

IST-652
Mini Project-2

Semi-Structure Data Analysis using Python

By:
Nivesh Vaze
nvvaze@syr.edu
411946624

Contents

Introduction	3
Dataset and Source	3
Part 1: Analysis of tweets with hashtag.....	3
Part 2: Extract tweets by Elon Musk for analysis	6
Part 3: Sentiment Analysis.....	8
Program Description	13

Introduction

The project is an attempt to extract data and analyze data from Twitter using python library Tweepy. Tweepy is an open-source Python package that gives you a very convenient way to access the Twitter API with Python. Tweepy includes a set of classes and methods that represent Twitter's models and API endpoints, and it transparently handles various implementation details.

The IST Mini project-2 is an attempt to extract and analyze semi-structure data from Twitter using Tweepy. The project is divided in three minor parts namely:

-

1. Extracting and analyzing tweet's containing twitter Hashtag
2. Analyzing a users' tweets and finding meaningful statistics
3. Sentiment analysis using TextBlob

Dataset and Source

The Dataset is in JSON format. I have utilized tweepy Cursor interface and user_timeline method to extract and iterate through tweets in python.

Part 1: Analysis of tweets with hashtag

In this part, I used '#covid19' search query to fetch 200 tweets from twitter. Thereafter, I initialized pandas' data frame to populate each tweet in it. I used 8 attributes from the twitter object namely - user_name, user_location, user_description, user_verified, date, text, hashtags, and source.

```
[255] #search for a hashtag with filter
      search_query = "#covid19 -filter:retweets"

[371] # get 200 tweets from the API using Tweepy cursor
      tweets = tw.Cursor(api.search,
                          q=search_query,
                          lang="en",
                          since="2020-09-16").items(200)

      # store the API responses in a list
      tweets_copy = []
      for tweet in tweets:
          tweets_copy.append(tweet)

      print("Total Tweets fetched:", len(tweets_copy))

Total Tweets fetched: 200
```

```
# populate the dataframe
for tweet in tweets_copy:
    hashtags = []
    try:
        for hashtag in tweet.entities["hashtags"]:
            hashtags.append(hashtag["text"])
        text = api.get_status(id=tweet.id, tweet_mode='extended').full_text
    except:
        pass
    tweets_df = tweets_df.append(pd.DataFrame({'user_name': tweet.user.name,
                                                'user_location': tweet.user.location,\
                                                'user_description': tweet.user.description,\
                                                'user_verified': tweet.user.verified,\
                                                'date': tweet.created_at,\
                                                'text': text,\
                                                'hashtags': [hashtags if hashtags else None],\
                                                'source': tweet.source}))

    tweets_df = tweets_df.reset_index(drop=True)
```

`tweets_df.head()`

- Above command is used to view the top 5 tweets in the data frame

The dimension of data frame is calculated using the shape command as shown below: -

```
tweets_df.shape  
  
(200, 8)
```

`tweets_df['user_verified'].value_counts()`

- Above command is used to get the pivot table for summary of verified accounts.

```
[342] tweets_df['user_verified'].value_counts()  
  
False    436  
True      64  
Name: user_verified, dtype: int64
```

```
▶ tweets_df.isnull().sum()  
  
user_name          0  
user_location      0  
user_description   0  
user_verified      0  
date               0  
text               0  
hashtags           212  
source             0  
dtype: int64
```

Check for null values using isnull method.

```
tweets_df.user_location.value_counts()
```

	120
Los Angeles, CA	75
Manila, Philippines	6
New Delhi, India	6
Manhattan, NY	6
...	
Hamilton City, New Zealand	1
Toronto, Canada	1
Virginia, USA	1
Toronto	1
Unlit Lowlands, Plague Island	1

Name: user_location, Length: 212, dtype: int64

Check for value counts on location attribute suggests that most number of tweets in the data frame are from California, followed by Philippines.

Part 2: Extract tweets by Elon Musk for analysis

In this part, I extracted 200 tweets by Elon Musk using `user_timeline()` method of `tweepy`, and then iterated through each tweet to add them to list.

```
#new_tweets1 = tw.Cursor(api.search, screen_name="@elonmusk", since="2020-5-31", tweet_mode='extended').items(1000)

new_tweets = api.user_timeline(screen_name = "elonmusk", count = 200, lang = "en",
tweet_mode = "extended")

list = []
for tweet in new_tweets:
    text = tweet._json["full_text"]

    refined_tweet = {'text' : text,
                     'favorite_count' : tweet.favorite_count,
                     'retweet_count' : tweet.retweet_count,
                     'created_at' : tweet.created_at}

    list.append(refined_tweet)

for tweet in new_tweets:
    text = tweet._json["full_text"]
```

```
[325] import pandas as pd
      df = pd.DataFrame(list)
      df.to_csv('refined_tweets.csv')
      df.shape

      (200, 4)
```

As seen above, the data frame dimension is 200 rows and 4 columns. Columns are namely text, favorite_count, retweet_count, and created_at.

Created_at column is a time stamp. We converted it to datetime type of pandas and extracted date from the same.

```
#convert to pandas date-time
df['created_at'] = pd.to_datetime(df['created_at'])

# remove time from Date and store it in a new column
df['Date'] = df['created_at'].dt.date
# display the dataframe
print(df)
```

Number of tweets per month are calculated with values counts command as shown below

#number of tweets by date df.Date.value_counts()	
2022-04-21	37
2022-04-09	20
2022-04-03	19
2022-04-08	16
2022-04-18	15
2022-04-14	14
2022-04-10	13
2022-04-16	12
2022-04-20	11
2022-04-07	10
2022-04-05	10
2022-04-06	7
2022-04-04	5
2022-04-15	4
2022-04-17	4
2022-04-19	2
2022-04-22	1
Name: Date, dtype: int64	

The final business questions are to query the tweet with the most favorite count and the tweet with most retweet count.

[329] #query the tweet with max favorite_count df[df.favorite_count == df.favorite_count.max()]					
	text	favorite_count	retweet_count	created_at	Date
90	i♥u	1216281	90258	2022-04-14 21:40:23	2022-04-14
[330] #query the tweet having max retweet count df[df.retweet_count == df.retweet_count.max()]					
	text	favorite_count	retweet_count	created_at	Date
98	I made an offer \nhttps://t.co/VvreuPMeLu	920993	114378	2022-04-14 11:23:21	2022-04-14

As shown above, we use max () function to answer these business queries.

Part 3: Sentiment Analysis

At first, only the tweet message is loaded into the data frame to make things simpler for analysis.

```
df = pd.DataFrame([tweet.full_text for tweet in new_tweets], columns = ["tweet"])
df.head()
```



tweet



0	@PPathole Yes
1	@28delayslater 🤖 100
2	Believe in the future
3	https://t.co/IOV4tWvyxv
4	https://t.co/TMOnmKBcWQ

The next process is to clean the data. i.e., to remove hashtags, mentions, retweets, links etc.

```
[332] import re

# Clean The Data
def cleantext(text):
    text = re.sub(r"@[A-Za-z0-9]+", "", text) # Remove Mentions
    text = re.sub(r"#", "", text) # Remove Hashtags Symbol
    text = re.sub(r"RT[\s]+", "", text) # Remove Retweets
    text = re.sub(r"https?:\/\/\/\S+", "", text) # Remove The Hyper Link

    return text

# Clean The Text
df["tweet"] = df["tweet"].apply(cleantext)
```

After this, we calculate subjectivity and polarity to understand the sentiment of a tweet.

```
[1] from textblob import TextBlob

# Get The Subjectivity
def sentiment_analysis(ds):
    sentiment = TextBlob(ds["tweet"]).sentiment
    return pd.Series([sentiment.subjectivity, sentiment.polarity])

# Adding Subjectivity & Polarity in our dataframe
df[["subjectivity", "polarity"]] = df.apply(sentiment_analysis, axis=1)

df
```

	tweet	subjectivity	polarity
0	Yes	0.000000	0.000000
1	👉 100	0.000000	0.000000
2	Believe in the future	0.125000	0.000000
3		0.000000	0.000000

Further, we create a word cloud to find the most frequent and relevant word used in the text message.


```
import matplotlib.pyplot as plt
from wordcloud import WordCloud

allwords = " ".join([twts for twts in df["tweet"]])
wordCloud = WordCloud(width = 1000, height = 1000, random_state = 21, max_font_size = 119).generate(allwords)
plt.figure(figsize=(20, 20), dpi=80)
plt.imshow(wordCloud, interpolation = "bilinear")
plt.axis("off")
plt.show()
```



```
def analysis(score):
    if score < 0:
        return "Negative"
    elif score == 0:
        return "Neutral"
    else:
        return "Positive"

# Create a New Analysis Column
df["analysis"] = df["polarity"].apply(analysis)
# Print The Data
df
```

	tweet	subjectivity	polarity	analysis
0	Yes	0.000000	0.000000	Neutral
1	 100	0.000000	0.000000	Neutral
2	Believe in the future	0.125000	0.000000	Neutral

Finally, positive, and negative tweets are highlighted and a ratio of length of positive tweets to negative tweets is calculated to comment on the sentiment analysis of a user's tweet.

```
positive_tweets = df[df['analysis'] == 'Positive']
negative_tweets = df[df['analysis'] == 'Negative']

print('positive tweets')
for i, row in positive_tweets[:5].iterrows():
    print(' - ' + row['tweet'])

print('negative tweets')
for i, row in negative_tweets[:5].iterrows():
    print(' - ' + row['tweet'])
```

```
[337] len(positive_tweets) / len(negative_tweets)

4.238095238095238
```

We can conclude that as the ratio is +ve, the overall tweets by user Elon Musk are positive in nature

Program Description

Below attached is the Jupyter notebook consisting of all the executable commands in Python. I have added the text blocks in each command to better understand the dynamics of each code block.



Semi_structure_DA.ip
ynb