintenance de toutes les informations nécessaires pendant la durée de vie de l'application. De l'autre côté, les workers ne sont responsables que de l'exécution du code qui leur a été attribué par le master, le rapport d'exécution sera renvoyé au master, figure 4.5.

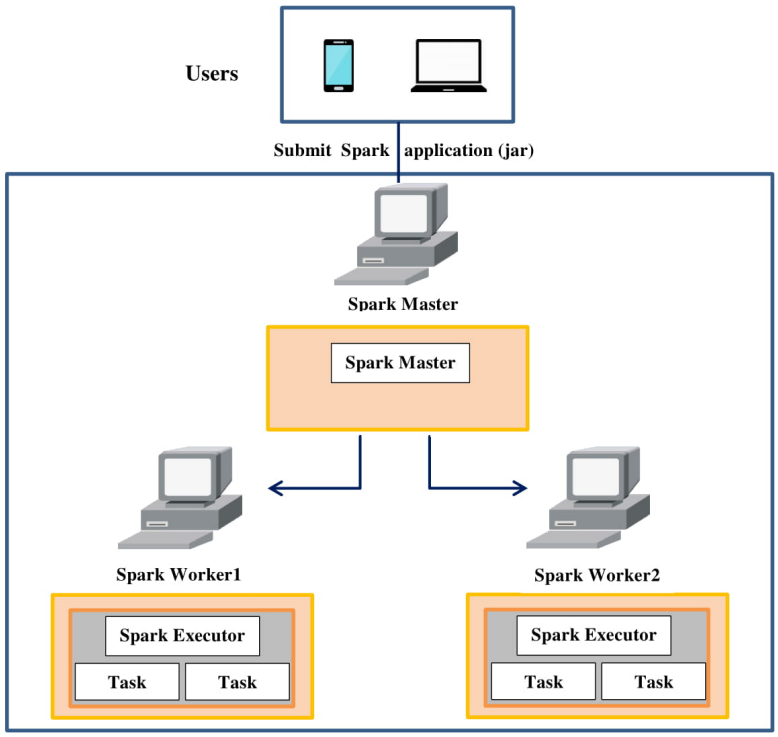

FIGURE 4.5: Architecture de cluster Spark.

3.2.3 Étude de cas : description de données

Pour mettre en œuvre le modèle proposé dans ce travail de recherche, deux bases de données sont utilisées. La première est liée au diabète, il provient d'un site de compétitions de science

de données nommé Kaggle[1]. Cette base comprend 15 000 enregistrements et neuf attributs, chaque enregistrement a huit attributs (Tableau4.1) et l'un des deux résultats possibles, à savoir si le patient est testé positif pour le diabète indiqué par 1 ou non indiqué par 0. Le tableau 4.2

TABLEAU 4.1: La description de la base de données de diabète

Numéro	Attributs	Description
1	Pregnancies	Grossesses
2	Plasma G	Glucose dans le plasma
3	Diastolic BP	La pression artérielle diastolique
4	Triceps Th	Épaisseur des triceps
5	Serum I	Sérum Insuline
6	BMI	L'indice de masse corporelle
7	Diabetes P	Pedigree du diabète
8	Age	Age en années
9	Num	Classe (1= présence de maladie, 0 = absence de maladie)

présente l'échantillon de données sur la maladie de diabète.

TABLEAU 4.2: Échantillon de la base de données de la maladie de diabète

Attributs								
Pregnancies	Plasma G	Diastolic BP	Triceps Th	Serum I	BMI	Diabetes P	Age	Classe
0	171	80	34	23	43,51	1,21	21	0
8	80	95	33	24	26,62	0,44	53	1
8	92	93	47	36	21,24	0,16	23	0
9	103	78	25	304	29,58	1,28	43	1

La deuxième base de données est 'processed.cleveland.data' de la maladie cardiaque (HD) (UCI, 2017) elle a été utilisée et analysée. Il s'agit d'une base de données étiquetée (labelled) qui comprend 303 enregistrements et 14 attributs (Tableau 4.3). Elle a été utilisée dans de nombreux travaux de recherche de ML. Pour chaque observation de la maladie cardiaque, nous avons construit une base de données étiquetées avec des attributs, où l'attribut d'étiquette de la classe est étiqueté avec deux classes, la présence de la maladie cardiaque et l'absence de la maladie cardiaque. Les valeurs des attributs de l'étiquette de classe ont été modifiées seulement par 0 et 1, où la valeur 1 indique la présence de la maladie cardiaque remplacent les valeurs 1, 2, 3 et 4, tandis que la valeur 0 indique l'absence de la maladie cardiaque, ce qui la transforme en une base de données de classe binaire. Le tableau 4.4 présente l'échantillon de données sur la maladie cardiaque. La base de donnée comportent 76 attributs bruts, dont seulement 14 sont effectivement utilisés, 139 (45,87 %) enregistrements présentent la présence de la maladie cardiaque tandis que 164 (54,13 %) présentent l'absence de la maladie cardiaque.

[1]https://www.kaggle.com/fmendes/diabetes-from-dat263x-lab01

TABLEAU 4.3: La description de la base de données de la maladie cardiaque

Numéro	Attributs	Description
1	Age	Age en années
2	Sex	Sex(1= masculin, 0= féminin)
3	Cp	Douleur de poitrine
4	Restbpss	La pression artérielle au repos
5	Chol	Serum Cholestoral
6	Fbs	Hyperglycémie
7	Restecg	Résultats électrocardiographiques au repos
8	Thalach	Fréquence cardiaque maximale
9	Oldpeak	Dépression du segment ST induite par l'exercice par rapport au repos
10	Exang	Angine induite par l'exercice
11	Slope	Pente du pic du segment ST
12	Ca	Nombre de vaisseaux principaux colorés par fluoroscopie
13	Thal	3(normal),6(fixed defect),7(reversable defect)
14	Num	Classe (1 présence de maladie, 0 = absence de maladie)

TABLEAU 4.4: Échantillon de la base de données de la maladie cardiaque

Attributs													
Age	Sex	Cp	Restbpss	chol	Fbs	Restecg	Thalach	Oldpeak	Exang	Slope	Ca	Thal	Num
63.0	1.0	1.0	145.0	233.0	1.0	2.0	150.0	0.0	2.3	3.0	0.0	6.0	0
67.0	1.0	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	2.0	3.0	3.0	1
67.0	1.0	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	2.0	2.0	7.0	1
37.0	1.0	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	3.0	0.0	3.0	0
41.0	0.0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	1.0	0.0	3.0	0

Dans ce module, les bases de données ont été analysées selon l'approche de l'analyse prédictive en utilisant l'environnement Spark. Les données sont chargées à partir du fichier csv dans un RDD. Nous utilisons la transformation Map sur le RDD, qui appliquera la fonction Parse RDD pour transformer chaque élément String du RDD en un RDD de point labellisé et de l'utiliser pour entraîner et tester le modèle de ML qui prédit l'état de santé (Figure 4.6). Comme ces travaux portent essentiellement sur la collection de données en temps réel, le traitement de données en continu, ML distribué et le stockage distribué, la base de données utilisée et l'état de santé attendu n'est pas très importante puisque ces données peuvent être facilement remplacées par une autre base de données pertinentes.

3.2.4 Implémentation du modèle de classification évolutif

La classification de l'état de santé provenant de flux de données a besoin de construire un modèle de classification, qui est capable de classer les attributs de chaque flux en présence ou en absence de la maladie. Par conséquent, un modèle adéquat et parallèle pour prédire l'état de

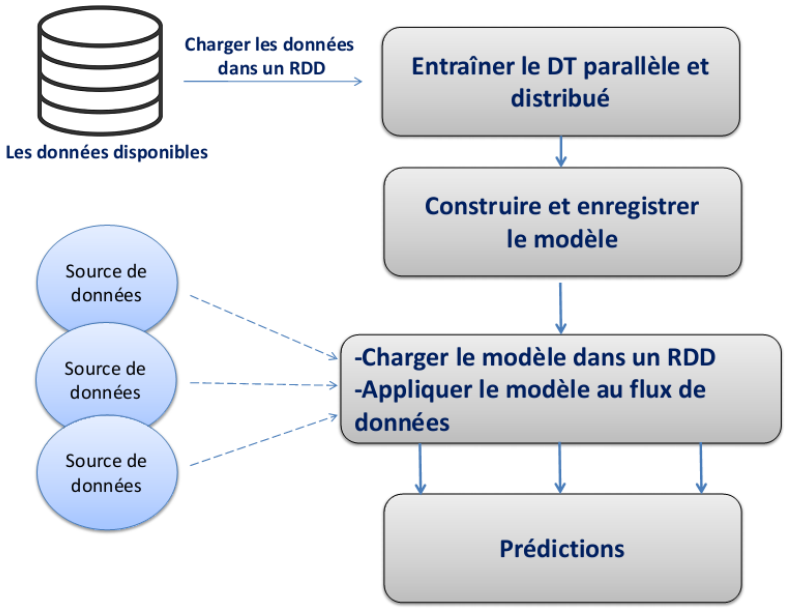

FIGURE 4.6: Le processus de ML avec Spark.

santé dans un contexte big data en utilisant Spark est nécessaire. Sur cette base, une adaptation du modèle DT est plus importante. Dans ce travail la parallélisation de DT est effectuée à l'aide de Spark MLlib.

L'algorithme 3 représente les étapes pour entraîner et tester le DT sur un environnement distribué basé sur Spark. Dans cette partie, il y a deux phases pour la classification de l'état de santé en temps réel. La première consiste à analyser la base de données sur les soins de santé pour construire le modèle de ML. La seconde utilise le modèle en production pour effectuer la classification sur le flux de données de santé en temps réel. Le processus de déroulement est présenté dans la figure 4.7. Les bases de données ont été divisées aléatoirement en une base d'apprentissage et une base de test, 70 % de données sont utilisées pour l'apprentissage du modèle et 30 % seront utilisées pour le test. Le modèle a été entraîné à l'aide de la base d'apprentissage qui utilise l'impureté de Gini pour classer les maladies. En utilisant l'impureté de Gini et en variant les paramètres maxBins, maxDepth, différents modèles du DT ont été testés trois fois, les valeurs moyennes de l'accuracy de classification sont calculées dans chaque cas.

3.3 Stockage et visualisation

3.3.1 Stockage distribué

Pour obtenir des modèles significatifs (Ali et al., 2016) à partir du big data générées par tous les utilisateurs, telles que les informations de diagnostiques sur les patients est également un grand défis. Les résultats prévus doivent être stockées de manière distribuée pour garantir

Algorithme 3 Le modèle DT sous Spark MLlib

Step1: Start new SparkContext
Laoding required package and APIs
sparkContext(master, appName, sparkHome)
step2: Load and parse the dataset into an RDD
rowData(RDD) : sc.textFile(path)
Data(RDD) : Map(pareseFunction(rowData)) parse each input line in parallel
Step3: Split the data into training and test sets
Set related parameters
trainData(RDD), testData(RDD):
randomSplit(Data)
trainData.cache(): cache the trainData in memory
testData.cache(): cache the testData in memory
train the model
step4: Test the model
LabelAndPredict(RDD) : Map(predictFunction(testData)) parse and predict each input line in parallel
Save model : save(sc, path)

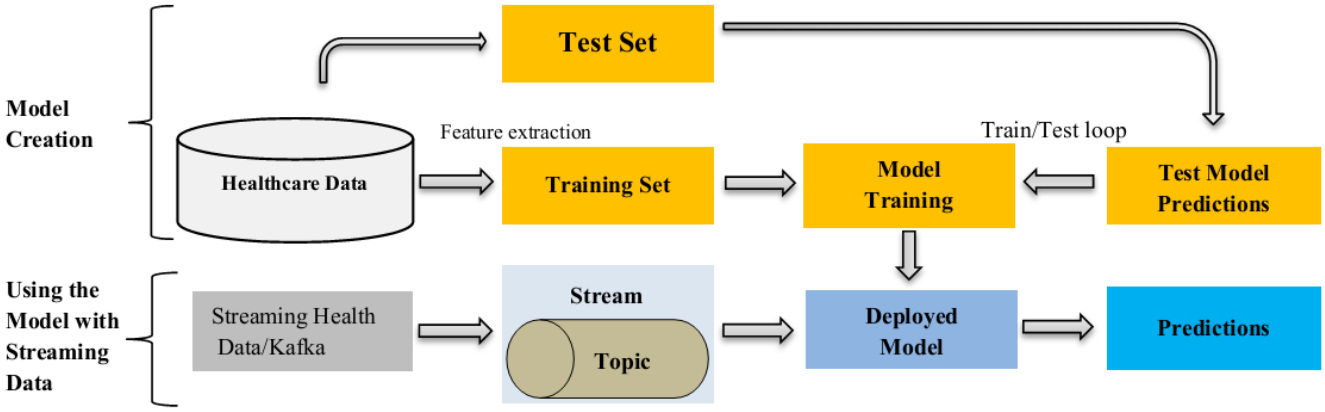

FIGURE 4.7: Le processus de ML.

la disponibilité de données avec aucun point de défaillance. Apache Cassandra [2] (Apache Cassandra, 2017) est un système de base de données distribué très puissant, particulièrement efficace pour prendre en charge de grands volumes d'enregistrements sur plusieurs serveurs. Cette base de données peut être facilement mise à l'échelle pour répondre à une augmentation soudaine de la demande. Il suffit de déployer une base de données en cluster à nœuds multiples avec Cassandra. De plus, Cassandra est très disponible et a l'avantage de ne pas avoir aucun point de défaillance (Lakshman and Malik, 2010). Cassandra offre également des vitesses d'écriture et de lecture extrêmement rapides avec Spark (Hassan and Bansal, 2018).

[2]https://cassandra.apache.org/

3.3.2 La visualisation

La visualisation de données est devenue une discipline à part entière, nécessaire pour visualiser la multitude de données auxquelles nous avons accès et pour communiquer les informations les plus pertinentes. Cette visualisation permettra entre autres d'expliquer et de mettre en évidence les informations clés qui seront exploitées à partir des bases de données (Ali et al., 2016). La visualisation de données est la représentation graphique de l'information, elle se situe à l'intersection des domaines de la communication, des sciences de l'information et du design. Le principal avantage de la visualisation de données n'est pas de rendre les données plus belles, mais de donner un aperçu aux données complexes en communiquant leurs aspects clés. Il permet aux décideurs de voir des analyses présentées visuellement qui joue un rôle clé dans la prise de décision.

Le stockage et la visualisation des enregistrements électroniques de santé peuvent aider à identifier les modèles du système de prévision des maladies, aider les prestataires des soins de santé à trouver un système précis et adapté aux besoins des patients, à prendre de meilleures décisions financières et sanitaires sur la base des prévisions de résultats faites par le système (Acharjya and Ahmed, 2016). En outre, grâce à la disponibilité de données, nous pouvons extraire la relation entre chaque attribut (variable dépendante) et le résultat (attribut désiré). En outre, le regroupement de dossiers des patients similaires peut aider les médecins à gagner du temps lorsqu'ils diagnostiquent des maladies. En fait, chaque groupe du patient peut subir le même type de traitement. Enfin, nous pouvons obtenir des informations utiles sur les maladies en nous basant sur les statistiques de données.

Ici, la visualisation de données est effectuée à l'aide de Zeppelin [3] (Apache zeppelin, 2017), un nootebook polyvalent basé sur le web qui permet l'analyse interactive de données, c'est un environnement d'analyse de données open source qui fonctionne au-dessous de l'Apache Spark. Avec l'utilisation du concept de dataframe et Spark SQL, la base de données peut être interrogée par différentes requêtes telles que le nombre d'instances, le nombre de différents cas selon le type de maladie et de faire des analyses de données plus sophistiquées et de haut niveau. La description de l'interrogation de données à l'aide de Spark SQL est la suivante :
- Spark SQL et SQL Context sont importés.
- Charger les données de la table Cassandra pour construire un RDD et exécuter les commandes d'action.
- Appeler la transformation Map et mapper le RDD en classe de cas de la maladie cardiaque.
- Convertir la classe de cas en cadre de données (dataframe) et créez un tableau temporel.
- Définir les requêtes exécutées en utilisant les requêtes de Spark SQL. Dans cette étape, nous avons étudié le temps de récupération des résultats des requêtes suivantes (Tableau 4.5) avec différentes tailles des enregistrements et de clusters multi-nœuds.

[3] https://zeppelin.apache.org/

TABLEAU 4.5: Liste des requêtes

Numéro	Requêtes
1	Le nombre total des patients (Enregistrement)
2	Le nombre total des patients atteints de maladie
3	Regrouper les patients par l'âge
4	Récupérer le dernier enregistrement

4 Expérimentation et résultats

4.1 Implémentation

La prévision et l'analyse de l'état de santé en temps réel basée sur Twitter, Kafka, Spark et Cassandra a été programmée en utilisant le langage de programmation Scala et Zeppelin comme plateforme de développement qui supporte de nombreux interpréteurs comme Scala, Spark et Cassandra. L'algorithme 4 décrit les principales étapes de la mise en œuvre de notre système. Tout d'abord, le système proposé est réalisé et testé sur un cluster de nœuds unique créé sur une machine autonome avec un core i7 processeurs et 8 Go de RAM, utilisant le système d'exploitation Ubuntu 16.04 via Spark plateforme qui intègre MLlib, notamment DT et RF avec Kafka streaming pour le traitement de données en temps réel.

Dans cette étape, toutes les expériences ont été menées avec les ressources informatiques disponibles sur un cluster composé d'un nœud master et de deux nœuds agissant comme des workers. Tout d'abord, un groupe appelé spark et un compte d'utilisateur appelé spuser ont été créés pour simplifier la communication entre les nœuds. Java et Scala ont été installées. Nous installons Open SSH Server, générons les paires de clés et la configuration ssh sans mot de passe entre les nœuds de sorte que le master Spark puisse se connecter, démarrer, arrêter et exécuter des tâches dans différents Workers. Nous avons décompressé et installé Spark, Kafka et Cassandra en un seul nœud. Deux topics et deux tables ont été créés. Nous éditons le fichier .bashrc situé dans le répertoire personnel de l'utilisateur et ajouter les variables d'environnement telles que JAVA_HOME et SPARK_HOME. Ajouter le fichier slaves dans $SPARK_HOME/conf qui doit inclure le nom d'hôte des workers. Pour avoir la même copie de différents frameworks, nous copions trois fois le dossier de configuration du cluster à nœud unique, en renommer un comme master et les autre comme worker1 et worker2. Modifier le nom d'hôte et les hôtes sur tous les nœuds. L'algorithme 5 représente les étapes de la mise en place du cluster.

Le tableau 4.6 présente les caractéristiques de notre cluster. Après avoir établi une connexion au flux de Twitter par le biais du Kafka streaming comme le montre la figure 4.2, l'application reçoit en permanence le flux de données provenant de plusieurs producteurs Kafka, une fois qu'il rencontre le flux de contrôle de l'état de santé, il extrait les valeurs d'attribut

Algorithme 4 Les étapes du traitement de framework proposé

Step 1 : Twitter authentification
We set up the authentication credentials of Twitter using the Twitter4J library: consumerKey, consumerSecret, accessToken and accessTokenSecret
Step 2 : Spark context
Create an instance of SparkContext(sc by default in Zeppelin notebook) and StreamingContext to use all Spark streaming features
Step 3 : Get Twitter streams
createStream method of TwitterUtils was used to get DStreams, TwitterUtils uses Twitter4J to get Twitter status streams
Step 4: Kafka producer
Create a Kafka producer which will receives Twitter data streams
filter tweets based on certain keyword
gets the tweets username and publishes to a specified Kafka topic using send method
Step 5: Get Kafka streams
Create the direct stream with the Kafka parameters and topic using createDirectStream method of KafkaUtils
Step 6: Data processing
Extract username and attributes from each tweet status using foreachRDD method
apply the machine learning model to predict health status
send appropriate message based on Twitter username using sendDirectMessage method
save all attributes and predicted label to Cassandra keyspace and table using saveToCassandra method
Step 7: Start the computation
Start Spark streaming context using start method

des événements de données envoyées par Kafka streaming et applique le modèle DT pour prédire l'état de santé. D'autre part, les détails de chaque cas qui sont définis précédemment seront persistés dans la table de la base de données Cassandra pour les interroger plus tard. Une fois que le système d'analyse de l'état de santé en temps réel a été testé avec succès sur un cluster à nœud unique, un cluster multi-nœuds avec un master et deux workers a été créé.

TABLEAU 4.6: Caractéristiques de cluster

Paramètres	Master	Worker
Cores	4	4
Mémoire	8 GB	4 GB
Système d'exploitation	Ubuntu 16.04	Ubuntu 16.04

4.2 Évaluation de modèle d'apprentissage

Comme c'est indiqué dans la discussion précédente, le modèle de ML est testé sur les bases de données relatives à la maladie cardiaque et le diabète dans le rapport 70-30. Différents modèles

Algorithme 5 Les étapes de mise en œuvre du cluster

Step 1: Create a group called spark, an user called spuser and add the spuser to sudoers list
addgroup spark
adduser –ingroup spark spuser
visudo
ALL= (ALL) ALL
Step 2: Install ssh server and change permissions and disable IPV6
apt-get install openssh-server
ssh-keygen
cat home/.ssh/id_rsa.pub ≫ home/.ssh/authorized_keys
chmod 700 home/.ssh/authorized_keys
Step 3: Install Java, Scala, Spark, Kafka, Cassandra and Zeppelin
Download zip file of all installation
Unpacked and move all these zip files to /usr/local/
Change permissions of all files to have all permissions for spuser
Step 4: Update home/.bashrc file
Add all necessary environment variables
Step 5: Setup multinode cluster
Copy the single node cluster 3 times
Rename one master and other as worker1 and worker2
Update hostname and hosts of all 3 nodes
Step 6: Starting the cluster
Open terminal (Ctr+Alt+T)
Cd /usr/local/kafka/bin
Start Zookeeper
Start Kafka
Create Kafka topic
Cd /usr/local/cassandra/bin
Start Cassandra
Create a keyspace and table
Cd /usr/local/zeppelin/bin
Start Zeppelin
Cd /usr/local/spark/sbin
Start Spark

de DT avec la variation de profondeurs d'arbre et la valeur de paramètre d'impureté ont été testés et les valeurs de précision sont calculées pour chaque cas. Sur la base de l'analyse des erreurs du modèle, il a été découvert que la prédiction la plus précise devient stable lorsque la profondeur de DT est égale à 8 et 6 pour le diabète et la maladie cardiaque respectivement. Le tableau 4.7 présente le taux de classification pour les deux bases de données.

TABLEAU 4.7: Le taux de classification de DT (MLlib)

Base de données	Diabète	Maladie cardiaque
maxBins	250	100
maxDepth	8	6
Sensibilité (%)	85.97	80.00
Spécificité (%)	94.38	85.36
Accuracy (%)	91.57	82.40

4.3 Les performances de Apache Spark vs Weka

Grâce à sa capacité de calcul en mémoire, Spark peut être plusieurs fois plus rapide que les autres technologies traditionnelles, en particulier pour le ML itératif. Afin de démontrer l'efficacité du système de prévision basé sur Spark en terme du temps d'apprentissage et de test du modèle de ML à travers une large base de données, d'autres enregistrements de données ont été simulés. Pour tester et démontrer l'extensibilité de notre approche, une étude comparative entre le temps d'exécution de DT basée sur Spark et Weka a été réalisée. Dans ce contexte, pour avoir une base de données de taille suffisante, nous augmentons le nombre d'enregistrements. Ensuite, la vitesse de l'algorithme DT utilisant scala et la bibliothèque MLlib, a été mesurée et comparée avec le même algorithme dans Weka. En effet, nous avons comparé le temps d'exécution de DT dans le cluster (Spark) contre Weka (K : mille, M : million). Les figures 4.8 et 4.9 représentent la comparaison des performances de la mise en œuvre de DT à l'aide de Spark et Weka. Dans cette simulation, lorsque le nombre d'enregistrements est supérieur ou égal ou à 3 millions, l'apprentissage de modèle n'est pas supporté en utilisant le simple DT dans Weka.

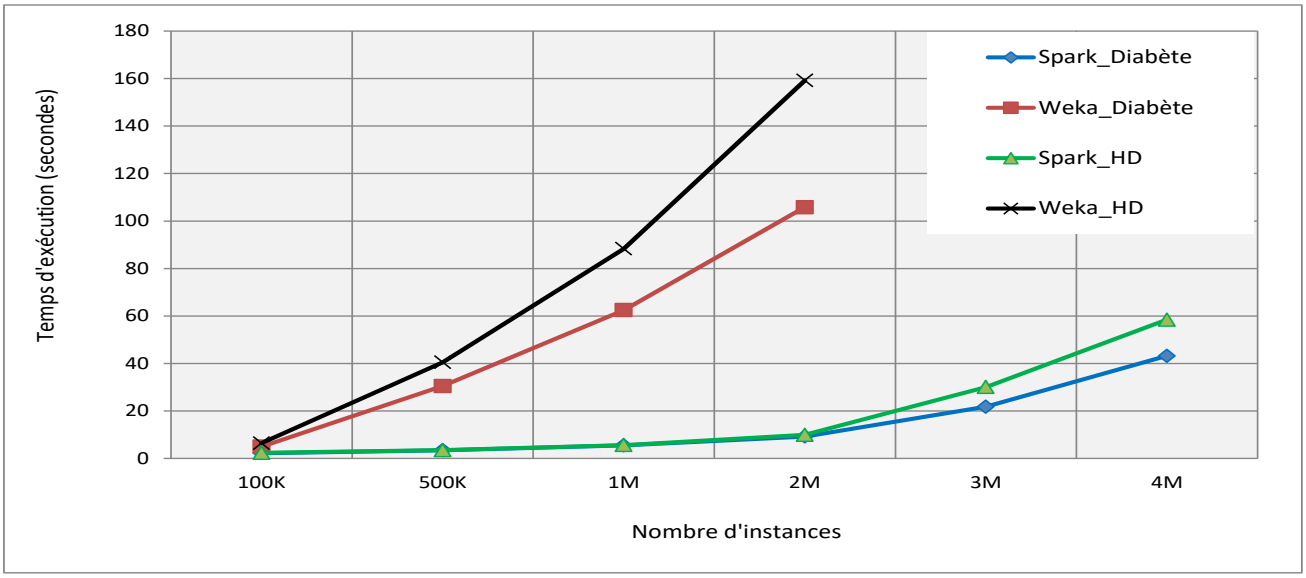

FIGURE 4.8: Temps d'exécution du DT en utilisant Spark MLlib et Weka, temps d'apprentissage de modèle.

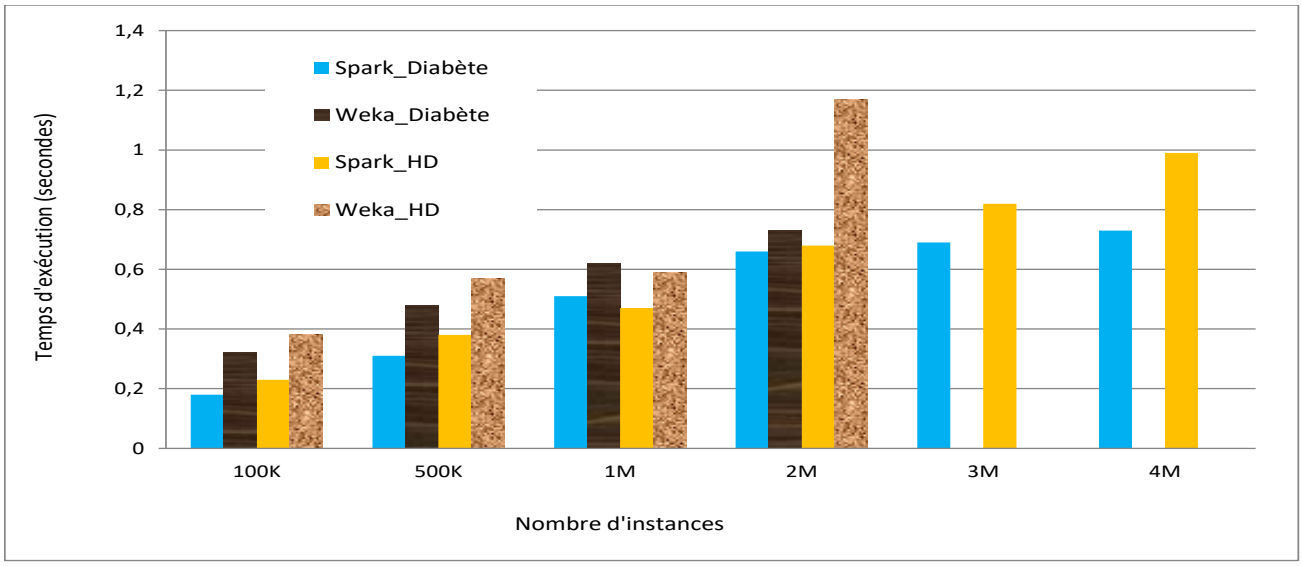

FIGURE 4.9: Temps d'exécution du DT en utilisant Spark MLlib et Weka, temps de test de modèle.

Comme on peut le voir sur le graphique, l'exécution du modèle DT avec Spark et la base de maladie cardiaque est plus rapide que le framework Weka. Il ne faut que 58.43 s pour entraîner le modèle dans le cas de 4 millions d'enregistrements alors que Weka prend 159,19 s pour entraîner le modèle dans le cas de 2 millions d'enregistrements. De même, il ne faut que 0,99 s pour tester le modèle dans le cas de 4 millions d'enregistrements alors que Weka prend 1,17 s pour tester le modèle dans le cas de 2 millions d'enregistrements. Le modèle DT proposé, basé sur Spark, met moins du temps pour entraîner et tester le modèle de ML. L'algorithme DT parallèle de Spark Mllib atteint une meilleure extensibilité grâce au calcul distribué sur les nœuds de cluster et au calcul en mémoire. Spark MLlib traite les données en moins de temps car il divise le travail en plusieurs tâches qui sont exécutées sur les nœuds workers. Après une analyse des résultats obtenus, on peut conclure que Spark constitue le meilleur moyen de mettre en œuvre le système proposé pour prévoir l'état de santé en temps réel.

4.4 Evolutivité du DT à l'aide de Spark

Cette étape consiste à évaluer les performances de l'algorithme DT basé sur Spark proposé dans un environnement parallèle et distribué. Différents nombres de nœuds et différentes tailles de base de données d'apprentissage sont prise en compte. Comme c'est mentionné précédemment, nous avons 2 nœuds et nos bases de données d'apprentissage varié de 100 milles à 4 millions en termes de nombre d'enregistrements. Les figures 4.10, 4.11, 4.12 et 4.13 illustrent le temps d'exécution de notre algorithme DT basé sur Spark avec un nombre différent de nœuds. Lorsque le nombre d'instances est respectivement de 2, 3 et 4 millions, nous pouvons voir que le temps d'exécution total diminue à mesure que le nombre de nœuds augmente. Cela indique

que plus le nombre de nœuds impliqués dans le calcul est élevé, plus l'algorithme sera rapide en raison du calcul parallèle et distribué. En revanche, le temps d'exécution de DT standard utilisant Weka avec deux nœuds reste stable en raison de l'environnement informatique non distribuée.

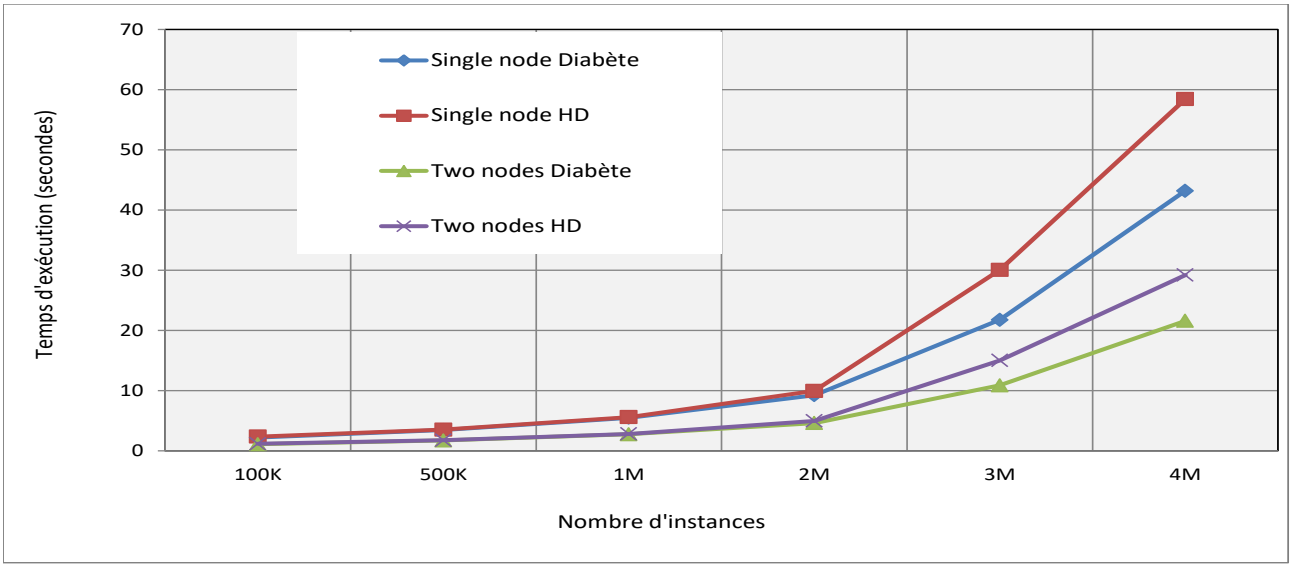

FIGURE 4.10: Temps d'exécution du DT en utilisant Spark pour différents nœuds, temps d'apprentissage de modèle.

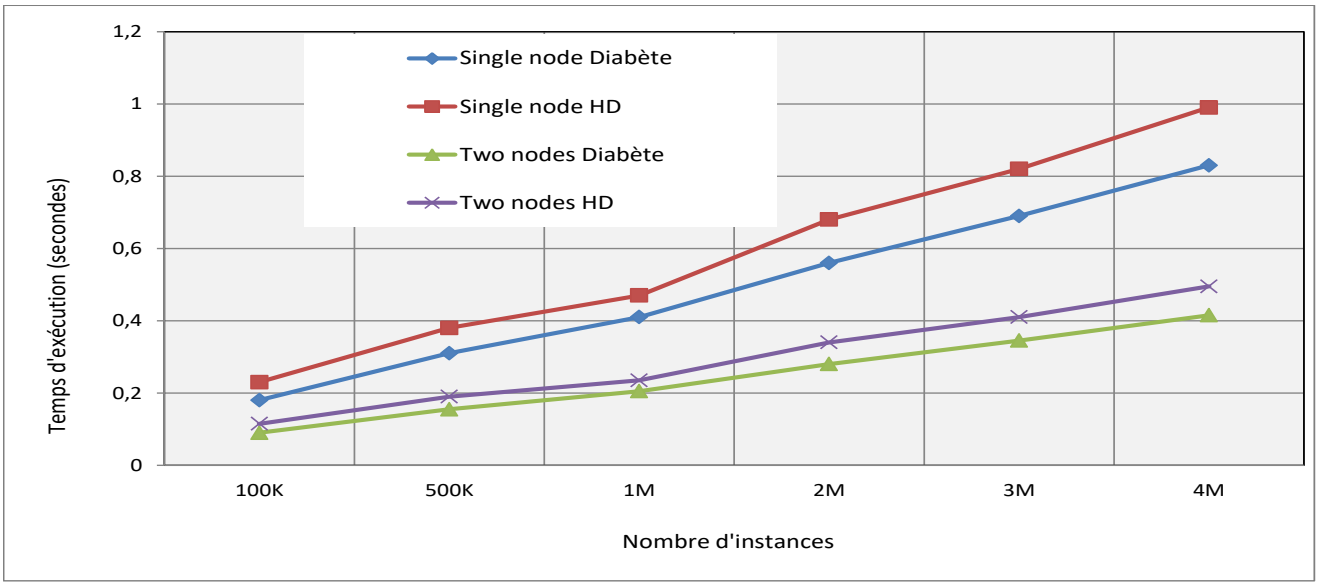

FIGURE 4.11: Temps d'exécution du DT en utilisant Spark pour différents nœuds, temps de test de modèle.

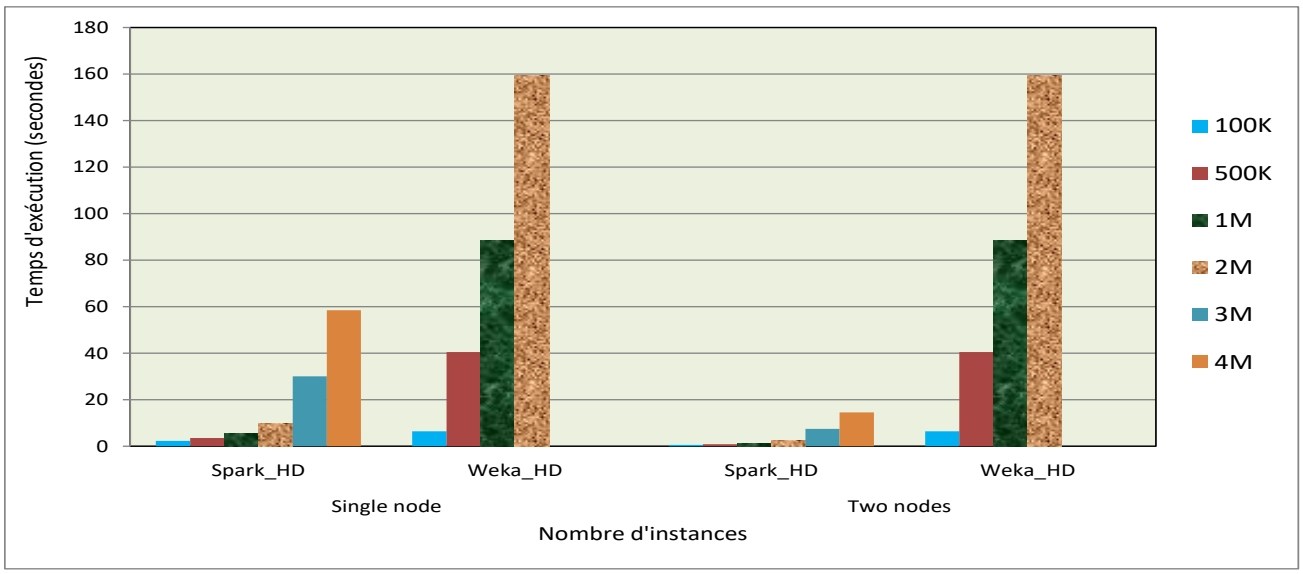

FIGURE 4.12: Temps d'exécution du DT en utilisant Spark et Weka pour différents nœuds, temps d'apprentissage de modèle.

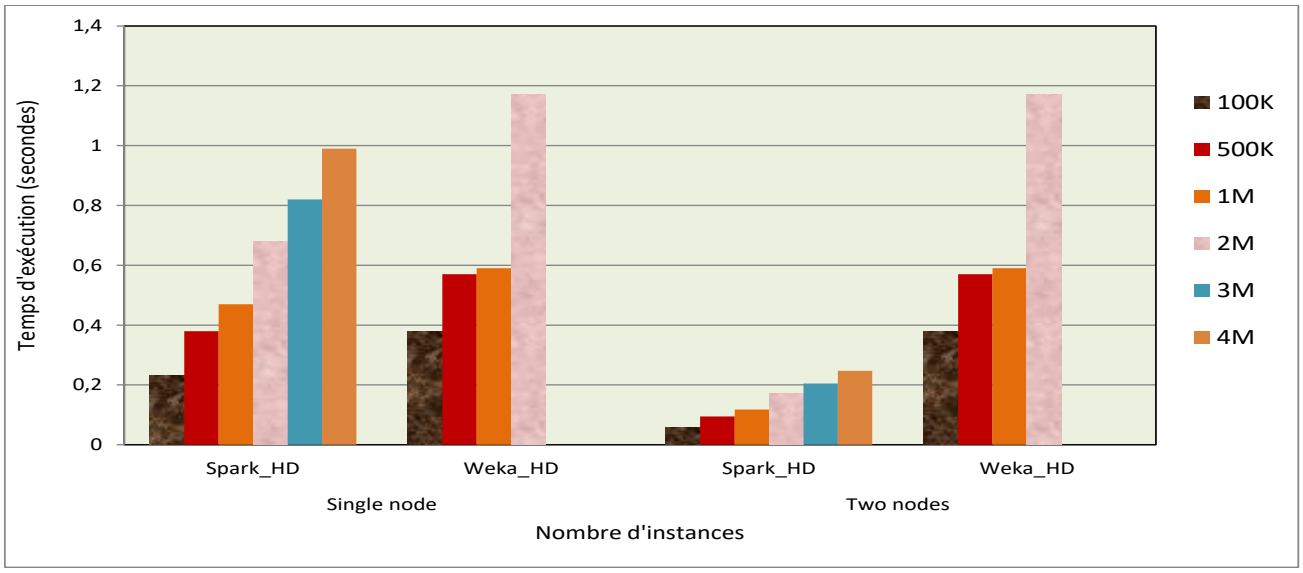

FIGURE 4.13: Temps d'exécution du DT en utilisant Spark et Weka pour différents nœuds, temps de test de modèle.

4.5 Le débit du cluster Spark

Dans un premier temps, les bases de données sur les maladies ont été traitées afin de construire des bases de données étiquetées. Le modèle DT était construit et testé séparément en faisant varier les paramètres tels que maxDepht, maxBins, l'erreur minimale du modèle est prise en compte en fonction de l'accuracy de classification du modèle (Tableau 4.7). Un modèle hors ligne a été créé et sauvegardé afin de pouvoir l'utiliser en temps réel, comme le montre la figure 4.7. Dans notre étude de cas, pour tester le débit de système, deux applications de

simulation agissant comme des producteurs de données de Kafka ont été créés, les producteurs de données consistent en deux applications de simulateur pour le flux des maladies, une pour le diabète et l'autre pour la maladie cardiaque (Figure 4.14). Nous utilisons deux producteurs,

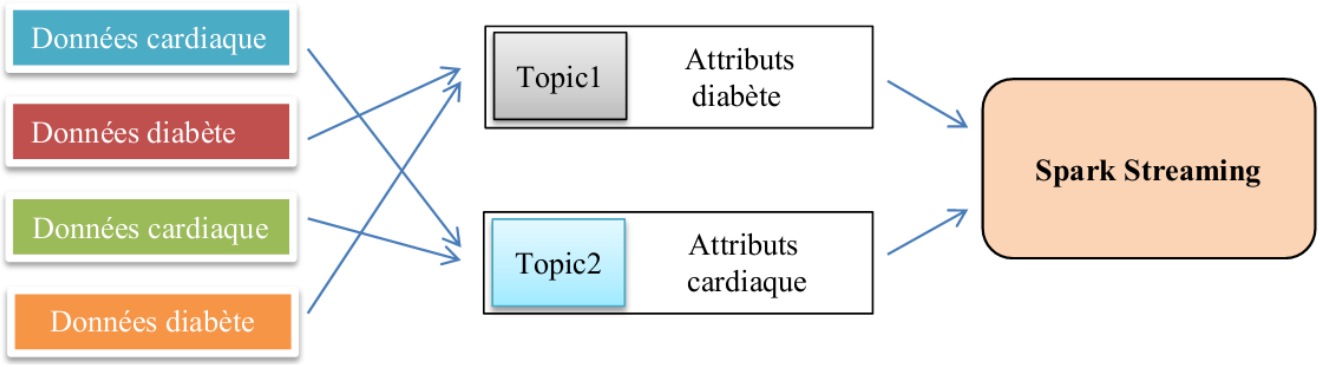

FIGURE 4.14: Simulation de flux de données.

chacun d'entre eux envoie environ 270000 (voir plus) d'événements par seconde par nœud dans un format prédéfini qui a été précédé par un mot clé comme le mot d'entête et qui sert à son tour au consommateur (Spark streaming). Nous avons réalisé trois scénarios avec un intervalle de 1, 2 et 3 secondes. Le tableau 4.8 présente les expériences menées. La figure 4.15 représente l'évaluation des performances du système proposé. Nous remarquons que le système peut traiter environ 550 k enregistrements par seconde/nœud, soit environ de 29 Mo par seconde/nœud.

TABLEAU 4.8: Les expériences menées

Nombre des nœuds	Événement/secondes	Kafka	Partitions	Spark workers	Cassandra
1	$\sim$ 550 000	1	1	1	1
2	$\sim$ 1 300 000	2	2	2	2

Comme on peut le voir sur la figure 4.15 :
- Plus le débit est important, plus le coût du temps de traitement est élevé.
- Plus de nœuds utilisés, moins le temps de traitement.
- Si l'on utilise suffisamment de nœuds, même si le débit est important, la performance peut être proche de l'optimum. Par exemple, étant donné un débit de 2.7 million, si nous utilisons 2 nœuds, le temps d'exécution est 2 secondes. Alors que le temps d'exécution est proche de 3 secondes lorsque nous utilisons un seul nœud. Le temps d'exécution d'un même débit diminue en ajoutant d'autres nœuds.
- Spark streaming avec ML distribué est un bon choix pour traiter les problèmes qui nécessitent un traitement en temps réel. En effet, en utilisant plus de nœuds, nous pouvons résoudre les problèmes du big data en particulier ceux liés à la prédiction en temps réel dans les soins de santé.

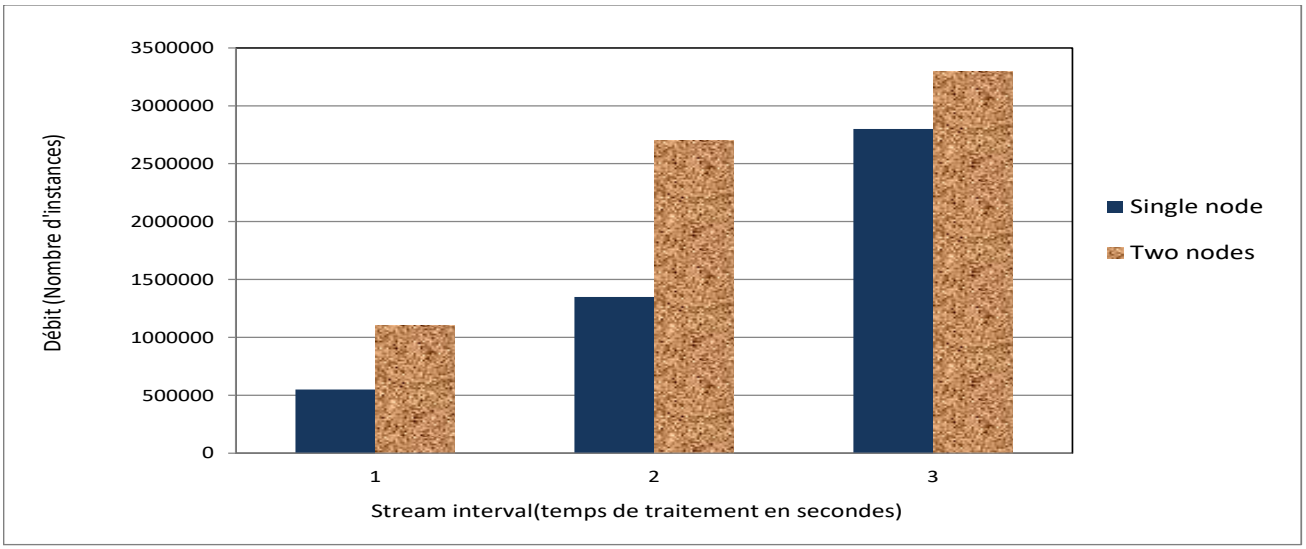

FIGURE 4.15: Temps de traitement du débit par rapport à l'augmentation de la taille de données pour différents nœuds et différents intervalles.

4.6 Débit des requêtes

La performance du temps de récupération des enregistrements est principalement calculée par rapport à une taille spécifique de données (3 millions d'enregistrements). La figure 4.16 montre le temps d'interrogation pour chaque requête en utilisant un système traditionnelle de gestion de base de données relationnelle (SGBDR) comme MySQL et Spark SQL. Pour la base de données de la maladie cardiaque, lors de chaque interrogation, le temps d'interrogation augmente de manière significative pour le SGBDR traditionnel plutôt que Spark. La figure ci-dessus montre que Spark accélère le temps d'exécution par rapport aux SGBDR traditionnels. Dans ce contexte, Spark peut aider à accélérer les demandes de rapports lents et à ajouter plus d'évolutivité aux requêtes de longue durée. En utilisant Zeppelin, un tableau de bord de données en temps réel a été créé qui récupérera les données de la base de données Cassandra et les affichent dans des graphiques et des tableaux. Cette application utilise Spark SQL et Angularjs pour pousser les données dans la page web dans des intervalles fixes, donc les données seront rafraîchie automatiquement.

4.7 Comparaison des performances de DT avec d'autres ML de MLlib

Après avoir testé le système à l'aide une application de simulation et avec le modèle DT, nous avons mené une comparaison entre le DT et les autres ML de MLlib pour choisir le meilleur modèle avec la base de données de la maladie cardiaque. Le tableau 4.9 donne les résultats des performances. La figure 4.17 illustre le taux d'accuracy.

Par la suite nous choisissons le modèle RF puisque il est le meilleur en termes d'accuracy. Il est intégré dans le système au lieu de DT. Après, deux comptes Twitter ont été créés. Les

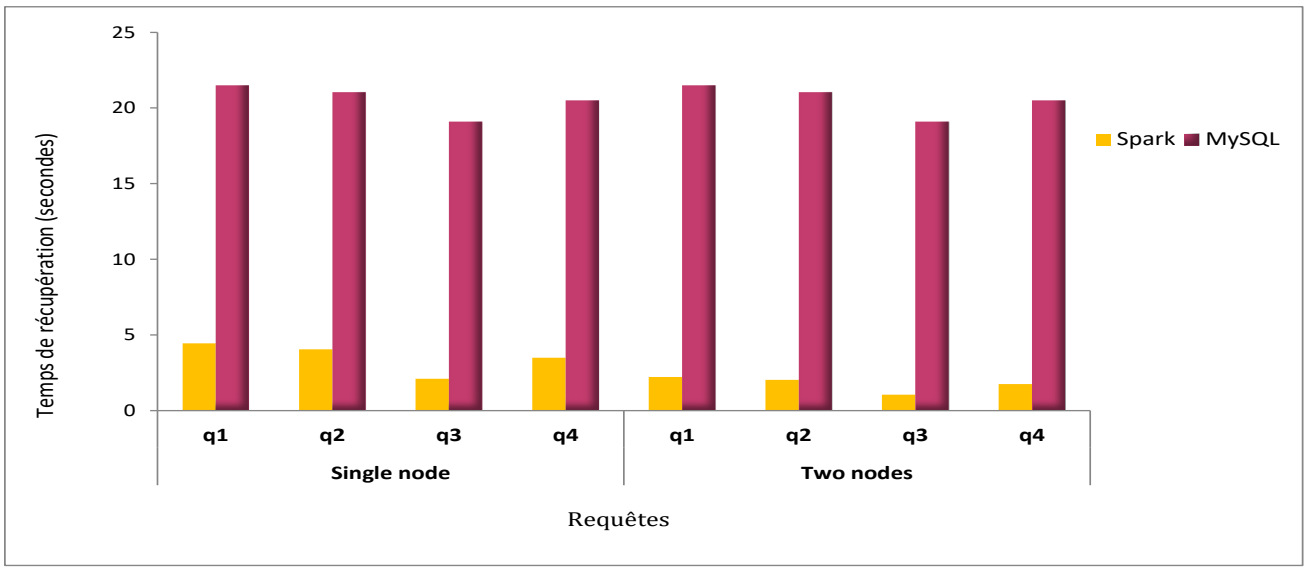

FIGURE 4.16: Comparaison de temps d'exécution des requêtes entre Spark SQL et MySQL.

attributs liés à la maladie cardiaque ont été tweetés dans un format prédéfinis et précédé d'un mot-clé à partir de statut de Twitter. Tous ces attributs tweetés ainsi que le nom d'utilisateur seront capturés et lus par Kafka streaming en temps réel, qui les servira à son tour pour le Spark streaming en tant que producteur de Kafka. Dans le Spark streaming, les valeurs des attributs et le nom de l'utilisateur ont été extraites. RF a été appliqué sur les attributs extraits pour prévoir l'état de santé et envoyer un message approprié indiquant l'absence ou la présence de maladie. Le résultat sera enregistré dans la base de données Cassandra pour une analyse

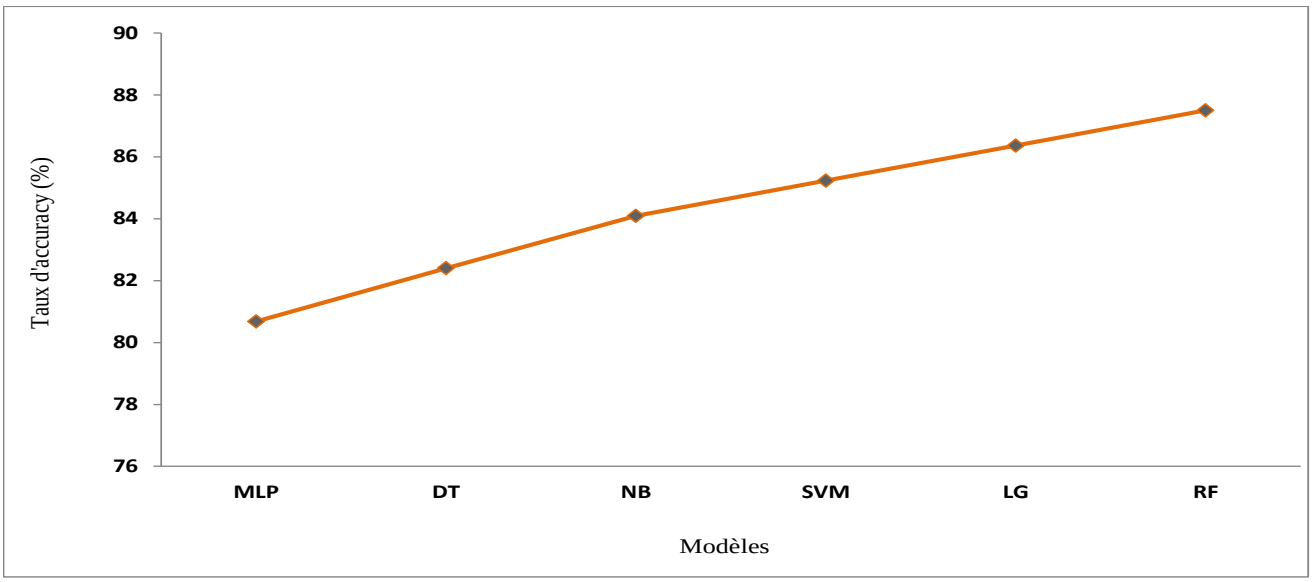

FIGURE 4.17: Comparaison de l'accuracy.

TABLEAU 4.9: Les mesures d'évaluation des performances avec d'autres modèles (MLlib)

Modèles	NB	SVM	MLP	DT	LG	RF
TP	41	40	36	41	40	39
TN	33	35	35	32	36	38
FP	5	5	10	5	6	5
FN	9	8	7	10	6	6
Sensibilité (%)	82,00	83,33	83,72	80,00	86,96	86,67
Spécificité (%)	86,84	87,50	77,78	85,36	85,71	88,37
Accuracy (%)	84.09	85.23	80.68	82.40	86.36	**87.50**

historiques et la visualisation de données. La connexion et l'interaction entre le système et les utilisateurs nécessitent l'investissement d'un ensemble d'infrastructures coûteuses en termes de compétences de programmation, du temps et de l'argent. L'utilisation de Twitter comme chaîne gratuite a grandement simplifiée la communication. Le tableau 4.10 présente une comparaison des architectures du big data existantes dans ce contexte avec l'architecture proposée. Sur la base des résultats, le système proposé peut être appliqué pour résoudre les problèmes d'analyse de données en temps réel pour l'IoT médicale.

TABLEAU 4.10: Comparaison de différentes architectures big data avec l'architecture proposée

Nom de l'architecture	Application	Système big data	L'analyse en temps réel	Observations.
A Real Time Patient Monitoring System for Heart Disease Prediction Using Random Forest Algorithm (Sreejith, Rahul, and Jisha, 2016)	La prédiction de la maladie cardiaque	Non	Non	-N'est pas tester avec du big data. -La prédiction du big data et le stockage de données est toujours le principal défi.

Diabetic data analysis in healthcare using Hadoop architecture over big data (Sampath et al., 2017b)	L'analyse de données du diabète	Oui	Non	-La prédiction est faite après l'étape de stockage de données - Streaming du big data n'est pas effectué. -La prédiction en temps réel n'est pas effectuée. -N'est pas convenable pour le traitement en temps réel.
System Framework for Cardiovascular Disease Prediction Based on Big Data Technology (Han et al., 2017a)	La prédiction de maladie cardiovasculaire	Oui	Non	-La prédiction est faite après l'étape de stockage de données. - Streaming du big data n'est pas effectué. -La prédiction en temps réel n'est pas effectuée. -N'est pas convenable pour le traitement en temps réel.
A novel three-tier Internet of Things architecture with machine learning algorithm for early detection of heart diseases (Kumar and Gandhi, 2018)	La prédiction de la maladie cardiaques	Oui	Non	-La prédiction est faite après l'étape de stockage de données. - Streaming du big data n'est pas effectué. -La prédiction en temps réel n'est pas effectuée. -N'est pas convenable pour le traitement en temps réel.
A new architecture of Internet of Things and big data ecosystem for secured smart healthcare monitoring and alerting system (Manogaran et al., 2018)	La prédiction de la maladie cardiaque	Oui	Non	-L'analyse en temps réel n'est pas effectuée. -La prédiction est faite après l'étape de stockage de données. -N'est pas convenable pour le traitement en temps réel.

Development of Big Data Predictive Analytics Model for Disease Prediction using Machine learning Technique (Venkatesh, Balasubramanian, and Kaliappan, 2019)	La prédiction de la maladie cardiaque	Oui	Non	-Spark streaming et le stockage de données ne sont pas couverts. -N'est pas convenable pour le traitement en temps réel.
Notre architecture	La prédiction de la maladie cardiaque	Oui	Oui	-L'analyse du big data en temps réel est le principal objectif. -La gestion de flux du big data est couverte. -Streaming du big data est couvert. -La prédiction est faite avant l'étape de stockage de données. -L'apprentissage automatique en temps réel est effectué. -Le stockage et la visualisation de données sont effectués. -L'interaction des utilisateurs est bien considérée et il est le principal objectif. -Très convenable pour le traitement en temps réel.

5 Conclusion

La quantité de données sur les soins de santé augmente constamment à un rythme alarmant dans des sources de données différentes et incohérentes. L'amélioration des résultats des patients nécessite la mise en place d'un système évolutif de prévision de l'état de santé en temps réel et une plateforme informatique de streaming. Cependant, ce volume vertigineux de données ne peut plus être collecté, traité, stocké et exploité par les solutions informatiques traditionnelles combinant des infrastructures physiques et des bases de données relationnelles.

Sur la base des défis évoqués précédemment, la technologie de l'information traditionnelle ne permet pas de traiter de données croissantes en évoluant avec du matériel en parallèle.

La réalisation et le développement d'un système de prédiction et d'analyse de données en temps réel en utilisant les outils d'analyse traditionnels exige une grande variété de compétences, des programmes intensifs et plus coûteux, une période de temps considérable et de l'argent. Cependant, lorsqu'on utilise les plateformes et les techniques traditionnelles de traitement de données, il devient difficile de traiter les énormes quantités de données. Par contre l'utilisation des outils open source du big data, en particulier Spark, a considérablement amélioré les performances et l'efficacité du système d'analyse, notamment en termes de temps de développement du système et de complexité des programmes, d'exécution et de la vitesse.

Conclusion générale et perspectives

1 Conclusion générale

Le nombre des patients ne cesse pas d'augmenter, ce qui implique une augmentation de données médicales produites et saisies quotidiennement dans les soins de santé pour amener des traitements et des tests. L'extraction automatique de connaissance qui utilise des techniques de DM et de ML sont capables de fournir une aide à la décision aux cliniciens et de découvrir de nouveaux modèles pertinents à partir des bases de données électroniques sur les patients. L'augmentation de données exige des changements de paradigme des méthodes traditionnelles dans l'application des techniques de DM et de ML sur ces données. En outre, grâce à la disponibilité de différentes technologies cliniques, les caractéristiques des maladies ont été recueillies pour la classification. Par conséquent, la sélection des caractéristiques et l'amélioration de la précision sont devenues une tâche difficile et longue. Pour résoudre ce problème, dans un premier temps, dans le premier chapitre, nous introduisons les méthodes de ML et leur application en santé. Ensuite, nous avons cité quelques challenges liés aux techniques de ML traditionnelles à savoir le volume de données, le nombre des caractéristiques et le traitement. Dans un deuxième temps, dans le chapitre 2, une approche combinant la réduction du nombre des caractéristiques et une méthode de ML plus efficace en termes du diagnostic a été proposée et validée sur le cancer du sein.

Plus récemment, le nombre des patients souffrant des maladies chroniques et d'autres sous surveillance a considérablement augmenté. Il est donc nécessaire d'assurer une solution efficace et évolutive pour prédire efficacement les maladies dans un délai court et très précis en faisant appel aux outils de traitement du big data. À ce stade, après avoir identifié les limites des méthodes traditionnelles, il était nécessaire de fournir une étude comparative actualisée des différents outils utilisés pour le traitement du big data et les mapper aux différents besoins de traitement. Cette étude a fait l'objet du chapitre 3. Ainsi, les performances de six

algorithmes de classification bien connus, à savoir NB, MLP, SVM, DT, RF et LG ont été évaluées pour prédire la maladie cardiaque en utilisant Apache Spark, un moteur rapide et général pour le traitement du big data avec sa bibliothèque d'apprentissage automatique, MLlib, pour le traitement de données en temps réel.

Dans le monde moderne, l'individu se préoccupe beaucoup de rester en bonne santé. Nous avons assisté à des progrès dans le domaine des soins de santé non seulement dans le diagnostic ou le traitement des maladies, mais aussi dans la prévision de la possibilité d'apparition d'une affection ou une détection précoce de celle-ci en se basant sur les bases de données des soins de santé actuellement disponibles. Les données sur les soins de santé varient entre dossiers médicaux électroniques aux données des capteurs portables de santé. Des versions plus avancées et précises de nombreux équipements et dispositifs des soins de santé, comme les glucomètres ou les tensiomètres, qui n'étaient auparavant disponibles que dans les laboratoires médicaux, sont maintenant disponibles dans de nombreux maison. Les capteurs portables de santé sont très populaires aujourd'hui, les montres intelligentes équipés d'un moniteur de fréquence cardiaque et d'accéléromètres sont abordables et acceptables pour tout le monde. La surveillance continue de la santé prend de plus en plus d'importance en tant que mode de vie en raison de la prise de conscience croissante, du fait que la prévention ou la détection précoce des maladies mortelles comme le cancer et la maladie cardiaque peuvent améliorer considérablement le taux de survie.

Souvent, la consultation d'un professionnel de santé se fait pour s'informer de l'état de santé malgré que les valeurs des attributs sont disponibles auprès de la personne, ce qui n'est pas une option envisageable à faire fréquemment. Les systèmes informatiques actuels peuvent prédire l'état de santé avec une précision raisonnable en appliquant des techniques avancées de ML aux données de santé déjà disponibles. A ce stade, dans le dernier chapitre nous avons proposé et développé une nouvelle architecture pour un système évolutif de prévision et d'analyse de l'état de santé en temps réel. Il consiste à collecter, filtrer, gérer, traiter, prédire et stocker les données de flux de santé. Le système proposé utilise Twitter comme source de données, Apache Kafka en tant qu'outil d'ingestion, Apache Spark en tant qu'outil de calcul en ligne, Apache Cassandra en tant que moteur de stockage et Apache Zeppelin comme outil de développement et reporting. DT et RF sont utilisés comme modèles de classifications de données. Le système utilise Twitter comme un canal de communication pour transmettre les données générées par les dispositifs médicaux et effectuer des analyses en temps réel en utilisant le streaming. Ainsi, les utilisateurs de Twitter tweetent des attributs liés à la santé, le système reçoit ces attributs, un modèle de ML est appliqué pour prédire l'état de santé et renvoyer un message de réponse à l'utilisateur. Les résultats expérimentaux montrent que le système proposé est capable de traiter et de prédire efficacement et en temps réel une grande quantité de données médicales rendues possibles par l'IoT à partir de sources de données distribuées et de différentes maladies.

2 Perspectives

Les travaux présentés dans cette thèse ont dégagés plusieurs pistes de recherche. Dans un premier temps, nous allons intégrer d'autres modèles de classification pour améliorer le taux de classification et renforcer le système par une source de données plus simple désigner spécifiquement aux utilisateurs, à savoir une application mobile, capture électronique de données pour le suivi médical etc. Dans un deuxième temps, vu le nombre de données collectées, nous allons proposer des méthodes pour réduire la dimension des bases de données. Nous souhaitons adapté l'architecture proposée aux problèmes introduite en big data et en apprentissage profonds. Finalement nous souhaitons étendre ces travaux dans d'autres domaine d'application telle que l'analyse de trafic routière en temps réel, la détection de fraude et d'intrusion dans les réseaux.

Références

Acharjya, Debi Prasanna and K Ahmed (2016). "A survey on big data analytics: challenges, open research issues and tools". In: *International Journal of Advanced Computer Science and Applications* 7.2, pp. 511–518.

Agrawal, Rakesh, Tomasz Imieliński, and Arun Swami (1993). "Mining association rules between sets of items in large databases". In: *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pp. 207–216.

Agrawal, Rakesh, Ramakrishnan Srikant, et al. (1994). "Fast algorithms for mining association rules". In: *Proc. 20th int. conf. very large data bases, VLDB*. Vol. 1215, pp. 487–499.

Ahamed, Bagrudeen Bazeer, Thirunavukarasu Ramkumar, and Shanmugasundaram Hariharan (2014). "Data integration progression in large data source using mapping affinity". In: *2014 7th International Conference on Advanced Software Engineering and Its Applications*. IEEE, pp. 16–21.

Akhtar, Usman, Asad Masood Khattak, and Sungyoung Lee (2016). "Challenges in managing real-time data in health information system (HIS)". In: *International Conference on Smart Homes and Health Telematics*. Springer, pp. 305–313.

Ali, Syed Mohd et al. (2016). "Big data visualization: Tools and challenges". In: *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*. IEEE, pp. 656–660.

Apache Cassandra (2017). *Official webpage of Apache cassandra*. http://cassandra.apache.org. Online; accessed 15 December 2017.

Apache kafka (2017). https://kafka.apache.org. accessed 15 December 2017.

Apache zeppelin (2017). *Official webpage of Apache zeppelin*. https://zeppelin.apache.org. Online; accessed 15 December 2017.

Arfat, Yasir et al. (2017). "Enabling smarter societies through mobile big data fogs and clouds". In: *Procedia Computer Science* 109, pp. 1128–1133.

Arya, Chandresh and Ritu Tiwari (2016). "Expert system for breast cancer diagnosis: A survey". In: *2016 International Conference on Computer Communication and Informatics (ICCCI)*. IEEE, pp. 1–9.

Aziz, Khadija, Dounia Zaidouni, and Mostafa Bellafkih (2018). "Big data processing using machine learning algorithms: Mllib and mahout use case". In: *Proceedings of the 12th international conference on intelligent systems: theories and applications*, pp. 1–6.

Basco, J Antony and NC Senthilkumar (2017). "Real-time analysis of healthcare using big data analytics". In: *Materials Science and Engineering Conference Series*. Vol. 263. 4, p. 042056.

Bauer H, Patel M, Veira J (2017). *The Internet of Things: sizing up the opportunity*. http://www.mckinsey.com/. accessed 15 December 2017.

Bhardwaj, Arpit and Aruna Tiwari (2015a). "Breast cancer diagnosis using genetically optimized neural network model". In: *Expert Systems with Applications* 42.10, pp. 4611–4620.

— (2015b). "Breast cancer diagnosis using genetically optimized neural network model". In: *Expert Systems with Applications* 42.10, pp. 4611–4620.

Bhargava, Sandeep, Dinesh Goyal, and Bright Keswani (2019). "Performance Comparison of Big Data Analytics Platforms". In: *International Journal of Engineering, Applied and Management Sciences Paradigms (IJEAM), ISSN*, pp. 2320–6608.

Bhosale, Harshawardhan S and Devendra P Gadekar (2014). "A review paper on big data and hadoop". In: *International Journal of Scientific and Research Publications* 4.10, pp. 1–7.

Borthakur, Dhruba et al. (2008). "HDFS architecture guide". In: *Hadoop apache project* 53.1-13, p. 2.

Brause, Rüdiger W (2001). "Medical analysis and diagnosis by neural networks". In: *International Symposium on Medical Data Analysis*. Springer, pp. 1–13.

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Breiman, Leo et al. (1984). *Classification and regression trees*. CRC press.

Chen, Chien-Hsing (2014). "A hybrid intelligent model of analyzing clinical breast cancer data using clustering techniques with feature selection". In: *Applied Soft Computing* 20, pp. 4–14.

Chen, Min et al. (2017). "Disease prediction by machine learning over big data from healthcare communities". In: *Ieee Access* 5, pp. 8869–8879.

Chen, Pengyu (2019). "Visualization of real-time monitoring datagraphic of urban environmental quality". In: *EURASIP Journal on Image and Video Processing* 2019.1, p. 42.

Chintapalli, Sanket et al. (2016). "Benchmarking streaming computation engines: Storm, flink and spark streaming". In: *2016 IEEE international parallel and distributed processing symposium workshops (IPDPSW)*. IEEE, pp. 1789–1792.

Cortes, Corinna and Vladimir Vapnik (1995). "Support-vector networks". In: *Machine learning* 20.3, pp. 273–297.

Cunha, Joao, Catarina Silva, and Mário Antunes (2015). "Health twitter big bata management with hadoop framework". In: *Procedia Computer Science* 64, pp. 425–431.

Curry, Edward, Freitas, et al. (2010). "The role of community-driven data curation for enterprises". In: *Linking enterprise data*. Springer, pp. 25–47.

Das, Deb Dutta et al. (2017). "Sentimental analysis for airline twitter data". In: *IOP Conference Series: Materials Science and Engineering*. Vol. 263. 4. IOP Publishing, p. 042067.

De Mauro, Andrea, Marco Greco, and Michele Grimaldi (2016). "A formal definition of Big Data based on its essential features". In: *Library Review*.

Dean, Jeffrey and Sanjay Ghemawat (2008). "MapReduce: simplified data processing on large clusters". In: *Communications of the ACM* 51.1, pp. 107–113.

Dua, Sumeet, U Rajendra Acharya, and Prerna Dua (2014). *Machine learning in healthcare informatics*. Vol. 56. Springer.

Dutta, Kaushik (2017). "Distributed computing technologies in Big Data analytics". In: *Distributed computing in Big Data analytics*. Springer, pp. 57–82.

Esfandiari, Nura et al. (2014). "Knowledge discovery in medicine: Current issue and future trend". In: *Expert Systems with Applications* 41.9, pp. 4434–4463.

Fayyad, Usama, Gregory Piatetsky-Shapiro, and Padhraic Smyth (1996). "From data mining to knowledge discovery in databases". In: *AI magazine* 17.3, pp. 37–37.

Feldman, Dan, Melanie Schmidt, and Christian Sohler (2013). "Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering". In: *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, pp. 1434–1453.

Feroz, Mohammed Nazim and Susan Mengel (2014). "Examination of data, rule generation and detection of phishing URLs using online logistic regression". In: *2014 IEEE International Conference on Big Data (Big Data)*. IEEE, pp. 241–250.

Garapati, Swarna Latha and Sridevi Garapati (2018). "Application of big data analytics: an innovation in health care". In: *Comput Intell* 14.1, pp. 15–27.

García, Salvador et al. (2016). "Big data preprocessing: methods and prospects". In: *Big Data Analytics* 1.1, p. 9.

Gouda, Karam and Mohammed Javeed Zaki (2001). "Efficiently mining maximal frequent itemsets". In: *Proceedings 2001 IEEE International Conference on Data Mining*. IEEE, pp. 163–170.

Graunt, John (1939). *Natural and political observations made upon the bills of mortality*. 2. The Johns Hopkins Press.

Han, Sang et al. (2017a). "System framework for cardiovascular disease prediction based on big data technology". In: *Symmetry* 9.12, p. 293.

Han, Sang Hun et al. (2017b). "System framework for cardiovascular disease prediction based on big data technology". In: *Symmetry* 9.12, p. 293.

Hassan, Mahmudul and Srividya K Bansal (2018). "Semantic data querying over NoSQL databases with Apache Spark". In: *2018 IEEE International Conference on Information Reuse and Integration (IRI)*. IEEE, pp. 364–371.

Hausenblas, Michael and Nathan Bijnens (2015). "Lambda architecture". In: *URL: http://lambda-architecture. net/. Luettu* 6, p. 2014.

Haykin, Simon (1994). *Neural networks: a comprehensive foundation*. Prentice Hall PTR.

Hazarika, Akaash Vishal, G Jagadeesh Sai Raghu Ram, and Eeti Jain (2017). "Performance comparision of Hadoop and spark engine". In: *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. IEEE, pp. 671–674.

Herland, Matthew, Taghi M Khoshgoftaar, and Randall Wald (2014a). "A review of data mining using big data in health informatics". In: *Journal of Big data* 1.1, pp. 1–35.

— (2014b). "A review of data mining using big data in health informatics". In: *Journal of Big Data* 1.1, p. 2.

Hermon, Rebecca and Patricia AH Williams (2014). "Big data in healthcare: What is it used for?" In:

Hosmer Jr, David W, Stanley Lemeshow, and Rodney X Sturdivant (2013). *Applied logistic regression*. Vol. 398. John Wiley & Sons.

Hunt, Patrick et al. (2010). "ZooKeeper: Wait-free Coordination for Internet-scale Systems." In: *USENIX annual technical conference*. Vol. 8. 9. Boston, MA, USA.

Karabatak, Murat and M Cevdet Ince (2009). "An expert system for detection of breast cancer based on association rules and neural network". In: *Expert systems with Applications* 36.2, pp. 3465–3469.

Khan, Zaheer et al. (2015). "Towards cloud based big data analytics for smart future cities". In: *Journal of Cloud Computing* 4.1, pp. 1–11.

Kim, Juhyeon and Hyunjung Shin (2013). "Breast cancer survivability prediction using labeled, unlabeled, and pseudo-labeled patient data". In: *Journal of the American Medical Informatics Association* 20.4, pp. 613–618.

Kim, Tae Woong et al. (2014). "A big data framework for u-healthcare systems utilizing vital signs". In: *2014 International Symposium on Computer, Consumer and Control*. IEEE, pp. 494–497.

Kodratoff, Yves (1998). "techniques et outils de l'extraction de connaissances à partir des données". In: *Signaux* 92.38-43, p. 10.

Kumar, Priyan Malarvizhi and Usha Devi Gandhi (2018). "A novel three-tier Internet of Things architecture with machine learning algorithm for early detection of heart diseases". In: *Computers & Electrical Engineering* 65, pp. 222–235.

Kumar, Sachin and Mikhail Zymbler (2019). "A machine learning approach to analyze customer satisfaction from airline tweets". In: *Journal of Big Data* 6.1, p. 62.

Lakshman, Avinash and Prashant Malik (2010). "Cassandra: a decentralized structured storage system". In: *ACM SIGOPS Operating Systems Review* 44.2, pp. 35–40.

Laney, Doug (2001). "3D data management: Controlling data volume, velocity, and variety. Application Delivery Strategies". In: *Recuperado de http://blogs. gartner. com/doug-laney/files/2012-/01/ad949-3D-Data-Management-Con-trolling-Data-Volume-Velocity-and-Variety. pdf*.

Lee, Kathy, Ankit Agrawal, and Alok Choudhary (2013). "Real-time disease surveillance using twitter data: demonstration on flu and cancer". In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1474–1477.

Lee, Kyong-Ha et al. (2012). "Parallel data processing with MapReduce: a survey". In: *AcM sIGMoD Record* 40.4, pp. 11–20.

Lewis, David D (1998). "Naive (Bayes) at forty: The independence assumption in information retrieval". In: *European conference on machine learning*. Springer, pp. 4–15.

Liu, Rong et al. (2014). "Big Data architecture for IT incident management". In: *Proceedings of 2014 IEEE International Conference on Service Operations and Logistics, and Informatics*. IEEE, pp. 424–429.

Liu, Xiao et al. (2017). "A hybrid classification system for heart disease diagnosis based on the RFRS method". In: *Computational and mathematical methods in medicine* 2017.

Manogaran, Gunasekaran and Daphne Lopez (2017). "A survey of big data architectures and machine learning algorithms in healthcare". In: *International Journal of Biomedical Engineering and Technology* 25.2-4, pp. 182–211.

— (2018). "Health data analytics using scalable logistic regression with stochastic gradient descent". In: *International Journal of Advanced Intelligence Paradigms* 10.1-2, pp. 118–132.

Manogaran, Gunasekaran et al. (2018). "A new architecture of Internet of Things and big data ecosystem for secured smart healthcare monitoring and alerting system". In: *Future Generation Computer Systems* 82, pp. 375–387.

Masethe, Hlaudi Daniel and Mosima Anna Masethe (2014a). "Prediction of heart disease using classification algorithms". In: *Proceedings of the world congress on Engineering and Computer Science*. Vol. 2, pp. 22–24.

— (2014b). "Prediction of heart disease using classification algorithms". In: *Proceedings of the world Congress on Engineering and computer Science*. Vol. 2, pp. 22–24.

Mathew, Prabha Susy and Anitha S Pillai (2016). "Big data challenges and solutions in healthcare: a survey". In: *Innovations in bio-inspired computing and applications*. Springer, pp. 543–553.

Mazumder, Sourav (2016). "Big data tools and platforms". In: *Big Data Concepts, Theories, and Applications*. Springer, pp. 29–128.

Mert, Ahmet et al. (2015). "Breast cancer detection with reduced feature set". In: *Computational and mathematical methods in medicine* 2015.

Mohammed, Ahmed Fuad, Vikas T Humbe, and Santosh S Chowhan (2016). "A review of big data environment and its related technologies". In: *2016 International Conference on Information Communication and Embedded Systems (ICICES)*. IEEE, pp. 1–5.

Mohktar, Mas S et al. (2015). "Predicting the risk of exacerbation in patients with chronic obstructive pulmonary disease using home telehealth measurement data". In: *Artificial intelligence in medicine* 63.1, pp. 51–59.

Nahato, Kindie Biredagn, Khanna Nehemiah Harichandran, and Kannan Arputharaj (2015). "Knowledge mining from clinical datasets using rough sets and backpropagation neural network". In: *Computational and mathematical methods in medicine* 2015.

Nair, Lekha R, Sujala D Shetty, and Siddhanth D Shetty (2018). "Applying spark based machine learning model on streaming big data for health status prediction". In: *Computers & Electrical Engineering* 65, pp. 393–399.

Nayak, Aparna, MM Manohara Pai, and Radhika M Pai (2016). "Prediction models for Indian stock market". In: *Procedia Computer Science* 89, pp. 441–449.

Neuvirth, Hani et al. (2011). "Toward personalized care management of patients at risk: the diabetes case study". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 395–403.

Nguyen, Quang, Hamed Valizadegan, and Milos Hauskrecht (2014). "Learning classification models with soft-label information". In: *Journal of the American Medical Informatics Association* 21.3, pp. 501–508.

Nightingale, Florence (1863). *Notes on hospitals*. Longman, Green, Longman, Roberts, and Green.

Nilashi, Mehrbakhsh et al. (2017). "A knowledge-based system for breast cancer classification using fuzzy logic method". In: *Telematics and Informatics* 34.4, pp. 133–144.

Osman, Ahmed M Shahat (2019). "A novel big data analytics framework for smart cities". In: *Future Generation Computer Systems* 91, pp. 620–633.

Pan, Shirui, Jia Wu, and Xingquan Zhu (2015). "CogBoost: Boosting for fast cost-sensitive graph classification". In: *IEEE Transactions on Knowledge and Data Engineering* 27.11, pp. 2933–2946.

Paulin, F and A Santhakumaran (2011). "Classification of breast cancer by comparing back propagation training algorithms". In: *International Journal on Computer Science and Engineering* 3.1, pp. 327–332.

Peng, Zhihao (2019). "Stocks Analysis and Prediction Using Big Data Analytics". In: *2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*. IEEE, pp. 309–312.

Piatetsky-Shapiro, Gregory (1991). "Discovery, analysis, and presentation of strong rules". In: *Knowledge discovery in databases*, pp. 229–238.

Prasad, Yamuna and Kanad K Biswas (2010). "PSO-SVM based classifiers: a comparative approach". In: *International Conference on Contemporary Computing*. Springer, pp. 241–252.

Qian, Buyue et al. (2015). "A relative similarity based method for interactive patient risk prediction". In: *Data Mining and Knowledge Discovery* 29.4, pp. 1070–1093.

Quinlan, J Ross (2014). *C4. 5: programs for machine learning*. Elsevier.

Raghupathi, Wullianallur and Viju Raghupathi (2014). "Big data analytics in healthcare: promise and potential". In: *Health information science and systems* 2.1, p. 3.

Rahman, Mohammad Naimur, Amir Esmailpour, and Junhui Zhao (2016). "Machine learning with big data an efficient electricity generation forecasting system". In: *Big Data Research* 5, pp. 9–15.

Rallapalli, Sreekanth, RR Gondkar, and Golajapu Venu Madhava Rao (2016). "Cloud based k-means clustering running as a Mapreduce job for big data healthcare analytics using Apache mahout". In: *Information systems design and intelligent applications*. Springer, pp. 127–135.

Rallapalli, Sreekanth and T Suryakanthi (2016). "Predicting the risk of diabetes in big data electronic health Records by using scalable random forest classification algorithm". In: *2016 International Conference on Advances in Computing and Communication Engineering (ICACCE)*. IEEE, pp. 281–284.

Rathore, M Mazhar et al. (2016). "Real-time medical emergency response system: exploiting IoT and big data for public health". In: *Journal of medical systems* 40.12, p. 283.

Rathore, M Mazhar et al. (2017). "Hadoop-Based Intelligent Care System (HICS) Analytical Approach for Big Data in IoT". In: *ACM Transactions on Internet Technology (TOIT)* 18.1, pp. 1–24.

Rehman, Muhammad Habib ur et al. (2016). "Big data reduction framework for value creation in sustainable enterprises". In: *International Journal of Information Management* 36.6, pp. 917–928.

Rohini, TV (2016). "Market Price Prediction Based on Neural Network Using Hadoop MapReduce Technique". In: *email: drssridhar@ yahoo. com* 89, pp. 441–449.

Saggi, Mandeep Kaur and Sushma Jain (2018). "A survey towards an integration of big data analytics to big insights for value-creation". In: *Information Processing & Management* 54.5, pp. 758–790.

Sampath, P et al. (2017a). "Diabetic data analysis in healthcare using Hadoop architecture over big data". In: *International Journal of Biomedical Engineering and Technology* 23.2-4, pp. 137–147.

— (2017b). "Diabetic data analysis in healthcare using Hadoop architecture over big data". In: *International Journal of Biomedical Engineering and Technology* 23.2-4, pp. 137–147.

Samuel, Arthur L (1959). "Some studies in machine learning using the game of checkers". In: *IBM Journal of research and development* 3.3, pp. 210–229.

Sangaiah, Ilangovan and A Vincent Antony Kumar (2019). "Improving medical diagnosis performance using hybrid feature selection via relieff and entropy based genetic search (RF-EGA) approach: application to breast cancer prediction". In: *Cluster Computing* 22.3, pp. 6899–6906.

Santhanam, T and MS Padmavathi (2014). "Comparison of K-Means clustering and statistical outliers in reducing medical datasets". In: *2014 International Conference on Science Engineering and Management Research (ICSEMR)*. IEEE, pp. 1–6.

Sarkar, Bidyut Biman et al. (2016). "Personal health record management system using hadoop framework: An application for smarter health care". In: *International Workshop Soft Computing Applications*. Springer, pp. 385–393.

Sehgal, Divya and Ambuj Kumar Agarwal (2018). *Real-time sentiment analysis of big data applications using Twitter data with Hadoop framework*.

Silipo, Rosaria and Phil Winters (2013). "Big data, smart energy, and predictive analytics". In: *Time Series Prediction of Smart Energy Data* 1, p. 37.

Simon, György J et al. (2013). "Extending association rule summarization techniques to assess risk of diabetes mellitus". In: *IEEE Transactions on Knowledge and Data Engineering* 27.1, pp. 130–141.

Sreejith, S, S Rahul, and RC Jisha (2016). "A real time patient monitoring system for heart disease prediction using random forest algorithm". In: *Advances in Signal Processing and Intelligent Recognition Systems*. Springer, pp. 485–500.

Stoean, Ruxandra and Catalin Stoean (2013). "Modeling medical decision making by support vector machines, explaining by rules of evolutionary algorithms with feature selection". In: *Expert Systems with Applications* 40.7, pp. 2677–2686.

Storey, Veda C and Il-Yeol Song (2017). "Big data technologies and management: What conceptual modeling can do". In: *Data & Knowledge Engineering* 108, pp. 50–67.

Sun, Dongdong, Minghui Wang, and Ao Li (2018). "A multimodal deep neural network for human breast cancer prognosis prediction by integrating multi-dimensional data". In: *IEEE/ACM transactions on computational biology and bioinformatics* 16.3, pp. 841–850.

Sun, Jimeng and Chandan K Reddy (2013). "Big data analytics for healthcare". In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1525–1525.

Suthaharan, Shan (2014). "Big data classification: Problems and challenges in network intrusion prediction with machine learning". In: *ACM SIGMETRICS Performance Evaluation Review* 41.4, pp. 70–73.

Sweilam, Nasser H, AA Tharwat, and NK Abdel Moniem (2010). "Support vector machine for diagnosis cancer disease: A comparative study". In: *Egyptian Informatics Journal* 11.2, pp. 81–92.

Ta, Van-Dai, Chuan-Ming Liu, and Goodwill Wandile Nkabinde (2016). "Big data stream computing in healthcare real-time analytics". In: *2016 IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*. IEEE, pp. 37–42.

Tartar, Ahmet, Niyazi Kilic, and Aydin Akan (2013). "Classification of pulmonary nodules by using hybrid features". In: *Computational and Mathematical Methods in Medicine* 2013.

Thommandram, Anirudh et al. (2013). "Classifying neonatal spells using real-time temporal analysis of physiological data streams: Algorithm development". In: *2013 IEEE Point-of-Care Healthcare Technologies (PHT)*. IEEE, pp. 240–243.

Tomar, Divya and Sonali Agarwal (2013a). "A survey on Data Mining approaches for Healthcare". In: *International Journal of Bio-Science and Bio-Technology* 5.5, pp. 241–266.

— (2013b). "A survey on Data Mining approaches for Healthcare". In: *International Journal of Bio-Science and Bio-Technology* 5.5, pp. 241–266.

Tran, Truyen et al. (2013). "An integrated framework for suicide risk prediction". In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1410–1418.

Tran, Truyen et al. (2015). "Stabilized sparse ordinal regression for medical risk stratification". In: *Knowledge and Information Systems* 43.3, pp. 555–582.

UCI (2017). *UCI*. https://archive.ics.uci.edu/ml/datasets/heart+Disease/. Online; accessed 15 December 2017.

Vapnik, Vladimir (2013). *The nature of statistical learning theory*. Springer science & business media.

Vavilapalli, Vinod Kumar et al. (2013). "Apache hadoop yarn: Yet another resource negotiator". In: *Proceedings of the 4th annual Symposium on Cloud Computing*, pp. 1–16.

Venkatesh, R, C Balasubramanian, and M Kaliappan (2019). "Development of Big Data Predictive Analytics Model for Disease Prediction using Machine learning Technique". In: *Journal of medical systems* 43.8, p. 272.

Wei, Jin-Mao, Shu-Qin Wang, and Xiao-Jie Yuan (2009). "Ensemble rough hypercuboid approach for classifying cancers". In: *IEEE Transactions on Knowledge and Data Engineering* 22.3, pp. 381–391.

WIlliam H. Wolberg (2017). *Breast Cancer Wisconsin (Original) Data Set*. https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/. accessed 15 December 2017.

Xu, Gaowei et al. (2016). "User behavior prediction model for smart home using parallelized neural network algorithm". In: *2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. IEEE, pp. 221–226.

Xu, Kaihe et al. (2015). "Privacy-preserving machine learning algorithms for big data systems". In: *2015 IEEE 35th international conference on distributed computing systems*. IEEE, pp. 318–327.

Xu, Ke et al. (2014). "A MapReduce based parallel SVM for email classification". In: *Journal of Networks* 9.6, p. 1640.

Yadranjiaghdam, Babak, Nathan Pool, and Nasseh Tabrizi (2016). "A survey on real-time big data analytics: applications and tools". In: *2016 international conference on computational science and computational intelligence (CSCI)*. IEEE, pp. 404–409.

Yu, Feng et al. (2018). "Stock Transaction Analysis System Based on Hadoop and Capital Flow". In: *2018 Sixth International Conference on Advanced Cloud and Big Data (CBD)*. IEEE, pp. 54–59.

Zaharia, Matei et al. (2010). "Spark: Cluster computing with working sets." In: *HotCloud* 10.10-10, p. 95.

Zaharia, Matei et al. (2012a). "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing". In: *Presented as part of the 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12)*, pp. 15–28.

Zaharia, Matei et al. (2013). "Discretized streams: Fault-tolerant streaming computation at scale". In: *Proceedings of the twenty-fourth ACM symposium on operating systems principles*, pp. 423–438.

Zaharia Mosharaf Chowdhury, Tathagata Das Ankur Dave Justin Ma Murphy McCauley Michael J. Franklin Scott Shenker et al. (2012b). "A Fault-Tolerant Abstraction for In-Memory Cluster Computing". In: *NSDI* 12.

Zemmal, Nawel et al. (2016). "Adaptive semi supervised support vector machine semi supervised learning with features cooperation for breast cancer classification". In: *Journal of Medical Imaging and Health Informatics* 6.1, pp. 53–62.

Zhai, Yiteng, Yew-Soon Ong, and Ivor W Tsang (2014). "The emerging" big dimensionality"". In: *IEEE Computational Intelligence Magazine* 9.3, pp. 14–26.

Zhao, Tingzhi et al. (2014). "Detecting abnormal patterns of daily activities for the elderly living alone". In: *International Conference on Health Information Science*. Springer, pp. 95–108.

Zheng, Bichen, Sang Won Yoon, and Sarah S Lam (2014). "Breast cancer diagnosis based on feature extraction using a hybrid of K-means and support vector machine algorithms". In: *Expert Systems with Applications* 41.4, pp. 1476–1482.

كلية العلوم و التقنيات فاس
ⵜⴰⵖⵉⵡⴰⵏⵜ ⵏ ⵜⵎⴰⵙⵙⴰⵏⵉⵏ ⴷ ⵜⵙⵏⵉⴳⵉⵜⵉⵏ
Faculté des Sciences et Techniques de Fès

جامعة سيدي محمد بن عبد الله
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵙⵉⴷⵉ ⵎⵓⵃⵎⵎⴰⴷ ⴱⵏ ⵄⴱⴷⵓⵍⵍⴰⵀ
Université Sidi Mohamed Ben Abdellah

Centre d'Etudes Doctorales : Sciences et Techniques et Sciences Médicales

Titre de la thèse: Machine learning pour l'aide au diagnostic médical en temps réel dans un environnement big data.

Nom du candidat: Abderrahmane ED-DAOUDY

Spécialité: Informatique

Résumé de la thèse

Dans cette thèse, nous présentons un certain nombre de solutions dédiées au diagnostic médical en termes d'architectures, de méthodologies et d'applications autour de deux grands paradigmes, à savoir le Data Mining (DM) avec Machine Learning (ML) et les technologies big data. Le premier volet de ce travail consiste à améliorer le taux de classification en intégrant des méthodes de sélection des attributs les plus significatifs. Pour ce faire, la méthode des règles d'association (Association Rules) a été utilisée pour éliminer les attributs non significatifs, puis un ensemble de modèles de classification les plus connus ont été appliqués pour distinguer l'état de santé. La méthode proposée a été validée sur le problème de classification des tumeurs. Le deuxième volet est consacré à la proposition d'une architecture du système de diagnostic et de suivi médicale introduisant à la fois les technologies émergentes de l'écosystème big data et ML parallèle et distribué. Il s'agit d'incorporer les nouvelles technologies du big data, distribuées et parallèle à la fois, afin d'apporter des solutions robustes, rapides et évolutives aux exigences de la collection, du calcul intensif, de l'analyse prédictive, du stockage et de la visualisation de données médicales. Pour ce faire, la démarche méthodologique et la réalisation du système sont présentées, ainsi que le déploiement d'une architecture de traitement parallèle et distribuée sur un cluster de nœuds combinant les technologies récentes du big data.

Pour valider notre approche, nous avons testé le système sur la maladie cardiaque et le diabète. Tout d'abord, nous transformons les algorithmes traditionnels en des algorithmes parallèles, distribués, évolutifs et rapides. Deuxièmement, en utilisant Spark streaming, le modèle le plus performant est appliqué au flux de données provenant de sources distribuées pour prédire l'état de santé en temps réel. Le résultat ainsi que le flux de données seront stockés dans une base de données distribuée en vue de mettre en œuvre le reporting et la surveillance en temps réel. Dans un premier lieu, pour tester notre système en termes de sources et du flux de données, nous avons opté pour une application de simulation agissante comme flux de données générés par différentes technologies cliniques (IoT). Dans un deuxième temps, nous intégrons le Twitter qui sert du canal de communication gratuit. Les utilisateurs tweetent leurs attributs liés à la santé. Le système reçoit tous les attributs souhaités, puis le modèle de ML est appliqué sur le flux de données pour prédire l'état de santé et envoyer un message approprié indiquant l'absence ou la présence de maladie. Les résultats montrent de meilleures performances et ont le potentiel d'ouvrir de nouvelles opportunités et perspectives en ML parallèle et distribué dans un environnement big data.

Mots-clés : Data mining, Big data, Machine learning parallèle et distribué, Temps réel, Classification, Apache Spark, Soins de santé, Traitement des Tweets, Traitement de flux, Internet des objets.