**PROJECT REPORT**

**Topic: Twitter Sentiment Analysis**

by

**Robin Thomas (17BCE1330)**

**Submitted to: Prof.SHRIDEVI S**

**CSE3024 – Web Mining**

****

***ABSTRACT***

Twitter is a well-known online social network and micro-blogging service which is known to allow users to post and interact with messages which are known as tweets. Twitter has roughly about 330 monthly active users and 500 million tweets are sent on average per day which adds up to a staggering amount of data. With these vast numbers, there is a need to manage and classify data efficiently. That’s where the concept of sentiment analysis comes in. Sentiment analysis, in simple words, refers to the classification of text based on the mood expressed by the user, known as sentiment. Analysis these sentiments is important for many applications like firms trying to find the response of products in the market, predicting political propaganda and stock exchange. Sentiment analysis can be achieved for the following applications by analyzing overall public sentiment towards that particular application with respect to time and finding the correlation between the public sentiment and the application’s value in the market. This could lead to three possibilities namely, positive response, neutral response and negative response. To validate the process of sentiment analysis and to predict its overall performance, the system should have categorized the data in accordance with the intuition of the user. The aim of our chosen project is to implement sentiment analysis that fairly accurately classifies the generated tweets into positive, neutral and negative responses.

***INTRODUCTION***

Twitter is a very useful platform and we have chosen to work with Twitter because it makes classification easier compared to other social media and websites.

Sentiment analysis is a broader topic and has many applications. It is very useful when we need to compute customer satisfaction metrics or when needed to identify detractors and promoters. It can also be used to find people who are happy with the services offered by a particular product. It has macro-scale applications like predicting the stock market rate of a particular firm.

Sentiment analysis falls under the domain of Pattern Classification and Data Mining. Sentiment analysis is also often called opinion mining. Sentiment analysis deals with predicting the mood of the person based on classifying tweets and using natural language processing (NLP) techniques. For example, let’s say the word ‘amazing’ means that the person is in a happy state while the word ‘uneasy’ can mean the person is not so happy about it. By using NLP techniques, we differentiate between the important words and the unnecessary words which can be used further for classification. Also, since the tweet can contain special characters like hashtags, we also are eliminating these delimiters for accurate classification of the text.

***BACKGROUND***

After analyzing 20 research papers, Here’s a look at some of the advantages and disadvantages.

* One of the research papers focused on how sentiment analysis could be used in predicting the Indian elections. A Twitter archiver tool was used to get tweets in the Hindi language. Data mining algorithms were performed on 42,235 tweets collected over a month referencing 5 political parties of India. A dictionary based Naïve Bayes Classifier was used along with SVM algorithm to build the classifier. The classifier classified tweets into positive, negative and neutral. And based on the results, the percentage was computed which predicted a 78.4% chance of BJP winning in the next election.
* Another research paper focused on sentiment analysis for the Uri terror attack. The sentiments that hinted at post-attack aftermath and survivors were analyzed and information such as number of retweets and the number of likes for each post helped at studying the information flow of data posted on Twitter. Also, this event caused a widespread reaction on other social media.
* Emoticons were utilized in another research paper to reduce dependency in machine learning techniques for sentiment classification. This paper aims to provide a good match between the domain and the time. They provided preliminary experiments with training data labelled with emoticons and has the potential of being independent of domain, topic and time.
* A real time classifier was built in another research paper. They propose a real time Twitter sentiment analysis and classification. They use queries and they get a graphical representation of the polarity of the tweets. Out of various algorithms, they have made use of Simple Voter and Naïve Bayes algorithm and have constructed a 3-way classifier. They have also shown that the system is accurate and efficient when Naïve Bayes algorithm is used.
* A novel hidden Naïve Bayes classifier was proposed in another research paper. According to them, Naïve Bayes classifier fails when datasets have a strong correlation between attributes due to the conditional independence assumption is not always true in the real world. In the latest Hidden Naïve Bayes algorithm, each attribute corresponds to a hidden parent that influences all other attributes. They also introduce a novel model PHNB (Packaged Hidden Naïve Bayes) and that PHNB significantly reduces the test time on many high dimensional datasets and has higher accuracy on some particular datasets.
* Another research paper introduces a novel solution to target-oriented sentiment summarization and sentiment analysis of short informal texts with a main focus on tweets. They introduce a hybrid polarity detection system and according to them, that system outperforms unigram-state-of-the-art baseline. It also exhibits high performance with various useful functionalities and features.
* More research papers were analyzed similarly and we found out, first hand, that sentiment analysis had a broader perspective and a wide range of applications that can solve most of the commonly faced issues in India and other countries. Not only that, it can create more and more opportunities for data scientists and other machine learning experts and for them to create better technology to better suit the current needs that will evolve over time. Sentiment analysis has been constantly evolving from the past decade and will continue to do so with the passage of time.

**PROPOSED ARCHITECTURE**

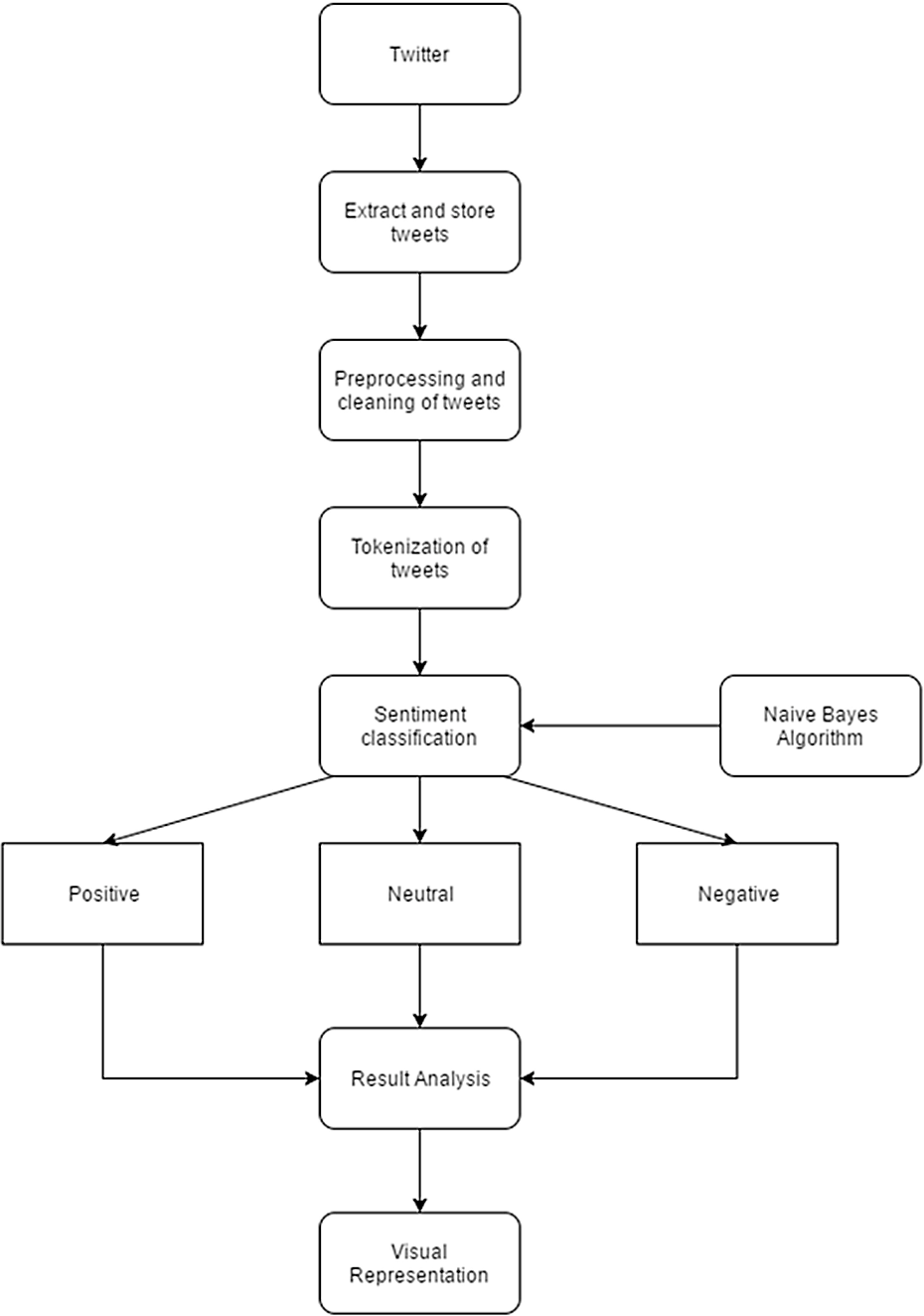
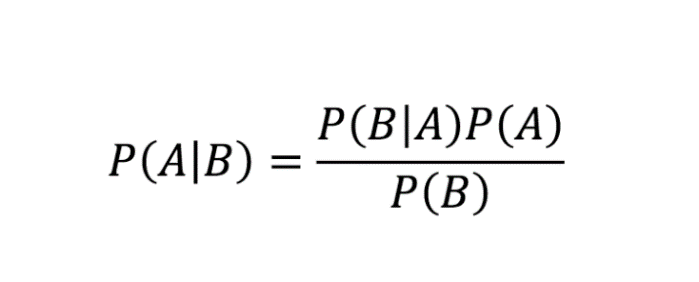
****

Figure 1 Architecture Diagram

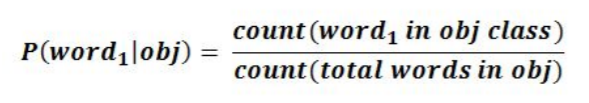
We have used a Naïve Bayes classifier for our project to visualize the sentiment analysis. The Naïve Bayes classifier works as follows:

A dataset is considered. The dataset will be divided into 2 parts, namely the feature matrix and the response vector. Feature matrix contains all vectors (rows) of the dataset in which each vector are dependent upon each other. The response vector, on the other hand, contains the value of class variable (the prediction or the output) for each row of the feature matrix.

The fundamental Naïve Bayes makes an assumption that each feature makes an independent and equal contribution to the outcome. These assumptions may not be true in the real world. Naïve Bayes is based on the Bayes theorem which finds the probability of an event occurring given the probability of another event that already occurred. The equation for Bayes theorem is given below:



Now, the response vector consists of both positive and negative features and hence Naïve Bayes must be considered for each of these types. First, the probability of a word in the training data belonging to a particular class using the below formula:



Then the probability of the tweet with respect to objective is calculated based on the Bayes theorem stated above.

Each feature in the feature matrix is represented as an array of objects and the Bayes theorem is applied for each type of response vector. The probabilities for each type of response vector is calculated simultaneously. Once the Bayes theorem is applied for each feature in the feature matrix against each type of response vector, the corresponding probabilities are multiplied together. The resulting values are then compared and whichever has a higher probability, is the most likely to occur.

Twitter has an inbuilt API that allows users to extract a maximum of 3200 tweets. But we need to get authorized by Twitter to be able to use their API. So first we created an account on Twitter and logged in to the Twitter developers’ section and applied for an access token which can be integrated with Python libraries to extract the tweets from Twitter. The authentication took us 2 days following which we were ready to extract tweets.

**IMPLEMENTATION**

The implementation is done using python and flask for the website. The code uses a sample training dataset for the Naïve Bayes classifier to understand the distinction between positive and negative tweets and then sets on to labelling words present as positive and negative which are then assessed so as to imply whether the overall sentiment is positive or negative. The website built purely using HTML, CSS and bootstrap is used for the User Interface where users can give a particular query and search for positive tweets under that query or negative tweets which are distinguished and displayed using the Naïve Bayes Classifier.

**Code:-**

**1.Main Script:-**

#Author Robin Thomas

from flask import Flask,render\_template,request,jsonify

import tweepy

from textblob import TextBlob

consumer\_key = 'A3tMsP66iCW5Gqeja8xiwOkmD'

consumer\_secret = 'qjw9CTugWwwrvWtyHrDgvPahE1RQmQUqWBQFWod6dWSAhsSwBI'

access\_token = '3052501375-FY3TxCbMMZRq0dSJO9SZ5KENnwsYoeCPBYl8jHX'

access\_token\_secret = 'ugi79nBbiY3BYaq758HZCyopOCs4wgDdML0pEK5Ntukmn'

auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret)

auth.set\_access\_token(access\_token, access\_token\_secret)

api = tweepy.API(auth)

#-------------------------------------------------------------------------

app = Flask(\_\_name\_\_)

@app.route("/")

def index():

    return render\_template('index.html')

@app.route("/search",methods=["POST"])

def search():

    search\_tweet = request.form.get("search\_query")

    t = []

    tweets = api.search(search\_tweet, tweet\_mode='extended')

    for tweet in tweets:

        polarity = TextBlob(tweet.full\_text).sentiment.polarity

        subjectivity = TextBlob(tweet.full\_text).sentiment.subjectivity

        t.append([tweet.full\_text,polarity,subjectivity])

    return jsonify({"success":True,"tweets":t})

class TwitterClient(object):

    '''

    Generic Twitter Class for sentiment analysis.

    '''

    def \_\_init\_\_(self):

        '''

        Class constructor or initialization method.

        '''

        '''# keys and tokens from the Twitter Dev Console

        consumer\_key = 'qBUPcn9g2BDgn4YuIThH1uamj'

        consumer\_secret = 'K3nrtfYWMMm5cgQKxmGNKc0hWRKEsvu4gi8LSKR4i4n2nCqRWl'

        access\_token = '1183353803354980352-XcB4ddx8348bgPV8pIMgWg347A3t3Y'

        access\_token\_secret = '25k7k8LMS8eTesUJUmbeFbsNTA87uly0Qvuzw0Ho4IyFT'

        # attempt authentication

        try:

            # create OAuthHandler object

            self.auth = OAuthHandler(consumer\_key, consumer\_secret)

            # set access token and secret

            self.auth.set\_access\_token(access\_token, access\_token\_secret)

            # create tweepy API object to fetch tweets

            self.api = tweepy.API(self.auth)

        except:

            print("Error: Authentication Failed") '''

    def clean\_tweet(self, tweet):

        '''

        Utility function to clean tweet text by removing links, special characters

        using simple regex statements.

        '''

        return ' '.join(re.sub("""(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])

                                    |(\w+:\/\/\S+)""", " ", tweet).split())

    def get\_tweet\_sentiment(self, tweet):

        '''

        Utility function to classify sentiment of passed tweet

        using textblob's sentiment method

        '''

        analysis = TextBlob(self.clean\_tweet(tweet))

        if analysis.sentiment.polarity > 0:

            return 'positive'

        elif analysis.sentiment.polarity == 0:

            return 'neutral'

        else:

            return 'negative'

    def get\_tweets(self, query, count = 10):

        '''

        Main function to fetch tweets and parse them.

        '''

        # empty list to store parsed tweets

        tweets = []

        try:

            # call twitter api to fetch tweets

            fetched\_tweets = self.api.search(q = query, count = count)

            # parsing tweets one by one

            for tweet in fetched\_tweets:

                # empty dictionary to store required params of a tweet

                parsed\_tweet = {}

                # saving text of tweet

                parsed\_tweet['text'] = tweet.text

                # saving sentiment of tweet

                parsed\_tweet['sentiment'] = self.get\_tweet\_sentiment(tweet.text)

                # appending parsed tweet to tweets list

                if tweet.retweet\_count > 0:

                    # if tweet has retweets, ensure that it is appended only once

                    if parsed\_tweet not in tweets:

                        tweets.append(parsed\_tweet)

                else:

                    tweets.append(parsed\_tweet)

            # return parsed tweets

            return tweets

        except tweepy.TweepError as e:

            # print error (if any)

            print("Error : " + str(e))

def MAIN():

    # creating object of TwitterClient Class

    api = TwitterClient()

    # calling function to get tweets

    tweets = api.get\_tweets(query = 'Donald Trump', count = 200)

    # picking positive tweets from tweets

    ptweets = [tweet for tweet in tweets if tweet['sentiment'] == 'positive']

    # percentage of positive tweets

    print("Positive tweets percentage: {} %".format(100\*len(ptweets)/len(tweets)))

    # picking negative tweets from tweets

    ntweets = [tweet for tweet in tweets if tweet['sentiment'] == 'negative']

    # percentage of negative tweets

    print("Negative tweets percentage: {} %".format(100\*len(ntweets)/len(tweets)))

    # percentage of neutral tweets

    print("Neutral tweets percentage: {} % \ ".format(100\*(len(tweets)-len(ntweets)-len(ptweets))/len(tweets)))

    analyser = SentimentIntensityAnalyzer() #using object for sentiment intensity analyzer

    # printing first 5 positive tweets

    print("\n\nPositive tweets:")

    for tweet in ptweets[:10]:

        print(tweet['text'])

        sentiment\_dict = analyser.polarity\_scores(tweet['text'])   #dictionary for which contains the scores

        print(str(sentiment\_dict))

    # printing first 5 negative tweets

    print("\n\nNegative tweets:")

    for tweet in ntweets[:10]:

        print(tweet['text'])

        sentiment\_dict = analyser.polarity\_scores(tweet['text'])

        print(str(sentiment\_dict))

app.run()

**2.Website script:-**

<!DOCTYPE html>

<html lang="en">

<!-- Developed By Robin Thomas-->

    <head>

        <meta charset="utf-8">

        <meta http-equiv="X-UA-Compatible" content="IE=edge">

        <meta name="author" content="Robin Thomas">

        <meta name="viewport" content="width=device-width, initial-scale=1">

        <title>Tweet Sentiment Analysis</title>

        <link rel="stylesheet" href="http://fonts.googleapis.com/css?family=Roboto:400,100,300,500">

        <link rel="stylesheet" href="static/bootstrap/css/bootstrap.min.css">

        <link rel="stylesheet" href="static/font-awesome/css/font-awesome.min.css">

        <link rel="stylesheet" href="static/css/form-elements.css">

        <link rel="stylesheet" href="static/css/style.css">

    </head>

    <body>

        <div class="top-content">

            <div class="inner-bg">

                <div class="container">

                    <div class="row">

                        <div class="col-sm-10 col-sm-offset-1 text">

                            <h1><strong>Tweets</strong> Analysis</h1>

                            <h2 style="color: white;"><strong>Developed By Robin Thomas</strong></h2>

                            <div class="description">

                                <p>

                                    Search for your favorite topic in twitter, using <strong>Sentiment</strong> filter!

                                </p>

                            </div>

                        </div>

                    </div>

                    <div class="row">

                        <div class="col-sm-12  form-box">

                            <div class="form-top">

                                <div class="form-top-left">

                                    <h3>What tweets you want to see!</h3>

                                    <p>Use the sentiment and factual filters below </p>

                                </div>

                                <div class="form-top-right">

                                    <i class="fa fa-twitter" style="color:#38A1F3;"></i>

                                </div>

                            </div>

                            <div class="form-bottom">

                                <form role="form" id="form" class="login-form">

                                    <div class="form-group">

                                        <label class="sr-only" for="form-username">Username</label>

                                        <input type="text" name="tweet\_search" placeholder="Enter a search topic" class="form-username form-control" id="form-username">

                                    </div>

                                    <div class="row search\_tweets">

                                                 <div class="col-sm-offset-3 col-sm-6">

                                                       <button id="search\_btn" class="btn">Search Tweets</button>

                                                 </div>

                                        </div>

                                    <div class="row reset\_filter">

                                                 <div class="col-sm-offset-3 col-sm-6">

                                                       <button data-toggle="tooltip" data-placement="top" title="Show all Tweets"  class="btn">Reset Filters</button>

                                                 </div>

                                        </div>

                                    <div class="row">

                                        <div class="col-sm-4 col-sm-offset-1">

                                            <div class="row">

                                                 <div class="col-sm">

                                                      <button  data-toggle="tooltip" data-placement="left" title="Show only Positive Tweets" class="btn" id="psenti">Positive Tweets</button>

                                                 </div>

                                                 <div class="col-sm">

                                                        <button  data-toggle="tooltip" data-placement="left" title="Show only Negative Tweets"  class="btn" id="nsenti">Negative Tweets</button>

                                                 </div>

                                            </div>

                                        </div>

                                        <div class="col-sm-4 col-sm-offset-2">

                                            <div class="row">

                                                 <div class="col-sm">

                                                      <button data-toggle="tooltip" data-placement="left" title="Tweets which are Opinion"  id="opi" class="btn">Opinion Tweet</button>

                                                 </div>

                                                 <div class="col-sm">

                                                        <button  data-toggle="tooltip" data-placement="left" title="Tweets which are factual"  id="fac" class="btn">Factual Tweet</button>

                                                 </div>

                                            </div>

                                        </div>

                                    </div>

                                </form>

                            </div>

                        </div>

                    </div>

                    <div id="search\_result">

                        <ol id="search\_list">

                        </ol>

                    </div>

                </div>

            </div>

        </div>

        <script src="static/js/jquery-1.11.1.min.js"></script>

        <script src="static/bootstrap/js/bootstrap.min.js"></script>

        <script src="static/js/jquery.backstretch.min.js"></script>

        <script src="static/js/scripts.js"></script>

        <script src="{{ url\_for('static',filename='index.js') }}"></script>

    </body>

</html>

**RESULTS AND DISCUSSION**

Performance Metrics:

There are many ways in which we can accurately obtain the performance metrics for evaluating a classifier and to understand how good a sentiment analysis model is. One of the most frequently used is known as *cross-validation*. Precision, recall, and accuracy are standard metrics used to evaluate the performance of a classifier. In order to compare all the existing sentiment analysis techniques, their accuracy was used as a metric for comparison.

Graphs depicting the different algorithms and their accuracy for performing Twitter Sentiment Analysis

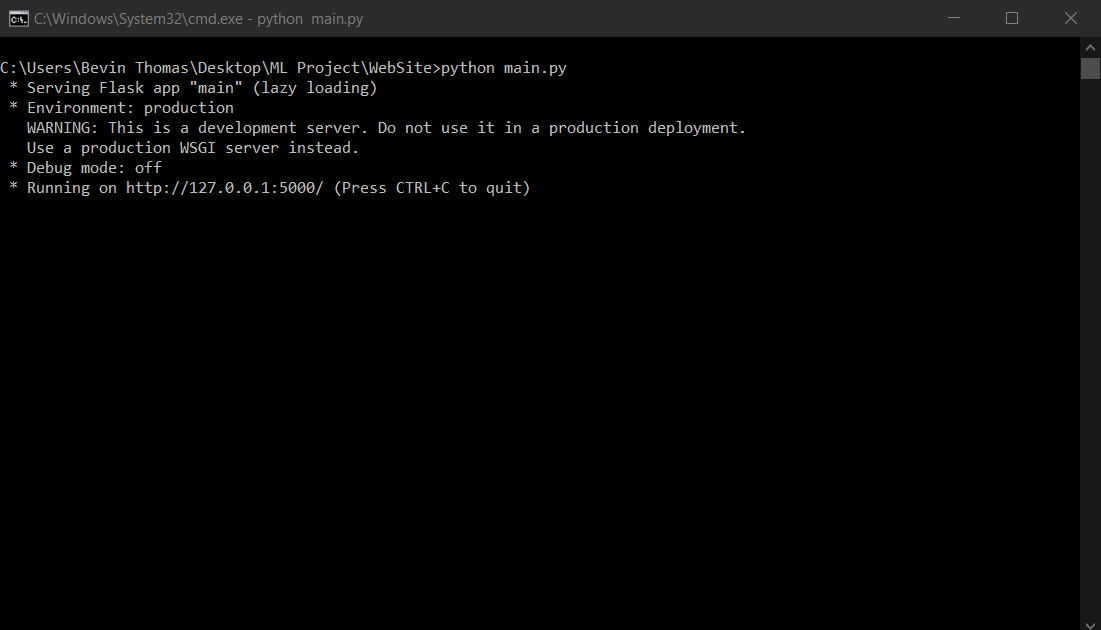


Figure 2 Output in CLI

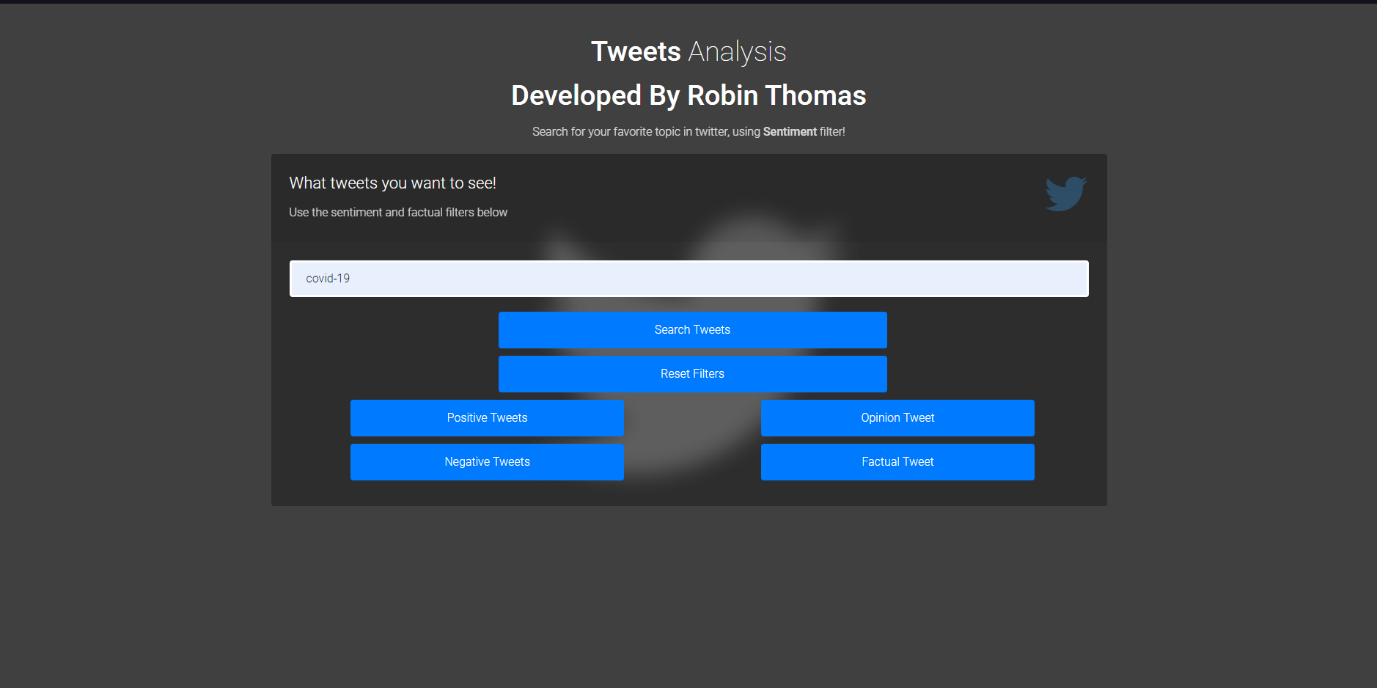


Figure 3 Website

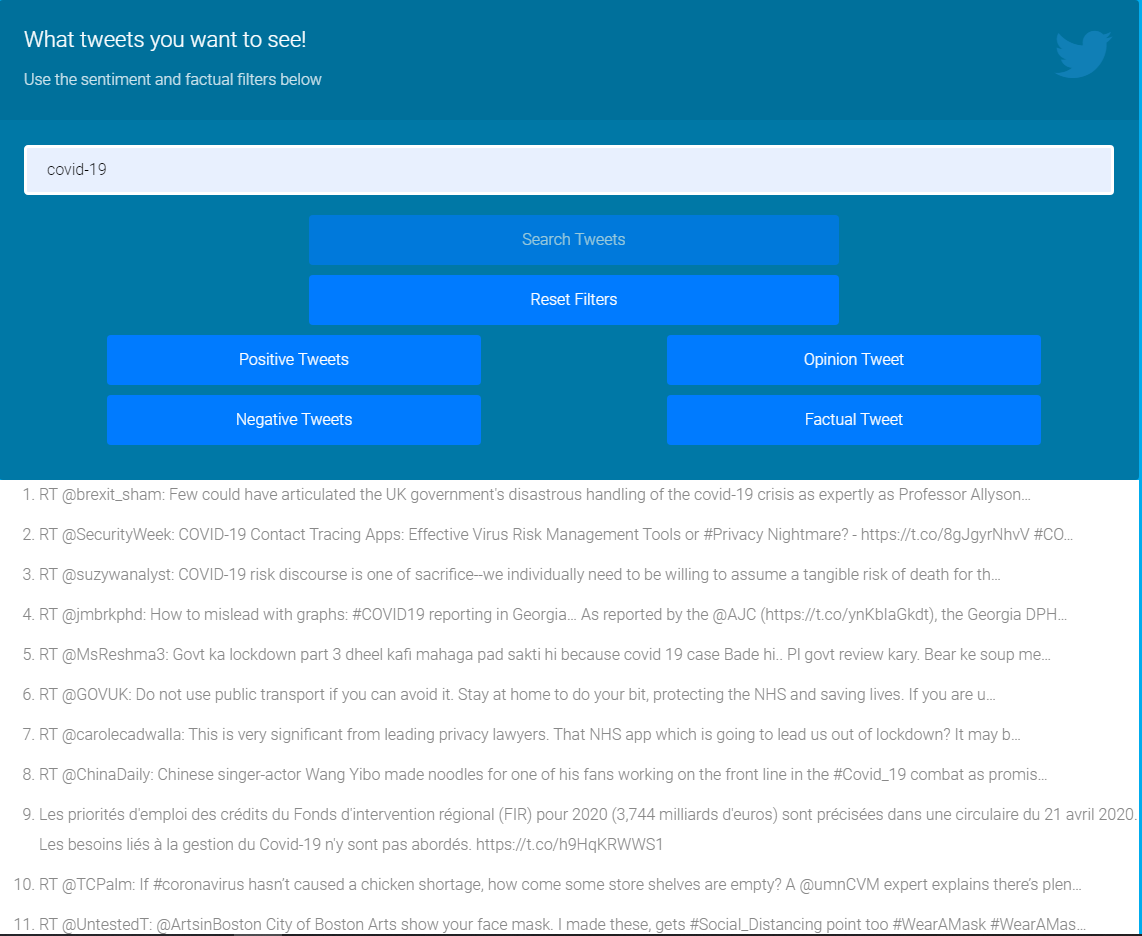
****

Figure 4 Search Results

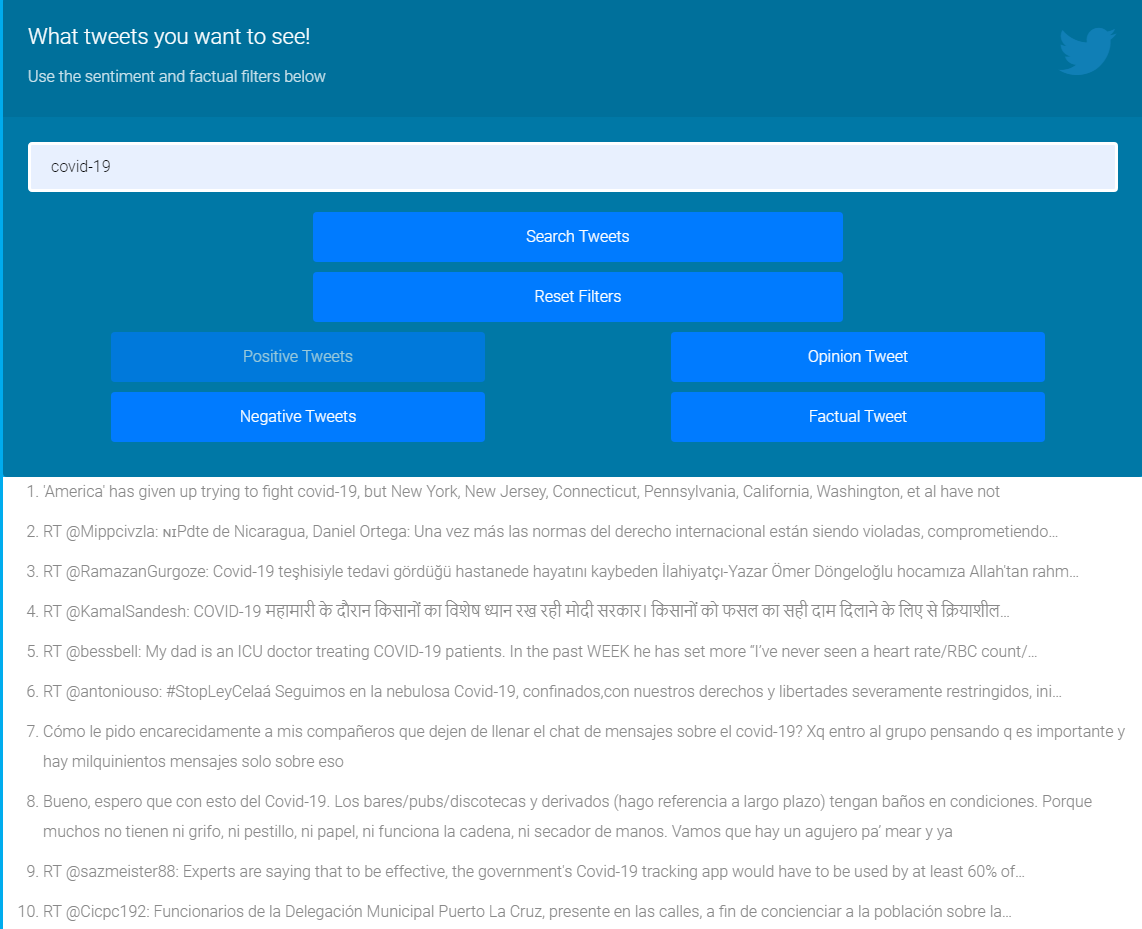


Figure 5 Positive tweets for search Covid-19

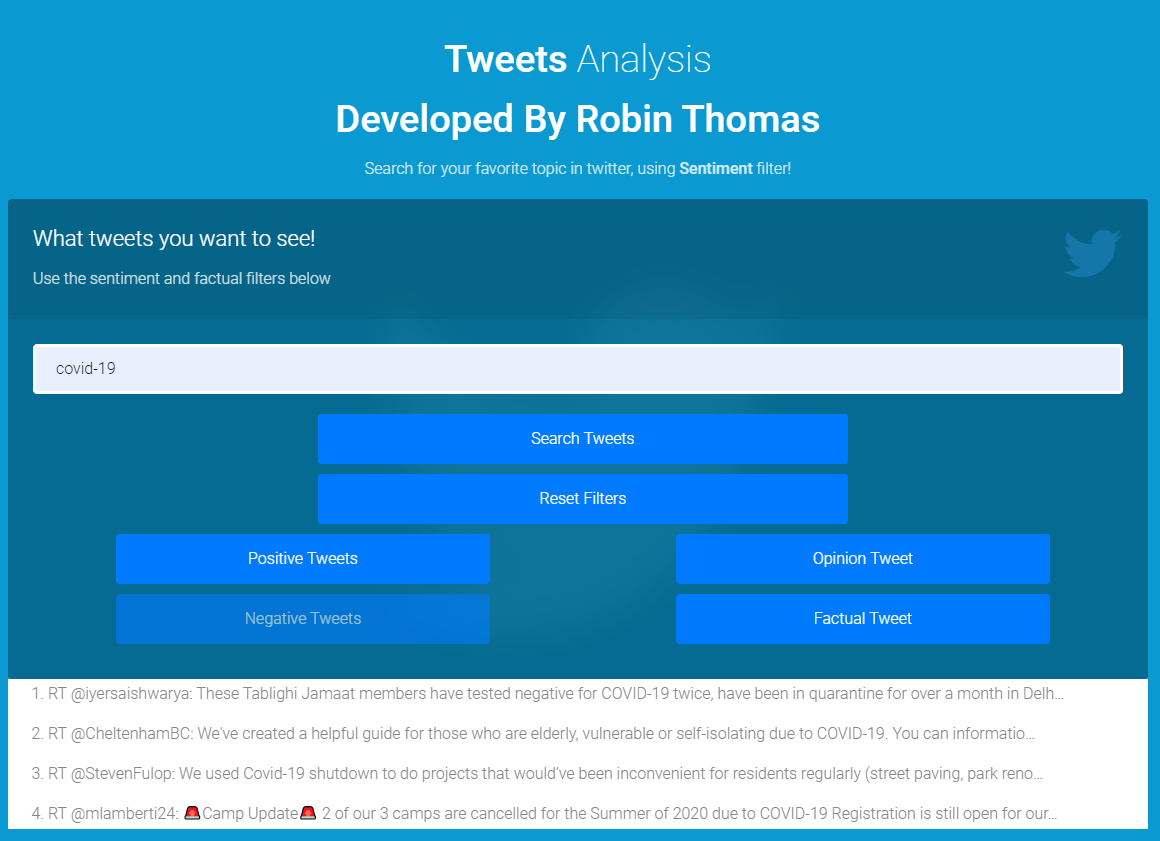


Figure 6 Negative tweets for search Covid-19

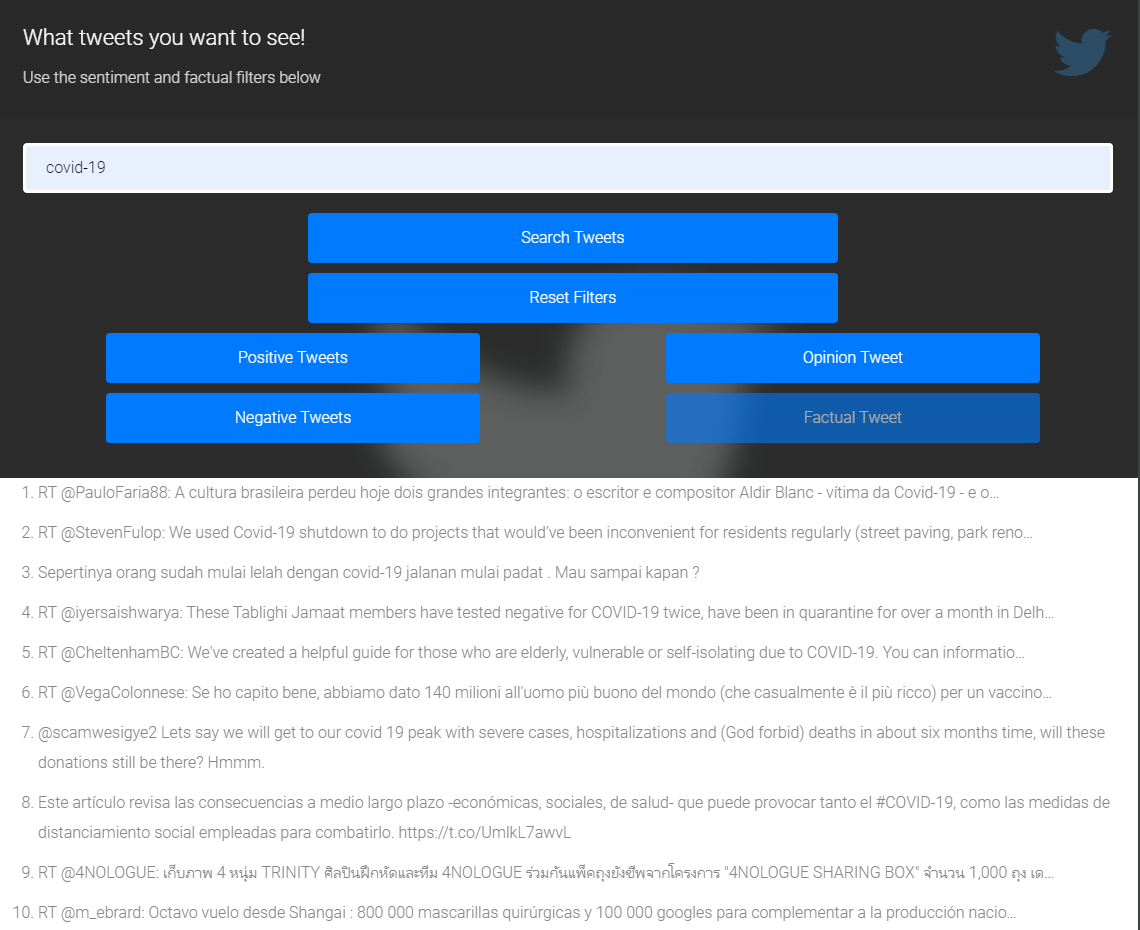
****

Figure 7 Factual Tweets for search Covid-19

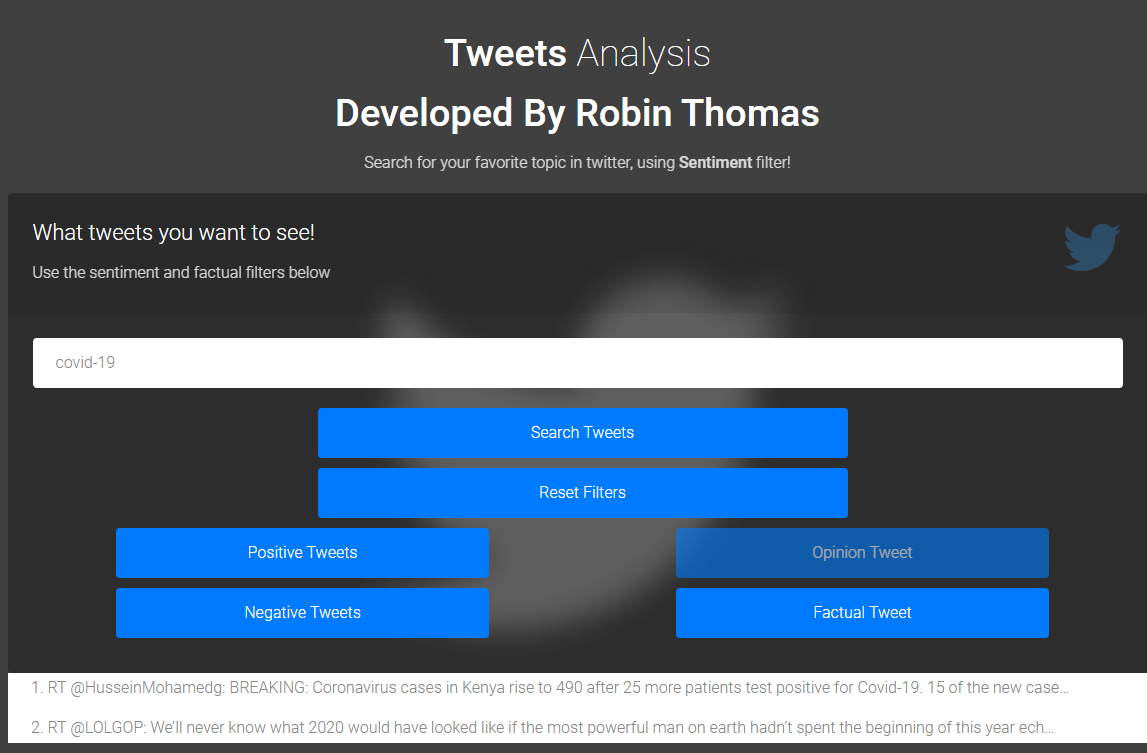
****

Figure 8 Opinion Tweets for Search Covid-19

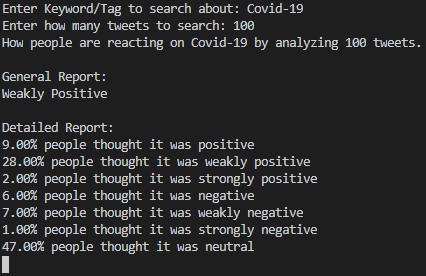


Figure 9 CLI Output 2

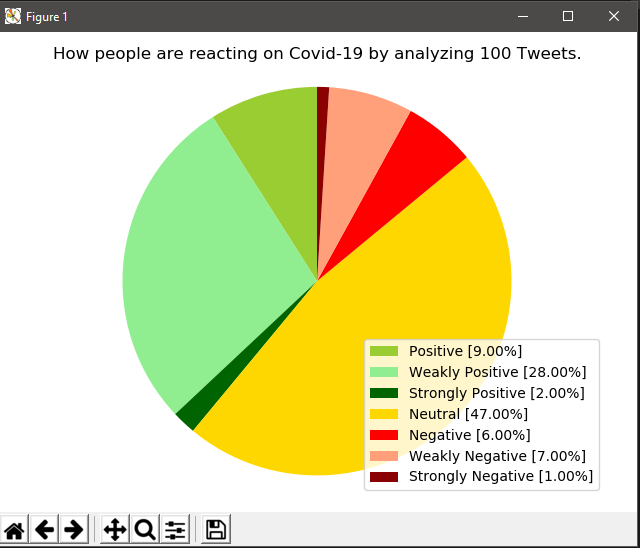


Figure 10 Pie Chart Depicting the percentage of Tweets

**CONCLUSION AND FUTURE WORKS**

As discussed earlier, Twitter sentiment analysis has a lot of applications. From our literature survey, we noticed that the accuracy of the algorithms was slightly lesser. Our approach has increased the accuracy compared to other algorithms. The purpose of our project was to accurately classify tweets into categories and predict parameters based on the results, which we have been able to achieve quite easily. As stated earlier, we have also developed a website that allows users to see the results for themselves and check the accuracy of our algorithm.

Our future works may include improving the accuracy even further than before and to try out new algorithms that may decrease the execution time and yield more results. This will in turn, make it easier to predict the required parameters efficiently.

**REFERENCES**

1. P. V, S. D and D. S. Anupama Kumar, "Real Time Sentiment Analysis Of Twitter Posts," 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), Bengaluru, India, 2018, pp. 29-34.doi: 10.1109/CSITSS.2018.8768774
2. M. Trupthi, S. Pabboju and G. Narasimha, "Sentiment Analysis on Twitter Using Streaming API," 2017 IEEE 7th International Advance Computing Conference (IACC), Hyderabad, 2017, pp. 915-919. doi: 10.1109/IACC.2017.0186
3. S. A. El Rahman, F. A. AlOtaibi and W. A. AlShehri, "Sentiment Analysis of Twitter Data," 2019 International Conference on Computer and Information Sciences (ICCIS), Sakaka, Saudi Arabia, 2019, pp. 1-4. doi: 10.1109/ICCISci.2019.8716464
4. Pak P. Paroubek "Twitter as a corpus for sentiment analysis and opinion mining" LREc vol. 10 pp. 1320-1326 May. 2010.
5. E. Boiy P. Hens K. Deschacht M. F. Moens "Automatic Sentiment Analysis in On-line Text" in ELPUB pp. 349-360 Jun. 2007.
6. Pang L. Lee "Thumbs up?: sentiment classification using machine learning techniques" Proc. ACL-02 conference on Empirical methods in natural language processing Association for Computational Lingustics pp. 79-86 Jul. 2002.
7. B. Pang L. Lee "Opinion mining and sentiment analysis" Foundations and Trends in Information Retrieval vol. 2 pp. 1-135 2008.
8. G. Vinodhini R. M. Chandrasekaran "Sentiment analysis and opinion mining: a survey" International Journal vol. 2 pp. 282-292 2012.
9. Pablo Gamallo Marcos Garcia "A Naive-Bayes Strategy for Sentiment Analysis on English Tweets" Proceedings of the 8th international Workshop on Semantic Evaluation (SemEval 2014) pp. 171-175 Aug 23-24 2014.
10. S. A. El Rahman, F. A. AlOtaibi and W. A. AlShehri, "Sentiment Analysis of Twitter Data," 2019 International Conference on Computer and Information Sciences (ICCIS), Sakaka, Saudi Arabia, 2019, pp. 1-4. doi: 10.1109/ICCISci.2019.8716464
11. V. Kharde S. Sonawane "Sentiment Analysis of Twitter Data: A Survey of Techniques" International Journal of Computer Applications vol. 139 pp. 11 2016.
12. H. Parveen and S. Pandey, "Sentiment analysis on Twitter Data-set using Naive Bayes algorithm," 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Bangalore, 2016, pp. 416-419. doi: 10.1109/ICATCCT.2016.7912034
13. Ö. Çoban and G. T. Özyer, "Word2vec and Clustering based Twitter Sentiment Analysis," 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, 2018, pp. 1-5. doi: 10.1109/IDAP.2018.8620757
14. J. Bollen H. Mao X. Zeng "Twitter mood predicts the stock market" Journal of Computational Science vol. 2 no. 1 pp. 1-8 2011.
15. D. Ayata M. Saraçlar A. Özgür "Turkish tweet sentiment analysis with word embedding and machine learning" Signal Processing and Communications Applications Conference pp. 1-4 2017.
16. E. Stamatatos N. Fakotakis G. Kokkinakis "Automatic text categorization in terms of genre and author" Computational linguistics vol. 26 no. 4 pp. 471-495 2000.
17. L. Jiang D. Wang Z. Cai X. Yan Survey of Improving Naive Bayes for Classification Springer-Verlag Berlin Heidelberg vol. 4632 pp. 134-145 2007.
18. Y. Ji, S. Yu and Y. Zhang, "A novel Naive Bayes model: Packaged Hidden Naive Bayes," 2011 6th IEEE Joint International Information Technology and Artificial Intelligence Conference, Chongqing, 2011, pp. 484-487. doi: 10.1109/ITAIC.2011.6030379
19. S. Al Shammari, "Real-time Twitter Sentiment Analysis using 3-way classifier," 2018 21st Saudi Computer Society National Computer Conference (NCC), Riyadh, 2018, pp. 1-3. doi: 10.1109/NCG.2018.8593205
20. P. Garg, H. Garg and V. Ranga, "Sentiment analysis of the Uri terror attack using Twitter," 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, 2017, pp. 17-20.  
    doi: 10.1109/CCAA.2017.8229812