

SENTIMENTAL ANALYSIS ON SOCIAL MEDIA

By

Karan Rohit (ID No. 16CEUSS004)

Bhumik Parmar (ID No. 16CEUSS007)

A project submitted

In

Partial fulfillment of the requirements

For the degree of

BACHELOR OF TECHNOLOGY

In

Computer Engineering

Internal Guide

Prof. Bhavika Gambhava

Assistant Professor

Dept. of Comp. Engg.

External Guide

Mr. Vishal Patel

Project Manager

BISAG



Faculty of Technology

Department of Computer Engineering

Dharmsinh Desai University

April 2020

CERTIFICATE

This is to certify that the project work titled Sentimental Analysis on
Social Media is the bonafide work of

Karan Rohit – 16CEUSS004

&

Bhumik Parmar - 16CEUSS007

carried out in the partial fulfillment of the degree of Bachelor of
Technology in Computer Engineering at Dharmsinh Desai University in
the academic session

December 2019 to April 2020.

Bhavika Gambhava

Assistant Professor

Dept. of Computer Engg.

Dr. C. K. Bhensdadia

Head

Dept. of Computer Engg.



Faculty of Technology

Department of Computer Engineering

Dharm Singh Desai University

April 2020

Join Feature In GIS

B.Tech-CE Semester- VIII

Prepared at



Bhaskaracharya Institute for Space Applications & Geo-informatics

Science & Technology Department, Govt. of Gujarat.

Gandhinagar

Prepared By

Karan G. Rohit

ID No. 16CEUSS004

Guided By:

Prof. Bhavika Gambhava

Department of Computer Engineering

DDU, Nadiad.

Bhumik Parmar

ID No. 16CEUSS007

External Guide:

Vishal Patel

Project Scientist

BISAG, Gandhinagar.

SUBMITTED TO

Dharmsinh Desai University



Faculty of Technology – Nadiad

Department of Science & Technology

Government of Gujarat

Phone: 079 - 23213081 Fax: 079 - 23213091

E-mail: info@bisag.gujarat.gov.in, website: www.bisag.gujarat.gov.in

CERTIFICATE

*This is to certify that the project report compiled by **Mr. Karan Rohit and Mr. Bhumik Parmar** students of 8th Semester **B.Tech-CE** from Faculty of Technology, Dharmsinh Desai University, Nadiad have completed their final semester project satisfactorily. To the best of our knowledge this is an original and bonafide work done by them. They have worked on web-based application for **“Sentimental Analysis on Social Media”**, starting from December 9th, 2019 to April 3rd, 2020.*

During their tenure at this Institute, they were found to be sincere and meticulous in their work. We appreciate their enthusiasm & dedication towards the work assigned to them.

We wish them every success.

Vishal Patel
Project Scientist,
BISAG, Gandhinagar

T. P. Singh
Director,
BISAG, Gandhinagar

Acknowledgements

We are deeply thankful to our advisor and Mentor, *Assistant Prof. Bhavika Gambhava* and Project Scientist **Mr. Vishal Patel** for helping us throughout the course in accomplishing our final project. Their guidance, support and motivation enabled us in achieving the objectives of the project.

Table Of Contents

Study Project		Page No.
Chapter 1	Introduction	9
Chapter 2	Software Requirement Specification	12
	2.1 Internal Interface requirement	
	2.2 External Interface requirement	
	2.3 Non-Functional Requirement	
	2.4 Other Requirements	
Chapter 3	UML Diagrams	17
	3.1 Sequence Diagram	
	3.2 State Diagram	
	3.3 Data Flow Diagram	
	3.4 Flow Chart	
Chapter 4	About the System	20
	4.1 Problem Definition	
	4.2 Technology Used	
	4.3 Working Procedure	
	4.4 Models Developed	
Chapter 5	Study and Survey of Related Systems	40
Chapter 6	Conclusion and Future Scope	43

Table of Tables

Contents	Page No.
Table 1: Keyword Sentiment Value 1	30
Table 2: Keyword Sentiment Value 2	30

Table of Figures

Contents	Page No.
Figure 1: Sequence Diagram	17
Figure 2: State Diagram	17
Figure 3: Data Flow Diagram	18
Figure 4: Flow Chart	19
Figure 5: Pie Chart 1	25
Figure 6: Command Line Output 1	25
Figure 7: Command Line Output 2	26
Figure 8: Live Tweets Output	27
Figure 9: Pie Chart 2	27
Figure 10: Command Line Output 3	28
Figure 11: UserID Input Page	32
Figure 12: UI Output	32
Figure 13: Tweets Output	33
Figure 14: Keyword Input UI	34
Figure 15: Keyword UI Output	34
Figure 16: Topic Input UI	35
Figure 17: Topic Output 1	36
Figure 18: Topic Output 2	36
Figure 19: Input Text Model UI	37
Figure 20: Different Options	38
Figure 21: Input Text UI Analysis	38
Figure 22: Input Keyword Analysis	39

Chapter 1

Introduction

A sentiment is a view or opinion that is held or expressed. Sentiment means a view or opinion, but it can also mean an emotion. Maybe you prefer tragic movies because you enjoy the sentiment of sadness. This meaning of sentiment is taken to an extreme in yet another version of the word, meaning something like "overdone, exaggerated feelings, especially of sadness or nostalgia." For example, let's take this sentence: "I don't find the app useful; it's really slow and constantly crashing." A sentiment analysis model automatically tags this as *Negative*.

Sentiment analysis (also known as **opinion mining** or **emotion AI**) refers to the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine.

Sentiment analysis is contextual mining of text which identifies and extracts subjective information in source material, and helping a business to understand the social sentiment of their brand, product or service while monitoring online conversations. However, analysis of social media streams is usually restricted to just basic sentiment analysis and count

based metrics. This is akin to just scratching the surface and missing out on those high value insights that are waiting to be discovered.

Let's say a company has just launched a new product feature and you notice a sharp increase in mentions on Twitter. However, receiving tons of mentions does not *necessarily* mean a good thing. Are customers tweeting more because they are expressing good things about this new product feature? Or, are customers actually complaining about the feature having lots of bugs? Performing twitter sentiment analysis can be excellent way to understand the tone of those mentions and obtain real-time insights on how users are perceiving your new product.

Because of sentiment analysis, companies can understand the reputation of their brand. By analyzing social media posts, customer feedback, or NPS(Net promoter Score) responses(among other sources of unstructured business data), they can be aware of how their customers *feel* about their product. They can also track specific topics and get relevant insights on how people are talking about those topics. Sentiment analysis is particularly useful for social media monitoring because it goes beyond metrics that focus on the number of likes or re-tweets , and provides a qualitative point of view.

With the recent advances in deep learning, the ability of algorithms to analyze text has improved considerably. Creative use of advanced artificial intelligence techniques can be an effective tool for doing in-depth research. Sentiment Analysis is the most common text classification tool that analyses an incoming message and tells whether the underlying sentiment is positive, negative our neutral. Performing sentiment analysis

on data from Twitter using machine learning can help companies understand how people are talking about their brand.

Twitter allows businesses to reach a broad audience and connect with customers without intermediaries. On the downside, it's harder for brands to quickly detect negative content, and if it goes viral you might end up with an unexpected PR crisis on your hands. This is one of the reasons why social listening – monitoring conversation and feedback in social media – has become a crucial process in social media marketing. Monitoring Twitter allows companies to understand their audience, keep on top of what's being said about their brand and their competitors, and discover new trends in the industry.

Nearly 80% of the world's digital data is unstructured , and data obtained from social media sources is no exception to that. Since the information is not organized in any predefined way, it's difficult to sort and analyze. Twitter sentiment analysis systems allow you to sort large sets of tweets and detect the polarity of each statement automatically.

If we want to perform Twitter sentiment analysis, the first step is to gather the data. This data is the data that we will use for training the model and running the actual sentiment analysis on Twitter data. There are two main type of tweets we can extract from Twitter: Current Tweets and Historical Tweets. Current Tweet is useful to track keywords or hashtags in real-time. Historical Tweets are used to search past tweets during a predefined time frame.

Note that no any algorithm or technique provides 100% accuracy or prediction on Sentiment analysis.

Chapter 2

Software Requirement Specification

SRS (Software Requirement Specification):

Internal Interface requirement:

Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.

Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.

Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.

Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here. The recent explosion in data pertaining to users on social media has created a great interest in performing sentiment analysis on this data using Big Data and Machine Learning principles to understand people's interests. This project intends to perform the same tasks. The difference between this project and other sentiment analysis tools is that, it will perform real time analysis of tweets based on hashtags and not on a stored archive.

Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.

The Product functions are:

- Collect tweets in a real time fashion i.e. from the twitter live stream based on specified hashtags.
- Remove redundant information from these collected tweets.
- Perform Sentiment Analysis on the tweets stored in the database to classify their nature viz. positive, negative and so on.
- Use a machine learning algorithm which will predict the 'mood' of the people with respect to that topic.

Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.

Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.

External Interface Requirement:

We classify External Interface in 4 types, those are:

User Interface:

Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.

Hardware interface:

Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.

Software Interface:

Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.

Communication Interface:

Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.

Non Functional Requirement:**Performance Requirements:**

If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.

Safety Requirements:

Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.

Security Requirements:

Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.

Software Quality Attributes:

Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are:

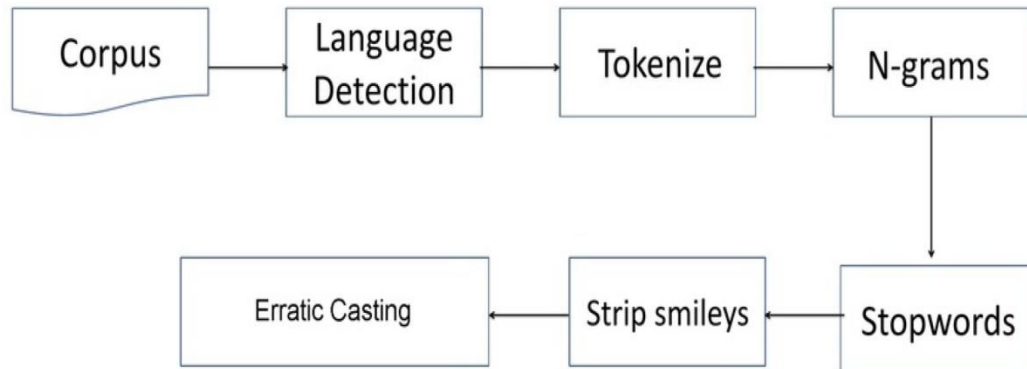
Adaptability, Availability, Correctness, Flexibility, Interoperability, Maintainability, Portability, Reliability, Reusability, Robustness, Testability and Usability. Write these to be Specific, Quantitative and Verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.

Other Requirements:

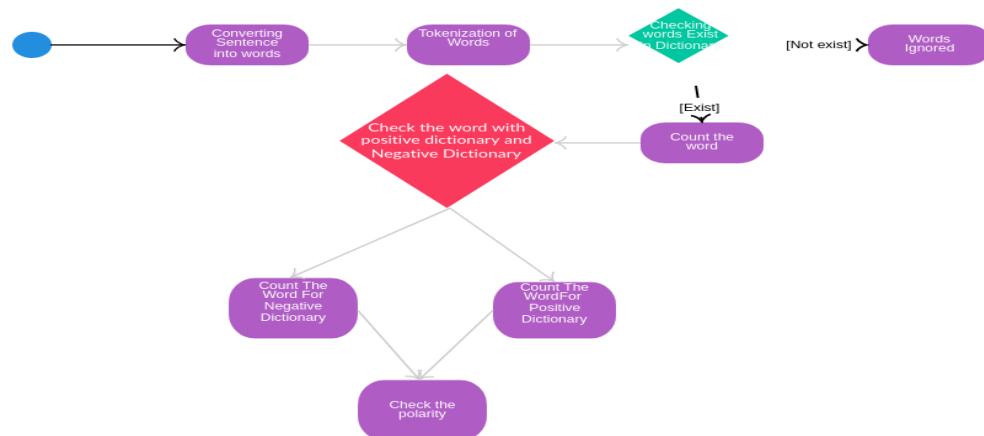
- Linux Operating System/Windows.
- Python Platform (Jupyter, Python3).
- NLTK package (TextBlob, Tweepy).
- Modern Web Browser (Chrome).
- Twitter API.

Chapter 3

UML Diagrams

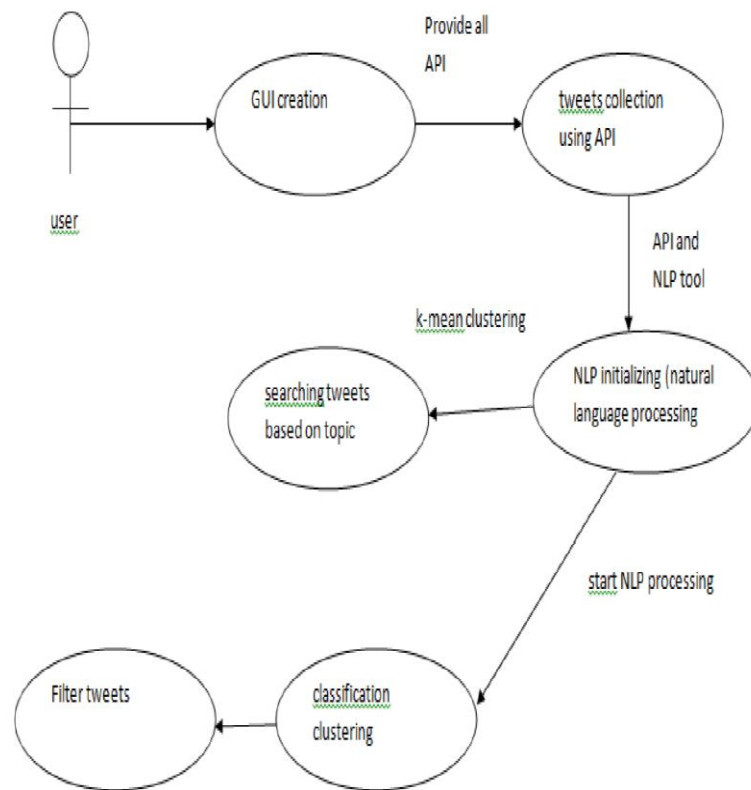
1. Sequence Diagram:

[Figure: 1] Sequence Diagram

2. State Diagram:

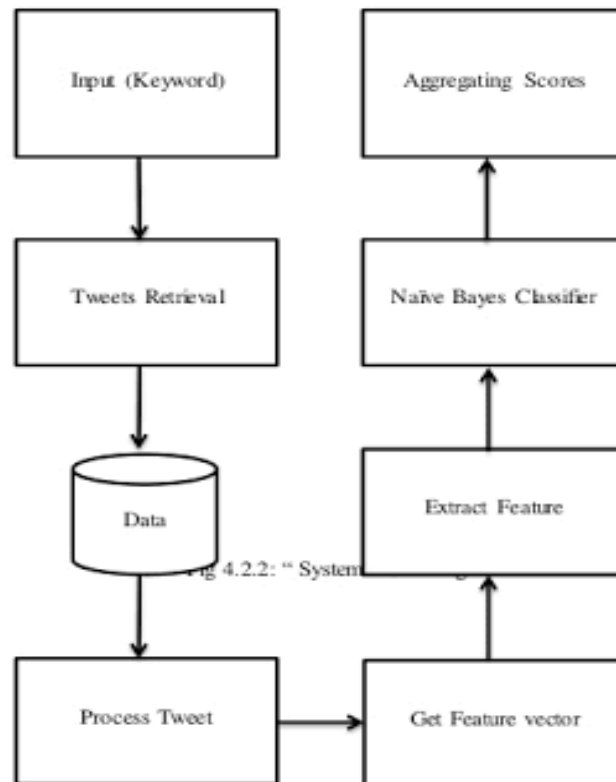
[Figure: 2] State Diagram

3. DFD(Data Flow Diagram):



[Figure: 3] Data Flow Diagram

4. Flow Chart:



[Figure: 4] Flow Chart

Chapter 4

About the System

Problem Definition:

Sentiment analysis of Twitter in the domain of micro-blogging is a relatively new research topic so there is still a lot of room for further research in this area. Decent amount of related prior work has been done on sentiment analysis of user reviews, documents, web blogs/articles and general phrase level sentiment analysis. These differ from twitter mainly because of the limit of 280 characters per tweet which forces the user to express opinion compressed in very short text. The best results reached in sentiment classification use supervised learning techniques such as Naive Bayes and Support Vector Machines, but the manual labeling required for the supervised approach is very expensive. Some work has been done on unsupervised and semi-supervised approaches, and there is a lot of room of improvement. Various researchers testing new features and classification techniques often just compare their results to base-line performance. There is a need of proper and formal comparisons between these results arrived through different features and classification techniques in order to select the best features and most efficient classification techniques for particular applications.

So in our system we are able to do sentimental analysis on tweets by some keywords like: good, bad, worst, happy etc.

For better understanding and practice we have created two kinds of models:-

1. Command line models.
2. Web based UI models.

About Technology we Used:

1. Flask Framework:

Flask is a micro web framework written in Python. It is classified as a micro-framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

2. NLTK(Natural Language Processing Toolkit):

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing for English written in the Python programming language.

3. Tweepy:

Twitter is a popular social network where users share messages called tweets. Twitter allows us to mine the data of any user using Twitter API or Tweepy. The data will be tweets extracted from the user. The first thing to do is to get the consumer key, consumer secret, access key and access secret from twitter developer available after proper reasoning for each user. These keys will help the API for authentication.

4. Matplotlib:

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.

5. TextBlob:

TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation and more.

WORKING PROCEDURE OF SENTIMENTAL ANALYSIS:

Input (Key-Word):

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

- Specific keyword to track/search for in the tweets
- Specific Twitter user according to their name
- Tweets originating from specific location(s) (only for geo-tagged tweets).

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

Since we wanted to increase the generality of our data, we acquired it in portions at different points of time instead of acquiring all of it at one go. If we used the latter approach then the generality of the tweets might have been compromised since a significant portion of the tweets would be referring to some certain trending topic and would thus have more or less of the same general mood or sentiment. This phenomenon has been observed when we were going through our sample of acquired tweets. For example the sample acquired near Christmas and New Year's had a significant portion of tweets referring to these joyous events and were thus of a generally positive sentiment. Sampling our data in portions at different points in time would thus try to minimize this problem. Thus forth, we acquired data at four different points which would be 17th of December 2015, 29th of December 2015, 19th of January 2016 and 8th of February 2016.

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

- Whether a tweet has been favorite
- 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

Tweets Retrieval:

Since human labeling is an expensive process we further filter out the tweets to be labeled so that we have the greatest amount of variation in tweets without the loss of generality. The filtering criteria applied are stated below:

- 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)

After this filtering roughly 30% of tweets remain for human labeling on average per sample, which made a total of 10,173 tweets to be labeled.

Data Preprocessing:

Data preprocessing consists of three steps:

- 1) Tokenization,
- 2) Normalization, and
- 3) Part-of-speech (POS) tagging.

Tokenization:

It is the process of breaking a stream of texts up into words, symbols and other meaningful elements called “tokens”. Tokens can be separated by whitespace characters and/or punctuation characters. It is done so that we can look at tokens as individual components that make up a tweet. Emoticons and abbreviations (e.g., *OMG*, *IKR* and *BRB*) are identified as part of the tokenization process and treated as individual tokens.

Part-of-speech:

POS-Tagging is the process of assigning a tag to each word in the sentence as to which grammatical part of speech that word belongs to, i.e. noun, verb, adjective, adverb, coordinating conjunction etc. For each tweet, we have features for counts of the number of verbs, adverbs, adjectives, nouns, and any other parts of speech.

Classified Tweets:

We labeled the tweets in three classes according to sentiments expressed/observed in the tweets: positive, negative and neutral. We gave the following guidelines to our labellers to help them in the labeling process:

Positive:

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

Negative:

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

Neutral:

If the user who wrote a tweet, expresses no personal sentiment/opinion in the tweet and merely transmits information. Advertisements of different products would be labeled under this category.

So let us discuss first command line models:-

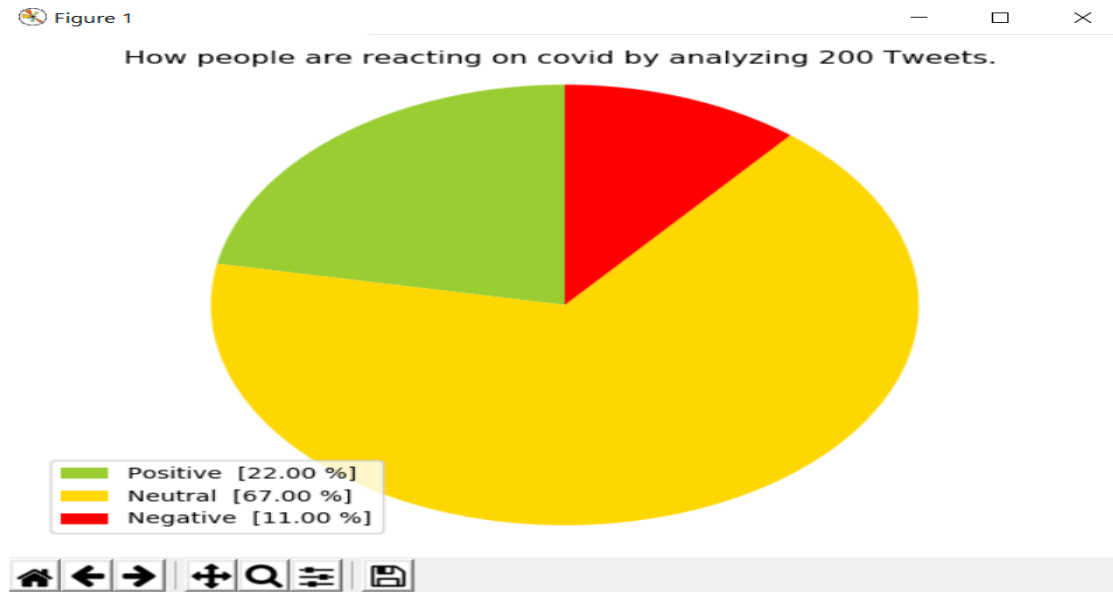
In command line models first we have created,

1. Keyword and count wise:-

In this we have made a python model in which user can provide a particular keyword along with limited no of counts to limit the no of fetching tweets.

It is showing a pie chart in three colors:-

1. Green :- Positive feedback
2. Yellow :- Neutral feedback
3. Red :- Negative feedback



[Figure: 5] Pie Chart 1

Also it prints the overall sentimental analysis on tweets like as shown below:-

```
Command Prompt - python "keyword & count wise.py"

(myvenv) C:\Users\karan\OneDrive\Desktop\BISAG_Proj\myvenv\Final Year Project\CMD line>python "keyword & count wise.py"
Enter keyword\hashtag to search about: covid
Enter How many tweets to analyze: 200
How people are reacting on covid by analyzing 200 Tweets.
Positive
```

[Figure: 6] Command line Output 1

Now we will discuss about our second Command line model:-

2. LIVE tweet analysis on stored keywords:-

It tries to print each fetched tweets on stored keyword along with their sentimental polarity and index. Like as shown below,

```

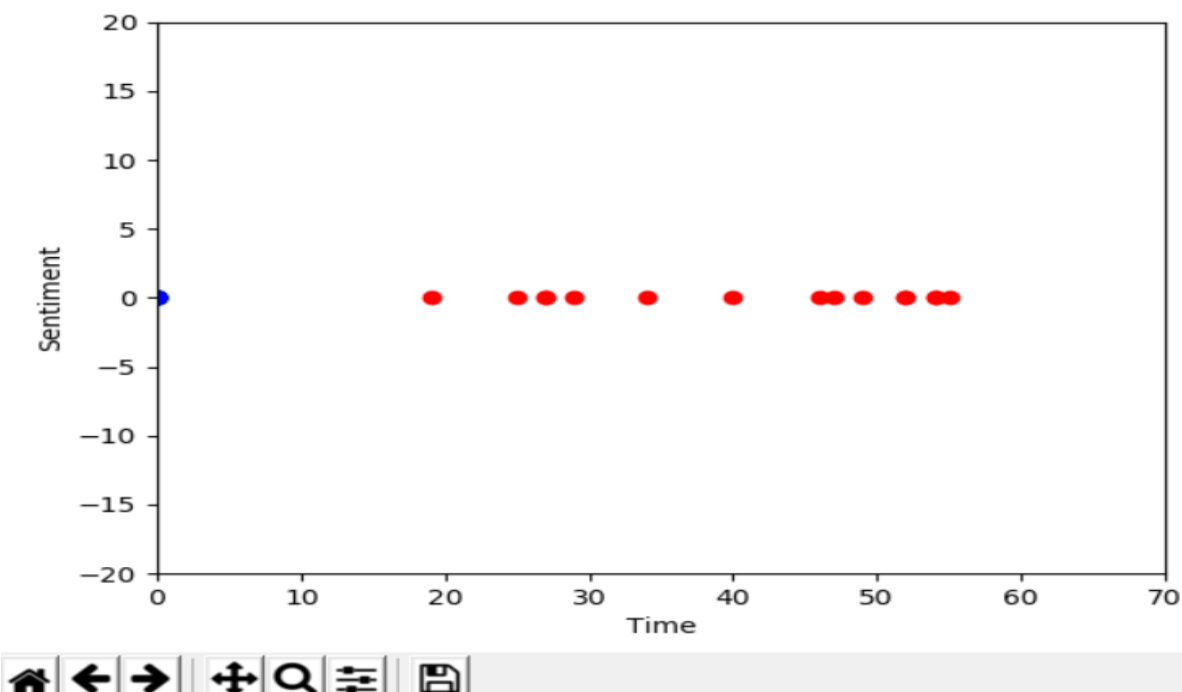
Command Prompt - python "LIVE tweet analysis on stored keyword.py"
RTIndurChhuganiFakeNarendraModijustnowbecamemyfollowerChiefMinisterinShriNarendraModifollowedmethensameNare
0.0
54
0.0 0 0.0
14
RTomthanviOneofthemostremarkablecasesisIndiawherePrimeMinisterNarendraModihasimposedathreeweeklockdownonhisbi
0.0
54
0.0 0 0.0
15
RTvinitgoenkaIndiahasseenasharperosionincivillibertiesunderthegovernmentoftheHindunationalistPrimeMinisterNarendraM
0.0
55
0.0 0 0.0
16
ReportPMnarendramodipaystributetomartyrsofJallianwalaBaghmassacrehttpstcoLjDldDjdyTimesNowviaUnfollowers
0.0
56
0.0 0 0.0
17
RTbainjalCOVIDAndnowitiscoronajihadinNarendraModishatefillednewIndiahttpstcoTzjSPqfKDmySWATAnalysisexclu
0.0
58
0.0 0 0.0
18
RTbainjalCOVIDAndnowitiscoronajihadinNarendraModishatefillednewIndiahttpstcoTzjSPqfKDmySWATAnalysisexclu
0.0
59
0.0 0 0.0

```

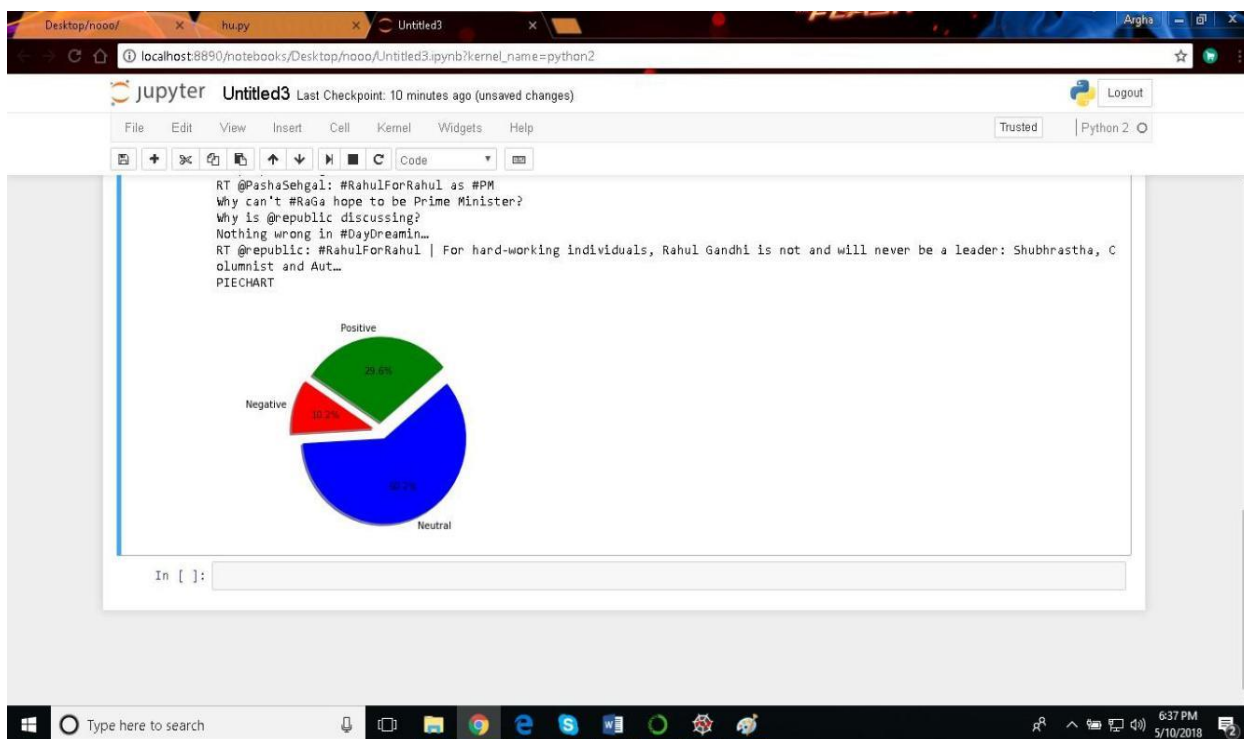
[Figure: 7] Command Line Output 2

And it prints live fetched tweet analysis in graphical format based on index and polarity that whether the tweet is positive, negative or neutral. Like as shown below,

Figure 1



[Figure: 8] Live Tweets Output



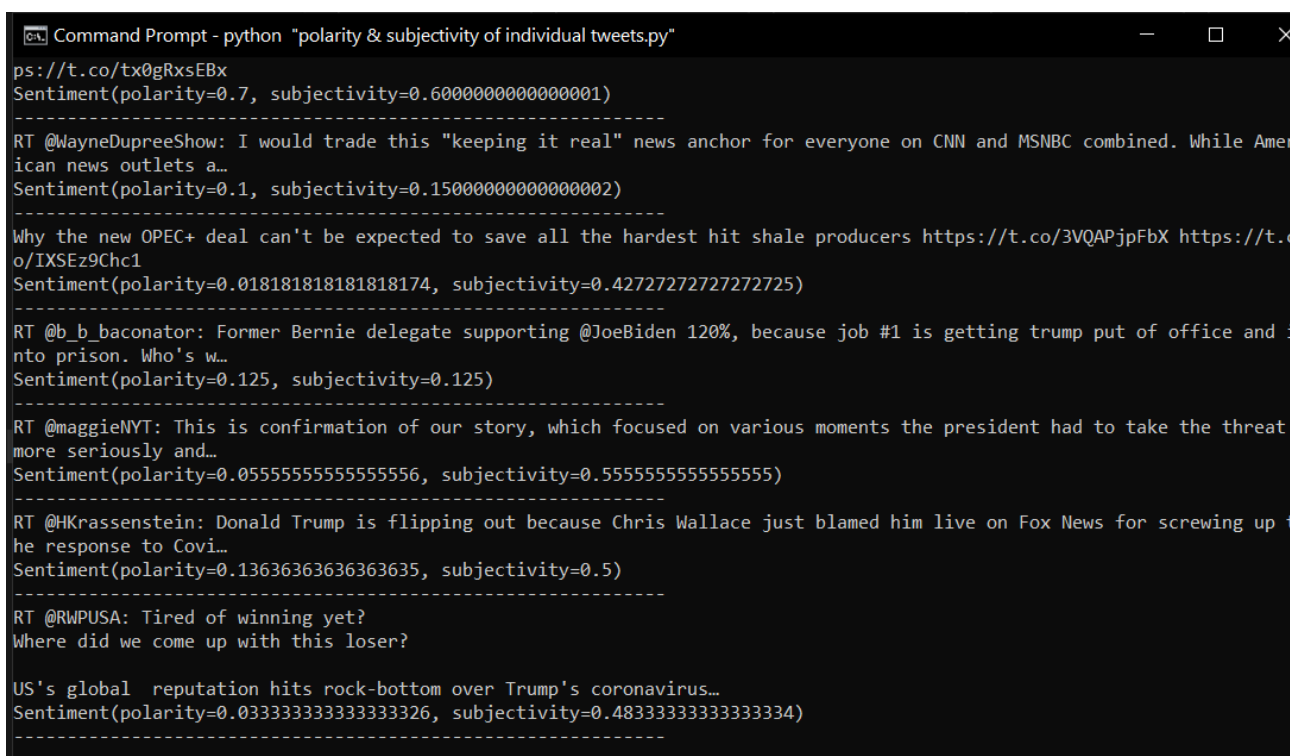
[Figure: 9] Pie Chart 2

Now we will discuss on our 3rd command line model:-

3. Polarity and subjectivity of individual tweets:-

Here it fetched all real time tweets on stored keyword based on recent timestamp.

It prints time stamp, sentimental polarity of individual tweets along with its subjectivity to user and content of tweet. Like as shown below,



```

Command Prompt - python "polarity & subjectivity of individual tweets.py"
ps://t.co/tx0gRxSEBx
Sentiment(polarity=0.7, subjectivity=0.6000000000000001)
-----
RT @WayneDupreeShow: I would trade this "keeping it real" news anchor for everyone on CNN and MSNBC combined. While Amer
ican news outlets a...
Sentiment(polarity=0.1, subjectivity=0.15000000000000002)
-----
Why the new OPEC+ deal can't be expected to save all the hardest hit shale producers https://t.co/3VQAPjpFbX https://t.co
o/IXSEz9Chc1
Sentiment(polarity=0.018181818181818174, subjectivity=0.42727272727272725)
-----
RT @b_b_baconator: Former Bernie delegate supporting @JoeBiden 120%, because job #1 is getting trump put of office and i
nto prison. Who's w...
Sentiment(polarity=0.125, subjectivity=0.125)
-----
RT @maggieNYT: This is confirmation of our story, which focused on various moments the president had to take the threat
more seriously and...
Sentiment(polarity=0.05555555555555556, subjectivity=0.5555555555555555)
-----
RT @HKrassenstein: Donald Trump is flipping out because Chris Wallace just blamed him live on Fox News for screwing up t
he response to Covi...
Sentiment(polarity=0.13636363636363635, subjectivity=0.5)
-----
RT @RWPUSA: Tired of winning yet?
Where did we come up with this loser?

US's global reputation hits rock-bottom over Trump's coronavirus...
Sentiment(polarity=0.03333333333333326, subjectivity=0.48333333333333334)
-----

```

[Figure: 10]Command line Output 3

Now we can move forward toward our Web based UI developed models:-

The TextBlob package for Python is a convenient way to do a lot of Natural Language Processing (NLP) tasks. For example:-

From TextBlob import TextBlob

TextBlob ("not a very great calculation").sentiment

This tells us that the English phrase “not a very great calculation” has a polarity of about -0.3, meaning it is slightly negative, and a subjectivity of about 0.6, meaning it is fairly subjective.

There are helpful comments like this one, which gives us more information about the numbers we're interested in:-

Each word in the lexicon has scores for:-

1) Polarity: negative vs. positive (-1.0 => +1.0)

2) Subjectivity: objective vs. subjective (+0.0 => +1.0)

3) Intensity: modifies next word? (x0.5 => x2.0)

The lexicon it refers to is in en-sentiment.xml, an XML document that includes the following four entries for the word “great”.

```
<word Form="great" cornetto svnset id="n_a-525317" wordnet id="a-01123879"
pos="JJ" sense="very good" polanty="1.0" subjectivity="1.0" intensity="1.0"
confidence="0.9" />
```

```
<word Form="great" wordnet id="a-011238818" pos="JJ" sense="of major
significance or importance" polanty="1.0" subjectivity="1.0" intensity="1.0"
confidence="0.9" />
```

```
<word Form="great" wordnet id="a-01123883" pos="JJ" sense="relativity large
in size or number or extent" polanty="0.4" subjectivity="0.2" intensity="1.0"
confidence="0.9" />
```

```
<word Form="great" wordnet id="a-01677433" pos="JJ" sense="remarkable or
out of the ordinary in degree or magnitude or effect" polanty="0.8"
subjectivity="0.8" intensity="1.0" confidence="0.9" />
```

In addition to the polarity, subjectivity and intensity mentioned in the comment above, there's also “confidence”, but we don't see this being used anywhere. In the case of “great” here, it's all the same part of speech (JJ, adjective), and the senses are themselves natural language and not used. To simplify for readability:

Word	Polarity	Subjectivity	Intensity
Great	1.0	1.0	1.0
Great	1.0	1.0	1.0
Great	0.4	0.2	1.0
Great	0.8	0.8	1.0

[Table: 1] Keyword Sentiment Value 1

When calculating sentiment for a single word, TextBlob uses a sophisticated technique known to

Mathematicians as “averaging”.

TextBlob (“great”).sentiment

Sentiment (polarity=0.8, subjectivity=0.75)

At this point we might feel as if we're touring a sausage factory. That feeling isn't going to go away, but remember how delicious sausage is! Even if there isn't a lot of magic here, the results can be useful—and you certainly can't beat it for convenience.

TextBlob doesn't handle negation, and that isn't anything!

TextBlob (“not great”).sentiment

Sentiment (polarity=-0.4, subjectivity=0.75)

Negation multiplies the polarity by -0.5, and doesn't affect subjectivity.

TextBlob also handles modifier words. Here's the summarized record for “very” from the lexicon:

Word	Polarity	Subjectivity	Intensity
Very	0.2	0.3	1.3

[Table: 2] Keyword Sentiment Value 2

Recognizing “very” as a modifier word, TextBlob will ignore polarity and subjectivity and just use intensity to modify the following word:

```
TextBlob("very great").sentiment
```

```
## Sentiment (polarity=1.0, subjectivity=0.9750000000000001)
```

The polarity gets maxed out at 1.0, but you can see that subjectivity is also modified by “very” to become $0.75 \cdot 1.3 = 0.975$. $0.75 \cdot 1.3 = 0.975$.

Negation combines with modifiers in an interesting way: in addition to multiplying by -0.5 for the polarity, the inverse intensity of the modifier enters for both polarity and subjectivity.

```
TextBlob("not very great").sentiment
```

```
#Sentiment (polarity=-0.3076923076923077, subjectivity=0.5769230769230769)
```

Polarity = $-0.5 \cdot 1.3 \cdot 0.8 \approx -0.31$ polarity = $-0.5 \cdot 1.3 \cdot 0.8 \approx -0.31$

Subjectivity = $1.3 \cdot 0.75 \approx 0.58$ subjectivity = $1.3 \cdot 0.75 \approx 0.58$

TextBlob will ignore one-letter words in its sentiment phrases, which means things like this will work just the same way:

```
TextBlob("not a very great").sentiment
```

```
##Sentiment (polarity=-0.3076923076923077, subjectivity=0.5769230769230769)
```

And TextBlob will ignore words it doesn't know anything about:

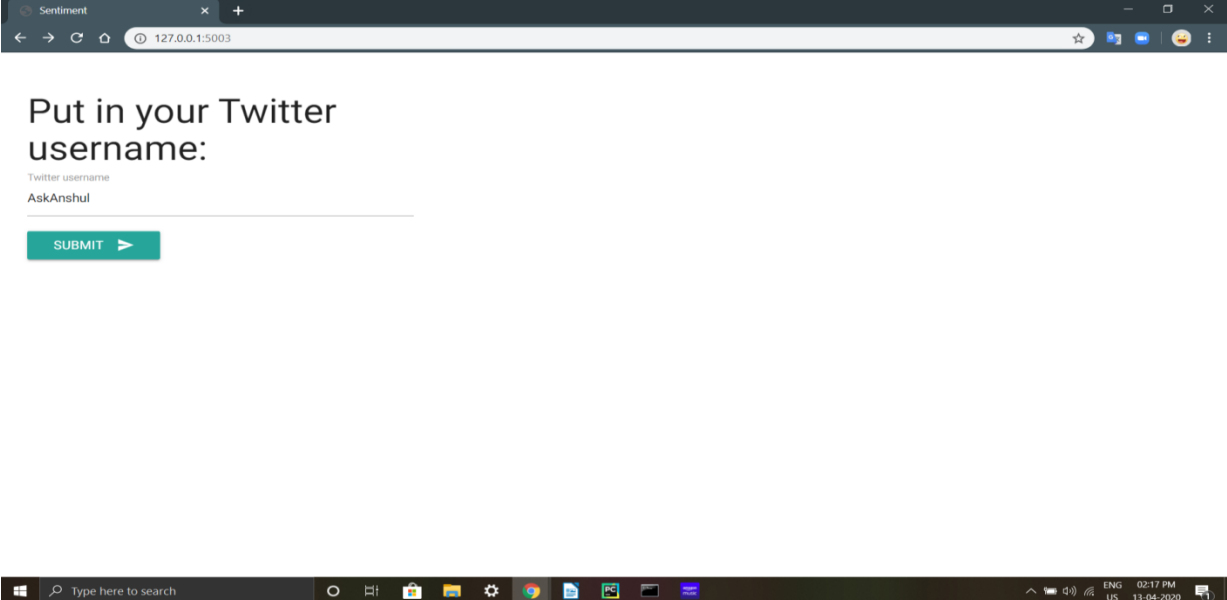
```
TextBlob("not a very great calculation").sentiment
```

```
##Sentiment (polarity=-0.3076923076923077,
subjectivity=0.5769230769230769)
```

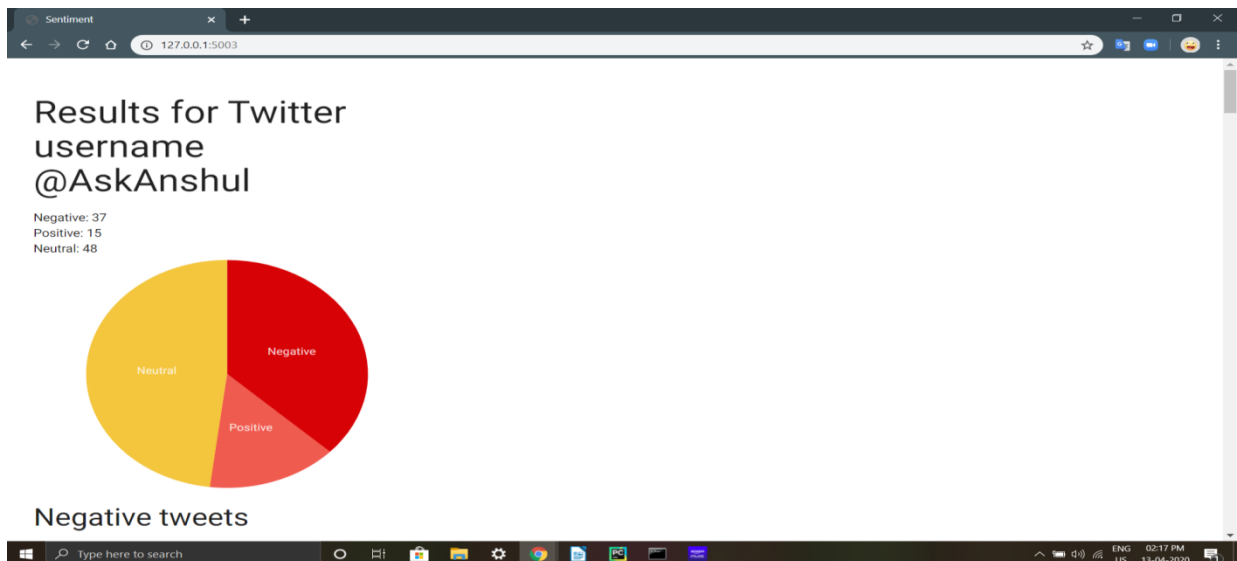
TextBlob goes along finding words and phrases it can assign polarity and subjectivity to, and it averages them all together for longer text.

We have built using flask python framework and also using Machine Learning Libraries like Tweepy. The following are the web based models.

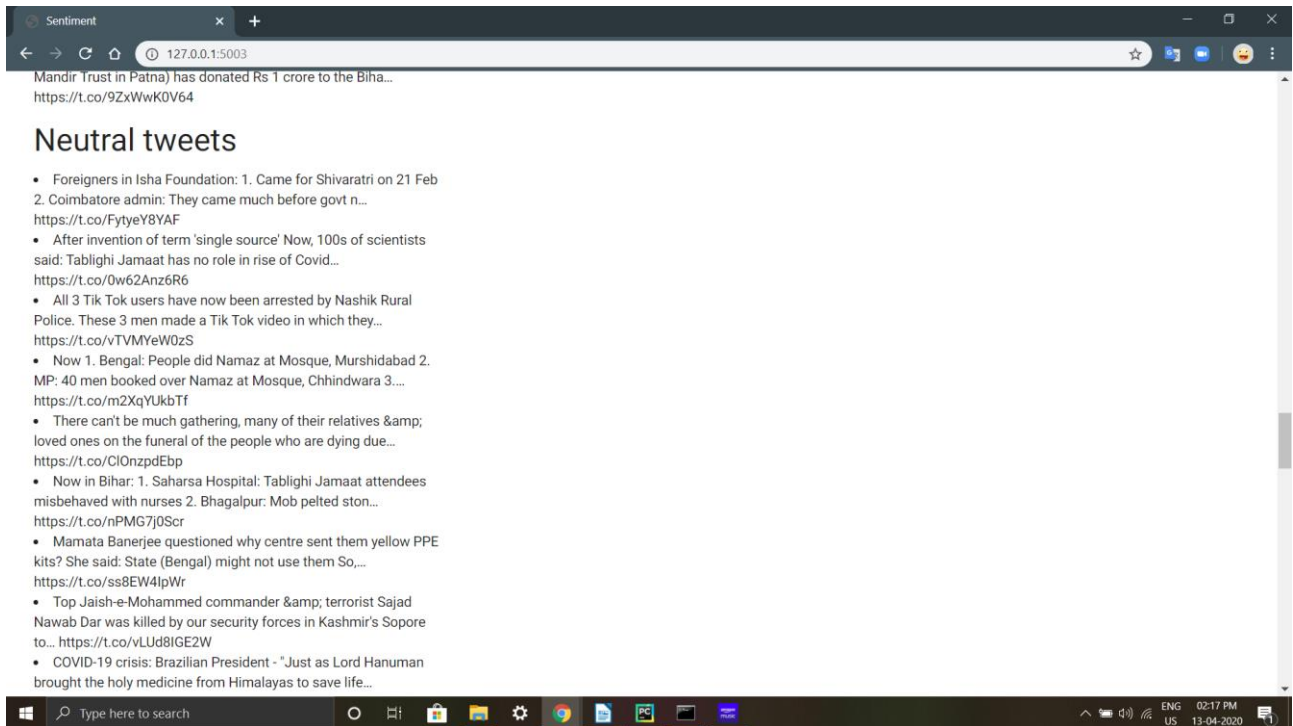
1.User-id based sentiment analysis with stored keywords:-



[Figure: 11]UserID Input page



[Figure: 12] UI Output



[Figure: 13] Tweets Output

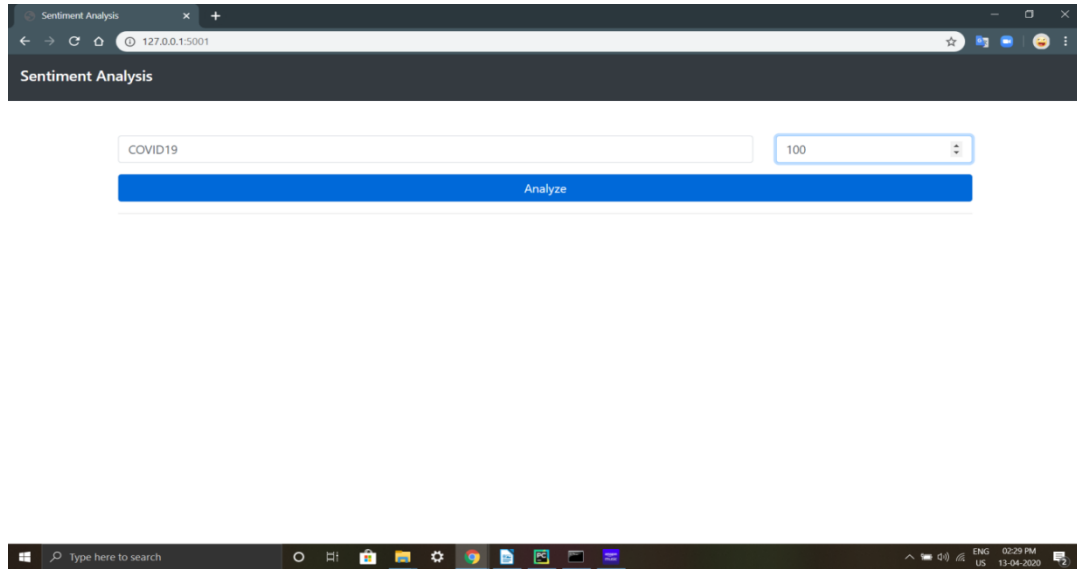
As they say that a picture is worth thousands of word like wise you can see that you just need to provide the username of particular user(it should be a public account) and it will fetch and plot it in pie chart along with its polarity that its *positive*, *negative* or *neutral*. By the way the fetched tweets are based on recent time stamp.

It also shows some recent categorized user tweets content also.

Now we will move toward on our next model,

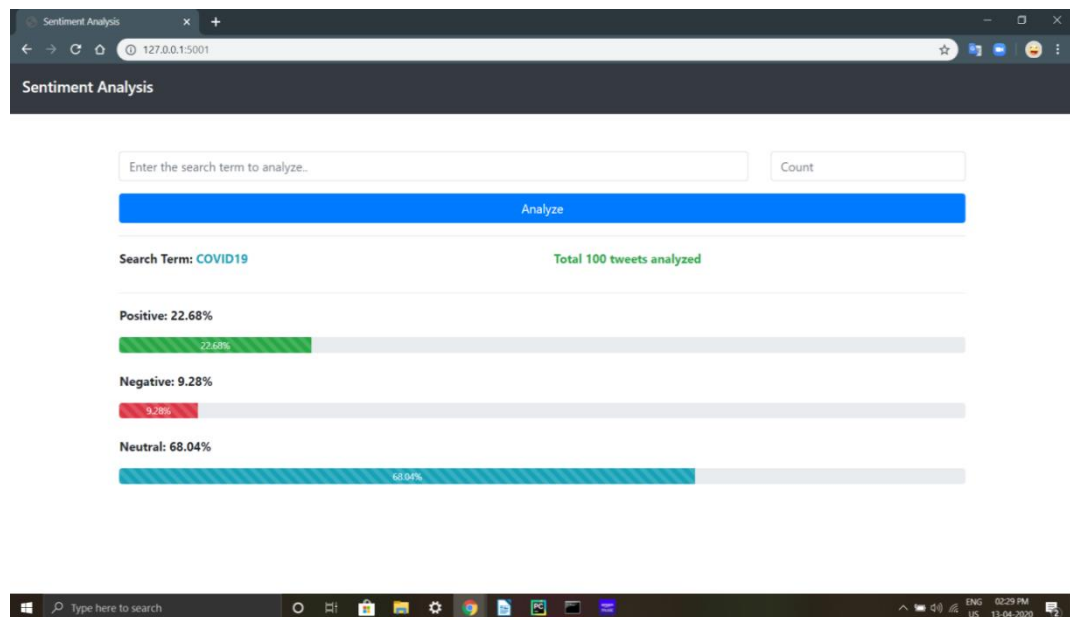
2. Keyword and count wise on live tweets:-

Here user will first enter the keyword for analysis and no of tweets to be fetched and submit for the analysis:-



[Figure: 14] Keyword Input UI

After that it will print the percentage wise polarity of tweets analysis that whether it's *negative*, *positive* or *neutral*.



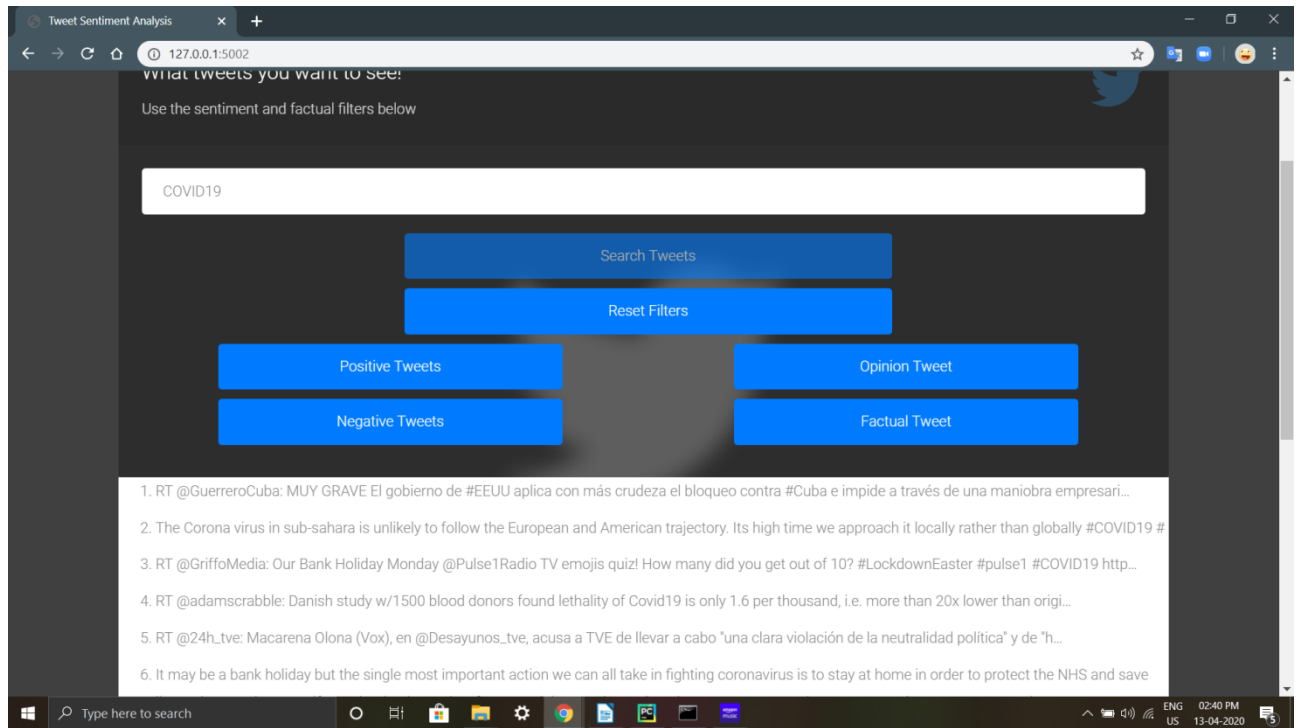
[Figure: 15] Keyword UI Output

Now we will move on to our third flask model.

3. Topic wise filtered sentimental analysis:

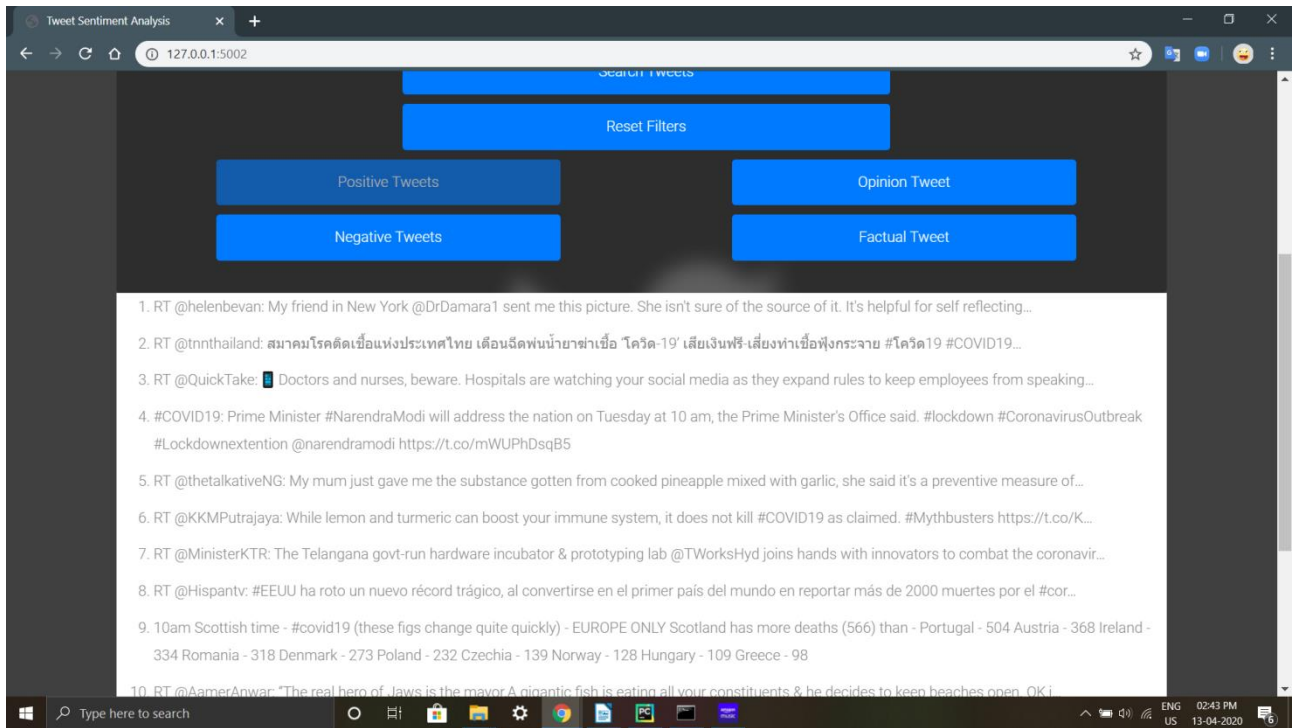
Here you will enter the keyword for search and do search tweets:-

As you can see that you can individually fetch *positive* or *negative* tweets:

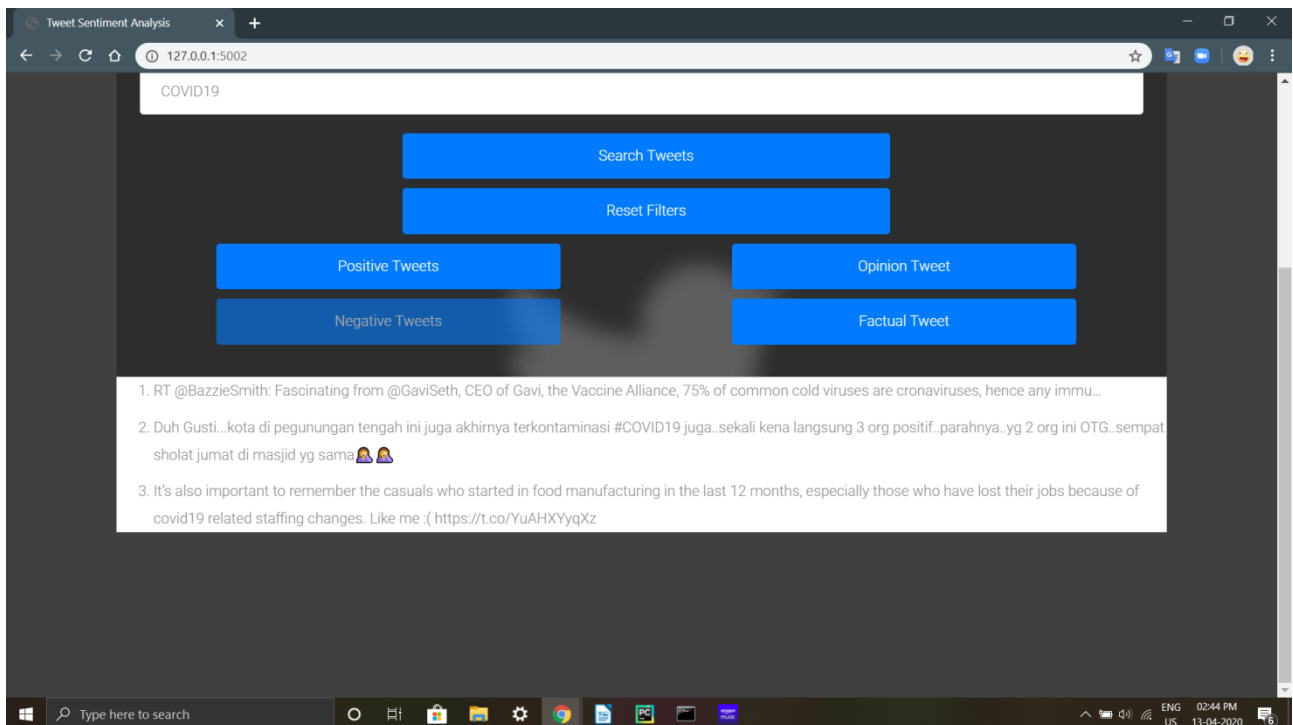


[Figure: 16] Topic Input UI

As you can see that you can individually fetch positive or negative tweets:



[Figure: 17] Topic Output 1

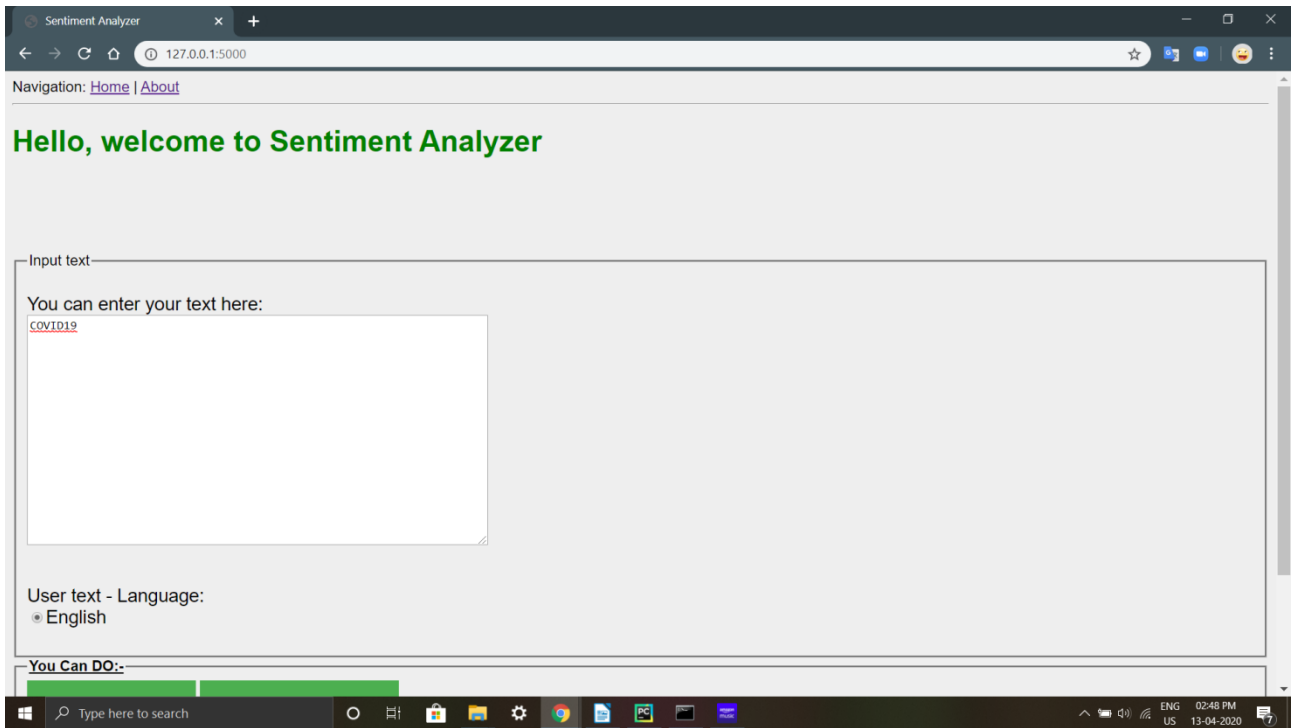


[Figure: 18] Topic Output 2

Also you can reset filters and do analysis again.

Now we move on to our next model,

4. Input text analysis on with stop-words :

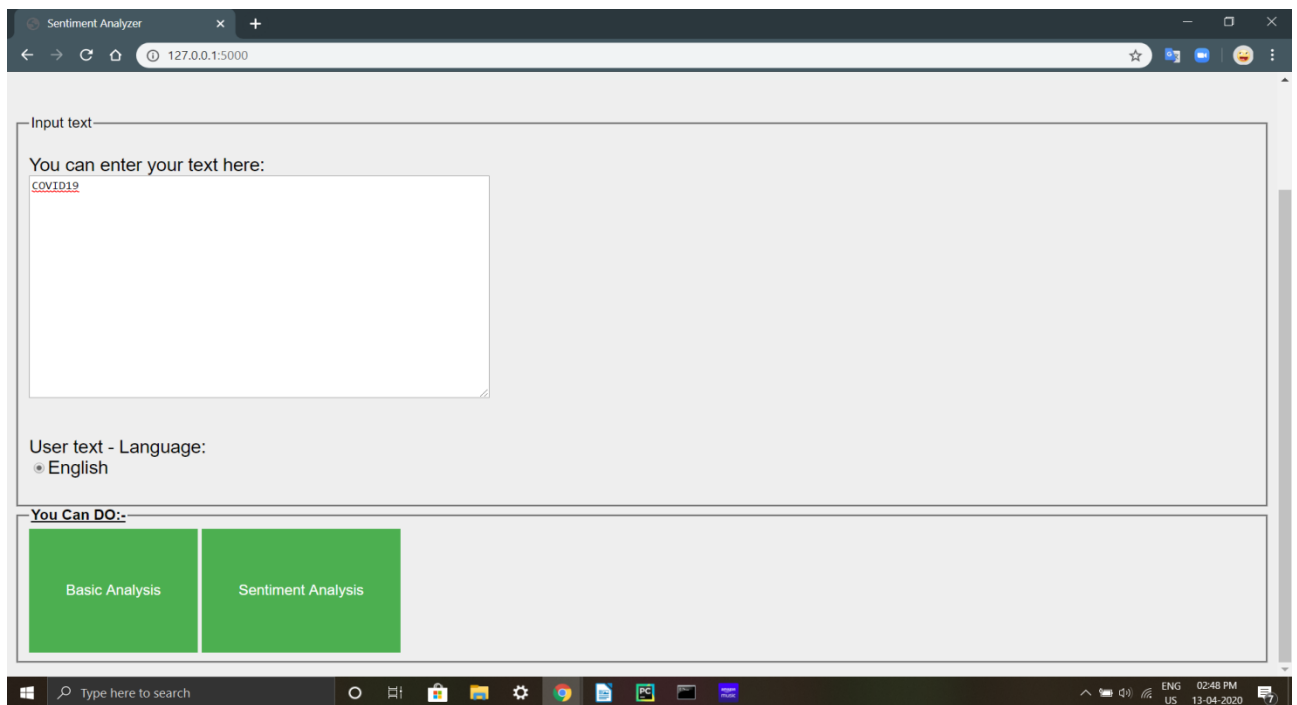


[Figure: 19] Input Text model UI

Here first you will enter your search keyword in search box:-

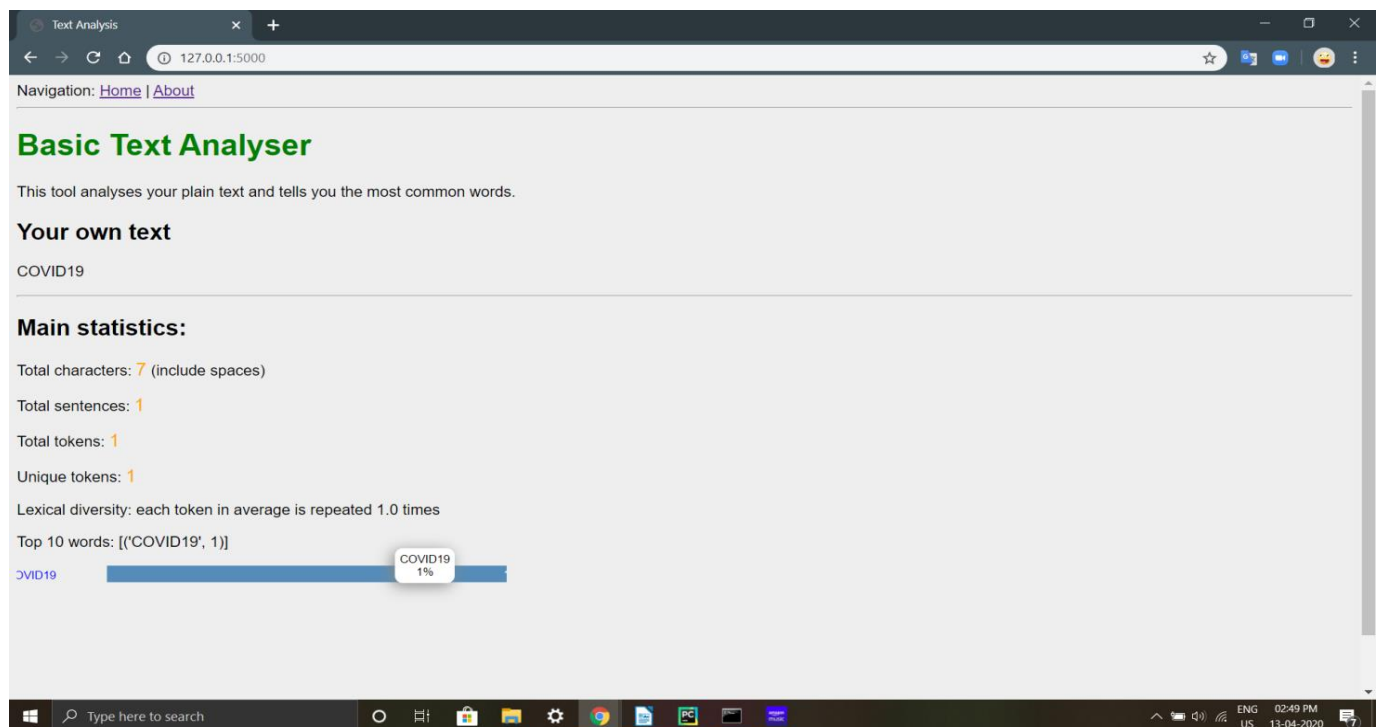
Now you have two options that is BASIC and ADVANCED analysis.

In basic sentimental analysis it will show you that how many sentences, characters and tokens are present in search box which you have entered.



[Figure: 20] Different Options

Basically its a word matrix type of thing.

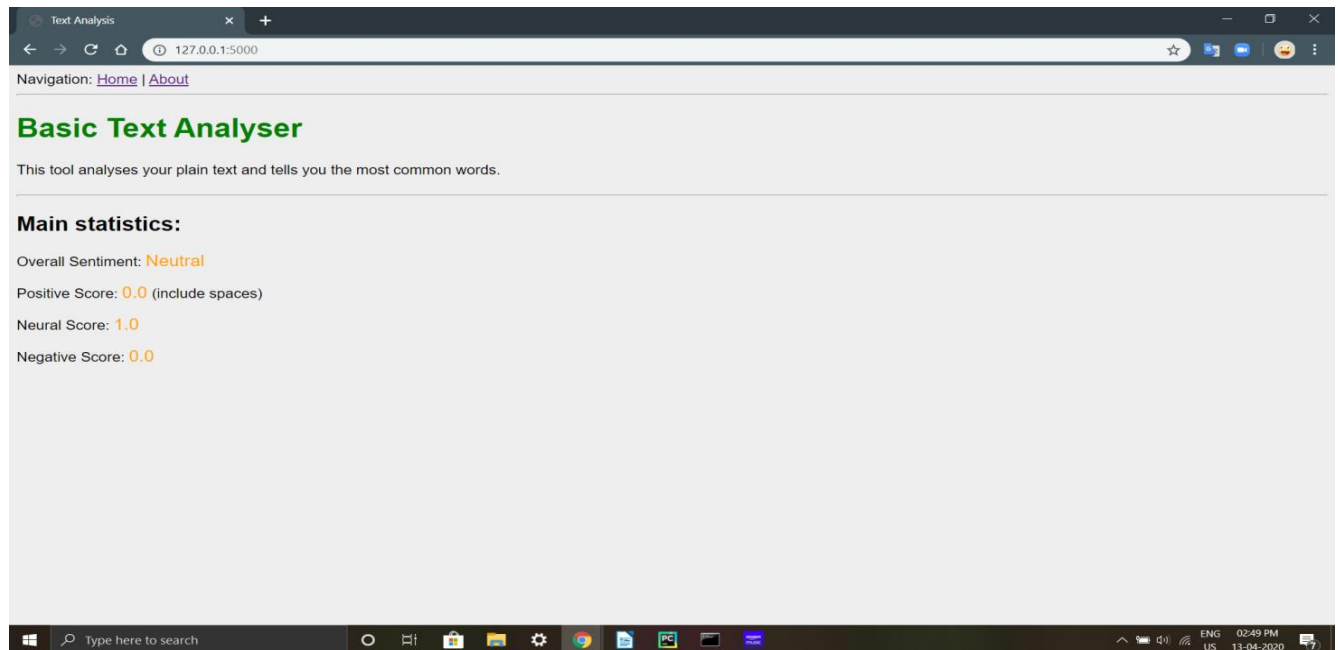


[Figure: 21] Input Text UI Analysis

And in advanced sentimental analysis, it will show you that how the polarity scores are positive, negative and neutral sentiments and also displays overall sentimental analysis results.

We have already mentioned that we have used python3 ML (machine learning) Libs like Tweepy, matplotlib, NLTK(natural language processing toolkit) and Flask framework for sentimental analysis. Also we have got the Twitter developers account credentials for analysis.

As an extension to this project we have gone for Facebook posts analysis but the Graph API which they have suggested is a paid version so we didn't proceed further.



[Figure: 22] Input Keyword Analysis

Chapter 5

Study and Survey of Related Systems

The research and analysis work took a good two three days to find out if the project could be completed within the allotted time frame or not. We went through different articles of Machine Learning and also went through different other articles to know which other technologies would be required to complete the project.

The development approach we took as a team to work out our project was to first research and study as many articles and content available from reliable sources. This was needed to be done as to check whether the project we took in hand was feasible or not. Also the research was to be done so as to find whether we could complete the given project in a given amount of time. Because time was also a priority and selecting a project which could not be completed in a given time was of no use.

We also divided our work into different parts so as to efficiently run the project. So first we decided to work on the backend portion as a whole and keeping the frontend portion of the project for the later part of the working stage. The idea was to initially develop a model which was a very basic model in the command prompt which would just accept the word and the number of tweets to be fetched and then give the output of the tweets in the prompt itself. And after developing the basic model, we then started further to add various features and make the frontend portion.

The main module of the system is the Twitter API. The Twitter API allows you to access the features of Twitter without having to go through the website interface. This can be useful for doing things like posting tweets or sending directed messages in an automated way with scripts. The Twitter platform offers access to corpus of data, via the API. It's important to note that the Twitter APIs are constantly evolving, and developing on the Twitter Platform is not a one-off event. The Twitter API (the term stands for Application Programming Interface) enables software developers to access and interact with public Twitter data. Developers can interact with this API by writing their own scripts or by using one of the open source libraries available in different programming languages.

The use of Twitter API was to fetch the real time tweets from the internet. After getting the keyword from the user, the tweets are fetched accordingly. The number of tweets to be analyzed is also taken from the user. So after applying our model, the exact number of tweets are fetched according the API and then the model works to produce the output of the tweet being positive, negative or neutral. The tweets fetched are stored in the database.

The main issue of the development was the approval of the developers at Twitter as it was a sensitive deal to allow us to fetch the tweets and also analyze them as giving the authority to anyone without any true meaning would cause damage and also be used for the wrong reasons. The sentiment analysis could be used for any politically motivated reason and hence the team at Twitter took lot of time to grant the access of the API. It took around a week of exchanging of emails to properly make them understand of why we wanted to use the Twitter API and what we wanted to do with the tweets fetched through the API.

While searching and analyzing the different articles and projects, we found out many different projects regarding the sentiment analysis of Twitter. We found a open-source project which had many different functionalities and features than our current project. In that project it was available that the users can know the sentiment of tweets based on a particular location. For example, the user enters the region as Gujarat, then the prompt appears to enter the number of tweets to be fetched and perform sentiment analysis. The model then would fetch the latest tweets irrespective of the keyword and would fetch the latest tweets of that particular region and perform sentiment analysis. We studied the project and found it useful to implement but because of time constraints and also the model being very complicated we decided not to pursue the model.

While searching and analyzing the different articles and projects, we found out many different projects regarding the sentiment analysis of Twitter. We also found a model which could tell the machine that the user has used for the tweeting purpose. There are different machines or systems available to tweet such as from the Android phone, it could be an iPhone or it could be the standard Twitter Desktop version. The model could tell after the tweets have been fetched about the system that has been used for tweeting. And in turn this could be used for determining the different types of machines and hardware available in the particular area for the time being. By knowing the number of different types of machines and hardware available in the region, it could be found out where the twitter is most used from.

We also found different models which had many nice features for the Frontend which made it the model looked very attractive. One of the project had the 3D view of the world which could be rotated and pointed to a particular place for performing the sentiment analysis of the tweets of that particular region. Also it had a nice graphic for the plotting of the graph which had an effect of fading in the screen.

We also found the model where in the sentiments of tweets of different users can be compared regarding the same key-word. This feature is useful for knowing the opinion of two different users on the same topic or key-word. It could also be used to derive what the different leaders have to say on a particular topic.

Chapter 6

Conclusion and future Scope

Nowadays, sentiment analysis or opinion mining is a hot topic in machine learning. We are still far to detect the sentiments of corpus of texts very accurately because of the complexity in the English language and even more if we consider other languages such as Chinese.

In this project we tried to show the basic way of classifying tweets into positive or negative category using Naive Bayes as baseline and how language models are related to the Naive Bayes and can produce better results. We could further improve our classifier by trying to extract more features from the tweets, trying different kinds of features, tuning the parameters of the naïve Bayes classifier, or trying another classifier all together.

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

One more feature we have that is worth exploring is whether the information about relative position of word in a tweet has any effect on the performance of the classifier. Although studies conducted explored a similar feature and reported negative results, their results were based on reviews which are very different from tweets and they worked on an extremely simple model.

In this research we are focussing on general sentiment analysis. There is potential of work in the field of sentiment analysis with partially known context. For example we noticed that users generally use our website for specific types of keywords which can be divided into a various distinct classes, namely: politics/politicians, celebrities, products/brands, sports/sportsmen and journalism/movies/music. So we can attempt to perform separate sentiment analysis on tweets that only belong to one of these classes (i.e. the training data would not be general but specific to one of these categories) and compare the results which we obtained if we apply general sentiment analysis on it instead.

Reference:-

- [1] Efthymios Kouloumpis and Johanna Moore, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 3, July 2012
- [2] S. Batra and D. Rao, "Entity Based Sentiment Analysis on Twitter", Stanford University, 2010
- [3] Saif M. Mohammad and Xiaodan Zhu, Sentiment Analysis on of social media texts, 2014
- [4] Ekaterina Kochmar, University of Cambridge, at the Cambridge Coding Academy Data Science. 2016
- [5] Manju Venugopalan and Deepa Gupta, Exploring Sentiment Analysis on Twitter Data, IEEE 2015
- [6] Brett Duncan and Yanqing Zhang, Neural Networks for Sentiment Analysis on Twitter. 2017
- [7] Afroze Ibrahim Baqapuri, Twitter Sentiment Analysis: The Good the Bad and the OMG!, Proceedings of the Fifth International AAI Conference on Weblogs and Social Media. 2011
- [8] Jin Bai, JianYun Nie. Using Language Models for Text Classification.
- [9] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, Rebecca Passonneau. Sentiment Analysis of Twitter Data.