

























# -\*- coding: utf-8 -\*-

"""Untitled2.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1nU0kbFDr1A99KzrvlzIaEmhgtA0FvxA-

"""

import tweepy

from textblob import TextBlob

from wordcloud import WordCloud

import pandas as pd

import numpy as np

import re

import matplotlib.pyplot as plt

plt.style.use('fivethirtyeight')

Consumer\_Key="WVAQeQyjglQaR2A9ELVrmAYUN"

Consumer\_Secret\_Key="IfEos3qgam3mmdjQc8dLCbp7mLpZJxhcvXM45SbhTLsfGKRoqh"

Access\_Token="1242004678839156738-nFCCCt43nfgB4DT1CqMVL9sECHKWii"

Access\_Token\_Secret="8mtfNKAmEXgEJaeznzzj0uSjMRmwgzFbTts4DqnASffvR"

#create a auth\_key

auth = tweepy.OAuthHandler(Consumer\_Key,Consumer\_Secret\_Key)

#create a access\_token

auth.set\_access\_token(Access\_Token,Access\_Token\_Secret)

#create a api while passing in the information

api = tweepy.API(auth)

#extract tweet from sushant\_sinha

posts = api.user\_timeline(screen\_name='SushantBSinha', count = 100, lang ="en", tweet\_mode="extended")

# Print the last 5 tweets

print("Show the 5 recent tweets:\n")

i=1;

for tweet in posts[0:5]:

print(str(i) + ') ' +tweet.full\_text + '"\n');

i=i+1;

df = pd.DataFrame([tweet.full\_text for tweet in posts], columns=['Tweets'])

# Show the first 5 rows of data

df .head()

#clean the data

#Removing @mentions

def cleanTxt(text):

text=re.sub(r'@[A-Za-z0-9]+', '', text)

# Removing '#' hash tag

text=re.sub(r'#', '', text)

# Removing Retweet

text=re.sub(r'RT[\s]+', '', text)

# Removing hyperlink

text=re.sub(r'https?:\/\/\S+', '', text)

return text

#clean the data and applying the function

df['Tweets']= df['Tweets'].apply(cleanTxt)

#displaying the clean data

df

# Create a function to get the subjectivity

def getSubjectivity(text):

return TextBlob(text).sentiment.subjectivity

# Create a function to get the polarity

def getPolarity(text):

return TextBlob(text).sentiment.polarity

# Create two new columns ‘Subjectivity’ & ‘Polarity’

df['Subjectivity'] = df['Tweets'].apply(getSubjectivity)

df['Polarity'] = df['Tweets'].apply(getPolarity)

# Show the new dataframe with columns ‘Subjectivity’ & ‘Polarity’

df

allWords = ' '.join([twts for twts in df['Tweets']])

wordCloud = WordCloud(width=500, height=300, random\_state=21, max\_font\_size=110).generate(allWords)

plt.imshow(wordCloud, interpolation="bilinear")

plt.axis('off')

plt.show()

# Create a function to compute negative (-1), neutral (®) and positive (+1) analysis

def getAnalysis(score):

if score < 0:

return 'Negative'

elif score ==0:

return 'Neutral'

else:

return 'Positive'

df['Analysis'] = df['Polarity'].apply(getAnalysis)

# Show the dataframe

df

print('Printing negative tweets:\n')

j=1

sortedDF = df.sort\_values(by=[ 'Polarity' ],ascending=False) #Sort the tweets

for i in range(0, sortedDF.shape[0] ):

if( sortedDF['Analysis'][i] == 'Negative'):

print(str(j) + ') '+sortedDF['Tweets'][i])

print()

j=j+1

print( 'Printing Positive tweets:\n')

j=1

sortedDF = df.sort\_values(by=['Polarity'],ascending=True) #Sort the tweets

for i in range(0, sortedDF.shape[0] ):

if( sortedDF['Analysis'][i] == 'Positive'):

print(str(j) + ') '+sortedDF[ 'Tweets'][i])

print()

j=j+1

#plotting the subjectivity and plarity

plt.figure(figsize=(8,6))

for i in range(0, df.shape[0]):

plt.scatter(df['Polarity'][i], df['Subjectivity'][i], color='Blue')

# plt.scatter(x,y,color)

plt.title('Sentiment Analysis')

plt.xlabel('Polarity')

plt.ylabel('Subjectivity')

plt.show()

#get the percentage of positive tweet

ptweets = df[df.Analysis == 'Positive']

ptweets = ptweets['Tweets']

#all the positive tweets

ptweets

round( (ptweets.shape[0] / df.shape[0]) \* 100 , 1)

#get the negative tweets

ntweets = df[df.Analysis == 'Negative']

ntweets = ntweets['Tweets' ]

ntweets

round( (ntweets.shape[0] / df.shape[0]) \* 100 , 1)

# Show the value counts

df['Analysis'].value\_counts()

# Plotting and visualizing the counts

plt.title('Sentiment Analysis')

plt.xlabel('Sentiment')

plt.ylabel('Counts')

df['Analysis'].value\_counts().plot(kind = 'bar')

plt.show()