Go Yaana Snippet

PROJECT REPORT

Twitter Sentiment Analysis

Introduction.

Twitter is a popular microblogging service where users create status messages (called "tweets")

This project report gives a brief idea or gist of the Sentiment Analysis project done at Go Yaana

The purpose of this project is to create an algorithm, which takes an "hashtag" as input and then does analysis of the tweets related to this hashtag. After the analysis is done , a pie-chart is printed which displays the percentages of various sentiments related to the hashtag.Then it outputs a word cloud which constitutes of the various words that were more frequently used in the tweets related to the tag. The density and size of these words are proportional their frequencies in the tweets.This work can further be modified by creating an interactive API which can ask the user for the input and then in addition to the mentioned outputs it can also print timlines of tweets, trends and images related to these tweets, location,etc.

In [27]:

```
from textblob import TextBlob
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import matplotlib.pyplot as plt
```

In [11]:

```
import tweepy

# Authenticate to Twitter
auth = tweepy.OAuthHandler("FSRKkhwpNMmxqePNYvLb9vfNJ",
    "eFUpsUjTGp1uIUdO1UUPLfX6aas8a29cqtxq92hVujcZiezY5C")
auth.set_access_token("1126809976259764224-fqUPI198oN6BbI2I7L9bGZG8nvk0sg",
    "rB1f55jomIRnqKDVWAhOHTGkFTxLd5Dqx2qiGUGNQJeTx")

api = tweepy.API(auth)

try:
    api.verify_credentials()
    print("Authentication OK")
except:
    print("Error during authentication")
```

```
Authentication OK
```

In [12]:

```
c=0
for tweet in api.search(q="fortunehotels", lang="en", rpp=100):
    print(f"{tweet.user.name}:{tweet.text}")
    c+=1
    print(c)
    #print()
```

```
Fortune Select Global:Sometimes we all need to just sit back and relax. With our #DayUseP
ackage, you'll exactly get what you need for a c… https://t.co/IrqRa5uhpr
1
Fortune Select Global:Our expert team at #FortuneSelectGlobal believes in a hassle-free w
ay of living. We provide you with hygienic laund… https://t.co/H7qcitYzJ2
```

```
2
Fortune Select Global:Our expert team at #FortuneSelectGlobal ensures regular cleaning &a
mp; sanitization of common areas to provide you with… https://t.co/Rp5ZSGK0IK
3
Fortune Select Global:The festival of Raksha Bandhan is to cherish the beautiful memories
and strengthen the bond you share with your sib… https://t.co/TKHLOKjQMu
4
```

In [11]:

```
print(c)
```

```
15
```

In [151]:

```
tw=[]
search_words = "#mussoorie"
date_since = "2021-06-1"
tweets = tweepy.Cursor(api.search,
              q=search_words,
              lang="en",
              since=date_since).items(500)
#print(tweets)
for tweet in tweets:
    tw.append(tweet.text)
    #print(tweet.text)
```

In [152]:

```
tw[2]
```

Out[152]:

```
'Mussoorie - of snowfall and rice bowls.\n.\n.\n.\n#goSTOPSShots #goSTOPSTravel #Mussoori
e #Workation #MountainLovers… https://t.co/VJpAvnLSZx'
```

In [153]:

```
import re

def decontracted(phrase):
    # specific
    phrase = re.sub(r"won't", "will not", phrase)
    phrase = re.sub(r"can\'t", "can not", phrase)

    # general
    phrase = re.sub(r"n\'t", " not", phrase)
    phrase = re.sub(r"\'re", " are", phrase)
    phrase = re.sub(r"\'s", " is", phrase)
    phrase = re.sub(r"\'d", " would", phrase)
    phrase = re.sub(r"\'ll", " will", phrase)
    phrase = re.sub(r"\'t", " not", phrase)
    phrase = re.sub(r"\'ve", " have", phrase)
    phrase = re.sub(r"\'m", " am", phrase)
    return phrase
stopwords= ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're"
, "you've",\
            "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', '
his', 'himself', \
            'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'th
ey', 'them', 'their',\
            'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "tha
t'll", 'these', 'those', \
            'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had
', 'having', 'do', 'does', \
            'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as',
'until', 'while', 'of', \
            'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through',
'during', 'before', 'after',\
            'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'ove
r', 'under', 'again', 'further',\
            'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any'
, 'both', 'each', 'few', 'more',\
            'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than', 'too',
'very', \
            's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now
', 'd', 'll', 'm', 'o', 're', \
            've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't",
'doesn', "doesn't", 'hadn',\
            "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'might
n', "mightn't", 'mustn',\
            "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wa
sn', "wasn't", 'weren', "weren't", \
            'won', "won't", 'wouldn', "wouldn't","https"]
```

In [154]:

```
from tqdm import tqdm
preprocessed_tweets = []
# tqdm is for printing the status bar
for sentance in tqdm(tw):
    sent = decontracted(sentance)
    sent = sent.replace('https://','')
    sent = sent.replace('\\r', ' ')
    sent = sent.replace('\\"', ' ')
    sent = sent.replace('\\n', ' ')
    sent = re.sub('[^A-Za-z0-9]+', ' ', sent)
# https://gist.github.com/sebleier/554280
    sent = ' '.join(e for e in sent.split() if e.lower() not in stopwords)
    preprocessed_tweets.append(sent.lower().strip())
```

```
100%|████████████████████████████████████████████████████████████████| 288/
288 [00:00<00:00, 5374.70it/s]
```

In [155]:

```
preprocessed_tweets[2]
```

Out[155]:

```
'mussoorie snowfall rice bowls gostopsshots gostopstravel mussoorie workation mountainlov
ers co vjpavnlszx'
```

In [156]:

```
def percentage(part,whole):
    return 100 * float(part)/float(whole)
```

In [157]:

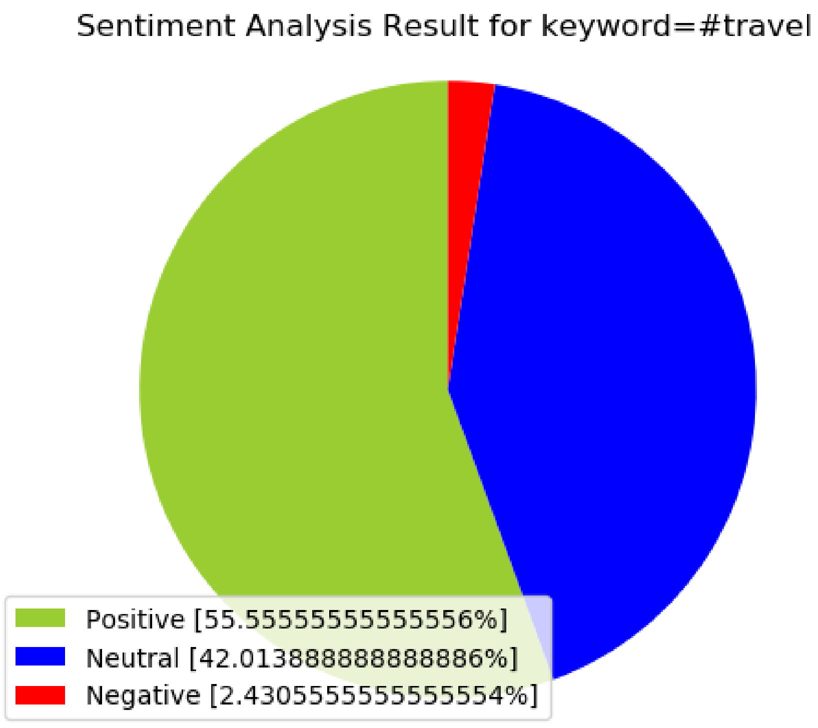
```
positive = 0
negative = 0
neutral = 0
polarity = 0
tweet_list = []
neutral_list = []
negative_list = []
positive_list = []
for tweet in tw:
    #tweet_list.append(tweet.text)
    analysis = TextBlob(tweet)
    score = SentimentIntensityAnalyzer().polarity_scores(tweet)
    neg = score['neg']
    neu = score['neu']
    pos = score['pos']
    comp = score['compound']
    polarity += analysis.sentiment.polarity
    if neg > pos:
        negative_list.append(tweet)
        negative += 1
    elif pos > neg:
        positive_list.append(tweet)
        positive += 1
    elif pos==neg:
        neutral_list.append(tweet)
        neutral+=1

    #print(score)
```

In [158]:

```
positive = percentage(positive, len(tw))
negative = percentage(negative, len(tw))
neutral  = percentage(neutral, len(tw))
polarity = percentage(polarity, len(tw))
```

In [159]:

```
labels = ['Positive ['+str(positive)+'%]' , 'Neutral ['+str(neutral)+'%]','Negative ['+st
r(negative)+'%]']
sizes = [positive, neutral, negative]
colors = ['yellowgreen', 'blue','red']
patches, texts = plt.pie(sizes,colors=colors, startangle=90)
plt.style.use('default')
plt.legend(labels)
plt.title('Sentiment Analysis Result for keyword='+keyword+' ')
plt.axis('equal')
plt.show()
```



In [160]:

```
strr=""
for i in positive_list:
    strr=strr+i
```

In [161]:

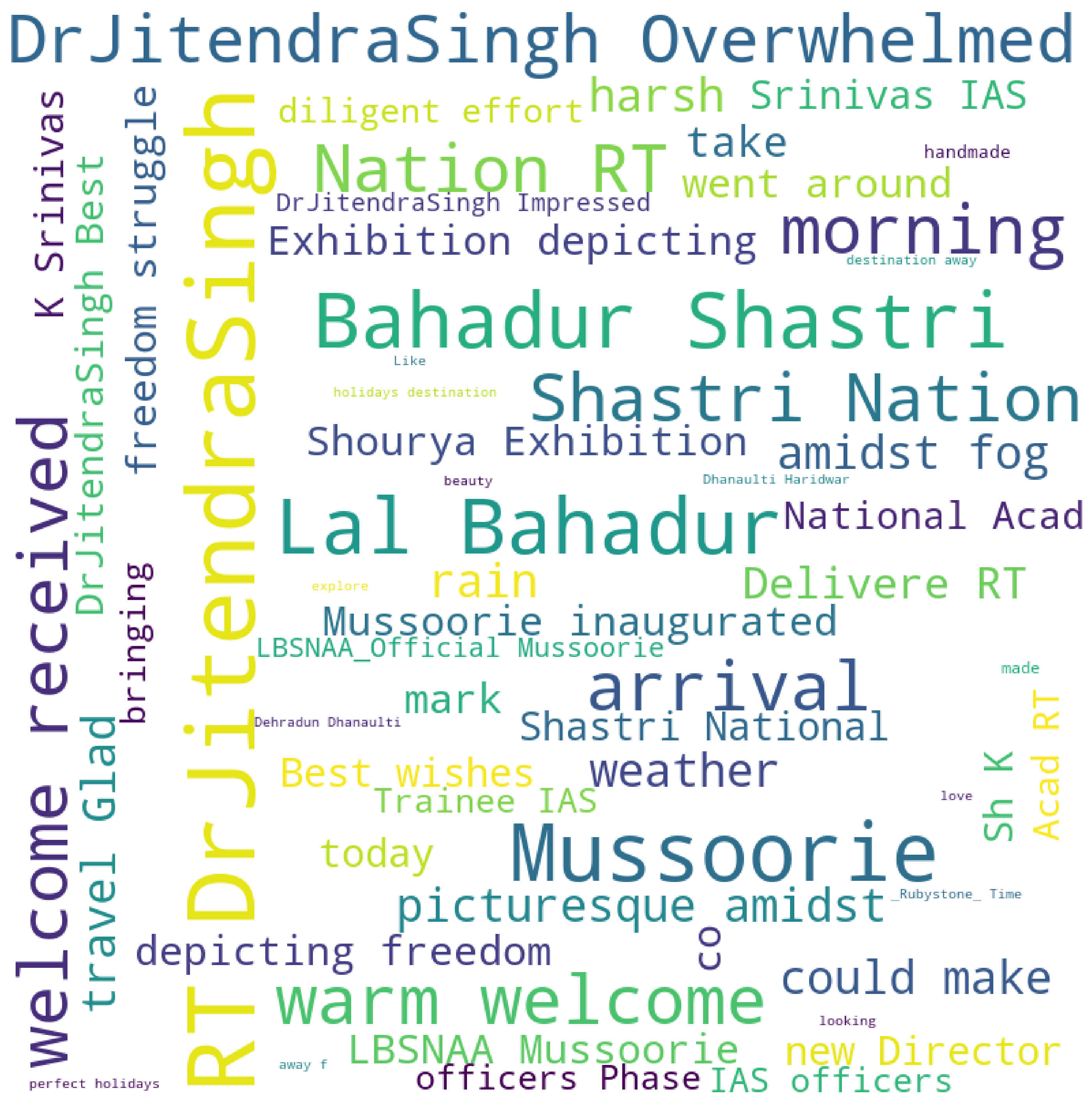
```
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
```

In [162]:

```
wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(strr)
```

In [163]:

```
plt.figure(figsize = (8, 8), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```



In [148]:

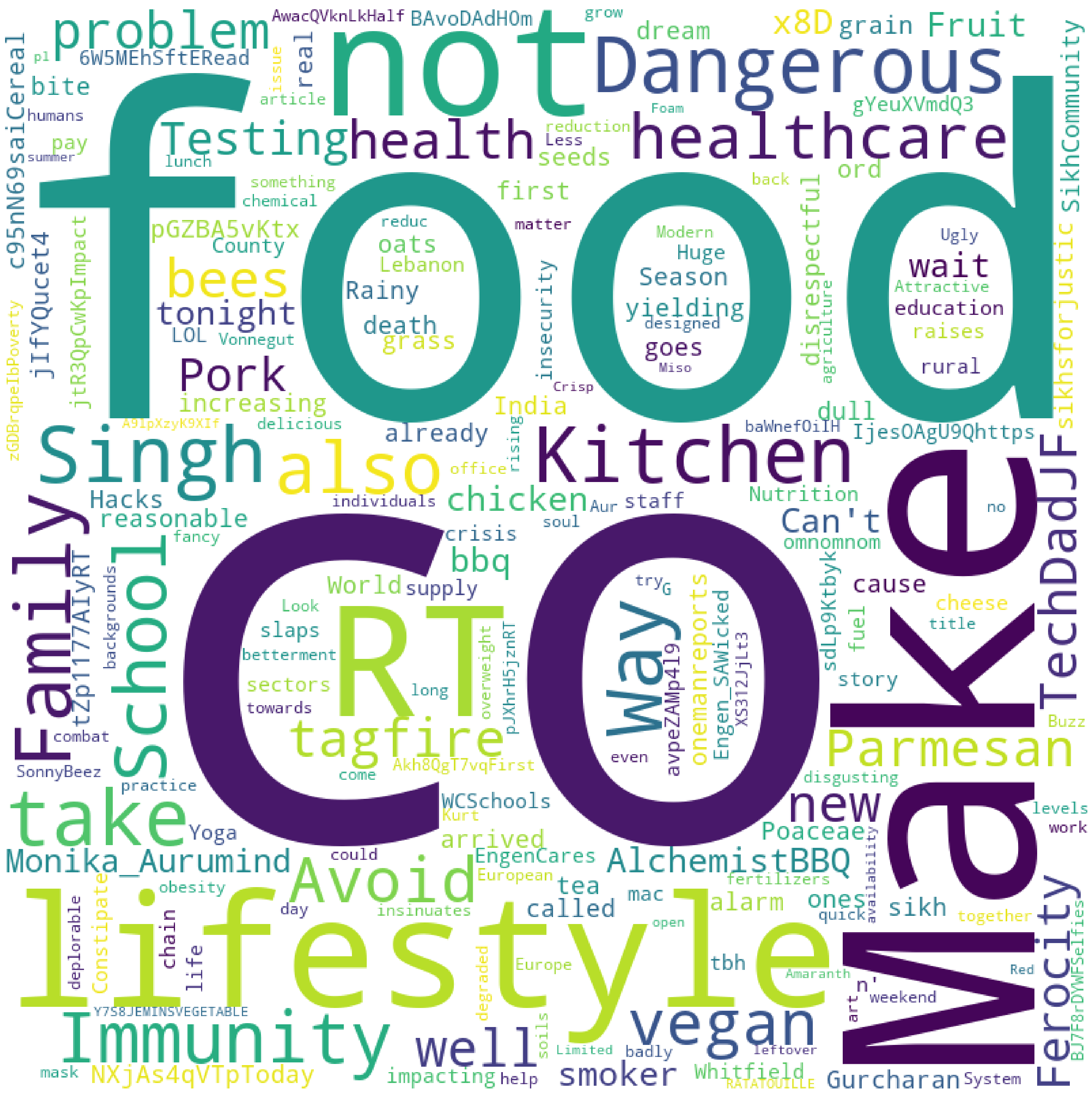
```
neg=""
for j in negative_list:
    neg=neg+j
```

In [149]:

```
wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(neg)
```

In [150]:

```
plt.figure(figsize = (8, 8), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```



Surya Kiran

Email-suryakirangorthi@gmail.com