SENTIMENT ANALYSIS

1. Introduction

The aim of the project is to analyze people sentiments related to hot trending topics across multiple social platforms.

This program picks hottest trends from Google.com at that second of the day and find out corresponding tweets from Twitter. Using those tweets, it extracts people reactions corresponding to that topic as “Did People like it”, “retweeted it” or “what’s average sentiment (positive /negative) for that tweet”. Also uses matplotlib library to plot correlation between tweets and sentiments.

2. Requirements

Following is the list of modules required to run Twitter Sentiment Analysis.:-
urllib.request :-This module fetch data from the web using http request
BeautifulSoup: - This is a library that formats web data. Also has in built functions to extract particular html tags.
Text Blob: Is a Sentiment Analysis library
Tweepy : Is a twitter Api library
Matplotlib : This module plots the graphical representation of analysis

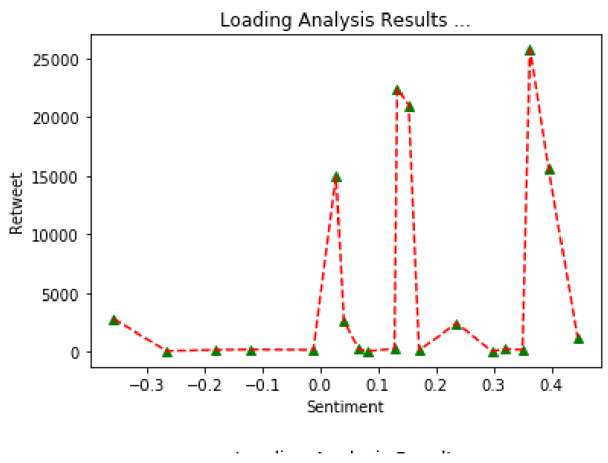
For Running Twitter Sentiment Analysis, we need to create Twitter account and create an application on twitter that will access twitter data. We can create application through this url https://apps.twitter.com/.
After creating Application, Twitter generates consumer key and access tokens, which are required for user authentication while calling twitter API to access tweets data from the program.

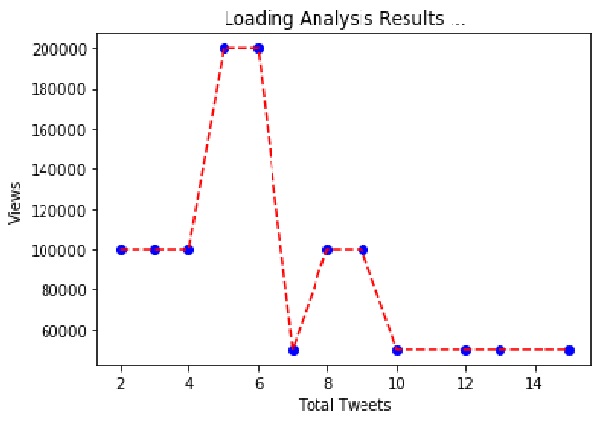
3. Description of the Python program

- This program starts by posting a urlib.request to fetch trending topics from web URL http://www.google.com/trends/hottrends/atom/feed and receive page data.

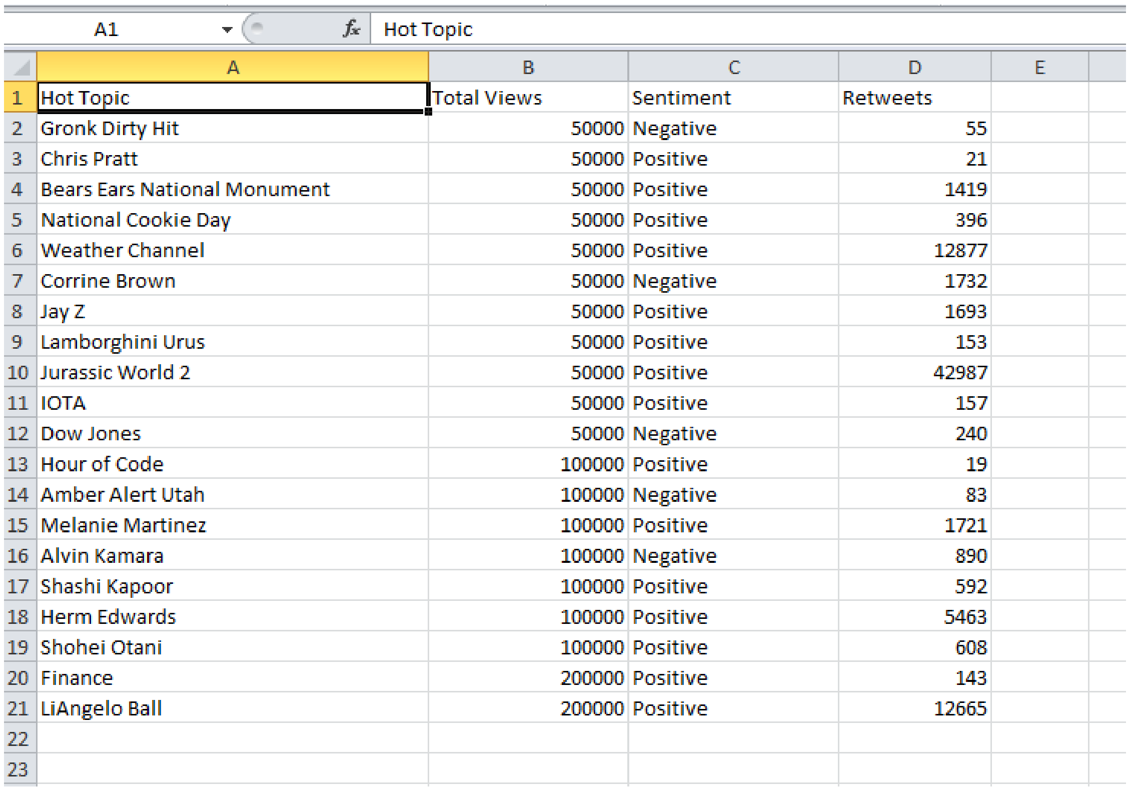
- After then extract hot topics and traffic information using Beautiful Soup library from the web data and save it in a dictionary data structure “trends”.
- Next it calls function “getSentiment” to get tweets and tweet attributes related to hot topics in a loop. This function returns average values of Sentiments, total tweets and retweets for all tweets related to that topic.
  To find out the sentiment polarity, it uses “TextBlob” library.
- Then it plots correlation between different attributes using matplotlib library by calling function “plotGraph(AvgSentiments,color_patt,xlabel,ylabel)
- It also saves tweet results in a “Sentiments.csv” file.

4. Screenshots of the program output





Picture1:-Sentiment Analysis Graph



A1 Hot Topic

	A	B	C	D	E
1	Hot Topic	Total Views	Sentiment	Retweets	
2	Gronk Dirty Hit	50000	Negative	55	
3	Chris Pratt	50000	Positive	21	
4	Bears Ears National Monument	50000	Positive	1419	
5	National Cookie Day	50000	Positive	396	
6	Weather Channel	50000	Positive	12877	
7	Corrine Brown	50000	Negative	1732	
8	Jay Z	50000	Positive	1693	
9	Lamborghini Urus	50000	Positive	153	
10	Jurassic World 2	50000	Positive	42987	
11	IOTA	50000	Positive	157	
12	Dow Jones	50000	Negative	240	
13	Hour of Code	100000	Positive	19	
14	Amber Alert Utah	100000	Negative	83	
15	Melanie Martinez	100000	Positive	1721	
16	Alvin Kamara	100000	Negative	890	
17	Shashi Kapoor	100000	Positive	592	
18	Herm Edwards	100000	Positive	5463	
19	Shohei Otani	100000	Positive	608	
20	Finance	200000	Positive	143	
21	LiAngelo Ball	200000	Positive	12665	
22					
23					

Picture 2:-CSV Output

5. Conclusion –

First plot in picture 1 shows a **conclusive** co-relation between sentiments and re-tweets. The output shows that tweets with negative sentiments are re- tweeted less comparative to positive sentiments tweets.

But on the other hand second plot of picture 1 shows a **non-conclusive** relation between web Traffic and Total tweets. That shows every metric doesn't generate any useful output.

The above conclusions are based out of various other plotting metrics' being tested using different combination of attributes as "likes", "retweets, "polarity", "web traffic" and "total tweets".

** Program Code**

```
import urllib.request
from bs4 import BeautifulSoup
from textblob import TextBlob
import tweepy
import matplotlib.pyplot as plt
import csv

# Step 1 - Authenticate
consumer_key= 'cyUgsIyEz9uWTQ58E9cKlXDLv'
consumer_secret= 'bpzeVGqEn48mlEdzrnDqxBJjQXzfNmQcsPiJPd8wCpdBPa2NtV'
access_token='817298744-0rjezQ7EcsSyLvlmC5Gl4PDsCDfhwKAaZxwSjIZy'
access_token_secret='lemVAyz76BHAqihG1Wij9U6C7qzfukGk8j7Uipmk1JZ18'

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

def getSentiment(api, key):
    public_tweets = api.search(key)
    AvgSentiment = 0
    noOfTweets = len(public_tweets)
    sum1 = 0
    avgRetweets=0
    total_rtw_ct =0
    for tweet in public_tweets:
        text = tweet.text
        rtw_count = tweet.retweet_count
        cleanedtext = ' '.join([word for word in text.split(' ') if len(word) > 0 and word[0] != '@'\
          and word[0] != '#'and 'http' not in word and word != 'RT'])
        #print(cleanedtext)
        analysis = TextBlob(cleanedtext)
        sentiment = analysis.sentiment.polarity
        sum1 += sentiment
        total_rtw_ct +=rtw_count
        if sentiment == 0:
            #ignore since not a opinion, its a general statement
            noOfTweets -= 1
    if noOfTweets > 0:
        AvgSentiment = sum1/noOfTweets
        avgRetweets=  total_rtw_ct/noOfTweets
    return (AvgSentiment,int(avgRetweets),noOfTweets)
```

```
def plotGraph(plotData,color_patt,xlabel,ylabel):
        lists=sorted(plotData.items())
        x,y= zip(*lists)
        plt.title("Loading Analysis Results ...")
        plt.xlabel(xlabel)
        plt.ylabel(ylabel)
        #plot data
        plt.plot(x,y,color_patt,x,y,'r--')
        #show plot
        plt.show()
#------------------------------------------------------------------------#

url=urllib.request.urlopen("http://www.google.com/trends/hottrends/atom/feed").read().decode("utf-
8")
soup=BeautifulSoup(url, features="xml")
title = []
for element in soup.find_all('title'):
        #print(element.string)
        if element.string == "Hot Trends":
                continue
        title.append(element.string)
views = []
for element in soup.find_all('approx_traffic'):
        view = element.string.replace(',','')
        view = view.strip('+')
        views.append(int(view))

i = 0
trends = dict()
for element in title:
        trends[element] = views[i]
        i += 1
trends = sorted(trends.items(), key=lambda x:x[1])
trends = dict(trends)

coordinates = dict()
flag = 0
plotData_sr={}
plotData_vl={}

sentiment=0
```

```
retweet =1
totaltweets=2
with open('sentiment.csv', 'w', newline='\n') as  f:
        writer = csv.writer(f)
        writer.writerow(['Hot Topic','Total Views', 'Sentiment','Retweets','Total tweets'])
        for key, value in trends.items():
                tweet_data = getSentiment(api, key)
                plotData_sr[tweet_data[sentiment]]= tweet_data[retweet]
                plotData_vl[ tweet_data[totaltweets]]= value
                if tweet_data[sentiment] <0:
                        sent_data ="Negative"
                else:
                        sent_data ="Positive"
                writer.writerow([key,value,sent_data,tweet_data[retweet],tweet_data[totaltweets]])
plotGraph(plotData_sr,'g^','Sentiment','Retweet')

plotGraph(plotData_vl,'bo','Total Tweets','Views')
```