COMPARATIVE STUDY OF RECURRENT NEURAL NETWORK AND SUPPORT VECTOR MACHINE FOR TEXT CLASSIFICATION

M.SC. PROJECT

BY

BAMGBOYE PELUMI OYELAKIN
19PGCD000070

SUPERVISOR
PROFESSOR ADEBIYI AYODELE

CO-SUPERVISOR
DR GBADAMOSI BATATUNDE

DEPERTMENT OF COMPUTER SCIENCE,
LANDMARK UNIVERITY, OMU ARAN.

MAY, 2021

COMPARATIVE STUDY OF RECURRENT NEURAL NETWORK AND SUPPORT VECTOR MACHINE FOR TEXT CLASSIFICATION

BAMGBOYE PELUMI OYELAKIN
(19PGCD000070)

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, COLLEGE OF PURE AND APPLIED SCIENCE, LANDMARK UNIVERSITY, OMU-ARAN, NIGERIA

IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF MASTERS OF SCIENCE (M.Sc) IN COMPUTER SCIENCE

MAY, 2021

DECLARATION

I, **BAMGBOYE PELUMI OYELAKIN** a Masters student in the Department of Computer Science, Landmark University, Omu-Aran, hereby declare that this dissertation entitled "**COMPARATIVE STUDY OF RECURRENT NEURAL NETWORK AND SUPPORT VECTOR MACHINE FOR TEXT CLASSIFICATION**", was written, implemented and submitted by me is based on my original work. Any material(s) obtained from other sources or work done by any other persons or institutions have been duly acknowledged.

_____

Student's Full Name and Matriculation Number

_____

Signature and Date

CERTIFICATION

This is to certify that this dissertation has been read and approved as meeting the requirements of the Department of Computer Science Landmark University, Omu-Aran, Nigeria, for the Award of Masters Degree

_____ _____

Professor Ayodele Adebiyi Date

(Supervisor)

_____ _____

Dr Babatunde Gbadamonsi Date

(Co-Supervisor)

_____ _____

Dr (Mrs) M.O Adebiyi Date

(Head of Department)

_____ _____

Professor Peter Idowu Date

(External Examiner)

DEDICATION

This work is dedicated to God who in his mercy made this thesis successful. Also, to my mother and my beloved siblings for their support in every way.

ACKNOWLEDGEMENT

To the Almighty God the giver of wisdom and life. I my sincere gratitude goes appreciate my supervisor, Professor Ayodele Adebiyi and my Co-supervisor, Dr. Babatunde Gbadamonsi for their mentorship and guidance. I want to use this medium to thank the H.O.D of computer science, Dr. (Mrs) Marion Adebiyi and the faculty and staff of Landmark University Omu-Aran. I will also like to thank Dr. Arowolo Olaolu, Dr. Asani Emmanuel and Mrs. Ogundokun Roseline for their support through the course of this project.

Finally, I want to thank the entire Bamgboye family for their support in every way.

ABSTRACT

Text is constantly generated from our day to day use of the internet, and these large amounts of data generated are mostly unfiltered. In most cases, unstructured data needs to be classified to improve the rate at which a given text is understood. Text classification is a branch of Natural Language Processing that is used to create a distinction in an unstructured text data.

As a result of this, Machine learning is widely used in the classification of textual data as a result of its ability to create complex prediction functions dynamically. Also, statistical models are commonly used to classify textual data because they are able to describe the relationship between two or more random variables.

Text classification consists of various branches which include sentiment analysis, document categorization, product labelling, and information retrieval. In an ecommerce environment, customers review is a way in which customer's opinion is given about a particular product and this can be negative, positive or neutral. However, there is always difficulty in classifying these opinions in ways in which they can be used to inform customer's decision about a product. Despite the advancement of machine learning models, certain limitations are prevalent in the traditional techniques like Decision Tree and Naïve Bayes.

In this study, a comparative study of Recurrent Neural Network (RNN) and Support Vector Machine (SVM) is done for a customer product review on whether they have positive or negative comments. The result of this work shows that RNN with an accuracy of 94.86% is better than the state of art SVM with an accuracy of 86.67%. The result of this work is not only better in terms if accuracy, but also other performance matrices.

TABLE OF CONTENTS

CHAPTER THREE: METHODOLOGY

CHAPTER FOUR: RESULT AND DISCUSSION

CHAPTER FIVE: SUMMARY, CONCLUSION AND RECOMMENDATION

LIST OF FIGURES

LIST OF TABLE

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

Text classification is a branch of natural language processing (NLP) which helps in filtering out irrelevant documents from a large dataset and because most textual data are unstructured, we constantly need to process and classify them in order to make sense of it and to have the precise understanding of the given information(Abiodun *et al*., 2019).

The process of information sharing is extremely difficult due to the ambiguity of natural languages. As a result of the expanded abundance of textual material and the rise in the need to provide versatile access to it, the classification of text becomes a critical task. Machine learning approaches and mathematical theory have recently been extended to text categorization(Liu *et al*., 2020).

The role of text categorization is to assign documents to already existing subjects, such as finance, sport and politics(Abdelaal *et al*., 2018). Text and sentence classification techniques are also a very important components in the study of NLP and they have been modelled for numerous area of application which includes analysis of sentiment, classification of document, Product labelling, information retrieval, hierarchical classification(Wehrmann *et al*., 2019).

In the early 1980's, researchers relentlessly looked into the field of text classification. As a result of the relative increase in the amount of data across the internet such as, electronic mails, news outlet, website, including countless properties available for scientific research purposes, with relative request for classification. As artificial

intelligence evolves, Frank Rosenblatt developed the first computer neural network called perceptron to mimic human brain and thinking processes. In order to help computers use natural languages, understand, and translate, Natural language processing (NLP) was introduced(Seising, 2018).

The usage of predictive models for NLP analyses grew significantly in the 1990s. Pure statistics NLP methods have become stable and efficient in maintaining the enormous flow of digital manuscript(Li *et al*., 2020).

Later in 1990s work on artificial intelligence swings from knowledge oriented approach to a data-driven method. Scientists have started to set up computer systems to analyse vast volumes of data and derive conclusions or benefit from the findings. At this time, machine learning method known as support vector machine (SVM) was applied to text categorization (Darling-Hammond *et al*., 2020). However, Artificial Neural Network (ANN) proved to generate a better result in this field over the years than models like SVM because artificial neural networks may have more than one output, while support vector machines have only one. ANN is part of many machine learning models that help to solve classification problems. as a result of its ability to dynamically construct complex prediction functions and mimic human reasoning in a way that no other method can do(Abiodun *et al*., 2019; Shanmuganathan, 2016).

Traditionally, text is understood as a piece of documented or spoken content in its natural form. Also, text can also be described as any kind of language that can be understood by a reader. It could be as simple as one or two words or as complex as a novel, which means, sequence of sentences that pertain together may be considered as a text. While classification, on the other hand, is a challenge of defining the groups in which new discovery is associated with, based on training collection of data comprising certain observation of which membership of the group is identified.

Examples are the assignment of a given email to the "spam" or "non-spam" class and the assignment of a diagnosis to a patient based on the patient's reported characteristics (blood pressure, sex, appearance or absence of such symptoms, etc.). Classification is an example of identification of trends. Text classification is a semi-supervised system designed for learning tasks which assigns documents in an automatic way to a collection of categories which are defined on its text data including features that were extracted (Wehrmann *et al*, 2019). Text classification is an automatic process which places significant focus on content creation, spam filtering, product review analysis, sentiment analysis, product marking and so on.

In this study, Recurrent Neural Network and Support Vector Machine Is implemented, for text classification on a customer review dataset in order to know if a product is good or bad.

1.2 Justification of the Study

The need to constantly classify text is fast becoming imperative as the number of textual data increases (Uçar *et al*, 2020). However, many techniques have been used in literature but the need to compare the existing technique in order to know which of the existing techniques performs better is important. Therefore, in this study, I compared recurrent neural network and support vector machine for text classification.

1.3 Statement of the Problem

With the proliferation of data, the classification of customer's opinion is a major issue in ecommerce. As a result of these, vendors frequently require customers to input their opinion of a product manually, this makes it impossible for other customers to shop for the same product based on other customer experiences. For text-based

product classification, the two machine learning models applied to ecommerce product categorization are Support Vector Machine and Recurrent Neural Network. RNN and SVM have proven to achieve great result in NLP which put them ahead of other machine learning techniques such as Gated Recurrent Unit(GRU), and Convolutional Neural Network (CNN), Logistic Regression, Decision tree and so on (Habimana *et al*., 2020). However, artificial intelligence researchers have been applying machine learning to classification of goods (Cioffi *et al*., 2020).

Therefore, as a result of the aforementioned problem and to proffer a suitable solution, customer review is big part of any sales shop whether they are online store or a local supermarket. Text classification technique helps customers to know if products have a positive or negative review, which enables sales companies to improve the customer experience and also improve the revenue of the company. In this study, the possibility of analysing product review using machine learning techniques to improve user experience was considered.

In this work, Recurrent Neural Network and Support Vector machine for customer opinion in ecommerce environment is implemented to know which one performs better in term of Accuracy, Precision, Specificity, Sensitivity, F-Score, Recall, False Positive Rate, False Negative Rate and Matthew Correlated Coefficient.

1.4 Aim and Objectives

The aim of this study is to undertake a comparative study of recurrent neural network and support vector machine for text classification in an ecommerce environment.

The specific objectives of the study are to:

1. Identify important features for text classification

2. Formulate the model for text classification using recurrent neural network and support vector machine algorithm.
3. Implement the model formulated in (2) above.
4. Evaluate the performance of the models.

1.5 Scope of the Study

The scope of this study focuses on Support Vector Machine and Recurrent Neural Network for text classification in an ecommerce environment. The implementation is done in a MATLAB 2015 environment using an existing dataset

1.6 Significance of the Study

This study seeks to improve customers experience in a way that they can shop for the right product based on other peoples experience with the product in the past. The study will also be useful for other researchers who need to explore the intricacy of Recurrent Neural Network and Support Vector Machine for text classification tasks.

1.7 Organization of the Work

The second chapter gives a review of related work while the third Chapter discusses methodology, analysis of the system and data used.

Chapter four covers the system implementation and experiments on data sets for machine training and classifications as well as comparative analysis of the experimental outcome.

Chapter five is about the summary of the work, limitations of the study, recommendations, conclusion, bibliography, and appendix of the thesis.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This chapter discusses various machine learning techniques for text classification, the review of related work and gaps identified in existing literature. However, in order to fully understand the concept of machine learning approach for text classification, they are categorized into three major areas as described in figure 2.1 which includes, unsupervised text classification, semi-supervised text classification and supervised text classification on the basis of learning theory applied to the data technique (Korde, 2012).

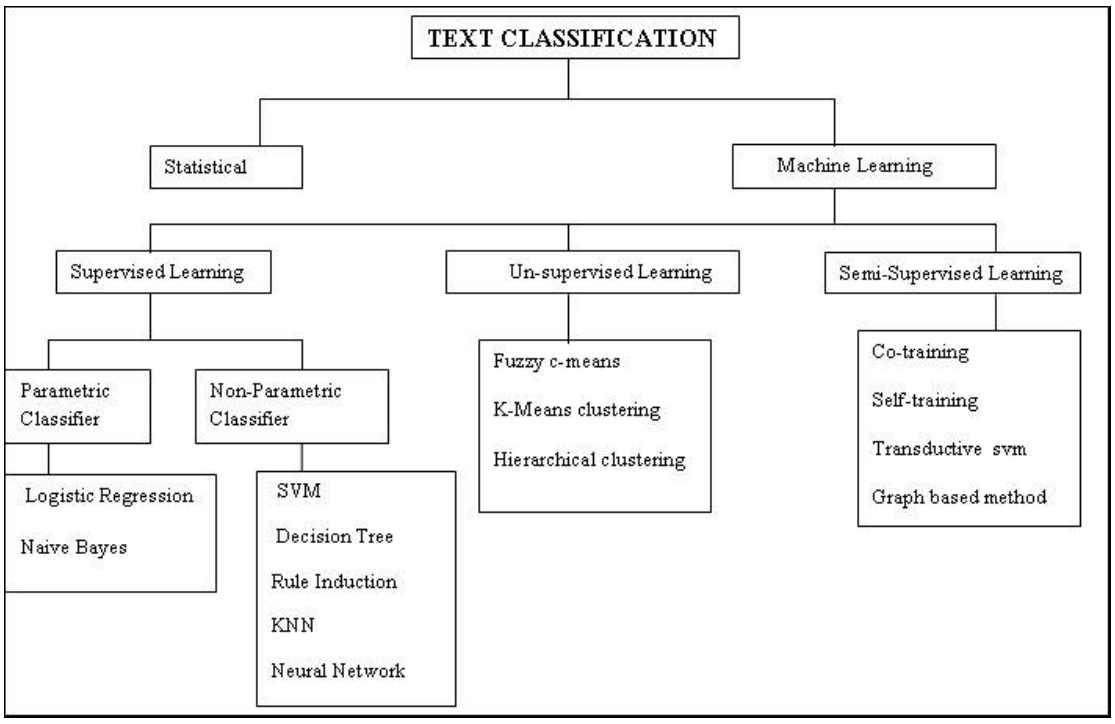

Figure 2.1: Representation of Various Classifiers(An *et al*., 2017)

2.2 Machine Learning Approach

In machine learning context, the categorization tasks is based on supervised learning theory, in cases that the algorithm is evaluated on class awareness prior to the real categorization step begin, while the unsupervised learning happens in cases where the data that is labeled are not visible(Brownlee, 2020). This method is dynamic and has consistency problems. It's fantastic in the classification of large amount of data. Also, Semi-supervised learning is done when data is unlabeled and partly labeled (Uçar *et al*, 2020).

It is therefore difficult to create a standard connection among the unlabeled and labeled sample. Performance is measured using measures such as precision, recall and accuracy. In case of large dataset, the error of categorization is smaller (Jin Huang *et*, 2005). It is taken into consideration that the collection of appropriate model for a given set of are important to categorization of text(Ramola *et al*, 2019).

2.2.1 Supervised learning

For a supervised learning model, it is often time the costly and complicated of all the machine learning methods, because it require human interaction when applying categories to groups that are not feasible in a huge amount of data. (M. Chen *et al*, 2011).

As different data are distributed, supervised learning becomes costly as a result of different outcome and diverse functional environment exist in case of mixed text. The commonly used supervised methods is the calculation of the highest probability, by doing this, the learning process may be generalized by different hypotheses. This types of data assumptions incorporate two models, such as non-parametric and parametric approaches(Dada *et al*., 2019; Hartmann *et al*., 2019)

2.2.1.1 Parametric models

A model that can quantify the data on the basis of the underlying properties is known parametric model (Soria *et al*, 2011). The Novel Logic regression and Naïve algorithms are parametric classifiers. While, k-nearest neighbor, Support Vector Machine, decision trees, Inference rule, and neural networks are non-parametric classifiers (Tsangaratos *et al*, 2016).

a. Naive Bayes Classifier

Naive Bayesian are probabilistic classification technique that are widely acceptable in Machine Learning (Thangaraj *et al*, 2018. However, the bayesian classifiers are used in statistics and also used learning process. Naïve Bayes uses a multinomial approach for large datasets, which means that the output may be improved by looking for the dependencies between the characteristics. It is primarily used in preprocessing of data as a result the simplicity of computing. (Zhang, 2005).

In a naïve bayes classifer the deep feature weighting measures the probabilities of the classifier by calculating the frequency of the weight of the training results. It addresses the issue of conditional independence presumption, which is a significant change in the Naïve Bayesian classification and correctly calculates the conditional probability(Hassan *et al*., 2007). While these weighting strategies feature certain flaws, such as low performance enhancement, weakened simplicity and increased model execution time, they act to reduce the expense of the computation related to the modeled data. In addition, the Naïve Bayes method may denote random characteristic dependences(Shang *et al,* 2016).

The utility of Naïve Bayes classifier is dependent on the accuracy of the probable restrictive probabilities. These terms cannot be correctly approximated because the training data is sparse. Any of the metaheuristic methods, such as Genetic Algorithms,

Differential Evolution and so on was then used to approximate the conditional probabilities. in certain cases, the advantages of Naïve Bayes are always affected by conditional assumption of freedom among attributes which determines the progress of categorization.(Diab *and* El Hindi, 2018; El Hindi, 2014).

b. Logistic regression

Logistic regression in supervised learning is when the choice of the right data to be classified in order to obtain a successful classification is based on the amount of time the classification is performed. The supervised learning method is then used to identify the most accurate classified data in a Machine Learning Models.(Schmidt *et al*, 2019).

Logistic regression are examples of linear classifiers which are ideal for broad and computationally intensive datasets, which makes them ideal for a huge amount of data (Ifrim *et al*, 2008; Kowsari *et al*., 2019).

2.2.1.2 The Non parametric models

This are model that are unable to coordinate data on the basis of the fundamental attributes that are considered in a classification process. Example of such models are Support vector machines, Recurrent Neural Networks, Decision Trees, and K Nearest Neighbor are mostly non-parametric classifiers(Burba *et al*., 2009; Thanh Noi, Kappas, 2017).

a. Support vector machine

Support Vector Machine (SVM) classifiers are supervised learning models that aim to find the optimum hyper plane separating two separate groups of data, as described in

Figure 2.2, it is also known as a supervised machine learning model that uses categorization techniques for multiple groups of classification issues. In SVM, once a data has been defined in a category for training, they are able to produce a new collection of information into different category. For instance, by giving a SVM model a certain set of input text and this model is trained, it is guaranteed that a new set of text will be generated into a different category as an output this is typically how this model work, and it is commonly utilized in the text classification because it allows us to categorize new set of data from a given input. It is also known to be fast and dependable in text classification for small dataset

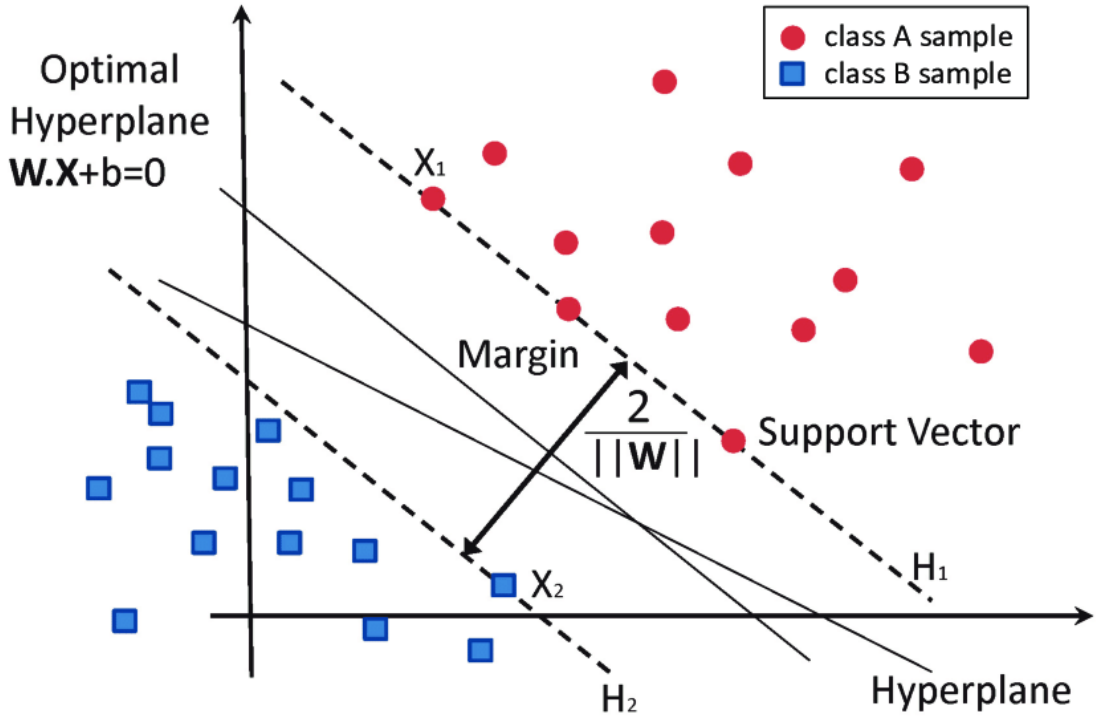

Figure 2.2; A typical SVM (García-Gonzalo *et al*., 2016)

This figure 2.2 shows the separation of margins using hyper plane H_1, and H_2 where the X_1 and X_2 shows the instance of the two classes they are trying to classify. Also, the W is the weight matrix applied to the hyper planes.

b. Decision trees (DT)

The Decision Trees (DT) are extremely logical structures in comparison with Neural Networks. They function in a series of test to predict values within a range of an available set of values. In DT training takes place in relation to some rational laws, analogous to the idea of neural network weights(Bologna *et al*, 2018; Kotsiantis, 2013).

The optimisation of decision-making trees is another area that needs to be extensively researched. In a recent publication, a genetic algorithm are coupled with a multi-labeled objective function that produces efficient trees with the best outputs.(Huang *and* Wang, 2006; Mehboob *et al*., 2016).

Although decision trees work well for data with minimally significant attributes, computational complexity increases complexity when the relationship between the attributes increases. Given all the strengths of the decision tree, the regular end-user could still not understand the background details that led to a clear decision on classification problems. (Peloia *et al*., 2019).

c. K-Nearest Neighbor

K-Nearest Neighbor (k-NN) are based on theories of the nearest training sets, the points of data that are related to one another as part a similar class, usually referred to as instance-based learning. Although it is robust for results that are not preprocessed, determining the value of k is difficult. The complexity of the computation is increasing within a high dimensional space. (Segata *et al*., 2010).

Through modifying the stimulation bias of k-Nearest Neighbor, the class imbalance issue of a set of data is solvable by uncommon class modeling, which is a major benefit of k-Nearest Neighbor, particularly when it is a classification task(Adewale *et al,* 2019*).* Any other existing research approaches on this subject include re-sampling,

cost-sensitive learning, and algorithm-specific learning methods. Extension of this work can be carried out in a variety of uncommon class situations and class instances on the grounds of the posterior probability of each class. K-Nearest Neighbor are also the most frequent classification of events, based on the context of the data points by majority decision.(Branco *et al*., 2016). This approach is very good for small samples.

d. Artificial neural networks

Recently, artificial intelligence has improved the chances for physicians with little to no mathematical knowledge to use the advantages of artificial intelligence-based diagnostic methods to boost service by offering strategies that reveal complicated correlations that cannot be reduced to an equation. Artificial Intelligence (AI) methods include rationale, consisting of inferences from facts and rules using heuristics, pattern matching or other search approaches, and has greatly contributed to the development of biology and medicine (Anwaitu Fraser *et al*., 2020).

Artificial Intelligence (AI) is a subset of computer science that covers a wide range of computing activities, ranging from classical algorithmic development to Machine Learning and deep learning algorithms (Di Franco *and* Santurro, 2020). These networks use simple numerical procedures, such as integer arithmetic, but are able to solve complex, non-linear problems. When a network has been sufficiently trained, it may be used to forecast a variance based on an independent (holdout) dataset, usually with minimal changes. The main components of the ANNs are neurons, which are organized into layers and fully connected to the next layer by a set of weights (edges). Each ANN consists of a single input layer, a single output layer, and at least one hidden layer. The most basic form of ANN is called the perceptron, first suggested by Rosenblatt, the building block of neural networks. In a perceptron, each input is

multiplied by a weight matrix, and aggregated by a mathematical function known as an activation function. (Mollalo *et al*., 2020).

e. Recurrent Neural Networks (RNN)

RNN are one of the many variations of ANN's. RNN's which are common amongst tasks such as speech recognition, translation but also time series forecasting (C. Olah; 2019). The decisions made in an RNN are based on both the current input and previous predictions; with this consideration it is clear how the application of RNN to sequential data is relevant. In deep neural networks Jürgen Schmidhuber introduces Credit Assignment Paths (CAP) (2014), which is a system for analyzing and classifying the depth of a neural network related to the depth of a problem. Hence, due to the reuse of previous data, RNN can potentially solve problems of unlimited depth such as text classification.

RNN's are well suited for handling cases where we have a sequence of input, rather than a single input, and are great for a problem where a sequence of data is propagated through the model to give one output. For example we can imagine training a model, which takes a input as sequence of word and outputs a sentiment that is associated with that phrase or sentence, alternatively we can train a model which takes in a sequence of input, propagate them through the recurrent neural network and return an output at each time step in the sequence . RNN have loops in them which allows information's to persist, and we call it recurrent because information is been passed in each cell from one time step to the next. We can also see RNN as unrolling each loops across time and if we do this we can see RNN as multiple copies of the same network where each copy is passing the same message unto its descendant. RNN generally have the chainlike structure which highlight why they are the best for tasks such as text classification.

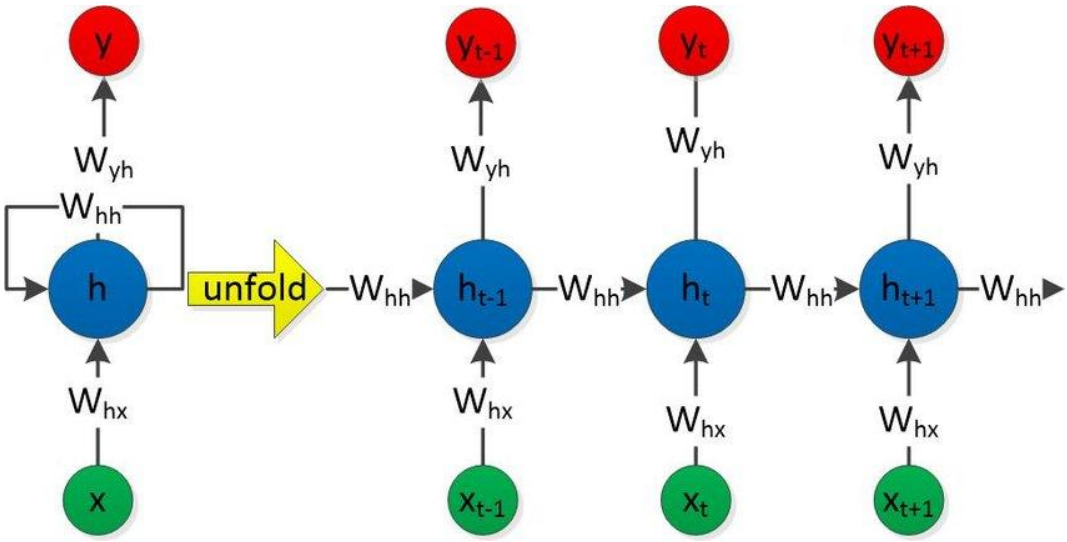

Figure 2.3 : A typical RNN Architecture (Seising, 2018)

Where h_t= *new state*

W_{hx}= *function parameter* by weight metrix

h_{t-1}= *old state*$(previous\ hidden\ state)$

x_t= *input vector at each timestep*

2.2.2 Unsupervised learning

Unsupervised learning is a type of Machine Learning models where assumptions are made from data by clustering data into separate clusters without a labeled answer which means (expected outcomes), the machine is not supplied with training data initially, it seems to be complicated, but as more data is fed into the model, the algorithm refines itself to performance. The Key component analysis, clustering and self-organizing maps are also used in unsupervised research. Clustering is the same in

many scenarios as unsupervised learning. Many times, the expert expertise needed to mark the materials is either non-existent or incomplete. In such a case, self-organizing maps and correlation coefficients are used to cluster documents and use them to mark documents for further processing (Shafiabady *et al*., 2016).

2.2.3 Semi-supervised learning

The semi-supervised learning is a blend of supervised and non-supervised learning methods (Aydav *et al*, 2020). This method of learning uses a limited amount of classified data and a significant number of unlabeled data for training. The labels are allocated by mixing unlabeled and labeled cases, since unlabeled data mitigates the impact of inadequacies in data that are labeled on the precision of the classifier. Some of the Semi Supervised Learning approaches include bootstrapping, self-training or co-training, transducing graph-based processes, Support Vector Machine and generative models (Mitchell, 2004; Pise *et al*, 2008).

Supervised learning approaches are successful where appropriate instances are defined. However, in many applications, such as object recognition, text and web page categorization, categorized instances are challenging, expensive or time-consuming to accomplish because they require empiric analysis or experienced human annotators. Semi-supervised machine learning models not only use named data and therefore unmarked data to produce a classifier. The goal of semi-supervised learning is to use unmarked instances and to combine knowledge in unmarked data with explicit details on the categorization of labeled data in order to increase the quality of the categorization.(Tanha *et al*., 2017).

Traditional text labeling approaches become null because there are no labeled data on a particular form of dataset, for example, data set are only available for optimistic

samples and not for negative cases. Semi-supervised algorithm based on maximal tolerance and group learning is proposed for the same purpose.(Chen *et al,* 2019).

The inaccessible class is removed roughly from the dataset and set to the labelled sample. The ensemble classifier iteratively constructs the margin between negative and positive classes to better estimate negative data, since negative data is combined with positive data. Consequently, classification is done by a composite methodology without the need for training samples. It removes the expense of hand labeling data, especially in the case of huge amount data.

The use of semi-supervised approaches is very useful for the extraction of information criteria. The role of semi-supervised algorithms in the hierarchical multi-label categorization is an area in which further research is still required. Self-training with a semi-supervised classifier for the hierarchical categorization of multi-labels is suggested. It has also proven to be a better way to accomplish automatic brand allocation (Benato et al., 2018).

2.3 Review of Related Works

Mayor *et al,* (2012) worked on document classification using support vector machine discussed in depth the implementation of the Support Vector Machine for the measurement of the word frequency of features used as Sports, Industry and Entertainment for categorization with the aid of a manual domain dictionary. They find out that it is comparatively less cumbersome to identify divisions that broadly characterize the knowledge stored in these sets. A package called LIBSVM was used in past literature to incorporate the SVM in their work.

The purpose of using LIBSVM is to classify positive data so that the classifier can correctly predict uncertain data, in other words test data. They proposed five major

steps that can be used for document classification which includes the following; the first step which is to collect the instances [i.e. News clipping] from the Newspapers or News sites. The second step which is the creation of domain dictionary in this step all the features from the instances are extracted and stored in file. When in the third stage classification of the classifier function, this move is to identify such instances obtained in subclasses as Athletics, Entertainment and Industry by means of a classifier function. In general, the classifier takes a single instance and then pairs it with features in the domain dictionary contain lots of features synonyms.

This mapping is performed to produce the threshold frequency for each feature and automatically create a text file for each feature. The fourth stage is the processing of the LIBSVM tool, which produces text files, which is then stored in the LIBSVM tool, which provides the accuracy score for the research classification, which is further qualified and projected to be evaluated. The last step is to evaluate the results obtained from the contour graph and to make conclusions on the efficiency of the categorization. The result of the accuracy of the LIBSVM is at 66.6% for sport and business while it achieves 100% accuracy in the business dataset that was used. They determined the average accuracy of the three classes which is 66.667%, and finally concluded the work by proving that they developed very efficient and reliable classification technique

Sharkah *et al*, (2015) worked on Text Classification Using Support Vector Machine, they used the concept of data mining incrementally to perform classification and supervised learning, by using multiple files for input, where the first and the second set of the data contains both training respectively for the classifier, they identified the vital part of modelling a classifier which is to decrease the dimensionality of attributes.

To achieve this, they performed a two key processing step on each data; by doing this they converted all the data into lowercase since it does not aid the process of classification. They have eliminated all the stop word from the training data set because they have no meaning that can boost the text categorization process. According to the authors, the keys to the hash table appearing in the process of preparation, the data whose actual value is the information which will be used to define the text as long as the hash table contains the correct data.

They categorized the dataset into two categories, where the first category is the counts of the amount of times a word appear in the instance of the first category, this process is repeated for the second category. However, once the hash table is loaded, it implies that the training process is completed. Therefore when the classification process begin they will make use of the hash table containing the data.

Chunting *et al.,* (2015) In their work on the Convolutional –Long Short Term Memory Neural Network for classification of text, they incorporate the strengths of both the Convo Net and Recurrent Neural Network architectures and introduce a new and coherent model called Convolutional-Long Short Term Memory for sentence description and text classification.

Convolutional Long Short Term Memory uses Convolutional Neural Network to derive a sequence of higher-level phrase representations and is fed into a long-term memory repetitive neural network to generate a phrase demonstration. Convolutional –Long Short Term Memory is capable of collecting both local phrase characteristics as well as national and temporal phrase semantics.

The RNN used in their work will manage sequences of any length and catch long-term dependencies. In order to prevent issues with the gradient bursting or vanishing gradient in the regular RNN, the Long Short-Term Memory RNN (LSTM) and other

versions were designed for improved memory. In addition to sequence-based or tree-based models, RNNs have obtained impressive results in sentence or text modelling.

The Stanford Sentiment Treebank benchmark used this data collection, which consists of 11855 film ratings and is divided into train 8544, development 1101 and test 2210. Sentences in this corpus are parsed and all phrases along with the sentences are fully annotated with 5 labels. The result of comparing the C-LSTM and other novel approach such as LSTM and bi-LSTM is that the C-LSTM with 94.6% result which outperforms the LSTM and bi-LSTM with 93.2% and 93.0% respectively. Although the C-LSTM is not the best approach for text classification, the result shows that the C-LSTM result is close to the popular SVM at 95.0%.

Shafiabady *et al.*, (2016) Worked on using an unsupervised clustering approach to training the Text Classification Support Vector Machine by defining a process that uses Self Organizing Maps (SOM) and, respectively, by using CorrCoef (CorrCoef) automated clustering.

As a consequence, clusters are used as labels to train the Support Vector Machine (SVM). However in their work, the basic concept behind the nature of the machine-labelled data learning scheme introduced in their work uses SOM and CorrCoef for the arrangement and marking of unlabelled data samples. They suggested that the SOM suffer from COD because it uses the Euclidean distance function for dimensionality reduction. Thus the elimination in dimensionality as accomplished by the use of the Concept Component Analysis (PCA) has tended to minimize the dimension of the original information to its major features.

The SOM is also used exclusively for the acquisition of labels for each of the data samples. They also introduced the integration of SOM and SVM, which leads to an unsupervised learning machine with a high generalization potential obtained from the

SVM properties. The results of their analysis were obtained by updating the map and the data threshold to reach the same number of clusters as the original class number. The results indicate that 20 newsgroups showed a categorization accuracy of more than 95%. The accuracy of the dataset validation collection is approximately 92 percent, which is higher than that suggested by AL-SVM Mohamed *et al* in their text classification study.

Pengfei *et al*, (2016) In their work on RNN for text categorization with multi-task learning used the multiple task learning, proposed three separate models of knowledge exchange with recurrent neural networks (RNN), relevant activities are merged into a single machine that is jointly qualified. The first model uses only one shared layer for all functions. The second model uses various layers for different activities, but each layer can read data from diverse layers.

The third model not only assigns a particular layer to each task, but also creates a common layer for all tasks. Aside from these, a gating function was added to allow the model to use shared knowledge selectively. Both of these activities were collectively learned by the whole network. They used innovative methods to incorporate RNN into a multi-learning system, which learns to translate arbitrary text into semantic vector representations that are both task-specific and shared layers. They have been able to show good effects on many text recognition activities.

The multi-task models used outperform much of the state-of-the-art baselines. The network parameters used in their work have been learned to decrease the cross-entropy of the expected and true populations. They noted that the deep neural model is well adapted for multitasking learning, as the features gained from the task can be useful for other tasks. The network has been prepared for back spreading and gradient-based optimization is done using the Ada grad upgrade rule.

In order to illustrate the efficacy of multi-task instruction, four separate text classification tasks for film are selected. Experimental findings suggest that models can boost the efficiency of a set of similar tasks by investigating shared characteristics.

Su *et al*, (2017) worked on accurate identification of words in scenes lacking character segmentation using a recurring neural network. They proposed a new Scene Text Recognition system that performs word level recognition without differentiation. Second, their model turns any word representation into a sequential signal for the text recognition of the scene. Also, it adapts the recurrent neural network (RNN) to the Long Short Term Memory, a tool widely used for handwriting recognition in recent years. Third, by integrating multiple RNNs, an appropriate method of identification is created that is capable of recognizing the text of the scene, including those strongly impacted, considering the segmentation of the character. The suggested methodology has been tested on four datasets, including three Rigorous Reading Competitions and a Google Street View Text Dataset composed mainly of external signboards and picture titles.

Wang, (2017) In their work on A Long Short Term Memory Approach to Short Text Opinion categorization with Word Embedding's Examine deeper word semantics using deep learning approaches that analyse the impact of word embedding and long-term memory (LSTM) on social media sentiment classification. Next, terms in posts are translated to vectors using word embedding templates.

The word series in sentences is then entered into the LSTM to acquire long distance semantic dependence between words. Experimental studies have shown that deep learning models can efficiently learn the use of terms in social media based on ample training samples. The dataset that were used in order to evaluate the performance of

the proposed approach include the English movie review from IMDB, a Chinese movie review comment from Douban and a Chinese post. The IMDB dataset used contain 50,000 review comment with 25,000 for training and 25,000 for testing respectively, More than 20,000 were used as testing as and more than 22,000 as test results.

In order to prevent influence from prior feedback on the same film, no more than 29 reviews were received for each film. The rating of each film was used as the ground reality, with a rating above seven as positive and a rating less than four as negative. In the second dataset, the top 200 films for each of the ten groups of Douban Movies were selected. The top 40 to 60 review comments were collected from each movie based on their success.

The simple reality of this data were following order: a rating of 4-5 as positive and a rating of 1 and 2 as negative. Other datasets that have been compiled and used from the popular social media site in Taiwan, and user reviews of likes and dislikes, are used as the simple truth of the dataset. Finally, they concluded that deep learning approaches such as LSTM demonstrate improved performance of sentiment categorization when more training data are usable.

Yin *et al.,* (2017) In their work on Comparative Study of Gated Recurrent Unit (GRU), Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN), for Natural Language Processing(NLP) A comprehensive comparison of GRU, CNN and RNN on a wide variety of NLP representation activities, aimed at providing specific instructions for the collection of DNNs, was made.

They noticed that the NLP gained greatly from DNN as a result of its high performance, with little to no need for engineered functionality. The work systematically compares CNNs, GRUs and LSTMs to a wide variety of NLP tasks:

sentiment/relationship grouping, textual implication, response collection, question-relationship matching in Freebase, which is used for route query and part of speech tagging.

Two main results are confirmed by the studies of their research. Including one which RNN and CNN have additional knowledge on text classification tasks to see which architecture fits well based on how necessary it is to grasp the sequence semantically. Also the learning rate adjusts output relatively smoothly, while changes in secret size and batch size result in significant variations. The Stanford Sentiment Treebank (SST) dataset that was used predicts the sentiment (positive or negative) of movie reviews. They were able to use the given split of 6920 train, 872 dev. and 1821 test sentences. This dataset was classified into four major categories which include the text classification, sentiment classification, and sequential order and context dependencies respectively and by doing this, they were able to discover some basic principles involved in utilizing CNNs/ RNNs. The result in their work shows that for all tasks and models and corresponding hyper parameters for Text classification the Gated Recurrent Unit (GRU) performs best with a performance of 86.32% and the CNN and LSTM with 82.2% and 84.51% respectively.

Wang, (2018) presented a Novel Approach Named Disconnected Recurrent Neural Network (DRNN) in their work on Disconnected Recurrent Neural Network for Text classification. The DRNN which is proposed to be a framework of RNN Requires position-invariance in the RNN. By restricting the width of the information flow in the RNN, the secret state at each time stage is limited to reflecting terms near the current location. The suggested model makes major changes over the RNN and CNN models and delivers the highest results on a variety of benchmark text classification data.

In order to reduce the burden of modelling the entire sentence, they limit the distance of information flow in RNN. Similar to other RNN variants, they were able to feed the input sequence into an RNN model and generate an output vector at each step. One essential distinction from Recurrent Neural Network is that the status of the proposed model at each stage is related only to the preceding k-1 terms, but not to all the preceding words. For Text classification they utilized GRU as recurrent unit of DRNN and get the context representation of each step where every context can be considered as a representation of a text fragment.

Then they fed the context vector into multi-layer perceptron (MLP) to extract high-level features. Before feeding the vector into MLP, they utilize batch normalization after the DRNN, so that the model can alleviate the internal covariant shift problem. Finally, they feed the text representation vector into a MLP with rectified linear unit (ReLu) activation and send the output of MLP to the softmax function to predict the probability of each category. Although the proposed DRNN does not does not compare to the novel RNN and CNN in terms of performance, the DRNN have an aggregate of 5.53, while the CNN and the RNN have 6.30 and 6.20 respectively. In this paper, they incorporated position invariant into RNN, so that the proposed model DRNN can both capture the key phrase and the long term short term dependencies.

Goudjil *et al.,* (2018) In their effort to create an Active Learning Model for Text Classification using SVM, aimed at reducing the number of samples labeled without compromising the efficiency of the classification by means of the Active Learning Support Vector Machine (AL-SVM), which leads to a large decrease in the expense of the labelling of samples and to the improvement of the training phase for classifiers. In their work, they use a series of SVM classifiers to pick the most insightful sample

for each active learning iteration, which increases the trust of the multi-class classification.

The data pool in the model they suggested is split into packets of similar size and executed concurrently. The sample chosen is then labelled and added to the training package. The dataset that was retrieved from the publicly accessible registry for single label text categorization was used to verify the text categorization basis. However, they stressed the main purpose of active learning, which is to reduce the effort to mark, without compromising the accuracy of classification, by intelligently selecting which samples should be labelled.

The suggested technique selects a batch of useful samples using the probability distributions produced by a set of multi-class SVM classifiers, which are then manually labelled by the expert. In the AL-SVM result, the collection of data was separated to include a training data set of 20 samples per class. The pool was divided into 200-sample sets. AL-SVM has been executed several times with different initial training sets and the same standard with 70 per cent accuracy.

Jang *et al.*, (2020)worked on Bi-LSTM Model to Increase Accuracy in Text Classification and Combining Word2vec CNN and Attention Mechanism, an attention-based Bi-LSTM and Convolutional Neural Network hybrid model was suggested that would draw on the advantages of LSTM and CNN with an additional attention mechanism.

The classifiers are trained using the Internet Movie Database (IMDB) video review information to measure the effectiveness of the proposed scheme, and the test results demonstrate that the suggested hybrid attention Bi-LSTM and CNN models produce more accurate classification results and also higher recall and F1 scores than the independent multi-layer perceptron (MLP), CNN or LSTM models and hybrid

models. To upset the LSTM's failure to understand the various relationships between the sections of a text. To counter this downside, a 1D CNN was proposed.

The key classification component of their model was built on an attention-based Bi-LSTM. While CNN decreases the input features to be used for estimation, the association between each word and the final classification is not the same for all input terms. They want to leverage all the benefits of CNN and Bi-LSTM in this report. Bi-LSTM is used to efficiently encrypt long-distance term dependences.

The findings of the experiment were carried out thus setting the number of epochs and increasing the quality of the feature from five thousand to fifteen thousand. When the data size was at its height, the suggested model had the highest accuracy at 90.2%, followed by the hybrid model at 87.90%, the CNN model showed the same accuracy as 87.1 per cent, the LSMT model at 82.5% and the MLP model at 72.6%. In terms of average precision, the suggested model was the highest at 87.4%, followed by CNN, Composite, LSTM, and MLP. The F1 score also showed the highest performance at 90.1%, when the data size was at its peak. The hybrid model and the proposed model demonstrate that the precision improves as the scale of the data increases. The proposed hybridized Bi-LSTM and CNN are thus less reliable in their work with limited datasets.

Yi *et al*., (2020) Worked on a Machine learning based customer sentiment analysis for recommending shoppers, shops based on customers' review and suggested solution to a computer-based learning method that uses a multi-class support vector machine (MSVM) to distinguish various types of thoughts and feelings on twitter.

The proposed solution entails the import of twitter data from the twitter source, followed by pre-processing data that requires multiple functions, such as the elimination of contents such as punctuations, incorrect phrases, repetitive or obsolete

terms and stop words. This is accompanied by the use of a multiclass class support vector machine as a classifier to distinguish emotions.

The performance of this Hybrid Recommendation System was evaluated using three metrics namely MAE (Mean absolute error), MSE (Mean squared error) and MAPE (mean absolute percentage error) and the results were compared and analyzed with other contemporary approaches.

The results in their work show that the MAE value of HRS is significantly lower than the existing approaches that were compared. Also, the MAPE value for HRS was nearly 98% which is a clear indicator of a high degree of accuracy. Similarly, the MSE value for Hours displayed a small deviation, which is another good indication of high precision and accuracy. As a result of this, it can be inferred that the proposed Hybrid Recommendation Method (HRS) clearly outperforms other contemporary approaches in terms of reliable estimation of consumer opinion in relation to the buying of a commodity in a specific store. However, dynamics behind the topic under discussion and the context of discussion still remain a black box.

2.4 Modelling Preference

Several supervised machine learning methods have been discussed in literature for the text and document categorization such as Naïve Bayes(Salmi & Rustam, 2019), Neural Networks (Anwaitu Fraser *et al*., 2020), Support Vector Machine(Shafiabady *et al*., 2016).

The Naïve Bayes classification model is based on the presumption of independence between every pair between attributes, includes a probabilistic categorization of the text document such that there is an appropriate number of training instances for each type. Since the Naïve Bayesian approach is solely mathematical, its implementation is

easy and learning time is shorter, but its output is not good for categories identified with very few attributes/characteristics. Hence, SVM or RNN is considered preferable. SVM is found to be very successful for two-class classification problems for example, the text message does not belong to a certain category. The viewpoint is graded as negative or positive. It is often observed that the efficiency of the Neural Network based text classification has been enhanced by assigning the probabilities obtained from the Naïve Bayesian approach as initial weights(Kowsari *et al*., 2019b). The Naïve Bayesian approach was used as a pre-processor for dimensional reduction followed by the SVM method for text classification. There is a need to experiment with more such hybrid approaches in order to obtain the greatest value from machine learning algorithms and to produce better classification performance.

2.5 Gaps Identified In Literature

In (sharkah *et al*, 2015) work, the data set that was used were relatively small, making it difficult to achieve optimum result for classification tasks. Therefore, the result generated on the SVM is not reliable when it comes to a large dataset as it will take more time to test and train. Also, the supervised learning techniques were only based on two files as input when there are chances of using multiple files.

Despite the efficiency of the AL-SVM proposed in (Goudjil *et al*, 2018) work, using the proposed method on labeled sample without any negative effect on the classification performance. The AL-SVM still had to be manually computed which is a major drawback in applying this method to a larger dataset.

(Chunting *et al*; 2015) was able to fill the gap in (Goudjil *et al*, 2018) work by using the concept of Self Organizing Map(SOM) which does the labeling automatically compared to the manual labeling technique in the aforementioned technique. The

SOM that was used in the automatic labelling was more effective when the Correlation Coefficient (CorrCoff) was applied to the existing dataset

Although the C-LSTM approach that was proposed by (Chutung *et al*, 2015) is a good one, as it harnesses the power of both CNN and RNN and then applied the LSTM which is a framework of RNN. The CNN was designed for image processing and this factor makes is less efficient to properly classify textual dataset.

(Wang *et al* 2017) worked on LSTM Approach to Short Text Sentiment Classification by applying the most commonly used dataset in classification task which is the IMDB dataset, but the downside to their work is based in the short text sentiment. However, the method can be applied to both long and short text sentiment.

In (Wang *et al*, 2018) worked on the DRNN for text classification, although novel, it still does not compare to the LSTM approach to text classification task. Also, the ReLu activation function that was used in their work is nothing compared to the sigmoid function in application Recurrent Neural Network

In this review section, we have identified that the major gaps in this research is based on two major factors which are Lack of sufficient dataset and Lack of applying the right method on text classification task for instance the using CNN for a classification task where methods such as statistical method such as Naïve Bayes, SVM and Deep Learning Techniques exist. (Mayor *et al*, 2012) discovered that, it is relatively less cumbersome to define categories that broadly classify the information contained in the collections of data they used.Despite the improved performance of the novel approach of the multi-task learning model proposed by (Pengfei *et al*, 2016). The result of the performance still doesn't measure up to the novel LSTM model with the highest performance when compared to MBOW, MV-RNN, RNTN, DCNN, and PV

CHAPTER THREE

METHODOLOGY

This chapter consists of the two methods and procedures used in the analysis and interpretation of the data. It also discusses the environment and the requirements in which the experiment was carried out. Table 3.1 discusses each objectives and the methodology that will be used to achieve them

Table 3.1 Research Methodology

Objectives	Activity/ Methodology
Identify important features for text classification	Extensive literature review
Formulate the model for text classification using recurrent neural network and support vector machine algorithm.	✓ Flow diagrams ✓ Pseudocodes ✓ Architectures
Implement the model designed formulated in (2) above.	✓ Using a customer review dataset known as customer review. ✓ MATLAB will be used for the analysis of the best technique.
Evaluate the performance of the models.	Result analysis through tables and charts.

3.1 Data Collection and Analysis

The dataset used in this study is from the Amazon product review for customer review. The Amazon dataset contains 3772 customers' review where 70% was used for training and 30% reviews was used for testing. The dataset was retrieved from a data hub repository and the comparative analysis using both the RNN and SVM was

done on both dataset at different time steps to determine which one gives a better performance. Both model was trained and tested in a MATLAB environment.

The datasets were obtained from an online repository for machine training. This study seeks to use open source dataset, which is retrieved from a https://data.world/datafiniti/consumer-reviews-of-amazon-products. This dataset contains about 3772 observation and 16 attributes specific to each customer shopping for a product or more on the amazon web page.

3.2 Overview of Research Methodologies

This section discusses the methods and techniques as well as the data and tools that was used to conduct the study of text classification using SVM and RNN. Figure 3.1 shows a block diagram representation of the work flow. This also seeks to solve the objective 2 in of our aims and objectives.

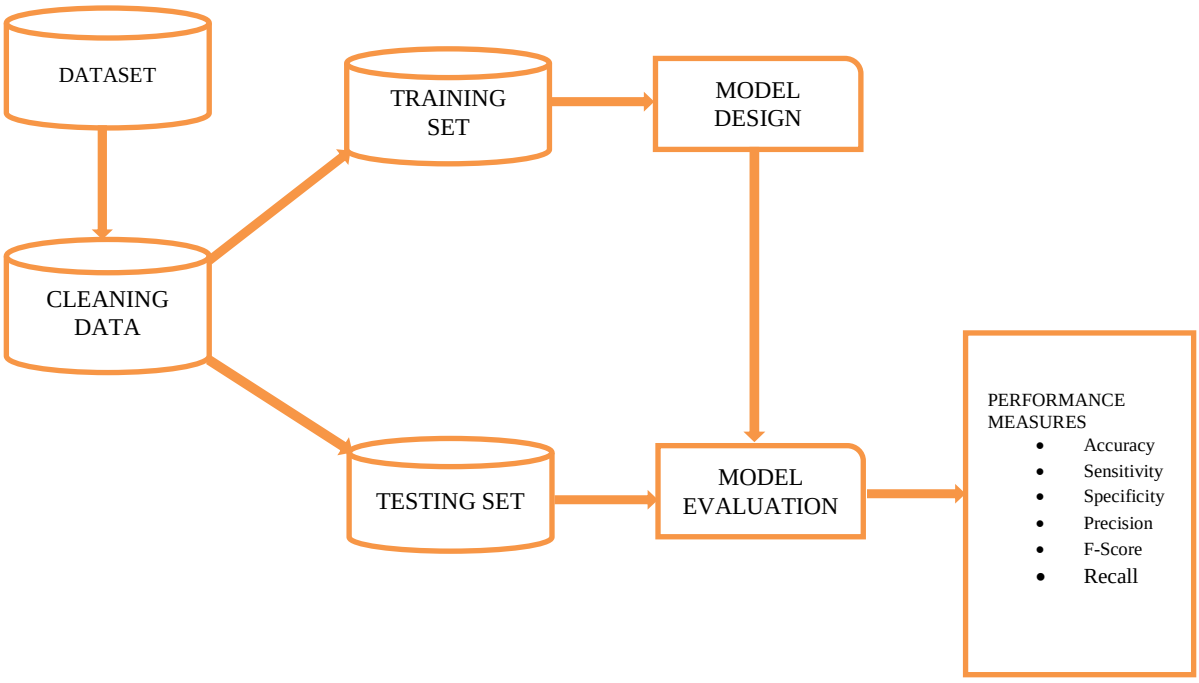

Figure 3.1: Research Framework

3.2.1 Training and Test Set

Since the training set sizes for the two models in this study are different, it is a plausible fact that there is substantially high performance of both models on classification task. To shed light on this, an incremental result on the datasets was presented.

The models (RNN and SVM) are built and optimized on the Amazon Customer Review is the major influences on people's opinion in textual form and a way to properly categorize products using the features on their label. The training and test set are split 70/30 whereas the training was done in two folds for cross validation.

Recurrent Neural Network

The Long Short Time Memory is an artificial Recurrent Neural Network architecture which was developed in order to solve the vanishing gradient problem in the Traditional Recurrent Neural Network. The Long Short Term Memory (LSTM) is the most common architecture in the world and is designed to help capture long-term dependencies. LSTM addresses the major problems with vanilla RNNs by adding a memory cell to recall values over arbitrary time Periods and three gates which are the input gate, exit gate, forget gate to control the flow of information(Minaee *et al.*, 2020).

In this study, Long Short Term Memory will be added to the traditional Recurrent Neural Network so as to achieve an optimal result.

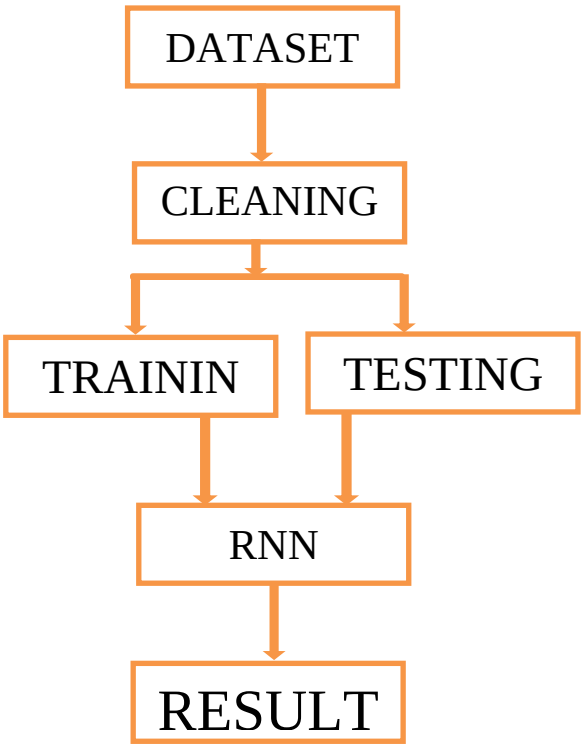

Figure 3.3: Recurrent Neural Network model Work flow

The process described in figure 3.2 shows the steps involved in the design of the Recurrent Neural Network model. The data is fed into the model and the cleaning process is done, the data is then divided into two major partitioning, which is the training and testing respectively. The classifier is then selected after which an output is generated in form of performance matrices

Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning method developed by Cortes and Vapnik in 1995 used for binary classification (Meyer *et al*; 2019). It has since then been developed for use in regression and outlier detection (Scikit-learn; 2019). SVM classifies data in different groups with support vectors used as divisors in a multidimensional space. Support vectors are hyper-planes which is a subspace of one dimension lower than its ambient space. Support vectors are calculated with a kernel function, the three most popular kernel functions are Linear, Polynomial and

Radial Basis Function (RBF), however there are several other kernel functions. To classify a linear classification problem a linear kernel function is recommended (Afonja; 2019). Linear kernel functions calculate linear support vectors that are placed in the multidimensional space with a maximum calculated space from different classification group elements. Nonlinear classification problems require a nonlinear kernel function such as polynomial- or RBF- kernel function. Polynomial kernels work in a similar way as linear functions, except they use a polynomial function that produces polynomial hyper plane curves as divisors in the multidimensional space. RBF kernels nonlinearly maps samples into a multidimensional space (Chang *et al*; 2016).

Parameter Optimization.

Parameter optimization or hyper-parameter optimization technique are methods of improving the performances matrices of Support Vector Machine in a text classification task such as the precision, accuracy and so on. In order to classify the results, Support Vector Machines first find the maximum margin that divides two groups and then output the hyper plane separator at the centre of the marginThe best kernel to be used depends on the classification problem and the data available, and typically has specific parameters that can be managed to fine-tune the output of the Support Vector Machines.(Gaspar *et al*, 2012).

In this study, RBF kernel Support Vector Machine as a result of its flexibility in the classification of non-linear datasets such as the Amazon customer review dataset.

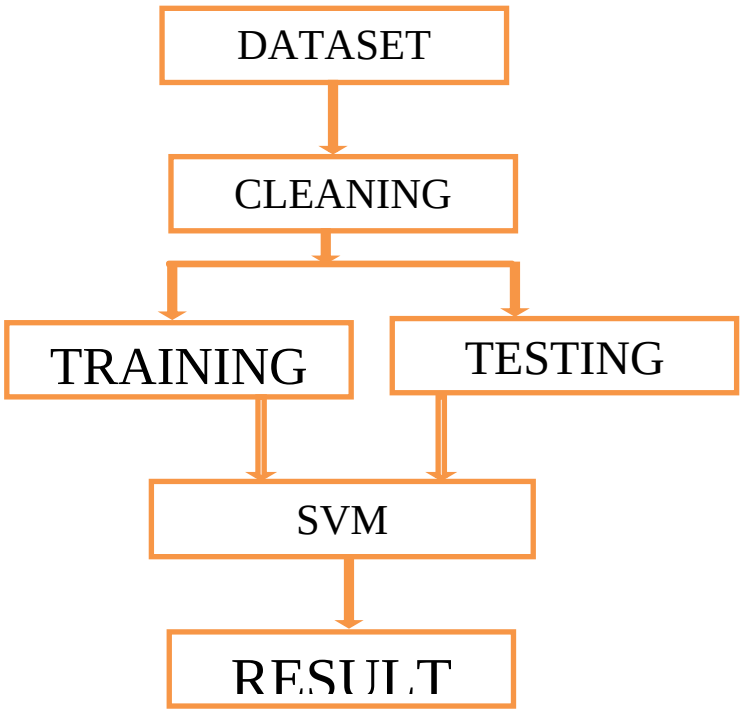

Figure 3.3: Support Vector machine work flow

Support vector machine is implemented in this work as described in figure 3.3 where the data is being fed into the model; it passes through a cleaning process and is partitioned into training and testing respectively. The Support vector machine classifier is then introduced for the test classification purpose, then a result is being generated in form of performance metrics. The table below shows the pseudo code in the table below describes how the Support Vector machine classifier works.

Table 3.2: Support Vector Machine Algorithm

Pseudo-code for SVM

Inputs : Determine the Various Training and Test data
Outputs: Select performance metrics
Select the optimal hyper plane for SVM
while(Stop condition is not met) **do**
Implement SVM training step for each data point
Implement SVM classification for testing data
end while
Return Performance metrics

3.3 Data Training or Model Building

This research adopts a collection of features to classify stream data for Amazon customers and adds additional features to boost classifier performance. Production of a deep learning algorithm for machine learning algorithm, which Are Recurrent Neural Network and Support Vector Machine (SVM) through machine learning resources in a MATLAB environment. The above algorithms have been used to train the data after which the model is designed, this model will be used to evaluate and forecast or correctly identify new instances. Often the key objective of splitting the data into the training set and testing set is to help construct a classifier with a minimal error rate.

3.4 Research Tools

MATLAB (Matrix Laboratory) is used to conduct the experiment as a result of its ease and beefy programming environment for engineers, architects, scientists, scholars, among others, MATLAB is a multi-world arithmetic processing environment and exclusive programming language developed by MathWork. Allows

application monitors, feature and knowledge plotting, algorithm execution, user interface writing in various languages such as C++, Java, FORTRAN, and Python programming languages.

MATLAB is a powerful graphical and computational tool used to answer comparatively complex science and engineering issues which is designed, developed and implemented.

Experimental Setup

In this chapter, the analysis and interpretation of data using MATLAB IDE and text classification toolkits is done. The data collected is compiled in Microsoft Excel .xlsx format. MATLAB is used to implement the experiment.

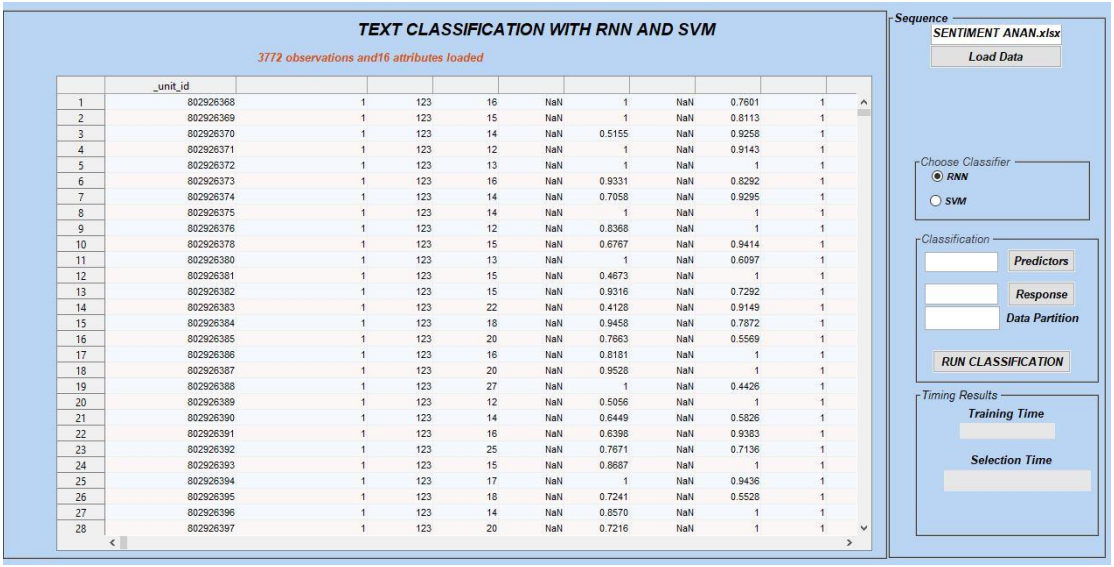

Figure 3.4 Load Data on the MATLAB Environment.

The loaded button is used to import the amazon dataset that was used as described in Figure 3.4 above, for better compilation and user friendliness, the MATLAB software developer tools (GUIDE) was also used to create an interactive environment for easy readability, formatting and interactivity. The GUI is partitioned into three different segments, which are described as seen below.

1. Load the data
2. Partition into testing and training instances
3. Classification

The GUI in figure 4.1 shows when an initial data is supplied to the software using the load tab at the menu bar. The amazon dataset which contains 3772 observations and 16 attribute

3.5 PERFORMANCE EVALUATION METRICES

The performance evaluation metrics of the classifiers is evaluated in terms of accuracy, sensitivity, specificity, precision, f score and recall**.** Assessing machine learning algorithms efficiency needs certain validation metrics. The uncertainty matrix is often used to evaluate four characteristics of the classification model; True positive (TP), True negative (TN), False Positive (FP), and False Negative (FN). It discovers the example categorized correctly and incorrectly from the data set sample given to test the model (Gu *et al*., 2009). This is described as below.

Where:

TPR = True positive rate

SPC = Specificity

PPV = Positive predictive value

NPV = Negative predictive value

FPR = False positive rate

FDR = False discovery rate

FNR = False Negative rate

ACC = Accuracy

F1 = F1 score

MCC = Matthews Correlation Coefficient

TP= True Positive

TN= True Negative

FP = False Positive

FN = False Negative

TPR = TP / (TP + FN)

SPC = TN / (FP + TN)

PPV = TP / (TP + FP)

NPV = TN / (TN + FN)

FPR = FP / (FP + TN)

FDR = FP / (FP + TP)

FNR = FN / (FN + TP)

ACC = (TP + TN) / (TP + TN + FP + FN)

F1 = 2TP / (2TP + FP + FN)

MCC = (TP x TN – FP x FN) / (sqrt((TP + FP) x (TP + FN) x (TN + FP) x (TN + FN))

CHAPTER FOUR

RESULT AND DISCUSSION

This chapter describes the implementation of both Recurrent Neural Network and Support Vector Machine for text classification task of a customer review of the popular amazon dataset in a MATLAB environment.

4.1 Recurrent Neural Network

In this study, Recurrent Neural Network is implemented and the result of the implementation is described below.

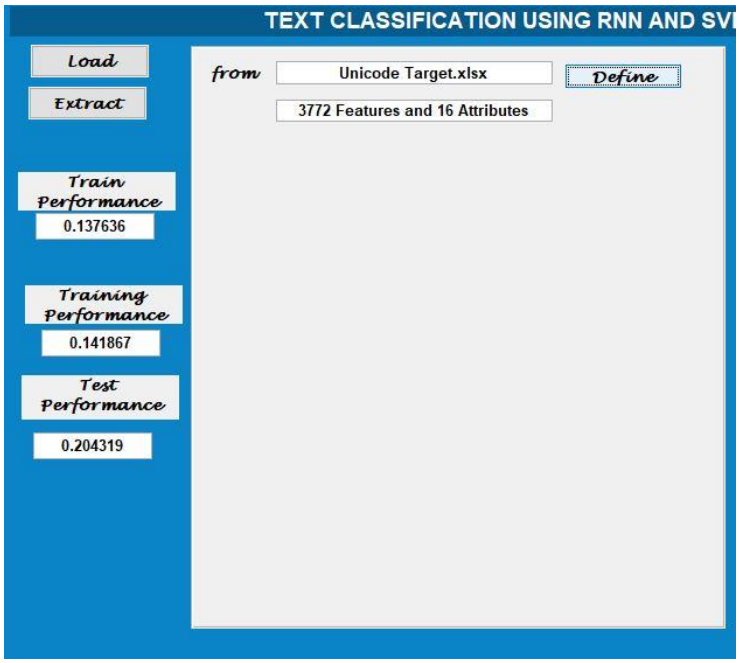

Figure 4.1: Representation of the Training and Testing performance of Recurrent Neural Network.

The RNN classification is performed using 9 input with 20 hidden layer and 2 output. Figure 4.2 shows the representation of the training performance in the classification of

RNN with a training performance of 0.137636 while the test performance is 0.204319 with a dataset that contains the total of 3772 features and 16 attributes. The features in the dataset is being partitioned into 70% for training and 30% for testing and this is done on the benchmark dataset to generate the best possible result.

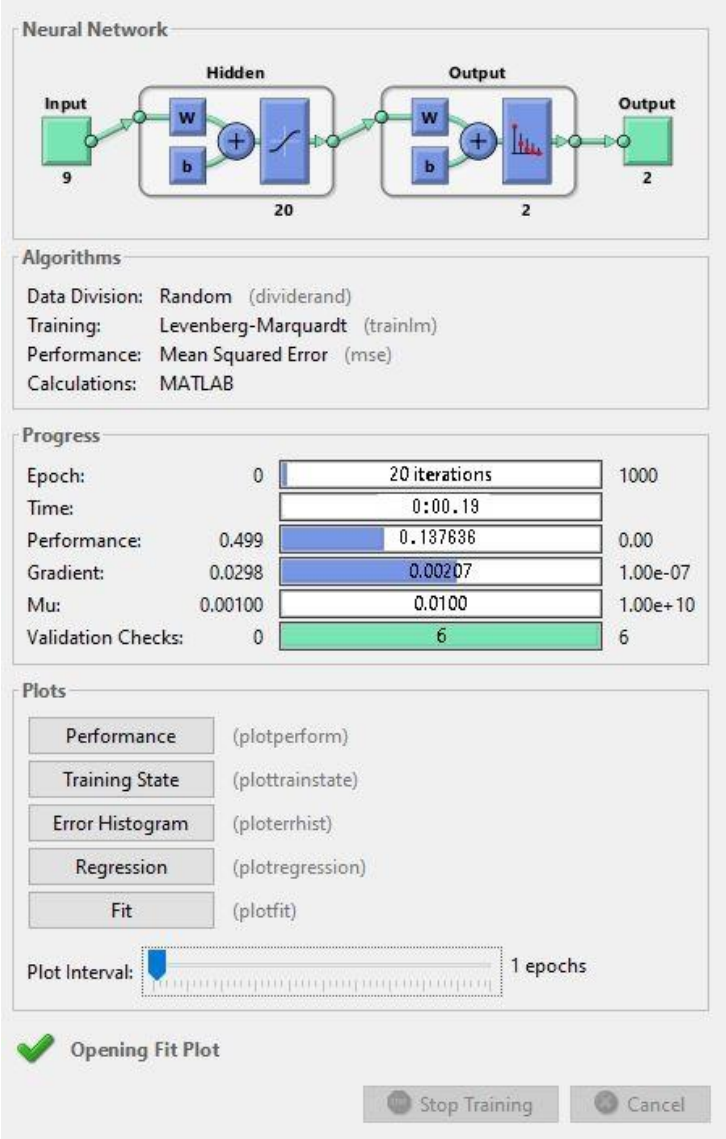

Figure 4.2 The Execution area of the Recurrent Neural Network

The Execution of the experiment shows the 9 input that was feed into the neural network at an epoch of 20 iteration over the 3772 sample data. Also, the hidden layer have 20 generated output after the first input within the time frame of 0.19 seconds.

Figure 4.3 also shows the performance of RNN at 0.137636 and other performance attribute used for the execution.

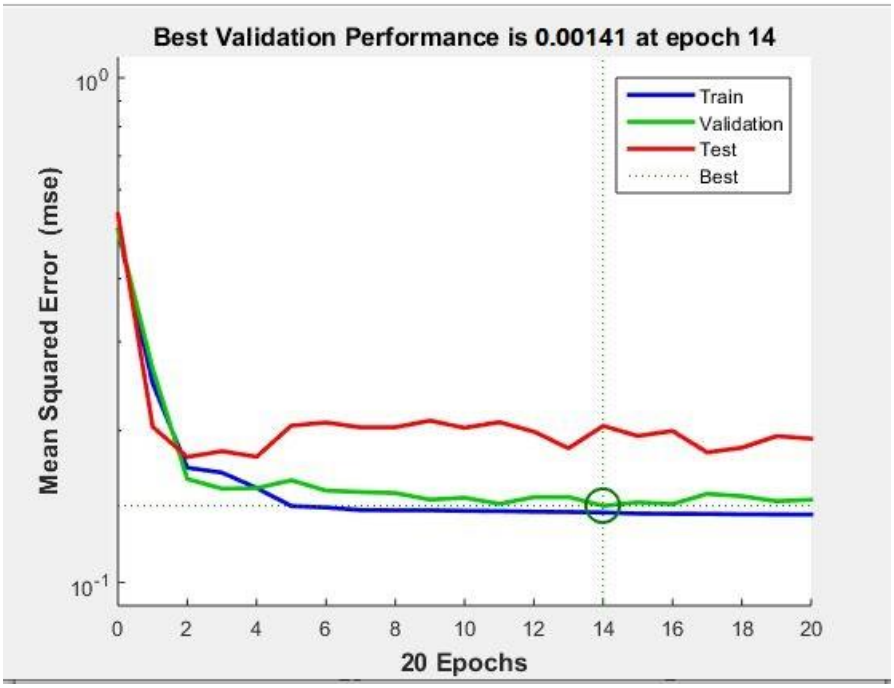

Figure 4.3: Mean Square Error for Epoch

At epoch 14 the result shows that the best fit iteration on the epoch at the mean square error level which is higher than $10^{-1.}$ Therefore, the best performance of the experiment at 0.00141 between the training, testing and validation curve as described in Figure 4.3.

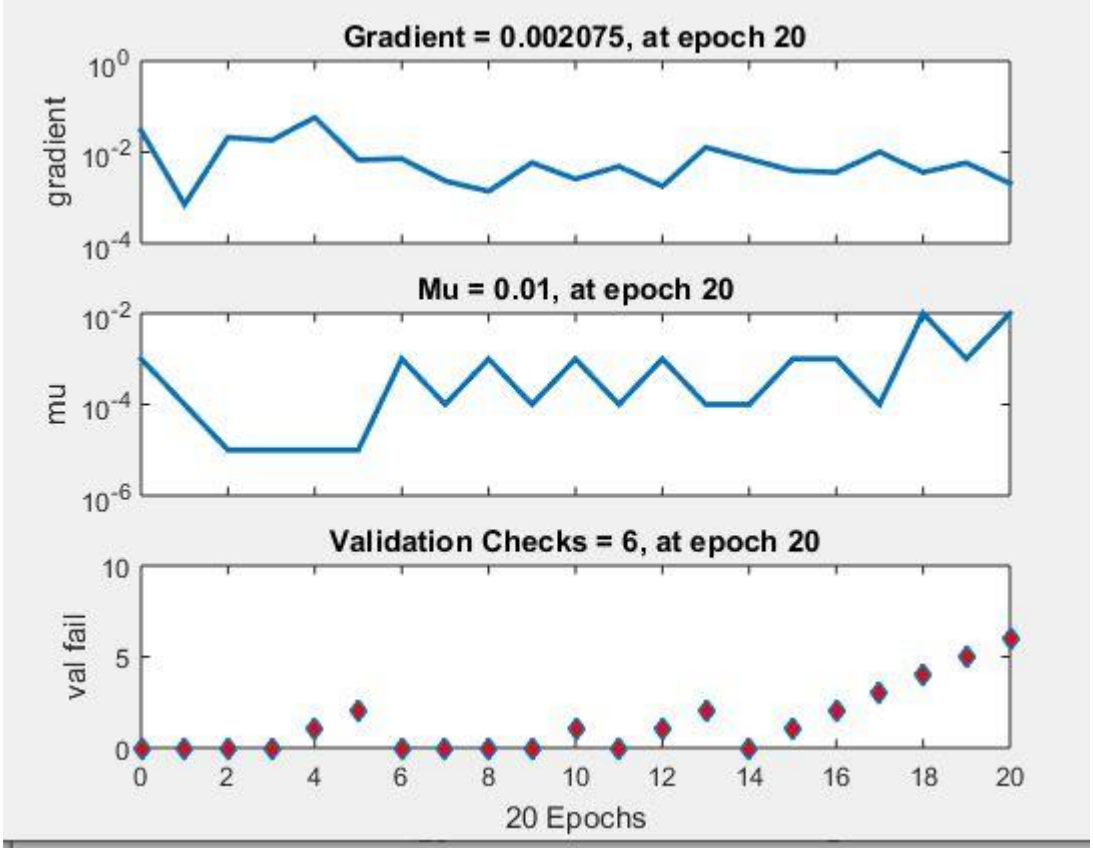

Figure 4.4 Training State Plot for RNN

Figure 4.4 demonstrates the validation check graphs for the complete optimization phase which display the training progress of data and gradients at the 20 epoch. Where the epoch shows the complete step used to update the values of weight. This also shows how the gradient and mu (weight shift of neural network) varies and the amount of validation tests carried out during the 20 epochs of the neural network training process. This outputs of the training phase demonstrate the performance of the Recurrent Neural Network.

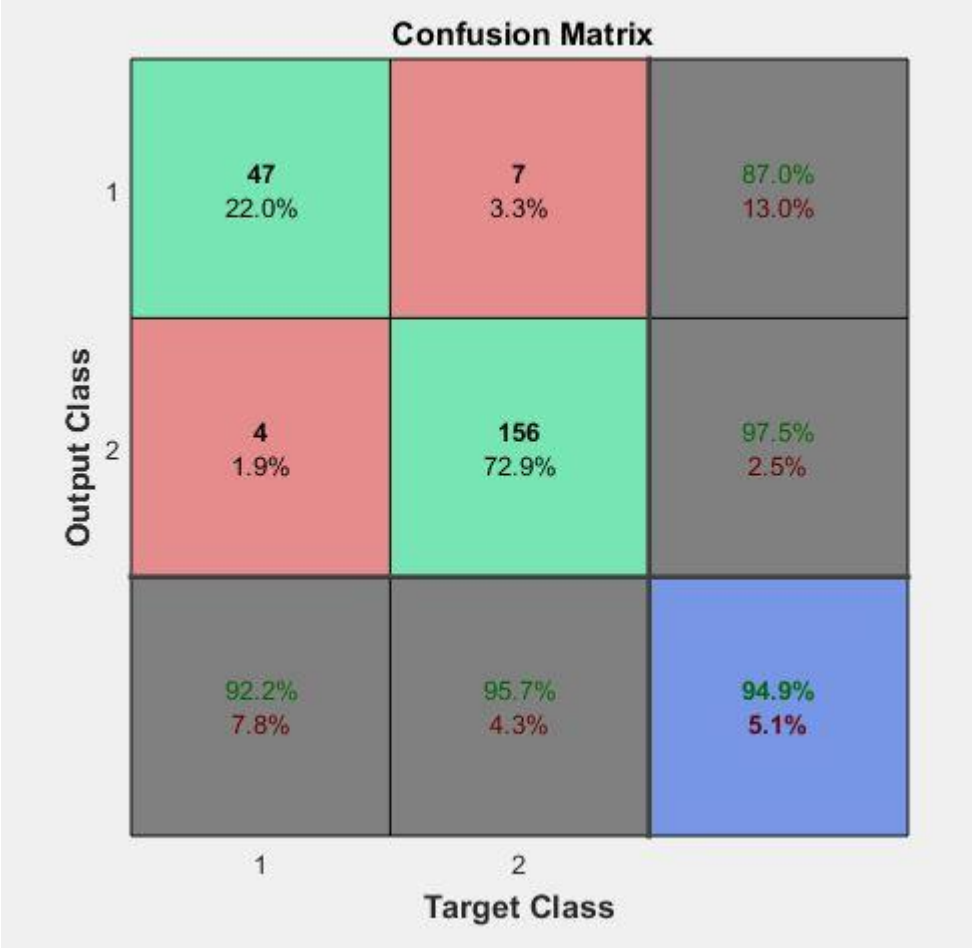

Figure 4.5 Confusion Matrix Table for RNN

In this study, the Confusion matrix of Recurrent Neural Network was computed with the customer review dataset. With metrics True Positive (TP) = 156, True Negative (TN) = 47. False Negative (FN) = 7, False positive (FP) = 4 to calculate the performance metrics in table 4.1.

ACCURACY: This is the simplest scoring measure. Is calculates the proportion of correctly classified instances

Accuracy = (TN + TN)(TP + TN +FP +FN)

=(156+47)(156+47+4+7)

=94.86%

SENSITIVITY: The sensitivity rate also referred to as the Recall or True positive, tells is how likely the test will come back positive on a sample

Sensistivity = TP/(TP + FN)

= 47/ (156 + 7)

=97.5%

SPECIFICITY: The specificity also known as the true negative relates to the classifier ability to identify negative results

Specificity = TN/(TP + TN)

= 47/(4 + 47)

= 87.04%

PRECISION: This is a measure of retrieval instance that are relevant

Precision = TP /(TP + FP)

= 156 (156 + 4)

= 95.71%

F-SCORE: Its is way to measure a model accuracy based on recall and precision

F-score = 2*TP/(2*TP + FP+FN)

= 2 * 156 / (2 * 156 + 4 + 7)

= 96. 59%

FALSE POSITITIVE RATE: it occurs when we accept a user to whom should be rejected.

False positive rate = FP/(FN + TN)

=4/ (7 + 47)

= 12.97%

FALSE NEGATIVE RATE : This occur when we reject a user to whom should be accepted.

False Negative Rate = FN /(FN + TP)

= 7 /(7 + 156)

= 0.25%

Table 4.1: Performance metrics for RNN

Performance Metrics	Results (%)
Sensitivity	97.5
Specificity	87.04
Precision	95.71
Negative predictive value	92.16
False Positive Rate	12.96
False Discovery Rate	4.29
False Negative Rate	0.25
Accuracy	94.86
F1 score	96.59
Matthews Correlation Coefficient	86.18

The significant application of the data obtained from the amazon dataset for customer review on a text classification task is not consistent and to achieve such classification, a variety of parameter, a variety of algorithms have been proposed. These algorithms typically require optimization of parameters to get detailed results. It is very important to find an optimal collection of textual data that can be used in classification whether it is true or false.

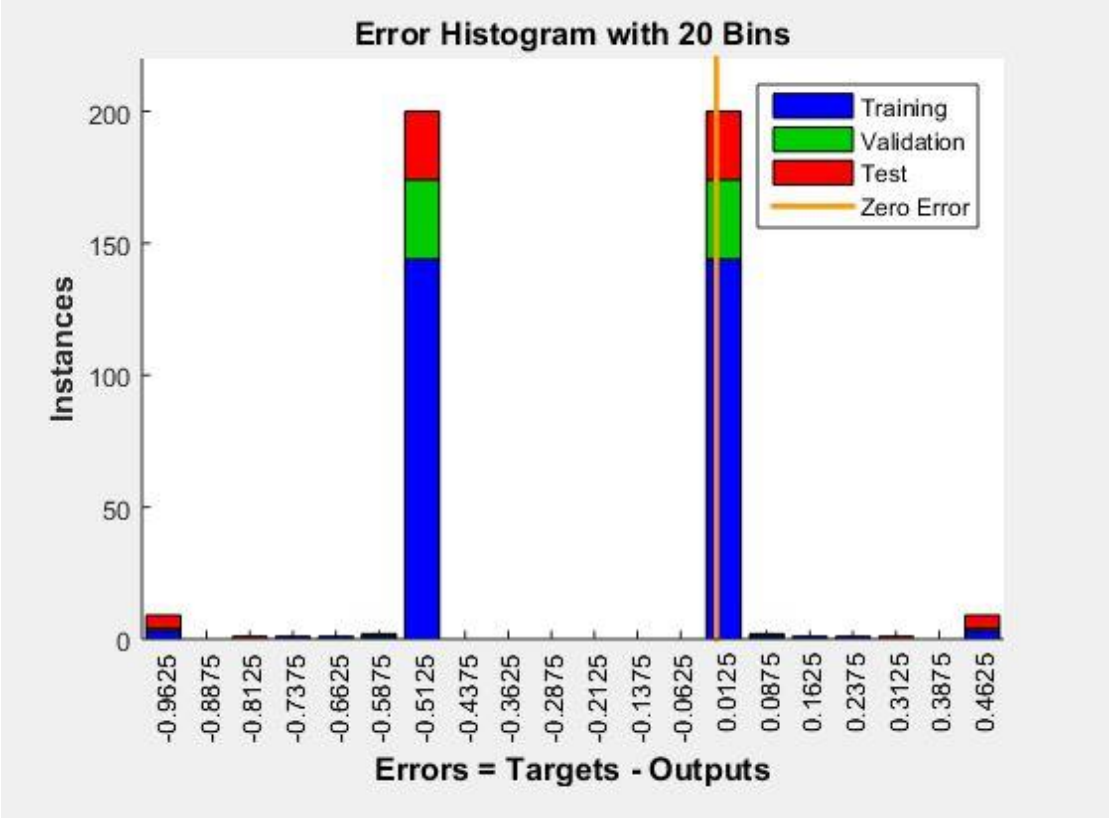

Figure 4.6: Error Histogram for RNN

Using RNN, for instances of about 200, there are 12 errors which are represented on the histogram with the highest recorded error at 0.0125. This is also the point at which there is no error recorded with over 200 instances of iteration. The RNN is passed through with 20 histogram Bins.

For evaluating the model, both short and long term dependencies obtained from the proposed recurrent neural network and normal recurrent model are evaluated using an error histogram are shown in figures 4.6. Based on the inclination and level of error per histogram bin, the Recurrent Neural Network model produces lower error rates in comparison with the state of art. The error histogram also creates a distinction between the training, testing, and validation which ranges between 150 to 200 instances in the classification. However, as a result of the proper partitioning of the training and testing of 70% and 30% respectively, we are able to achieve an

equivalent result in areas where are there is least possible error and in areas where there is highest number of error.

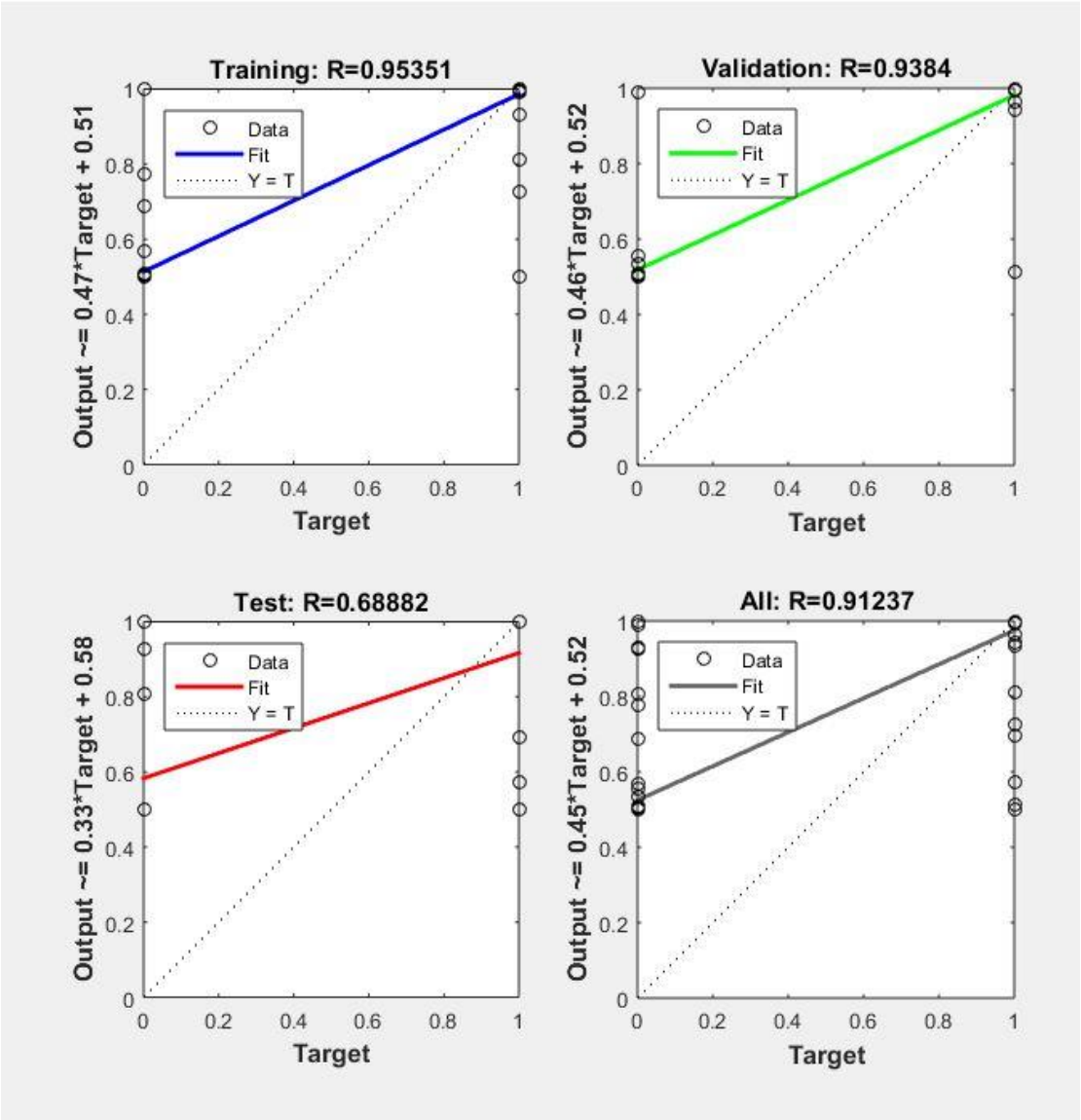

Figure 4.7. Regression plot for RNN

The regression chart shown in figure 4.7 indicates the relationship between the outputs of the network and the goals. Essentially, a perfect scenario means that the preparation was fine, which in essence means that the results of the network and the goals are precisely the same, but this is hardly the scenario. As a result of this, the regression values in the figure above calculate the association between outputs and goals. Each of the four plots has a dotted line reflecting optimal outcomes, and the

solid line holds a significant match line from our datasets. The point of intersection is the linear regression line between both the outputs of the produced network and the predicted information.

4.2 Support Vector Machine

In this study, Support Vector Machine is implemented and below is the result for the experiment.

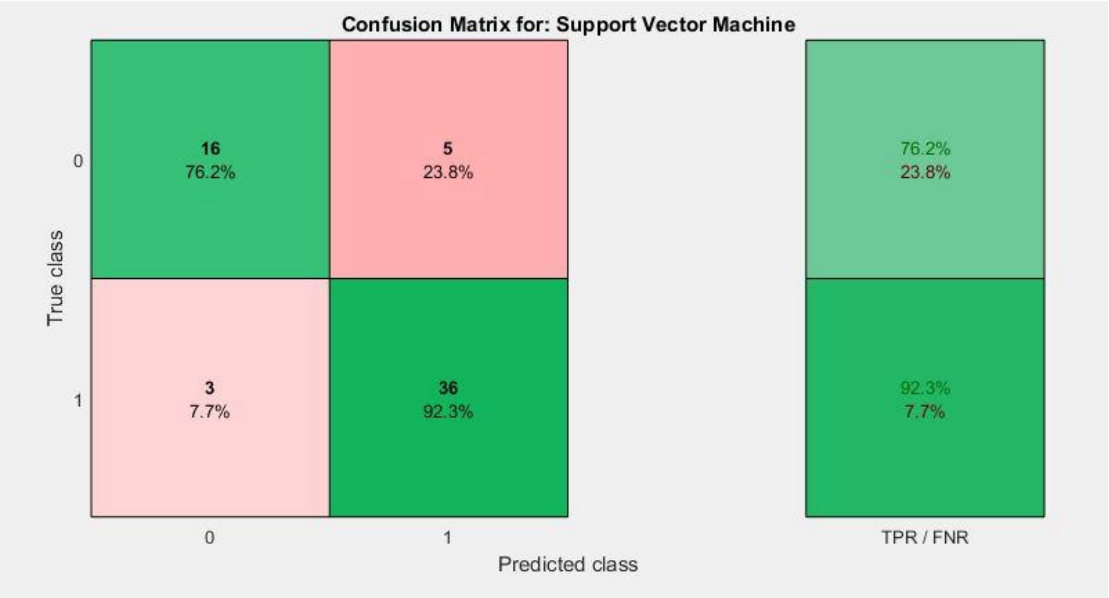

Figure 4.8: Confusion matrix for SVM

Figure 4.8 shows the confusion matrix of support vector machine (SVM) classifier computed with the amazon customer review dataset. With the confusion metrics True Positive (TP) = 36, True Negative (TN) = 16. False Negative (FN) = 3, False positive (FP) = 5 to calculate the performance matrix in table 4.2.

Accuracy = (TP + TN)(TP + TN +FP +FN)

=(36+16)(36+16+5+3)

=86.67%

Sensistivity = TP/(TP + FN)

= 36/ (36 + 3)

=92.3%

Specificity = TN/(TP + TN)

= 16/(36 + 3)

= 76.19%

Precision = TP /(TP + FP)

= 36 (36 + 5)

= 8.7%

F-score = 2*TP/(2*TP + FP+FN)

= 2 * 36 / (2 * 36 + 5 + 3)

= 90.0%

False positive rate = FP/(FN + TN)

=5/ (3 + 16)

= 23.81%

False Negative Rate = FN /(FN + TP)

= 3 /(3 + 36)

= 0.76%

Table 4.2: Performance metrics of SVM

Performance Metrics	Results (%)
Sensitivity	92.3
Specificity	76.19
Precision	87.8
Negative predictive value	84.21
True Positive Rate	23.81

False Discovery Rate	1.22
False Negative Rate	0.76
Accuracy	86.67
F1 score	90.0
Matthews Correlation Coefficient	70.23

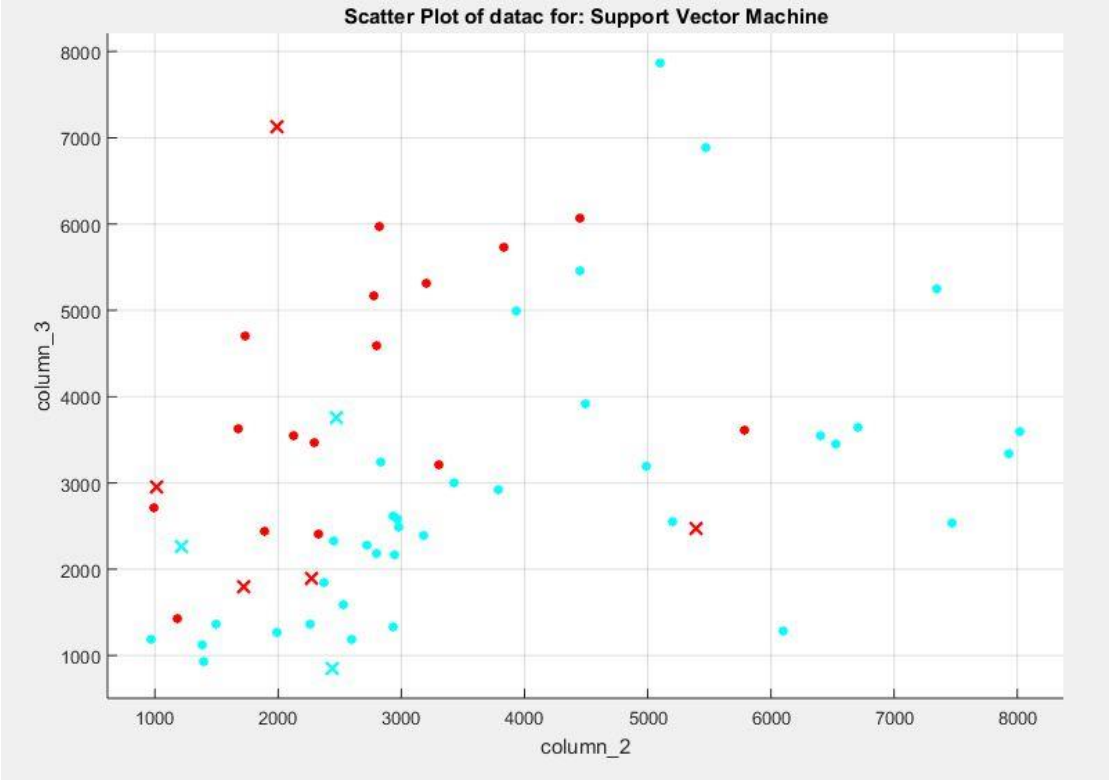

Figure 4.9 Scatter plot of data for Support Vector Machine

The results of the classification shows the difference between the similarities of each sample mean of each class. The different colours on the graph as described in the graphs in figure 4.9 shows that there is a clear distinctions between the positive and negative reviews. Therefore, classification using the amazon customer review dataset is done linearly, making it is easy to separate between the positive and negative sentiment in peoples review concerning a product.

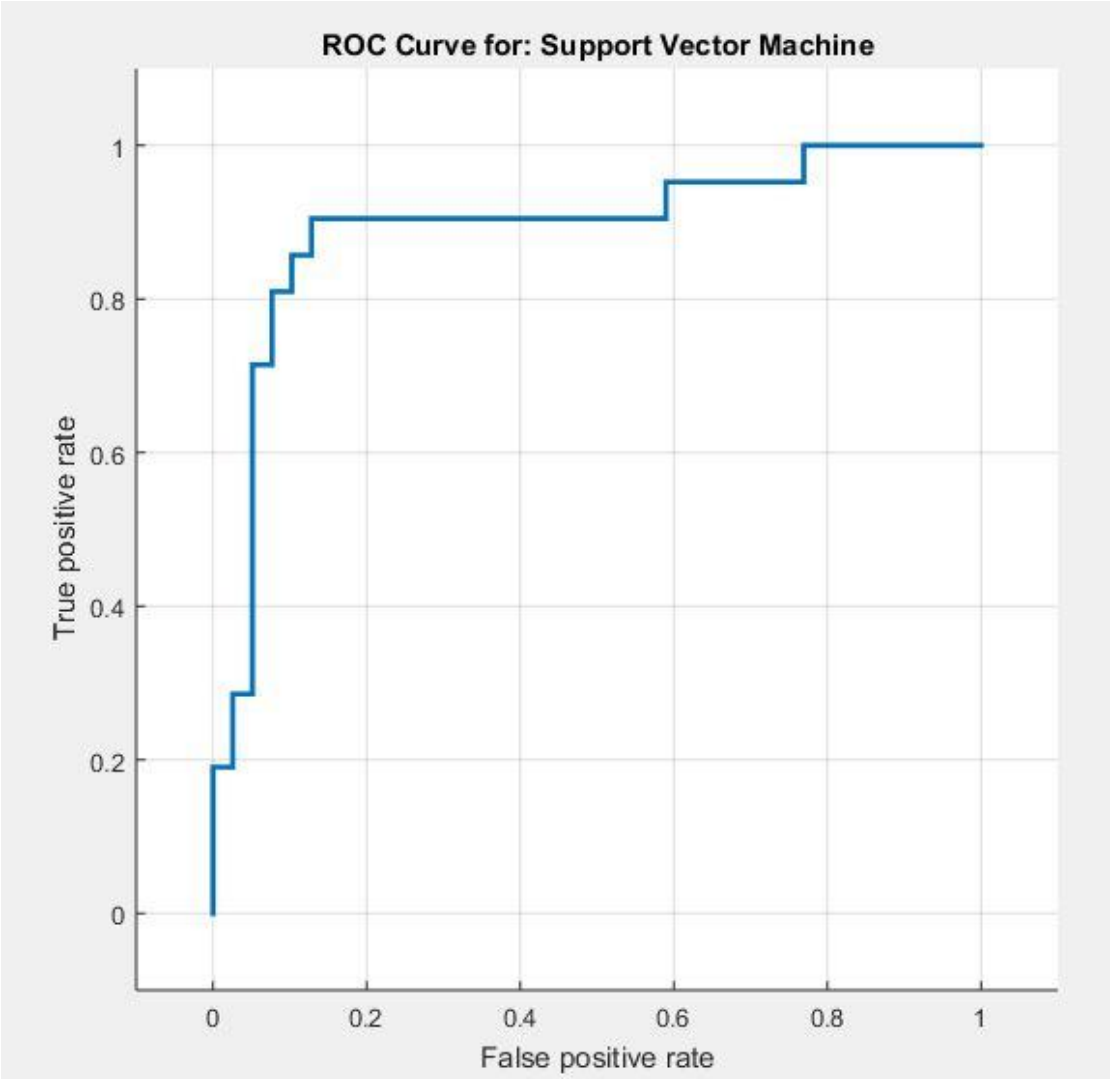

Figure 4.10 ROC Curve for Support Vector Machine

Figure 4.10 shows the Receiver Operating Characteristic curve, with the true positive rate of the SVM after the training and the testing of the 3772 features and the 16 attributes of the amazon customer review dataset, based on the 70 to 30 percentage partitioning rate. This ROC curve shows that the positive result for most of review increase relatively at each time step between 0 and 1 and these shows that the true negative result of the SVM generate a positive output on the classification task.

4.3 COMPARATIVE ANALYSIS

Table 4.3 Performance matrix table for RNN and SVM

Performance matrices	Result for SVM%	Result for RNN%
Sensitivity	92.3	97.5
Specificity	76.17	87.04
Precision	87.8	95.71
Negative predictive value	84.21	92.16
False positive value	23.81	12.96
False discovery rate	1.22	4.29
False Negative rate	0.76	0.25
Accuracy	86.67	94.86
F1 Score	90.0	96.59
Matthew correlation coefficient	70.23	86.18

The comparative analysis of Recurrent Neural Network and Support vector Machine results are carried out and shown in table 4.3. However, the results shows that Recurrent Neural Network (RNN) with accuracy of 94.86% outperforms Support Vector Machine (SVM) with accuracy of 86.67. Also, the result shows that the Recurrent Neural network is better than the Support Vector Machine in terms of Specificity, sensitivity, F1 score, and recall.

In this study, experiment has been performed to classify a pre-processed dataset to know if a customer opinion is Positive or Negative. The classification techniques that was employed in this study is Recurrent Neural Network and Support Vector Machine, the Recurrent Neural Network outperforms the support Vector machine in term of accuracy and precision.

4.4 Comparative Performance Measures of other Technique

The result of the experiment in this study is further compared with existing methods in literature and has proven to be an efficient method which can be adopted by researchers for further investigations and improvements.

4.4 Performance matrix of other techniques

Authors	Techniques	Accuracy %
(Jang *et al*., 2020)	Bi-LSTM	91.41
(Putong *and* Suharjito, 2020)	Naïve bayes + word2Vec	62.30
(An *et al*., 2017)	Character-level CNN	77.8
(Tsangaratos *and* Ilia, 2016)	Logistic Regression	86.2
(Ranjan *and* Prasad, 2018)	BPLion Neural Network	70.0
(Dwivedi *and* Arya, 2016)	KNN	76.0
(Chaya *et al*, 2020)	Multivariate Bernoulli NB	70.96
(Caccamisi *et al*., 2020)	Naïve bayes	87.45

Table 4.4 shows the comparison between the proposed methods and other methods that have been implemented in literature on using machine learning techniques for text classification task. With the result in the table above, the Recurrent Neural Network in this study achieves a better result than every other techniques with the accuracy of 94.86%

CHAPTER FIVE

SUMMARY, CONCLUSION, AND RECOMMENDATION

5.1 Summary

In this study, a comparative study of Recurrent Neural Network and Support Vector Machine for text classification was done for customer review on an ecommerce benchmark dataset. The implementation was done on both algorithms at each time step and result is generated in terms of F1 score, Accuracy, Specificity, Recall and Precision. At the end of the experimentation, Recurrent Neural Network performed better than the Support Vector Machine in term of accuracy. Hence in an ecommerce environment for a positive or negative customer review, RNN will assist customers in recognizing merchant's opinion concerning a product, and to inform other people's decision regarding a particular product. It is also noted that the two algorithms that was proposed in this study was far better than existing literatures in terms of classification as describes in table 4.6.

5.2 Future Work and Recommendation

In the future, work can be done to enhance and improve these two techniques in order to achieve better result. The application of this study can be seen in large ecommerce environment, where there is the need to attract more customer and make more sales on products. Therefore, with the advancement of Machine Learning methods in our day to day activities, the needs to implement other nuance techniques. Work can also be done to enhance the performance of RNN and SVM by introducing hybridized technique such as clustering techniques before classification.

5.3 Contribution to knowledge

The main input if this study to the body of knowledge is to understand the inner working of Recurrent Neural Network and Support Vector Machine in a way that it can be understood by other researchers in a way that it can be further improved and worked on to build new and enhanced system.

This study implements Support Vector Machine and Recurrent Neural Network on a customer review dataset for text classification.

5.4 Conclusion

In this study, Recurrent Neural Network and Support Vector Machine was implemented to create a distinction between customer opinions about a product, using the amazon product review dataset. With the accuracy of this study, a clear distinction has been made on which algorithm performs better in the classification of textual data In an Ecommerce environment.

REFERENCES

Abdelaal, H. M., Elmahdy, A. N., Halawa, A. A., and Youness, H. A. (2018). Improve the automatic classification accuracy for Arabic tweets using ensemble methods. *Journal of Electrical Systems and Information Technology*, 5(3), 363–370. https://doi.org/10.1016/j.jesit.2018.03.001

Abiodun, O. I., Kiru, M. U., Jantan, A., Omolara, A. E., Dada, K. V., Umar, A. M., Linus, O. U., Arshad, H., Kazaure, A. A., and Gana, U. (2019). Comprehensive Review of Artificial Neural Network Applications to Pattern Recognition. *IEEE Access*, 7, 158820–158846. https://doi.org/10.1109/ACCESS.2019.2945545

Adewale Aderoju, S., and Teju Jolayemi, E. (2019). Issues of Class Imbalance in Classification of Binary Data: A Review. *International Journal of Data Science and Analysis*, 5(6), 123. https://doi.org/10.11648/j.ijdsa.20190506.13

An, Y., Tang, X., and Xie, B. (2017). Sentiment analysis for short Chinese text based on character-level methods. *2017 9th International Conference on Knowledge and Smart Technology (KST)*, 78–82. https://doi.org/10.1109/KST.2017.7886093

Anwaitu Fraser, E., Okonkwo, and Obikwelu R. (2020). Artificial Neural Networks for Medical Diagnosis: A Review of Recent Trends. *International Journal of Computer Science & Engineering Survey*, 11(3), 1–11. https://doi.org/10.5121/ijcses.2020.11301

Bologna, G., and Hayashi, Y. (2018). A Comparison Study on Rule Extraction from Neural Network Ensembles, Boosted Shallow Trees, and SVMs. *Applied Computational Intelligence and Soft Computing*, 2018, 1–20. https://doi.org/10.1155/2018/4084850

Branco, P., Torgo, L., and Ribeiro, R. P. (2016). A Survey of Predictive Modeling on Imbalanced Domains. *ACM Computing Surveys*, 49(2), 1–50. https://doi.org/10.1145/2907070

Brownlee, J. (2020). Supervised and unsupervised machine learning algorithms. *Machine Learning Mastery*. https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

Burba, F., Ferraty, F., and Vieu, P. (2009). k -Nearest Neighbour method in functional nonparametric regression. *Journal of Nonparametric Statistics*, 21(4), 453–469. https://doi.org/10.1080/10485250802668909

Caccamisi, A., Jørgensen, L., Dalianis, H., and Rosenlund, M. (2020). Natural language processing and machine learning to enable automatic extraction and classification of patients' smoking status from electronic medical records. *Upsala Journal of Medical Sciences*, 125(4), 316–324. https://doi.org/10.1080/03009734.2020.1792010

Chaya Liebeskind, S. L. (2020). Deep Learning for Period Classification of Historical Hebrew Texts. *Journal of Data Mining and Digital Humanities* ISSN 2416-5999, an Open-Access Journal.

Chen, J., Feng, J., Sun, X., and Liu, Y. (2019). Co-Training Semi-Supervised Deep Learning for Sentiment Classification of MOOC Forum Posts. *Symmetry*, 12(1), 8. https://doi.org/10.3390/sym12010008

Chen, M., Sun, J.-T., Ni, X., and Chen, Y. (2011). Improving context-aware query classification via adaptive self-training. *Proceedings of the 20th ACM International Conference on Information and Knowledge Management - CIKM '11*, 115. https://doi.org/10.1145/2063576.2063598

Cioffi, R., Travaglioni, M., Piscitelli, G., Petrillo, A., and De Felice, F. (2020). Artificial Intelligence and Machine Learning Applications in Smart Production: Progress, Trends, and Directions. *Sustainability*, 12(2), 492. https://doi.org/10.3390/su12020492

Dada, E. G., Bassi, J. S., Chiroma, H., Abdulhamid, S. M., Adetunmbi, A. O., and Ajibuwa, O. E. (2019). Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, 5(6), e01802. https://doi.org/10.1016/j.heliyon.2019.e01802

Darling-Hammond, L., Flook, L., Cook-Harvey, C., Barron, B., and Osher, D. (2020). Implications for educational practice of the science of learning and development. *Applied Developmental Science*, 24(2), 97–140. https://doi.org/10.1080/10888691.2018.1537791

Di Franco, G., and Santurro, M. (2020). Machine learning, artificial neural networks and social research. *Quality & Quantity*. https://doi.org/10.1007/s11135-020-01037-y

Diab, D. M., and El Hindi, K. (2018). Using differential evolution for improving distance measures of nominal values. *Applied Soft Computing*, 64, 14–34. https://doi.org/10.1016/j.asoc.2017.12.007

Dwivedi, S. K., and Arya, C. (2016). Automatic Text Classification in Information retrieval. *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies - ICTCS '16*, 1–6. https://doi.org/10.1145/2905055.2905191

El Hindi, K. (2014). Fine tuning the Naïve Bayesian learning algorithm. *AI Communications*, 27(2), 133–141. https://doi.org/10.3233/AIC-130588

Gaspar, P., Carbonell, J., & Oliveira, J. L. (2012). On the parameter optimization of Support Vector Machines for binary classification. *Journal of Integrative Bioinformatics*, *9*(3), 201. https://doi.org/10.1515/jib-2012-201

Habimana, O., Li, Y., Li, R., Gu, X., and Yan, W. (2020). Attentive convolutional gated recurrent network: a contextual model to sentiment analysis. *International Journal of Machine Learning and Cybernetics*, 11(12), 2637–2651. https://doi.org/10.1007/s13042-020-01135-1

Hartmann, J., Huppertz, J., Schamp, C., and Heitmann, M. (2019). Comparing automated text classification methods. *International Journal of Research in Marketing*, 36(1), 20–38. https://doi.org/10.1016/j.ijresmar.2018.09.009

Hassan, S., Mihalcea, R., and Banea, C. (2007). Random Walk Term Weighting For Improved Text Classification. *International Journal of Semantic Computing*, 01(04), 421–439. https://doi.org/10.1142/S1793351X07000263

Hassani, H., Beneki, C., Unger, S., Mazinani, M. T., and Yeganegi, M. R. (2020). Text Mining in Big Data Analytics. *Big Data and Cognitive Computing*, 4(1), 1. https://doi.org/10.3390/bdcc4010001

Huang, C.-L., and Wang, C.-J. (2006). A GA-based feature selection and parameters optimization for support vector machines. *Expert Systems with Applications*, 31(2), 231–240. https://doi.org/10.1016/j.eswa.2005.09.024

Ifrim, G., Bakir, G., and Weikum, G. (2008). Fast logistic regression for text categorization with variable-length n-grams. *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 08*, 354. https://doi.org/10.1145/1401890.1401936

Jang, B., Kim, M., Harerimana, G., Kang, S., and Kim, J. W. (2020). Bi-LSTM Model to Increase Accuracy in Text Classification: Combining Word2vec CNN and

Attention Mechanism. *Applied Sciences*, 10(17), 5841. https://doi.org/10.3390/app10175841

Jin Huang, and Ling, C. X. (2005). Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 17(3), 299–310. https://doi.org/10.1109/TKDE.2005.50

Korde, V. (2012). Text Classification and Classifiers:A Survey. *International Journal of Artificial Intelligence & Applications*, 3(2), 85–99. https://doi.org/10.5121/ijaia.2012.3208

Kotsiantis, S. B. (2013). Decision trees: a recent overview. *Artificial Intelligence Review*, 39(4), 261–283. https://doi.org/10.1007/s10462-011-9272-4

Kowsari, Jafari Meimandi, Heidarysafa, Mendu, Barnes, and Brown. (2019a). Text Classification Algorithms: A Survey. Information, 10(4), 150. https://doi.org/10.3390/info10040150

Kowsari, Jafari Meimandi, Heidarysafa, Mendu, Barnes, and Brown. (2019b). Text Classification Algorithms: A Survey. *Information*, 10(4), 150. https://doi.org/10.3390/info10040150

Liu, Y., Ju, S., Wang, J., and Su, C. (2020). A New Feature Selection Method for Text Classification Based on Independent Feature Space Search. *Mathematical Problems in Engineering*, 2020, 1–14. https://doi.org/10.1155/2020/6076272

Mehboob, U., Qadir, J., Ali, S., and Vasilakos, A. (2016). Genetic algorithms in wireless networking: techniques, applications, and issues. *Soft Computing*, 20(6), 2467–2501. https://doi.org/10.1007/s00500-016-2070-9

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., and Gao, J. (2020). Deep Learning Based Text Classification: A Comprehensive Review. *ArXiv*. arXiv.

Mitchell, T. M. (2004). The Role of Unlabeled Data in Supervised Learning. In Language, Knowledge, and Representation (pp. 103–111). *Springer Netherlands*. https://doi.org/10.1007/978-1-4020-2783-3_7

Mollalo, A., Rivera, K. M., and Vahedi, B. (2020). Artificial Neural Network Modeling of Novel Coronavirus (COVID-19) Incidence Rates across the Continental United States. *International Journal of Environmental Research and Public Health*, 17(12), 4204. https://doi.org/10.3390/ijerph17124204

Peloia, P. R., Bocca, F. F., and Rodrigues, L. H. A. (2019). Identification of patterns for increasing production with decision trees in sugarcane mill data. *Scientia Agricola*, 76(4), 281–289. https://doi.org/10.1590/1678-992x-2017-0239

Pengfei Liu, Xipeng Qiu, uanjing H. (2016). Recurrent Neural Network for Text Classification with Multi-Task Learning. *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*, 2873–2879.

Pise, N. N., and Kulkarni, P. (2008). A Survey of Semi-Supervised Learning Methods. *2008 International Conference on Computational Intelligence and Security*, 30–34. https://doi.org/10.1109/CIS.2008.204

Putong, M. W., and Suharjito, S. (2020). Classification Model of Contact Center Customers Emails Using Machine Learning. *Advances in Science, Technology and Engineering Systems Journal*, 5(1), 174–182. https://doi.org/10.25046/aj050123

Ramola, R., Jain, S., and Radivojac, P. (2019). Estimating classification accuracy in positive-unlabeled learning: characterization and correction strategies. Pacific Symposium on Biocomputing. *Pacific Symposium on Biocomputing*, 24, 124–135. http://www.ncbi.nlm.nih.gov/pubmed/30864316

Ranjan, N. M., and Prasad, R. S. (2018). Automatic text classification using BPLion-neural network and semantic word processing. *The Imaging Science Journal*, 66(2), 69–83. https://doi.org/10.1080/13682199.2017.1376781

Schmidt, J., Marques, M. R. G., Botti, S., and Marques, M. A. L. (2019). Recent advances and applications of machine learning in solid-state materials science. *Npj Computational Materials*, 5(1), 83. https://doi.org/10.1038/s41524-019-0221-0

Segata, N., Blanzieri, E., Delany, S. J., and Cunningham, P. (2010). Noise reduction for instance-based learning with a local maximal margin approach. *Journal of Intelligent Information Systems*, 35(2), 301–331. https://doi.org/10.1007/s10844-009-0101-z

Seising, R. (2018). The Emergence of Fuzzy Sets in the Decade of the Perceptron—Lotfi A. Zadeh's and Frank Rosenblatt's Research Work on Pattern Classification. *Mathematics*, 6(7), 110. https://doi.org/10.3390/math6070110

Shafiabady, N., Lee, L. H., Rajkumar, R., Kallimani, V. P., Akram, N. A., and Isa, D. (2016). Using unsupervised clustering approach to train the Support Vector Machine for text classification. *Neurocomputing*, 211, 4–10. https://doi.org/10.1016/j.neucom.2015.10.137

Shang, S., Shi, M., Shang, W., and Hong, Z. (2016). Improved Feature Weight Algorithm and Its Application to Text Classification. *Mathematical Problems in Engineering*, 2016, 1–12. https://doi.org/10.1155/2016/7819626

Shanmuganathan, S. (2016). Artificial Neural Network Modelling: An Introduction (pp. 1–14). https://doi.org/10.1007/978-3-319-28495-8_1

Sharkah anurag, Chatterjee saptarhi, Das writayan, Datta debabrata (2015). Text Classification Using support vector machine. *International Journal of Engineering Science Invention*, 4(11), 33–37.

Soria, D., Garibaldi, J. M., Ambrogi, F., Biganzoli, E. M., and Ellis, I. O. (2011). A 'non-parametric' version of the naive Bayes classifier. *Knowledge-Based Systems*, 24(6), 775–784. https://doi.org/10.1016/j.knosys.2011.02.014

Su, B., and Lu, S. (2017). Accurate recognition of words in scenes without character segmentation using recurrent neural network. *Pattern Recognition*, 63, 397–405. https://doi.org/10.1016/j.patcog.2016.10.016

Tanha, J., van Someren, M., and Afsarmanesh, H. (2017). Semi-supervised self-training for decision tree classifiers. *International Journal of Machine Learning and Cybernetics*, 8(1), 355–370. https://doi.org/10.1007/s13042-015-0328-7

Thangaraj, M., and Sivakami, M. (2018). Text Classification Techniques: A Literature Review. *Interdisciplinary Journal of Information, Knowledge, and Management*, 13, 117–135. https://doi.org/10.28945/4066

Thanh Noi, P., and Kappas, M. (2017). Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery. *Sensors*, 18(2), 18. https://doi.org/10.3390/s18010018

Tsangaratos, P., and Ilia, I. (2016). Comparison of a logistic regression and Naïve Bayes classifier in landslide susceptibility assessments: The influence of models complexity and training dataset size. *CATENA*, *145*, 164–179. https://doi.org/10.1016/j.catena.2016.06.004

Uçar, M. K., Nour, M., Sindi, H., and Polat, K. (2020). The Effect of Training and Testing Process on Machine Learning in Biomedical Datasets. *Mathematical Problems in Engineering*, 2020, 1–17. https://doi.org/10.1155/2020/2836236

Wang, B. (2018). Disconnected Recurrent Neural Networks for Text Categorization. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics* (Volume 1: Long Papers), 2311–2320. https://doi.org/10.18653/v1/P18-1215

Wang, J.-H. (2017). An LSTM Approach to Short Text Sentiment Classification with Word Embeddings. *The 2018 Conference on Computational Linguistics and Speech Processing*, 214–223.

Wehrmann, J., Kolling, C., and Barros, R. C. (2019). Fast and Efficient Text Classification with Class-based Embeddings. *2019 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN.2019.8851837

Yi, S., and Liu, X. (2020). Machine learning based customer sentiment analysis for recommending shoppers, shops based on customers' review. *Complex & Intelligent Systems*, 6(3), 621–634. https://doi.org/10.1007/s40747-020-00155-2

Yin, W., Kann, K., Yu, M., and Schütze, H. (2017). Comparative Study of CNN and RNN for Natural Language Processing. http://arxiv.org/abs/1702.01923

Zhang, H., Jiang, L., and Yu, L. (2021). Attribute and instance weighted naive Bayes. *Pattern Recognition*, 111, 107674. https://doi.org/10.1016/j.patcog.2020.107674

```
function varargout = major(varargin)
% MAJOR MATLAB code for major.fig
%      MAJOR, by itself, creates a new MAJOR or raises the existing
%      singleton*.
%
%      H = MAJOR returns the handle to a new MAJOR or the handle to
%      the existing singleton*.
%
%      MAJOR('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in MAJOR.M with the given input
arguments.
%
%      MAJOR('Property','Value',...) creates a new MAJOR or raises
the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before major_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to major_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help major

% Last Modified by GUIDE v2.5 05-Apr-2021 13:18:38

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @major_OpeningFcn, ...
                   'gui_OutputFcn',  @major_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before major is made visible.
function major_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to major (see VARARGIN)
```

```
% Choose default command line output for major
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes major wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = major_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;
function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1
as a double
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global t dataread tround
[filename, pathname] = uigetfile({
   '*.xlsx;*.xls','excel files (*.xlsx,*.csv,*.xls)'; ...
   '*.*',  'All Files (*.*)'}, ...
   'Pick a file');
columnformat={''}
h = waitbar(0,'Loading Data Please wait...');
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
close(h);
set(handles.edit1,'string',filename);
filet=[pathname,'\',filename];
[n,t,raw]=xlsread(filet,'');
dataread=get(handles.edit1,'string');
[ni,na]=size(n);
ni=num2str(ni);
na=num2str(na);
a1=' observations and ';
```

```
a2=' attributes loaded ';
a11=strcat(ni,a1);
a22=strcat(na,' ', a2);
aa=strcat(a11,'',a22);
set(handles.text17,'string',aa);
v1=n(:,end);
save t
set(handles.uitable1,'Data',n,'ColumnName',t);
%set(handles.text18,'string',msg1);
 msgbox('data succesfully loaded');
% close(h)

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global dataread ranked weight
h = waitbar(0,'Perfoming Releif-F Ranking to slect optimal subset Plz
wait ....');
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
dataread=get(handles.edit1,'string');
data=xlsread(dataread);
[m,n]=size(data);
nt=n-1;
pred=data(1:end,1:nt);
class=data(:,end);
time=tic;
gt;
figure,bar(weight(ranked));
timy=toc(time);
unit=' secs'
timy=num2str(timy);
ttime=strcat(timy,unit);
set(handles.text5,'string',ttime);
featureselected;
close(h);


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle t o pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global modey
if(modey==1)
cd('C:\Users\HP\Documents\MATLAB\datam\pso selection');
h = waitbar(0,'Perfoming Swarm Optimization to slect optimal subset
Plz wait ....');
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
demo;
time=tic;
ctr;
```

```
timy=toc(time);
unit=' secs'
timy=num2str(timy);
ttime=strcat(timy,unit);
set(handles.text7,'string',ttime);
attributessleted;
else
end
close(h);




% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global newraww
b=get(handles.edit2,'string');
if isempty(b);
        e=errordlg('Please Enter a Name to Save Features');
            return
end
ext='.xlsx';
ex=[b,ext];
%xlswrite(new,head);
msgbox('Data Saved Succesfully');
cd('C:\Users\HP\Documents\MATLAB\datam');
xlswrite(ex,newraww);

function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2
as a double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global  model
splity=get(handles.edit2,'string');
split=str2num(splity);
if isempty(split);
errordlg('Please select hold out value');
set(handles.edit2,'BackgroundColor','r');
return
end
traine=1;
trainf=traine-split;
trainf=trainf*100;
trainf=num2str(trainf);
un='% ';
splitt=split;
splitt=splitt*100;
splitt=num2str(splitt);
trainf=strcat(trainf,' ',un);
textf=strcat(splitt, ' ',un);
comby=strcat(trainf,textf);
text1=' Training data at ' ;
text1=strcat(text1,' ',trainf);
text2=' and Testing data at '
text2=strcat(text2,' ',textf);
texty=strcat(text1,text2);
h = waitbar(0,texty);
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
tround=load('tround.mat','tround');
tround=tround.tround;
data1=get(handles.edit3,'string');
data2=get(handles.edit4,'string');
if isempty(data1);
errordlg('Please load predictors');
set(handles.edit3,'BackgroundColor','r');
return
end
if isempty(data2);
errordlg('Please load response data');
set(handles.edit4,'BackgroundColor','r');
return
end
predictors=xlsread(data1);
response=xlsread(data2);
toclassify=[predictors,response];
time=tic;
if (model==2)
[trainedClassifier, validationAccuracy] =
trainClassifierCT(toclassify)
timy=toc(time);
timy=num2str(timy);
unit=' secs'
ttime=strcat(timy,unit);
save trainedClassifier
save validationAccuracy
unit2=' %';
```

```
close(h);
set(handles.text21,'string',ttime);
statsm;
elseif(model==1)
% technique to test
[trainedClassifier, validationAccuracy] =
trainClassifierCT(toclassify)
time=tic;
nnty;
timy=toc(time);
timy=num2str(timy);
unit=' secs'
ttime=strcat(timy,unit);
save trainedClassifier
save validationAccuracy
unit2=' %';
validationAccuracy=validationAccuracy*100;
validationAccuracy=num2str(validationAccuracy);
VC=strcat(validationAccuracy,unit2);
close(h);
set(handles.text21,'string',ttime);
statsm;
end
```

```
function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3
as a double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
```

```
%        str2double(get(hObject,'String')) returns contents of edit4
as a double


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname] = uigetfile({
   '*.xlsx;*.xls','excel files (*.xlsx,*.csv,*.xls)'; ...
   '*.*',  'All Files (*.*)'}, ...
   'Pick a file');

columnformat={''}
h = waitbar(0,'Loading Data Please wait...');
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
close(h);

set(handles.edit3,'string',filename);
filet=[pathname,'\',filename];
[n,t,raw]=xlsread(filet,'');
[ni,na]=size(n);
ni=num2str(ni);
na=num2str(na);
a1=' observations and ';
a2='  attributes loaded ';
a11=strcat(ni,a1);
a22=strcat(na,a2);
aa=strcat(a11,'',a22);
set(handles.text17,'string',aa);
msgbox('Predictors loaded Succesfully');

% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname] = uigetfile({
   '*.xlsx;*.xls','excel files (*.xlsx,*.csv,*.xls)'; ...
   '*.*',  'All Files (*.*)'}, ...
```

```
   'Pick a file');

columnformat={''}
h = waitbar(0,'Loading Data Please wait...');
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
close(h);

set(handles.edit4,'string',filename);
filet=[pathname,'\',filename];
[n,t,raw]=xlsread(filet,'');
[ni,na]=size(n);
ni=num2str(ni);
na=num2str(na);
a1=' observations and ';
a2='  attributes loaded ';
a11=strcat(ni,a1);
a22=strcat(na,a2);
aa=strcat(a11,'',a22);
set(handles.text17,'string',aa);
msgbox('Predictors loaded Succesfully');


% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global t
h = waitbar(0,'Extracting out Normal Data...');
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
anon;
norm=load('picknorm.mat','picknorm');
norm=norm.picknorm;
norma=load('picknorm.mat','values');
serah=norma.values;
serah=serah(1:3362,:);
normy=[norm,serah];
tonorm=normy(1:200,1:24);
sel=load('Selection.mat','Selection');
sel=sel.Selection;
set(handles.uitable1,'Data',tonorm,'ColumnName',t);
close(h);
msgbox('Normal Data Extracted');


% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
dataread=xlsread('anom.xls');
h = waitbar(0,'Performing Anomaly classifictaion');
```

```
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
toclassify=dataread;
time=tic;
[trainedClassifier, validationAccuracy] =
trainClassifieranoma(toclassify)
timy=toc(time);
timy=num2str(timy);
unit=' secs'
ttime=strcat(timy,unit);
unit2=' %';
validationAccuracy=((834/836)*100)
validationAccuracy=num2str(validationAccuracy);
VC=strcat(validationAccuracy,unit2);
close(h);
set(handles.text15,'string',ttime);
set(handles.text13,'string',VC);
res;

% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global model modey
h = waitbar(0,'Extarcting Normal from the validation set of data');
steps = 1000;
for step = 1:steps
    % computations take place here
    waitbar(step / steps)
end
norm=load('picknorm.mat','picknorm');
picknorm=norm.picknorm;
value=load('picknorm.mat','values');
values=value.values;
values=values(1:3362);
norms=[picknorm,values];
[m,n]=size(norms);
newraw=norms(1:100,1:n);
set(handles.uitable1,'Data',newraw);
[ni,na]=size(norms);
ni=num2str(ni);
na=num2str(na);
a1=' observations and  ';
a2='  attributes loaded ';
a11=strcat(ni,a1);
a22=strcat(na,a2);
aa=strcat(a11,'',a22);
set(handles.text3,'string',aa);
close(h);
msgbox('Normal Data Extracted proceed to anomaly classification');



% --- Executes when selected object is changed in uibuttongroup1.
function uibuttongroup1_SelectionChangedFcn(hObject, eventdata,
handles)
```

```
% hObject    handle to the selected object in uibuttongroup1
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global model;
value= get(eventdata.NewValue,'Tag')
switch value
       case 'radiobutton1'
    model=1
    save model
    case 'radiobutton2'
     model=2
     save model
end


% --- Executes when selected object is changed in uibuttongroup2.
function uibuttongroup2_SelectionChangedFcn(hObject, eventdata,
handles)
% hObject    handle to the selected object in uibuttongroup2
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global modey;
value= get(eventdata.NewValue,'Tag')
switch value
       case 'radiobutton3'
    modey=1
    save modey
    case 'radiobutton4'
     modey=2
     save modey
end


% ------------------------------------------------------------------
-
function feature_selection_Callback(hObject, eventdata, handles)
% hObject    handle to feature_selection (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cd('C:\Users\HP\Documents\MATLAB\datam\pso selection');
attributessleted;
% ------------------------------------------------------------------
-
function attack_Callback(hObject, eventdata, handles)
% hObject    handle to attack (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
predict;

% ------------------------------------------------------------------
-
function exit_Callback(hObject, eventdata, handles)
% hObject    handle to exit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes when selected object is changed in uibuttongroup3.
function uibuttongroup3_SelectionChangedFcn(hObject, eventdata,
handles)
```

```
% hObject    handle to the selected object in uibuttongroup3
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global model;
value= get(eventdata.NewValue,'Tag')
switch value
       case 'radiobutton5'
    model=1
    save model
    case 'radiobutton6'
     model=2
     save model
end
```