THE ANALYSIS ON
THE POTENTIAL BREAST CANCER BY
USING BIG DATA ENVIRONMENT

YEE MON EI

M.C.Sc. **SEPTEMBER 2022**

THE ANALYSIS ON THE POTENTIAL BREAST CANCER BY USING BIG DATA ENVIRONMENT

BY

YEE MON EI

B.C.Sc.

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Computer Science (M.C.Sc.)

University of Computer Studies, Yangon

September 2022

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

…..…… ……..……

Date Yee Mon Ei

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my sincere thanks to all my teachers who gave me many valuable advice and information. I am also grateful to all respectable people who directly or indirectly contributed towards the success of this thesis.

I would like to express my respectful thanks to **Dr. Mie Mie Khin**, Rector of the University of Computer Studies, Yangon for her kind permission to conduct this thesis.

I would also like to offer my deep and sincere gratitude to my supervisor, **Dr. Thida Aung**, Lecturer, Faculty of Computer Science, the University of Computer Studies, Yangon, her effort, time in reading and patience to help me in accomplishing this paper. It was a great privilege and honor to work and study under her guidance.

I would like to express my gratitude and appreciation to **Dr. Tinzar Thaw** and **Dr. Si Si Mar Win**, Professor of Faculty of Computer Science, University of Computer Studies, Yangon who offer me an unrestricted support and valuable and timely advice and suggestions for the completion of this work.

I also thank **Daw Aye Aye Khine**, Associate Professor & Head, Department of English, University of Computer Studies, Yangon, for her kind suggestion in writing my thesis documentation.

I would like to thank all my teachers for their motivated, encouragement and recommending the thesis.

Furthermore, I am extremely grateful to my companions who gave me their precious ideas and invaluable knowledge throughout this thesis. Finally, I am extending my heartfelt thanks to my family for their encouragement and support to accomplish this work.

ABSTRACT

Nowadays, big data is widely used in healthcare for prediction of diseases. Breast cancer is the most occurred cancer disease in the world that occurs in a woman. If this disease is detected in early stages, there will be a better chance for curing. In this system, a scalable and fault tolerant pipeline model is proposed for analyzing big cancer data and predicting the cancerous cells. Nowadays, a large amount of digital data is generated from everywhere, every second of the day. One of the challenges is the volume of generated data with high dimensionality. Most of traditional machine learning algorithms are not good in training time and classification result to find hidden insights from these high dimensional data. This model is developed on Apache Spark Framework using Random Forest algorithm and the used data source is Wisconsin Diagnosis Breast Cancer Dataset of the University of California at Irvine (UCI) Machine Learning Repository. This system is implemented using Apache Spark-based Random Forest algorithm in order to compare with Naïve Bayes in terms of accuracy, precision, recall and f-measure. The analysis of evaluation results describes the achievement of the proposed system with the accuracy of 98.2% in the Big Data Analytics Environment. The proposed system is implemented by Scala programming language on Linux platform.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES **PAGES**

LIST OF TABLES

LIST OF EQUATIONS

CHAPTER 1

INTRODUCTION

Nowadays, the data rate increment based on the technology development occurs volume of data with many characteristics becomes big data. The big data exists by many formats of data. Although data storage occurs no expensive, this increasing data in many resources of data by many formats of data happens new problems relating to data processing. The linkage of created data with various events is done so quicker and more accurate network connections, inexpensive storage in healthcare sectors and the application of devices for the creation of mobile phones, gps enabled devices and so on. The occurrence of data statistics with data collection is formed according to many devices, equipment, telecommunications and so on.

Big data has been applied for huge volume of data in managing, analyzing, and storing as the utilization of vast information resources. Big data contains various data records exceeding the capability of conventional data warehouse. In various business sectors, the data is vast or faster. It has the ability in assisting the organizations for consolidation its recommendations, providing fast and simple answers. Big data has the potential effect on the control for structured, semi-structured, and unstructured data. Big data can handle the retrieval, storage, and operation to vast amount of data. Organizations are recognized that it has a hidden pattern in these data sets. The index finding, the specification of business patterns, the creation of new products, the comprehension of customer relations or machine learning are the application of big data. The variation of format in big data is greatly. Word documents, spreadsheets and relational data bases are instances of semi-structured and structured data while emails or social media posts are instances of unstructured data. Data is with various format in certain path. Three different data formats are:

- **Structured data:** is the information existing in a static field in a record including information in spreadsheets and relational databases.
- **Semi-structured data:** is the information not existing in a relational database they have some organizational features that can occur the easier analysis than structured data.

- **Unstructured data:** is the information residing in the non-relational database with no pre-specified data model. It contains text and multimedia content consisting e-mail, audio file, and so on.

Moreover, the utilization of data analytics is done for the extraction of valid, useful, and unknown patterns and information from large datasets, for the detection of important relationships between the variables which are maintained. As the vast volume of big data grows, they are becoming bigger interested in generation of hidden patterns from huge amount of data.

The conventional environment differs from the aspects of the big data environment depending upon the large amount of dataset. The conventional database systems, dealing with data analytics as relational databases, encounter big data problems in the restriction of operation capabilities. Therefore, organizations are attempting for big data analytics for analysis huge volume of data with quickly and reveal last unseen patterns, and customer intelligence.

The Hadoop platform was implemented on the big data analytics while providing reliability, scalability, and manageability with the MapReduce. Hadoop is comprised of two principal components: the Hadoop distributed file system for big data storage and MapReduce for big data analytics. The operation of scalability, redundancy, and reliable storage for large scale application is provided by the HDFS. As the splitting of incoming data and arrangement among nodes in the cluster, HDFS is optimal for large files. Moreover, the storage of HDFS provides the reliability and availability if the occurrence of node failures happens during the distribution of data replication. Two types of nodes are name node and data nodes. The data is kept at data nodes based on the replica placement policy of HDFS. Name node is the master node and it controls the data nodes.

It is a cluster computing framework that is open-sourced. At the top of the Hadoop Distributed File System (HDFS), this framework is constructed. It performs distributed data processing and distributes data for the separation of worker nodes to perform processing. A master node performs the management of worker nodes by dispatching and scheduling of distributed tasks. Therefore, a cluster manager and distributed storage system is needed by this framework. This framework has faster in-memory data engine and developer-friendly API that provides in the choice of this framework.

The foundation of Apache spark, Spark Core, works as execution engine. The scheduling, I/O functionalities and dispatching of distributed tasks is supported by spark core. It is possibly performed by an application programming interface (API) in R, Scala, Java, and Python. This can support a wide variety of languages with interfaces available for many languages. It supports faster speed utilization of in-memory computing capabilities.

The four components of spark core are Spark Streaming, Spark SQL, Machine Learning library and GraphX. The module of Spark, Spark SQl is associated with structured data that utilized data frames. Spark SQL possess an interface as SQL for query data processing. The addition of real-time data processing to batch processing of Spark is performed by Spark Streaming. The incoming data stream is broken down into micro batches and its processing is same as batch processing. For graph structures processing, Graph X is a distributed framework at the top of Spark. Users are allowed for the building and processing of interactive graph structured data. The distributed implementation of the machine learning pipelines is allowed by the machine learning library with the significant decrement of the overall processing time.

The model is used to predict classes of objects with unknown class labels. The classification has been successfully applied to a variety of application areas, including medical diagnostics, weather forecasting, credit approval, customer segmentation, and fraud detection across different offerings. Classification is obviously useful for many decision-making issues where decisions are made about data elements (depending on the class to which they belong). A model is developed with a spark machine learning pipeline that processes voluminous data fast and accurately. In this system, Random Forest is used for building prediction model which predict breast cancer data.

1.1 Objectives of the System

The main objectives of the thesis are:

- to investigate the architectures of Hadoop and Apache Spark
- To develop a high-performance and scalable potential Breast Cancer analysis on the Big Data Analytics Platform
- To implement the Breast Cancer Prediction System on Apache Spark
- To analyze the results of the Apache Spark-based Random Forest

1.2 Organization of the System

This thesis is mainly composed of five chapters.

Chapter 1 is the introductory section where the introduction to big data, the objectives, and the organization of the thesis are presented.

Chapter 2 describes the background theory related to big data, big data processing frameworks, applications of big data, machine learning, Hadoop architecture, and classification algorithms, and the related work.

Chapter 3 presents the design of the proposed system, describing system flow, the detailed steps of preprocessing, the Sparke-based Random Forest algorithm, and performance evaluation used.

Chapter 4 mainly describes the implementation of the proposed system in detail, including the experimental setup, the system's implementation, the Spark-based Random Forest and the experimental result.

Finally, Chapter 5 concludes this thesis and further extensions of the proposed system.

CHAPTER 2

THEORETICAL BACKGROUND

Nowadays, the generation in huge amount of information at every second is done by Internet so that it is characterized as a big storage. The existence of data 2.7 Zettabytes in today digital universe is described by the IBM Big Data Flood Infographic. There are 100 Terabytes daily update data and estimation in 35 Zettabytes data generation leads to many tasks on social networks annually by 2020 according to statistics from Facebook.

The development of cloud computing coincides big data analytics. The storage of large volumes of data in different formats has become a big issue. Big data is a description of data with a complex nature that needs improvement in approaches, techniques, and methods for obtaining, operating, and keeping it effectively. Big data possesses 7 elements [23]: volume, velocity, variety, value, veracity, variability, and visibility as follows:

- Volume: The amount of created data has risen in recent years.
- Velocity: The production rate of data has increased.
- Variety: The various format of data from structured to unstructured data.
- Value: The efficiency of the data utilization useful to organizations.
- Veracity: The requirement relating with uncertain data is one factor in big data.
- Variability: Various data structures and data transformation to convert data.
- Visibility: The separation of data from various resources makes up big data.

In last years, many researchers have investigated big data analytics as the growth of data becomes in forecasted amount and complexity. Various organizations collect huge volume of data in analysis to extract options for providing better decisions.

2.1 Big Data Analytics

Analytics of data is the principle utilization of data that can be operated with the utilization of data mining, statistical analysis, machine learning, and mathematical models for giving accurate decisions. Data analytics is the conversion of data into decisions by analyzing it for problem-solving and decision-producing. Moreover, the

combination of data mining techniques and traditional data analytics is called big data analytics. Giving the primary framework for analyzing, model building, and forecasting the patterns of customers, services, and so on. Hadoop is capable of analyzing and storing large amounts of data. The useless operation time can be reduced with a parallel operation platform.

2.1.1 Types of Data Analytics

There are four main types of data analytics:

- Descriptive Analytics: employs data mining and aggregation techniques to aid comprehension of the previous period and result: "What has happened?"
- Diagnostic Analytics: determines data for the question "Why did it happen?"
- Predictive Analytics: applying statistical and forecasting methods to the future and resulting in: "What could happen in the future?"
- Prescriptive Analytics: apply simulation and scalarization methods to give the advice for possible results and outcomes: “What should we do to occur in future?”

Therefore, the question: "What has happened?" is answered by both descriptive analytics, which applies data mining and aggregation approaches to provide decisions by applying statistical models like regression and forecasts to comprehend the future. Various methods contain future outcomes forecasting by utilizing.

2.1.2 Predictive Data Analytics

Predictive data analytics is the process of extracting insights from an existing data set to predict future outcomes and trends with the goal of predicting what will happen in the future. Data is transformed into valuable, actionable insights with an acceptable level of reliability to determine the future outcome of an event. In addition, it consists of many statistical methods such as modeling, data mining, and machine learning. For example, predictive models describe the patterns identified in historical and transactional data when defining risks and business opportunities.

However, there are several ways to analyze big data. The underlying problem may be a statistical problem. The more significant improvements in dimensionality reduction, data quality, predictive capability, and reliability of the learning models can be achieved with big data. Models with a large number of variables or observations

require dimensionality reduction to avoid having too much noisy data that can affect model estimation.

2.2 Big Data Processing Frameworks

Divide and Conquer is the key factor of parallel processing. Therefore, independent division process of data is performed in parallel. This is the imagination of the matrix multiplication so that each operation can be divided and then processed it. The processing of big data process is the division of a problem into many small operations, and combination of these operations into a single result. This is certainly impossible to apply the parallel operation as the best usage if the operation is dependent. Therefore, it is necessary for saving and processing of data considering these points.

2.2.1 MapReduce

MapReduce is the most widely used distributed data processing framework of the MapReduce in order to process huge amount of data in distributed manner, such as Apache Hadoop [6, 7, 8, 11, 16]. The following characteristics of data processing with the MapReduce are:

- Its operation is performed via regular computer that has built-in hard disk, not a special data storage. Each computer has extremely weak correlation where expansion can be hundreds and thousands of computers.
- System and hardware errors are supposed as general conditions, rather than exceptional conditions since the participation of many computers in processing.
- Many complicated problems can be solved by the simple and abstract basic operation of Map and Reduce. Parallel data processing can be easily performed by programmers who do not know parallel processing.
- High throughput is supported by using many computers.

The incoming data stored at HDFS is split to available worker and represented (Map) a value type, and outputs are maintained in a local disk. The compilation of data is done by reducing worker and the output file is produced. The gap between the processing node and the source data location with the placement of worker in the location based on network switch where data is stored is reduced by applying the data locality concept as the characteristics of data storage. The implementation of each

worker can be performed in various languages through streaming interface (standard in/out).

2.2.2 Dryad

Microsoft developed the distributed platform, Dryad [20] in order to support large-scale, parallel, and fault tolerant operation of processing jobs. Since it provides the arbitrary execution with directed acyclic graphs (DAGs), it is more flexible than MapReduce. Moreover, MapReduce can only perform the simple and two-step execution chain of a map with a reduce. Dryad applies its own high-level language known as DryadLINQ [21, 33]. Map and Reduce functions should be applied by developers who use MapReduce framework; therefore, if using Dryad, they should do a graph in order to perform the processing of the corresponding data. DAG type data flow processing can be provided by Dryad. There are the barriers to entry in using the system for inexperienced data analyzers, developers, and data minors although enough functions for processing big data are provided by parallel data operation framework.

2.2.3 Apache Spark

In order to provide data analytics to be faster to run and write, Apache Spark [34] is a parallel data distributed computing framework. Apache Spark supports the general execution model to provide the optimum arbitrary operator graphs, and provides in-memory computing, for faster data querying than disk storage as Hadoop to provide faster programs running. Moreover, Apache Spark supports concise and clean application programming interfaces (APIs) in Java, Scala, and Python in order to provide faster programming. This was originally introduced for two applications where data placement in memory aids: interactive data mining and iterative algorithms, that are common in machine learning. But it can be used for general data processing. Apache Spark is also the engine that resides behind Shark, a fully and compatible data warehousing system with Apache Hive. It can process any data source supported by Hadoop providing it easier to run over existing data although Apache Spark is a new engine.

The obvious issue with hadoop is that read and write calculations to and from hard drives are operated by hadoop for the incoming mapreduce task. Apache Spark

can handle in-memory computing and serialize RDDs for storage. Furthermore, it can do interactive calculations, whereas batch computing is produced by Hadoop. Mapreduce provides a solution to the problems of inefficient repetitive operations. Spark can be applied to handle the issue of repetitive operations while producing effective operations on resilient distributed datasets. The conversion of resilient distributed datasets to direct acyclic graphs is done, and that is planned and operated with the respective nodes in the cluster. Spark owns various benefits in matrix operations as follows:

- Resilient distributed datasets, the storage abstraction of Spark, is a distributed and fault-tolerance vector for developers who can carry out a subset of operations from a regular local vector.
- The execution engine and user-defined data partitioning are allowed by RDDs for co-scheduling tasks for data movement and co-partitioning RDDs.
- To construct an RDD, the lineage of operations is logged by spark for automatic reconstruction of broken partitions upon failure without concern for
- Spark also supports a high-level API in Scala that can aid in the creation of a coherent API for matrix computation.

RDD objects from loaded data are created by Apache Spark at the initial stage of data processing. The parallel scheduling and processing is done by the worker nodes in the cluster. The evaluation is controlled by the driver, the main program. In order to optimize RDD dataset processing, Apache Spark provide machine learning algorithms such as classification, clustering, and data dimensionality reduction. Nowadays, Apache Spark is more popular for scaling up various data processing applications. Apache Spark supports a machine learning library known as "MLlib" to provide machine learning data processing.

2.2.4 Stratosphere

Stratosphere [2] is a massively parallel data processing system. Stratosphere contains the Parallelization Contracts (PACT) programming model [3], a declarative query language Meteor [19], and Nephele [5], the execution engine. The expression of their queries is done by users with the Meteor language. The application of high-level operators like filter to semi-structured data sets is done. Directed acyclic graphs of second-order functions, Pact programs, are included in meteor operators. The Pact

programming model applies the general concept of the MapReduce. Moreover, second order functions, three supplemental are supported by Pact for effective implementation of equi-joins, cross products, and groupings from two sources. The optimization of Pact programs and the compilation of these programs into data flow graphs are done in parallel processing with the Nephele execution engine. Thus, the transformation of complex user queries into optimized parallel execution graphs can be provided by Stratosphere. The submission of these program to the PACT compiler is done to perform the execution of a PACT program. The translation of the program to a data flow program is done by the compiler and then, submitting to the Nephele system is done to perform parallel execution. Processing data is maintained at HDFS.

2.2.5 GridGain

GridGain [30] is Java-based middleware in order to provide big data in-memory execution in a distributed environment. GridGain is developed on high performance in-memory data platform integration with fast in-memory MapReduce execution with in-memory data grid technique providing easier to apply and to scale software. The processing of data terabytes on 1000s of nodes in short period of time with GridGain can be done. Typically, GridGain resides between analytics, BI or transactional applications, business and long-term data storage like Enterprise Resource Planning (ERP) or Hadoop HDFS, Relational Database Management System (RDBMS), and supports low latency data storage and processing and in-memory data platform for high performance.

Hadoop and GridGain are developed to provide parallel distributed data processing. Nevertheless, they provide for very various purposes and in many cases are very supplementary to each other. While GridGain provides low latency real-time in-memory processing on non-transactional and transactional data, Hadoop provides batch-oriented offline processing for historical and data analytics where latencies and transactions do not really matter. Depending on very different conditions make GridGain and Hadoop very supplementary with each other. GridGain in-memory data grid and compute grid perform directly in real-time with user application with caching and partitioning data within data grid, and operating in-memory calculations and SQL queries on it. If data is historical, snapshotting of this data into HDFS is done in where

data analysis is performed with Hadoop MapReduce and analytical tools from Hadoop ecosystem.

2.2.6 Storm

Storm [22] is an open source and free distributed real-time computation system. Storm provides reliable and real-time unbounded data streams processing, while Hadoop supported on batch processing. It is very easy to be applied with any programming language. It has been applied in various conditions such as online machine learning, distributed Remote Procedure Call (RPC), real-time analytics, continuous computation, ETL, and etc. It is fast and a benchmark over million tuples processes per second at each node. Storm is fault-tolerant and scalable data processing, and is easy to execute and set up. It integrates with database and queuing techniques. Data streams are consumed by Storm topology and then those streams are arbitrarily processed with complex ways, the streams between each calculation step are repartitioned whenever needed. It is similar to a Hadoop cluster. Storm topologies are run whereas on Hadoop MapReduce jobs are run. Topologies and jobs are very distinct -- one main difference is that a topology processes messages forever whereas a MapReduce job eventually completes. Two types of nodes in Storm cluster are: the master node and corresponding worker nodes. A daemon called Nimbus is run by the master node like Hadoop's JobTracker. Nimbus performs code distribution in the cluster, tasks placement to nodes, and failure detection.

A daemon called the Supervisor is run by each worker node. The supervisor detects for task placement to nodes and stopping and starting worker processes depending on the assignment of Nimbus. A subset of a topology is executed by each worker process; many worker processes distributed among many nodes are included in a running topology. The cooperation among the Supervisors and Nimbus is performed by a Zookeeper cluster. Moreover, Supervisor daemons and the Nimbus daemon are stateless and fail-fast; all condition is kept on local disk or in Zookeeper.

2.3 Distributed File Systems

A file system is a procedure which operates the controlling of access, storage and computation. This is the logical disk element for controlling the internal calculation of disk relating with system and human. The file operation at specific generation is

taken till certain protection and corruption of threats in the system. In this distributed file system, data access and sharing are performed at host machine. This is effective in information sharing to users with the manner of authorized. Data maintenance and file sharing are permitted by the server to host users. However, the access right is provided to clients by the server and this server own full authority on the data. It includes clients, service, and server. The server is the application for providing services and operations at one machine. The client is the job that can demand the service by the actions becoming the client interface that is constructed with the set of file actions like writing, deleting, creating and reading. The service is the set of software operating at many machines and provides the specific actions to unknown users. The distribution of server, storage and clients are done on the machines in the distributed system. They possess many free storage machines organized by a single centralized place and the service is done on the network.

2.3.1 Google File System

The data operation requirements of google is handled by the generation of distributed file system. The google file system has the capability of availability, fault tolerance, scalability, reliability, and performance in the assignment of networks to nodes. This is constructed by many storage systems with lower cost hardware elements. It includes a master and various client and the users does the access.

The master maintains the information of all file operations. This includes the block location, the mapping among files and blocks, access information, and the namespace. It controls system operations like the server migration, release control, and useless chunks collection. The connection with master and client is done at every interval by delivering heartbeat notifications for doing the situation collection and producing the instructions.

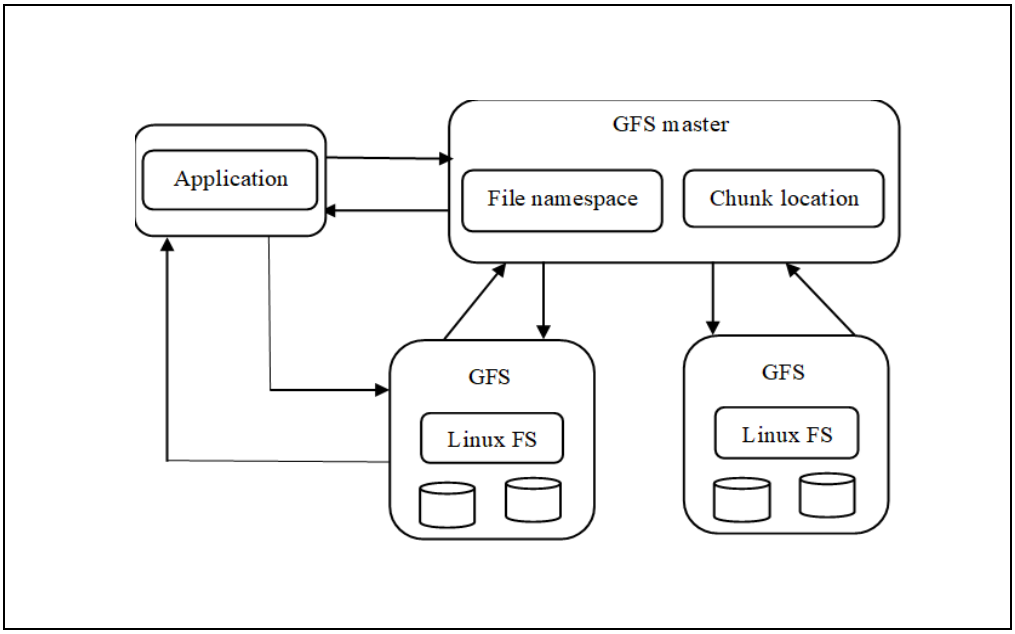

Figure 2.1 General Architecture of GFS

The splitting of the incoming data into 64MB-sized blocks is done. These blocks are kept on local disks with block servers like linux files and write or read chunk data defined by a byte range and chunk handle. The replication of every block is performed at various block servers for reliability. The default number of replicas is three, whereas various replica numbers are user-defined in various places.

2.3.2 Hadoop Distributed File System

The Hadoop distributed file system was introduced by Yahoo and that is the same with the Google file system. However, it is open-source and lighter than the Google File System. The larger cluster is constructed by the interconnection of nodes in the cluster. It includes a namenode for file management and datanodes for block storage in files. The HDFS architecture is shown in figure 2.2.

It operates the separate storage of application data and file system metadata. Like in other distributed file systems, metadata information is maintained at a dedicated server, while the namenode and application data are maintained at other servers, called datanodes.

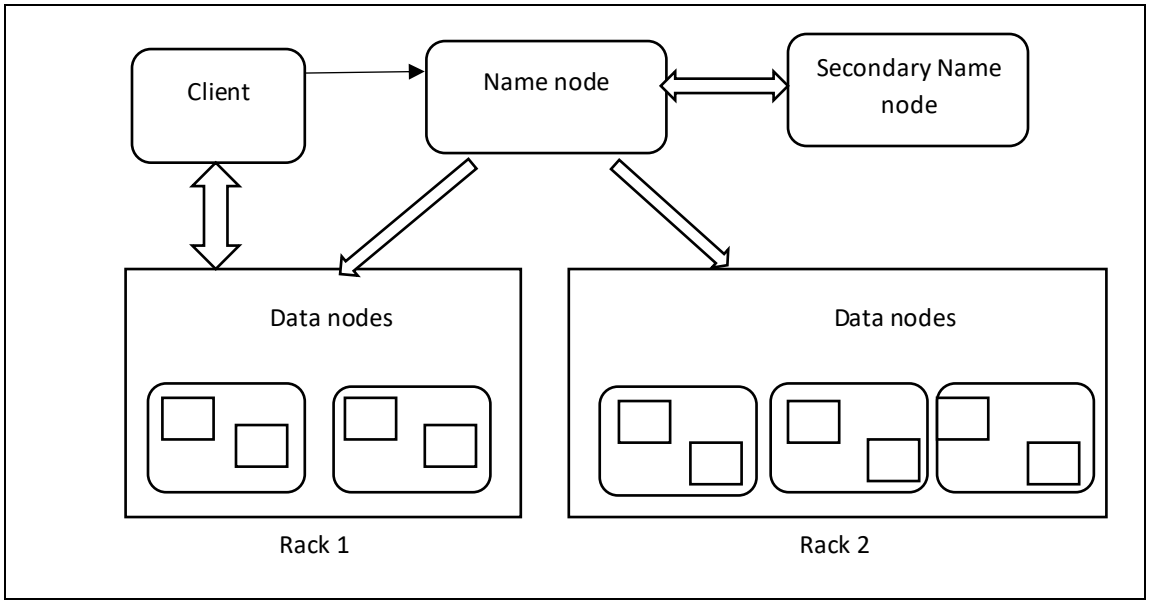

Figure 2.2 General Architecture of HDFS

A factor is the name node dependency for the management of all data block executions in the file system. As a consequence, it becomes a single point of failure and a bottleneck resource. To solve these issues, a distributed file scheme was proposed in [13]. The system applies a light-weight front-end server for making the connection of many name nodes with all requests. This produces a workload distribution of a name node to various data nodes.

2.4 Applications of Big Data

The main aim of big data analytics is to help organizations in the generation of many business. Organizations in various domains are making an investment in big data applications to decide huge datasets for finding. The industry domains applying big data are:

- **Banking:** In this domain, big data is widely applied for fraud detection of the banking sectors. The abuse of debit and credit cards, the storage of inspection tracks, the prevention of venture credit hazard, modification of customer statistics, IT strategy fulfillment analytics, and public analytics for business are investigated.
- **Insurance:** In this domain, big data is applied for good consumer observation from insurance organizations. The improvement of scan discovery has been

achieved. The real-time analytics is deployed in huge volume of data from social media.

- **Healthcare:** In this domain, big data is applied for healthcare application. With the big data analytics, all patient records, descriptive information, and treatment planning are accessed in efficiently. Moreover, the effective classification of diseases can be performed in the medical diagnosis.
- **Manufacturing:** In this domain, the improvement in result and quality is provided with the minimal of waste products. Many manufacturers are related on analytics at which they require for handing problems more quickly and generating many agile business decisions.

2.5 Machine Learning

Machine learning is a kind of artificial intelligence directed at enabling machines to execute their tasks with skillfulness, with the programs building on learning from their last experience. The principal factors of artificial intelligence are the capabilities for pursuing purposes and planning to next activities. The division of data to various groups and giving prediction on next information events according to last data are included in the possible actions. It is the expected appealing fact for the manual building and the application of machine learning has increased among computer science in the most recent decade. Data is the most principal in machine learning and the application of the learning algorithm is done for achieving and studying properties or knowledge by the data. The amount of dataset has the actions on the forecasting and learning performance. In machine learning, the learning methods can be divided into taxonomies according to the expected result of the method.

2.5.1 Fundamental Aspects of Machine Learning

Various facts need to be decided in the design of the machine learning method for gaining good performance. Designing machine learning contains the following:

(i) Pre-processing

The primary stage of machine learning is preprocessing at where the raw data are required to preprocess and prepare by utilizing preprocessing methods according to some earlier system than feeding into the system. For instance, the conversion of document images into binary format is performed before the putting with various kinds

of optical character recognition or layout analysis. If the machine learning platform does not solve incomplete datasets, the input raw data may be incomplete as missing values in other events. For this condition, the suitable developed preprocessing stage can solve by filling this gap in the dataset with applying other optimal statistical methods or averaging. Many steps of preprocessing contain for removing outliers or noise in the dataset. So, the selection and providing usage of suitable preprocessing approach may possess the potential effects on other steps done by machine learning.

(ii) Feature Selection

The processing of patterns is performed and the patterns are divided into many measurement metrics known as features. A suitable feature set must be selected for pattern recognition. The chosen features must provide the satisfaction specific facts in order to be most efficient. With the irrelevant features of the input data removed, they reduce resource and memory consumption and computation time. Although prior knowledge of the domain of the problem may be required, selecting the appropriate features is a risky job that involves many implementations.

(iii) Model Selection

The accuracy of the classification depends upon the choice of model as many kinds of models may provide many estimations on various problem domains. The most accurate estimations suffer in forecasting improvement and higher classification rates. Not only selecting a model takes place in the performance evaluation but also the quality and amount of instance data are important for the deciding the accuracy of the classifier.

(iv) Training, Testing and Optimization

They are implemented with data sample after choosing the classification approach. Data training is the model training taken on a data subset. The generation of the model from the training step is done for the further decision and testing in a continuous testing stage by utilizing testing instance. The problem is that the operation time for each step may occur excessive according to the methods applied. Generally, many iterations happen during training and testing stages within the parameter optimization may need the significant operation needs. The answer to this problem is the efficient application of machine learning methods can be provided with using the parallelization in the system. As a consequence, the system is authorized to apply

various processors and a large number of processing machines at the same time. This is where the aspects of distributed operation occur in the picture.

2.5.2 Types of Machine Learning Algorithms

There are four main groups according to their purpose:

- Supervised learning
- Unsupervised learning
- Semi-supervised learning
- Reinforcement learning

(1) Supervised learning

The job of this procedure is to provide rules which provides the outcome of the provided input. Till the performance of the model is higher effective, the training phase must obey for continuing. This system must be ready for the associate classification to outcome objects hidden along the training step after the complement of training procedure. Generally, the classification procedure is basically accurate and quick.

In supervised learning approaches, the training of the model is done by a dataset including labelled instances. Therefore, this learning approaches point out the forecasting of the output values for the new data with dependencies and connection modelling among input features and labels in the dataset that may be forecasting after the training stage. Popular supervised learning approaches are Bayesian statistics, lazy learning, support vector machines, Bayesian networks, artificial neural network, nearest neighbor algorithm, gaussian process regression, decision trees, hidden Markov model, boosting, ensembles classifiers, linear classifiers etc. The classification of supervised learning approaches into regression and classification.

- **Regression:** The aim of regression method is for the forecasting of the outcome of associated input. For example, in order to forecast the value of some product, as the price of a stock or the price of a house in a specified town. There have various elements that can be provided the forecasting by applying regression.
- **Classification:** The aim of classification method is for providing class assignments. This has the ability to forecast the outcome value, and the data is

classified into "classes". For instance, the recognition of an automobile type in a photo, of today's weather, and of spam mail.

(2) Unsupervised learning

Unsupervised learning is learning where the forecasted outcome labels are unknown. The training of the model is done by applying this unlabeled data. These approaches direct the search for the hidden layers as well as the realization of sets of photos with similar cars, but there is a risk in the implementation and it cannot be utilized as a supervised learning method. For producing accurate results, the internal data structure may provide the information. The training of model is done by unlabeled dataset in this learning. The most commonly applied approach is the clustering in this unsupervised learning. Clustering is commonly applied for detection of pattern and modelling of description. As there has no labels for learning, the approaches applied in this learning searches the unlabeled input data with grouping and summarization of associated data points, with patterns detection for achieving specific information and producing forecasting. Mostly applied unsupervised learning methods are hierarchical clustering, apriori algorithm, outlier detection, clustering, self-organization map, and éclat method. Clustering is one kind of unsupervised learning methods.

- **Clustering:** This can be applied to discovering variations and similarities. It assembles the same things together. However, it doesn't require understanding any class labels, but the system can know the data itself and cluster it well. In comparison with classification, the outcome labels are not pre-known. This type of learning algorithm can handle various issues.

(3) Semi-supervised learning

Something is required among these two types of machine learning methods, unsupervised and supervised learning for all the investigations. This can be applied semi-supervised learning in such conditions, that refers to a learning process where many outcome values are missing. This requires for using each unsupervised and supervised path as for producing useful outcomes. It is usually the event in medical fields, where medical doctors do not own for performing the manual classification all ways of health issue based on the overwhelming huge volume of data.

(4) Reinforcement Learning

The expected outcome value is explicitly unknown, but the system can provide feedback on the expected outcome. Reinforcement learning is learning provided with such feedback. It is applied in order to train artificial intelligence in gaming on the nero game and might be used in schools. The specific title is studied by the students after they sit an exam, and the students are provided by the teacher with grades without any definition of what the answers were right or not. Reinforcement learning is also a kind of artificial intelligence. In this learning, the continuous learning method is done from the environment in which it is operating, depending upon the reward thing. The main objective is for maximizing the cumulative reward until the full range in the possible events achievement. Mostly applied reinforcement learning methods are; temporal difference, deep adversarial network, and q-learning.

2.6 Classification Algorithms

Classification approach is utilized for the preparation of dataset. The results are kept for using later; this step is called learn model. The checking of data is done with the classifier. The accuracy is good when the result of the classification is the same with the known class that refers the training instance and the classification method possess the better performance. When they possess bad classification and bad training dataset, the system achieves the poor performance in the testing. The models are learned with applying one learning methods from training dataset. The testing is performed with testing data to produce testing instances [18]. Various classification methods are:

(i) Random Forest
(ii) Support Vector Machine
(iii) K-nearest neighbor
(iv) Logistic Regression
(v) Decision Tree
(vi) Naïve Bayes

2.6.1 Random Forest

It is a supervised algorithm and a kind of overall tutorial. It is a very versatile algorithm that can perform both regression and classification. It is based on a decision

tree. Basically, multiple decision trees can be created and merged to get a result. The algorithm considers only a subset of the features. It has the same hyperparameters as the decision tree. The advantage of random forests is that they work very efficiently with large data sets. This adds more randomness to the model, making it a better model. The disadvantage of this model is that it uses a large number of shafts to slow down the speed.

2.6.2 Support Vector Machine

It is a supervised learning algorithm that classifies cases by delimiters. It works by mapping the data to a high-dimensional feature space and finding the delimiter. Find the n-dimensional space in which you want to classify the data points. This algorithm finds the optimal level with the best margin. The boundaries that classify data points are called hyperplanes. Data points are classified according to their position relative to the hyperplane. SVM tuning settings are kernel, gamma, and regularization settings. The linear kernel predicts a new input through the point product between the input vector and the support vector. Mapping data to a high-dimensional space is called a kernel ring. Kernel functions can be linear, polynomial, RBF, or sigmoid. The normalization parameter is a C parameter with a default value of 10. Less normalization means incorrect classification. A small gamma value means you can't find a range of data. The model can be improved by increasing the importance of the classification of each data element.

Support vector machines (SVMs) are based on the principle of structural risk minimization (SRM), and SRM results are classifiers with the lowest expected risk and therefore better generalizations in the test set. The simplest form of SVM is a hyperplane classifier. The strength of SVMs lies in their ability to implicitly convert data into a high-dimensional space and create linear binary classifiers in that high-dimensional space. This happens implicitly, so neither the dimensionality of the data nor the dilution of the data in high-dimensional space is an SVM problem without performing calculations in high-dimensional space. The hyperplane of a high-dimensional transformation space leads to complex decision-making surfaces in the input data space [15].

SVMs have been successfully applied to certain types of classification problems and have consistently performed better than other nonlinear classifiers such

as neural networks and mixtures of Gaussian distributions. The development of efficient optimization schemes has led to the use of SVMs for the classification of more important tasks, such as text classification. In the early 90s, there were some early efforts to apply SVMs to speaker recognition. These efforts were only successful due to the lack of effective implementation of the SVM estimation process at the time. SVMs are also being applied to simple telephone classification tasks, and the results are very encouraging.

SVMs are not designed for the temporal structure of data and are therefore not suitable for modeling audio data that evolves over time. To account for temporal fluctuations in audio data, SVM/HMM hybrids have been proposed, where SVM is used as part of the post-processing phase. In this hybrid approach, SVMs are used to process the information provided by the basic HMM system in order to arrive at a definitive hypothesis. Basic HMM systems are used to provide segmentation to construct the input function vector of an SVM that classifies data based on distanceTo integrate SVMs into an HMM framework, these distances must be converted into later probabilities. Several schemes were studied to convert SVM distance to probability in order to adapt SVM classifiers to HMM-based ASR systems. However, the use of SVMs in speech recognition is limited. The advantage of the SVM is that it is an algorithm suitable for estimation in a high-dimensional space and very efficient in memory. The downside of SVMs is that they can suffer from overfitting and work very well with small data sets.

2.6.3 K-nearest Neighbor

An algorithm to classify similar cases. Results are only generated on request. Since there is no learning phase, this is called a lazy learner. The advantage of ANN is that it is one of the simplest algorithms, since only the value of k and the Euclidean distance must be calculated. Due to the lazy learning feature, it can be faster than other algorithms. This works well for multi-class problems. The disadvantage of ANN is that the algorithm does not go through an information gathering phase, so the algorithm may not generalize well. Large data sets are slower because all distances of unknown elements must be sorted and calculated. For best results, normalization of the ANN algorithm data is required. Used for classification and regression. In both cases, the

entry consists of the following training examples in the dataset. The output depends on whether k-NN is used for classification or regression:

- In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.
- In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

k-NN is a type of classification in which a function is approached only locally and all calculations are carried forward until the function is evaluated. Since this algorithm relies on distance for classification, normalization of training data can significantly improve their accuracy if the characteristics represent different physical units or occur at very different scales [31]. For classification and regression, a useful technique is to assign weights to neighbors' contributions, so that closer neighbors contribute more to the average than more distant neighbors. For example, a typical weighting scheme is to give each neighborhood a weight of 1/d, where d is the distance to the neighborhood. Neighbors are taken from a set of objects whose class (for the k-NN classification) or object property values (for the k-NN regression) are known. You can think of this as a training set for an algorithm, but it doesn't require an explicit training step.

2.6.4 Logistic Regression

Logistic regression is defined as a supervised machine learning algorithm that accomplishes binary classification tasks by predicting the probability of an outcome, event, or observation. A logistic regression model predicts a dependent data variable by analyzing the relationship between one or more existing independent variables. For example, a logistic regression could be used to predict whether a political candidate will win or lose an election or whether a high school student will be admitted or not to a particular college. These binary outcomes allow straightforward decisions between two alternatives. Logical regression analyzes the relationship between one or more independent variables and classifies data into discrete classes. It is extensively used in

predictive modeling, where the model estimates the mathematical probability of whether an instance belongs to a specific category or not.

2.6.5 Decision Tree

Decision trees are trees which performs the classification of instances with the sorting according to feature values. Every branch node is described with the selection among many alternatives and every leaf node describes the decision in the decision tree. They are widely applied for achieving information in the decision process. They start by the root node for performing the activities. The iterative splitting of every node is done based on the decision tree classification through the root node. The final outcome is the tree at where every node describes the possible event and result. The mostly common decision tree techniques are C4.5, CART, and ID3 (iterative dichotomies) [26]. The extension of iterative dichotomiser is C4.5. The created decision trees with C4.5 are utilized for the classification as statistical classifier. This tree develops with depth-first approach and permits pruning of outcoming trees. They may relate with missing values, numeric and noisy data. The backward is performed at the creation and the removal is done with the replacement of leaf nodes. The advantages of C4.5 algorithm is:

- The balanced tree is built.
- To produce the decision, the highest normalized information gain is selected.
- This is utilized for solving discrete and continuous data.

2.6.6 Naïve Bayes

Naïve Bayes is a popular algorithm for classifying text. The Bayesian Classification represents a supervised learning method as well as a statistical method for classification. The classifier is a straightforward and powerful algorithm for the classification task. Multinomial Naïve Bayes is a specific instance of Naïve Bayes where the P (Feature | Class) follows multinomial distribution like word counts, probabilities, decision making. The general term Naïve Bayes refers the strong independence assumptions in the model, rather than the particular distribution of each feature.

Naïve Bayes model assumes that each of the features it uses are conditionally independent of one another given some class. Naïve Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of features in a learning problem. The classifier often performs much more complicated solutions. A Naïve Bayes classifier works by figuring out the probability of different attributes of the data. being associated with a certain class. To understand the Naïve Bayes classifier, need to understand the Bayes theorem. So, let's first discuss the Bayes Theorem.

Even though, there is a data set with millions of records with some attributes, it can evaluate by the Naive Bayes approach. The data set available publicly that can be used for the process.

2.7 Related Works

Many researchers use Hadoop and Apache Spark architectures to facilitate their research.

V. Deshwal and M. Sharma implemented a breast cancer prediction model on a support vector machine using grid research[9]. The first model of a support vector machine is tested without raster search. The support vector machine model is then tested in the raster search. Finally, a comparative analysis was carried out on the basis of the results. A new model was built. The newly designed model is based on rasterizing the data before inserting it into the predictions, which improves the results.

M. J. Domingo, B. D. Gerardo, and R. P. Medina analyzed the potential of using a hybrid of the fuzzy decision tree in classifying a stage of breast cancer that can be used by experts in the SEER dataset [10]. Three (3) performance indicators are used to compare the two (2) algorithms: accuracy, sensitivity, and specificity. The result shows that the performance of Fuzzy Decision Tree is significantly higher than the traditional J8 Decision Tree. Performance comparison shows that the fuzzy decision tree achieved a higher accuracy of 99.96%, the sensitivity of 99.26% and specificity of 99.98% than the decision tree classification technique with 99.75%, 94.31%, and 99.86% respectively. The simulation results showed fuzzy decision tree classifier classifies the data with 165,124 instances which is equivalent to 99.97% and only 351 incorrect classified instances or 0.21%.

B. Sathiyabhama et. al. implemented breast cancer prediction and diagnosis method [29]. In this paper, Spark based methodology for breast cancer prediction. The

advantage of our proposed methodology is that the computational complexity and response time are reduced as related to the traditional single core machine learning techniques. In addition, experimentation analysis on three different classifiers with WBCD dataset demonstrates that Random Forest Classifier outperforms than other models. It is observed that the Random Forest Classifier attained the highest classification accuracies of 96.4912% for 80–20 data separation. The experimental outcomes also demonstrate the superiority of the Random Forest Classifier in terms of sensitivity, specificity, precision, and F-Measure.

Sahan et al. presented a hybrid soft computing technique in breast cancer prediction. The proposed system consists of two phases [28]. Initially, Fuzzy-Artificial Immune System (FAIS) was performed for dimensionality reduction. Furthermore, K-Nearest Neighbor applied for classification in the selected features from FAIS. In this manner, the efficiency of the prediction model increases and execution time also reduced. The proposed methodology obtained a high validation accuracy of 99.14% via 10-fold cross-validation.

Nahato et al. developed a classifier that learns attributes from clinical datasets to predict disease. After processing the missing values, a roughly defined and indistinguishable relational method is used that uses an upstream propagation neural network [24]. This classification was used in the hepatitis, Wisconsin breast cancer and heart disease statlog dataset collected by the University of California, Irvine (UCI). The accuracy of hepatitis, heart disease, breast cancer and heart disease was 97.3%, 90.4% and 98.6%, respectively.

Aaron and Taghi presented a cloud-based approach to learning from electronic health record data and demonstrated its usefulness in predicting melanoma risk [27]. For data preprocessing and markup, we used hybrid distributed processing with Apache Spark and undistributed processing with scikit-learn to test machine learning models. In addition, they showed the performance-enhancing effect of training dataset samples. The success of the predictions was achieved by a gradient boost classifier, cross-validated cross-validation as specificity of 0.688, AUC of 0.799 and sensitivity of 0.753.

Wang et al. [32] proposed an automatic image processing system for classification of breast cell nuclei. The authors utilized breast cell histopathology (BCH) images for breast cancer prediction. In this study, BCH images are segmented

with wavelet decomposition and multi-scale region growing techniques. The hybrid intelligent technique, which includes support vector machine and genetic algorithm used for extract optimal features (4 shape-based and 138 texture-based features). It has achieved 96% classification accuracy with 68 BCH images. However, to demonstrate the applicability of the proposed system, assessment on a huge dataset is preferred.

Nilashi et al. [25] introduced a knowledge-based system, which combination of Expectation Maximization to cluster the data into related groups and Classification and Regression Trees (CART) to produce the fuzzy rules to be used for the prediction of breast cancer. Additional, to overcome the multicollinearity issue, Principal Component analysis (PCA) is incorporated with the proposed system. This technique demonstrations decent accuracy on mammography mass datasets. Apache spark-based tree classifiers are not applied to predict the breast cancer problem. This motivated to apply it to breast cancer prediction.

According to the knowledge gained from the previous related works, it is found that Random Forest is a widely used classification algorithm in breast cancer prediction. And it is also found that faster schema for random forest is achieved using the Apache Spark paradigm. Thus, in this thesis, breast cancer prediction is performed by using Random Forest and the Random Forest algorithm is speeded up by using the Apache Spark paradigm.

CHAPTER 3

DESIGN OF THE SYSTEM

The main target of this thesis is to develop a high-performance and scalable breast cancer prediction system on the big data analytics platform, Apache Spark. This model is developed on Apache Spark Framework using Machine Learning and the used data source is Wisconsin Diagnosis Breast Cancer Dataset taken from the University of California at Irvine (UCI) Machine Learning Repository Set which categorizes breast tumor cases as either benign or malignant based on 10 features to predict the diagnosis. Performance evaluation of a machine learning classifier is made using the holdout method. The cancer prediction system will help medical professionals reach conclusions based on the clinical data of patients and also perform with high levels of accuracy.

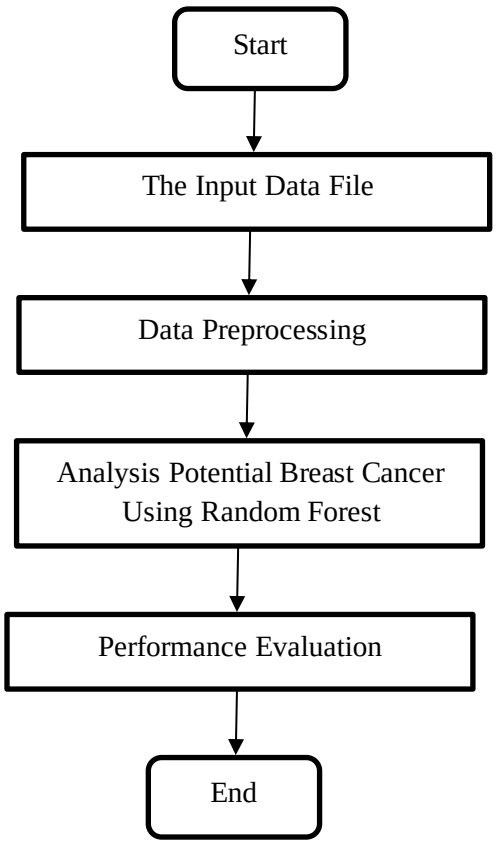

Figure 3.1 System Flow Diagram

3.1 Overview of The Proposed System

This system proposes a model developed with spark machine learning pipeline that processes voluminous data fast and accurate. Original cancer data set of UCI Machine learning repository is used as input for analysis. Apache Spark is a scalable, fault tolerant in memory computing engine that handles big data. It provides rich library to implement machine learning algorithms in an effective manner. In Machine learning, Pipeline is used to chain the prediction process which executes continuously. Apache Spark an open-sourced cluster computing framework.

This framework is built at the top of the Hadoop Distributed File System (HDFS). It performs distributed data processing and distributes data for the separation of worker nodes to perform processing. A master node performs the management of worker nodes by dispatching and scheduling of distributed tasks. Therefore, a cluster manager and distributed storage system is needed by this framework. This framework has faster in-memory data engine and developer-friendly API that provides in the choice of this framework.

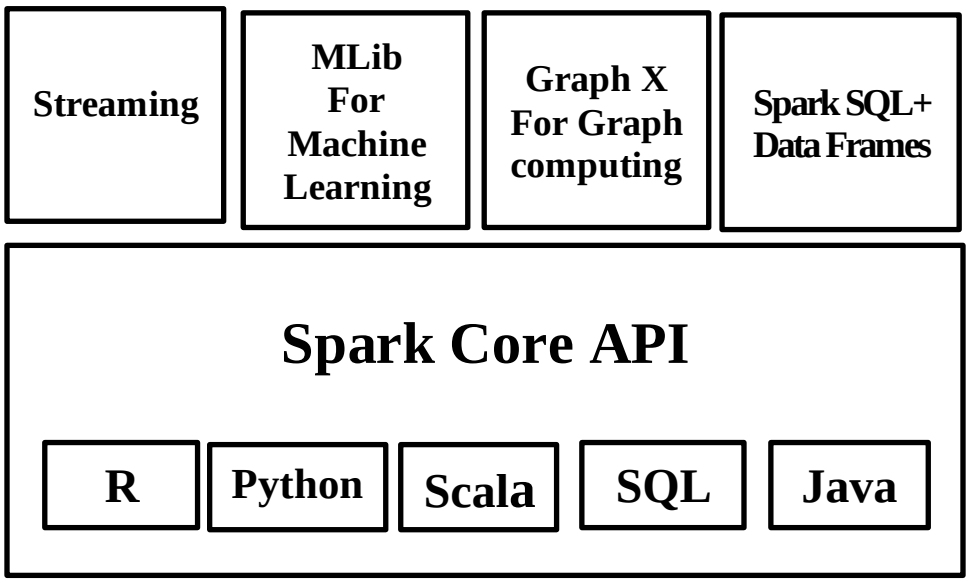

Figure 3.2 Apache Spark Ecosystem [36]

The foundation of Apache spark, Spark Core, works as execution engine. The scheduling, I/O functionalities and dispatching of distributed tasks is supported by spark core. It is possibly performed by an application programming interface (API) in

R, Scala, Java, and Python. This can support a wide variety of languages with interfaces available for many languages. It supports faster speed utilization of in-memory computing capabilities. This API provides higher-level programming method by a driver program which performs the calling of parallel operations on a Resilient Distributed Dataset (RDD) with function passing to Spark. The parallel execution in the available clusters by scheduled by this core. Resilient Distributed Dataset (RDD) was the primary API until the version of spark 2.1.x. It is fault-tolerant and read-only collection of datasets which is distributed over the cluster machines to perform processing. The dataset API is enhanced even though the still use of RDD starting from spark version 2.2.x. Broadcast variables that aids in sharing of common read-only data among clusters and accumulators that provides reduction of program are used by spark than the use of RDD [37].

The four components of spark core are Spark Streaming, Spark SQL, Machine Learning library and GraphX. The module of Spark, Spark SQL is associated with structured data that utilized data frames. Spark SQL possess an interface as SQL for query data processing. The addition of real-time data processing to batch processing of Spark is performed by Spark Streaming. The incoming data stream is broken down into micro batches and its processing is same as batch processing. For graph structures processing, Graph X is a distributed framework at the top of Spark. Users are allowed for the building and processing of interactive graph structured data. The distributed implementation of the machine learning pipelines is allowed by the machine learning library with the significant decrement of the overall processing time.

Many machine learning algorithms such as classification, clustering, collaborative filtering, regression and dimensionality reduction are implemented in a distributed manner. These algorithms perform the easy execution of feature extraction, selection, and transformation on structured dataset. Moreover, they support tools for constructing, tuning ML pipelines and evaluating along with loading and keeping algorithms, pipelines and models [35].

For Spark-based Random Forest training, the training set is loaded into the Hadoop File System (HDFS). Then, the training is performed on Hadoop with the loaded training set. After model training, the generated trained model is saved in HDFS and then the copy is sent to the local file system for performance evaluation. After the training is completed, a performance evaluation is performed. For performance

implementation, the testing set from a random split of the dataset is done. After the testing data is loaded, the target class of testing data is predicted using a random forest model. The testing of the proposed system calculates the output values by using the trained model values. The proposed system' performance evaluation will measure by accuracy, precision, recall, and f-measure. The details of performance evaluation are described in the next section.

Big Data is a huge amount of data that is cumbersome in traditional systems and comes in the form of unstructured, structured and partially structured data. Machine learning is a concept of implementing algorithms to train systems comparable to human learning. Pipelines are machine learning standards that can chain linear sequences of data transformations to evaluate the modeling process. This ensures that all stages of the pipeline evaluate available data, such as training records and cross-validation processes.

The Spark Machine Learning library provides a standard application programming interface to facilitate the implementation of machine learning algorithms. By combining multiple algorithms, it offers the ability to develop pipelines and workflows. Each ML pipeline contains the following components:

- **Data Frame:** This is the basic component of the spark SQL that handles columned data. It is used to store feature vectors, labels, predictions and data. Spark ML API can work with data frames.
- **Transformer:** This is a ML algorithm that transform data frame features into predictions.
- **Estimator:** A ML learning algorithm that trains the data frames to generate the model
- **Pipeline:** A process that creates a workflow by chaining multiple estimators and transformers.
- **Parameters:** A parameter set that can be shared by transformers and estimators.

The Pipeline process consists of four phases:

- **Data Ingestion:** In this phase, input data loaded into spark system and converted into data frames.
- **Data Preparation:** Input data may not be proper and may contain missing values. In this phase, prepare the proper input data for the model by eliminating

or predicting the missing values. Output data is used to build and train the model.

- **Train and build Model:** In this phase, features are added to data frames and transform them to predictions with data frames by training the model.
- **Predictions:** In this phase we evaluate the predictions from the model using train data and test data.

In this system, Wisconsin Diagnostic data from Breast Cancer Data Set of UCI machine learning repository is used for the ML algorithms which predict breast cancer [38]. This dataset is selected mainly for it is from a reliable source and is publicly available real-world breast cancer data set. This dataset contains tumor features acquired from a Digital image of breast Fine Needle Aspirates (FNA). Ten real-valued features are computed for each cell nucleus as follows: Every patient, 10 attributes of cell nuclei (noticeable in DFNA) are gathered: radius, texture, perimeter, area, smoothness, compactness, concavity, symmetry, concave points and fractal dimension. The features given are the characteristics of the cell nuclei computed from the fine needle aspirate (FNA) of a breast mass. The dataset is about the patients who were detected with 2 kinds of breast cancer: a) Malignant or b) Benign.

This dataset contains following attributes. (a)Clump thickness: Cancer cells are accumulated in multiple layers where normal cells are grouped in monophonic layers. (b) Uniformity of cell size/shape: Cancer cell differs in size and shape. This attribute is use to identify that whether the sample is cancerous or not. (c) Marginal adhesion: Normal cells tend to penetrate together whereas cancer cells cannot penetrate. (d) Epithelial cell size: Epithelial cells that are significantly enlarged if it is impacted by cancer (e) Bare nuclei: This value is available only in normal cells. (f) Bland Chromatin: It deals with the surface of sample cell. If the cell is cancerous the surface will be rough where it is softer for normal cell. (g) Normal nucleoli: The small visible structures of nucleus are called Nucleoli. This Nucleoli is too small and invisible in normal cell whereas very noticeable in cancerous cells.

In Data preprocessing, the conversion of unstructured data into structured data is performed. The steps in data preprocessing are

- Reading the dataset
- Check for missing values and fill with required data
- Splitting data into dependent and independent data.

- Label encoding
- One hot encoding (Binarization)
- Splitting the independent and dependent values into train and test sets.

After data preprocessing, the prediction of breast cancer is computed through our proposed model Apache Spark-Based Random Forest. The Steps of Prediction Analysis using Random Forest in Spark are:

- Input Pre-processed dataset in the form of an RDD
- Convert RDD to Data Frame (DF)
- Read Features and Labels from DF
- One Hot Encoding of the non-numeric features
- String Indexing of each encoded feature
- Vector assembly of one-hot-encoded features and numeric features
- Convert the assembled vector into a Pipeline
- Fit and Transform the Pipeline into a suitable form for Spark to read
- Train the model using random forest-based features using the training data
- Test on the whole data to obtain prediction value of the label

Random forest is a supervised learning algorithm that creates a forest randomly. This forest, is a set of decision trees, most of the times trained with the bagging method. The essential idea of bagging is to average many noisy but approximately impartial models, and therefore reduce the variation. This is one kind of ensemble learning algorithm. It's a randomized ensembles of decision trees. The introductory unit is a binary tree that's constructed using recursive partitioning (RPART). This is generally grown by the methodology of CART, an approach in that the binary splitting by recursively partition of the tree into near homogeneous terminal or homogeneous nodes. The splitting of data from the parent tree into its son nodes is done by a good binary tree such that the unity enhancement in the son nodes is done from the parent.

This is constructed with hundreds to thousands of trees, in where each tree is constructed with a bootstrap sample of the original data. It differs from CART as they're constructed non-deterministically using a two- stage randomization procedure. It considers only a subset of features. Also, it has same hyperparameters as a decision tree. Advantages are that it performs very effectively on large scale dataset. It can handle for both classification and regression problems. It becomes a better model by

adding more randomness to the model. Each tree is constructed using the following algorithm:

- Let N be the number of test cases, M is the number of variables in the classifier.
- Let me be the number of input variables to be used to determine the decision in a given node; m<**M**.
- Choose a training set for this tree and use the rest of the test cases to estimate the error.
- For each node of the tree, randomly choose m variables on which to base the decision. Calculate the best partition of the training set from the m variables.

For prediction a new case is pushed down the tree. Then it is assigned the label of the terminal node where it ends. This process is iterated by all the trees in the assembly, and the label that gets the most incidents is reported as the prediction.

The step of Random Forest prediction is as follows:

- Step 1: Read the input data set for "n" features.
- Step 2: Choose subset of features and name it as "i" from "n" features randomly
- Step 3: Compute the node "n" among "i" features base on finest fit
- Step 4: Base on best split, divide the node into child nodes.
- Step 5: Repeat 2-4 steps until got "j" nodes i.e., trees
- Step 6: Repeat 2-5 step for "m" times to get "m" number of trees to create a forest.
- Step 7: Test features and generated discussion trees are used for prediction. This is stored as target
- Step 8: Calculate the generalized value called vote for each predicted target
- Step 9. The high voted target is considered as final prediction.

3.2 Evaluation of the Performance of Methods

In all learning algorithm, evaluating performance is a fundamental aspect. A measurement is needed to determine the effectiveness of the learning algorithm used for a system. Not only it is important in order to compare competing algorithms, but in much case is an integral part of the learning algorithms itself. Estimating classifier accuracy is important in that it allows one to evaluate how accurately a given classifier will label future data. For determining the effectiveness of the algorithm, commonly

used measurements include classification accuracy, F-Measure, precision, and recall. These measurements can be calculated by the classification results commonly tabulated in a matrix format called a Confusion Matrix. Classification accuracy is defined as the percentage of the examples correctly classified by the algorithm. Bootstrap, Holdout, and Cross-validation methods are common techniques for accessing classifier accuracy based on randomly sampled partitions of the given data.

3.2.1 Confusion Matrix Performance

In a classic binary classification problem, the classifier labels the items as either positive or negative. A confusion matrix summarizes the outcome of the algorithm in a matrix format. In our binary example, the confusion matrix would have four outcomes:

True positives (TP): These refer to the positive tuples that were correctly labeled by the classifier. Let *TP* be the number of true positives.

True negatives (TN): These are the negative tuples that were correctly labeled by the classifier. Let *TN* be the number of true negatives.

False positives (FP): These are the negative tuples that were incorrectly labeled as positive. Let *FP* be the number of false positives.

False negatives (FN): These are the positive tuples that were mislabeled as negative. Let *FN* be the number of false negatives.

These terms are summarized in the confusion matrix of table 3.1. The confusion matrix is a useful tool for analyzing how well your classifier can recognize tuples of different classes. TP and TN tell the user when the classifier is getting things right, while FP and FN tell the user when the classifier is getting things wrong (i.e., mislabeling). Given m classes (where m = 2), a confusion matrix is a table of at least size m by m. An entry, CMi,j in the first m rows and m columns indicates the number of tuples of class i that were labeled by the classifier as class j.

Table 3.1 Confusion Matrix

Confusion Matrix		Predicted class:		
		Positive	Negative	Total
Actual Class	Positive	TP	FN	P
	Negative	FP	TN	N
	Total	P’	N’	P+N

For a classifier to have good accuracy, ideally most of the tuples would be represented along the diagonal of the confusion matrix, from entry $CM_{1,1}$ *to entry* $CM_{m,m}$, with the rest of the entries being zero or close to zero. That is, ideally, *FP* and *FN* are around zero.

The simplest performance measure is accuracy. The overall effectiveness of the algorithm is calculated by dividing the correct labeling against all classifications. The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier. That is,

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \qquad 3.1$$

The accuracy determined may not be an adequate performance measure when the number of negative cases is much greater than the number of positive cases.

The *precision* and *recall* measures are also widely used in classification. **Precision** can be thought of as a measure of *exactness* (i.e., what percentage of tuples labeled as positive are actually such), whereas **recall** is a measure of completen/ess (what percentage of positive tuples are labeled as such). If recall seems familiar, that’s because it is the same as sensitivity (or the *true positive rate*). These measures can be computed as

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \qquad 3.2$$

$$\text{Recall} = \text{Sensitivity} = \frac{\text{TP}}{\text{TP+FN}} \quad 3.3$$

F-Measure (Lewis and Gale, 1994) is one of the popular metrics used as a performance measure. The measure itself is computed using two other performance measures, precision and recall. Based on these definitions F-measure is defined as follows:

$$\text{F-measure} = \frac{2\times\text{precision}\times\text{recall}}{\text{precision+recall}} \quad 3.4$$

In essence, the F-Measure is the harmonic mean of the recall and precision measures.

3.2.2 Cross-validation Method

In k-fold cross-validation, the random partition of initial data is done into k mutually exclusive subsets or folds of approximately equal size. A learning algorithm is trained and tested k times; each time it is tested on one of the k-folds and trained using the remaining k-1 folds. The cross-validation estimate of accuracy is the overall number of correct classifications from the k iterations, divided by the number of examples in the initial data.

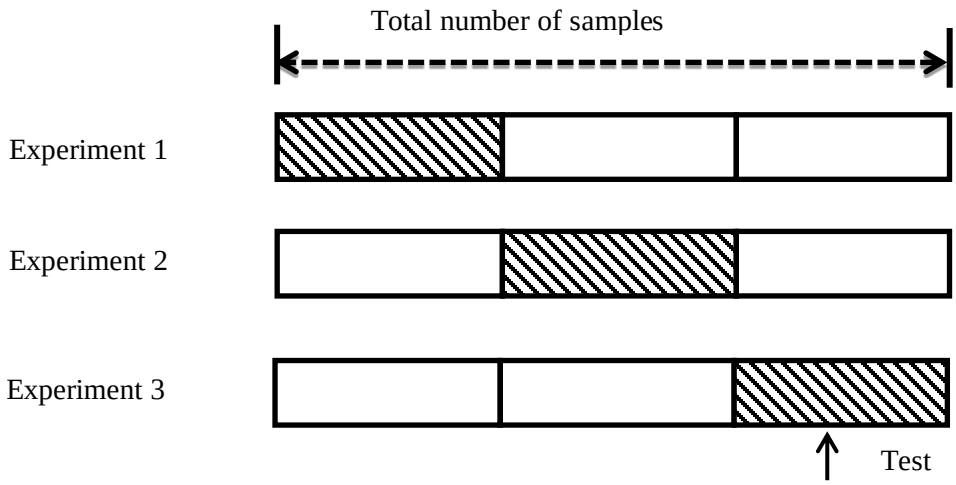

Figure 3.3 Cross-validation Method

CHPAPTER 4

IMPLEMENTATION OF THE SYSTEM

The main purpose of the chapter is to describe the experimental environment and implementation procedures of the proposed system. The performance of the proposed system has been evaluated on analytics of big data platform using "Apache Spark".

4.1 Experimental Setup

A Hadoop cluster (Pseudo Distributed Mode) is established on a Linux virtual machine in a VMWare workstation to implement the proposed system. In pseudo-distributed mode, the name node and data node reside on the same machine, and the master and slave servers run on this machine. The configuration is as follows:

The specification of host machine is as follows:

- Intel ® Core i7-8550U CPU @ 3.7GHz,
- 8GB Memory,
- 1TB Hard Disk

The software components for each virtual machine are as follows:

- MLlib Machine Learning Library
- Hadoop 3.2.2
- Flume 1.9.0
- Spark 3.1.2
- Scala 2.12.3

This system is implemented on the Apache Hadoop big data analytics platform, which offers good capabilities to analyze data for achieving a better understanding of the data. The proposed system using the Apache Spark architecture is described. There are four main parts:

- Data Collection
- Data Analytics
- Data Processing
- Data Storage

Data Ingestion Layer: In this layer, breast cancer data is collected by Apache Flume. For offline processing, the collected data is pushed into the HDFS.

Storage Layer: In the Storage Layer, HDFS is used to store scalable and reliable data. HDFS provides a master/slave architecture, and a single Name Node acts as the primary server. The Name Node performs the operations of the file system namespace, such as opening, closing, and renaming.

Processing Layer: Yarn Cluster Manager and Spark executives are in the processing layer to process large amounts of data in parallel on a product hardware cluster in a reliable and fault-tolerant manner. The YARN Resource Manager keeps track of the resources of each node manager YARN. The Node Manager manages resources on the slave node. Each slave node may have one or many resources available at the same time, and each executor can have a work process. And, as planned, each task is carried out in a separate JVM managed by the operator. The data is loaded in terms of the RDD partition into multiple handlers and the conversion is applied to these RDD partitions. The code operates on a topic called "work. The manager is designed as a multi-threaded process and hence can run multiple tasks simultaneously.

Analytics layer: Offline training is implemented in the analytics layer. All of the processes from the Analytics Layer are executed in a distributed manner by using the Spark engine. Breast Cancer prediction is implemented by using Spark MLlib.

4.2 Implementation of the System

In this system, a breast cancer dataset from UCI is used. This system is implemented with dataset size 600,000 breast cancer data. This system is tested with data size ratio: (Training – 80%, Testing – 20%). Firstly, randomly selected 80% records as training data and 20 % records for testing from this dataset. The popular performance measures (accuracy, recall, f-measure, and precision) are evaluated for this proposed system analysis. The breast cancer dataset is shown in figure 4.2.

id	diagnosis	radius_mea	texture_mea	perimeter_r	area_mean	smoothness	compactnes	concavity_r	concave po	symmetry_r	fractal_dim
842302	M	17.99	10.38	122.8	1001	0.1184	0.2776	0.3001	0.1471	0.2419	0.07871
842517	M	20.57	17.77	132.9	1326	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667
84300903	M	19.69	21.25	130	1203	0.1096	0.1599	0.1974	0.1279	0.2069	0.05999
84348301	M	11.42	20.38	77.58	386.1	0.1425	0.2839	0.2414	0.1052	0.2597	0.09744
84358402	M	20.29	14.34	135.1	1297	0.1003	0.1328	0.198	0.1043	0.1809	0.05883
843786	M	12.45	15.7	82.57	477.1	0.1278	0.17	0.1578	0.08089	0.2087	0.07613
844359	M	18.25	19.98	119.6	1040	0.09463	0.109	0.1127	0.074	0.1794	0.05742
84458202	M	13.71	20.83	90.2	577.9	0.1189	0.1645	0.09366	0.05985	0.2196	0.07451
844981	M	13	21.82	87.5	519.8	0.1273	0.1932	0.1859	0.09353	0.235	0.07389
84501001	M	12.46	24.04	83.97	475.9	0.1186	0.2396	0.2273	0.08543	0.203	0.08243
845636	M	16.02	23.24	102.7	797.8	0.08206	0.06669	0.03299	0.03323	0.1528	0.05697
84610002	M	15.78	17.89	103.6	781	0.0971	0.1292	0.09954	0.06606	0.1842	0.06082
846226	M	19.17	24.8	132.4	1123	0.0974	0.2458	0.2065	0.1118	0.2397	0.078
846381	M	15.85	23.95	103.7	782.7	0.08401	0.1002	0.09938	0.05364	0.1847	0.05338
84667401	M	13.73	22.61	93.6	578.3	0.1131	0.2293	0.2128	0.08025	0.2069	0.07682
84799002	M	14.54	27.54	96.73	658.8	0.1139	0.1595	0.1639	0.07364	0.2303	0.07077
848406	M	14.68	20.13	94.74	684.5	0.09867	0.072	0.07395	0.05259	0.1586	0.05922
84862001	M	16.13	20.68	108.1	798.8	0.117	0.2022	0.1722	0.1028	0.2164	0.07356
849014	M	19.81	22.15	130	1260	0.09831	0.1027	0.1479	0.09498	0.1582	0.05395
8510426	B	13.54	14.36	87.46	566.3	0.09779	0.08129	0.06664	0.04781	0.1885	0.05766
8510653	B	13.08	15.71	85.63	520	0.1075	0.127	0.04568	0.0311	0.1967	0.06811

Figure 4.1 Breast Cancer Dataset

The attribute information in the dataset are:

1. ID number
2. Diagnosis (M = malignant, B = benign)

Ten real-valued features are computed for each cell nucleus:

a. radius (mean of distances from center to points on the perimeter)
b. texture (standard deviation of gray-scale values)
c. perimeter
d. area
e. smoothness (local variation in radius lengths)
f. compactness (perimeter^2 / area - 1.0)
g. concavity (severity of concave portions of the contour)
h. concave points (number of concave portions of the contour)
i. symmetry
j. fractal dimension ("coastline approximation" - 1)

The number of attributes is 10, with a single class representing the dependent variable making it 11 fields, which is the estimation outcome of the machine-learning algorithm. All values of the attributes in the domain are numerical as presented in Table 4.1. Class value applicable in the distribution of Benign and Malignant used for the breast cancer prediction is shown in Table 4.1 where class value 0 is interpreted as "Benign" that signifies Non-invasive. The type of breast cancer that don't develop into or attack normal tissues inside or beyond the breast while class value 1 is interpreted as "Malignant" that signifies any of Invasive, Metastatic or Intrinsic type of breast cancer.

Table 4.1. Attributes of Breast Cancer Dataset

Number	Attribute	Domain
1	Sample code number	id number
2	Clump Thickness	1 - 10
3	Uniformity of Cell Size	1 - 10
4	Uniformity of Cell Shape	1 - 10
5	Marginal Adhesion	1 - 10
6	Single Epithelial Cell Size	1 - 10
7	Bare Nuclei	1 - 10
8	Bland Chromatin	1 - 10
9	Normal Nucleoli	1 - 10
10	Mitoses	1 - 10
11	Class	(0 for benign, 1 for malignant)

4.3 Apache Spark-based Random Forest Prediction

This system develops an Apache Spark-based model for processing voluminous data in order to be accurate and fast. Original cancer data set of UCI Machine learning repository is used as input for prediction. Apache Spark is a fault tolerant and scalable in memory computing engine for solving big data. It supports rich library for machine learning algorithms implementation with an effective manner. Pipeline is utilized for the chain of the continuous prediction process in Machine learning. Data preprocessing is performed by substituting missing values from the input dataset. After data preprocessing, the data storage is done on HDFS with the splitting into training and testing sets. 80% of the dataset is used as the training and the remaining 20% is used as

the testing. Random Forest can be trained using the training set on a local file system. For Spark-based Random Forest training, the training set is loaded into the Hadoop File System (HDFS). Then, the training is performed on Hadoop with the loaded training set. After model training, the generated trained model is saved in HDFS and then the copy is sent to the local file system for performance evaluation.

The actions are performed in order to execute the Apache Spark-based Random Forest program in Hadoop. Firstly, the namenode is formatted by typing the command: hdfs namenode-format. After that, all of the daemons of Hadoop are needed to run by typing the "start-all.sh" script in a Linux terminal. If all of the Hadoop daemons are running successfully, the program can be run in Hadoop. Then, the condition of namenode is checked by typing http://localhost:9870/ from the web browser.

After that, all of the daemons of Spark are needed to run by typing the "start-master.sh" script in a Linux terminal. If all of the Spark daemons are running successfully, the program can be run in Spark.. Then, the condition of Spark is checked by typing http://127.0.0.1:8080/ from the web browser.

To run the program in Hadoop, the input directory is needed to be created in HDFS by the commands "hadoop fs –mkdir/createdirectoryname". The file that the user wants to process needs to be uploaded from the local machine to HDFS by specifying the file location. To achieve that, the command "hadoop fs –put /inputfilelocation/data.csv/createdirectoryname" is run, which copies the data from the local file system to the folder of /createdirectoryname/in HDFS.

4.4 Evaluation of Experimental Results

In this system, breast cancer dataset from UCI is used. This system is implemented with dataset size 600,000 breast cancer data. This system is tested with data size ratio: (Training – 80%, Testing – 20%). Firstly, randomly selected 80% records as training data and 20 % records for testing from this dataset. This breast cancer prediction system is performed with Apache Spark-based Random Forest Model. Then, the breast cancer prediction system is performed with Apache Spark-based Naïve Bayes to evaluate Random Forest Classifier outperforms than Naïve Bayes in prediction. The popular performance measures (accuracy, recall, f-measure, and precision) are evaluated for this proposed system analysis.

Machine Learning Algorithm	Benign			Malignant		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Random Forest	0.99	0.98	0.98	0.96	0.99	0.97

Figure 4.2 Performance Results of Random Forest

Figure 4.7 shows the results for the performance of naïve bayes-based classifier for type Benign and Malignant in breast cancer prediction.

Machine Learning Algorithm	Benign			Malignant		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Naïve Bayes	0.91	0.94	0.92	0.89	0.84	0.86

Figure 4.3 Performance Results of Naïve Bayes

The comparative results for the performance of proposed random forest-based classifier and naïve bayes classification for type: Benign are illustrated in Figure 4.8.

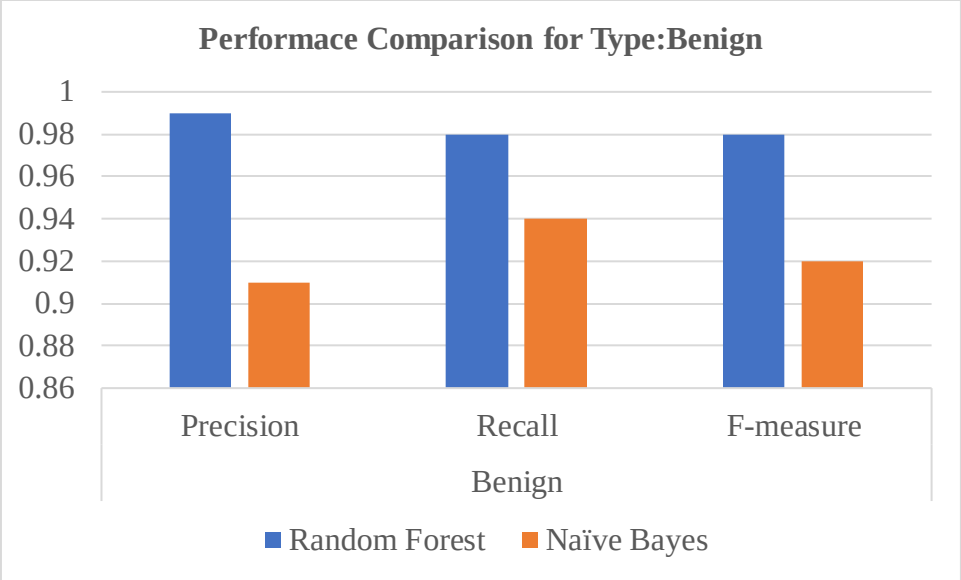

Figure 4.4 Performance Comparison of Random Forest and Naïve Bayes for Type: Benign

The comparative results for the performance of proposed random forest-based classifier and naïve bayes classification for type: Malignant are illustrated in Figure 4.9.

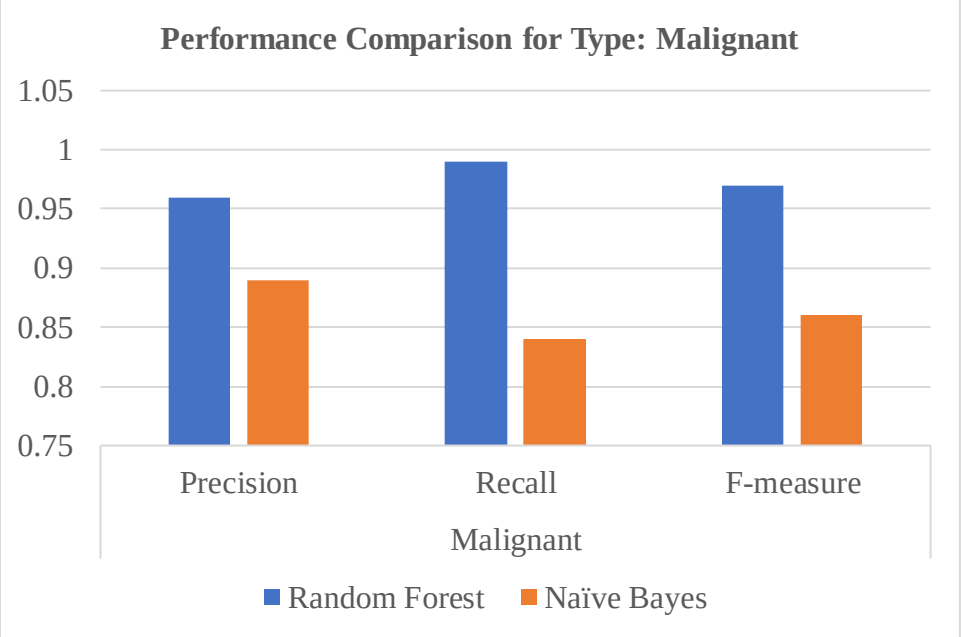

Figure 4.5 Performance Comparison of Random Forest and Naïve Bayes for Type: Malignant

The comparative results for the performance of proposed random forest-based classifier and naïve bayes classification are illustrated in Figure 4.10.

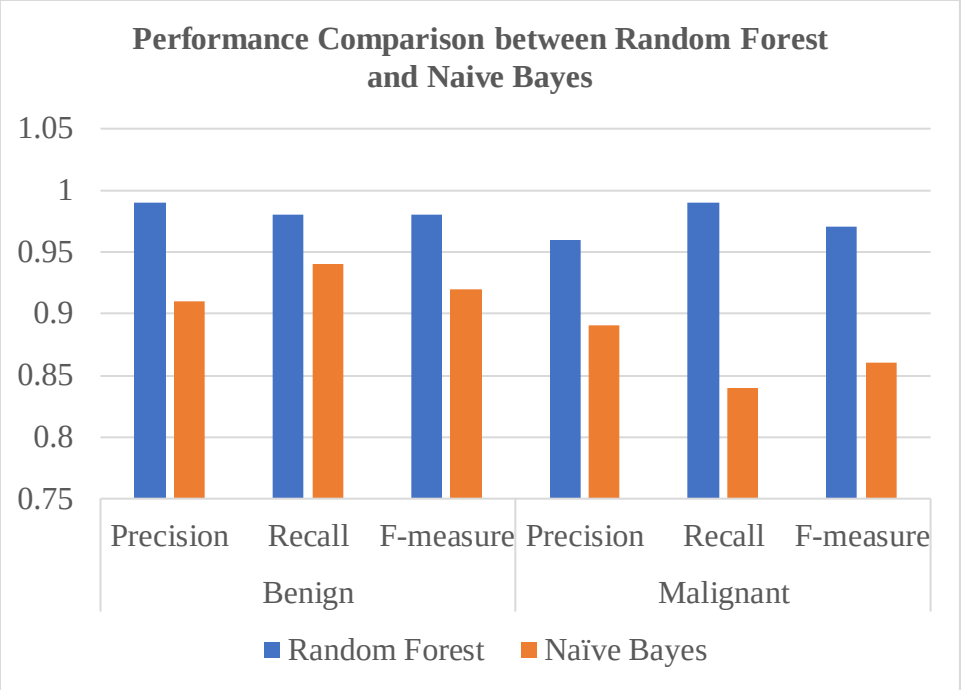

Figure 4.6 Performance Comparison between Random Forest and Naïve Bayes

The comparisons of accuracy between Random Forest and Naïve Bayes on breast cancer dataset are illustrated in Figure 4.11 and Figure 4.12.

Machine Learning Algorithm	Accuracy (%)
Random Forest	98.1
Naïve Bayes	90.5

Figure 4.7 Accuracy Results of Random Forest and Naïve Bayes

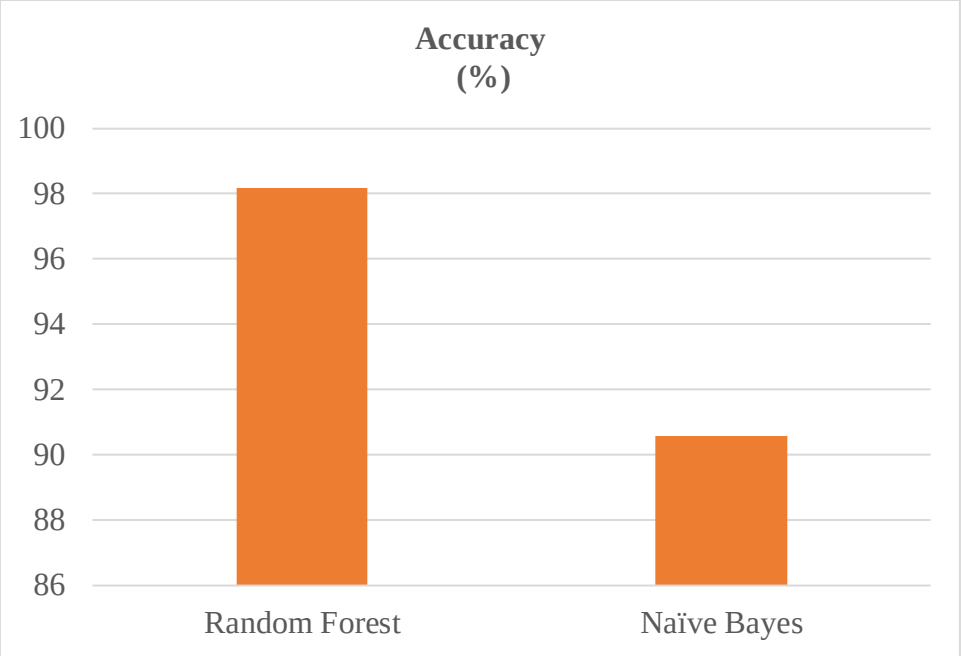

Figure 4.8 Accuracy Comparisons of Random Forest and Naïve Bayes

According to the evaluation results, Random Forest classifier with proposed breast cancer classification model achieves the best optimal accuracy than Naïve Bayes classifier. To run the program the command is typed by "**python3 main.py**". After that the first page is found as following in Figure 4.9. The corresponding files are uploaded as shown in Figure 4.10. Then, the results files are described in Figure 4.11. by clicking the Next button.

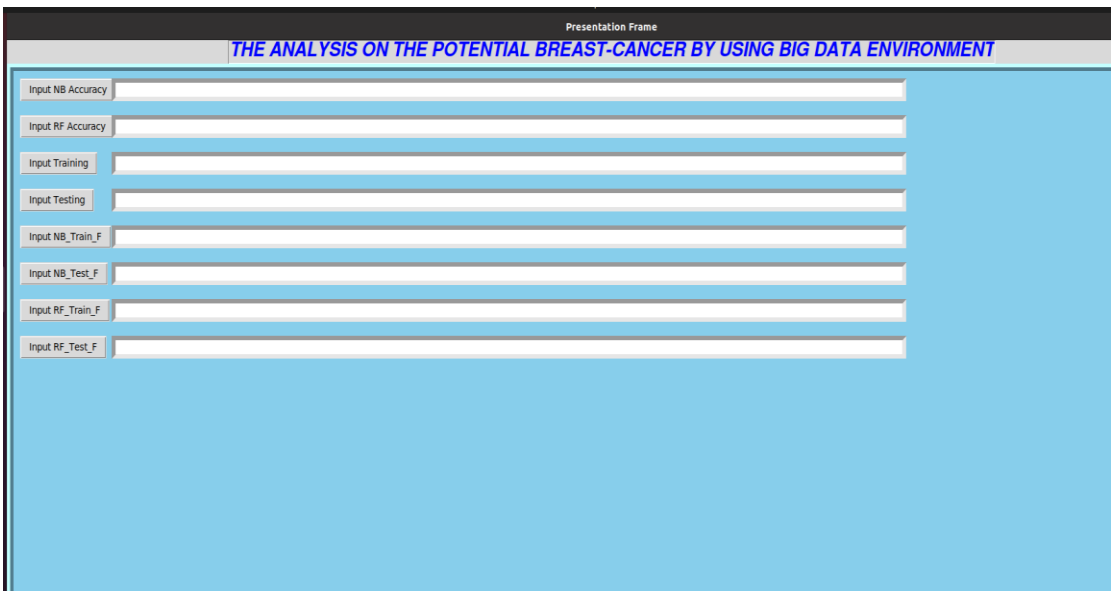

Figure 4.9 Main Page

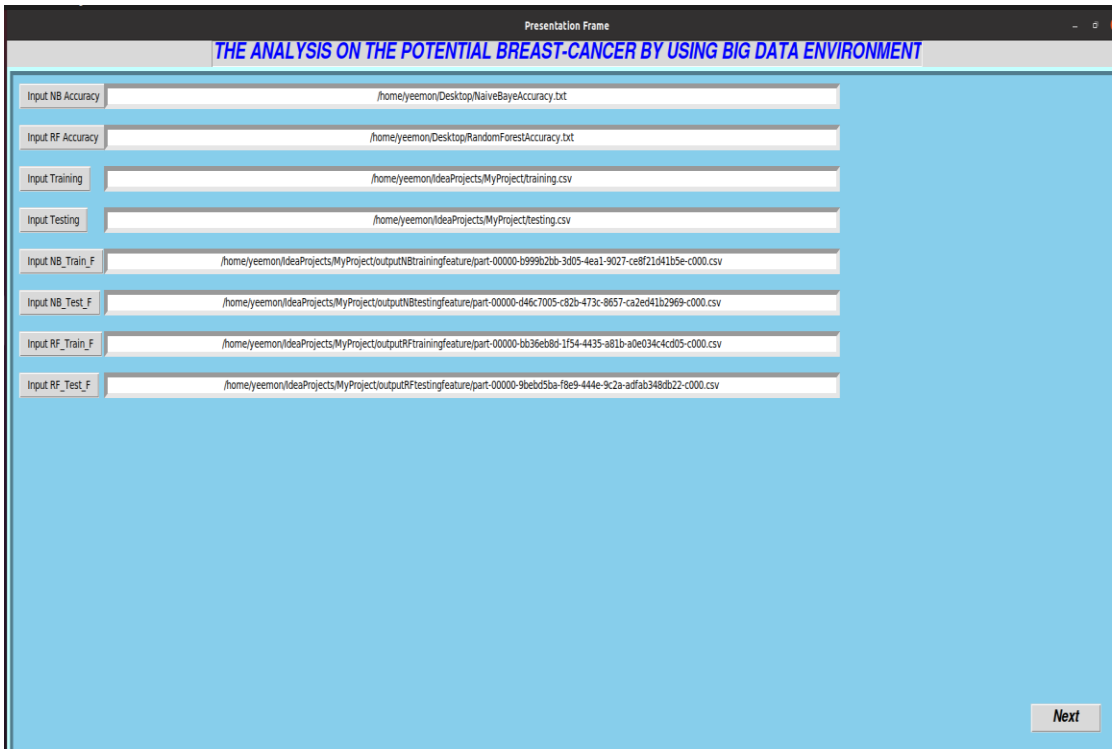

Figure 4.10 Loading Files

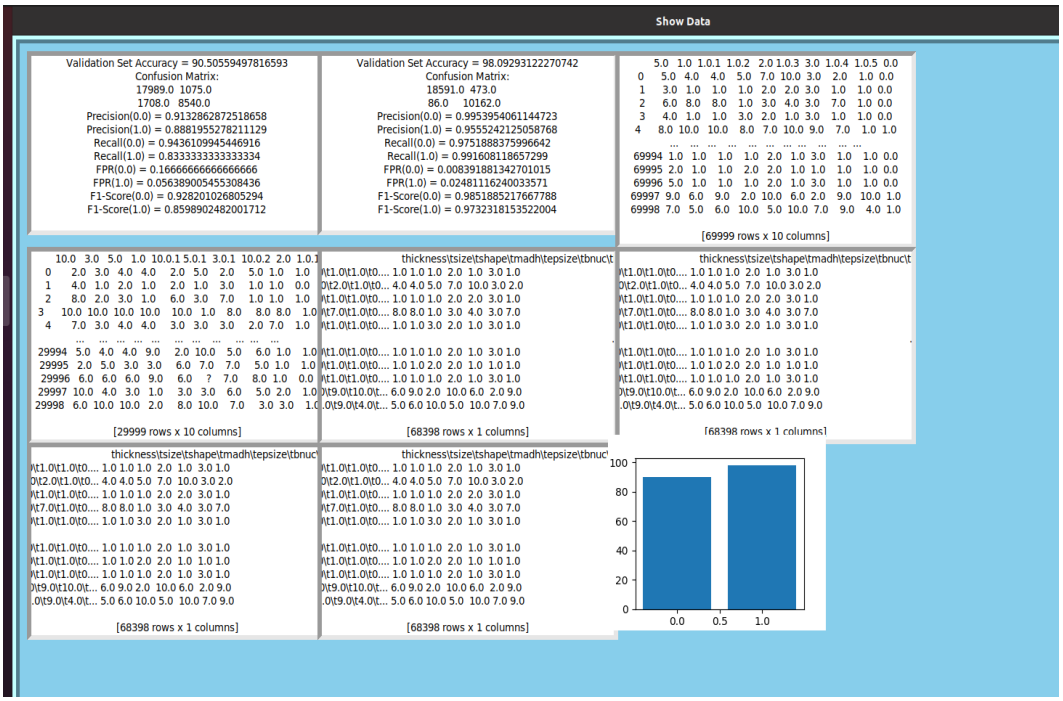

Figure 4.11 Comparison Results between Random Forest and Naïve Bayes

CHAPTER 5

CONCLUSION

Computer-based diagnosis systems are playing an increasingly important role in health care facilities. They may improve the quality of the diagnosis process in accuracy and efficiency, and the patients can save money and time. The automatic diagnosis of breast cancer is an essential real-world medical problem. Detection of breast cancer in an early stage is the key to treatment. Breast cancer is the second most common cancer in women in the world. Among the other kinds of diseases, breast cancer causes a greater number of deaths in many countries. The earlier notification of breast cancer provides a treatment chance; therefore, a massive amount of observation is performed for the identification of breast cancer in its early steps. Many tree-based machine learning algorithms will not have the ability to solve large amounts of complex data. This issue is addressed through efficient tree-based classifiers like the Random Forest classifier with the Apache Spark framework.

This proposed prediction model is adaptable to classical machine learning algorithms with data generally, and forecasting methods are possible for achieving a result with the phenomenon of big data. In order to improve the survivability rate among patients with breast cancer, the system is implemented for the breast cancer prediction method. In this system, a spark-based random forest approach is used for breast cancer prediction. The proposed approach is evaluated and compared using the Wisconsin breast cancer dataset. The advantage of the proposed system is that the computational complexity and response time are reduced compared to traditional machine learning approaches as the implementation on the big data analytics platform, Apache Spark. The experimental outcomes also demonstrate the superiority of the random forest classifier in terms of accuracy, recall, precision, and f-measure.

5.1 Restrictions and Future Expansions

The comparative study has few limitations. The system is proposed for the forecasting of only two stages of breast cancer, and the user must know the symptoms of the breast cancer. The system is implemented only by using the Random Forest algorithm and the Nave Bayes method. The future work will extend. Furthermore, the system implementation is performed only for the pseudo-distributed mode in offline forecasting. As a future work, the system implementation will be done for the fully distributed mode cluster in online forecasting to provide the experiments with huge datasets.

AUTHOR'S PUBLICATIONS

[1] Yee Mon Ei, Thida Aung, "Breast Cancer Predictive Analytics Using Big Data Environment", Parallel & Soft Computing, University of Computer Studies, Yangon, 2022.

REFERENCES

[1] P. Across and H. Hardware, “The Hadoop Distributed File System: Architecture and Design,” pp. 1–14, 2007.

[2] A. Alexandrov, D. Battré, S. Ewen, M. Heimel, et al., “Massively Parallel Data Analysis with PACTs on Nephele”, In Proceedings of the VLDB Endowment, Vol. 3, No. 2, 2010, pp. 1625–1628.

[3] A. Alexandrov, S. Ewen, M. Heimel, F. Hueske, et al., “MapReduce and PACT – Comparing Data Parallel Programming Models”, In Proceedings of the 14th Conference on Database Systems for Business, Technology, and Web (BTW 2011), Bonn, Germany, February 28- March 4, 2011, pp. 25-44.

[4] S. Banumathi, A. Aloysius, "Predictive Analytics Concepts in Big Data- A Survey", In International Journal of Advanced Research in Computer Science, Vol. 8, No. 8, September 2017.

[5] D. Battré, S. Ewen, F. Hueske, O. Kao, et al., “Nephele/PACTs: A Programming Model and Execution Framework for Web-Scale Analytical Processing”, In Proceedings of the 1st ACM Symposium on Cloud Computing (SoCC ’10), June 10-11, 2010, Indianapolis, IN, USA, pp. 119-130.

[6] C. T. Chu, S. K. Kim, Y. A. Lin, Y. Yu, et al., “Map-Reduce for Machine Learning on Multicore”, Advances in Neural Information Processing Systems (NIPS ’06), MIT Press, 2006, pp. 281-288.

[7] J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters”, In Proceedings of the 6th Symposium on Operating Systems Design and Implementation, San Francisco, CA, USA, December 6-8, 2004, pp. 137-149.

[8] J. Dean and S. Ghemawat, “MapReduce: A Flexible Data Processing Tool”, Communications of the ACM, Vol.53, No.1, January 2010, pp. 72-77.

[9] V. Deshwal and M. Sharma, “Breast Cancer Detection using SVM Classifier with Grid Search Technique”, International Journal of Computer Applications (0975 – 8887), Volume 178 – No. 31, July 2019.

[10] M. J. Domingo, B. D. Gerardo, and R. P. Medina, “Fuzzy Decision Tree for Breast Cancer Prediction”, In Proceedings of 2019 International Conference on Advanced Information Science and System (AISS’19). Singapore.

[11] J. Ekanayake, S. Pallickara, and G. Fox, “MapReduce for Data Intensive Scientific Analyses”, In Proceedings of the IEEE 4th International Conference on eScience (eScience’08), Washington, DC, USA, December 7-12, 2008, pp. 277-284.

[12] T. Eswari, P. Sampath, and S. Lavanya, "Predictive methodology for diabetic data analysis in big data", Procedia Computer Science, 2015, pp. 203-208.

[13] D. Fesehaye and N. G. Ave, “A Scalable Distributed File System for Cloud Computing,” 2010.

[14] C. Florina, G. Elena, “Perspectives on Big Data and Big Data Analytics”, 2013.

[15] A. Ganapathiraju," Support Vector Machines for Speech Recognition", Ph.D. Dissertation, Mississipi State University, Mississippi State, Mississippi, USA, 2002.

[16] J. Gentile, “Hadoop MapReduce”, June 28, 2012. [Online]. Available: http://bigdata.globant.com/. [Accessed: 14-January-2014].

[17] S. Ghemawat, H. Gobioff, and S.-T. Leung, “The Google file system,” ACM SIGOPS Oper. Syst. Rev., vol. 37, no. 5, p. 29, Dec. 2003.

[18] J. Han, and M. Kamber, “Data Mining, Concept and Techniques”, Sixed edition, Morgan Kaufmann Publishers, 2006, ISBN 1-55860-494-8.

[19] A. Heise, A. Rheinl¨ander, M. Leich, F. Naumann, et al., “Meteor/ Sopremo: An Extensible Query Language and Operator Model”, In Proceedings of the International Workshop on End-to-end Management of Big Data in conjunction with VLDB 2012, Istanbul, Turkey, 2012, August 27-31, 2012.

[20] M. Isard, M. Budiu, Y. Yu, A. Birrell, et al., “Dryad: Distributed Data-Parallel Programs from Sequential Building Blocks”, In Proceedings of the European Conference on Computer Systems (EuroSys), Lisbon, Portugal, March 21-23, 2007, pp. 59-72.

[21] M. Isard and Y. Yu, “Distributed Data-Parallel Computing Using a High-Level Programming Language”, In Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD 2009), Providence, Rhode Island, USA, June 29- July 2, 2009, pp. 987–994.

[22] N. Marz, “Storm: Distributed and Fault-Tolerant Realtime Computation”, 3-January-2012. [Online]. Available: [Accessed: 14-January-2014].

[23] C. Meng, W. Ye, and M. Ping, "Effective statistical methods for big data analytics", In Handbook of Research on Applied Cybernetics and Systems Science, IGI Global, 2017, pp. 280-299.

[24] K. Nahato, K. Harichandran, and K. Arputharaj, “Knowledge Mining from Clinical Datasets Using Rough Sets and Backpropagation Neural Network”, Computational and Mathematical Methods in Medicine. 2015, 2015:1-13.

[25] M. Nilashi, O. Ahmadi, and L. Shahmoradi, “A knowledge-based system for breast cancer classification using fuzzy logic method. Telematics and Informatics, 34(4), 133-144, 2017.

[26] R. Revathy and R. Lawrance, "Comparative Analysis of C4.5 and C5.0 Algorithms on Crop Pest Data", International Journal of Innovative Research in Computer and Communication Engineering, no., vol. 5, pp. 50-58, 2017.

[27] A. Richter, and T. Khoshgoftaar, “Efficient learning from big data for cancer risk modeling: A case study with melanoma”, 2020.

[28] S. Sahana, K. Polat, H. Kodaz, and S. Günes, “A new hybrid method based on fuzzy-artificial immune system and knn algorithm for breast cancer diagnosis”, Computers in Biology and Medicine, 377, 415-423 ,2007.

[29] B. Sathiyabhama, S. U. Kumar, and J. Jayanthi, “Spark based Framework for Breast Cancer Analysis”, Proceedings of the International Conference on Intelligent Computing Systems (ICICS 2017), December 2017.

[30] D. Setrakyan, “GridGain and Hadoop: Differences and Synergies”, 20-November-2012. [Online]. Available: [Accessed: 14-January-2014].

[31] H. Trevor, “The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations”. Tibshirani, Robert., Friedman, J. H. (Jerome H.). New York: Springer. ISBN 0-387-95284-5. OCLC 4680922, 2001.

[32] P. Wang, X. Hu, and X. Zhu, “Automatic cell nuclei segmentation and classification of breast cancer histopathology images”, Signal Processing, 122, 1-13, 2016.

[33] Y. Yu, M. Isard, D. Fetterly, M. Budiu, et al., “DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language”, In Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (OSDI’ 08), Berkeley, CA, USA, December 8-10, 2008, pp. 1-14.

[34] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, et al., “Spark: Cluster Computing with Working Sets”, In Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud’ 10), Boston, MA, June 22, 2010, pp. 10-10.

[35] MLlib: Main guide Spark 2.3.0. Documentation https://spark.apache.org/docs/2.3.0/ml-guide.html”, 2018.

[36] What is Apache Spark? https://databricks.com/spark/about”, 2018.

[37] What is Apache Spark? The big data analytics platform explained https://www.infoworld.com/article/3236869/analytics/what-is-apache-spark-the-big-data-analytics-platform-explained.html, 2018.

[38] https://archive.ics.uci.edu/ml/datasets/breast+cancer