Disease Analysis based on Twitter Data

Harika Reddy Vundyala
Computer Science
University of Missouri-Kansas City
hvf2m@umkc.edu

Rakesh Reddy Gaddam
Computer Science
University of Missouri-Kansas City
rgynf@umkc.edu

ABSTRACT

There are numerous social media platforms like Facebook, Instagram, and twitter. These platforms have unlimited data which can be used to analyze many trends in the present day. In this project we are going to utilize the twitter data to represent the disease analysis. That is, we are going to retrieve data from twitter using twitter API and there is also an alternate way to retrieve tweet data using Twint or snscrape library for retrieving unlimited tweets without any authentication. By using pyspark and twitter tweet data from 2008 to present day, we will portray the comparison between the disease using matplotlib, this will explain which disease has affected the people the most. Also, we shall be presenting a comparison between pre and post covid. The main motive of this project is to help create awareness among general people and healthcare departments to concentrate on preventing or working on treatments for respective diseases.

KEYWORDS

Tweepy, stylecloud, twitter API, pyspark, snscrape, twint

1. INTRODUCTION

Twitter is the most trending social media platform right now, it was founded on March 21, 2006, in San Francisco. It is trending because it is owned by Elon Musk recently and all famous celebrities use twitter. It is a platform used for many things such as sharing information. One can tweet about something, and it can be retweeted, and a lot of discussions can be done on that tweet by users in comment section.

Twitter also has a section to check out trending topics, one can see what's going on in the world, the tweet or topic which is being discussed most will be in the top position.

To perform Big Data Analysis, we can make use of any field and can take data related to that field from twitter. We thought, we could select the "Disease" field and retrieve the data from twitter. Based on the tweets from twitter, we wanted to predict few analysis on Diseases using the tweets tweeted by different people. Firstly, we tried to collect the tweets from twitter API based on some key words related to Disease using Hashtag. For that we generated the API keys and have written the code for it, but we couldn't retrieve the data through it as we are having only essential access rather than elevated access to developer account. So, for now with no twitter API and no authentication, we have retrieved the data using python package called snscrape. Later we analyze the data we retrieved and perform few SQL queries and represent the results in graphical way.

As mentioned, we would like to perform few SQL queries on the following tasks.

1. What are the most popular diseases all over the world
2. Comparison between pre and post covid
3. What are tweets posted by verified users
4. What are the most Popular Tweeted Words

2. RELATED WORK

There were many works related to big data analytics. Firstly, we were looking into big data analytics concept like how it works, how it benefits and what challenges it has. So, this information will let us know how to go ahead with our project. With all the learnings from our class and referring websites such as Tableau, Towardsdatascience, we are performing this project.

The most important feature for any Big Data related project is Dataset. So as aforementioned, we are retrieving the twitter data through twitter API. Before proceeding with it, it is necessary to refer the information how we need to do it. We have referred into few works which are related to this which explains how to retrieve the data with twitter API. Also, with no API and no authentication, we can retrieve the data using a python package. Towards data science website excellently handles all the big data concepts and we have gone through this python package called snscrape in towardsdatasceince where it explained all those in a clear manner. That led us to retrieve the twitter tweets.

Next, we are going to do this project on pyspark and as we have performed queries earlier which was given for the programming assignment 1, it has been very helpful to perform SQL queries now.

3. PROPOSED TECHNIQUES AND TOOLS

To perform the tasks mentioned in the Introduction, we will perform the following steps.

Steps:

1. Collecting twitter tweets
2. Data Processing in spark
3. Visualizing outcomes

3.1 Tweepy

Tweepy is a python library for accessing the twitter API. Simply, we can say it as an interface to access twitter API from any python application. With tweepy, we can do things such as like, comment, retweet and so much more. We can install tweepy simply by typing the command pip install tweepy

3.2 Twitter Developer Account

We have created a developer account for the API keys to retrieve tweets. For that we need to have a developer twitter account with an elevated access where we can generate API key, API Key Secret, Access Token, Access Token secret to retrieve the tweets.

Following are the keys we have generated.

API Key –
ClucZCPhH7xFwNPNZrPc7JE45

API Key Secret -
c0O6ODpBvBNsUL89ARfFw8WNCeon7X6fmEDeA0zOKIwAfVOSxr

Access Token –
1589031861069676545-FnAfrdNWcjbV7FDdy7zN95GEj9UEPh

Access Token Secret –
rBKJtnyCPTSrPXH8e9xAqxejCKCawNhgGAjNWeurxihLL

3.3 snscrape

There is also another way for retrieving the tweets if you don't have a twitter API. The most advisable way of extracting tweets is using **TWINT or snscrape** library. snscrape library will help in getting unlimited tweets without any authentication with just a few lines of code. The biggest advantage of this kind of library is that we can be able to retrieve unlimited tweets without any restrictions and even the past tweets from the year 2006 unlike twitter API. This is extremely useful library for making word clouds.

3.4 Stylecloud

Stylecloud is a python package which has popular word cloud package, and it has useful features which creates truly unique word clouds. By default, it handles stopwords and has an amazing feature where we can create theme on basis of font awesome icons. That means, it uses as a logo of which you want like google, twitter icon so on.

4. DATA AND METHODOLOGY

As mentioned, there are two ways for retrieving twitter data which is with twitter API and snscraper library (without API). Now, we'll have a look on how to retrieve tweets with twitter API followed by using snscraper library.

4.1 Collecting the tweets with twitter API

As aforementioned, we are going to retrieve the tweets using the keys generated by establishing the connection to twitter API.

```
consumer_key = "ClucZCPhH7xFwNPNZrPc7JE45"
consumer_secret = "c0O6ODpBvBNsUL89ARfFw8WNCeon7X6fmEDeA0zOKIwAfVOSxr"
access_token = "1589031861069676545-FnAfrdNWcjbV7FDdy7zN95GEj9UEPh"
access_token_secret = "rBKJtnyCPTSrPXH8e9xAqxejCKCawNhgGAjNWeurxihLL"
```

```
#Authenticate to Twitter
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)
```

Fig 4.1.1: Twitter Authentication

We have taken the generated keys into the colaboratory, and the next part is tweepy authentication. If this successfully runs without any kind of error, then we have successfully managed to authenticate and establish the connection to twitter API through tweepy. With this, we can be able to extract tweets of particular account, particular hashtag and many more. Here we have given the username as "yusufsarwar" to extract the tweets of Professor's account.

```
username='yusufsarwar'
tweets = api.user_timeline(screen_name=username, count=1,
                           include_rts=True, tweet_mode='extended')
columns=['User','User_id','screen_name','description',
         'followers_count','friends_count','text']
data=[]
for tweet in tweets:
  data.append([tweet.user.name,tweet.user.id, tweet.user.screen_name,
               tweet.user.description, tweet.user.followers_count,
               tweet.user.friends_count,tweet.full_text])
df=pd.DataFrame(data,columns=columns)
df.head()
```

Fig 4.2.2: Displaying the code for retrieving tweets of a user

4.2 Collecting tweets using snscrape

Here, we'll see how to extract tweets using snscraper and perform the tasks mentioned in the Introduction. For that we have taken the data based on diseases. Given few hashtags in twitter account such as #Malaria, #Diarrhea, etc. And collected the data within 2008-2022 period

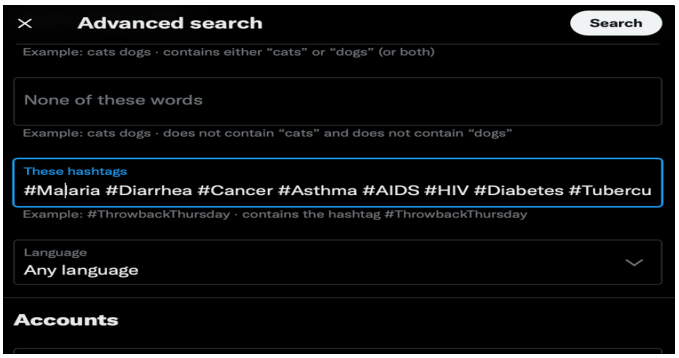


Fig 4.2.1: Displaying the advanced search of twitter

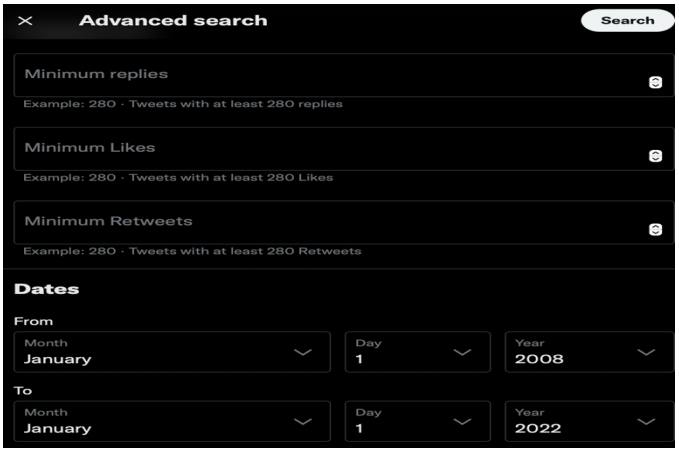
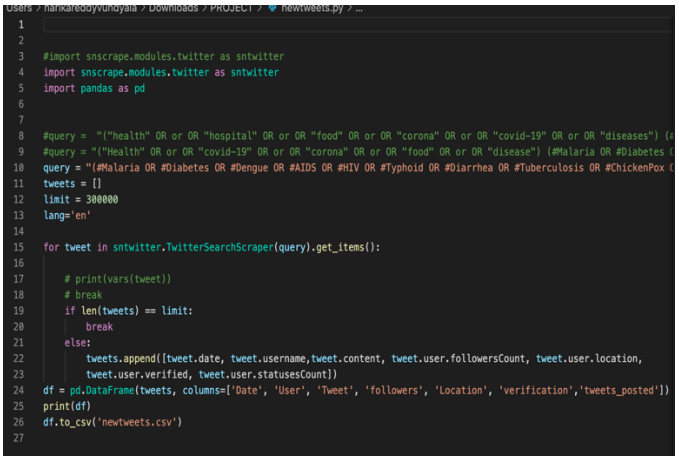


Fig 4.2.2: Displaying the advanced search of twitter

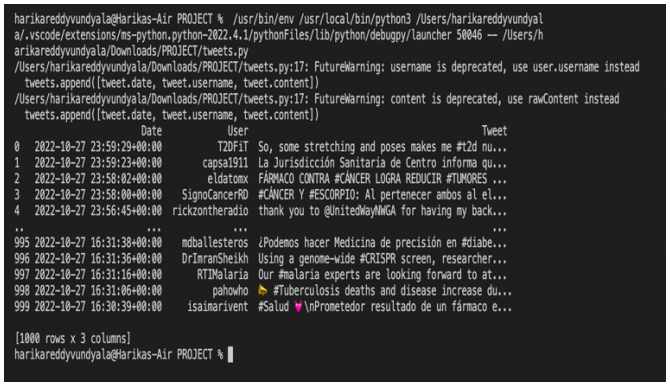
Fig 4.2.1 and 4.2.2 lets us select which period of tweets we need or what must be included in it and not included in it so on. Next, we've written the code in visual code using python package snscrape as mentioned.



```
#import snscrape.modules.twitter as sntwitter
import snscrape.modules.twitter as sntwitter
import pandas as pd


#query =  "("health" OR or OR "hospital" OR or OR "food" OR or OR "corona" OR or OR "covid-19" OR or OR "diseases") (
#query = "("Health" OR or OR "covid-19" OR or OR "corona" OR or OR "food" OR or OR "disease") (#Malaria OR #Diabetes
query = "(#Malaria OR #Diabetes OR #Dengue OR #AIDS OR #HIV OR #Typhoid OR #Diarrhea OR #Tuberculosis OR #ChickenPox
tweets = []
limit = 300000
lang='en'

for tweet in sntwitter.TwitterSearchScraper(query).get_items():

    # print(vars(tweet))
    # break
    if len(tweets) == limit:
        break
    else:
        tweets.append([tweet.date, tweet.username,tweet.content, tweet.user.followersCount, tweet.user.location,
        tweet.user.verified, tweet.user.statusesCount])
df = pd.DataFrame(tweets, columns=['Date', 'User', 'Tweet', 'followers', 'Location', 'verification','tweets_posted'])
print(df)
df.to_csv('newtweets.csv')
```

Fig 4.2.3: Displaying the code for retrieving tweets

As shown in the Fig 4.2.1, those hashtags will be our query. We basically hit search, copy those, and paste it under query then that will let you retrieve the tweets. Here we have given the limit as 300000 and has given date, content, user, followers, verification, location, followers, tweets posted as attributes. Here, the verifications describe which user has a verified account or not, if the user is verified, we will be notified as True. Next, Locations says the location of the tweet that has been posted. Tweets posted describes the number of tweets posted by a particular user. As seen in the last statement in Fig 4.2.3, we are saving these data into the csv file



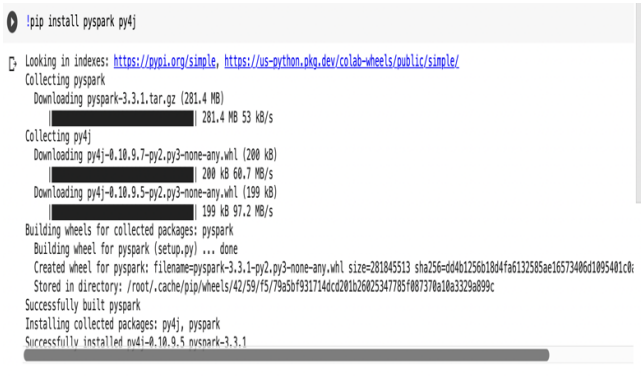
```
harikareddyvundyala@Harikas-Air PROJECT %  /usr/bin/env /usr/local/bin/python3 /Users/harikareddyvundyal
a/.vscode/extensions/ms-python.python-2022.4.1/pythonFiles/lib/python/debugpy/launcher 50046 -- /Users/h
arikareddyvundyala/Downloads/PROJECT/tweets.py
/Users/harikareddyvundyala/Downloads/PROJECT/tweets.py:17: FutureWarning: username is deprecated, use user.username instead
  tweets.append([tweet.date, tweet.username, tweet.content])
/Users/harikareddyvundyala/Downloads/PROJECT/tweets.py:17: FutureWarning: content is deprecated, use rawContent instead
  tweets.append([tweet.date, tweet.username, tweet.content])
                    Date            User                                              Tweet
0    2022-10-27 23:59:29+00:00       T2DFiT  So, some stretching and poses makes me #t2d nu...
1    2022-10-27 23:59:23+00:00      capsa1911  La Jurisdicción Sanitaria de Centro informa qu...
2    2022-10-27 23:58:02+00:00       eldatomx  FÁRMACO CONTRA #CÁNCER LOGRA REDUCIR #TUMORES ...
3    2022-10-27 23:58:00+00:00   SignoCancerRD  #CÁNCER Y #ESCORPIO: Al pertenecer ambos al el...
4    2022-10-27 23:56:45+00:00  rickzontheradio  thank you to @UnitedWayNWGA for having my back...
..                         ...             ...                                               ...
995  2022-10-27 16:31:38+00:00   mdballesteros  ¿Podemos hacer Medicina de precisión en #diabe...
996  2022-10-27 16:31:36+00:00   DrImranSheikh  Using a genome-wide #CRISPR screen, researcher...
997  2022-10-27 16:31:16+00:00      RTIMalaria  Our #malaria experts are looking forward to at...
998  2022-10-27 16:31:06+00:00        pahowho  #Tuberculosis deaths and disease increase du...
999  2022-10-27 16:30:39+00:00    isaimarivent  #Salud \nPrometedor resultado de un fármaco e...

[1000 rows x 3 columns]
harikareddyvundyala@Harikas-Air PROJECT %
```

Fig 4.2.4: Displaying the terminal after running the code

Fig 4.2.4 shows the data we retrieved from the twitter. And we will import this csv file in colaboratory and run the tasks.

4.3 Data Processing in Spark

Firstly, we have imported all the required packages and libraries such as Py4j by starting a new Google Colaboratory and have started a spark session.



```
!pip install pyspark py4j

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pyspark
  Downloading pyspark-3.3.1.tar.gz (281.4 MB)
     |████████████████████████████████| 281.4 MB 53 kB/s
Collecting py4j
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
     |████████████████████████████████| 200 kB 60.7 MB/s
  Downloading py4j-0.10.9.5-py2.py3-none-any.whl (199 kB)
     |████████████████████████████████| 199 kB 97.2 MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.3.1-py2.py3-none-any.whl size=281845513 sha256=dd4b1256b18d4fa6132585ae16573406d1095401c8
  Stored in directory: /root/.cache/pip/wheels/42/59/f5/79a5bf931714dcd201b26025347785f087370a10a3329a899c
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.5 pyspark-3.3.1
```

Fig 4.3.1: Initializing Spark session

Now we have pulled the data set which is in csv format where we have retrieved the tweets in visual code using snscraper and created a data frame in spark. As part of task, we have selected Appname and Installation count with limit as 20. Here, as mentioned we have loaded the dataset into google colab and have copied the file path after creating the data frame.

4.4 Dataset

```
[2] from pyspark.sql import SparkSession
```

```
[3] spark = SparkSession.builder.appName("twitter_pyspark").getOrCreate()
```

```
[4] df = spark.read.csv("/content/tweetdata.csv", header=True, inferSchema=True)
```

Fig 4.4.1: Initializing Spark session

Displaying the records using show() function.

```
+-------------------+-------------+--------------------+---------+------------+-------------+
|               Date|         User|               Tweet|followers|verification|tweets_posted|
+-------------------+-------------+--------------------+---------+------------+-------------+
|2022-10-27 23:16:...|pikesley_ebooks|Patrol cover-vers...|       62|       False|        75755|
|2022-10-27 23:12:...|    BridgeGap1|@TCLChalk @Southl...|      641|       False|         3288|
|2022-10-27 23:08:...|  BonitaMelvin|@BonitaMelvin #co...|      144|       False|          442|
|2022-10-27 23:07:...|  MedTechStrat|Water vapor techn...|     3844|       False|         5612|
|2022-10-27 23:00:...|    5tephanieM|@KizzyPhD I don't...|      324|       False|         7416|
|2022-10-27 22:53:...|icecreamofhour|The ice cream for...|        2|       False|         1875|
|2022-10-27 22:34:...| NewFoundWings|I feel as though ...|      734|       False|        21487|
|2022-10-27 22:34:...|   KablamoLong|Hillston Randomis...|        0|       False|          230|
|2022-10-27 22:27:...|  silver_charm|Awful lot of peop...|      276|       False|         9334|
|2022-10-27 22:16:...| Junobutclover|weezer is a slur ...|        3|       False|           77|
|2022-10-27 22:13:...|   KablamoLong|Wollongong Random...|        0|       False|          230|
|2022-10-27 22:09:...|   naomisloyan|I had some good n...|      521|       False|         3113|
|2022-10-27 22:00:...|      moebotte|@HanZou10470 so c...|        6|       False|         1747|
|2022-10-27 21:52:...|  bigolpoofter|@danahull @WHCOS ...|     1602|       False|        40984|
|2022-10-27 21:46:...|       milposp|#Russia is sendin...|      135|       False|         6865|
|2022-10-27 21:40:...|   MTrevNewton|I am looking look...|      845|       False|         2542|
|2022-10-27 21:28:...|  craigjlennox|#Bravo for openly...|       40|       False|          734|
|2022-10-27 21:01:...| newlypositive|And @CheddarGorge...|     2649|       False|        24451|
|2022-10-27 21:01:...| HeatherHenry4|@NaomiBe44702921 ...|     5006|       False|        19113|
|2022-10-27 20:55:...|   eveslipknot|@CheddarGorgeous ...|      101|       False|         2537|
+-------------------+-------------+--------------------+---------+------------+-------------+
only showing top 20 rows
```

Fig 4.4.2: Streaming twitter tweets

Here displaying the first 20 records of data. Next, we started creating a new column for performing the first query

5. ANALYSIS AND RESULTS

5.1 Different Popular diseases all over the world

In this query, we are counting the total number of popular diseases all over the world using spark.sql function. For that we have assigned the keywords called diseases to the associated tweets and have written the appropriate query.

```
df2.createOrReplaceTempView("TWEETS")
df3=spark.sql('''SELECT Date, User, lower(Tweet) as tweets,followers,verification,tweets_posted,
CASE
WHEN lower(Tweet) like "%diabetes%" THEN 'Diabetes'
WHEN lower(Tweet) like "%cancer%" THEN 'Cancer'
WHEN lower(Tweet) like "%malaria%" THEN 'Malaria'
WHEN lower(Tweet) like "%dengue%" THEN 'Dengue'
WHEN lower(Tweet) like "%diarrhea%" THEN 'Diarrhea'
WHEN lower(Tweet) like "%tuberculosis%" THEN 'Tuberculosis'
WHEN lower(Tweet) like "%hiv%" THEN 'HIV'
WHEN lower(Tweet) like "%aids%" THEN 'AIDS'
WHEN lower(Tweet) like "%kidneycancer%" THEN 'KidneyCancer'
WHEN lower(Tweet) like "%asthma%" THEN 'Asthma'
WHEN lower(Tweet) like "%braincancer%" THEN 'BrainCancer'
WHEN lower(Tweet) like "%lungcancer%" THEN 'LungCancer'
WHEN lower(Tweet) like "%cold%" THEN 'Cold'
WHEN lower(Tweet) like "%fever%" THEN 'Fever'
WHEN lower(Tweet) like "%typhoid%" THEN 'Typhoid'
ELSE 'NA'
END AS Diseases
FROM TWEETS''')
```

Fig 5.1.1: Code for different popular diseases

```
+------------+-----------------------+
|    Diseases|tweets_count_on_diseases|
+------------+-----------------------+
|      Cancer|                  94499|
|    Diabetes|                  25940|
|         HIV|                  21044|
|        Cold|                  17318|
|      Asthma|                  15018|
|     Malaria|                   9283|
|        AIDS|                   5701|
|     Typhoid|                   4145|
|       Fever|                   3890|
|      Dengue|                   2029|
|    Diarrhea|                   2006|
|Tuberculosis|                   1828|
+------------+-----------------------+
```

Fig 5.1.2: Result for popular disease count

Now, we are using matplotlib for visual representation of the results.

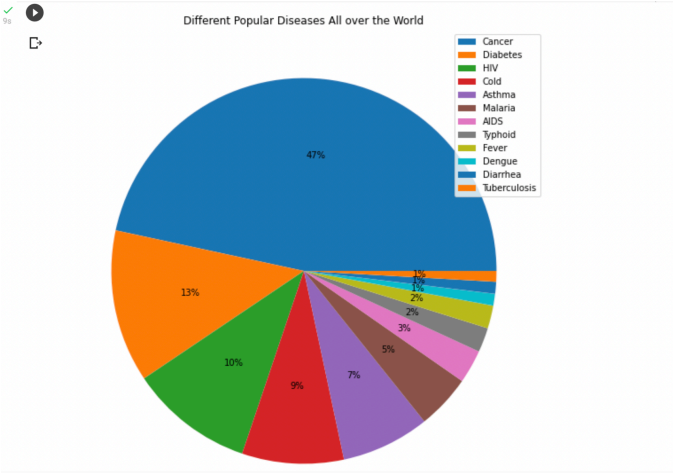


Fig 5.1.3: Pie chart of popular disease count

5.2 Popular Diseases in covid time

In this query, we analyze the difference between results of normal time and the covid period. As expected, we have analyzed the major change in disease cold and asthma where in the normal time the cold and asthma count were more than half of the count in covid time.

Query: Popular Diseases in Covid time

```
#query popular diseases in covid time
from pyspark.sql import functions as F
from pyspark.sql.functions import month,year
from pyspark.sql.functions import to_date
df3.createOrReplaceTempView("TWEE")
df4=spark.sql('''SELECT Diseases as Diseases1,COUNT(*) as tweet_count FROM TWEE group by Diseases order by 2 DESC''')
covid_time= df3.filter("date BETWEEN '2020-03-03' AND '2022-03-03'")

covid_time = covid_time.groupby('Diseases').agg(F.count('Diseases').alias('tweet_count_during_covid'))

#sort the resulted data in descending order
covid_time = covid_time.sort(covid_time.tweet_count_during_covid.desc())
df5 = covid_time.join(df4, covid_time.Diseases==df4.Diseases1, "inner")
df5.show()
df5 = df5.toPandas()
#df5.info()

df5.plot(x="Diseases", y=["tweet_count_during_covid","tweet_count"], kind="bar", figsize=(9, 8))
```

Fig 5.2.1: Code for disease count in covid

Diseases	tweet_count_during_covid	Diseases1	tweet_count
Cold	11105	Cold	17318
Tuberculosis	1200	Tuberculosis	1828
Diabetes	17042	Diabetes	25940
Fever	2959	Fever	3890
Malaria	5634	Malaria	9283
Cancer	61002	Cancer	94499
AIDS	4155	AIDS	5701
Typhoid	1380	Typhoid	4145
Asthma	10059	Asthma	15018
HIV	14760	HIV	21044
Dengue	1264	Dengue	2029
Diarrhea	1378	Diarrhea	2006

Fig 5.2.2: Result for diseases in covid

Now, we are using matplotlib for the representation of results in the bar plot

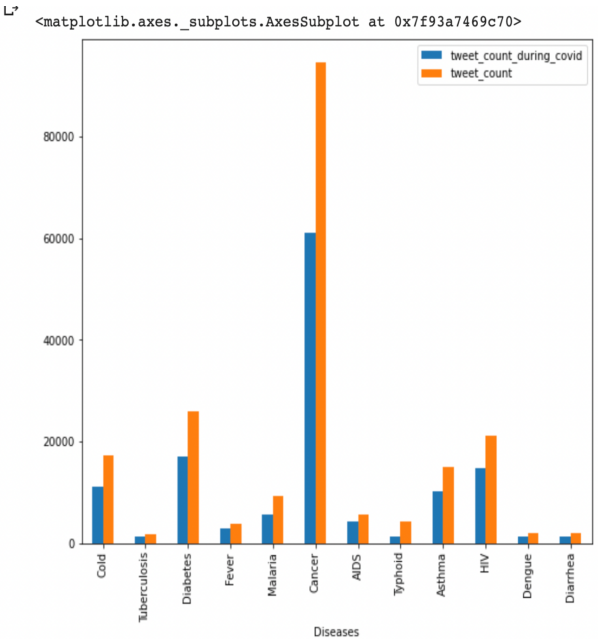


Fig 5.2.3 Bar plot for covid disease count

5.3 Tweets posted by verified users

In this query, we are analyzing the tweets related to diseases posted by the verified users and comparing them with original number of tweets they have posted. Also, this query helps how well the content posted by verified users will be reached to people as the verified users are generally celebrities, politicians and so on. Simply, people usually believe that the tweets posted by the verified users are legit and start believing the tweets and it will help in creating the awareness about the diseases.

Query Tweets Posted by Verified Users

```
#query4 Tweets posted by Verified Users
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plot
data1=df3.filter(df3.verification==True)
#data1.show()
#data1.count()
verified_users = data1.groupby('User','followers','tweets_posted').agg((F.count('User').alias('tweets_on_diseases')))
verified_users= verified_users.filter((verified_users.tweets_on_diseases>=50))
verified_users = verified_users.sort(verified_users.tweets_on_diseases.desc())
verified_users.show()
df6=verified_users.toPandas()
df6['followers1']=df6['followers'].astype(str).astype(int)
df6['tweets_posted']=df6['tweets_posted'].astype(str).astype(int)
df6['tweets']=df6['tweets_on_diseases'].astype(str).astype(int)

df6.plot(x="User", y=["tweets_posted", "followers1" ], kind="bar", figsize=(9, 8))
```

Fig 5.3.1: Code for verified user tweets

User	followers	tweets_posted	tweets_on_diseases
ViiVUS	6947	6383	221
JFreemanDaily	5747	21194	121
TedOkonCOA	5313	23587	100
LizHighleyman	14328	30822	98
LifeExtension	151682	233405	73
parthaskar	43005	141943	71
diaTribeNews	20971	20645	55
NovoNordiskLive	10901	2908	53
GHTCoalition	7420	11037	50

Fig 5.3.2: Result for verified user tweets

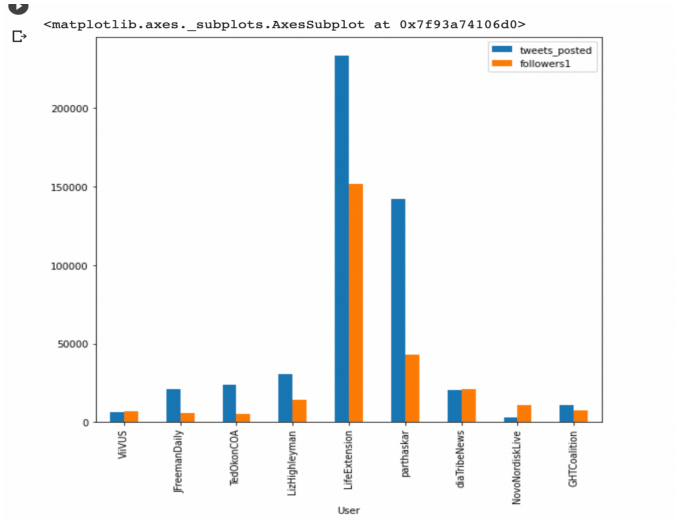


Fig 5.3.3: Bar plot between followers count and tweets posted

5.4 Most Tweeted words

Here, we are using style cloud library for representing the most tweeted words in twitter image.

```
import stylecloud

stylecloud.gen_stylecloud(file_path='/content/tweets2.csv/part-00000-90f9b879-fbfa-46e0-bfb4-e1b8b093c05c-c000.csv',
                          icon_name='fab fa-twitter',
                          palette='colorbrewer.diverging.Spectral_11', #colorbrewer.qualitative.Paired_3',
                          background_color='black',
                          gradient='horizontal',
                          stopwords = True ,
                          custom_stopwords = ['TCO','T','CO','HTTPS','RT','THE','IS','WITH','ON','HTTPS','AMP','i','and','me','my','myself','we','our',
                                              'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours','yourself', 'yourselves',
                                              'he', 'him', 'his', 'himself','she', "she's",'her','hers', 'herself','it', "it's",'its', 'itself', 'they',
                                              'them', 'their','theirs','themselves', 'what', 'which', 'who','whom', 'this','that', "that'll", 'these',
                                              'those', 'am','is', 'are', 'was','were','be', 'been', 'being', 'have', 'has','had', 'having', 'do', 'does',
                                              'did','doing','a', 'an','the', 'and', 'but', 'if', 'or','because', 'as','until','while','of', 'at', 'by',
                                              'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below','to',
                                              'from', 'up','down','in','out', 'on', 'off', 'over', 'under','again', 'further','then','once', 'here','there',
                                              'when', 'where', 'why', 'how','all', 'any', 'both', 'each','few','more', 'most', 'other', 'some', 'such',
                                              'nor', 'not', 'only', 'own', 'same', 'so','than','too','very','s', 't','can','will', 'just','don', "don't",
                                              'should', "should've",'now', 'd','ll','m', 'o', 're','ve','y','ain', 'aren', "aren't",'couldn', "couldn't",
                                              'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't",'haven', "haven't",'isn', "isn't",
                                              'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't",'shouldn', "shouldn't",
                                              'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn',"wouldn't",'for','no'])
```

```
from IPython.display import Image
Image('stylecloud.png')
```

Fig 5.4.1: Code for most tweeted words

Fig5.4.2: Visual representation of most tweeted words

6. CONCLUSION

This project gives a detailed explanation of the current diseases using the data retrieved from Twitter using the Twitter API. This shows how data from any social media platform can be beneficial and used for good causes. The bar graphs give a clear difference between various diseases which is easily comprehended by anyone.

REFERENCES

[1] *Authentication.* (n.d.). Retrieved from Developer Platform:https://towardsdatascience.com/an-extensive-guide-to-collecting-tweets-from-twitter-api-v2-for-academic-research-using-python-3-518fcb71df2a

[2] Edward, A. (2021). *Extensive guide for collecting tweets from twitter API.* Retrieved from Towards Data Science: https://towardsdatascience.com/an-extensive-guide-to-collecting-tweets-from-twitter-api-v2-for-academic-research-using-python-3-518fcb71df2a

[3] Raja, A. M. (2019). *Generate modern stylish wordcloud.* Retrieved from TowrdsDataScience: https://towardsdatascience.com/generate-modern-stylish-wordcloud-with-stylecloud-9cbb059696d2

[4] Sistilli, A. (2019). *twitter data mining using python.* Retrieved from Developers: https://towardsdatascience.com/generate-modern-stylish-wordcloud-with-stylecloud-9cbb059696d2

[5] *Twitter API.* (2019). Retrieved from Developer Platform: https://developer.twitter.com/en/docs/twitter-api