Social Media analyzing

prepared by :

mohammad alowibdi

Supervised By :

Dr . chieck alloul

Dr . Mariam elmasry

introduction

Social media activity has huge impact on society And it must be consider as the most important crowd behavior rapid change till now and among These apps there is Twitter this app has huge number of users around the glob and we are going to explore it briefly because the company already know the reflection of the app on society they provide API to Crawl data from their server .

Content

- ✓ approach Analysis
- ✓ Import Necessary Dependencies
- ✓ Read and Load the Dataset
- ✓ Exploratory data analysis
- ✓ Data processing & cleaning
- ✓ Analyze sentiment of tweets
- ✓ Classifications

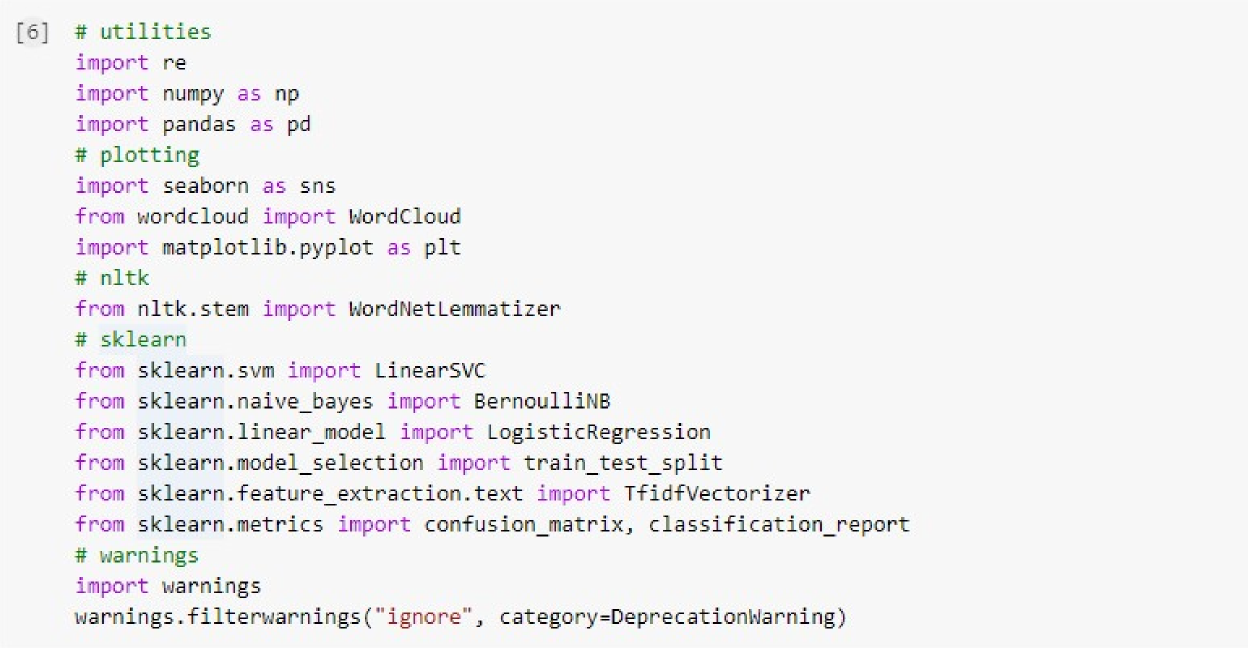
Approach Analysis

1. Get data from twitter account
2. Arrange data
3. Explore the data
4. Clean and process the data

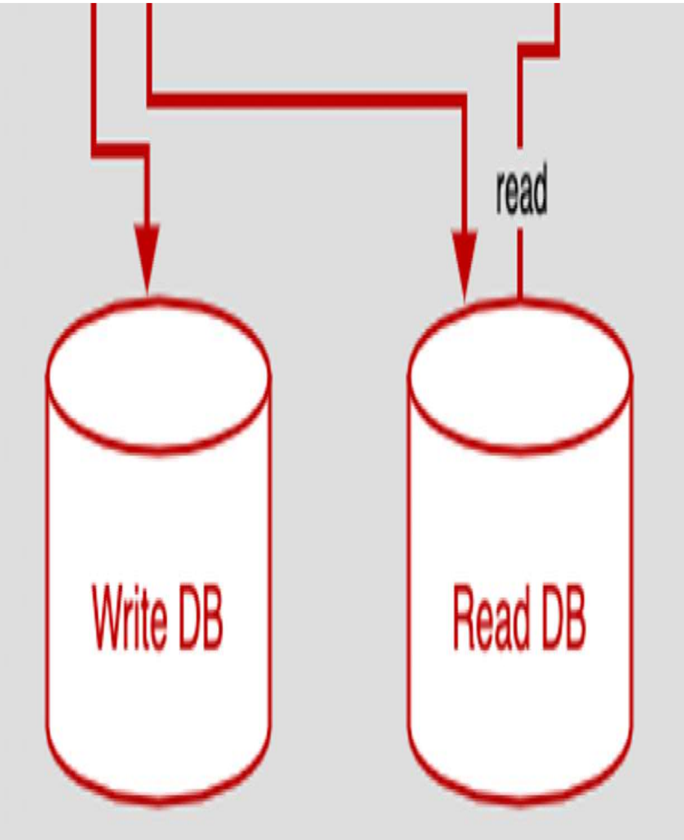
Results

1. Get sentiment of tweets
2. Get a lot of information by classification of tweets

Import Necessary Dependencies

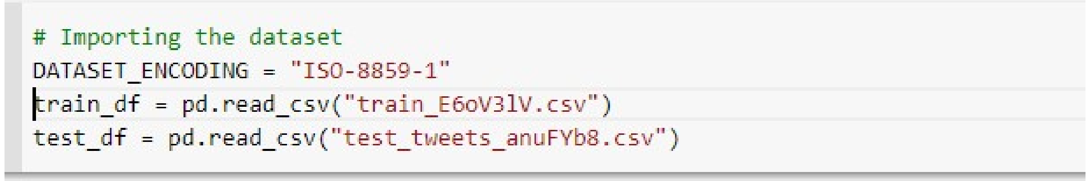


```
# utilities
import re
import numpy as np
import pandas as pd
# plotting
import seaborn as sns
from wordcloud import WordCloud
import matplotlib.pyplot as plt
# nltk
from nltk.stem import WordNetLemmatizer
# sklearn
from sklearn.svm import LinearSVC
from sklearn.naive_bayes import BernoulliNB
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix, classification_report
# warnings
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
```



Read and Load the Dataset

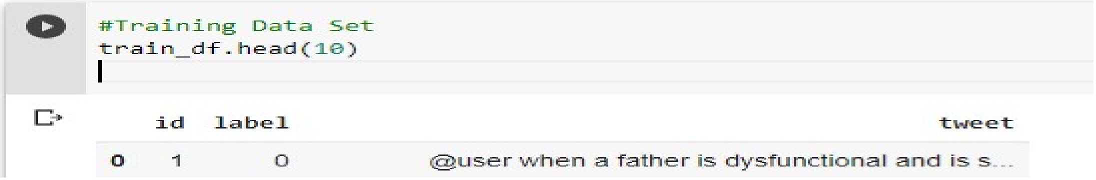
- I use google colab to make it easy to work from several machines
- Download demo data from kaggle.com
- Imports all data to pandas



```
# Importing the dataset
DATASET_ENCODING = "ISO-8859-1"
train_df = pd.read_csv("train_E6oV3lV.csv")
test_df = pd.read_csv("test_tweets_anuFYb8.csv")
```

Exploratory data analysis

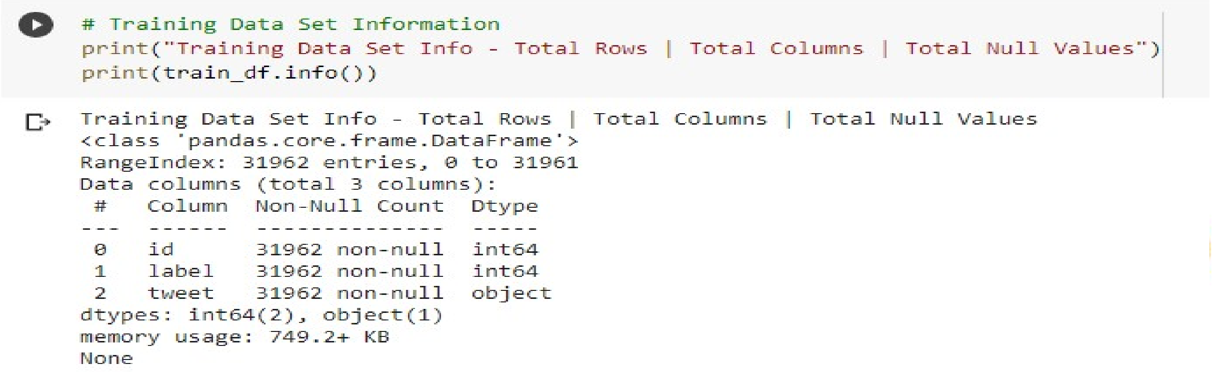
- I got the header first



```
#Training Data Set
train_df.head(10)
```

```
    id  label                                         tweet
0    1      0    @user when a father is dysfunctional and is s...
```

- Then get the information of the data set to identify data types

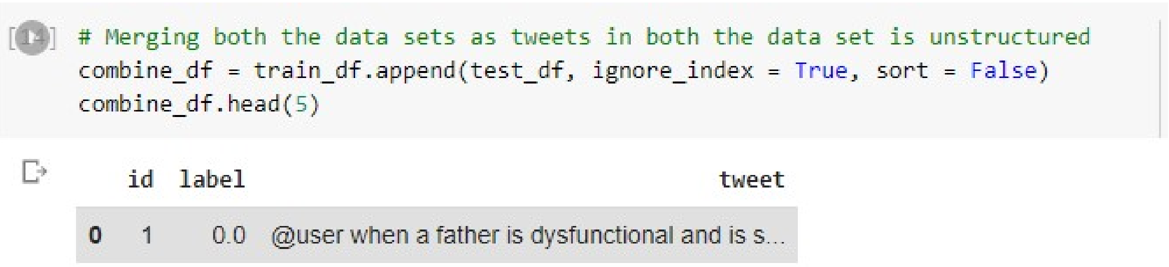


```
# Training Data Set Information
print("Training Data Set Info - Total Rows | Total Columns | Total Null Values")
print(train_df.info())
```

```
Training Data Set Info - Total Rows | Total Columns | Total Null Values
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31962 entries, 0 to 31961
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   id      31962 non-null  int64 
 1   label   31962 non-null  int64 
 2   tweet   31962 non-null  object
dtypes: int64(2), object(1)
memory usage: 749.2+ KB
None
```

Exploratory data analysis

- structure the tweets, remove the unwanted words, replace the misspelled words with the correct ones, replace the abbreviation with full words



```
# Merging both the data sets as tweets in both the data set is unstructured
combine_df = train_df.append(test_df, ignore_index = True, sort = False)
combine_df.head(5)
```

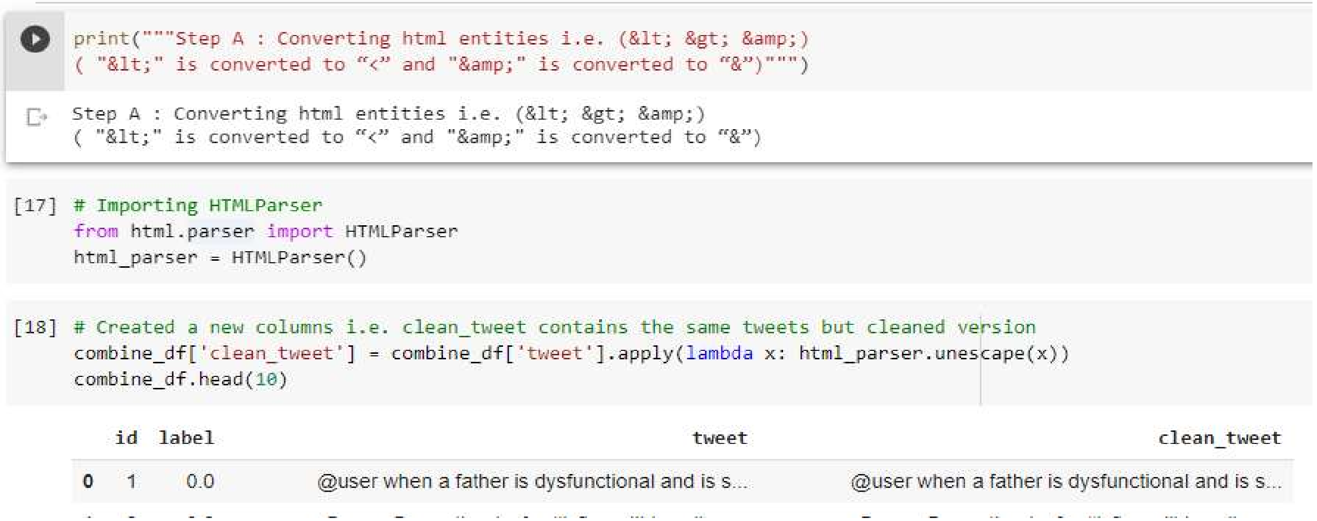
```
      id  label                                            tweet
0     1     0.0   @user when a father is dysfunctional and is s...
```

Data processing & cleaning

- Step A : Converting html entities
- Step B : Removing "@user" from all the tweets
- Step C : Changing all the tweets into lowercase
- Step D : Apostrophe Lookup
- Step E : Short Word Lookup
- Step F : Emoticon Lookup
- Step H : Replacing Special Characters with space
- Step I : Replacing Numbers (integers) with space
- Step J : Removing words whom length is 1

Data processing & cleaning

Step A : Converting html entities



```
print("""Step A : Converting html entities i.e. (&lt; &gt; &amp;)
( "&lt;" is converted to “<” and "&amp;" is converted to “&”)""")
```

```
Step A : Converting html entities i.e. (&lt; &gt; &amp;)
( "&lt;" is converted to “<” and "&amp;" is converted to “&”)
```

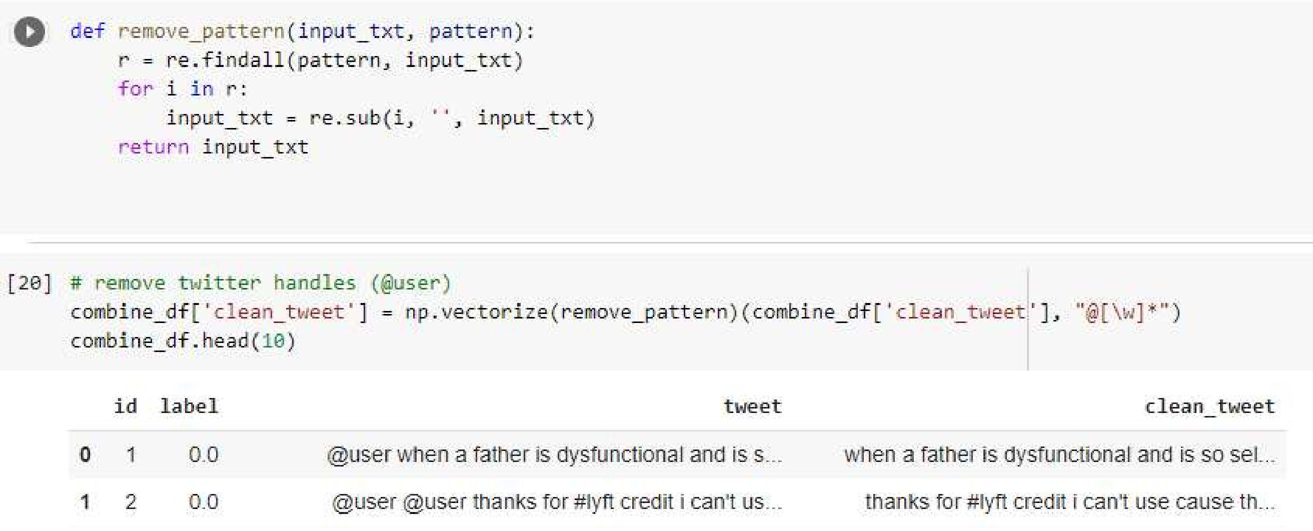
```
[17] # Importing HTMLParser
     from html.parser import HTMLParser
     html_parser = HTMLParser()
```

```
[18] # Created a new columns i.e. clean_tweet contains the same tweets but cleaned version
     combine_df['clean_tweet'] = combine_df['tweet'].apply(lambda x: html_parser.unescape(x))
     combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	@user when a father is dysfunctional and is s...

Data processing & cleaning

Step B : Removing "@user" from all the tweets



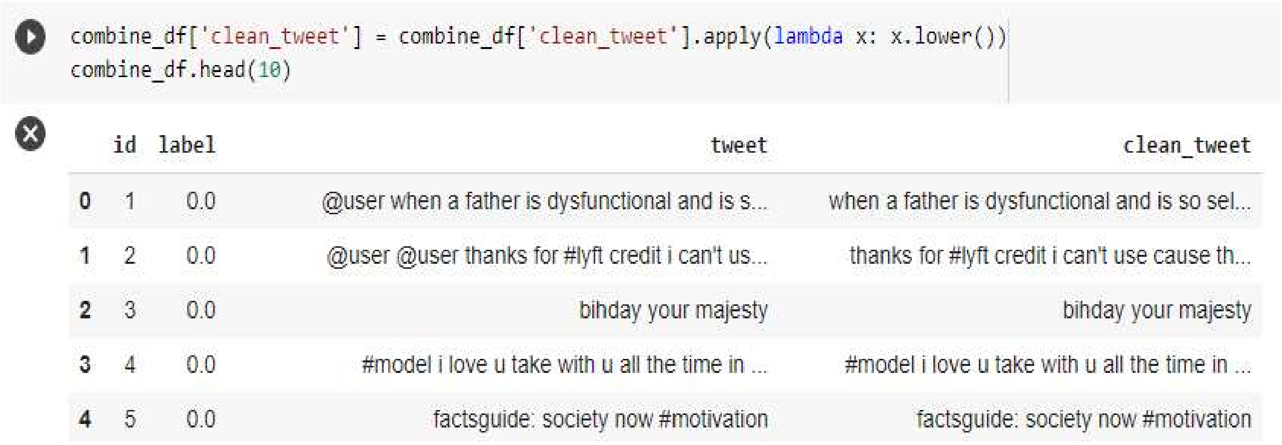
```
def remove_pattern(input_txt, pattern):
    r = re.findall(pattern, input_txt)
    for i in r:
        input_txt = re.sub(i, '', input_txt)
    return input_txt
```

```
[20] # remove twitter handles (@user)
     combine_df['clean_tweet'] = np.vectorize(remove_pattern)(combine_df['clean_tweet'], "@[\w]*")
     combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i can't use cause th...

Data processing & cleaning

- Step C : Changing all the tweets into lowercase



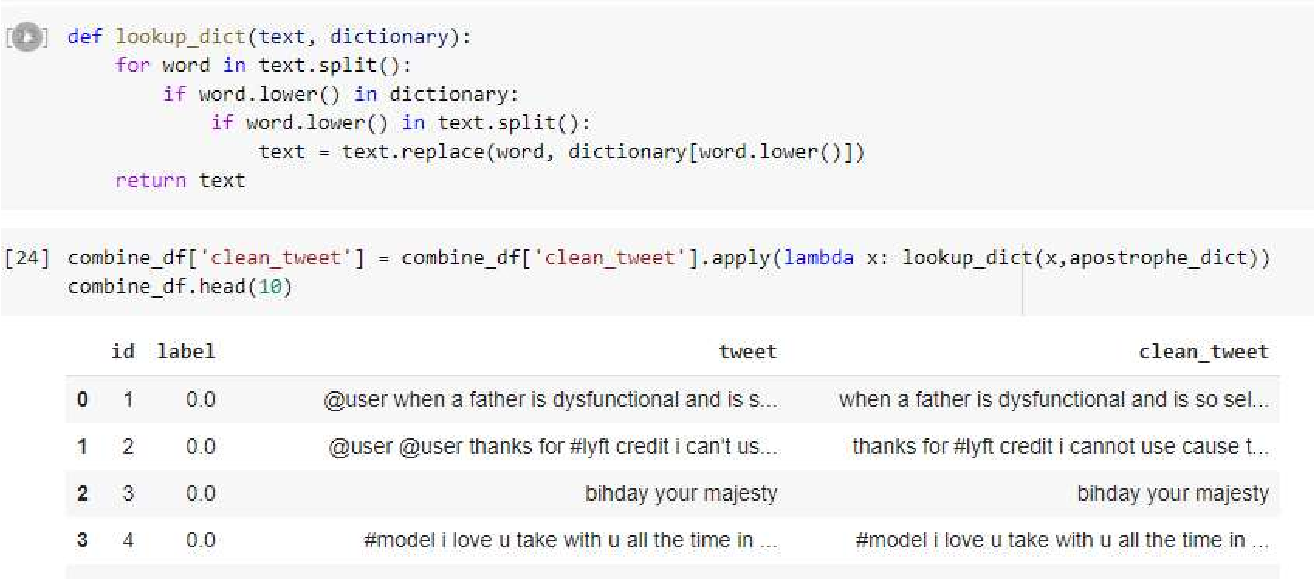
```
combine_df['clean_tweet'] = combine_df['clean_tweet'].apply(lambda x: x.lower())
combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i can't use cause th...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	#model i love u take with u all the time in ...
4	5	0.0	factsguide: society now #motivation	factsguide: society now #motivation

Data processing & cleaning

Step D : Apostrophe Lookup

First we create Apostrophe Dictionary then use it



```
def lookup_dict(text, dictionary):
    for word in text.split():
        if word.lower() in dictionary:
            if word.lower() in text.split():
                text = text.replace(word, dictionary[word.lower()])
    return text
```

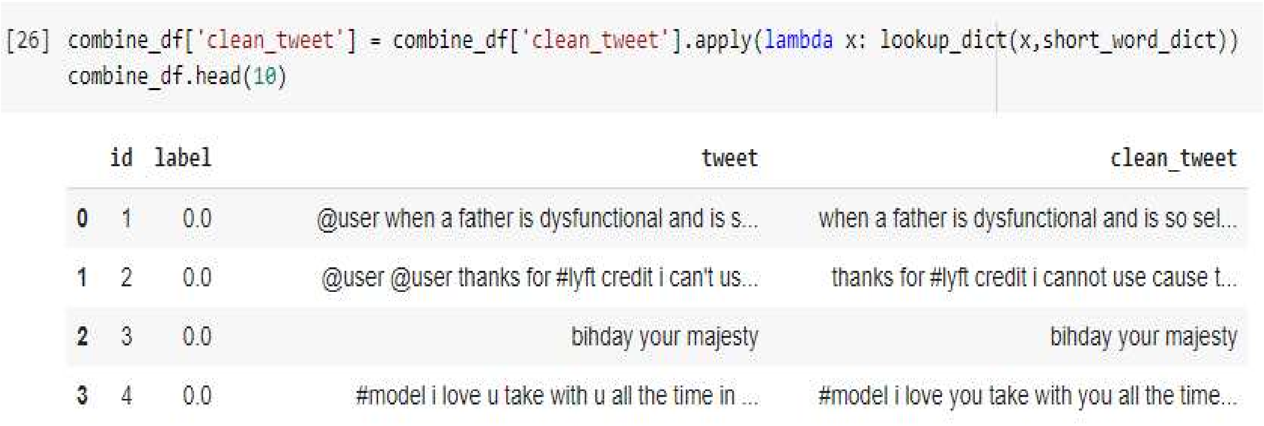
```
[24] combine_df['clean_tweet'] = combine_df['clean_tweet'].apply(lambda x: lookup_dict(x,apostrophe_dict))
     combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i cannot use cause t...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	#model i love u take with u all the time in ...

Data processing & cleaning

Step E : Short Word Lookup

First we create most known Short Words Dictionary then use it



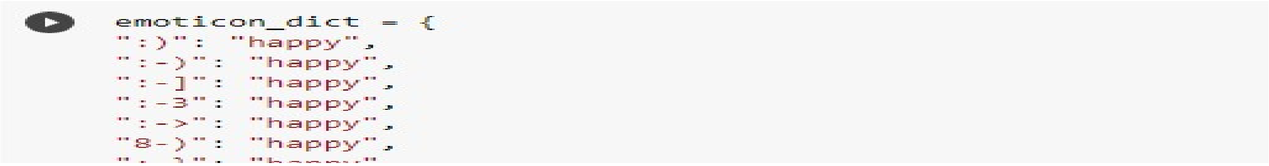
```
[26] combine_df['clean_tweet'] = combine_df['clean_tweet'].apply(lambda x: lookup_dict(x,short_word_dict))
     combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i cannot use cause t...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	#model i love you take with you all the time...

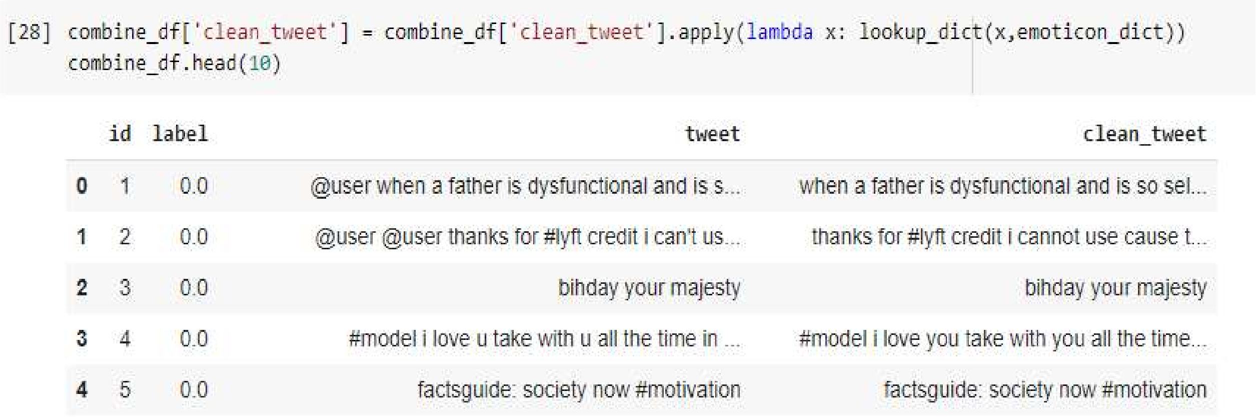
Data processing & cleaning

Step F : Emoticon Lookup

First we create most known Emoticon Dictionary then use it



```
emoticon_dict = {
":)":  "happy",
":-)":  "happy",
":-]":  "happy",
":-3":  "happy",
":->":  "happy",
"8-)":  "happy",
```

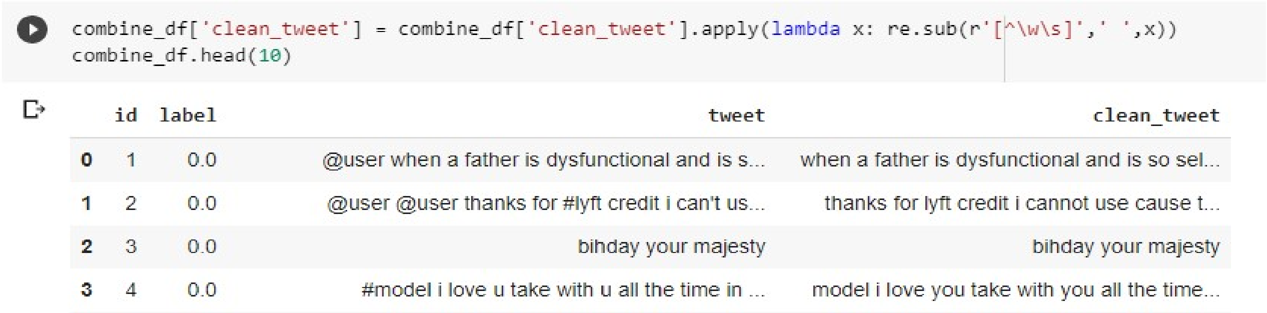


```
[28] combine_df['clean_tweet'] = combine_df['clean_tweet'].apply(lambda x: lookup_dict(x,emoticon_dict))
     combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i cannot use cause t...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	#model i love you take with you all the time...
4	5	0.0	factsguide: society now #motivation	factsguide: society now #motivation

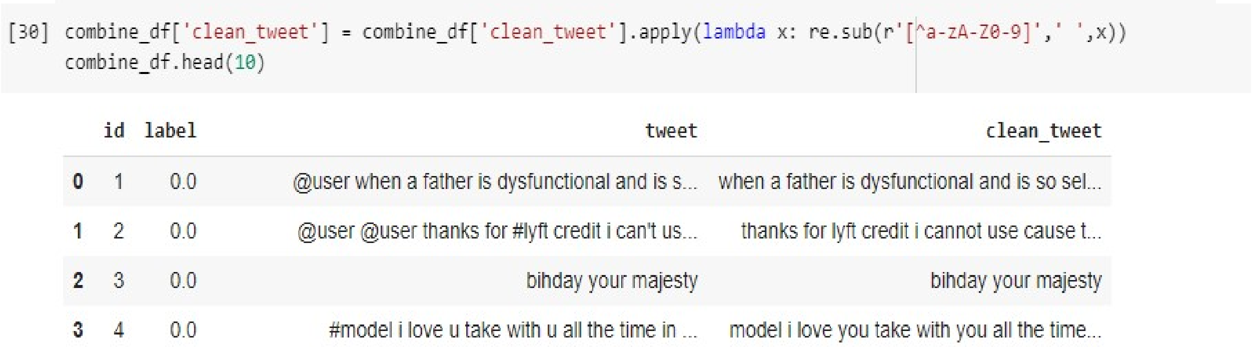
Data processing & cleaning

Step H : Replacing Special Characters with space



```
combine_df['clean_tweet'] = combine_df['clean_tweet'].apply(lambda x: re.sub(r'[^\w\s]',' ',x))
combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for lyft credit i cannot use cause t...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	model i love you take with you all the time...

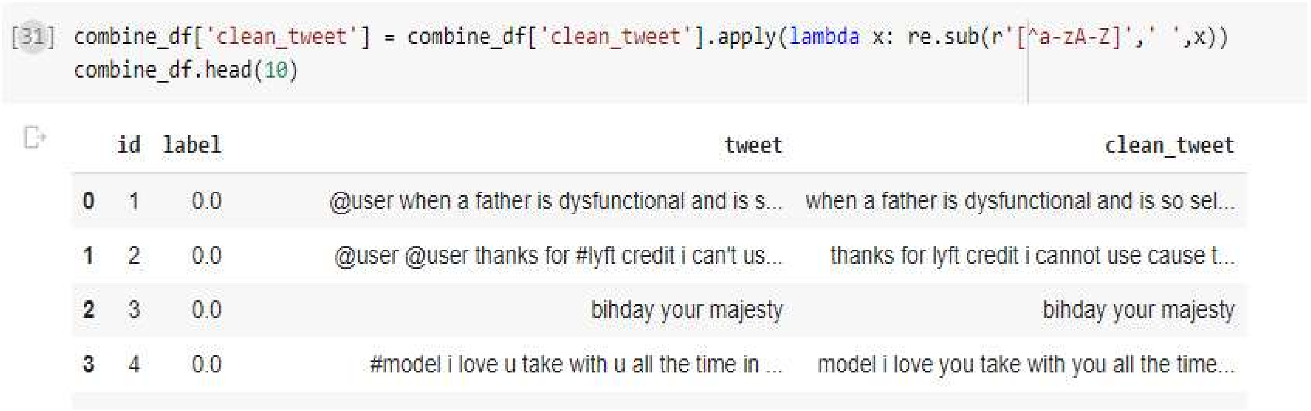


```
[30] combine_df['clean_tweet'] = combine_df['clean_tweet'].apply(lambda x: re.sub(r'[^a-zA-Z0-9]',' ',x))
     combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for lyft credit i cannot use cause t...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	model i love you take with you all the time...

Data processing & cleaning

Step I : Replacing Numbers (integers) with space

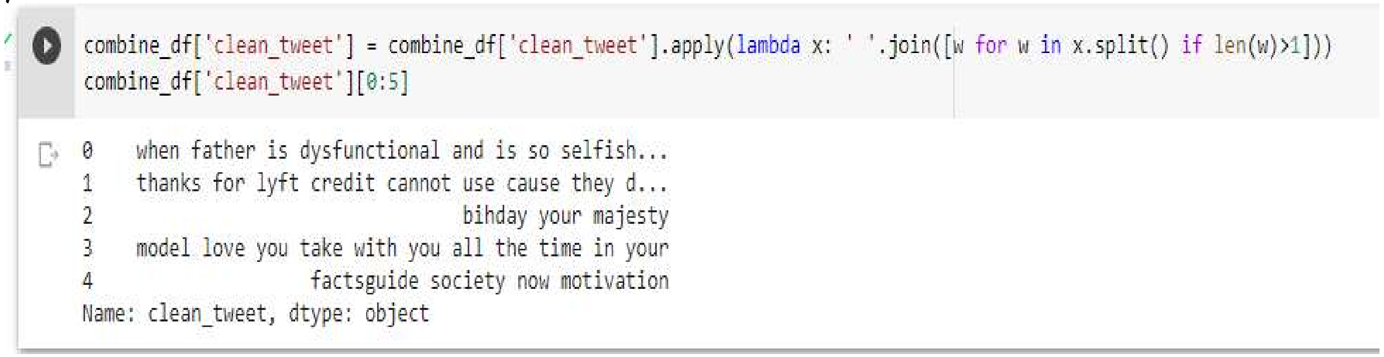


```
[31] combine_df['clean_tweet'] = combine_df['clean_tweet'].apply(lambda x: re.sub(r'[^a-zA-Z]',' ',x))
     combine_df.head(10)
```

	id	label	tweet	clean_tweet
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for lyft credit i cannot use cause t...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	model i love you take with you all the time...

Data processing & cleaning

Step J : Removing words whom length is 1

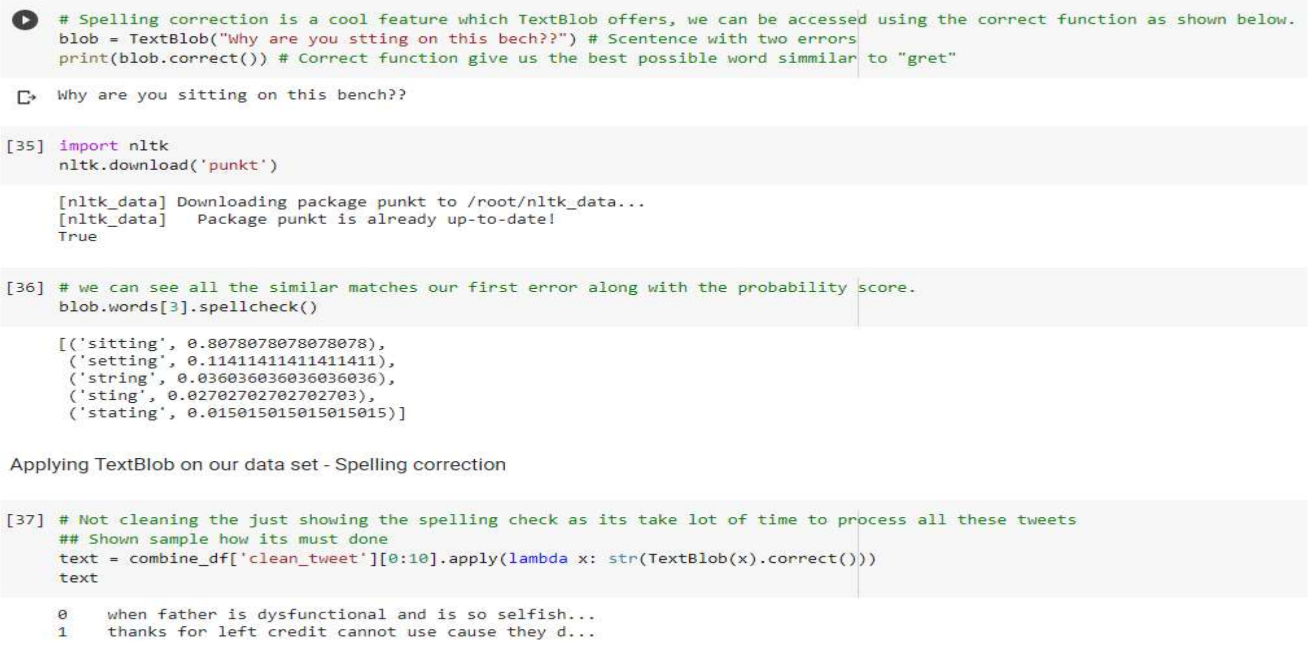


```
combine_df['clean_tweet'] = combine_df['clean_tweet'].apply(lambda x: ' '.join([w for w in x.split() if len(w)>1]))
combine_df['clean_tweet'][0:5]
```

```
0    when father is dysfunctional and is so selfish...
1    thanks for lyft credit cannot use cause they d...
2                                bihday your majesty
3    model love you take with you all the time in your
4                   factsguide society now motivation
Name: clean_tweet, dtype: object
```

Data processing & cleaning

Last step spell checking



```
# Spelling correction is a cool feature which TextBlob offers, we can be accessed using the correct function as shown below.
blob = TextBlob("Why are you stting on this bech??") # Scentence with two errors
print(blob.correct()) # Correct function give us the best possible word simmilar to "gret"
```

```
Why are you sitting on this bench??
```

```
[35] import nltk
     nltk.download('punkt')
```

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

```
[36] # we can see all the similar matches our first error along with the probability score.
     blob.words[3].spellcheck()
```

```
[('sitting', 0.8078078078078078),
 ('setting', 0.11411411411411411),
 ('string', 0.036036036036036036),
 ('sting', 0.02702702702702703),
 ('stating', 0.015015015015015015)]
```

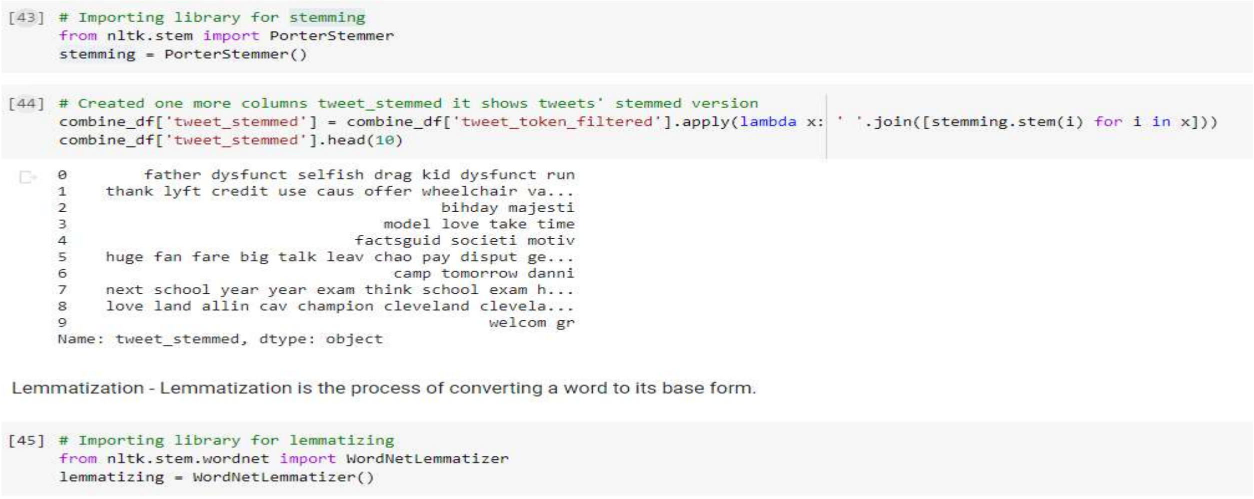
Applying TextBlob on our data set - Spelling correction

```
[37] # Not cleaning the just showing the spelling check as its take lot of time to process all these tweets
     ## Shown sample how its must done
     text = combine_df['clean_tweet'][0:10].apply(lambda x: str(TextBlob(x).correct()))
     text
```

```
0    when father is dysfunctional and is so selfish...
1    thanks for left credit cannot use cause they d...
```

Data processing & cleaning

Stemming and Lemmatization



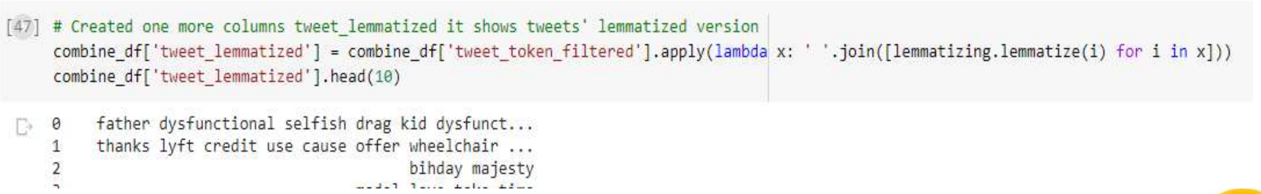
```
[43] # Importing library for stemming
     from nltk.stem import PorterStemmer
     stemming = PorterStemmer()
```

```
[44] # Created one more columns tweet_stemmed it shows tweets' stemmed version
     combine_df['tweet_stemmed'] = combine_df['tweet_token_filtered'].apply(lambda x: ' '.join([stemming.stem(i) for i in x]))
     combine_df['tweet_stemmed'].head(10)
```

```
     0          father dysfunct selfish drag kid dysfunct run
     1     thank lyft credit use caus offer wheelchair va...
     2                                     bihday majesti
     3                               model love take time
     4                            factsguid societi motiv
     5     huge fan fare big talk leav chao pay disput ge...
     6                                camp tomorrow danni
     7     next school year year exam think school exam h...
     8     love land allin cav champion cleveland clevela...
     9                                           welcom gr
     Name: tweet_stemmed, dtype: object
```

Lemmatization - Lemmatization is the process of converting a word to its base form.

```
[45] # Importing library for lemmatizing
     from nltk.stem.wordnet import WordNetLemmatizer
     lemmatizing = WordNetLemmatizer()
```

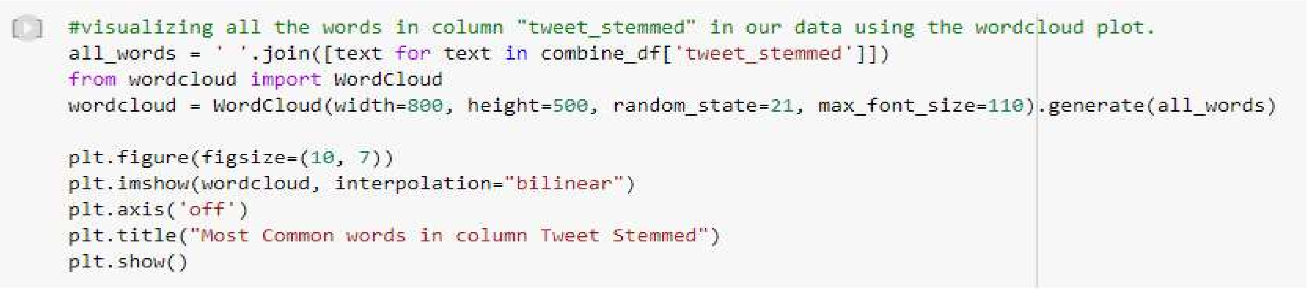


```
[47] # Created one more columns tweet_lemmatized it shows tweets' lemmatized version
     combine_df['tweet_lemmatized'] = combine_df['tweet_token_filtered'].apply(lambda x: ' '.join([lemmatizing.lemmatize(i) for i in x]))
     combine_df['tweet_lemmatized'].head(10)
```

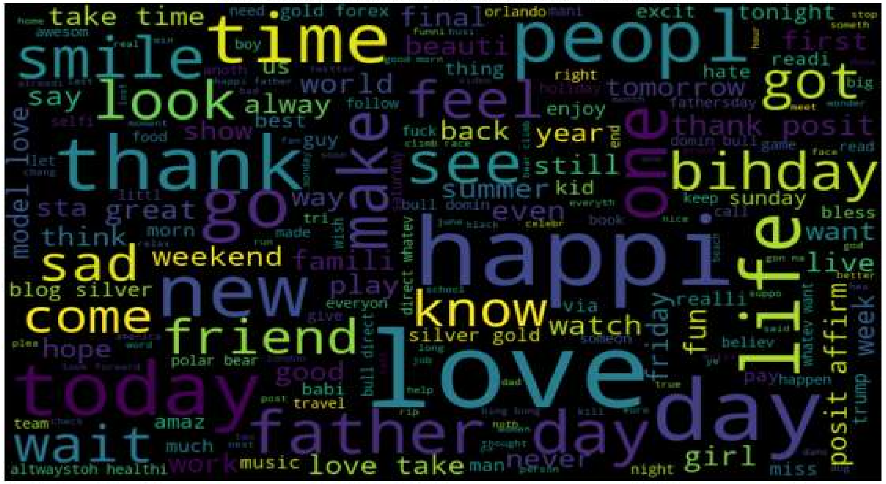
```
     0     father dysfunctional selfish drag kid dysfunct...
     1     thanks lyft credit use cause offer wheelchair ...
     2                                     bihday majesty
     3                               model love take time
```

Analyze sentiment of tweets

Most common words in tweet column



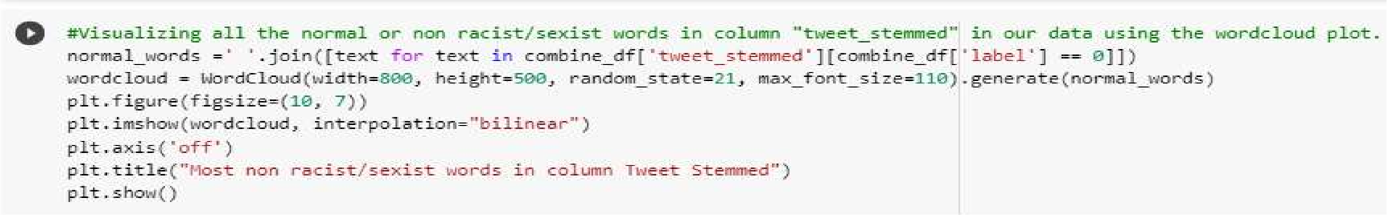
```
#visualizing all the words in column "tweet_stemmed" in our data using the wordcloud plot.
all_words = ' '.join([text for text in combine_df['tweet_stemmed']])
from wordcloud import WordCloud
wordcloud = WordCloud(width=800, height=500, random_state=21, max_font_size=110).generate(all_words)

plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.title("Most Common words in column Tweet Stemmed")
plt.show()
```



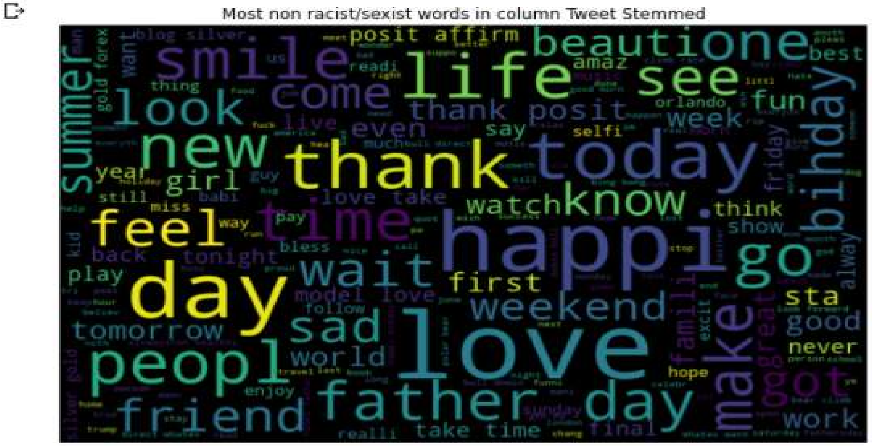
Most Common words in column Tweet Stemmed

Analyze sentiment of tweets

Most common words in non racist/sexist tweets



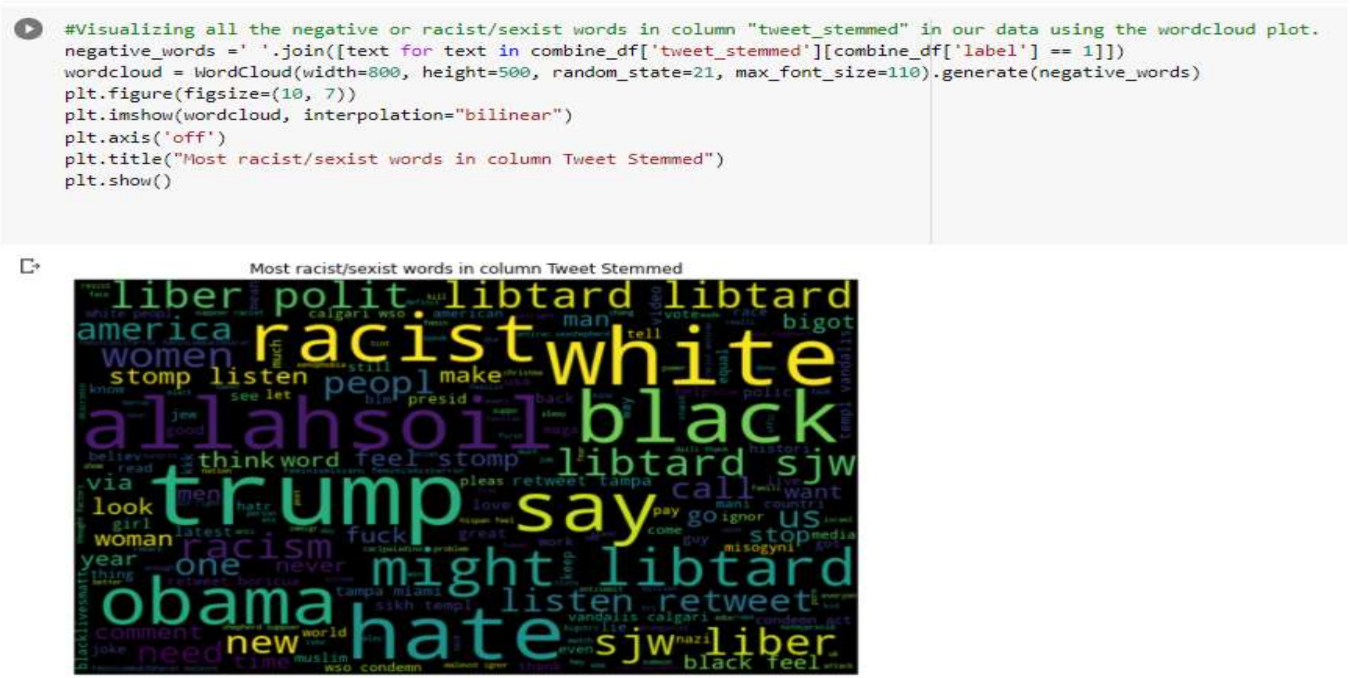
```
#Visualizing all the normal or non racist/sexist words in column "tweet_stemmed" in our data using the wordcloud plot.
normal_words =' '.join([text for text in combine_df['tweet_stemmed'][combine_df['label'] == 0]])
wordcloud = WordCloud(width=800, height=500, random_state=21, max_font_size=110).generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.title("Most non racist/sexist words in column Tweet Stemmed")
plt.show()
```



Most non racist/sexist words in column Tweet Stemmed

Analyze sentiment of tweets

Most common words in racist/sexist tweets



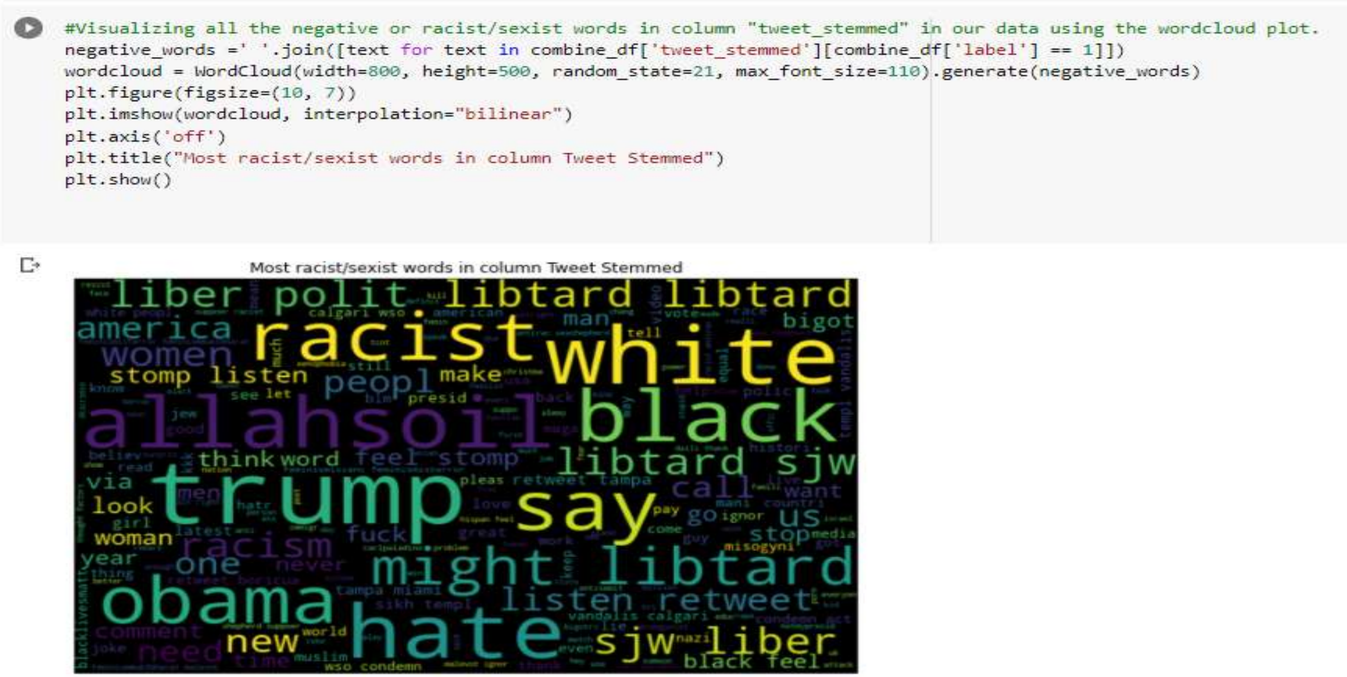
```
#Visualizing all the negative or racist/sexist words in column "tweet_stemmed" in our data using the wordcloud plot.
negative_words =' '.join([text for text in combine_df['tweet_stemmed'][combine_df['label'] == 1]])
wordcloud = WordCloud(width=800, height=500, random_state=21, max_font_size=110).generate(negative_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.title("Most racist/sexist words in column Tweet Stemmed")
plt.show()
```

Most racist/sexist words in column Tweet Stemmed

liber polit libtard libtard
america racist white bigot
women stomp listen peopl make
allahsoil black
think word feel stomp libtard sjw
trump say call
look girl woman racism fuck stop media
year one might libtard
obama listen retweet
new world hate sjw liber
black feel

Analyze sentiment of tweets

Most common words in racist/sexist tweets



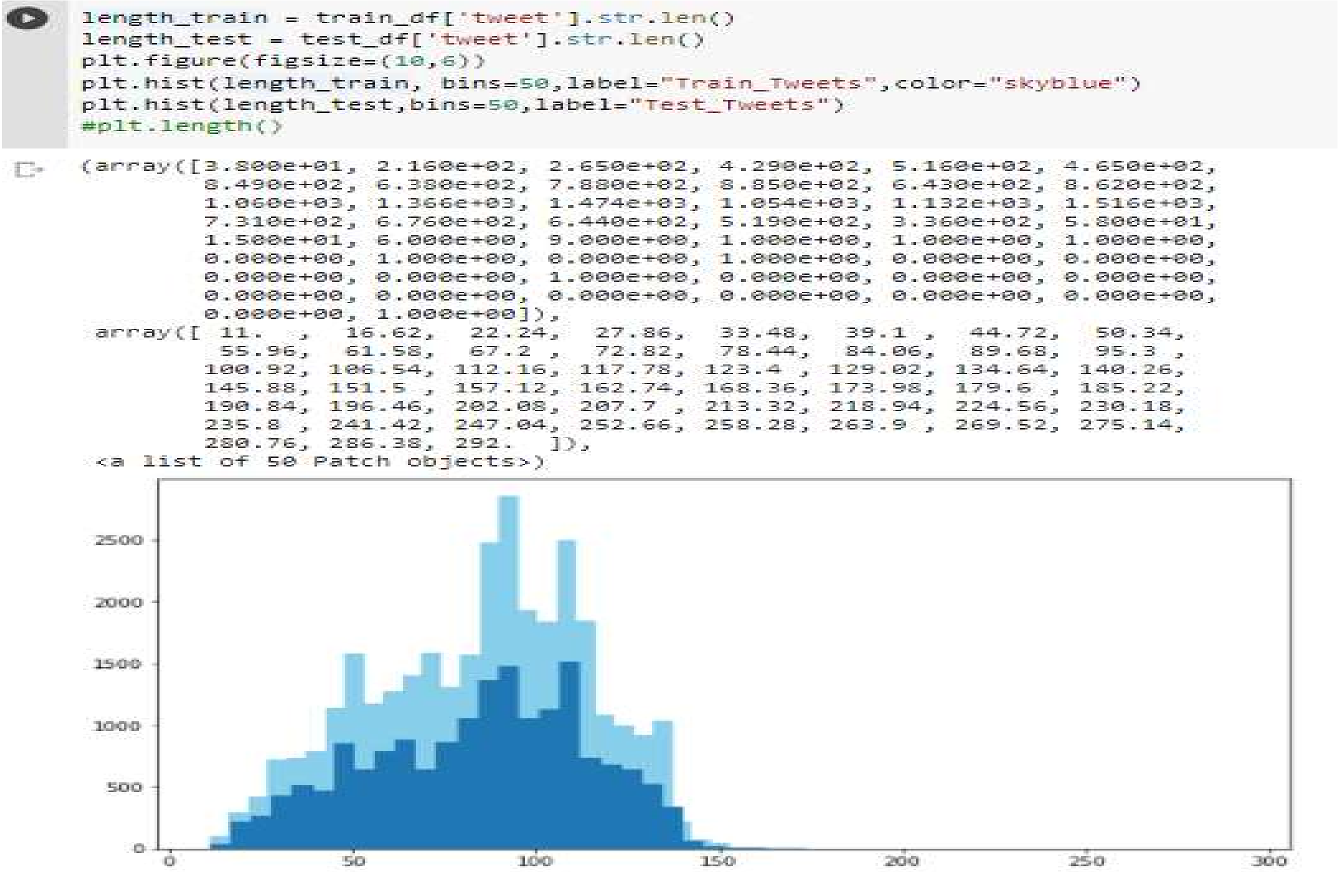
```
#Visualizing all the negative or racist/sexist words in column "tweet_stemmed" in our data using the wordcloud plot.
negative_words =' '.join([text for text in combine_df['tweet_stemmed'][combine_df['label'] == 1]])
wordcloud = WordCloud(width=800, height=500, random_state=21, max_font_size=110).generate(negative_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.title("Most racist/sexist words in column Tweet Stemmed")
plt.show()
```

Most racist/sexist words in column Tweet Stemmed

liber polit libtard libtard
america racist white bigot
women stomp listen peopl make
allahsoil black
think word feel stomp libtard sjw
trump say call
look girl woman racism
year one might libtard
obama listen retweet
new hate sjw liber black feel

Classifications

- length of the Tweets in our Train data



```
length_train = train_df['tweet'].str.len()
length_test = test_df['tweet'].str.len()
plt.figure(figsize=(10,6))
plt.hist(length_train, bins=50,label="Train_Tweets",color="skyblue")
plt.hist(length_test,bins=50,label="Test_Tweets")
#plt.length()
```

```
(array([3.800e+01, 2.160e+02, 2.650e+02, 4.290e+02, 5.160e+02, 4.650e+02,
        8.490e+02, 6.380e+02, 7.880e+02, 8.850e+02, 6.430e+02, 8.620e+02,
        1.060e+03, 1.366e+03, 1.474e+03, 1.054e+03, 1.132e+03, 1.516e+03,
        7.310e+02, 6.760e+02, 6.440e+02, 5.190e+02, 3.360e+02, 5.800e+01,
        1.500e+01, 6.000e+00, 9.000e+00, 1.000e+00, 1.000e+00, 1.000e+00,
        0.000e+00, 1.000e+00, 0.000e+00, 1.000e+00, 0.000e+00, 0.000e+00,
        0.000e+00, 0.000e+00, 1.000e+00, 0.000e+00, 0.000e+00, 0.000e+00,
        0.000e+00, 0.000e+00, 0.000e+00, 0.000e+00, 0.000e+00, 0.000e+00,
        0.000e+00, 1.000e+00]),
 array([ 11.  ,  16.62,  22.24,  27.86,  33.48,  39.1 ,  44.72,  50.34,
         55.96,  61.58,  67.2 ,  72.82,  78.44,  84.06,  89.68,  95.3 ,
        100.92, 106.54, 112.16, 117.78, 123.4 , 129.02, 134.64, 140.26,
        145.88, 151.5 , 157.12, 162.74, 168.36, 173.98, 179.6 , 185.22,
        190.84, 196.46, 202.08, 207.7 , 213.32, 218.94, 224.56, 230.18,
        235.8 , 241.42, 247.04, 252.66, 258.28, 263.9 , 269.52, 275.14,
        280.76, 286.38, 292.   ]),
 <a list of 50 Patch objects>)
```

THANK YOU!