# FEATURE SELECTION AND MAPREDUCE-BASED NEURAL NETWORK CLASSIFICATION FOR BIG DATA

**CHIT THU SHINE**

**M.C.Sc.** **DECEMBER 2018**

# FEATURE SELECTION AND MAPREDUCE-BASED NEURAL NETWORK CLASSIFICATION FOR BIG DATA

**By**

**Chit Thu Shine**

**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**

**December 2018**

# ACKNOWLEDGEMENTS

I wish to express my deepest gratitude and sincere appreciations to the following persons, who have contributed directly or indirectly towards the completion of this thesis and helped me make this dissertation possible.

I would like to express my gratitude and my sincere thanks to **Prof. Dr. Mie Mie Thet Thwin**, the Rector, the University of Computer Studies, Yangon, for allowing me to develop this thesis and giving me general guidance and workable environment during the period of study.

My heartfelt thanks and respect go to my supervisor, **Dr. Thi Thi Soe Nyunt,** Professor, Head of Faculty of Computer Science, the University of Computer Studies, Yangon, for her invaluable recommendations regarding the thesis topic, giving me detailed guidance throughout the work of this thesis and invaluable guidance and support, as Deans of Master's Courses, throughout the development of the thesis.

I would like to express my respectful gratitude to **Daw Ni Ni San**, Lecturer, the Department of Language, the University of Computer Studies, Yangon, for editing my thesis from the language point of view.

I would like to acknowledge my thanks to my teachers of the University of Computer Studies, Yangon and the University of Computer Studies (Thaton), and all of my dear teachers from childhood to the present time.

In addition, I would like to thank the board of examiners for making precious comments and detailed corrections to my thesis and those who are pressing power to improve the end result.

Last but not least, I especially thank my parents and my family for raising me and inspiring me all the time. Finally, I am grateful to my colleagues and all my friends for their cooperation and help.

# ABSTRACT

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. Backpropagation Neural Network, one of the most popular Artificial Neural Networks, is widely used in many classification applications. To reduce the data dimension, feature selection is needed to consider. MapReduce is a software framework for writing applications which are run on Hadoop that supports rapid computation and processing of Big Data.

First, the data preprocessing is performed by substituting missing values. And then, the dimension of data is reduced using Chi-square feature selection method. After that, Backpropagation Neural Network with MapReduce paradigm is used for classification. For this MapReduce-based Neural Network classifier, it is constructed using one and two hidden layers. The outputs of the proposed system are the performance measures which involve the training time, accuracy and number of selected features. The experiments have made with feature selection and without feature selection. Then, the results are compared with the results obtained from WEKA tool and Conventional Backpropagation Neural Network. Six different datasets (Thyroid Disease Diagnosis, Diabetics Diagnosis, Insurance Classification, Intrusion Detection, Customer Churn Prediction and Human Activity Recognition) are used as case study. Based on the experimental results, the MapReduce-based Neural Network algorithm gives the superior efficiency in training time faster than the WEKA tool in large dataset. And it is also found that feature selection can retain a suitably accuracy in representing the original features by selection a minimal feature subset from a problem domain. The proposed system is implemented by Java programming language on Linux platform.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF EQUATIONS

# CHAPTER 1

# INTRODUCTION

## 1.1 Introduction to Big Data

Nowadays, unprecedented rate of data according to the development of technology in Internet of Things (IoT) devices and social media, structured and unstructured, is generated across the globe. This leads to volume of data with large number of features, attributes or characteristics that leads to curse of dimensionality which technically constitutes the term known as 'Big Data'. This volume of high-dimensional data continues to grow with different kinds of data formats. Conversely, the fee of data storage continues to fall which results in data storing being more reachable. Even though creating data storage is getting less expensive and more easily available, the increasing volume of data in different data formats and from different kinds of data sources creates new issues with regards to data processing, including in its analysis and in integrating Big Data into business intelligence process [6]. Mostly, big data can be characterized with the following six V's.

i. Volume: the amount of data generated
ii. Velocity: the rate at which the data is being generated
iii. Variety: the heterogeneity of data sources
iv. Veracity: the quality of data to process
v. Variability: data whose meaning is constantly changing
vi. Value: how important or crucial the value is for business to possess the information (essential to possess, nice to possess, …)

In this century, people do not simply want to store data and therefore, they try to understand about data analytics that learns the importance and meaning of data, and use it to support them in decision making. Data analytics is that the process by applying algorithms so as to investigate the datasets and extract unknown and useful relationships, patterns, and information. In addition, data analytics are used to extract previously unknown, valid and useful patterns and information from large datasets, furthermore to detect vital relationships among the variables that are currently stored.

As the amount of big data continues to exponentially grow, they are becoming bigger interested in how to get or generate hidden insights from these large amounts of data. Therefore, they are trying to get the opportunities from these large amounts of

data and gain all of the valuable insights and benefits, consequently adopting big data analytics in order to unlock business profits and make faster and better business decisions.

One of the first things to consider when there have to manage in dealing with big data, is where and how this data will be stored once it is acquired. The traditional ways for the storage of structured data and retrieval process include relational databases, data marts, and data warehouses. The data from operational data stores is stored using Extract, Load, Transform (ELT) or Extract, Transform, Load (ETL) tools that extract the data from outside sources, transform it according to operational needs, and finally load the data into the relational database or data warehouse. Therefore, the data is transformed, cleaned, and listed before being made available for online analytical functions and data mining.

However, the traditional environment differs from the aspects of big data environment according to previous mentioned different characteristics of big data. The conventional database system related to data analytics, like relational databases, are faced big data issues in the limit of data processing capacity. For this reason, organizations are trying for big data analytics in order to analyze large scale amounts of data faster, and reveal previously patterns that are unseen, customer intelligence, and sentiment. A desire for new innovative methods and tools specialized for big data analytics, in addition the required paradigms for managing and storing such increased multitudes of data that is how and where this massive data will be stored once it's acquired, have been arisen.

Accordingly, different solutions that range from distributed systems and Massive Parallel Processing (MPP) databases for supporting high query performance and platform scalability, to non-relational or in-memory databases have been used for big data analytics. Non-relational databases, such as NoSQL which stands for "Not Only SQL", were developed for storing and managing non-relational or unstructured data. NoSQL databases are used to overcome the issues for data scaling, data flexibility, and simplified application development and deployment. Unlike with relational databases, kind of NoSQL databases separate data management and data storage. Instead of having it written in databases specific languages, such kind of databases rather focus on the high-performance data storage for scalability, and allow for data management tasks in order to be written in the application.

Alternatively, a framework called Hadoop was implemented to support in performing big data analytics which provides scalability, reliability, and manageability by providing the implemented MapReduce paradigm, which is discussed more detail in the next Chapter 2, along with the storage and analytics together. Hadoop is composed by the two core components: the Hadoop File System (HDFS) for the big data storage, and MapReduce for big data analytics.

The HDFS storage function supports the functionality of scalable, redundant and reliable storage and fast access to this large-scale data storage. It optimized for large files because a single file is split into blocks according to block size and distributed them across cluster nodes. Additionally, the data that is stored in HDFS is ensured for reliability and availability even the node failures occur because the data are distributed by replication mechanism. There are two kinds of HDFS nodes: the Data Nodes and the Name Nodes. The Data Nodes in which the data is stored as a file blocks according to the replication factor among the multiple Data Nodes. Name Node is the node in which metadata about file part that stored in Data Nodes are stored and regulates the action between the client and the Data Nodes, which directs the client to the particular Data Node in which the requested data is contained.

MapReduce is a parallel computational model and software framework, which is inspired by the two functional languages called “Map” and “Reduce”, which is suitable for parallel processing of large amount data stored in HDFS. It performs the data analytics functions and data processing. The MapReduce job first maps input values to output as a combination of <key, value> pairs. The “Map” function is partitioned from large computation tasks into smaller tasks. Each task performs some computation with to the assigned set of <key, value> pairs and generate a set of intermediate <key, value> pairs as output. This output is then assigned as the input to the “Reduce” function. Each “Reduce” then aggregates all intermediate results from multiple mappers associate with the same intermediate key, to emit the final output of the computational task.

When deeper analysis is required to find insights that are hidden from high voluminous data, machine learning may be more suitable to use [12]. Machine learning is a part an Artificial Intelligence (AI) technology that automate complex decision-making and problem-solving tasks. Benefits from machine learning from Big Data analytics range from healthcare domain, government domain, finance domain, marketing domain and many more. But machine learning algorithms work slowly for

large datasets. One way to address this issue is to make feature selection before machine learning model is built. It has become one of the important issues in classification because it results in less training time and may also have a considerable effect on the classifier's accuracy. The ultimate goal of feature selection is to select an optimal features subset that are suitable to use in model construction.

In pattern recognition and many other classification applications, artificial neural networks (ANNs) have been broadly used. Backpropagation Neural Network (BPNN) is one of the most popular ANNs. Artificial Neural Network is learned by using backpropagation algorithm. It can approximate any continuous non-linear functions by arbitrary precision with a sufficient quantity of neurons. When the size of the training data is large, BPNN training normally requires a great amount of time. To fulfill the potentials of neural networks in big data applications, the computation process must be speeded up with parallel computing techniques. Therefore, new ideas in terms of BPNN classification by using parallel environment like Hadoop were provided according to the development of cloud platforms. The following section discusses some related works of this thesis.

## 1.2 Related Works

Many researchers employed feature selection before model construction and parallel design for traditional data mining algorithms using Hadoop and MapReduce architecture are admitted to facilitate for their researches.

Changlong Li et. al. [2] implemented an Artificial Neural Network with MapReduce paradigm. In this research, they used datasets with the size of 1GB, 10GB, 50GB, and 100GB respectively. They made experiments in Hadoop Fully Distributed mode. Their experimental results show optimizing the system performance and speeding the system up in Hadoop Fully Distributed mode.

Rachana Sharma et. al. [13] implemented two traditional machine learning algorithms (Naïve Bayes and K-Nearest Neighbors) using MapReduce paradigm. They have also implemented the standard algorithms for both the classifier in WEKA 3.7 and compared the results of classifier in terms of accuracy and training time for both platforms. In this research, they used NSL-KDD dataset that has 41 features and 5 target classes. Their experiments showed MapReduce platform is faster than WEKA. It is also

found that WEKA face scalability issue as they move from 20% of the dataset to 100%, while MapReduce prove to be more efficient with larger datasets.

Navjit Singh and Anantdeep Kaur [9] admitted paper that present about feature selection for artificial neural network based intrusion detection system. They used NSL-KDD dataset that has 41 features and 5 target classes. In this paper, the performance in terms of Detection Accuracy (DA) of Multilayer Perceptron (MLP) using three feature selection methods: Chi-square, Gain Ratio and Information Gain were compared. The result showed that the Chi-square gave the best detection accuracy and fastest method out of the all.

Yang Liu et. al. [18] introduced a computationally fast parallel neural network approach utilizing Hadoop and MapReduce architecture. In this paper, they used CUG dataset with 24 features with 2 target classes. They achieved a faster scheme through parallelism by splitting the training data into data chunks and processing them in parallel. This support to train large-scale training data in parallel to speed up the training process.

According to the knowledge that gained from the previous related works, it is found that ANN is a widely used classification algorithm and Chi-square gave the best detection accuracy and fastest method for ANN compared with other feature selection methods. And it is also found that faster schema for ANN using MapReduce paradigm. Thus, in this thesis, the feature selection is performed by using Chi-square feature selection method and ANN algorithm is speeded up by using MapReduce paradigm. The experimental results of the proposed system are then compared with the results obtained from conventional neural network and WEKA tool.

## 1.3 Objectives of the Thesis

The main objectives of the thesis are as follows:

- To study Hadoop and MapReduce architecture
- To learn feature selection methods and artificial neural network
- To apply backpropagation neural network classification algorithm with Hadoop/MapReduce architecture
- To explore the effects of feature selection method on classification performance

- To analyze the results of conventional neural network and MapReduce-based neural network

## 1.4 Organization of the Thesis

This thesis is mainly composed of five chapters.

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

Chapter 2 describes the background theory related to big data, feature selection, machine learning, artificial neural network, Hadoop architecture and performance evaluation in detail.

Chapter 3 presents the design of the proposed system describing system flow, the detail steps of preprocessing, feature selection using Chi-square method, MapReduce-based Backpropagation Neural Network algorithm, performance evaluation and description about datasets used.

Chapter 4 mainly describes the implementation of the proposed system in detail that includes the experimental setup, the system's implementation, the MapReduce-based Neural Network job execution and the experimental result.

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

# CHAPTER 2

# BACKGROUND THEORY

## 2.1 Big Data

Big Data is defined as complex and voluminous amount of data which requires new technologies, tools, frameworks and architectures to get hidden insights that can give profit to businesses and organizations. Nowadays, there are many new sources of big data such as location data that arises from traffic management, and personal devices tracking such as mobile smart phones. Big Data has emerged as a result of our nature of living in a society which tends to increase use of data intensive technologies. Due to the voluminous of data, it becomes very difficult to make effective analysis using the existing traditional techniques.

Since Big Data is becoming a popular technology in the applications of market which can bring huge amount of profits to the organizations of business, it becomes necessary to understand the characteristics associated in bringing and adapting to this technology. The concept of Big Data means a dataset that grows continuously so much which is very difficult to manage using existing traditional database management concepts and techniques. The difficulties can be consisted of collecting the data, data storage, data processing, data searching, data analysis and data visualization etc.

Big Data due to its various characteristics like volume, velocity, variety, variability, value and veracity put forward many challenges. In addition to these various characteristics of data stored in different sectors, the types of data generated and stored also differ markedly from industry to industry. Typically, Big Data could be found in three forms: Structured Data, Unstructured Data and Semi-structured Data [20].

- **Structured Data:** It is a type of data that can be stored, accessed and processed in the form of fixed format.
- **Unstructured Data:** The structure or from of data to access, store and process is unknown. Examples of unstructured data are social media data, images, and videos and so on.
- **Semi-structured Data:** It can contain both of structured and unstructured data. It can be seen as a structured form but it is not specified. Data represented in XML file is one of examples of Semi-structured data.

### 2.1.1 Characteristics of Big Data

Big Data can be characterized by the following six V's model and the overview of these six characteristics is shown in Figure 2.1 [20].

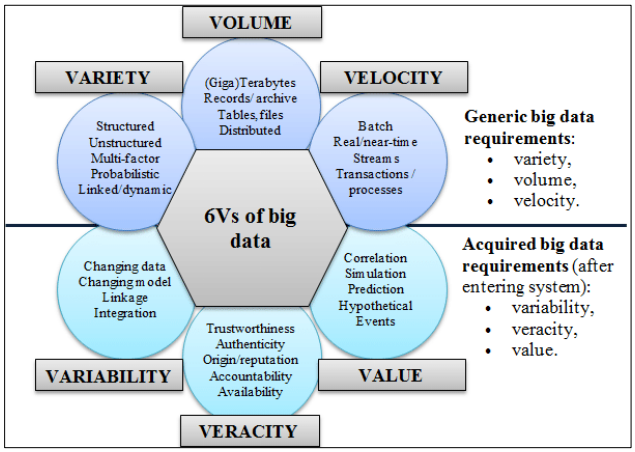



**Figure 2.1** The characteristics of big data

- **Data Volume:** A large amount of digital data is being generated from different sources which includes social medias, sensor devices and business transactions on a daily basis. Handling with these voluminous data is the most important challenge for conventional architecture. The primary concerns are storage capacity, and processing capacity to analyze it.
- **Data Velocity:** Typically, velocity refers to the rate at which the data is being generated in real-time which can be both the rate of new input data from different sources and the rate of conduct analysis to meet the demands.
- **Data Variety:** One of the main attractions of big data analytics is that data from different sources can support valuable insights by integration and aggregation. This leads to the requirement of processing structured data, unstructured data and semi-structured data.
- **Data Veracity:** The next aspect of Big Data is the quality of data to process. It refers to the noise, biases, and abnormality in data. It is also important because the accuracy of analysis depends on the data veracity of different sources. Therefore, it needs to spend time on cleaning data and organizing to clean 'dirty data' from accumulating data in the system.

- **Data Variability:** Data variability refers to nature of data whose meaning is constantly changing. It also major issue because of data dimensions resulting from multiple different kinds of data types and sources.
- **Data Value:** As the data keep by different organizations is being employed by them for data analytics, it's important to consider the value of big data. Big data has many valuable big value to increase efficiency, productivity, and revenues, reducing risk and lowering costs in businesses management.

### 2.1.2 Applications of Big Data

The main objective of Big Data applications is to assist organizations that create additional informative business decisions by analyzing large scale amount of data. The organizations of different domains are investing in Big Data applications, for examining large datasets to uncover all hidden patterns, unknown correlations, market trends, customer preferences and other useful business information. Some industry domains using big data technology are listed [19] as follows.

- **Banking:** Big data is hugely used in the fraud detection in the banking sectors. In banking sectors where the big data is implemented, all the mischief tasks are critical tasks. It detects the misuse of credit and debit cards, repository of inspection tracks, treatment of venture credit hazard, business clarity, and alteration of customer statistics, IT action analytics, public analytics for business, and IT strategy fulfillment analytics.
- **Insurance:** The big data as well enables for the better purchaser preservation from insurance agencies. In the claim administration, extrapolative big data business analytics has been utilized to provide more rapid service given that enormous quantity of information can be worked on particularly in the countersigning period. Scam discovery has also been improved. In the course of large data from digital conduits and social media, real-time dominant of allurements throughout the argument series is employed to afford insights.
- **Healthcare:** The big data is efficiently used in the field of medicine and healthcare. By analyzing Big Data, all the patients' history in terms of patient records, treatment plans, prescription information, etc., can be accessed quickly. Effective management leads to improve patients' health care by analyzing hidden insights. And it can also be used in the medical diagnosis to classify patient's disease efficiently.

- **Manufacturing:** Manufacturers can improve their quality and output whereas minimizing waste wherever processes are known as the most effective key factors in today's extremely competitive market. Many manufacturers are engaged on analytics wherever they need to solve issues quicker and create additional agile business decisions.
- **Retail:** A large amount of customer relationship data is generated every day. Therefore, maintaining these large-scale data is the most challenge within the retail business. Retailers should have distinctive marketing ideas to sell their merchandise to customers, the most effective way to make transactions handling, and applying impermanent ways of using innovative ideas using Big Data to boost their business.

## 2.2 Feature Selection Methods

The term feature selection refers to tasks of identifying and finding that hopefully the best subset of the input feature set with respect the target task (classification accuracy). The main objective of feature selection is typically to minimize the overall cost of measurement acquisition and some features are discarded and then there is no need to obtain them and the selected features retain their original physical interpretation. Feature selection is also known to as variable selection, variable subset selection, attribute selection or feature reduction. It is the process of identifying and removing as much irrelevant and redundant information as possible. This reduces the data dimensionality and can support learning algorithms to operate quicker and more effectively. In some cases, accuracy on future classification can be improved; in others, the result is a more compact, easily interpreted representation of the target concept [8].

There are many reasons to consider that feature selection should be treated or not in classification. Some of the reasons to use feature selection are [15]:

- It supports the machine learning algorithm to train quicker.
- It reduces the complexness of a model and makes it easier to interpret.
- It can improve the accuracy of a model if the proper feature subset is chosen.

The existing feature subset selection can be classified into four broad categories based on evaluation criteria namely filter, wrapper, hybrid and embedded method [17].

- Filter method: filter features are passed to training process.

- Wrapper method: feature selection as wrapper around the classifier training process.
- Hybrid method: combines filter and wrapper models by applying them in different stages of the selection process.
- Embedded method: feature selection is embedded within the training algorithm.

### 2.2.1 Filter Method

A filter is defined as a feature selection method where search criteria is performed independently to find appropriate feature subset without reference to the machine learning algorithm. Features are selected on the idea of their scores in varied applied the proper statistical tests for their correlation with the variable of outcome value. The model for filter feature selection method is shown in Figure 2.2.

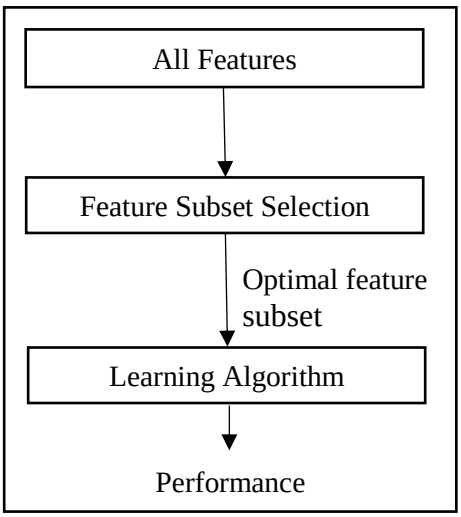



**Figure 2.2** Filter feature selection method

It generally uses as a preprocessing step. Some feature selection methods of filter method are [11]:

- **Chi-square:** Pearson's Chi-square test is a kind of statistical test which can be applied to evaluate the correlation between the two variables. This is called hypothesis testing. The critical value of Chi-square statistical test is determined by the two parameters: (1) the null hypothesis and (2) the number of degree of freedom (df). The null hypothesis states that any observed pattern is due solely to chance and that, hence, no relationship exists. Thus, an objective of Chi-square statistical test is to examine whether the null hypothesis can be rejected. The formula for calculating the Chi-square test requires to calculate a value for

each cell and then add them all up. The larger the number of observations tend to the larger the value of this test statistic. This problem is controlled by the degree of freedom statistic. Every statistical test has its own way for degrees of freedom (df) calculation. The degree of freedom for any Chi-square test is defined by subtracting one from the number of features of the dataset.

- **Correlation:** The Correlation Feature Selection (CFS) measure is used to evaluate feature subset on the basis of the hypothesis: "Good subset of feature can contain highly correlated features with the classification, nevertheless unrelated to each other alternative".
- **Entropy:** Entropy is a kind of test to measure the amount of information in a randomized variable: it is the average length of the message required to transmit an outcome of that variable using optimal code.
- **One-attributed-rule (OneR):** The concept of one-attribute-rule (OneR) algorithm is to find the one attribute to use that produces fewest prediction errors.

### 2.2.2 Wrapper Method

In wrapper method, feature selection is wrapped around the classifier training process. The feature selection is processed as a black box using the induction algorithm. The strategy of the wrapper method is to use an induction algorithm to estimate the merit of the searched feature subset on the training data and using the estimated accuracy of the resulting classifier as it's metric. The wrapper methods often result better than filter methods because they are tuned to the specific interaction between an induction algorithm and its training data. Thus, the wrapper methods consider the final induction algorithm. The model for wrapper feature selection method is given in Figure 2.3. Some feature selection methods of wrapper method are listed [15].

- **Forward selection:** Forward selection is iterative method in which it starts with no feature and, in each iteration, it adds the feature which improve the performance of the classifier model. The iteration keeps running until an addition of a new variable doesn't improve the model performance.
- **Backward elimination:** In backward elimination, it starts with all of the features and, in each iteration, it removes the least significant feature that can

support to improve the performance of the classifier model. The iteration is performed till no improvement is discovered on the removal of features.

- **Recursive feature elimination:** It's a kind of greedy optimization algorithm which aims to seek out the most effective feature set. It creates classifier models repeatedly and keeps aside the worst or the best performing feature at every iteration. It then constructs the next classifier model with the left features till all the features are exhausted. After that, it ranks the features according to the order of feature elimination.

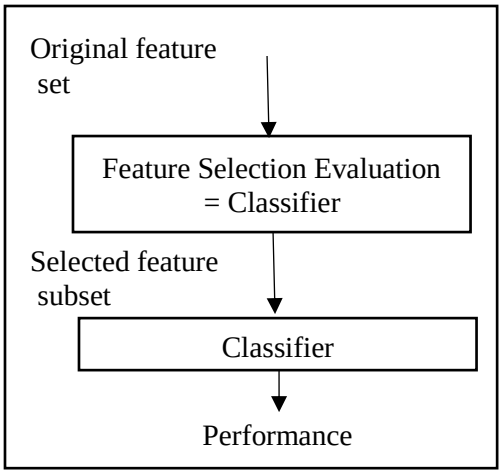



**Figure 2.3** Wrapper feature selection method

### 2.2.3 Hybrid Method

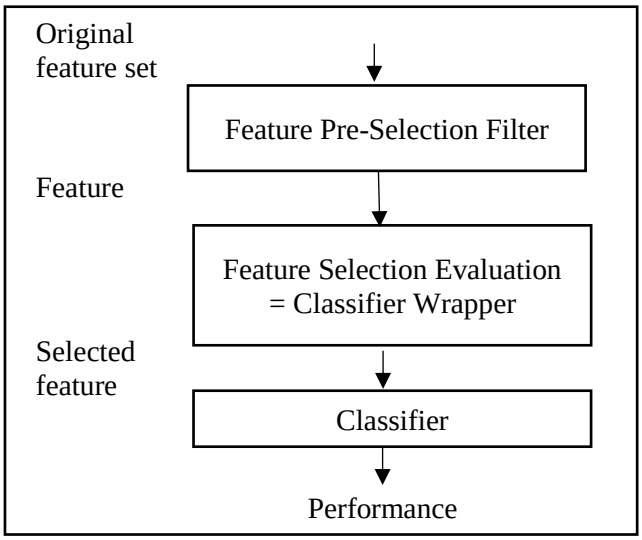



**Figure 2.4** Hybrid feature selection method

The model for hybrid feature selection method is shown in Figure 2.4. Hybrid method combines filter and wrapper methods by applying them in different stages of the selection process. The first step is typically based on filter method to reduce the number of features that is used in the second stage. Afterwards a wrapper method is employed to select the desired number of features using this reduced set.

### 2.2.4 Embedded Method

Embedded method attempts to find an optimal subset of features best contribute to the accuracy of the classifier model while the model is being created. The most common type of embedded feature selection methods are regularization methods which are also called penalization methods that introduce additional constraints into the optimization of a predictive algorithm (such as a regression algorithm) that bias the model toward lower complexity (fewer coefficients). The model for embedded feature selection method is given in Figure 2.5 [15].

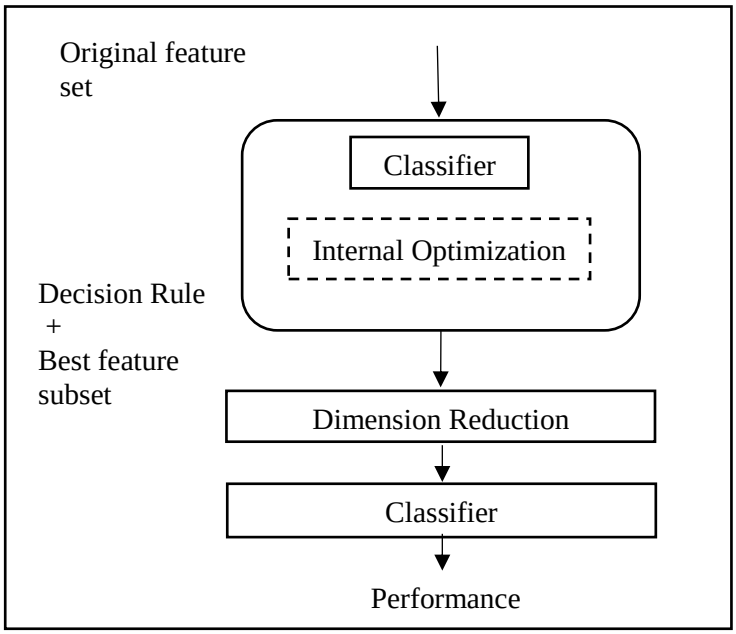



**Figure 2.5** Embedded feature selection method

## 2.3 Machine Learning

Machine learning is a branch of artificial intelligence that aims at enabling machines to perform their jobs skillfully by the construction of programs which learn from past experience. The fundamental characteristics of intelligent behavior are the abilities to pursue goals and to plan for future actions. Possible activities include

categorizing data into different groups and making forecasts about future information patterns dependent on the past or experienced data. This is often an exceptionally appealing option to manually construction them, and in the most recent decade the utilization of machine learning has spread quickly all through computer science and beyond.

In machine learning, data is the most important, and the learning algorithm is employed to get and learn knowledge or properties from the data. The quality or amount of the dataset can have an effect on the learning and prediction performance. The learning algorithms of machine learning can be organized into taxonomy, based on the desired outcome of the algorithm.

### 2.3.1 Fundamental Aspects of Machine Learning

There are many fundamental aspects to be considered when designing a machine learning application, aiming to achieve a good generalization performance. The design of a machine learning system in general usually involves the following fundamental aspects: [1]

#### (i) Pre-processing

Pre-processing is the basis step of machine learning for which some given raw data have to be pre-processed and prepared via applying pre-processing algorithms that depend on the specific application earlier than they may be fed into the system. As an example, document images regularly need to convert to binary format before engaging in any sort of optical character recognition (OCR) or layout analysis. In other cases, the input raw data can be incomplete because of missing values, while the supposed machine learning paradigm does not handle incomplete datasets. In this situation, an appropriately designed pre-processing state could assist with filling these gaps in the dataset by using averaging or the use of other proper statistical approaches. The pre-processing phase may consist of steps to remove noise or outliers from the dataset. Therefore, choosing and making use of the proper pre-processing method can have a substantial effect on further steps taken by machine learning.

## (ii) Feature Selection

Generally, the patterns to be processed and categorized are represented via various measurement metrics referred to as features. To recognize patterns, a proper set of features must be chosen. These features have to satisfy certain aspects so that it will be most effective. By removing invariant to irrelevant features of the input data, this sufficiently effect to compact in size that minimize memory and resource consumption and computation time. The selection of features may require prior knowledge of the problem domain, but choosing appropriate features is often a tricky task, and it sometimes involves lengthy evaluations [1].

## (iii) Model Selection

The performance of a classifier also relies on the model selection because various type of models can offer various approximations for different problem domains. Greater accurate approximations result in improved predictions and better classification rates. Not only the volume and quality of available sample data are important to determine the performance of a classifier, but also choosing a model may come into play in the performance requirements.

## (iv) Training, Testing and Optimization

After selecting a classification scheme, it needs to be evaluated against example data. Model training is performed on a data subset, also referred to as training data. The model that was generated from the training phase has to be further examined and tested in a subsequent testing phase by using a data subset called testing data. One issue is that the computation time for each phase may become quite excessive depending on the schemes used. In particular, the large number of iterations occurring within the training and testing phases during parameter optimization may require significant computational requirements. A solution to this problem for the more effective deployment of machine learning techniques can be achieved by utilizing parallelism within the system. As a result, the system can be empowered to use many processors, and even an excessive number of processing machines, at the same time. This is where the concept of distributed computing comes into the picture [1].

### 2.3.2 Types of Machine Learning Algorithms

Machine learning comes in many alternative flavors, counting on the learning algorithm and its objectives. It can be divided into four main groups based on their purpose [3] on most.

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

### (1) Supervised learning

Supervised learning adapt a kind of model to generate the known outputs from a training subset of the original dataset (e.g. classify automobile types on photos). In the first step, the system receives both of input data and output data. The task of this process is to make applicable rules that map the output of the given input. The training process ought to continue till the extent of model performance is high enough. After the training process is finished, the system ought to be ready to classify associate an output objects that it hasn't seen throughout the training phase. Mostly, this classification process is basically quick and accurate. Supervised learning issues are further classified into classification and regression issues.

- **Regression:** Regression technique aims to predict the value of associated output. For instance, to predict the worth of some product, like a worth of a house in a very specific town or the worth of a stock. There is a large variety of things which can be made prediction by using regression.
- **Classification:** Classification technique aims to make class assignments. It can be able to predict the response value and also the data is classified into "classes". For example, recognition of a type of automobile in a photo, of the mail spam, and of today's weather.

### (2) Unsupervised learning

Unsupervised learning is a learning in which the predicted output variables are unknown. The model is trained using these unlabeled data. These techniques aim to find hidden structures, like realize groups of photos with similar cars, however it is a bit troublesome to implement and isn't used as wide as supervised learning technique.

Unsupervised learning can be used as a preceding step before applying supervised learning. The internal structure of data might support information on a way to get better outputs. One of the categories of unsupervised techniques is clustering.

- **Clustering:** Clustering can be used to find out similarities and variations. It groups similar things together. Here it doesn't need to know any class labels, however the system can understand data itself and cluster it well. In contract to classification, the output labels are not known beforehand. This kind of learning algorithm can facilitate to solve several obstacles, like produce clusters of similar tweets based on their content, realize groups of photos with similar cars, or determine different kinds of news.

### (3) Semi-supervised learning

Sometimes, it is needed something between these two varieties of machine learning techniques, supervised and unsupervised learning for all the observations. In such situations it can be used Semi-supervised learning, which refers to a learning process in which lots of output values (the ones that are aimed to predict) are missing. It needs to apply each supervised and unsupervised ways so as to get useful results. This is usually the case in medical applications, in which medical doctors are not able to manually process classification all sorts of health problem according to the overwhelming large amounts of data.

### (4) Reinforcement learning

Sometimes, the desired output value is unknown explicitly, however the system can support feedback on the provided output. Learning supported such feedback is termed Reinforcement Learning. This is used for training the artificial intelligence of gaming in the NERO game and could be found in schools. The students learn for a specific title (reinforcement learning), then they make sitting an exam, and the teacher provides them grades while not specifying which answers were wrong or not.

## 2.4 Artificial Neural Network

Artificial Neural Network (ANN) is a branch of artificial intelligence and that inspired by human nervous system. The nervous system [14] is composed of tens of millions of interconnected nerve cells in which everyone plays a specific task. For the

reason that interconnections among the ones cells are complicated, the final results of these multi simple task's accomplishments is the execution of a larger more complex task. In a comparable manner, Artificial Neural Networks (ANNs) are composed of a huge quantity of interconnecting neurons. The essence of ANNs are the mimicking of how the nervous system inside the human processes information.

Neural networks can be really stated as a processing approach, especially based on mimicking the human brain in which huge interconnected layers process information simultaneously and adapt over the course of multiple runs. Thus, a neural network tries to generalize form a recognized data to new unknown data. Table 2.1 indicates the common phrases used in reference to neural networks and their analogous on brain expressions.

**Table 2.1** Terminology and analogies of Artificial Neural Network

| **ANN Terminology** | **Brain Equivalent** |
|---|---|
| Node | Neuron |
| Interconnect | Synapse |
| Weight | Simulation |

Artificial Neural Networks (ANNs) were utilized in an extensive sort of application. ANNs are excellent tools for approximating any given function of single or multi-variable inputs and outputs. Because of the flexibility inherent in ANNs' configurability, they can be used in many decision-making processes. Pattern recognition and numerical classifications are ones of those typical applications that use ANNs [4], [5].

Artificial Neural Networks achievement at system modelling for exceedingly complicated physical processes can be attributed to the original architecture on which they are based, the human brain. At present, brain function is not absolutely understood. A brain neuron collects signals from other neurons of the Central Nervous System (CNS), through structures referred to as dendrites as shown in Figure 2.6. The neuron sends out spikes of electrical activity through a long thin strand called an axon. This axon splits into lots of branches. At the end of a branch, a structure called a synapse converts the activity from the axon into electrical effects that may excite or inhibit activity in the connected neurons.

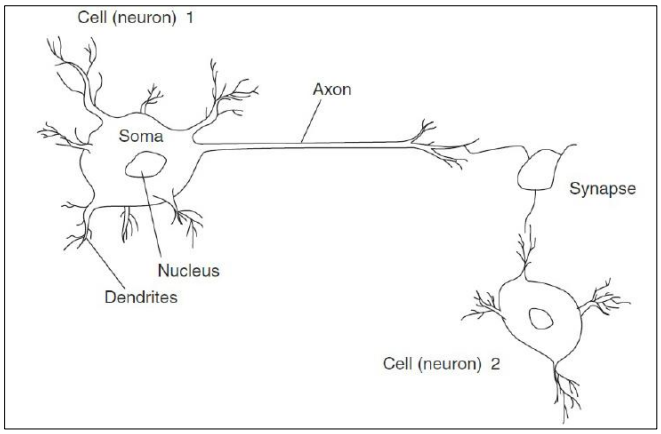



**Figure 2.6** The structure of human neuron

When a neuron receives an excitatory input that is adequately large in comparison with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by converting the effectiveness of the synapses in order to have an effect on other modification.

### 2.4.1 Structure of Artificial Neural Network

An artificial neural network is constructed by an input layer, quantity of hidden layers and output layer. The main duty of the input layer is to receive the external measurements 'data' and propagate it forwards to the hidden layer or layers. In every layer, there are number of nodes called neurons, and each neuron is capable of communicating with its neighbors through weights and is capable of changing its present state depending on the signal it receives. Figure 2.7 shows a scalar product of one artificial neuron. The neurons are connected to each other in layers with the first layer that is called the input layer and the final layer that is called the output layer. The layers between those two layers are called the hidden layers. The objective of a neural network is, for a given set of inputs, to give a set of desired or expected outputs. The weights or what it is referred to as synaptic weights are updated by the usage of certain learning algorithm until the error between the actual output of the neural network and the desired output is reduced to an acceptable level which will be relying on the function being approximated [14].

The quantity of hidden layers isn’t limited in to a certain number. An activation function is presented at each node in the hidden layer in which manipulating the output that is passed to the next layer. An activation function of the output layer is also presented to limit the output under certain utilization. One more important parametric

structure of the network is the number of nodes being included in the hidden layer or layers. The wide variety of these nodes influences the accuracy of the results to some degree. It is usually beneficial to have a large number of hidden nodes at the same time serving extremely complex networks. Determining the best quantity of nodes is not an easy task, but generally it is determined by means of repeat testing the network until getting the favored one. However, a trade-off is involving, the more complex the network is, the longer training time is needed. Not only training effort exponentially increases as the number of layer increases which will increase the computation required for training but also it may negatively impact in the results. For that reason, using more than three hidden layers is rare in most systems [14].

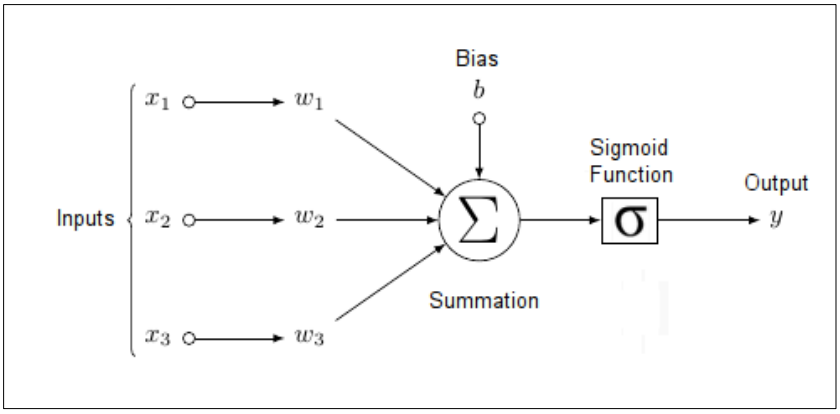



**Figure 2.7** A typical scalar summation of an artificial neuron

## 2.4.2 Types of Activation Function

The artificial neuron like the biological neuron described in Figure 2.6 is a processing element. An output for this artificial neuron is calculated by multiplying its inputs by a weight vector. The results are then summed collectively and an activation function is carried to the sum. The activation function is a function used to transform the activation level of a unit into an output signal. Normally, activation functions have a squashing effect; they contain the output within a range. There are numerous activation functions that can be applied to neural networks; two important activation functions are linear activation and non-linear activation [14].

### 2.4.2.1 Linear Activation Function

The first is the linear transform function that's shown in Figure 2.8, or pure-line function. It is described as follows.

$$f(x) = x \tag{2.1}$$

Neurons of this kind are used as linear approximations on the function being approximated.

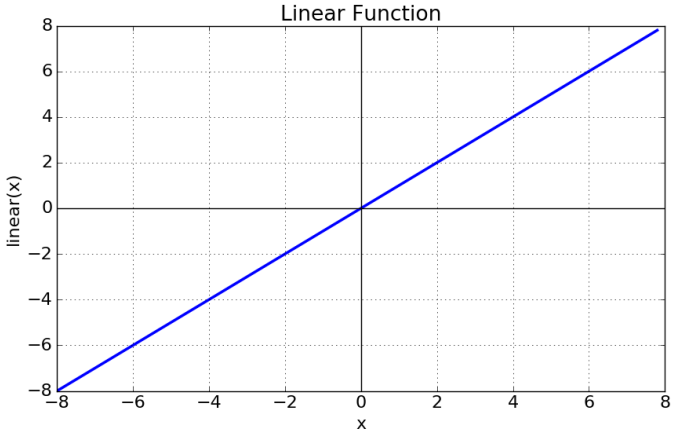



**Figure 2.8** Linear activation function

### 2.4.2.2 Non-Linear Activation Function

There are several forms of non-linear activation functions; the two most common are the log-sigmoid transfer function and the tan-sigmoid transfer function. Plots of these differentiable, non-linear activation functions are illustrated in Figure 2.9 and Figure 2.10 respectively. They are usually utilized in networks which are trained with back-propagation.

### (i) Logistic Activation Function

The first sigmoid activation function which is referred to as logistic activation function is shown in Figure 2.9 and defined as follows.

$$logsig(x) = \frac{1}{1+exp(-\beta(x))} \tag{2.2}$$

The value of $\beta$ can be changed in which it changes the shape of the sigmoid. As $\beta$ tends toward infinity it behaves more and more like a hard-limiter where the slope of the sigmoid is zero. In the case when the slope is not zero, the output range is squeezed between 0 and 1. It is used in each the hidden and output layers of the constructed network. It supports to get a reasonable accuracy with the proposed non-linear problem.

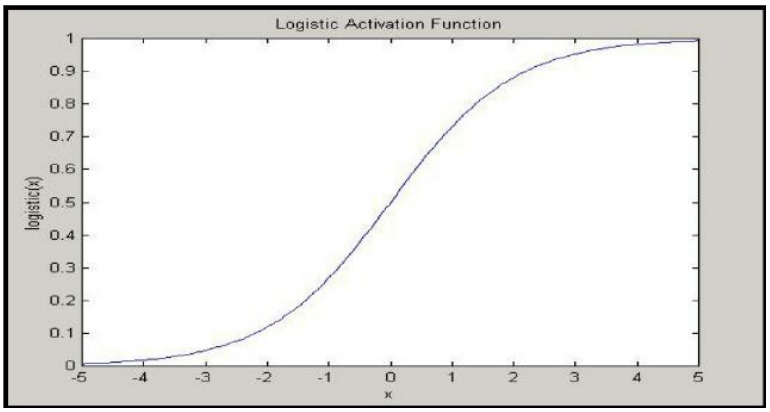



**Figure 2.9** Log-sigmoid activation function

## (ii) Hyperbolic Tangent Activation Function

The second one stated activation function is known as hyperbolic tangent activation function. In contrast to log-sigmoid activation function, this function squeezes the output between -1 and 1. Tan-sigmoid activation function is described in Figure 2.10 and defined as:

$$tansig(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)} \quad (2.3)$$

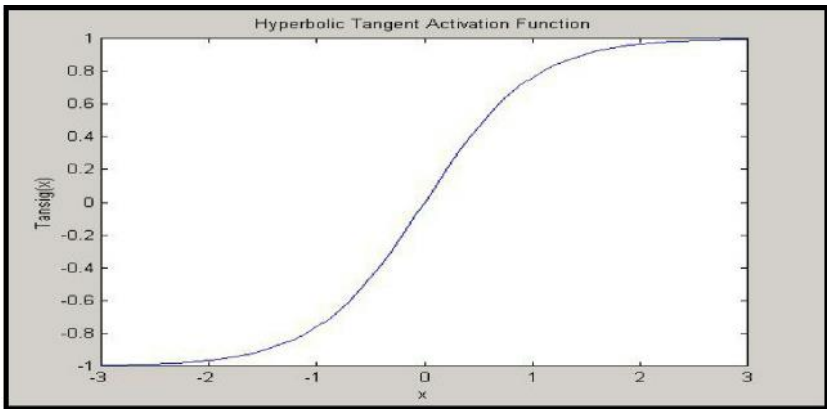



**Figure 2.10** Tan-sigmoid activation function

## 2.4.3 Artificial Neural Network Topologies

ANN can be typically categorized into two groups of topologies based on connection patterns between their units and data propagation among them. These include feedback neural networks and feed-forward neural networks.

- **Feedback neural networks:** This kind of neural network has feedback connections i.e. cycles or loops are present within this type of network. In such networks, data flows from input to output units or vice-versa. Feedback occurs

in a dynamical system when the output of an element in the system affects its precise input, which give rises to 1 or more closed paths for signal transmission across the system [16]. They are also referred to as the recurrent networks. Kohonen networks, and Hopfield networks are examples of feedback neural networks.

- **Feedforward neural networks:** In this type of network, the data commonly flows from input to output units in feedforward fashion. This is one of the simplest ANN where the information moves in only one direction from input to hidden (if present) to output nodes.

### 2.4.4 Multi-Layer Perceptron

There is an extensive variety of ANNs, however the most commonly used ANN is the Multi-Layer Perceptron (MLP), called feedforward neural network. The most commonly learning algorithm that is used within the feedforward MLP is called error back-propagation. The MLP with back-propagation algorithm is generally characterized by the presence of an input layer, one or more hidden layers, and an output layer as described in Figure 2.11.

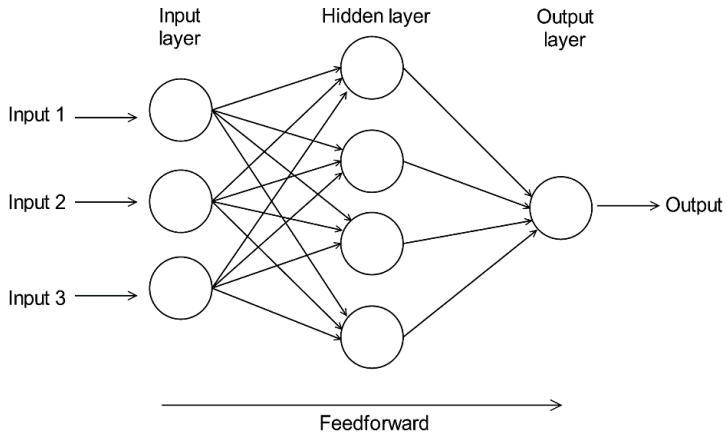



**Figure 2.11** General multilayers feedforward network

The input signal propagates through the network in a forward direction, on a layer-by-layer basis [16]. MLP have been successfully applied to solve numerous diverse and complex problems [16] by using a supervised learning process. Finally, a set of outputs are produced because of the actual response of the network. This generates a difference between the desired output and the output of the network (error). The synaptic weights of all networks are all fixed during the forward pass. Similarly,

during the backward pass, error signal is computed and it propagates backward through the network. Finally, the synaptic weights are all adjusted in accordance with an error-correction rule. This is how this particular algorithm received its name from 'back-propagation'. Training time can range from seconds to days depending on the problem domain, the quantity and representation of data, and the implementation of the network.

### 2.4.5 Backpropagation Learning Algorithm

Backpropagation which, extra descriptively, can be known as back-error propagation, is the most widely used supervised learning algorithm in neural computing [4]. A backpropagation network is very easy to implement, and it includes one or more hidden layers with its input and output layers. This kind of network is taken into consideration feedforward because there aren't any interconnections between the output of a processing element and the input of a node within the same layer or in a previous layer. Externally provided correct patterns are compared with the neural networks output during (supervised) training, and feedback or precisely a back propagate is used to adjust the neurons connecting weights until the network has classified all the training patterns as effectively as preferred.

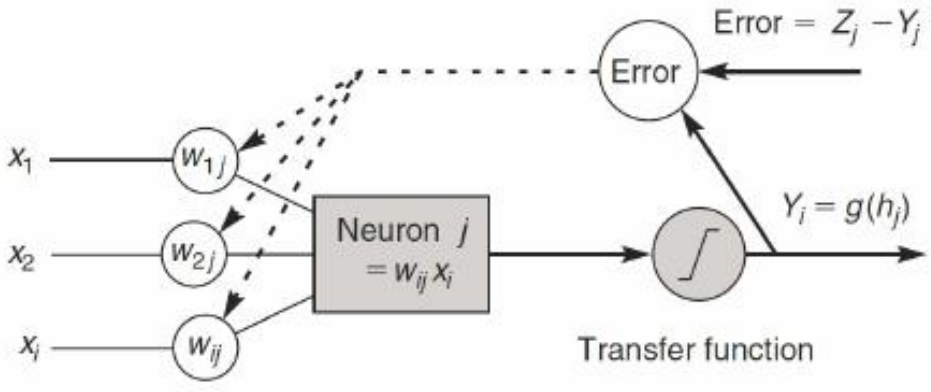



**Figure 2.12** Backpropagation of error for a single neuron

Starting with the output layer, errors among the actual and desired outputs are used to get the correct result of the weights for the connections to the previous layer. Backpropagation of error for a single neuron is shown in Figure 2.12.

The learning algorithm includes the subsequent procedures:

- Initialize weights of neurons with random values and set alternative parameters.
- Read the input vector and the desired output.

- Compute the actual output value by the calculations, working forward through all the layers specified.
- Compute the error.
- Change the weights by working backward from the output layer through the hidden layers.

This procedure is repeated for the whole set of input vectors until the desired output and the real output agree within some predetermined tolerance. Given the calculation requirements for one iteration, a large network can take a totally long time to train; therefore, in one variation, a set of cases are run forward and an aggregated error is fed backward to speed up learning. Sometimes, relying on the initial random weights and network parameters, the network does not converge to a satisfactory performance level. When this is the case, new random weights must be generated, and the network parameters, or even its structure, can also need to be modified before any other try is made.

## 2.5 Hadoop Architecture

To address challenges of big data processing requirements, the Hadoop framework is designed to provide a reliable, shared storage and analysis infrastructure to the user community. Apache Hadoop is an open source distributed processing framework that was designed to run on low-cost hardware. The core components of Hadoop are Hadoop File System (HDFS) and MapReduce processing framework. While data processing functionality of the Hadoop framework is supported by MapReduce, the storage portion is presented by HDFS. The MapReduce functionality is designed as a tool to analyze data deeply and the transformation for huge datasets. Hadoop supports the users to analyze or explore complex datasets by using customized analysis commands or scripts.

In Hadoop 1.X, first release of Hadoop, it was started with two main components, the Hadoop Distributed File System (HDFS) and the MapReduce. There were some drawbacks by using Hadoop 1.X [10].

For example, there was only one, centralized, JobTracker, which had to perform many activities like Resource Management and Allocation, Job scheduling and execution etc. In case of a failure, the jobs in the system had to start all over again. Also, there was only one Name Node that stored all the metadata about the files/blocks stored

in the HDFS. In addition, Hadoop 1.X was mainly UNIX based and big companies running Windows Microsoft servers could not use Hadoop.

In order to overcome the limitations of Hadoop 1, the architecture changed by adding a new component. The upgrade to Hadoop 2 led to a more flexible system, in terms of availability and scalability. The difference of architecture between Hadoop 1.X and 2.X is shown in Figure 2.13. Hadoop 2 is characterized and always followed by the abbreviation YARN which is explained in subsection 2.5.3. In section 2.5.1, HDFS is explained detail including its features and architecture. The MapReduce is further discussed in the section 2.5.2.

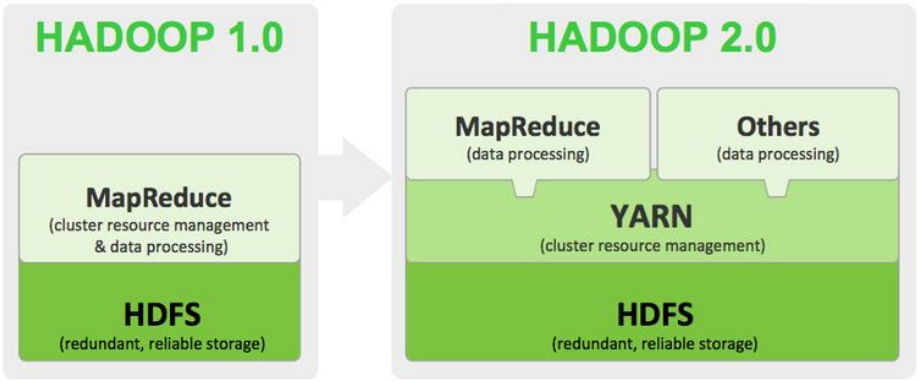



**Figure 2.13** Difference of architecture between Hadoop 1.X and 2.X

### 2.5.1 Hadoop Distributed File System (HDFS)

The main purpose of Hadoop Distributed File System (HDFS) is to store very large-scale datasets reliably, and to process those datasets at high bandwidth. In a large cluster, thousands of servers can directly be attached for storage and execution of user application tasks. By distributing computation and storage across several servers, the resource can grow economically with demand at every size. When data to make analytics arrives in the Hadoop cluster, HDFS breaks it into smaller chunks and redistributes these chunks among the different nodes (servers) that are engaged in the Hadoop cluster. Each chunk of the entire dataset can be resided on each server or node, and it makes conceivable each data chunk is replicated on other server or node. This replication features support reliability of data that resides in HDFS [10]. There are many other features of HDFS. The details of common features of HDFS are discussed in the Section 2.5.1.1. In Section 2.5.1.2, the architecture of HDFS also cover the detailed architecture of Hadoop HDFS i.e NameNode, DataNode in HDFS, JobTracker, and TaskTracker in HDFS.

### 2.5.1.1 Hadoop Distributed File System's Features

HDFS has many common characteristics like other distributed file systems. The features of HDFS can be categorized as follows [1]:

- **Very large datasets:** HDFS enables the processing of massive amounts of data in the order of petabyte scales on distributed file systems.
- **Streaming data access:** The HDFS process follows the model of write once and then read many times. This approach not only minimizes data coherency issues, but it also over-simplifies high throughput data access, making it efficient and suitable for MapReduce applications, which is discussed later in this chapter.
- **Commodity hardware:** Hadoop can run on inexpensive machines with noticeably low power consumption. This brings the power of fault tolerance to the scheme, where basic failures can be recovered and compensated with minimum or no effects on the total functionality.
- **Data reliability:** As data is stored on multiple nodes and racks in HDFS architecture, data will be easily accessible in case of sudden failures. In fact, the placement of replicas is one of the key differentiators between HDFS and other distributed file system mechanisms.
- **Portability:** HDFS is designed so that it can be easily moved from one platform to another, thereby facilitating its use as a platform of choice for many applications.

### 2.5.1.2 Hadoop Distributed File System's Architecture

The architecture of HDFS is illustrated in Figure 2.14. Each file is stored in HDFS as a metadata file and a collection of blocks. While blocks store the content, the metadata file records blocks location and replication. When a large file is saved in HDFS, it first breaks in block-sized chunks that finally stored as independent units. The default size of HDFS block-size is 128 MB and the replication parameter at three, which means that each block is stored three times across the cluster. For performance or reliability reasons these parameters could be customized by means of xml configuration files.

It consists of one NameNode and a number of DataNodes, and it has master/slave architecture. The master server, referred to as the NameNode, divides files

into blocks and distributes them among the cluster members with replication to support fault tolerance. It also keeps all metadata about stored files and arranges the system namespace. The slaves, referred to as DataNodes, are the actual repository of the data blocks; they respond to read/write requests from clients and distribute replication tasks as instructed by the NameNode. In response to a request, the relevant data is retrieved from the HDFS and sent to a set of pre-allocated compute nodes to implement parallel scanning. To achieve its goal, HDFS uses the JobTracker and TaskTracker functions. JobTracker is responsible for scheduling and assigning the relevant tasks to TaskTrackers, while TaskTrackers are only responsible for accomplishing the jobs they are assigned to. Upon completion of the job, TaskTrackers notifies JobTracker about the result of the work (success/failure), and in case of failure, the JobTracker reschedules the failed operations [1].

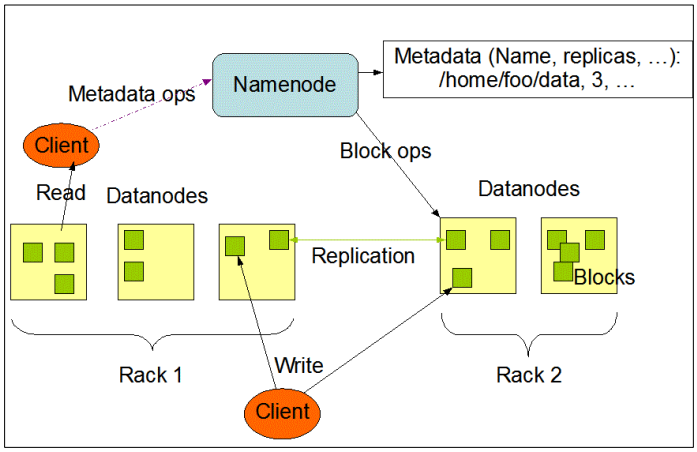



**Figure 2.14** HDFS architecture

## 2.5.2 MapReduce Framework

MapReduce framework is a computation software model for writing applications which are run on Hadoop. In the MapReduce data-processing approach, all processing operations are expressed using two main primitives – (a) a map function to accept a <key, value> pair, perform some computations and generate a set of intermediate <key, value> pairs as output, and (b) a reduce function to aggregate all intermediate results from multiple mappers associated with the same intermediate key, perform some computations including sorting and merging the intermediate results and emit the final output of computation results [1].

The data to be processed by an individual mapper is represented by block in HDFS because a single Map task instance is created for every data block in the HDFS,

which is relevant to the input data [7]. Therefore, the number of map tasks is equal to the number of block split. These simplified functions permit users to build and deploy parallel data-processing jobs without the need to explicitly coordinate parallel sub-tasks with distributed file storage. As a result, the MapReduce abstraction can vastly improve user productivity and experience.

### 2.5.3 Yet Another Resource Negotiator (YARN)

The architecture of YARN abstracts out the Resource Manager, which is responsible for the monitoring of the resource usage in the cluster, the activities taking place and also, allocates the resources. The planning and the execution of Hadoop jobs are handled by the Application Master. Every application has its own Application Master. In the YARN architecture the MapReduce is considered as an application, so if there are three MapReduce jobs running, every one of them will have their own Application Master.

In order to run any application, the Application Master will request resources from the Resource Manager, in terms of RAM, CPU and memory.

In every node of the Hadoop cluster there is a daemon running, called Node Manager. The Node Manager is the contact point between the Resource Manager and the available nodes. The jobs to run are scheduled by the Resource Manager, keeping metadata in order to recover in case of any Resource Manager crash. The Application Master then, requests resource, executes the tasks and handles any job failures.

Therefore, the new processing models in Hadoop are offered by YARN in Hadoop 2. It consists of a cluster with a resource management system, allowing every distributed program to run and analyze big datasets in a Hadoop cluster [10].

## 2.6 Performance Evaluation

Evaluating the performance of learning algorithm is a fundamental aspect of machine learning. Estimating classifier accuracy is important in that it allows to evaluate how accurately a given classifier will label future data, that is, data on which the classifier has not been trained and also help in the comparison of different classifiers.

An estimate of classification accuracy on new instances is the most common performance evaluation criteria, although others based on information theory have been

suggested. Feature selection is considered successful if the dimensionality of the data is reduced and the accuracy of a learning algorithm improves or remains the same. Classification accuracy is defined as the percentage of test examples correctly classified by the algorithm. Measuring accuracy on a test set of examples is better than using the training set because examples in the test set have not been used to induce concept descriptions. Using the training set to measure accuracy typically provide an optimistically biased estimate and can result in misleading estimates due to over specialization of the learning algorithm to the data.

Data partitioning into training and testing sets can be made in different ways. A learning algorithm is trained on training data and accuracy is predicted by making test on testing set. Doing this provides a more reliable estimate of the true accuracy of an algorithm. Holdout and cross-validation are two common techniques for assessing classifier accuracy, based on randomly sampled partitions of the given data [17].

### 2.6.1 Holdout Method

In the holdout method, the given data were randomly partitioned into two independent sets, a training set and a test set. Typically, two thirds of the data are allocated to the training set, and the remaining one third is allocated to the test set. The training set was used to derive the classifier, whose accuracy was estimated with the test set. Random sampling was a variation of the holdout method in which the holdout method is repeated *k* times. The overall accuracy estimated was taken as the average of the accuracies obtained from each iteration [17].

### 2.6.2 Cross-validation

In *k*–fold cross-validation, the data was randomly split into *k* mutually exclusive subsets or “folds” of approximately equal size. A learning algorithm was 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 was the overall number of correct classifications from the *k* iterations, divided by the number of examples in the initial data [17].

# CHAPTER 3
# DESIGN OF THE PROPOSED SYSTEM

The main target of this thesis is to analyze the effect of feature selection and the new trend of MapReduce on machine learning. Six different datasets are used as case study for experiment. They are Customer Churn Prediction dataset, Insurance Classification dataset, Intrusion Detection dataset, Thyroid Disease Diagnosis dataset, Diabetics Diagnosis dataset and Human Activity Recognition dataset. Chi-square method is used as filter feature selection model, i.e the optimal feature subset is chosen without reference to classifier model construction and then the result optimal feature subset is fed into the classifier model construction. For classifier model construction, Artificial Neural Network with Backpropagation learning algorithm is chosen. Performance evaluation of machine learning classifier is made using holdout method. It is implemented on the Linux platform with java programming language.

## 3.1 Overview of the Proposed System

Figure 3.1 shows the overview of the proposed system. Data preprocessing is performed by substituting missing values from the input dataset. The proposed system is built by integrating feature selection and artificial neural network classifier. Feature selection is performed to select optimal features from the original dataset by using Chi-square based feature selection method. Each feature of the optimal feature subset can have different range of values. Processing these data that have different range of values directly in artificial neural network training can have effect on the efficiency of classifier. Therefore, the result feature subset undergoes a normalization process before applying these data in artificial neural network training. After normalization process, the data is stored on local file system by splitting it into training and testing set. The data is split randomly using holdout method. Two thirds of the data are to the training and remaining one third is allocated as the test sets.

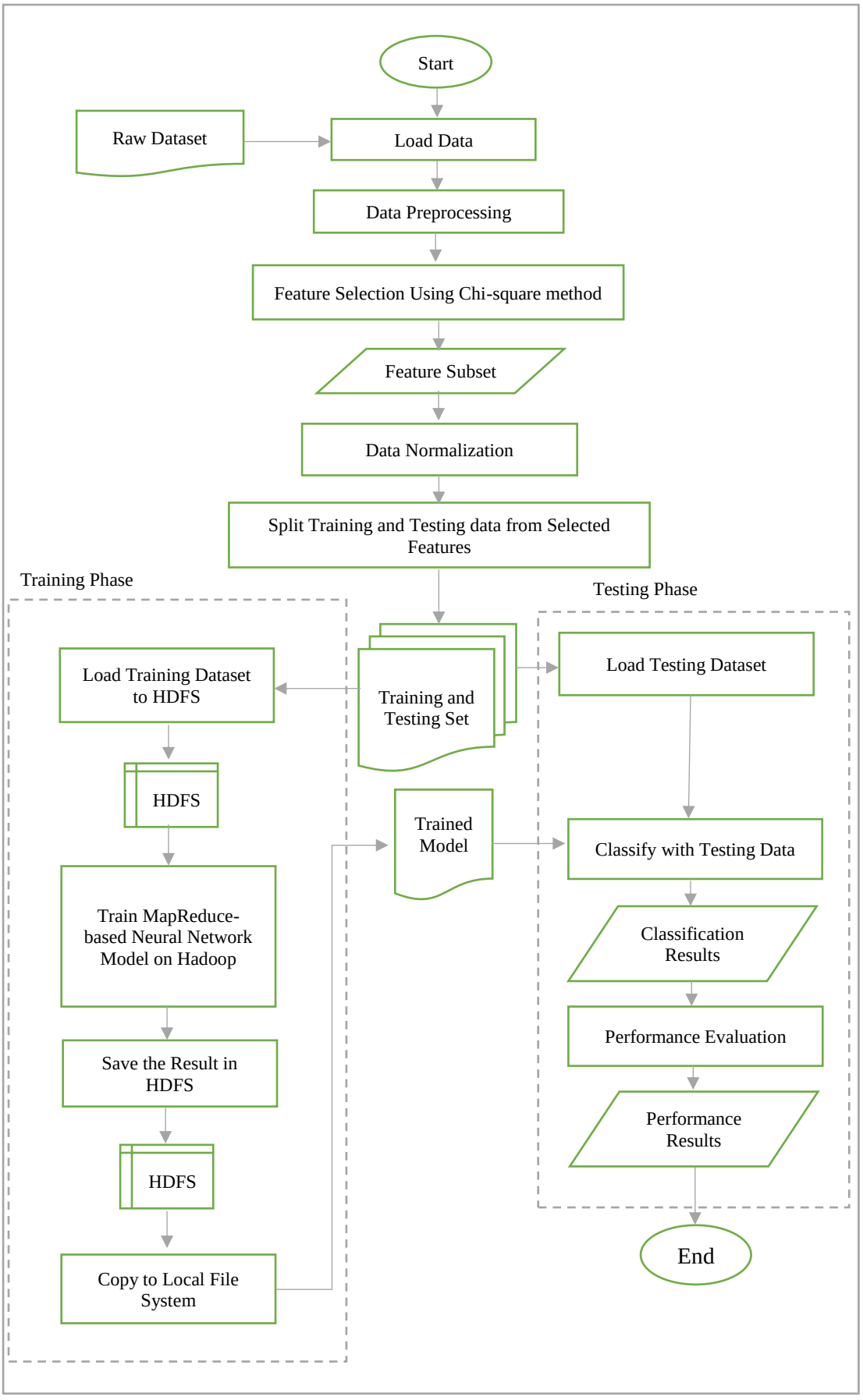



**Figure 3.1** Overview of the proposed system

Convention Backpropagation Neural Network (BPNN) can be trained using the training set on local file system. To train MapReduce-based Backpropagation Neural Network (MRBPNN) model, the training set is needed to load into the Hadoop File System (HDFS). MRBPNN training is performed on Hadoop with the loaded training set. After model training is finished, the generated trained model is saved in HDFS and then copied it into local file system to make performance evaluation.

After the training phase of the proposed system is completed, performance evaluation is performed. To make performance evaluation, the testing dataset is loaded from holdout split of the original dataset. After the testing dataset is loaded, the target class for each records of testing data are predicted using artificial neural network model that was successfully trained in the training phase. In the testing phase of the proposed system, only feedforward process that calculates the output values is performed using the weight values from the trained model. The target class of testing record is predicted according to the output values of feedforward process. Training time, accuracy and number of selected features are the key metrics of performance evaluation of the proposed system. The detail of performance evaluation is described in the next section.

In general, the proposed system is divided into three major stages.

**Stage 1**: Preprocessing of the data by substituting missing value.

**Stage 2**: Optimal features are selected by the aid of Chi-square feature selection method.

**Stage 3**: Artificial neural network classifier for predicting unknown class label by using holdout performance evaluation method.

In Stage 1, the entire dataset is preprocessed by RStudio. In stage 2, the Chi-square feature selection method is employed to identify the best optimal feature subset for classification. The optimal features that filtered by Chi-square feature selection method are used for classification by ANN in Stage 3.

## 3.2 Preprocessing

In this stage, data cleaning is performed as a preprocessing step. Data cleaning makes data to be more consistent by checking and validations. This stage converts the data to get standard formats by fixing data ranges, encoding to standard file format like excel sheets or database formats depending on the model used. The downloaded data file is firstly converted to ".csv" file format. After that, missing value is checked by RStudio editor. If the missing value is found, arithmetic mean substitution is used to

replace missing value in that editor. After data preprocessing is finished, the dataset is ready to use for feature selection.

## 3.3 Feature Selection Using Chi-square Method

After the original dataset has been preprocessed by substituting missing values, feature selection is performed to select the optimal feature subset. Feature selection is one of the most fundamental tasks in the workflow of machine learning for classification. The number of features needed to successfully perform a given classification task depends on the discriminatory qualities of the selected features. The kind of selection process ensures that those features which are highly skewed towards the presence of a particular class label are picked for the learning process.

In this stage, Chi-square feature selection method is used to select the most important features that can support to decide the target classes of classification problem. It is used as a filter model because feature selection is performed regardless of classifier model. Chi-square method is used to test independence level to identify whether there is a considerable relationship between two attributes. The $\chi^2$ value is computed as

$$\chi 2 = \sum_{i=1}^{R} \sum_{j=1}^{C} \frac{(o_{ij} - e_{ij})^2}{e_{ij}} \quad (3.1)$$

Where,

$o_{ij}$ = observed frequency (actual count)

$e_{ij}$ = expected frequency which is computed as

$$e_{ij} = \frac{n_{*j} n_{i*}}{n} \quad (3.2)$$

Where,

$n_{i*}$= total number of samples with $i^{th}$ the feature value.

$n_{*j}$= total number of samples in class j.

n = total number for samples.

Hypothesis testing needs to specify a confidence level that is also known as significance level. Features with P-values which are greater than critical value is selected as optimal features and be processed in the classification stage. If not, those features are eliminated. Chi-square hypothesis consists of the following steps:

- **Step 1:** Calculate the number of degree of freedom. Degree of freedom equals to the number of features minus one. If there are 306 features in the dataset, the degree of freedom is 306-1=305.

- **Step 2:** Specify the confidence level that the phenomenon would occur by chance. The probability of 0.01 is used as confidence level because many researchers recommend this value.
- **Step 3:** Calculate P-value by the calculated $\chi 2$ value and degree of freedom.
- **Step 4:** Make hypothesis testing in this step. Each feature is tested whether support or reject the null hypothesis by its P-value. The null hypothesis assumes that there is no significant difference between the two attributes. If the P-value is less than or equal to chosen confidence level (0.01), then the null hypothesis is rejected, otherwise it accepts the null hypothesis. Features which reject the null hypothesis are selected as optimal features.

Chi-square method extracts the list of optimal features and ranks them by the importance level of each feature. The resultant optimal feature subset can be used for model training and testing to access the accuracy of the classifier. Processing the optimal feature subset directly in the artificial neural network model training is time consuming. Therefore, feature subset undergoes a normalization process that transformed each data of feature subset in a particular range [0, 1]. After normalization process finishes, the normalized dataset is split randomly into training and testing set by holdout method. Finally, the training and testing sets with optimal features are ready to apply in the artificial neural network classifier for classification.

## 3.4 MapReduce-based Backpropagation Neural Network Algorithm

Artificial Neural Network is used to train the data for recognition model. The artificial neural network is learned by Backpropagation Learning Algorithm. The sigmoid activation function is used that support the output range of calculation to squeeze between 0 and 1. The theorical background of Backpropagation Learning Algorithm and sigmoid activation function is already discussed in Chapter 2.

In case of BPNN in MapReduce, each mapper constructs one BPNN for each input split (chunk). The number of mappers is determined by the number of splits for the input path. In this experiment, input split size is specified as four megabytes. The number of mappers is equal to the number of input splits. Each split is responsible for each mapper using the map function. The reducer part takes the results of individual mappers and processes them to get the final result. The idea of the model is to build individual classifier on each split. And the reducer chooses the best weight that fit in

the smallest error rate from all of these classifiers and save the final trained results to HDFS. The detail of MapReduce-based backpropagation neural network algorithm is described in Figure 3.2.

**Algorithm: MapReduce-based backpropagation neural network algorithm**

**Input:** The user provides the dataset name and number of hidden layers via input arguments.

**Output:** The final received Weights are the trained result we want.

**m** mappers and **one** reducer

Each mapper constructs one BPNN with input node (iNode), output node (oNode) and hidden node (hNode) that was computed by iNode and oNode for each data split.

1. Initialize Randomize Weights ()
2. For iteration <= maxEpoch do
3. Each $neuron_j$ of hidden layer computes

$$I_{jh} = \sum_{i=1}^{n} w_{ij}\ a_i + \theta$$

$$o_{jh} = \frac{1}{\left(1+e^{-I_{jh}}\right)}$$

4. Each $neuron_j$ in output layer computes

$$I_{jo} = \sum_{i=1}^{h} w_{ij}\ o_{jh} + \theta$$

$$o_{jo} = \frac{1}{\left(1+e^{-I_{jo}}\right)}$$

5. In each output, compute

$$Err_{jo} = o_{jo}(1-o_{jo})\ (target_j - o_{jo})$$

$$mse = (target - o_{jo})^2$$

$$E[e^2] = E[e^2] + mse$$

6. In hidden layer, compute

$$Err_{jh} = o_{jh}(1-o_{jh})\ \sum_{i=1}^{n} Err_i w_{io}$$

7. For all weight between input and hidden layer do

$$\Delta w_{jh} = w_{ij} + Err_{jh} * o_{jh}$$

8. End for
9. For all weight between hidden and output layer do

$$\Delta w_{jo} = w_{ij} + Err_{jo} * o_{jh}$$

10. End for
11. iteration++;
12. End for
13. $E[e^2] = E[e^2]$/mapper input size

14. For all weight between input and hidden layer

output <key, value> pair: <$w_{jh}$, $E[e^2]$+':'+$\Delta w_{jh}$ >

15. End for

16. For all weight between hidden and output layer

output <key, value> pair: <$w_{jo}$, $E[e^2]$+':'+$\Delta w_{jo}$>

17. End for

**Mapper process finish**

18. Reducer collects weight values that have the same key and chooses the best weight value from them

19. For all values of same key do

bestWeight = weight value with the smallest error

20. end for

21. output <key, value> pair: <key, bestWeight>

**End**

**Figure 3.2** MapReduce-based backpropagation neural network algorithm

In order to train MapReduce-based Backpropagation Neural Network, it is needed to provide the dataset name and the number of hidden layers. The algorithm is trained according to the training data of the user's given dataset and the number of hidden layers. After that, layer nodes for input dataset are prepared for each layer. The number input nodes in input layer is the number of input features and the number of output nodes in output layer is the number of class label. The number of hidden nodes for each hidden layer is calculated by 2/3 the size of the input layer plus the size of the output layer. After layer nodes have been prepared, weight values for each link between layers are initialized by using randomly weights.

After the parameters and structure for artificial neural network is ready, each mapper constructs one backpropagation neural network. There is 'n' number of backpropagation neural networks according to block size of dataset. In addition, all of these neural networks have exactly the same parameters and structure. Initially, the MapReduce job first maps input record to a combination of <key, value> pairs where byte offset is the key and the whole input record represents the value. Then, each neural network processes the feedforward process (Step 3 to 4 in algorithm) that computes the output values for each node of each layer. After feedforward process is finished, each mapper's neural network starts the back-propagation process (Step 5 to 10 in algorithm). It computes the error values and updates weights values to get new weight

values for each neuron in each layer. Repeat the feed forward and back-propagation process until all the instances have been processed for training and the maximum iteration is satisfied. At last, each mapper outputs and sends intermediate outputs to reducer in the form of $<w_{jh}, E[e^2]+':'+\Delta w_{jh}>$ where $w_{jh}$ is the key that represents the link index and $E[e^2]+':'+\Delta w_{jh}$ indicates the value in which two combination of global average error and weight value for the link index.

After mapper process is finished, the reducer collects and merges all the outputs of mappers by grouping key value of mapper's output. It chooses the best weight value from a set of weight values that have the same key. The chosen weight values that has the smallest global average error then generates as reducer's output in the form of <key, bestWeight> where key is the link index between layers and bestWeight indicates the chosen weight value for the link index. Finally, the outputs of reducer are then written to Hadoop File System (HDFS) and copied to local file system. In this case, the final trained result represents the trained model that is good fit for the training data. The trained model is then used for performance evaluation that is discussed in the next section.

## 3.5 Performance Evaluation

Three key performance evaluators are used. They are number of features selected, training time and classification accuracy. Using these evaluators, the comparison analysis is performed on single hidden layer implementation of ANN and two different hidden layer implementation of ANN using six different datasets.

For single hidden layer implementation of ANN, the following four comparisons are made on MapReduce-based Neural Network, WEKA tool and Conventional Neural Network.

1) Training time comparison of classifiers on complete feature
2) Accuracy comparison of classifiers on complete features
3) Training time comparison of classifiers on features selected subset
4) Accuracy comparison of classifiers on features selected subset

For two different hidden layer implementation of ANN, two more analysis are made on MapReduce-based Neural Network.

1) Training time comparison between two different hidden layers
2) Accuracy comparison between two different hidden layers

All experiments are carried out using holdout method. The given data is randomly partitioned into two independent sets:

1) the training set (used for learning classifier) and
2) the testing set (used for evaluating classifier)

Two thirds of the data are allocated to the training set, and remaining one third is allocated as the test set.

The classifier's accuracy, the possibility of the algorithm that is able to correctly predict positive and negative tuple, is calculated by the following equation.

$$Accuracy = \frac{TP+TN}{Total\ Number\ of\ Test\ Data} \quad (3.3)$$

Where,

TP = positive tuples correctly classified as positive

NP = negative tuples correctly classified as negative

## 3.6 Datasets Used in the Experiment

To evaluate the performances of the system, six different datasets are used as case study. Diabetic Diagnosis and Intrusion Detection (KDD 99), Customer Churn Prediction (KDD 2009), Thyroid Disease Diagnosis, Insurance Classification, Human Activity Recognition. Two different datasets, Thyroid Disease Diagnosis and Intrusion Detection, are taken from UCI machine learning repository. The other three datasets, Insurance Classification, Customer Churn Prediction and Human Activity Recognition, are taken from OpenML repository and the remaining one is taken from GitHub repository. The main characteristics of these datasets: the number of instances, number of attributes, feature type, missing values exit or not, the train/testing size that randomly split by holdout method and number of classes are as shown in Table 3.1.

**Table 3.1** Dataset description

| Dataset Name | No. of Instances | No. of features | Feature Type | Missing Values | Train/ Testing Size | No. of classes |
|---|---|---|---|---|---|---|
| Diabetics Diagnosis | 101767 | 37 | numeric, nominal | yes | 68183/ 33584 | 2 |
| Intrusion Detection | 125975 | 41 | numeric, nominal | no | 84403/ 41572 | 2 |
| Customer Churn Prediction | 15333 | 306 | numeric | no | 10272/ 5061 | 2 |

| Dataset Name | No. of Instances | No. of features | Feature Type | Missing Values | Train/ Testing Size | No. of classes |
|---|---|---|---|---|---|---|
| Thyroid Disease Diagnosis | 9172 | 29 | numeric, nominal | yes | 6145/ 3027 | 2 |
| Insurance Classification | 9823 | 85 | numeric | no | 6581/ 3242 | 2 |
| Human Activity Recognition | 4856 | 351 | numeric | no | 3254/ 1602 | 3 |

The brief description of each datasets are as follows.

**Diabetics Diagnosis:** This dataset contains 101,767 instances. The task is to distinguish whether the patients suffer diabetics cancer or not. There are 37 features whose values are numeric and nominal. There are 15 numeric features and 22 nominals.

**Intrusion Detection (KDD 99):** This is the dataset used to build a network intrusion detector, a predictive model capable of distinguishing between “bad" connections, called intrusions or attacks, and "good" normal connections. This dataset contains 125975 records with 41 number of features. Seven features are nominal and the remaining are continuous.

**Customer Churn Prediction (KDD 2009):** This dataset offers the opportunity to work on large marketing databases from the French Telecom company Orange to predict the propensity of customers to switch provider (churn). Churn rate is a measure of the number of people or items moving into or out of a group over a particular amount of time. The dataset contains 15,333 samples with 306 number of attributes and two number of classes. All features are continuous.

**Thyroid Disease Diagnosis:** This dataset contains 9172 records. The problem is to identify whether or not a patient suffered hypothyroid. It has 7 numeric features and the remaining are nominal.

**Insurance Classification:** The data was supplied by the Dutch data mining company Sentient Machine Research and is based on a real-world insurance business problem. This dataset contains 9823 customers’ records, including the information of whether or not they have a caravan insurance policy. Each record consists of 85 features and all of these features are numeric.

**Human Activity Recognition:** Human Activity Recognition (HAR) database built from the recordings of 30 subjects performing activities of daily living (ADL) while

carrying a waist-mounted smartphone with embedded inertial sensors. This dataset contains 4856 human's activity data, and the data is used to distinguish whether the person is LAYING, SITTING OR WALKING. Each record consists of 351 features and all of these features are numeric.

# CHAPTER 4

# IMPLEMENTATION OF THE PROPOSED SYSTEM

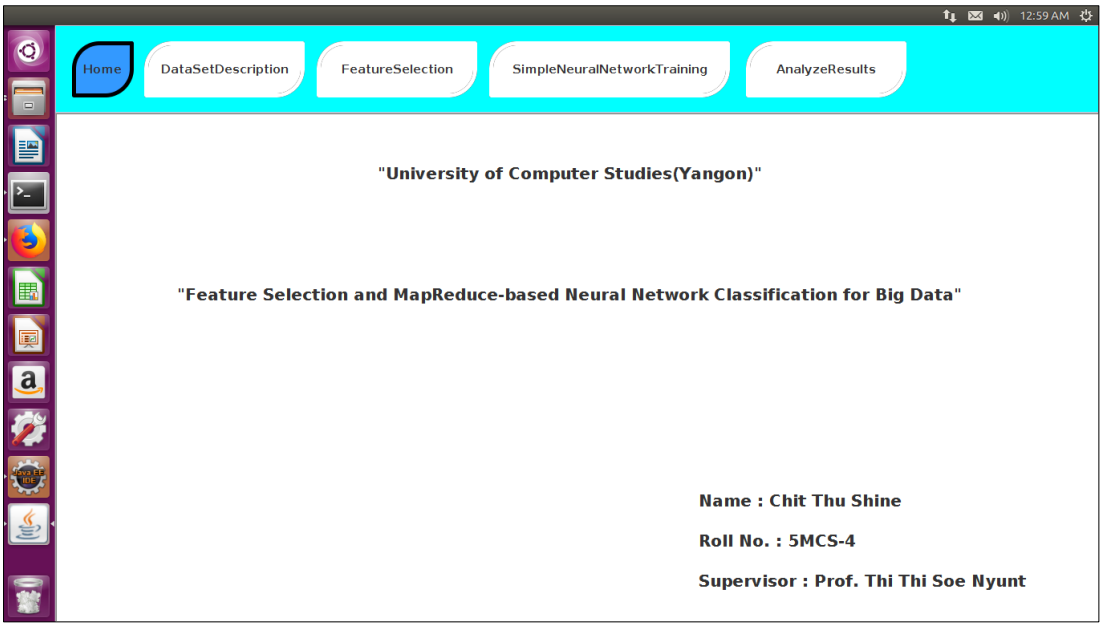
## 4.1 Experimental Setup

In order to execute and evaluate the proposed system, a Hadoop cluster (Pseudo Distributed Mode) is stablished on Linux Virtual Machine in VMWare Workstation. In Pseudo Distributed Mode, NameNode and DataNode are resided on the same computer, and master and slave servers actually run on the same server. The configuration for Linux Virtual Machine is as follows:

- Hardware configurations
  - CPU: Core i7@3 GHz with 4 number of core processors
  - Memory (RAM): 6 GB
  - Hard disk: 85GB
- Software requirements
  - Ubuntu 16.04
  - JDK 1.8
  - Hadoop version: 2.7.2 64 bits

## 4.2 Implementation of the System

When the system is started, the main interface screen is appeared as shown in Figure 4.1. The main form consists of four menus namely, DataSetDescription menu, FeatureSelection menu, SimpleNeuralNetworkTraining menu, and AnalyzeResults menu respectively. DataSetDescription menu is used for the detail description of the dataset used in the system. Feature selection before model training is made by using FeatureSelection menu. SimpleNeuralNetworkTraining menu is used to train conventional neural network. And, the last menu of the system, AnalyzeResults menu, allows to view the detail analysis results of the proposed system for all experiments.




**Figrue 4.1** Main form of the proposed system

The dataset description form is shown Figure 4.2 allows to see the detail description of the datasets used in this system. By choosing desire dataset from the left sidebar, the detail description of chosen dataset can be seen like Figure 4.3.


**Figrue 4.2** Dataset description form

The detail description of ThyroidDisease dataset is shown in Figure 4.3. The detail description of datasets includes number of instances, attributes, target classes and domain area where this dataset can be applied.

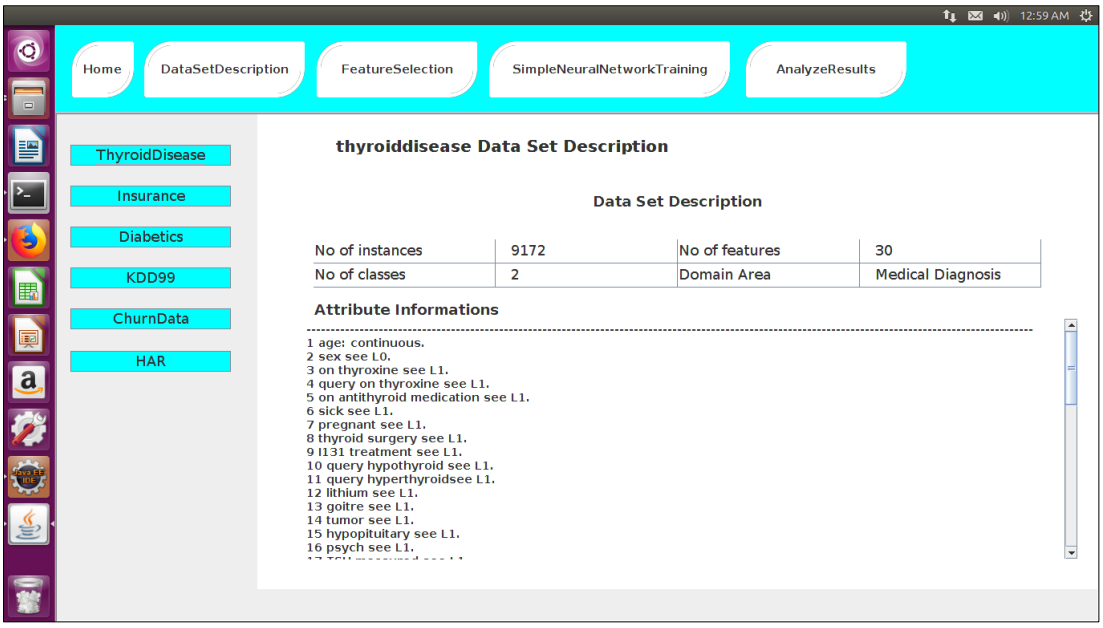



**Figure 4.3** Dataset description form for ThyroidDisease dataset

The feature selection form is shown in Figure 4.4 allows user to select features from the original datasets. By choosing desire dataset from the left sidebar, the system selects the optimal features and presents back to the user. The result of feature selection for ThyroidDisease dataset is shown in Figure 4.5.


**Figure 4.4** Feature selection form

Figure 4.5 presents a screenshot showing the feature selection results of ThyroidDisease dataset. The calculated Chi-square value, P-value, and induction whether accept or reject the null hypothesis of each attribute can be seen. And the number of selected optimal features by ranking is also listed. After feature selection

process has finished, features selected subset is saved into the local file system by splitting training and testing set. Holdout method is used to randomly split the selected optimal features set. MapReduce-based Neural Network is trained using the training set. Therefore, the copy of the training set is then loaded into Hadoop Distributed File system (HDFS) by clicking button 'Load To HDFS'.

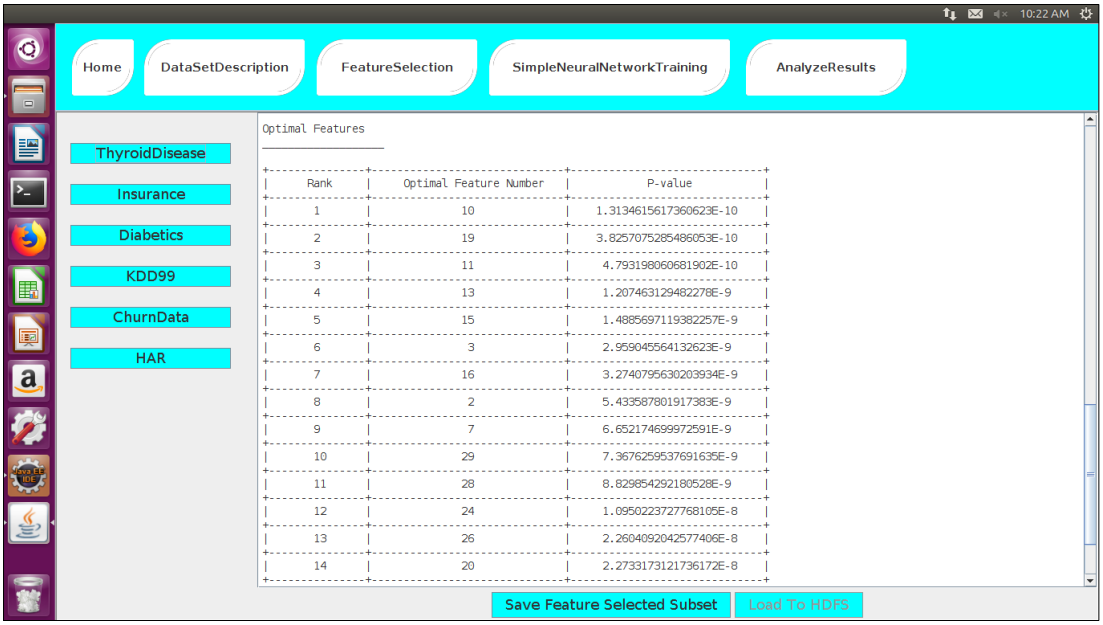



**Figure 4.5** Result of feature selection form for ThyroidDisease dataset

The loaded file can be checked by using the NameNode URL 'https://localhost:50070' on the web browser. Figure 4.6 shows the screenshot for the loaded training file into HDFS.

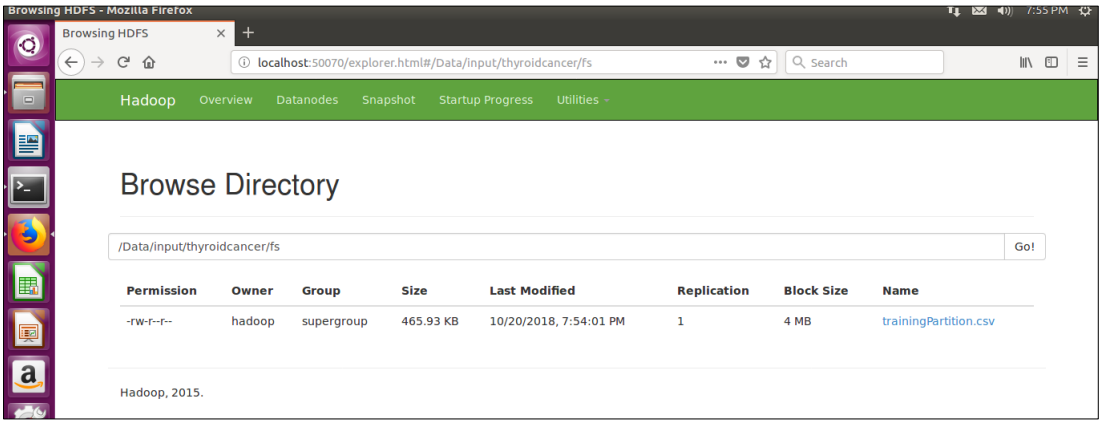



**Figure 4.6** The loaded file to HDFS

Conventional backpropagation neural network is trained by using SimpleNeuralNetworkTraining menu shown in Figure 4.7. The dataset can be chosen

to train it from the combo box. Figure 4.8 shows the conventional backpropagation neural network training for ThyroidDisease dataset.

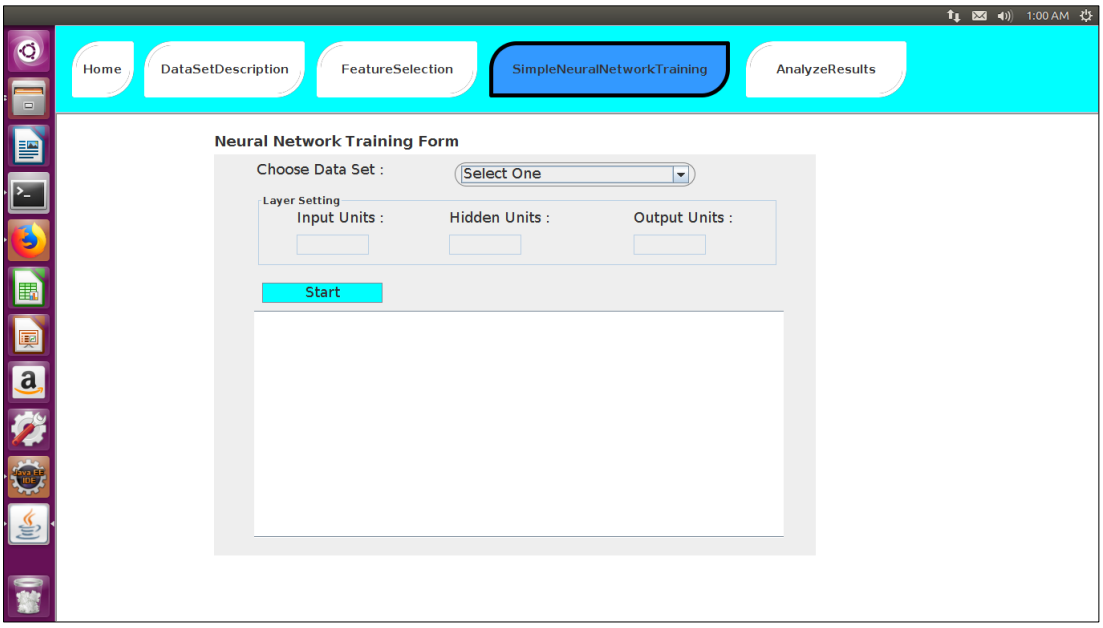



**Figure 4.7** Conventional neural network training form

Figure 4.8 presents a screenshot showing the training form of the conventional backpropagation neural network algorithm. By selecting the desire dataset; the system will automatically fill the number of input node and number of output node based on the dataset. The number of hidden nodes is then calculated by the system based on the number of input nodes and output nodes of the dataset. 'Start' button is used to train Conventional Neural Network.

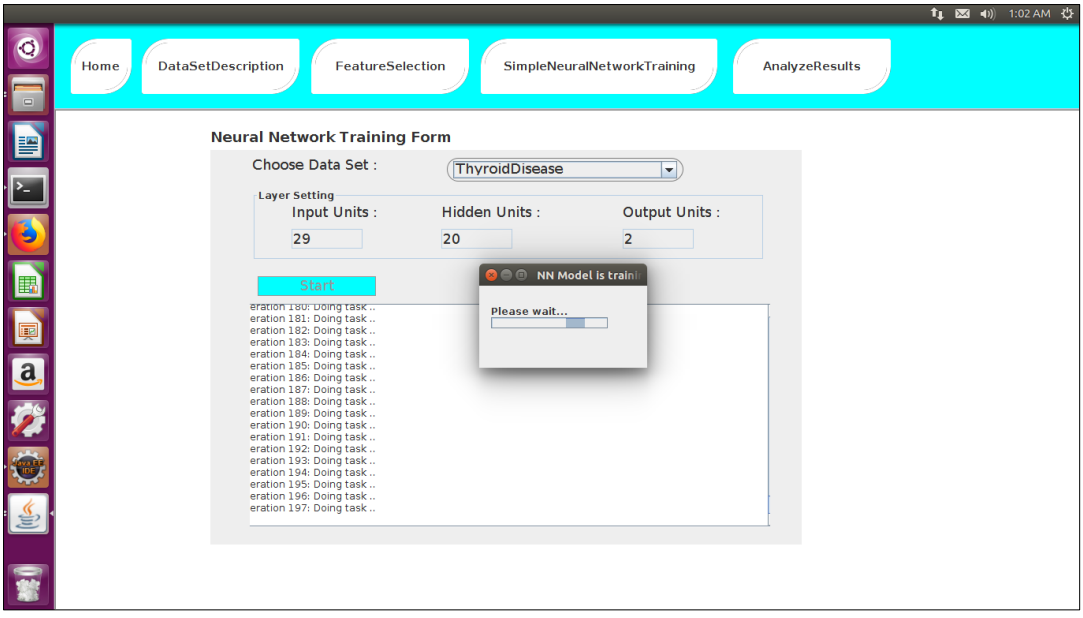



**Figure 4.8** Conventional neural network training form for ThyroidDisease dataset

After the training of conventional neural network is finished, the training time and accuracy is shown like in Figure 4.9. The trained neural network model is then saved into local file system. This model can be used for evaluation in the next time.

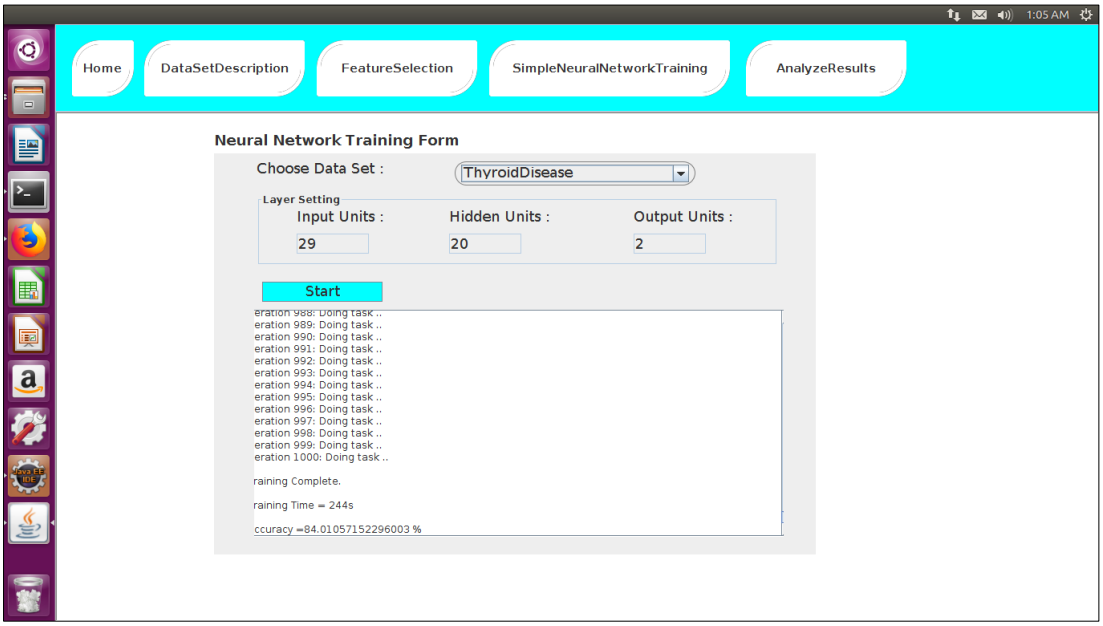



**Figure 4.9** Result of conventional neural network training result form for ThyroidDisease dataset

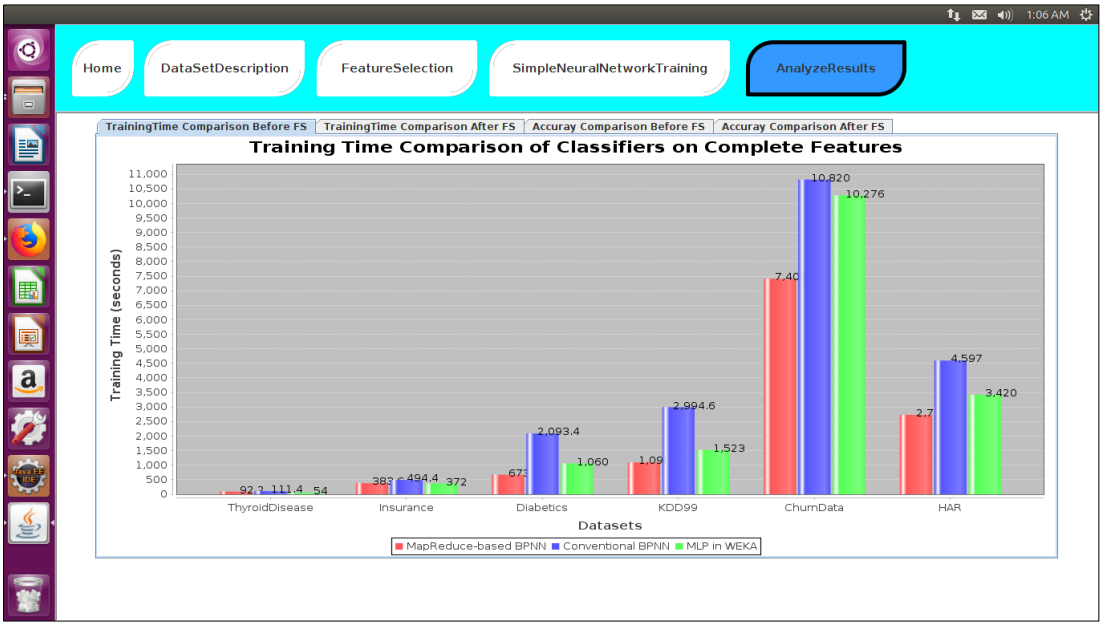



**Figure 4.10** Analysis results form that show the training time comparison of classifiers on complete features

The detail comparison results can be seen by the final menu, AnalyzeResults. Training time and accuracy comparison of classifiers on complete features and features

selected subset are shown. By clicking AnalyzeResults menu, the four variants of comparison can be seen. Figure 4.10 shows training time comparison of classifiers on complete features. Figure 4.11 shows training time comparison of classifiers on features selected subset. The accuracy comparison of classifiers on complete features and features selected subset can be seen in Figure 4.12 and Figure 4.13 respectively.

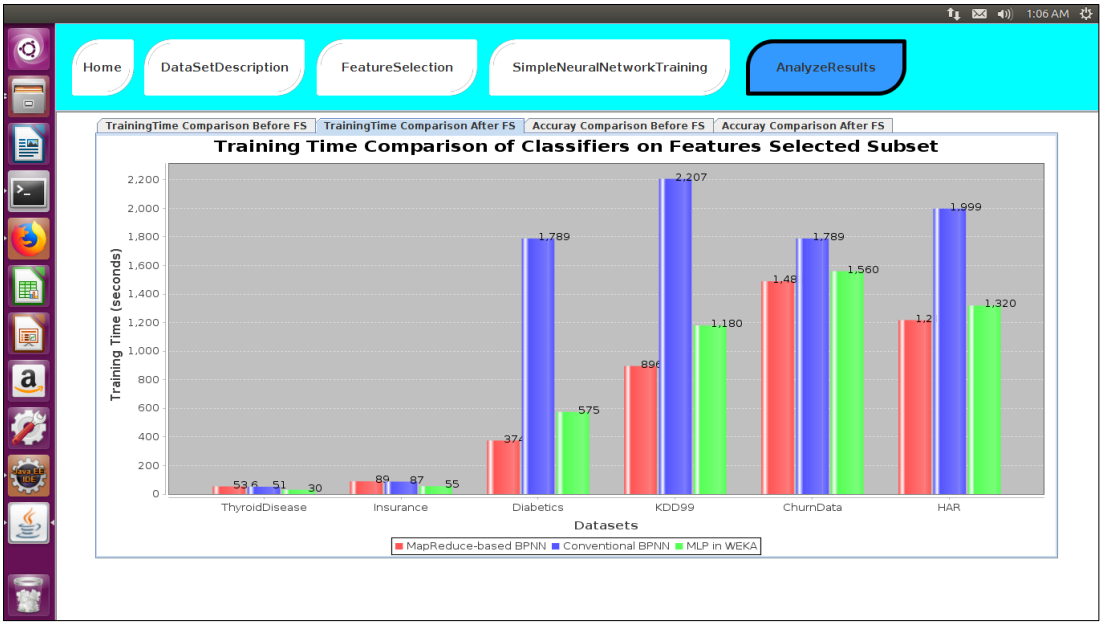



**Figure 4.11** Analysis results form that show the training time comparison of classifiers on features selected subset

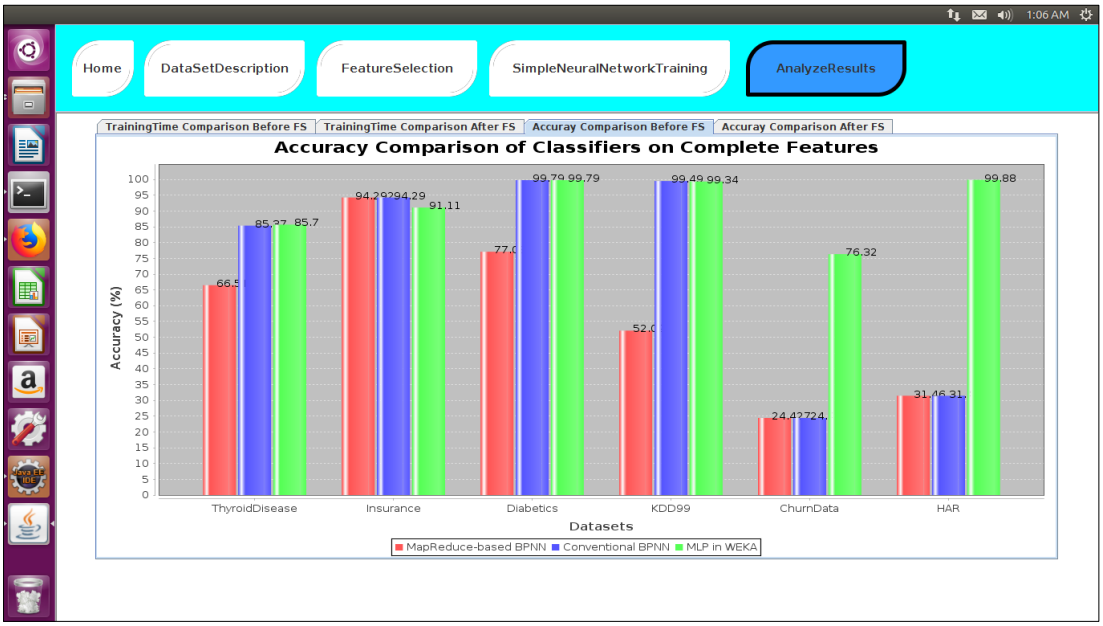



**Figure 4.12** Analysis results form that show the accuracy comparison of classifiers on complete features

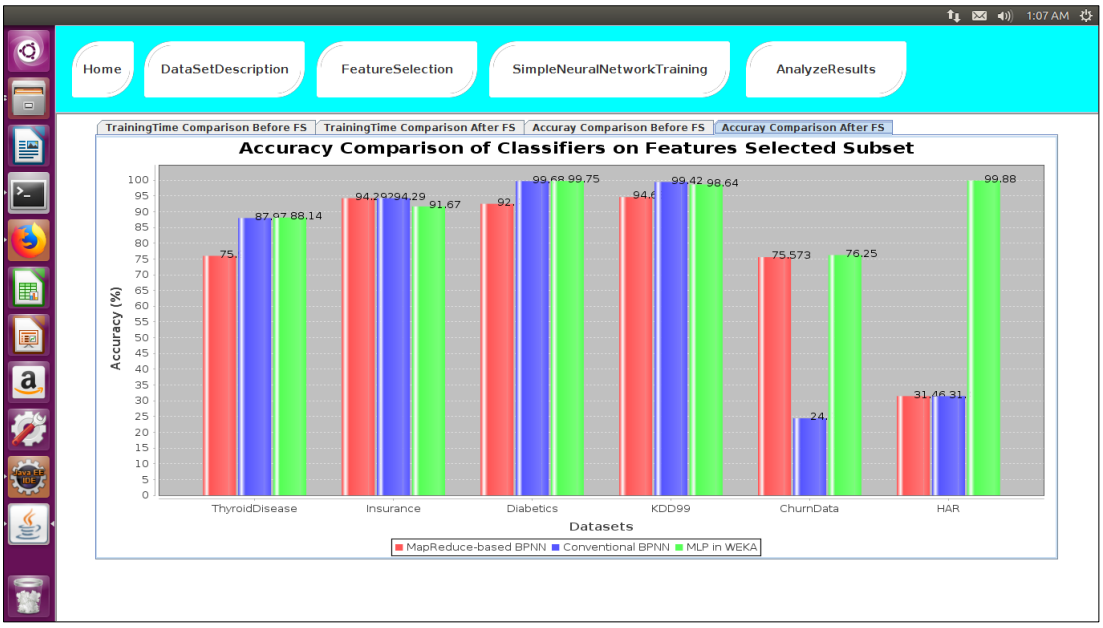



**Figure 4.13** Analysis results form that show the accuracy comparison of classifiers on features selected subset

## 4.3 MapReduce-based Backpropagation Neural Network Job Execution

In order to execute MapReduce-based Neural Network program in Hadoop, the actions are performed. Initially, all of the daemons of Hadoop are needed to run by typing "start-all.sh" script in Linux terminal. If all of Hadoop daemons are running successfully, the program can be run in Hadoop. To run the program in Hadoop, the input and output directories are needed to create in HDFS by the commands "hdfs dfs –mkdir inputfilelocation" and "hdfs dfs –mkdir outputfilelocation". The file that the user wants to process is needed to upload from local machine to HDFS by specifying the file location. To achieve that, the command "hdfs dfs –copyFromLocal trainingPartition.csv /inputfilelocation/" is run which copy the data from local file system to the folder of /inputfilelocation/ in HDFS.

After the input file for model training and the output directory are ready, the next step is to run script in the Linux command line that is to execute MapReduce-based Neural Network program in Hadoop.

```
yarn jar MRBNN.jar datasetname noOfHiddenLayer
```

The yarn jar command is a built-in algorithm of MapReduce offered by Apache Hadoop. In this command, the dataset name and number of hidden layers are needed to

specify. According to the dataset name and number of hidden layers, it is constructed a neural network using corresponding input and output directory.

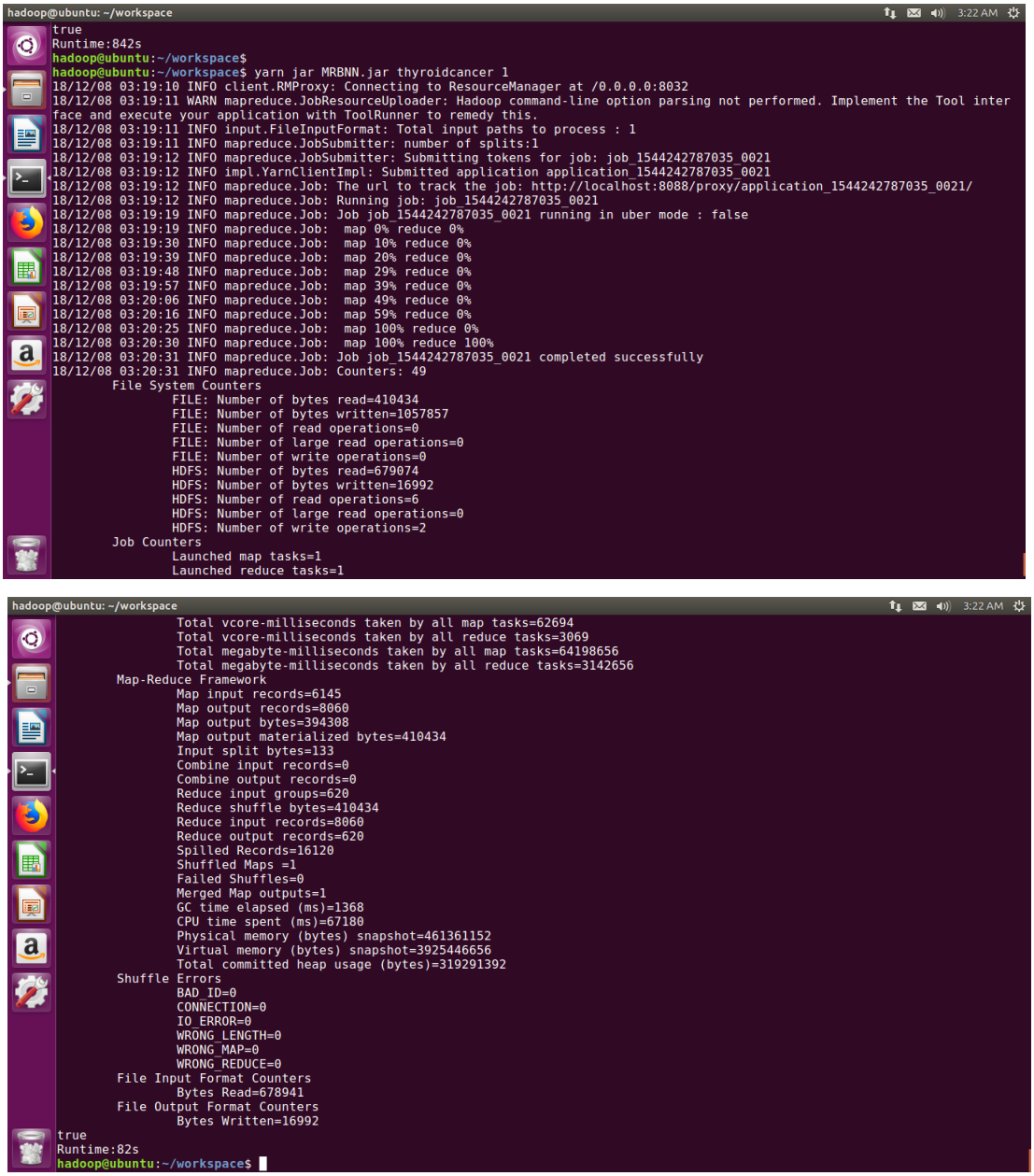



**Figure 4.14** MapReduce-based neural network job execution

Figure 4.14 shows the neural network training for ThyroidDisease dataset in Hadoop with MapReduce paradigm. In this Figure 4.14, it is depicted the output from execution of the MapReduce job. It can be seen that the mapping and reducing procedure with percentage during the action. The detail of job execution can also be found by exploring the task's link that is shown in Figure 4.15. This figure shows the elapsed time for each map task of the job execution. After the job is successfully processed, it outputs the status 'true'. The amount of time required for the execution of

the job is calculated by subtracting the finish timestamp and the start time of the job. The file produced from the execution can be seen by typing the output file directory in browser. The screenshot for the output file in HDFS after the execution of MapReduce job is shown in Figure 4.16. This output file 'part-r-00000' is the training result of the execution and it is used in performance evaluation with testing set.

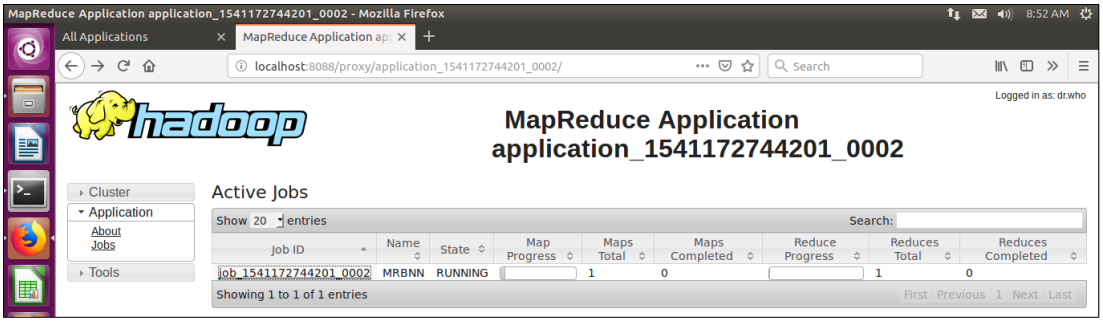



**Figure 4.15** Elapsed time of each map task for MapReduce job execution

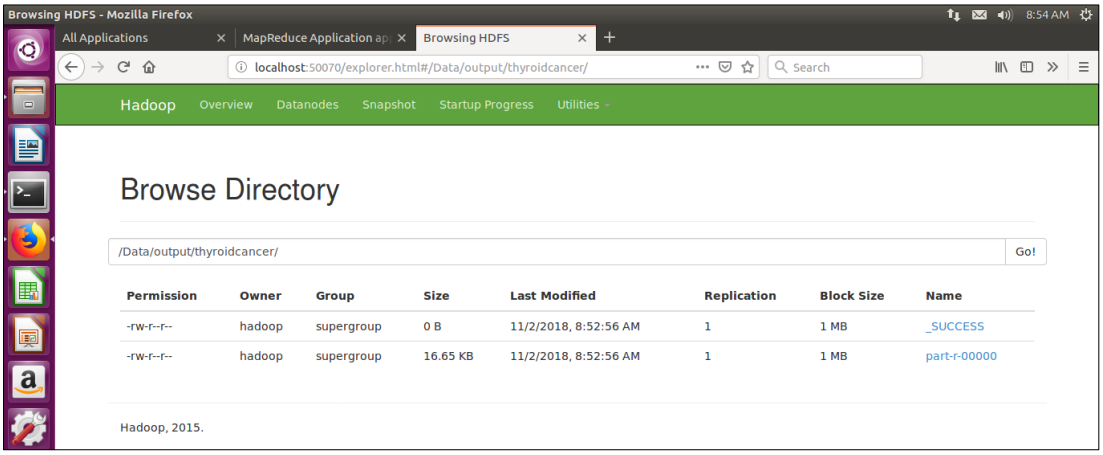



**Figure 4.16** The HDFS file after the execution of MapReduce job

## 4.4 Experiment Result

The analysis includes comparison of three Neural Network models: MapReduce-based BPNN, Multi-Layer Perceptron (MLP) that was implemented in the tool of WEKA 3.8 and Conventional BPNN. Chi-square feature selection method is employed to select features before passing the datasets to the classifier. By using Chi-square feature selection algorithm, it reduced features from 29 to 16, 37 to 26, 41 to 35, 85 to 29, 306 to 126 and 351 to 223 in Thyroid Disease, Diabetic, KDD99, Insurance, Churn Data and HAR datasets respectively. The result details of the average training time in seconds and classification accuracy are presented before feature selection and

after feature selection in Table 4.1. The result details of number of selected features, training time in seconds and accuracy of these three models are presented in that table.

**Table 4.1** Experimental result in details

| Dataset Name | All attributes without feature selection | | | | | | | Selected features set using Chi-square method | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | #of features | MapReduce-based BPNN | | MLP in WEKA | | Conventional BPNN | | #the number of selected features | MapReduce-based BPNN | | MLP in WEKA | | Conventional BPNN | | |
| | | Time (sec) | Accuracy | Time (sec) | Accuracy | Time (sec) | Accuracy | | Time (sec) | Accuracy | Time (sec) | Accuracy | Time (sec) | Accuracy | |
| Diabetics Data | 37 | 674 | 77.081 | 1060 | 99.79 | 2093 | 99.79 | 26 | 374 | 92.49 | 575 | 99.75 | 1789 | 99.68 | |
| KDD 99 | 41 | 1095 | 52.1 | 1523 | 99.34 | 2995 | 99.49 | 35 | 896 | 94.64 | 1180 | 98.64 | 2207 | 99.42 | |
| Churn Data | 306 | 7409 | 24.25 | 10276 | 76.32 | 10820 | 24.43 | 126 | 1490 | 75.57 | 1560 | 76.25 | 1789 | 24.43 | |
| Thyroid Disease | 29 | 92 | 66.514 | 54 | 85.70 | 111 | 85.37 | 16 | 54 | 75.96 | 30 | 88.14 | 51 | 87.97 | |
| Insurance | 85 | 384 | 94.29 | 372 | 91.11 | 494 | 94.29 | 29 | 89 | 94.29 | 55 | 91.67 | 87 | 94.29 | |
| HAR | 351 | 2529 | 31.46 | 3420 | 99.88 | 4597 | 31.46 | 223 | 1218 | 31.46 | 1320 | 99.38 | 1999 | 31.46 | |

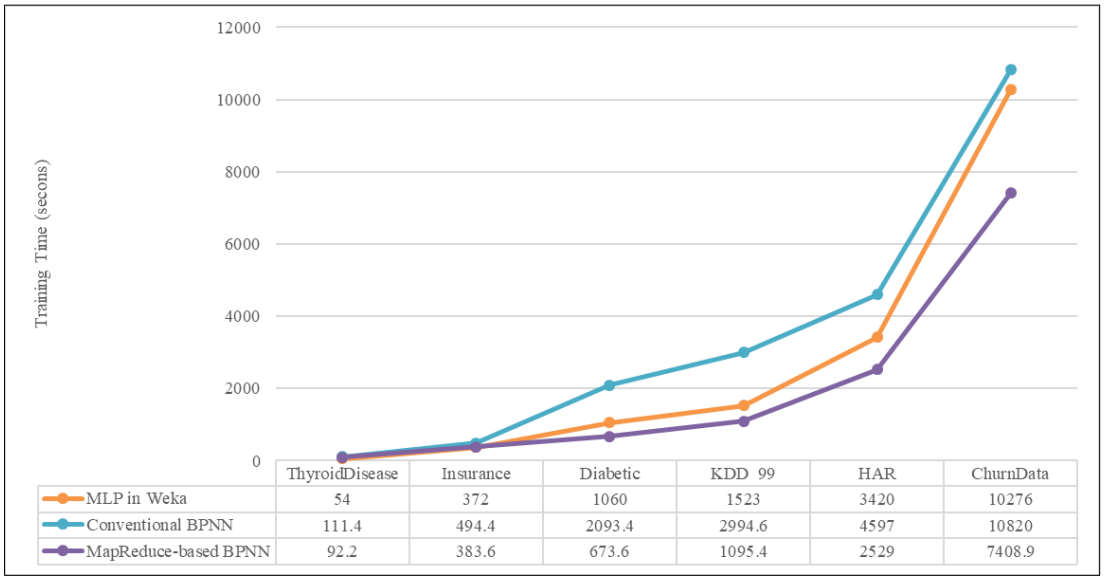
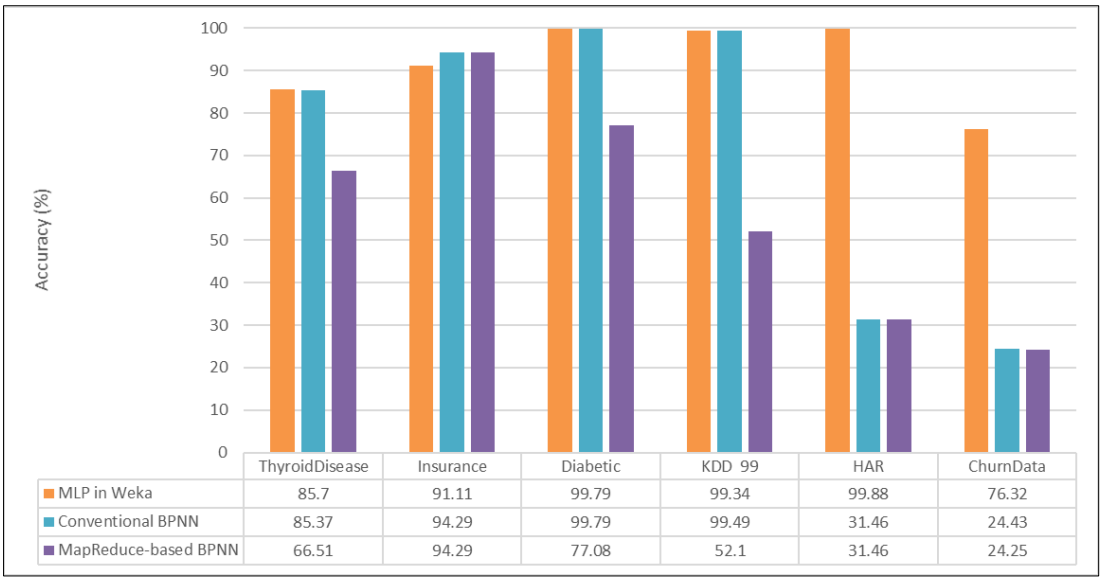



**Figure 4.17** Training time comparison of classifiers on complete features

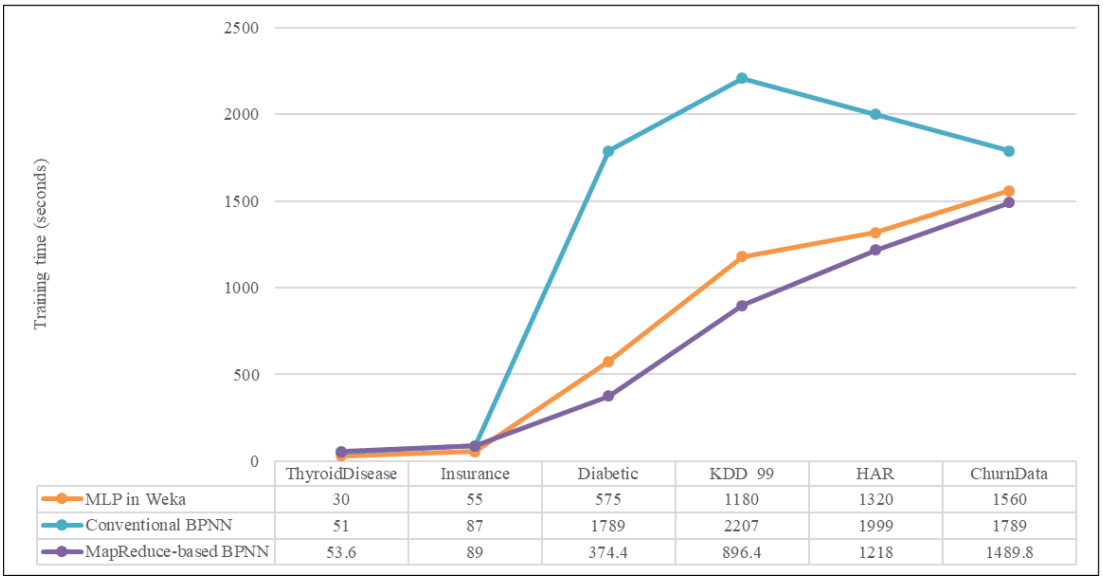
Figure 4.17 shows the training time comparison of the three models' construction on six datasets with complete features. It is observed that MapReduce-based BPNN is the fastest method out of the all when the dataset becomes large. Conventional BPNN is the laziest model to train for all datasets. According to Table 4.1 and Figure 4.17, training time of MapReduce-based BPNN is slightly longer than MLP in WEKA tool in small datasets (Thyroid Disease and Insurance).




| | ThyroidDisease | Insurance | Diabetic | KDD 99 | HAR | ChurnData |
|---|---|---|---|---|---|---|
| MLP in Weka | 85.7 | 91.11 | 99.79 | 99.34 | 99.88 | 76.32 |
| Conventional BPNN | 85.37 | 94.29 | 99.79 | 99.49 | 31.46 | 24.43 |
| MapReduce-based BPNN | 66.51 | 94.29 | 77.08 | 52.1 | 31.46 | 24.25 |

**Figure 4.18** Accuracy comparison of classifiers on complete features

Figure 4.18 shows the classification accuracy comparison of the three models on six datasets with complete features. According to the nature of MapReduce-based BPNN algorithm that trains on subset of dataset by splitting the original dataset, it is observed that the classification accuracy of the sequential execution MLP in WEKA tool is higher than the classification accuracy of MapReduce-based BPNN algorithm except from insurance dataset.

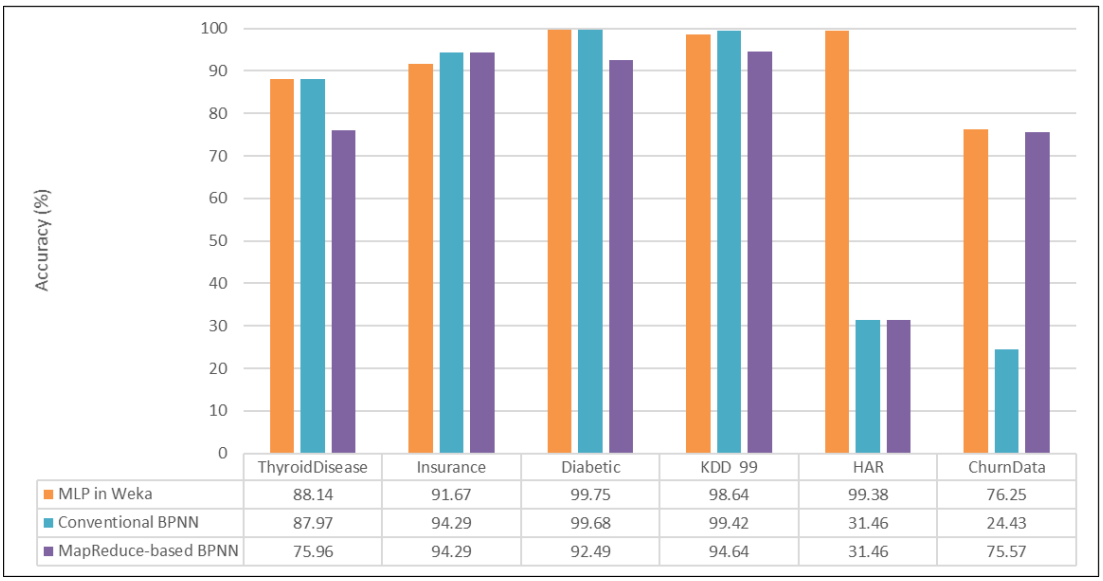



| | ThyroidDisease | Insurance | Diabetic | KDD 99 | HAR | ChurnData |
|---|---|---|---|---|---|---|
| MLP in Weka | 30 | 55 | 575 | 1180 | 1320 | 1560 |
| Conventional BPNN | 51 | 87 | 1789 | 2207 | 1999 | 1789 |
| MapReduce-based BPNN | 53.6 | 89 | 374.4 | 896.4 | 1218 | 1489.8 |

**Figure 4.19** Training time comparison of classifiers on features selected subset

Figure 4.19 shows the training time comparison of the three models on six datasets after making feature selection. These three models were constructed by passing features subset that was generated by Chi-square feature selection method. The detail

comparison results with the reduced number of features are presented in Table 4.1. By comparing Figure 4.17 and Figure 4.19, it can be seen that model construction time after making feature selection is significantly reduced for all of the three models on all of six datasets.




| | ThyroidDisease | Insurance | Diabetic | KDD 99 | HAR | ChurnData |
|---|---|---|---|---|---|---|
| MLP in Weka | 88.14 | 91.67 | 99.75 | 98.64 | 99.38 | 76.25 |
| Conventional BPNN | 87.97 | 94.29 | 99.68 | 99.42 | 31.46 | 24.43 |
| MapReduce-based BPNN | 75.96 | 94.29 | 92.49 | 94.64 | 31.46 | 75.57 |

**Figure 4.20** Accuracy comparison of classifiers on features selected subset

Figure 4.20 shows the classification accuracy comparison of the three models on six datasets after making feature selection. By comparing Figure 4.20 and Figure 4.18, it can be seen that MapReduce-based BPNN is affected by Chi-square feature selection method because the classification accuracy significantly increases on features selected subset. And it is also found that the accuracy of MLP in WEKA tool and Conventional BPNN for the Thyroid Disease dataset is increased from 85.7 to 88.14 and 85.37 to 87.97 respectively. The accuracy is stable in the remaining datasets by feature selection. According to these results, it is observed that feature selection can retain a suitably accuracy that represent in the complete features by selecting minimal features subset from a problem domain that helps to decrease training time.

The analysis is also made by increasing one more hidden layer in MapReduce-based BPNN algorithm implementation. The training time and accuracy comparison of MapReduce-based BPNN algorithm on two different hidden layers are shown in Figure 4.21 and 4.22 respectively. By comparing single hidden layer and two hidden layer implementation, the classification accuracy of two hidden layer implementation isn't so different from the single hidden layer implementation. One of the case studies, ThyroidDisease dataset, is only affected that tends to increase accuracy from 67% to

74% but the accuracy of the remaining case studies is stable. Although accuracy isn't so different, increasing one more hidden layer in MapReduce-based BPNN algorithm suffers more training time to build classifier model.

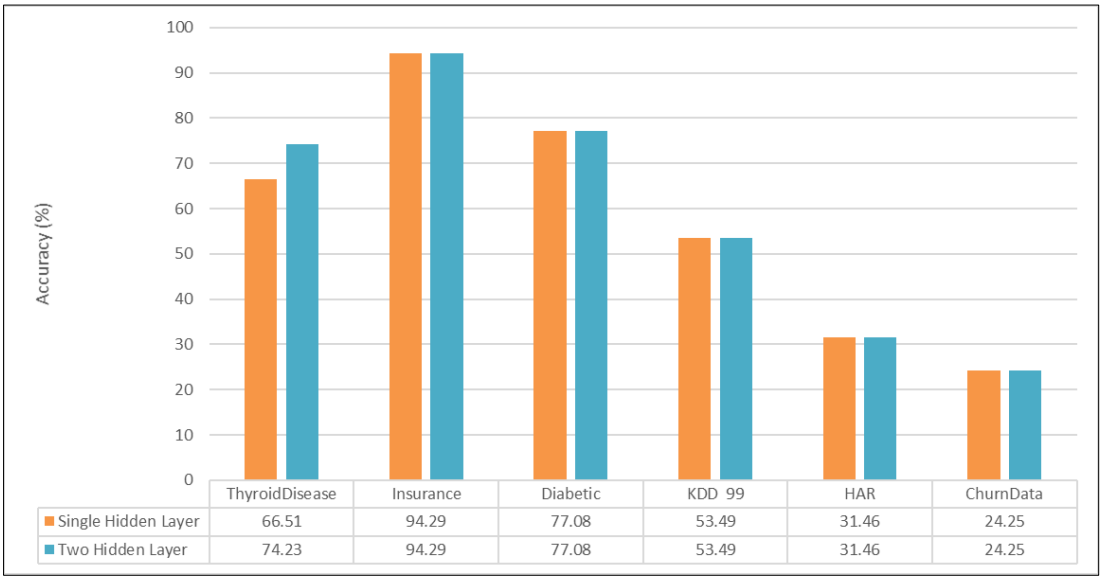



**Figure 4.21** Training time comparison of MapReduce-based BPNN on two different hidden layers

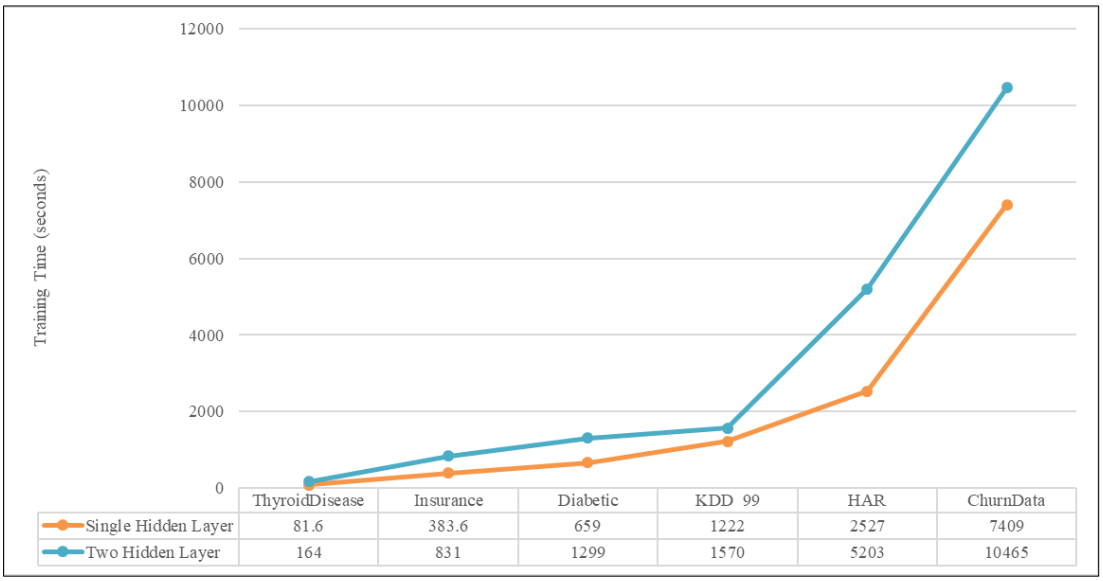



**Figure 4.22** Accuracy comparison of MapReduce-based BPNN on two different hidden layers

# CHAPTER 5

# CONCLUSION

## 5.1 Conclusion

A large amount of digital data is generated every data. This leads to high dimension of big data. To reduce the high dimension of big data, Chi-square method is applied and features are selected according to the threshold value. Then the selected features are divided into training and testing data and save in separate file. MapReduce based BPNN model is created to get the training model. Two types of BPNN model, only one hidden layer and two hidden layers are used in training the classification of BPNN model. Using the trained BPNN model, the classification performance is measured using the testing data. The experiment is conducted with six different datasets which vary from patient health data to customer data. The performance measures such as training time and accuracy are being analyzed. The system is designed and implemented using Java programming language on Linux platform. The experimental results are compared with WEKA and conventional model.

For BPNN classification model which has trained with only one hidden layer, four comparisons in training time and accuracy on complete features and selected features subset have made. It is observed that the MapReduce-based BPNN is the fastest method out of the all when the dataset becomes large while it is slightly longer than WEKA in small datasets (ThyroidDisease diagnosis and Insurance). Conventional BPNN is the laziest model to train for all datasets. Therefore, it can be concluded that MapReduce is a powerful framework for the processing and analysis of huge amounts of data. According to the nature of MapReduce-based BPNN algorithm that trains on subset of dataset by splitting the dataset, it is observed that the classification accuracy of the sequential execution in WEKA is higher that the classification accuracy of MapReduce-based BPNN except from the Insurance dataset.

By making feature selection, the dimensions of the dataset are reduced minimum 15% to maximum 65% from the original feature subset. It is found that model construction time after making feature selection to original datasets is significantly reduced for all of the models on all of six datasets. After making feature selection, the accuracy is significantly increased in MapReduce-based BPNN on selected optimal features. On the other hand, the accuracy is not decreased in WEKA and conventional

BPNN when the dimension of dataset is reduced. Thus, it can also conclude that feature selection can retain a suitably accuracy in representing the original features by selecting a minimal feature subset from a problem domain.

For BPNN classification model which has trained with two hidden layer, two more analysis is also made in training time and accuracy with MapReduce-based BPNN on complete features. In this experiment, increasing one more hidden layer in MapReduce-based BPNN suffers more training time to build classifier model. But the accuracy isn't significantly affected because only one dataset, ThyroidDisease diagnosis, is affected that tends to increase accuracy from 67% to 74%. Therefore, it can also be concluded that increasing hidden layer in MapReduce-based BPNN tends to increase training time although it's not significant in classification accuracy when compared to classification of BPNN model with single hidden layer used.

## 5.2 Limitations and Further Extensions

There are some limitations in the proposed system. In this system, feature selection is performed by Chi-square method only and feature set size of the dataset varies 29 to 351. Missing values are replaced by the arithmetic mean substitution. For MapReduce-based BPNN, only one and two hidden layers used. The system has experimented in Hadoop pseudo distributed mode with single NameNode and single DataNode.

In the future, another feature selection method such as Gain Ratio, Entropy and Correlation can also be used. It can also be compared with other feature selection method such as wrapper, hybrid and embedded method. It can also implement MapReduce-based algorithm on Fully Distributed mode cluster that can support to make experiments with hug datasets with high dimensional data. Other framework like Spark can also be used to compare the performance measure with Hadoop/MapReduce architecture.

# AUTHOR'S PUBLICATIONS

[1] Chit Thu Shine, Thi Thi Soe Nyunt, *"Feature Selection and MapReduce-based Neural Network Classification for Big Data"*, the Proceedings of the 9th Conference on Parallel and Soft Computing (PSC 2018), Yangon, Myanmar, 2018.

[2] Chit Thu Shine, Thi Thi Soe Nyunt, *"Scalable Neural Network Using Hadoop and MapReduce"*, Journal of Network Security and Data Mining, HBRP Publication, 2018, pp. 7-14.

# REFERENCES


[1] Amir Hossein Basirat, *"Distributed Associative Memory Approach for Cloud Computing Environments"*, ME Thesis, Monash University, 2016.

[2] Changlong Li, Xuehai Zhou, Kun Lu, Chao Wang, Dong Dai. *"Implementing of Artificial Neural Networks in MapReduce Optimization"*, [accessed November 25 2018].

[3] Grzegorz Mrukwa, *"Types of Machine Learning Algorithms: Supervised and Unsupervised Learning"*, [online] available: https://www.netguru.co/blog/ types-of-machine-learning-algorithms-supervised-and-unsupervised-learning, October 2018.

[4] Hong-Xing Li and C.L.P. Chen, *"The Equivalence Between Fuzzy Logic Systems and Feedforward Neural Networks"*, IEEE Transactions on Neural Networks, Volume: 11, Issue: 2 , Mar 2000.

[5] K.J. Hunt, D. Sbarbaro, R. Z_ bikowski, and PJ Gawthrop, *"Neural Networks for Control Systems - A Survey*", Automatica, Volume 28, Issue 6, November 1992.

[6] Martin Lněnička, Renáta Máchová, Jitka Komárková, and Ivana Čermáková, *"Components of Big Data Analytics for Strategic Management of Enterprise Architecture"*, 2nd International Conference on Strategic Management, May 2017.

[7] Miroslav Vozábal, *"Tools and Methods for Big Data Analysis"*, ME Thesis, University of West Bohemia, June 2016.

[8] Nang Saing Moon Khan, *"A Wrapper-Based Hybrid Approach to Feature Selection for Artificial Neural Network"*, Ph.D Thesis, University of Computer Studies, Yangon, May 2004.

[9] Navjit Singh, Anantdeep Kaur, M. Tech, *"Feature Selection for Artificial Neural Network Based Intrusion Detection System"*, International Journal for Technological Research in Engineering Volume 2, Issue 11, July 2015.

[10] Petsas Konstantinos, *"Study of Technologies/Research Systems for Big Scientific Data Analytics"*, ME Thesis, University of Piraeus, 2016.

[11] Prem R. Adhikari, *"What Kinds of Filter Methods are there in Feature Selection?"*, [online] available: https://www.quora.com/What-kinds-of-filter-methods-are-there-in-feature-selection, February 2016.

[12] Rong Gu, Furao Shen, and Yihua Huang, *"A Parallel Computing Platform for Training Large Scale Neural Networks",* IEEE International Conference on Big Data, October 2013, pp. 376–384.

[13] Rachana Sharma, Priyanka Sharma, Preeti Mishra and Emmanuel S. Pilli, *"Towards MapReduce Based Classification Approaches for Intrusion Detection*". 6th International Conference - Cloud System and Big Data Engineering (Confluence). January 2016.

[14] Raul Rojas, *"Neural Networks: A Systematic Introduction"*, Springer-Verlag, Berlin, New-York, 1996**.**

[15] Saurav Kaushik, *"Introduction to Feature Selection Methods with An Example (Or How to Select the Right Variables?"*, [online] available: https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/, December 2016.

[16] Simon S. Haykin, *"Neural Networks: A Comprehensive Foundation"*, Second Edition, Pearson Education, 1994, ISBN 81-7808-300-0.

[17] Than Than New, *"Hybrid Approaches to Improve Performance of Classification by Using Feature Selection"*, Ph.D Thesis, University of Computer Studies, Yangon, May 2004.

[18] Yang Liu, Youbo Liu, Junyong Liu, Maozhen Li, Tingjian Liu, Gareth Taylor, and Kunyu Zuo, *"A MapReduce Based High Performance Neural Network in Enabling Fast Stability Assessment of Power Systems",* Hindawi Mathematical Problems in Engineering Volume 2017, February 2017.

[19] Online Document, *"Big Data Overview"*, [online] available: https://intellipaat.com/tutorial/hadoop-tutorial/big-data-overview/, [accessed November 25 2018].

[20] Online Document, *"Introduction to BIG DATA: Types, Characteristics & Benefits"*, [online] available: https://www.guru99.com/what-is-big-data.html, [accessed November 25 2018].

# APPENDIX: HADOOP INSTALLATION

In this section, the installation of Hadoop on a pseudo distributed mode will be presented as a step by step procedure. As a start, virtual machine is created with memory 4GB, 4 number of core processor and hard disk size with 85GB. Chosen operation system for pseudo distributed mode Hadoop installation in this master thesis is Ubuntu 16.04. The following procedure will have to do.

1) This step is to install the openssh-server that securely allows to communicate the nodes of the Hadoop Cluster, without password prompt during workflows. After the installation finishes, it is essential to generate a ssh key for the user and copy the id_rsa.pub to the "authorized_keys" file, within the "~/.ssh" directory.
2) Java 8 programming language is required to install in Step 2, because Hadoop is a Java based framework.
3) Afterwards, the Hadoop 2.7.2.tar.gz is downloaded for the installation of this master thesis and extracted tar file into the folder, "/usr/local/Hadoop-2.7.2".
4) Continuing, some of Java and Hadoop variables are needed to export in the file "~/.bashrc". The following commands are added in the end of file:

   ```
   export JAVA_HOME=/usr/local/java/jdk1.8.0_91
   export PATH=$PATH:/usr/local/java/jdk.8.0_91/bin
   export HADOOP_HOME=/usr/local/hadoop-2.7.2
   export PATH=$PATH:$HADOOP_ HOME /bin
   export PATH=$PATH:$HADOOP_ HOME /sbin
   export HADOOP_MAPRED_HOME=$HADOOP_ HOME
   export HADOOP_COMMON_HOME=$HADOOP_ HOME
   export HADOOP_HDFS_HOME=$HADOOP_ HOME
   export YARN_HOME=$HADOOP_ HOME
   export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_ HOME /lib/native
   export HADOOP_OPTS="-Djava.library.path=$HADOOP_ HOME /lib/native"
   ```

   Also, JAVA_HOME variable is added to file "hadoop-env.sh" that is in the directory of "/usr/local/hadoop-2.7.2/etc/hadoop"

   ```
   export JAVA_HOME=/usr/local/java/jdk1.8.0_91
   ```

5) The next step is to configure Hadoop File System (HDFS). The HDFS directories for name node and slave node are created by the following commands in terminal:

   ```
   sudo mkdir -p /usr/local/hadoop-2.7.2/hdfs/namenode
   sudo mkdir -p /usr/localhadoop-2.7.2/hdfs/datanode
   ```

Also, the permission for the user to access that directories is given by the following command

```
sudo chmod 777 -R /usr/local/hadoop-2.7.2
```

And, the "core-site.xml" file that reside in the "/usr/local/hadoop-2.72/ etc/hadoop/" is needed to update. So, the following lines are added to the <configuration></configuration> tags like:

```
<property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:9000</value>
</property>
```

In addition, "hdfs-site.xml" file that is in the same directory is also added the lines to the `<configuration></configuration> tags`:

```
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>file:/usr/local/hadoop-2.7.2/hdfs/namenode</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:/usr/local/hadoop-2.7.2/hdfs/datanode</value>
</property>
<property>
  <name>dfs.blocksize</name>
  <value>4194304</value>
  <description>Block Size is specified as 4MB</description>
</property>
```

Above the directories for name node and data node, block size of data that is going to save data in the HDFS and the directory to access a list of files that connect with the name node can be seen.

6) This step is to configure Yet Another Resource Negotiator (YARN). The resource manager is configured by adding the following lines to the `<configuration></configuration> tags of` "yarn-site.xml" file that is in the directory of "/usr/local/hadoop-2.7.2/etc/hadoop".

```
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
```

```
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
 </property>
```

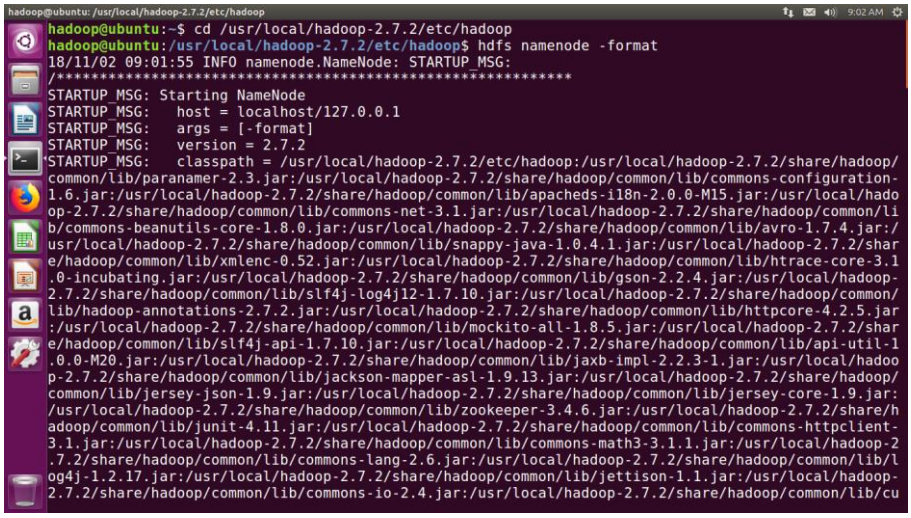
The `<configuration></configuration> tags of` "mapred-site.xml" file is also added the lines:

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

7) After above all steps are configured, the name node is needed to format by using the following command:

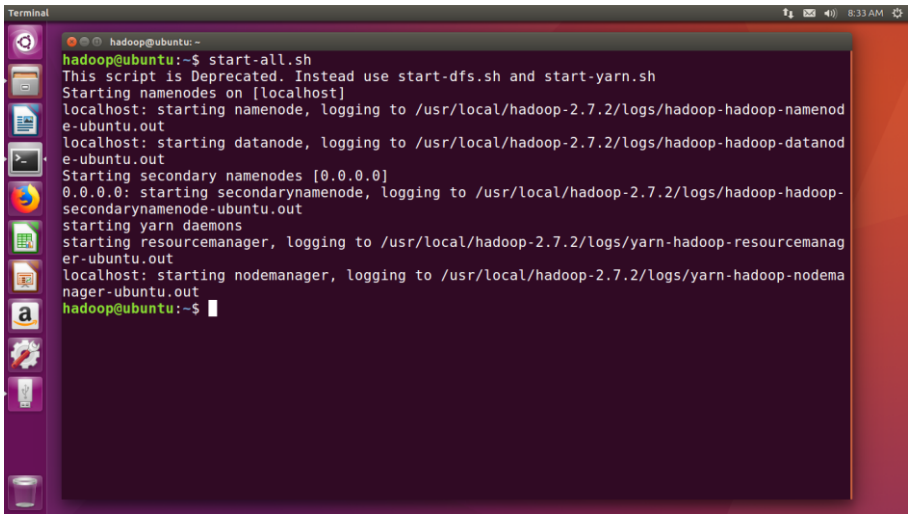
hdfs namenode -format

This command is used to start the name node in Hadoop.



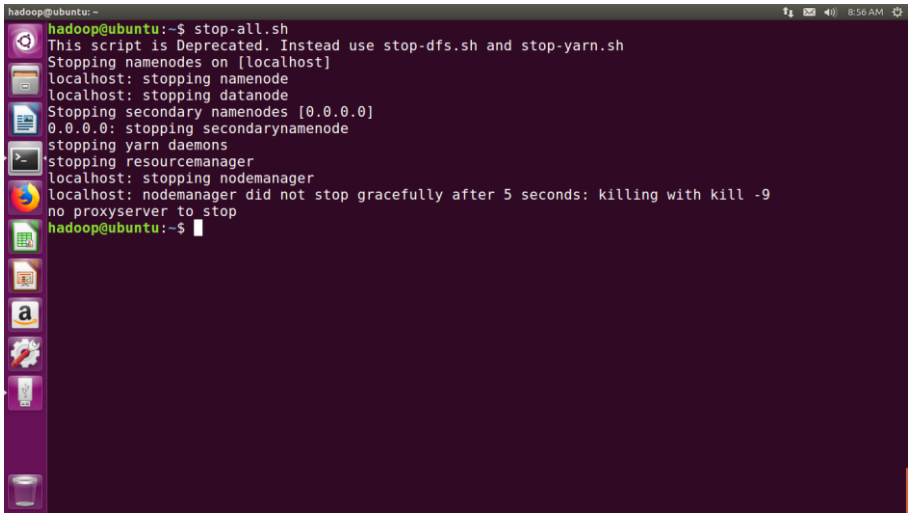

After formatting name node, the dfs and yarn daemons can be run. These daemons are started by typing the following command in the terminal of name node.

start-all.sh

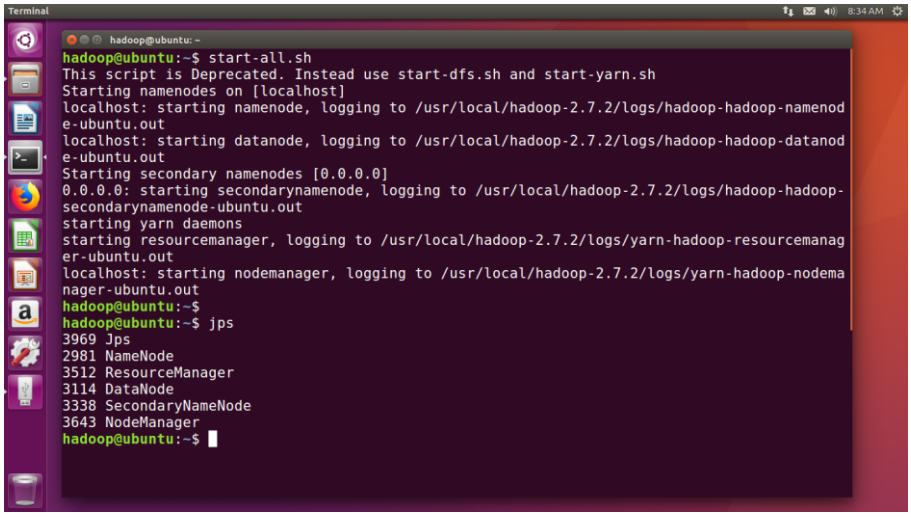


To stop all the daemons running in Hadoop, the following command can be used.

stop-all.sh

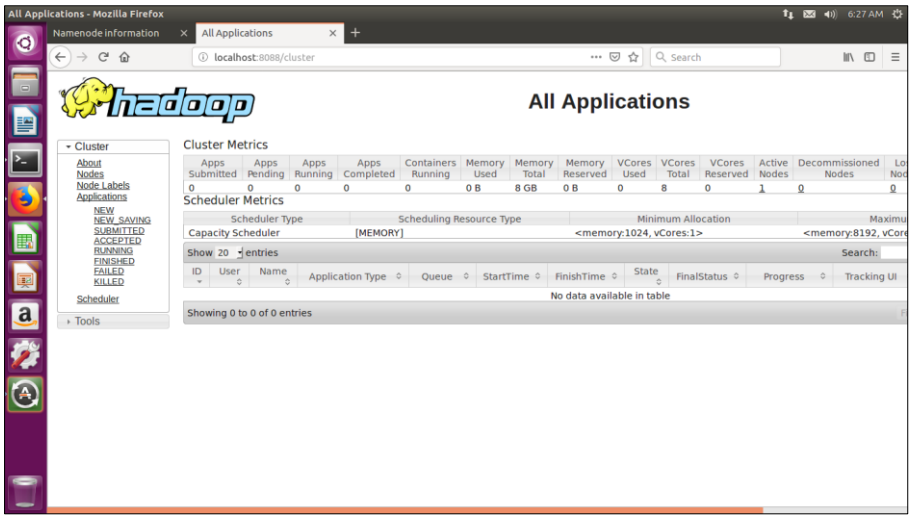



8) By using the commands “jps” in the terminal of name node, a list of the running services can be checked to identify whether they are running as expected or not.




After Hadoop installation steps is completed, it can be login to browser by using the links that will mention below.

http://localhost:8088




http://localhost:50070