

HELWAN UNIVERSITY  
Faculty of Computers and Artificial Intelligence  
Computer Science Department

# Twitter Sentiment Analysis Using NLP

A graduation project dissertation by:

Kareem Magdy Ahmed [20150389]  
Omar Adel Mohamed [20150355]  
Mohamed Ahmed Shawky [20150413]  
Hossam Abdel-Monem Mohamed [20150207]  
Esraa Gamal Atito [20160061]  
Mahmoud Aly Abdel-Aziz [20140386]

Submitted in partial fulfilment of the requirements for the degree of Bachelor of Science  
in Computers & Artificial Intelligence, at the Computer Science Department, the Faculty  
of Computers & Artificial Intelligence, Helwan University

Supervised by:

Ass. Prof. Hala Abdel-Galil

July 27, 2020

## ABSTRACT

Analysis of public information from social media could yield interesting results and insights into the world of public opinions about almost any product, service or personality. Social network data is one of the most effective and accurate indicators of public sentiment. The explosion of Web has led to increased activity in Podcasting, Blogging, Tagging, Contributing to RSS, Social Bookmarking, and Social Networking. As a result there has been an eruption of interest in people to mine these vast resources of data for opinions. Sentiment Analysis or Opinion Mining is the computational treatment of opinions, sentiments and subjectivity of text. In this paper we will be discussing a methodology which allows utilization and interpretation of twitter data to determine public opinions.

Developing a program for sentiment analysis is an approach to be used to computationally measure customers' perceptions. This paper reports on the design of a sentiment analysis, extracting and training a vast amount of tweets. Results classify customers' perspective via tweets into positive, negative and neutral, which is represented in a pie chart, bar diagram using JavaScript, CSS and HTML pages.

# Table of Contents

<b>Table &amp; Figures.....</b>	<b>5</b>
<b>1 Introduction .....</b>	<b>6</b>
1.1 INTRODUCTION .....	6
1.2 MOTIVATION .....	6
1.3 TARGET USERS.....	6
1.4 OBJECTIVES .....	6
1.5 PROJECT SUMMARY AND PURPOSE .....	7
1.6 OVERVIEW OF THE PROJECT .....	7
1.7 PROBLEM DEFINITION.....	7
1.8 CHALLENGES .....	8
1.9 SCOPE .....	8
<b>2 Similar systems.....</b>	<b>9</b>
2.1 SIMILAR SYSTEMS DESCRIPTIONS .....	9
Paper 1.....	9
Paper 2.....	10
Paper 3.....	12
Paper 4.....	15
Paper 5.....	15
Paper 6.....	18
Paper 7.....	19
2.2 COMPARISON WITH PROPOSED PROJECT .....	20
<b>3 Feasibility Study.....</b>	<b>21</b>
3.1 FEASIBILITY ANALYSIS .....	21
3.1.1 Technical Feasibility.....	21
3.1.2 Operational Feasibility.....	22
3.1.3 Economic Feasibility .....	22
3.1.4 Schedule Feasibility .....	22
3.2 REQUIREMENT DEFEMINATION .....	22
3.2.1 Functional Requirements .....	23
3.2.2 Non-Functional Requirements .....	23
<b>4 System Design and Architecture .....</b>	<b>24</b>
4.1 INTRODUCTION .....	24
4.2 DIAGRAMS .....	24
4.2.1 Activity Diagram .....	24
4.2.2 Use Case Diagram.....	26
4.2.3 System Flow Diagram .....	27

4.2.4 Data Flow Diagram .....	28
4.2.5 Sequence Diagram.....	29
4.2.6 Flow Chart Diagram .....	30
<b>5 Implementation.....</b>	<b>31</b>
5.1 INTRODUCTION .....	31
5.2 NATURAL LANGUAGE PROCESSING.....	31
5.3 SENTIMENT ANALYSIS.....	31
5.4 ALGORITHMS .....	32
5.4.1 RNN and LSTM .....	32
5.4.2 Entropy Loss.....	32
5.4.3 Optimizer .....	33
5.5 DIFFICULTIES .....	33
5.6 TOOLS .....	34
5.7 DATASET.....	34
5.8 RESULTS .....	36
5.9 GUI AND CODE .....	37
GUI.....	37
Model Training Code .....	39
<b>6 Conclusion .....</b>	<b>42</b>
6.1 CONCLUSION.....	42
6.2 FUTURE WORK .....	42
<b>References .....</b>	<b>43</b>

## Tables:-

Table 2.1.....	9
Table 2.2.....	10
Table 2.3.....	11
Table 2.4.....	13
Table 2.5.....	14
Table 2.6.....	15
Table 2.7.....	17
Table 2.8.....	19

## Figures:-

Figure 1 Activity Diagram.....	25
Figure 2 Use Case Diagram .....	26
Figure 3 System Flow Diagram.....	27
Figure 4 Dataflow Diagram .....	28
Figure 5 Sequence Diagram .....	29
Figure 6 Flow Chart Diagram .....	30
Figure 7 simple of the data set .....	35
Figure 8 Dataset Labels Distribution .....	35
Figure 9 Classification Report .....	36
Figure 10 Confusion Matrix.....	36
Figure 11 GUI Home.....	37
Figure 12 GUI Pie Chart.....	37
Figure 13 GUI Bar Chart .....	38
Figure 14 GUI Tweets .....	38
Figure 15 Preprocessing Code.....	39
Figure 16 Word2Vec Code .....	39
Figure 17 Tokenize Text Code .....	40
Figure 18 Embedding Layer Code .....	40
Figure 19 Build and Compile Code.....	40
Figure 20 Train and Evaluate Code .....	41
Figure 21 Decode Sentiment and Perform Prediction Code .....	41

# Chapter 1

## Introduction to The System

### 1.1 Introduction

Social media is used nowadays by billions of people, contain their opinions, emotions, thoughts and feelings. Twitter is a preferred destination for reviews, it used by companies and organizations as an evaluation measure, and they make polling for their products and services. Tweets and re-tweets are very helpful indicators for public sentiment and provide us with information which is helpful to any company aiming for higher sales, better growth or hoping to correct a failing product. The tweets number are huge which makes it hard to analysis them by humans, so NLP, data mining and machine learning is helpful to process huge data analysis but it is difficult for machine to understand the meaning of human languages due to the ambiguity of some words meaning or not standard spellings, so that the computer struggle understanding our languages.

### 1.2 Motivation

People write tweets about their opinion or thoughts about products, so it is hard for companies to manually read all these tweets or comments and get information from them so sentimental analysis is very helpful for decrease the cost and time for classify a great number of statements in twitter.

### 1.3 Target Users

Our main users are managers and owners of organizations that want a feedback from user tweets about their product.

### 1.4 objectives

Our objective is to reduce the cost of reading and analysis of people and costumers' opinions that they post on social media so owners and company managers can get information of how to modify their products to satisfy customers' needs and increasing the accuracy of classifying the tweets.

## **1.5 Project summary and purpose**

This project of analyzing sentiments of tweets comes under the domain of “Pattern Classification” and “Data Mining”. Both of these terms are very closely related and intertwined, and they can be formally defined as the process of discovering “useful” patterns in large set of data, either automatically (unsupervised) or semi automatically (supervised). The project would heavily rely on techniques of “Natural Language Processing” in extracting significant patterns and features from the large dataset of tweets and on “Machine Learning” techniques for accurately classifying individual unlabeled data samples (tweets) according to which ever pattern model best describes them.

### **1.5 Overview of the project**

This proposal entitled “Twitty Mood” is a web application which is used to analyze the tweets. We will be performing sentiment analysis on tweets and determine if it is positive, negative or neutral. This web application can be used by any organization office to review their works, Political leaders to review people opinions about their plans or any other company to review their products or brands.

The main feature of our web application is that it helps to determine the opinion about the peoples on products, government work, politics or any other by analyzing the tweets. Our system is capable of training the new tweets taking reference to previously trained tweets. The computed or analyzed data will be represented in various diagram such as Pie- chart and Bar graph with ability to review tweets.

## **1.6 Problem definition**

The algorithm proposed works on Twitter Data, primarily it collects the tweets and then study it with the help of different statistical computing procedures we are using TWINT to collect our tweets and pass it to our algorithm.

## 1.8 challenges

The corpus is a set of positive, negative and neutral tweets, contain emotions, non ASCII characters, Hashtags, the ambiguity of some word meanings that the same word has many different meanings also some non-standard English words like happyyy , saaaad which makes the computer to not give the accurate classification for the input, also the tokenization of statements which split words which may relate together to give a meaning like “not like” it may classify it wrong.

## 1.9 Scope

This project will be helpful to the companies, political parties as well as to the common people. It will be helpful to political party for reviewing about the program that they are going to do or the program that they have performed. similar companies also can get review about their new product on newly released hardware and software. Also the movie maker can take review on the currently running movie. The system user types a statement, the statement is preprocessed and is passed to the classifier that return a positive or negative for the managers using the system and get information about the opinion of the users of the system.



## Chapter 2

### Similar Systems

#### 2.1 Similar System Description

##### Paper 1:

The authors in [1] present a new ensemble method that uses a lexicon based sentiment score as input feature for the machine learning approach. The combined method proved to produce more precise classifications. also show that employing a cost-sensitive classifier for highly unbalanced datasets yields an improvement of sentiment classification performance up to 70%.

One of the most popular microblogging platforms is Twitter. It has been growing steadily for the last several years and has become a meeting point for a diverse range of people: students, professionals, celebrities, companies and politicians. This popularity of Twitter results in the enormous amount of information being passed through the service, covering a wide range of topics from people well-being to the opinions about the brands, products, politicians and social events. In this contexts Twitter becomes a powerful tool for predictions. For example, (Asur and Huberman, 2010) was able to predict from Twitter analytics the amount of ticket sales at the opening weekend for movies with 97.3% accuracy, higher than the one achieved by the Hollywood Stock Exchange, a known prediction tool for the movies.

As it can be observed from tables and , the addition of the “Lexicon Sentiment” feature and other manually constructed features allowed to increase all performance measures significantly for 3 classifiers. For example, the accuracy of Naive Bayes classifier was increased by 7%, accuracy of Decision Trees was increased by over 9%, and the accuracy of SVM improved by 4.5%

Method	Tokens Type	Folds Number	Accuracy	Precision	Recall	F-Score
Naive Bayes	uni/bigrams	5	81.5%	0.82	0.82	0.82
Decision Trees	uni/bigrams	5	80.57%	0.81	0.81	0.81
SVM	uni/bigrams	5	86.62%	0.87	0.87	0.87

*Table 2.1: Scenario 1: 5-fold cross-validation test on a movies reviews dataset using only N-grams as features.*

<b>Method</b>	<b>Tokens Type</b>	<b>Folds Number</b>	<b>Accuracy</b>	<b>Precision</b>	<b>Recall</b>	<b>F-Score</b>
<b>Naive Bayes</b>	uni/bigrams	5	88.54%	0.89	0.86	0.86
<b>Decision Trees</b>	uni/bigrams	5	89.9%	0.90	0.90	0.90
<b>SVM</b>	uni/bigrams	5	91.17%	0.91	0.91	0.91

*Table 2.2: Scenario 2: 5-fold cross-validation test on a movies reviews dataset using traditional N-grams features in combination with manually constructed features: lexicon sentiment score, number of different parts-of-speech, number of emoticons, number of elongated words, etc.*

In the machine learning approach, they propose to use a lexicon sentiment obtained during the lexicon based classification as an input feature for training classifiers. The ranking of all features based on the information gain scores during the feature selection process revealed that the lexicon feature appeared on the top of the list, confirming its relevance in sentiment classification. They also demonstrated that in case of highly unbalanced datasets the utilisation of cost-sensitive classifiers improves accuracy of class prediction: on the benchmark Twitter dataset a cost-sensitive SVM yielded 7% increase in performance over a standard SVM.

## **Paper 2:**

The authors [2] in This paper presents an unsupervised method for analyzing tweet sentiments. Polarity of tweets is evaluated by using three sentiment lexicons—SenticNet, SentiWordNet and SentislangNet. SentislangNet is a sentiment lexicon built from SenticNet and SentiWordNet for slangs and acronyms. Experimental results show fairly good FF-score. The method is implemented and tested in parallel python framework and is shown to scale well with large volume of data on multiple cores.

They propose an unsupervised and domain-independent approach by using the polarity scores from three lexical resources—SentiWordNet 3.0, SenticNet 2 and SentislangNet. SentiWordNet contains polarity scores of uni-grams. SenticNet 2.0 is a publicly available semantic resource which contains commonly used polarity concepts. The method exploits SenticNet along with SentiWordNet to analyse twitter sentiments. They have also created a sentiment lexicon for slangs and acronyms called SentislangNet using SenticNet and SentiWordNet. The algorithm has been implemented in parallel python framework and shown to scale well with multiple cores for large volume of data.

Our approach is compared with two unsupervised methods based on SentiWordNet. Few recent approaches have used SentiWordNet lexical resource to implement sentiment analysis methods. Since their data sets are not public, we implemented and tested two methods to compare the performance.

The performance of the sentiment analyser on two data sets is shown in Table below. Results show that our unsupervised method (SN-SWN) achieved reasonably good recall and F-measure and outperforms other baseline methods (SWN, RW-SWN). We found that using other sentiment lexicons (SenticNet, SentislangNet) along with SentiWordNet has improved the accuracy of sentiment analysis.

Methods	Corpus	Precision	Recall	F-score
Performance				
SWN	Data set#1	59.8	56.24	57.97
	Data set#2	58.87	54.87	56.8
RW-SWN	Data set#1	63.7	63.11	63.42
	Data set#2	60.5	59.89	60.09
SN-SWN	Data set#1	64.64	74.35	69.16
	Data set#2	62.9	72.73	67.46

Table 2.3 performance comparison of sentiment analysis methods

In the results They found that SenticNet has contributed polarity scores of several *n*grams commonly present in tweets and hence enhanced the accuracy of sentiment analyzer. Including more concepts in SenticNet may increase the accuracy further.

### Paper 3:

The authors [3] said because of comprises of people's views is becoming very important in order to gauge public opinion on a particular topic of interest. It can help evaluate consumer satisfaction about some products, customers' interests and preferences, political viewpoints and many others. Indeed, number of surveys shows that: • 91% of people visited a store because of an online experience. Among which 22% were influenced by Twitter and Facebook experiences • 72% of consumers trust online reviews as much as personal recommendations • 78% of consumers state that posts made by companies on social media influence their purchases .

Sentiment analysis on text is a very difficult task by itself, given the unstructured or in the best cases ill structured nature of text along with the context complexity, let alone extracting sentiment from a text as noisy as social media text. There are some difficulties inherent in analyzing sentiment from social media . One example is “False negatives” where words such as “crying” and “crap” generally suggest negativity, yet they imply positive sentiment when used in a sentence such as “I was crying with joy” or “Holy crap! This is great”. Another example is “Conditional sentiment,” such as “If someone doesn't call me back, I will never do business with them again.” These examples show how sentiment analysis of social media text can be hard Moreover, the process gets even harder with the use of emoticons such as “☺” (“smiley”) and hash-tags such as “#happy” to express feelings ironically or sarcastically.

The study analyzes formally the performance of sentiment classification methods based on fair experimental setups. They analyze unigrams, as well as, bigrams as features spaces. For example for a tweet “I Love Kindle, It’s Amazing”, unigrams = {I, Love, Kindle, Its, Amazing}, bigrams = {I Love, Love Kindle, Kindle Its, Its Amazing}. Moreover, they analyze term frequency representation of dataset (i.e. the number of occurrences of a term in a document), as well as, term presence representation (i.e. the occurrence or absence of a term in a document regardless of how many times it occurred). For training and testing we are using Stanford Testing Dataset

In this comparative study the need to evaluate the performance of Multinomial NB, Bernoulli NB and SVM in sentiment mining of Twitter data. The selected classifiers are the most commonly used machine learning classifiers . For comparison They use a selected Twitter dataset, apply suitable preprocessing steps then produce the dataset with unigrams and

bigrams, one time with term frequencies and one time with term presence (i.e. polarity dataset). Afterwards, the three selected classifiers are trained with the four variations of the input dataset and the accuracy results are compared along with training time

For results:

	Dataset Type	Unigrams (1442 Features)	Time (Sec)	Bigrams (4150 Features)	Time (Sec)
<b>BNB</b>	<b>Polarity</b>	<b>76.6%</b>	<b>0.22</b>	<b>70.75 %</b>	<b>0.61</b>
	<b>Frequency</b>	<b>75.21 %</b>	<b>0.24</b>	<b>65.18 %</b>	<b>0.61</b>
<b>MNB</b>	<b>Polarity</b>	<b>79.39 %</b>	<b>0.13</b>	<b>75.77 %</b>	<b>0.13</b>
	<b>Frequency</b>	<b><u>81.34 %</u></b>	<b>0.05</b>	<b>72.14 %</b>	<b>0.11</b>
<b>SVM</b>	<b>Polarity</b>	<b>74.37 %</b>	<b>4.34</b>	<b>74.09 %</b>	<b>12.16</b>
	<b>Frequency</b>	<b>77.16 %</b>	<b>4.22</b>	<b>69.95 %</b>	<b>12.95</b>

Table 2.4 - Testing Results -10-Fold Cross Validation

Table shows the experimental classification results for Bernoulli Naïve Bayes, Multinomial Naïve Bayes and SVM classifiers with 10-fold cross validation. The results show that overall accuracy for unigrams datasets are higher than the accuracy for bigrams datasets. Furthermore, training time for unigrams dataset is in general less than bigrams. This is expected since bigrams produce larger feature space. Multinomial Naïve Bayes produced the best classification results with frequency, unigrams dataset. SVM requires the longest training time to build the model and does not outperform other classifiers in the context of sentiment analysis of Twitter, which makes it less preferable choice for sentiment analysis compared to multinomial NB that produces good accuracy results at very high training speed.

	Dataset Type	Unigrams (1442 Features)	Time (Sec)	Bigrams (4150 Features)	Time (Sec)
<b>BNB</b>	<b>Polarity</b>	<b>73.76%</b>	<b>0.22</b>	<b>73.77 %</b>	<b>0.59</b>
	<b>Frequency</b>	<b>73.77 %</b>	<b>0.25</b>	<b>68.03%</b>	<b>0.59</b>
<b>MNB</b>	<b>Polarity</b>	<b><u>82.78%</u></b>	<b>0.13</b>	<b>80.32 %</b>	<b>0.11</b>
	<b>Frequency</b>	<b>80.32 %</b>	<b>0.05</b>	<b>77.86%</b>	<b>0.09</b>
<b>SVM</b>	<b>Polarity</b>	<b>79.50 %</b>	<b>4.34</b>	<b>72.95 %</b>	<b>12.23</b>
	<b>Frequency</b>	<b>80.32%</b>	<b>4.22</b>	<b>66.39%</b>	<b>12.42</b>

Table 2.5 - Testing Results - Dataset split into 66% training set Conclusion

Table shows the experimental classification results for Bernoulli Naïve Bayes, Multinomial Naïve Bayes and SVM classifiers with STS set divided into training set and testing set. The first observation is that training time did not significantly change. This means that using either method, cross validation or training set would take comparable training time for model building. For MNB and BNB the classification results of bigrams with training outperformed the classification results with cross validation. For Bernoulli NB the accuracy of unigrams dropped compared to significant increase in the performance of SVM. However, overall accuracy results for unigrams still outperform bigrams. This is expected as well, since bigrams produce datasets that are sparser given the limit of 140 character of Twitter. Moreover, multinomial NB still outperforms other classifiers. We deduce that MNB is less affected by the data sparsity problem inherent in Twitter datasets

The experimental results show that Multinomial Naïve Bayes classifier outperformed other classifiers examined in the study in the context of Twitter sentiment analysis being less affected by the sparsity of Twitter dataset. Unigrams as a form of representing dataset feature proved to be more effective in the context of Twitter sentiment analysis as they produce less sparse datasets. From the experiments, they could not get proof on best choice for frequency vs. polarity representation of data. Finally, despite the strong capabilities of SVM, it generated the least accuracy results taking the longest processing time, it proved to be negatively affected by data sparsity, making it less preferable choice for Twitter sentiment analysis.

#### Paper 4:

the authors [4] explore the role of text pre-processing in sentiment analysis, and report on experimental results that demonstrate that with appropriate feature selection and representation, sentiment analysis accuracies using support vector machines (SVM) in this area may be significantly improved. The level of accuracy achieved is shown to be comparable to the ones achieved in topic categorisation although sentiment analysis is considered to be a much harder problem in the literature.

Table 2.6 compares the classifier performances resulting from the classification on both not pre-processed and preprocessed data for each of the features matrices (TF-IDF, FF, FP).

	<i>TF-IDF</i>		<i>FF</i>			<i>FP</i>		
	no pre-proc	pre-proc	no pre-proc1	no pre-proc2	pre-proc	no pre-proc1	no pre-proc2	pre-proc
<i>Accuracy</i>	78.33	81.5	72.7	76.33	83	82.7	82.33	83
<i>Precision</i>	76.66	83	NA	77.33	80	NA	80	82
<i>Recall</i>	79.31	80.58	NA	76.31	85.86	NA	83.9	83.67
<i>F-Measure</i>	77.96	81.77	NA	76.82	82.83	NA	81.9	82.82

They have reported extensive experimental results, showing that, appropriate text pre-processing methods including data transformation and filtering can significantly enhance the classifier's performance.

#### Paper 5:

The authors [5] see that social media are usually used to express opinions and feelings about companies, products, services, hobbies, politics, etc. Therefore, enterprises, organizations, governments, and different groups in general have shown interest in the opinions that users have for their activities. They are also interested to know the way users use these media, the communication behavior, and some users attributes such as gender, age, geographical location, political orientation, etc. In general, the main aim is to provide personalized services, particularized offers, or simply to know what people think about something in order to improve their activities.

The nature of these texts poses new challenges for researchers in Natural Language Processing (NLP). In some cases, the tweets are written with ungrammatical sentences with a lot of emoticons, abbreviations, specific terminology, slang, etc.

Therefore, the usual techniques of NLP must be adapted to these characteristics of the language, and new approaches must be proposed in order to successfully address this problem. NLP tools like POS taggers, parsers, or Named Entity Recognition (NER) tools usually fail when processing tweets because they generally are trained on grammatical texts and they perform poorly in micro-blogging texts.

They present a system for addressing the task of political tendency identification of Twitter users based on SA techniques. For each user, they collect all their tweets and they extract all the entities related to the political subject. Then, they automatically assign a polarity to these entities and define a political tendency metric that uses this entity polarity information combined with another tendency metric for classifying the political tendency of each user in four categories: Left, Right, Center, or Undefined. The evaluation of the system is performed on the General Corpus, a corpus of Spanish tweets provided by the organization of the TASS2013 workshop.

The system consists of 4 modules. The first module is the Pre-processing module, which performs the tokenization, lemmatization, and Named Entities recognition of the input tweet. A lemma reduction and a POS tagging process is also carried out in this module. The second module is optional. It allows us to obtain the polarity of the entities contained in the tweet. If we omitted this step the global polarity of the tweet is obtained. The third module is the Feature Extraction module, which selects the features from the pre-processed tweet (or from the segments of tweets) and obtains a feature vector. Some features require the use of a polarity lexicon of lemmas and words. To determine the best features, a tuning process is required during the training phase. The fourth module is the Polarity Classifier module, which uses a classifier. We used the NU-SVM algorithm (Scholkopf et al., 2000) from an external library called " LibSVM ", which is very efficient software for building SVM classifiers. It is easy to integrate this software with WEKA thus allowing us to use all of WEKA's features. We used the bag of words approach to represent each tweet as a feature vector that contains the frequency of the selected features of the training set.) to assign a polarity label to the tweet.



The evaluated system on the SA tasks defined at the TASS2013 workshop. Two different sub-tasks called 5-level and 3-level were proposed. Both sub-tasks differ only in the polarity granularity considered. The 5-level sub-task uses the labels N, N+, P, P+, and NEU. The 3-level sub-task uses the labels N, P, and NEU. In both sub-tasks, an additional label (NONE) was used to represent tweets with no polarity. The accuracy results obtained on the unseen data test were:  $62.88\% \pm 0.38\%$  for 5-level task and  $70.25\% \pm 0.36\%$  for 3-level task. This results outperformed all the approaches at TASS2013 workshop with statistical significance (with a 95% level of confidence). The official results ranged from 61.6% to 13.5% for the 5-level task and from 66.3% to 38.8% for the 3-level task. The F1 result obtained in the Sentiment Analysis at Entity level task was worse ( $F1=0.40$ ), but it still is the best result reported in the sentiment analysis at entity level task at TASS2013 competition.

Table 1.6 summarizes the experimental results of our proposal. The table includes both the overall results (Global) and the results for each one of the political tendencies (Left, Right, Centre, and Undefined). It also includes the distribution of the tendencies in the gold-standard (%Ref). For the global result, the precision and the recall are the same since each user in the test set had a tendency assigned and the task consist to assign a tendency to all the users.

Tendency	%Ref	Precision	Recall	$F_1$
Left	21.5	0.658	0.735	0.694
Centre	17.7	0.478	0.393	0.431
Right	39.9	0.786	0.698	0.739
Undefined	20.9	0.780	0.970	0.865
Global	100	0.709	0.709	0.709

Table 2.7: Experimental results obtained in the political tendency identification task of TASS2013

The result obtained by the system (0.709) is the best result reported so far for this corpus, to their knowledge. The tendency for what they get better results is the Undefined ( $F1=0.865$ ). They consider the political tendency of a user to be Undefined if he did not have any tweet that references any of the majority parties. This assumption may be too strict for common users, but it seems reasonable for the well-know users that form the test corpus. The tendency that system had more trouble identifying was Centre ( $F1=0.431$ ). The tendency of a user can be identified as Centre when he expressed -in his tweets- opinions about entities related to center parties, even when these opinions were negative. This is because the neutral value of Centre

entities. In addition, users with opinions on right and left parties with the same polarity may be identified as Centre, which can be wrong in many cases.

## **Paper 6:**

In this paper [6] The authors noticed that few works have investigated the impact of training set size on classification accuracy pertaining to sentiment analysis. Such insight into the effect of the training set is important because of the amount of time and effort required to label training examples. The contributions of this paper are primarily twofold: 1) to measure the effect of varying training set size on SVM and Naïve Bayes Sentiment classification accuracies and 2) to determine which ensemble type (i.e. 1 or 2) performs the best in terms of F-score and accuracy. The remainder of the paper is organized as follows.

The aim of the study was to study the effect of varying the training set size on the learning curves of both SVM and Naïve Bayes when used in twitter sentiment classification. In addition, they also examined the impact of the training set on different ensemble fusion types. To study these effects, they evaluated the different classifiers on ten training sets, each with different sizes, and then, trained the two ensembles with the training sets. The two ensembles developed are identical except in the fusion method. Ensemble 1 fused the output of an SVM and Naïve Bayes classifier using AND fusion and Ensemble 2 fuses the outputs the same classifiers using OR fusion. The effects of varying the training set sizes on Naïve Bayes, SVM, Ensemble 1, and Ensemble 2 were recorded. They will compare between the different classifiers using the classification accuracy and the F-score.

They have shown in this paper that changing the training set size did not significantly affect the sentiment classification accuracy when using SVM or Naïve Bayes. As a consequence, they concluded that in cases where the manual training set labelling is expensive, SVM and Naïve Bayes could be trained on 20% to 80% of the labelled training set and still produce acceptable classification accuracies. In terms of specific classifier performance, results indicate that SVM and Naïve Bayes classifiers in terms of classification accuracy, it was shown that SVM surpassed the Naïve Bayes when using certain training set sizes, but often times, they both had similar or equal accuracies at most training set sizes. In most cases the Naïve Bayes classifier could be used instead of SVM if a faster run time is needed due to certain system constraints. Naïve Bayes gave comparable accuracies to SVM with a faster run time. However, SVM has shown to

be more robust as a model when compared to Naïve Bayes as it had better F-scores on all datasets. Moreover, they developed two ensembles to test how varying the training set sizes on an ensemble that is comprised of an SVM and a Naïve Bayes classifiers affects their classification accuracies and F-scores.

Training Set	Classification Accuracy				F-Score			
	<i>SVM</i>	<i>Naïve Bayes</i>	<i>Ensemble 1</i>	<i>Ensemble 2</i>	<i>SVM</i>	<i>Naïve Bayes</i>	<i>Ensemble 1</i>	<i>Ensemble 2</i>
10%	73.27%	67.00%	73.27%	67.01%	0.7085	0.5092	0.7085	0.5092
20%	73.27%	73.27%	73.15%	73.40%	0.7053	0.6532	0.7047	0.6533
30%	74.55%	75.06%	75.06%	74.55%	0.7215	0.6885	0.7284	0.6794
40%	75.19%	75.32%	75.45%	75.06%	0.7295	0.6997	0.7344	0.6927
50%	75.06%	75.06%	75.45%	74.65%	0.7279	0.6919	0.7332	0.6850
60%	75.70%	75.58%	76.47%	74.81%	0.7332	0.7028	0.7439	0.6896
70%	75.44%	75.58%	75.70%	75.32%	0.7301	0.7021	0.7343	0.6966
80%	75.06%	75.19%	75.19%	75.06%	0.7226	0.6955	0.7252	0.6919
90%	76.47%	75.32%	76.34%	75.45%	0.7389	0.6974	0.7395	0.6954
100%	76.47%	74.68%	76.21%	74.94%	0.737	0.6892	0.7365	0.6882

Table 2.8 Classification accuracy and F-score

As a result, it was shown that by combining Naïve Bayes classifier with SVM using AND-Type fusion, the overall output has improved the classification accuracy and F-score of SVM. The main reason was that Naïve Bayes reversed some of the false positives produced by SVM. On the other hand, the ORing approach used in Ensemble 2 did not improve the overall accuracy of SVM.

## Paper 7:

This paper [7] mentioned that tweets are usually composed of incomplete, noisy and poorly structured sentences, irregular expressions, ill-formed words and non-dictionary terms. Before selection features, a series of pre-processing(e.g., removal stopwords, removal URLs, replace negations etc.) will be applied for reducing the amount of noise in the tweets. The pre-

processing are done extensively in existing approaches, especially in machine learning based approaches

The authors [6] evaluated the effects of various pre-processing methods on sentiment classification, including removing URLs, replacing negation, reverting repeated letters, removing stopword ,removing numbers and expanding acronym. They used two features models and four classifiers to identifying tweets sentiment polarity on five Twitter datasets. Experimental results show that the performance of sentiment classification obtains improvement after expanding acronym and replacing negation, while hardly change when removal URL and removal stopwords are applied.

Experimental results show that the same pre-processing method affects performance of sentiment classifiers similarly, while NB and RF classifiers are more sensitive than LR and SVM classifiers. One of factor that may affect results of sentiment classification is the choice of the sentiment classifier and the features used for classifier training.

## **2.2 Comparison with Proposed Project**

Many of these systems use machine learning as naive classifier, SVM which give accuracy about 70 % or a bit more, we used ITSM which gives us accuracy about 80% in addition to friendly and good user interface, easy for end users to interact with.

# Chapter 3

## Feasibility Study

### 3.1 Feasibility Analysis

A feasibility study is a preliminary study which investigates the information of prospective users and determines the resources requirements, costs, benefits and feasibility of proposed system. A feasibility study takes into account various constraints within which the system should be implemented and operated. In this stage, the resource needed for the implementation such as computing equipment, manpower and costs are estimated. The estimated are compared with available resources and a cost benefit analysis of the system is made. The feasibility analysis activity involves the analysis of the problem and collection of all relevant information relating to the project. The main objectives of the feasibility study are to determine whether the project would be feasible in terms of economic feasibility, technical feasibility and operational feasibility and schedule feasibility or not. It is to make sure that the input data which are required for the project are available. Thus we evaluated the feasibility of the system in terms of the following categories:

- Technical feasibility
- Operational feasibility
- Economic feasibility
- Schedule feasibility

---

#### 3.1.1 Technical Feasibility

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at the point in time there is no any detailed designed of the system, making it difficult to access issues like performance, costs (on account of the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis; understand the different technologies involved in the proposed system. Before commencing the project, we have to be very clear about what are the technologies that are to be required for the development of the new system. Is the required technology available? Our system "Twitty Mood" is technically feasible since all the required tools are easily available. Python and Php with Javascript can be easily handled. Although all tools seems to be easily available there are challenges too.

### 3.1.2 Operational Feasibility

Proposed project is beneficial only if it can be turned into information systems that will meet the operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to Implementation?

The proposed was to make a simplified web application. It is simpler to operate and can be used in any webpages. It is free and not costly to operate.

### 3.1.3 Economic Feasibility

Economic feasibility attempts to weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system. A simple economic analysis which gives the actual comparison of costs and benefits are much more meaningful in this case. In addition, this proves to be useful point of reference to compare actual costs as the project progresses. There could be various types of intangible benefits on account of automation. These could increase improvement in product quality, better decision making, and timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information.

This is a web based application. Creation of application is not costly.

### 3.1.4 Schedule Feasibility

A project will fail if it takes too long to be completed before it is useful. Typically, this means estimating how long the system will take to develop, and if it can be completed in a given period of time using some methods like payback period. Schedule feasibility is a measure how reasonable the project timetable is. Given our technical expertise, are the project deadlines reasonable? Some project is initiated

with specific deadlines. It is necessary to determine whether the deadlines are mandatory or desirable.

A minor deviation can be encountered in the original schedule decided at the beginning of the project. The application development is feasible in terms of schedule.

---

## 3.2 Requirement Definition

After the extensive analysis of the problems in the system, we are familiarized with the requirement that the current system needs. The requirement that the system needs is categorized into the functional and non-functional requirements. These requirements are listed below:

### 3.2.1 Functional Requirements

Functional requirement are the functions or features that must be included in any system to satisfy the business needs and be acceptable to the users. Based on this, the functional requirements that the system must require are as follows:

- System should be able to process the new tweets stored after retrieval
- System should be able to analyze data and classify each tweet polarity

### 3.2.2 Non-Functional Requirements

Non-functional requirements is a description of features, characteristics and attribute of the system as well as any constraints that may limit the boundaries of the proposed system.

The non-functional requirements are essentially based on the performance, information, economy, control and security efficiency and services.

Based on these the non-functional requirements are as follows:

- User friendly
- System should provide better accuracy
- To perform with efficient throughput and response time

# Chapter 4

## 4. System Design and Architecture

### 4.1 Introduction

The purpose of this chapter is to represent a software detailed design about the project requirements and it gives a complete illustration about the functions and users scenarios where each function is clearly illustrated.

### 4.2 Diagrams

#### 4.2.1 Activity Diagram

The activity diagram in figure 1 shows the activity in the whole system of our twitter sentiment analysis system, the user open the system and search for a certain topic the system extracts the data from twitter, it process the tweets and passes it to the model.

The model sends the results back to the system, and the system display the results in charts for the user.



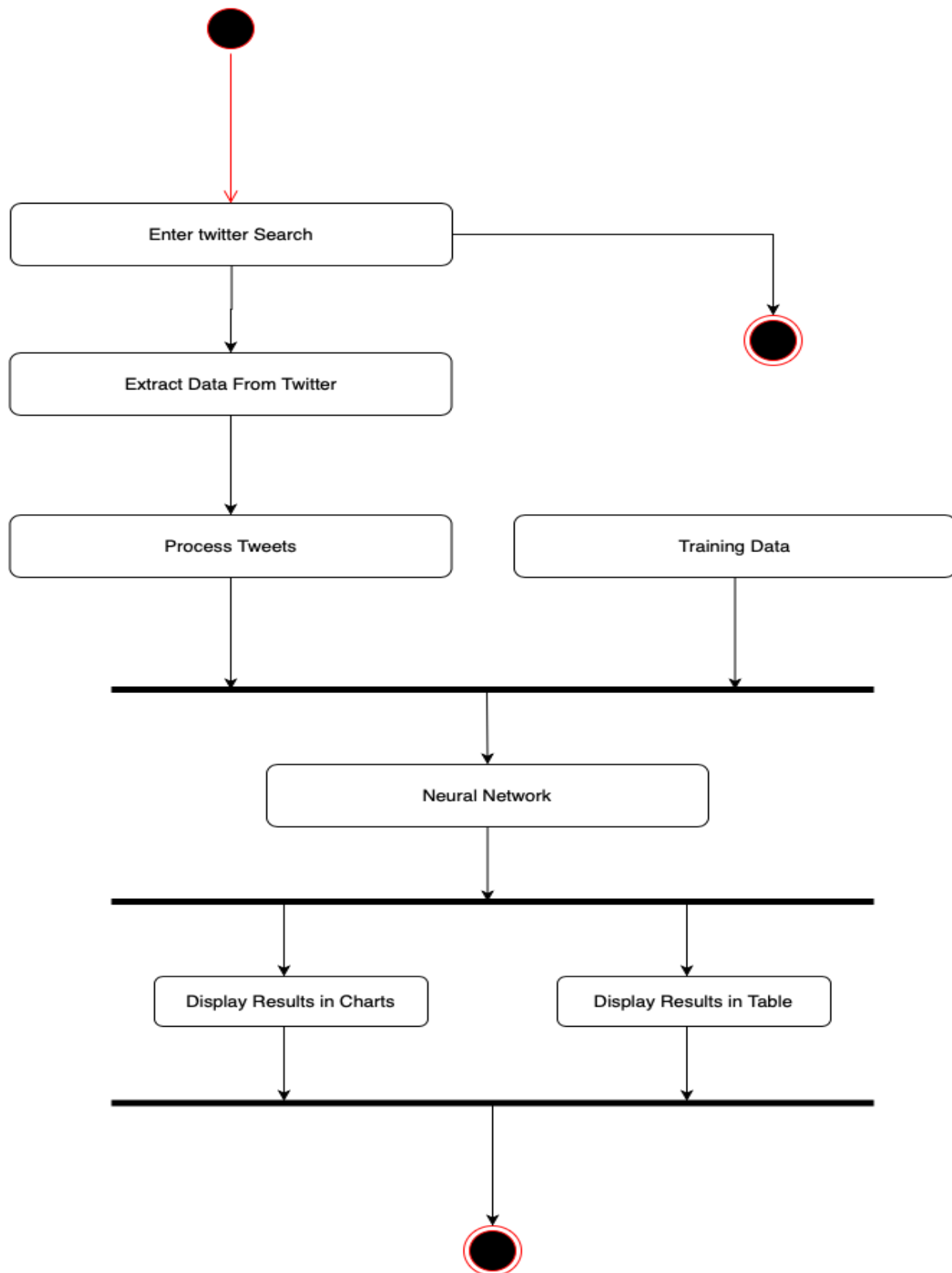


Figure 1 Activity Diagram

## 4.2.2 Use Case Diagram

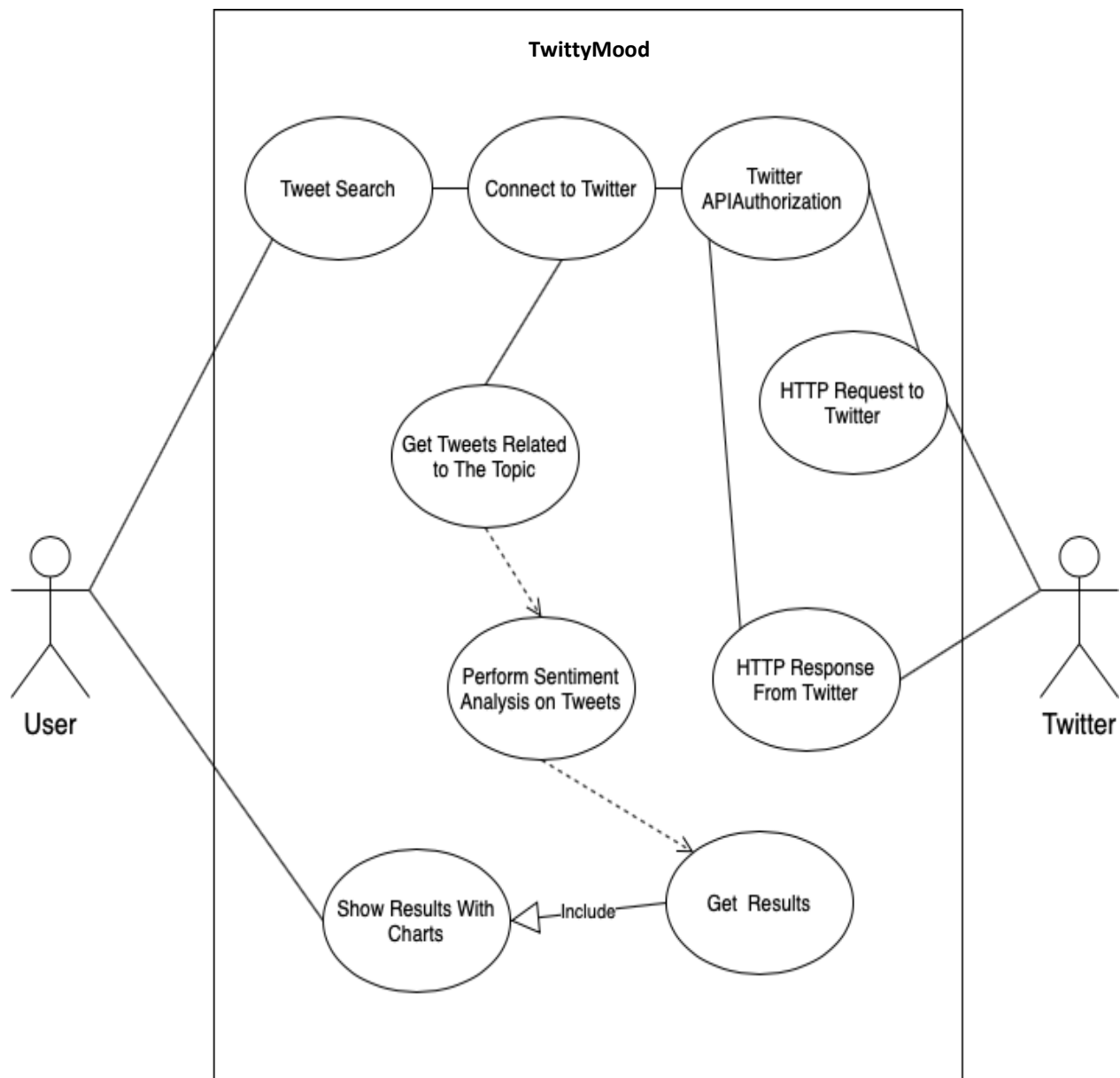


Figure 2 Use Case Diagram

### Description:

In the Use Case diagrams in figure 2 the user interacts with the system when he searches for a certain subject. The system connects to twitter using TWINT and get the related tweets. The system performs sentiment analysis on the tweets using the model and it returns to the user the results in diagrams with further information about the tweets.

#### 4.2.3 System Flow Diagram

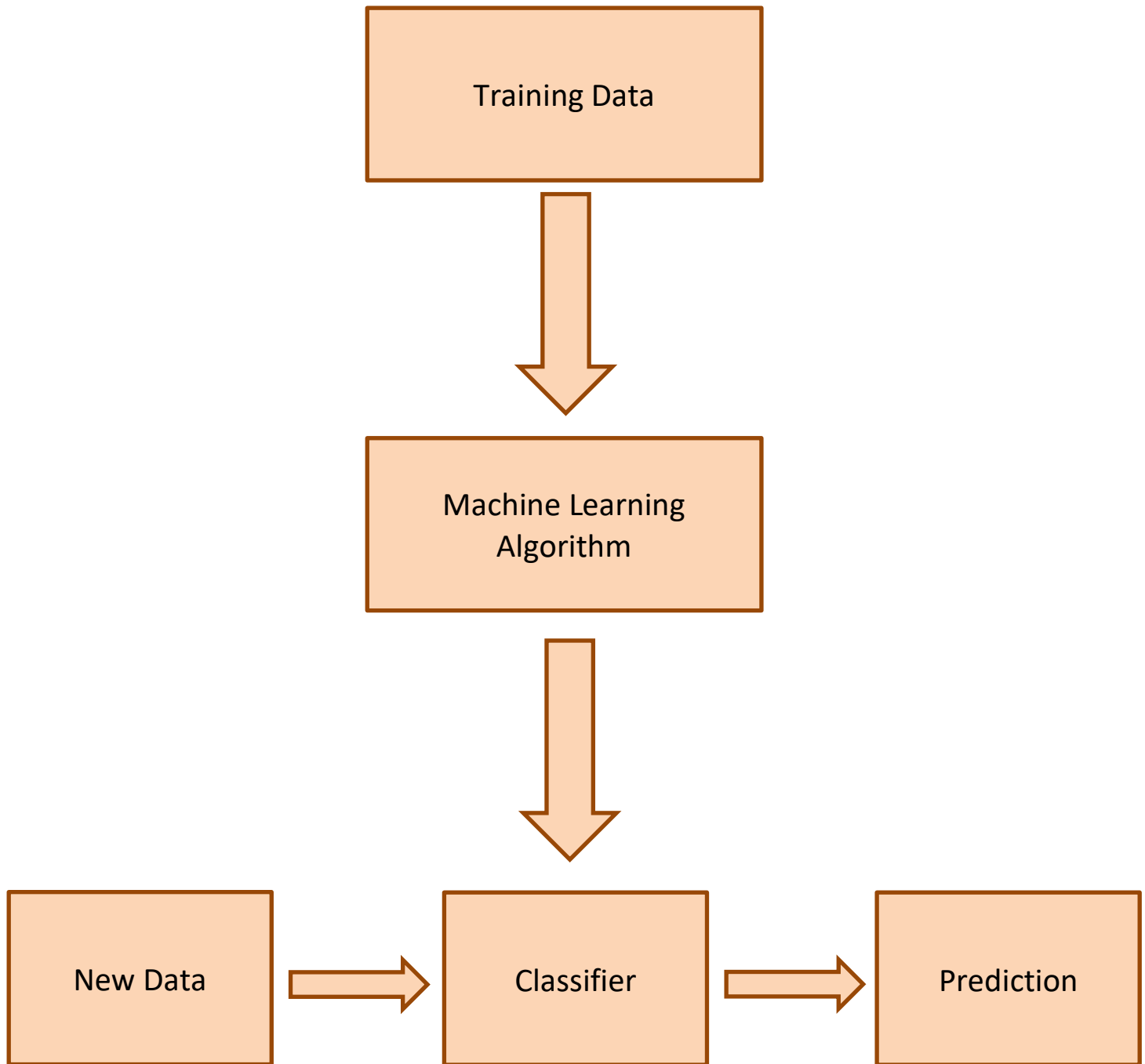


Figure 3 System Flow Diagram

#### 4.2.5 Data Flow Diagram

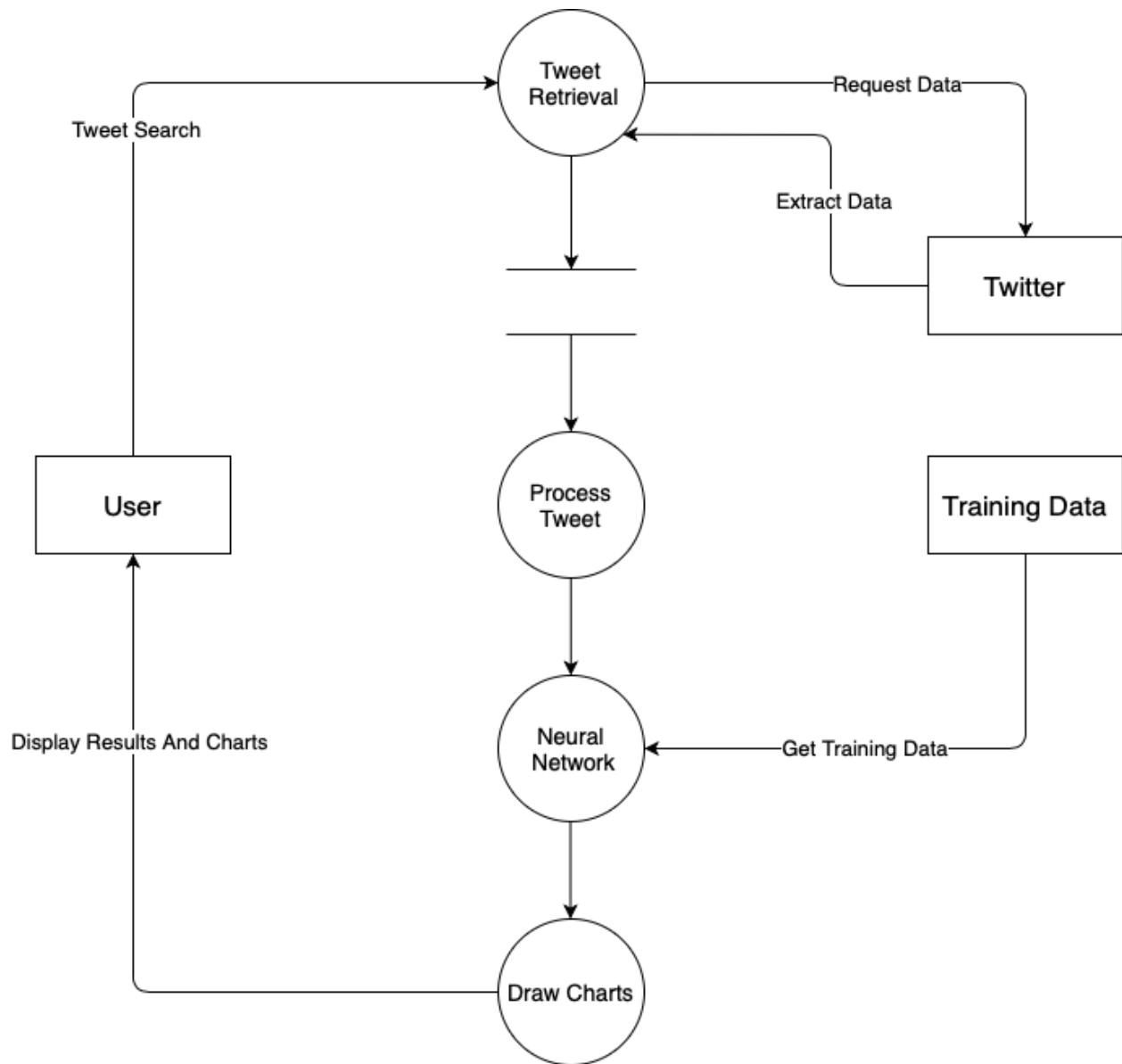


Figure 4 Dataflow Diagram

#### 4.2.5 Sequence Diagram

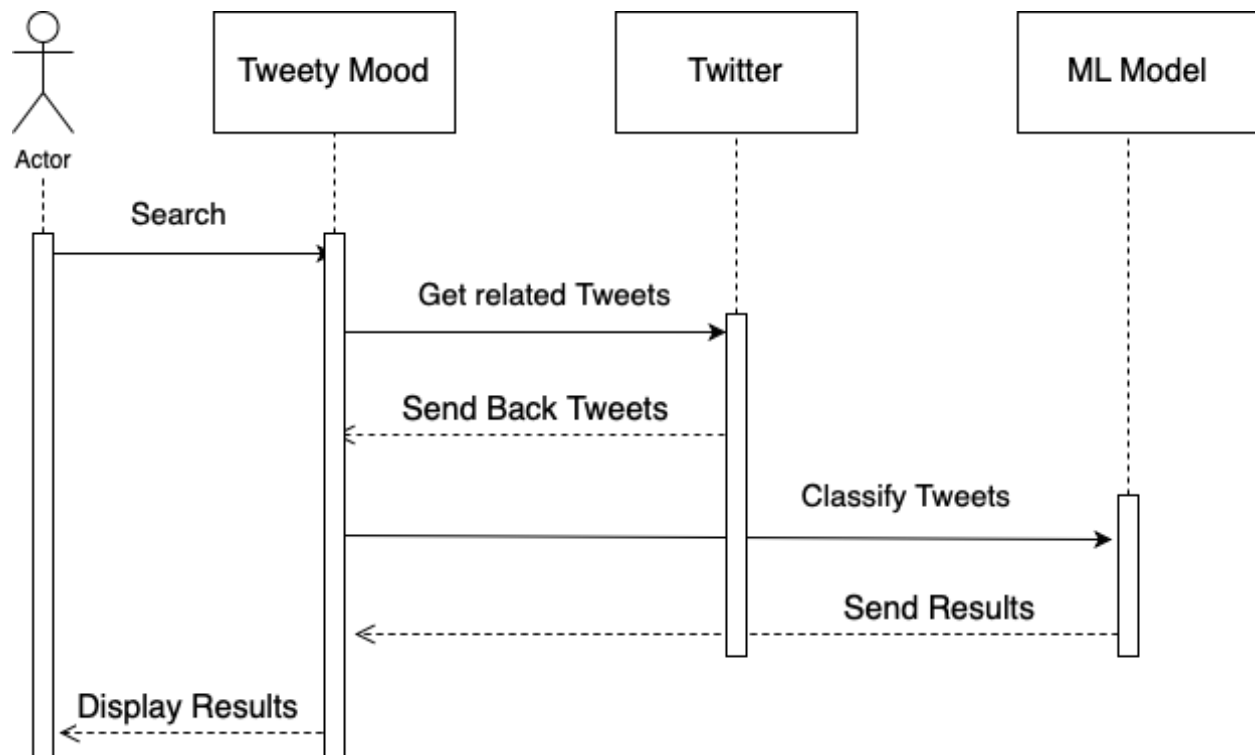


Figure 5 Sequence Diagram

#### Description:

In the sequence diagrams in figure 5 it shows that the goal of the system is to start to get the sentiment on tweets after the user searches for them, so it sends request to twitter to get the data and twitter response with the related tweets, it asks the model for classifying the data, then the model response with the results to the system, and last the system displays the results to the user.

#### 4.2.6 Flow Chart Diagram

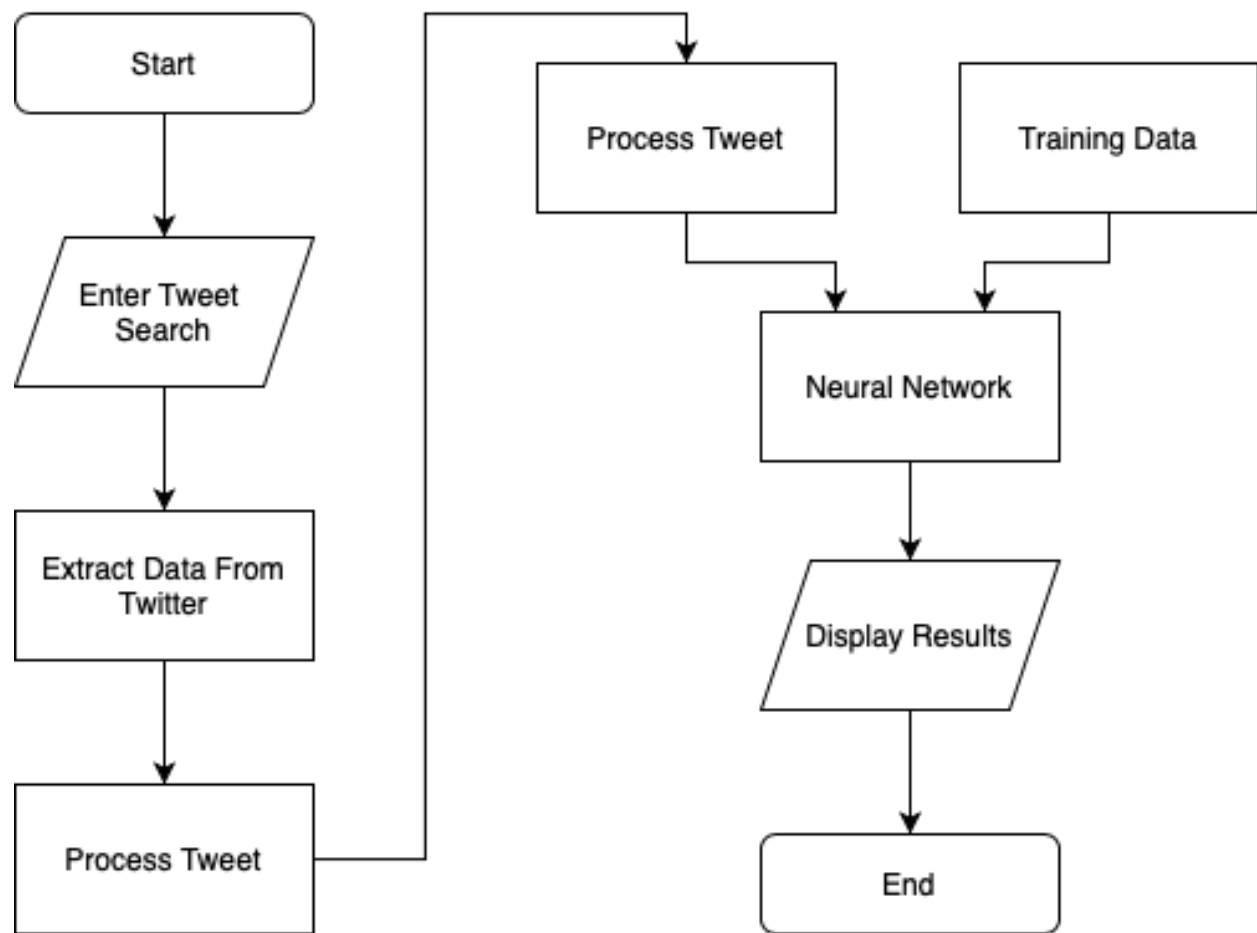


Figure 6 Flow Chart Diagram

# Chapter 5

## Implementation

### 5.1 Introduction

The purpose of this chapter is to clarify how Natural Language Processing and Sentiment Analysis work, what are the algorithms we used and why. This chapter also gives some information about very important and useful libraries which are used in this application.

### 5.2 Natural Language Processing

Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data. Some NLP techniques we used are:

#### Tokenization

It is the process of segmenting running text into sentences and words. In essence, it's the task of cutting a text into pieces called tokens, and at the same time throwing away certain characters, such as punctuation.

#### Stop Words Removal

Includes getting rid of common language articles, pronouns and prepositions such as “and”, “the” or “to” in English. In this process some very common words that appear to provide little or no value to the NLP objective are filtered and excluded from the text to be processed.

#### Stemming

Refers to the process of slicing the end or the beginning of words with the intention of removing affixes to reach the root of those words.

### 5.3 Sentiment Analysis

The process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive, negative, or neutral. Organizations can determine what customers are saying about a service or product by identifying and extracting information in sources like social media. This sentiment analysis can provide a lot of information about customers' choices and their decision drivers.

In this project we create a Recurrent Neural Network (RNN) using Python to implement NLP and the “Tensorflow Keras” library to apply the sentiment analysis.

The steps to computer coding sentiment analysis are:

- Preprocessing and tokenize the training dataset
  - Training a Word2Vec model
  - Create embedding layer
  - Build model and fit train data
  - Fetch tweets using TWINT library
  - Feed new data and evaluate prediction
- 

## 5.4 Algorithms

### 5.4.1 RNN and LSTM

Recurrent neural networks (RNN) are a class of neural networks that is powerful for modeling sequence data such as natural language. Schematically, a RNN layer uses a for loop to iterate over the timesteps of a sequence, while maintaining an internal state that encodes information about the timesteps it has seen so far.

One robust RNN architecture is the Long Short-term Memory (LSTM) model which incorporates a gating mechanism (with three gates) to control the memory retention operation. In the present paper, the LSTM implementation in the TensorFlow library was employed.

### 5.4.2 Entropy Loss

Information entropy is an evaluation measure that yields the intrinsic uncertainty of a process, represented by a random Variable  $X$ . Assuming that  $X$  has  $n \in \mathbb{N}$  possible different outcomes, and a probability distribution  $P(X)$  is assigned to  $X$ , the entropy of  $X$  with respect to  $P(X)$  is defined as :

$$H_{\beta}(X) := - \sum_{i=1}^n p_i \log_{\beta} p_i,$$

where  $p_i = P(X = x_i)$  for  $i \in \{1 \dots n\}$ .



### 5.4.3 Optimizer

Adam is an adaptive learning rate optimization algorithm that's been designed specifically for training deep neural networks. Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first order and second order moments. The algorithm leverages the power of adaptive learning rates methods to find individual learning rates for each parameter.

---

### 5.5 Difficulties

1- the corpus is a set of positive, negative and neutral tweets, contain emotions, non ascii characters, hashtags, the ambiguity of some word meanings that the same word has many different meanings and Identifying subjective parts of text: Subjective parts represent sentiment-bearing content. The same word can be treated as subjective in one case, or an objective in some other. This makes it difficult to identify the subjective portions of text.

Example:

1. The language of the Mr. Dennis was very crude.
2. Crude oil is obtained by extraction from the sea beds. The word "crude" is used as an opinion in first example, while it is completely objective in the second example also some non-standard English words like happyyy , saaaad which makes the computer not give accurate classification for the input , also the tokenization of statements which split words which may relate to gather to give a meaning like "not like" it may classify it wrong .

2- Domain dependence: The same sentence or phrase can have different meanings in different domains. Example: The word "unpredictable" is positive in the domain of movies, dramas, etc. But if the same word is used in the context of a vehicle's steering, then it has a negative opinion.

Also, Sarcasm Detection: Sarcastic sentences express negative opinion about a target using positive words in unique way. Example: "Nice perfume. You must shower in it." The sentence contains only positive words but actually it expresses a negative sentiment.

another problem is the great size of data which is 1.6 million tweets which its preprocessing and the layers of the model took long time for training on the corpus about 4 hours, so we saved the trained model so there is no need for training it again

## 5.6 Tools

### Python:

In our project we used python it is simple to develop, and it is good for natural language processing. Python libraries TensorFlow, Keras, Genism Word2Vec, TWINT, NLTK, Numpy and Scipy.

### TWINT:

TWINT is an advanced Twitter scraping tool written in Python that allows for scraping Tweets from Twitter profiles without using Twitter's API.

We use it to filter tweets and get the input data to perform the analysis. We chose this module over Twitter's API for various reasons, such as:

- Can fetch almost all Tweets (Twitter API limits to last 3200 Tweets only);
  - Fast initial setup;
  - Can be used anonymously and without Twitter sign up;
  - No rate limitations.
- 

## 5.7 Dataset

This is the sentiment140 dataset. It contains 1,600,000 tweets extracted using the twitter api. The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment.

### Content:

It contains the following 6 fields:

1. target: the polarity of the tweet (0 = negative, 4 = positive)
2. ids: The id of the tweet
3. date: the date of the tweet
4. flag: The query. If there is no query, then this value is NO\_QUERY.
5. user: the user that tweeted
6. text: the text of the tweet


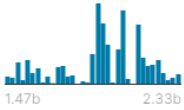
# 0	# 1467810369	Mon Apr 06 22:1...	NO_QUERY	_TheSpecialOne_	@switchfoot htt...
target	id	date	flag	user	text
		774362 unique values	1 unique value	659775 unique values	1581465 unique values
0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by texting it... and might cry as a result School today ...
0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Managed to save 50% The rest go out of bounds
0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all. i'm mad. why am i here? because I can't see you all o...

Figure 7 simple of the data set

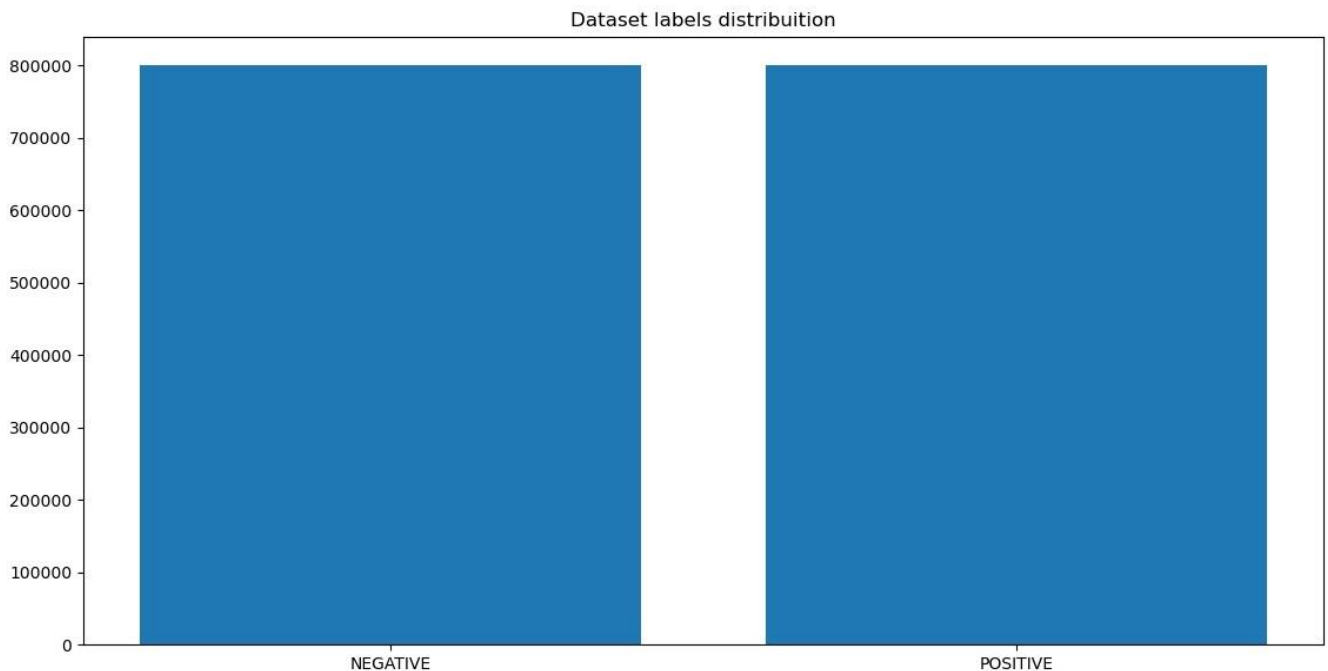


Figure 8 Dataset Labels Distribution

## 5.8 Results

Accuracy:

79 %

Classification report:

	precision	recall	f1-score	support
NEGATIVE	0.80	0.78	0.79	159494
POSITIVE	0.78	0.80	0.79	160506
micro avg	0.79	0.79	0.79	320000
macro avg	0.79	0.79	0.79	320000
weighted avg	0.79	0.79	0.79	320000

Figure 9 Classification Report

Confusion matrix :

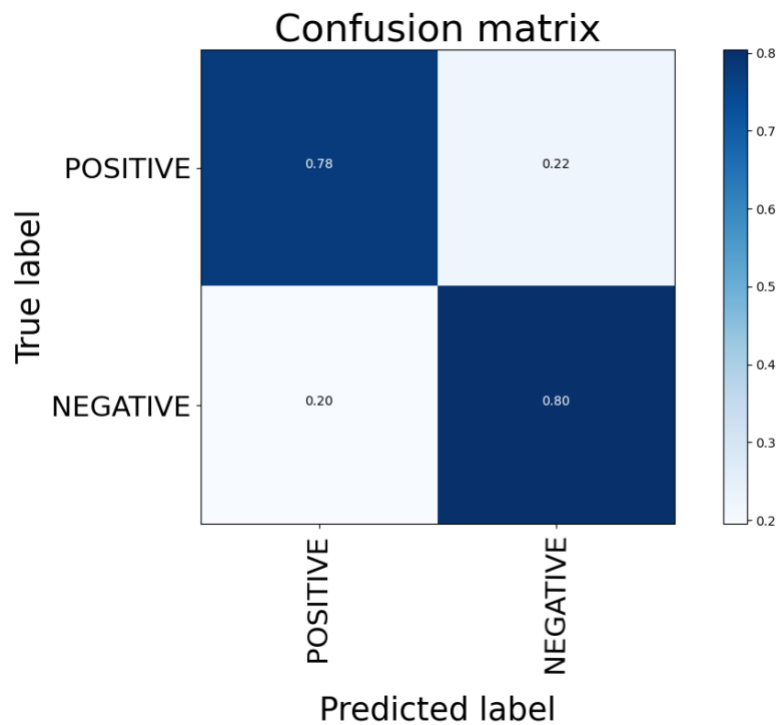


Figure 10 Confusion Matrix

## 5.9 GUI and Code

### GUI

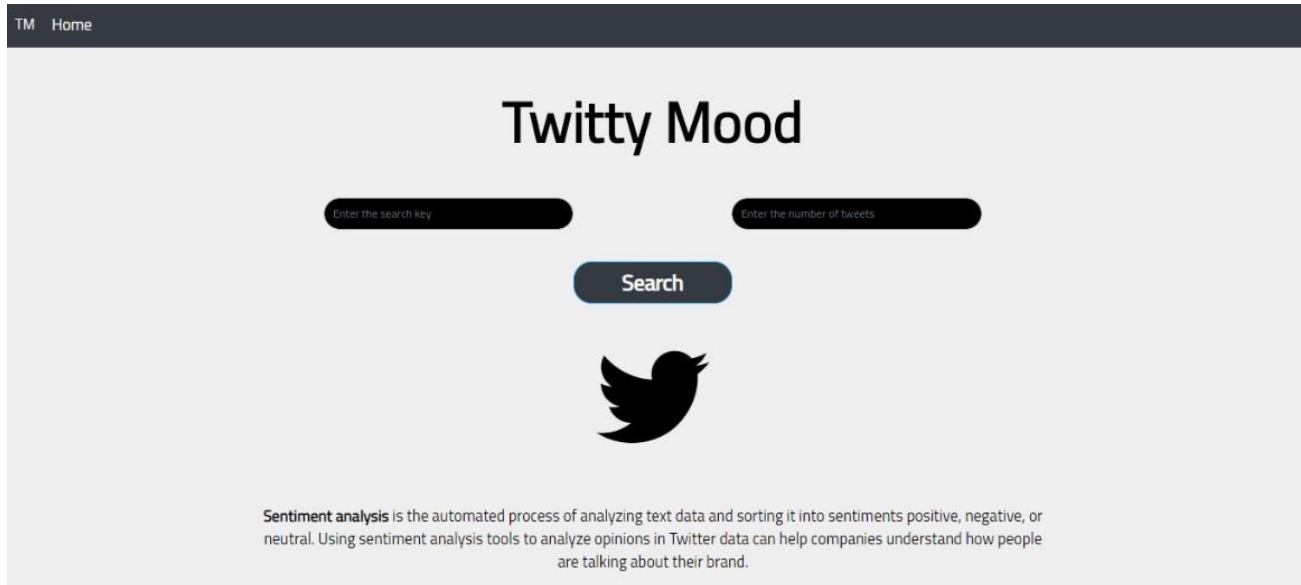


Figure 11 GUI Home

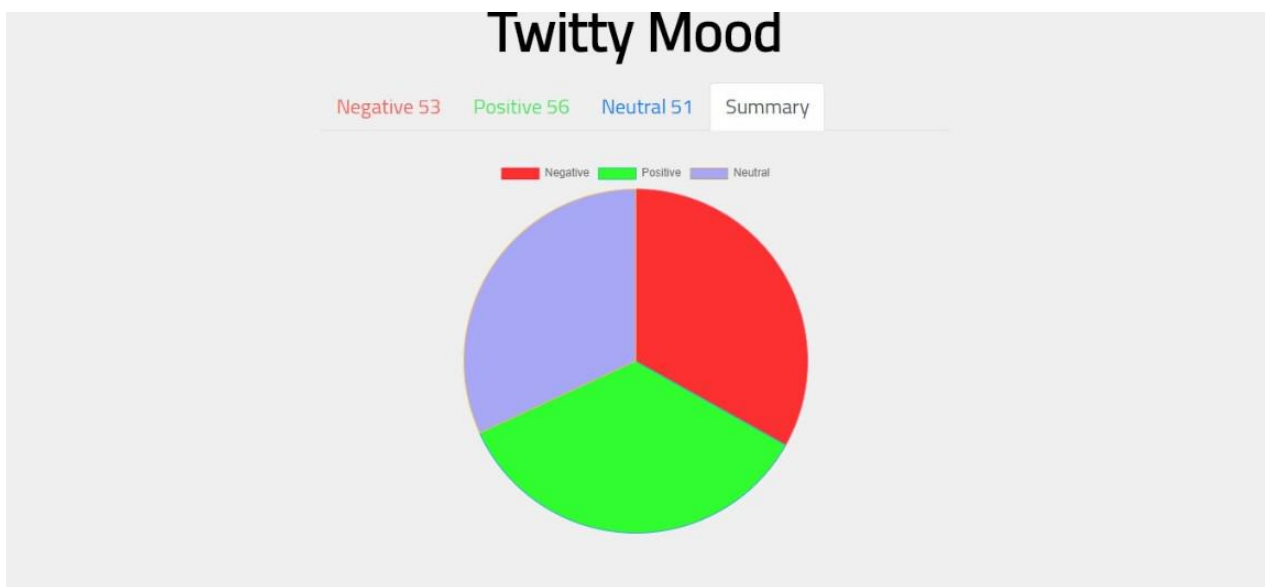


Figure 12 GUI Pie Chart

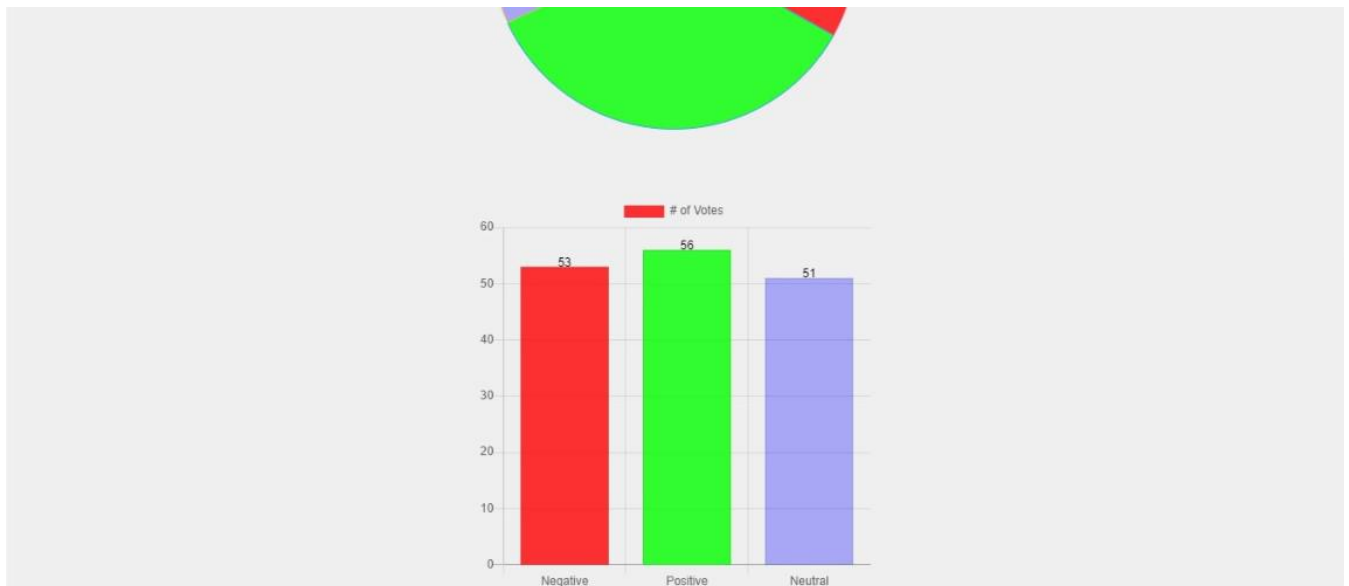


Figure 13 GUI Bar Chart

TM
HOME

## Twitty Mood

Negative 53
Positive 56
Neutral 51
Summary

POSITIVE

**Username :** undercoverindi

**Tweet:** Fun fact: On June 7th 1975 Sony introduced the Betamax videocassette recorder for sale to the public.

❤️❤️

pic.twitter.com/1NZMFxTkWF

2020-07-26 [Go see it](#)

POSITIVE

**Username :** bauer1596

**Tweet:** Nah personally I'm not that hyped for Cyberpunk but everyone who has played it early said it's amazing. So I think it could definitely win GOTY.

2020-07-26 [Go see it](#)

POSITIVE

**Username :** jrgabledon

**Tweet:** Remake Jak and Daxter. All of them

2020-07-26 [Go see it](#)

Figure 14 GUI Tweets

## Model Training Code:

- **Preprocessing**

```
[ ] stop_words = stopwords.words("english")
    stemmer = SnowballStemmer("english")

[ ] def preprocess(text, stem=False):
    # Remove link,user and special characters
    text = re.sub(TEXT_CLEANING_RE, ' ', str(text).lower()).strip()
    tokens = []
    for token in text.split():
        if token not in stop_words:
            if stem:
                tokens.append(stemmer.stem(token))
            else:
                tokens.append(token)
    return " ".join(tokens)

[ ] %%time
    df.text = df.text.apply(lambda x: preprocess(x))
```

Figure 15 Preprocessing Code

- **Word2Vec**

```
[ ] %%time
    documents = [_text.split() for _text in df_train.text]

[ ] w2v_model = gensim.models.word2vec.Word2Vec(size=W2V_SIZE,
                                                window=W2V_WINDOW,
                                                min_count=W2V_MIN_COUNT,
                                                workers=8)

[ ] w2v_model.build_vocab(documents)

[ ] words = w2v_model.wv.vocab.keys()
    vocab_size = len(words)
    print("Vocab size", vocab_size)

[ ] %%time
    w2v_model.train(documents, total_examples=len(documents), epochs=W2V_EPOCH)

[ ] w2v_model.save(WORD2VEC_MODEL)
```

Figure 16 Word2Vec Code

- **Tokenize text**

```
[ ] %%time
tokenizer = Tokenizer()
tokenizer.fit_on_texts(df_train.text)

vocab_size = len(tokenizer.word_index) + 1
print("Total words", vocab_size)

[ ] %%time
x_train = pad_sequences(tokenizer.texts_to_sequences(df_train.text), maxlen=SEQUENCE_LENGTH)
x_test = pad_sequences(tokenizer.texts_to_sequences(df_test.text), maxlen=SEQUENCE_LENGTH)

[ ] pickle.dump(tokenizer, open(TOKENIZER_MODEL, "wb"), protocol=0)
```

Figure 17 Tokenize Text Code

- **Embedding layer, Build and Compile Model**

```
[ ] embedding_matrix = np.zeros((vocab_size, W2V_SIZE))
for word, i in tokenizer.word_index.items():
    if word in w2v_model.wv:
        embedding_matrix[i] = w2v_model.wv[word]
print(embedding_matrix.shape)

[ ] embedding_layer = Embedding(vocab_size, W2V_SIZE, weights=[embedding_matrix], input_length=SEQUENCE_LENGTH, trainable=False)
```

Figure 18 Embedding Layer Code

```
[ ] model = Sequential()
model.add(embedding_layer)
model.add(Dropout(0.5))
model.add(LSTM(100, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))

model.summary()
```

Compile model

```
[ ] model.compile(loss='binary_crossentropy',
                  optimizer="adam",
                  metrics=['accuracy'])
```

Figure 19 Build and Compile Code



- **Train and Evaluate**

```
[ ] %%time
history = model.fit(x_train, y_train,
                    batch_size=BATCH_SIZE,
                    epochs=EPOCHS,
                    validation_split=0.1,
                    verbose=1,
                    callbacks=callbacks)

[ ] model.save(KERAS_MODEL)

[ ] %%time
score = model.evaluate(x_test, y_test, batch_size=BATCH_SIZE)
print()
print("ACCURACY:", score[1])
print("LOSS:", score[0])
```

Figure 20 Train and Evaluate Code

- **Decode Sentiment and perform prediction on input data**

```
[ ] def decode_sentiment(score, include_neutral=True):
    if include_neutral:
        label = NEUTRAL
        if score <= SENTIMENT_THRESHOLDS[0]:
            label = NEGATIVE
        elif score >= SENTIMENT_THRESHOLDS[1]:
            label = POSITIVE

    return label
    else:
        return NEGATIVE if score < 0.5 else POSITIVE

[ ] def predict(text, include_neutral=True):
    start_at = time.time()
    # Tokenize text
    x_test = keras.preprocessing.sequence.pad_sequences(tokenizer.texts_to_sequences([text]), maxlen=SEQUENCE_LENGTH)
    # Predict
    score = model.predict([x_test])[0]
    # Decode sentiment
    label = decode_sentiment(score, include_neutral=include_neutral)

    return {"label": label, "score": float(score),
            "elapsed_time": time.time()-start_at}
```

Figure 21 Decode Sentiment and Perform Prediction Code

# Chapter 6

## Conclusion

### 6.1 Conclusion

The experimental studies performed through the chapters, successfully show that neural network long short term memory is more accurate than Naïve classifier for sentiment classification, comparatively outperforming accurate results. For the datasets used, we recorded consistent accuracy of almost 70% for naïve classifier, 80% for LSTM. The first method that we approached for our problem is Naïve Bayes. It is mainly based on the independence assumption. Training is very easy and fast In this approach each attribute in each class is considered separately. Testing is straightforward, calculating the conditional probabilities from the data available. One of the major task is to find the sentiment polarities which is very important in this approach to obtain desired output. In the Naïve Bayes approach we only considered the words that are available in our dataset and calculated their conditional probabilities. We have obtained successful results after applying this approach to our problem.

But the LSTM deep learning approach that contain 3 layers that extracted more features, gives us a higher accuracy which is 80%.

### 6.2 Future work

We're looking forward enhancing our application with newest and stronger algorithms and classifiers that will extract new features, also Interpreting Sarcasm: The proposed approach is currently incapable of interpreting sarcasm. In general sarcasm is the use of irony to mock or convey contempt, in the context of current work sarcasm transforms the polarity of an apparently positive or negative utterance into its opposite. This limitation Conclusion and Future Work can be overcome by exhaustive study of fundamentals in discourse-driven sentiment analysis. The main goal of this approach is to empirically identify lexical and pragmatic factors that distinguish sarcastic, positive and negative.

We also will create a model to classify both the positive and negative tweets by the topic, or what the user like or dislike in the product returning what is the negative tweets is about , eg user commented “the camera quality is low , I dislike it ” it will return (negative tweet , camera).

And finally we can use a financial support in the future to enhance our application to support long term analysis that enables our user to do analysis for a long period of time in the same key word to determine the behavior of the curve of their performance.

---

### **References:**

- [1 ] Olga Kolchyna , Tharsis T. P. Souza , Philip C. Treleaven and Tomaso Aste “Twitter Sentiment Analysis: Lexicon Method, Machine Learning Method and Their Combination”
- [2] Rafeeqe Pandarachalil • Selvaraju Sendhilkumar • G. S. Mahalakshmi “ Twitter Sentiment Analysis for Large-Scale Data: An Unsupervised Approach”
- [3] Heba M. Ismail, Saad Harous, Boumediene Belkhouche “ A Comparative Analysis of Machine Learning Classifiers for Twitter Sentiment Analysis”
- [4] Emma HaddiXiaohui LiuYong Shi ,” The Role of Text Pre-processing in Sentiment Analysis”
- [5] Ferran Pla and Llu'is-F. Hurtado,” Political Tendency Identification in Twitter using Sentiment Analysis Techniques”
- [6] Omar Abdelwahab , Mohamed Bahgat , Christopher J. Lowrance , Adel Elmaghraby,” Effect of Training Set Size on SVM and Naïve Bayes for Twitter Sentiment Analysis”
- [7] Zhao jianqiang, “ Pre-processing Boosting Twitter Sentiment Analysis?”