VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belgaum – 590 018

A project report on

"A Study on Fake News Detection using Naïve Bayes, SVM, Neural Networks and LSTM"

Submitted in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE & ENGINEERING

by

Prannay S Reddy (1CR15CS114)
Diana Elizabeth Roy (1CR15CS056)
M Keerthana (1CR15CS088)
Manoj P (1CR15CS094)

Under the guidance of

Poonam V Tijare
Assistant Professor
Dept. of CSE, CMRIT, Bengaluru

CMR INSTITUTE OF TECHNOLOGY

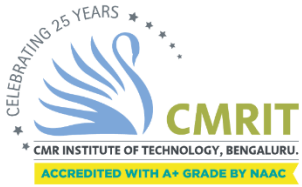
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

#132, AECS Layout, IT Park Road, Bengaluru-560037

2018-19

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belgaum – 590 018



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Certificate

This is to certify that the project entitled, **"A Study on Fake News Detection using Naïve Bayes, SVM, Neural Networks and LSTM"**, is a bonafide work carried out by **Prannay S Reddy (1CR15CS114)**, **Diana Elizabeth Roy (1CR15CS056)**, **M Keerthana (1CR15CS088), Manoj P (1CR15CS094)** in partial fulfillment of the award of the degree of Bachelor of Engineering in Computer Science & Engineering of Visvesvaraya Technological University, Belgaum, during the year 2018-19. It is certified that all corrections/suggestions indicated during reviews have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the said Degree.

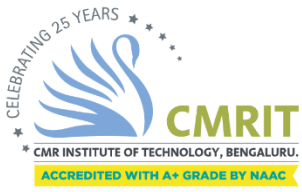
Name & Signature of Guide	Name & Signature of HOD	Signature of Principal
(Poonam V Tijare)	(Dr. Jhansi Rani P)	(Dr. Sanjay Jain)

External Viva

Name of the Examiners	Signature with date
1.	
2.	

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belgaum – 590 018



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Self Declaration

We, **Prannay S Reddy (1CR15CS114)**, **Diana Elizabeth Roy (1CR15CS056), M Keerthana (1CR15CS088)** and **Manoj P (1CR15CS094)** bonafide students of CMR Institute of Technology, Bangalore, hereby declare that the report entitled **"**A Study on Fake News Detection using Naïve Bayes, SVM, Neural Networks and LSTM**"** has been carried out by us under the guidance of **Poonam V Tijare, Assistant Professor,** CMRIT Bangalore, in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in **Computer Science Engineering**, of the Visvesvaraya Technological University, Belgaum during the academic year 2018-2019. The work done in this dissertation report is original and it has not been submitted for any other degree in any university.

Prannay S Reddy (1CR15CS114)
Diana Elizabeth Roy (1CR15CS056)
M Keerthana (1CR15CS088)
Manoj P (1CR15CS094)

ABSTARCT

The proliferation of misleading information in everyday access media outlets such as social media feeds, news blogs, and online newspapers have made it challenging to identify trustworthy news sources, thus increasing the need for computational tools able to provide insights into the reliability of online content. We focus on fake news detection exploring the applications of Natural Language Processing and Machine Learning. Learning techniques are used to identify fake news accurately. Pre-processing tools are used to clean the data and apply feature extraction on them. Then a fake news detection model is built using four different techniques, Naïve Bayes, SVM, Neural Networks and LSTM. We investigate and compare the accuracy of the techniques to find out the best fit technique for the model.

Acknowledgment

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of people who made it possible, success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance.

So with gratitude, we acknowledge all those whose guidance and encouragement served as a beacon of light and crowned our effort with success.

We would like to thank **Dr. Jhansi Rani P.**, Professor and HOD, Department of Computer Science who shared her opinion and experience through which we received the required information crucial for the project.

We consider it a privilege and honor to express my sincere gratitude to our guide **Poonam V Tijare .,** Assistant Professor, Department of Computer Science & Engineering, for his/her valuable guidance throughout the tenure of this review.

Finally, we would like to thank all our family members and friends whose encouragement and support was invaluable.

Prannay S Reddy(1CR15CS114)
Diana Elizabeth Roy(1CR15CS056)
M Keerthana(1CR15CS088)
Manoj P(1CR15CS094)

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

Page no.

Figures

Tables

CHAPTER 1

INTRODUCTION

Fake news is a phenomenon which is having a significant impact on our social life, in particular in the political world. Fake news detection is an emerging research area which is gaining interest but involved some challenges due to the limited amount of resources (i.e., datasets, published literature) available.

We propose in this paper, a fake news detection model that use n-gram analysis and machine learning techniques. We investigate and compare two different features extraction techniques and six different machine classification techniques. Experimental evaluation yields the best performance using Term Frequency-Inverted Document Frequency (TF-IDF) as feature extraction technique, and Linear Support Vector Machine (LSVM) as a classifier, with an accuracy of 90%.

In the recent years, online content has been playing a significant role in swaying users decisions and opinions. Opinions such as online reviews are the main source of information for e-commerce customers to help with gaining insight into the products they are planning to buy.

Recently it has become apparent that opinion spam does not only exist in product reviews and customers' feedback. In fact, fake news and misleading articles is another form of opinion spam, which has gained traction. Some of the biggest sources of spreading fake news or rumours are social media websites such as Google Plus, Facebook, Twitters, and other social media outlet.

Even though the problem of fake news is not a new issue, detecting fake news is believed to be a complex task given that humans tend to believe misleading information and the lack of control of the spread of fake content. Fake news has been getting more attention in the last couple of years, especially since the US election in 2016. It is tough for humans to detect fake news. It can be argued that the only way for a person to manually identify fake

news is to have a vast knowledge of the covered topic. Even with the knowledge, it is considerably hard to successfully identify if the information in the article is real or fake. The open nature of the web and social media in addition to the recent advance in computer science simplify the process of creating and spreading fake news. While it is easier to understand and trace the intention and the impact of fake reviews, the intention, and the impact of creating propaganda by spreading fake news cannot be measured or understood easily. For instance, it is clear that fake review affects the product owner, customer and online stores; on the other hand, it is not easy to identify the entities affected by the fake news. This is because identifying these entities require measuring the news propagation, which has shown to be complex and resource intensive. Trend Micro, a cyber security company, analysed hundreds of fake news services provider around the globe. They reported that it is effortless to purchase one of those services. In fact, according to the report, it is much cheaper for politicians and political parties to use those services to manipulate election outcomes and people opinions about certain topics. Detecting fake news is believed to be a complex task and much harder than detecting fake product reviews given that they spread easily using social media and word of mouth.

We present in this paper an n-gram features based approach to detect fake news, which consists of using text analysis based on n-gram features and machine learning classification techniques. We study and compare six different supervised classification techniques, namely, K-Nearest Neighbour (KNN), Support Vector Machine (SVM), Logistic Regression (LR), Linear Support Vector Machine (LSVM), Decision tree (DT) and Stochastic Gradient Descent (SGD). Experimental evaluation is conducted using a dataset compiled from real and fake news websites, yielding very encouraging results.

Research on fake news detection is still at an early stage, as this is a relatively recent phenomenon, at least regarding the interest raised by society. We review some of the published work in the following. In general, Fake news could be categorized into three groups. The first group is fake news, which is news that is completely fake and is made up by the writers of the articles. The second group is fake satire news, which is fake news whose main purpose is to provide humour to the readers. The third group is poorly written

news articles, which have some degree of real news, but they are not entirely accurate. In short, it is news that uses, for example, quotes from political figures to report a fully fake story. Usually, this kind of news is designed to promote certain agenda or biased opinion.

Rubin et al. discuss three types of fake news. Each is a representation of inaccurate or deceptive reporting. Furthermore, the authors weigh the different kinds of fake news and the pros and cons of using different text analytics and predictive modelling methods in detecting them. In this paper, they separated the fake news types into three groups:

- Serious fabrications are news not published in mainstream or participant media, yellow press or tabloids, which as such, will be harder to collect.
- Large-Scale hoaxes are creative and unique and often appear on multiple platforms. The authors argued that it may require methods beyond text analytics to detect this type of fake news.
- Humorous fake news, are intended by their writers to be entertaining, mocking, and even absurd. According to the authors, the nature of the style of this type of fake news could have an adverse effect on the effectiveness of text classification techniques.

The authors argued that the latest advance in natural language processing (NLP) and deception detection could be helpful in detecting deceptive news. However, the lack of available corpora for predictive modelling is an important limiting factor in designing effective models to detect fake news.

Horne et al. illustrated how obvious it is to distinguish between fake and honest articles. According to their observations, fake news titles have fewer stop-words and nouns, while having more nouns and verbs. They extracted different features grouped into three categories as follows:

- Complexity features calculate the complexity and readability of the text.

- Psychology features illustrate and measure the cognitive process and personal concerns underlying the writings, such as the number of emotion words and casual words.
- Stylistic features reflect the style of the writers and syntax of the text, such as the number of verbs and the number of nouns.

The aforementioned features were used to build an SVM classification model. The authors used a dataset consisting of real news from BuzzFeed and other news websites, and Burfoot and Baldwin's satire dataset to test their model. When they compared real news against satire articles (humorous article), they achieved 91% accuracy. However, the accuracy dropped to 71% when predicting fake news against real news.

Wang et al. introduced LIAR, a new dataset that can be used for automatic fake news detection. Thought LIAR is considerably bigger in size, unlike other data sets, this data set does not contain full articles, it contains 12800 manually labelled short statements from politicalFact.com.

Rubin et al. proposed a model to identify satire and humour news articles. They examined and inspected 360 Satirical news articles in mainly four domains, namely, civics, science, business, and what they called "soft news" ('entertainment/gossip articles'). They proposed an SVM classification model using mainly five features developed based on their analysis of the satirical news. The five features are Absurdity, Humour, Grammar, Negative Affect, and Punctuation. Their highest precision of 90% was achieved using only three combinations of features which are Absurdity, Grammar, and Punctuation.

CHAPTER 2

LITERATURE SURVEY

A literature survey depicts the various analyses and research made in the field of interest of the project and the results already published. It is an important part as it gives a direction in the area of your research. It helps to set a goal for your analysis- thus deriving at the problem statement. After taking into account various parameters and the extent of the project, the following papers were analyzed:

1. In the paper by Shu A et al., a detailed study was done on identifying fake news on social media, including fake news characterizations on psychology and social theories, existing algorithms from a data mining perspective, evaluation metrics and representative datasets. They also discuss related research areas, open problems, and future research directions for fake news detection on social media. [1]
2. The paper is written by Rubin et al. deals with the domain of fake news which is composed of satirical news. Satire news intentionally provides hints revealing its own deception. While fake news wants the readers to believe a false fact, satire news must eventually be understood as a jest. This paper provides an in-depth view of the features of humor and satire news along with the style of the authors reporting. The paper has considered the news articles from twelve contemporary news topics in four different domains which are civics, science, business, and soft news. The paper proposes a Support Vector Machine based algorithm which can detect satire news based on features like Absurdity, Humor, Grammar, and Punctuation. The models achieved an accuracy of 90% and a recall of 84%. The aim is to reduce the negative impact of satire news on readers. [2]

3. In the paper by Kelly Stahl et al., they have considered past and current techniques for fake news identification in text formats while elucidating how and why fake news exists in any case. This paper incorporates a dialog on Linguistic Cue and

Network Analysis. It also approaches and proposes a three-section strategy utilizing Naïve Bayes Classifier, Support Vector Machines, and Semantic Analysis as an apt method to distinguish fake news via social media. [3]

4. In this paper by Marco L. Delia Vedov et al., detection is based on social context models, such as mapping the news' diffusion pattern. They first propose a novel ML fake news detection method which, by combining news content and social context features, increasing their already high accuracy by up to 4.8%. They have implemented their method within a Facebook Messenger chatbot and validated it with a real-world application, obtaining high accuracy. They presented a novel automatic fake news detection method that combines social and content signals. They built on the work by E. Tacchini, G. Ballarin, M. L. Della Vedova, S. Moret, and L. de Alfaro. Their hypothesis is: although social-based methods offer good performances, these performances get worse for news items that collect only a few social interactions, and therefore the combination with content-based approaches can increase the overall detection accuracy. They have combined content-based and social-based methods based on a threshold rule which, despite its simplicity, is able to capture well the different contributions of these two approaches and to outperform other more sophisticated - methods presented in the literature. [4]

5. In the research paper written by Mykhailo Granik et al., they have explored a simple approach to detect fake news using Naïve Bayes classifier. Their approach was conducted on Facebook news posts rather than news articles on the internet. The paper achieved an accuracy of 74% in the classification process. It also went on to state that simple machine learning models like the Naïve Bayes can achieve a moderate accuracy and in future, more artificial intelligence techniques could be used to tackle the menace of fake news. [5]

6. The paper by Namwon Kim et al. detects fake news based on unified key sentence information. The model extracts the key sentences of the article to the question

from the article and then later combining the word vector for each key sentence. They make use of a Korean dataset for their analysis. It performs an efficient matching operation for the word vectors obtained using bidirectional LSTM. They have achieved an accuracy ranging from 64 to 69%. Their future plan is to develop a more advanced model which applies the model independently to each key sentence. [6]

7. In this paper by Hadeer Ahmed et al., they have presented a detection model for fake news using n-gram analysis through the lenses of different features extraction techniques. Furthermore, they investigated two different feature extraction techniques and six different machine learning techniques. The proposed model achieves its highest accuracy when using unigram features and Linear SVM classifier. [7]

8. In this paper by Saranya Krishnan et al., they have proposed a generalized framework to predict tweet credibility. First, they have extracted the essential features and user features through the Twitter API. If a tweet has an image or image URL, the reverse image search is performed to check whether the same image has been tagged with different information in the past. In addition, if any URL is present in any tweet, it will be cross-checked against the fake news sources to see whether it is a part of a fake news websites dataset. All these features are then used by the data mining algorithms to classify tweets as fake or real. [8]

9. In this paper by Stefan Helmstetter et al., they discuss a weakly supervised approach, which automatically collects a large- scale, but very noisy training dataset comprising hundreds of thousands of tweets. During collection, they automatically labeled tweets by their source, i.e., trustworthy or untrustworthy source, and train a classifier on this dataset. They then used a classifier for a different classification target, i.e., the classification of fake and non-fake tweets. Although the labels are not accurate according to the new classification target (not

all tweets by an untrustworthy source need to be fake news, and vice versa), they show that despite this unclean inaccurate dataset, they have detected fake news with high accuracy. [9]

10. In this paper by Akshay Jain, they propose a method for "fake news" detection and ways to apply it on Facebook, one of the most popular online social media platforms. This method uses Naive Bayes classification model to predict whether a post on Facebook will be labeled as REAL or FAKE. They have suggested many techniques in the paper for improving the accuracy of classification. [10]

CHAPTER 3

THEORETICAL BACKGROUND

The theoretical concepts and algorithms used in the development of the proposed system are explained in the subsequent sections.

3.1 NATURAL LANGUAGE PROCESSING

Natural language processing (NLP) is an area of computer science and artificial intelligence that is concerned with the interaction between computers and humans in natural language. The ultimate goal of NLP is to enable computers to understand language as well as we do. It is the driving force behind things like virtual assistants, speech recognition, sentiment analysis, automatic text summarization, machine translation and much more.

3.1.1 Introduction

Natural Language Processing (NLP) is the intersection of Computer Science, Linguistics and Machine Learning that is concerned with the communication between computers and humans in natural language. NLP is all about enabling computers to understand and generate human language. Applications of NLP techniques are Voice Assistants like Alexa and Siri but also things like Machine Translation and text-filtering. NLP is one of the fields that heavily benefited from the recent advances in Machine Learning, especially from Deep Learning techniques. The field is divided into the three following parts:

Speech Recognition—The translation of spoken language into text.

Natural Language Understanding—The computers ability to understand what we say.

Natural Language Generation—The generation of natural language by a computer.

Human language is special for several reasons. It is specifically constructed to convey the speaker/writers meaning. It is a complex system, although little children can learn it pretty quickly. Another remarkable thing about human language is that it is all about symbols.

According to Chris Manning (Machine Learning Professor at Stanford University), it is a discrete, symbolic, categorical signalling system. This means that you can convey the same meaning by using different ways, like speech, gesture, signs etc. The encoding of these by the human brain is a continuous pattern of activation, where the symbols are transmitted via continuous signals of sound and vision. Understanding human language is considered a difficult task due to its complexity. For example, there is an infinite number of different ways to arrange words in a sentence. Also, words can have several meanings and contextual information is necessary to correctly interpret sentences. Every Language is more or less unique and ambiguous. Just take a look at the following newspaper headline „The Pope's baby steps on gays". This sentence clearly has two very different interpretations, which is a pretty good example of the challenges in NLP.

3.1.2 Syntactic & Semantic Analysis

Syntactic Analysis (Syntax) and Semantic Analysis (Semantic) are the two main techniques that lead to the understanding of natural language. Language is a set of valid sentences, but what makes a sentence valid? Actually, you can break validity down into two things: Syntax and Semantics. The term Syntax refers to the grammatical structure of the text whereas the term Semantics refers to the meaning that is conveyed by it. However, a sentence that is syntactically correct, does not have to be semantically correct. Just take a look at the following example. The sentence "cows flow supremely" is grammatically valid (subject—verb—adverb) but does not make any sense.

Syntactic Analysis:

Therefore, Semantic Analysis is the process of understanding the meaning and interpretation of words, signs, and sentence structure. This enables computers partly to understand natural language the way humans do, involving meaning and context. I say partly because Semantic Analysis is one of the toughest parts of NLP and not fully solved yet. For example, Speech Recognition has become very good and works almost flawlessly but we are still lacking this kind of proficiency in Natural Language Understanding (e.g Semantic). Your phone basically understands what you have said but often can't do anything with it because it doesn't understand the meaning behind it. Also, note that some of the technologies out there only make you think they understand the meaning of a text.

An approach based on keywords or statistics or even pure machine learning may be using a matching or frequency technique for clues as to what a text is "about." These methods are limited because they are not looking at the real underlying meaning

3.1.3 Components of NLP

Five main Component of Natural Language processing as shown in figure 3.1.3 are:

1. Morphological and Lexical Analysis
2. Syntactic Analysis
3. Semantic Analysis
4. Discourse Integration
5. Pragmatic Analysis

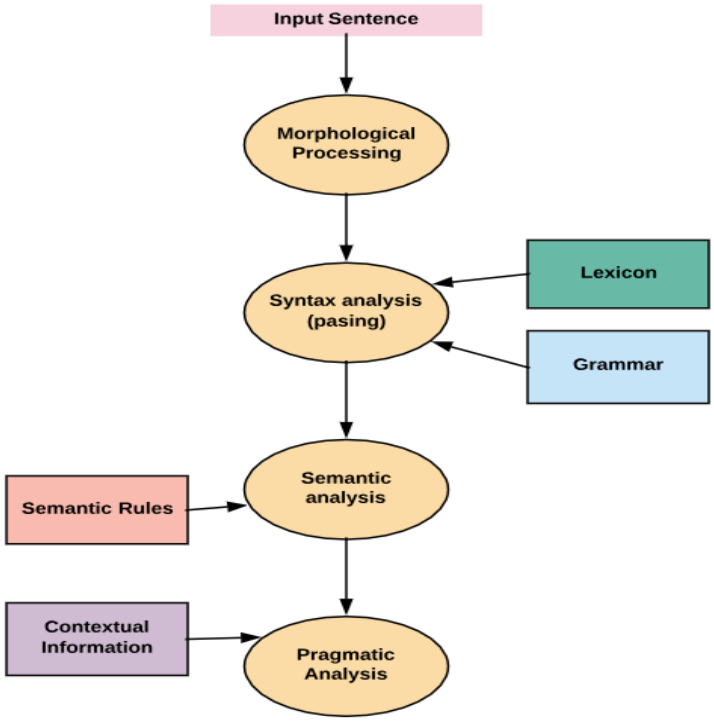


Fig 3.1.3: Components of NLP

1. Morphological and Lexical Analysis

Lexical analysis is a vocabulary that includes its words and expressions. It depicts analysing, identifying and description of the structure of words. It includes dividing a text into paragraphs, words and the sentences

Individual words are analysed into their components, and nonword tokens such as punctuations are separated from the words.

2. Semantic Analysis

Semantic Analysis is a structure created by the syntactic analyser which assigns meanings. This component transfers linear sequences of words into structures. It shows how the words are associated with each other.

Semantics focuses only on the literal meaning of words, phrases, and sentences. This only abstracts the dictionary meaning or the real meaning from the given context. The structures assigned by the syntactic analyser always have assigned meaning

E.g.. "colourless green idea." This would be rejected by the Symantec analysis as colourless Here; green doesn't make any sense.

3. Pragmatic Analysis

Pragmatic Analysis deals with the overall communicative and social content and its effect on interpretation. It means abstracting or deriving the meaningful use of language in situations. In this analysis, the main focus always on what was said in reinterpreted on what is meant.

Pragmatic analysis helps users to discover this intended effect by applying a set of rules that characterize cooperative dialogues.

E.g., “close the window?” should be interpreted as a request instead of an order.

4. Syntax analysis

The words are commonly accepted as being the smallest units of syntax. The syntax refers to the principles and rules that govern the sentence structure of any individual languages.

Syntax focus about the proper ordering of words which can affect its meaning. This involves analysis of the words in a sentence by following the grammatical structure of the sentence. The words are transformed into the structure to show hows the word are related to each other.

5. Discourse Integration

It means a sense of the context. The meaning of any single sentence which depends upon that sentences. It also considers the meaning of the following sentence.

For example, the word "that" in the sentence "He wanted that" depends upon the prior discourse context.

3.1.4 NLP and writing systems

The kind of writing system used for a language is one of the deciding factors in determining the best approach for text pre-processing. Writing systems can be

1. Logographic: A Large number of individual symbols represent words. Example Japanese, Mandarin
2. Syllabic: Individual symbols represent syllables
3. Alphabetic: Individual symbols represent sound

Majority of the writing systems use the Syllabic or Alphabetic system. Even English, with its relatively simple writing system based on the Roman alphabet, utilizes logographic symbols which include Arabic numerals, Currency symbols (S, £), and other special symbols.

This pose following challenges

- Extracting meaning(semantics) from a text is a challenge
- NLP is dependent on the quality of the corpus. If the domain is vast, it's difficult to understand context.
- There is a dependence on the character set and language

3.1.5 How to implement NLP

Below, given are popular methods used for Natural Learning Process:

Machine learning: The learning nlp procedures used during machine learning. It automatically focuses on the most common cases. So when we write rules by hand, it is often not correct at all concerned about human errors.

Statistical inference: NLP can make use of statistical inference algorithms. It helps you to produce models that are robust. e.g., containing words or structures which are known to everyone.

3.1.6 NLP Examples

Today, Natural process learning technology is widely used technology.

Here, are common Application' of NLP:

Information retrieval & Web Search

Google, Yahoo, Bing, and other search engines base their machine translation technology on NLP deep learning models. It allows algorithms to read text on a webpage, interpret its meaning and translate it to another language.

Grammar Correction:

NLP technique is widely used by word processor software like MS-word for spelling correction & grammar check.

Question Answering

Type in keywords to ask Questions in Natural Language.

Text Summarization

The process of summarising important information from a source to produce a shortened version

Machine Translation

Use of computer applications to translate text or speech from one natural language to another.

Sentiment analysis

NLP helps companies to analyze a large number of reviews on a product. It also allows their customers to give a review of the particular product.

3.1.7 Future of NLP

- Human readable natural language processing is the biggest Al- problem. It is all most same as solving the central artificial intelligence problem and making computers as intelligent as people.
- Future computers or machines with the help of NLP will able to learn from the information online and apply that in the real world, however, lots of work need to on this regard.
- Natural language toolkit or nltk become more effective
- Combined with natural language generation, computers will become more capable of receiving and giving useful and resourceful information or data.

3.1.8 Advantages of NLP

- Users can ask questions about any subject and get a direct response within seconds.
- NLP system provides answers to the questions in natural language
- NLP system offers exact answers to the questions, no unnecessary or unwanted information
- The accuracy of the answers increases with the amount of relevant information provided in the question.
- NLP process helps computers communicate with humans in their language and scales other language-related tasks
- Allows you to perform more language-based data compares to a human being without fatigue and in an unbiased and consistent way.
- Structuring a highly unstructured data source

3.1.9 Disadvantages of NLP

- Complex Query Language- the system may not be able to provide the correct answer it the question that is poorly worded or ambiguous.
- The system is built for a single and specific task only; it is unable to adapt to new domains and problems because of limited functions.
- NLP system doesn't have a user interface which lacks features that allow users to further interact with the system

3.2 MACHINE LEARNING

The term Machine Learning was coined by Arthur Samuel in 1959, an American pioneer in the field of computer gaming and artificial intelligence and stated that "it gives computers the ability to learn without being explicitly programmed". And in 1997, Tom Mitchell gave a "well-posed" mathematical and relational definition that "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

3.2.1 Classification of Machine Learning

Machine learning implementations are classified into three major categories, depending on the nature of the learning "signal" or "response" available to a learning system which are as follows:-

1. **Supervised learning :** When an algorithm learns from example data and associated target responses that can consist of numeric values or string labels, such as classes or tags, in order to later predict the correct response when posed with new examples comes under the category of Supervised learning. This approach is indeed similar to human learning under the supervision of a teacher.

The teacher provides good examples for the student to memorize, and the student then derives general rules from these specific examples.

2. **Unsupervised learning :**Whereas when an algorithm learns from plain examples without any associated response, leaving to the algorithm to determine the data patterns on its own. This type of algorithm tends to restructure the data into something else, such as new features that may represent a class or a new series of un-correlated values. They are quite useful in providing humans with insights into the meaning of data and new useful inputs to supervised machine learning algorithms.
   As a kind of learning, it resembles the methods humans use to figure out that certain objects or events are from the same class, such as by observing the degree of similarity between objects. Some recommendation systems that you find on the web in the form of marketing automation are based on this type of learning.
3. **Reinforcement learning :** When you present the algorithm with examples that lack labels, as in unsupervised learning. However, you can accompany an example with positive or negative feedback according to the solution the algorithm proposes comes under the category of Reinforcement learning, which is connected to applications for which the algorithm must make decisions (so the product is prescriptive, not just descriptive, as in unsupervised learning), and the decisions bear consequences. In the human world, it is just like learning by trial and error.
4. Errors help you learn because they have a penalty added (cost, loss of time, regret, pain, and so on), teaching you that a certain course of action is less likely to succeed than others. An interesting example of reinforcement learning occurs when computers learn to play video games by themselves.
   In this case, an application presents the algorithm with examples of specific situations, such as having the gamer stuck in a maze while avoiding an enemy. The application lets the algorithm know the outcome of actions it takes, and learning occurs while trying to avoid what it discovers to be dangerous and to pursue survival. You can have a look at how the company Google DeepMind

has created a reinforcement learning program that plays old Atari's videogames. When watching the video, notice how the program is initially clumsy and unskilled but steadily improves with training until it becomes a champion.

5. **Semi-supervised learning :** where an incomplete training signal is given: a training set with some (often many) of the target outputs missing. There is a special case of this principle known as Transduction where the entire set of problem instances is known at learning time, except that part of the targets are missing.

3.2.2 Categorizing on the basis of required Output

Another categorization of machine learning tasks arises when one considers the desired output of a machine-learned system:

1. **Classification :** When inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes. This is typically tackled in a supervised way. Spam filtering is an example of classification, where the inputs are email (or other) messages and the classes are "spam" and "not spam".
2. **Regression :** Which is also a supervised problem, A case when the outputs are continuous rather than discrete.
3. **Clustering :** When a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically an unsupervised task.

3.2.3 Classification vs Regression

Classification is the process of finding or discovering a model or function which helps in separating the data into multiple categorical classes i.e. discrete values. In classification, data is categorized under different labels according to some parameters given in input and then the labels are predicted for the data. The derived mapping function could be

demonstrated in the form of "IF-THEN" rules. The classification process deal with the problems where the data can be divided into binary or multiple discrete labels.

Let's take an example, suppose we want to predict the possibility of the wining of match by Team A on the basis of some parameters recorded earlier. Then there would be two labels Yes and No.

Figure 3.2.3.1 indicates the comparison of both binary classification and multi-class classification.

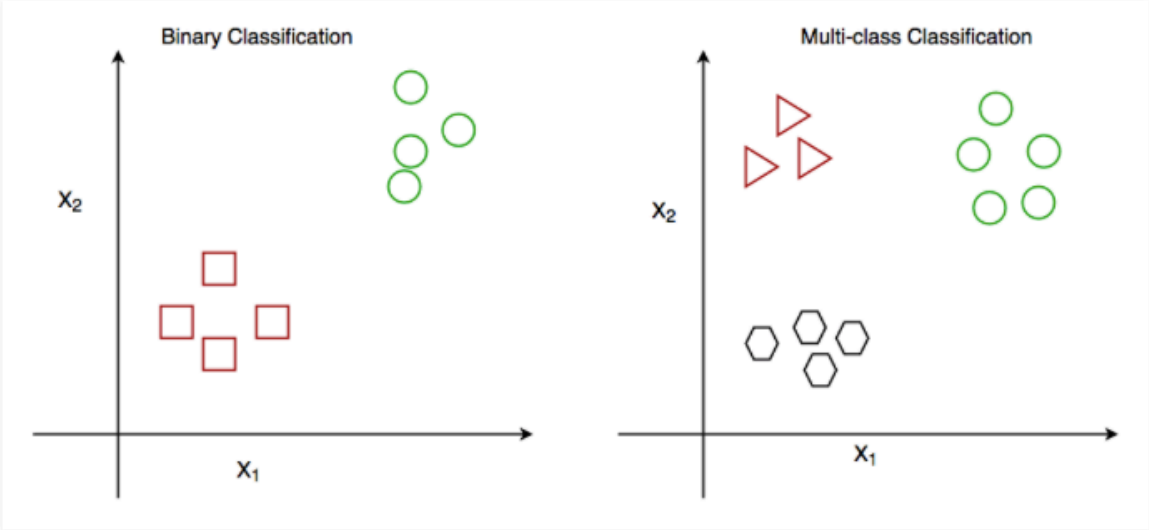


Fig 3.2.3.1: Binary Classification and Multiclass Classification [20]

Regression is the process of finding a model or function for distinguishing the data into continuous real values instead of using classes or discrete values. It can also identify the distribution movement depending on the historical data. Because a regression predictive model predicts a quantity, therefore, the skill of the model must be reported as an error in those predictions

Let's take a similar example as given in figure 3.2.3.2 of regression also, where we are finding the possibility of rain in some particular regions with the help of some parameters recorded earlier. Then there is a probability associated with the rain.

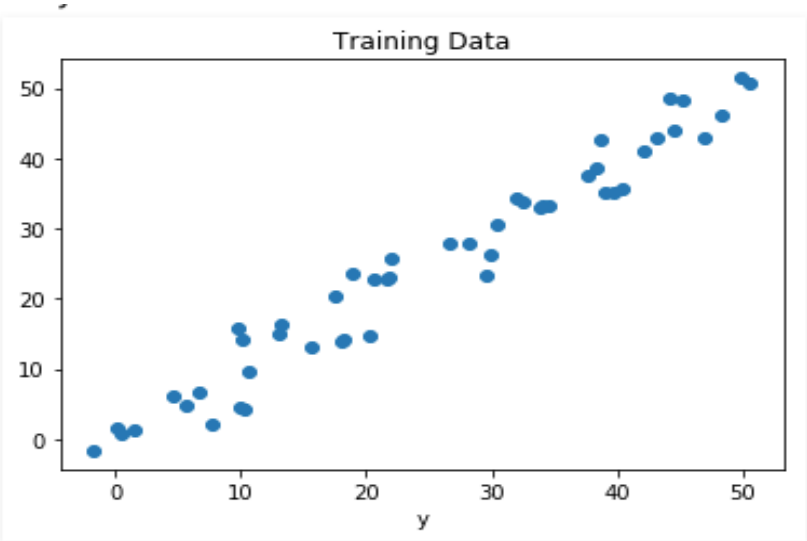


Fig 3.2.3.2: Regression of Day vs Rainfall (in mm) [21]

3.3 Naïve Bayes

Naïve Bayes is derived from Bayes Theorem, which is used for calculating conditional probability, the "probability that something will happen, given that something else has already occurred". Thus, it is possible to compute the likelihood of a certain outcome by using past knowledge of it . Furthermore, as shown in figure 3.3, Naïve Bayes is a type of classifier considered to be a supervised learning algorithm, which belongs to the Machine Language class and works by predicting "membership probabilities" for each individual class, for instance, the likelihood that the given evidence, or record, belongs to a certain class . The class with the greatest, or highest probability, shall be determined the "most likely class," which is also known as Maximum A Posteriori (MAP). The Naive Bayes Rule is based on the Bayes' theorem:

$$P(c|x) = \frac{P(x|C)P(c)}{P(x)} \qquad (1)$$

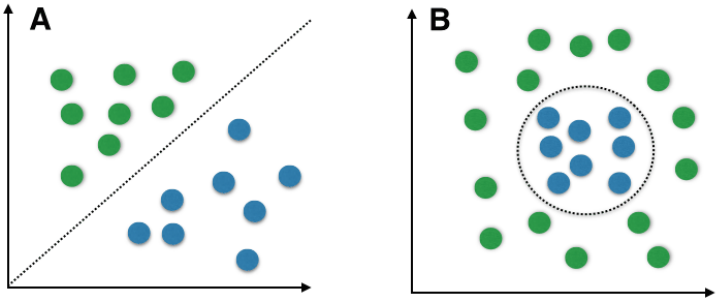


Fig 3.3: Naïve Bayes [22]

3.4 Support Vector Machine:

A support vector machine (SVM), which can be used interchangeably with a support vector network (SVN), is also considered to be a supervised learning algorithm. As shown in figure 3.4, SVMs work by being trained with specific data already organized into two different categories. Hence, the model is constructed after it has already been trained. Furthermore, the goal of the SVM method is to distinguish which category any new data falls under, in addition, it must also maximize the margin between the two classes [13]. The optimal goal is that the SVM will find a hyperplane that divides the dataset into two groups. The Radial Basis Function kernel is implemented in the project. The reason this kernel is used is because two Doc2Vec feature vectors will be close to each other if their corresponding documents are similar, so the distance computed by the kernel function should still represent the original distance. The Radial Basis function on two samples x *and* x' ' is given by:

$$K(x, x') = \exp\left(-\frac{||x-x'||^2}{2\sigma^2}\right) \qquad (2)$$

Where $||x - x'||^2$ represents the squared Euclidean distance and σ is a free parameter.

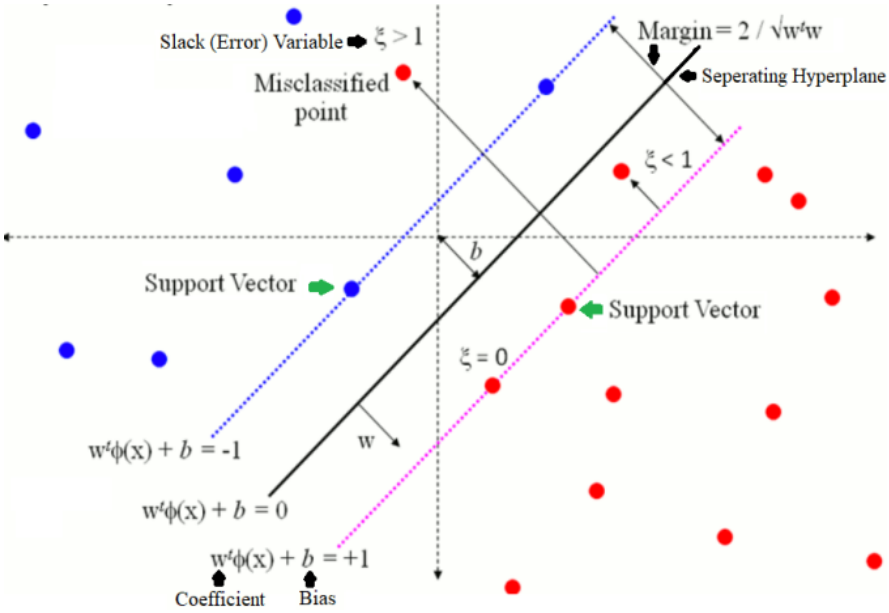


Fig 3.4: Support vector machine [23]

3.5 Long Short-Term Memory:

As shown in figure 3.5, the Long Short-Term Memory (LSTM) networks are the extension for the recurrent neural networks, that basically extends their memory. The building units for the layers of an RNN are the units of an LSTM, which is then often called an LSTM network. To remember LSTM's inputs over a long period of time, the LSTM's enable RNN's. Since the information is present in the memory for LSTM's, that is an analogy to the memory of the computer since even the LSTM can read, write and delete information from its memory. The LSTM is still a neural network. But different from the fully connected neural network, it has cycled in the neuron connections. Here, i_t , f_t , and O_t represent the equations of the input, forget, and output gates respectively. The weights are represented by w and σ is the sigmoid function :

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \quad (3)$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f) \quad (4)$$

$$O_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \quad (5)$$

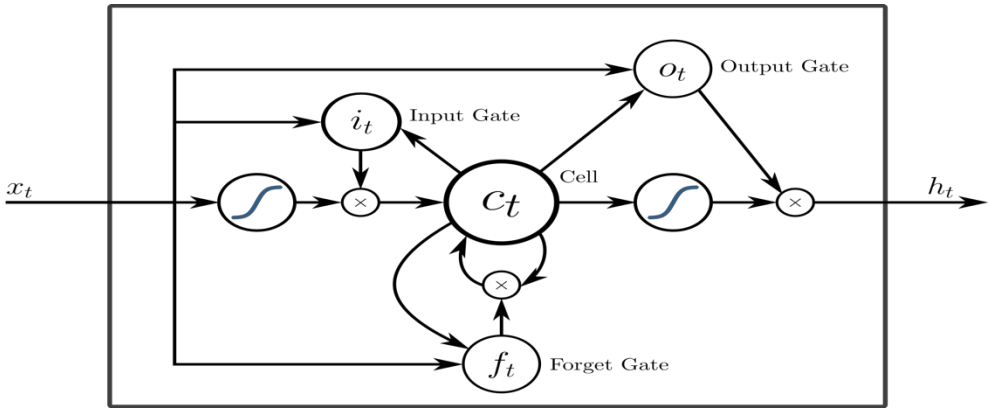


Fig 3.5: LSTM [19]

3.6 Neural Network:

Artificial Neural Networks in which the association between the units do not form a cycle are called Feedforward neural networks. Feedforward neural networks were the first type of artificial neural network invented and are simpler than recurrent neural networks. These networks are named so because the information moves linearly in the network through the input layer, then the hidden layer and ultimately through the output layer. The paper implements one feed-forward neural network models using Keras. The papers neural network implementation uses three hidden layers. In the Keras implementation layers of size 256, 256, and 80 are used, interspersed with dropout layers to avoid overfitting. For the activation function, the Rectified Linear Unit (ReLU)is chosen, which has been found to perform well in NLP applications [15]. In the below equations, $f(x)$ and $f'(x)$ represent the equation and the derivative of the Rectified Linear Unit (ReLU).

$$f(x) = \begin{cases} 0 \ for \ x < 0 \\ x \ for \ x \geq 0 \end{cases} \quad (6)$$

$$f'(x) = \begin{cases} 0 \ for \ x < 0 \\ 1 \ for \ x \geq 0 \end{cases} \quad (7)$$

3.7 CLASSIFICATION PROCESS

Fig 3.7 is a schematic representation of the classification process. It starts with preprocessing the data set, by removing unnecessary characters and words from the data. The Doc2Vec process is used, and a vector is formed representing the documents involved.

The last step in the classification process is to train the classifier. The paper investigates different classifiers to predict the class of the documents. The data is partitioned into train and test. The testing data has 26000 corpus of news articles and the training data has 5000 corpus of news articles.

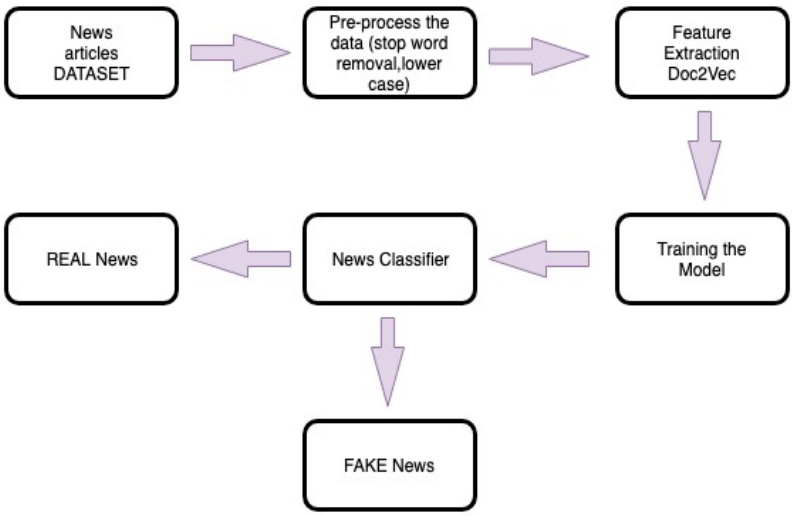


Fig 3.7: Classification Process

CHAPTER 4

SYSTEM REQUIREMENT SPECIFICATION

A **software requirements specification (SRS)** is a detailed description of the software system to be developed. It portrays the **functional** and **non-functional** requirements and may also include a set of use cases that describe user interactions that the software must provide.

Software Requirement Specification is a fundamental part of the document as it is the root foundation of the software development process. At the basic level, it is an organization's understanding of a customer or potential client's expectations, requirements and dependencies at a particular point of time, before any design or development work.

4.1 Functional Requirements

A function is defined as a set of inputs, behavior and the outputs. A functional requirement can be defined as a specification which defines the function of the concerned system. This may include specific functionality such as technical details, calculations, data manipulation and processing etc. The functional requirement documents the operations and the activities that a system should be able to perform.

The system design contains details about implementing the functional requirement. Below listed are the functional requirements in this system:

- Data collection - The data set used in this project is UCI known as UCI SMS spam and the dataset includes 5575 SMS classified into spam and ham. The corpus has been collected from free or free for research sources at the Web.
- Data Preprocessing - The purpose of preprocessing is to convert raw data into a form that fits machine learning. Structured and clean data allows a data scientist to get more precise results from an applied machine learning model. The technique includes data formatting, cleaning, and sampling.
- Dataset splitting -A dataset used for machine learning should be partitioned into three subsets — training, test, and validation sets.

- Model training -After a data scientist has preprocessed the collected data and split it into train and test can proceed with model training. This process starts with "feeding" the algorithm with training data. An algorithm will process data and output a model that is able to find a target value (attribute) in new data an answer you want to get with predictive analysis. The purpose of model training is to develop a model.
- Model evaluation and testing –The aim of this step is to develop a simple model that would be able to formulate a target value fast and well enough. A data scientist can achieve this goal through model tuning. That's the optimization of model parameters to achieve an algorithm's best performance.

4.2 Non-functional Requirements

While functional requirements specify the behavior of the system and tell us what the system is supposed to do, the non-functional requirements are mostly concerned with how the system is supposed to be. The system architecture contains details about implementing these non-functional requirements.

The non-functional requirements specify the criteria that are used to evaluate the operation of a system, instead of a specific behavior. These requirements are also often termed as "quality attributes", "constraints" and "non-behavioral requirements".

The following is a list of non-functional requirements.

- Response Time
- Availability
- Stability
- Maintainability
- Usability

4.3 Hardware Requirements

The hardware requirements include the requirements specification of the physical computer resources for a system to work efficiently. The hardware requirements may serve as the

basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system.

- Processor : Any Processor above 500 MHz
- Ram : 4 GB
- Hard Disk : 4 GB
- Input device : Standard Keyboard and Mouse.
- Output device : VGA and High Resolution Monitor.

4.4 Software Requirements

Operating System : Windows 7 or higher

Programming : Python 3.6 and related libraries – Keras, Numpy, Gensim

Software Description:

Python

Python is an interpreted high-level programming language for all-purpose programming. Created by Guido Van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that alter clear programming on each tiny and huge scales.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

4.5 System Analysis

System Analysis is the method of understanding an existing problem, defining requirements and evaluating the best possible solution. It enables thinking about the organization and its concerned problems as well as the technologies which can help in coming up with solutions.

System Analysis can be seen as a great problem solving technique that breaks down the system into component modules and evaluates each module to determine how efficiently they work individually and how well they interact with each other, in order to accomplish a specific task. The feasibility study plays a very important role in system analysis as it provides the target for design and development.

4.5.1 Feasibility Study:

Feasibility study can be understood as the assessment of the practicality of the proposed system or application. The feasibility study is also the key to evaluate the potential of a project's success. There are two most important to measure the feasibility which includes the total cost involved and the value to be attained on completion. Usually, a feasibility study always precedes the technical development and implementation.

Risk analysis is very closely related to feasibility study. If there is a greater project risk, the feasibility of creating quality software is thus reduced. When we talk about feasibility study at depth, there are three primary areas of interest in this context:

1. Performance Analysis
2. Technical Analysis
3. Economic Analysis

All the three types of feasibility tests are useful in determining the overall practicality or feasibility of the proposed system. These tests enable to evaluate whether the considered solution to satisfy the requirements is practical and workable in the software developed. Thus, it is important to perform these tests individually.

4.5.1.1 Performance Analysis:

The software system is made to run in suitable environment. The most important aim of

performance analysis is to provide the end user with the required functionality of the system. The user must obtain the desirable results in an accurate and time-efficient manner. The process of performance definition should be carried out in parallel with performance analysis to set the milestone for efficient performance. The goal of development process to achieve requirements satisfaction is to carry out performance analysis as early as possible. The aim of the present system is to predict the unwanted messages in a more efficient and smarter way.

The benchmark of performance for our system can be set by the following performance definition:

- ✓ The user should obtain relevant results as per the situation demand.
- ✓ The user should get accurate prediction of the SMS messages.
- ✓ The user should get these results in a limited time period, so there is enough time for conducting analysis.
- ✓ The output of the system should give the user a clear idea of whether the message is spam or ham.
- ✓ The system should benefit the user to make consequent important decisions regarding the messages.

4.5.1.2 Technical Analysis:

This feasibility test evaluates whether or not the proposed system will work and give results when it is fully developed and installed. It also analyses the presence of any obstacles in the implementation of the software application. After all a system can be termed as efficient only if it can overcome all the technical requirements of the user.

The technical modules and software required to implement this project are freely available. The project is implemented using Python and other machine learning algorithms. Since the resources are available, it can be concluded that the system is technically feasible.

4.5.1.3 Economic Analysis:

The economic analysis is performed to estimate the development cost weighed against the benefits which will be derived from the developed system.

For this project, only a computer system is required. The features desired from the system

can be uncovered by any operating system. For running this system, a Mac/Ubuntu/Windows operating system installed on our machine is used. Thus it can be concluded that the system is economically feasible.

CHAPTER 5

SYSTEM DESIGN

System design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specific requirements. Systems design is the application of systems theory to product development.

5.1 Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

In the diagram, classes are represented with boxes that contain three compartments:

- The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
- The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.
- The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

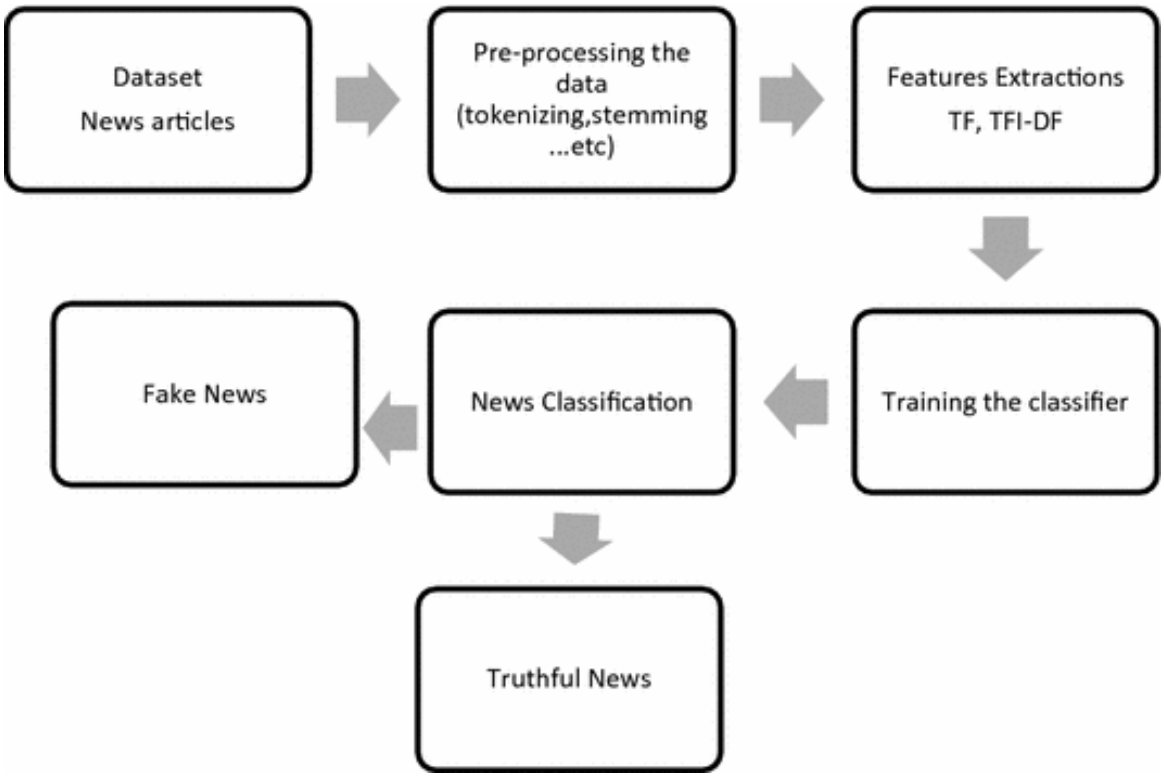


Fig 5.1: Class diagram of Fake News Detection Model [18]

The class diagram in figure 5.1, explains about the properties and functions of each class. The classes are dataset news articles , pre-processing the data , feature extractions , news classification and truthful news. In the above diagram, every class is represented with attributes and operations.

5.2 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. A use-case diagram can help provide a higher-level view of the system.

They provide the simplified and graphical representation of what the system must actually do. Due to their simplistic nature, use case diagrams can be a good communication tool for

stakeholders. The drawings attempt to mimic the real world and provide a view for the stakeholder to understand how the system is going to be designed.

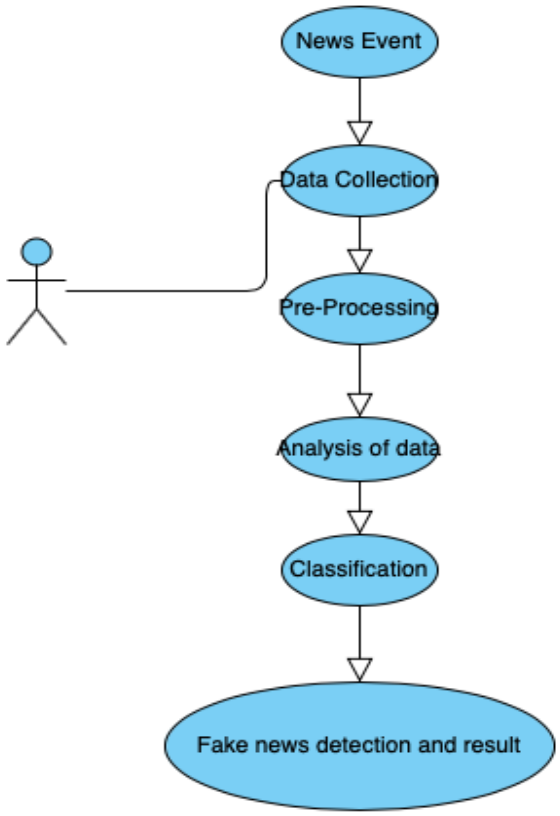


Fig 5.2: Use Case Diagram of Fake News Detection Model

The above figure 5.2, represents use case diagram of proposed system, for fake news detection. The actor and use case are represented. An eclipse shape represents the use case namely news event, data collection, pre-processing, analysis of data, classification and fake news detection result.

5.3 Sequence Diagram

A sequence diagram as shown in figure 5.3 shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of news collected by the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the

system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the news exchanged if they are fake or real, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

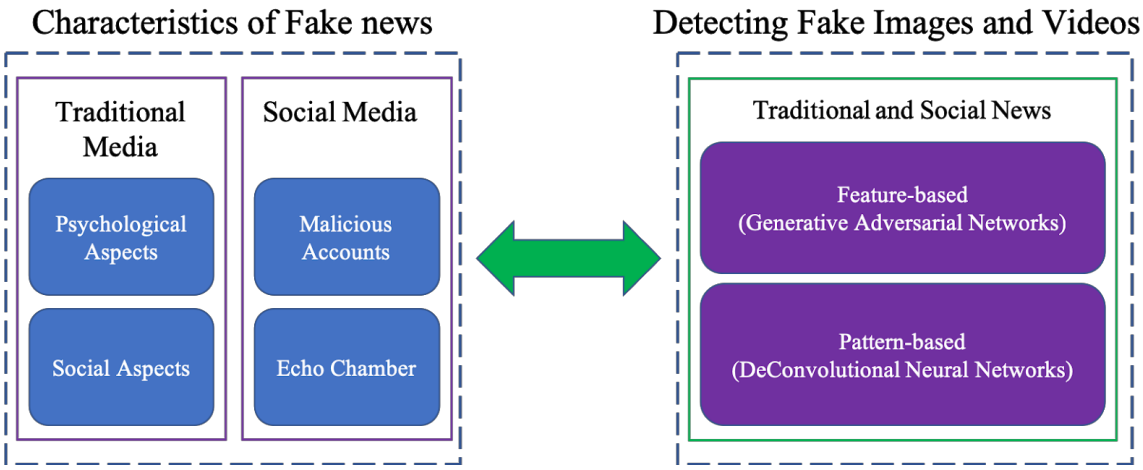


Fig 5.3: Sequence Diagram of Fake News Detection model [16]

5.4 System Architecture

Figure 5.4 represents system architecture of the proposed system, where we are applying Doc2Vec to predict fake news. There are three modules namely Extracting the data, Data pre-processing and Prediction.

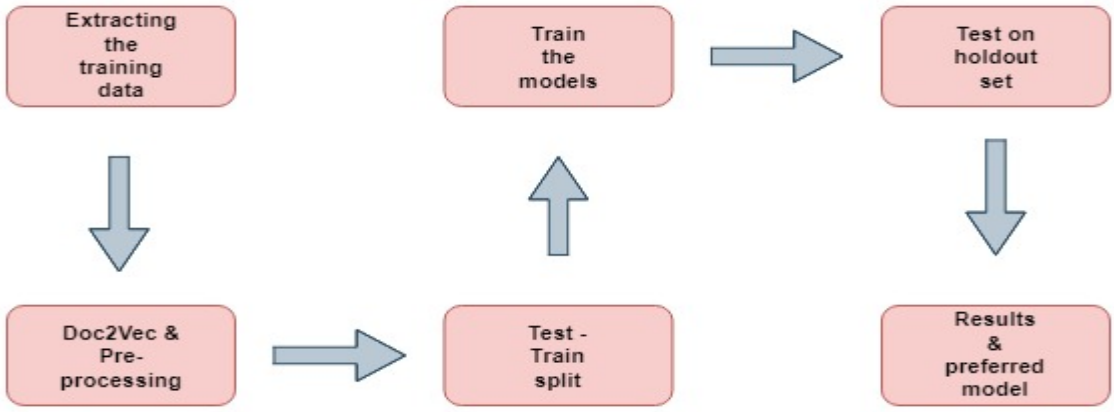


Fig 5.4: Architecture Diagram of Fake News Detection Model

5.5 Deployment Diagram

In the deployment diagram as shown in figure 5.5, the UML models the physical deployment of artifacts on nodes. The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

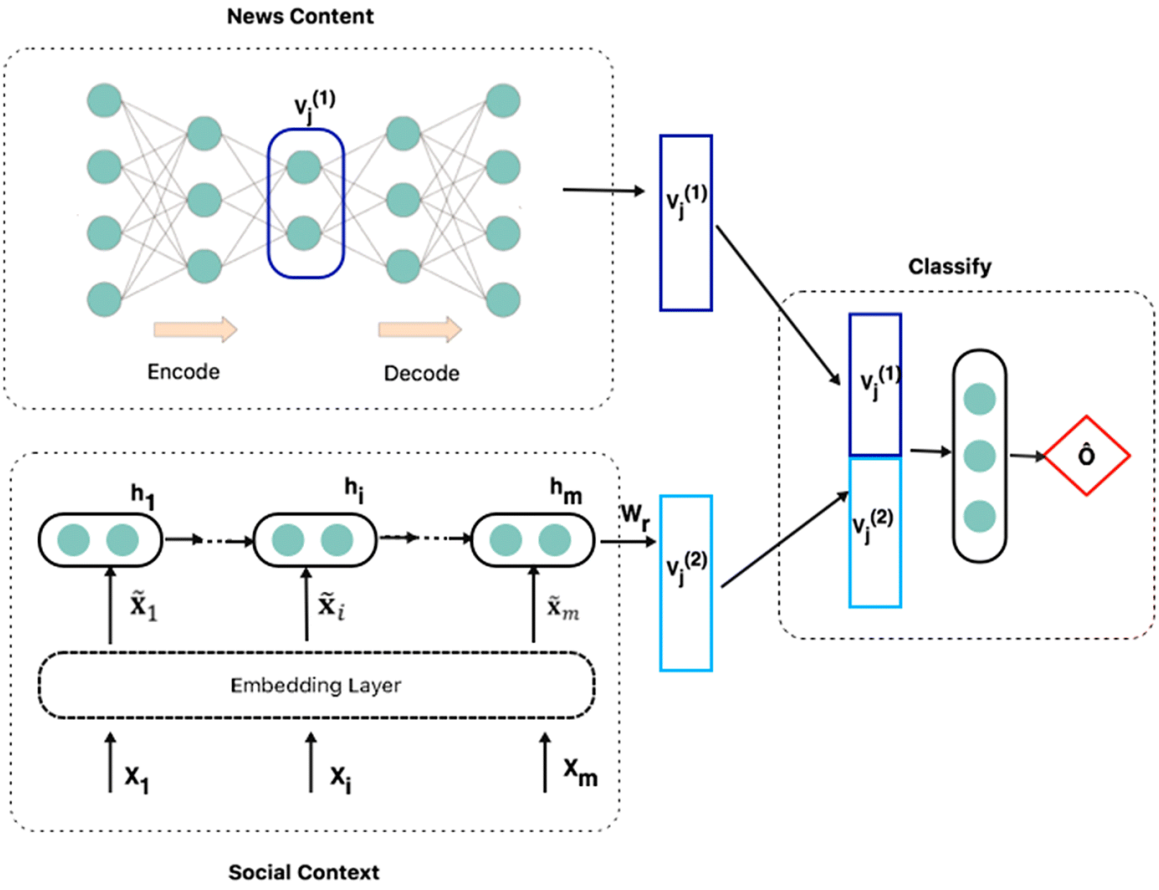


Fig 5.5: Deployment Diagram of Fake News Detection Model [17]

CHAPTER 6

IMPLEMENTATION

6.1 DATASET DESCRIPTION

The datasets used for this project were drawn from Kaggle. The training dataset has about 26000 rows of data from various articles on the internet. There was a lot of pre-processing to be done before the news articles could be used. A full training dataset has the following attributes:

1. id: unique id for a news article
2. title: the title of a news article
3. author: author of the news article
4. text: the text of the article; incomplete in some cases
5. label: a label that marks the article as potentially unreliable
    - 1: fake
    - 0: real

6.2 FEATURE EXTRACTION AND PRE-PROCESSING

Before representing the data using the various models, the data need to be subjected to certain refinements like stop-word removal, tokenization, a lower casing, and punctuation removal. This will help us reduce the size of actual data by removing the irrelevant information that exists in the data. A generic processing function was created to eliminate punctuation as well as non-letter characters for each document. Then the letters in the document were lowercased before removing the stop words.

Stop words are insignificant words in a language that will create noise when used as features in text classification. These are words commonly used a lot in sentences to help connect thought or to assist in the sentence structure. Articles, prepositions and conjunctions and some pronouns are considered stop words. Then the common words were removed such as, a, about, an, are, as, at, be, by, for, from, how, in, is, of, on, or, that, the,

these, this, too, was, what, when, where, who, will, etc. This produces a comma-separated list of words, which can be input into the Doc2Vec algorithm to produce a fixed length embedding vector for each article. The embeddings used for the majority of our modelling are generated using the Doc2Vec model. The aim is to produce a vector representation of each article.

Doc2Vec is a model developed in 2014 based on the existing Word2Vec model, which generates vector representations for words [11]. Word2Vec represents documents by combining the vectors of the individual words, but in doing so it loses all word order information. Doc2Vec expands on Word2Vec by adding a "document vector" to the output representation, which contains some information about the document as a whole and allows the model to learn some information about word order.

6.3 SOURCE CODE

APPENDIX A: LSTM.py

APPENDIX B: Fake news.py

CHAPTER 7

RESULTS AND PERFORMANCE EVALUATION

The results arrived at by calculating the accuracies of the various models mentioned above. The accuracy, precisions and f1 score can be computed with the help of confusion matrices. A single confusion matrix was created for each model. The values shown are the averaged values over successive trials.

Table 7: Results of Algorithm

Model	Accuracy
Naïve Bayes	71.84%
Support Vector Machine	87.37%
LSTM	94.27%
Keras Based Neural Network	90.62%

Based on the results in Table 7, the graph in Fig 7 is constructed by taking different algorithms on X-axis and accuracy on the Y-axis. It is inferred that LSTM provides us with the highest accuracy followed by Keras Neural Network, SVM and finally Naïve Bayes. The Keras based Neural network has a good accuracy that almost matches LSTM. LSTM provides such high results because the text is inherently a serialized object.

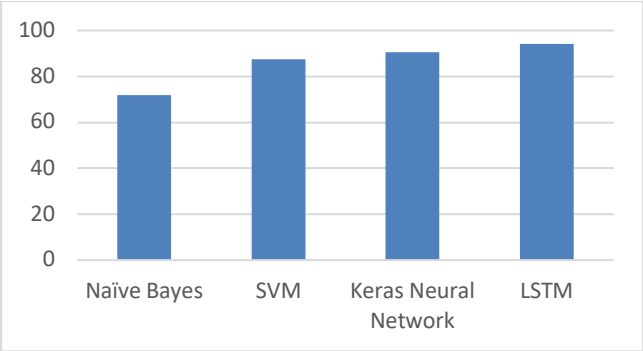


Fig 7: Comparison of Algorithm Results

CHAPTER 8

CONCLUSION AND FUTURE WORK

The detection model for fake news has been implemented through different machine learning techniques. Furthermore, the four methods are investigated and compared with their accuracies. The proposed model achieves its highest accuracy when using LSTM. The highest accuracy score is 94%.

Fake news detection is an emerging research area which has a scarce number of datasets. There are no data on real-time news or regarding the current affairs. The current model is run against the existing dataset, showing that the model performs well against it.

In our future work, news article data can be considered related to recent incidents in the corpus of data. The next step then would be to train the model and analyse how the accuracies vary with the new data to further improve it.

REFERENCES

1. Shu K., Sliva A., Wang S., Tang J., Liu H., *Fake News Detection on Social Media: A Data Mining Perspective,* ACM SIGKDD Explorations Newsletter, 2017, 19(1), 22-36.

2. Rubin, V., Conroy N., Chen Y., Cornwell S., *Fake News or Truth? Using Satirical Cues to Detect Potentially Misleading News*, Proceedings of the Second Workshop on Computational Approaches to Deception Detection Association for Computational Linguistics, 2016.

3. Kelly Stahl., *Fake news detection in social media*, B.S. Candidate, Department of Mathematics and Department of Computer Sciences, California State University Stanislaus, 2018.

4. Marco L. Delia Vedova, Stefano Moret, Eugenio Tacchini, Massimo Di Pierro, *Automatic Online Fake News Detection Combining Content and Social Signals*, ResearchGate May 2018.

5. Mykhailo Granik, Volodymyr Mesyura, *Fake News Detection Using Naive Bayes Classifier*, 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON).

6. Namwon Kim, Deokjin Seo, Chang-Sung Jeong, *FAMOUS: Fake News Detection Model based on Unified Key Sentence Information*, 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)

7. Hadeer Ahmed, Issa Traore, Sherif Saad, *Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques,* ResearchGate May 2017.

8. Saranya Krishnan, Min Chen., *Identifying tweets with fake news*, 2018 IEEE Conference on Information Reuse Integration for Data Science.

9. Stefan Helmstetter, Heiko Paulhem, *Weakly Supervised Learning for Fake news detection on Twitter*, 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM).

10. Akshay Jain, *Fake News Detection*, IEEE 2018 International Students' Conference on Electrical, Electronics and Computer Sciences.

11. Quoc L., Mikolov T., *Distributed Representations of Sentences and Documents*, Arxiv Archive May 2014.

12. Saxena, R., *How the Naive Bayes Classifier works in Machine Learning*, http://dataaspirant.com/2017/02/06/naive-bayes-classifier-machine-learning/, Retrieved: April 15, 2019.

13. Brambrick, Aylien, *Support Vector Machines: A Simple Explanation,* https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html, Retrieved: April 20, 2019.

14. Hochreiter, S., Jrgen, S., *Long short-term memory,* Institute of Bioinformatics Publications Newsletter, October 1997.

15. Goldberg, Y., *A Primer on Neural Network Models for Natural Language Processing*, Arxiv Archive 2015.

16. https://www.google.com/search?client=safari&rls=en&q=deployment+diagram+for+fake+news+detection&ie=UTF-8&oe=UTF-8

17. https://www.google.com/search?q=deployment+diagram+for+fake+news+detection&client=safari&rls=en&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjXr_Dev6niAhVp73MBHVLvBo8Q_AUIDigB&biw=1440&bih=837#imgrc=lsc-sHyDYwq2_M:

18. https://www.google.com/search?q=class+diagram+for+fake+news+detection&client=safari&rls=en&source=lnms&tbm=isch&sa=X&ved=0ahUKEwj15YXKnafiAhUS8XMBHdLhAUEQ_AUIDigB&biw=1440&bih=837#imgrc=26MRM3bk715SKM:

19. https://www.google.com/search?q=lstm&client=safari&rls=en&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjw3KGTwqniAhVQ6nMBHYdrBbIQ_AUIDigB&biw=1440&bih=837#imgrc=osu-IXgM4TCE1M:

20. https://www.google.com/search?q=Binary+Classification+and+Multiclass+Classification&client=safari&rls=en&source=lnms&tbm=isch&sa=X&ved=0ahUKEwj_1peuw6niAhVh63MBHd3hA8QQ_AUIDygC&biw=1440&bih=837#imgrc=hymhPlvT6xuX3M:

21. https://www.google.com/search?q=Regression+of+Day+vs+Rainfall+(in+mm)&client=safari&rls=en&source=lnms&tbm=isch&sa=X&ved=0ahUKEwif9Lb_w6niAhUw8HMBHRO2A-0Q_AUIDigB&biw=1440&bih=837#imgrc=Sn-Z0yMiiDR5uM:

22. https://www.google.com/search?q=Na%C3%AFve+Bayes&client=safari&rls=en&source=lnms&tbm=isch&sa=X&ved=0ahUKEwi37p7axKniAhX54XMBHbnRD_MQ_AUIDygC&biw=1440&bih=837#imgrc=rr5Zof0ybullxM:

23. https://www.google.com/search?q=Support+vector+machine&client=safari&rls=en&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiA0eCaxaniAhVEg-YKHf6WDcoQ_AUIDigB&biw=1440&bih=837#imgrc=cjeTGYu-WB0zjM:

APPENDIX A

LSTM.py

```
import numpy as np
import re
import string
import pandas as pd
from gensim.models import Doc2Vec
from gensim.models.doc2vec import LabeledSentence
from gensim import utils
from nltk.corpus import stopwords


def textClean(text):
    text = re.sub(r"[^A-Za-z0-9^,!.\/'+-=]", " ", text)
    text = text.lower().split()
    stops = set(stopwords.words("english"))
    text = [w for w in text if not w in stops]
    text = " ".join(text)
    return (text)


def cleanup(text):
    text = textClean(text)
    text = text.translate(str.maketrans("", "", string.punctuation))
    return text
```

```
def constructLabeledSentences(data):
    sentences = []
    for index, row in data.iteritems():
        sentences.append(LabeledSentence(utils.to_unicode(row).split(), ['Text' + '_%s' % str(index)]))
    return sentences

def clean_data():
    path = 'train.csv'
    vector_dimension=300

    data = pd.read_csv(path)

    missing_rows = []
    for i in range(len(data)):
        if data.loc[i, 'text'] != data.loc[i, 'text']:
            missing_rows.append(i)
    data = data.drop(missing_rows).reset_index().drop(['index','id'],axis=1)

    for i in range(len(data)):
        data.loc[i, 'text'] = cleanup(data.loc[i,'text'])

    data = data.sample(frac=1).reset_index(drop=True)

    x = data.loc[:,'text'].values
    y = data.loc[:,'label'].values

    train_size = int(0.8 * len(y))
    test_size = len(x) - train_size

    xtr = x[:train_size]
```

```
    xte = x[train_size:]
    ytr = y[:train_size]
    yte = y[train_size:]

    np.save('xtr_shuffled.npy',xtr)
    np.save('xte_shuffled.npy',xte)
    np.save('ytr_shuffled.npy',ytr)
    np.save('yte_shuffled.npy',yte)
import numpy as np
from keras.datasets import imdb
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers.embeddings import Embedding
from keras.preprocessing import sequence
from collections import Counter
import os
import matplotlib.pyplot as plt
import scikitplot.plotters as skplt


top_words = 5000
epoch_num = 5
batch_size = 64

def plot_cmat(yte, ypred):
    '''Plotting confusion matrix'''
    skplt.plot_confusion_matrix(yte, ypred)
    plt.show()

if not os.path.isfile('./xtr_shuffled.npy') or \
```

```
    not os.path.isfile('./xte_shuffled.npy') or \
    not os.path.isfile('./ytr_shuffled.npy') or \
    not os.path.isfile('./yte_shuffled.npy'):
    clean_data()


xtr = np.load('./xtr_shuffled.npy')
xte = np.load('./xte_shuffled.npy')
y_train = np.load('./ytr_shuffled.npy')
y_test = np.load('./yte_shuffled.npy')

cnt = Counter()
x_train = []
for x in xtr:
    x_train.append(x.split())
    for word in x_train[-1]:
        cnt[word] += 1

# Storing most common words
most_common = cnt.most_common(top_words + 1)
word_bank = {}
id_num = 1
for word, freq in most_common:
    word_bank[word] = id_num
    id_num += 1

# Encode the sentences
for news in x_train:
    i = 0
    while i < len(news):
        if news[i] in word_bank:
```

```
            news[i] = word_bank[news[i]]
            i += 1
        else:
            del news[i]

y_train = list(y_train)
y_test = list(y_test)

# Delete the short news
i = 0
while i < len(x_train):
    if len(x_train[i]) > 10:
        i += 1
    else:
        del x_train[i]
        del y_train[i]

# Generating test data
x_test = []
for x in xte:
    x_test.append(x.split())

# Encode the sentences
for news in x_test:
    i = 0
    while i < len(news):
        if news[i] in word_bank:
            news[i] = word_bank[news[i]]
            i += 1
        else:
            del news[i]
```

```
# Truncate and pad input sequences
max_review_length = 500
X_train = sequence.pad_sequences(x_train, maxlen=max_review_length)
X_test = sequence.pad_sequences(x_test, maxlen=max_review_length)

# Convert to numpy arrays
y_train = np.array(y_train)
y_test = np.array(y_test)

# Create the model
embedding_vecor_length = 32
model = Sequential()
model.add(Embedding(top_words+2, embedding_vecor_length,
input_length=max_review_length))
model.add(LSTM(100))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=epoch_num,
batch_size=batch_size)

# Final evaluation of the model
scores = model.evaluate(X_test, y_test, verbose=0)
print("Accuracy= %.2f%%" % (scores[1]*100))

# Draw the confusion matrix
y_pred = model.predict_classes(X_test)
plot_cmat(y_test, y_pred)
```

APPENDIX B

Fake news.py

```
import numpy as np
import re
import string
import pandas as pd
from gensim.models import Doc2Vec
from gensim.models.doc2vec import LabeledSentence
from gensim import utils
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords')


def textClean(text):
    text = re.sub(r"[^A-Za-z0-9^,!.\/'+-=]", " ", text)
    text = text.lower().split()
    stops = set(stopwords.words("english"))
    text = [w for w in text if not w in stops]
    text = " ".join(text)
    return (text)


def cleanup(text):
    text = textClean(text)
    text = text.translate(str.maketrans("", "", string.punctuation))
    return text
```

```
def constructLabeledSentences(data):
    sentences = []
    for index, row in data.iteritems():
        sentences.append(LabeledSentence(utils.to_unicode(row).split(), ['Text' + '_%s' % str(index)]))
    return sentences


def getEmbeddings(path,vector_dimension=300):
    data = pd.read_csv(path)

    missing_rows = []
    for i in range(len(data)):
        if data.loc[i, 'text'] != data.loc[i, 'text']:
            missing_rows.append(i)
    data = data.drop(missing_rows).reset_index().drop(['index','id'],axis=1)

    for i in range(len(data)):
        data.loc[i, 'text'] = cleanup(data.loc[i,'text'])

    x = constructLabeledSentences(data['text'])
    y = data['label'].values

    text_model = Doc2Vec(min_count=1, window=5, vector_size=vector_dimension, sample=1e-4, negative=5, workers=7, epochs=10,
                         seed=1)
    text_model.build_vocab(x)
    text_model.train(x, total_examples=text_model.corpus_count, epochs=text_model.iter)

    train_size = int(0.8 * len(x))
    test_size = len(x) - train_size
```

```
    text_train_arrays = np.zeros((train_size, vector_dimension))
    text_test_arrays = np.zeros((test_size, vector_dimension))
    train_labels = np.zeros(train_size)
    test_labels = np.zeros(test_size)

    for i in range(train_size):
        text_train_arrays[i] = text_model.docvecs['Text_' + str(i)]
        train_labels[i] = y[i]

    j = 0
    for i in range(train_size, train_size + test_size):
        text_test_arrays[j] = text_model.docvecs['Text_' + str(i)]
        test_labels[j] = y[i]
        j = j + 1

    return text_train_arrays, text_test_arrays, train_labels, test_labels
import numpy as np
from sklearn.svm import SVC
import matplotlib.pyplot as plt
import scikitplot.plotters as skplt


def plot_cmat(yte, ypred):
    '''Plotting confusion matrix'''
    skplt.plot_confusion_matrix(yte,ypred)
    plt.show()


xtr,xte,ytr,yte = getEmbeddings("train.csv")
np.save('./xtr', xtr)
```

```
np.save('./xte', xte)
np.save('./ytr', ytr)
np.save('./yte', yte)

xtr = np.load('./xtr.npy')
xte = np.load('./xte.npy')
ytr = np.load('./ytr.npy')
yte = np.load('./yte.npy')

clf = SVC()
clf.fit(xtr, ytr)
y_pred = clf.predict(xte)
m = yte.shape[0]
n = (yte != y_pred).sum()
print("Accuracy = " + format((m-n)/m*100, '.2f') + "%")   # 88.42%

plot_cmat(yte, y_pred)
```