Abstract

This analysis deals with identifying and understanding different opinions of people activity in social media (Twitter). It contains vast amount of data which includes tweets, updates, status, blog posts etc. In this report we are trying to analyze twitter activities in June month at Europe region.

Learning From This Course Work:

- Explored large dataset of tweets collected from the Twitter API.
- Handling big data sets in efficient manner.
- Analysis of different graphs etc.

Processing Huge Dataset

Once we downloaded the dataset we got huge number of files(in .json format) that contains lot of redundant data. I am following some steps to simplify this data.

Step 1: Identify required attributes to analyze, and create separate file for those attributes. It will help us to minimize the dataset and it makes easier to process the data.

Based on my analysis we require few columns(attributes) they are **Twiiter Id, UserId, Text, Created_at, Country, Geo, User_mentions** fields.

Once we identified the data fields now I am going to extract this data fields from json files and I will create separate single csv file(twitter_file.csv).

```
path_to_file = 'D:\Computer Sceince\Introduction_data_science\CourseWork\TwitterJune2021\TwitterJune2021\geoEurope'
json_path = os.path.join(path_to_file)

cols = ['id', 'text', 'created_at', 'entities', 'geo', 'place', 'user']
count = 0
for fname  in glob(path_to_file + "\\" +'*.json'):
    df = pd.read_json(fname, lines=True)
    if count == 0:
        df[cols].to_csv('twitter_file.csv', mode = 'a', header=True)
        count = 1
    else:
        df[cols].to_csv('twitter_file.csv', mode = 'a', header=False)

    count += 1
    print(count)
```

From above snippet code I am taking Some specific objects from all json files and creating twitter_file.csv file.

Now in my twitter_file.csv file it contains **Id, Text, created_at, entities, geo, place, user** columns.

Step 2: We don't need every column in csv file we need some metadata from that column, now again I am going to extract this metadata and store it in Database.

Now I am creating a table to store this metadata and inserting this data values into my database.

```
In [3]: conn = sqlite3.connect("TWITTER.db") # Creating new db
        c = conn.cursor()
        c.execute('''CREATE TABLE twitter(Id INTEGER, User_Id INTEGER, Text TEXT NOCASE, DateOfTweet TIMESTAMP, CountryName TEXT NOCASE,
        LattLong TEXT NOCASE, UserMentions TEXT NOCASE)''')
```

In the above snippet code I am creating database (TWITTER.db) and table name called **twitter.**

Now I am going to insert data to this table from twitter_file.csv file

```
chunksize = 10 * 1000
def process(data):
    for ind in data.index:
        c.execute("INSERT INTO twitter VALUES (:Id, :User_id, :Text, :DateOfTweet, :CountryName, :LattLong, :UserMentions)", {
                'Id' : int(data['id'][ind]),
                'User_id' : int(user_id(data['user'][ind])),
                'Text' : data['text'][ind],
                'DateOfTweet' : Time_stamp(data['created_at'][ind]),
                'CountryName' : country_or_none(data['place'][ind]),
                'LattLong': lat_or_non(data['geo'][ind]),
                'UserMentions': user_mention(data['entities'][ind])
            })

    count += chunksize



filename = 'twitter_file.csv'
with pd.read_csv(filename, chunksize=chunksize) as reader:
    for chunk in reader:
        chunk.drop_duplicates(subset='text',inplace=True)  # Drop duplicates in Text field
        chunk = chunk[chunk['id'].notna()]                # Exclude nan values from the row['id']
        process(chunk)
```

In the above snippet code, I am reading chunk by chink data from csv file after that I am dropping some duplicates and excluding nan values from **Id**.

While inserting I am extracting each metadata by calling some functions (for each colums have separate function).

```
def lat_or_non(geo):
    if pd.isnull(geo):
        return '0.0'
    else:
        y = eval(geo)
        return (','.join([str(x) for x in y['coordinates']]))

def user_mention(entities):
    #print(entities)
    my_list = []
    if pd.isnull(entities):
        return 'NONE'
    else:
        y = eval(entities)
        if y.get('user_mentions'):
            my_list = [x for x in y.get('user_mentions')]
            return (','.join([str(x['id']) for x in my_list]))
        else:
            return 'NONE'
```

I didn’t include all functions here because it will consume more space.

Now my insertion is complete, now we can easily extract this data by querying.

PART1:

Q1: Count the total number of tweets, describing how you deal with duplicates or other anomalies in the data set.

Ans:

The total Number of tweets in the given dataset is **13861411**

But after refining duplicates and Nan values in the dataset The total number of tweets got reduced and I am using only this reduced dataset for analysis.

I am comparing text field in the dataset if the same text is repeated in the data, I will mark it has duplicate and I am not inserting those rows into database. I am taking only specific colum ns from dataset and I am avoiding duplicates and other null values. We are checking Nan val ues in Id field from whole dataset, if it is Nan I am ignoring those rows for database insertion.

I already explained in above part (Processing Huge dataset) with sample code.

After these steps my Total number of Tweets is **13803777**

```
In [5]: # PART1: Q1
        Total_twitts = 0
        c.execute("""SELECT * FROM twitter""")
        #rows = c.fetchall()
        for row in c:
            Total_twitts += 1
        print("Total number of tweets: ", Total_twitts)

        Total number of tweets:  13803777
```

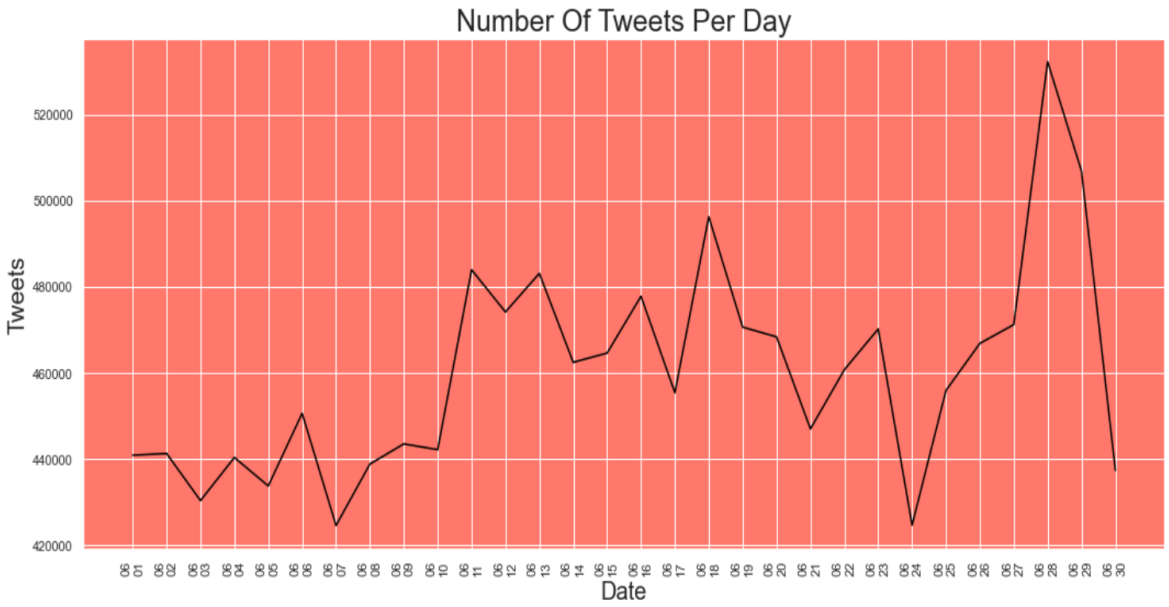
Q2: Plot a time-series of the number of tweets by day using the whole dataset and comment on what you see

Ans:

From my query, I am going to group the tweets by each day and count the how many number of tweets per day, Once I got the Date with number of tweets we are going to draw the time series of the data. Below is the query to get number of tweets per each day.

```
3]: # PART1: Q2
    c.execute("""SELECT date(DateOfTweet), count(*) FROM twitter WHERE date(DateofTweet)!= "2021-05-31" GROUP BY date(DateOfTweet) "'
    rows = c.fetchall()
```

Once we got the data with number of tweets, I am going to plot the time series using matplotl ib library. I got below graph after plotting the time series of each day.



I have Excluded 2021-05-31 tweets because it contains only 1hr data, we can't compare this 1hr data with other days because other days have complete 24hr data.
As We can see in the graph in the beginning of June month tweets are low compared to end of the month, in the mid of the month tweets are steadily increasing again one day it came down but it next days it got increased and it reaches highest number of tweets in that month (2021-06-28). We got more number of tweets at end of month, it may be because at the end of month Europe having some couple of foot ball matches, so people are posting their opinion or wishes to their countries.

Q3: Using a box and whisker diagram the average number of tweets on the weekdays in the d ataset to the numbers for weekend days. Are there statistically significant differences between the number of tweets on weekdays and weekends?
Ans:

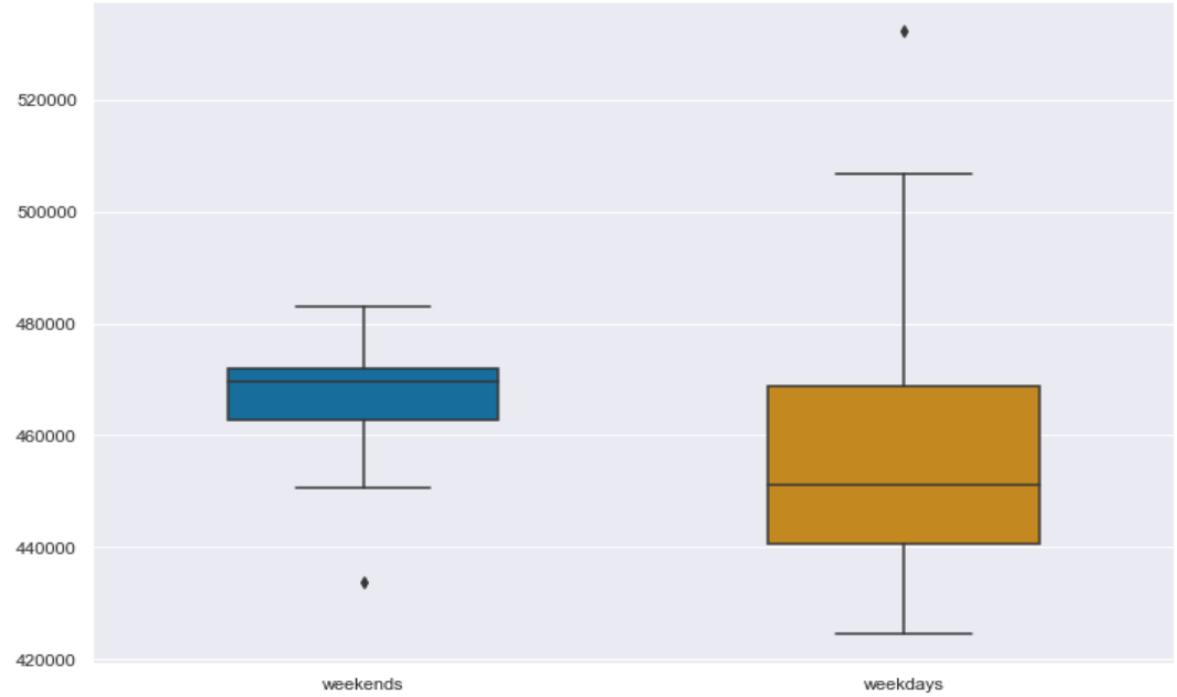
First I am going to fetch all the days of tweet using GROUP BY method, here also I am not including 2021-05-31 data. After getting each day tweets I am segregating weekday and weekend tweets separately and I am plotting this tweets using box plot.

```
[4]: #PART1: Q3
c.execute("""SELECT date(DateOfTweet), count(*) FROM twitter WHERE date(DateofTweet)!= "2021-05-31" GROUP BY date(DateOfTweet)""")
rows = c.fetchall()
print(rows)
```

```
[('2021-06-01', 440881), ('2021-06-02', 441272), ('2021-06-03', 430327), ('2021-06-04', 440352), ('2021-06-05', 433727), ('2021
-06-06', 450613), ('2021-06-07', 424548), ('2021-06-08', 438822), ('2021-06-09', 443503), ('2021-06-10', 442165), ('2021-06-1
1', 483974), ('2021-06-12', 474154), ('2021-06-13', 483133), ('2021-06-14', 462463), ('2021-06-15', 464610), ('2021-06-16', 477
808), ('2021-06-17', 455398), ('2021-06-18', 496277), ('2021-06-19', 470643), ('2021-06-20', 468380), ('2021-06-21', 446989),
('2021-06-22', 460724), ('2021-06-23', 470212), ('2021-06-24', 424614), ('2021-06-25', 455917), ('2021-06-26', 466859), ('2021-
06-27', 471212), ('2021-06-28', 532319), ('2021-06-29', 506884), ('2021-06-30', 437353)]
```

```
lst_weekend_data = []
lst_weekday_data = []
lst_of_dates = [a_tuple[0] for a_tuple in rows]
xaxis = [datetime.datetime.strptime(d, '%Y-%m-%d') for d in lst_of_dates]
for i, j in zip(xaxis, [a_tuple[1] for a_tuple in rows]):
    if i.weekday() > 4:
        lst_weekend_data.append(int(j)) # Its a Weekend
    else:
        lst_weekday_data.append(int(j)) # Its a Weekday
```

Using lst_weekend_data[] and lst_weekday_data[] we are displaying box plot.



As from the graph there is a statistically significant difference between weekend and weekdays. From the above box plot in weekends the median is closer to the top of the box, then the distribution is **negatively skewed** but in weekdays distribution is **positively skewed**.

Q4: Plot the average number of tweets at each hour of the day for weekdays and weekends an d comment. You should have two plots where the x-axis is time of day (from midnight to mid night) and the y-axis shows the number of tweets
Ans:

Group by Hour and date, we will get per day and hour of tweets. Here also I am excluding 2021-05-31 data.

```
[27]: # PART1:Q4
c.execute("""SELECT date(DateOfTweet), count(*), strftime ('%H',DateOfTweet) hour FROM twitter
          WHERE date(DateofTweet)!= "2021-05-31" GROUP BY strftime ('%H',DateOfTweet), date(DateOfTweet)""")
rows = c.fetchall()
print(rows)
```

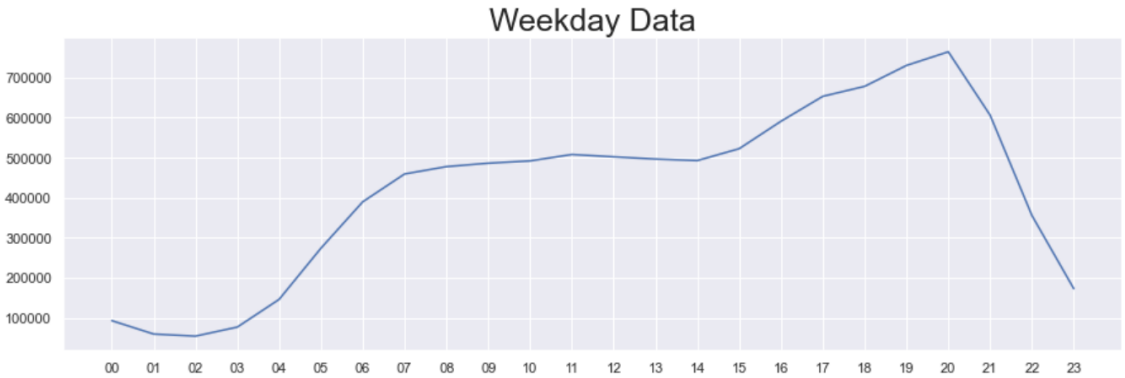
If we print rows we will get below format (date, number of tweets per hr, Hr)

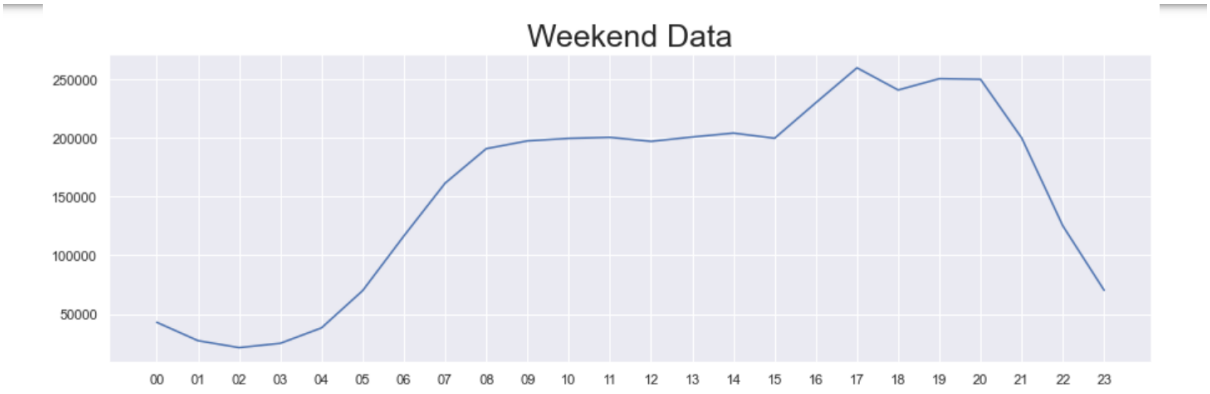
```
[('2021-06-01', 4190, '00'), ('2021-06-02', 4055, '00'), ('2021-06-03',
4051, '00'), ('2021-06-04', 4475, '00'),..... ('2021-06-30', 4427, '00'),
('2021-06-01', 2629, '01'), ('2021-06-02', 2539, '01'), ('2021-06-03',
2774, '01'),..... ('2021-06-25', 9307, '23'), ('2021-06-26', 9534, '23'),
('2021-06-27', 8304, '23'), ('2021-06-28', 9246, '23'), ('2021-06-29',
8089, '23')]
```

Now we are segregating weekday and weekend data and we are plotting the graphs(weekend and weekday).

```
[32]: Weekend = {}
Weekday = {}
for i in rows:
    date_time_obj = datetime.strptime(i[0], '%Y-%m-%d')  # To convert str into datetime
    if date_time_obj.date().weekday() > 4:               # Check for weekday or weekend
        if i[2] in Weekend:
            Weekend[i[2]] += i[1]
        else:
            Weekend[i[2]] = i[1]
    else:
        if i[2] in Weekday:
            Weekday[i[2]] += i[1]
        else:
            Weekday[i[2]] = i[1]
```

After that We are plotting Average number of tweets at each hour for both weekend and week day data.





As we can see in the graph most number of tweets is done at night time(between 17 - 21) and minimum number of tweets is recorded at 01 – 04 time.
All most all people is sleeping in this time (01 - 04) so we can easily know why tweets are recorded less number at this time, and between 17 – 21 we got most number of tweets in weekdays as well as weekend because at this time most of the people have free time(in home, taking rest or something).

PART2:

Q1: Draw a map of Europe showing the location of the GPS-tagged tweets - these are tweets which have a “coordinates” field in the metadata.
Ans:

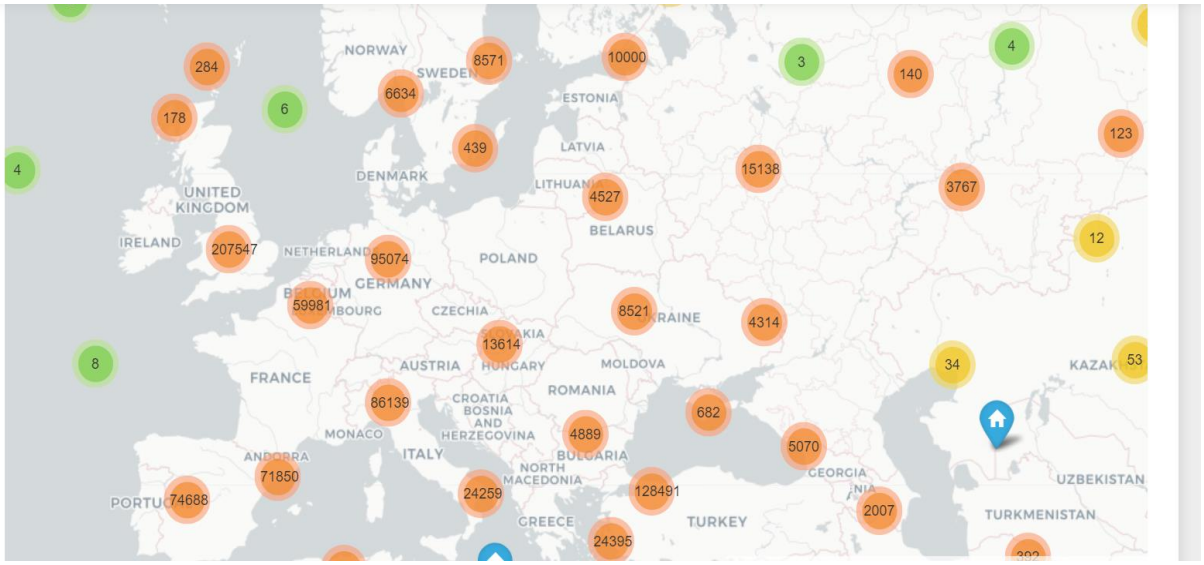
In my case I am storing all the geo-tagged coordinates in LattLong field(database), while inserting we are checking geo-tag values is Nan or not, if it is Nan we are inserting 0.0 in database.
While Extracting I am taking all LattLong values which are not equal to “0.0”(Because while inserting I am checking Coordinates values from metadata if Coordinate values are null I am i nserting ‘0.0’).

[14]:
```
#PART2
geo_loc = []
c.execute("""SELECT Id, LattLong FROM twitter WHERE LattLong != "0.0" """)
rows = c.fetchall()
for row in rows:
    geo_loc.append(row)
```

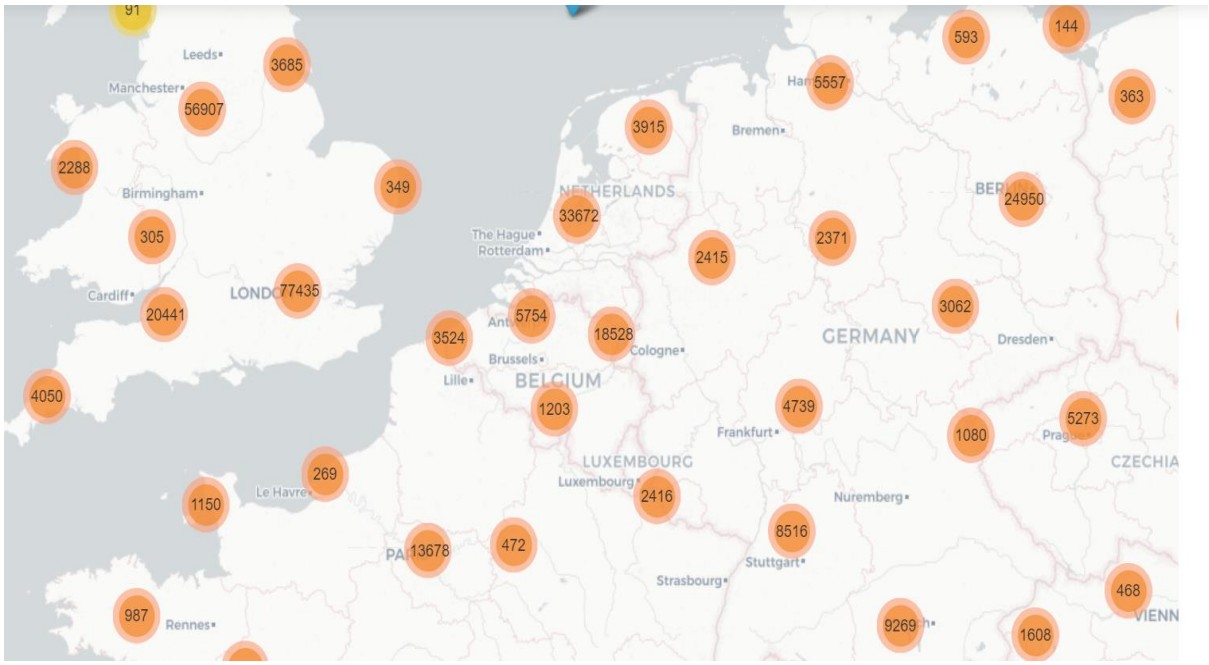
By this above query I got some millions of tweets, I am using **FastMarkerCluster**() to Display the is locations. This method is going to group the given data and when we zoom in it will spread out and display more locations.
I am using cluster mechanism to display tweets because if we display all geo tagged location in a single attempt location are jumbled and we can’t see properly. The below 5 images shows how the cluster will work.

Euro1: In the First Image I am showing Europe map with GPS tagged tweets from dataset.

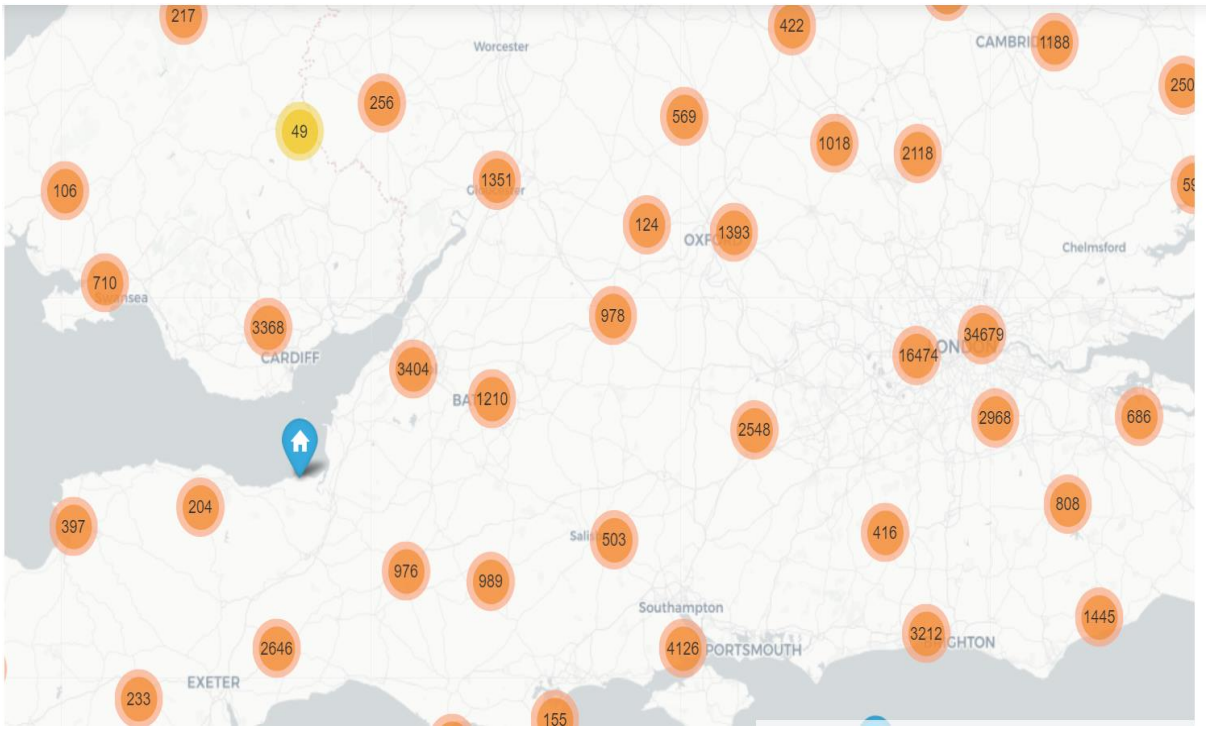
Euro2: In this image we are Zoom in of Euro1 Image. As we can see if we zoom in the tweet locations are spreading and displays accurate number of tagged tweets.



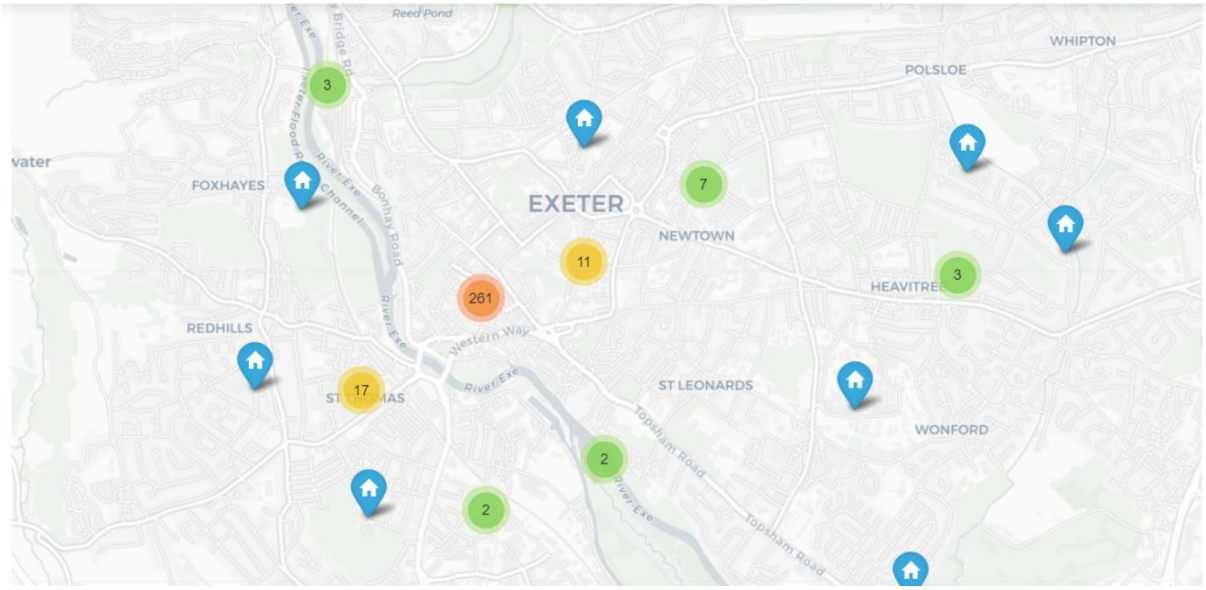
Euro3: Again, I zoomed in and it displays more accurate numbers for particular locations.



Euro4: In this image I am zooming near to UK, so the UK location tweets are spreading out and displaying accurate numbers for regions.



Euro5: In this image I have went through near Exeter city and displaying tweet locations.



Q2: Explain any patterns you observe.
Ans:

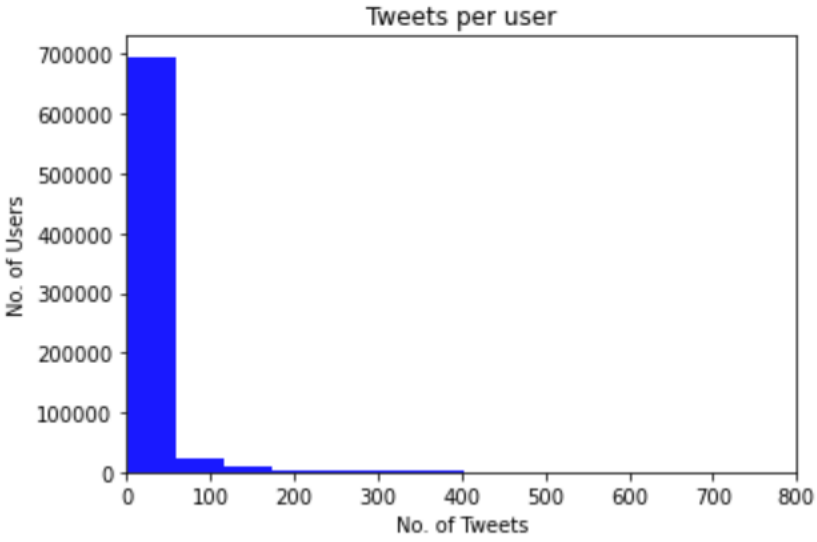
As we can see in the map (First Image, Euro1), most number of tweets are coming from Uni ted Kingdom and Turkey. Less number of tweets are coming from Iceland, Norway and Sweden (North of Europe). And another pattern we can see the highest number of tweets are coming from major cities.
Example: London in UK, Istanbul in Turkey, Madrid in Spain, and another exiting fact there is some football match in the month of June so we can see lot of tagged tweets from Barcelona.

PART3:
Q1: Make a histogram of tweets per user with number of users on the y-axis and number of tweets they make on the x-axis. Discuss the distribution that you see.
Ans:

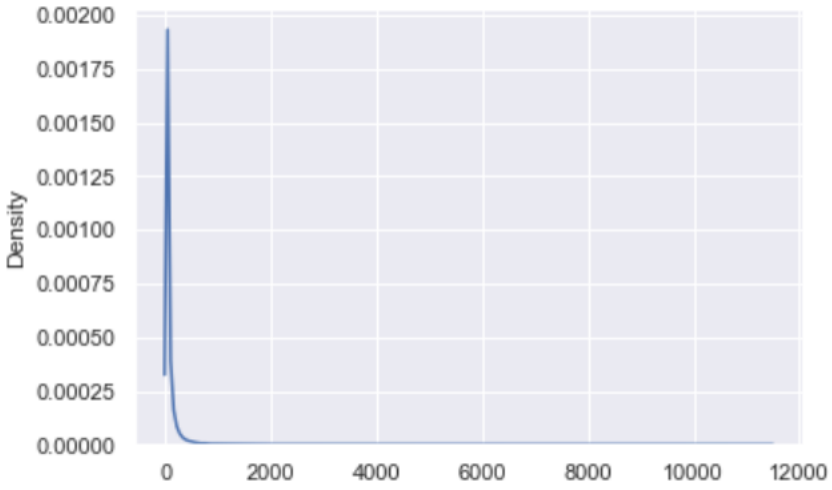
Here I am grouping by User_id to get the number of tweets per user.

```
4]: c.execute("""SELECT User_Id, count(*) FROM twitter GROUP BY User_Id""")
    userTweets = c.fetchall()
    histData = [i[1] for i in userTweets]
```

From the above code I am taking number of tweets into a list. After that I am plotting histogram using that list.



As we can see in the above graph almost 92% of the users sends below 50 tweets in a month, only few users send above 100 tweets and some of the users (less than 1%) sends above 400 tweets in a month.
I am adding another plot(sns) to understand better distribution for this dataset.



If we calculate mean, mode and median for dataset mean is greater than mode and median so it is positively skewed distribution.

Q2: Find the top-5 users by total number of tweets. Do you think any are automated accounts (aka. bots)? Justify your answer.
Ans:
I have identified top 5 users with total number of tweets in a month.

```
4]: # PART3:Q2
    c.execute("""SELECT User_Id, count(*) FROM twitter GROUP BY User_Id ORDER BY COUNT(User_Id) DESC""")
    rows = c.fetchall()
    print(rows)
```

From the above query we will get most number of tweets done by user from descending order.
The top 5 users are
Top 5 User : 1384110594, 1211606433399095296, 1382496899744301057, 974188940973486081, 161262801
This above 5 Accounts are automated accounts.

- ID: 1384110594:
  This user sends repeated text at same time to same group of people everyday e.g :'Its one O'clock in the Evening(in spanish)'
- ID: 1211606433399095296 :
  This user sends good morning and good evening messages at same and tagging same people everyday.
- ID: 1382496899744301057 :
  This user sharing same links to some group of users and I can see repeated messages
- ID: 974188940973486081 :
  This user promoting his shop by sharing same info with different link every time.
- ID: 161262801 :
  From this Id user posting tracks every time by adding same hashtags

From the above points we can conclude these users accounts is automated.

Q3: Find the 5 users who receive the most mentions and comment on this.
Ans:
First we are going to extract all data in the UserMentions column and after that we will convert this data into list then we can comment.

```
[59]: #PART3: Q3
      c.execute("""SELECT UserMentions FROM twitter WHERE UserMentions != "NONE" """)
      rows = c.fetchall()
      print('DONE')
```

After getting all rows now we are converting this data into single list and count which UserMentions is max.
I got below people have most user mentioned from the dataset

- 10228272 **:@RTErdogan** (Tukish politician)
  He got the most mentions from the users, people are tagging him and asking some questions.

- 68034431 : **@YouTube**
  people are sharing some video link from you tube, this is the second highest user_tags
- 3131144855 : **@BorisJohnson**
  people are tagging him by pointing some issues and some people are wishing him for his decisions about his country
- 510948637 : **@dimash_official**
  It is a musical band id, people are wishing and complementing this band.
- 335141638 : **@BTS_twt**
  This user is a singer, he had lot of fans and people are tagging him and praising for his voice.

Top 5 mentions are celebrities from different fields either politician, singer, music band and one is social media channel.

Q4: Calculate how often users in the UK, France, Germany, Italy and Turkey mention users in each of the other 4 countries. You should compute 25 numbers e.g. UK mentions UK, UK mentions France, France mentions UK etc. Comment on any patterns you observe.
Ans:

To calculate Usermentions in 5 different country I am writing a query like, taking all users in one dictionary(key=country name, values= list of user_id) and in another dictionary I am ta king all user_mentions(key= country name, value= user_mention_id).

```
# PART3:Q4
countries = ["United Kingdom", "France", "Germany", "Italy", "Turkey"]
user_mentions = {}
user_id_country = {}
for country in countries:
    c.execute("""SELECT UserMentions FROM twitter WHERE UserMentions != "NONE" AND CountryName in ('{}')""".format(country))
    rows = c.fetchall()
    user_mentions[country] = rows

print("DONE")
for country in countries:
    c.execute("""SELECT User_Id FROM twitter WHERE CountryName in ('{}')""".format(country))
    rows = c.fetchall()
    user_id_country[country] = rows
```

Now user_mentions contain countries with respective user_mention ID, and use_id_country c ontains countries with respective user_id.
Now I am using below loop to map the user_mention and user with all 5 countries.

```
count = {}
for i,j in user_mentions.items():        # user_mention list
    print(i, len(j))
    for k, l in user_id_country.items():   # user id list
        for item in l:
            if item in j:
                str1 = k + " mentions " + i
                if str1 in count:
                    count[str1] += 1
                    print("FOUND")
                else:
                    count[str1] = 1
```

Now we are going to print the count{} from the above snippet code.

```
{'Uniter Kingdom mentions United Kingdom': 948634, 'Uniter Kingdom mentions France': 84837, 'Uniter Kingdom mentions Germany':
52874, 'Uniter Kingdom mentions Italy': 20876, 'Uniter Kingdom mentions Turkey': 53456, 'France mentions United Kingdom': 5738
2, 'France mentions France': 203871, 'France mentions Germany': 43781, 'France mentions Italy': 24672, 'France mentions Turke
y': 5678, 'Germany mentions United Kingdom': 9897, 'Germany mentions France': 5892, 'Germany mentions Germany': 44908, 'Germany
mentions Italy': 10478, 'Germany mentions Turkey': 2767, 'Italy mentions United Kingdom': 15467, 'Italy mentions France': 8745,
'Italy mentions Germany': 6785, 'Italy mentions Italy': 43568, 'Italy mentions Turkey': 6890, 'Turkey mentions United Kingdom':
20897, 'Turkey mentions France': 15980, 'Turkey mentions Germany': 4579, 'Turkey mentions Italy': 5632, 'Turkey mentions Turke
y': 45678}
```

From the above details we can see most of the user mentions done by the same country people, For example UK mention UK, Italy mentions Italy etc. and another pattern we can observe the countries who had match between each other there user mention rate is also high i n number, for example France mentions Germany, France mention Italy etc.

PART4:

Q1: Identify 3 days with unusually high activity in 3 different countries of your choosing. For example you could choose one day in the UK, one in France and one in Turkey. Describe and justify how you identify 'unusual' days.

Ans:

I am choosing United Kingdom, France and Switzerland to identify high activities.

I am writing a query to sort out high activity days, first I am taking one country and group by date with max number of tweets.

```
[20]: countries = ["United Kingdom", "France","Switzerland"]
      data = {}
      for country in countries:
          c.execute("""SELECT date(DateOfTweet), COUNT(*) FROM twitter WHERE CountryName in ('{}')
          GROUP BY date(DateOfTweet) ORDER BY COUNT(date(DateOfTweet)) DESC LIMIT 1""".format(country))
          rows = c.fetchall()
          data[country] = rows
```

In the query I am just taking highest number of tweets in June month for particular countries with date. If I print data{} it look like below format.

```
[21]: for i, j in data.items():
          print("High Activity in",  i , j)

      High Activity in United Kingdom [('2021-06-18', 153467)]
      High Activity in France [('2021-06-28', 53692)]
      High Activity in Switzerland [('2021-06-28', 1937)]
```

United Kingdom:

Normally UK per day tweets is between 100000 -135000 but I can see 29th and 18th days tweets are high compared to all other day, among 29th and 18th, 18th June have max number of tweets. And in 18th June they had football match between England vs Scotland So in that day we can see high number of tweets.

France:

If we see other days of France the tweets in the range of 30000 – 40000 but In 2021- 6 – 28 They is a football match between France and Switzerland so we can see high number of tweets in that day compare to all other days in france .

Switzerland:

Normally Switzerland per day tweets is in the range of 900-1500, but 2021-06-28 is recorded as highest number of tweets in june month because there is a football match between France and Switzerland so people are active in that day.

Q2: Characterise each of these three days. Exactly how you do this is up to you, but for exam ple you could:

- Display some indicative Tweets.
- Make a word cloud from the tweet text.
- Plot tweets locations on a map.

Indicative Tweets:

Indicative tweets are identified by hyper links in a text, for getting indicative tweets I am searching hyperlinks in a text, if a text contains hyperlinks I am take
it has an indicative tweet.

```
]: c.execute("""SELECT * FROM twitter WHERE CountryName='United Kingdom' AND
              (date(DateOfTweet) = '2021-06-18' AND Text LIKE '%@England%')""")
   rows_uk = c.fetchall()
```

In the above query I am taking all the tweets from United Kingdom for 2021-06-18. After that we are displaying few indicative tweets from the rows. I followed same step for other two countries (France and Switzerland)

Indicative Tweets: The below tweets are Indicated @England and people are wishing to England for football match.

```
(1405734142508253186, 833276507128791041, 'Good morning twitter 🌎, bit
of a lay in today 🤧🤧🤧😉😉😉😛😛🤣🤣🤣,quick fire 4 hrs work,then
chill out ready for @England g… https://t.co/v20XJWZn4o', '2021-06-18 0
3:48:31', 'United Kingdom', '0.0', '1903382054')

(1405752533889404928, 579963946, '@England Play your strongest FIT 11,
for some who have not played in other tournaments this is their first r
eal tes… https://t.co/qfDQmRTw96', '2021-06-18 05:01:36', 'United Kingd
om', '0.0', '1903382054')

(1405802087775150081, 2783899629, "@England don't u let us down #MATCHD
AY #ENGSCO #EURO2020 #ThreeLions https://t.co/9FZ2Va6SjO", '2021-06-18
08:18:30', 'United Kingdom', '0.0', '1903382054')
```

The below Tweets are indicated at @France people are tagging france and wishing their team in French language .

```
(1409376916642091010, 1038885266, '@mmegeboudier @greg_binary @francein
fo Sans compter les vélos électriques cassés, les piquouses de dope gas
pillées.… https://t.co/xhJhtaD2UX', '2021-06-28 05:03:36', 'France', '0
.0', '3231560723,1308293677127696385,38395124')

(1409389152760696833, 305390796, 'Encore un choix impossible entre @fra
nceinter et @franceinfo ce matin : @RobertMenardFR ou @louis_aliot ….😱
', '2021-06-28 05:52:13', 'France', '0.0', '34867057,38395124,113234234
0,316956459')
```

The below Tweets are indicated at @Switzerland people are Exited to watch football match.

```
(1409625662739132416, 518856061, 'Yes watching #football with The Dad .
. France FR vs. Swiss CH 3-3 damn exciting game ... Now is over time ..
Great… https://t.co/DlJ78iWEWT', '2021-06-28 21:32:02', 'Switzerland',
'47.41287182,9.06562928', 'NONE')

(1409632357250846720, 21245141, 'Ohhh what a match. Now this was nerve
tickling good football ⚽! 🎉Well done CHFR\n#frasui 💪💪 🎉 #euro2020
@ Zürich, S… https://t.co/JRFoqQydUB', '2021-06-28 21:58:38', 'Switzerl
and', '47.3846,8.531600000000001', 'NONE')
```

Word Cloud:

We are creating word cloud based on Text field in the, while creating word cloud I am doing some text cleaning to make it proper word cloud.

```
30]: #PART4:Q2: 1&2
     def cleanTxt(text):
         text = re.sub(r'@[A-Za-z0-9]+', '', text)
         text = re.sub(r'#', '', text)
         text = re.sub(r'RT[\s]+', '', text)
         text = re.sub(r'https:\/\/\S+:', '', text)
         return text

     for rows in (rows_uk, rows_france, rows_swiss):
         print(len(rows))
         out = [row[2] for row in rows]
         cleanedList = [x for x in out if str(x) != 'nan']
         str_cat = ""
         for i in cleanedList:
             out_l = cleanTxt(i)
             str_cat += ''.join(out_l)

         wordcloud = WordCloud().generate(str_cat)

         # Display the generated image:
         plt.imshow(wordcloud, interpolation='bilinear')
         plt.axis("off")
         plt.show()
```

After executing above code we will get below word cloud for three different countries.

United Kingdom word cloud:

France word cloud:

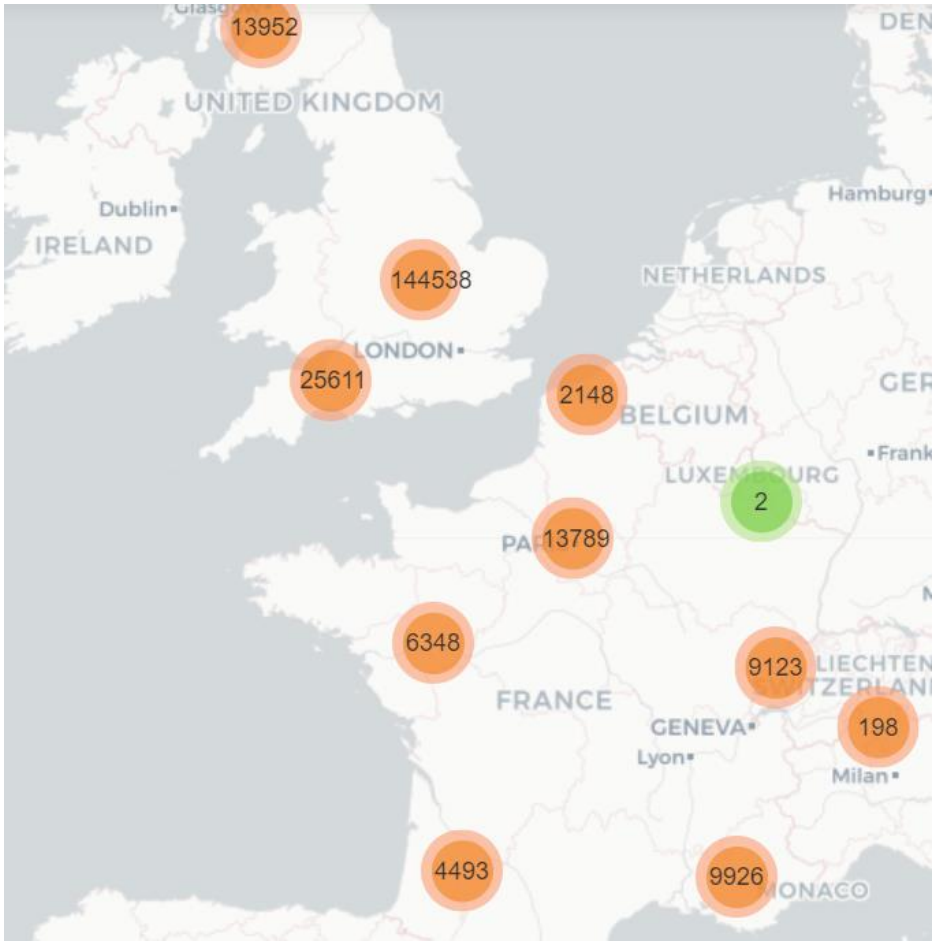
Switzerland word cloud:

Plot tweets locations on a map:
To get tweets location I am reading LattLong field from database for particular countries.

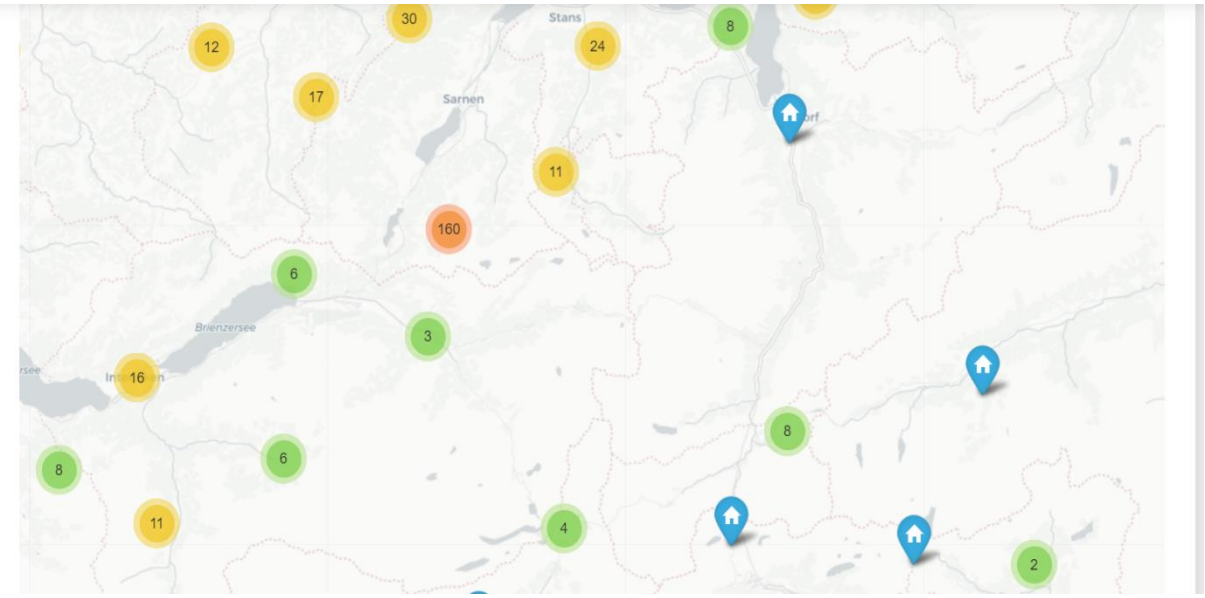
```
32]: #PART4:Q3
countries = ["United Kingdom", "France", "Switzerland"]
data = {}
for country in countries:
    c.execute("""SELECT Id, LattLong FROM twitter WHERE LattLong != "0.0" AND CountryName in ('{}')""".format(country))
    rows = c.fetchall()
    data[country] = rows
```

After getting LattLong for tweet location I am using FastMarkerCluster() to point out the locations on map.
As we can see in the map the tweet's locations are coming from England, France and Switzerland countries.

Map1: Again I have used cluster mechanism to display the tweets.



Map2: Tweets location from exact points, I have zoomed in some part of Switzerland and displays the location of tweets



PART5:
Q1 : The strengths and weaknesses of Twitter as a data source from a technical/statistical Perspective.
Ans:

Pros:

- Twitter data source contains all objects to perform statistical analysis, Data can be acc essed through twitter API. By the method of filtering, we can fetch the particular data for analysis.
- Twitter objects divided into sub-objects (Tweet object, user object, Entities Object, G eo Object etc), These sub-objects help us to identify data resources without any confu sion.
- Each object contains own attributes, this attribute will help us to process or select data based on requirements. Eg: geo locations, place, hashtags, texts are used to identify s pecific discussion.

Cons:

- Due to the large data sets, we got lot of redundant data in json files. One challenge is l ike we need to be careful. while selecting columns and making smaller files from give n data.
- This data contains some unnecessary fields, like simple tweet from user contains lot o f empty objects this leads us to huge data set and NaN values.
- Not all the people use twitter to express their opinions, so there is a lot of chances conclusions can be inaccurate.
- Not all the people in twitter are active only 20% are active this can be lead us to taking opinions of few people and again our final results can be erroneous.
- Some of the tweets are in different language so it is difficulty to analyse actual intention of the user.

Q2: Biases in Twitter data and how they might be mitigated.
Ans:

- First thing is population, The number of populations in the dataset vs the actual population is different. If we take only populations from the dataset for people opinion then the conclusion is going to be mislead.
- People are using different language to express their opinion, this can cause our models will perform poor in some situations. To overcome this one we need to build strong models that should be trained with different categories.
- We can see lot of bots tweets from the dataset, This can be the main problem while gi ving conclusion for particular topics. First, we need to identify and remove or make flag mark these tweets while analysis.
- Due to the privacy and content of the data some results are not matching with actual analysis results

Q3: Ethical and legal concerns about using Twitter data.
Ans:
Ethical Concerns: In accumulating and retrieving tweets to form large datasets it may be failed to obtain informed consent from all of the participants, simply due to the volume of tweets retrieved. Dummy and non-human accounts create lot of problems, these things sometimes ethically incorrect. There are also ethical issues if you decide to reproduce tweets in an academic publication, which have to be handled with care especially concerning tweets related to sensitive topics i.e., obtaining consent before disclosing user IDs or tweets.

Legal Concerns: Sharing of data (user tweets) is prohibited under Twitter's API Terms of Se rvice, however, people (researchers) can share the tweet ID numbers, associated with each tweet, which can be used by other researchers to obtain Twitter datasets. If, for any reason, it is not possible to share tweet Identification numbers then sharing the keywords and retrieval time of the data, may allow researchers to obtain a similar dataset. Some Terms and conditions of twitter services specifically state user's posts that are available for public and ea sier to access.

Q4: The use of Twitter to study the effectiveness of lockdown policies.
Ans:

- Twitter is one of the main platforms where the lockdown policies information's are spreading quickly. Eg. covid guidelines, preventive measure, rules and regulations for public, maintaining safe distance with people, updates about vaccines etc.
- When the lockdown was announced by government with different countries, This information also announced in twitter in that time the tweets spreads within few minutes across the borders. This information is very useful because travellers can easily get to know about country policies and lockdown procedures.
- There is a small initiative taken care by twitter to give latest updates about covid and lockdown policy. Twitter added new tab about covid, this can be used to identify latest information about covid and travel restrictions across the borders etc.
- Twitter is also keep on giving updates about positive, negative cases and death rates in each country, safety measures about covid etc.
- Twitter contains lots of threads i.e discussing about lockdown policies and people opinion for these policies. Some of the threads are discussing about people are struggling in because of lockdown. Some people lost their jobs and they are expressing their emotions in tweets etc.

These data can be used to know about the effectiveness of lockdown and also impact of human life during lockdown.