Sentiment Analysis of Twitter data

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

**BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING**

BY

**Aditi Bhalawat**

**EN18CS301011**

Under the Guidance of

**Internal Guide -: Mr Ashish Kumawat**

**External Guide -: Mr. Tanuj Soni**



**Department of Computer Science & Engineering**

**Faculty of Engineering**

**MEDI-CAPS UNIVERSITY, INDORE- 453331**

**January 2022 - May 2022**

Report Approval

The project work **“Sentiment Analysis of Twitter Data”** is hereby approved as a creditable study of an engineering/computer application subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approve any statement made, opinion expressed, or conclusion drawn therein; but approve the “Project Report” only for the purpose for which it has been submitted.

Internal Examiner

Name:

Designation

Affiliation

External Examiner

Name:

Designation

Affiliation

Declaration

I hereby declare that the project entitled **“Sentiment Analysis of Twitter Data”** submittedin partial fulfillment for the award of the degree of Bachelor of Technology in ‘Computer Science and Engineering’ completed under the supervision of **<Name, designation and department of the Guide(s)>,** Faculty of Engineering, Medi-Caps University Indore is an authentic work.

Further, I/we declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

**Signature and name of the student(s) with date**

Certificate

I, **<Name of the Guide(s)>** certify that the project entitled **“<Title of the Project>”** submittedin partial fulfillment for the award of the degree of Bachelor of Technology by **<Name of the student(s)>** istherecordcarried out by him/them under my/our guidance and that the work has not formed the basis of award of any other degree elsewhere.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Name of Internal Guide>

<Name of the Department>

Medi-Caps University, Indore

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Name of External Guide (If any)>

<Name of the Department>

Name of the Organization

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

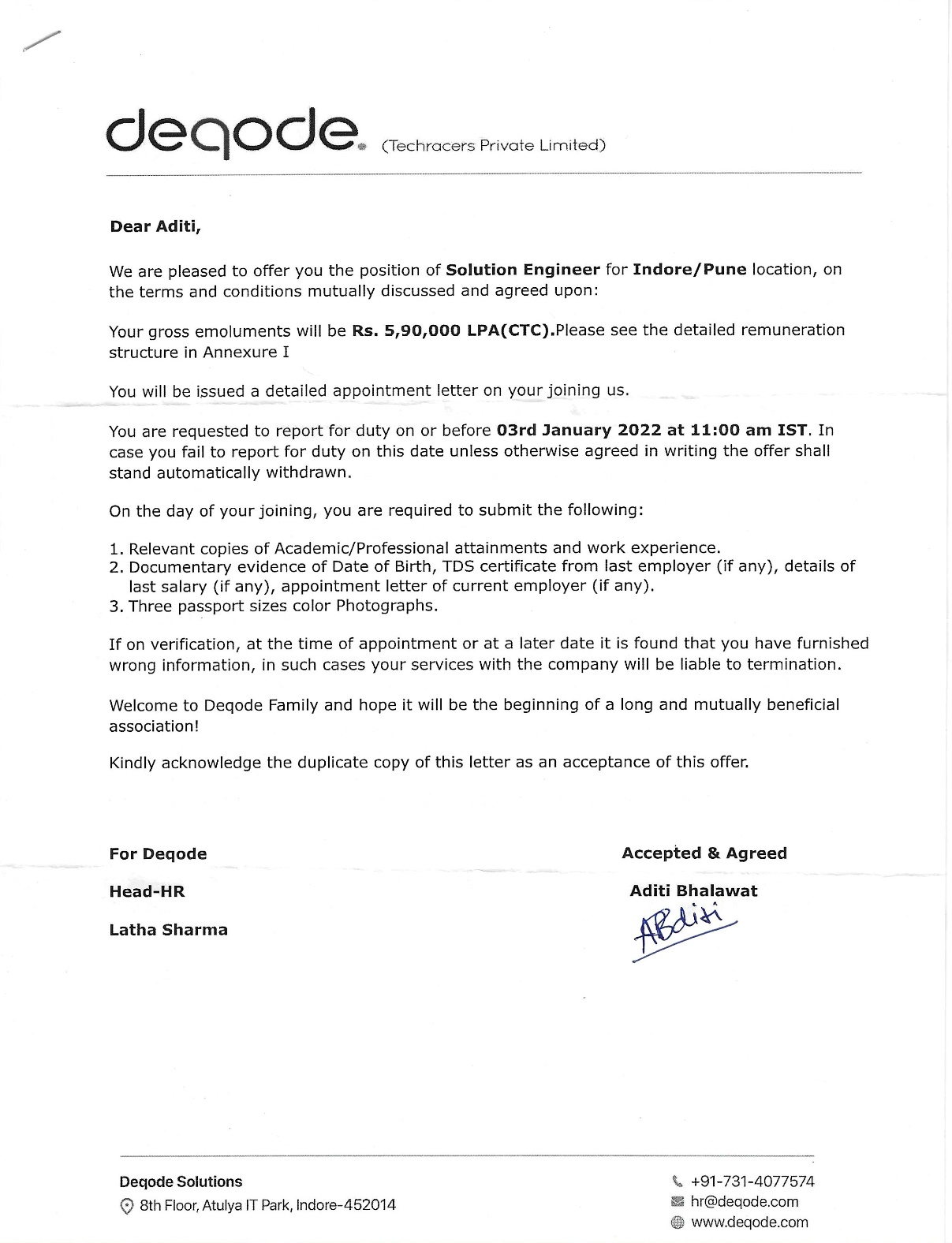
Dr. Pramod S. Nair

Head of the Department

Computer Science & Engineering

Medi-Caps University, Indore

Offer Letter of the Project work-II/Internship

****

Acknowledgements

I would like to express my deepest gratitude to Honorable Chancellor, **Shri R C Mittal,** who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) Dileep K Patnayak,** Vice Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) D K Panda,** Pro Vice Chancellor, **Dr. Suresh Jain,** DeanFaculty of Engineering, Medi-Caps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Dr. Pramod S. Nair** for his continuous encouragement for betterment of the project.

I express my heartfelt gratitude to my **External Guide, Mr. Tanuj Soni**, Project Lead, Deqode Pvt. Ltd as well as to my Internal Guide, Dr**. Debendra Kumar Panda,** Professor, Department of Electronics Engineering, MU, without whose continuous help and support, this project would ever have reached to the completion.

I would also like to thank my team at Deqode who extended their kind support and help towards the completion of this project.

It is their help and support, due to which we became able to complete the design and technical report. Without their support this report would not have been possible.

**Aditi Bhalawat**

B.Tech. VIII Year

Department of Computer Science & Engineering

Faculty of Engineering

Medi-Caps University, Indore

Abstract

Nowadays, people from all around the world use social media sites to share information. Twitter for example is a platform in which users send, read posts known as ‘tweets’ and interact with different communities. Users share their daily lives, post their opinions on everything such as brands and places. Companies can benefit from this massive platform by collecting data related to opinions on them. The aim of our project is to present a model that can perform sentiment analysis of real data collected from Twitter. Data in Twitter is highly unstructured which makes it difficult to analyze. However, our proposed model is different from prior work in this field because it combined the use of supervised and unsupervised machine learning algorithms.

The process of performing sentiment analysis is as follows: Tweet extracted directly from Twitter API, then cleaning and discovery of data performed. After that the data were fed into several models for the purpose of training. Each tweet is classified based on its sentiment whether it is positive, negative or neutral. Data was collected on crypto tokens and went through their tweets to see which one could grow or lose on the market.

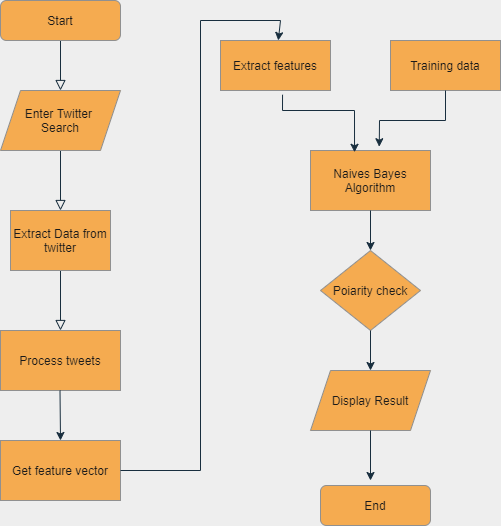
Different machine learning algorithms were used. The results from these models were tested using various testing metrics like cross validation and f-score. Moreover, our model demonstrates strong performance on mining texts extracted directly from Twitter.

Table of Contents

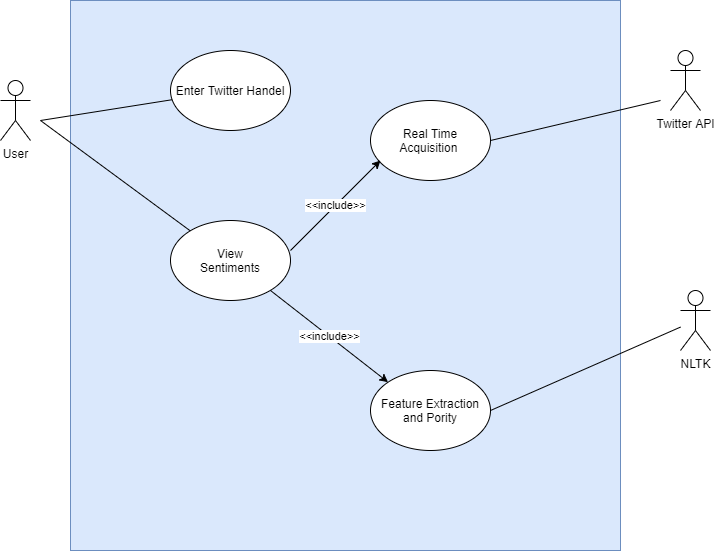
|  |  |  |
| --- | --- | --- |
|  |  | **Page No.** |
|  |  |  |
|  | Report Approval | ii |
|  | Declaration | iii |
|  | Certificate | iv |
|  | Offer Letter of the Project work-II/Internship | v |
|  | Completion letter/certificate | vi |
|  | Acknowledgement | vii |
|  | Abstract | viii |
|  | Table of Contents | ix |
|  | List of figures | x |
| Chapter 1 | Introduction |  |
|  | 1.1 Introduction | 1 |
|  | 1.2 Literature Review |  |
|  | 1.3 Objectives |  |
|  | 1.4 Scope |  |
|  | 1.5 Feasibility Study |  |
|  | 1.6 Source of Data |  |
| Chapter 2 | Methodology |  |
|  | 2.1 Data Extraction and Cleaning |  |
|  | 2.2 Human Labeling |  |
|  | 2.3 Feature Extraction |  |
|  | 2.4 Model Building |  |
| Chapter 3 | Code Flow |  |
|  | 3.1 Pre Processing |  |
|  | 3.2 Data Cleaning |  |
|  | 3.3 Analysis Sentiment |  |
|  | 3.4 User interface |  |
| Chapter 4 | Results |  |
| Chapter 5 | Conclusions |  |
| Chapter 6 | References |  |

List of Figures

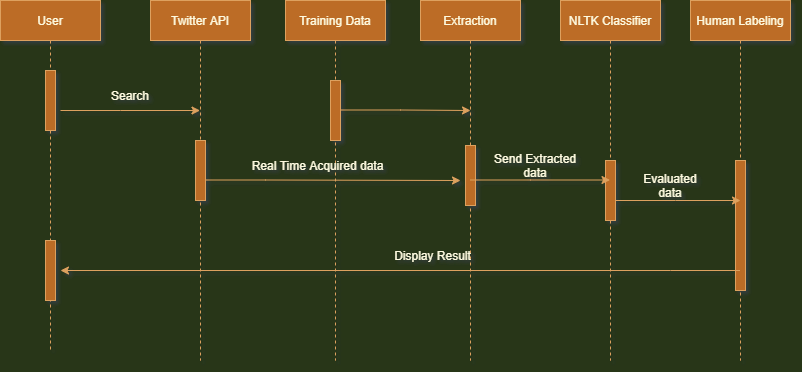
**Flow chart**



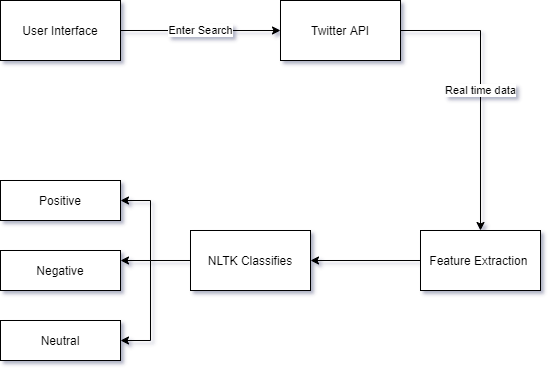
**Use Case Diagram**



**Sequence Diagram**



**Collaboration Diagram**



**Chapter-1**

Introduction

**Introduction**

Online social media such as Twitter, Facebook, and Instagram allow users to communicate with the whole world. Write their own opinions about products or share their moments, even influence politics and companies.

Twitter for example, almost every huge company has an account on Twitter to know about their customers' feedback about their services or products. Sentiment analysis, known as opinion mining, for classifying specific words into positive or negative.

In this project, we used sentiment analysis to classify specific English tweets about crypto tokens. our research was determining which one is better than the other. Specifically, we examined whether specific tweets are positive, negative, neutral.

This project contains implementation of Naive Bayes using sentiment140 training data using Twitter database and proposes a method to improve classification.

Use of Naive Bayes can improve accuracy of classification of tweets, by providing positivity, negativity and objectivity score of words present in tweets.

For actual implementation of this system python with NLTK and python-Twitter APIs are used. We'll use a logistic regression classifier, bag-of-words features, and polarity lexicons (both in-built and external).

**We'll also create our own pre-processing module to handle raw tweets:-**

**3-way classification**

It means the tweets after analyzing will be classified into Positive, Negative and Neutral.

**About Sentiment Analysis**

Sentiment analysis refers to the broad area of natural language processing which deals with the computational study of opinions, sentiments and emotions expressed in text.

**The challenge**

The challenge is to gather all such relevant data, detect and summarize the overall sentiment on a topic.

**Literature Review**

Sentiment analysis in the domain of micro-blogging is a relatively new research topic so there is still a lot of room for further research in this area. Decent amount of related prior work has been done on sentiment analysis of user reviews, documents, web blogs/articles and general phrase level sentiment analysis.

These differ from twitter mainly because of the limit of 140 characters per tweet which forces the user to express opinion compressed in very short text. The best results reached in sentiment classification use supervised learning techniques such as Naive Bayes and Support Vector Machines, but the manual labeling required for the supervised approach is very expensive. Some work has been done on unsupervised and semi-supervised approaches, and there is a lot of room of improvement.

Various researchers testing new features and classification techniques often just compare their results to base-line performance. There is a need for proper and formal comparisons between these results arrived through different features and classification techniques in order to select the best features and most efficient classification techniques for particular applications.

**Objectives**

**Classify the polarity**

The main objective of Twitter sentiment analysis is classifying the polarity of a given text and acquired tweet at the document, sentence or feature/aspect level.

**Empirical study**

It provides means for empirically studying properties of social interactions so that we can determine the attitude of the mass is positive, negative or neutral towards the subject of interest.

**Promote Research**

Classifier codes in python using NLTK that can be used in NLP in order to promote research that will lead to a better understanding of house sentiment is conveyed in tweets and texts.

**Scope**

Talking about the scope of this project this system can be helpful in various applications and various domains like review related websites for example for movie reviews, product reviews etc.

Also, in the subcomponent technology field like for detecting antagonistic, heated language in mails, spam detection, context sensitive information detection.

Also, in business and government intelligence by knowing the consumer attitudes and trends

And also, in getting the political feedback like knowing the public opinion for political leaders for their notions about rules and regulations in place etc.

**Feasibility study**

**Technical Feasibility**

Every bit of the code will be in python as all the libraries used from data processing to deployment are available with very descriptive documentation.

**Operational Feasibility**

This project is easy to operate and maintain. It uses a user-friendly web interface to easily work with it. It can be maintained by anyone who is fluent in python.

**Economic Feasibility**

To manage and maintain this project in inexpensive as compared to the operational facilitation it provides for the professionals which can save thousands of bucks and the most important entity i.e., time

**Source of Data:**

The Twitter API gives developers access to most of Twitter’s functionality. You can use the API to read and write information related to twitter entities such as tweets, users, and trends.

Technically, the API exposes dozens of HTTP endpoints related to:

* Tweets
* Retweets
* Likes
* Direct messages
* Favorites
* Trends
* Media

The Twitter API uses OAuth, a widely used open authorization protocol, to authenticate all the requests. Before making any call to the Twitter API, you need to create and configure your authentication credentials. Later in this article, you’ll find detailed instructions for this.

You can leverage the Twitter API to build different kinds of automations, such as bots, analytics, and other tools. Keep in mind that Twitter imposes certain restrictions and policies about what you can and cannot build using its API. This is done to guarantee users a good experience. The development of tools to spam, mislead users, and so on is forbidden.

The Twitter API also imposes **rate limits** about how frequently you’re allowed to invoke API methods. If you exceed these limits, you’ll have to wait between 5 and 15 minutes to be able to use the API again.

**Chapter-2**

Methodology

This project focuses on mining tweets written in English. We are interested in seeing who people think about crypto currency in terms of how good/bad reviews are. Analyzing people’s opinions and what they think about a product from their tweets on social media could be a valuable thing for any business. In our project, we extracted tweets from Twitter using python language. In order to extract tweets from Twitter, Twitter API was used to create Twitter applications and get authorization. By using tweepy package you can extract tweets up to 3200 only.

**Data Extraction and Cleaning:**

Data in the form of raw tweets is acquired by using the python library “twint” which provides a package for simple twitter streaming. This API allows two modes of accessing tweets: SampleStream and FilterStream. SampleStream simply delivers a small, random sample of all the tweets streaming at a real time. FilterStream delivers tweet which match a certain criteria. It can filter the delivered tweets according to three criteria:

• Specific keyword(s) to track/search for in the tweets

• Specific Twitter user(s) according to their user-id’s

• Tweets originating from specific location(s) (only for geo-tagged tweets).

A programmer can specify any single one of these filtering criteria or a multiple combination of these. But for our purpose we have no such restriction and will thus stick to the SampleStream mode.

Since we wanted to increase the generality of our data, we acquired it in portions at different points of time instead of acquiring all of it at one go. If we used the latter approach then the generality of the tweets might have been compromised since a significant portion of the tweets would be referring to some certain trending topic and would thus have more or less of the same general mood or sentiment. This phenomenon has been observed when we were going through our sample of acquired tweets.

For example the sample acquired near Christmas and New Year’s had a significant portion of tweets referring to these joyous events and were thus of a generally positive sentiment. Sampling our data in portions at different points in time would thus try to minimize this problem.

A tweet acquired by this method has a lot of raw information in it which we may or may not find useful for our particular application. It comes in the form of the python “dictionary” data type with various key-value pairs. A list of some key-value pairs are given below:

• Whether a tweet has been favorited

• User ID • Screen name of the user

• Original Text of the tweet

• Presence of hashtags

• Whether it is a retweet

• Language under which the twitter user has registered their account

• Geo-tag location of the tweet

• Date and time when the tweet was created

Since this is a lot of information, we only filter out the information that we need and discard the rest.

**Human Labeling:**

For the purpose of human labeling, we made three copies of the tweets so that they can be labeled by four individual sources. This is done so that we can take the average opinion of people on the sentiment of the tweet and in this way the noise and inaccuracies in labeling can be minimized. Generally speaking the more copies of labels, the better it is, but we have to keep the cost of labeling in our mind, hence we reached the reasonable figure of three.

We labeled the tweets in three classes according to sentiments expressed/observed in the tweets: positive, negative, neutral/objective and ambiguous.

We gave the following guidelines to our labelers to help them in the labeling process:

**Positive:** If the entire tweet has a positive/happy/excited/joyful attitude or if something is mentioned with positive connotations. Also if more than one sentiment is expressed in the tweet but the positive sentiment is more dominant. Example: “4 more years of being in shithole Australia then I move to the USA! :D”.

**Negative:** If the entire tweet has a negative/sad/displeased attitude or if something is mentioned with negative connotations. Also if more than one sentiment is expressed in the tweet but the negative sentiment is more dominant. Example: “I want an android now this iPhone is boring :S”.

**Neutral/Objective:** If the creator of the tweet expresses no personal sentiment/opinion in the tweet and merely transmits information. Advertisements of different products would be labelled under this category. Example: “US House Speaker vows to stop Obama contraceptive rule...

**Feature Extraction:**

Now that we have arrived at our training set we need to extract useful features from it which can be used in the process of classification. But first we will discuss some text formatting techniques which will aid us in feature extraction:

• Tokenization: It is the process of breaking a stream of text up into words, symbols and other meaningful elements called “tokens”. Tokens can be separated by whitespace characters and/or punctuation characters. It is done so that we can look at tokens as individual components that make up a tweet .

• Url’s and user references (identified by tokens “http” and “@”) are removed if we are interested in only analyzing the text of the tweet.

• Punctuation marks and digits/numerals may be removed if for example we wish to compare the tweet to a list of English words.

• Lowercase Conversion: Tweets may be normalized by converting it to lowercase which makes its comparison with an English dictionary easier.

• Stemming: It is the text normalizing process of reducing a derived word to its root or stem . For example a stemmer would reduce the phrases “stemmer”, “stemmed”, “stemming” to the root word “stem”. Advantage of stemming is that it makes comparison between words simpler, as we do not need to deal with complex grammatical transformations of the word. In our case we employed the algorithm of “porter stemming” on both the tweets and the dictionary, whenever there was a need for comparison.

• Stop-words removal: Stop words are a class of some extremely common words which hold no additional information when used in a text and are thus claimed to be useless . Examples include “a”, “an”, “the”, “he”, “she”, “by”, “on”, etc. It is sometimes convenient to remove these words because they hold no additional information since they are used almost equally in all classes of text, for example when computing prior-sentiment-polarity of words in a tweet according to their frequency of occurrence in different classes and using this polarity to calculate the average sentiment of the tweet over the set of words used in that tweet.

• Parts-of-Speech Tagging: POS-Tagging is the process of assigning a tag to each word in the sentence as to which grammatical part of speech that word belongs to, i.e. noun, verb, adjective, adverb, coordinating conjunction etc.

**Model Building:**

In this phase, after preparing a tweet (removing unnecessary symbols), each tweet was labeled as 1, -1, 0. (That’s it: positive, negative, or natural) using an unsupervised learning algorithm. Since we do not have pre-classified data, a lexicon-based model is used to classify tweets. By using two text files containing a list of positive and negative words, along with more words related to our domain. Each word within each tweet is compared to positive and negative documents in order to find matching words, and classify tweets whether it has more positive or negative words.

After that, multiple supervised learning algorithms applied for the purpose of training: Naive Bayes, support vector machine (SVM), maximum entropy, decision tree, random forest and bagging.

* Naïve Bayes: is defined as a classifier used to determine the most probable class label for each object.
* Support vector machine: is defined as a supervised model, used for classification, regression analysis.
* Maximum entropy: is a classifier used for a large variety of text classification.
* Decision tree: are flexible algorithms used to assign labels based on the highest score.
* Random forest: is a supervised algorithm for constructing multiple decision trees.
* Bagging: is a classifier used to take multiple random samples and use each sample separately to construct a prediction model.

For analyzing the tweets we used TextBlob.TextBlob is a python library and offers a simple API to access its methods and perform basic NLP tasks.

## NLP tasks using TextBlob

### Tokenization

Tokenization refers to dividing text or a sentence into a sequence of tokens, which roughly correspond to “words”. This is one of the basic tasks of NLP. To do this using TextBlob, follow the two steps:

1. Create a textblob object and pass a string with it.
2. Call functions of textblob in order to do a specific task.

### Noun Phrase Extraction

Since we extracted the words in the previous section, instead of that we can just extract out the noun phrases from the textblob. Noun Phrase extraction is particularly important when you want to analyze the “who” in a sentence.

### Part-of-speech Tagging

Part-of-speech tagging or grammatical tagging is a method to mark words present in a text on the basis of its definition and context. In simple words, it tells whether a word is a noun, or an adjective, or a verb, etc. This is just a complete version of noun phrase extraction, where we want to find all the parts of speech in a sentence.

## Creation of Word Cloud:

Word Clouds (WordClouds) are quite often called Tag clouds, but I prefer the term word cloud. I think this term is more general and easier to understand by most people. The term tag is used for annotating texts and especially websites. This means finding out the most important words or terms characterizing or classifying a text. In the early days of web development people had to tag their websites so that search engines could more easily classify them. Spemmer used this to manipulate the search engines by giving incorrect or even misleading tags so that their websites ranked higher. Google changed this by automatically finding out the importance of the text components. Google more or less disregards the tags which the owners of the websites assigned to their pages. "Word clouds' ' as we use them also automatically find out what are the most important words. Of course, we do it naively by just counting the number of occurrences and using stop words.

Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analyzing data from social network websites.

**Advantages of Word Clouds :**

1. Analyzing customer and employee feedback.
2. Identifying new SEO keywords to target.

**Drawbacks of Word Clouds :**

1. Word Clouds are not perfect for every situation.
2. Data should be optimized for context.

**Chapter-2**

**Code Flow**

**Pre Processing**

The first thing that we'll do is preprocess the tweets so that they're easier to deal with, and ready for feature extraction, and training by the classifiers.

To start with we're going to extract the tweets from the json file, read each line and store the tweets, labels in separate lists.

Then for the preprocessing, we'll:

* **segment** tweets into sentences using an NTLK segmenter
* **tokenize** the sentences using an NLTK tokenizer
* **lowercase** all the words
* **remove twitter usernames** beginning with @ using regex
* **remove URLs** starting with http using regex
* **process hashtags** ,for this we'll tokenize hashtags, and try to break down multi-word hashtags using a MaxMatch algorithm, and the English word dictionary supplied with NLTK.

As we can see from the above example, it incorrectly breaks down the word 'casestudy', by returning 'cases', instead of 'case' for the first iteration., which would have been a better output. This is because it *greedily* extract 'cases' first.For an improvement, we can implement an algorithm that better counts the total number of successful matches in the result of the maxmatch process, and return the one with the highest successful match count.

**Clean tweets**

Data cleaning is the process of preparing data for analysis by removing or modifying data that is incorrect, incomplete, irrelevant, duplicated, or improperly formatted.

Delete words from the tweets containing username or symbols or web url .Also after viewing through the tweets we came across irrelevant strings as ([RT]) therefore need to remove that too.

The main aim of Data Cleaning is **to identify and remove errors & duplicate data, in order to create a reliable dataset**. This improves the quality of the training data for analytics and enables accurate decision-making.Preparing your data helps you maintain quality and makes for more accurate analytics, which increases effective, intelligent decision-making.

**Analyze Sentiment**

Return Positive,negative or Neutral on the basis of tweets after cleaning.This will be done on the basis of polarity of tweets using TextBlob library.If

analysis**.**sentiment**.**polarity **>** 0 : Positive

analysis**.**sentiment**.**polarity **==** 0 : Neutral

analysis**.**sentiment**.**polarity **<** 0 : Negative

TextBlob is a python library for Natural Language Processing (NLP).TextBlob actively used Natural Language ToolKit (NLTK) to achieve its tasks. NLTK is a library which gives an easy access to a lot of lexical resources and allows users to work with categorization, classification and many other tasks. TextBlob is a simple library which supports complex analysis and operations on textual data.

For **lexicon-based approaches**, a sentiment is defined by its semantic orientation and the intensity of each word in the sentence. This requires a pre-defined dictionary classifying negative and positive words. Generally, a text message will be represented by bag of words. After assigning individual scores to all the words, final sentiment is calculated by some pooling operation like taking an average of all the sentiments.

TextBlob returns **polarity** and **subjectivity** of a sentence. Polarity lies between [-1,1], -1 defines a negative sentiment and 1 defines a positive sentiment. Negation words reverse the polarity. TextBlob has semantic labels that help with fine-grained analysis. For example — emoticons, exclamation marks, emojis, etc. Subjectivity lies between [0,1]. **Subjectivity quantifies the amount of personal opinion and factual information contained in the text. The higher subjectivity means that the text contains personal opinion rather than factual information**. TextBlob has one more parameter — intensity. TextBlob calculates subjectivity by looking at the ‘**intensity**’. Intensity determines if a word *modifies* the next word. For English, adverbs are used as modifiers (‘very good’).

For example: We calculated polarity and subjectivity for “I do not like this example at all, it is too boring”. For this particular example, polarity = -1 and subjectivity is 1, which is fair.

However, for the sentence “This was a helpful example but I would prefer another one”. It returns **0.0 for both subjectivity and polarity** which is not the finest answer we’d expect.

It is expected that if the library returns exactly 0.0 either if your sentence didn’t contain any words that had a polarity in the NLTK training set or because TextBlob uses a weighted average sentiment score over all the words in each sample. This easily diffuses out the effect of sentences with widely varying polarities between words in our case : ‘helpful’ and ‘but’.

**Create WordCloud**

The next step is to convert each processed tweet into a bag-of-words feature dictionary. We'll allow for options to remove stopwords during the process, and also to remove \_rare\_ words, i.e. words occurring less than n times across the whole training set.After removing stop\_words we create the WordCloud of the dataset.

A word cloud is a visually prominent presentation of “keywords” that appear frequently in text data. The rendering of keywords forms a cloud-like color picture, so that you can appreciate the main text data at a glance.

The principles of generating a word cloud are not complicated, and can be roughly divided into several steps:

First, segment text data. This is also the first step in NLP text processing. For the process\_text() method in wordcloud, it is mainly the processing of stop words.

Secondly, calculate the frequency of each word in the text and generate a hash table. Word frequency calculation is equivalent to word count, the first case of various distributed computing platforms, and has the same status as hello world programs in various languages.

Thirdly, generate a picture layout proportionally based on the value of the word frequency. The class IntegralOccupancyMap is the algorithm of the word cloud and the core of the word cloud data visualization method.

Next, generate pictures on the word cloud layout diagram according to the corresponding word frequency. The core method is generate\_from\_frequencies, whether it is generate() or generate\_from\_text(), it will eventually reach generate\_from\_frequencies.

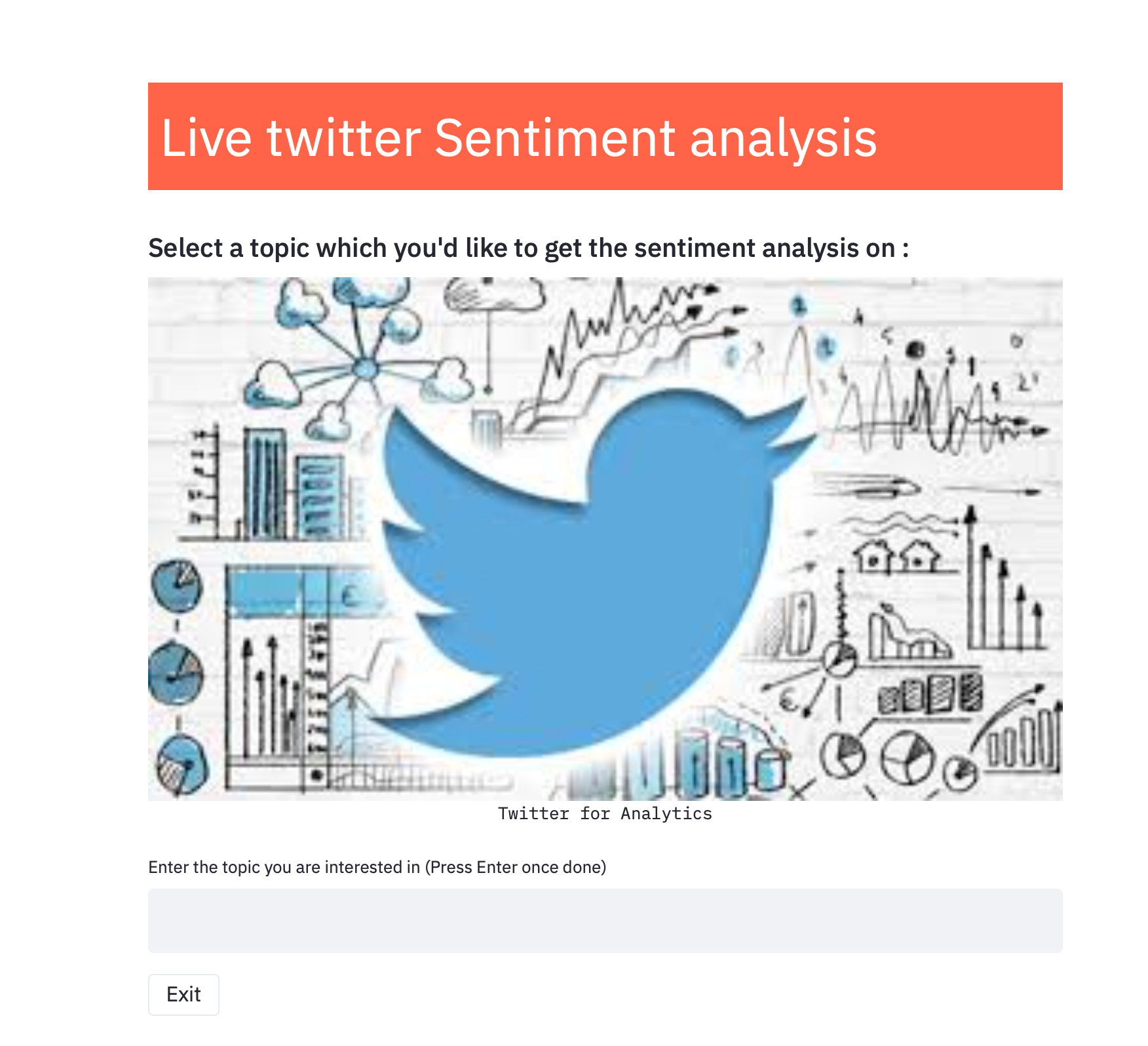
Finally, complete the coloring of each word on the word cloud, the default is random coloring.

**User interface:**

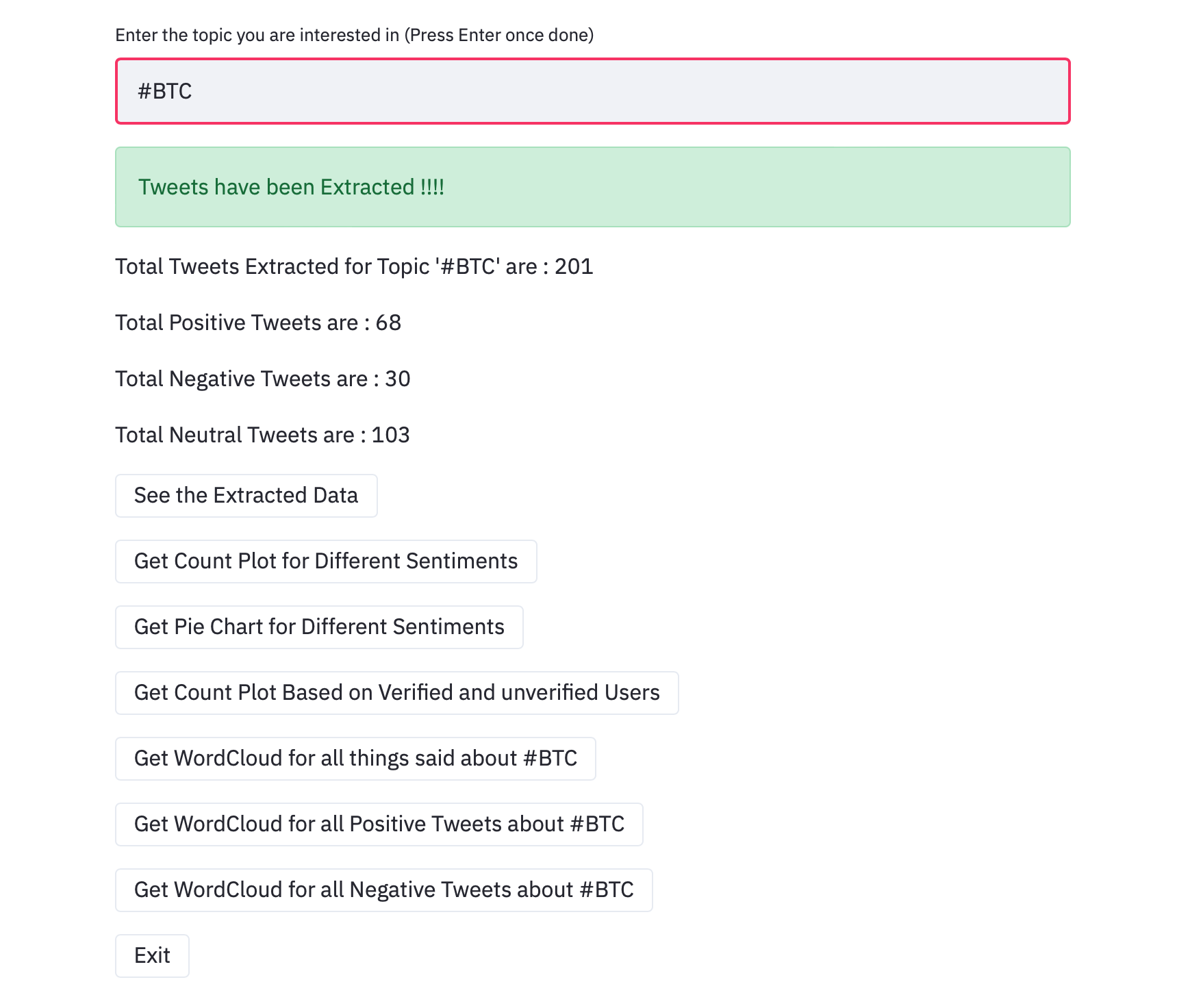
**F**or UI we Streamlit.Streamlit is **an open source app framework in Python language**. It helps us create web apps for data science and machine learning in a short time. It is compatible with major Python libraries such as scikit-learn, Keras, PyTorch, SymPy(latex), NumPy, pandas, Matplotlib etc.

**First Page :**

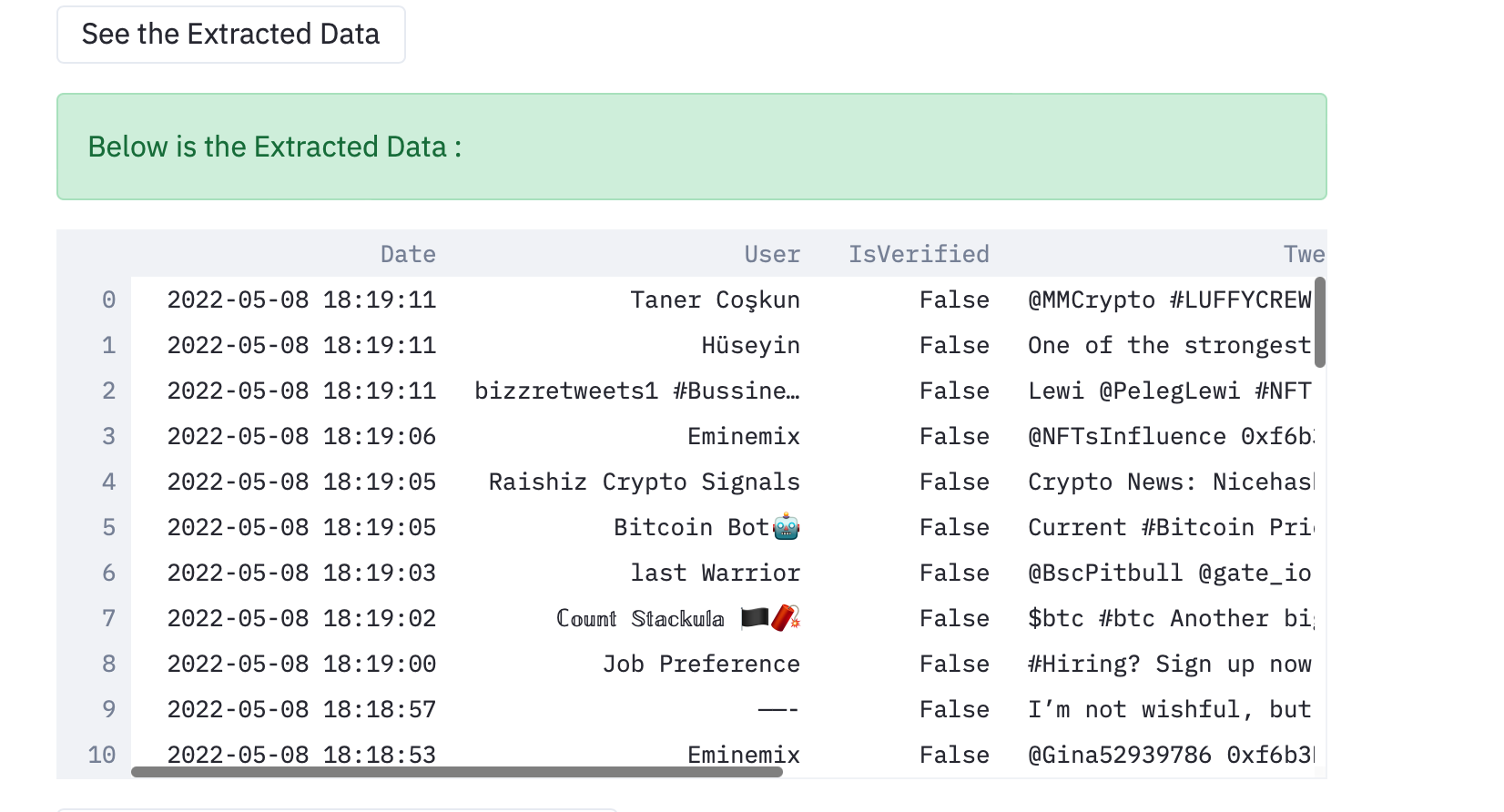
Here write the topic or token whose sentiment you need to know it will give information of tweets of about 100 tweets .Because that's the limit we have defined in code.We can increase that.



**Functionality After searching about keywords:**

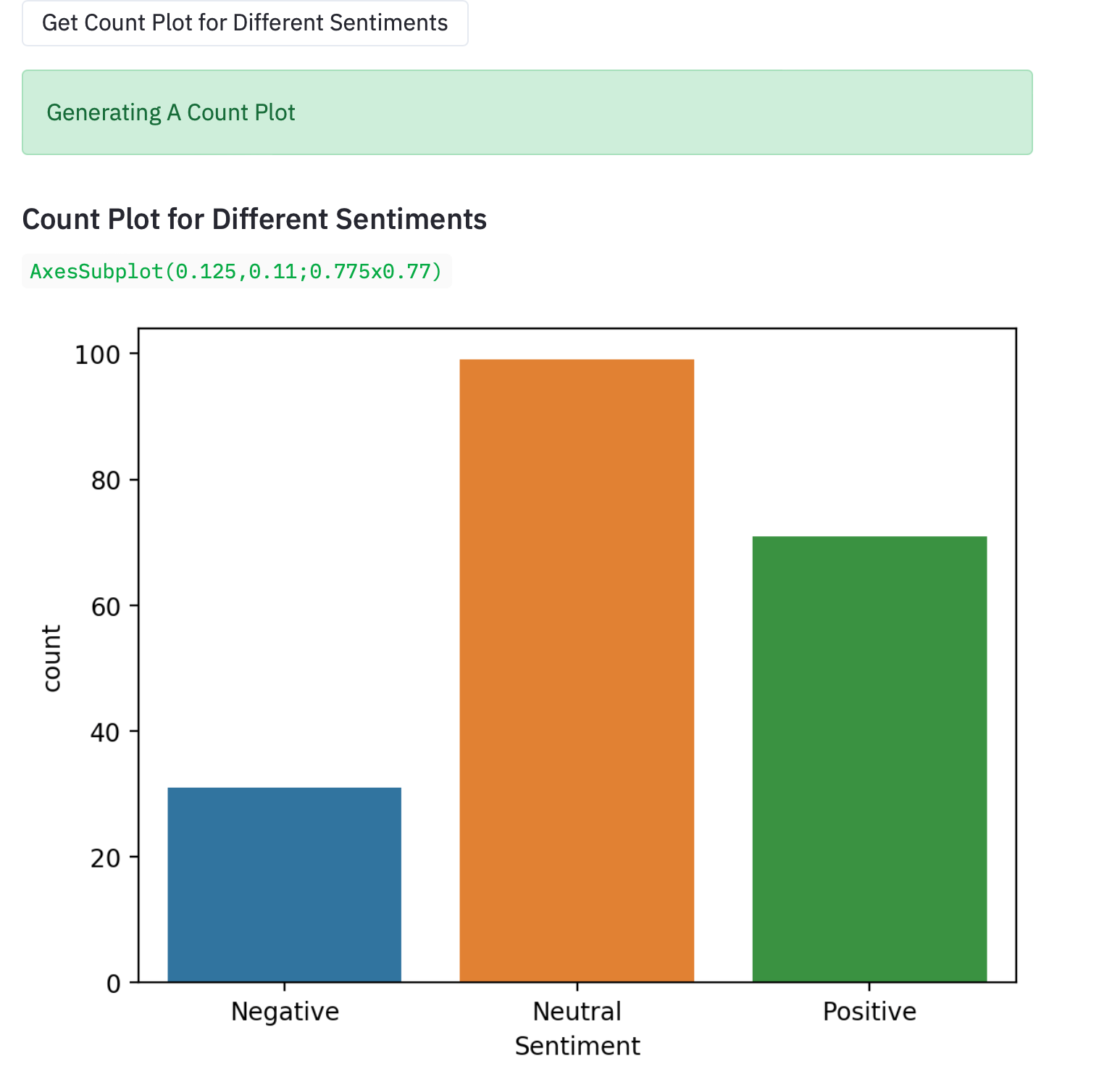


**DataSet of extracted tweets:**

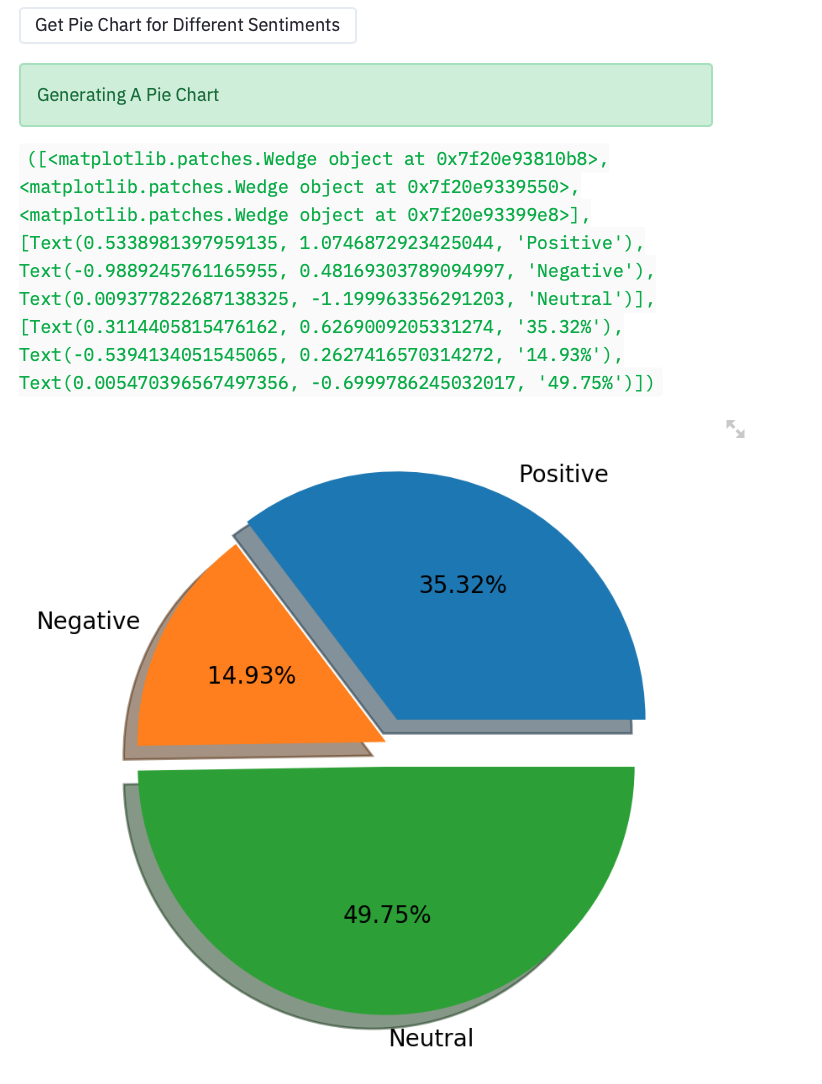




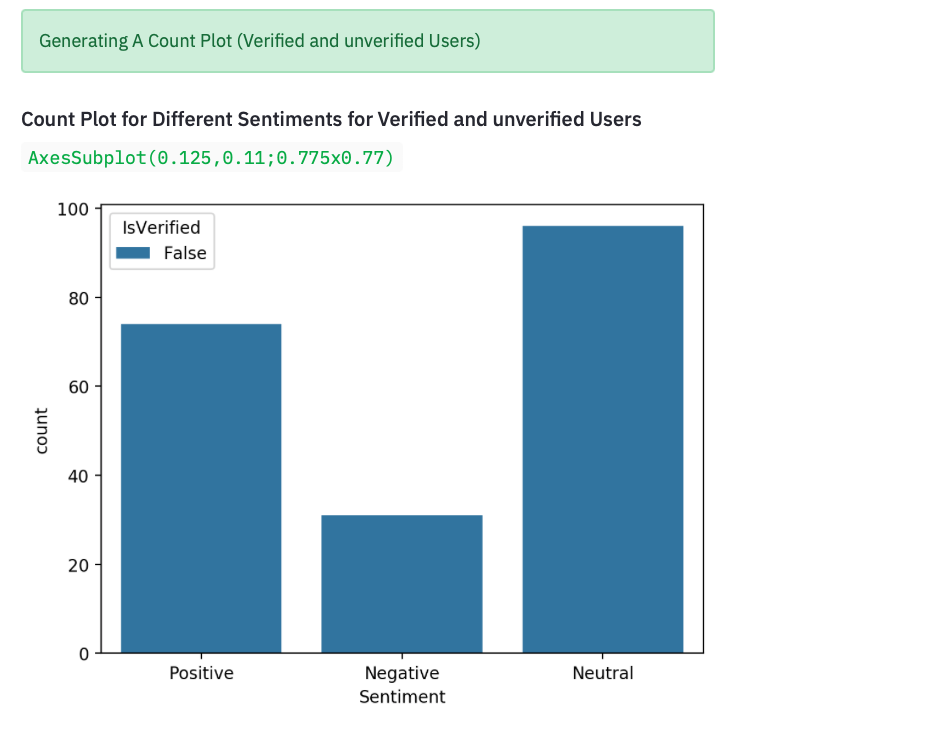
**Count plot of positive and negative tweets:**



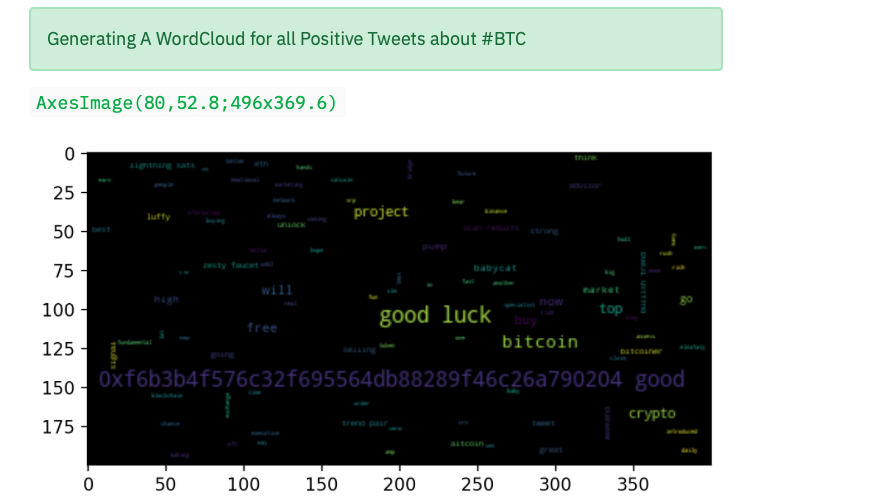
**Pie chart of sentiments:**

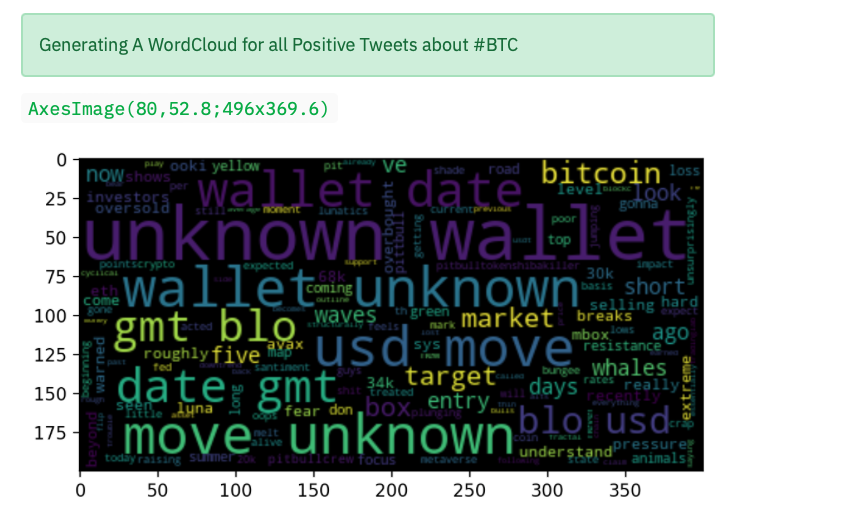


**Count plot based on verified and unverified user:**



**WordsClouds of positive , Negative and both:**







**Chapter-3**

Result and Discussion

In this paper, data extracted directly from Twitter API were used to train and test the models. A lexicon-based classifier used a manually created lexicon to find the sentiment of each tweet. Our proposed methodology used a novel approach for using both supervised and unsupervised modeling. As a result, the prediction showed improvements in comparison to existing work where a label data is present. Our model combined several algorithms to get the fit model for our data. Some metrics were used to validate and test the accuracy of each model as follows

**A. Measurements**

Recall: is defined as the number of true positives divided by the number of true positives plus the number of false negatives.

Precision: is defined as the number of true positives divided by the number of true positives plus the number of false positives

Fscore: is a measure of how accurate a model is by using precision and recall following the formula

F1\_Score = 2 \* ((Precision \* Recall) / (Precision + Recall))

**B. Cross validation**

In cross validation, the original training data set is divided into four groups, 4-fold cross validation for testing and training.

So we get to know how accurate the model’s predictions is when comparing the model’s predictions on the validation set and the actual labels of the data points. After applying validation techniques on the models, the prediction accuracy is found as indicated in Tables.

We will first present our results for the objective / subjective and positive / negative classifications. These results act as the first step of our classification approach. We only use the short-listed features for both of these results. This means that for the objective / subjective classification we have 5 features and for positive / negative classification we have 3 features. For both of these results we use the Naïve Bayes classification algorithm, because that is the algorithm we are employing in our actual classification approach at the first step. Furthermore all the figures reported are the result of 10-fold cross validation. We take an average of each of the 10 values we get from the cross validation. We make a condition while reporting the results of polarity classification (which differentiates between positive and negative classes) that only subjective labeled tweets are used to calculate these results. However, in case of final classification approach, any such condition is removed and basically both objectivity and polarity classifications are applied to all tweets regardless of whether they are labeled objective or subjective

**Chapter-4**

Conclusion

Sentiment analysis is a field of study for analyzing opinions expressed in text in several social media sites. Our proposed model used several algorithms to enhance the accuracy of classifying tweets as positive, negative and neutral. Our presented methodology combined the use of unsupervised machine learning algorithms where previously labeled data did not exist at first using lexicon-based algorithms. After that data were fed into several supervised models. For testing various metrics used, and it is shown that based on cross validation, maximum entropy has the highest accuracy. Same methodology can be used in various fields, detecting rumors

On Twitter regarding the spread of diseases. For future work, an algorithm that can automatically classify tweets would be an interesting area of research.

The task of sentiment analysis, especially in the domain of micro-bloging, is still in the developing stage and far from complete. So we propose a couple of ideas which we feel are worth exploring in the future and may result in further improved performance.

Right now we have worked with only the very simplest unigram models; we can improve those models by adding extra information like closeness of the word with a negation word. We could specify a window prior to the word (a window could for example be of 2 or 3 words) under consideration and the effect of negation may be incorporated into the model if it lies within that window.

The closer the negation word is to the unigram word whose prior polarity is to be calculated, the more it should affect the polarity. For example if the negation is right next to the word, it may simply reverse the polarity of that word and farther the negation is from the word the more minimized ifs effect should be. Apart from this, we are currently only focusing on unigrams and the effect of bigrams and trigrams may be explored.

As reported in the literature review section when bigrams are used along with unigrams this usually enhances performance. However for bigrams and trigrams to be an effective feature we need a much more labeled data set than our meager 9,000 tweets. Right now we are exploring Parts of Speech separate from the unigram models, we can try to incorporate POS information within our unigram models in future.

So say instead of calculating a single probability for each word like P(word | obj) we could instead have multiple probabilities for each according to the Part of Speech the word belongs to. For example we may have P(word | obj, verb), P(word | obj, noun) and P(word | obj, adjective).

Pang et al. used a somewhat similar approach and claims that appending POS information for every unigram results in no significant change in performance (with Naive Bayes performing slightly better and SVM having a slight decrease in performance), while there is a significant decrease in accuracy if only adjective unigrams are used as features.

However these results are for classification of reviews and may be verified for sentiment analysis on micro blogging websites like Twitter. One more feature we that is worth exploring is whether the information about relative position of word in a tweet has any effect on the performance of the classifier.

Although Pang et al. explored a similar feature and reported negative results, their results were based on reviews which are very different from tweets and they worked on an extremely simple model. In this research we are focussing on general sentiment analysis. There is potential for work in the field of sentiment analysis with partially known context.

For example, we noticed that users generally use our website for specific types of keywords which can be divided into a couple of distinct classes, namely: politics/politicians, celebrities, products/brands, sports/sportsmen, media/movies/music.

So we can attempt to perform separate sentiment analysis on tweets that only belong to one of these classes (i.e. the training data would not be general but specific to one of these categories) and compare the results we get if we apply general sentiment analysis on it instead.

**Chapter-5**

References

[1] Levy, M. (2016). Playing with Twitter Data. [Blog] R-bloggers. Available at: <https://www.r-bloggers.com/playing-with-twitter-data/> [Accessed 7 Feb. 2018].

[2] Popularity Analysis for Saudi Telecom Companies Based on Twitter Data. (2013). Research Journal of Applied Sciences, Engineering and Technology. [online] Available at: http://maxwellsci.com/print/rjaset/v6-4676-4680.pdf [Accessed 1 Feb. 2018].

[3] Zhao, Y. (2016). Twitter Data Analysis with R – Text Mining and Social Network Analysis. [online] University of Canberra, p.40. Available at:https://paulvanderlaken.files.wordpress.com/2017/08/rdatamining-slides-twitter-analysis.pdf [Accessed 7 Feb. 2018].

[4] Alrubaiee, H., Qiu, R., Alomar, K. and Li, D. (2016). Sentiment Analysis of Arabic Tweets in e-Learning. Journal of Computer Science. [online] Available at: http://thescipub.com/PDF/jcssp.2016.553.563.pdf [Accessed 7 Feb. 2018].

[5] Qamar, A., Alsuhibany, S. and Ahmed, S. (2017). Sentiment Classification of Twitter Data Belonging to Saudi Arabian Telecommunication Companies. (IJACSA) International Journal of Advanced Computer Science and Applications, [online] 8. Available <https://thesai.org/Downloads/Volume8No1/Paper_50-> Sentiment\_Classification\_of\_Twitter\_Data\_Belonging.pdf [Accessed 1 Feb. 2018].

[6] R. M. Duwairi and I.Qarqaz, “A framework for Arabic sentiment analysis using supervised classification” , Int. J. Data Mining, Modelling and Management, Vol. 8, No. 4, pp.369-381 , 2016.

[7] Hossam S. Ibrahim, Sherif M. Abdou, Mervat Gheith, “Sentiment Analysis For Modern Standard Arabic And Colloquial”, International Journal on Natural Language Computing (IJNLC), Vol. 4, No.2, pp. 95-109, April 2015.

[8] Assiri, A., Emam, A. and Al-Dossari, H. (2016). Saudi Twitter Corpus for Sentiment Analysis. International Journal of Computer and Information Engineering, [online] 10. Available at: http://waset.org/publications/10003483/saudi-twitter-corpus-for-sentiment-analysis [Accessed 1 Mar. 2018].

[9] L. Wasser and C. Farmer, "Sentiment Analysis of Colorado Flood Tweets in R", Earth Lab, 2018. [Online]. Available: https://earthdatascience.org/courses/earth-analytics/get-data-using-apis/sentiment-analysis-of-twitter-data-r/. [Accessed: 01- Mar- 2018].

[10] D. Robinson, "Text analysis of Trump's tweets confirms he writes only the (angrier) Android half", Variance explained, 2016.

[11] \_\_\_\_\_"A Common Database Interface (DBI)", cran.r-R, 2003. [Online]. Available: https://cran.r-project.org/web/packages/DBI/vignettes/DBI-1.html. [Accessed: 25- Mar- 2018].

[12] V. Kharde and S. Sonawane, "Sentiment Analysis of Twitter Data: A Survey of Techniques", International Journal of Computer Applications, vol. 139, p. 11, 2016.