



# **Sentiment Analysis for Stimulus check using Twitter Scraping**



**ISDS-558  
Group 13**

**Submitted to: -**  
Dr. Daoji Li

**Submitted by: -**  
Ami Thakkar  
Sagar Pathak  
Satish Pinnuri

## **Introduction**

The Internal Revenue Service(IRS) has announced on April 8 to send electronic payments (stimulus checks) to 115 million Americans as part of the \$2 trillion coronavirus law. It is to support increasing unemployment cases day by day due to pandemic. This has generated a lot of buzz among Americans. It has mixed reviews for how this will support the economy of Americans and glitches related to it. In our project, we will gather data from twitter using scrapping and run sentimental analysis on it. We have selected three different time frames to extract data and analyze trends for changing views regarding the Stimulus check.

## **Data extraction**

We extracted data by using the GetOldTweets3 package. The Keyword we are looking for is #Stimuluscheck. Data size is set to 15000 and data is received in the form of an array of arrays. We have extracted data for three timelines:

1. 04/12/2020 – 04/15/2020
2. 04/20/2020 – 04/23/2020
3. 04/29/2020 – 05/01/2020

In Data Extraction, we are getting information like Date, Time, and related Tweets. Further, we have converted data into Data Frame which is as follow:

Here we can notice that we have received untidy data containing punctuation marks, hyperlinks, parenthesis, and commas which are referred to as stop words. For sentimental analysis, we are more focused on words that provide some meaning, and the need for removal of stop words arises.

## **Data Pre-processing**

Data Cleaning is used to remove unwanted symbols. As for hyperlinks, symbols, RT (retweet), mentions do not add any information but create noise here. We have considered removing it so that we can focus on the context of the text. Moreover, our focus is analyzing the percentage of

people having positive, negative, and neutral views regarding Stimulus check and retweets will create noise counting it multiple times. We have also used some STOPWORDS like COVID, Coronavirus, F\*\*\*, sh\*\*, stimulus which does not add any contribution to sentimental analysis.

## Analysis

Currently, we have extracted and cleaned twitter data of Stimulus-check. Now, the goal is to analyze the data for the sentiment analysis. We will carry out the following tasks to know views of people in three different time frames:

- Percentage of positive and negative tweets in all three time frames.
- Trends in positive, negative and neutral tweet
- Word cloud for positive and negative words for all three time frames.
- Most frequently used words

In this project, we have used an open source text mining data analysis model called TextBlob.

## TextBlob Algorithm

TextBlob provides an API that can perform different Natural Language Processing (NLP) tasks like Noun Phrase Extraction, Classification (Naive Bayes, Decision Tree), Sentiment Analysis, Part-of-Speech Tagging, Language Translation, and Detection, Spelling Correction, etc( Mukesh Chapagain, 2018).

TextBlob is built upon **Natural Language Toolkit (NLTK)**.

Sentiment Analysis in simple terms is, analyzing the sentiment of a given text or document and categorizing the text/document into a specific class or category (like positive or negative or neutral). In other words, we can say that sentiment analysis classifies any particular text or document as positive or negative or neutral. Basically, the classification is done for two classes: positive and negative. However, we can add more classes like neutral, highly positive, highly negative, etc.

## Procedure

### 1. **Step 1:** Install the TextBlob

```
pip install -U textblob
```

### 2. **Step 2:** Getting sentiment of the tweet:

We have created a function that returns the sentiment of the given tweet. The sentiment can be either positive or negative or neutral. This sentiment is decided based on the keywords used in the tweet.

To get these values we pass the cleaned tweet text to the TextBlob class that creates a TextBlob object. The object contains the sentiment polarity and subjectivity of the text. The measures are: if the polarity is greater than zero: the tweet is considered as positive if the value is less than zero: it is considered negative and if the value is zero: the tweet is considered neutral.

In the code snippet, we have segregated the tweets based on the tweet values and stored in three different arrays.

### 3. **Step 3:** Process tweets

The next step is to create a new function that gets the tweet sentiment and returns an array of tweets and their respective sentiment value. This function helps us to append tweet sentiment value to the tweet text and return an array of processed tweets. These processed tweets can be further used to calculate the percentage of tweet sentiments in step 4

**Output:**

```
{'text': 'Has any received their #Stimuluscheck yet? Just asking ..', 'sentiment': 'neutral'}

{'text': '@realDonaldTrump 20 weeks for Americans to receive checks is complete incompetence. #Stimuluscheck #coronavirus', 'sentiment': 'positive'}

{'text': 'Who got that stim check? #CoronavirusOutbreak #QuarantineLife #Stimuluscheck', 'sentiment': 'neutral'}

{'text': 'When you see that your #Government #Stimuluscheck has been sent but you're bank account still has a negative balance #economy #broke #Jobs #CancelTheDebt #BailOutThePeople', 'sentiment': 'negative'}

{'text': 'Thank you Mr Trump I'm getting a whole half inch, I'll be above 4 for the first time many blessings #Stimuluscheck ', 'sentiment': 'positive'}

{'text': 'Check your balance: Coronavirus stimulus money starts to flow into bank accounts @COVID19 #coronavirus #Stimuluscheck ', 'sentiment': 'neutral'}

{'text': '#Stimuluscheck ', 'sentiment': 'neutral'}
```

**4. Step 4: Getting tweet percentage**

The last step in this process is to get the percentages of positive, negative, and neutral tweets from the whole tweets. This percentage calculation can be done by summing up the respective tweet count(positive, negative, and neutral) and dividing the individual tweet count by the total number of tweets multiplied by 100.

**Output:**

Postive Tweets		Count: 875 , Percent: 35.17 %
Negative Tweets		Count: 316 , Percent: 12.7 %
Neutral Tweets		Count: 1297 , Percent: 52.13 %
Postive Tweets		Count: 921 , Percent: 38.04 %
Negative Tweets		Count: 498 , Percent: 20.57 %
Neutral Tweets		Count: 1002 , Percent: 41.39 %
Postive Tweets		Count: 387 , Percent: 40.87 %
Negative Tweets		Count: 222 , Percent: 23.44 %
Neutral Tweets		Count: 338 , Percent: 35.69 %

### 5. Step 5:

This is the entire process for calculating the sentiment values for tweets and getting the percentage values for positive, negative, and neutral tweets. As we are planning to get sentiment values for three different time intervals we need to repeat steps: 1 to 4 to get those sentiment values. After getting the sentiment values for three different time frames, we have planned to check the trend lines for public views. How the sentiment values have been changed based on time.

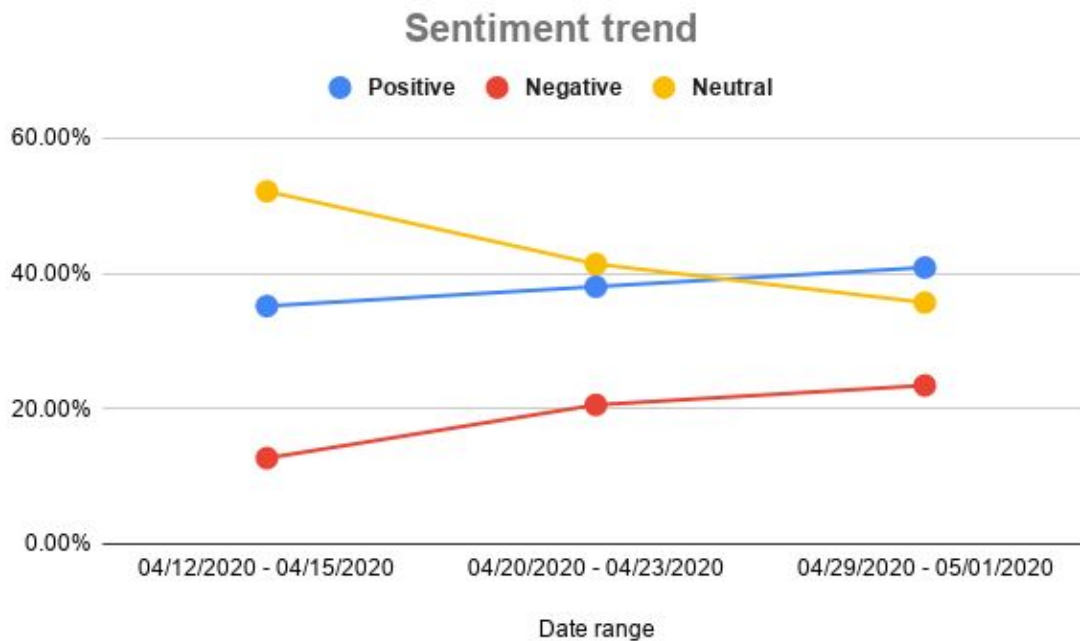
### Calculate sentiment trend

Now we have sentiment values(percentages) of the tweets for three different time frames. The considered date ranges and the percentage values are tabulated below. Using this data we have visualized the trend of public views on stimulus check.

Date	Positive	Negative	Neutral
04/12/2020 - 04/15/2020	35.17%	12.70%	52.13%
04/20/2020 - 04/23/2020	38.04%	20.57%	41.39%
04/29/2020 - 05/01/2020	40.87%	23.44%	35.69%

### Trend analysis

By looking at the visualizations it is clear that the majority of the public has reacted positively for the stimulus check initiative. Although the neutral views have been the highest in the very first date range, it is gradually decreased with the increasing time frame. The positive and negative trends have been consistently increasing throughout the three date ranges while the neutral trend has been decreasing


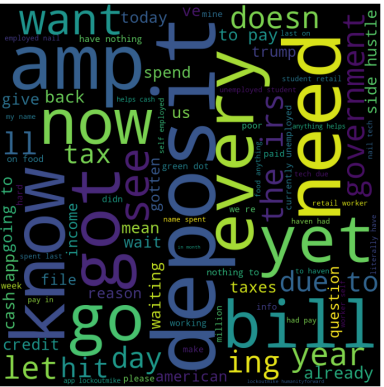
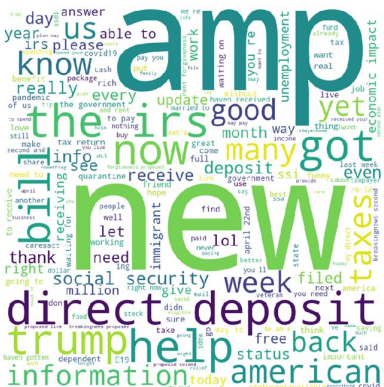
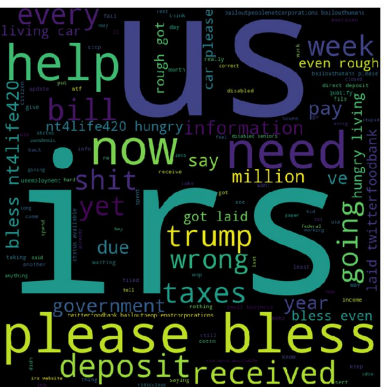




## Creating a Word Cloud

**Word cloud** is an image composed of words used in a particular text or subject, in which the size of each word indicates its frequency or importance. So, the more often a specific word appears in the given text, the bigger and bolder it appears in the word cloud.

We are trying to analyze the most frequent words or buzz going on around people during three-time frames.

1. Following are the word clouds for three different time frames
  - a. The positive word cloud is represented with the white background
  - b. The negative word cloud is represented with the black background

Time Frame	Positive Word Cloud	Negative Word Cloud
04/12/2020 - 04/15/2020	 <p>use enough going to got take now hit pending trump know the irs new taxes amp way yet happy american thank better later live month right bill us people lol pay every direct deposit even receive please</p>	 <p>want today doesn't amp spend give back now us need government know see every yet bill go waiting taxes due to credit day hit american ing year already</p>
04/20/2020 - 04/23/2020	 <p>day answer us able to amp work year irs please know cash pay you package really every update good yet the irs good many got bill receiving find new taxes direct deposit trump help status back information today american</p>	 <p>every living car help us week even rough bill nt4life420 hungry information now say million ve irs yet trump wrong going please bless deposit received</p>
04/29/2020 - 05/01/2020	 <p>year lawsuit filed section lawsuit just published employee in sba changed new eidl pploan forgiveness right thank direct deposit in violation published exclusive help trump 10k sba grant application</p>	 <p>has wave just people who will there has purported second coming glitch causing purported website round second pua application just published website glitch application in amp be second in california coming second exclusive new second round</p>



## **Word cloud analysis:**

- **Word list for time frame 04/12/2020 - 04/15/2020:**

**Positive word list:**

Deposit , Receive, good, lol, Happy, free, funds, exclusive

**Negative word list:**

Spend, spent last, Unemployed student, currently unemployed, Wait, Question, side hustle

**Inference:** This is the initial phase when the Stimulus check was announced. People are happy out receiving free funds. As well as unemployment is increasing day by day. People are waiting for their stimulus check and have many questions

- **Word list for time frame 04/20/2020 - 04/23/2020:**

**Positive word list:**

Thank, Benefit, love, Rich, Direct Deposit, Information, better

**Negative word list:**

even rough, get laid, hungry, disabled seniors, wrong, shit, pandemic, the wrong feel

**Inference:** People have started receiving their checks but on the other side the situation is getting worse and people are getting laid off.

- **Word list for time frame 04/29/2020 - 05/01/2020:**

**Positive word list:**

loan forgiveness, Grant, Care, Thank, Sure

**Negative word list:**

Second wave, Incorrect, Denial, Unemployment, Website glitch, Exhausted, Wrong

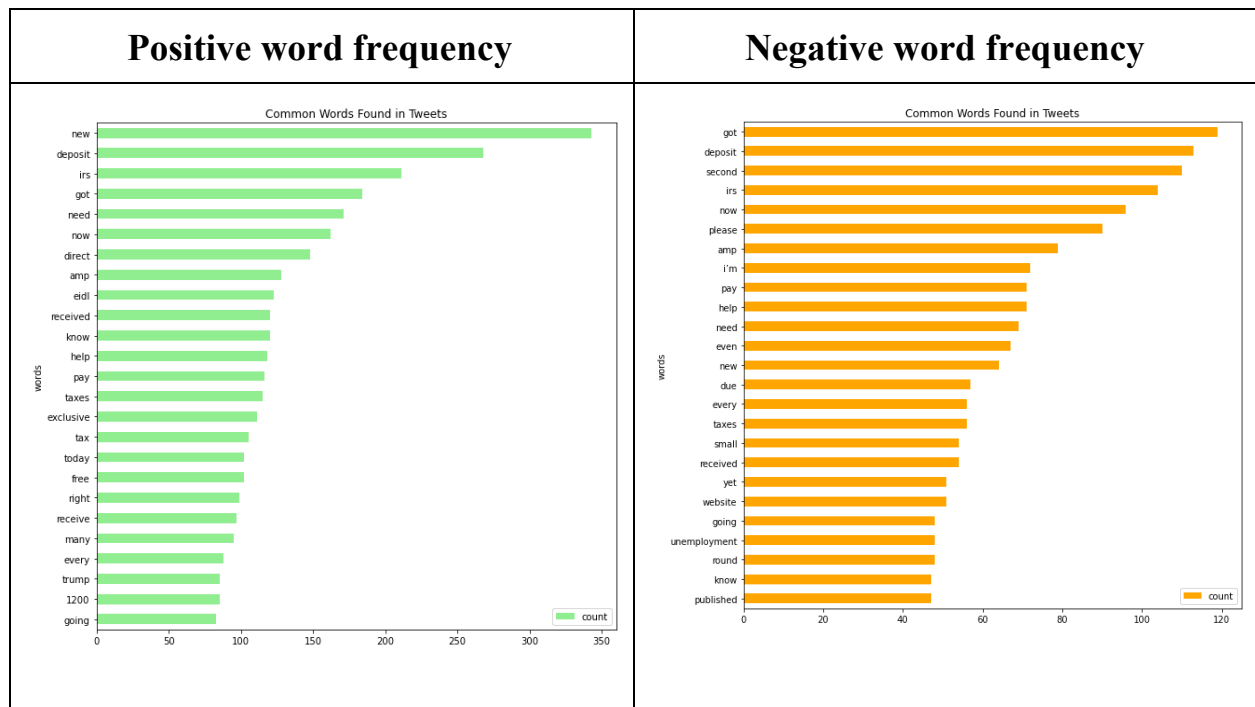
**Inference:** Government is helping people with loan forgiveness but unemployment is at its peak and people are demanding a second round of Stimulus Check.

## High-frequency words

We have generated high frequency words from positive and negative tweets using Collection library in Python. Following are most used Positive and Negative words during three time frames.

**Most used Positive words:** Deposit, received, free, got, etc.

**Most used Negative words:** Please, help, unemployment, etc.



## Conclusion

Here comes the inferences part from the sentiment analysis of twitter data.

Firstly, from trend analysis we can summarize 41% of people are positive, 21% of people are negative and 38% were neutral to this stimulus check initiative.

Next, from the word-cloud analysis, more positive keywords are being used when compared to negative keywords. Positive Keywords like help, need, happy, free, food, benefit, thank indicates that this initiative is really helping people who are in need.

Hence, from both the trend analysis and the word cloud analysis we can infer that people are really happy with this initiative and will also be extremely happy if the government can provide the second round of stimulus checks.

## References

<http://blog.chapagain.com.np/python-twitter-sentiment-analysis-on-real-time-tweets-using-textblob/>

<https://pypi.org/project/GetOldTweets3/>

<https://python-graph-gallery.com/wordcloud/>

<https://stackoverflow.com/questions/44750574/creating-wordcloud-using-python>

<https://www.digitalocean.com/community/tutorials/how-to-perform-sentiment-analysis-in-python-3-using-the-natural-language-toolkit-nltk>

## Appendix: Python code

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:
import pandas as pd
import GetOldTweets3 as got
from nltk.tokenize import TweetTokenizer
import nltk
from nltk.corpus import stopwords
import re, string
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import re # importing regex
import string
nltk.download('stopwords')

# In[2]:
text_query = '#stimuluscheck'
count = 15000
# Creation of query object
tweetCriteria =
```

```

got.manager.TweetCriteria().setQuerySearch(text_query).setSince("2020-
04-12").setUntil("2020-04-15").setMaxTweets(count)# Creation of list
that contains all tweets
tweets = got.manager.TweetManager.getTweets(tweetCriteria)
# Creating list of chosen tweet data
text_tweets = [[tweet.date, tweet.text] for tweet in tweets]
text_tweets

```

```

# In[3]:
#storing in dataframe
df = pd.DataFrame(text_tweets)
df.head()
df=df.rename(columns={0: "Date", 1: "Tweet"})
df["Tweet"]

```

```

# In[7]:
# Creation of query object
tweetCriteria1 =
got.manager.TweetCriteria().setQuerySearch(text_query).setSince("2020-
04-20").setUntil("2020-04-23").setMaxTweets(count)# Creation of list
that contains all tweets
tweets1 = got.manager.TweetManager.getTweets(tweetCriteria1)
# Creating list of chosen tweet data
text_tweets1 = [[tweet.date, tweet.text] for tweet in tweets1]
text_tweets1

```

```

# In[9]:
#storing in dataframe
df1 = pd.DataFrame(text_tweets1)
df1.head()
df1=df1.rename(columns={0: "Date", 1: "Tweet"})
df1["Tweet"]

```

```

# In[10]:
# Creation of query object
tweetCriteria2 =
got.manager.TweetCriteria().setQuerySearch(text_query).setSince("2020-
04-29").setUntil("2020-05-01").setMaxTweets(count)# Creation of list
that contains all tweets
tweets2 = got.manager.TweetManager.getTweets(tweetCriteria2)
# Creating list of chosen tweet data
text_tweets2 = [[tweet.date, tweet.text] for tweet in tweets2]
text_tweets2

```

```

# In[12]:
#storing in dataframe
df2 = pd.DataFrame(text_tweets2)
df2.head()
df2=df2.rename(columns={0: "Date", 1: "Tweet"})
df2['Tweet']

# In[8]:
#data cleaning
import re # importing regex
import string
def clean_tweet(tweet):
    '''
    Remove unnecessary things from the tweet
    like mentions, hashtags, URL links, punctuations
    '''
    # remove old style retweet text "RT"
    tweet = re.sub(r'^RT[\s]+', '', tweet)

    # remove hyperlinks
    tweet = re.sub(r'https?:\/\/.*[\r\n]*', '', tweet)

    # remove hashtags
    # only removing the hash # sign from the word
    tweet = re.sub(r'#', '', tweet)
    tweet = re.sub(r"stimulus", " ", tweet)
    tweet = re.sub(r"Stimulus", " ", tweet)
    tweet = re.sub(r"stimulus check", " ", tweet)
    tweet = re.sub(r"stimuluscheck", " ", tweet)
    tweet = re.sub(r"check", " ", tweet)
    tweet = re.sub(r"Check", " ", tweet)
    tweet = re.sub(r"money", " ", tweet)
    tweet = re.sub(r"payment", " ", tweet)
    tweet = re.sub(r"Payment", " ", tweet)
    tweet = re.sub(r"COVID19", " ", tweet)
    tweet = re.sub(r"COVID", " ", tweet)
    tweet = re.sub(r"Coronavirus", " ", tweet)
    tweet = re.sub(r"coronavirus", " ", tweet)
    tweet = re.sub(r"bank", " ", tweet)
    tweet = re.sub(r"account", " ", tweet)
    tweet = re.sub(r"people", " ", tweet)
    tweet = re.sub(r"LALATE", " ", tweet)
    tweet = re.sub(r"lalate", " ", tweet)
    tweet = re.sub(r"fuck", " ", tweet)
    tweet = re.sub(r"fucking", " ", tweet)
    tweet = re.sub(r"still", " ", tweet)

```

```

tweet = re.sub(r"will", " ", tweet)
tweet = re.sub(r"don", " ", tweet)
tweet = re.sub(r"time", " ", tweet)
tweet = re.sub(r"getting", " ", tweet)
tweet = re.sub(r"irs", " ", tweet)
tweet = re.sub(r"i'm", " ", tweet)
tweet = re.sub(r"one", " ", tweet)
# remove mentions
tweet = re.sub(r'@[A-Za-z0-9]+', '', tweet)

# remove punctuations like quote, exclamation sign, etc.
# we replace them with a space
tweet = re.sub(r'['+string.punctuation+']+', ' ', tweet)
return tweet

# In[15]:

print(df.size)
print(df1.size)
print(df2.size)
#creating temporary variable
df_temp = df
df1_temp = df1
df2_temp = df2

# In[10]:
from textblob import TextBlob, Word, Blobber
#function to calculate sentiment of tweet
def get_tweet_sentiment1(tweet):
    '''
    Get sentiment value of the tweet text
    It can be either positive, negative or neutral
    '''
    sentiment = pd.DataFrame(columns=['Clean_Text', 'Sentiment'])
    # create TextBlob object of the passed tweet text
    for i in tweet:
        #print(i)
        blob = TextBlob(clean_tweet(i))
        #print(blob)
        # get sentiment
        if blob.sentiment.polarity > 0:
            sentiment =
sentiment.append({'Clean_Text':str(blob), 'Sentiment':"Positive"}, ignore_index=True)

        elif blob.sentiment.polarity< 0:
            sentiment =

```

```
sentiment.append({'Clean_Text':str(blob),'Sentiment':"Negative"},ignore_index=True)
```

```
    else:
        sentiment =
sentiment.append({'Clean_Text':str(blob),'Sentiment':"Neutral"},ignore_index=True)
    #print(sentiment[0])
    return sentiment['Clean_Text'], sentiment['Sentiment']
```

```
# In[11]:
#finding sentiments for tweet
df_temp['Clean_Text'],df_temp['Sentiment'] =
get_tweet_sentiment1(df_temp['Tweet'])
df1_temp['Clean_Text'],df1_temp['Sentiment'] =
get_tweet_sentiment1(df1_temp['Tweet'])
df2_temp['Clean_Text'],df2_temp['Sentiment'] =
get_tweet_sentiment1(df2_temp['Tweet'])
```

```
# In[12]:
df_temp
```

```
# In[13]:
#function to calculate percentage of positive, negative and neutral
#sentiment
```

```
def sentiment_analysis(data):
    positive_count = len(data[data['Sentiment'] == 'Positive'])
    negative_count = len(data[data['Sentiment'] == 'Negative'])
    neutral_count = len(data[data['Sentiment'] == 'Neutral'])
```

```
    positive_percent = round(100 * positive_count /
len(data['Sentiment']),2)
    negative_percent = round(100 * negative_count /
len(data['Sentiment']),2)
    neutral_percent = round(100 * neutral_count /
len(data['Sentiment']),2)
```

```
    print ('Positive Tweets | Count: {} , Percent: {} %' .
format(positive_count, positive_percent))
    print ('Negative Tweets | Count: {} , Percent: {} %' .
format(negative_count, negative_percent))
    print ('Neutral Tweets | Count: {} , Percent: {} %' .
format(neutral_count, neutral_percent))
    print("\n")
```

```
# In[14]:
sentiment_analysis(df_temp)
```

```

sentiment_analysis(df1_temp)
sentiment_analysis(df2_temp)

# In[15]:
Positive_tweets1= df_temp[df_temp['Sentiment']=='Positive']
Positive_tweets2= df1_temp[df1_temp['Sentiment']=='Positive']
Positive_tweets3= df2_temp[df2_temp['Sentiment']=='Positive']

Negative_tweets1= df_temp[df_temp['Sentiment']=='Negative']
Negative_tweets2= df1_temp[df1_temp['Sentiment']=='Negative']
Negative_tweets3= df2_temp[df2_temp['Sentiment']=='Negative']

# In[16]:
#plotting word cloud
def plot_wordcloud(words,sentiment):
    comment_words = ''
    stopwords = set(STOPWORDS)
# iterate through the csv file
    for val in words:

        # typecaste each val to string
        val = str(val)

        # split the value
        tokens = val.split()

        # Converts each token into lowercase
        for i in range(len(tokens)):
            tokens[i] = tokens[i].lower()

        comment_words += " ".join(tokens)+" "
    if(sentiment == 'positive'):
        wordcloud = WordCloud(width = 800, height = 800,
                                background_color='white',
                                stopwords = stopwords,
                                min_font_size = 10).generate(comment_words)
    else:
        wordcloud = WordCloud(width = 800, height = 800,
                                background_color='black',
                                stopwords = stopwords,
                                min_font_size = 10).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (10, 10), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)

```



```

plt.show()

# In[30]:
plot_wordcloud(Positive_tweets1['Clean_Text'], "positive")

# In[31]:
plot_wordcloud(Negative_tweets1['Clean_Text'], "negative")

# In[36]:
plot_wordcloud(Negative_tweets2['Clean_Text'], "negative")

# In[38]:
plot_wordcloud(Positive_tweets3['Clean_Text'], "positive")

# In[39]:
plot_wordcloud(Negative_tweets3['Clean_Text'], "negative")

# In[43]:
#plotting word cloud
def plot_common_words(text, sentiment):
    stopwords = set(STOPWORDS)
    total_words = []
    filtered_sentence = []
# iterate through the csv file
    for val in text:

        # typecaste each val to string
        val = str(val)

        # split the value
        tokens = val.split()

        # Converts each token into lowercase
        for i in range(len(tokens)):
            tokens[i] = tokens[i].lower()
        total_words.append(tokens)

    for words in total_words:
        for w in words:
            if w not in stopwords:
                if len(w) > 2:
                    filtered_sentence.append(w)

```

```

tweet_collection = collections.Counter(filtered_sentence)

tweet_collection.most_common(25)
clean_tweets_nsw = pd.DataFrame(tweet_collection.most_common(25),
                                columns=['words', 'count'])

fig, ax = plt.subplots(figsize=(10, 10))

# Plot horizontal bar graph
if sentiment== "positive":
    color="lightgreen"
else:
    color="orange"

clean_tweets_nsw.sort_values(by='count').plot.barh(x='words',
                                                    y='count',
                                                    ax=ax,
                                                    color=color)

ax.set_title("Common Words Found in Tweets (Without Stop Words)")

plt.show()

# In[45]:
import collections
First_word= pd.DataFrame()
Second_word= pd.DataFrame()
Third_word= pd.DataFrame()

First_word['Tweet']=Positive_tweets1['Clean_Text'].append(Positive_tweets2['Clean_Text']).append(Positive_tweets3['Clean_Text'])
Second_word['Tweet']=Negative_tweets1['Clean_Text'].append(Negative_tweets2['Clean_Text']).append(Negative_tweets3['Clean_Text'])

plot_common_words(First_word['Tweet'], "positive")
plot_common_words(Second_word['Tweet'], "negative")

```