A Mini Project Report on

Sentiment Analyzer

Submitted to the Department of Computer Science & Engineering, GNITS in the partial fulfillment of the academic requirement for the award of B. Tech (CSE) under JNTUH

By
D. Sreelekha (16251A0510)
G. Kaumudi (16251A0515)
K. Nirmala (17255A0508)

Under the guidance of

Dr. N. Kalyani
Professor

Department of Computer Science & Engineering
G. Narayanamma Institute of Technology & Science
(Autonomous) (for Women)
Shaikpet, Hyderabad- 500 104.

Affiliated to
Jawaharlal Nehru Technological University Hyderabad
Hyderabad – 500 085
Nov, 2019

Department of Computer Science & Engineering
G. Narayanamma Institute of Technology & Science
(Autonomous) (for Women)
Accredited by NBA & NAAC &Affiliated to JNTUH
Hyderabad-500104, T. S, INDIA

Certificate

This is to certify that the Project report on "**Sentiment Analyzer**" is a bonafide work carried out by **D. Sreelekha (16251A0510), G. Kaumudi (16251A0515), K. Nirmala (17255A0508)** in the partial fulfillment for the award of B. Tech degree in Computer Science & Engineering, G. Narayanamma Institute of Technology & Science, Shaikpet, Hyderabad, affiliated to Jawaharlal Nehru Technological University, Hyderabad under our guidance and supervision.

The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Internal Guide
Dr. N. Kalyani
Professor

Head of the Department
Dr. M. Seetha
Professor & Head
Department of CSE

External Examiner

Acknowledgements

We would like to express our sincere thanks to **Dr K. Ramesh Reddy, Principal,** GNITS, for providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. M. Seetha, Professor and HOD**, Dept. of CSE, GNITS for all the timely support and valuable suggestions during the period of our miniproject.

We are extremely thankful to **Mrs. T. Swapna, Asst. Prof, Mrs. Nandadevi D.R. Asst. Prof,** Dept. of CSE, GNITS, the miniproject coordinators and **Dr. N. Kalyani, Professor,** Dept. of CSE, GNITS for their encouragement and support throughout the project.

We are extremely thankful and indebted to our internal guide, **Dr. N. Kalyani, Professor,** Dept. of CSE, GNITS for her constant guidance, continuous advice, encouragement and moral support throughout the miniproject.

Finally, we would also like to thank all the faculty and staff of CSE Department who helped us directly or indirectly, parents and friends for their cooperation in completing the miniproject work.

D. Sreelekha (16251A0510)
G. Kaumudi (16251A0515)
K. Nirmala (17255A0508)

ABSTRACT

Ever since the earlier times, opinions have been best put forth through natural language. Sentimental mining is an important research area and due to huge number of daily posts on social networks, extracting people's opinion is a challenging task. About 90 percent of today's data has been provided during the last two years and getting insight into this large scale data is not trivial. Though pictures and illustrations are widely used to get the message across, they would not possibly deliver the exact sentiment. Natural language is a complex combination of various words to deliver the intentions. This kind of data is largely available in different social networking sites and is highly unstructured and heterogeneous.

In this project, the aim is to analyze comments and feedback in the form of text to classify them as positive, neutral or negative opinions. To analyze this data, this project aims to design a "Sentiment Analyzer" that uses standard technology for sentiment analysis like Support Vector Machine algorithm. This application is be used to analyze the information on Twitter about different brands and products in the domain of electronic products and suggest a best product. The project achieved best classification results for Support Vector Machine algorithm with respect to the accuracy, precision and recall metrics and hence accurately suggests the best choice.

Table of Contents

1. INTRODUCTION

Twitter has emerged as a major micro-blogging website, having over 100 million users generating over 500 million tweets every day. With such large audience, Twitter has consistently attracted users to convey their opinions and perspective about any issue, brand, company or any other topic of interest. Due to this reason, Twitter is used as an informative source by many organizations, institutions and companies.

On Twitter, users are allowed to share their opinions in the form of tweets, using only 140 characters. This leads to people compacting their statements by using slang, abbreviations, emoticons, short forms etc. Along with this, people convey their opinions by using sarcasm and polysemy (The ambiguity of an individual word to express two or more different meanings).Hence it is justified to term the Twitter language as unstructured.

In order to extract sentiment from tweets, sentiment analysis is used by analyzing the words of the text. The results from this can be used in many areas like analyzing and monitoring changes of sentiment with an event, sentiments regarding a particular brand or release of a particular product in the domain of electronics.

A lot of research has been done on Twitter data in order to classify the tweets and analyze the results. In this project the primary work is on how to perform sentiment analysis on Twitter data using Python. The project implements a systematic process to retrieve data from Twitter and process in for further analysis and use the final statistics to get an insight of the best products that are available currently in the market.

A web application is developed that serves as an interface where the user can interact with the system to compare between different brands and products. Once the user provides the input, searching is done to obtain the most recent tweets with the help of the Twitter API. Therefore, this project helps in real time analysis of opinion on products and brands.

1.1. Objectives:

- To extract the most recent and relevant tweets using Twitter Search API for analyzing the sentiment of the product.
- To implement an algorithm for automatic classification of text into positive, negative or neutral.
- To reduce the labour cost with the use of Twitter API Search using the tweepy package.
- To easily identify customer opinions without many expenses for business firms with the Sentiment Analyzer.
- To determine which is the best brand and best model for the domain of electronic products.
- To provide user friendly interface with easy navigation through the web application.
- Graphical representation of sentiment in the form of Pie-Chart.

1.2. Methodology:

The basic steps for performing sentiment analysis includes data collection, pre-processing of data, feature extraction, selecting baseline features, sentiment detection and performing classification either using simple computation or else machine learning approaches. Sentiment analysis approaches can be broadly categorized in two classes – lexicon based and machine learning based. Lexicon based approach is unsupervised as it proposes to perform analysis using lexicons and a scoring method to evaluate opinions [6]. Whereas machine learning approach involves use of feature extraction and training the model using feature set and some dataset. For example, there can be three class tweet sentiment classification (positive, negative and neutral).

Performing sentiment analysis is challenging on Twitter data, as mentioned earlier. Here are the reasons for this:

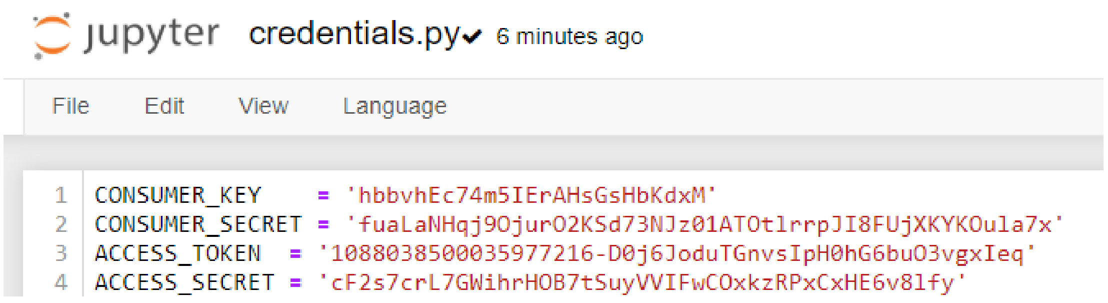
i. **Limited tweet size:** with just 140 characters in hand, compact statements are generated, which results sparse set of features.

ii. **Use of slang:** these words are different from English words and it can make an approach outdated because of the evolutionary use of slangs.

iii. **Twitter features:** it allows the use of hash tags, user reference and URLs. These require different processing than other words.

iv. **User variety:** the users express their opinions in a variety of ways, some using different language in between, while others using repeated words or symbols to convey an emotion.

In order to perform sentiment analysis, it is required to collect data from the desired source which is the Twitter website. This is done by creating a Twitter Search API using a Twitter account. The authentication of this API provides us with access tokens. With these Twitter authenticates future requests for extracting tweets by the web application. These keys are shown in the figure 1.1 below.



```python
CONSUMER_KEY    = 'hbbvhEc74m5IErAHsGsHbKdxM'
CONSUMER_SECRET = 'fuaLaNHqj9OjurO2KSd73NJz01ATOtlrrpJI8FUjXKYKOula7x'
ACCESS_TOKEN  = '1088038500035977216-D0j6JoduTGnvsIpH0hG6buO3vgxIeq'
ACCESS_SECRET = 'cF2s7crL7GWihrHOB7tSuyVVIFwCOxkzRPxCxHE6v8lfy'
```

Fig 1.1. Twitter API credentials

Of the four keys, shown above, the CONSUMER_KEY and CONSUMER_SECRET are used to authenticate the API every time a request is made to extract tweets.

The tweets extracted as described above undergo various steps of pre-processing which make them more machine sensible than its previous form. Later each of these tweets is classified as belonging to one of the three classes as mentioned above. All the classified tweets belonging to the reviews of a certain product are combined giving an aggregate percentage of the emotions towards that product.

- **Tweet Collection:**

Tweet collection involves gathering relevant tweets about the particular area of interest. The tweets are collected using Twitter's streaming API. The format of the retrieved text is converted as per convenience. The dataset collected is imperative for the efficiency of the model. The division of dataset into training and testing sets is also a deciding factor for the efficiency of the model. The training set is the main aspect upon which the results depend.

- **Preprocessing of Data:**

  Preprocessing includes the following things.

  - All URL (e.g. www.xyz.com), hash tags (e.g. #topic) and targets (@username) are removed.
  - Uppercase letters are converted into Lowercase.
  - All the texts are broken down into tokens. This process is called tokenization. For example “this is an amazing phone” is broken into individual tokens such as ‘this’, ‘is’, ‘an’, ‘amazing’, ‘phone’. On encountering a space, a token is identified.
  - Stop words like articles, prepositions, conjunctions, and pronouns are removed. Stop words provide little or no information.

- **Feature Extraction:**

A feature is a piece of information that can be used as a characteristic which can assist in solving a problem. The quality and quantity of features is very important as they are important for the results generated by the selected model.

Selection of useful words from tweets is feature extraction [5] [12].

- **Unigram features** – one word is considered at a time and decided whether it is capable of being a feature.
- **N-gram features** – more than one word is considered at a time. These words are consecutive in the text that is considered for processing. This project uses 2 gram features of the tweets of the training data.

Frequency analysis is a method to collect features with highest frequencies. Further, they removed some of them due to the presence of words with similar sentiment (for example happy, joy, ecstatic etc.) and created a group of these words. Along with this affinity analysis is performed.

- **Training the Model:**

The extracted features are vectorized and are transformed to numerical values which are then fed to the machine learning model. Different models are trained on the training dataset of labeled electronic products and chose the best classification algorithm by comparing their accuracy scores. The model with the highest accuracy score is selected and saved.

- **Classification:**

  The tweets searched with the Twitter API are cleaned and normalized. The saved model is used to predict the class label for each of these tweets. All the predictions are aggregated to give percentage of tweets for all the three class labels.

1.3. Organization of the project:

This project is organized in such a way that it follows a series of fixed pages. The next chapter gives a brief introduction on Theoretical analysis of the proposed project which consists of the work related to Existing systems and proposed system followed by Advantages, Disadvantages and Applications. Chapter 3 describes the complete architecture along with modules and Support Vector Machine algorithm is explained in detail. Chapter 4 describes the Implementation, coding and Technology used, discussion on UML diagrams and results, few screenshots of related work in Chapter 5. Finally, the Conclusions and Scope for Future work is stated in Chapter 6.

2. THEORETICAL ANALYSIS OF THE SENTIMENT ANALYZER

2.1. Literature Review:

Sentiment analysis has been handled as a Natural Language Processing task at many levels of granularity. Starting from being a document level classification task (Turney, 2002; Pang and Lee, 2004) [10, 11], it has been handled at the sentence level (Hu and Liu [9], 2004; Kim and Hovy, 2004) and more recently at the phrase level. Microblog data like Twitter, on which users post real time reactions to and opinions about everything, poses newer and different challenges. Some of the early and recent results on sentiment analysis of Twitter data are by Go et al. (2009) [8], (Bermingham and smeaton, 2010) They use tweet sending in positive emotions like “☺” as positive and negative emoticons like “☹” as negative. They build models using Naïve Bayes, Support Vector Machines (SVM), and they report SVM outperforms other classifiers. In terms of feature space, they try a Unigram, Bigram model in conjunction with parts-of-speech (POS) features. They use polarity prediction from websites as noisy labels to train a model. They propose the use of syntax features of tweets like prior polarity of words. They perform extensive feature analysis and feature selection and demonstrate that abstract linguistic analysis features contributes to the classifier accuracy. For feature selection suggested to remove [10] objective sentences by extracting subjective ones.

2.2. Existing System:

The existing system works only on the dataset which is constrained to a particular topic. The existing systems [7] also do not determine the measure of impact the results determined can have on the particular field taken into consideration and it does not allow retrieval of data based on the query entered by the user i.e. it has constrained scope. In simple words, it works on static data rather than dynamic data. Unsupervised algorithms like Vector Quantization, are used for data compression, pattern recognition, facial and speech recognition, etc and therefore cannot be used in determining sentiment in twitter data. Apriori algorithm fails to handle large datasets and as a result can generate faulty results.

2.3. Proposed System:

In the proposed system, tweets are retrieved from twitter using twitter API. The collected tweets will be subjected to preprocessing. Supervised algorithm is applied on the data [7]. The Supervised algorithm used in the system is Support Vector Machine (SVM) [1]. The results of the algorithms i.e. the sentiment will be represented in graphical manner (pie charts/bar charts). The proposed system is more effective than the existing one. This is because one will be able to know how the statistics determined from the representation of the result can have an impact in a particular field.

The objective is to review the product based on the comments given by the customer. The comments show the opinion of the user towards the product. These comments may be positive, negative or neutral. These comments may be in the form of sentences. In order to gain the sentiment of the user, these sentences have to be segregated into words in which they are processed using algorithms. Due to rapid growth of data in E-commerce, it is used to reveal the quality of product. As these websites have become major source for the customers to get rating of a product but due to huge amount of data available it becomes difficult for them to make decisions.

Proposed system summarizes the feedbacks collected from twitter, extracts the opinions from all this information, gives an overall view of the product, that could save time and ease the decision process of the customer.

2.4. Advantages and Disadvantages of proposed system:

2.4.1. Advantages:

- Sentiment analysis is a useful tool for any organization or group for which public sentiment or attitude towards them is important for their success.
- On social media, blogs, and online forums millions of people are busily discussing and reviewing businesses, companies, and organizations. And those opinions are being ‘listened to’ and analyzed.

- Those being discussed are making use of this enormous amount of data by using computer programs that don't just locate all mentions of their products, services, or business, but also determine the emotions and attitudes behind the words being used.
- The results from sentiment analysis help businesses understand the conversations and discussions taking place about them, and helps them react and take action accordingly.
- They can quickly identify any negative sentiments being expressed, and turn poor customer experiences into very good ones.
- They can create better products and services, and they can formulate the marketing messages they send out according to the sentiments being expressed by their target audience or customers.
- They can also conduct market research into general sentiment around key issues, topics, products, and services, before developing and launching their own new services, products or features.

2.4.2. Disadvantages:

- Sometimes computer programs have problems recognizing things like sarcasm and irony, negations jokes-the sort of things a person would have trouble identifying.
- Cannot identify underlying meaning of the Emoticons.

2.5. Applications:

- **Reputation Management:**

  It can also be called as brand monitoring. It is well known how much good reputation means these days when the majority of us check social media reviews as well as review sites before making a purchase decision. One has to be aware of those opinions in the first place. That's where social media monitoring combined with sentiment analysis comes in.

- **Customer support:**

  Social media are channels of communication with the customers these days, and whenever they are unhappy about something related to you, whether or not it’s your fault, they all call out on Face book/Twitter/Instagram.

- **Competitor Monitoring:**

  Chances are some of the competitors are getting bad press online. It’s where one could step in as long as one is aware of those negative mentions.

3. SENTIMENT ANALYZER WITH SVM

3.1. Architecture:

The block diagram below shows the architecture of the project. The first step is collection of data from Twitter as per user input, then this data is preprocessed in the text preparation step. The machine learning model is run on this text to detect sentiment and classify the text accordingly. These results are appropriately represented to the user.

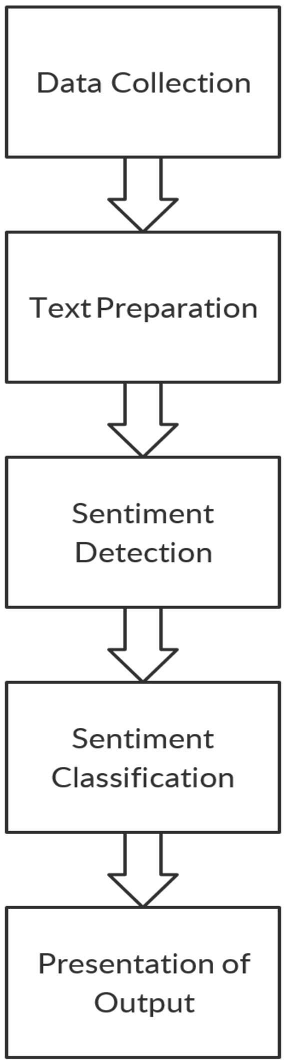


Fig 3.1 Architecture of Sentiment Analyzer

Methods of Sentiment Analysis:

- **Data Collection:**

  Consumers usually express their sentiments on public forums like the blogs, discussion boards, product reviews as well as on their private logs – Social network sites like Facebook and Twitter. Opinions and feelings are expressed in different way, with different vocabulary, Context of writing, usage of short forms and slang, making the data huge and disorganized. Manual analysis of sentiment data is virtually impossible. Therefore, Special programming languages like python are used to process and analyze the data.

- **Text Preparation:**

  Text preparation is nothing but filtering the extracted data before analysis. It includes identifying and eliminating non-textual content and content that is irrelevant to the area of study from the data.

- **Sentiment Detection:**

  At this stage, each sentence of the review and opinion is examined for subjectivity. Sentences with subjective expressions are retained and that which conveys objective expressions are discarded. Sentiment analysis is done at different levels using common computational techniques like Unigrams, lemmas, negation and so on.

- **Sentiment Classification:**

  Sentiments can be broadly classified into three groups, positive, negative and neutral. At this stage of sentiment analysis methodology, each subjective sentence detected is classified into groups-positive, negative, good, bad, like, dislike.

- **Presentation of Output:**

  The main idea of sentiment analysis is to convert unstructured text into meaningful information. After the completion of analysis, the text results are displayed on graphs like pie chart.

Preprocessing Methods:

The dataset will go through the pre-processing task such as tokenization, stop word removal, lowercase conversion and stemming. Tokenization is the procedure of splitting a text into words, phrases, or other meaningful parts, namely tokens. Stop words are the words that are commonly encountered in texts without dependency to a particular topic such as conjunctions, prepositions, etc. Another preprocessing step is lowercase conversion. All uppercase characters are usually converted to their lowercase forms before the classification stages. Finally, stemming process where root word is obtained, stem of derived words.

Feature Extraction:

Feature Extraction is an important step when dealing with natural languages because the text collected is not in a form understandable by a computer. In the fields of computational linguistics, an n-gram is a contiguous sequence of n items from a given sample of text. The items can be syllables, letters or words. A n-gram of size 1 is referred to as a “unigram”, size 2 is a “bigram”.

If a tweet that goes something like “I do not like the views” then one cannot just feed this into a learning algorithm. Depending on how the tokenizer is made, the previous tweet is transformed into something like [‘I’, ‘do’, ‘not’, ‘like’, ‘the’, ‘views’]. This is a unigram list of tokens. Bigram list of tokens would be [‘I do’, ‘do not’, ‘not like’, ‘like the’, ‘the views’] i.e. taking two tokens at a time. For each token count is calculated which is understandable by a computer.

A feature is a piece of information that can be used as a characteristic which can assist in solving a problem. The quality and quantity of features is very important as they are important for the results generated by the selected model.

Text Classification Method Selection:

Support Vector Machine (SVM) has been chosen for the classification in the experiments. The support-vector machine is a learning machine for three-group classification problems. It is used to classify the texts into positives, negatives and neutrals. SVM works well for text classification due to its advantages such as potential to handle large features. Another advantage is SVM is robust when there is a sparse set of examples and also because most of the problem are linearly separable.

Training the model:

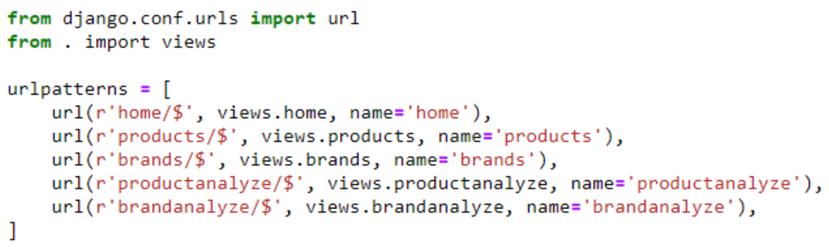
The extracted features are vectorized and are transformed to numerical values which are then fed to the machine learning model. Different models are trained on the training dataset of labeled electronic products and chose the best classification algorithm by comparing their accuracy scores. The model with the highest accuracy score is selected and saved.

3.2. Django Module Description:

The Django web framework [13] provides these essential modules for the development of the web application.

3.2.1. urls.py:

This file is located under /home/username/dango. ‘dango’ is the project name. This module takes care of connecting the URLs (Uniform Resource Locator) to the views of the project. Every URL has a corresponding view which is capable of rendering an HTML page when the URL is requested.



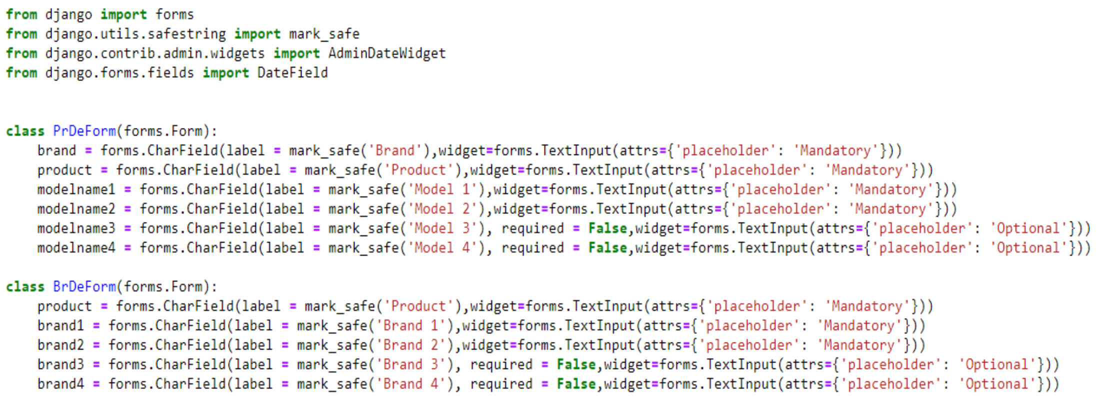
```
from django.conf.urls import url
from . import views

urlpatterns = [
    url(r'home/$', views.home, name='home'),
    url(r'products/$', views.products, name='products'),
    url(r'brands/$', views.brands, name='brands'),
    url(r'productanalyze/$', views.productanalyze, name='productanalyze'),
    url(r'brandanalyze/$', views.brandanalyze, name='brandanalyze'),
]
```

Fig 3.2 URLs Module

This module contains all the URLs corresponding to different web pages in the web application. It maps the URLs of these pages to their corresponding views with the url() method of the Django's configuration i.e., django.conf module.

3.2.2. forms.py:

The 'forms' module generates the forms for the appropriate web pages as HTML (Hyper Text Markup Language) forms. The created forms can easily validate the user data without further explicit validations with the help of Python's 'safestring' module.



```python
from django import forms
from django.utils.safestring import mark_safe
from django.contrib.admin.widgets import AdminDateWidget
from django.forms.fields import DateField


class PrDeForm(forms.Form):
    brand = forms.CharField(label = mark_safe('Brand'),widget=forms.TextInput(attrs={'placeholder': 'Mandatory'}))
    product = forms.CharField(label = mark_safe('Product'),widget=forms.TextInput(attrs={'placeholder': 'Mandatory'}))
    modelname1 = forms.CharField(label = mark_safe('Model 1'),widget=forms.TextInput(attrs={'placeholder': 'Mandatory'}))
    modelname2 = forms.CharField(label = mark_safe('Model 2'),widget=forms.TextInput(attrs={'placeholder': 'Mandatory'}))
    modelname3 = forms.CharField(label = mark_safe('Model 3'), required = False,widget=forms.TextInput(attrs={'placeholder': 'Optional'}))
    modelname4 = forms.CharField(label = mark_safe('Model 4'), required = False,widget=forms.TextInput(attrs={'placeholder': 'Optional'}))

class BrDeForm(forms.Form):
    product = forms.CharField(label = mark_safe('Product'),widget=forms.TextInput(attrs={'placeholder': 'Mandatory'}))
    brand1 = forms.CharField(label = mark_safe('Brand 1'),widget=forms.TextInput(attrs={'placeholder': 'Mandatory'}))
    brand2 = forms.CharField(label = mark_safe('Brand 2'),widget=forms.TextInput(attrs={'placeholder': 'Mandatory'}))
    brand3 = forms.CharField(label = mark_safe('Brand 3'), required = False,widget=forms.TextInput(attrs={'placeholder': 'Optional'}))
    brand4 = forms.CharField(label = mark_safe('Brand 4'), required = False,widget=forms.TextInput(attrs={'placeholder': 'Optional'}))
```

Fig 3.3 Forms Module

In this module, two forms are created. The forms created have text fields in HTML format. However, they do not implicitly create the submit button or formaction here. The submit button and formaction is written in HTML. The form PrDeForm has brand, product, modelname1, modelname2, modelname3, modelname4 as its fields. Of these, only modelname3 and modelname4 are optional fields as two models are essential to make a comparison.

In the BrDeForm, there are five fields: product, brand1, brand2, brrand3 and brand4. Of these, brand3 and brand4 are optional fields. Hence the web app provides comparison between two, three and four different brands of a product or different models of a brand of the product.

3.2.3. views.py:

The views module renders the web pages with which the user interacts with the web app. All the modules are integrated into the views module as shown below.

```python
from django.shortcuts import render
from .forms import PrDeForm
from .forms import BrDeForm
from . import extractorclone
import pandas as pd
```

Fig 3.4 Integration of Forms Module to render views by the Views Module

When the user requests for a URL to the server, it is first mapped to the appropriate view by the URLs module as discussed previously. Then, the corresponding view is rendered as a web page to the user. To use the Django framework for this, import ‘render’ method for the purpose of providing respective web pages by URL.

```python
def home(request):
    return render(request, 'LSVC/home.html')
```

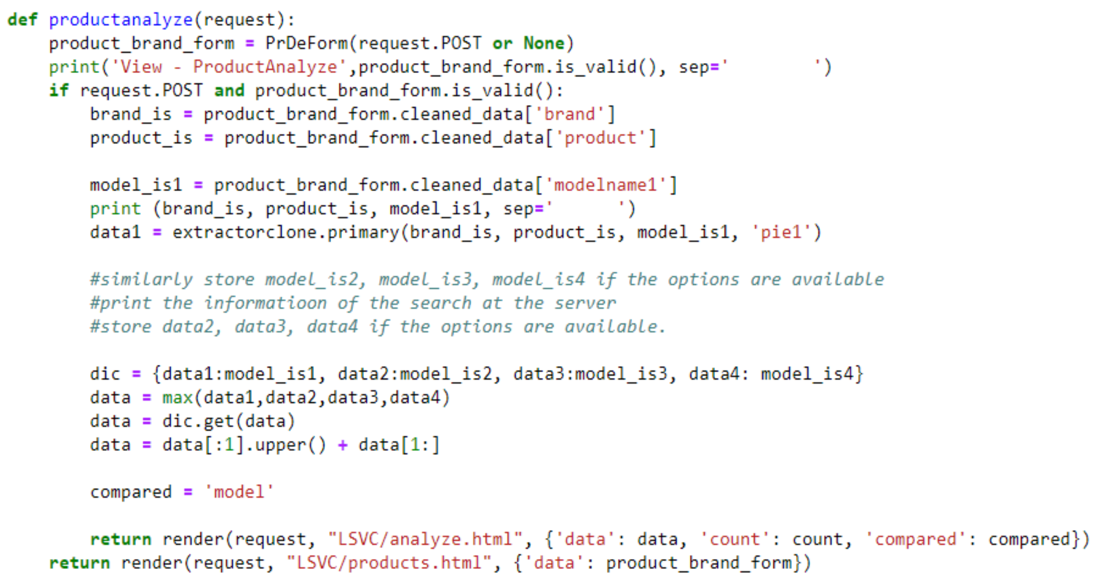
Fig 3.5 Rendering Home HTML

The method ‘home’ renders the home page in HTML format when an HTTP request is made for the URL – ‘LSCV/home.html’. This view, i.e., the method ‘home’ is corresponded to render home HTML page by the URLs module as seen in section 3.2.1.

```python
def products(request):
    form = PrDeForm(request.POST)
    if form.is_valid() and form.cleaned_data():
        form.save()
        brand = form.cleaned_data['brand']
        prod = form.cleaned_data['product']
        modelname1 = form.cleaned_data['modelname1']
        modelname2 = form.cleaned_data['modelname2']
        modelname3 = form.cleaned_data['modelname3']
        modelname4 = form.cleaned_data['modelname4']
    pform = PrDeForm()
    return render(request, 'LSVC/products.html', {'form': form})
```

Fig 3.6 Rendering Products HTML View that contains the Product Details Form

When the product view is requested, it renders the webpage after generating the form called PrDeFrom for the user to fill in the product details. If the user fills in the form with the valid data, the form is saved and used in the product analyze function, the code of which is shown below. The function stores the user inputs, extracts corresponding tweets and performs sentiment analysis. This final result is rendered as analyze.html page.



```python
def productanalyze(request):
    product_brand_form = PrDeForm(request.POST or None)
    print('View - ProductAnalyze',product_brand_form.is_valid(), sep='        ')
    if request.POST and product_brand_form.is_valid():
        brand_is = product_brand_form.cleaned_data['brand']
        product_is = product_brand_form.cleaned_data['product']

        model_is1 = product_brand_form.cleaned_data['modelname1']
        print (brand_is, product_is, model_is1, sep='      ')
        data1 = extractorclone.primary(brand_is, product_is, model_is1, 'pie1')

        #similarly store model_is2, model_is3, model_is4 if the options are available
        #print the informatioon of the search at the server
        #store data2, data3, data4 if the options are available.

        dic = {data1:model_is1, data2:model_is2, data3:model_is3, data4: model_is4}
        data = max(data1,data2,data3,data4)
        data = dic.get(data)
        data = data[:1].upper() + data[1:]

        compared = 'model'

        return render(request, "LSVC/analyze.html", {'data': data, 'count': count, 'compared': compared})
    return render(request, "LSVC/products.html", {'data': product_brand_form})
```

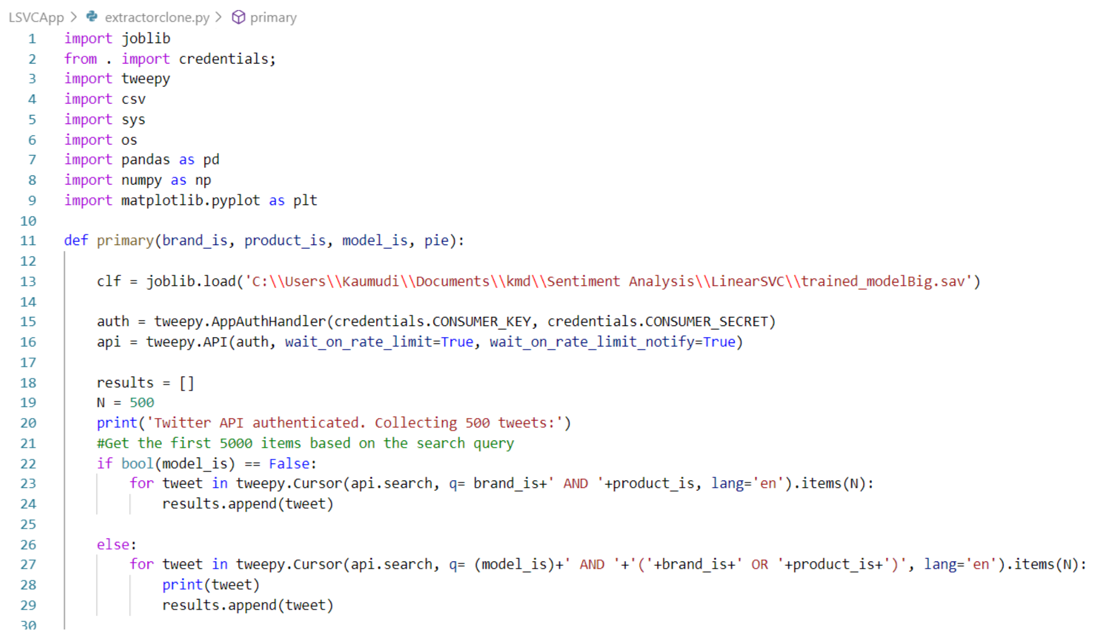
Fig 3.7 Rendering the Analyze HTML Page

Similar functions i.e., ‘brand’ and ‘brandanalyze’ take care of processing the sentiment and handling the web pages pertaining to the best choice of given brands for a particular product.

3.2.4. Extraction and Analysis module:

The extraction and analysis module has only one method called ‘primary’. This method extracts the tweets from the Twitter Search API, then preprocesses and analyzes the sentiments of the tweets and finally creates a graphical representation of the statistics of analysis which are displayed as the result to the user.

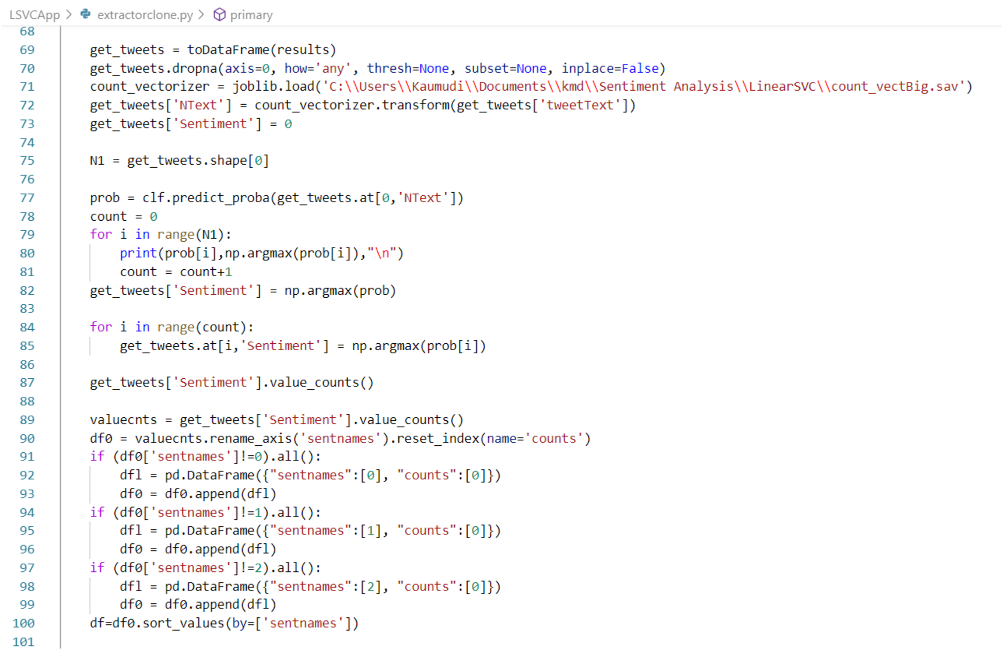
The figure below shows part of the code of the primary method which connects to the Twitter Search API authentically using the keys from the Twitter API credentials file. It retrieves a maximum of 500 tweets for every option given by the user.


LSVCApp > extractorclone.py > primary

```python
import joblib
from . import credentials;
import tweepy
import csv
import sys
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

def primary(brand_is, product_is, model_is, pie):

    clf = joblib.load('C:\\Users\\Kaumudi\\Documents\\kmd\\Sentiment Analysis\\LinearSVC\\trained_modelBig.sav')

    auth = tweepy.AppAuthHandler(credentials.CONSUMER_KEY, credentials.CONSUMER_SECRET)
    api = tweepy.API(auth, wait_on_rate_limit=True, wait_on_rate_limit_notify=True)

    results = []
    N = 500
    print('Twitter API authenticated. Collecting 500 tweets:')
    #Get the first 5000 items based on the search query
    if bool(model_is) == False:
        for tweet in tweepy.Cursor(api.search, q= brand_is+' AND '+product_is, lang='en').items(N):
            results.append(tweet)

    else:
        for tweet in tweepy.Cursor(api.search, q= (model_is)+' AND '+'('+brand_is+' OR '+product_is+')', lang='en').items(N):
            print(tweet)
            results.append(tweet)

```

Fig 3.8 Extraction of Relevant Tweets

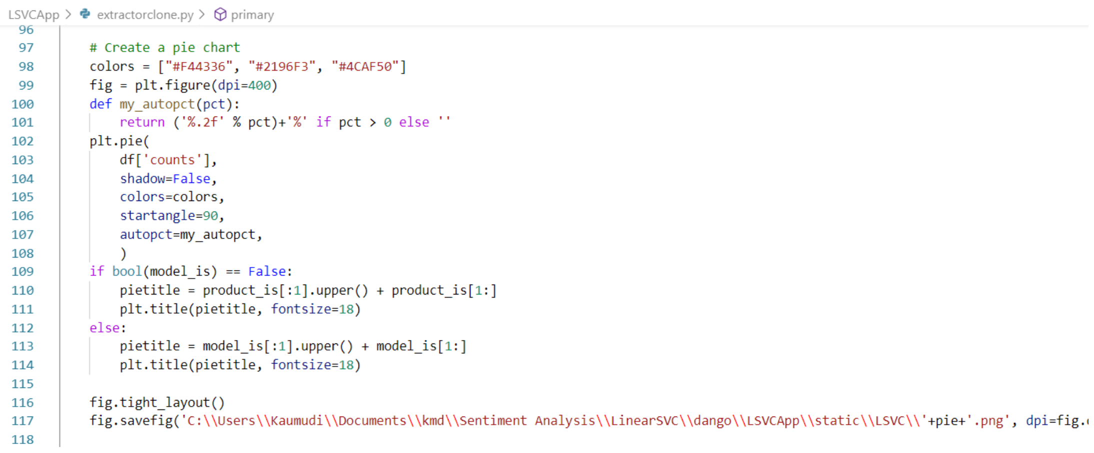
Once the tweets are extracted, they are stored in a pandas dataframe. The figure below shows the part of primary method which transforms the tweets into vectorized data. Then one vs rest classifier is applied on all transformed tweets. The result of this is a list of probabilities that the tweets may belong to either of the three classes under consideration, i.e., positive, negative and neutral.


LSVCApp > extractorclone.py > primary

```python

    get_tweets = toDataFrame(results)
    get_tweets.dropna(axis=0, how='any', thresh=None, subset=None, inplace=False)
    count_vectorizer = joblib.load('C:\\Users\\Kaumudi\\Documents\\kmd\\Sentiment Analysis\\LinearSVC\\count_vectBig.sav')
    get_tweets['NText'] = count_vectorizer.transform(get_tweets['tweetText'])
    get_tweets['Sentiment'] = 0

    N1 = get_tweets.shape[0]

    prob = clf.predict_proba(get_tweets.at[0,'NText'])
    count = 0
    for i in range(N1):
        print(prob[i],np.argmax(prob[i]),"\n")
        count = count+1
    get_tweets['Sentiment'] = np.argmax(prob)

    for i in range(count):
        get_tweets.at[i,'Sentiment'] = np.argmax(prob[i])

    get_tweets['Sentiment'].value_counts()

    valuecnts = get_tweets['Sentiment'].value_counts()
    df0 = valuecnts.rename_axis('sentnames').reset_index(name='counts')
    if (df0['sentnames']!=0).all():
        dfl = pd.DataFrame({"sentnames":[0], "counts":[0]})
        df0 = df0.append(dfl)
    if (df0['sentnames']!=1).all():
        dfl = pd.DataFrame({"sentnames":[1], "counts":[0]})
        df0 = df0.append(dfl)
    if (df0['sentnames']!=2).all():
        dfl = pd.DataFrame({"sentnames":[2], "counts":[0]})
        df0 = df0.append(dfl)
    df=df0.sort_values(by=['sentnames'])

```

Fig 3.9 Preprocessing and Sentiment Analysis of Extracted Tweets

Once these probabilities are obtained, the tweets is assigned the class with the maximum probability and this class is stored in a new column of the dataframe. Then, value counts of the sentiment column are obtained and stored in a new dataframe.


LSVCApp > extractorclone.py > primary

```
    # Create a pie chart
    colors = ["#F44336", "#2196F3", "#4CAF50"]
    fig = plt.figure(dpi=400)
    def my_autopct(pct):
        return ('%.2f' % pct)+'%' if pct > 0 else ''
    plt.pie(
        df['counts'],
        shadow=False,
        colors=colors,
        startangle=90,
        autopct=my_autopct,
        )
    if bool(model_is) == False:
        pietitle = product_is[:1].upper() + product_is[1:]
        plt.title(pietitle, fontsize=18)
    else:
        pietitle = model_is[:1].upper() + model_is[1:]
        plt.title(pietitle, fontsize=18)

    fig.tight_layout()
    fig.savefig('C:\\Users\\Kaumudi\\Documents\\kmd\\Sentiment Analysis\\LinearSVC\\dango\\LSVCApp\\static\\LSVC\\'+pie+'.png', dpi=fig.
```

Fig 3.10 Graphical Representation of the Results

The above figure shows the next step of the primary method. It creates a pie chart for every option given by the user with the statistics of analysis obtained above. The pie charts are saved and later rendered in HTML format to the user.

3.3. Support Vector Machine Algorithm:

Support vector machines [1] are supervised models with associated learning algorithms that analyze data used for classification and regression analysis. It makes use of the concept of decision planes that define decision boundaries.

It was originally designed for binary classification. It can also be extended to multi-class by combining multiple SVM's. It gives an excellent result for text categorization tasks such as sentiment analysis. SVM performs classification by finding an optimal hyper-plane that separates two classes. The optimal hyper-plane has maximum margin. The distance between nearest data point and hyper-plane is called as margin. The point that lies nearest to hyper-plane is called support vector.

Here the data is divided into positive, negative and neutral. Multinomial classification is the problem of classifying instances into one of three or more classes.

One versus rest classifier:

One-vs-rest strategy involves training a single classifier per class, with the samples of that class as positive samples and all other samples as negative.

Here, Positive data points are compared with negative and neutral, Negative data points are compared with positive and neutral, Neutral data points are compared with positive and negative.

Linear Kernel:

Linear kernel is used when the data is linearly separable, that is, it can be separated using a single line. It is one of the most common kernels to be used. It is mostly used when there are a large number of features in a particular data set.

Advantages:

1. Training a SVM with a linear kernel is faster than with any other kernel.

2. When training a SVM with a linear kernel, only the optimization of the **C** Regularization parameter is required.

C Parameter:

The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyper-plane if the hyper-plane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyper-plane, even if that hyper-plane misclassifies more points. For very tiny values of C, you should get misclassified examples, often even if you're training data that is linearly separable.

Gamma Parameter:

The gamma parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.

4. IMPLEMENTATION

4.1. UML Diagrams:

4.1.1. Flow Chart:

Flow chart is basically used to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Fig.4.1. depict the flow chart of the Sentiment analyzer.

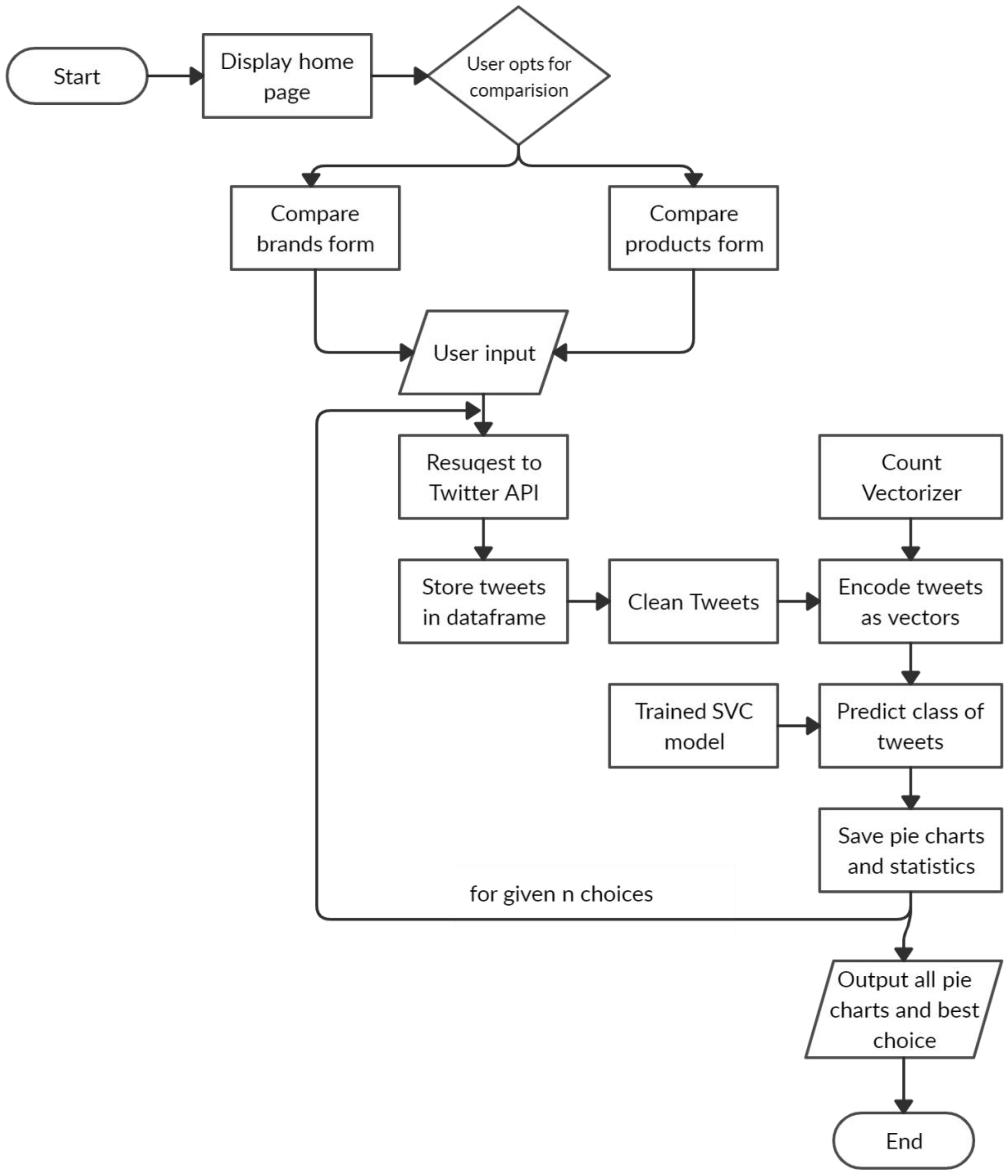


Fig 4.1. Sentiment Analyzer Flow chart

4.1.2. Data Flow Diagram:

A data flow diagram is a way of representing a flow of a data of a process or a system. The Data flow diagram (DFD) also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Fig 4.2. illustrate the DFD for Sentiment analyzer.

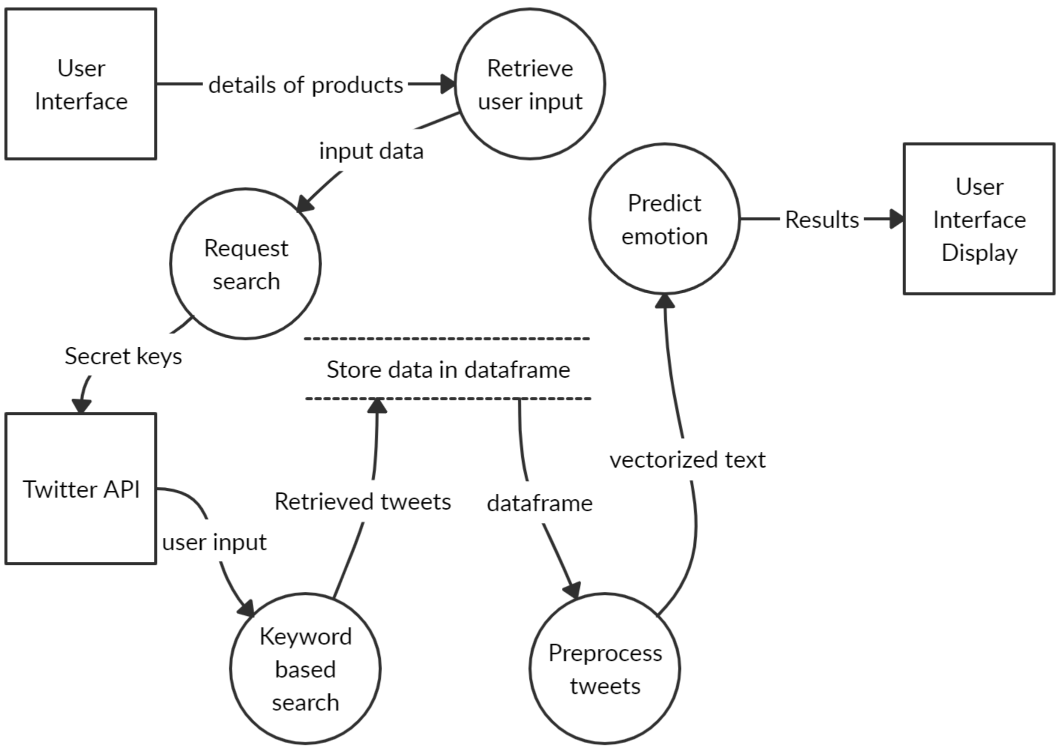


Fig 4.2. Data Flow Diagram

4.1.3. Use Case Diagram:

The use case diagram allows a designer to graphically show these use cases and the actors that use them. An actor is a role that a user plays in the system. It is important to distinguish between a user and an actor (better thought of as a role). A user of the system may play several different roles through the course of his/her or its jobs (since an actor may be another system). Examples of actors are salesperson, manager, support person and web store system. It is possible that the same person may be a sales person and also provide support. When creating a use case model individuals are not concerned, only the roles that they play are concerned. Fig 4.3 illustrates the use case diagram for the entire project.

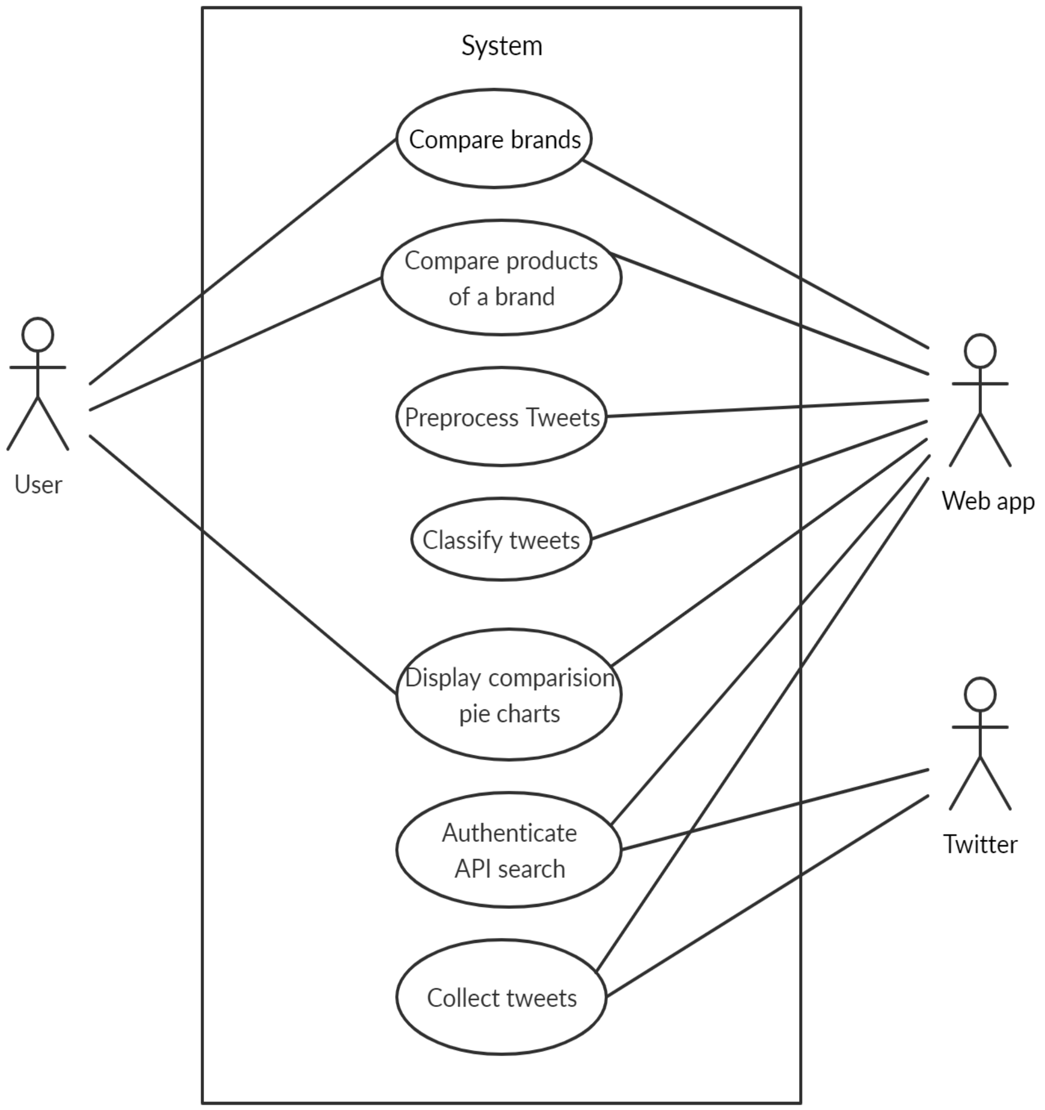


Fig 4.3. Use Case Diagram

4.1.4. Sequence Diagram:

Sequence diagrams emphasize the order in which things happen, while collaboration diagrams give more flexibility in their layout. Sequence diagram for Sentiment analyzer is shown in Fig 4.4.

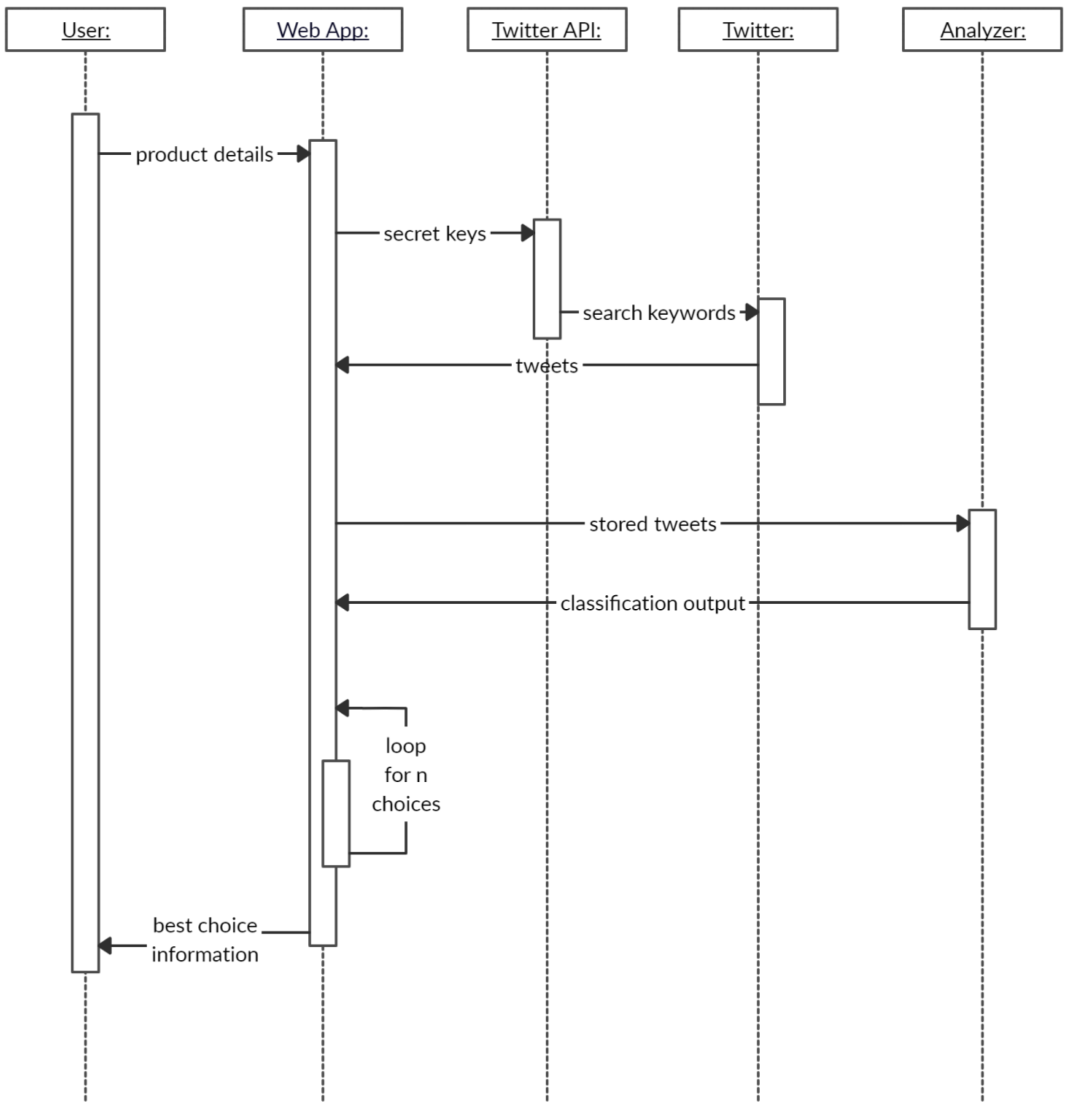


Fig 4.4. Sequence diagram

4.2. Technologies Used:

4.2.1. Machine Learning:

Machine learning for natural language processing [4] and text analytics involves using machine learning algorithms to understand the meaning of text documents.

These documents can be just about anything that contains text: Social media comments, online reviews.

The role of machine learning in Natural Language Processing (NLP) [2] and text analytics is to improve, accelerate and automate the underlying text analytics functions and NLP features that turn unstructured text into usable data and insights.

Supervised Machine Learning for Natural Language Processing and Text Analytics:

In supervised machine learning, a batch of text documents are tagged or annotated with examples of what the machine should look for and how it should interpret that aspect. These documents are used to train a statistical model.

The most popular supervised NLP machine learning algorithms are

- Support Vector Machines
- Bayesian Networks
- Maximum Entropy
- Neural Networks/Deep Learning

Tokenization:

Tokenization [5] involves breaking a text document into pieces that a machine can understand, such as words.

Names Entity Recognition:

At their simplest, named entities are people, places, and things (products) mentioned in a text document. Unfortunately, entities can also be hash tags, emails, mailing addresses, phone numbers, and Twitter handles.

4.2.2. Python:

Python is an interpreted language, high-level language, general purpose programming language. Sentiment analysis commonly uses python technology. Sentiment analysis can be done by rolling out the application from scratch, or maybe by using one of the many excellent open source libraries, such as scikit-learn.

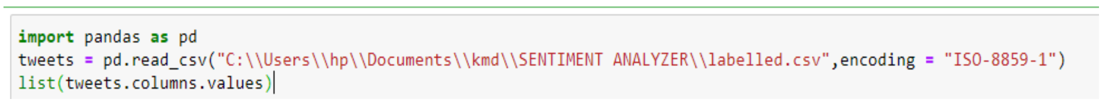
Installation:

Tweepy: tweepy [14] is the python client for the official Twitter API [3].

Install it using following pip command:

Pip install tweepy

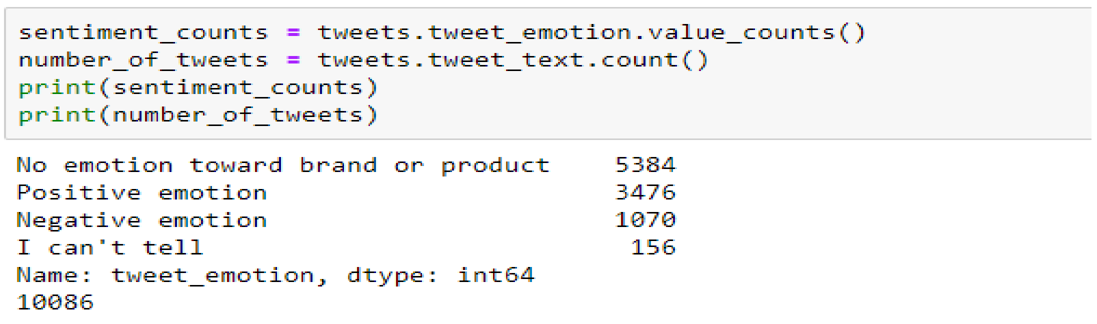
# Importing pandas



```
import pandas as pd
tweets = pd.read_csv("C:\\Users\\hp\\Documents\\kmd\\SENTIMENT ANALYZER\\labelled.csv",encoding = "ISO-8859-1")
list(tweets.columns.values)
```

Pandas DataFrames is used to import and inspect a variety of datasets. Pandas is one of those packages and makes importing and analyzing data much easier. By, default the rows not satisfying the condition are filled with NaN value.

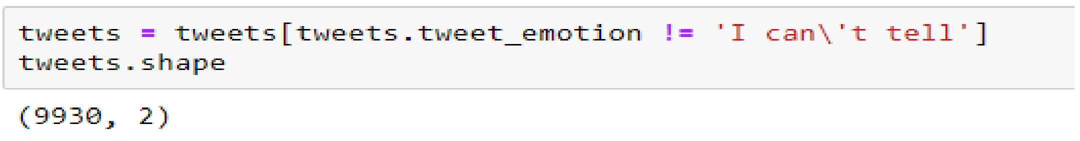
# Calculating total number of tweets and sentiment count.



```
sentiment_counts = tweets.tweet_emotion.value_counts()
number_of_tweets = tweets.tweet_text.count()
print(sentiment_counts)
print(number_of_tweets)
```

```
No emotion toward brand or product    5384
Positive emotion                      3476
Negative emotion                      1070
I can't tell                           156
Name: tweet_emotion, dtype: int64
10086
```

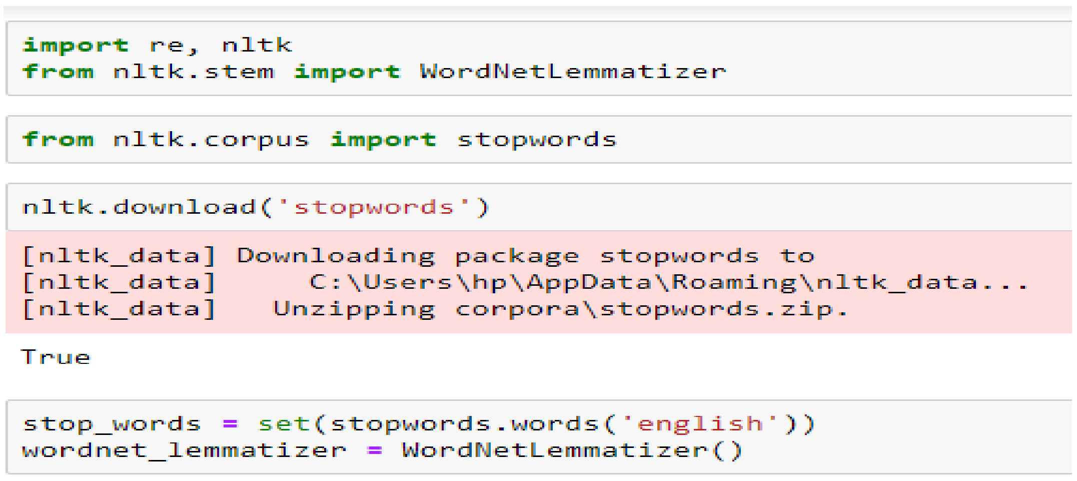
# Removing unnecessary data



```
tweets = tweets[tweets.tweet_emotion != 'I can\'t tell']
tweets.shape
```

```
(9930, 2)
```

# Importing WordNetLemmatizer and stopwords from nltk



```
import re, nltk
from nltk.stem import WordNetLemmatizer
```

```
from nltk.corpus import stopwords
```

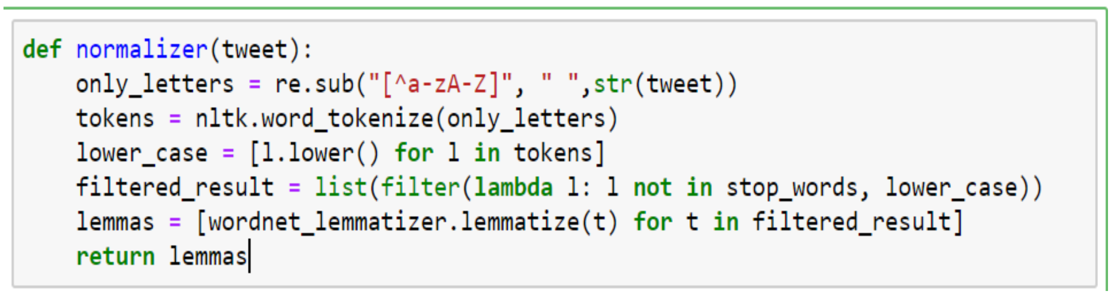
```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\hp\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

```
True
```

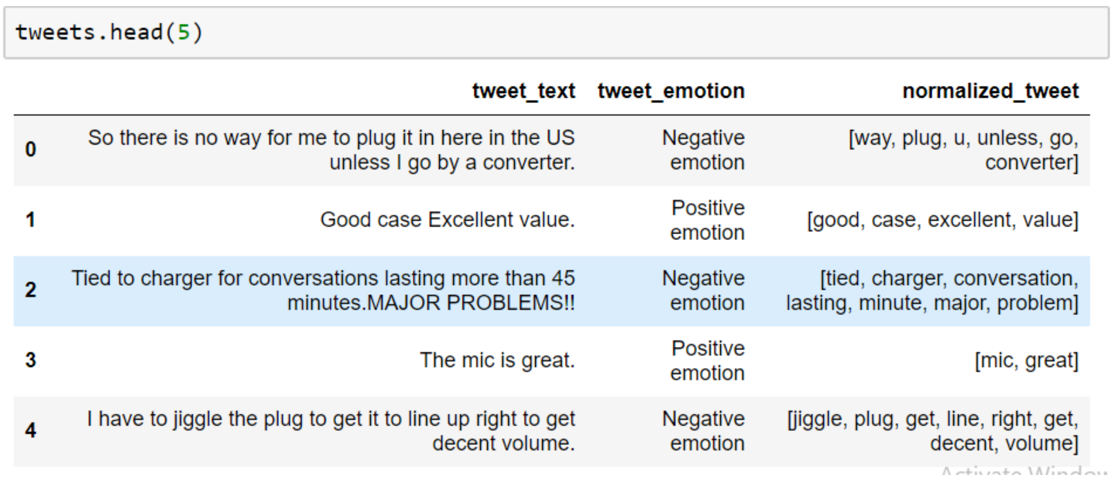
```
stop_words = set(stopwords.words('english'))
wordnet_lemmatizer = WordNetLemmatizer()
```

# Normalizing the tweets



```
def normalizer(tweet):
    only_letters = re.sub("[^a-zA-Z]", " ",str(tweet))
    tokens = nltk.word_tokenize(only_letters)
    lower_case = [l.lower() for l in tokens]
    filtered_result = list(filter(lambda l: l not in stop_words, lower_case))
    lemmas = [wordnet_lemmatizer.lemmatize(t) for t in filtered_result]
    return lemmas
```

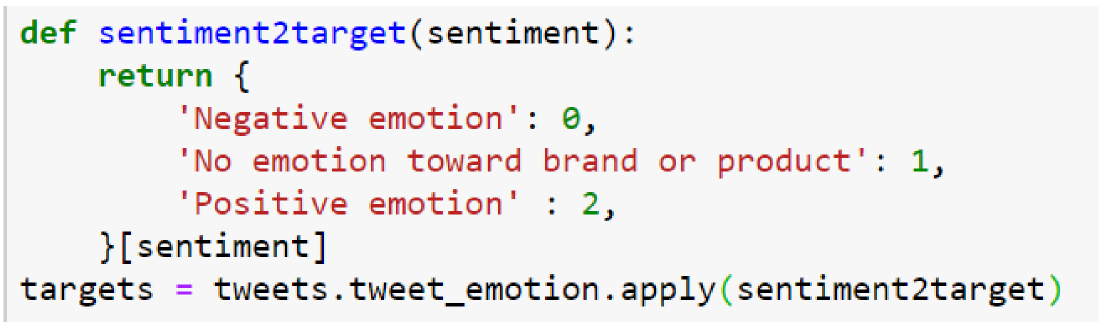
# Displaying the collected tweets



```
tweets.head(5)
```

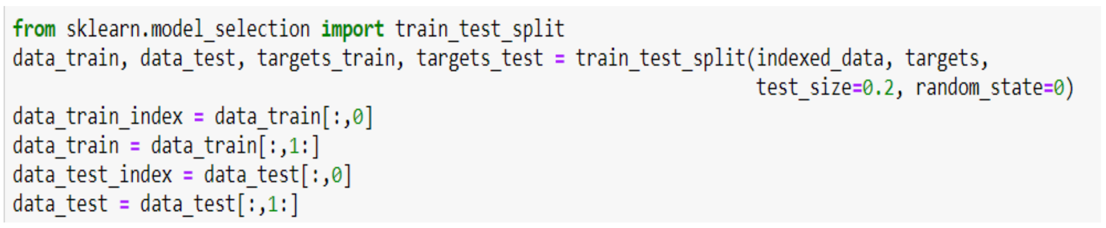
	tweet_text	tweet_emotion	normalized_tweet
0	So there is no way for me to plug it in here in the US unless I go by a converter.	Negative emotion	[way, plug, u, unless, go, converter]
1	Good case Excellent value.	Positive emotion	[good, case, excellent, value]
2	Tied to charger for conversations lasting more than 45 minutes.MAJOR PROBLEMS!!	Negative emotion	[tied, charger, conversation, lasting, minute, major, problem]
3	The mic is great.	Positive emotion	[mic, great]
4	I have to jiggle the plug to get it to line up right to get decent volume.	Negative emotion	[jiggle, plug, get, line, right, get, decent, volume]

# Giving polarity score of the sentiments



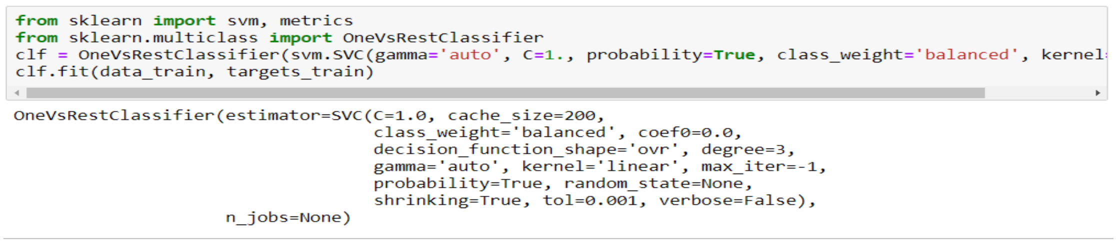
```
def sentiment2target(sentiment):
    return {
        'Negative emotion': 0,
        'No emotion toward brand or product': 1,
        'Positive emotion' : 2,
    }[sentiment]
targets = tweets.tweet_emotion.apply(sentiment2target)
```

#Training and testing the data



```
from sklearn.model_selection import train_test_split
data_train, data_test, targets_train, targets_test = train_test_split(indexed_data, targets,
                                                                       test_size=0.2, random_state=0)
data_train_index = data_train[:,0]
data_train = data_train[:,1:]
data_test_index = data_test[:,0]
data_test = data_test[:,1:]
```

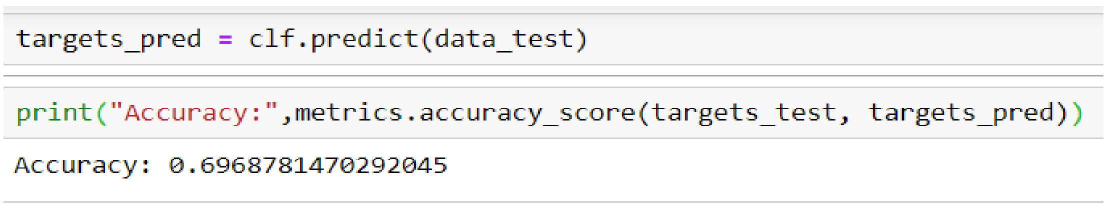
# Applying one versus rest classifier



```
from sklearn import svm, metrics
from sklearn.multiclass import OneVsRestClassifier
clf = OneVsRestClassifier(svm.SVC(gamma='auto', C=1., probability=True, class_weight='balanced', kernel
clf.fit(data_train, targets_train)
```

```
OneVsRestClassifier(estimator=SVC(C=1.0, cache_size=200,
                                  class_weight='balanced', coef0=0.0,
                                  decision_function_shape='ovr', degree=3,
                                  gamma='auto', kernel='linear', max_iter=-1,
                                  probability=True, random_state=None,
                                  shrinking=True, tol=0.001, verbose=False),
                    n_jobs=None)
```

# Prediction and Accuracy



```
targets_pred = clf.predict(data_test)
```

```
print("Accuracy:",metrics.accuracy_score(targets_test, targets_pred))
```

```
Accuracy: 0.6968781470292045
```

4.2.3. Django:

Django is an open-source python web framework used for rapid development, pragmatic, maintainable, clean design, and secures websites. A web application framework is a toolkit of all components need for application development. The main goal of the Django framework is to allow developers to focus on components of the application that are new instead of spending time on already developed components. It enables users to focus on developing components needed for this application.

Now for the frontend to be able to access these functions, make the functions as API's. This can be done easily using Django Framework.

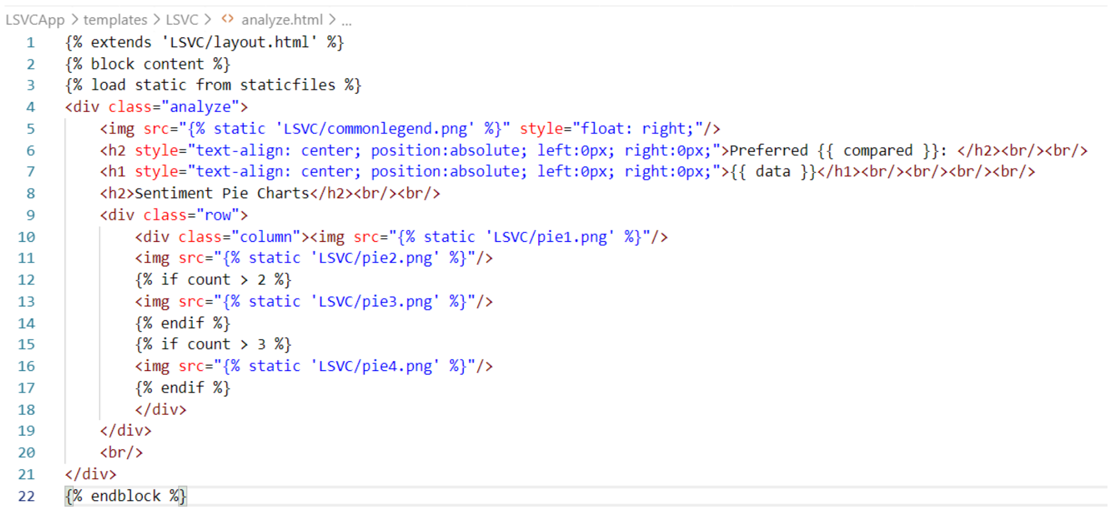
Simply install it using pip install djangorestframework and add @api_view[“GET”] before every function. Do not forget to make the additions to the settings.py file. Run the server using python.manage.py runserver.

Uses of Django:

- Django is time-tested
- Application Development
- Operating System Dependent
- Excellent Documentation for real-world application
- Scalable and Reliable
- Community support
- Robustness

4.2.4. Jinja:

Jinja is a template engine for web templating for Python. It can be used to generate any mark-up or source code. In Sentiment Analyzer, it is used for creating HTML from Python code. Jinja helps in embedding logic into the HTML files. It compiles Python code just in time. It also prevents cross-site scripting attacks.


LSVCApp > templates > LSVC > analyze.html > ...

```
{% extends 'LSVC/layout.html' %}
{% block content %}
{% load static from staticfiles %}
<div class="analyze">
    <img src="{% static 'LSVC/commonlegend.png' %}" style="float: right;"/>
    <h2 style="text-align: center; position:absolute; left:0px; right:0px;">Preferred {{ compared }}: </h2><br/><br/>
    <h1 style="text-align: center; position:absolute; left:0px; right:0px;">{{ data }}</h1><br/><br/><br/><br/>
    <h2>Sentiment Pie Charts</h2><br/><br/>
    <div class="row">
        <div class="column"><img src="{% static 'LSVC/pie1.png' %}"/>
        <img src="{% static 'LSVC/pie2.png' %}"/>
        {% if count > 2 %}
        <img src="{% static 'LSVC/pie3.png' %}"/>
        {% endif %}
        {% if count > 3 %}
        <img src="{% static 'LSVC/pie4.png' %}"/>
        {% endif %}
        </div>
    </div>
    <br/>
</div>
{% endblock %}
```

Fig 4.5. Jinja template in analyze.html file

In the above figure the output page of the web app is shown. Here, text in between '{{' and '}}' corresponds to Python variables and that in between '{%' and '%}' corresponds to the Python logic.

5. RESULTS

5.1 Discussion on Results:

Support Vector Machine algorithm is implemented in Python with NLTK. About 2000 tweets of people about different products and brands are used. Each tweet is labeled with polarity 0 or 1 or 2. According to the polarity tweet is classified either to Negative class or Neutral class or Positive class. Two types of Support Vector Machine are applied for classification of Twitter data. Eighty percent data are used for training and twenty percent data are used for testing.

The performance of the proposed algorithm is analyzed using four parameters i.e. Accuracy, Recall, and Precision.

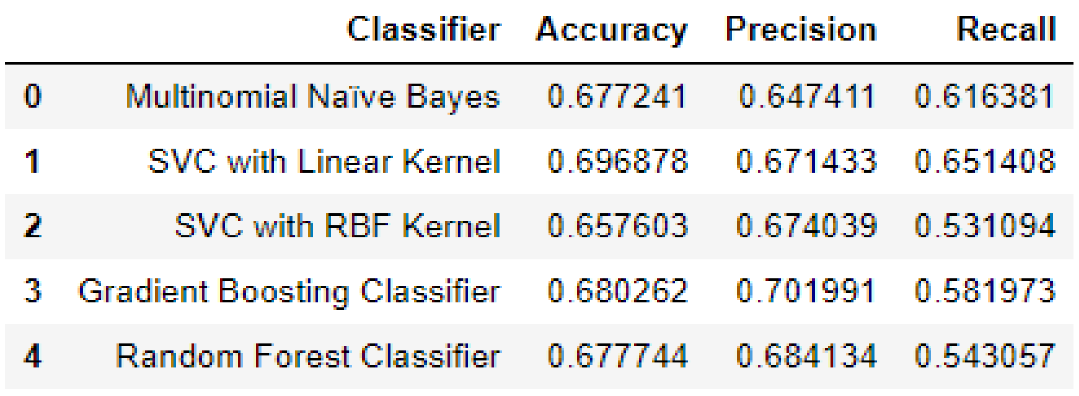
Accuracy = (TP+TN) / (TP+FP+TN+FN)

Recall = TP / (TP+FN)

Precision = TP / (TP+FP)

Here, True positive (TP) defines the number of positive tweets that are correctly classified, as positive, whereas false positive (FP) is the number of negative tweets that are incorrectly classified as positive. True negative (TN) is the number of negative instances that are correctly classified as negative and false negative (FN) is the number of positive tuples that are incorrectly classified as negative tweets.

The type of SVM applied for classification of tweets is Linear SVM. It has been analyzed that the Linear SVM gives more accuracy compared to SVM with Radial Basis (RBF) kernel, Multinomial Naïve Bayes, Gradient Boosting Classifier and Random Forest Classifier. The proposed approach is compared to other sentiment analysis techniques in the table below.



	Classifier	Accuracy	Precision	Recall
0	Multinomial Naïve Bayes	0.677241	0.647411	0.616381
1	SVC with Linear Kernel	0.696878	0.671433	0.651408
2	SVC with RBF Kernel	0.657603	0.674039	0.531094
3	Gradient Boosting Classifier	0.680262	0.701991	0.581973
4	Random Forest Classifier	0.677744	0.684134	0.543057

Fig 5.1 Comparison between algorithms

Above table shows the accuracy, precision and recall for different algorithms. It can be concluded that Support Vector Classifier with Linear Kernel is much effective in sentiment classification as it shows accuracy of 0.696878, precision of 0.671433, and recall of 0.651408 which is large when compared to other algorithms. Though, Support Vector Classifier with RBF kernel has the similar precision value with Linear kernel, it is not widely used because Linear kernel is simple, consumes less time and easy to implement.

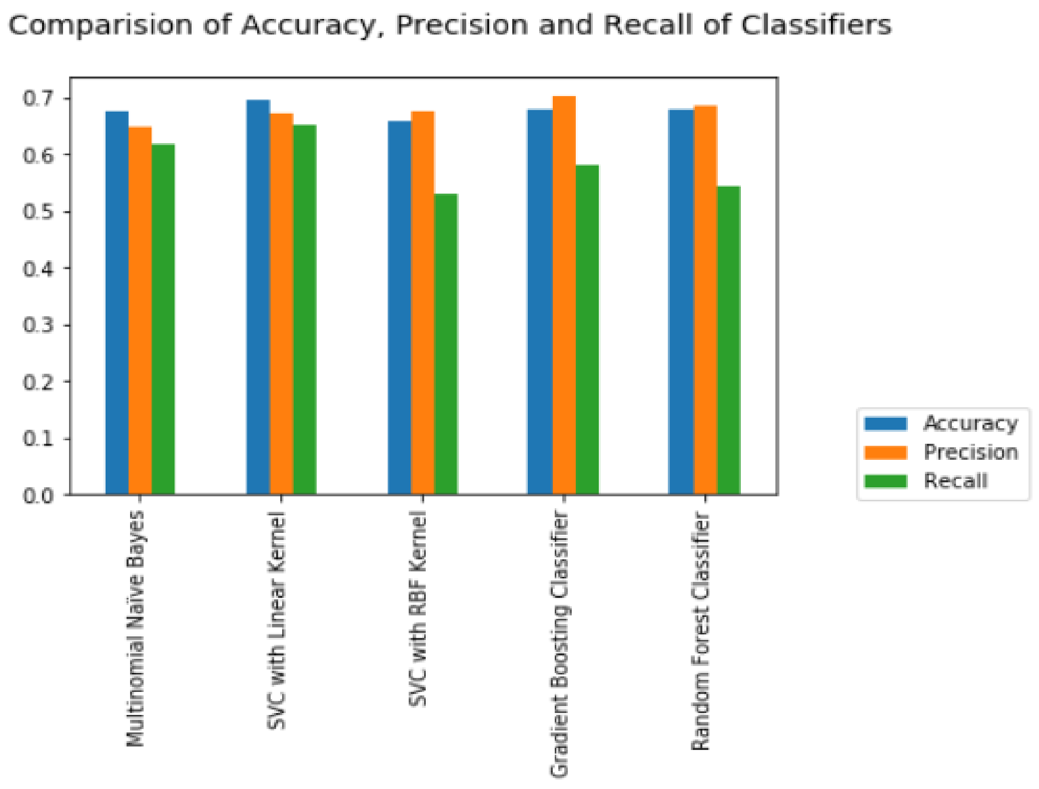
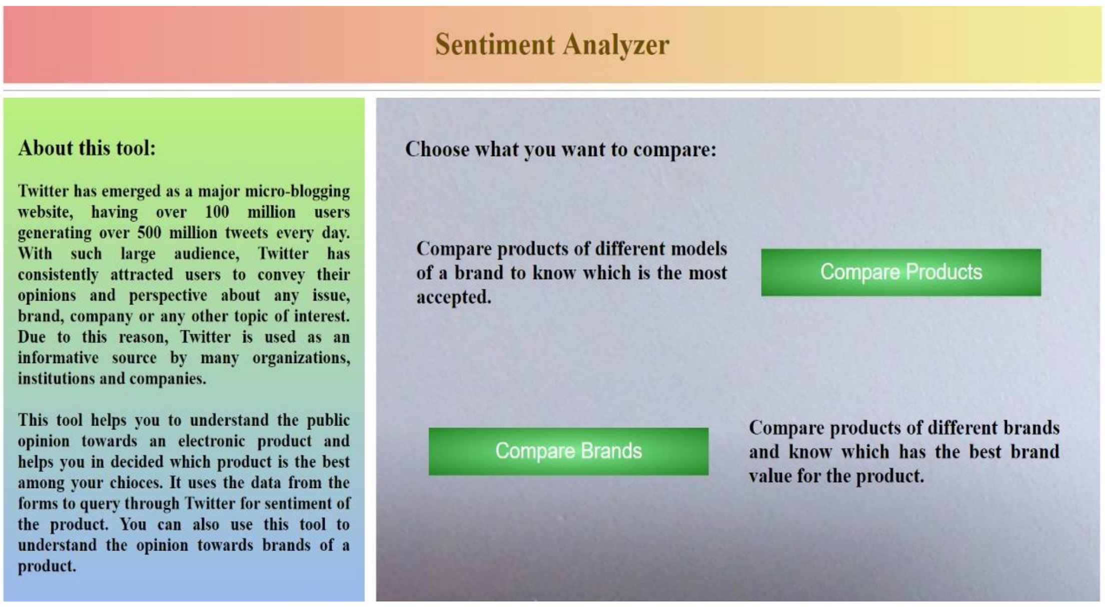


Fig 5.2 Bar Graph showing comparison between algorithms

The graph above is a pictorial representation of the values of the accuracies, precisions and recalls of different algorithms that have been run on the product reviews dataset for sentiment classification. From these results, the best working algorithm, i.e., SVC with linear kernel is chosen.

5.2 Screenshots:



Sentiment Analyzer

About this tool:

Twitter has emerged as a major micro-blogging website, having over 100 million users generating over 500 million tweets every day. With such large audience, Twitter has consistently attracted users to convey their opinions and perspective about any issue, brand, company or any other topic of interest. Due to this reason, Twitter is used as an informative source by many organizations, institutions and companies.

This tool helps you to understand the public opinion towards an electronic product and helps you in decided which product is the best among your chioces. It uses the data from the forms to query through Twitter for sentiment of the product. You can also use this tool to understand the opinion towards brands of a product.

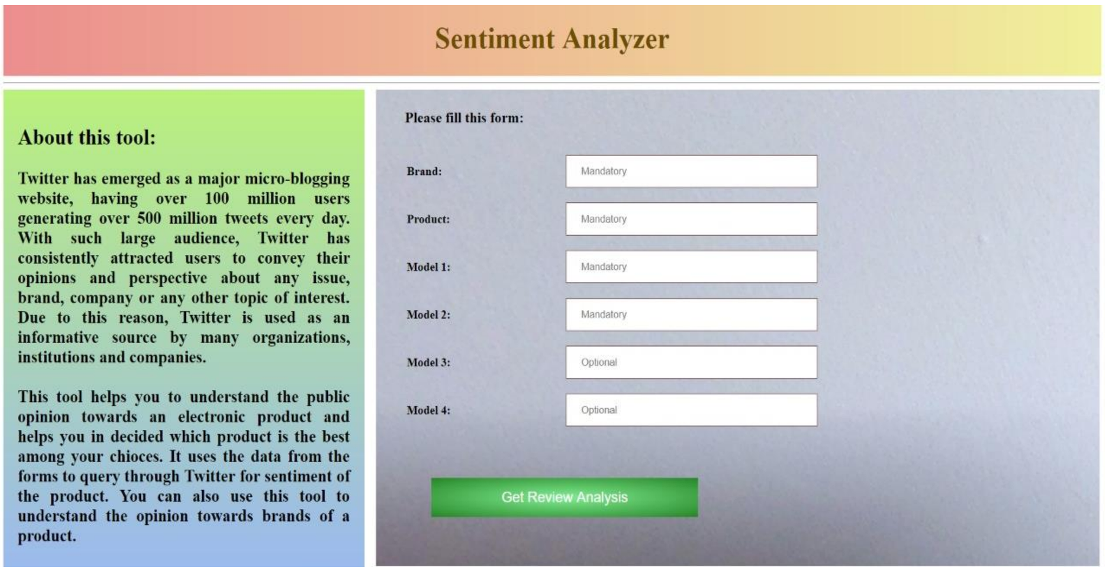
Choose what you want to compare:

Compare products of different models of a brand to know which is the most accepted.

Compare products of different brands and know which has the best brand value for the product.

Fig 5.3 Home Page

Home page contains two options. First option compares the different products, and the second option compares the different brands. Upon clicking the "Compare Products" option, the user would be redirected to the window shown in fig 5.3



Sentiment Analyzer

About this tool:

Twitter has emerged as a major micro-blogging website, having over 100 million users generating over 500 million tweets every day. With such large audience, Twitter has consistently attracted users to convey their opinions and perspective about any issue, brand, company or any other topic of interest. Due to this reason, Twitter is used as an informative source by many organizations, institutions and companies.

This tool helps you to understand the public opinion towards an electronic product and helps you in decided which product is the best among your chioces. It uses the data from the forms to query through Twitter for sentiment of the product. You can also use this tool to understand the opinion towards brands of a product.

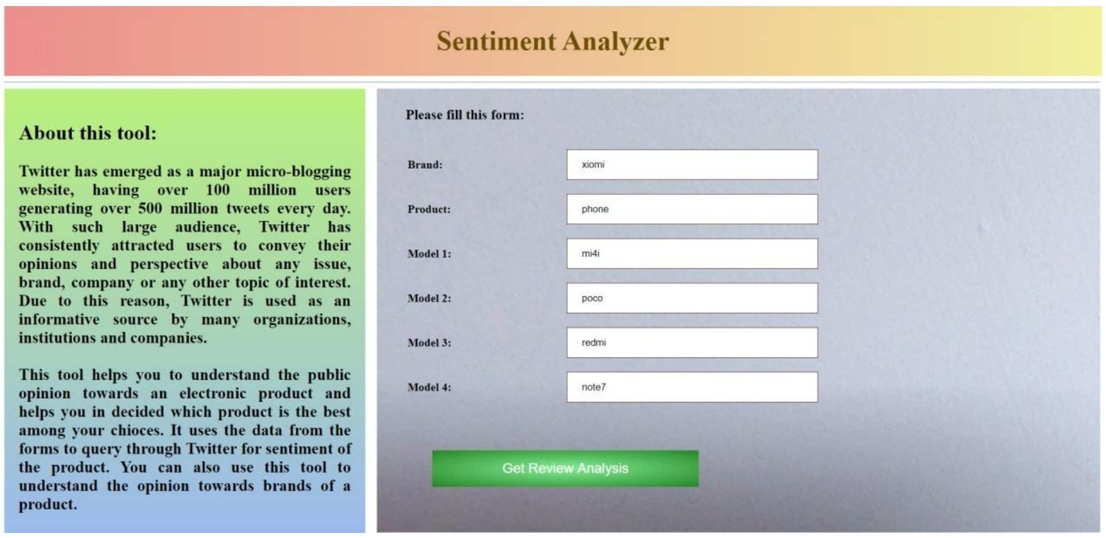
Please fill this form:

Brand:	Mandatory
Product:	Mandatory
Model 1:	Mandatory
Model 2:	Mandatory
Model 3:	Optional
Model 4:	Optional

Fig 5.4 Product Comparison screen

The above fig 5.4 shows the interface for 'Product comparison' where user is allowed to enter the text. After the text is entered click on 'Get Review Analysis' button to get required result.

To get the desired result, Brand name, Product name with at least two models should be given as input. Model 3 and Model 4 are optional. Brand and Product names should be of Electronic devices.



Sentiment Analyzer

About this tool:

Twitter has emerged as a major micro-blogging website, having over 100 million users generating over 500 million tweets every day. With such large audience, Twitter has consistently attracted users to convey their opinions and perspective about any issue, brand, company or any other topic of interest. Due to this reason, Twitter is used as an informative source by many organizations, institutions and companies.

This tool helps you to understand the public opinion towards an electronic product and helps you in decided which product is the best among your chioces. It uses the data from the forms to query through Twitter for sentiment of the product. You can also use this tool to understand the opinion towards brands of a product.

Please fill this form:

Brand:	xiomi
Product:	phone
Model 1:	mi4i
Model 2:	poco
Model 3:	redmi
Model 4:	note7

Fig 5.5 Product Comparison form

Once the user enters the product name and various models as input, the interface looks like the Fig 5.5 with the form filled in. Now the user is ready to click on the button 'Get Review Analysis' and get the required result.

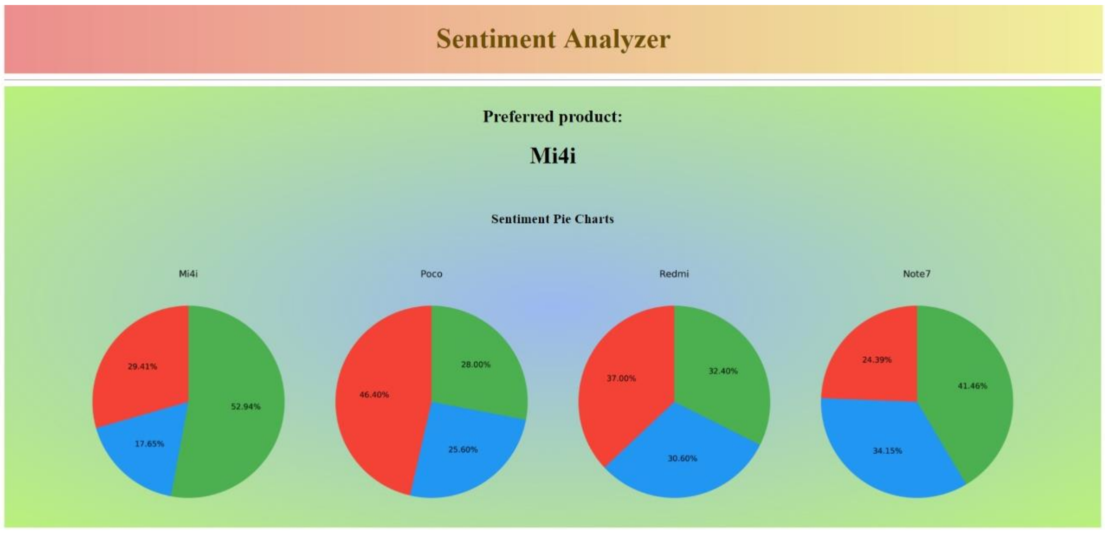
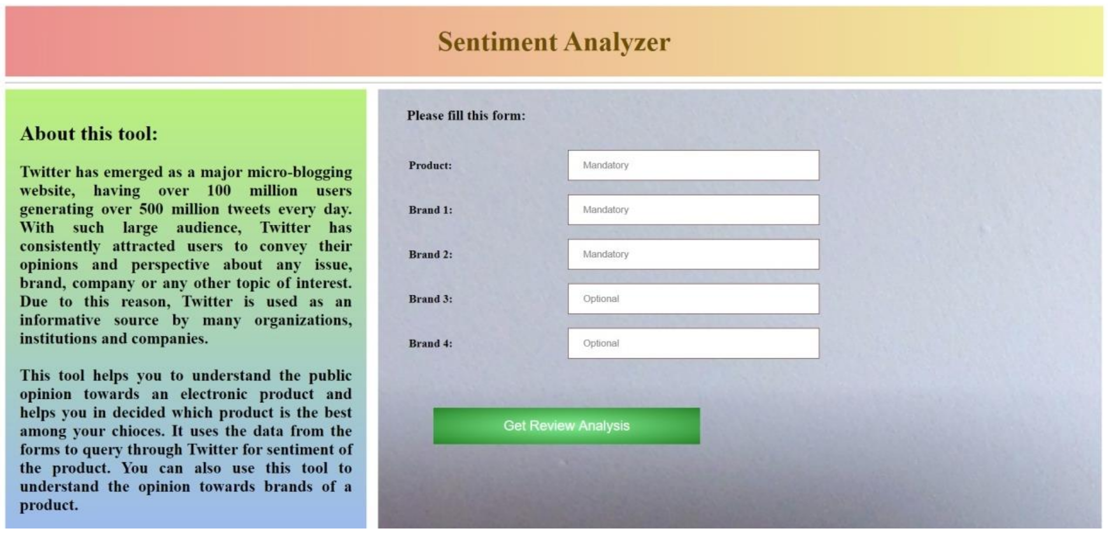


Fig 5.6 Output

After giving an input and clicking on the ‘Get Review Analysis’ button, the answer is generated in the form of ‘pie charts’ as in the Fig 5.6.

The color Green depicts the positive sentiment, Blue color depicts the neutral sentiment and the color Red depicts the Negative sentiment. Among the models, Mi4i, Poco, Redmi, and Note7, the preferred product is Mi4i as it shows positive sentiment with52.94%. Neutral sentiment with 17.65%. Negative sentiment with 29.41%. This result can be changed according to the time, because tweets are collected from Twitter API which is a real time API.



Sentiment Analyzer

About this tool:

Twitter has emerged as a major micro-blogging website, having over 100 million users generating over 500 million tweets every day. With such large audience, Twitter has consistently attracted users to convey their opinions and perspective about any issue, brand, company or any other topic of interest. Due to this reason, Twitter is used as an informative source by many organizations, institutions and companies.

This tool helps you to understand the public opinion towards an electronic product and helps you in decided which product is the best among your chioces. It uses the data from the forms to query through Twitter for sentiment of the product. You can also use this tool to understand the opinion towards brands of a product.

Please fill this form:

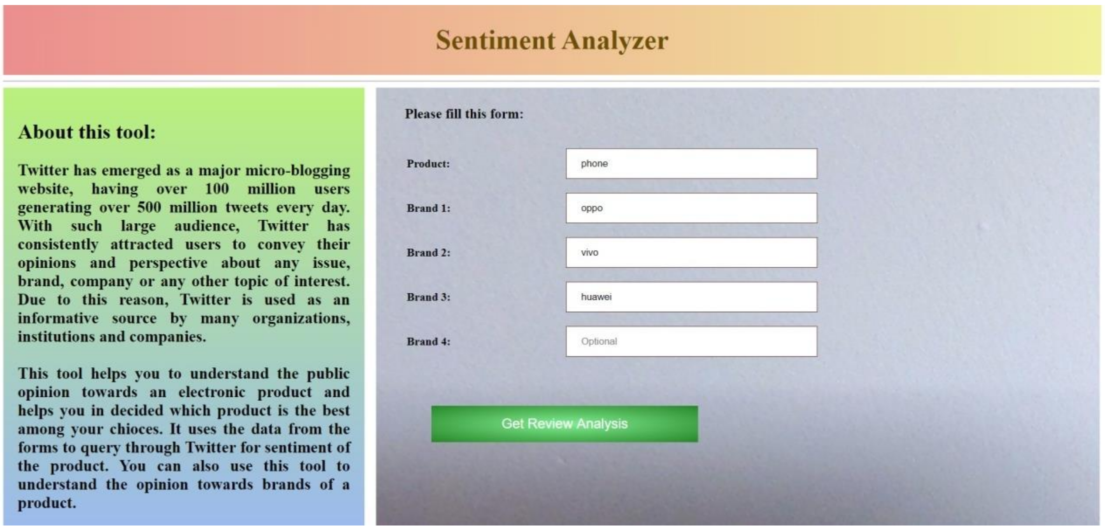
Product:	Mandatory
Brand 1:	Mandatory
Brand 2:	Mandatory
Brand 3:	Optional
Brand 4:	Optional

Fig 5.7 Brand Comparison Screen

Home page in fig 5.3 contains an option ‘Compare Brands’. Upon clicking this option, the user would be redirected to another window which looks like fig 5.7. Here the user enters the text and clicks on ‘Get Review Analysis’ button to obtain the desired result.

Product name with at least two brands should be given as input to compare and get the desired result. Brand 3 and Brand 4 are optional.

Once the user enters the product name and various brands as input, the interface looks like fig 5.8. Now the user is ready to click on the button ‘Get Review Analysis’ and get the required result.



Sentiment Analyzer

About this tool:

Twitter has emerged as a major micro-blogging website, having over 100 million users generating over 500 million tweets every day. With such large audience, Twitter has consistently attracted users to convey their opinions and perspective about any issue, brand, company or any other topic of interest. Due to this reason, Twitter is used as an informative source by many organizations, institutions and companies.

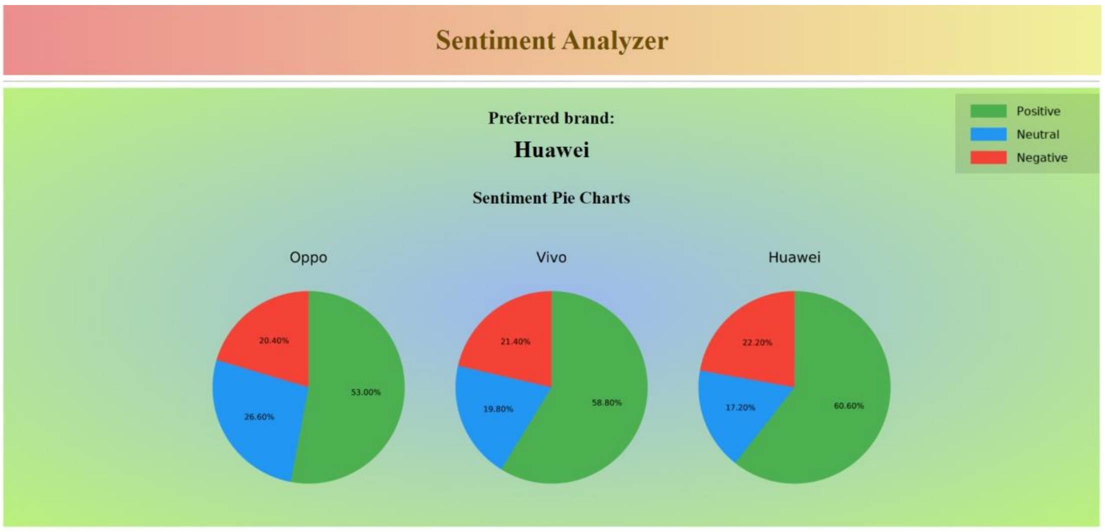
This tool helps you to understand the public opinion towards an electronic product and helps you in decided which product is the best among your chioces. It uses the data from the forms to query through Twitter for sentiment of the product. You can also use this tool to understand the opinion towards brands of a product.

Please fill this form:

Product:	phone
Brand 1:	oppo
Brand 2:	vivo
Brand 3:	huawei
Brand 4:	Optional

Fig 5.8 Brand Comparison Form

After giving an input and clicking on ‘Get Review Analysis’ button, the output is generated in the form of ‘pie charts’ as in the fig 5.9.



Sentiment Analyzer

Preferred brand:

Huawei

Sentiment Pie Charts

Fig 5.9 Output

Among the brands Oppo, Vivo and Huawei, the preferred brand is Huawei with 60.60% positive sentiment, 17.20% neutral sentiment and 32.29% negative sentiment. This result also changes according to the time due to twitter API.

6. CONCLUSION AND SCOPE FOR FUTURE ENHANCEMENTS

6.1 Conclusion:

Sentiment analysis is the process of categorizing a person’s opinion expressed in the form of text. The most recent and relevant tweets are extracted using Twitter Search API. Linear SVM is applied for the classification of sentiments of different electronic products. In addition to this feature selection is also applied here. The performance of SVM is analyzed on various measures, i.e. accuracy, precision, and recall. Labour cost is reduced using the tweepy package. Customer opinions are easily identified. It provides user friendly interface for customer experience and graphical representation of sentiment in the form of pie chart. It helps user to choose best brand and best model of Electronic products.

6.2 Future scope:

The future of sentiment analysis is going to continue to big deeper, far past the surface of the number of likes, comments and shares, and aim to reach, and truly understand, the significance of social-media interactions and what they tell us about the producers behind the screens.

Deeper, Broader Insights from Sentiment Analysis:

Sentiment analysis is getting better because social media is increasingly more emotive and expressive. Twitter introduced “Reactions”, which allows it users to not just ‘Like’, but attach an emoticon, whether it is a heart, a shocked face, angry face, etc. To leverage social media data that for sentiment analysis, this provides an entirely new layer of data that was not available before and to categorize these emoticons and predict correct sentiment.

Not Just For Electronic Products And Products:

Again, sentiment analysis is on the verge of breaking into new areas of application. In future, Sentiment analysis can be used in various other areas.

References:

1. Aamera Z. H. Khan, Dr. Mohammad Atique and Dr. V. M. Thakare, “Sentiment Analysis Using Support Vector Machine,” International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 4, p.p. 105-108, April 2015.
2. A. Nisha Jebaseeli and E. Kirubakaran, “A Survey on Sentiment Analysis of (Product) Reviews,” International Journal of Computer Applications, Volume 47– No.11, p.p. 36-39, June 2012.
3. Ankita Gupta, Jyotika Pruthi and Neha Sahu, “Sentiment Analysis of Tweets using Machine Learning Approach,” International Journal of Computer Science and Mobile Computing, Volume 6, Issue 4, p.p. 444-458, , April- 2017.
4. Dipak R. Kawade and Dr. Kavita S. Oza, “Sentiment Analysis: Machine Learning Approach,” International Journal of Engineering and Technology, Volume 9 No 3, p.p. 2183-2186, July 2017.
5. Dr. S. Vijayarani and Ms. R. Janani, “Text Mining: Open Source Tokenization Tools – An Analysis,” Advanced Computational Intelligence: An International Journal, Volume 3, No.1, p.p. 37-47, January 2016
6. Gangawane A. A. and H. B. Torvi, “Opinion Mining and Sentiment Analysis on Twitter,” International Journal of Innovative Research in Science, Engineering and Technology, Volume 6, Issue 7, p.p. 12989-12996, July 2017.
7. IJIRST – International Journal for Innovative Research in Science and Technology | Volume 3 | Issue 10 |March 2017 ISSN (online): 2349-6010
8. Kim S-M, Hovy E (2004) Determining the sentiment of opinions in Proceedings of the 20th international conference on Computational Linguistics, page 1367. Association for Computational Linguistics, Stroudsburg, PA, USA.
9. Liu B (2010) Sentiment analysis and subjectivity in Handbook of Natural Language Processing, second edition. Taylor and Francis Group, Boca. Liu B, Cheng J (2005) Opinion observer: Analyzing and Comparing opinions: Proceedings of the 14th International Conference on World Wide Web.

10. Pang B, Lee L (2004) A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, Stroudsburg, PA, USA.
11. Pang B, Lee L (2008) Opinion minimg and sentiment analysis. Found Trends Inf Retr2 (1-2): 1-135.
12. Muhammad Zubair Asghar, Aurangzeb Khan, Shakeel Ahmad and Fazal Masud Kundi, "A Review of Feature Extraction in Sentiment Analysis," Journal of Basic and Applied Scientific Research, ISSN 2090-4304, 4(3), p.p.181-186, 2014.
13. Django Framework: https://docs.djangoproject.com/en/2.1/
14. Twitter Search API: https://developer.twitter.com/en/docs/

Glossary

- **Machine Learning:** Scientific study of algorithms and statistical models that computer systems use to perform a specific task.
- **Natural Language Processing (NLP):** Ability for computers to understand human languages.
- **One-vs-rest classifier:** Strategy involves training a single classifier per class, with the samples of that class.
- **Sentiment Analysis:** Categorizing opinions expressed in a piece of text is positive, negative, or neutral.
- **Support Vector Machine (SVM):** Analyzes data for classification and regression analysis.
- **Twitter API:** Set of URL’s that helps to access many features of Twitter.