import re

# Python's built-in "re" module provides excellent support for [regular expressions](https://regular-expressions.mobi/tutorial.html)

import tweepy

# **Tweepy** is open-sourced, hosted on GitHub and enables Python to communicate with Twitter platform and use its API

from tweepy import OauthHandler

# Tweepy supports accessing Twitter via Basic Authentication and the newer method, OAuth. Twitter has stopped accepting Basic Authentication so OAuth is now the only way to use the Twitter API.

from textblob import TextBlob

#**TextBlob** is a Python (2 and 3) library for processing textual data

class TwitterClient(object):

#Generic Twitter Class for sentiment analysis.

def \_\_init\_\_(self):

# keys and tokens from the Twitter Dev Console

#In order to access the Twitter, that is to get recent tweets and twitter followers count, you

need the four keys such as Consumer Key, Consumer Secret, Acess token, Access Token

Secret.

consumer\_key = 'wKs4oj5j7EpRh5WuOjz5smPj6'

consumer\_secret = 'XVnMSaQqkAnH3l9CssedNNgQehOpkvlpRaIwDTTMjU7Byxyw6o'

access\_token = '897431456842039296-si6Y2NTTNrGVsVsJS5hX6bs46dGLurV'

access\_token\_secret = 'eC8VhlQZ1hbrnN3PuDBf7i4ZO5Iq6oHaFaxRdofXibeXf'

# attempt authentication

try:

# create OAuthHandler object

self.auth = OAuthHandler(consumer\_key, consumer\_secret)

# set access token and secret

self.auth.set\_access\_token(access\_token, access\_token\_secret)

# create tweepy API object to fetch tweets

self.api = tweepy.API(self.auth)

except:

print("Error: Authentication Failed")

def clean\_tweet(self, tweet):

'''

Utility function to clean tweet text by removing links, special characters

using simple regex statements.

'''

return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)", " ", tweet).split())

# The **re**.**sub**() function in the **re** module can be used to replace substrings. The syntax for **re**.**sub**() is **re**.**sub**(pattern,repl,string)

def get\_tweet\_sentiment(self, tweet):

'''

Utility function to classify sentiment of passed tweet

using textblob's sentiment method

'''

# create TextBlob object of passed tweet text

analysis = TextBlob(self.clean\_tweet(tweet))

# set sentiment

if analysis.sentiment.polarity > 0:

return 'positive'

elif analysis.sentiment.polarity == 0:

return 'neutral'

else:

return 'negative'

def get\_tweets(self, query, count = 10):

'''

Main function to fetch tweets and parse them.

'''

# empty list to store parsed tweets

tweets = []

try:

# call twitter api to fetch tweets

fetched\_tweets = self.api.search(q = query, count = count)

# parsing tweets one by one

for tweet in fetched\_tweets:

# empty dictionary to store required params of a tweet

parsed\_tweet = {}

# saving text of tweet

parsed\_tweet['text'] = tweet.text

# saving sentiment of tweet

parsed\_tweet['sentiment'] = self.get\_tweet\_sentiment(tweet.text)

# appending parsed tweet to tweets list

if tweet.retweet\_count > 0:

# if tweet has retweets, ensure that it is appended only once

if parsed\_tweet not in tweets:

tweets.append(parsed\_tweet)

else:

tweets.append(parsed\_tweet)

# return parsed tweets

return tweets

except tweepy.TweepError as e:

# print error (if any)

print("Error : " + str(e))

def main():

# creating object of TwitterClient Class

api = TwitterClient()

# calling function to get tweets

tweets = api.get\_tweets(query = 'Rahul Gandhi', count = 200)

# picking positive tweets from tweets

ptweets = [tweet for tweet in tweets if tweet['sentiment'] == 'positive']

# percentage of positive tweets

print("Hate tweets percentage: {} %".format(100\*len(ptweets)/len(tweets)))

# picking negative tweets from tweets

ntweets = [tweet for tweet in tweets if tweet['sentiment'] == 'negative']

# percentage of negative tweets

print("Love tweets percentage: {} %".format(100\*len(ntweets)/len(tweets)))

# percentage of neutral tweets

print("Neutral tweets percentage: {} %".format(100\*(len(tweets) - len(ntweets) - len(ptweets))/len(tweets)))

# printing first 5 positive tweets

print("\n\nPositive tweets:")

for tweet in ptweets[:10]:

print(tweet['text'])

# printing first 5 negative tweets

print("\n\nNegative tweets:")

for tweet in ntweets[:10]:

print(tweet['text'])

if \_\_name\_\_ == "\_\_main\_\_":

# calling main function

main()