

TWITTER SENTIMENTAL ANALYSIS
Project Report

*Submitted in fulfillment of the requirement for
the degree of Bachelor of Technology*

*In
Computer Science and Engineering and Information Technology*



By

Divyam Kudeshia (161296)

Anand Vashista (161308)

Under the supervision of

Dr. Rakesh Kanji

Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology Wagnaghat, Solan-173234, Himachal Pradesh

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this report entitled “Twitter Sentimental Analysis” in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering/Information Technology submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2020 to May 2020 under the supervision of Dr. Rakesh Kanji (Department of CSE & IT). The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature)

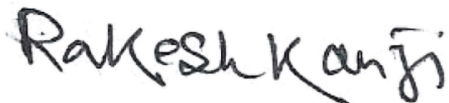


Divyam Kudeshia (161296)



Anand Vashistha (161308)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.



(Supervisor Signature)

Supervisor Name: Dr. Rakesh Kanji

Department name: Computer Science and Engineering and Information Technology

Dated:

ACKNOWLEDGEMENT

I want to take this chance to thank almighty for blessing me with his grace and taking my job to a successful culmination. We owe our profound gratitude to our project supervisor, Dr. Rakesh Kanji who took keen interest and guided us all along in my project work titled - “Twitter Sentimental Analysis” till the completion of our project by providing all the necessary information for developing the project. The project development helped us in research and we got to know a lot of new things in our domain. We are really thankful to him.

TABLE OF CONTENTS

TWITTER SENTIMENT ANALYSIS.....	
Candidate's Declaration.....	I
Acknowledgement.....	II
Abstract.....	III
Chapter-1 Introduction.....	6-9
Chapter-2 Literature Survey.....	10-18
Chapter-3 System Development.....	19-36
Chapter-4 Performance Analysis.....	37-52
Chapter-5 Conclusions.....	53
References	

ABSTRACT

This project principally addresses the matter of sentiment analysis in twitter; that's categorizing tweets consistent with the feelings expressed in them i.e. positive, negative or neutral. Twitter is a web micro blogging and social networking platform that permits the users to put in writing short standing of most length one hundred forty-five characters. it's a speedily growing service with over 205 million registered users out of that a hundred and ten million square measure active users and largely 1/2 them go online twitter on a usual - generating nearly 250 million tweets per day. because of this vast quantity of usage we tend to hope to attain a picture of public sentiment by analyzing the feelings articulated within the tweets. Analyzing the general public sentiment is a vital facet for several applications like corporations making an attempt to appear the responses of their merchandise within the market, statement political elections and predicting socioeconomic spectacles like stock exchanges. The goal of this project is to develop a purposeful classifier for associate correct and involuntary sentiment classification of unknown tweet stream.

CHAPTER 1

INTRODUCTION:

- **Inspiration**

We chose to work with Twitter because we think it is a good approximation of public sentiment, as opposed to traditional Internet articles and web blogs. The reason behind that is the amount of relevant data is much bigger for Twitter than for traditional blogging sites. The response on Twitter is quicker and more common (the number of users who tweet are more than those who write web blogs on a daily basis). Public sentiment analysis is important in macro-level socio-economic events such as estimating the stock market rate of a particular company. This can be done by analyzing the public sentiment towards the company over time, and by using economics to determine the relationship between public sentiment and the company's stock market value. To assess how well the market will react to their product, which sectors of market will react positively and which sector will have the adverse reaction (from Twitter for certain locations, which allows us to download geo tagged tweets currently). You can analyze the reasons behind geographical distinguished response about the information and to analyze, whether they are suitable for market segments such as creating the most appropriate solutions for their product in a more customized way to be marketed. Predicting the results of popular political elections and surveys is also an emerging application to sentiment analysis. One such research was conducted by Tumasjan et al in Germany for predicting the result of federal elections in which concluded that twitter is a good reflection of offline sentiment.

- **Introducing Domain**

This project of analysing the sentiments of tweets falls into the realm of "pattern classification" and "data mining". These two terms are closely related and interconnected, and can be formally defined as the process of searching for "useful" patterns in big set of data, either inevitably i.e. unsupervised or semi-automatic i.e. supervised. However, the project heavily rely on the methods of "natural language processing" and "machine learning" in defining important patterns and features from a large data set of tweets, the model is strictly based on unmodified data patterns (tweets) and on "Machine learning" techniques to accurately classify them.

The features used for modeling different patterns and classification can be alienated into two main groups: formal language based and informal blogging. Language based features are related to formal linguistics and prior knowledge of individual words and phrases is part of polarity and speech tagging of sentences. Prior sentiment polarization means that certain words and phrases usually have a natural tendency to express specific and specific emotions. Foreg., the word "outstanding" has a strong positive meaning, and the word "wicked" has a strong negative meaning. So, when a word is used with a +ve meaning in a sentence the whole is likely to convey a +ve emotion. Parts of speech tagging on the other hand, are syntactic approach to the problem. This means that each person in a sentence automatically recognizes what the word is: noun pronoun verb adjective verb, deflection,. Samples canbe obtained from the analysis of the frequency distribution (individually ether or mutually with some of these portions of speech) in certain class of labelled tweets. Twitter based features are more informally related to how people express themselfe on online social platforms and their emotions inthe limited140-character space that Twitter provides. These includes Twitterhashtags, retweets, wordcapitalization, wordlength, question-marks, URLs in tweets, exclamation marks, Internet emoticons & Internet shorthand / slang.

Classification methods can also divide into two categories: supervised versus unsupervised versus non-compatible versus adaptable or reinforcement methods. The supervised approach isthat already have data-labels available & we use them to train classifier .Classifier traning involves using pre-labels to extractfeatures that best

differentiate patterns & differences b/w different classes, and classify and then describe the best model accordingly.

For example, if we come up with a simplified model, the neutral tweet contains an average of 0.003 exclamation marks per tweet, a sentiment bearing tweet has 0.8 and we need to classify a tweet there is an exclamation mark (ignoring all features) it would have been classified as subjective since an exclamation mark is close to the model of 0.8 exclamation marks. Unsupervised classification is known as when we don't have labelled data for our training. It deals with these adaptive classification techniques as well as the feedback from environment. In our case, the response from the environment is on the form of a human being telling classifier whether he has done good or bad in the classifying a particular tweet and the classification must learn from this point of view.

There are 2 other types of adaptive methods: **passive & active**. There are passive methods that only use feedback to learn about the environment (in this case our model for every three classes of tweets means improvement), but current classification algorithm holds it well. The active approach adapts the classification algorithm to time consuming reality.

Several matrices have been proposed for computing & for comparing the results of our experiments. Some mostly common metrics are: Precision, Recall, Accuracy, F1 Measure, True Rate and False Alarm Rate (These matrices are computed individually for each class & then total classification performance on an average.) Below is our problem example of how to calculate the metric we need.

	Machine says yes	Machine says no
Human says yes	tp	fn
Human says no	fp	tn

Table 1: A Typical 2x2 Confusion Matrix

- Precision(P) = $\frac{tp}{tp+fp}$
- Recall(R) = $\frac{tp}{tp+fn}$
- Accuracy(A) = $\frac{tp+tn}{tp+fn+fp+tn}$
- F1 = $2 \cdot \frac{P \cdot R}{P+R}$
- True Rate(T) = $\frac{tp}{tp+fn}$
- False-alarm Rate(F) = $\frac{fp}{fp+fn}$

CHAPTER 2:

LITERATURE SURVEY:

- **Restrictions to Prior - Art**

Sentiment analysis is a relatively new research topic in the field of microblogging, so there is plenty of room for further research in this area. A small amount of relevant prior work has been done in sentiment analysis of user reviews, documents, web blogs articles & general phrase level sentiment analysis. Due to the limit of 140 characters per tweet, which forces the user to express compressed views on a short text. The better results involved in the sentiment classification are the uses of supervised learning techniques such as naive bayes & SVM(Support Vector Machines), but manual labeling required for the supervised approach is costly. Some of the work was done on the unsupervised & semi supervised approaches and there is great scope for improvement. Various researchers testing new features and classification methods have compared the results to baseline-line performance. There is need of better & more formal comparisons b/w the results through many distinct features & classification methods to select best features & most efficient classification methods for the particular apps.

- **Related-work**

The bag of words model is one of the most widely used feature models for an almost all text classification tasks, with its simplicity and good performance. The model refers to text classified as the bag or collection of individual words with links or dependencies to a word, meaning it completely ignores the grammar and order of words in the text. This model is also very popular in sentiment analysis & has been used by various researchers. The simplest way to incorporate this model into our classification is to use unigram as a feature. Generally speaking, n gram is the sequence of "n" words in our text, completely independent of any other word or gram on the text.

Therefore, we believe that the text to be classified as a unigram is only a collection of individual words, and the presence or absence of another word in a text does not affect the likelihood of a word occurring. It is quite simple but has been shown to provide good performance. The simple way of using unigram is to assign them the fixed pre-polarization & to average the overall polarity of the text, where the overall polarity of the text is calculated by summing the different prior polarities of the individual unigrams. If the word is used as a symbol of positivity, the earlier polarity of the word is positive, e.g. the word "good". However, the word is usually negative if it is associated with negative implications, for example "wicked". There may also be polarity in the model, which means how specific the term is to a particular class.

The word "overwhelming" probably has a strong subjective polarity, while the word "good" has a strong positive polarity, but perhaps with weak individualism.

There are three ways to use the pre-polarity of words to characterize. An unsupervised simple approach is to use publicly available online dictionaries, which map a word to its previous polarity. Multi-Perspective-Questioning Answering (MPQA) is a subjectivity dictionary that maps a total of 4,850 words whether "positive" or "negative" and whether they are "strong" or "weak" themes. SentiWordNet 3.0 is another resource that gives the possibility of every word in the positive, negative and neutral classes.

The second approach is to construct a positive pre-polarity dictionary according to the occurrence of each word in each particular class from our training data. For example, if a particular word in our training dataset (compared to other classes) occurs more frequently in positively labeled phrases, we calculate the probability that one word belongs to the positive class rather than the other class. This approach has been shown to provide better performance, as the earlier polarization of words is more suitable and fitted with a particular type of text, and is not as common as the previous approach. However, the latter is the supervised approach because the training data has to be labeled to the suitable class before it is possible to calculate the relative occurrence of a word in each of the class. Performance reductions were identified by Kouloumpis et al. using dictionary word features with custom n-gram word features built from the training data, as opposed to using n-grams alone.

The third approach is the mediation between the two approaches. In this, we build our own polarization but not necessarily from our training data, so we do not need to label training data. One way to do this is proposed by Turnty et al. The pre semantic orientation (polarity) of a word or phrase is calculated by reciprocating the information with the word "excellent" and subtracting the result from the word "poor" with the interaction of that word or phrase. They used the number of results from the relevant search query's online search engine to calculate mutual information. The final principle they use is as follows:

$$Polarity(phrase) = \log_2 \frac{hits(phrase NEAR "excellent").hits("poor")}{hits(phrase NEAR "poor").hits("excellent")}$$

Where hits (phrase "excellent") is the number of documents given by the search engine (whose polarity must be calculated) and the word "excellent" occur together. hits ("excellent") means the number of documents that contain the word "excellent" co-occur. Prabowo et al. came up with this idea and used 120 positive words and 120 negative seeds of data to conduct an internet search. So, the overall semantic orientation of the word under consideration can be found by averaging the word's proximity to each word's seed words.

Another graphical method of quantifying the polarity of adjectives is discussed by Hatzivasiloglou et al. . The process involves first identifying all the adjective combinations from the corpus and then categorizing each pair of adjectives using an algorithm(supervised). A graph is constructed in which node adjectives and links represent the same or different semantic orientation. Finally, the clustering algorithm is implemented, which divides the graph into two subsets, which means that the nodes in the subset have basically the same orientation links, and the links between the two subsets have basically different orientations. Most subsets have positive adjectives and the other has negatives.

Many researchers in this field have already used dictionaries of publicly available emotion, while others have also explored building their own pre-polar dictionaries.

A fundamental problem with the approach of prior polarity identified by Wilson et al. He distinguishes between prior polarity and contextual polarity. They also say that the prior polarity of a word may actually be different from the word used in a particular context. Let's take following phrase as an example:

Philip Clapp, president of the National Environment Trust, sums up well the general thrust of the reaction of environmental movements: "There is no cause at all to believe that the polluters are suddenly going to become appropriate."

In this example, the four underlined words "Trust", "well", "cause" and "appropriate" are positive references when viewed without reference to the phrase, but are not used here to convey a positive emotion. This leads to the conclusion that the word "Trust" may commonly be used in positive sentences, but this does not rule out the possibility that it is also found in non positive sentences.

Prior polarization of individual words (whether words are generally positive or negative perception) is not the only problem. Due to the contextual polarity of the phrase, explores some other features, including grammatical and syntactic relationships between words to improve their classification.

The performance of Twitter sentiment analysis may be closely related to phrase level sentiment analysis. In 2005, Wilson and others presented a seminal paper on phrase-level sentiment analysis. It identifies a new approach to the problem by first classifying the phrases according to subjectivity (polar) and the objectivity (neutral) and categorizing the subjective-categorical phrases as positive or negative. Many subjective phrases uses prepositional expressions, which in particular contributes to the classification of subjective phrases. If we use a simple classification we assume that the contextual polarity of the term is equal to its prior polarity, the result is approximately 48%. The novel classification system proposed contains a list of general features that

contain information about the contextual polarity, resulting in a significant improvement in the performance (in terms of accuracy) of the classification process. The results of this paper are presented in the following table:

Features	Accuracy	Subjective F.	Objective F.
Word tokens	73.6	55.7	81.2
Words + prior polarity	74.2	60.6	80.7
28 features	75.9	63.6	82.1

Table 2: Step 1 results for Objective / Subjective Classification in [16]

Features	Accuracy	Positive F.	Negative F.	Both F.	Objective F.
Word tokens	61.7	61.2	73.1	14.6	37.7
Word + prior	63.0	61.6	75.5	14.6	40.7
10 features	65.7	65.1	77.2	16.1	46.2

Table 3: Step 2 results for Polarity Classification in [16]

One way to reduce the independence condition and incorporate partial references into our word model is to use bigrams and trigrams as well as unigrams. Bigram is a collection of two mutually exclusive words in a text, and trigram is a collection of three consecutive words. Thus, we can calculate the prior polarity or the probability of the bigram / trigram of the particular class – instead of prior polarity of separate class. Many researchers have experimented with them, saying that if we have to use one of them, unigram perform better, and some of the unigram with bigram can give better results. Trigrams generally have poor performance, as reported by Pak et al. . Performance reduction using trigrams because there is a compromise between capturing more complex patterns and word coverage when moving to higher numbered grams. Some researchers have tried to include disclaimers in the Unigram word model. Pang et al. And Pakl et al. used a model in which prior polairty was reversed to the word, meaning denial (like "no", "not", "don't", etc.). So, some relevant information is included in the word model.

Grammatical features (such as Parts of speech tagging" or POS tagging) are also commonly used in this domain. The concept of tagging every word of a tweet with reference to any part of speech is: noun, pronoun, verb, adjective, intensity, etc. The concept is to identify and use models based on this POS which we can use in the classification process. For example, it has been reported that objective tweets have more common nouns and third party verbs than subjective tweets, so if a tweet is classified, the greater use of generic nouns and verbs is usually in the third person, that tweet is objective (according to this feature). Similarly, subjective tweets contain more adverbs, adjectives, and interjections. These relationships are established in the figures below:

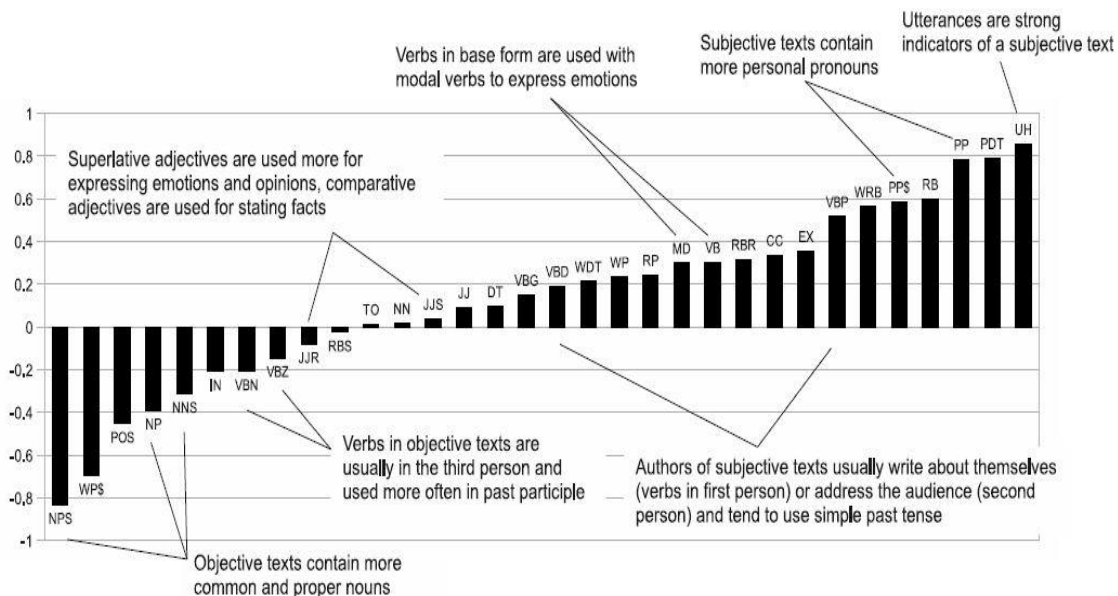


Figure 1: Using POS Tagging as features for objectivity/subjectivity classification

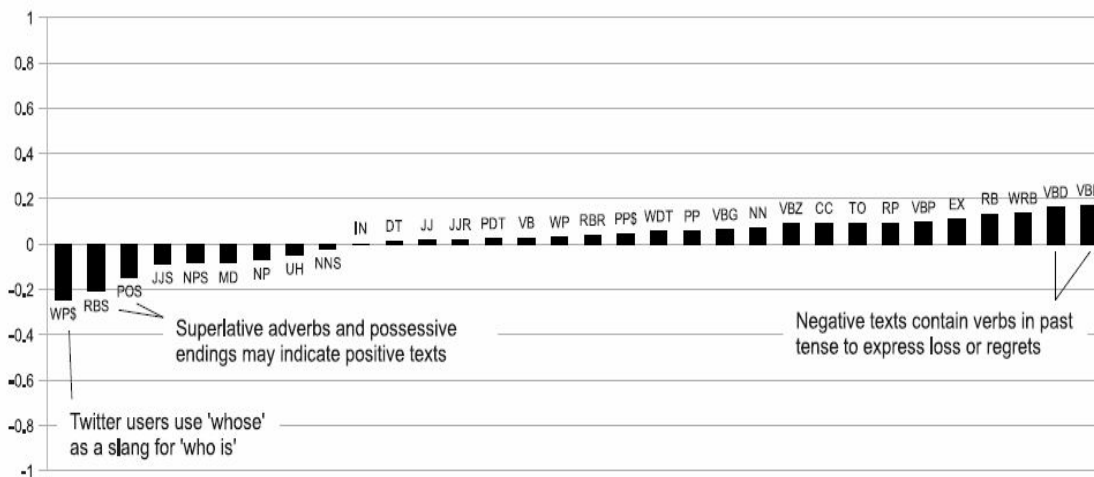


Figure 2: Using POS Tagging as features in positive/negative classification

However, there is still controversy over whether part of speech is a useful feature of sentiment classification. Some researchers argue in favor of better POS features while others do not recommend them.

Apart from these, there is much work to be done in searching for a feature class that is only suitable for the microblogging domain. The presence of URLs in a tweet and the number of capitalized letter / letters in a tweet were noted by Koulompis et al. and Barbosa et al. . Koulompis represents positive results for the use of features such as emoticons and Internet slang terms. Brady et al does study on the lengthening of words as a symbol of subjectivity in a tweet. The paper reports positive results of their study, suggesting that if a word is more frequent, the term is considered a strong sign of subjectivity.

Naive bayes classifier and state vector machines are the most commonly used classification techniques. Some researchers, such as Barbosa et al. publish good results for support vector machines, while Pak et al. support Naive Bayes.

Although it continues to add more labeled tweets to training data, it has been observed that having a larger training sample is somewhat payoff to a certain degree, then the accuracy of the classification is almost constant. Barbosa et al. for training classifiers used tweets labeled by Internet sources instead of hand labeling. While doing so the accuracy of the labeled models is lost (modelled as the increase in noise) but if the accuracy of the training label exceeds 50%, the higher the label, the higher the classification result accuracy. So, if there are a large number of tweets in this way, then our labels will make noise, shall be inaccurate and can be compensated for the mistake. On the other hand Pak et al And Go et al use the presence of positive or negative emoticons to assign labels to tweets . Similar to the above case, they used a large number of tweets to reduce the impact of noise on their training data.

Some early work in this area categorized the text as positive or negative, assuming that all of the data presented was subjective. If this is good assumption for things like movie review but, while analysing tweets and blogs, we should consider a lot of objective text, so the inclusion of a neutral class in the classification process has now become an standard.

There has been a lot of recent research about the classification of tweets according to the mood they express, which takes it a step further. Bolan et al explores this area and develops a technique that classifies tweets into 6 different moods: tension, depression, anger, vigour, fatigue, and confusion. They use an extended version of Profile of Mood States (POMS): a widely accepted psychometric device. They create a word dictionary and assign weights to each of the six moods, and then refer to each tweet as a vector corresponding to these 6 dimensions. However, no details have been given as to how he built his customized dictionary and what methods he used for classification.

CHAPTER-3

SYSTEM DEVELOPMENT

There are five fundamental classes in which the emotion analysis for functional classification process can be distinguished .

These are listed below:

- I. Data Acquisition
- II. Human Labelling
- III. Feature Extraction
- IV. Classification

Data Acquisition :

Python library “tweetstream” is used a for the extraction of the data of underdone tweets. The Twitter streaming API [26] package is provided through this library only. Sample stream & Filter stream are two ways allowed by API for access. For the tweets that are broadcasted in the real time, not a large but a random small sample is given by the sample stream. Filter Stream provides tweets that meet certain criterion. Tweets delivered are filtered according to these three criterion:

1. To track or search particular keywords in tweets
2. Particular Twitter users as stated by their user-id
3. Specific location originated location tweet(s) (for geo-tagged tweets).

The programming individual can point out one or several combination of these filter criterion. Since there are no such restrictions whatsoever therefore we will always use SampleStream mode as it is beneficial to us.

Since, we want to increment in the data integrity, we have gotten parts of it at different times than we get it all at once. If we apply not the former one but the latter approach, the normality of messages can be compromised because a substantial section of messages contain a specific trending topic. Thus, more or less the same general emotion. When we are going through the sample of message we got this event came through. For example,

for Christmas and New Year related samples, there is an important part of tweets that refer to these joyful events and is generally positive. Sampling our data at different points tries to minimize this problem. For four different points or dates, namely Feb 8th, Dec 29th, Jan 19th and Dec 17th.

The tweets obtained in this way contains a lot of raw information, which may not be relevant to our specific program. Also, various key pairs are found when the data type “Dictionary” of Python is considered. Some of the key-value pairs are as follows:

- Whether a tweet has been favourited
- User ID
- Screen name of the user
- Original Text of the tweet
- Presence of hashtags
- Whether it is a re-tweet
- Language under which the twitter user has registered their account
- Geo-tag location of the tweet
- Date and time when the tweet was created

Fig from[17]

We only filter the info we need and we leave the rest as there is so much of information. We save the real content of the messages in a different file and redirect all of them for our specific application, indicating that the English language is specified for the user’s social platform account. The users’ account language is given as “lang” and the tweet’s text is represented as “text” i.e. dictionary text.

Since individual labeling is a money consuming process, more of the tweets are filtered out and we need to mark them so that we can make a great diversity without the usual differences in tweets without losing the norm. The relevant filter criterion are as follows:

- Remove Retweets (any tweet which contains the string “RT”)
- Remove very short tweets (tweet with length less than 20 characters)
- Remove non-English tweets (by comparing the words of the tweets with a list of 2,000 common English words, tweets with less than 15% of content matching threshold are discarded)
- Remove similar tweets (by comparing every tweet with every other tweet, tweets with more than 90% of content matching with some other tweet is discarded)

Fig from[18]

This is followed by an average of 30% of tweets per filtered sample for human labeling, totaling 10,173 tweets.

Human Labelling :

For every tweet, three copies are prepared for individual marking or labelling so as to make them marked them by four different sources. Reduction in noise and in the inaccuracy in marking scheme and the acquisition of public opinion on the emotion of a specific tweet is successfully done due to the above mentioned labelling. Generally speaking more copies of the label may be better for us, but we have to remember the cost of labelling, so we have reached a reasonable figure of three.

According to the emotions expressed in the message on Twitter, the tweet is categorized into these four categories: positive(+), negative(-), neutral/objective(0), and ambiguous. To help with the labeling process we have given our labels the following guidelines:

- **Positive :** If the entire tweet has a positive / ecstatic / happy / cheerful / happy attitude or a reference to some good idea. Although if more than one emotion is expressed in a twitter post, affirmative feelings are more effective. For example: “I moved to the USA after 4 years with a shithole in Australia! 😊.

- **Negative** : If the entire tweet has a negative / sad / depressed attitude or is mentioned with some negative(-) perception. Although if more than one sentiment is expressed in a twitter post, negative(-) emotion is more effective. Example: "I want to go outside, but Now this Coronavirus is holding us"☹.
- **Neutral / Objective**: If the Tweet creator does not express personal feelings / opinions in the tweet and only spread the information. Ads for various products are labeled 'neutral'. For example: "India pledges to end Coronavirus spread from the country".
- **Ambiguous** : If more than one emotion is expressed in a tweet that is equally powerful and more explicit without a specific emotion. When it is clear that some personal views are expressed here, it is difficult / impossible to understand a properly articulated sentiment due to lack of context. Example: "I like and dislike chocos equally". At last, if the tweet indication is not clear from the available info. Example: "Sky is blue".
- **< Blank >**: If this is related to a language other than English, leave the tweet at random so that it is ignored in the training statistics.

In addition, it is suggested that labels exclude personal biases from labeling and do no make-up, that is, to view the tweet from the point of the view and personal info in the current personal post an not to view it through any extra personal info.

Once it is labeled with four classes, to get an average of the opinions of the three people together is the next step. The way we did it was by majority vote.

So for example, if two labels are in agreement on a particular tweet, we label the entire tweet. If the 3 labels are not similar, we would have found the post to be "out of reach of the majority vote." We came up with the following statistics for each category after a majority vote.

- Positive: 2573 tweets
- Negative: 1777 tweets
- Neutral: 4573 tweets
- Ambiguous: 751 tweets
- Unable for reaching majority vote: 391 tweets
- Unlabelled non-English tweets: 368 tweets

So if we only included tweets that could get a positive, negative or neutral majority vote, we would be left with 8963 tweets for the training set . These include 4543 objective tweets and 4420 subjective tweets (total of positive(+) & negative(-) tweets).

We also calculated the human-to-human contract for our tweet labeling work, and the results are as follows:

	Human 1: Human 2	Human 2: Human 3	Human 1: Human 3
Strict	58.9%	59.9%	62.5%
Lenient	65.1%	67.1%	73.0%

Table 4: Human-Human Agreement in Tweet Labelling

The "hard agreement" measure in the above matrix, where all the labels mentioned by humans must match in all cases, but the "liberal" measure is not considered a disagreement if one person finds the tweet "unclear" and another. So, in the case of a "liberal" rating, a non-liberal class maps to another class.

Therefore, it shows that the differentiation of emotions is also a difficult task for people, as the one-to-one contract is in the range of 60-70% (based on the definition of our agreement). We now consider another table presented by Kim et al. It refers to the human-to-human agreement in terms of labeling individual adjectives and verbs. [14].

	Adjectives	Verbs
	Human 1: Human 2	Human 1: Human 3
Strict	76.19%	62.35%
Lenient	88.96%	85.06%

Table 5: Human- Human Agreement in Verbs / Adjectives Labelling [6]

When the differentiation is in between in these three classes i.e. positive(+),negative(-) and neutral(0) then a narrow measurement occurs here. Whereas when the liberal measures are considered the classification makes positive(+) and negative(-) tweets into one category, and we are left with only subjective i.e. neutral class. These results indicate our first claim that analytical analysis of emotions is not a simple task. These results far outweigh our consensus results because in this case people are asked to mark individual words, which is relatively not a difficult task than marking the entire posts.

Feature Extraction :

For the training set of the given extracted data, we again have to extract the features that are useful in the classification process. But text formatting techniques are discussed first as it helps in the retrieval of the aforementioned useful features.

- Tokenization :The process of dividing the flow of text sentence into various symbols ,words and other elements which are meaningful called “tokens” is known as tokenization. These token can simply divided by whitespace character “ ” or punctuation characters like ?,!,%,(,) etc.
- If we are only focused on examining tweet text, the URL and user references (marked by the "http" and "@" tokens) will be deleted.

- If we want to compare a tweet in a list of English words, then we can remove the punctuation marks and digits.
- Lowercase conversion: Tweets can be simplified by turning it into a lowercase letter, which makes it easier to compare with the English dictionary.
- Steaming: The process of generalization of the reduction process of the derived words to its root words is called “Steaming”. For example: “Stem” is the reduction of the root words like “stammered”, “stemmed”, “stemming”. The benefit of the stem is that it helps to make comparisons between words, because we do not have to deal with complex language changes of the word. “Porter stemming” algorithm is used if we need a comparison in this case for both the twitter posts and the dictionary.
- Stop-Word Removal: These are the words that are referred as the most common word class. When used in text it doesn’t contain any additional info and therefore are termed as ‘useless’. For Examples are “A”, "The", "He", "By", "She", "On" . As there is no additional info provided by these words and as they are used for the same meaning in mostly every section of text, it is very convenient to remove these, for example, the frequency(f) of occurrence in various sections is reason behind the pre-emotion-polarization of a post. Average sentiment of the twitter post over the group of the words used in that specific tweet is measured by applying this polarity.
- Parts-of-speech tagging : This is the process of assignment of a tag to each word in a text under which the grammatical component of speech belongs to the word, such as noun, verb , adjective , adverb , etc.

As some of the methods of text formatting have been discussed, we will go over the set of attributes that we find. It helps our classification to distinguish between several sections if any is a variable, this can be seen below. The Objectivity / Background Classification and the Positivity / Negative Classification (discussed in detail in the next section), are the 2 types of classification in this system(s) . As suggested by the name, the former one is to distinguish

between objective(obj) & subjective(sub) sections, and the latter one is to distinguish between positive(+) & negative(-) sections.

Following is the attributes' list found for the subjective / subjective classification

- Number of exclamation marks in a tweet
- Number of question marks in a tweet
- Presence of exclamation marks in a tweet
- Presence of question marks in a tweet
- Presence of url in a tweet
- Presence of emoticons in a tweet
- Unigram word models calculated using Naive Bayes
- Prior polarity of words through online lexicon MPQA
- Number of digits in a tweet
- Number of capitalized words in a tweet
- Number of capitalized characters in a tweet
- Number of punctuation marks / symbols in a tweet
- Ratio of non-dictionary words to the total number of words in the tweet
- Length of the tweet
- Number of adjectives in a tweet
- Number of comparative adjectives in a tweet
- Number of superlative adjectives in a tweet

Fig from[19]

- Number of base-form verbs in a tweet
- Number of past tense verbs in a tweet
- Number of present participle verbs in a tweet
- Number of past participle verbs in a tweet
- Number of 3rd person singular present verbs in a tweet
- Number of non-3rd person singular present verbs in a tweet
- Number of adverbs in a tweet
- Number of personal pronouns in a tweet
- Number of possessive pronouns in a tweet
- Number of singular proper noun in a tweet
- Number of plural proper noun in a tweet
- Number of possessive endings in a tweet
- Number of wh-pronouns in a tweet
- Number of adjectives of all forms in a tweet
- Number of verbs of all forms in a tweet
- Number of nouns of all forms in a tweet
- Number of pronouns of all forms in a tweet

Fig from [19]

Positive(+)/Negative(-) Classification's list of attribute is as follows:

- Total emoticon score (where 1 is added to the score for positive emoticons and 1 is subtracted for negative emoticons).
- Online Polarity Dictionary Total score from MPQA (where a strong positive word in a tweet increases the score by 1.0 and a weak negative word reduces the score by 0.5)
- Unigram Word Models were calculated using Naive Bayes
- The total number of emoticons in the tweet
- A is the number of positive emoticons in a tweet
- A is the number of negative emoticons in a tweet
- Q is the number of positive words from the MPQA dictionary in the tweet
- Q is the number of negative words from the MPQA dictionary in the tweet
- A is the number of base-form functions in a tweet
- A is the number of past verbs in a tweet
- A is the number of current cell faces in a tweet
- A is the number of past participle verbs in a tweet
- 3 person specific current actions in a tweet
- A is the number of single non-current single verbs in a tweet
- A is the number of plural nouns in a tweet
- A is the number of correct nouns in a tweet
- A is the number of cardinal numbers in a tweet
- The number of coordinating conjunctions in a tweet
- A is the number of verbs in a tweet
- W-adverbs in A Tweet
- A is the number of actions of all the forms in a tweet

Fig from[19]

By using Naïve Bayes, the mathematical and analytical reasoning on how to calculate the unigram word can be given. Also, the calculation of probability of the word which belongs to a class named in our training set is the basic idea. By using formulas of mathematics, an example can simply be presented for the calculation of the probability of the clause which belongs to the subjective and subjective class. Similar measures should be taken for both the positive(+) & negative(-) categories.

We begin by calculating the probability of a word in our training data for a particular class:

$$P(\mathit{word}_1|\mathit{obj}) = \frac{\mathit{count}(\mathit{word}_1 \text{ in } \mathit{obj} \text{ class})}{\mathit{count}(\mathit{total} \ \mathit{words} \ \text{in} \ \mathit{obj})}$$

We now state the Bayes' rule. According to this rule, if we want to find the probability of whether a tweet is objective, we need to compute the probability of tweet given the objective class and the prior probability of objective class. The term $P(\mathit{tweet})$ can be substituted with $P(\mathit{tweet} / \mathit{obj}) + P(\mathit{tweet} / \mathit{subj})$.

$$P(\mathit{obj}|\mathit{tweet}) = \frac{P(\mathit{tweet}|\mathit{obj}).P(\mathit{obj})}{P(\mathit{tweet})}$$

The estimation of the likability of the tweet being passed to an objective category can simply be done as we hit the independence of unigrams in a message post (i.e. the likability of the other word which occurs in a text is not affected by the encounter of one word in the text). The related objective(obj) class is the product of the the above mentioned likelihood of all the words in the message post. The predictive probability of the Objective category can be ignored again if the equal category size for the Subjective & Objective category is taken.

$$P(\mathit{obj}|\mathit{tweet}) = \frac{\prod_{i=1}^N [P(\mathit{word}_i|\mathit{obj})]}{\prod_{i=1}^N [P(\mathit{word}_i|\mathit{obj}) + \prod_{i=1}^N [P(\mathit{word}_i|\mathit{subj})]}$$

Now that we can give a specific tweet an objectivity, we can easily quantify the likelihood of a given content by the previous word removal in comparison to 1. As 1 is always added to the probabilities. Now, if we know $P(\text{objective}|\text{tweet})$ then we know $P(\text{subjective}|\text{tweet})$ automatically.

$$P(\text{subj}|\text{tweet}) = 1 - P(\text{obj}|\text{tweet})$$

Hence, calculation of $P(\text{obj}|\text{tweet})$ for each tweet is done & this term is used now as a lone attribute in our objective/subjective differentiation.

With this algorithmic approach, there are potentially 2 major problem. The computation becomes more money consuming and time consuming when the list of the clauses used in the data set becomes bigger due to the inclusion of all the single words from the text. Atleast 5 times the words used in our system data is included to address this. It reduces our dictionary from 11,216 to 2,320 for subjective / objective classification. The size of the Unigram Dictionary for positive / negative classification has been reduced from 6,502 to 1,235 words.

The problem in which a particular word appears only in one class and does not occur in another class in our training sets is crucial and is the second one after the former (for example when the word is only mentioned once). For a situation like this, our taxonomy classifies that class into a specific category (other attributes in the tweet doesn't matter) because it has the same name. This method/approach is very stringent and makes for great fit. To avoid this, a technique called "Laplace Smoothing" is used. We replace the formula to calculate the probability of a class name in the following formula:

$$P(\text{word}_1|\text{obj}) = \frac{\text{count}(\text{word}_1 \text{ in obj class}) + x}{\text{count}(\text{total words in obj}) + x(\text{total unique words in obj})}$$

x: arbitrary constant also known as factor of smoothing.

1. How It Works Even though the number of words in a particular class is zero, the probability of a word in some class is not always equal to zero because the fraction has a small value. A more smaller non-zero probability is the one which is replaced with the zero probability instead when it occurs .

Selection of the better attributes from a big pool of attributes is the third and the last problem in the feature/attribute selection. Achievement of the highest precision and accuracy with a small number of attributes is the main goal here. Our classification adds new features to the strength of the problem and hence the difficulty of our classification. This increase in difficulty may be uneven and may be squared, so your preference is given to keep the features small. The other problem we have with many factors is that our training data is too large and can confuse classification when it is separated from an unknown group of tests, thus limiting program accuracy. For solving this problem, we take out the most relevant attributes by calculating the available information for all attributes in the session and marking properties for more information. In this feature selection task, we use the machine learning tool weka. [17].

33 attributes is searched for the classification of objective & subjective class and weka ML tool is used for the calculation of the info gained from each feature. Shown below is the resultant graph:

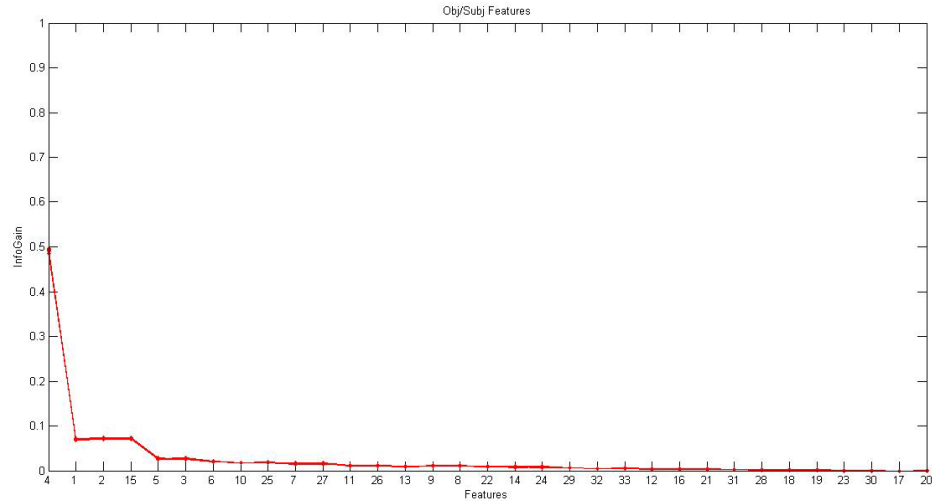


Figure 3: Information Gain of Objectivity / Subjectivity Features

10 of the other graphs are super-imposed in the above graph basically, each of which we have done 10 times from cross validation to fold. Since we have seen that all the graphs overlap, almost every time the results are similar, which conveys that the attributes chosen work successfully in every conditions. The top five attributes which can be taken from this graph are as follows:

1. The Unigram Word Model (for the predictive potential of words related to subjective / subjective classes)
2. URL presence in a tweet
3. The presence of emoticons in a tweet
4. Number of personal pronouns in a tweet
5. Number of exclamation marks in a tweet

Similarly twenty two various properties are taken and searched for the classification of positive(+) and negative(-). ML tool WEKA is used for the calculation of the info gain from these attributes. Shown below is the resultant graph:

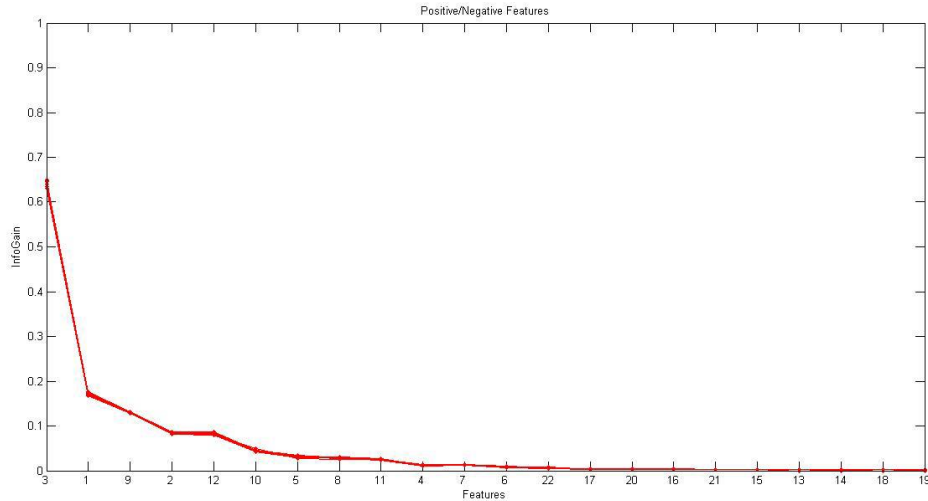


Figure 4: Information Gain of Positive / Negative (Polarity) Features

10 of the other graphs are super-imposed in the above graph basically, each of which we have done 10 times from cross validation to fold. Since we have seen that all the graphs overlap, almost every time the result obtained is similar, which conveys that the attributes work best in every condition. The top five traits has been taken up, 2 of which are recurring symptoms and only 3 traits remain for our positive / negative classification:

1. Unigram Word Model (for potential before positive or negative classes)
2. Number of positive emoticons in a tweet
3. Number of negative emoticons in a tweet

We ignore unnecessary attributes as additional info is not provided among the above attributes:

- Emoticon score for tweet
- MPQA score for Tweet

Classification:

The process of dividing the information given into separate categories by focusing onto some common paradigm obtained in one category, but differing from designs obtained in other categories is known as Pattern Categorization/ Classification. Creation of a process so that all the tweets can be categorized into the following four categories is the main goal here. These categories are: Ambiguous, neutral(0), positive(+) and negative(-).

The analysis of contextual emotion & general emotion are the two types of analysis done in this concept. Particular section of a tweet when categorized accounts for the analysis of contextual emotions, for example, for the tweet "I will eat a fancy meal after eating this ordinary meal for 15 days" 😊. A reference emotion class classification identifies ordinary meal with a -ve impression. And fancy meal with a +ve impression. General sentiment analysis, deals with the general mood of the entire text (in this case the tweet). Since the tweet mentioned earlier is overall positive, definitive common sense classifies it positively. For our particular project we will only deal with the final case, namely a general (complete) analysis of the entire message post. We made inflation categorization to determine the good, the bad or the two (some researchers are included) in both categories and others not installed).

This differentiation system is usually a two-stage approach in this domain. The first objectivity classification takes place, which classifies a message post into classes of objective or subjective. We performed polarity classification (tweets categorized subjectively by objectivity classification) to determine positive, negative, or both (some researchers included both category and some not included). An increase in performance compared to the before single-step method.

A different but effective method is applied here as opposed to the the method used here. The foremost point or step proposed in this that each message post must pass through 2 classification: the classification for objective and the classification for polarity. The first one differentiates a tweet among the above mentioned categories, and the second one among the categories of +ve(positive) and -ve(negative). A small set of attributes are used for these categorizations and then the Naïve Bayes method can be used after applying which and after the foremost step every post has either of zero(0) or one(1). One number represents the

possibility of the extent relation of the post with respect to the objectivity and the other shows the possibility of the extent of relation of the post with respect to positivity. As The calculation of the remaining probability of subjectivity of a post is simple by using easy difference therefore the other possibilities are not needed.

The other remaining numbers are considered here as a different attribute for this other differentiation in this step, with the attribute set taken as 2 . ML tool WEKA is used and the following ML algos are applied for this other differentiation in order to get a better outcome:

- K-Means Clustering
- Support Vector Machine
- Logistic Regression
- K Nearest Neighbours
- Naive Bayes
- Rule Based Classifiers

To comprehend the process or the working of the task, a plot from the real test group is displayed from one of our cross-verifications or on the 2-dimensional space below:

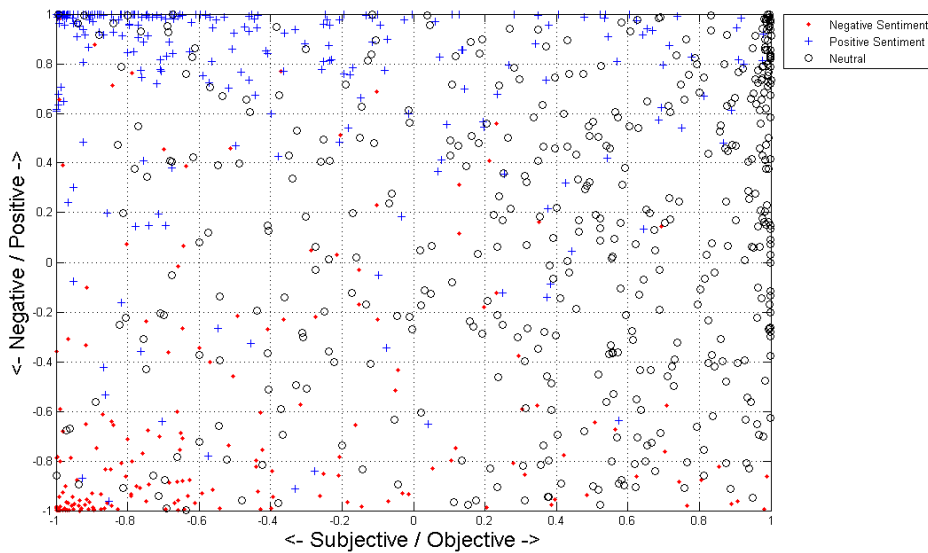


Figure 5: 2-d Scater Plot after Step 1

The labels in this image represent the distribution of how true ground truths and assorted data points are actually scattered across space. When we go right, tweets become a growing target and tweets become positive as we move up. The results of our classification system are described in the next section of this report.

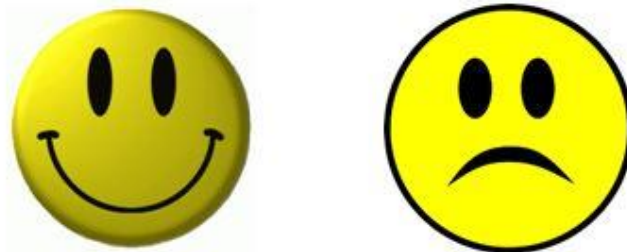
CHAPTER-4

PERFORMANCE ANALYSIS

TWITTER API:

The Twitter API allows you to use Twitter's features without having to go through the website interface. It can be used for posting a tweet or sending a guided message in an automated way by posting scripts.

Say for example that you're chatting with Twitter, and people are asked to receive a personal reminder tweet before getting started. If there are a hundred, it can take a lot of work to manually tweet everyone. However, with a list of usernames and a script that accesses Twitter via the API, you can automatically send reminder tweets so that it is completed quickly and easily.



TWEEPY: THE LIBRARY OF PYTHON FOR THE API OF TWITTER

To install the latest version from PyPI is by using pip:

- `pip install tweepy`

This example will download your home timeline tweets and print each one of their texts. Twitter requires all requests to use for authentication.

```
import tweepy

auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token (access_token, access_token_secret)

api = tweepy.API(auth)

public_tweets = api.home_timeline()
for tweet in public_tweets:
    print(tweet.text)
```

Fig from [21]

Models:

When we start an API method, when it returns to us, the Tweepy Model class is an example. It contains the returned data from Twitter that can be used within our application. For example the following code gives us a user model:

```
# Get the User object for twitter...
user = tweepy.api.get_user('twitter')
```

Fig from [21]

The model contains data and some assistive methods we can use:

```
print user.screen_name
print user.followers_count
for friend in user.friends():
    print friend.screen_name
```

Fig from [21]

TextBlob:

TextBlob provides a simple API to use the Python library and approaches to accomplish various NLP functions.

They are like the Python Strings that is one of the best things about TextBlob. Now, we may change it as we performed on Python. Below are some funtions.

```
string1 = TextBlob("Analytics")  
string1[1:5]    ### extracting 1 to 5 letters  
TextBlob("naly")
```

```
string1.upper() ## to upper case the entire text  
TextBlob("ANALYTICS")
```

```
string2 = TextBlob("Vidhya")
```

```
## concat two sentences similar as python  
string1 + " " + string2  
TextBlob("Analytics Vidhya")
```

Fig from [22]

Intalling:

In a simple task, to set up a textblob on your system, we have to only open the Anaconda prompt and implement the below command:

```
pip install -U textblob
```

Fig from [22]

Tokenization:

Tokenization refers to the division of a paragraph in the series of tokens, corresponding the letter. That's the usage of the N L P. We need to follow below commands for the usage of TextBlob:

1. Creation of an object & passing the strings through it.
2. TextBlob works to perform a specific task.

```
from textblob import TextBlob

blob = TextBlob("Analytics Vidhya is a great platform to learn data science. \n It
helps community through blogs, hackathons, discussions,etc.")
```

Fig from [22]

Noun Phrase Extraction

Since we have summarized the words, we can only take out the phrase of noun instead. Taking out of the noun phrase should be very important if we wanted the analization of “who” . Let us take an eg.

```
blob = TextBlob("Analytics Vidhya is a great platform to learn data science.")

for np in blob.noun_phrases:

    print (np)

>> analytics vidhya

great platform

data science
```

Fig from [22]

Part- of- speech Tagging

POS technique of tagging of identifying text by words based on how it is defined and in context. Under simpler word, a word noun either adj. or verbb is referred. That's the full version of the removal of the phrases of noun, in which we want to find a sentence-part in a sentence.

Let's examine the textblobtags

```
for words, tag in blob.tags:  
  
    print (words, tag)  
  
>> Analytics NNS  
  
Vidhya NNP  
  
is VBZ  
  
a DT  
  
great JJ  
  
platform NN  
  
to TO  
  
learn VB  
  
data NNS
```

Fig from [22]

Words Inflection & Lemmatization

To send the grammar meanings Inflection the word-forming method, the letters are attached to the root word is used. The method of inflection is easy, that is, letters you take from the textBlob may easily converted to single form or more than one form.

```
blob = TextBlob("Analytics Vidhya is a great platform to learn data science. \n It
helps community through blogs, hackathons, discussions,etc.")

print (blob.sentences[1].words[1])

print (blob.sentences[1].words[1].singularize())

>> helps

help
```

Fig from [22]

The built-in object in Word form is also available in the TextBlob library. We make the object of a letter & then we run the method:

```
from textblob import Word

w = Word('Platform')

w.pluralize()

>>'Platforms'
```

Fig from [22]

```
## using tags

for word,pos in blob.tags:

    if pos == 'NN':

        print (word.pluralize())

>> platforms

sciences
```

Fig from [22]

Words can be lemmatized using the function of lemmatize.

```
## lemmatization

w = Word('running')

w.lemmatize("v") ## v here represents verb

>> 'run'
```

Fig from [22]

N-grams

The grouping of several letters is n-gram. n-gram (n is greater than one) is usually better than letters and language modeling may be done through it. It may be accessed easy by making use of the n-gram method that gives us the node of N ordinal terms.

```
for ngram in blob.ngrams(2):  
  
    print (ngram)  
  
>> ['Analytics', 'Vidhya']  
  
['Vidhya', 'is']  
  
['is', 'a']  
  
['a', 'great']  
  
['great', 'platform']  
  
['platform', 'to']  
  
['to', 'learn']  
  
['learn', 'data']  
  
['data', 'science']
```

Fig from [22]

Implementation:

```
import sys,tweepy, csv, re
from textblob import TextBlob
import matplotlib.pyplot as plt

class SentimentAnalysis:

    def __init__(self):
        self.tweets = []
        self.tweetText = []

    def DownloadData(self):
        # authenticating
        consumerKey = 'xphfjVj8xtoAJvt14h6Evj77'
        consumerSecret = 'aheilY0UnTIYgfXpOCszlNujzB3tQgBnATfMp9gnOY1gFT3Sv'
        accessToken = '1263128586396921856-huxDeKAlmVLoPyUN5k0dpyxGSAU2vF'
        accessTokenSecret = '3R4cx8KzIYzMrhc5YAcD19DWS8bcbiB8SAdUm0Bsn6Ct8'
        auth = tweepy.OAuthHandler(consumerKey, consumerSecret)
        auth.set_access_token(accessToken, accessTokenSecret)
        api = tweepy.API(auth)

        # input for term to be searched and how many tweets to search
        searchTerm = input("Enter Keyword/Tag to search about: ")
        NoOfTerms = int(input("Enter how many tweets to search: "))
```

```
# searching for tweets
self.tweets = tweepy.Cursor(api.search, q=searchTerm, lang = "en").items(NoOfTerms)

# Open/create a file to append data to
csvFile = open('result.csv', 'a')

# Use csv writer
csvWriter = csv.writer(csvFile)

# creating some variables to store info
polarity = 0
positive = 0
wpositive = 0
spositive = 0
negative = 0
wnegative = 0
snegative = 0
neutral = 0
```

```

# iterating through tweets fetched
for tweet in self.tweets:
    #Append to temp so that we can store in csv later. I use encode UTF-8
    self.tweetText.append(self.cleanTweet(tweet.text).encode('utf-8'))
    # print (tweet.text.translate(non_bmp_map)) #print tweet's text
    analysis = TextBlob(tweet.text)
    # print(analysis.sentiment) # print tweet's polarity
    polarity += analysis.sentiment.polarity # adding up polarities to find the average later

    if (analysis.sentiment.polarity == 0): # adding reaction of how people are reacting to find
average later
        neutral += 1
    elif (analysis.sentiment.polarity > 0 and analysis.sentiment.polarity <= 0.3):
        wpositive += 1
    elif (analysis.sentiment.polarity > 0.3 and analysis.sentiment.polarity <= 0.6):
        positive += 1
    elif (analysis.sentiment.polarity > 0.6 and analysis.sentiment.polarity <= 1):
        spositive += 1
    elif (analysis.sentiment.polarity > -0.3 and analysis.sentiment.polarity <= 0):
        wnegative += 1
    elif (analysis.sentiment.polarity > -0.6 and analysis.sentiment.polarity <= -0.3):
        negative += 1
    elif (analysis.sentiment.polarity > -1 and analysis.sentiment.polarity <= -0.6):
        snegative += 1

```

```
# Write to csv and close csv file
    csvWriter.writerow(self.tweetText)
    csvFile.close()

# finding average of how people are reacting
    positive = self.percentage(positive, NoOfTerms)
    wpositive = self.percentage(wpositive, NoOfTerms)
    spositive = self.percentage(spositive, NoOfTerms)
    negative = self.percentage(negative, NoOfTerms)
    wnegative = self.percentage(wnegative, NoOfTerms)
    snegative = self.percentage(snegative, NoOfTerms)
    neutral = self.percentage(neutral, NoOfTerms)

# finding average reaction
    polarity = polarity / NoOfTerms

# printing out data
    print("How people are reacting on " + searchTerm + " by analyzing " + str(NoOfTerms) + " tweets.")
    print()
    print("General Report: ")
```



```

if (polarity == 0):
    print("Neutral")
elif (polarity > 0 and polarity <= 0.3):
    print("Weakly Positive")
elif (polarity > 0.3 and polarity <= 0.6):
    print("Positive")
elif (polarity > 0.6 and polarity <= 1):
    print("Strongly Positive")
elif (polarity > -0.3 and polarity <= 0):
    print("Weakly Negative")
elif (polarity > -0.6 and polarity <= -0.3):
    print("Negative")
elif (polarity > -1 and polarity <= -0.6):
    print("Strongly Negative")
print()

print("Detailed Report: ")
print(str(positive) + "% people thought it was positive")
print(str(wpositive) + "% people thought it was weakly positive")
print(str(spositive) + "% people thought it was strongly positive")
print(str(negative) + "% people thought it was negative")
print(str(wnegative) + "% people thought it was weakly negative")
print(str(egative) + "% people thought it was strongly negative")
print(str(neutral) + "% people thought it was neutral")
self.plotPieChart(positive, wpositive, spositive, negative, wnegative, snegative, neutral, searchTerm,
NoOfTerms)

```

```

def cleanTweet(self, tweet):
    # Remove Links, Special Characters etc from tweet
    return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t]) | (\w +:\ / \ / \S +)", "", tweet).split())

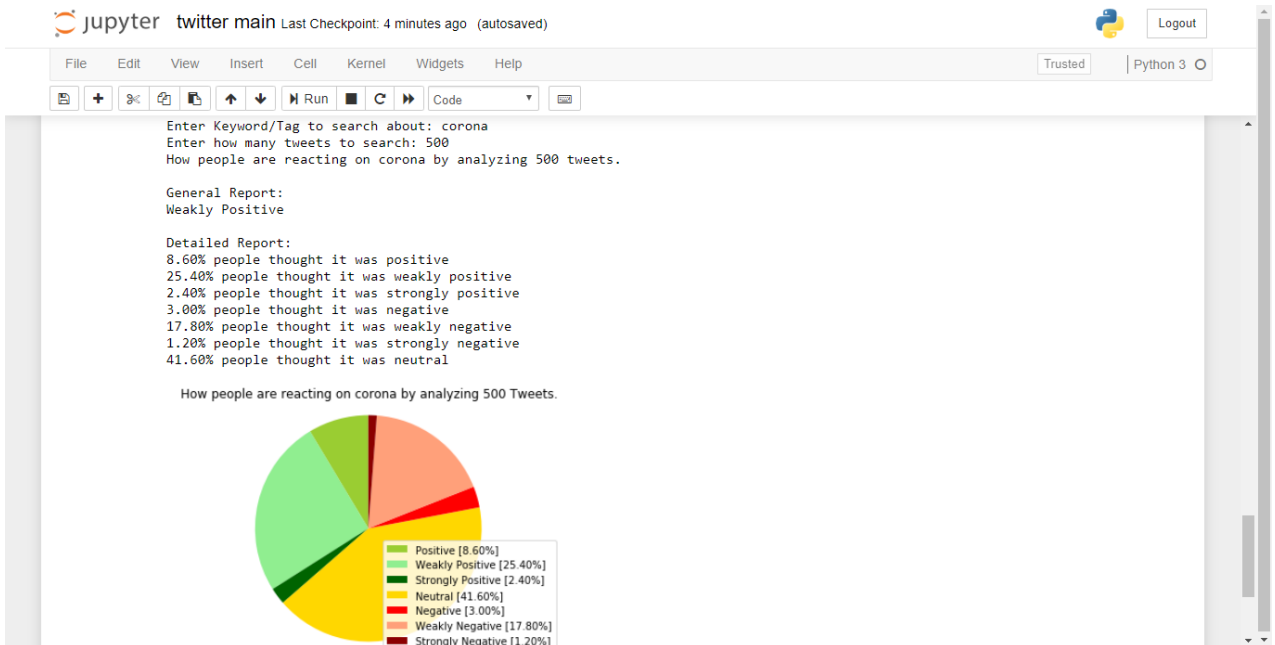
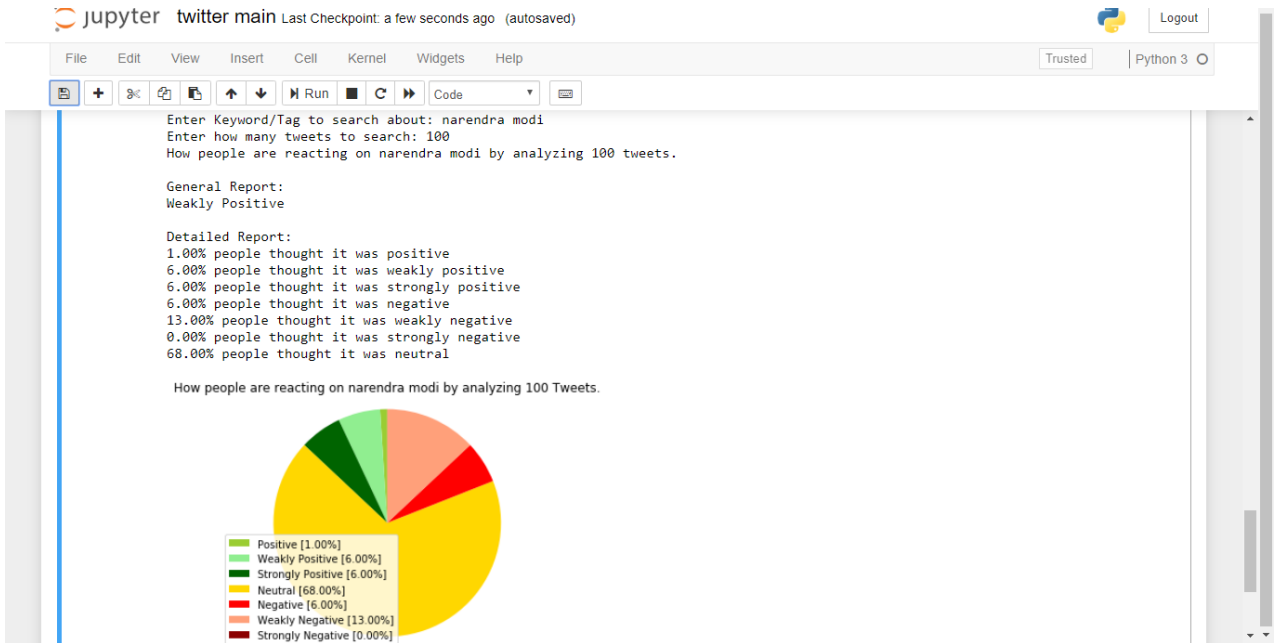
# function to calculate percentage
def percentage(self, part, whole):
    temp = 100 * float(part) / float(whole)
    return format(temp, '.2f')

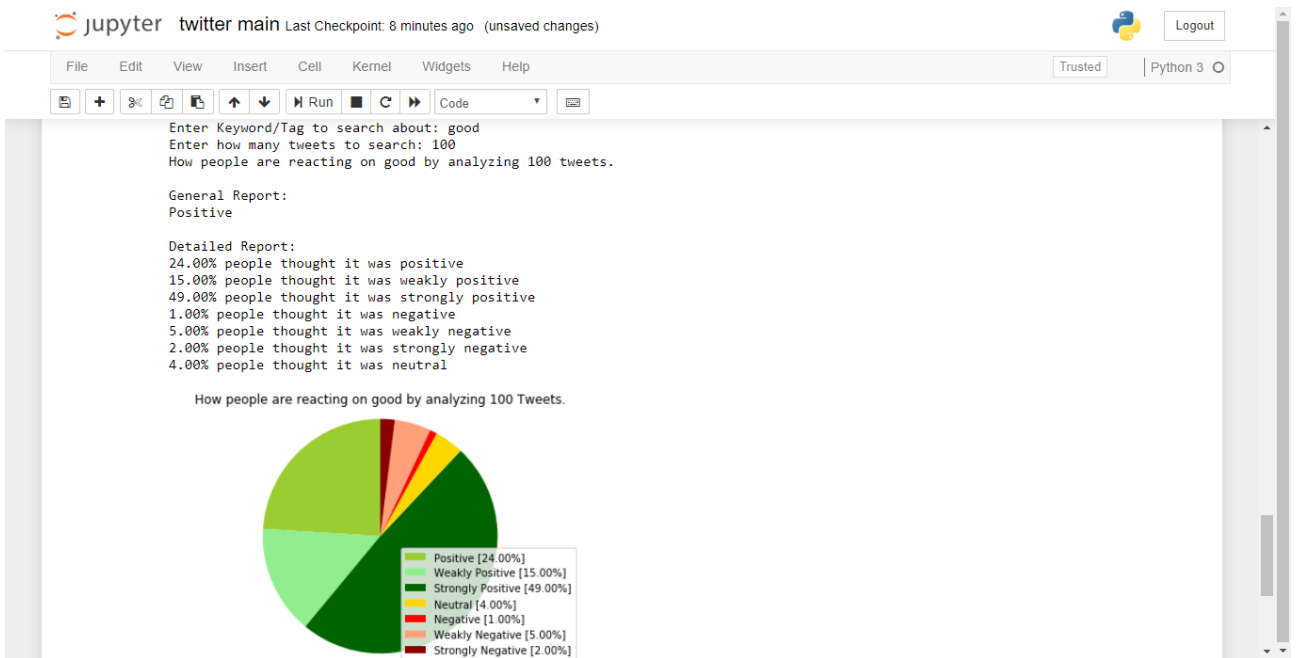
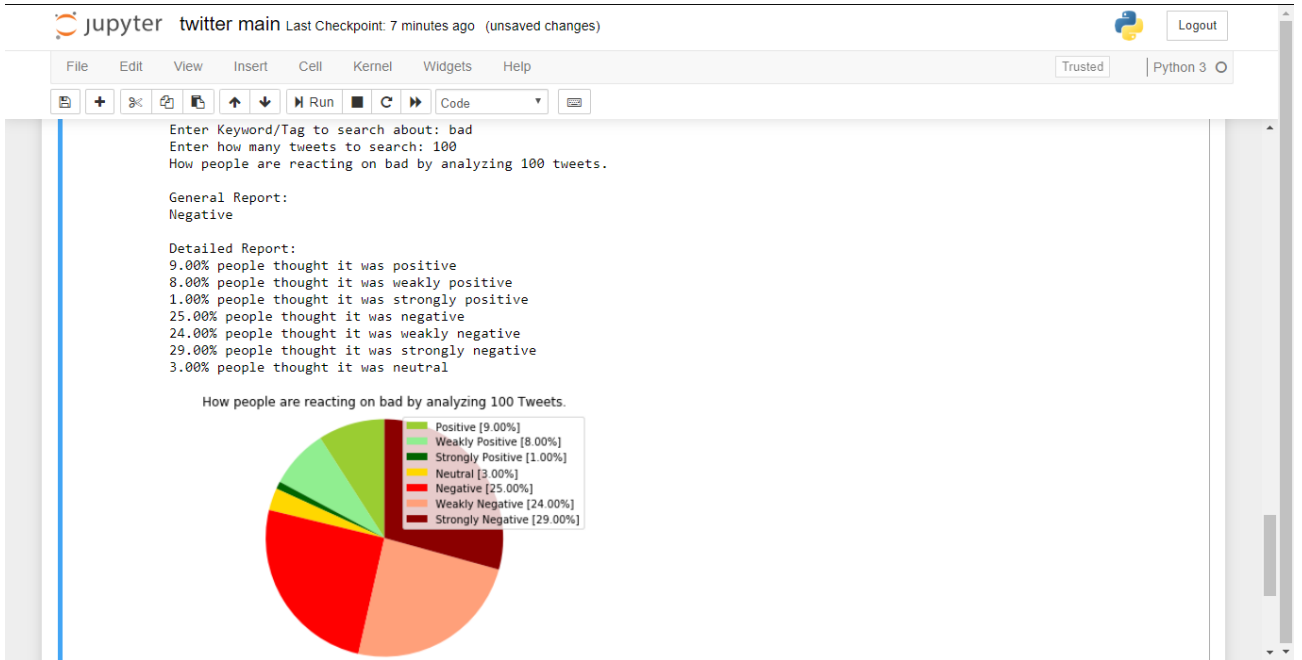
def plotPieChart(self, positive, wpositive, spositive, negative, wnegative, snegative, neutral,
searchTerm, noOfSearchTerms):
    labels = ["Positive [' + str(positive) + '%]", "Weakly Positive [' + str(wpositive) + '%]", "Strongly
Positive [' + str(spositive) + '%]", "Neutral [' + str(neutral) + '%]",
            "Negative [' + str(negative) + '%]", "Weakly Negative [' + str(wnegative) + '%]", "Strongly
Negative [' + str(snegative) + '%]"]
    sizes = [positive, wpositive, spositive, neutral, negative, wnegative, snegative]
    colors = ['yellowgreen', 'lightgreen', 'darkgreen', 'gold', 'red', 'lightsalmon', 'darkred']
    patches, texts = plt.pie(sizes, colors=colors, startangle=90)
    plt.legend(patches, labels, loc="best")
    plt.title('How people are reacting on ' + searchTerm + ' by analyzing ' + str(noOfSearchTerms) + '
Tweets.')
    plt.axis('equal')
    plt.tight_layout()
    plt.show()

if __name__ == "__main__":
    sa = SentimentAnalysis()
    sa.DownloadData()

```

Results:





CHAPTER-5

CONCLUSIONS

Under the field of microblogging the work is underdeveloped and is incomplete especially the work of SA. Therefore, we suggest some of the approaches which thought would be worth checking with time and can lead to better performance.

Currently we only work with very simple Unigram models; it may be improved through additional knowledge such as word proximity with negative words. We can specify a window before the word under consideration (for example a window may be two - three letters) and effective -ve can be included in this model when that window is present. Close to the word negativity, the older the words, the earlier the polarity must be calculated, which greatly affects the polarity. Foregg, we may reciprocate polarity of letter while a rejection, its letter next and remove negative letter, and if the word has little effect.

Further, the current focus is on the Uni-grams thus an impact on Bi-grams and tri-grams. Which is mentioned in the chapter2, it generally improves performance when using the bigrams with unigrams.

However, in order to be an effective feature for Bigrams and Trigrams, our measurement requires a labeled data set of more than 9,000 tweets.

At last a try can be done to build humanconfidence on system. Foregg. We may make every labeled tweert into a 2d +vity/-vity frame iff there are five human labels

Yes, only only four are happy. A result was made through more number of votes. People can make their own methods to come abovewith customized class bondaries, where all 5 labels are given the maximum weightage for accepted tweets, and the number of deals begins to decrease, so that the weight is allotted. Thus the humanfaith cause may be imagined on SA.

- [15] Stefano Baccianella, Andrea Esuli, Fabrizio Sebastiani. SENTIWORDNET 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. *In Proceedings of international conference on Language Resources and Evaluation (LREC)*, 2010.
- [16] Theresa Wilson, Janyce Wiebe and Paul Hoffmann. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. *In the Annual Meeting of Association of Computational Linguistics: Human Language Technologies (ACL-HLT)*, 2005.
- [17] Ian H. Witten, Eibe Frank & Mark A. Hall. Data Mining – Practical Machine Learning Tools and Techniques.
- [19] Ricgard O. Duda, Peter E. Hart & David G. Stork: Pattern Classification.
- [20] Steven Bird, Ewan Klein & Edward Loper. Natural Language Processing with Python.
- [21] <https://medium.com/@jasonrigden/tweetpt-a-python-library-for-the-twitter-api-9d0537dcebd4>
- [22] <https://www.analyticsvidhya.com/blog/2018/02/natural-language-processing-for-beginners-using-textblob/>

TWITTER SENTIMENTAL ANALYSIS

ORIGINALITY REPORT

6%

SIMILARITY INDEX

5%

INTERNET SOURCES

1%

PUBLICATIONS

3%

STUDENT PAPERS

PRIMARY SOURCES

1

docplayer.net

Internet Source

4%

2

Submitted to Higher Education Commission
Pakistan

Student Paper

1%

3

Submitted to Bahcesehir University

Student Paper

<1%

4

arxiv.org

Internet Source

<1%

Exclude quotes Off

Exclude matches Off

Exclude bibliography Off