Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
Information Technology

A ROBUST NETWORK IMPLEMENTATION FOR HADOOP AND MACHINE LEARNING IN HEALTHCARE

A Thesis Submitted

to the Council of the College of Information Technology for Postgraduate Studies of University of Babylon in Partial Fulfillment of the Requirements for the Degree of Master in Information Technology / Information Networks.

By

Mukalad Faleh Hassan Ali

Supervised by

Asst.Prof.Dr. Furkan Hassan Saleh

Asst. Prof.Dr. Mehdi Ebady Manaa Mehdi

2021 A.D. **1443 A.H**

بسم الله الرحمن الرحيم

يرفع الله الذين آمنوا منكم
والذين أوتوا العلم درجات والله
بما تعملون خبير

صدق الله العظيم

سورة المجادلة . آية 11

Certification of the Examination Committee

We hereby certify that we have studied the thesis entitled (**A ROBUST NETWORK IMPLEMENTATION FOR HADOOP AND MACHINE LEARNING IN HEALTHCARE**) presented by the student (**MUKALAD FALEH HASSAN ALI**) and examined her in its content and what is related to it, and that, in our opinion, it is adequate with (**Excellent**) standing as a thesis for the degree of Master in Information Technology-Information Networks.

Signature:
Name: Zaki Saeed Tywofik
Title: Asst. Prof. Dr.
Date: / / 2021
(**Chairman**)

Signature:
Name: Mahdi Salih Neama Almhanna
Title: Asst. Prof. Dr.
Date: / / 2021
(**Member**)

Signature:
Name: Tariq Alwan Kadhum
Title: Lecturer
Date: / / 2021
(**Member**)

Signature:
Name: Mehdi Ebady Manaa
Title: Asst. Prof. Dr.
Date: / / 2021
(**Member and Supervisor**)

Signature:
Name: Furkan Hassan Saleh
Title: Asst. Prof. Dr.
Date: / / 2021
(**Member and Supervisor**)

Approved by the Dean of the College of Information Technology, University of Babylon.

Signature:
Name: Hussein Atiyah. Lafta
Title: Professor
Date: / / 2021
(**Dean of Collage of Information Technology**)

Supervisor Certification

I certify that the thesis entitled (**A ROBUST NETWORK IMPLEMENTATION FOR HADOOP AND MACHINE LEARNING IN HEALTHCARE**) was prepared under my supervision at the department of Information Networks/ College of Information Technology / University of Babylon as partial fulfillment of the requirements of the degree of Master in Information Technology- Information Networks.

Signature:	Signature:
Supervisor Name: **Asst. Prof.Dr. Mehdi Ebady Manaa**	Supervisor Name: **Asst.Prof.Dr. Furkan Hassan Saleh**
Date: / /2021	Date: / /2021

The Head of the Department Certification

In view of the available recommendations, I forward the thesis entitled "**A ROBUST NETWORK IMPLEMENTATION FOR HADOOP AND MACHINE LEARNING IN HEALTHCARE"** for debate by the examination committee.

Signature:

Prof. Dr. **Saad Talib Hasson Aljebori**

Head of Information Networks Department

Date: / /2021

Declaration

I hereby declare that this Dissertation, submitted to the University of Babylon in partial fulfillment of requirement for the degree of Master of Information Technology-Information Networks has not been submitted as an exercise for a similar degree at any other university. I also certify that this work described here is entirely my own except for experts and summaries whose sources are appropriately cited in the references.

Signature:

Name : **Mukalad Faleh Hassan Ali**

Date: **/ / 2021**

Dedication

I dedicate this thesis

To the soul of my father

To the fountain of patience my mother

To my greatest lovely

To my supervisor

To my family

To my friends

Researcher

Acknowledgement

In the name of God, Most Gracious, Most Merciful, At first, Praise be to God and thanks to God and the satisfaction of parents and conciliation only from God greatest praise is to **Allah** for His assistance in facing the difficulty that I met in my study, and for always helping me to achieve my aims, also for His great graces and boons all the time.

I would like to express my deepest thanks to my supervisor **Asst. Prof.Dr. Mehdi Ebady Manaa** and **Asst.Prof.Dr. Furkan Hassan Saleh** for their valuable advice, motivation, guidance, and for so many fruitful discussions throughout the preparation of this thesis.

I would like to extend my respect and deepest gratitude to the College of Information Technology.

Sincere appreciation and love go to my family, my father's soul who he was always be with me in my heart and my dear mother that whatever I did to her will not reward her they provide me with optimism and pure affection and they give me great hope, encouragement and they have stood with me in every step in this research. I dedicate this work and give special thanks to those who encouraged me to continue my scientific career, my wife, and my children.

Finally, Sincere thanks and appreciation to all friends, colleagues and loved ones

Researcher

Abstract

Stroke is the third most common cause of death and the most common cause of long-term disability among adults around the world. Therefore, stroke prediction and diagnosis is a very important issue. Early awareness of different warning signs of stroke can minimize the stroke. Data mining techniques come in handy to help determine the correlations between individual patient characterization data, extract from the medical information system the knowledge necessary to predict and treat various diseases.

The implemented system aims to maximize the accuracy to predict stroke disease and minimize time to build a model using the Hadoop Map-Reduce programming model with machine learning, where time affects the computation parameters of the system. The proposed approach considered gender, age, hypertension, heart disease, smoking status feature attributes to predict stroke. It based on three case studies the 1st case study is based on data mining/machine learning with DT, SVM, and RF respectively. The 2nd case study is a data mining/ Hadoop-Count / machine learning and the 3rd case study is a data mining / Hadoop-Weight / machine learning with NB, SVM, and DT machine learning algorithms for both Hadoop states.

The performance evaluation reveals that 2nd and 3rd case studies Hadoop count / Hadoop–Weight respectively provided the highest accuracy of about 98.646 % of Naïve Bays, DT, and SVM, with decreased number of instances of the dataset from 43400 records into 2585 records of the testing dataset and decreased time to build system with Hadoop/Weight process into NB as 28 ms, SVM as 371 ms, and DT as 473ms compared with the other related works in the same field and the same dataset(Big Healthcare Stroke Dataset) as it gives the best accuracy for the prediction of stroke disease. Besides, it improved the data mining case through count/weight of Hadoop by increasing consistency among attributes. Alongside, the proportion of the model's predictions of whether a class is a stroke or not stroke showed the high Detection Rate (DR) results from Hadoop/Count - Hadoop/Weight is about 100% of NB, SVM, and DT algorithms, alongside, Recall as a measure of success stroke prediction quantity about 100% from Hadoop/Count - Hadoop/Weight of NB, SVM, and DT algorithms. Also, the false-positive rate (7) and false-negative rate (0) of the Hadoop studies are the lowest compared to other research directions based on machine learning algorithms.

Table of Contents

CHAPTER ONE GENERAL INTRODUCTION 1

CHAPTER TWO THEORETICAL BACKGROUND

List of Tables

Chapter 1

Chapter 2

Chapter 3

Chapter 4

List of Figures

Chapter 1

Chapter 2

Chapter 3

Chapter 4

List of Algorithms

Chapter 2

Chapter 3

List of Abbreviations

Abbreviation	Description
ANN	Artificial Neural Networks
AUC	Area Under the Curve
DNN	Deep Neural Network
DR	Detection Rate
DT	Decision Tree
FAR	False Alert Rate
FN	False Negative
FP	False Positive
HDFS	Hadoop Distributed File System
IDF	Inverse Document Frequency
IoT	Internet of things
JDK	Java Development Kit
JRE	Java Runtime Environment
KDD	Knowledge Discovery in Databases
KNN	k -nearest neighbor
MLR	Multiple Linear Regression
NB	Naive Bayes
NN	Neural Network
PLR	logistic regression
RF	Random Forest
RMSE	Root Mean Squared Error
SGD	Stochastic Gradient Boosting
SVM	Support Vector Machine
TF	Term Frequency
TN	True Negative
TP	True Positive
TPR	True positive rate

List of Symbols	
Symbol	**Description**
$\mathbf{x_i}$	Observations Time series
$\hat{\mathbf{x}}_{\mathbf{i}}$	Forecasted Time set
$\mathbf{X_{obs}}$	Observed Values for RMSE
$\mathbf{X_{model}}$	Modeled Values at time/place

List of Thesis Related Publications

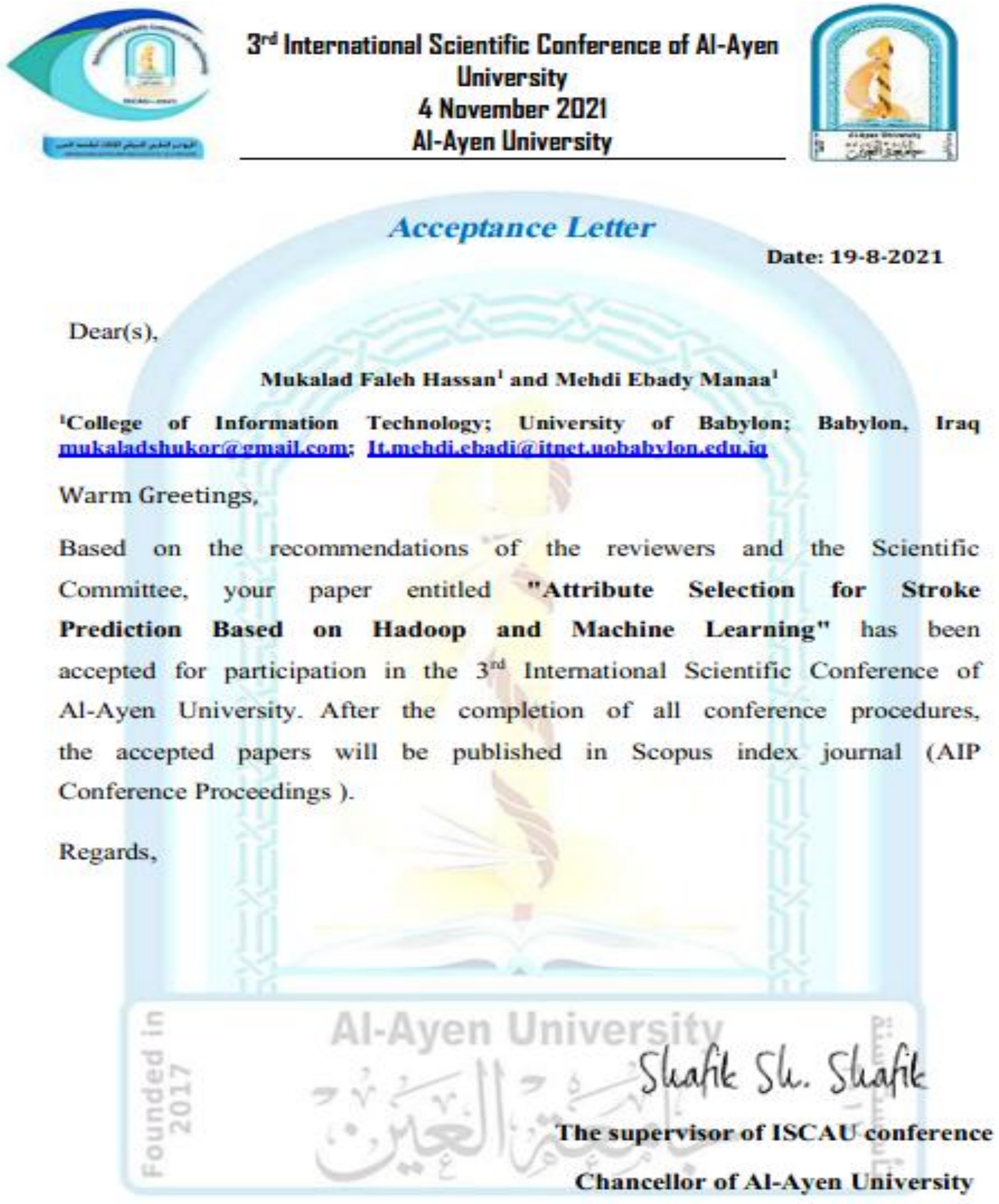
Name of Conference: The 3rd International Scientific Conference of Al-Ayen University, published in Scopus index journal (International Journal of Modern Education and Computer Science), journal (AIP Conference Proceedings).

- **1st Paper Title: Attribute Selection for Stroke Prediction Based on Hadoop and Machine Learning.**
- **2nd Paper Title: Big Data Processing with Hadoop and Data Mining**
- **Authors:**
  - Mukalad Faleh Hassan
  - Asst. Prof.Dr. Mehdi Ebady Manaa

Information Network Department, Information Technology College, Babylon University.

Email: mukaladshukor@gmail.com

Email: It.mehdi.ebadi@itnet.uobabylon.edu.iq



3rd International Scientific Conference of Al-Ayen University
4 November 2021
Al-Ayen University

Acceptance Letter

Date: 19-8-2021

Dear(s),

Mukalad Faleh Hassan[1] and Mehdi Ebady Manaa[1]

[1]College of Information Technology; University of Babylon; Babylon, Iraq
mukaladshukor@gmail.com; It.mehdi.ebadi@itnet.uobabylon.edu.iq

Warm Greetings,

Based on the recommendations of the reviewers and the Scientific Committee, your paper entitled **"Attribute Selection for Stroke Prediction Based on Hadoop and Machine Learning"** has been accepted for participation in the 3rd International Scientific Conference of Al-Ayen University. After the completion of all conference procedures, the accepted papers will be published in Scopus index journal (AIP Conference Proceedings).

Regards,

Shafik Sh. Shafik

The supervisor of ISCAU conference

Chancellor of Al-Ayen University

The 3rd International Scientific Conference of Al-Ayen University
4 November 2021
Al-Ayen University

Acceptance Letter

Date: 19-08-2021

Dears Mukalad Faleh Hassan, Mehdi Ebady Manaa

Warm Greetings,

Based on the recommendations of the reviewers and the Scientific Committee, your paper entitled **" Big Data Processing with Hadoop and Data Mining** "has been accepted for participation in the 3rd International Scientific Conference of Al-Ayen University. After the completion of all conference procedures, the accepted papers will be published in Scopus index journal (Lecture Notes in Networks and Systems-Springer).

Regards,

Shafik Sh. Shafik

The supervisor of ISCAU conference

Chancellor of Al-Ayen University

Chapter One

General Introduction

1.1 Introduction

Big data is a large and complex amount of data that cannot be using traditional analysis methods. This data can be in structured, semi-structured, and unstructured forms. The massive flow of data has led to the need for a better analytical methods, as traditional methods have become inefficient for processing big data [1]. Therefore, there are frameworks display to analysis, store, and process the large amount of data such as Apache Hadoop and Apache Spark [2].

The general characteristics of the big data are that the size of data is too big, the generation of data is too fast and most of the times the data is not directly in the form suitable for the database systems. Big data generated comes from three primary sources: Social Data (such as Likes, Tweets & Retweets, Comments, Video Uploads, and general media that are uploaded and shared via the world's favorite social media platforms.), Machine Data (such as industrial equipment, IoT sensors, web logs which track user behavior), Transactional Data (such as the daily transactions that take place both online and offline such as Invoices, payment orders, storage records, delivery receipts) [3].

Big data typically refer to the following three types based on data sources from physical, cyber, IoT sensors, and social worlds, as shown in Figure (1.1) [4], [5]:

Nature data: we can imagine that data coming from the nature in our earth will be a great potential data source, such as satellite data from outer space.

Life data: it is a big an enormous project on the study of biological body, especially the exploration of the human body still have a lot of challenges, such as biological data.

Sociality data*:* with the fast development of digital mobile products and network, large volumes of sociality data are generating every day in our life, such as voice and video data.

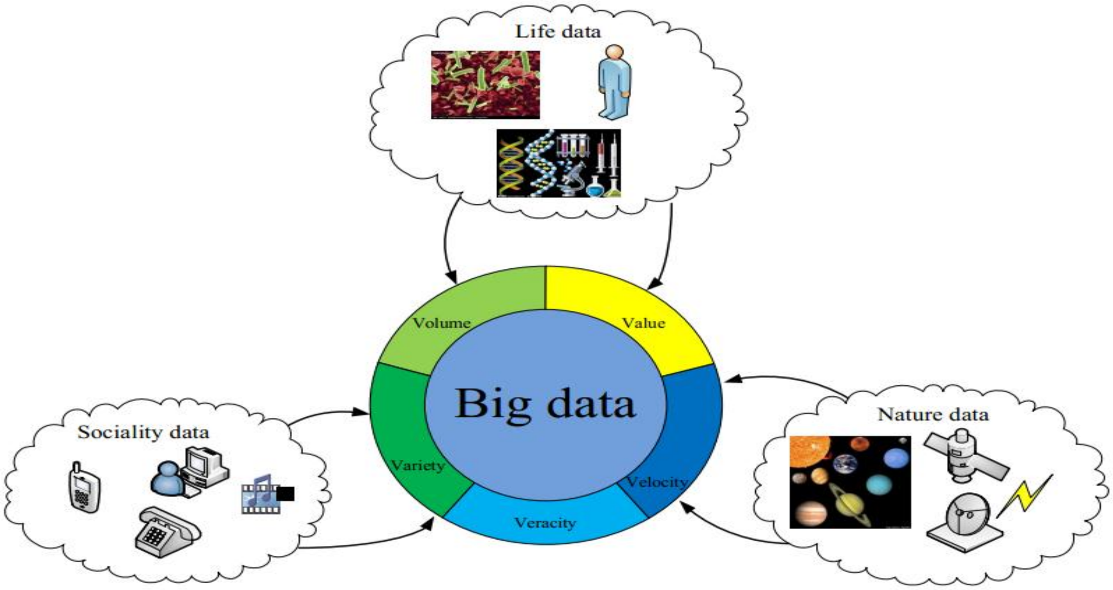


Figure 1.1: Big Data Types and Characteristics[5].

Data pre-processing refers to the set of techniques used to transform raw data from the original data source. In this scenario data is converted an a useable form, free from errors that might introduce inaccuracies in the existing system. The instance and feature selection are implemented in this method [6].

As the speed of data volume increases exponentially, there is a need for efficient distributed big data system that can store the massive volume of data. Currently, the most well-known Big data systems are:

Hadoop, MongoDB, Amazon Web Services, Google BigQuery, Microsoft Azure, IBM Big data and many other. Apache Hadoop is a framework that perform distributed processing of massive datasets across clusters of computers that scale up from a single server to thousands. Apache Hadoop is an open source framework for reliable, scalable and distributed computing over a massive amount of data developed in Java and consist of four main subprojects: MapReduce, Hadoop Distributed File System (HDFS), YARN, and common Hadoop utilities like Hbase, Zookeeper, Avro and some other [7].

The power of the Hadoop is in the parallel access to data that can reside on a single node or on thousands of nodes. When the data is loaded into the system it is split into chunks of data. It stores and process the huge voluminous amount of data with their strong Hadoop ecosystem. The importance of Hadoop efficiency is yielded from the big data strategy of parallel reading of large data files that are stored in internode network in a cluster especial Healthcare big data which it refers to the large volumes and complex of electronic data sets which are difficult to manage by using traditional software and hardware systems, modern healthcare systems need to handle large volumes of batch data which successfully manage by Hadoop [8].

Machine learning techniques have been widely adopted in a several of massive and complex data-intensive fields such as medicine, astronomy, biology, and so on, for these techniques provide possible solutions to mine the information hidden in the data [9]. It is a highly interdisciplinary field building upon ideas from many different kinds of fields such as artificial intelligence, optimization theory, information theory, statistics, cognitive science, optimal control, and many other disciplines of science, engineering, and mathematics[10].

1.2 Related Work

In this section, the most related works in term of big data processing using data mining, Hadoop and Machine Learning have been discussed and overviewed as follows:

The study to predict stroke risk proposed by (Li et al., 2019). They employed Naive Bayes, Decision Tree, and Neural Network to analyze data to predict stroke. In their study, they used accuracy and AUC as their pointer's assessment. All of this algorithm, they classified decision tree and naive Bayes gave the most accurate. The results showed the accuracy of DT as 82.22 % and NB(84.24 %) as the most accurate [11].

A different distributed machine learning algorithms for stroke prediction on the Healthcare Dataset Stroke have been compared in (Ali et al., 2019). The work implemented by a big data platform that is Apache Spark. Apache Spark is one of the most popular big data platforms that handle big data and includes a MLlib library. The results showed that Random Forest Classifier has achieved the best accuracy at 90 % [12]. While the proposed system is used Apache Hadoop/Mapreduce approach.

Deep Neural Network has been used in (Cheon et al., 2019) to detect stroke using medical service use and health behavior data, besides identified 15,099 patients with stroke. Alongside, they compared the used method (a scaled PCA/deep neural network (DNN) approach) to five other machine-learning methods. The area under the curve (AUC) value method was 83.48% [13].While the proposed system is used Big healthcare dataset with 43400 records.

Stroke prediction through deep learning is proposed by (Karthik et al., 2020). The knowledge of medical domain problems could not be traced accurately by the traditional predictive models. The outcome of the study was more accurate than a scoring system in the medical domain in the prediction of stroke [14].

Classification of stroke through machine learning techniques is discussed by (Stančin et al., 2020), and they have reviewed many works with the perspective of classification. Their work discussed two algorithmic approaches, decision tree and k -nearest neighbor (KNN). It concluded that decision tree performed better than KNN algorithm[15].

Artificial Neural Networks (ANN) for the prediction of Thromboembolic stroke disease has been used in (Alotaibi et al., 2020). The healthcare dataset stroke data with eight important attributes of a patient have been used. This research work shows ANN based prediction of stroke disease by improving the accuracy to 89% with a higher consistent rate. The ANN exhibits the right performance levels for the prediction of stroke disease [16].

Designing statistical assessment healthcare information system for diabetics analysis using big data has been proposed in (Sivaparthipan et al., 2020) by using a model of a statistical assessment, healthcare information system for Diabetes Analysis employing big data. The performance metric such as accuracy and F-measure for the proposed statistical assessment model is evaluated by Hadoop framework, ANN, and KNN the results are comparatively higher than existing methods [17].

In the same context, the performance of KNN, Multiple Linear Regression (MLR), and a Regression Tree model to predict the stroke severity have been compared in (Uyanık et al., 2020); the results showed KNN has better accuracy than other models[18].

Support Vector Machine (SVM), Stochastic Gradient Boosting (SGB), and penalized logistic regression (PLR) have been used in (Sampurnima et al., 2020) to predict stroke for the collected dataset from TurgutOzal Medical Centre, Inonu University, Malatya, Turkey. The findings of the research proved that SVM achieved the highest accuracy of 98% [19].

An improved artificial neural network, SVM and Hadoop have been presented in (Liu et al., 2020) with enabling the high-performance classification for the imbalanced large volume data. The parallelization is based on the data separation, and the parallelization is implemented using the Hadoop framework. To overcome the classification accuracy loss issue caused by the separation, the weighted voting is presented to improve the classification accuracy. The experimental results show the effectiveness of the presented classification algorithm [20].

A machine learning models for predicting stroke has been developed in (Ali et al., 2020) with imbalanced data in an elderly population in China. Machine learning methods such as regularized logistic regression (RLR), support vector machine (SVM), and random forest (RF) were used to predicting stroke with demographic, lifestyle, and clinical variables. Accuracy, sensitivity, specificity, and areas under the receiver operating characteristic curves (AUCs) were used for performance comparison [21].

Predicting ischemic stroke proposed in (Tozlu et al., 2020) by using two ANN models on the dataset from Sugam Multispecialty Hospital, Kumbakonam, Tamil Nadu, India. And the researchers concluded that the accuracy rates achieved 79.2% and 95.1% [22].

While classify stroke has been proposed in (Govindarajan et al., 2020) that combines text mining tools and machine learning algorithms. The data were fed into various machine learning algorithms such as artificial neural networks, support vector machine, boosting and bagging and random forests. Among these algorithms, artificial neural networks trained with a stochastic gradient descent algorithm outperformed the other algorithms with a higher classification accuracy of 95% and a smaller standard deviation of 14.69 [23].While the proposed system based on the Data mining for data preprocessing and evaluated output testing dataset with machine learning classifiers.

Early prediction of stroke diseases has been proposed in (Emon et al., 2020) using different machine learning approaches. Using these high features attributes, ten different classifiers have been trained, namely: Logistics Regression, Stochastic Gradient Descent, Decision Tree Classifier, AdaBoost Classifier, Gaussian Classifier, Quadratic Discriminant Analysis, Multilayer Perceptron Classifier, K-Neighbors Classifier, Gradient Boosting Classifier, XGBoost Classifier for predicting the stroke. The results show weighted voting is almost the perfect classifier for predicting the stroke that can be used by physicians and patients to prescribe and early detect a potential stroke [24].While the proposed system is based on the

Hadoop/Weight approach for high accurate stroke detection and early predication.

Prediction of new prescription requirements for diabetes patients using big data technologies has been suggested by (Bakırarar et al., 2021) using the data mining technologies of random forest and multilayer perception with the help of big data technologies, prediction of new prescription was made on these data. Accuracy of the random forest and multilayer perceptron methods using the Mahout technology were found to be 0.879 and 0.849 respectively whereas Accuracy of the random forest and multilayer perceptron methods using the Scala technology were found to be 0.849 and 0.870 [25].

Analyzing the performance of stroke prediction using machine learning has been proposed in (Gangavarapu et al., 2021), they have taken various physiological factors and used machine learning algorithms like Logistic Regression, Decision Tree Classification, Random Forest Classification, K-Nearest Neighbors, Support Vector Machine and Naïve Bayes Classification to train five different models for accurate prediction. The algorithm that best performed this task is Naïve Bayes that gave an accuracy of approximately 82% [26].

The best Machine Learning algorithm for identification of heart diseases has been proposed in (Harish et al., 2021). The proposed work compares the precision of three well-known classification algorithms, Decision Tree and Naïve Bayes, Random Forest for the prediction of heart disease. The result indicates that the Random Forest algorithm is the most efficient algorithm for prediction of heart disease with accuracy score of 97.17% [27]. While the proposed system is based on Stroke disease.

Table 1.1 illustrates the aims of previous researchers and the main methods which are used to achieved the works.

Table 1.1: The literature survey summary

Ref, Year	Methods	Aims	Results
[11], 2019	NB, DT, and NN	Analyze data to predict stroke	DT (82.22 %) and NB(84.24 %) the most accurate
[12], 2019	Apache Spark with data mining and big data, RF	Stroke prediction on the Healthcare Dataset Stroke	RF best accuracy at 90 %.
[13], 2019	Deep Neural Network	Detect stroke using medical service use and health behavior data	Accuracy 83% with Area Under the Curve (AUC) was 83.48%.
[14], 2020	Deep Learning	Stroke prediction through deep learning	High accurate (97%)
[15], 2020	DT and KNN	Classification of stroke through machine learning techniques	DT(97.8%) better than KNN (97%)
[16],2020	ANN	ANN for the prediction of Thromboembolic stroke disease	Improving the accuracy to 89% with a higher consistent rate
[17],2020	Hadoop framework, with machine learning as ANN, KNN,	Designing statistical assessment healthcare information system for diabetics analysis using big data	ANN with high accuracy as 95.5% and KNN as 93.8%
[18],2020	MLR and (RT)	Predict the stroke severity have been compared	KNN has better accuracy than other models as 95%

[19],2019	SVM, SGB and PLR	Predict stroke for the collected dataset	SVM achieved the highest accuracy of 98%.
[20],2020	Hadoop framework, SVM , back propagation neural network (BPNN)	High-performance large-scale data classification	BPNN and SVM for Testing dataset accuracy about 96.667%
[21], 2020	RLR, SVM, RF	Predicting stroke with imbalanced data in an elderly population in China.	Sensitivity and AUC reached 0.78 (95% CI, 0.73–0.83) for RF and 0.72 (95% CI, 0.71–0.73) for RLR
[22],2020	Two ANN Models	Predicting ischemic stroke	ANN achieved 79.2% and 95.1% .
[23], 2020	ANN, SVM, boosting and bagging and RF.	Classify stroke with combines text mining tools and machine learning algorithms.	ANN and SVM higher classification accuracy of 95%
[24], 2020	LR, SGD, DT, AdaBoost, Gaussian, QuadraticDiscriminant Analysis, MLP, KNN, Gradient Boosting , XGBoost	Early prediction of stroke diseases	Weighted voting is almost the perfect classifier with (97 %)
[25],2021	RF, NN(MLP)	Prediction of new prescription requirements for diabetes patients	Accuracy of RF as 0.879 and NN as 0.849
[26], 2021	SVM, LR,DT, RF, KNN and NB	Analyzing the Performance of Stroke Prediction	NB performs best with an accuracy of 82%.
[27], 2021	DT and NB, RF	Identification and prediction of heart diseases	RF is the most efficient algorithm of 97.17%.

1.3 Problem Definition

- Healthcare systems are being digitally transformed by technological enhancements in medical information systems, electronic medical records, wearable and smart devices, and handheld devices. This increase in medical big data, alongside the development of computational techniques in healthcare, has enabled a big problem faced to predicted diseases [28].

- Stroke diseases are one of the most challenging problems faced by the Health Care system all over the world. Knowing and treatment these diseases are very important nowadays. With the expanding count of deaths because of heart illnesses and Stroke, it is the necessary to build up a system to foresee heart ailments precisely [29].
- Besides, one of the largest problems of big data is an incapacity to process huge quantity of knowledge in a typical time. So that, there is a necessity to discover effective methods to work with fields like storing of data, real-time processing, information extraction and abstract model generation [30]. To address these problems :
- Hadoop is one of the keys solution for big data analysis especially in our case of big healthcare data, it is effected on accuracy in the main case. We need a specific approach to make balance between accuracy and time , and to deal with this case and we propose Hadoop(Count/Weight) to reduce building time and increase accuracy at the same time.

1.4 Thesis Contribution

The major contributions are represented by :

- Hadoop/Count to decrease the number of records in big healthcare dataset based on MapReduce approach as Key/Value approach depending on the similarity of attributes values in the dataset.
- Hadoop/Weight to provide attribute weight used as a key weight parameter useful for increase accuracy results within machine learning classifiers.

- Furthermore, Hadoop(Count/Weight) is implemented to decrease the time and computation power of CPU and RAM memory for execution time by providing fast and cost-effective solutions for big dataset.

1.5 Aims of the Study

The main aims of the study are :

- To design data mining, Hadoop and machine learning classifiers are suggested to be as a solution. MapReduce has a huge capacity to handle a large quantity of any type of data. MapReduce is common in healthcare field because its effective analytical way in health data, it provides access to treatment to users, discovers the causes of diseases (stroke) and knows the proper management.
- To implement Weight-count attributes to enhance accuracy results, with decrease number of records in dataset and decrease processing time by using 1 master and 2 slaves networks.
- To find an early predict stroke occurrence based on different attributes and machine learning in the proposed dataset which it helps to predict stroke before it happens and decease side effects.

1.6 Thesis Outline

Furthermore, this thesis contains four chapters in addition to chapter one:

Chapter Two: It presents the big data preprocessing, big data characteristics Furthermore, data mining techniques as well as the hadoop features and MapReduce job, the used machine learning algorithms as well as the used evaluation parameters and the characteristics of the big healthcare dataset.

Chapter Three: It presents the proposed system and illustrates the practical stages of the system as the data mining preprocessing, hadoop count/weight approach and machine learning, in addition to the system installation requirements and explains the proposed machine algorithms system.

Chapter Four: It describes the results and evaluates the used system based on three case studies as Data Mining /Machine Learning result, Hadoop/Count and Hadoop/Weight results.

Chapter Five: It presents the results conclusion. Also, it described the future works suggestions.

Chapter Two

Theoretical Background

2.1. Introduction

Healthcare acquired its current influence regarding big data technology due to the fact that the data sources involved in healthcare are well-known for their volume, heterogeneous complexity, and high dynamism. In the context of big data, the success of healthcare applications depends solely on the underlying architecture and utilization of appropriate tools. Big data technology has many areas of application in healthcare, such as predictive modeling and clinical decision support disease or safety surveillance, public health, and research. Big data analytics in medicine and healthcare covers integration and analysis of large amounts of complex heterogeneous biomedical data and electronic health records data. There are different tools for analysis big healthcare data, Hadoop is one of the most used tool in this side, as it is an efficient and rapid data processing platform for the healthcare system [31].

2.2 Big data

Big data is a collection of data sets or a combination of data sets. The concept of big data has been endemic within digital communication and information science since the earliest days of computing. Big data is growing day by day because data is created by everyone and for everything from mobile devices, call centers, web servers, and social networking sites, etc [32]. But the challenge is that it is too large, too fast and hard to handle for traditional database and existing technologies. Many organizations gather the massive amounts of data generated from high-volume transactions like call centers, sensors, web logs, and digital images. The success of their business depends on meeting big data challenges while continually improving operational efficiency[33].

Big data are continuously including more and more data sets with high volume beyond the capability of regularly used software tools to capture, curate, handle and process data set within a tolerable elapsed time. A huge amount of data sets is created every second from every part of the world i.e. the volume of data can never be reduce but increases day by day[34].

2.2.1 Big data Characteristics

Big data has got numerous definitions from researchers, organizations, and individuals. In 2001, industry analyst Doung Laney (currently with Gartener), articulated the mainstream of definition of big data regarding in terms of three V's; Volume, Velocity, and Variety [3]. SAS (Statistical Analysis System) has added two additional dimensions i.e. Variability and complexity [34].

Further, Oracle has defined big data in terms of four V's i.e. Volume, Velocity, Variety and Value [35]. Furthermore, Oguntimilehin A, presented big data in terms of five V's Volume, Velocity, Variety, Variability, Value and a Complexity[36] .All the characteristics has been listed and defined in Table (2.1). These characteristics provide research horizon to the researcher and practitioners in order to effectively manage big data.

Table 2.1: Big Data Characteristics[36].

S. No.	Big Data Characteristics	Concept illustration	Description
1	Volume	Size of Data	Quantity of collected and stored data. Data size is in TB, PB.
2	Velocity	Speed of Data	The transfer rate of data between source and destination
3	Value	Importance of Data	It simply represents the business value to be

			derived from big data.
4	Variety	Type of Data	Different type of data like pictures, videos, audio etc. arrives at the receiving end.
5	Veracity	Data Quality	Accurate analysis of captured data is virtually worthless if it’s not accurate.
6	Validity	Data Authenticity	Correctness or accuracy of data used to extract result in the form of information.
7	Volatility	Duration of Usefulness	Big data volatility means the stored data and how long is useful to the user.
8	Visualization	Data Process/ Data act	It is a process of representing abstract.
9	Virality	Spread Speed	It is defined as the rate at which the data is broadcast/spread by a user and received by different users for their use.
10	Viscosity	Lag of Event	It is a time difference the event occurred and the event being described.
11	Variability	Data Differentiation	Data arrives constantly from different sources and how efficiently it differentiates between noisy data or important data.
12	Venue	Different Platform	Various types of data arrived from different sources via different platforms like personnel system, private & public cloud etc.
13	Vocabulary	Data Terminology	Data terminology likes data model, data structures etc.
14	Vagueness	Indistinctness of existence in a Data	Vagueness concerns the reality in information that suggested little or no thought about what each might convey.
15	Complexity	Correlation of Data	Data comes from different sources and it is necessary to figure out the changes whether small or large in data with respect to the previously arrived data so that information can get quickly.

2.2.2 Big Data Analytics

Big data refers to data that exceeds the typical storage, processing, and computing capacity of conventional databases and data analysis techniques. As a resource, big data requires tools and methods that can be applied to analyze and extract patterns from large-scale data [37].

Big Data Analytics refers to collecting, organizing, analyzing large data sets to discover different patterns and other useful information. Big data analytics is a set of technologies and techniques that require new forms of integration to disclose large hidden values from large datasets that are different from the usual ones, more complex, and of a large enormous scale. It mainly focuses on solving new problems or old problems in better and effective ways[38].Types of Big Data analytics explained as :

A. Descriptive Analytics

It consists of asking the question:What is happening? It is a preliminary stage of data processing that creates a set of historical data. Data mining methods organize data and help uncover patterns that offer insight.[38].

B. Diagnostic Analytics

It consists of asking the question: Why did it happen? Diagnostic analytics looks for the root cause of a problem. It is used to determine why something happened. This type attempts to find and understand the causes of events and behaviors[39].

C. Predictive Analytics

It consists of asking the question: What is likely to happen? It uses past data in order to predict the future. It is all about forecasting. Predictive analytics uses many techniques like data mining and artificial intelligence to analyze current data and make scenarios of what might happen[39]. In the proposed system, it serves to predict the type of stroke disease beforehand hence the treatment actions can be carried out early and more appropriate to avoid worsening the patients' condition status.

D. Prescriptive Analytics

It consists of asking the question: What should be done? It is dedicated to finding the right action to be taken. Descriptive analytics provide a historical data, and predictive analytics help forecast what might happen. Prescriptive analytics uses these parameters to find the best solution[39].

2.2.3 Challenges to Handle Big Data

The programmers have to take decisions due to large availability of raw and complex data. An organization can collect, store, and analyze these large datasets in a number of ways. The Business can even use robust big data tools to store, access, and manage the structured and unstructured data collected from various sources in a faster and more efficient way. There are few challenges to address when handling big chunks of data. Some challenges listed below[40]:

A. Handling a Large Amount of Data

The large availability of data makes the difficulty is making decisions. For example, the large amount of information streaming in from our phones, computers, networks , IoT sensors for parking meters, buses, trains, and planes is truly it[41].

This data exceeds the amount of data that can be stored and computed, as well as retrieved. The challenge is not so much the availability, but the management of this data. Along with rise in unstructured data, the availability of data is in multiple formats such as video, audio, social media, smart device data etc. Some of the newest ways developed to manage this data are a hybrid of relational databases combined with NoSQL databases[41].

B. Data Complexity

With the huge updating of data in every second, organizations need to be aware of handling it too. For example, if a retail company wants to analyze customer behavior, real-time data from their current purchases can help. There are Data Analysis tools available for the same – Veracity and Velocity[42].

C. Shortage of Skilled Resources

There is a shortage of skilled Big Data professionals available at this time. This has become mentioned by many enterprises seeking to better utilize Big Data and build more effective Data Analysis systems[42].

2.3 Data Mining

The goal of this technique is to find patterns that were previously unknown. Once these patterns are found they can further be used to make certain decisions for development of their businesses. Vast amounts of raw data is surrounding us in our world, data that cannot be directly treated by humans or manual applications[43].

The performance and quality of the knowledge extracted by a data mining method in any framework does not only depends on the design and performance of the method but is also very dependent on the quality and suitability of such data[44]. The Knowledge Discovery in Databases process showed in Figure (2.1).

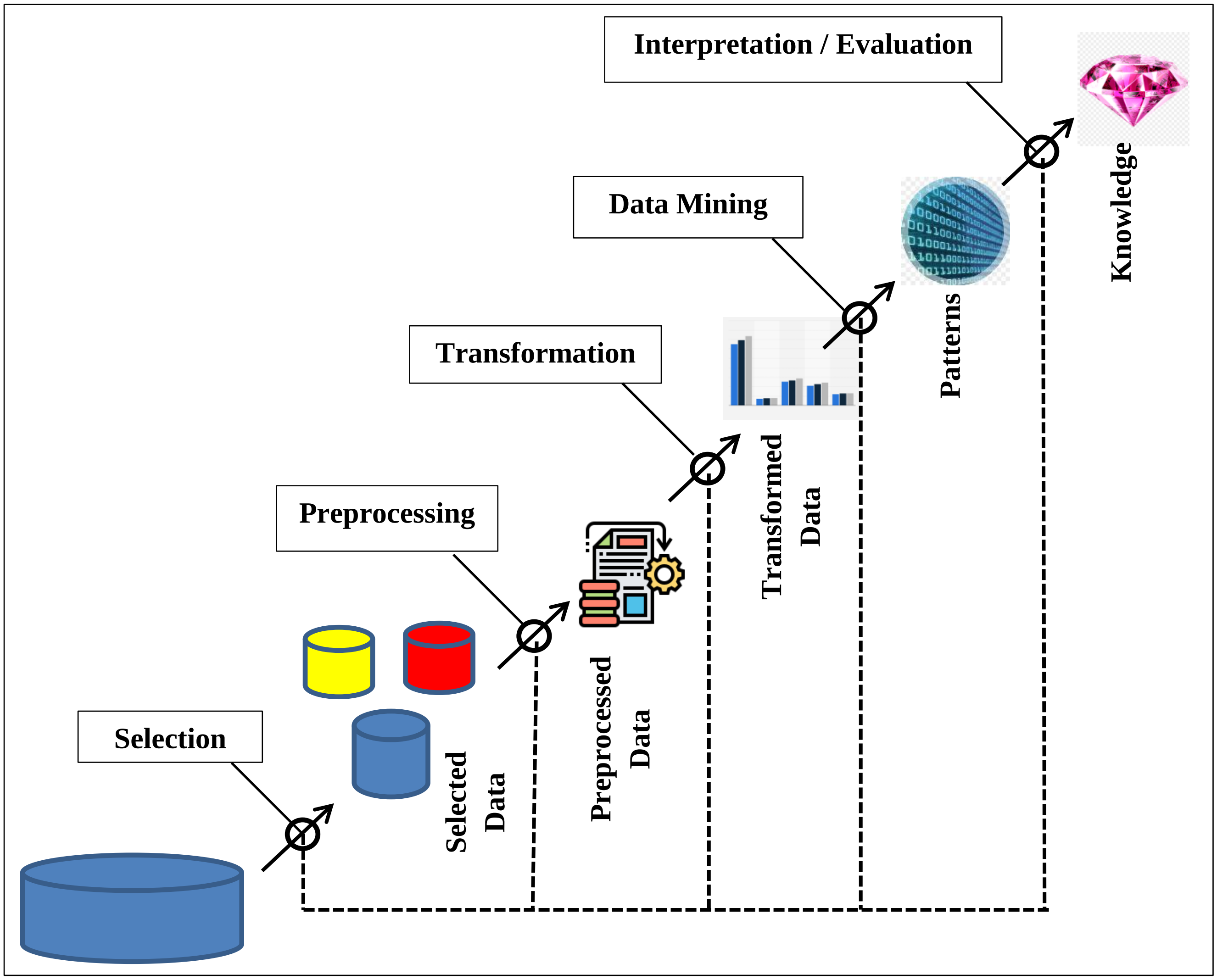


Figure 2.1 : The Knowledge Discovery in Databases (KDD) Process.

2.3.1 Data preprocessing

The set of techniques used prior to the application of a data mining method is named as data preprocessing for data mining and it is known to be one of the most meaningful issues within the famous Knowledge Discovery from Data process [45]. The knowledge discovery process usually involves seven phases from start to finish.

- Phase 1: Data Integration : Collect data from sources
- Phase 2 : Data Selection : Select useful data
- Phase 3 : Data Cleaning : Rid data of errors, missing values, inconsistent data
- Phase 4 : Data Transformation : Normalization, smoothing, other forms appropriate for data mining
- Phase 5 : Data Mining : Apply mining techniques to discover patterns
- Phase 6 : Pattern Evaluation / Presentation : Visualization and removing redundant patterns.
- Phase 7 : Knowledge Discovery : Use to make decisions

The preprocessing tasks showed in Figure (2.2)

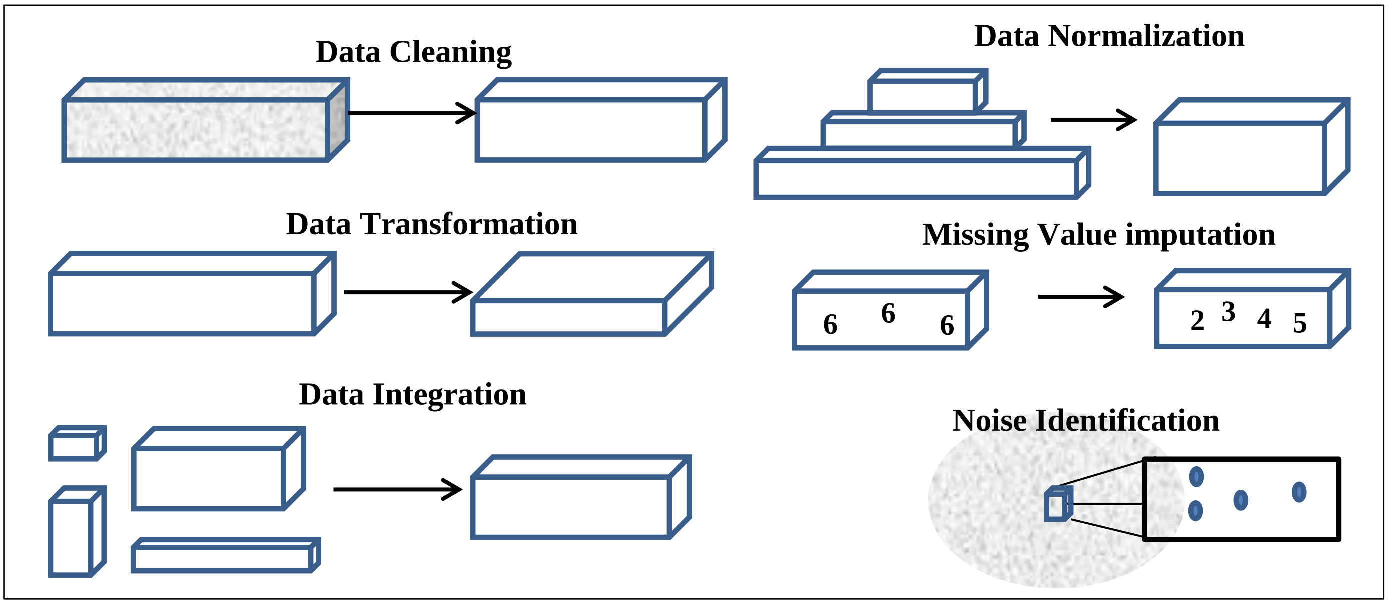


Figure 2.2.: Data Preprocessing Tasks

After the application of a successful data preprocessing stage, the final data set obtained can be regarded as a reliable and suitable source for any data mining algorithm applied afterwards.

Data preprocessing is not only limited to classical data mining tasks, as classification or regression. More and more researchers in novel data mining fields are paying increasingly attention to data preprocessing as a tool to improve their models. This wider adoption of data preprocessing techniques is resulting in adaptations of known models for related frameworks, or completely novel proposals[46].

A. Imperfect Data

Most techniques in data mining rely on a data set that is supposedly complete or noise-free. However, real-world data is far from being clean or complete. In data preprocessing it is common to employ techniques to either removing the noisy data or to impute (fill in) the missing data. The following two sections are devoted two missing values imputation [47].

B. Missing Values

One big assumption made by data mining techniques is that the data set is complete. The presence of missing values is, however, very common in the acquisition processes. A missing value is a datum that has not been stored or gathered due to a faulty sampling process, cost restrictions or limitations in the acquisition process[48].

C. Feature Indexers and Encoders

These functions convert features from one type to another using indexing or encoding techniques.

- **StringIndexer**: converts a column of string into a column of numerical indices. The indices are ordered by label frequencies.
- **OneHotEncoder**: maps a column of strings to a column of unique binary vectors. This encoding allows better representation of categorical features since it removes the numerical order imposed by the previous method.
- **VectorIndexer**: automatically decides which features are categorical and transform them to category indices[49].

D. Other Pre-processing Methods for Text Mining

Text mining techniques try to structure the input text, yielding structured patterns of information.

TF-IDF: This tool is aimed at quantifying how relevant each term is to a document, given a complete set of documents. Term Frequency (TF) measures the number of times that a term appears in a documents, whereas Inverse Document Frequency (IDF) measures how much information is given by a term according to its document frequency. TF is implemented using feature hashing for a better performance, so that each raw feature is mapped into an index[50].

2.4 Apache Hadoop

Big Data are collections of information that would have been considered gigantic, impossible to store and process, a decade ago. The processing of such large quantities of data imposes particular methods. A classic database management system is unable to process as much information. Hadoop is an open source software product (or, more accurately, software library framework) that is collaboratively produced and freely distributed by the Apache Foundation effectively Hadoop is a distributed data processing and management system. It contains many components, including: HDFS, YARN, Map Reduce. HDFS is a distributed file system that provides high-performance access to data across Hadoop clusters [52].

Hadoop is a very interesting application that crosses lines between network, storage, and application - but this is one of the key reasons the Map Reduce function is incredibly efficient. Figure (2.3) Network topology of the Hadoop used for testing, including IP addresses, hostnames, nodes and dataset blocks.

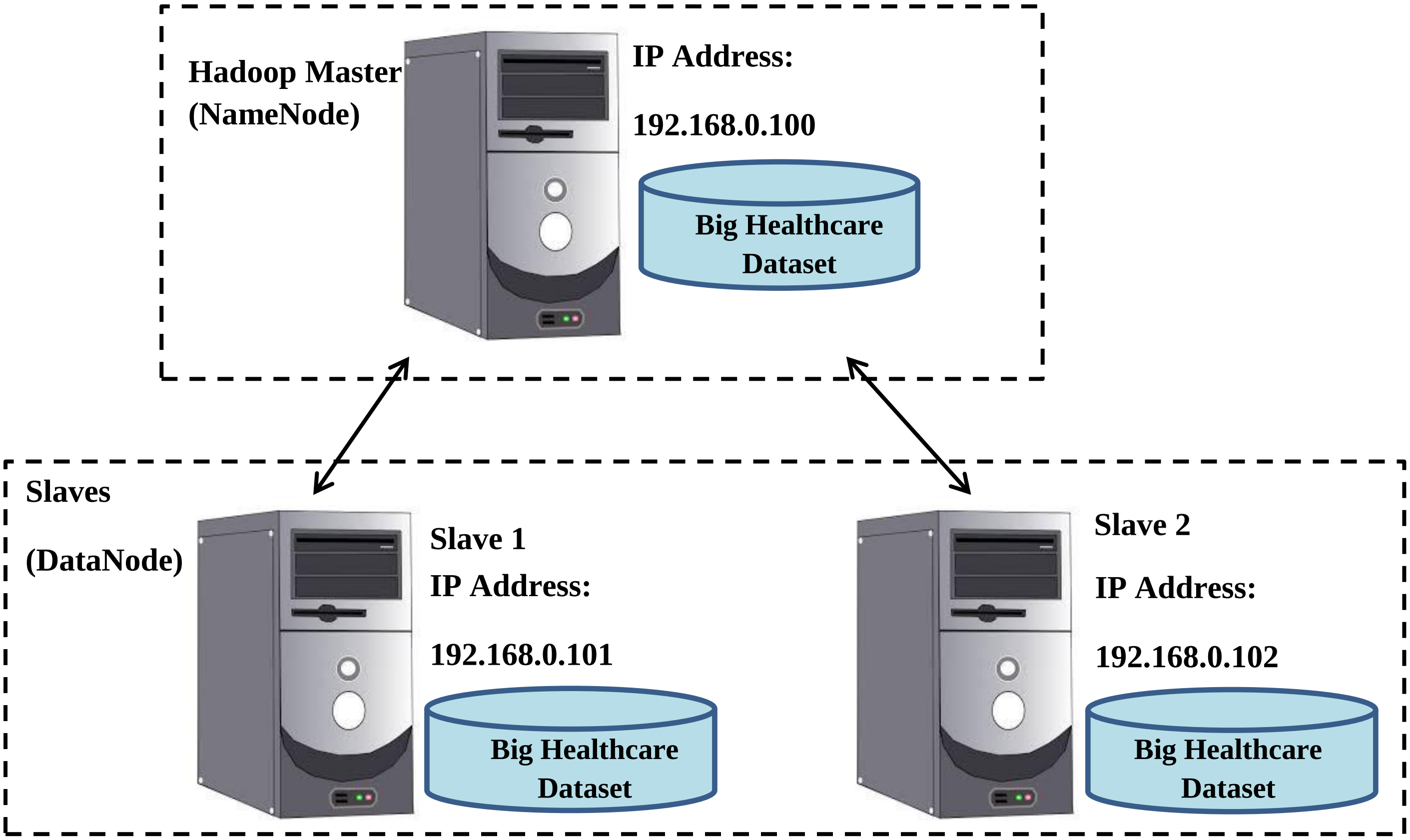


Figure 2.3: Hadoop Network Topology of the proposed System.

MapReduce is a core component of the Apache Hadoop software framework. Hadoop enables resilient, distributed processing of massive unstructured data sets across commodity computer clusters, in which each node of the cluster includes its own storage. MapReduce serves two essential functions: It parcels out work to various nodes within the cluster or map, and it organizes and reduces the results from each node into a cohesive answer to a query[53]. Hadoop relies on two servers:

A. JobTracker

There is only one JobTracker per Hadoop cluster. It receives Map/Reduce tasks to run and organizes their execution on the cluster. When you submit your code to be executed on the Hadoop cluster, it is the JobTracker‟s responsibility to build an execution plan, as it shown in Figure(2.4) [53]

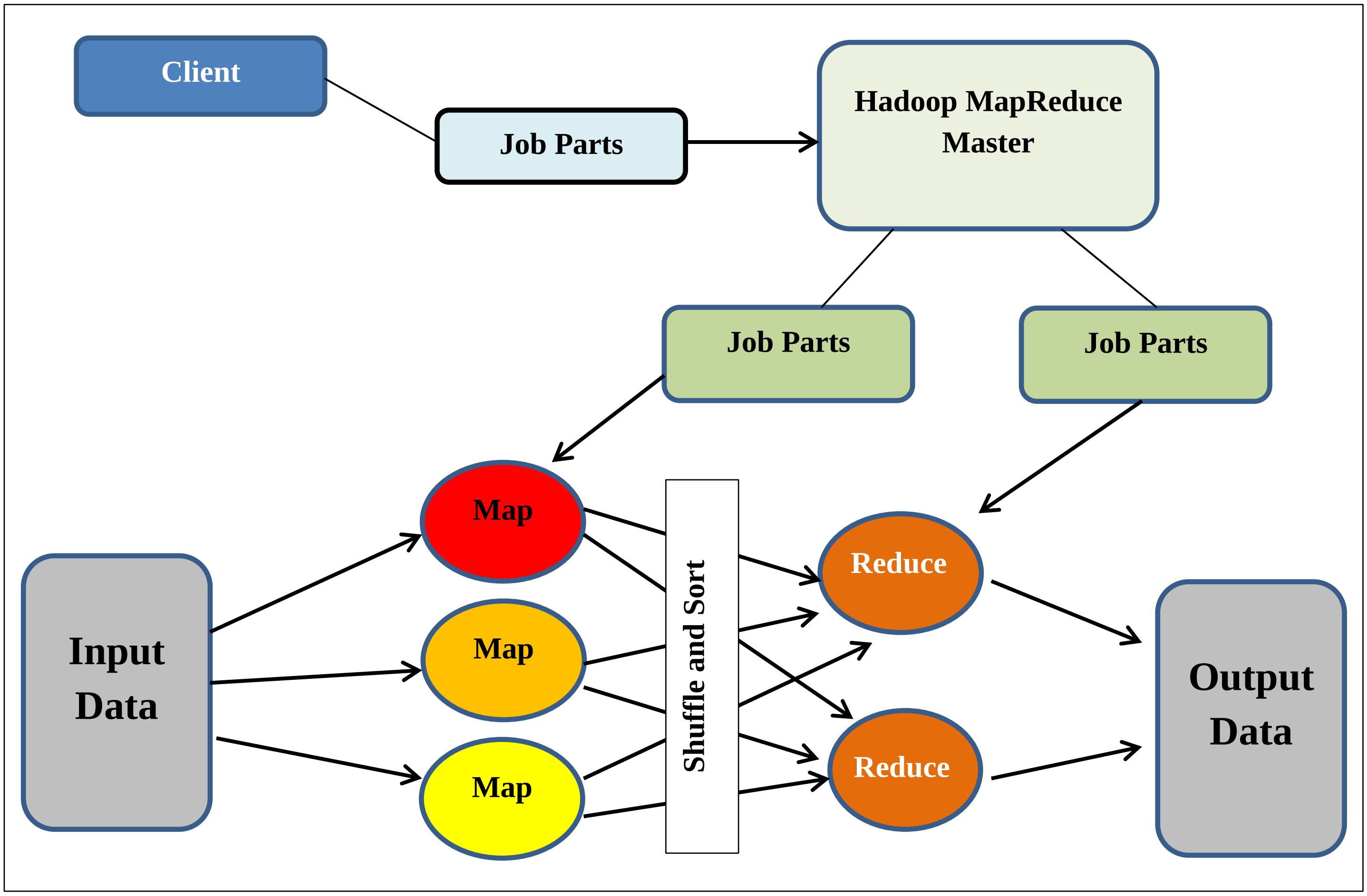


Figure 2.4: A Client Submitting a Job to MapReduce.

B. Task Tracker

Several per cluster. Executes the Map/Reduce work itself (as a Map and Reduce task with the associated input data). The JobTracker server is in communication with HDFS; it knows where the Map/Reduce program input data is and where the output data must be stored.

HDFS relies on two servers:

- NameNode: unique on the cluster. It stores information about file names and their characteristics. It is the master of the HDFS that controls slave DataNode.
- Secondary NameNode: The Secondary NameNode monitors the state of the HDFS cluster and takes “snapshots” of the data contained in the NameNode. If the NameNode fails, then the Secondary NameNode can be used in place of the NameNode[54].
- DataNode: multiple by cluster. Stores the contents of the files themselves, fragmented into blocks, as shown in Figure(2.5)

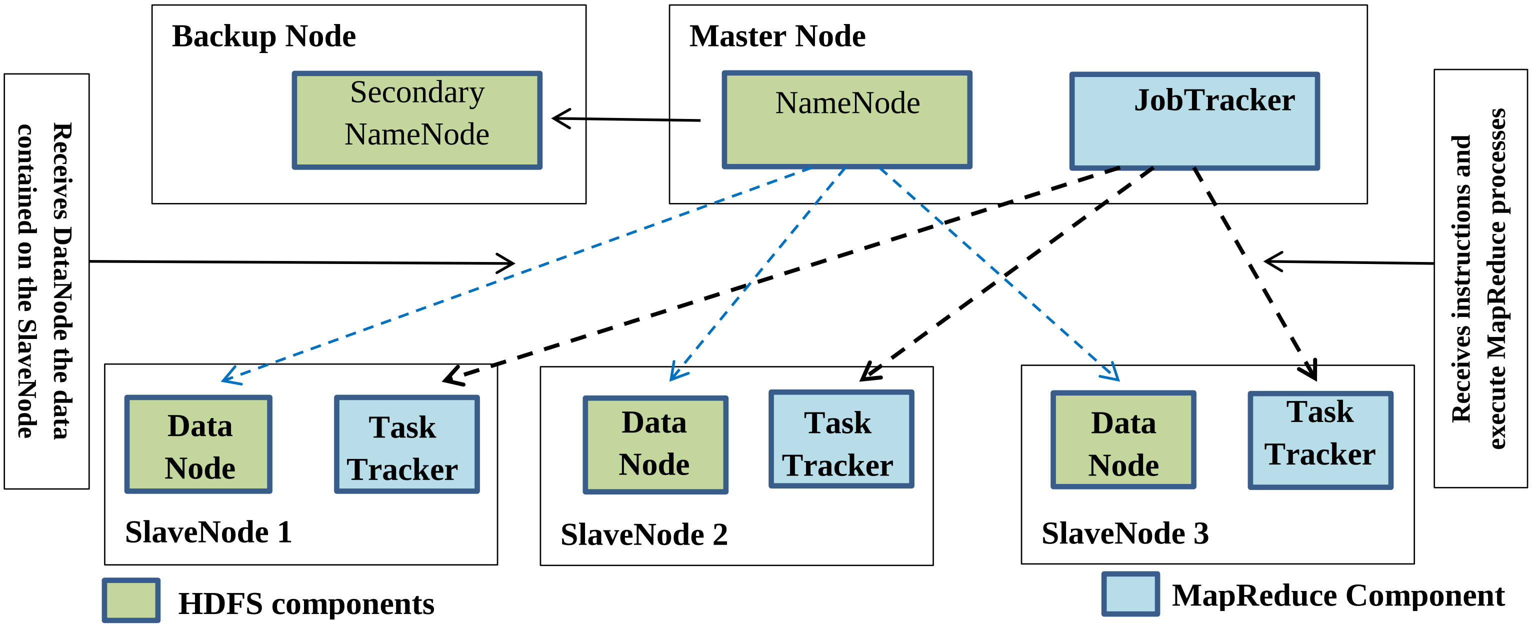


Figure 2.5: Name Node and Data node Architecture of Hadoop.

2.4.1 Basic Components of Hadoop

The main basic component of the hadoop based on the following :

2.4.1.1 The Hadoop Distributed File System (HDFS)

The Hadoop Distributed File System (HDFS) is, as the name already states, a distributed file system that runs on commodity hardware. Like other distributed file systems it provides access to files and directories that are stored over different machines on the network transparently to the user application[55].

HDFS there are two different types of servers: NameNodes and DataNodes. While there is only one NameNode, the number of DataNodes is not restricted. The NameNode serves all metadata operations on the file system like creating, opening, closing or renaming files and directories[55].

Therefore it manages the complete structure of the file system. Internally a file is broken up into one or more data blocks and these data blocks are stored on one or more DataNodes. The knowledge which data blocks form a specific file resides on the NameNode, hence the client receives the list of data blocks from the NameNode and can later on contact the DataNodes directly in order to read or write the data[55].

2.4.1.2 The Basics of MapReduce

MapReduce is a Java environment for writing programs intended for YARN. Java is not the simplest language for this, there are packages to import and class paths to provide. The data exchanged between Map and Reduce, in the entire job are pairs (key, value)[56]:

- Key: it is any type of data: integer, text.

- Value: it is any type of data.

The two functions Map and Reduce receive and send such pairs. Figure (2.6) Shows Basic flow of MapReduce[56].

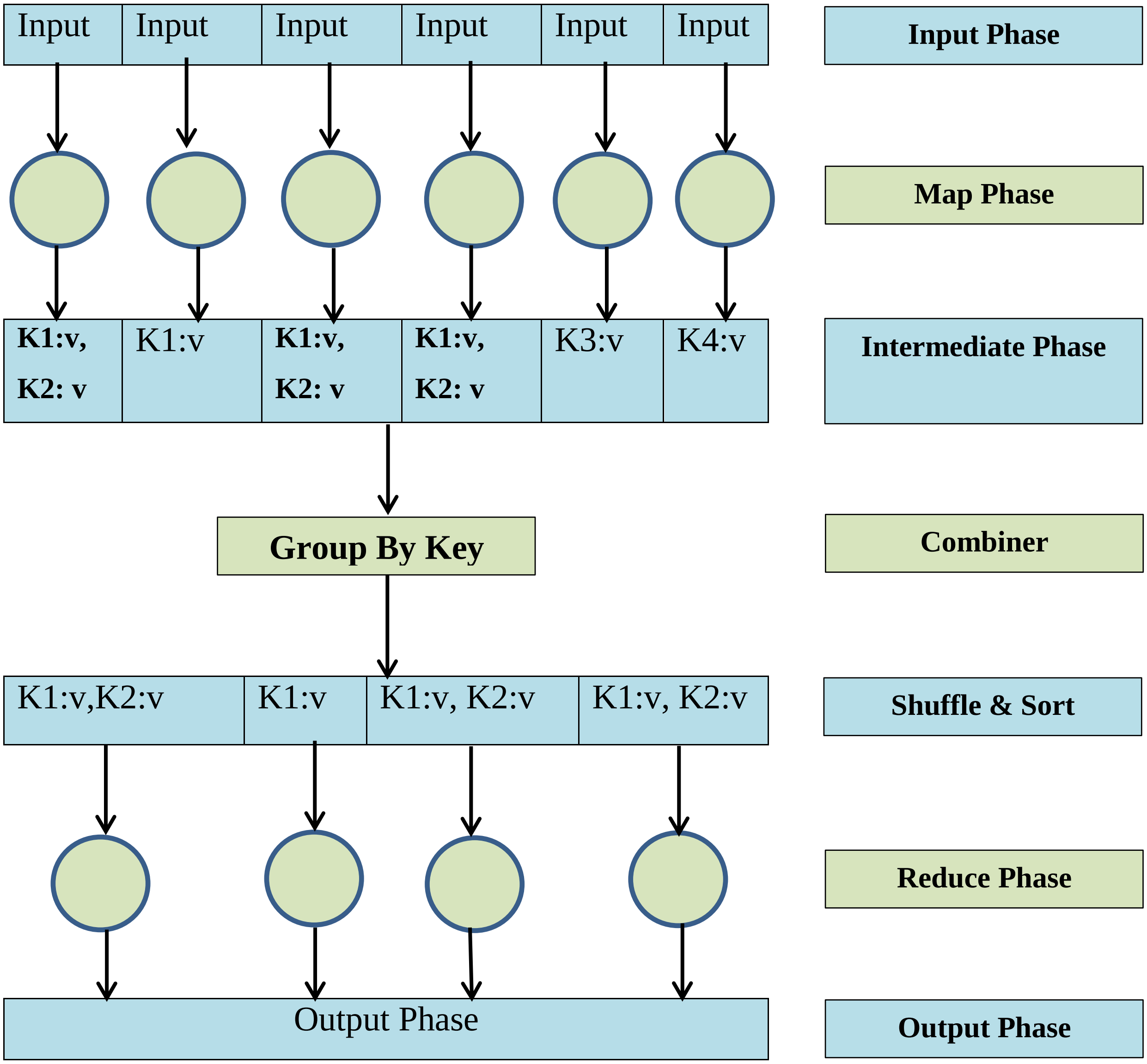


Figure 2.6: Basic flow of MapReduce.

A. Map

The Map function receives an input pair and can produce any number of pairs in output: none, one or more, every node performs the map task to the local data, and writes the output to a temporary storage. A master

Node coordinates that for replicated copies of input data, only one is processed. This first function set of data them converts it to other set of data[57].

B. Shuffle

It is the second step in MR Algorithm. It is named also as (Combine Function). Worker nodes redistribute data depended on the output keys that created via the map task. Such that all data belonged to one key is located on the similar worker node.

C. Reduce

The Reduce function receives a list of input pairs. These are the pairs produced by the instances of Map. Reduce can produce any number of output pairs, but most of the time it is one. Figure (2.7) Shows High-level Hadoop architecture[58].

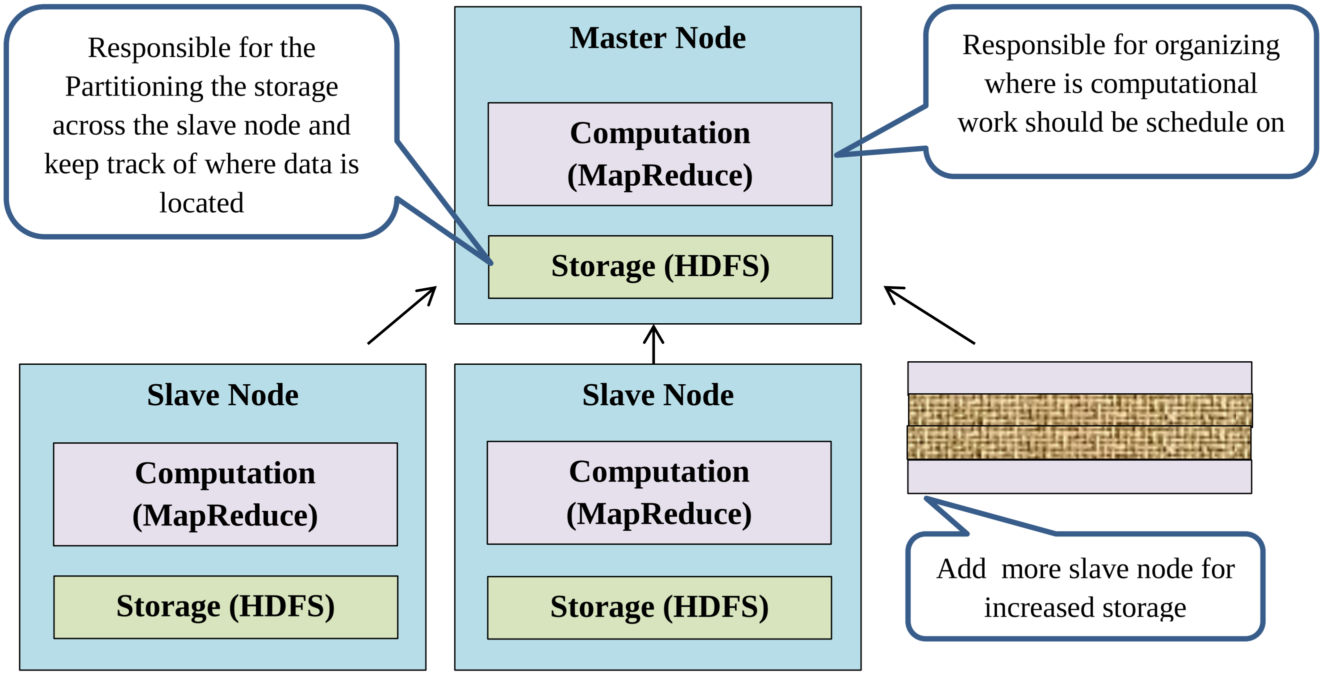

Responsible for the Partitioning the storage across the slave node and keep track of where data is located
Master Node
Computation (MapReduce)
Storage (HDFS)
Responsible for organizing where is computational work should be schedule on
Slave Node
Computation (MapReduce)
Storage (HDFS)
Slave Node
Computation (MapReduce)
Storage (HDFS)
Add more slave node for increased storage

Figure 2.7: High-level Hadoop Architecture [58]

D. Steps for a MapRedcue job

1. Pre-processing of input data, eg: decompression of files
2. Split: Separate data into separately process able blocks and formatted (key, value), eg in rows or tuples
3. Map: application of the map function on all the pairs (key, value) formed from the input data, this produces other pairs (key, value) output .
4. Shuffle & Sort: redistribution of data so that the pairs produced by Map having the same keys are on the same machines.
5. Reduce: Aggregation of pairs with the same key to get the final result[59].

2.4.2 Advantages of Hadoop

Major Advantages of Hadoop are sorted as follow [60] :

A. Scalable

Hadoop is a highly scalable storage platform because it can store and distribute very large data sets across hundreds of inexpensive servers that operate in parallel. Unlike traditional relational database systems (RDBMS) that can't scale to process large amounts of data.

B. Cost-effective

Hadoop also offers a cost-effective storage solution for businesses exploding data sets. The problem with traditional relational database management systems is that it is extremely cost prohibitive to scale to such a degree in order to process such massive volumes of data.

C. Flexible

Hadoop enables businesses to easily access new data sources and tap into different types of data (both structured and unstructured) to generate value from that data.

D. Fast

Hadoop's unique storage method is based on a distributed file system that basically 'maps' data wherever it is located on a cluster. The tools for data processing are often on the same servers where the data is located, resulting in much faster data processing.

E. Resilient to failure

A key advantage of using Hadoop is its fault tolerance. When data is sent to an individual node, that data is also replicated to other nodes in the cluster, which means that in the event of failure, there is another copy available for use[60].

2.5 Taxonomy of Machine Learning Algorithms

The task of choosing a machine learning algorithm includes feature matching of the data to be learned based on existing approaches. Taxonomy of machine learning algorithms is discussed below [61]:

2.5.1 The Supervised Learning/Predictive Models

Supervised learning algorithms are used to construct predictive models. A predictive model predicts missing value using other values present in the dataset. Supervised learning algorithm has a set of input data and also a set of output, and builds a model to make realistic predictions for the response to new dataset. Supervised learning includes Decision Tree, Bayesian Method, Artificial Neural Network, etc.[62].

The proposed system based on these machine learning algorithm shown as follows:

A- Naive Bayes (NB) Algorithm

Naïve Bayes is the most popular method of classification algorithms that used filtering applications and this popularity back to quick training speeds they attain and extremely high accuracy in spite of their relative simplicity to implement. Also it is one of the simplest methods of classification in machine learning. It is depended on Bayes' theory with some independent assumptions between the predictor [63]. The Bayes Theorem equation is [63].

$$P(A|B) = \frac{P(A)*P(B|A)}{P(B)} \qquad (2.1)\ [63]$$

Where

- A& B are events
- P(A) and P(B) are the probabilities of A and B without regard for each other
- P(A|B) is the conditional probability, the probability of A given that B is true
- P(B|A) is the probability of B given that A is true [64]

The Algorithm (2.1) shows the steps of Naive Bayes algorithm[64].

Algorithm (2.1): Naive Bayes (NB) Algorithm
Input: Training dataset T, F= (f1, f2, f3,.., fn) **Output**: A class of testing dataset
Step 1: Read the training dataset T.

Step 2: Calculate the mean and standard deviation of the predictor variables in each class.

Step 3: Repeat Calculate the probability of fi using the gauss density equation in each class;

- Until the probability of all predictor variables (f1, f2, f3,.., fn) has been calculated.

Step 4: Calculate the likelihood for each class.

Step 5: Get the greatest likelihood;

End Algorithm

B- Random Forest (RF) Algorithm

Random Forest is a supervised classifier and an ensemble-learning algorithm that generates many individual learners. It uses a bagging concept to construct a random set of data to build a decision tree. In the standard tree every node is separated using the best split of all the variables, while in a random forest, each node is split using the best of the subset of randomly chosen predicators at that node[65].

The algorithm can handle regression and classification problems. The classification mechanisms are as follows: the random trees classifier gets the input feature vector, classifies it with every tree in the forest, and outputs the class label that have received the majority of “votes”[65].

In Machine Learning, random trees are essentially the combination of two existing methods, single model trees are combined with Random Forest ideas. First, as in Bagging, the training data is sampled with a replacement for every single tree. Secondly, when growing a tree, only a random subset of all attributes is considered at each node rather than always computing the best possible split for each node, and the best split for that subset is calculated, as it showed in Algorithm (2.2)[65].

Algorithm (2.2):Random Forest Algorithm

Input : dataset = pd.read_csv(path, names = headernames)
Output : Subset of features

```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 4].values
dataset.head()
sklearn.ensemble import RandomForestClassifier
    classifier = RandomForestClassifier(n_estimators = 50)
     classifier.fit(X_train, y_train)
  from sklearn.metrics
    import classification_report, confusion_matrix, accuracy_score
    result = confusion_matrix(y_test, y_pred)
        print("Confusion Matrix:")
        print(result)
    result1 = classification_report(y_test, y_pred)
        print("Classification Report:",)
        print (result1)
Output: result2 = accuracy_score(y_test,y_pred)
print("Accuracy:",result2)
```

End Algorithm

C- Decision Tree (DT) Algorithm

It consists of tree internal nodes, which are labeled by the term, branches departing from them labeled by a test measure on the weight, and leaf nodes representing corresponding class labels. A decision tree can classification a text document starting from the root through the query structure until it reaches a certain leaf, which represents the target for document classification. Most of the training data does not suit in the construction of decision tree memory since it is inefficient due to swapping of training tuples [67]. This algorithms can works with nominal, ordinal, interval and ration dataset types. Algorithm (2.3) shows simple example of Decision Tree

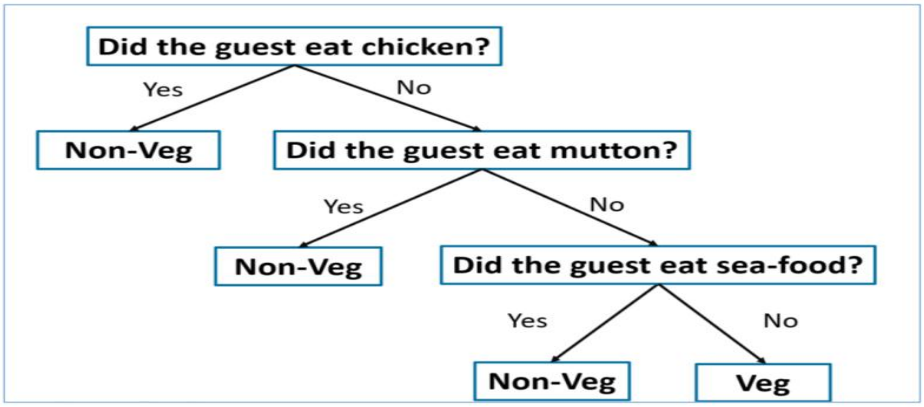

Did the guest eat chicken?
Yes
No
Non-Veg
Did the guest eat mutton?
Yes
No
Non-Veg
Did the guest eat sea-food?
Yes
No
Non-Veg
Veg

Figure 2.8: Decision Tree Example[67].

Algorithm (2.3): Decision Tree algorithm (DT) Algorithm

Input: Training dataset S, Classes X, attribute A. Value v

Output: classifier

```
Calculate Entropy by applying equation (2.1)
    Calculate InformationGain by applying equation (2.2)
    Function BuildDesTree (S, Asplit)
      For i in attributeList
          Compute InformationGain (S,i)
          Append  InformationGain (S,i) to IGLIST
          Amax = attributemax(IGLIST)
 If InformationGain ( S , Amax ) > InformationGain ( S , Asplit )
then
            For all v ∈ to val(Amax)
               Subset = ( x ∈ S | Xmax = v)
    Build (Subset, Amax)
```

End Algorithm

D. Support Vector Machine (SVM) Algorithm

Support Vector Machine (SVM) that can be defined as supervised machine learning method that are more common used for classification problems. The main goal of SVM is determining the ideal dividing hyperplane in order to obtain maximizes the margin of the training data. Classification algorithm means it is used to predict if there's something that belongs to a particular category, as showed in Algorithm (2.4)[68], [69].

Algorithm (2.4): Support Vector Machine (SVM) Algorithm

Input: Determine the various training and testing data

Output: Predicated Class Y

```
candidateSV = {closest pair from opposite classes }
while there are violating points do
        Find a violator
        candidateSV = candidateSV ∪violator
        if any $\alpha p < 0$ due to addition of c to S then
            candidateSV = candidateSV \p
            repeat till all such points are pruned
        end if
End Algorithm
```

2.5.2 Unsupervised Learning

Descriptive models are developed using unsupervised learning method. In this model we have known set of inputs but output is unknown. Unsupervised learning is mostly used on transactional data. This method includes clustering algorithms like k-Means clustering and k-Medians clustering [70].

2.5.3 Semi-supervised Learning

Semi supervised learning method uses both labeled and unlabeled data on training dataset. Classification, Regression techniques come under semi supervised learning. logistic regression, linear regression are examples of regression techniques[71].

2.6 The used Dataset

Healthcare dataset stroke was used to train and test models for predicting stroke disease. This dataset consists of 10 independent variables as features and one dependent variable as the class label that is used to predict heart disease.

The features' name are gender, age, hypertension, heart_disease, ever_married, work_type, residence_type,_avg glucose_level, bmi and smoking status. The class label has two values which are: 0 represents the absence of stroke disease; while the value 1 represents the presence of stroke disease[72].

The clinical measurements in the used dataset as (e.g. Hypertension, heart_disease, age, family history of disease) for a number of patients, as well as information about whether each patient has had a stroke. In practice, we want this method to accurately predict stroke risk for future patients based on their clinical measurements.

2.7 Evaluation Metric

The evaluation metrics were defined based on the confusion matrix, as shown in equations (2.1) to (2.5).

A- Precision

It is the number of TP divided by the number of TP and FP. The precision can be computed based on Equation (2.2)[73].

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (2.2)\ [73]$$

B- Accuracy

It is the number of correct predictions which is divided by the total number of predictions. The accuracy can be computed based on Equation (2.3) [74].

$$\text{Accuracy} = \frac{\text{TP + TN}}{\text{TP+TN+FP + FN}} \quad (2.3)\ [74]$$

C- Recall

It is the number of TP divided by the number of TP and the number of FN. This metric can be computed based on Equation (2.4) [75].

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (2.4)\ [75].$$

D- F1-score

It is the result of 2*((precision*recall)/ (precision + recall)). It is also called the f1- score or the f1- measure. An equation of this metric can be computed based on Equation (2.5) [76].

$$\text{F1} - \text{score} = \frac{(2*\text{TP})}{(2*\text{TP+FN+FP})} \quad (2.5)[76]$$

E-Detection Rate (DR)

Aka Recall, Sensitivity, Hit rate, or True positive rate (TPR), it is the measure of identified positive (anomaly) instances from all the actual positive instances, defined as the ratio of correct positive predictions to the total number of positive predictions .This metric can be computed based on Equation (2.6) [77].

$$\text{Detection Rate(DR)} = \frac{\text{TP}}{\text{TP+ FN}} \quad (2.6)[77]$$

F-False Alert Rate (FAR)

Aka fall-out or False positive rate (FPR), represents the proportion of negative prediction; this is mistakenly considered as positive (anomaly) for all negative predictions. The lower value is the better. This metric can be computed based on Equation (2.7) [78].

$$\text{False Alert Rate}(\text{FAR}) = \frac{\text{FP}}{\text{FP+ TN}} \quad (2.7)[78]$$

G- Error rate

It can be defined as the number of all wrong predictions divided by the entire number of dataset predictions, as showed in Figure (2.9) [79].

$$\text{ERR} = \frac{\text{b+c}}{\text{a+ b+c+d}} \quad (2.8)\ [79]$$

Besides in some cases, it is calculated as follow:

Error Rate = Incorrect Predictions / Total Predictions (2.9) [80]

Error Rate = 1 – Accuracy (2.10)[80]

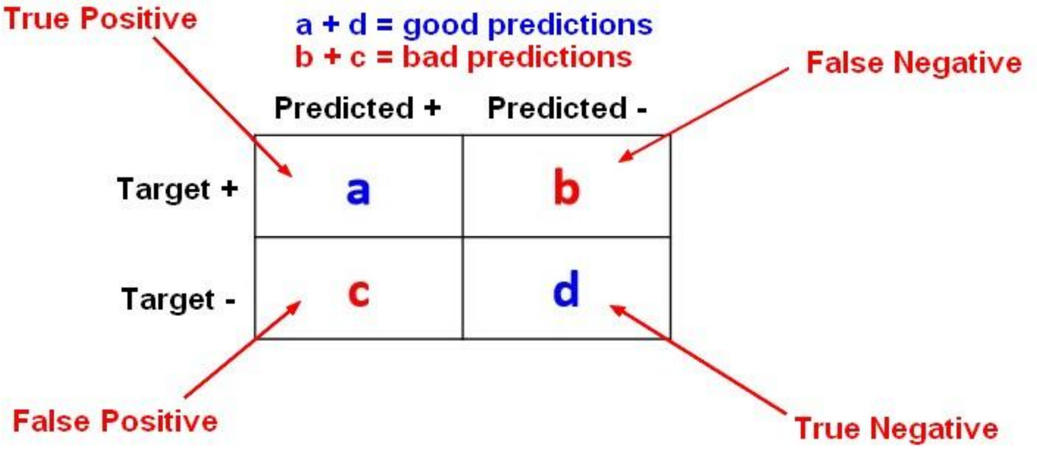


Figure 2.9 :Example of Error rate calculation.

H- Mean Absolute Error

It is a quantity utilized to measure how close forecasts or exceptions are to the definitive results. The mean absolute error is given via[81]:

$$\text{MAE}= \frac{\text{SAE}}{\text{N}} = \frac{\sum_{i=1}^{n}|x_i - \hat{x}_i|}{\text{N}} \ldots\ldots\ldots (2.11)\ [81]$$

That MEN is the collection of the absolute errors (or deviations), N is the number of non-missing data points, $\mathbf{x_i}$ is the real observations time series, $\hat{\mathbf{x}}_\mathbf{i}$ is the evaluated or forecasted time set

I- Root Mean Squared Error

It is the square root of the mean squares of the values. It squares the errors before the mean is calculated [82] and RMSE gives a relatively high weight to big errors. The RMSE of an algorithm exception with respect to the evaluated parameter $\mathbf{x_{model}}$ is referred to the square root of the mean squared error:

$$\text{RMSE} = \sqrt{\frac{\sum(x_{obs,i-}x_{model,i})}{\text{N}}} \ldots\ldots. (2.12)\ [82]$$

Where $\mathbf{X_{obs}}$ is observed values and $\mathbf{X_{model}}$ is modelled values at time/place i. Relative and absolute values are different that the absolute error is the measure of result deviation from the actual value. While relative error is a the percentage measure compared to the actual value[83].

2.8 Confusion Matrix

The performance evaluating of models are used the confusion matrix to calculate accuracy, precision, recall, and f-measure. Confusion matrix describes the performance of a model on set of test data. It gives two types of correct predictions and two types of incorrect predictions for the classifier. Table (2.2) shows the confusion matrix. TP is the predicted output as true positive, TN is the predicted output as true negative, FP is the predicted output as false positive, and FN is the predicted output as a false negative. The accuracy, precision, recall, and f-measure (f-score) are defined in the following[84]:

There are various measures can used to evaluate the classification approach. Confusion matrix was chose to get main performance measures for evaluation. It used to describe the performance of a classification approach or "classifier” by test data [85].

- **True Positive (TP)**: denotes to the positive examples that are properly classified.
- **False Negative (FN)**: denotes to the positive examples that are incorrectly classified.
- **False Positive (FP)**: denotes to the negative examples that are incorrectly predicted and classified.
- **True Negative (TN)**: denotes to the negative instances that are properly predicted by the classification model[86].

Table 2.2: Confusion Matrixes.

Confusion Matrix		**Predicated Class**	
		Positive +	**Negative -**
Actual Class	**Positive +**	**TP**	**FN**
	Negative -	**FP**	**TN**

Chapter Three

The Proposed Approach

3.1 Overview

This chapter explains the main steps of the proposed system implementation, configuration and installation for Java and Hadoop system. The state of both data mining and Hadoop(Count/Weight) has been explained in this chapter. In addition, it is also involve an explanation of the proposed algorithms as RF, NB, SVM, and DT algorithms.

3.2 The proposed system implementation steps

The proposed system is implemented in Java Eclipse, and Hadoop. The machine learning algorithms are implemented with Java and MapReduce of Apache Hadoop implemented to decrease time and system computation of CPU and RAM memory through execution time by providing an efficient method for big dataset processes. Hadoop comprises of two parts, which are the storage part and the other being the data processing part.

The storage part is called the Hadoop Distributed File System (HDFS) and the processing part is called MapReduce, while the proposed system based on the second part as the MapReduce. The proposed system data mining/ machine algorithm with Hadoop steps consisted of three main stages as follow:

Stage one: Data mining pre-processing on full data set to transform the row data of big healthcare dataset in a useful and efficient format.

Stage two: Passing new dataset (efficient dataset) after pre-processing stage to Hadoop MapReduce to create key, value attributes.

Stage Three: Using Machine learning algorithms and find results, as it showed in Figure (3.1).

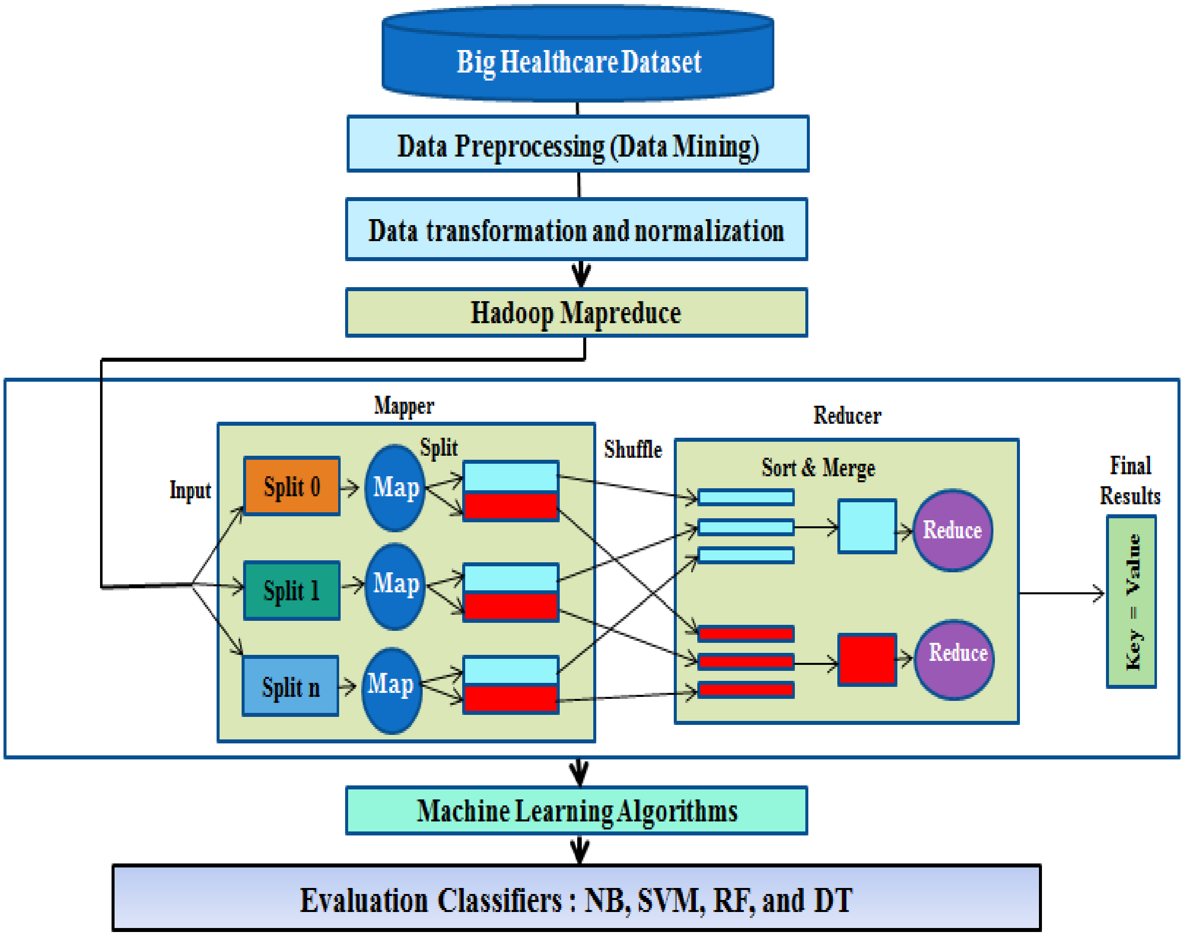


Figure 3.1: The proposed System Stages.

3.2.1 Data Mining Pre-processing

Data pre-processing is a data mining technique which is used to transform the row data in a useful and efficient format, alongside, it based on the useful data from preprocessing to evaluated with machine learning classifiers as it showed in Figure (3.2) as the main steps of data preprocessing.

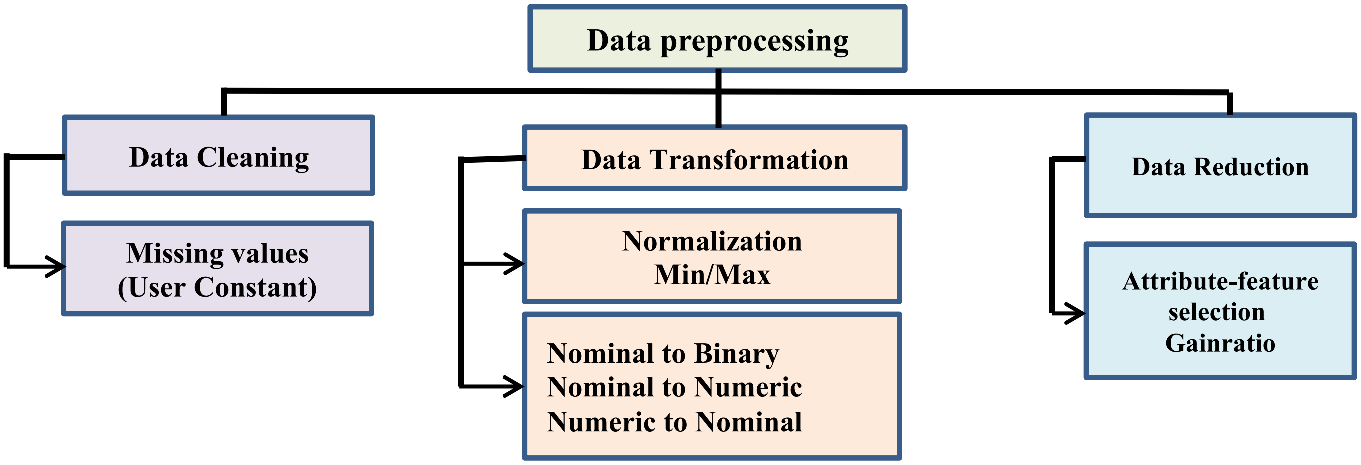


Figure 3.2: The used Data Mining Pre-processing Methods.

A- Normalization : It is done in order to scale the data values in a specified range.

B- Attribute-feature selection: The highly relevant attributes should be used, rest all can be discarded. For performing attribute selection, one can use level of significance and p- value of the attribute. P-value gives us the probability of finding an observation under an assumption that a particular hypothesis is true. This probability is used to accept or reject that hypothesis. It used to find relation between two variables. The attribute having p-value greater than significance level can be discarded. new attributes are constructed from the given set of attributes to help the mining process. In this work, this method used as gain ratio to determine the splits and to select the most important features.

C- Missing values: It can be replaced by the minimum, maximum or average value of that Attribute. Zero can also be used to replace missing values. Any replenishment value can also be specified as a replacement of missing values. Numeric replacement for numeric features and ‘constant’ for categorical features. In this work, this method used as user constant value to replace missed value attribute in dataset records especially in smoking attribute as it used “Unknown” value for not clear state of smoking.

D- Nominal to Binary : The proposed system used this method to convert nominal attribute values into binary state. In this methodology, the used system is based on converting nominal (string values) in attribute big healthcare dataset into binary data as (0, 1) as it is optimal to use in the proposed machine learning algorithm to increase accuracy of prediction.

E- Nominal to Numeric : It used to convert nominal values in dataset into numeric values. In this work, the method is used to convert string value in dataset attribute such as ever_married, work_type, and residence_type attributes.

F- Numeric to Nominal : It used to convert numeric values into nominal values. In this work, the used method applied on class value to deal with them as nominal value through classify class state as stroke (class 1) and non-stroke(class 0).

In addition, collecting the healthcare dataset stroke data , it is a mix of both categorical and numeric data and since ML algorithms understand data of numeric nature let’s encode our categorical data into numeric ones using Label Encoder, or one hot encoding. Label Encoder(data mining transformation techniques) is a technique that will convert categorical data into numeric data. It takes value in ascending order and converts it into numeric data from 0 to n-1. While the used system steps of the data mining with machine learning classifiers showed in Figure (3.3). After the processes of data mining, the dataset splitted into training 70% and testing 30% as the best splitting ratio, then training passed to M.L and testing to trained classifiers to build the evaluation model based on the testing dataset.

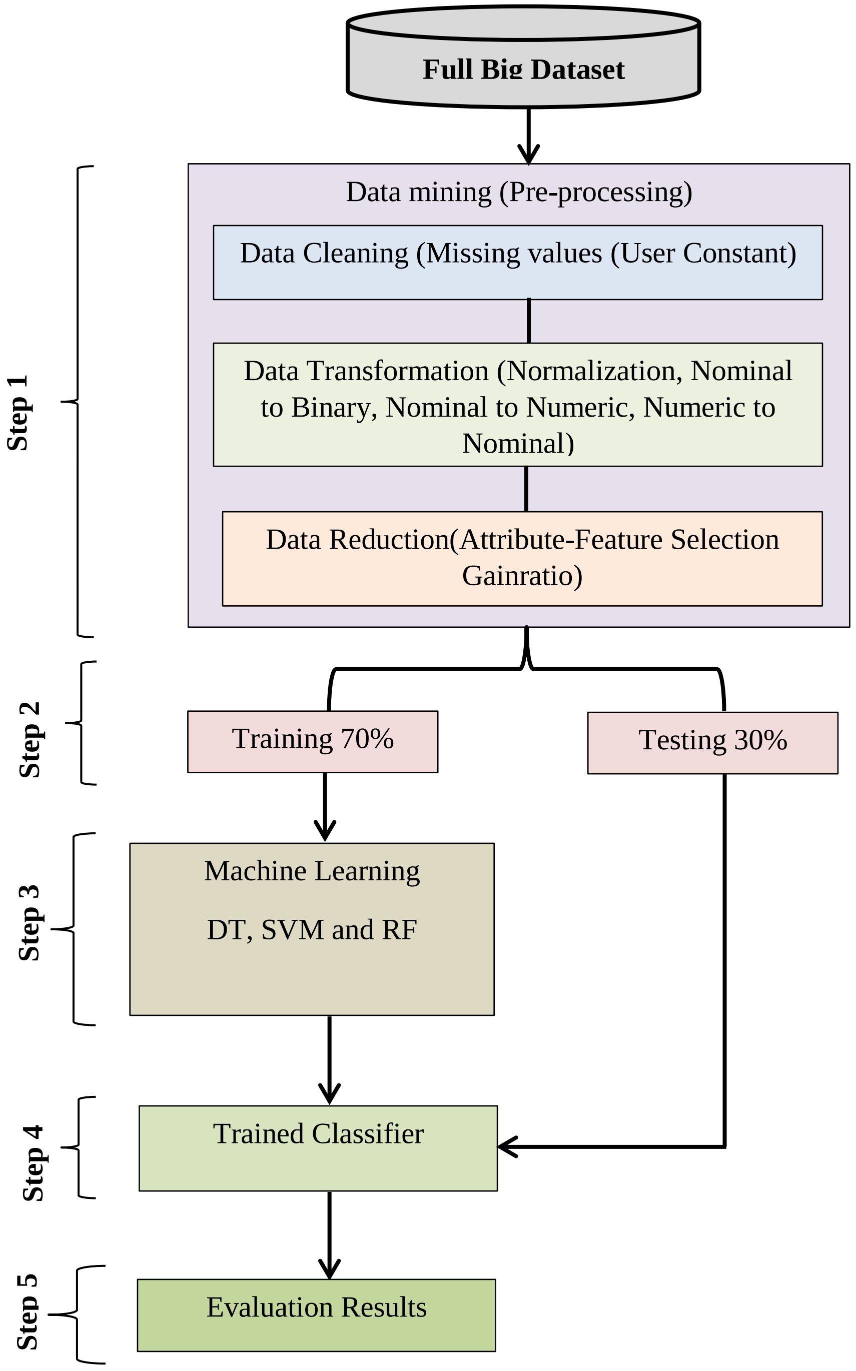

Full Big Dataset
Data mining (Pre-processing)
Data Cleaning (Missing values (User Constant)
Data Transformation (Normalization, Nominal to Binary, Nominal to Numeric, Numeric to Nominal)
Data Reduction(Attribute-Feature Selection Gainratio)
Training 70%
Testing 30%
Machine Learning
DT, SVM and RF
Trained Classifier
Evaluation Results
Step 1
Step 2
Step 3
Step 4
Step 5

Figure 3.3: The proposed Machine Learning system Model.

Besides, Table (3.1) showed the used data preprocessing methods for each attribute in the big healthcare data set.

Table 3.1: Data mining methodology for each attributes in Dataset.

Column Name	Type	Value	Data Mining (Pre-processing)
Gender	string	[Male Female]	Transformation(Nominal to Binary)
age	Double	[0.08] – [82]	Attribute-feature selection (GainRatio)
hypertension	integer	[0 1]	Normalization (Min/Max)
heart_disease	integer	[0 1]	Normalization (Min/Max)
ever_married	String	[Yes No]	Transformation (Nominal to Binary)
work_type	String	[Children, Private, Self-employed, Govt_job, Never_worked]	Transformation (Nominal to Numeric)
Residence_type	String	[Urban Rural]	Transformation (Nominal to Binary)
smoking_status	String	[formerly smoked, never smoked, smokes, Unknown]	Transformation (Nominal to Numeric) , Cleaning Replace Missing Value((User Constant))
stroke	Integer	[0 1]	Transformation (Numeric to Nominal)

3.2.2 Hadoop MapReduce

In the proposed system Hadoop is implementing a mapping system to locate data in a cluster. Besides, it processes the used Big healthcare dataset in less time. It depends on the DataNode to save real data in HDFS, and DataNodes sends information to the NameNode about the files and blocks packed in that node and responds to the NameNode for whole file system procedures.

The proposed system shows the major functions of Hadoop MapReduce for count/weight states as follow :

- Input and output: each a set of key/value pairs.
- The functions implemented as:

A- Map (k1, v1) -> list(k2, v2)

- takes an input key/value pair
- produces a set of intermediate key/value pairs, and then passed to shuffle processs.

B- Reduce (k2, list(v2)) -> list(k3, v3)

- takes a set of values for an intermediate key
- produces a set of output value

The MapReduce framework guarantees that all values associated with the same key are brought together in the reducer

A- Map task is the first phase in MapReduce model. It receives input functions (considered as Datasets) and small them into minimal sub-task. After that, needed computation is performed on every sub-function in parallel. The mapper task is processing of input data. In general, the input data is in of shape of a file or directory and is saved in the HDFS. It crossed to the mapper task. The mapper becomes to process the data and produces many small blocks of data, as showed in Algorithm (3.1).

Algorithm (3.1) Hadoop/mapper using Associative Arrays

Input : Large number of attributes/records of (Big Healthcare Dataset)

Output : For every attributes line in a document output (Key, "value")

```
Class Mapper
method Map(docid a,doc d)
H ←new AssociativeArray
for all term t ∈doc d do
H{t}←H{t}+ 1 // counts for entire document
for all term t ∈H do
Emit(term t,count H{t}
```

End of Algorithm

B- Reduce task is the final phase in MapReduce technique. It receives a listing of <Key, List<Value>> sorted couples from Shuffle task and implement reduce step. The output of Mapper class is utilized like an input via reducer class that seeks matching couples and reduces them. This phase is the integration of the shuffle phase and the Reduce phase. The Reducer's task is processing the data, which delivered from the mapper. Then, it produces a modern collection of output that will be sorted in the HDFS, as showed in Algorithm (3.2).

Algorithm (3.2) Hadoop/Reducer using Associative Arrays

Input : Output of Mapper class (Key, "value")

Output : Sum all occurrences of attributes line and output (attribute row, total-count)

```
class Reducer
    method reduce(term t, counts [c1,  c2,  ..])
        sum = 0;
        for all counts c in [c1, c2, ...] do
            sum = sum + c;
        emit(t,  sum);
```

End of Algorithm

In Hadoop, configures are involved etc/hadoop/under /conf. Individual configuration files will be discovered for various Hadoop elements.

In addition, the TF-weight score was considered to be of high importance in balancing weight between less common words and most common words. Each token in review is calculated via the TF; such frequency was offset via the frequency related to the token in whole dataset. The value of TF indicates the token significance to documents in dataset. The weight of Hadoop/Weight term computes normalized Term Frequency (TF), how many time a word shown in the document, divided by the overall number of words in the same document. TF-Weight is explained in Algorithm (3.3).

Algorithm (3.3): Hadoop/Weight TF Algorithm

Input: Preprocessing Dataset(Count column)

Output: TF/Weight for each attributes(count value)

Process:

```
- For each attributes line( 9 attributes rows with count value) in dataset do
- Compute frequency of each count value as shown in following:
    TF (t) = (Number of times attributes count (A) appears in a document) / (Total number of terms in the document)
- End for
```

End of Algorithm

The proposed system approach of hadoop/count and Hadoop/Weight with TF showed in Figure (3.4).

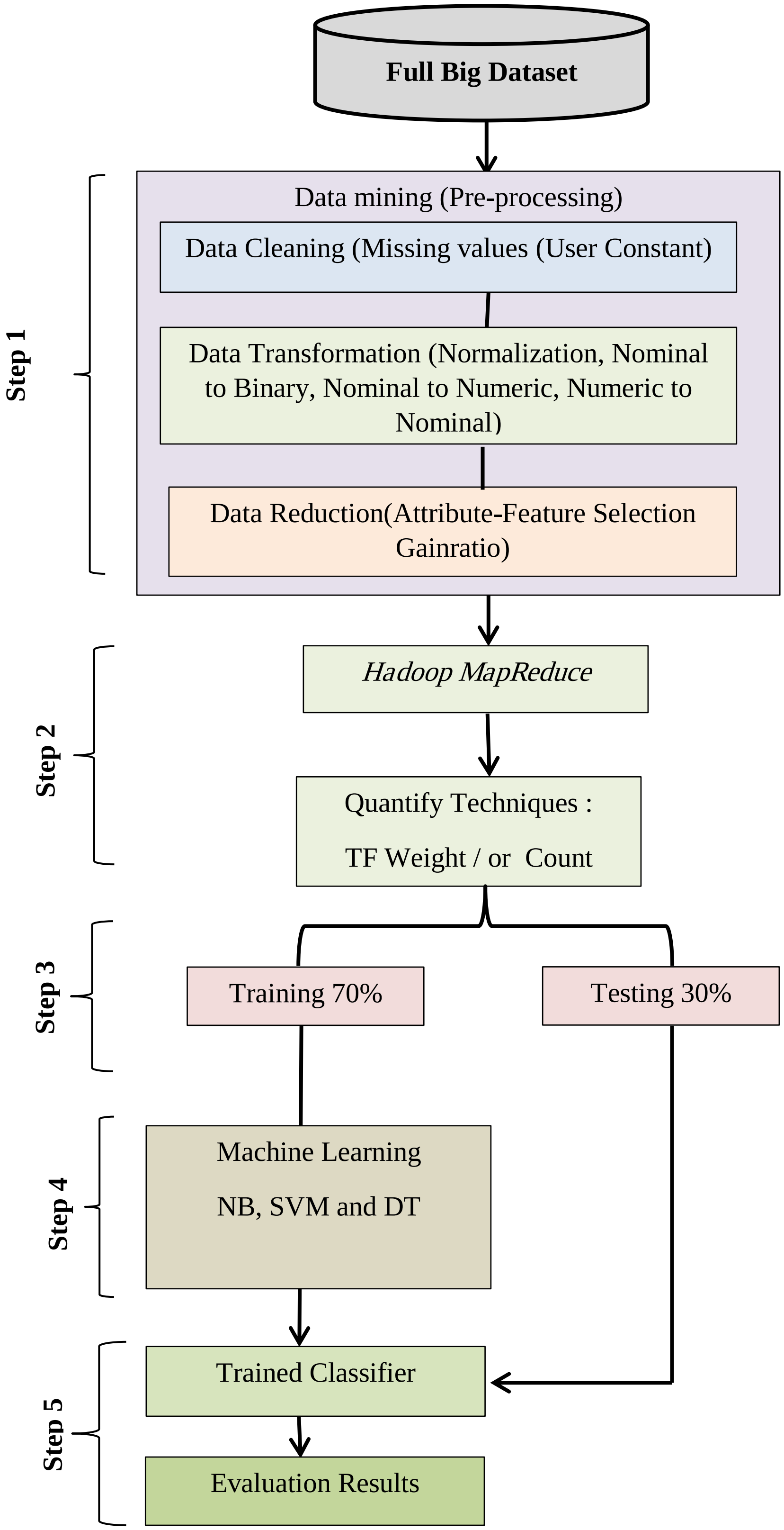

Full Big Dataset
Data mining (Pre-processing)
Data Cleaning (Missing values (User Constant)
Data Transformation (Normalization, Nominal to Binary, Nominal to Numeric, Numeric to Nominal)
Data Reduction(Attribute-Feature Selection Gainratio)
Step 1
Hadoop MapReduce
Quantify Techniques :
TF Weight / or Count
Step 2
Training 70%
Testing 30%
Step 3
Machine Learning
NB, SVM and DT
Step 4
Trained Classifier
Evaluation Results
Step 5

Figure 3.4: The proposed system Model.

3.2.3 Machine Learning Algorithms

The proposed system based on top 4 machine learning algorithms from the point of view accuracy and building time. Regarding the present study, experiments have been conducted on (Random forest(RF), Naïve Bays (NB), Support vector machine (SVM) , Decision tree(DT)).

3.2.3.1 Random Forest (RF) Algorithm

It is an (ensemble classifier) consisting of multiple decision-trees and outcomes the class results mode of each tree. It can accommodate many input variables and selects instances automatically without pruning, trees are grown to maximum depth and each of them is classified separately.

3.2.3.2 Naive Bayes (NB) Algorithm

The technique of Naive Bayes1classifier depended on the so-called Bayesian theorem. In spite of its simplicity but can superior to many sophisticated classification methods. We can define the classifier as machine learning model that used to distinguish different objects based on definite features. Naive Bayes is probabilistic a model in machine learning that is used in the classification task.

3.2.3.3 Support Vector Machine (SVM) Algorithm

A support vector machine (SVM) is a non-probabilistic binary linear classifier. The non-probabilistic aspect is its key strength. This aspect is in contrast with probabilistic classifiers such as the Naïve Bayes. That is, an SVM separates data across a decision boundary (plane) determined by only a small subset of the data (feature vectors). The data subset that supports the decision boundary are aptly called the support vectors. The remaining feature vectors of the dataset do not have any influence in determining the position of the decision boundary in the feature space.

3.2.3.4 Decision Tree (DT) Algorithm

It is based on establishing classification systems based on multiple covariates or for developing prediction algorithms for a target variable. This method classifies a population into branch-like segments that construct an inverted tree with a root node, internal nodes, and leaf nodes. The algorithm is non-parametric and can efficiently deal with large, complicated datasets without imposing a complicated parametric structure.

While, as mentioned above the Pseudo code prediction of the proposed system is based on the data mining /Hadoop/count/weight and machine learning, it is explained in Figure (3.5).

Pseudo code Prediction of big healthcare data by using data mining / Hadoop/MapReduce and machine learning classifiers algorithms	
Input	Big healthcare Stroke data (dataset)
Output	Prediction of stroke
1.	procedure Mapper(key, value = o_i)
2.	for all object o_i in HDFS − block do
3.	dataBlock ← load(o_i);
4.	end for
5.	for all feature X_f in X do
6.	if (IsContinuos(X_f)) then
7.	$B_{z,f}$ ← ComputeEquiFrequencyBinBoundaries(dataBlock, f);
8.	Call Output(<key = f, value = $B_{z,f}$ >);

9.	end if
10.	end For
11.	end procedure
12.	procedure Reducer(key = f, value = List($B_{z,f}$))
13.	B_f $\leftarrow$ newSortedList()
14.	for all $B_{z,f}$ in List($B_{z,f}$) do
15.	B_f $\leftarrow$ insert($B_{z,f}$);
16.	end for
17.	Call Output(<key = f, value = B_f >);
18.	end procedure
19.	procedure RANDOM FOREST(etc..)(<key = f, value = B_f >);
22.	For *all i* to *c* do
23.	Randomly sample the training data *D* with replacement to produce D_i
24.	Create a root node, $N_i$containing D_i
25.	Call Build classifier (N_i)
26.	end For
27.	Build classifier (N):
28.	if *N* contains instances of only one class then
29.	Return
30.	Else
31.	Select the feature *F* with the highest information gain
33.	for *all i* to *f* do
34.	Set the contents of N_i to D_i, where $D_i$is all instances in *N* that match F_i
35.	Call Build classifier (N_i)
36.	end for

```
end if
end procedure
procedure Machine learning classifier (<key = f, value = $B_f$ >);
If(D is “pure”) ||(another stopping characteristics match ) then
end if
For all attribute $\alpha \in D$ D do
  Count characteristics of impurity task if the split is on $\alpha$
  $\alpha_{best}$ ← best attribute based on above Counted characteristics
  classifier ← produce a decision node that test $\alpha_{best}$ in the root
  $D_V$ ← Induced sub-dataset from D depend on $\alpha_{best}$
  Return tree
end For
end procedure
```

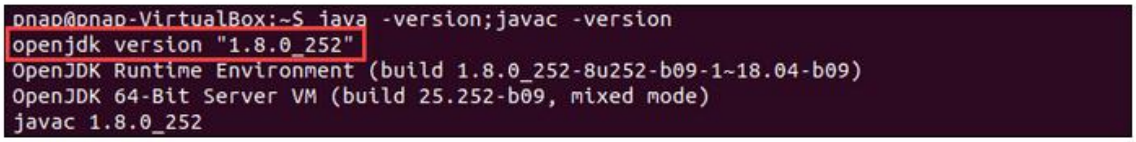
Figure 3.5: Pseudo code Prediction of big healthcare Stroke Dataset.

3.3 System Installation Requirements

The proposed system installation is based on deploying Hadoop services on a single node and multimode case are installed on Linux and also tested in Windows 7 64-bits. The installation requirements explained as showed in Appendix A.

While the main system steps explained in Appendix B .

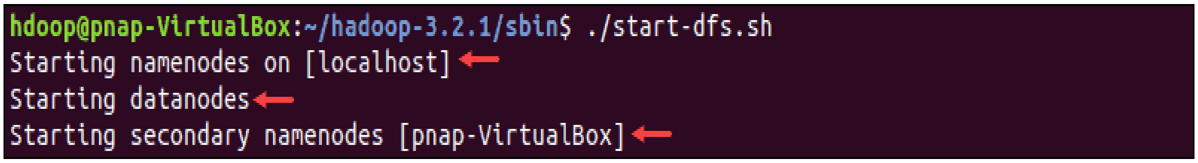
As well as, the explanation of each steps of installation in Linux explained in Appendix C. Also, to clarify the Java copy version installed in the computer, it is possible to rely on the following command in Figure (3.6).



```
pnap@pnap-VirtualBox:~$ java -version;javac -version
openjdk version "1.8.0_252"
OpenJDK Runtime Environment (build 1.8.0_252-8u252-b09-1~18.04-b09)
OpenJDK 64-Bit Server VM (build 25.252-b09, mixed mode)
javac 1.8.0_252
```

Figure 3.6:Check running and Java version.

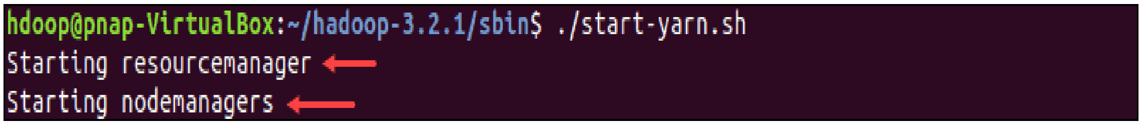
Besides, it is possible to explain how the system takes some time to carry out the Nodes and assign tasks, as showed in Figure (3.7)



```
hdoop@pnap-VirtualBox:~/hadoop-3.2.1/sbin$ ./start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [pnap-VirtualBox]
```

Figure 3.7: Hadoop start over processes.

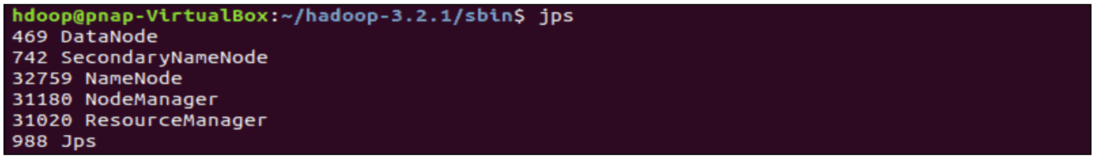
Once the installation is completed correctly, it is possible to access and operate the system through command shown in Appendix D. Besides, the management of resource and nodes showed in Figure (3.8)



```
hdoop@pnap-VirtualBox:~/hadoop-3.2.1/sbin$ ./start-yarn.sh
Starting resourcemanager
Starting nodemanagers
```

Figure 3.8: Running the Yarn process of Hadoop.

It can also be ensured in a very simple way that all the elements are active and working properly as explained in command Appendix E, with Figure (3.9).



```
hdoop@pnap-VirtualBox:~/hadoop-3.2.1/sbin$ jps
469 DataNode
742 SecondaryNameNode
32759 NameNode
31180 NodeManager
31020 ResourceManager
988 Jps
```

Figure 3.9: The running Hadoop jobs.

3.3.1 Configuration the proposed Environment

The proposed system is based on the JAVA and Hadoop environments and the main configuration for both showed as follow :

3.3.1.1 JAVA Eclipse Configuration

The main class of the proposed algorithms code in Java as follows in the Algorithm(3.4) of the Java machine learning class.

Algorithm(3.4): Machine learning main Class Algorithm

Input : Big Healthcare Stroke Dataset

Output: Machine learning evaluation results

```
Begin
    -  converters.ConverterUtils.DataSource.
Step1:  For all classifiers do
        Begin
            DataSource source = new DataSource("C:\\Big healthcare stroke.csv");
            Instances dataset = source.getDataSet(); //Getting data source
            M.L type Rule = new M.L type (); // building classifier algorithm
            Rule1.buildClassifier(dataset);
        End
Step 2: For each Building Evaluators do
        Begin
            Evaluation eval = new Evaluation(dataset);
            Rest Dataset for Evaluation
            DataSource source1 = new DataSource("C:\\new stroke.csv ");
            Instances testdataset = source1.getDataSet();
            Set class index to the last attribute
            eval.evaluateModel(Rule1, testdataset);
        End
Step 3: For all evaluation results do
        Begin
            Print evaluation results based on the machine learning type
            eval.pctCorrect(),eval.pctIncorrect(),eval.meanAbsoluteError(),eval.relative
            AbsoluteError(), eval.rootRelativeSquaredError(), eval.precision(),
            eval.recall(), eval.fMeasure(), eval.errorRate(), eval.avgCost()
        Then Building the confusion Matrix parameters
    End
```

3.3.1.2 Apache Hadoop Configuration

Apache Hadoop configuration in Linux 18.04 LTS also it configures as an extension with Eclipse

Step1: Configuring Hadoop in Eclipse

1-Install the eclipse
- Eclipse Oxygen
2-Download: hadoop-eclipse-plugin-2.6.jar
3-copy the hadoop-eclipse-plugin-2.6.jar and paste it to eclipse dropins folder.
4-Go to the eclipse > Window > Perspective > open perspective > others
Map/Reduce is chosen from the menu > Ok
Now there is DFS Location
5- After this, configure hadoop is needed:
Start all process

Eclipse > Right Click > Create New Location

Step 2 create lib folder inside program project. Right click on program->New->Folder called lib

Step 3 Copy the Three jar file in lib folder and Create Three java class.

Jar file name and location.

A) /hadoop-2.6.0/share/hadoop/common/lib : commons-cli-1.2.jar

b) /hadoop-2.6.0/share/hadoop/common : hadoop-common-2.6.0.jar

C) /hadoop-2.6.0/share/hadoop/mapreduce/ : hadoop-mapredure-client-core-2.6.0.jar

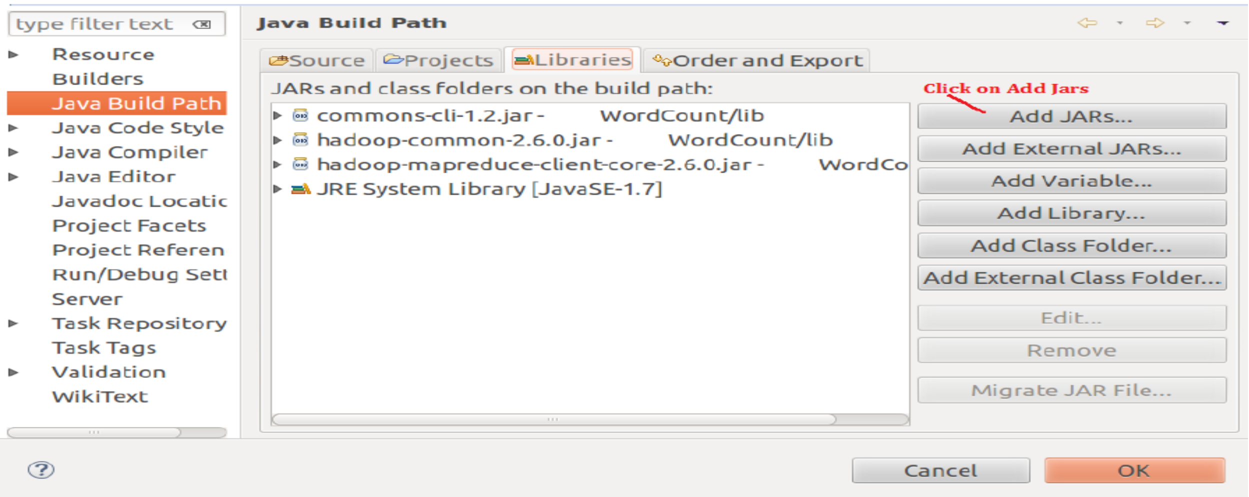
Step 4 Three java file for Driver code, Mapper code, Reducer code.

MyDrive.java: main class

MyMapper.java

MyReducer.java

Step 5 set java build path (class path) by Right Click on ->program2Project-> Properties->JavaBuildPath->Libaries->Click on Add jar and find three jar file in lib folder of program2 project. Figure (3.10) explains the way of adding jar by using eclipse



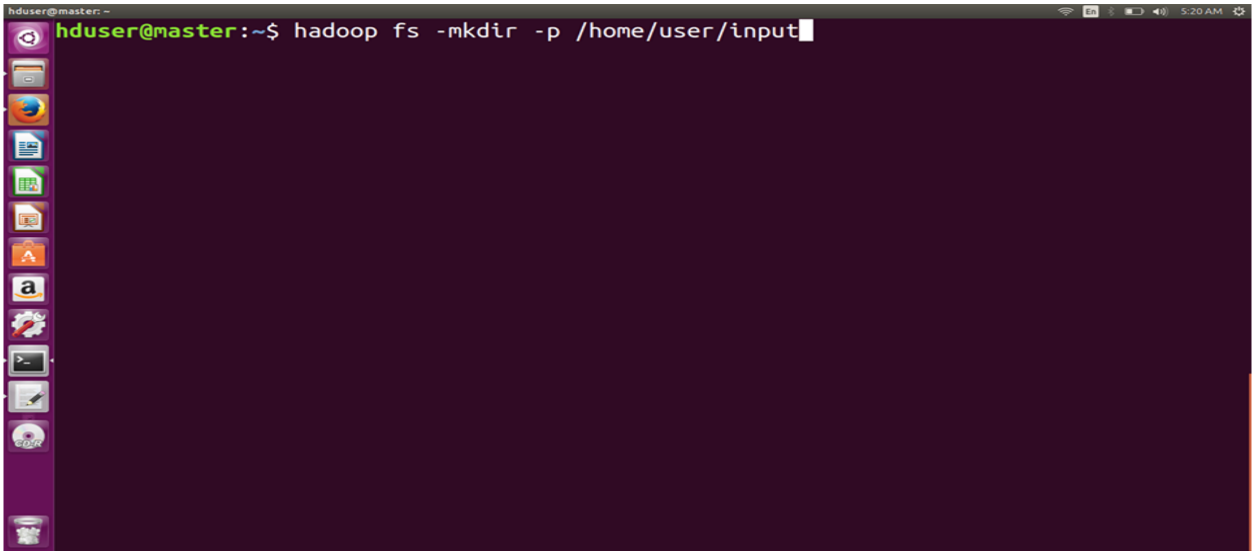
JARs and class folders on the build path:

- commons-cli-1.2.jar - WordCount/lib
- hadoop-common-2.6.0.jar - WordCount/lib
- hadoop-mapreduce-client-core-2.6.0.jar - WordCo
- JRE System Library [JavaSE-1.7]

Figure 3.10: The Way of Adding jar by Using Eclipse

Step 6 Create Jar file Right click on program2Project->Export->java->jar file->Browse->give jar test5.jar filename ->OK->finish.

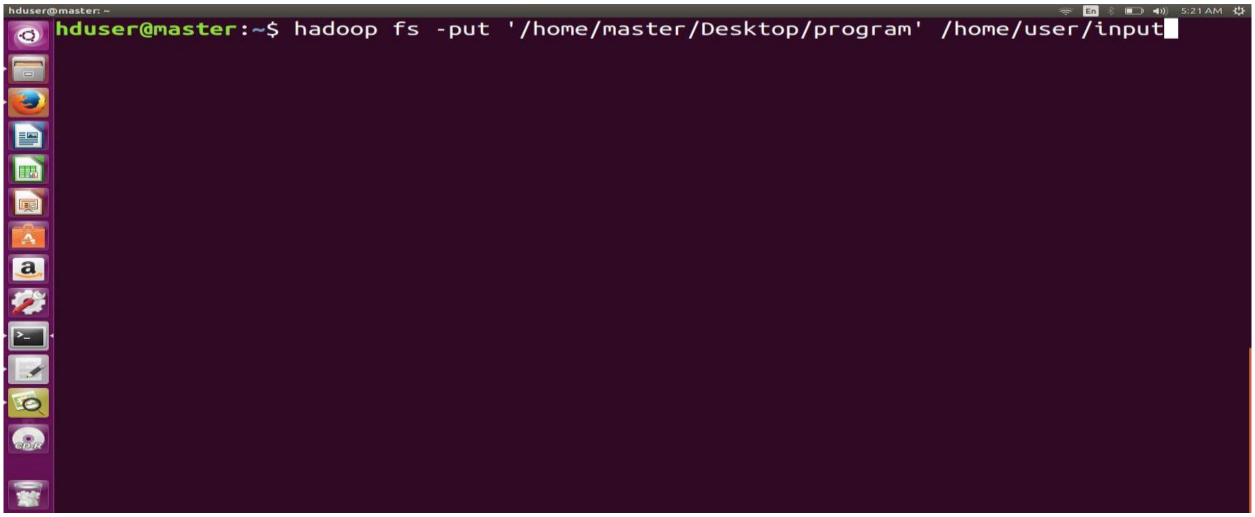
Step 7 create directory inside hdfs name is /home/user/input. Figure (3.11) shows the way of creating file in hdfs.



```
hduser@master:~$ hadoop fs -mkdir -p /home/user/input
```

Figure 3.11 : The way of Creating File in HDFS.

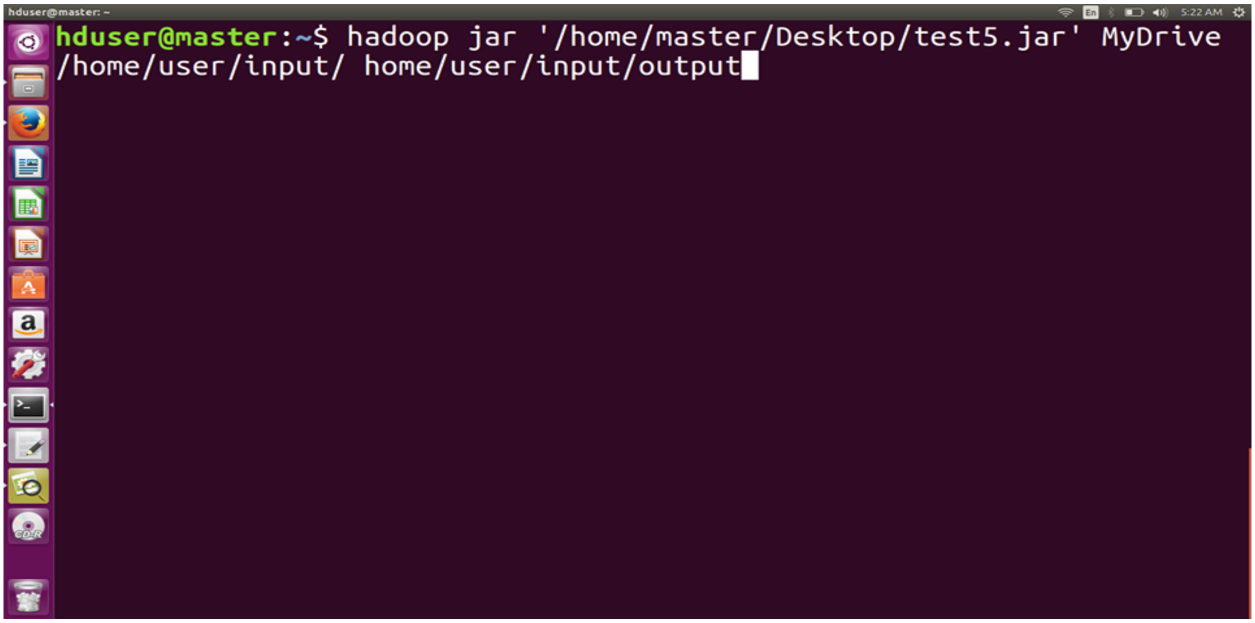
Step 8: move inputfile.txt in hdfs /home/user/input director. Figure (3.12) explains the way of transferring file in hdfs.



```
hduser@master:~$ hadoop fs -put '/home/master/Desktop/program' /home/user/input
```

Figure 3.12 : The Way of Transferring File in HDFS.

Step 9: Run BDSstroke1.jar file in HDFS. Figure (3.8) explain how run any code in hadoop.



```
hduser@master:~$ hadoop jar '/home/master/Desktop/test5.jar' MyDrive
/home/user/input/ home/user/input/output
```

Figure 3.13: The Run any Code in Hadoop.

In Hadoop, configs are involved etc/hadoop/under /conf. Individual configuration files will be discovered for various Hadoop elements, and it is worth supplying a rapid overview of them in Table 3.2 the working directory and configuring Hadoop setup files are set. At last, hadoop in Eclipse is configured so can work with MapReduce.

Table 3.2: Hadoop Configuration Files.

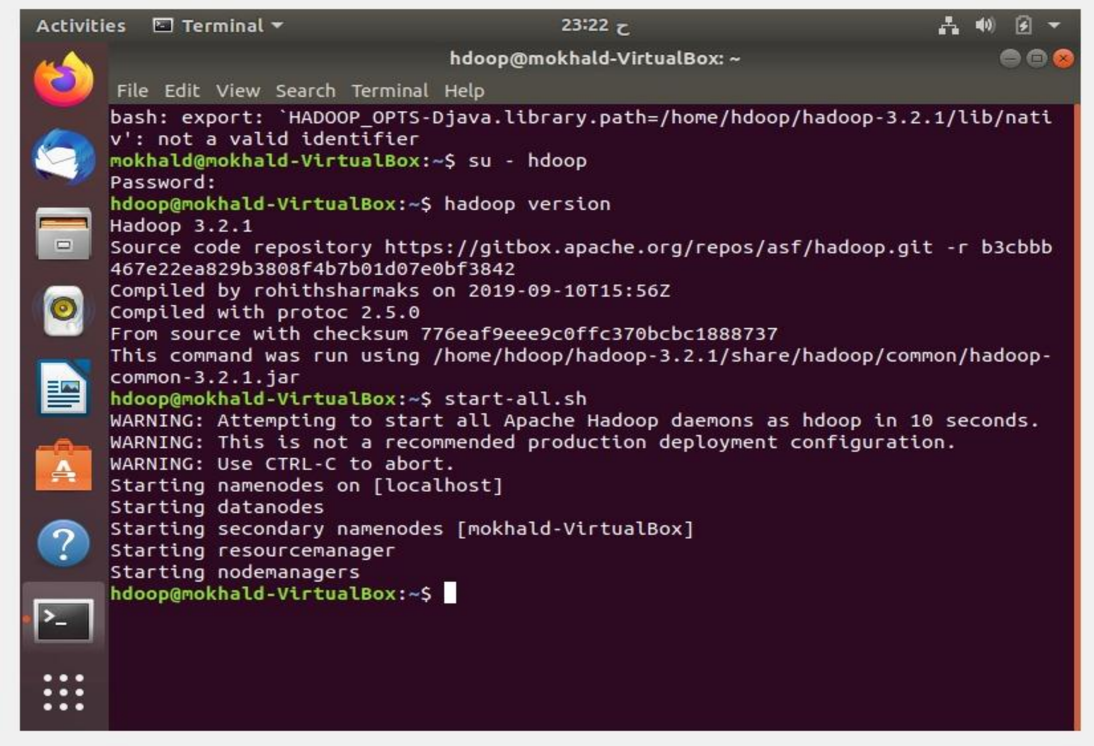
Filename	Description
hadoop-env.sh	This file involves some environment changeable settings utilized via Hadoop to influence some sides of Hadoop daemon behavior. Specification directory places like the log file and the places of the master and slave files is applied here.
core-site.xml	Involves system-level Hadoop configuration components, like the HDFS URL, the Hadoop temporal directory, and script location. Settings in this file

	exceed the settings in core-default.xml.
hdfs-site.xml	This file involves the configuration settings of HDFS like size of block , default file replication count and licenses testing on HDFS
mapred-site.xml	HDFS settings like the default number of reduce functions, default minimum /maximum function memory sizes.
yarn-site.xml	YARN configuration choices are saved in the hadoop-3.x.x/etc/hadoop/yarn-site.xml file and are modifiable via the root user. This file involves configuration data that exceed the default counts for YARN parameters. Exceeds of the default counts for core configuration characteristics are saved in the Default YARN parameters file.
masters	Involves a listing of hosts, which are Hadoop masters. This term is misunderstood and should have been named secondary-masters. When Hadoop is begun, it will start NameNode and JobTracker on the localhost from that the first command is put to begin, after that SSH to whole nodes in this file to begin the SecondaryNameNode.
Slaves	Involves a listing of hosts, which are Hadoop slaves. When Hadoop is started, it will SSH to every host in this file and begin the DataNode and TaskTracker daemons.

The main steps of the start Hadoop Cluster as follow :

1- The main steps of the start Hadoop Cluster as follow in Appendix F .

Similar to above but start/stop HDFS daemons separately on whole nodes from the master machine. This will begin a Namenode, a DataNode, and SecondaryNameNode on the machine. The command (JPS) (Java Virtual Machine Process Status Tool) is utilized to test whole Hadoop daemons such as NameNode, DataNode. That is carrying out on the machine. Test whether the predicted Hadoop processes are carrying out. Figure (3.14) shows start HDFS daemons

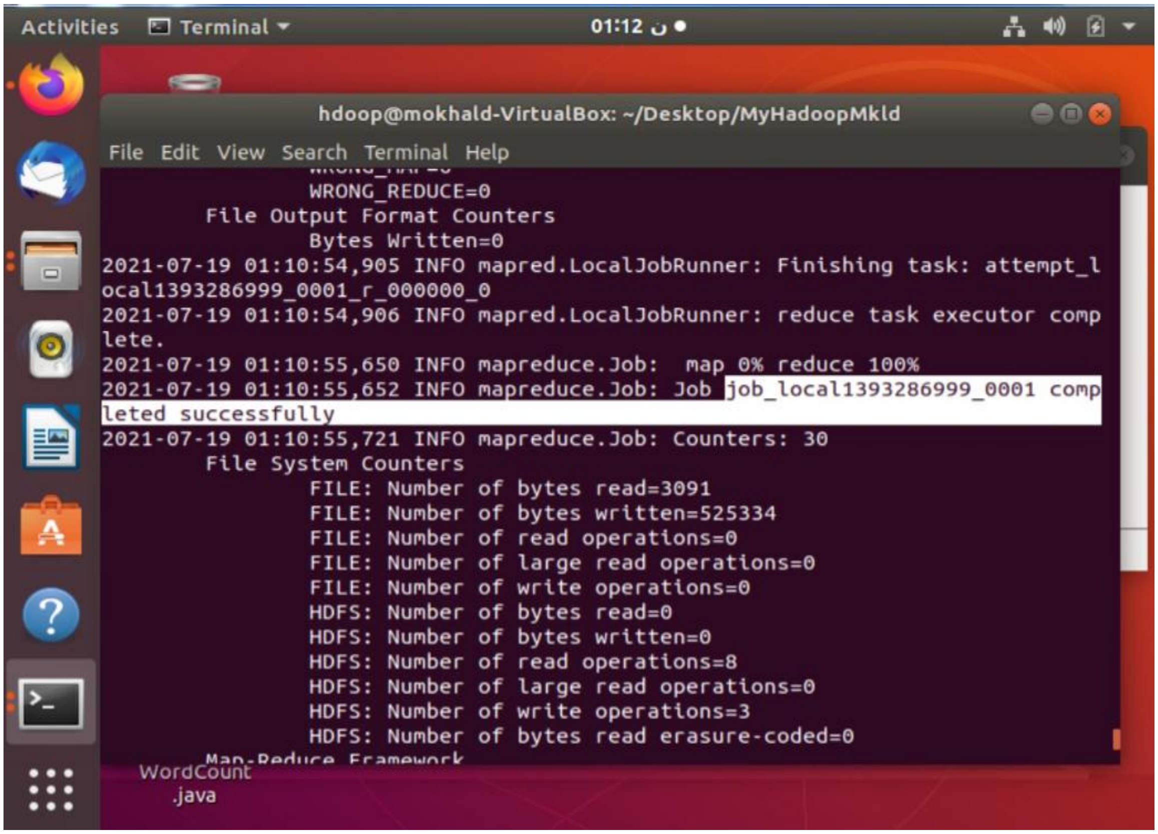

Activities
Terminal
23:22 ج
hdoop@mokhald-VirtualBox: ~
File Edit View Search Terminal Help

```
bash: export: `HADOOP_OPTS-Djava.library.path=/home/hdoop/hadoop-3.2.1/lib/nati
v': not a valid identifier
mokhald@mokhald-VirtualBox:~$ su - hdoop
Password:
hdoop@mokhald-VirtualBox:~$ hadoop version
Hadoop 3.2.1
Source code repository https://gitbox.apache.org/repos/asf/hadoop.git -r b3cbbb
467e22ea829b3808f4b7b01d07e0bf3842
Compiled by rohithsharmaks on 2019-09-10T15:56Z
Compiled with protoc 2.5.0
From source with checksum 776eaf9eee9c0ffc370bcbc1888737
This command was run using /home/hdoop/hadoop-3.2.1/share/hadoop/common/hadoop-
common-3.2.1.jar
hdoop@mokhald-VirtualBox:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hdoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [mokhald-VirtualBox]
Starting resourcemanager
Starting nodemanagers
hdoop@mokhald-VirtualBox:~$ 
```

Figure 3.14 : The HDFS Daemons Running.

2- The process of start and end showed in Appendix G.

Utilized to start and stop whole hadoop daemons immediately. Putting it out on the master machine will start/stop the daemons on whole nodes of a cluster. The command (jps) JPS (Java Virtual Machine Process Status Tool) is a command is utilized to test whole Hadoop daemons such as NameNode, DataNode, Resource Manager, Node Manager. That are carrying on the machine. Test whether the predicted Hadoop processes are carrying out. Figure (3.15) explains start whole hadoop


Activities
Terminal
01:12
hdoop@mokhald-VirtualBox: ~/Desktop/MyHadoopMkld
File Edit View Search Terminal Help
WordCount
.java

```
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Output Format Counters
                Bytes Written=0
2021-07-19 01:10:54,905 INFO mapred.LocalJobRunner: Finishing task: attempt_l
ocal1393286999_0001_r_000000_0
2021-07-19 01:10:54,906 INFO mapred.LocalJobRunner: reduce task executor comp
lete.
2021-07-19 01:10:55,650 INFO mapreduce.Job:  map 0% reduce 100%
2021-07-19 01:10:55,652 INFO mapreduce.Job: Job job_local1393286999_0001 comp
leted successfully
2021-07-19 01:10:55,721 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=3091
                FILE: Number of bytes written=525334
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=0
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=8
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=3
                HDFS: Number of bytes read erasure-coded=0
        Map-Reduce Framework
```

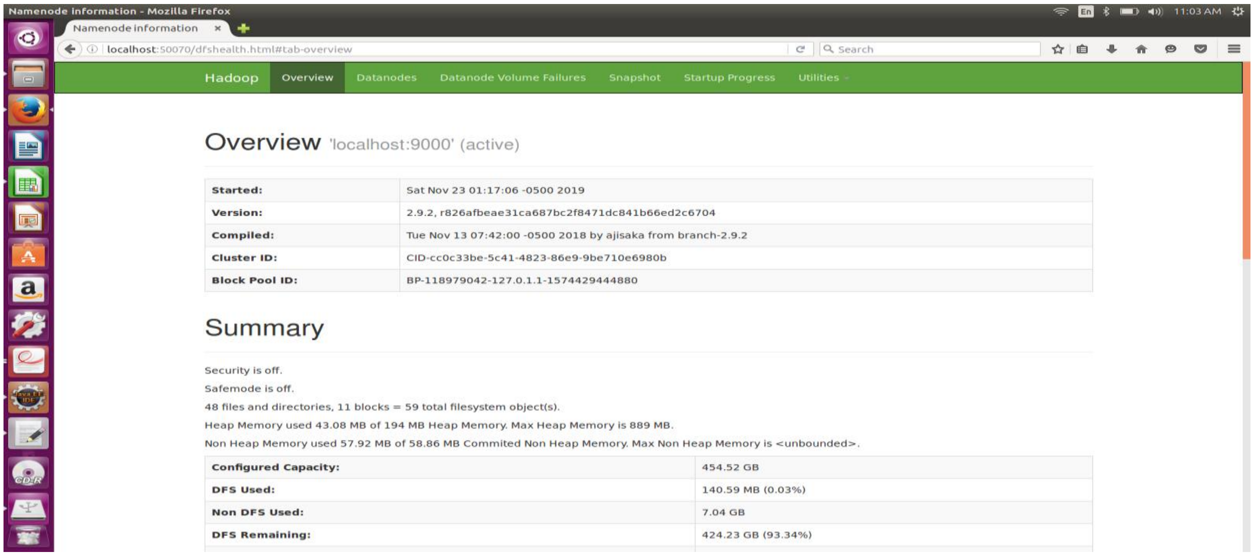
Figure 3.15: Start Whole Hadoop

3-There is a group of ports that they can use in the Hadoop.

Table 3.3: The used URL for the proposed Node/Node Manager.

NameNode	http://localhost:50070
ResourceManager	http://localhost:8088
MapReduceJobHistory Server	http://localhost:19888

4- Check if the hadoop is accessible through browser by hitting the below URLs. Figure (3.16) explain Hadoop NameNode started on default port 50070.



Overview 'localhost:9000' (active)

Started:	Sat Nov 23 01:17:06 -0500 2019
Version:	2.9.2, r826afbeae31ca687bc2f8471dc841b66ed2c6704
Compiled:	Tue Nov 13 07:42:00 -0500 2018 by ajisaka from branch-2.9.2
Cluster ID:	CID-cc0c33be-5c41-4823-86e9-9be710e6980b
Block Pool ID:	BP-118979042-127.0.1.1-1574429444880

Summary

Security is off.

Safemode is off.

48 files and directories, 11 blocks = 59 total filesystem object(s).

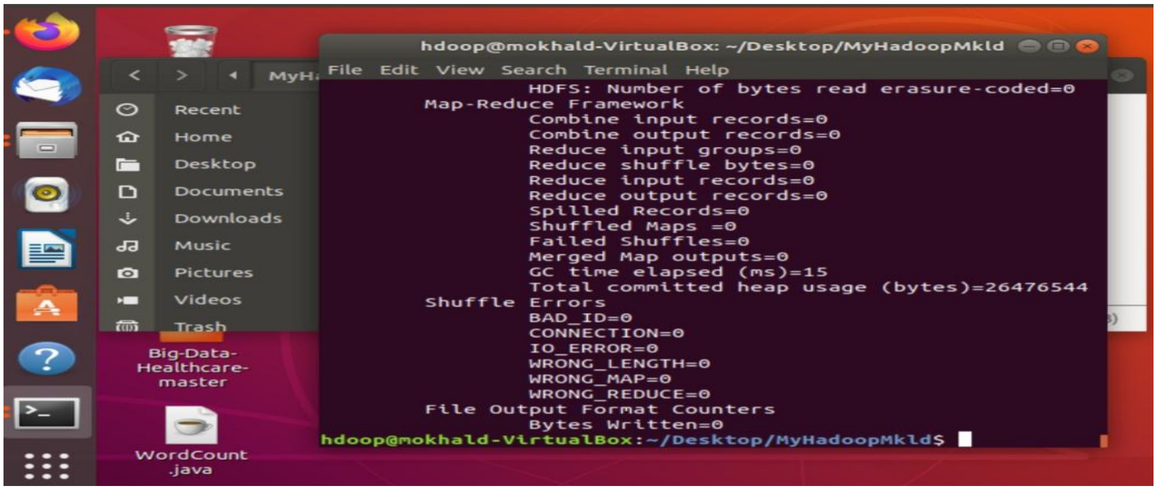
Heap Memory used 43.08 MB of 194 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 57.92 MB of 58.86 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	454.52 GB
DFS Used:	140.59 MB (0.03%)
Non DFS Used:	7.04 GB
DFS Remaining:	424.23 GB (93.34%)

Figure 3.16: NameNode on Default Port.

5- When writing the command (start) will display all nodes on Hadoop master nodes contains two or more DataNodes in a distributed Hadoop environment. Both Hadoop Single Node Cluster (Pseudo Distributed Mode) and Hadoop Multi-Node Cluster (Fully Distributed Mode) are applied. The Figure (3.17) shows the work of other nodes (slave1, slave2).



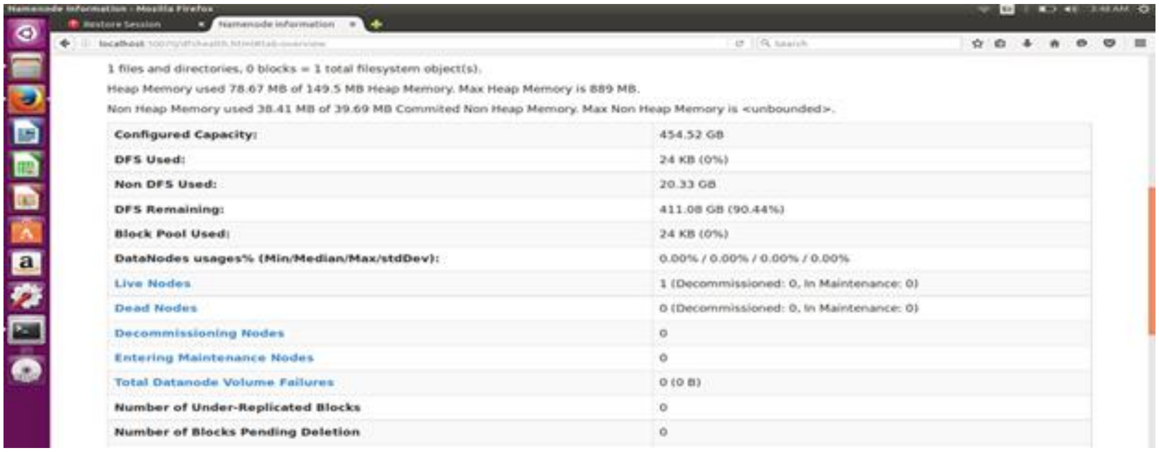
hdoop@mokhald-VirtualBox: ~/Desktop/MyHadoopMkld
File Edit View Search Terminal Help
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Big-Data-Healthcare-master
WordCount.java

```
                HDFS: Number of bytes read erasure-coded=0
        Map-Reduce Framework
                Combine input records=0
                Combine output records=0
                Reduce input groups=0
                Reduce shuffle bytes=0
                Reduce input records=0
                Reduce output records=0
                Spilled Records=0
                Shuffled Maps =0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=15
                Total committed heap usage (bytes)=26476544
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Output Format Counters
                Bytes Written=0
hdoop@mokhald-VirtualBox:~/Desktop/MyHadoopMkld$
```

Figure 3.17: Running of the Hadoop Single and Multi-Cluster Nodes

NameNode only packs the metadata of HDFS – the directory tree of whole files in the file system, and tracks the files across the cluster.

NameNode doesn't save the real data or the dataset. After configuration is finished, NameNode on ports is checked. The Figure (3.18) shows NameNode.


Namenode information - Mozilla Firefox
Restore Session
Namenode information
localhost:50070/dfshealth.html#tab-overview
Search

1 files and directories, 0 blocks = 1 total filesystem object(s).

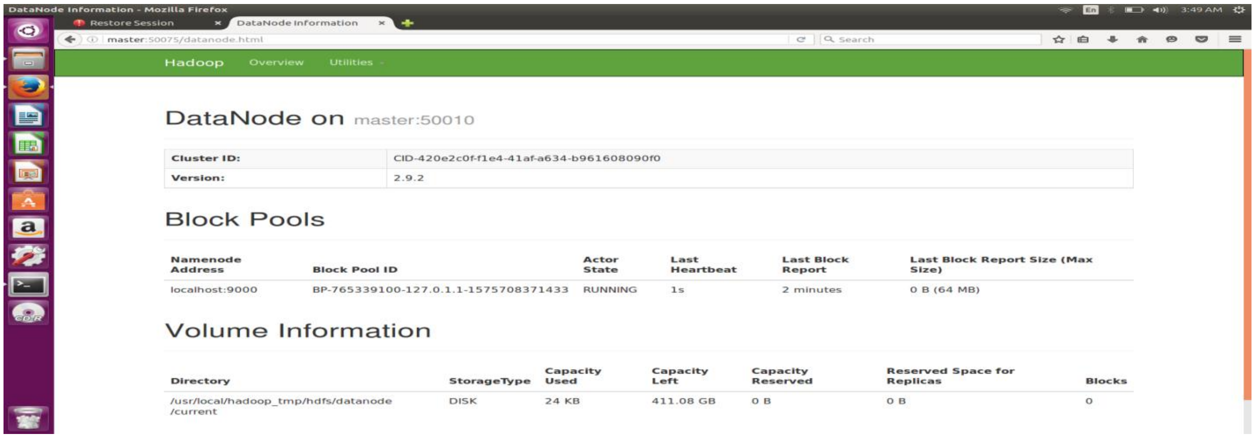
Heap Memory used 78.67 MB of 149.5 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 38.41 MB of 39.69 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	454.52 GB
DFS Used:	24 KB (0%)
Non DFS Used:	20.33 GB
DFS Remaining:	411.08 GB (90.44%)
Block Pool Used:	24 KB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	1 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	0
Number of Blocks Pending Deletion	0

Figure 3.18: the Running of NameNode on Localhost.

6- DataNode is also called a Slave node. In Hadoop HDFS Architecture, DataNode saves real data in HDFS. DataNodes sends information to the NameNode about the files and blocks packed in that node and responds to the NameNode for whole filesystem procedures. The Figure below shows the checking of DataNode on ports.the Figure (3.19) shows DataNode



DataNode on master:50010

Cluster ID:	CID-420e2c0f-f1e4-41af-a634-b961608090f0
Version:	2.9.2

Block Pools

Namenode Address	Block Pool ID	Actor State	Last Heartbeat	Last Block Report	Last Block Report Size (Max Size)
localhost:9000	BP-765339100-127.0.1.1-1575708371433	RUNNING	1s	2 minutes	0 B (64 MB)

Volume Information

Directory	StorageType	Capacity Used	Capacity Left	Capacity Reserved	Reserved Space for Replicas	Blocks
/usr/local/hadoop_tmp/hdfs/datanode /current	DISK	24 KB	411.08 GB	0 B	0 B	0

Figure 3.19: DataNode on Localhost.

Chapter Four

The Implementation، Results and Discussion

4.1 Overview

This chapter discusses the results for the proposed system that are described in chapter three. The proposed system implemented by study three cases where the first one with data mining (data preprocessing) and machine learning classifiers (DT, SVM, and RF)**,** while in the second case study based on the Hadoop /count case study and evaluated with machine learning as (NB, SVM, and DT), while third case study is the Hadoop/Weight-TF and the machine learning algorithms as (NB, SVM, and DT).

4.2 The proposed System Implementation

The proposed system based on three case study with machine learning as showed in Figure (4.1).

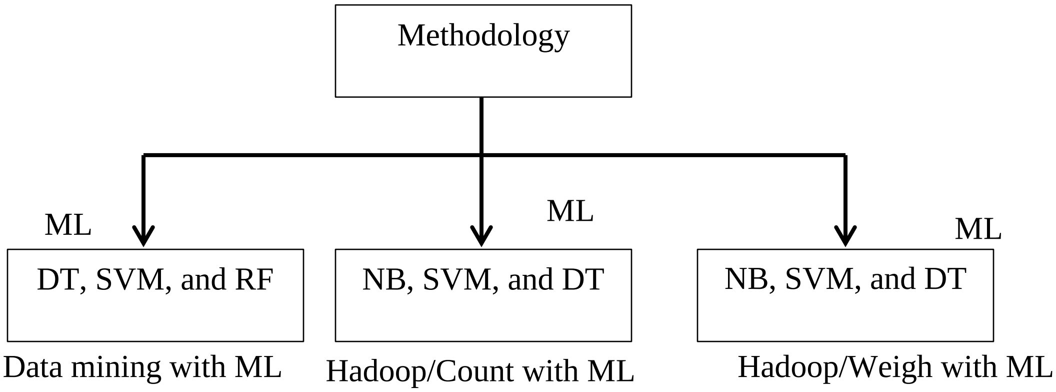


Figure 4.1: The used Machine Learning Algorithms.

The proposed system has been implemented in an environment with the following specifications showed in Table (4.1). Besides, The code of proposed the system has been written in java programming language and the Hadoop environment. Moreover, The Healthcare Stroke Dataset was used in this thesis.

Table 4.1: Environment specifications for the proposed system.

Operating Systems	Windows 7
CPU	Core (TM) I7-3630
RAM	8.00 GB , 8.00 GB
Implementation Tools	Java, Eclipse IDE for Java EE Developers Luna SR2 v4.9 , Eclipse-Hadoop
Operating Systems	Linux
CPU	Core (TM) I7-3630
RAM	8.00 GB , 8.00 GB
Implementation Tools	Hadoop 3.2.1

While Table (4.2) presents the specifications of each compute node, IP Address and the count of nodes that represents master and slave that Apache Hadoop multi node cluster. Table (4.2) shows characteristics for each node in the Hadoop.

Table 4.2: The Specification of the Hadoop Compute Node

Node IP	**The behavior**	**The node CPU**	**RAM**	**OS type**
192.168.0.100	Master (NameNode)	Core i7	8 GB	64-bit
192.168.0.101	Slave 1 (DataNode)	Core i7	8 GB	64-bit
192.168.0.102	Slave 2 (DataNode)	Core i7	8 GB	64-bit

4.2.1 The results of Data Mining/Machine learning for Big Data Analysis (Case 1)

The proposed system is based in the 1st case study on Data mining (Preprocessing) which is evaluated with machine learning classifiers with the maximum accuracy and minimum time required to build the system. The results show the Decision Tree (DT), Support Vector Machine (SVM), and Random Forest (RF) algorithms are the best classifiers in the first case study.

Besides, Table (4.3) and Table (4.4) show the attribute statistics of the main dataset fields before Hadoop MapReduce as the data mining (preprocessing case study).

Table 4.3: Dataset statistics before Hadoop processes.

Attributes	Min	Max	Mean	StdDev
gender	0	2	0.592	0.492
age	0	1	0.514	0.275
hypertension	0	1	0.094	0.291
heart_disease	0	1	0.048	0.213
ever_married	0	1	0.644	0.479
work_type	0	4	1.551	1.263
Residence_type	0	1	0.501	0.5
smoking_status	0	3	1.169	1.027
stroke	0	1	0.018	0.133

Table 4.4: Number of records and attributed of big Healthcare dataset.

Healthcare dataset features	
Number of attributes	9
Sum of Weights (records)	43400

Besides, the main accuracy details of the proposed 1st case study using Data mining / machine learning on the big healthcare dataset without Hdoop show DT is the high accuracy as 98.5407 %, and time to build model is 1091 ms, while SVM is the same accuracy of DT but the time take to build model is 2459 ms. Besides, RF is the low result accuracy as 98.1183 %, and time take to build model is 3615 ms.

Table (4.5) shows the accuracy and time details with the confusion matrix evaluated parameters as False Positive Rate and False Negative Rate of Stroke data case based on the DT, SVM, and RF.

Table 4.5: The results of machine Learning for Big Data Analysis Case Study.

Item	**Method Name**	**Accuracy**	**Confusion Matrix**		**Time**
			False Positive Rate	**False Negative Rate**	
1	Decision Tree	98.5407 %	38	0	1091 ms
2	Support Vector Machine	98.5407 %	38	0	2459 ms
3	Random Forest	98.1183 %	37	12	3615 ms

Table (4.6) presents the results of correctly classified with incorrectly classified instances of the proposed data mining (preprocessing) with the high accurate machine learning algorithms (Decision Tree, Support Vector Machine , Random Forest) on the testing dataset used (Big Healthcare Dataset).

Table 4.6 : Correctly / Incorrectly Classified Testing Instances of the data Preprocessing (1^{st} Case Study).

Machine learning algorithm	**Correctly Classified**	**Incorrectly Classified**
Decision Tree	12830 = 98.54 %	190.99= 1.46 %
Support Vector Machine	12830 = 98.54 %	190.99= 1.46 %
Random Forest	12775 = 98.12 %	245= 1.88 %
Total Number of Testing Instances	13020	
Number of data attributes	9	

Furthermore, the evaluation criteria used in 1st of the the proposed system as Mean Absolute Error (MAE), Root Mean Squared Error(RMSE), and Error Rate showed in Table (4.7) and Figure (4.2) show the prediction of the evaluation criteria of the proposed algorithms, the SVM as 0.0146 almost lower the MAE value, so it is the better compared with others. DT results of the RMSE statistic is the lower as the better 0.1199 compared with other algorithms. DT and SVM results of error rate was better for theses algorithms compared with RF algorithms.

Table 4.7: MAE and RMSE for the big data mining/machine learning 1st *Case Study.*

Evaluation Criteria	Predication		
	DT	SVM	RF
Mean Absolute Error	0.0308	0.0146	0.0301
Root Mean Squared Error	0.1199	0.1208	0.132
Error Rate	0.0145	0.0145	0.0188

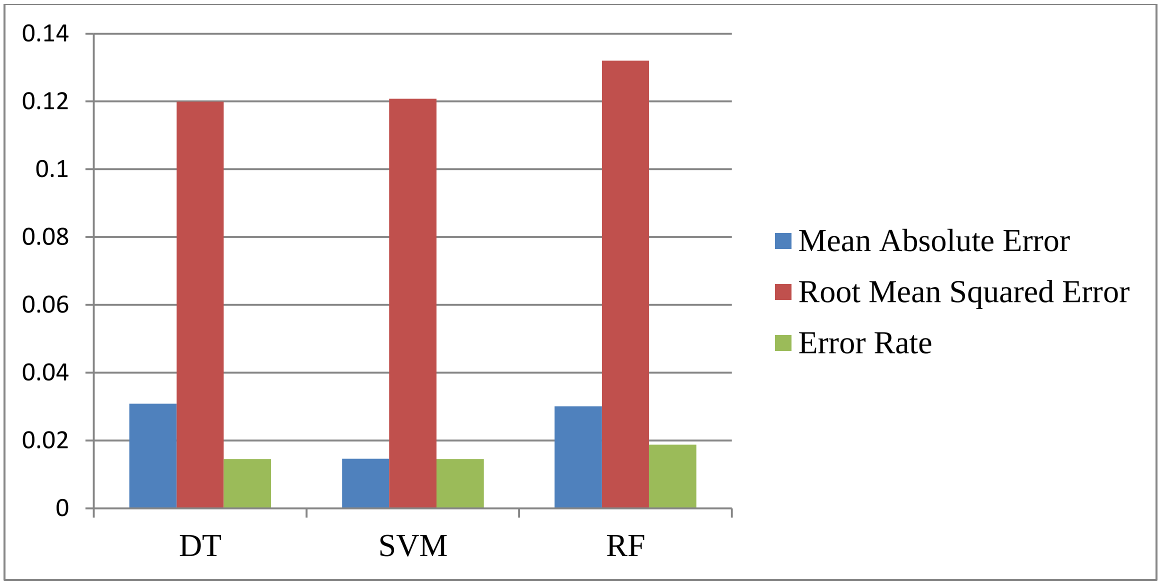


Figure 4.2 : The main evaluation Parameters MAE, RMSE for the big data Preprocessing Analysis Case

Besides, there are other evaluation classifiers as in Table (4.8) of the proposed system based on the three machine learning classifiers and they are implemented for both normal case without stroke(class 0) and with stroke(class 1) as shown in Figure (4.3) of stroke case based on confusion matrix values. RF precision as 0.98571 can be seen as a measure of high quality to return more relevant results than irrelevant ones, and recall as a measure of quantity.

DT and SVM high recall as 100% mean that the algorithms return most of the relevant results. Alongside, DT and SVM as 0.99264 have high F-measure, so they are the better compared with others of this parameter. The better results of Detection Rate (DR), and False Alert Rate (FAR) with high value so the DT and SVM are better as 100% achieved for DR and FAR evaluation parameters.

Table 4.8: Evaluation Details of the presence of stroke disease of the machine learning for big data analysis.

Evaluation Parameters	**Machine Learning Algorithms**		
	DT	**SVM**	**RF**
Precision	0.98540	0.98540	0.98571
Recall	1.0	1.0	0.99532
F-Measure	0.99264	0.99264	0.99049
Detection Rate (DR)	1	1	0.9953
False Alert Rate (FAR)	1	1	0.9736
TP Rate	2566	2566	2554
TN Rate	0	0	1

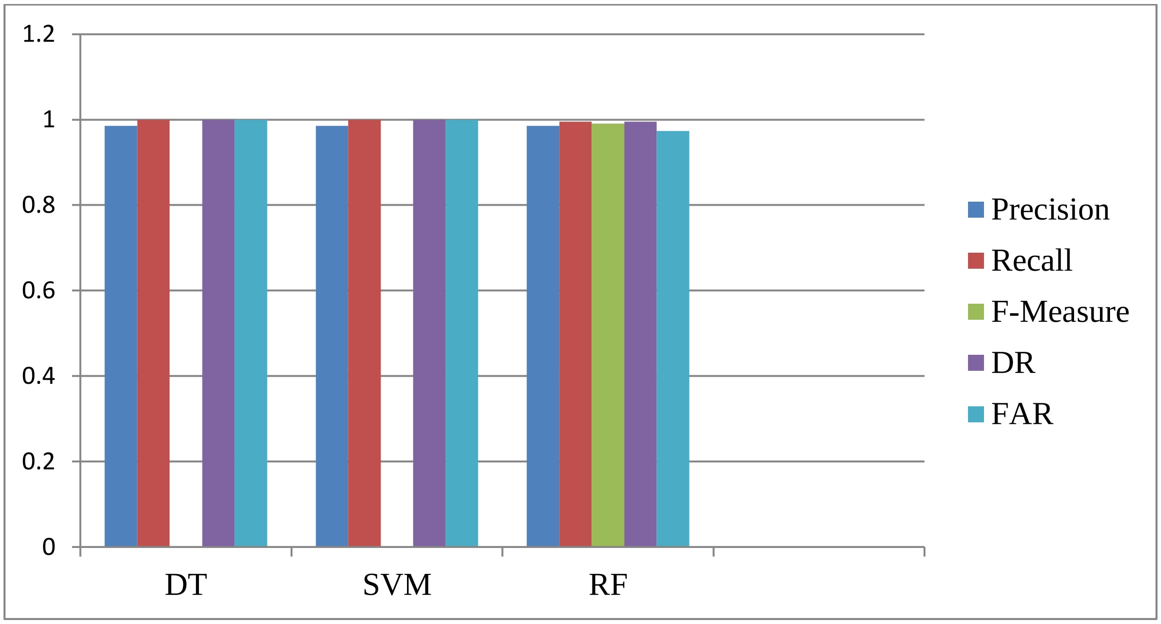


Figure 4.3: Precision, Recall, F-Measure, DR and FAR of the Data Mining/machine learning 1st Case Study.

4.2.2 The results of Hadoop/count for big data analysis (Case 2)

The proposed system implemented in 2nd case study based on the Hadoop/count approach to decease number of records and decease time to build model, with take into consideration increase accuracy at the same time. The used database analysis with Hadoop/MapReduce and the Table (4.9) and Table (4.10) show the attribute statistics of the main dataset fields with Hadoop/MapReduce.

The min value represents by the minimum value of the attributes in the dataset, Max value represents by the maximum value of the attribute while the Mean value represents as the average or the summation of all attribute values divided by the total number of records, besides the StdDev value represents standard deviation value based on the standard equation based on values and Mean.

Table 4.9: Dataset Statistics after Hadoop Processes .

Attributes	Min	Max	Mean	StdDev
gender	0	2	0.579	0.494
age	0	1	0.533	0.266
hypertension	0	1	0.095	0.295
heart_disease	0	1	0.05	0.218
ever_married	0	1	0.674	0.469
work_type	0	4	1.612	1.254
Residence_type	0	1	0.49	0.5
smoking_status	0	3	1.202	1.025
stroke	0	1	0.017	0.128
Proposed/Count	**1**	**143**	**19.41**	**23.681**
Proposed/Weight	**0**	**0.077**	**0.016**	**0.017**

Table 4.10: Number of records and attributed of Healthcare dataset after Hadoop.

Healthcare dataset features	
Number of attributes	10
Sum of Weights	8618

Besides, the main accuracy details of the proposed 2nd case study using Hadoop/ Count and machine learning on the big healthcare dataset. The results show NB, SVM, and DT is the high accuracy as 98.646 %, and time to build model is 63 ms, 313 ms, and 520 ms respectively.

Table (4.11) shows the accuracy and time details of 2nd case study with the confusion matrix evaluated parameters as False Positive Rate and False Negative Rate of Stroke data case based on the NB, SVM, and DT.

Table 4.11: The results of machine learning for big data analysis and Hadoop/Count.

Item	Method name	Accuracy	Confusion Matrix		Time
			False positive Rate	False Negative Rate	
1	Naïve Bays	98.646 %	7	0	63 ms
2	Support Vector Machine	98.646 %	7	0	313 ms
3	Decision Tree	98.646 %	7	0	520 ms

While Table (4.12) shows the results of correctly classified with incorrectly classified Instances of the proposed Hadoop/count with the high accurate machine learning algorithms (NB, SVM , DT) on the testing dataset used (Big Healthcare Dataset).

Table 4.12 : Correctly / Incorrectly Classified Testing Instances of the Hadoop/ Count (2nd Case Study).

Machine learning algorithm	Correctly Classified	Incorrectly Classified
Naïve Bays	2550 = 98.65 %	35.998 = 1.35 %
Support Vector Machine	2550 = 98.65 %	34.998 = 1.35 %
Decision Tree	2550 = 98.645 %	34.998 = 1.35 %
Total Number of Testing Instances	2585	
Number of data attributes	10	

Moreover, the evaluation criteria used in 2nd case study of the proposed system showed in Table (4.13) and Figure (4.4) show the prediction of the evaluation criteria of the proposed algorithms, the SVM as 0.0135 is almost lower the MAE value, so it is the better compared with others. DT results of the RMSE statistic is the lower as the better about 0.1157 compared with other algorithms. DT and SVM results of error rate was better for theses algorithms compared with NB algorithms.

Table 4.13: MAE and RMSE for the big data analysis and Hadoop/Count.

Evaluation Criteria	Predication		
	NB	SVM	DT
Mean Absolute Error	0.0412	0.0135	0.0309
Root Mean Squared Error	0.1209	0.1164	0.1157
Error Rate	0.01353	0.01350	0.01350

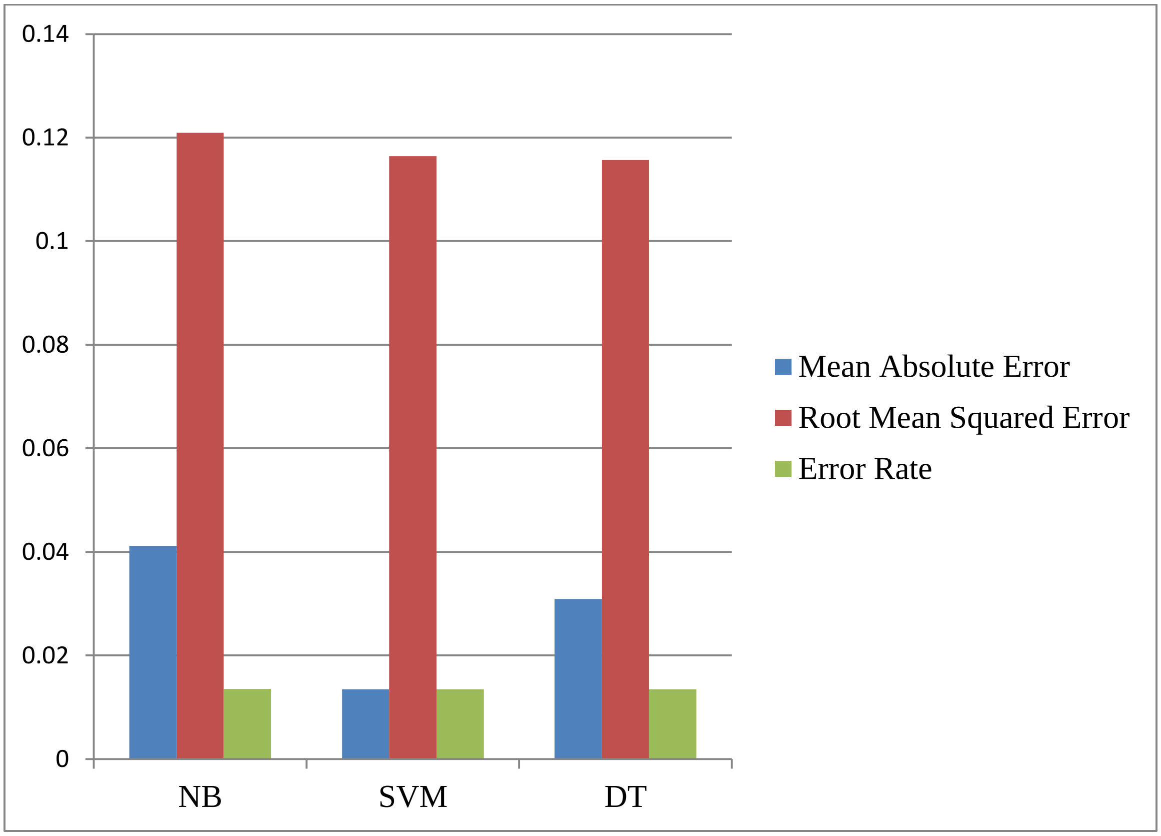


Figure 4.4 : The main evaluation Parameters MAE, RMSE and RAE for the big data analysis and Hadoop/Count 2nd Case Study.

Besides, in Table (4.14) of the proposed system based on the three machine learning classifiers(NB, SVM, DT) and they are implemented for both normal case without stroke(class 0) and with stroke(class 1) as shown in Figure (4.5) of stroke case based on confusion matrix values. NB,SVM, and DT precision as 0.98646 are the high quality algorithms for relevant result. Besides, NB,SVM, and DT as 100% high recall results. Alongside, NB,SVM, and DT have high F-measure as the better results.

The better results of Detection Rate (DR), and False Alert Rate (FAR) with high value so the NB,SVM, and DT are better as 100% achieved for DR and FAR evaluation parameters.

Table 4.14: Evaluation Details of the presence of stroke disease of the machine learning for big data analysis and Hadoop/Count.

Evaluation Parameters	**Machine Learning Algorithms**		
	NB	**SVM**	**DT**
Precision	0.98646	0.98646	0.98646
Recall	1.0	1.0	1.0
F-Measure	0.99318	0.99318	0.99318
Detection Rate (DR)	1	1	1
False Alert Rate (FAR)	1	1	1
TP Rate	510	510	510
TN Rate	0	0	0

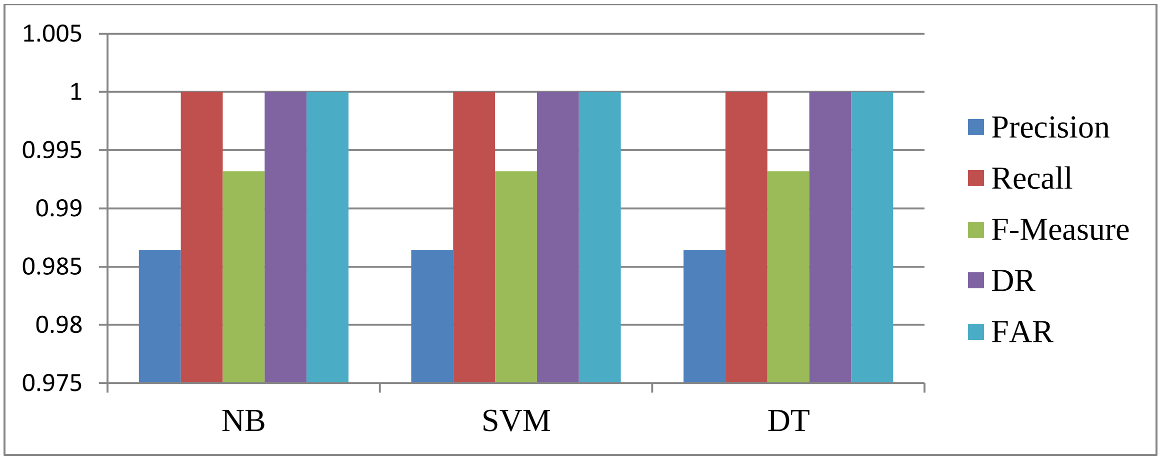


Figure 4.5: Precision, Recall, F-Measure, DR and FAR of Hadoop/count 2^{nd} Case Study.

4.2.3 The results of Hadoop/Weight for Dig Data Analysis (Case 3)

The proposed system was implemented in the 3rd case study is based on the Hadoop/Weight approach to enhance accuracy and decrease the time required to build the model. Table (5.15) showed the accuracy and time takes to build system model for each machine learning classifiers with the maximum accuracy and minimum time values.

Table 5.15: The results of machine learning for big data analysis and Hadoop/Weight

Item	Method name	Accuracy	Confusion Matrix		Time
			False positive Rate	False Negative Rate	
1	Naïve Bays	98.646 %	7	0	28 ms
2	Support Vector Machine	98.646 %	7	0	371 ms
3	Decision Tree	98.646 %	7	0	473 ms

While Table (4.16) shows the results of correctly classified with incorrectly classified instances of the proposed Hadoop/Weight with the high accurate machine learning algorithms (NB, SVM , DT) on the testing dataset used (Big Healthcare Dataset).

Table 4.16 : Correctly / Incorrectly Classified Testing Instances of the data Hadoop/Weight (3rd Case Study).

Machine learning algorithm	Correctly Classified	Incorrectly Classified
Naïve Bays	2550= 98.6460 %	34.998 = 1.35 %
Support Vector Machine	2550= 98.6460 %	34.998 = 1.35 %
Decision Tree	2550 = 98.6460 %	34.998 = 1.35 %
Total Number of Testing Instances	2585	
Number of data attributes	10	

Furthermore, the evaluation criteria used in 2nd case study of the proposed system showed in Table 4.17 and Figure 4.5 show the prediction of the evaluation criteria of the proposed algorithms, the SVM as 0.0135 is almost lower the MAE value, so it is the better compared with others. DT results of the RMSE statistic is the lower as the better about 0.1157 compared with other algorithms. Error rate was the same for all used algorithms as NB, SVM, and DT.

Table 4.17: MAE, RMSE, and Error Rate for the big data analysis and Hadoop/Weight

Evaluation Criteria	Predication		
	NB	SVM	DT
Mean Absolute Error	0.0448	0.0135	0.0309
Root Mean Squared Error	0.1233	0.1164	0.1157
Error Rate	0.0135	0.0135	0.0135

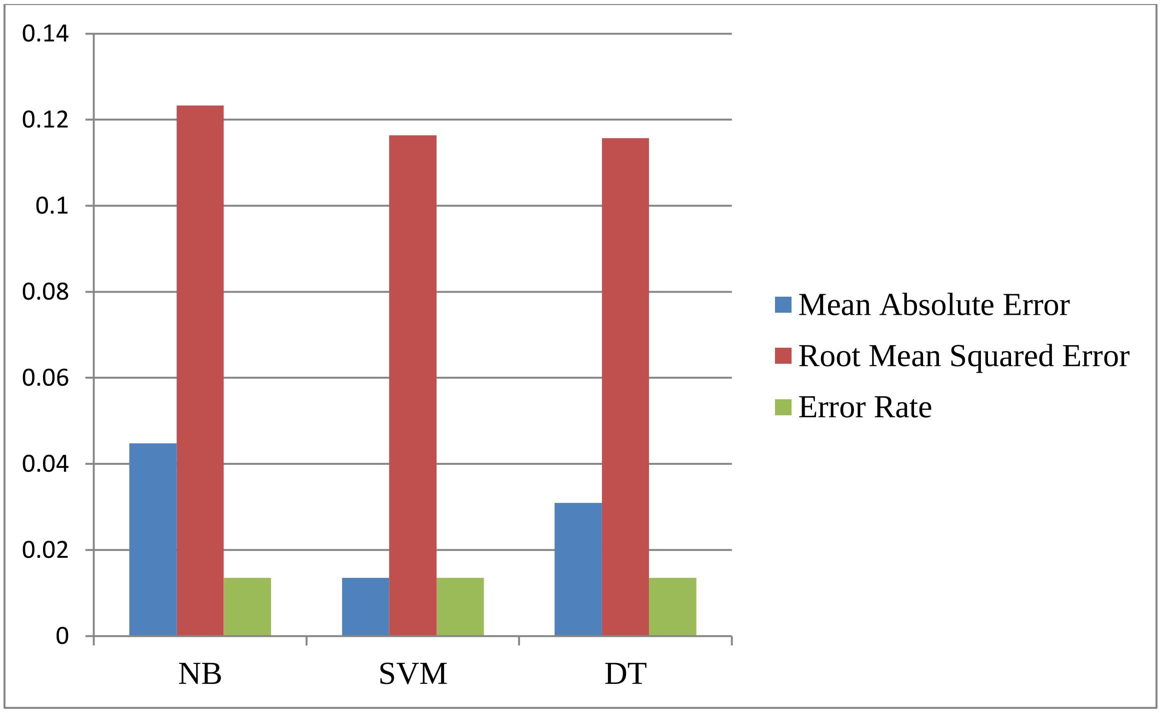


Figure 4.6: The main evaluation Parameters MAE, RMSE and Error Rate for the big data analysis and Hadoop/Weight 3rd Case Study

Besides, in Table (4.18) and Figure (4.6) of the proposed system of Hadoop/weight with machine learning algorithms as NB,SVM, and DT, the precision as 0.98646 are the high quality algorithms for all the used algorithms. Besides, NB,SVM, and DT as 100% high recall results. Alongside, NB,SVM, and DT have high F-measure as the better results. The better results of Detection Rate (DR), and False Alert Rate (FAR) with high value so the NB,SVM, and DT are better as 100% achieved for DR and FAR evaluation parameters.

Table 4.18 : Evaluation Details of the presence of stroke disease of the machine learning for big data analysis and Hadoop/Weight

Evaluation Parameters	**Machine Learning Algorithms**		
	NB	**SVM**	**DT**
Precision	0.98646	0.98646	0.98646
Recall	1.0	1.0	1.0
F-Measure	0.99318	0.99318	0.99318
Detection Rate (DR)	1	1	1
False Alert Rate (FAR)	1	1	1
TP Rate	510	510	510
TN Rate	0	0	0

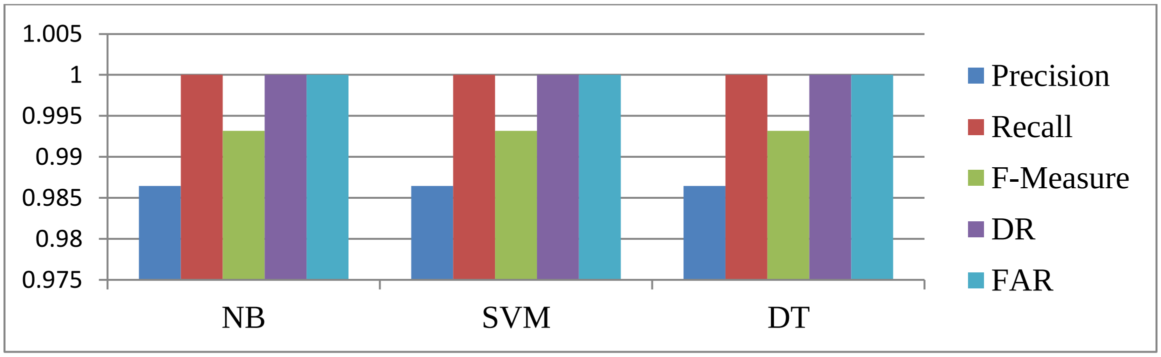


Figure 4.7: Precision, Recall, F-Measure, DR and FAR of the Hadoop–Weight Case Study.

4.3 System Comparison

The proposed system is compared based on the three case studies and they are evaluated based on the common evaluation parameters on this side and Table 4.19 showed the time comparison. The better result (minimum building time) from the 1st case study is as DT is 1091 ms and SVM is 2459 ms, while the low results (Long Time) from RF is 3615 ms. Besides, the better result from the 2nd case study as NB is 63 ms and SVM is 313 ms, while the low results (Long Time) from DT is 520 ms. Alongside, the better result from the 3rd case study as NB is 28 ms and SVM is 371 ms, while the low results (Long Time) from DT is 473 ms. The better result of the time evaluation parameter for all machine learning classifiers in the proposed system is in the NB as 28 ms.

Table 4.19 : Time Taken to build Model Comparisons.

Time Taken to build Model in Millisecond (ms)	**Data Mining / Preprocessing 1st Case Study**		
	DT	**SVM**	**RF**
	1091 ms	2459 ms	3615 ms
	Hadoop/Count 2nd Case Study		
	NB	**SVM**	**DT**
	63 ms	313 ms	520 ms
	Hadoop/Weight 3rd Case Study		
	NB	**SVM**	**DT**
	28 ms	371 ms	473 ms

While the accuracy results of the data mining (preprocessing), Hadoop/count and Hadoop/Weight showed in Table 4.20.

Table 4.20 :Accuracy Details of Normal Data of (without Data Preprocessing).

Data Mining / Preprocessing 1st Case Study			
Accuracy Details of Stroke	**Machine Learning Algorithms**		
	DT	**SVM**	**RF**
Accuracy	**98.5407 %**	**98.5407 %**	**98.1183 %**
Hadoop/Count 2nd Case Study			
Accuracy Details of Stroke	**Machine Learning Algorithms**		
	NB	**SVM**	**DT**
Accuracy	**98.646 %**	**98.646 %**	**98.646 %**
Hadoop/Weight 3rd Case Study			
Accuracy Details of Stroke	**Machine Learning Algorithms**		
	NB	**SVM**	**DT**
Accuracy	**98.646 %**	**98.646 %**	**98.646 %**

Besides, the system comparison with the other related works showed in the Table (4.21). The better accuracy result of the proposed system of all case studies are showed in the case of Haoop/Count and Hadoop/weight of NB, SVM, and DT as 98.646 %.

Table 4.21 :The results of Big Healthcare Dataset Analysis with the compared systems.

Ref.No	**Year**	**Method Name**	**Accuracy**	**Time**
[13]	2019	DNN	83%	/
[21]	2020	Support Vector Machine	77 %	/
		Random Forest	90 %	/

		Decision Tree	79 %	/
[23]	2020	Neural Network	95 %	/
		Support Vector Machine	91.5 %	2280 ms
		Random Forest	90.9 %	7240 ms
[24]	2020	Weighted Voting	97 %	/
[27]	2021	Random Forest	97.17%	/
		Decision Tree	94.24%	/
		Naïve Bayes	86.04%	/
Proposed-Preprocessing		**Decision Tree**	**98.5407 %**	**1091 ms**
		Support Vector Machine	**98.5407 %**	**2459 ms**
		Random Forest	**98.1183 %**	**3615 ms**
Hadoop/count		**Naïve Bays**	**98.646 %**	**63 ms**
		Support Vector Machine	**98.646 %**	**313 ms**
		Decision Tree	**98.646 %**	**520 ms**
Hadoop/Weight(TF)		**Naïve Bays**	**98.646 %**	**28 ms**
		Support Vector Machine	**98.646 %**	**371 ms**
		Decision Tree	**98.646 %**	**473 ms**

4.4 **Summary**

A stroke big healthcare dataset has been used in this work and there are three case study are used in this chapter to predict and evaluate results of Stroke disease. The 1st case study is based on the data mining for data preprocessing and then the data evaluated with machine learning for testing data set records. The 2nd case study is based on the Hdoop approach with count feature as Hadoop/Count to minimize number of records in big dataset and decrease resource allocation besides, increase accuracy of stroke prediction and decrease time to build model. The 3rd case study is based on the Hadoop/Weight to maximize accuracy and decrease time to build as the optimal case study with Hadoop approach. In addition, this chapter explained the system comparison with other related works in the same side.

Chapter Five

Conclusions and Suggestions for Future Works

5.1 Conclusions

This chapter explains the proposed system conclusions and the main suggestions for future works they can be summarized as :

1- The implemented system is based on three case study as the 1st is preprocessing data mining / machine learning, the 2nd case study is Hadoop/Count and the 3rd case study is Hadoop/Weight. Besides, it is based on the three machine learning for each case study to predict stroke in big healthcare dataset.

2- The study is aimed to address the issues of computation parameters exhausted by machine learning through processing big dataset.

3- The performance evaluation reveals that Hadoop count / Hadoop–Weight provided the highest accuracy of about 98.646 % of Naïve Bays, Support Vector Machine, Decision Tree with decreased number of instances of the dataset from 43400 records into 2585 records and decreased time to build system with Hadoop processes compared to other research directions based on machine learning algorithms.

4- The better the performance of the model at distinguishing between the positive and negative classes of the presence of stroke disease (Class 1) as the proportion of the model's predictions as DR of whether a class is a Stroke or not showed the high results from preprocessing case study about 100% of SVM, DT, and low DR of RF as 0.9953. Alongside, the proposed Hadoop/Count with 100% for NB, SVM, and DT Besides, the high DR of Hadoop/Weight is about 100% of NB, SVM, and DT So, if we can maintain this disease from an early stage then it will help to reduce stoke in our life.

5- Implementing the hadoop Mapreduce approach to decrease number of records in the big healthcare stroke dataset.

6- The results show the proposed model succeeded in predicting stroke with high accuracy of the Hadoop/weight of NB algorithm as 98.646 %.

7- The proposed system showed better time to build model in the case of NB as 28 ms in the Hadoop/Weight case study.

5.2 Suggestions for Future Works

There are numerous considerations can be realized for future expansion of present research through utilizing the following propositions:

1- Implementing online Apache Spark as big data platform based on frequent Item sets concept to extract the interesting rules in the Dataset.

2- Implementing the clustering algorithms for instance (K-means clustering algorithm) in case of without class appearance in dataset such as clustering algorithms for IoT data analysis.

3- Evaluating the proposed system based imaging, such as brain CT scan and MRI, together with an existing model that will boost performance indices.

4- Measuring and collecting real-time bio-signals in driving or sleeping services as well as walking during everyday life, along with conducting research and development on the provision of more comprehensive forecasting services for stroke diseases.

REFERENCES

[1] Kumar, S., & Singh, M. (2018). Big data analytics for healthcare industry: impact, applications, and tools. *Big data mining and analytics*, *2*(1), 48-57.

[2] Ahmed, N., Barczak, A. L., Susnjak, T., & Rashid, M. A. (2020). A comprehensive performance analysis of Apache Hadoop and Apache Spark for large scale data sets using HiBench. *Journal of Big Data*, *7*(1), 1-18.

[3] Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Khan, S. U. (2015). The rise of "big data" on cloud computing: Review and open research issues. *Information systems*, *47*, 98-115.

[4] Gavhane, A., Kokkula, G., Pandya, I., & Devadkar, K. (2018, March). Prediction of heart disease using machine learning. In *2018 second international conference on electronics, communication and aerospace technology (ICECA)* (pp. 1275-1278). IEEE.

[5] Li, X., Bian, D., Yu, J., Li, M., & Zhao, D. (2019). Using machine learning models to improve stroke risk level classification methods of China national stroke screening. *BMC medical informatics and decision making*, *19*(1), 1-7.

[6] Gravesteijn, B. Y., Nieboer, D., Ercole, A., Lingsma, H. F., Nelson, D., Van Calster, B., ... & Puybasset, L. (2020). Machine learning algorithms performed no better than regression models for prognostication in traumatic brain injury. *Journal of clinical epidemiology*, *122*, 95-107.

[7] Chicco, D., & Jurman, G. (2020). Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. *BMC medical informatics and decision making*, *20*(1), 1-16.

[8] Gupta, D. L., Malviya, A. K., & Singh, S. (2012). Performance analysis of classification tree learning algorithms. *International Journal of Computer Applications*, *55*(6).

[9] Karaca, Y., Moonis, M., & Baleanu, D. (2020). Fractal and multifractional-based predictive optimization model for stroke subtypes' classification. *Chaos, Solitons & Fractals*, *136*, 109820.

[10] Wang, Y., Nazir, S., & Shafiq, M. (2021). An overview on analyzing deep learning and transfer learning approaches for health monitoring. *Computational and Mathematical Methods in Medicine*, *2021*.

[11] Li, X., Bian, D., Yu, J., Li, M., & Zhao, D. (2019). Using machine learning models to improve stroke risk level classification methods of China national stroke screening. *BMC medical informatics and decision making*, *19*(1), 1-7.

[12] Ali, A. A. (2019). Stroke Prediction using Distributed Machine Learning Based on Apache Spark. *Stroke*, *28*(15), 89-97.

[13] Cheon, S., Kim, J., & Lim, J. (2019). The use of deep learning to predict stroke patient mortality. *International journal of environmental research and public health*, *16*(11), 1876.

[14] Karthik, R., Menaka, R., Johnson, A., & Anand, S. (2020). Neuroimaging and deep learning for brain stroke detection-A review of recent advancements and future prospects. *Computer Methods and Programs in Biomedicine*, 105728.

[15] Stančin, I., & Jović, A. (2020, May). An overview and comparison of free Python libraries for data mining and big data analysis. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (pp. 977-982). IEEE.

[16] Alotaibi, S. R. (2020). Applications of artificial intelligence and big data analytics in m-health: a healthcare system perspective. *Journal of Healthcare Engineering*, *2020*.

[17] Sivaparthipan, C. B., Karthikeyan, N., & Karthik, S. (2020). Designing statistical assessment healthcare information system for diabetics analysis using big data. Multimedia Tools and Applications, 79(13), 8431-8444.

[18] Uyanık, T., Karatuğ, Ç., & Arslanoğlu, Y. (2020). Machine learning approach to ship fuel consumption: A case of container vessel. *Transportation Research Part D: Transport and Environment*, *84*, 102389.

[19] Sampurnima, P., Satapathy, S. K., Mishra, S., & Mallick, P. K. (2019, December). Hyperspectral Image Classification Using Stochastic Gradient Descent Based Support Vector Machine. In *International Conference on Biologically Inspired Techniques in Many-Criteria Decision Making* (pp. 78-84). Springer, Cham.

[20] Liu, Y., Li, X., Chen, X., Wang, X., & Li, H. (2020). High-performance machine learning for large-scale data classification considering class imbalance. *Scientific Programming*, *2020*.

[21] Ali, A. A. (2019). Stroke Prediction using Distributed Machine Learning Based on Apache Spark. *Stroke*, *28*(15), 89-97.

[22] Tozlu, C., Edwards, D., Boes, A., Labar, D., Tsagaris, K. Z., Silverstein, J., ... & Kuceyeski, A. (2020). Machine learning methods predict individual upper-limb motor impairment following therapy in chronic stroke. *Neurorehabilitation and neural repair*, *34*(5), 428-439.

[23] Govindarajan, P., Soundarapandian, R. K., Gandomi, A. H., Patan, R., Jayaraman, P., & Manikandan, R. (2020). Classification of stroke disease using machine learning algorithms. *Neural Computing and Applications*, *32*(3), 817-828.

[24] Emon, M. U., Keya, M. S., Meghla, T. I., Rahman, M. M., Al Mamun, M. S., & Kaiser, M. S. (2020, November). Performance Analysis of Machine Learning Approaches in Stroke Prediction. In *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)* (pp. 1464-1469). IEEE.

[25] Bakırarar, B., Yüksel, C., & Yavuz, Y. (2021). Prediction of new prescription requirements for diabetes patients using big data technologies. *Journal of Health Research*.

[26] Gangavarapu. S, Gorli L A. Kumari, Analyzing the Performance of Stroke Prediction using ML Classification Algorithms. (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 12, No. 6, 2021

[27] Harish, C. S., Vamsi, G. K., Akhil, G. J. P., Sravan, J. H., & Chowdary, V. M. Prediction of Heart Stroke using A Novel Framework–PySpark, 2021.

[28] Siddique, S., & Chow, J. C. (2021). Machine learning in healthcare communication. *Encyclopedia*, *1*(1), 220-239.

[29] Lip, G. Y., Genaidy, A., Tran, G., Marroquin, P., Estes, C., & Sloop, S. (2021). Improving stroke risk prediction in the general population: Common clinical rules, a new multimorbid index and machine learning based algorithms. *Thrombosis and haemostasis*, (AAM).

[30] Choi, Y. A., Park, S., Jun, J. A., Ho, C. M. B., Pyo, C. S., Lee, H., & Yu, J. (2021). Machine-Learning-Based Elderly Stroke Monitoring System Using Electroencephalography Vital Signals. *Applied Sciences*, *11*(4), 1761.

[31] O'Driscoll, A., Daugelaite, J., & Sleator, R. D. (2013). 'Big data', Hadoop and cloud computing in genomics. *Journal of biomedical informatics*, *46*(5), 774-781.

[32] Kaisler, S., Armour, F., Espinosa, J. A., & Money, W. (2013, January). Big data: Issues and challenges moving forward. In *2013 46th Hawaii international conference on system sciences* (pp. 995-1004). IEEE.

[33] Wang, G., Gunasekaran, A., Ngai, E. W., & Papadopoulos, T. (2016). Big data analytics in logistics and supply chain management: Certain investigations for research and applications. *International Journal of Production Economics*, *176*, 98-110.

[34] Kapil, G., Agrawal, A., & Khan, R. A. (2016, October). A study of big data characteristics. In *2016 International Conference on Communication and Electronics Systems (ICCES)* (pp. 1-4). IEEE.

[35] van Altena, A. J., Moerland, P. D., Zwinderman, A. H., & Olabarriaga, S. D. (2016). Understanding big data themes from scientific biomedical literature through topic modeling. *Journal of Big Data*, *3*(1), 1-21.

[36] Al-Mekhlal, M., & Khwaja, A. A. (2019, August). A Synthesis of Big Data Definition and Characteristics. In *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)* (pp. 314-322). IEEE.

[37] Jan, B., Farman, H., Khan, M., Imran, M., Islam, I. U., Ahmad, A., ... & Jeon, G. (2019). Deep learning in big data analytics: a comparative study. *Computers & Electrical Engineering*, *75*, 275-287.

[38] Verma, J. P., Agrawal, S., Patel, B., & Patel, A. (2016). Big data analytics: challenges and applications for text, audio, video, and social media data". *International Journal on Soft Computing, Artificial Intelligence and Applications (IJSCAI)*, *5*(1), 41-51.

[39] Bakdi, M., & Chadli, W. (2021). Big Data: An Overview. *Big Data Analytics*, 3-13.

[40] Rehman, A., Naz, S., & Razzak, I. (2021). Leveraging big data analytics in healthcare enhancement: trends, challenges and opportunities. *Multimedia Systems*, 1-33.

[41] Waller, M. A., & Fawcett, S. E. (2013). Data science, predictive analytics, and big data: a revolution that will transform supply chain design and management.

[42] Ranjan, J., & Foropon, C. (2021). Big data analytics in building the competitive intelligence of organizations. *International Journal of Information Management*, *56*, 102231.

[43] Pacha, N. H., Khebazi, F. Z., & Mazouz, N. (2021). Data Mining and Its Contribution to Decision-Making in Business Organizations. In *Big Data Analytics* (pp. 67-80). Apple Academic Press.

[44] García, S., Luengo, J., & Herrera, F. (2015). *Data preprocessing in data mining* (Vol. 72). Cham, Switzerland: Springer International Publishing.

[45] Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1996, August). Knowledge Discovery and Data Mining: Towards a Unifying Framework. In *KDD* (Vol. 96, pp. 82-88).

[46] Banaee, H., Ahmed, M. U., & Loutfi, A. (2013). Data mining for wearable sensors in health monitoring systems: a review of recent trends and challenges. *Sensors*, *13*(12), 17472-17500.

[47] Weber, B. G., & Mateas, M. (2009, September). A data mining approach to strategy prediction. In *2009 IEEE Symposium on Computational Intelligence and Games* (pp. 140-147). IEEE.

[48] Farhangfar, A., Kurgan, L., & Dy, J. (2008). Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition*, *41*(12), 3692-3705.

[49] Sayed, H., Abdel-Fattah, M. A., & Kholief, S. (2018). Predicting potential banking customer churn using apache spark ML and MLlib packages: a comparative study. *IJACSA) International Journal of Advanced Computer Science and Applications*, *9*, 674-677.

[50] Yan, J. Text Mining with R: A Tidy Approach, by Julia Silge and David Robinson. Sebastopol, CA: O’Reilly Media, 2017. ISBN 978-1-491-98165-8. XI+ 184 pages. *Natural Language Engineering*, 1-3.

[51] Vargas-Solar, G., Zechinelli-Martini, J. L., & Espinosa-Oviedo, J. A. (2017). Big data management: what to keep from the past to face future challenges?. *Data Science and Engineering*, *2*(4), 328-345.

[52] Bansod, A. (2015). Efficient big data analysis with apache spark in HDFS. *International Journal of Engineering and Advanced Technology (IJEAT)*, *4*(6), 313-315.

[53] Honjo, T., & Oikawa, K. (2013, October). Hardware acceleration of hadoop mapreduce. In *2013 IEEE International Conference on Big Data* (pp. 118-124). IEEE.

[54] Garlasu, D., Sandulescu, V., Halcu, I., Neculoiu, G., Grigoriu, O., Marinescu, M., & Marinescu, V. (2013, January). A big data implementation based on Grid computing. In *2013 11th RoEduNet International Conference* (pp. 1-4). IEEE.

[55] Kaushik, R. T., Bhandarkar, M., & Nahrstedt, K. (2010, November). Evaluation and analysis of greenhdfs: A self-adaptive, energy-conserving variant of the hadoop distributed file system. In *2010 IEEE Second International Conference on Cloud Computing Technology and Science* (pp. 274-287). IEEE.

[56] Prabhu, C. S. R., Chivukula, A. S., Mogadala, A., Ghosh, R., & Livingston, L. J. (2019). Big Data Tools—Hadoop Ecosystem, Spark and NoSQL Databases. In *Big Data Analytics: Systems, Algorithms, Applications* (pp. 83-165). Springer, Singapore.

[57] Anderson, J. C., Lehnardt, J., & Slater, N. (2010). *CouchDB: the definitive guide: time to relax*. " O'Reilly Media, Inc.".

[58] Li, B., Mazur, E., Diao, Y., McGregor, A., & Shenoy, P. (2011, June). A platform for scalable one-pass analytics using mapreduce. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data* (pp. 985-996).

[59] Mao, Y., Morris, R., & Kaashoek, F. (2010). *Optimizing MapReduce for multicore architectures*. Technical Report MIT-CSAIL-TR-2010-020, MIT.

[60] Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, *160*(1), 3-24.

[61] Medina-Ortiz, D., Contreras, S., Quiroz, C., & Olivera-Nappa, Á. (2020). Development of supervised learning predictive models for highly non-linear biological, biomedical, and general datasets. *Frontiers in molecular biosciences*, *7*, 13.

[62] Singh, A., Thakur, N., & Sharma, A. (2016, March). A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 1310-1315). Ieee.

[63] Zhang, H., Liu, C. T., Mao, J., Shen, C., Xie, R. L., & Mu, B. (2020). Development of novel in silico prediction model for drug-induced ototoxicity by using naïve Bayes classifier approach. *Toxicology in Vitro*, *65*, 104812.

[64] Chen, S., Webb, G. I., Liu, L., & Ma, X. (2020). A novel selective naïve Bayes algorithm. *Knowledge-Based Systems*, *192*, 105361.

[65] Sahin, E. K., Colkesen, I., & Kavzoglu, T. (2020). A comparative assessment of canonical correlation forest, random forest, rotation forest and logistic regression methods for landslide susceptibility mapping. *Geocarto International*, *35*(4), 341-363.

[66] Zhang, S. (2020). Cost-sensitive KNN classification. *Neurocomputing*, *391*, 234-242.

[67] Yuan, B. W., Luo, X. G., Zhang, Z. L., Yu, Y., Huo, H. W., Johannes, T., & Zou, X. D. (2021). A novel density-based adaptive k nearest neighbor method for dealing with overlapping problem in imbalanced datasets. *Neural Computing and Applications*, *33*(9), 4457-4481.

[68] de Lima, M. D., e Lima, J. D. O. R., & Barbosa, R. M. (2020). Medical data set classification using a new feature selection algorithm combined with twin-bounded support vector machine. *Medical & biological engineering & computing*, *58*(3), 519-528.

[69] Jaini, S. N. B., Lee, D. W., Lee, S. J., Kim, M. R., & Son, G. H. (2021). Indirect tool monitoring in drilling based on gap sensor signal and multilayer perceptron feed forward neural network. *Journal of Intelligent Manufacturing*, *32*(6), 1605-1619.

[70] Jafari, M., Wang, Y., Amiryousefi, A., & Tang, J. (2020). Unsupervised learning and multipartite network models: a promising approach for understanding traditional medicine. *Frontiers in pharmacology*, *11*, 1319.

[71] Yu, F., Wang, D., Chen, Y., Karianakis, N., Shen, T., Yu, P., ... & Chen, X. (2019). Unsupervised domain adaptation for object detection via cross-domain semi-supervised learning. *arXiv preprint arXiv:1911.07158*.

[72] Reddy, T., Bhattacharya, S., Maddikunta, P. K. R., Hakak, S., Khan, W. Z., Bashir, A. K., ... & Tariq, U. (2020). Antlion re-sampling based deep neural network model for classification of imbalanced multimodal stroke dataset. *Multimedia Tools and Applications*, 1-25.

[73] Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.*, *3*(Mar), 1289-1305.

[74] Solhmirzaei, R., Salehi, H., Kodur, V., & Naser, M. Z. (2020). Machine learning framework for predicting failure mode and shear capacity of ultra high performance concrete beams. *Engineering structures*, *224*, 111221.

[75] Gumaei, A., Al-Rakhami, M., Hassan, M. M., Alamri, A., Alhussein, M., Razzaque, M. A., & Fortino, G. (2020). A deep learning-based driver distraction identification framework over edge cloud. *Neural Computing and Applications*, 1-16.

[76] Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics*, *21*(1), 1-13.

[77] Powers, D. M. (2020). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*.

[78] Umar, M. A., & Zhanfang, C. (2020). Effects of Feature Selection and Normalization on Network Intrusion Detection.

[79] Uthayakumar, J., Metawa, N., Shankar, K., & Lakshmanaprabu, S. K. (2020). Intelligent hybrid model for financial crisis prediction using machine learning techniques. *Information Systems and e-Business Management*, *18*(4), 617-645.

[80] Hossin, M., & Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, *5*(2), 1.

[81] Wang, W., & Lu, Y. (2018, March). Analysis of the mean absolute error (MAE) and the root mean square error (RMSE) in assessing rounding model. In *IOP conference series: materials science and engineering* (Vol. 324, No. 1, p. 012049). IOP Publishing.

[82] Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. *Geoscientific model development*, *7*(3), 1247-1250.

[83] Tofallis, C. (2015). A better measure of relative prediction accuracy for model selection and model estimation. *Journal of the Operational Research Society*, *66*(8), 1352-1362.

[84] Bowes, D., Hall, T., & Gray, D. (2012, September). Comparing the performance of fault prediction models which report multiple performance measures: recomputing the confusion matrix. In *Proceedings of the 8th international conference on predictive models in software engineering* (pp. 109-118).

[85] Hossin, M., & Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, *5*(2), 1.

[86] Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information processing & management*, *45*(4), 427-437.

Appendix of Chapter 3

Appendix A

- Access to a terminal window/command line
- Sudo or root privileges on local /remote machines

Appendix B

1. Install OpenJDK on Ubuntu.
2. Set Up a Non-Root User for Hadoop Environment. Install OpenSSH on Ubuntu 18.04.
3. Download and Install Hadoop on Ubuntu.
4. Single Node Hadoop Deployment (Pseudo-Distributed Mode) Configure Hadoop Environment Variables (bashrc).
5. Format HDFS NameNode.
6. Start Hadoop Cluster.

Appendix C

Step1: Install OpenJDK on Ubuntu

The Hadoop framework is written in Java, and its services require a compatible Java Runtime Environment (JRE) and Java Development Kit (JDK). Use the following command to update your system before initiating a new installation:

```
sudo apt-get update
```

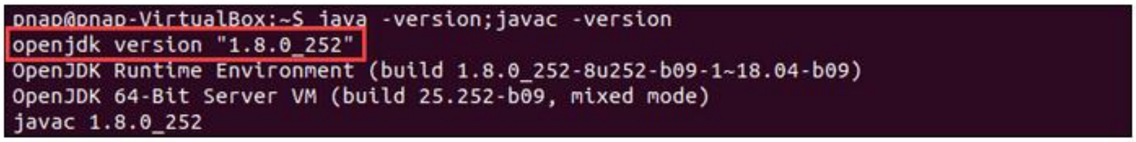
At the moment, Apache Hadoop 3.2.1 fully supports Java 8. The OpenJDK 8 package in Ubuntu contains both the runtime environment and development kit. Type the following command in your terminal to install OpenJDK 8:

```
sudo apt install openjdk-8-jdk -y
```

The OpenJDK or Oracle Java version can affect how elements of a Hadoop ecosystem interact. To install a specific Java version, check out our detailed guide on how to install Java on Ubuntu. Once the installation process is complete, verify the current Java version:

```
java -version; javac -version
```

The output informs you which Java edition is in use.



```
pnap@pnap-VirtualBox:~$ java -version;javac -version
openjdk version "1.8.0_252"
OpenJDK Runtime Environment (build 1.8.0_252-8u252-b09-1~18.04-b09)
OpenJDK 64-Bit Server VM (build 25.252-b09, mixed mode)
javac 1.8.0_252
```

Step 2: Set Up a Non-Root User for Hadoop Environment

It is advisable to create a non-root user, specifically for the Hadoop environment. A distinct user improves security and helps you manage your cluster more efficiently. To ensure the smooth functioning of Hadoop services, the user should have the ability to establish a passwordless SSH connection with the localhost.

Step 3: Install OpenSSH on Ubuntu

Install the OpenSSH server and client using the following command:

```
sudo apt install openssh-server openssh-client -y
```

Step 4 : Create Hadoop User

Utilize the **adduser** command to create a new Hadoop user:

```
sudo adduser hdoop
su - hdoop
```

Step 5: Single Node Hadoop Deployment (Pseudo-Distributed Mode)

Hadoop excels when deployed in a fully distributed mode on a large cluster of networked servers. However, if you are new to Hadoop and want to explore basic commands or test applications, you can configure Hadoop on a single node.

This setup, also called pseudo-distributed mode, allows each Hadoop daemon to run as a single Java process. A Hadoop environment is configured by editing a set of configuration files:

- bashrc
- hadoop-env.sh
- core-site.xml
- hdfs-site.xml
- mapred-site-xml
- yarn-site.xml

- Configure Hadoop Environment Variables (bashrc)

  Edit the .bashrc shell configuration file using a text editor of your choice (we will be using nano):

```
sudo nano .bashrc
```

Define the Hadoop environment variables by adding the following content to the end of the file for Hadoop related options:

```
 export HADOOP_HOME=/home/hdoop/hadoop-3.2.1

export HADOOP_INSTALL=$HADOOP_HOME

export HADOOP_MAPRED_HOME=$HADOOP_HOME

export HADOOP_COMMON_HOME=$HADOOP_HOME

export HADOOP_HDFS_HOME=$HADOOP_HOME

export YARN_HOME=$HADOOP_HOME

export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native

export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin

export HADOOP_OPTS"-Djava.library.path=$HADOOP_HOME/lib/nativ"
```

Once you add the variables, save and exit the .bashrc file.

- **Edit hadoop-env.sh File**

When setting up a single node Hadoop cluster, you need to define which Java implementation is to be utilized. Use the previously created $HADOOP_HOME variable to access the hadoop-env.sh file:

```
sudo nano $HADOOP_HOME/etc/hadoop/hadoop-env.sh
```

and add the code section inside hadoop-env.sh file:

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

- **Edit core-site.xml File**

The core-site.xml file defines HDFS and Hadoop core properties.

Open the core-site.xml file in a text editor:

```
sudo nano $HADOOP_HOME/etc/hadoop/core-site.xml
```

Add the following configuration to override the default values for the temporary directory and add your HDFS URL to replace the default local file system setting:

```
<configuration>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/home/hdoop/tmpdata</value>
</property>
<property>
  <name>fs.default.name</name>
  <value>hdfs://127.0.0.1:9000</value>
</property>
</configuration>
```

- **Edit hdfs-site.xml File**

The properties in the hdfs-site.xml file govern the location for storing node metadata, fsimage file, and edit log file. Configure the file by defining the NameNode and DataNode storage directories.

Use the following command to open the hdfs-site.xml file for editing:

```
sudo nano $HADOOP_HOME/etc/hadoop/hdfs-site.xml
```

Add the following configuration to the file and, if needed, adjust the NameNode and DataNode directories to your custom locations:

```
<configuration>
<property>
  <name>dfs.data.dir</name>
  <value>/home/hdoop/dfsdata/namenode</value>
</property>
<property>
  <name>dfs.data.dir</name>
  <value>/home/hdoop/dfsdata/datanode</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
</configuration>
```

- **Edit mapred-site.xml File**

Use the following command to access the *mapred-site.xml* file and **define MapReduce values:**

```
sudo nano $HADOOP_HOME/etc/hadoop/mapred-site.xml
```

Add the following configuration to change the default MapReduce framework name value to **yarn**:

```
<configuration>
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
</configuration>
```

- **Edit yarn-site.xml File**

The *yarn-site.xml* file is used to define settings relevant to **YARN.** It contains configurations for the **Node Manager, Resource Manager, Containers,** and **Application Master**.

Open the *yarn-site.xml* file in a text editor:

```
sudo nano $HADOOP_HOME/etc/hadoop/yarn-site.xml
```

Append the following configuration to the file:

```
<configuration>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>127.0.0.1</value>
</property>
<property>
  <name>yarn.acl.enable</name>
  <value>0</value>
</property>
<property>
  <name>yarn.nodemanager.env-whitelist</name>
  </property>
</configuration>
```

- **Format HDFS NameNode**

It is important to format the NameNode before starting Hadoop services for the first time:

```
hdfs namenode -format
```

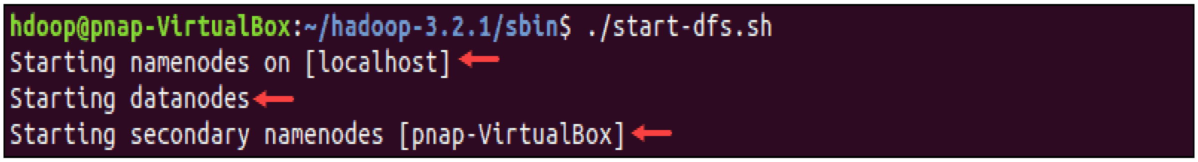
The shutdown notification signifies the end of the NameNode format process.

- Start Hadoop Cluster

Navigate to the hadoop-3.2.1/sbin directory and execute the following commands to start the NameNode and DataNode:

```
./start-dfs.sh
```

The system takes a few moments to initiate the necessary nodes.

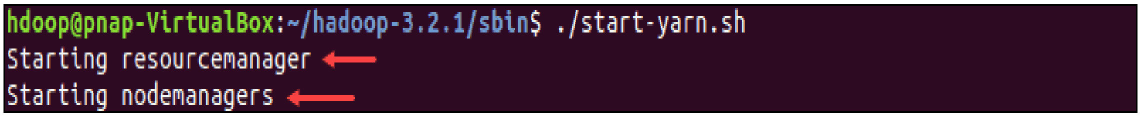


```
hdoop@pnap-VirtualBox:~/hadoop-3.2.1/sbin$ ./start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [pnap-VirtualBox]
```

Once the namenode, datanodes, and secondary namenode are up and running, start the YARN resource and nodemanagers by typing:

```
./start-yarn.sh
```

As with the previous command, the output informs you that the processes are starting.



```
hdoop@pnap-VirtualBox:~/hadoop-3.2.1/sbin$ ./start-yarn.sh
Starting resourcemanager
Starting nodemanagers
```

Type this simple command to check if all the daemons are active and running as Java processes:

```
jps
```

If everything is working as intended, the resulting list of running Java processes contains all the HDFS and YARN daemons.

Appendix D

```
./start-yarn.sh
```

Appendix E

```
jps
```

Appendix F

1- Run the Command: $ /usr/local/hadoop/bin/start-dfs.sh, stop-dfs.sh

Appendix G

2- Run the Command: $ /usr/local/hadoop/bin/start-all.sh,stop-all.sh

الخلاصة

السكتة الدماغية هي السبب الثالث الأكثر شيوعا للوفاة والإعاقة طويلة الامد بين البالغين في جميع أنحاء العالم. ولذلك ، فإن التنبؤ بالسكتة الدماغية والتشخيص مسألة بالغة الأهمية. والوعي المبكر بمختلف علامات التحذير من السكتة الدماغية يمكن أن يقلل من السكتة الدماغية. وتتيح تقنيات استخراج البيانات امكانية المساعدة في تحديد الروابط بين خصائص بيانات المرضى ، أي استخلاص المعرفة اللازمة من نظام المعلومات الطبية للتنبؤ بمختلف الأمراض وعلاجها.

ويهدف النظام المستخدم إلى زيادة الدقة إلى أقصى حد للتنبؤ بأمراض السكتة الدماغية وتقليل الوقت اللازم لبناء النموذج التنبؤي باستخدام الهادوب مقلل/الخريطة مع تعلم الالة ، حيث يؤثر الوقت على المعالجات الاحتسابية للنظام. يأخذ النظام المقترح بعمله نظر الاعتبار لنوع الجنس والعمر وارتفاع ضغط الدم وأمراض القلب وحالة التدخين في سمات التنبؤ بالسكتة الدماغية. حيث يستند إلى ثلاث حالات للدراسة ، تستند حالة الدراسة الأولى على استخراج البيانات / التعلم الآلي مع خوارزميات التعلم المتمثلة بال DT و SVM و RF على التوالي. بينما حالة الدراسة الثانية تتمثل استخراج البيانات / الهادوب -العداد / التعلم الالي ، وحالة الدراسة الثالثة معتمدة على استخراج البيانات / الهادوب-الوزن / التعلم الالي المتمثل بخوارزميات ال NB، SVM و DT لكلتا حالتين الدراسة الخاصة بالهادوب.

يكشف تقييم الأداء للنظام المقترح أن دراستي الحالة الثانية والثالثة على التوالي المستندة على الهادوب وخوارزميات التعلم الالي المتمثلة بال DT ، NBو SVM ،قدمتا اعلى دقة تقدر بحوالي ٩٨.٦٤٦% لهذه الخوارزميات المذكورة، مع انخفاض عدد القيود بقاعدة البيانات ضمن مجموع البيانات من ٤٣٤٠٠ قيد إلى ٢٥٨٥ قيد ضمن مجموع بيانات حالة الاختبار لقاعدة البيانات وتناقص الوقت المحدد لبناء النظام لحالة الهادوب / الوزن إلى٢٨ ملي ثانية لخوارزمية NB ،وخوارزمية SVM إلى ٣٧١ ملي ثانية ، بينما خوارزمية ال DT إلى ٤٧٣ملي ثانية مقارنة مع الأعمال الأخرى ذات الصلة في نفس المجال ونفس قاعدة البيانات (قاعدة البيانات الضخمة الخاصة بمرض السكتة الدماغية) والتي اعطيت أفضل دقة للتنبؤ بمرض السكتة الدماغية. بجانب ذلك يضيف النظام المقترح الذي يستند على الهادوب تحسين حالة التنقيب عن البيانات بعمل توازان وتألف للحقول ضمن قاعدة البيانات بتقليل عدد القيود واضافة ميزة العداد والوزن لها. وعلاوة على ذلك ، فإن نسبة تنبؤات النظام المقترح بشأن ما إذا كانت فئة ما من السكتة الدماغية أم لا أظهرت النتائج ان معدل الكشف المرتفع من هادوب/العداد - هادوب/الوزن حوالي ١٠٠% لخوارزميات NB ، SVM ، و DT ، إلى جانب ذلك ، فإن Recall كمقياس لكمية التنبؤ الناجحة بالسكتة الدماغية حوالي ١٠٠% لحالتي الهادوب. كما أن المعدل الإيجابي الزائف (٧) والمعدل السلبي الزائف (٠) في دراسات هادوب هما الأدنى مقارنة بالتوجهات البحثية الأخرى القائمة على خوارزميات التعلم الآلي.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل
تكنولوجيا المعلومات

تنفيذ شبكة قوية للهادوب والتعلم الالي في مجال الرعاية الصحية

رسالة مقدمة

إلى مجلس كلية تكنولوجيا المعلومات في جامعة بابل والتي هي جزء من متطلبات الحصول على درجة الماجستير في تكنولوجيا المعلومات / شبكات المعلومات

من قبل الطالب

مخلد فالح حسن علي

باشراف

أ.م.د فرقان حسن صالح

أ.م.د مهدي عبادي مانع مهدي

٢٠٢١ م

١٤٤٣ هـ