FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERING

Department of Computer Engineering

Course, Subject & Experiment Details

Practical No:	5
Title:	Develop Content (text, emoticons, image, audio, video) based social media analytics model for business. (e.g., Content Based Analysis: Topic, Issue, Trend, sentiment/opinion analysis, audio, video, image analytics)
Name of the Student:	Warren Fernandes
Roll No:	8940
Date of Performance:	21/02/2023
Date of Submission:	07/03/2023

Evaluation:

Sr. No.	Rubric	Grade
1	On time submission/completion (2)	
2	Preparedness (2)	
3	Skill (4)	
4	Output (2)	

Signature of the Teacher

Collect text data using Twitter APIs.

There are a lot of free APIs through which we can collect data and use it to solve problems. We will learn the Twitter API in particular (as it can be used in many applications of NLP like product reviews, sentiment analysis,....).

Problem

You want to collect text data using Twitter APIs.

Solution

Twitter has a gigantic amount of data with a lot of value in it. Social media marketers are making their living from it. There is an enormous amount of tweets every day, and every tweet has some story to tell. When all of this data is collected and analyzed, it gives a tremendous amount of insights to a business about their company, product, service, etc.

How It Works

Log in to the Twitter developer portal

Create your own app in the Twitter developer portal, and get the keys mentioned below. Once you have these credentials, you can start pulling data. Keys needed:

> • consumer key: Key associated with the application (Twitter, Facebook, etc.).

> • consumer secret: Password used to authenticate with the authentication server (Twitter, Facebook, etc.).

> • access token: Key given to the client after successful authentication of above keys.

> • access token secret: Password for the access key.

```
# Install tweepy
!pip install tweepy
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: tweepy in /usr/local/lib/python3.8/dist-packages (3.10.0)
Requirement already satisfied: requests[socks]>=2.11.1 in /usr/local/lib/python3.8/dist-packages (from tweepy) (2.25.1)
Requirement already satisfied: six>=1.10.0 in /usr/local/lib/python3.8/dist-packages (from tweepy) (1.15.0)
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.8/dist-packages (from tweepy) (1.3.1)
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.8/dist-packages (from requests-oauthlib>=0.7.0->tweepy) (3
Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.8/dist-packages (from requests[socks]>=2.11.1->tweepy) (
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.8/dist-packages (from requests[socks]>=2.11.1->tweep
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packages (from requests[socks]>=2.11.1->tweepy) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/dist-packages (from requests[socks]>=2.11.1->tweepy)
Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in /usr/local/lib/python3.8/dist-packages (from requests[socks]>=2.11.1->twee
```

```
# Once all the credentials are in place, use the code below to fetch the data.

# Import the libraries
import numpy as np
import tweepy
import json
import pandas as pd
from tweepy import OAuthHandler

# credentials  --> put your credentials here
consumer_key = ""
consumer_secret = ""
access_token = ""
access_token_secret = ""

# calling API
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)

# Provide the query you want to pull the data. For example,
# pulling data for "bollywood stars" or "US unemployment"  or "IIT Admissions" or ...
query = "IIT Admissions 2022"

# Fetching tweets
Tweets = api.search(query, count = 10, lang='en', exclude='retweets',tweet_mode='extended')

# The query above will pull the top 10 tweets when the term "IIT Admissions 2022"
# is searched. The API will pull English tweets since the language
```

```
# given is ‘en’ and it will exclude retweets.
# language codes possible are : https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes
```

Getting the Tweets + Some Attributes

In this section, we will get some tweets plus some of their related attributes and store them in a structured format.

If we are interested in getting more than 100 tweets at a time, which we are in our case, we will not be able to do so by just using api.search. We will need to use tweepy.Cursor which will allow us to get as many tweets as we desire.

For our purpose, the end result is that it will just keep going on fetching tweets until we ask it to stop by breaking the loop.

```
# start by creating an empty DataFrame with the columns we'll need
df = pd.DataFrame(columns = ['Tweets', 'User', 'User_statuses_count',
                             'user_followers', 'User_location', 'User_verified',
                             'fav_count', 'rt_count', 'tweet_date'])
```

```
# Next, lets define a function as follows.

def stream(data, file_name):
    i = 0
    for tweet in tweepy.Cursor(api.search, q=data, count=100, lang='en').items():
        print(i, end='\r')
        df.loc[i, 'Tweets'] = tweet.text
        df.loc[i, 'User'] = tweet.user.name

        # indicates the no. of times the user as tweeted
        df.loc[i, 'User_statuses_count'] = tweet.user.statuses_count

          # indicates the no. of times the user as tweeted
        df.loc[i, 'user_followers'] = tweet.user.followers_count
        df.loc[i, 'User_location'] = tweet.user.location
        df.loc[i, 'User_verified'] = tweet.user.verified
        df.loc[i, 'fav_count'] = tweet.favorite_count

        # It's the number of tweets that given user has marked as favourite.
        df.loc[i, 'rt_count'] = tweet.retweet_count
        df.loc[i, 'tweet_date'] = tweet.created_at

        i+=1
        if i == 1000:
            break
        else:
            pass
```

Let's look at this function from the inside out:

> First, we followed the same methodology of getting each tweet in a for loop, but this time from tweepy.Cursor.

> Inside tweepy.Cursor, we pass our api.search and the attributes we want: q = data: data will be whatever piece of text we pass into the stream function to ask our api.search to search for just like we did passing "un unemployment" in the previous example.

count = 100: Here we are setting the number of tweets to return to 100, via api.search, which is the maximum possible number.

lang = 'en': Here I am simply filtering results to return tweets in English only.

Now, since we put our api.search into tweepy.Cursor, it will not just stop at the first 100 tweets. It will instead keep going on forever; that's why we are using i as a counter to stop the loop after 1000 iterations.

> Next, I am filling my DataFrame with the attributes I am interested in and during each iteration making use of the .loc method in Pandas and my i counter.

The attributes I am passing into each column are self explanatory and you can look into the Twitter API documentation for what other attributes are available and play around with those.

> Finally I am saving the result into an excel file using "df.to_excel" and here I am using a placeholder {} instead of naming the file inside the function because I want to be able to name the file myself when I run the function.

Now, I can just call my function as follows, looking for tweets about *"Some Text of your Choice"* again and naming my file "my_tweets."

```
stream(data = ['India'], file_name = 'my_tweets') #some text of your choice
```

```
df.head()
```

	Tweets	User	User_statuses_count	user_followers	User_location	User_verified	fav_co
0	RT @RomanaDaljeet: Saint Gurmeet Ram Rahim Sin...	Amar Jeet	24704	24		False	
1	RT @Officialteam_vs: India Emerges as Global L...	Manish Jadhav	39026	521		False	
2	RT @RenukaJain6: 👍👍Opposition parties in Ind...	chitusinha	22527	59		False	
3	RT @dodo: Stray dog found in	Angelique	2785	16		False	

Let's Analyze Some Tweets

```
# importing TextBlob. It has build-in sentiment property
from textblob import TextBlob

# The sentiment property returns a named tuple of the form
# Sentiment(polarity,subjectivity). The polarity score is a float
# within the range [-1.0, 1.0].
# The subjectivity is a float within the range [0.0, 1.0]
# where 0.0 is very objective and 1.0 is very subjective.
```

> I would like to add an extra column to this DataFrame that indicates the sentiment of a tweet.

> We will also need to add another column with the tweets stripped of useless symbols, then run the sentiment analyzer on those cleaned up tweets to be more effective.

Let's start by writing our tweets cleaning function:

```
import re

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

```
# Let's also write our sentiment analyzer function:

def analyze_sentiment(tweet):
   analysis = TextBlob(tweet)
   if analysis.sentiment.polarity > 0.1:
       return 'Positive'
   elif analysis.sentiment.polarity < -0.1:
       return 'Negative'
   else:
       return 'Neutral'
```

```
# Now let's create our new columns:

# Now let's create our new columns:
df['clean_tweet'] = df['Tweets'].apply(lambda x: clean_tweet(x))

df['Sentiment'] = df['clean_tweet'].apply(lambda x: analyze_sentiment(x))
```

```
# Let's look at some random rows to make sure our functions
# worked correctly.
# Example (300th row):
n=150
print('Original tweet:\n'+ df['Tweets'][n])
print()
print('Clean tweet:\n'+df['clean_tweet'][n])
print()
print('Sentiment:\n'+df['Sentiment'][n])
```

```
Original tweet:
RT @follow_amj: I still remember when Dr.Manmohan Singh was Prime Minister the leader of opposition Shri Atal Behari Bajpayee went

Clean tweet:
RT amj I still remember when Dr Manmohan Singh was Prime Minister the leader of opposition Shri Atal Behari Bajpayee went to the U

Sentiment:
Neutral
```

```
# find no. of positive sentiments
pos = 0
for i in range(0,1000):
  if df['Sentiment'][i] == 'Positive':
    pos = pos+1
print(pos)
```

```
330
```

```
# find no. of negative sentiments
neg = 0
for i in range(0,1000):
  if df['Sentiment'][i] == 'Negative':
    neg = neg+1
print(neg)
```

```
91
```

```
# find no. of neutral sentiments
neu = 0
for i in range(0,1000):
  if df['Sentiment'][i] == 'Neutral':
    neu = neu+1
print(neu)
```

```
579
```

```
print(neu+pos+neg)
```

```
1000
```

```
```

Conclusion:

Developing a content-based social media analytics model for business is a practical experiment that can be implemented successfully. By analyzing the text, emoticons, images, audio, and video content, the model can extract valuable insights related to topics, issues, trends, sentiment/opinion analysis, and image and video analytics. These insights can help businesses understand their audience's preferences and needs, identify areas of improvement, and develop effective marketing strategies. Furthermore, by leveraging social media analytics, businesses can enhance their brand reputation and customer engagement. Overall, this experiment has the potential to offer businesses a competitive edge and improve their overall performance in the marketplace.