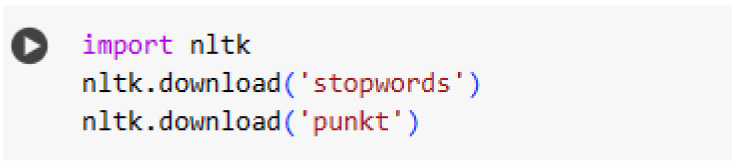
IMPLEMENTATION

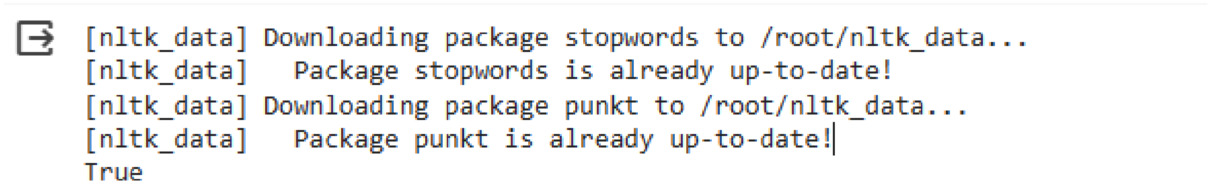
TWITTER DATASET: AVENGERS ENDGAME

4.1 SOURCE CODE AND OUTPUT SCREENS

1.Importing nltk module to download stopwords package

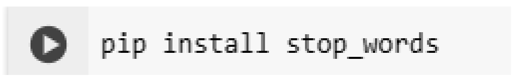


```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
```

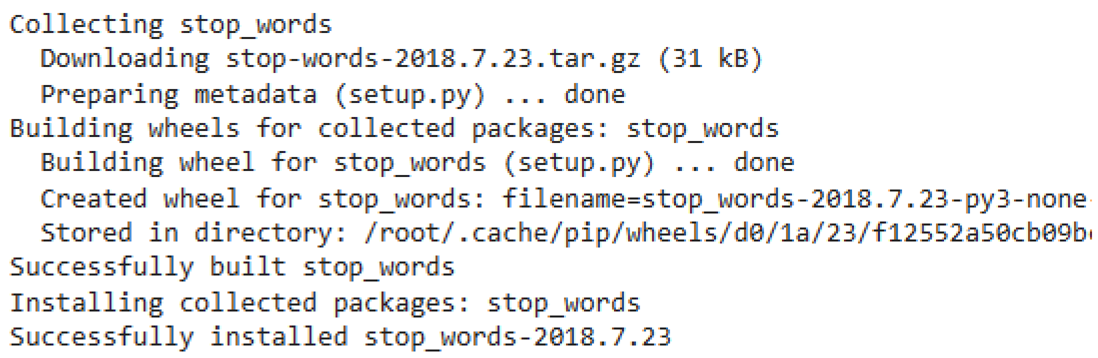


```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

2.Installing stopwords

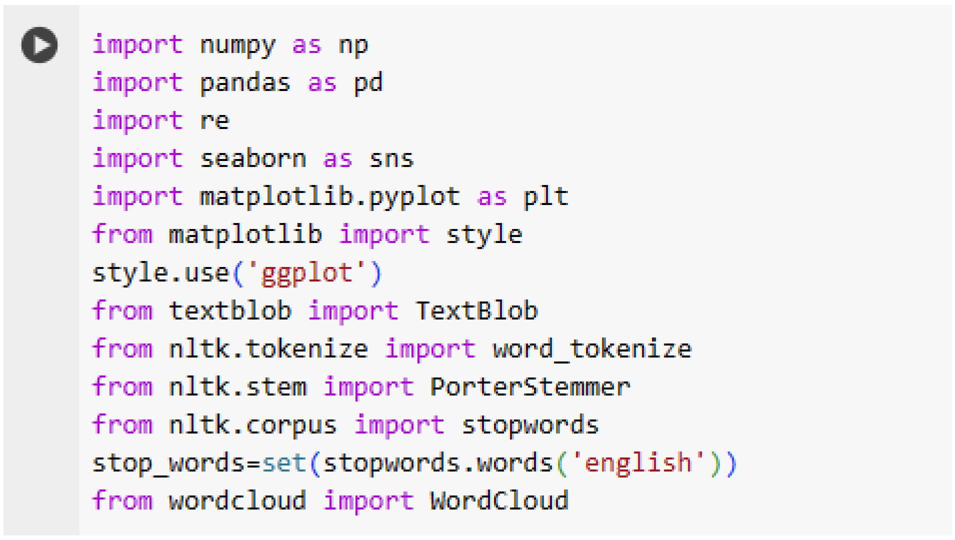


```
pip install stop_words
```



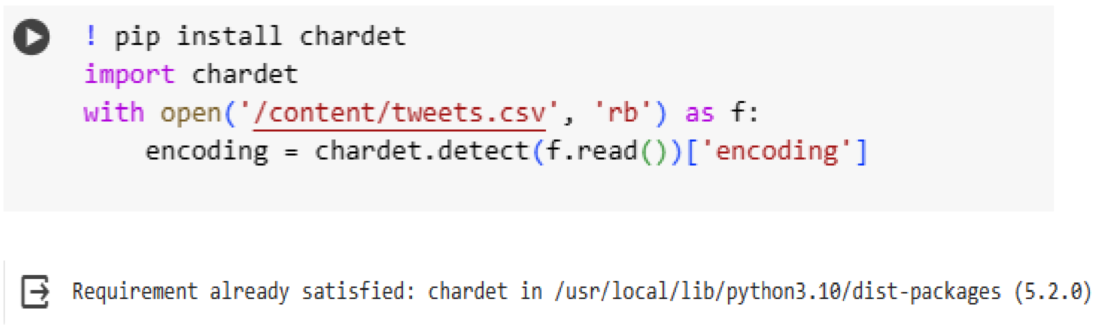
```
Collecting stop_words
  Downloading stop-words-2018.7.23.tar.gz (31 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: stop_words
  Building wheel for stop_words (setup.py) ... done
  Created wheel for stop_words: filename=stop_words-2018.7.23-py3-none
  Stored in directory: /root/.cache/pip/wheels/d0/1a/23/f12552a50cb09b
Successfully built stop_words
Installing collected packages: stop_words
Successfully installed stop_words-2018.7.23
```

3.Importing all necessary Libraries.



```python
import numpy as np
import pandas as pd
import re
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import style
style.use('ggplot')
from textblob import TextBlob
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from nltk.corpus import stopwords
stop_words=set(stopwords.words('english'))
from wordcloud import WordCloud
```

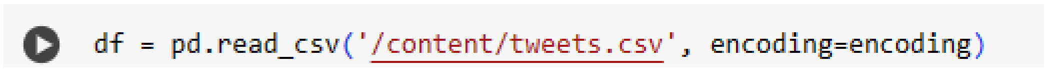
4.This code snippet installs the chardet library via pip, imports it, and then detects the encoding of the file located at '/content/tweets.csv' in a single line.



```python
! pip install chardet
import chardet
with open('/content/tweets.csv', 'rb') as f:
    encoding = chardet.detect(f.read())['encoding']
```

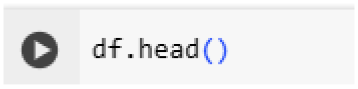
```
Requirement already satisfied: chardet in /usr/local/lib/python3.10/dist-packages (5.2.0)
```

5.Reading the csv located at the given path into pandas dataframe.



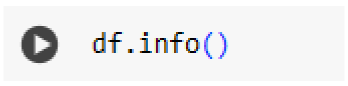
```python
df = pd.read_csv('/content/tweets.csv', encoding=encoding)
```

6.Printing first 5 rows of pandas dataframe.

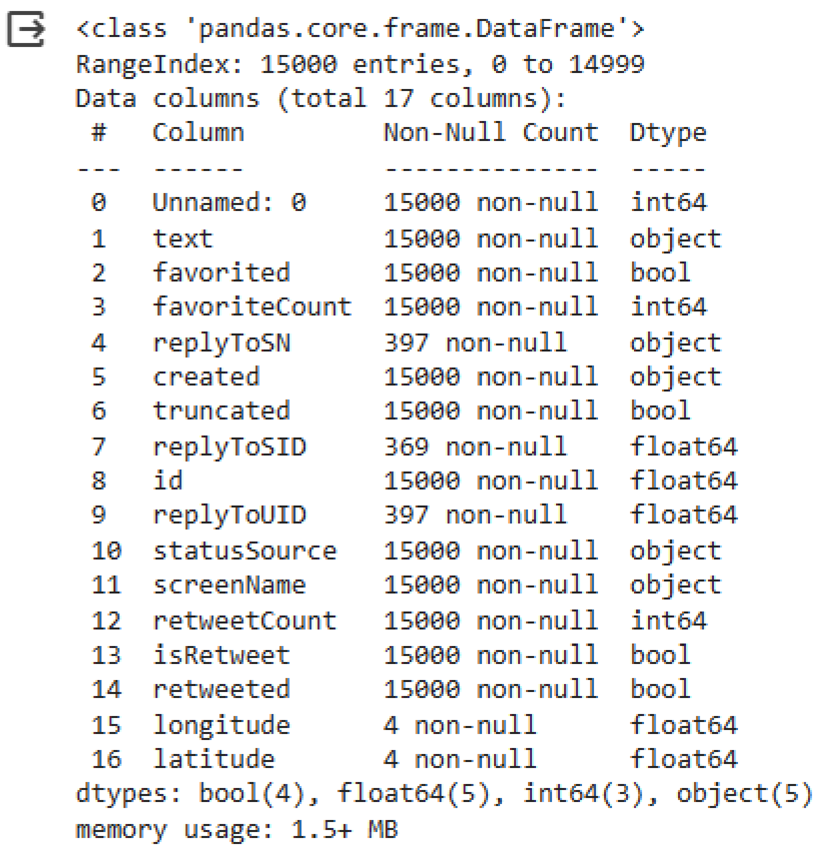


```python
df.head()
```

7. This method is used to display a concise summary of a pandas DataFrame.

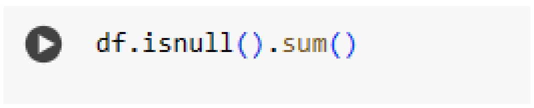


```
df.info()
```

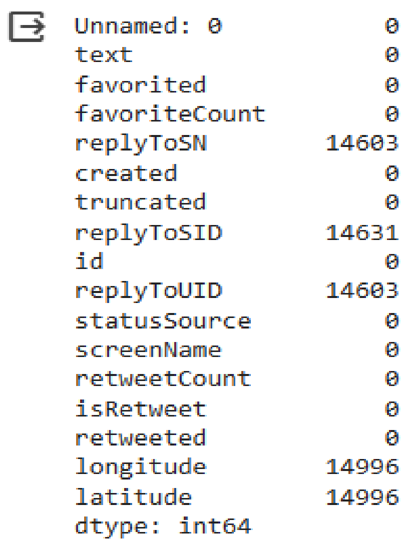


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 17 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Unnamed: 0     15000 non-null  int64  
 1   text           15000 non-null  object 
 2   favorited      15000 non-null  bool   
 3   favoriteCount  15000 non-null  int64  
 4   replyToSN      397 non-null    object 
 5   created        15000 non-null  object 
 6   truncated      15000 non-null  bool   
 7   replyToSID     369 non-null    float64
 8   id             15000 non-null  float64
 9   replyToUID     397 non-null    float64
 10  statusSource   15000 non-null  object 
 11  screenName     15000 non-null  object 
 12  retweetCount   15000 non-null  int64  
 13  isRetweet      15000 non-null  bool   
 14  retweeted      15000 non-null  bool   
 15  longitude      4 non-null      float64
 16  latitude       4 non-null      float64
dtypes: bool(4), float64(5), int64(3), object(5)
memory usage: 1.5+ MB
```

8. This method calculates and returns the total number of missing values in each column.

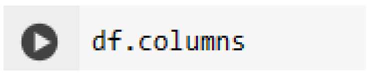


```
df.isnull().sum()
```

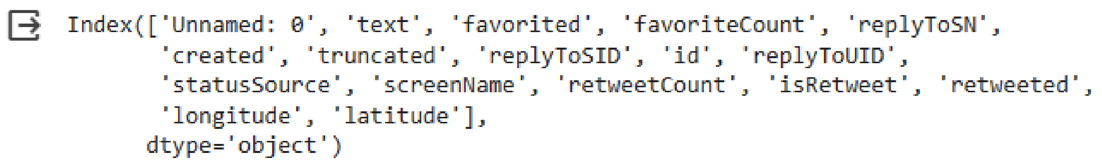


```
Unnamed: 0           0
text                 0
favorited            0
favoriteCount        0
replyToSN        14603
created              0
truncated            0
replyToSID       14631
id                   0
replyToUID       14603
statusSource         0
screenName           0
retweetCount         0
isRetweet            0
retweeted            0
longitude        14996
latitude         14996
dtype: int64
```

9. This method accesses the column labels of the pandas DataFrame df.

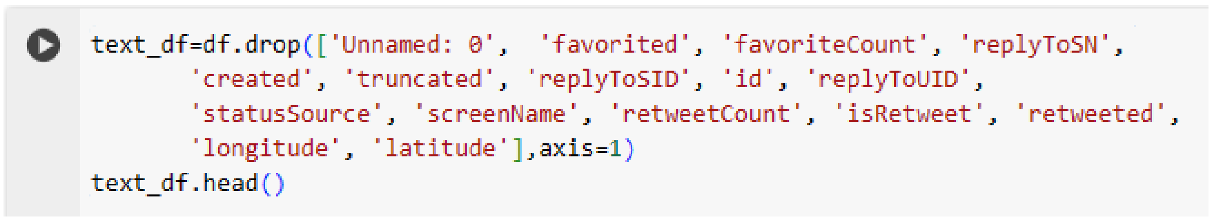


```
df.columns
```

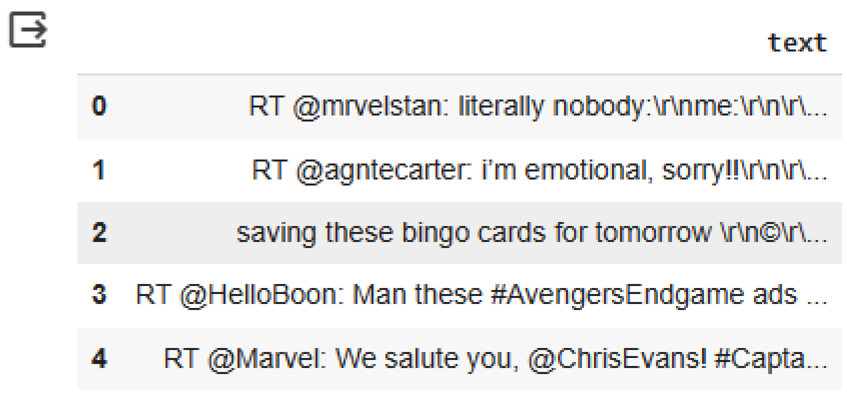


```
Index(['Unnamed: 0', 'text', 'favorited', 'favoriteCount', 'replyToSN',
       'created', 'truncated', 'replyToSID', 'id', 'replyToUID',
       'statusSource', 'screenName', 'retweetCount', 'isRetweet', 'retweeted',
       'longitude', 'latitude'],
      dtype='object')
```

10. This code removes specified columns from the DataFrame df and assigns the resulting DataFrame to text_df, then displays the first few rows of text_df.

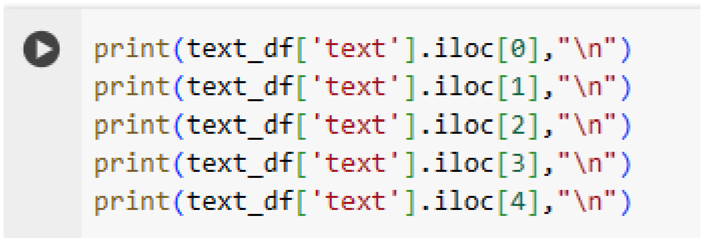


```
text_df=df.drop(['Unnamed: 0',  'favorited', 'favoriteCount', 'replyToSN',
       'created', 'truncated', 'replyToSID', 'id', 'replyToUID',
       'statusSource', 'screenName', 'retweetCount', 'isRetweet', 'retweeted',
       'longitude', 'latitude'],axis=1)
text_df.head()
```

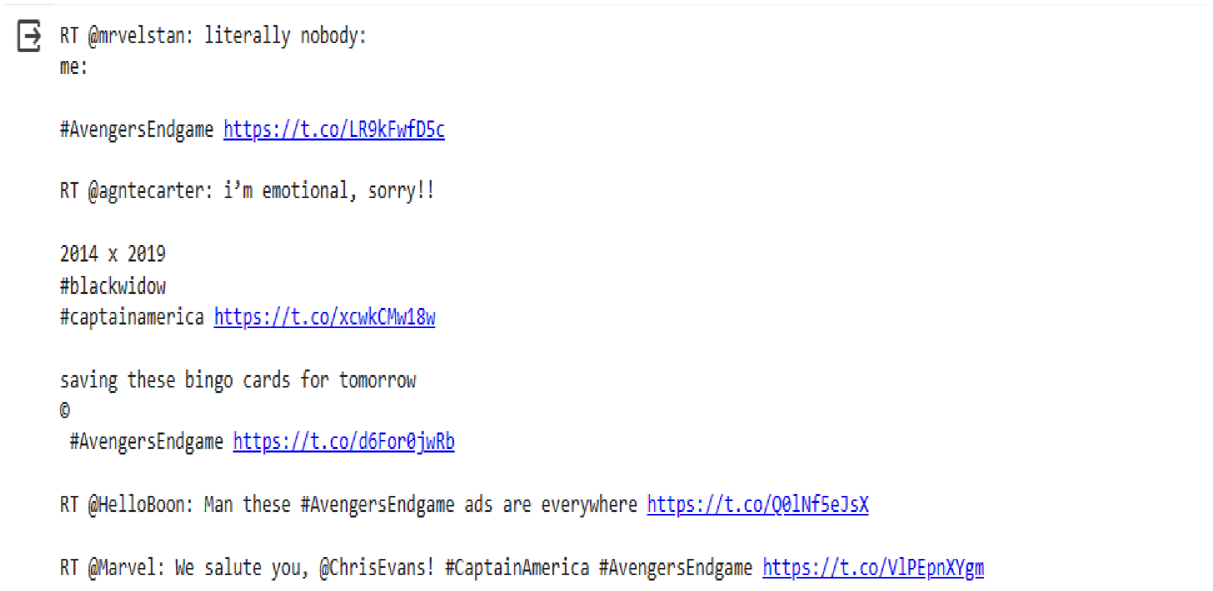


	text
0	RT @mrvelstan: literally nobody:\r\nme:\r\n\r\...
1	RT @agntecarter: i'm emotional, sorry!!\r\n\r\...
2	saving these bingo cards for tomorrow \r\n©\r\...
3	RT @HelloBoon: Man these #AvengersEndgame ads ...
4	RT @Marvel: We salute you, @ChrisEvans! #Capta...

11. These print statements display the text content of the first five rows of the 'text' column.

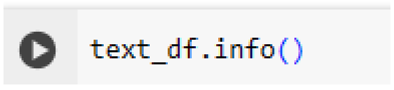


```
print(text_df['text'].iloc[0],"\n")
print(text_df['text'].iloc[1],"\n")
print(text_df['text'].iloc[2],"\n")
print(text_df['text'].iloc[3],"\n")
print(text_df['text'].iloc[4],"\n")
```

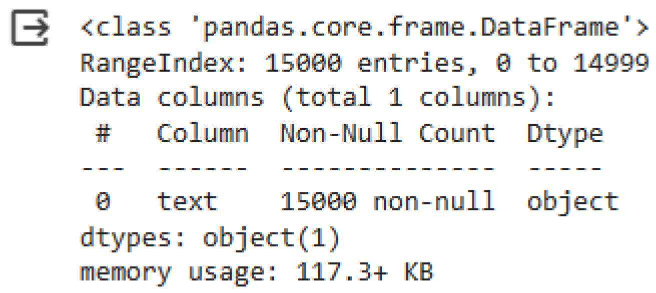


```
RT @mrvelstan: literally nobody:
me:

#AvengersEndgame https://t.co/LR9kFwfD5c

RT @agntecarter: i’m emotional, sorry!!

2014 x 2019
#blackwidow
#captainamerica https://t.co/xcwkCMw18w

saving these bingo cards for tomorrow
©
 #AvengersEndgame https://t.co/d6For0jwRb

RT @HelloBoon: Man these #AvengersEndgame ads are everywhere https://t.co/Q0lNf5eJsX

RT @Marvel: We salute you, @ChrisEvans! #CaptainAmerica #AvengersEndgame https://t.co/VlPEpnXYgm
```

12.This method shows concise summary of dataframe after dropping the unnecessary columns.

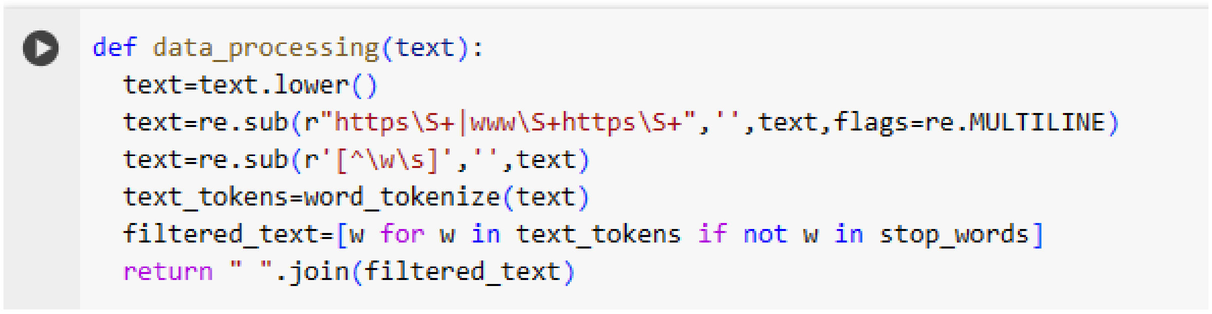


```
text_df.info()
```

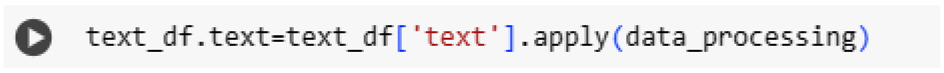


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    15000 non-null  object
dtypes: object(1)
memory usage: 117.3+ KB
```

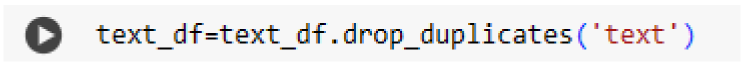
13. This function data_processing processes the input text by converting it to lowercase, removing URLs, non-alphanumeric characters, and stop words, tokenizing the text into words, and finally joining the filtered words back into a single string.


```
def data_processing(text):
  text=text.lower()
  text=re.sub(r"https\S+|www\S+https\S+",'',text,flags=re.MULTILINE)
  text=re.sub(r'[^\w\s]','',text)
  text_tokens=word_tokenize(text)
  filtered_text=[w for w in text_tokens if not w in stop_words]
  return " ".join(filtered_text)
```

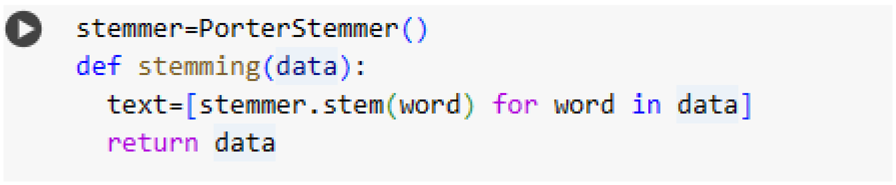
14. This code applies the function data_processing to each element in the 'text' column of the DataFrame text_df and assigns the processed text back to the 'text' column.


```
text_df.text=text_df['text'].apply(data_processing)
```

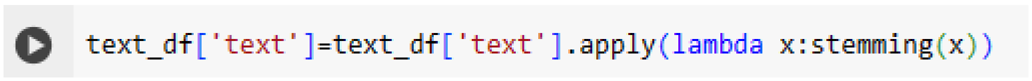
15.This line drops duplicate rows based on the values in the 'text' column of the DataFrame text_df and reassigns the result to text_df.

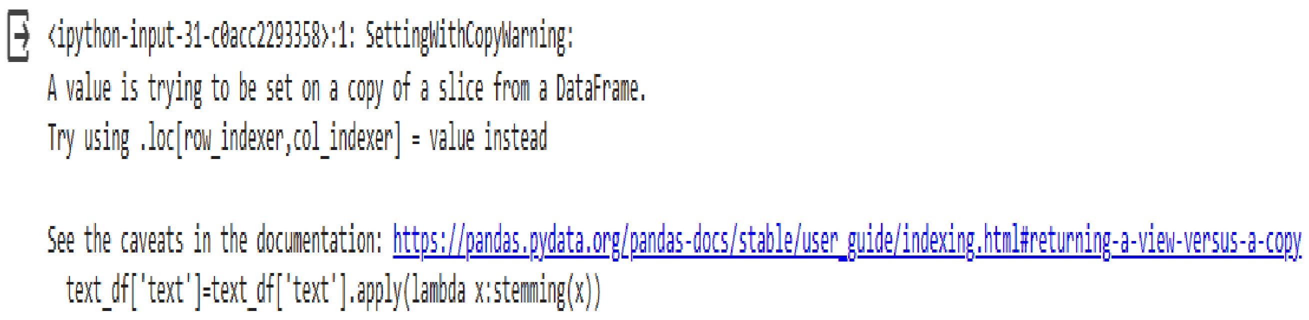

```
text_df=text_df.drop_duplicates('text')
```

16. This code defines a function called stemming that applies the Porter stemming algorithm to each word in the input list data and returns the stemmed words.


```
stemmer=PorterStemmer()
def stemming(data):
  text=[stemmer.stem(word) for word in data]
  return data
```

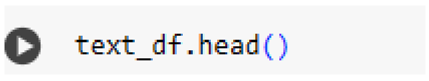
17. This applies the stemming function to each element in the 'text' column of the DataFrame text_df using a lambda function, resulting in each word being stemmed within each text entry.


```
text_df['text']=text_df['text'].apply(lambda x:stemming(x))
```

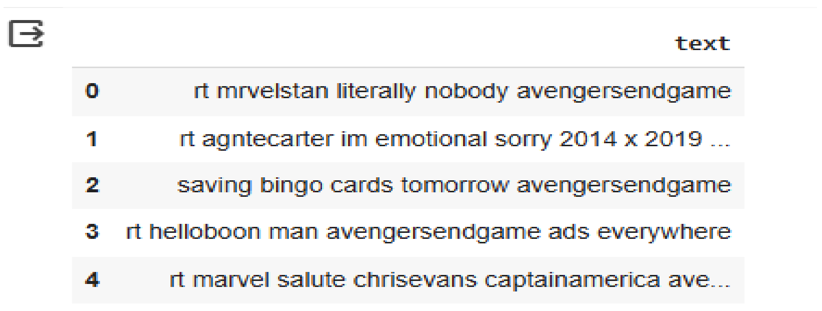


```
<ipython-input-31-c0acc2293358>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  text_df['text']=text_df['text'].apply(lambda x:stemming(x))
```

18. The output of text_df.head() would display the first few rows of the DataFrame, where each text entry in the 'text' column has been stemmed.

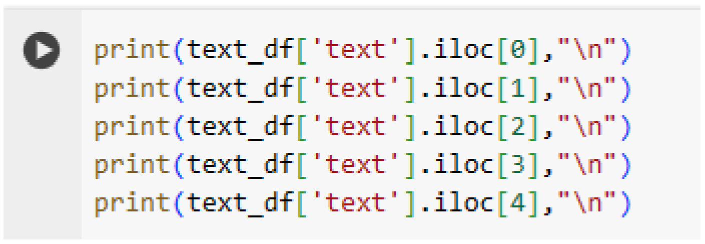


```
text_df.head()
```



	text
0	rt mrvelstan literally nobody avengersendgame
1	rt agntecarter im emotional sorry 2014 x 2019 ...
2	saving bingo cards tomorrow avengersendgame
3	rt helloboon man avengersendgame ads everywhere
4	rt marvel salute chrisevans captainamerica ave...

19.Prints the first 5 rows of text column after stemming.



```
print(text_df['text'].iloc[0],"\n")
print(text_df['text'].iloc[1],"\n")
print(text_df['text'].iloc[2],"\n")
print(text_df['text'].iloc[3],"\n")
print(text_df['text'].iloc[4],"\n")
```

```
rt mrvelstan literally nobody avengersendgame

rt agntecarter im emotional sorry 2014 x 2019 blackwidow captainamerica

saving bingo cards tomorrow avengersendgame

rt helloboon man avengersendgame ads everywhere

rt marvel salute chrisevans captainamerica avengersendgame
```

20.This method prints the concise summary but the text column will now consists of stemmed words.

```
text_df.info()
```

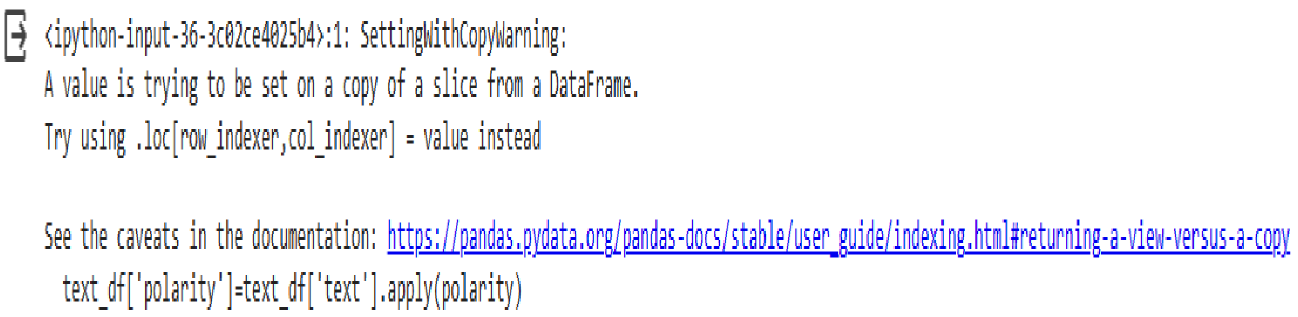
```
<class 'pandas.core.frame.DataFrame'>
Index: 2686 entries, 0 to 14997
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    2686 non-null   object
dtypes: object(1)
memory usage: 42.0+ KB
```

21. This function calculates the polarity of sentiment for a given text using the TextBlob library.

```
def polarity(text):
  return TextBlob(text).sentiment.polarity
```

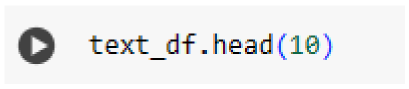
22. This line of code calculates the polarity score for each text entry in the 'text' column of the DataFrame text_df using the polarity function, and assigns the resulting polarity scores to a new column named 'polarity'.

```
text_df['polarity']=text_df['text'].apply(polarity)
```

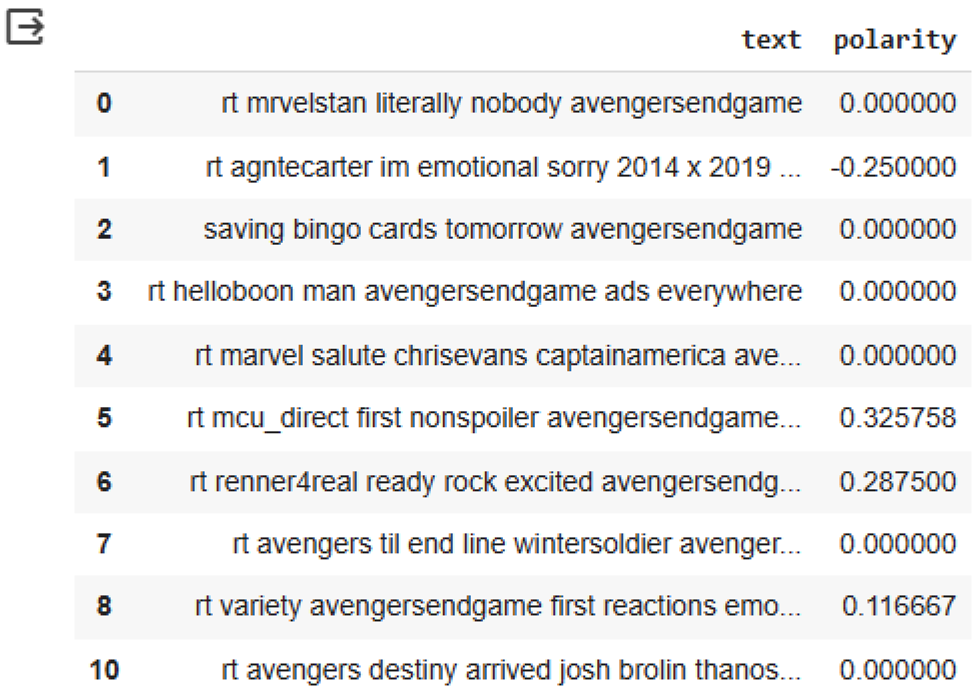


```
<ipython-input-36-3c02ce4025b4>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  text_df['polarity']=text_df['text'].apply(polarity)
```

23.Displays first 10 rows of text and polarity columns.



```
text_df.head(10)
```



	text	polarity
0	rt mrvelstan literally nobody avengersendgame	0.000000
1	rt agntecarter im emotional sorry 2014 x 2019 ...	-0.250000
2	saving bingo cards tomorrow avengersendgame	0.000000
3	rt helloboon man avengersendgame ads everywhere	0.000000
4	rt marvel salute chrisevans captainamerica ave...	0.000000
5	rt mcu_direct first nonspoiler avengersendgame...	0.325758
6	rt renner4real ready rock excited avengersendg...	0.287500
7	rt avengers til end line wintersoldier avenger...	0.000000
8	rt variety avengersendgame first reactions emo...	0.116667
10	rt avengers destiny arrived josh brolin thanos...	0.000000

24. This function, sentiment, takes a polarity score label as input and returns a corresponding sentiment label based on the polarity score. If the polarity score is less than 0, it returns "Negative". If the polarity score is equal to 0, it returns "Neutral". If the polarity score is greater than 0, it returns "Positive".

```
def sentiment(label):
              if label<0:
                return "Negative"
              elif label==0:
                return "Neutral"
              elif label>0:
                return "Positive"
```

25. This line of code applies the sentiment function to each polarity score in the 'polarity' column of the DataFrame text_df and assigns the resulting sentiment labels to a new column named 'sentiment'.

```
text_df['sentiment']=text_df['polarity'].apply(sentiment)
```

```
<ipython-input-39-8f1655674b26>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  text_df['sentiment']=text_df['polarity'].apply(sentiment)
```

26.Displays the contents of text column.

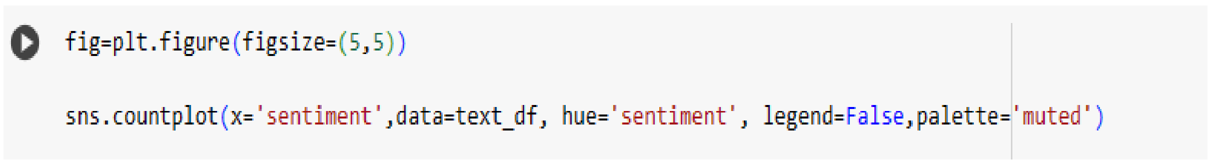
```
text_df["text"]
```

```
0               rt mrvelstan literally nobody avengersendgame
1         rt agntecarter im emotional sorry 2014 x 2019 ...
2                 saving bingo cards tomorrow avengersendgame
3             rt helloboon man avengersendgame ads everywhere
4         rt marvel salute chrisevans captainamerica ave...
                                ...
14979     im like today replace santa endgame avengersen...
14981     rt natportman_news natalie attended premiere a...
14982     long tomorrows double bill infinity war follow...
14989     ticketnew noted agreed teamu2714ufe0f u0001f38...
14997     spicinemas kindly announce approximate timings...
Name: text, Length: 2686, dtype: object
```

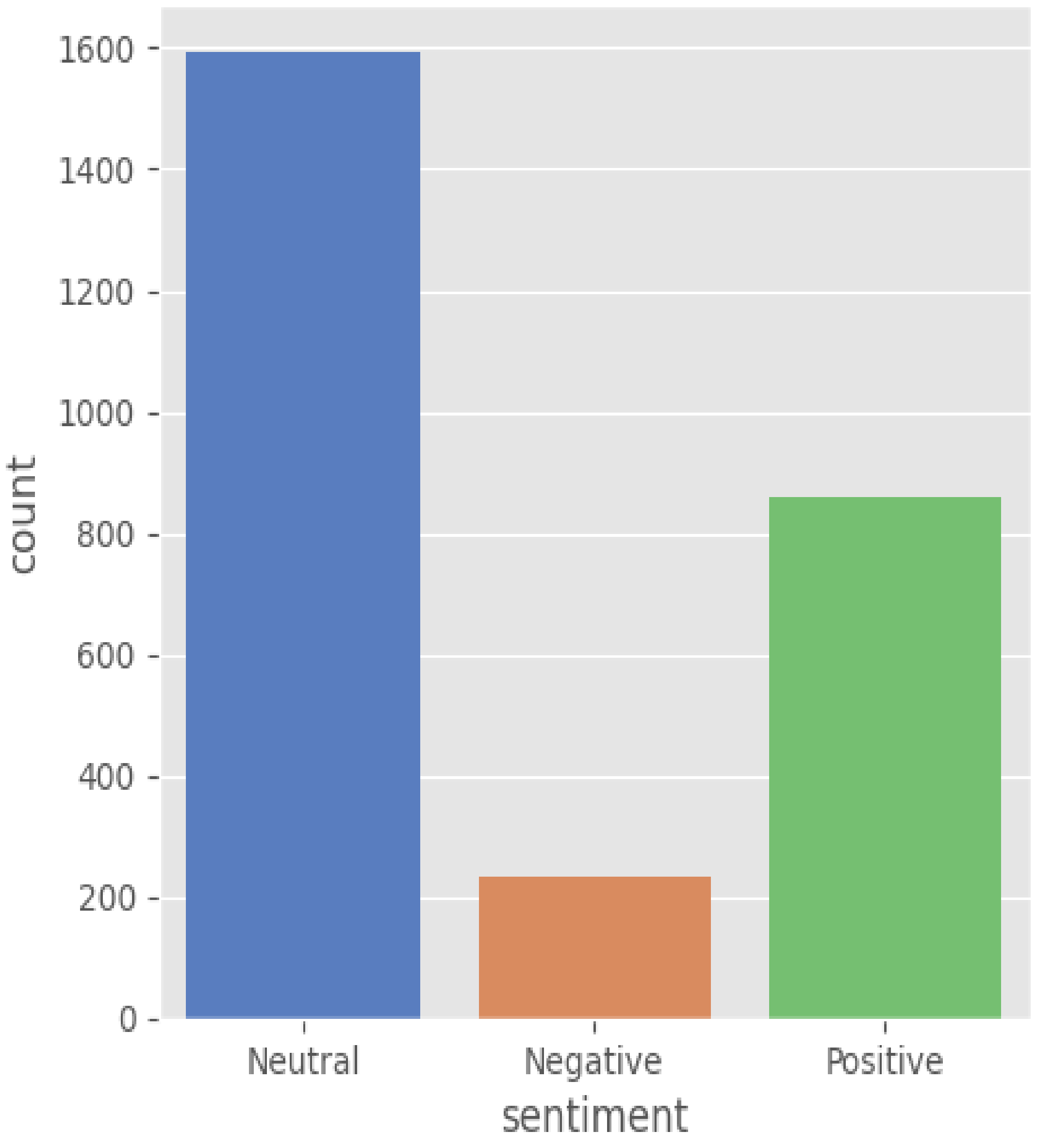
27. This code imports the matplotlib.pyplot module and the seaborn library, allowing you to create and customize visualizations in Python.

```
import matplotlib.pyplot as plt
import seaborn as sns
```

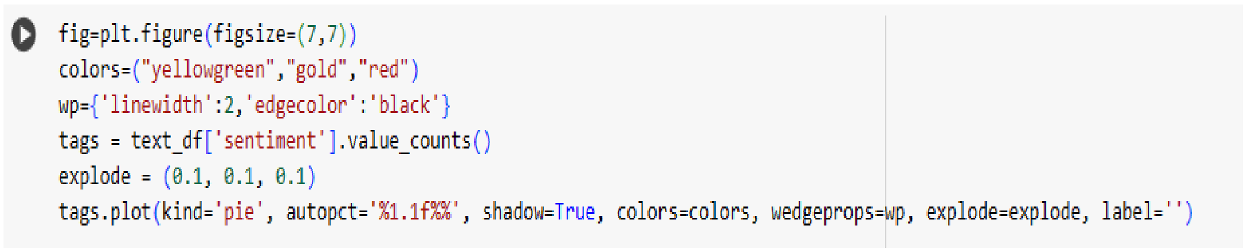
28. This code creates a countplot to visualize the distribution of sentiment labels in the DataFrame text_df, using seaborn. The figure size is set to (5, 5) inches, and the color palette is 'muted'. The legend is turned off for simplicity.



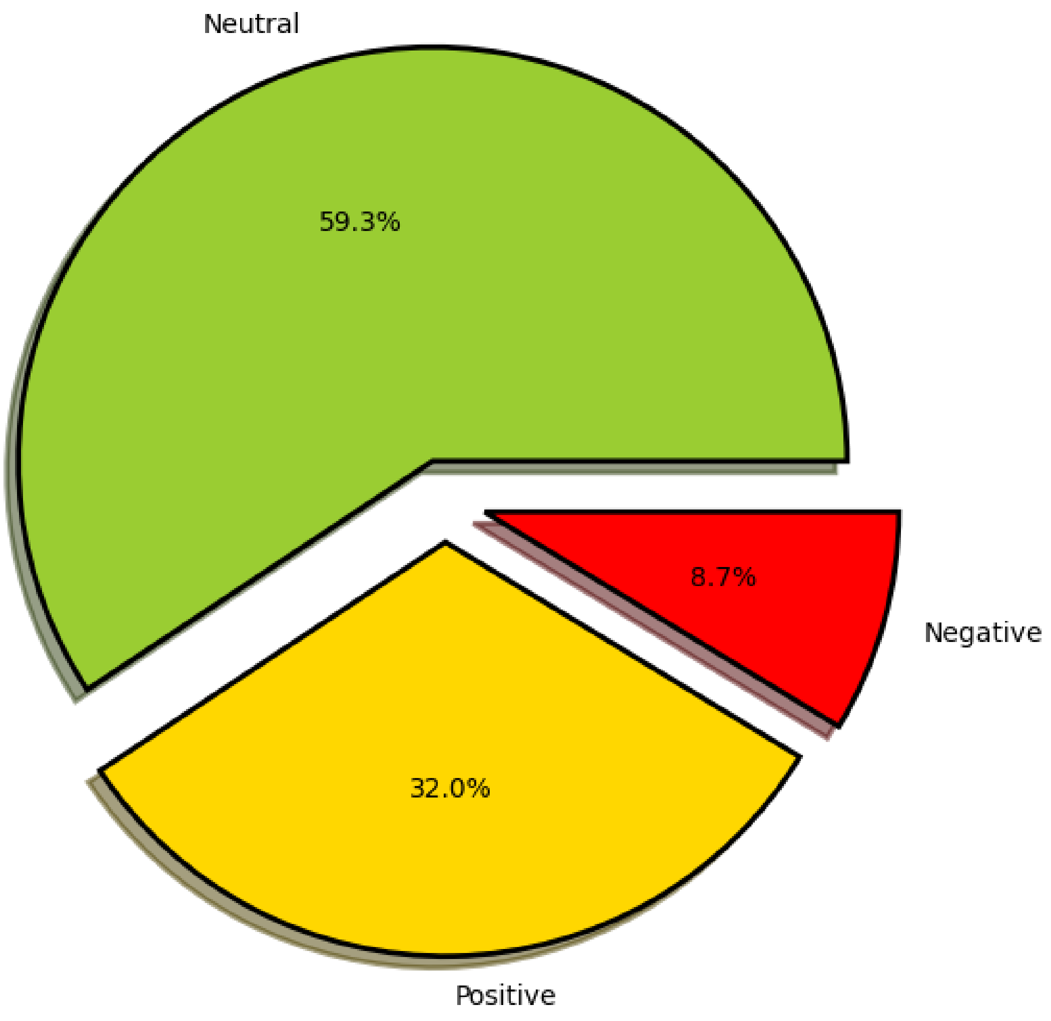
```
fig=plt.figure(figsize=(5,5))

sns.countplot(x='sentiment',data=text_df, hue='sentiment', legend=False,palette='muted')
```



29. This code creates a pie chart to display the distribution of sentiment labels in the DataFrame text_df. The chart has a size of (7, 7) inches and uses colors 'yellowgreen', 'gold', and 'red'. Each slice has a black edge, and the percentage of each label is shown inside its slice. An explosion effect is applied for better visualization.

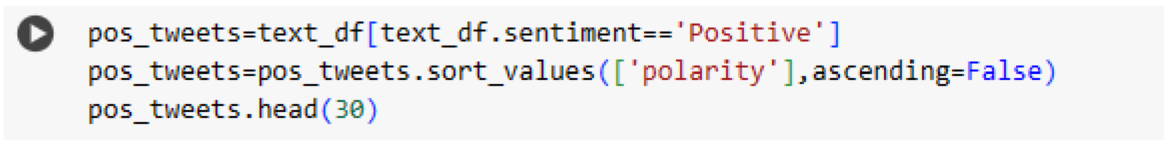


```
fig=plt.figure(figsize=(7,7))
colors=("yellowgreen","gold","red")
wp={'linewidth':2,'edgecolor':'black'}
tags = text_df['sentiment'].value_counts()
explode = (0.1, 0.1, 0.1)
tags.plot(kind='pie', autopct='%1.1f%%', shadow=True, colors=colors, wedgeprops=wp, explode=explode, label='')
```

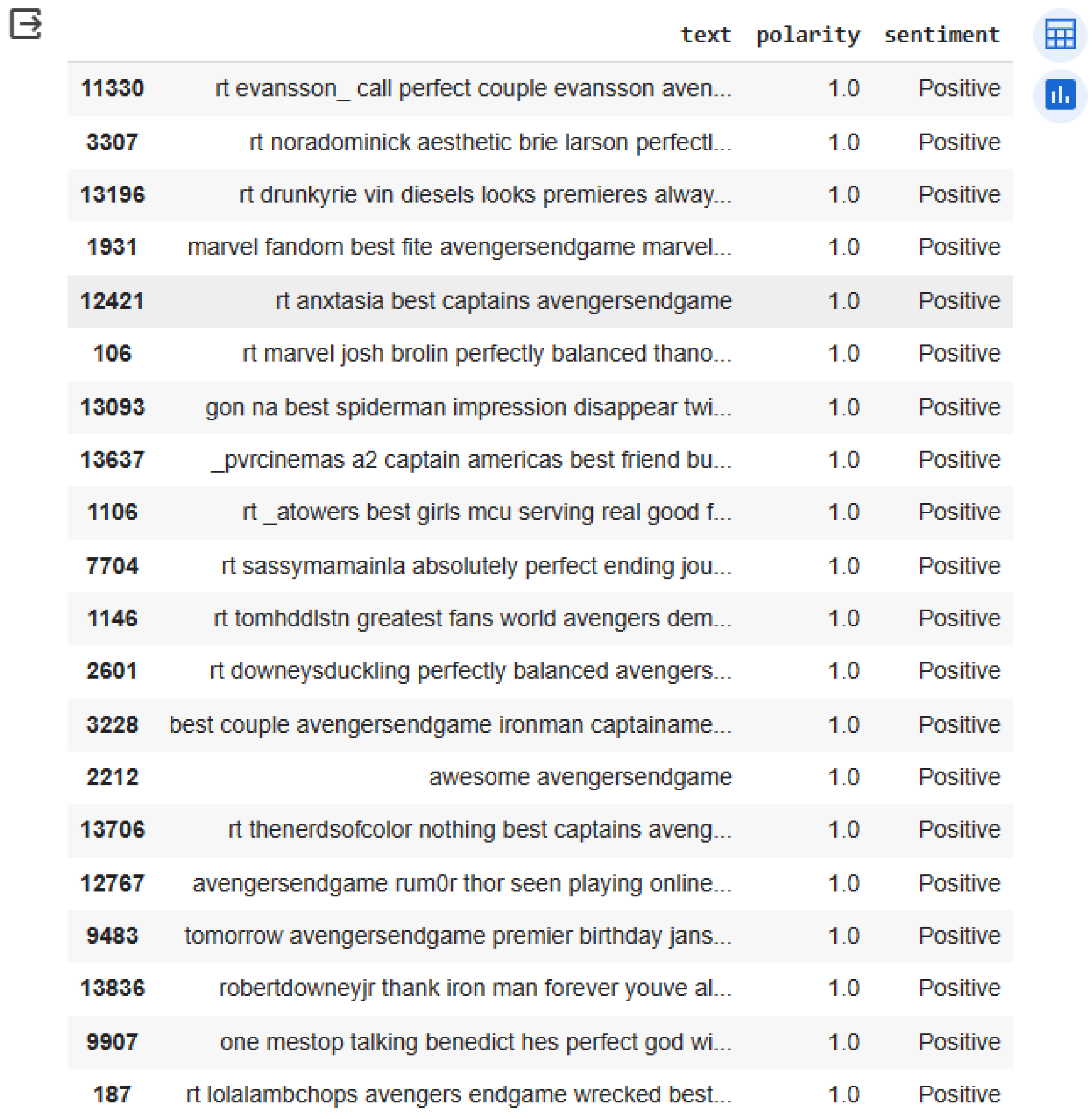


.

30.This code retrieves the top 20 positive tweets where the sentiment label is Positive.



```
pos_tweets=text_df[text_df.sentiment=='Positive']
pos_tweets=pos_tweets.sort_values(['polarity'],ascending=False)
pos_tweets.head(30)
```

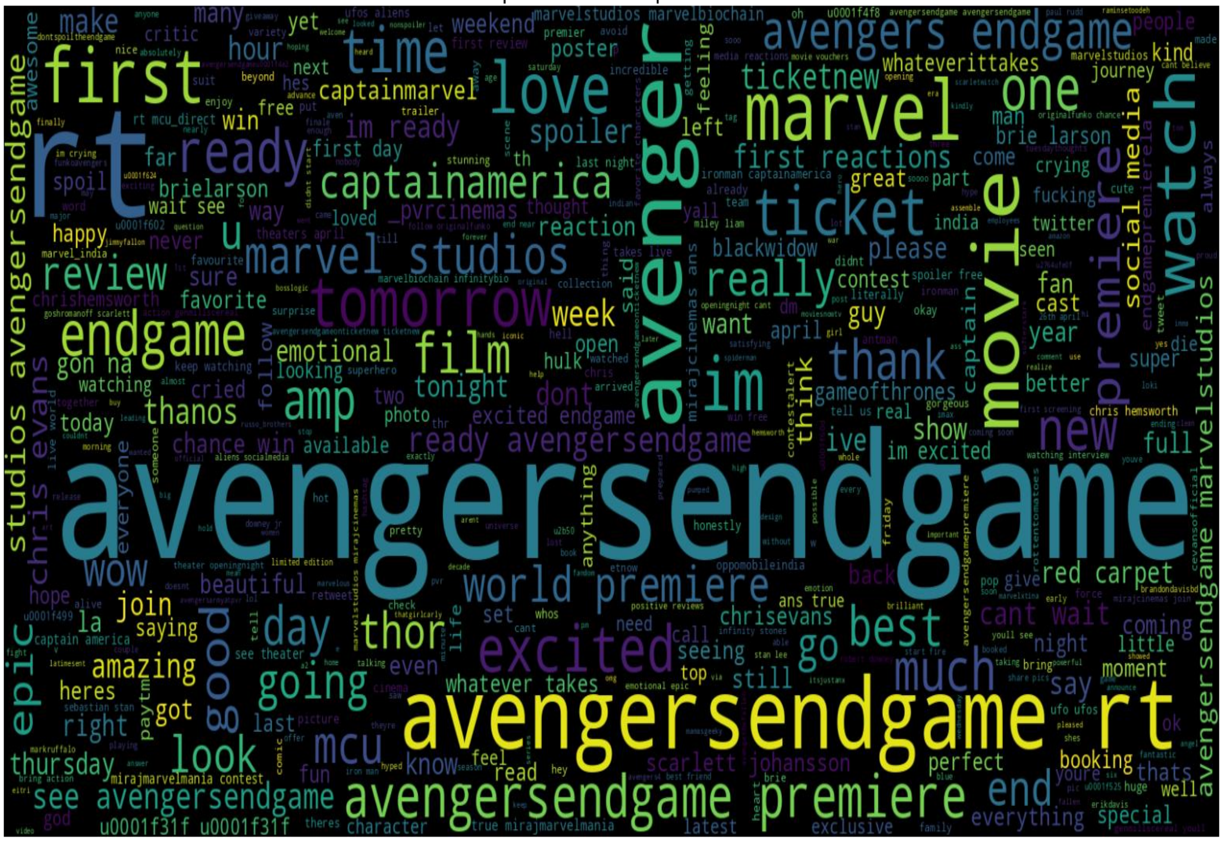


	text	polarity	sentiment
11330	rt evansson_ call perfect couple evansson aven...	1.0	Positive
3307	rt noradominick aesthetic brie larson perfectl...	1.0	Positive
13196	rt drunkyrie vin diesels looks premieres alway...	1.0	Positive
1931	marvel fandom best fite avengersendgame marvel...	1.0	Positive
12421	rt anxtasia best captains avengersendgame	1.0	Positive
106	rt marvel josh brolin perfectly balanced thano...	1.0	Positive
13093	gon na best spiderman impression disappear twi...	1.0	Positive
13637	_pvrcinemas a2 captain americas best friend bu...	1.0	Positive
1106	rt _atowers best girls mcu serving real good f...	1.0	Positive
7704	rt sassymamainla absolutely perfect ending jou...	1.0	Positive
1146	rt tomhddlstn greatest fans world avengers dem...	1.0	Positive
2601	rt downeysduckling perfectly balanced avengers...	1.0	Positive
3228	best couple avengersendgame ironman captainame...	1.0	Positive
2212	awesome avengersendgame	1.0	Positive
13706	rt thenerdsofcolor nothing best captains aveng...	1.0	Positive
12767	avengersendgame rum0r thor seen playing online...	1.0	Positive
9483	tomorrow avengersendgame premier birthday jans...	1.0	Positive
13836	robertdowneyjr thank iron man forever youve al...	1.0	Positive
9907	one mestop talking benedict hes perfect god wi...	1.0	Positive
187	rt lolalambchops avengers endgame wrecked best...	1.0	Positive

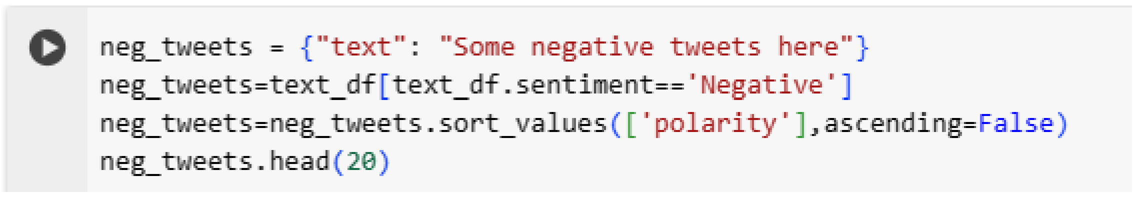
31.Displays the most frequent words in positive tweets in Word Cloud.

```python
text=' '.join([word for word in pos_tweets['text']])
plt.figure(figsize=(20,15),facecolor='None')
wordcloud=WordCloud(max_words=500,width=1600,height=800).generate(text)
plt.imshow(wordcloud,interpolation='bilinear')
plt.axis("off")
plt.title('Most frequent words in positive tweets',fontsize=19)
plt.show()
```

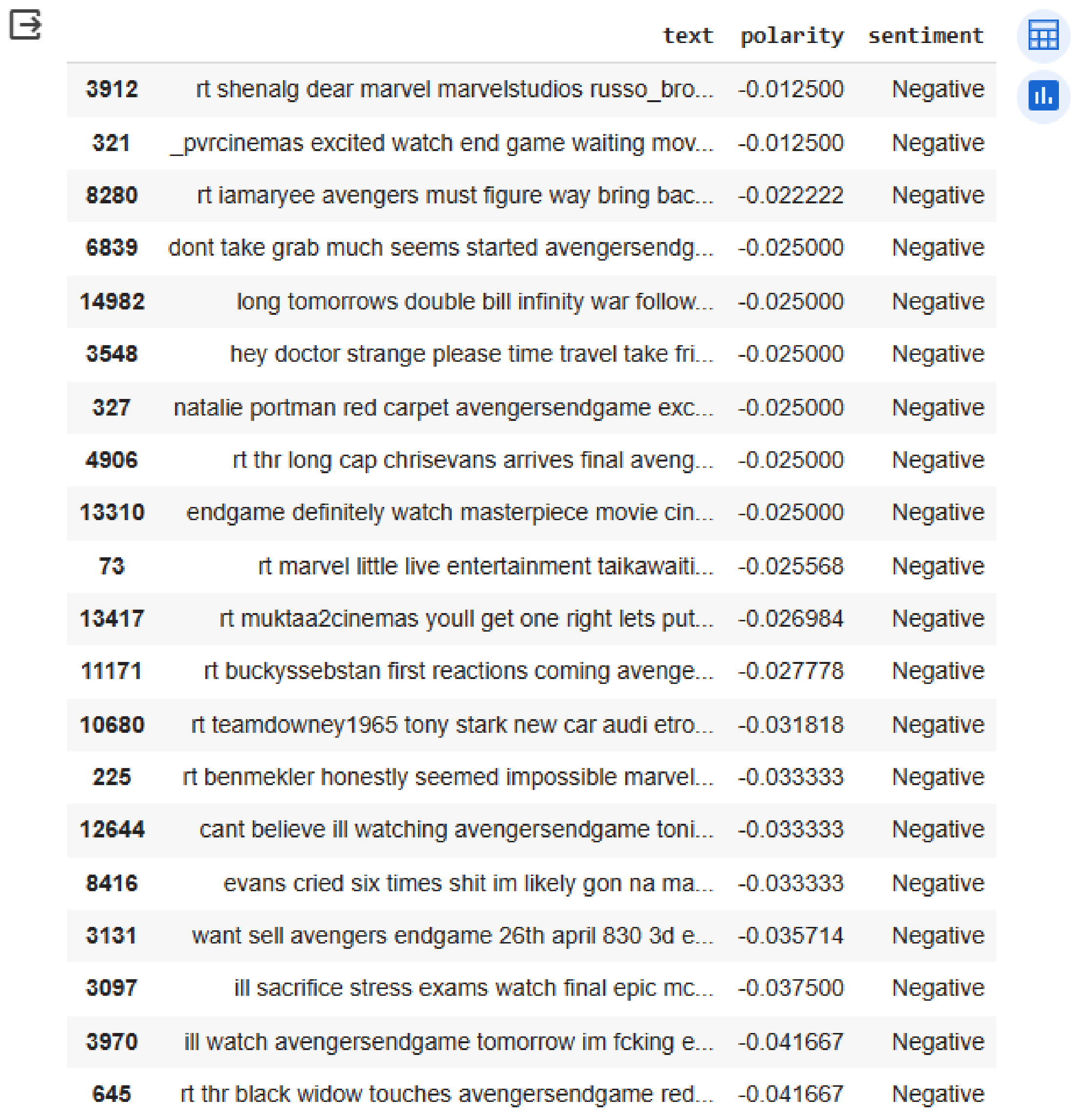
Most frequent words in positive tweets



32. This code retrieves the top 20 negative tweets where the sentiment label is Negative.

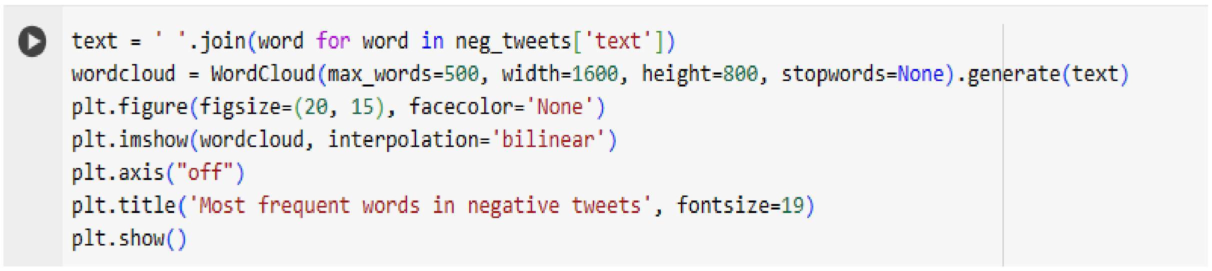


```
neg_tweets = {"text": "Some negative tweets here"}
neg_tweets=text_df[text_df.sentiment=='Negative']
neg_tweets=neg_tweets.sort_values(['polarity'],ascending=False)
neg_tweets.head(20)
```



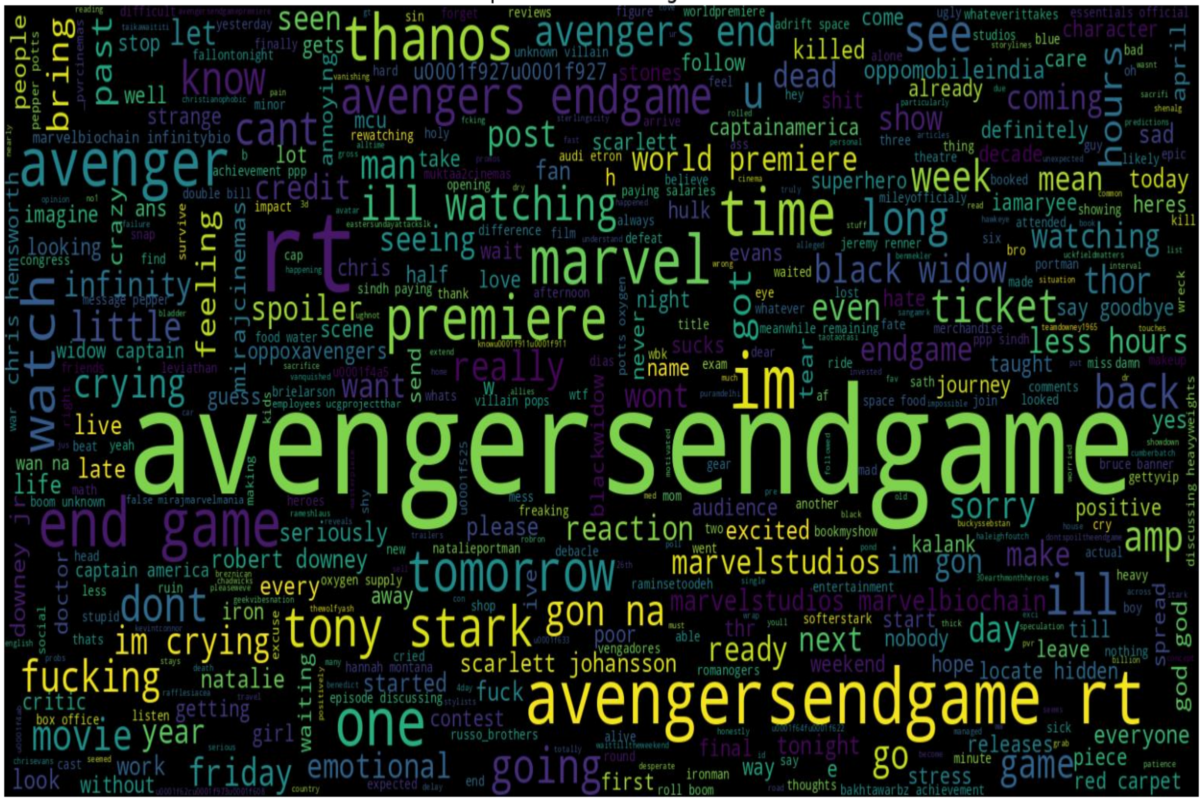
	text	polarity	sentiment
3912	rt shenalg dear marvel marvelstudios russo_bro...	-0.012500	Negative
321	_pvrcinemas excited watch end game waiting mov...	-0.012500	Negative
8280	rt iamaryee avengers must figure way bring bac...	-0.022222	Negative
6839	dont take grab much seems started avengersendg...	-0.025000	Negative
14982	long tomorrows double bill infinity war follow...	-0.025000	Negative
3548	hey doctor strange please time travel take fri...	-0.025000	Negative
327	natalie portman red carpet avengersendgame exc...	-0.025000	Negative
4906	rt thr long cap chrisevans arrives final aveng...	-0.025000	Negative
13310	endgame definitely watch masterpiece movie cin...	-0.025000	Negative
73	rt marvel little live entertainment taikawaiti...	-0.025568	Negative
13417	rt muktaa2cinemas youll get one right lets put...	-0.026984	Negative
11171	rt buckyssebstan first reactions coming avenge...	-0.027778	Negative
10680	rt teamdowney1965 tony stark new car audi etro...	-0.031818	Negative
225	rt benmekler honestly seemed impossible marvel...	-0.033333	Negative
12644	cant believe ill watching avengersendgame toni...	-0.033333	Negative
8416	evans cried six times shit im likely gon na ma...	-0.033333	Negative
3131	want sell avengers endgame 26th april 830 3d e...	-0.035714	Negative
3097	ill sacrifice stress exams watch final epic mc...	-0.037500	Negative
3970	ill watch avengersendgame tomorrow im fcking e...	-0.041667	Negative
645	rt thr black widow touches avengersendgame red...	-0.041667	Negative

33. Displays the most frequent words in negative tweets in Word Cloud.

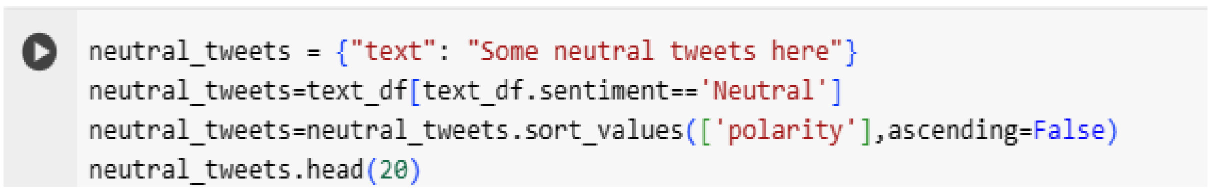


```
text = ' '.join(word for word in neg_tweets['text'])
wordcloud = WordCloud(max_words=500, width=1600, height=800, stopwords=None).generate(text)
plt.figure(figsize=(20, 15), facecolor='None')
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('Most frequent words in negative tweets', fontsize=19)
plt.show()
```

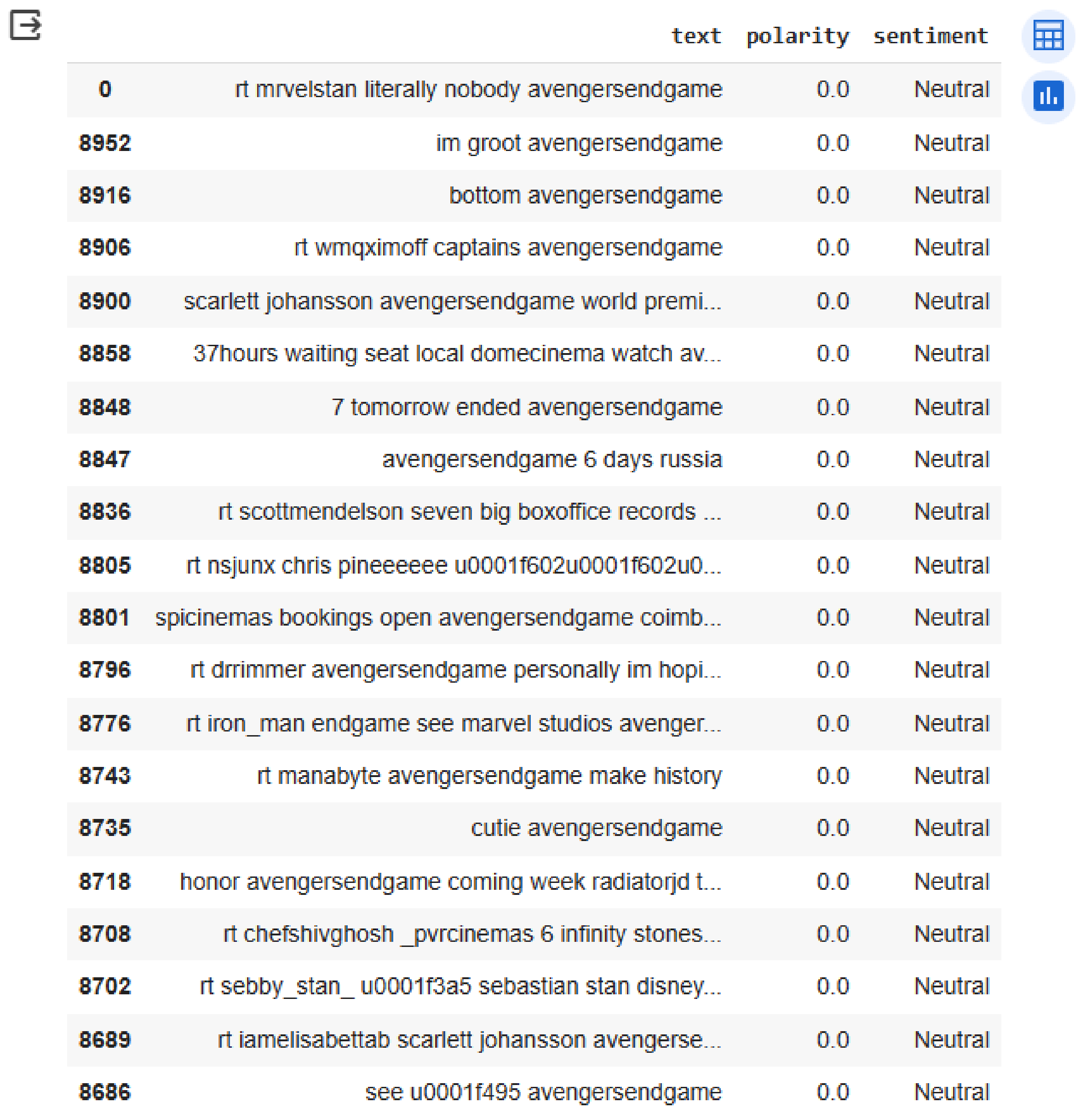
Most frequent words in negative tweets



34. This code retrieves the top 20 neutral tweets where the sentiment label is Neutral.

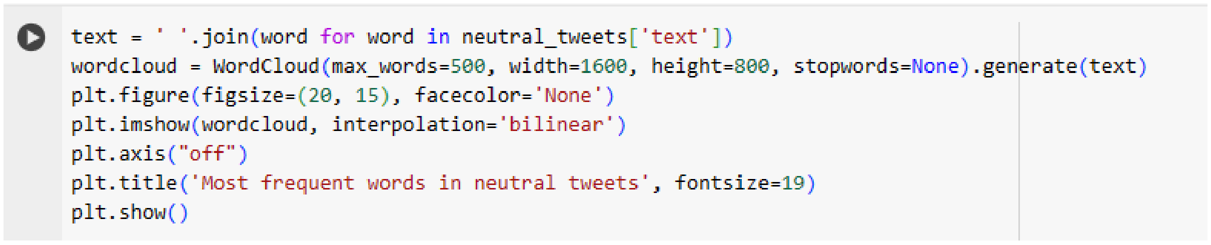


```
neutral_tweets = {"text": "Some neutral tweets here"}
neutral_tweets=text_df[text_df.sentiment=='Neutral']
neutral_tweets=neutral_tweets.sort_values(['polarity'],ascending=False)
neutral_tweets.head(20)
```



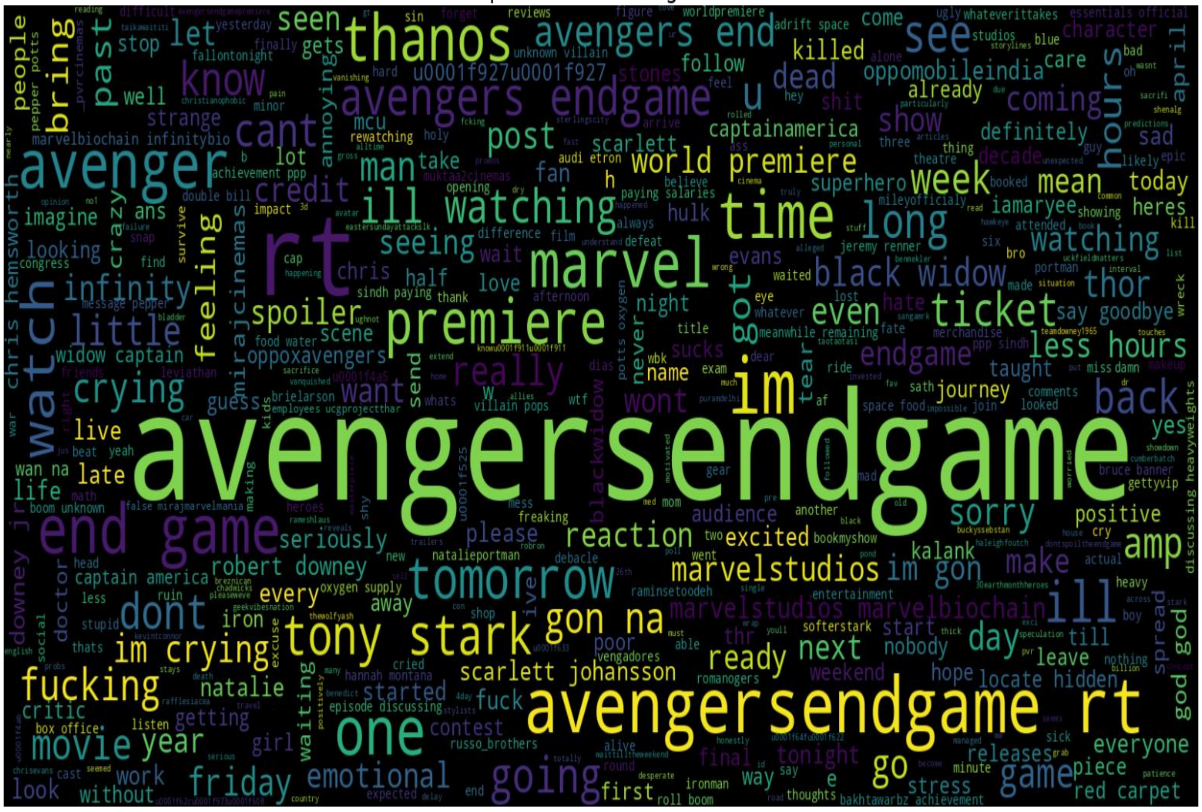
	text	polarity	sentiment
0	rt mrvelstan literally nobody avengersendgame	0.0	Neutral
8952	im groot avengersendgame	0.0	Neutral
8916	bottom avengersendgame	0.0	Neutral
8906	rt wmqximoff captains avengersendgame	0.0	Neutral
8900	scarlett johansson avengersendgame world premi...	0.0	Neutral
8858	37hours waiting seat local domecinema watch av...	0.0	Neutral
8848	7 tomorrow ended avengersendgame	0.0	Neutral
8847	avengersendgame 6 days russia	0.0	Neutral
8836	rt scottmendelson seven big boxoffice records ...	0.0	Neutral
8805	rt nsjunx chris pineeeeee u0001f602u0001f602u0...	0.0	Neutral
8801	spicinemas bookings open avengersendgame coimb...	0.0	Neutral
8796	rt drrimmer avengersendgame personally im hopi...	0.0	Neutral
8776	rt iron_man endgame see marvel studios avenger...	0.0	Neutral
8743	rt manabyte avengersendgame make history	0.0	Neutral
8735	cutie avengersendgame	0.0	Neutral
8718	honor avengersendgame coming week radiatorjd t...	0.0	Neutral
8708	rt chefshivghosh _pvrcinemas 6 infinity stones...	0.0	Neutral
8702	rt sebby_stan_ u0001f3a5 sebastian stan disney...	0.0	Neutral
8689	rt iamelisabettab scarlett johansson avengerse...	0.0	Neutral
8686	see u0001f495 avengersendgame	0.0	Neutral

35. Displays the most frequent words in neutral tweets in Word Cloud.



```
text = ' '.join(word for word in neutral_tweets['text'])
wordcloud = WordCloud(max_words=500, width=1600, height=800, stopwords=None).generate(text)
plt.figure(figsize=(20, 15), facecolor='None')
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('Most frequent words in neutral tweets', fontsize=19)
plt.show()
```

Most frequent words in negative tweets

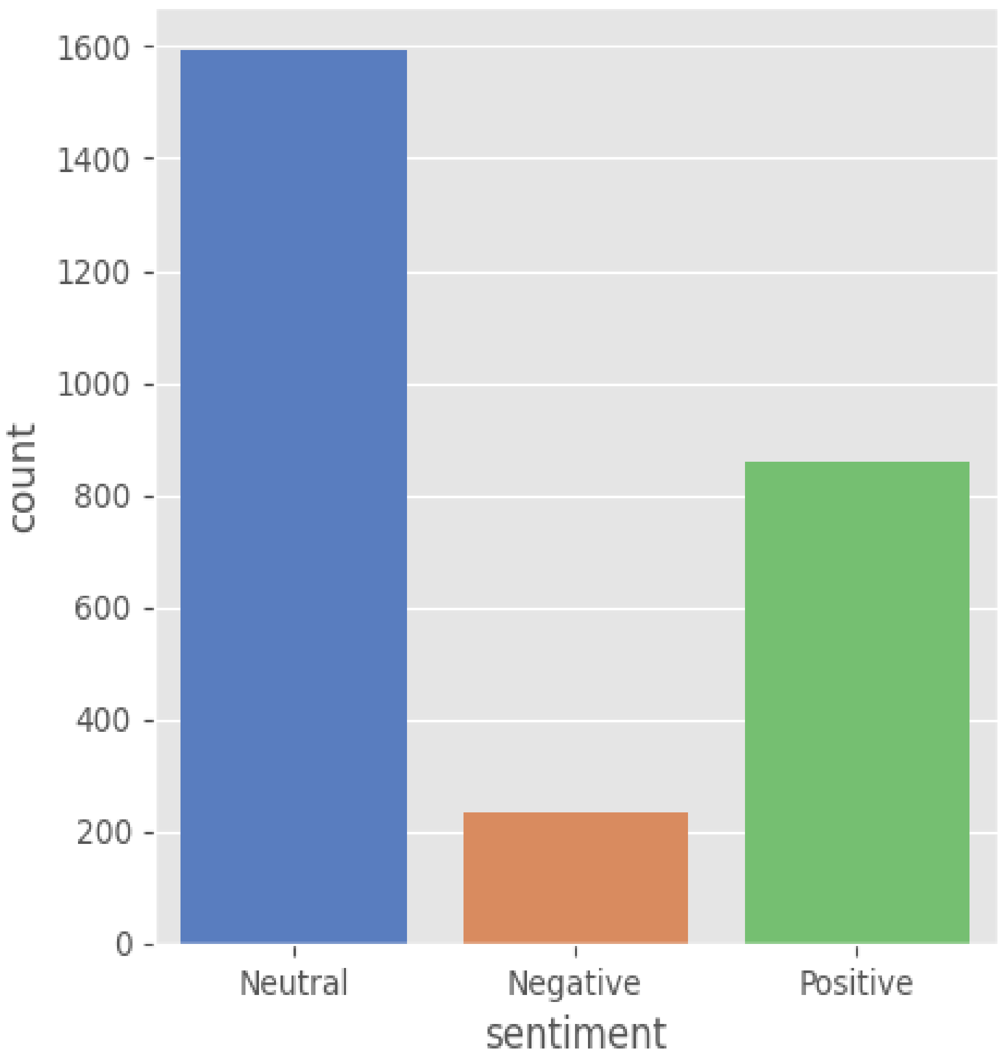


VISUALIZATION

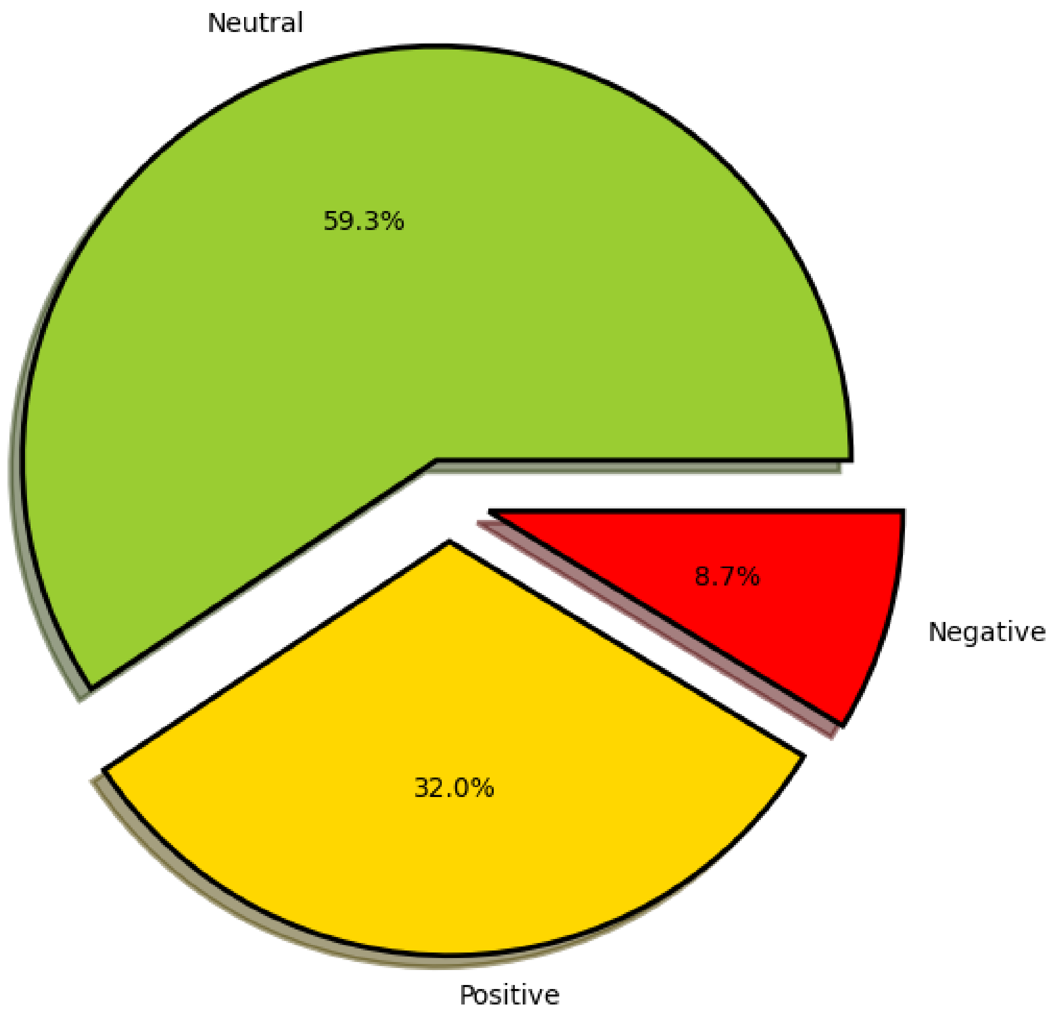
The major purpose of our project is to provide visualization by using matplotlib and seaborn libraries. By using these libraries we have shown visualization in bar graph, pie chart and the most frequent words used in positive, negative and neutral tweets are shown effectively by displaying in Word Cloud.

BAR GRAPH REPRESENTATION:

In this bar graph x-axis represents the sentiment and y-axis represents the count.



PIE CHART REPRESENTATION:


Neutral
Negative
Positive