ANALYSIS ON US ELECTION USING SOCIAL MEDIA

TEAM MEMBERS:

1. NIKHITHA, KOLLURI (16290856)

2. CHAITANYA, MALLEPUDI (16291222)

3. ESWAR, VALLURU (16292742)

4. SRI SAI NITHIN CHOWDARY, DUKKIPATI (16291615)

2. INTRODUCTION:

• MOTIVATION:

The world has been advancing significantly day by day, so as the data. Nowadays the data has become so vast that it could not be stored in a single cluster or a machine, thus the concept of big data arisen, and its services have been implemented in various sectors. Many tools and technologies are arriving in the market very frequently to work with this big data. We thought of analyzing social media using the existing techniques, visualize the data, aggregate it, and model it in a user-friendly way.

• SIGNIFICANCE:

People are very curious about the recent trending issues across the globe, Since the 2020 US elections have been trending now the people want to know about the different views across the country on participating election candidates. So, we are collecting social media data to analyze different views and opinions, so that people can get awareness.

• OBJECTIVES:

Our Undertaking's fundamental thought is to do the ETL cycle utilizing Spark Cluster Handling, Cleaning the data, performing sentimental analysis, visualizing the data, and Incorporating visualization with Web UI. The wellspring of our framework is Twitter information and we would utilize Tweepy API to gather the information.

• FEATURES:

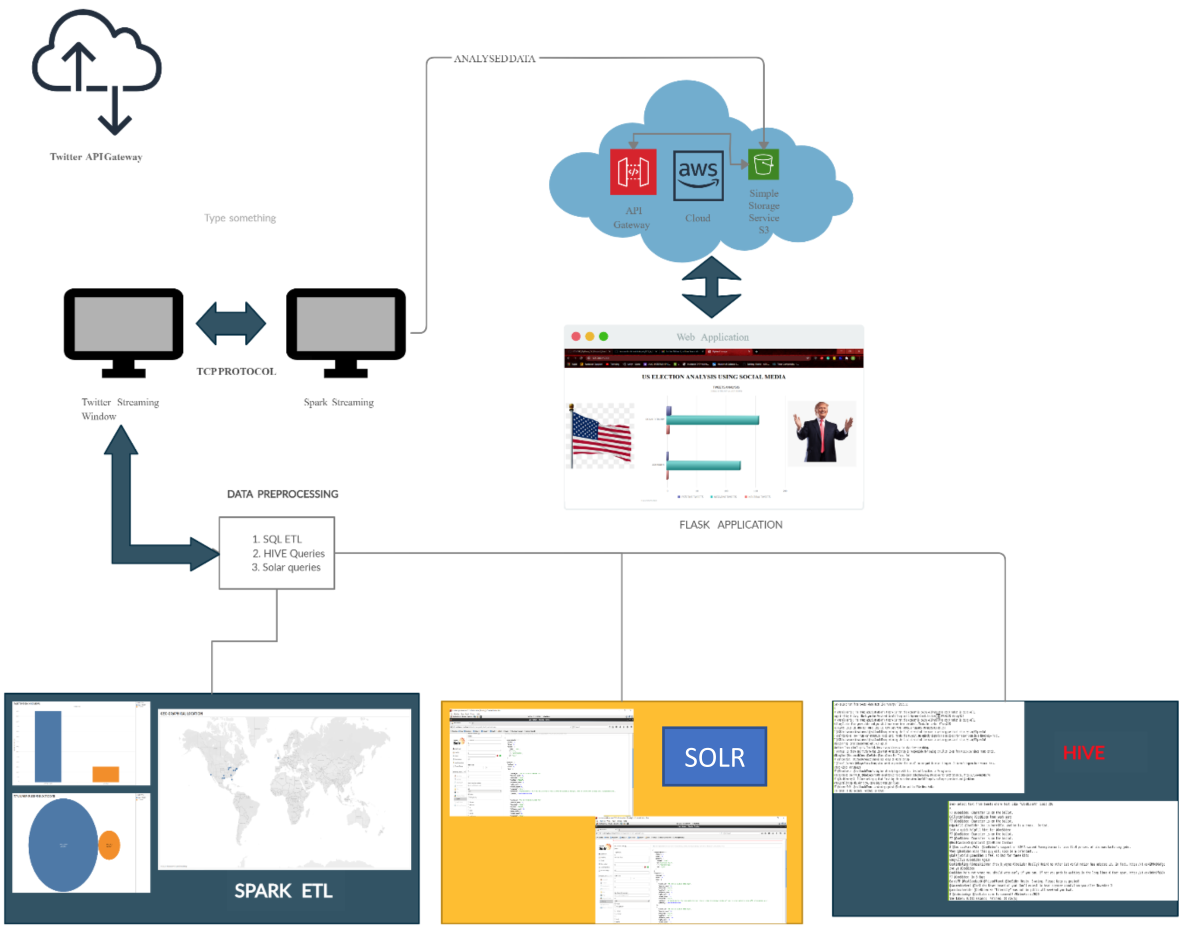
This project includes streaming of live Twitter data and processing it through SPARK ETL and using TextBlob from NLTK to process the data and predict the sentiment and Creating a Web UI incorporating the visualization of data and deploying it to the Cloud environment.

3. Background:

We all are pretty much interested in developing machine learning algorithms but most of the time we have a well formatted data which in ready made in csv format so that we can apply what ever algorithm we need to apply in easiest possible manner with the help of all available open source predefined ML libraries, but the main challenge for the scientist comes while collecting the data. With the bigdata courses that we had taken in pervious semesters, we got awareness on hadoop and spark eco system. So, with all the knowledge that we have gained and with all the available resources we collected the data from different data sources and analysed them. To make our application real time in this real world we choose to stream the data instead of collecting them and later working on them. Thus proceeded with spark streaming.

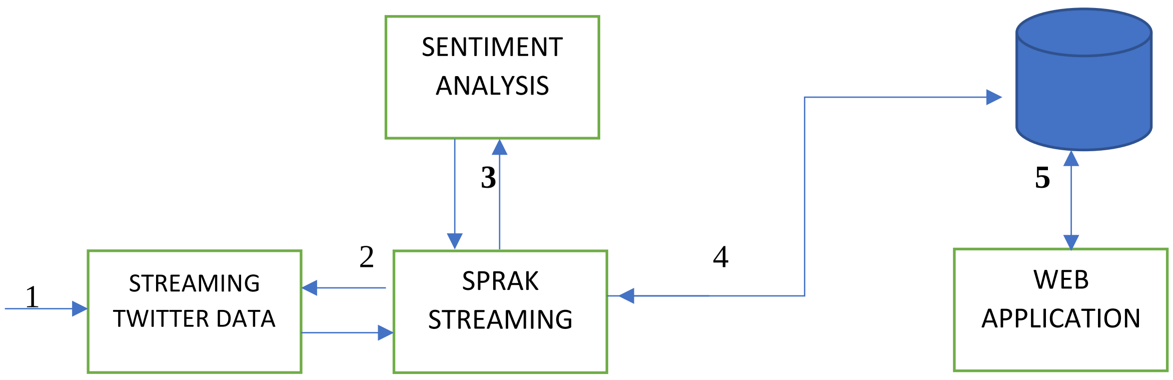
4. OUR MODEL:

4.1. ARCHITECTURE:



1. We have done some data pre-processing using Hive, Solr, and SparkSql after collecting tweets with tweepy API using python.
2. Streamed and processed data using spark streaming and provided sentimental analysis over the data.
3. We stream the data from twitter using spark streaming.
4. Stored the data after performing sentiment analysis to AWS s3 bucket
5. Finally web application retrieves the updated data from AWS API.

4.2 WORKFLOW:



In here, we are streaming the data from twitter using tweepy, done some data pre-processing using Hive, Solr, and SparkSql, Streamed and processed data using spark streaming and provided sentimental analysis over the data, stored the data to AWS s3 bucket and, Finally web application retrieves the updated data from database.

5. DATA SET:

Twitter data on Donald Trump and Joe Biden (US ELECTION 2020)

This dataset has all the recent tweets about Joe Biden and Donald Trump and tweet json object has all the attributes which gives a precise and vital view about what and where things are going around the candidates.

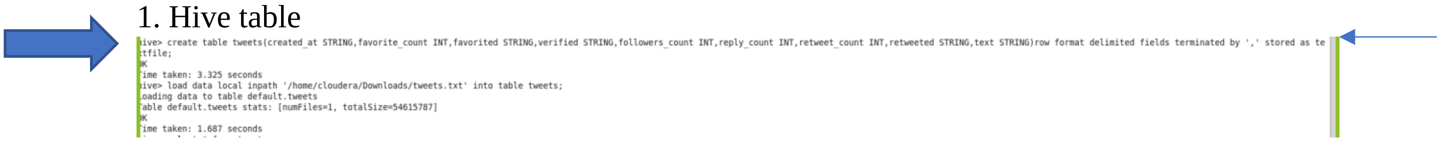
6. ANALYSIS OF DATA:

6.1. DATA PRE-PROCESSING:

6.1.1. HIVE:

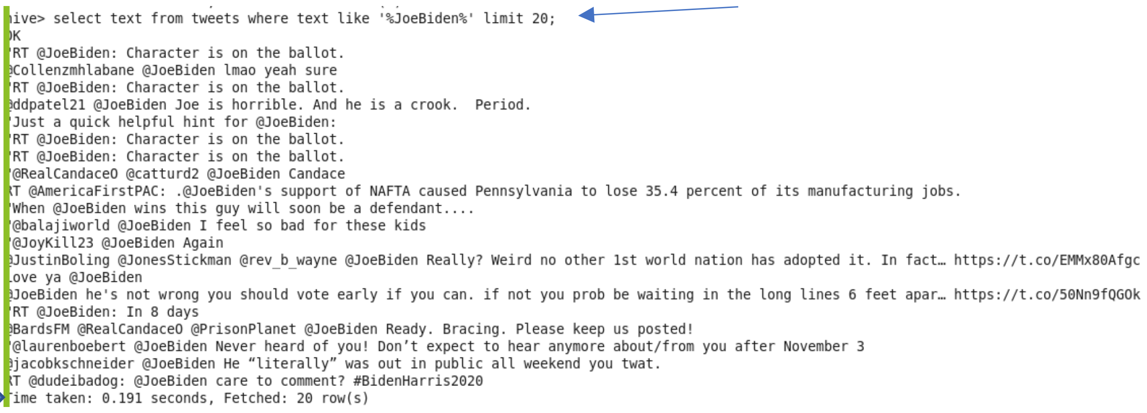
Firstly, we have created hive table and loaded the json data that we have streamed using the tweepy library in python. We have written some queries to analyse whether the streamed data is related to our topic or not. Further, we have verified whether there is any imbalance for the candidates, so that there are no inconsistencies during the sentiment analysis phase.

1. Hive table



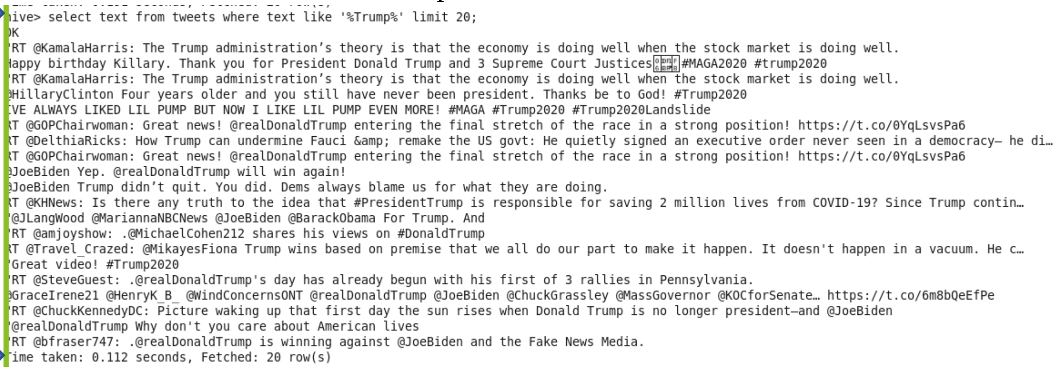
```
ive> create table tweets(created_at STRING,favorite_count INT,favorited STRING,verified STRING,followers_count INT,reply_count INT,retweet_count INT,retweeted STRING,text STRING)row format delimited fields terminated by ',' stored as te
tfile;
K
ime taken: 3.325 seconds
ive> load data local inpath '/home/cloudera/Downloads/tweets.txt' into table tweets;
oading data to table default.tweets
able default.tweets stats: [numFiles=1, totalSize=54615787]
K
ime taken: 1.687 seconds
```

2. Tweets that are Joe Biden



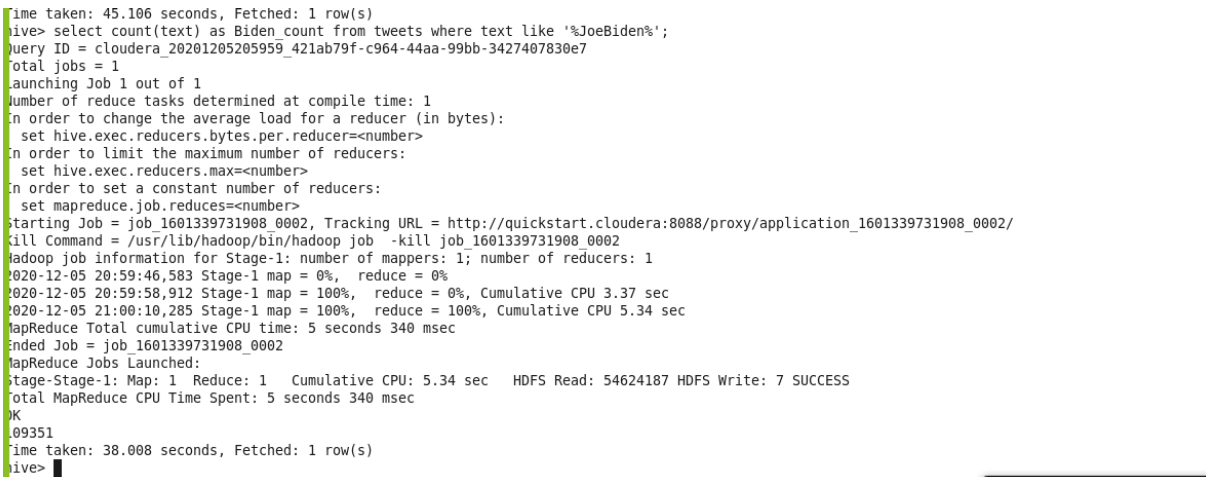
```
ive> select text from tweets where text like '%JoeBiden%' limit 20;
K
'RT @JoeBiden: Character is on the ballot.
Collenzmhlabane @JoeBiden lmao yeah sure
'RT @JoeBiden: Character is on the ballot.
ddpatel21 @JoeBiden Joe is horrible. And he is a crook.  Period.
'Just a quick helpful hint for @JoeBiden:
'RT @JoeBiden: Character is on the ballot.
'RT @JoeBiden: Character is on the ballot.
'@RealCandaceO @catturd2 @JoeBiden Candace
T @AmericaFirstPAC: .@JoeBiden's support of NAFTA caused Pennsylvania to lose 35.4 percent of its manufacturing jobs.
'When @JoeBiden wins this guy will soon be a defendant....
'@balajiworld @JoeBiden I feel so bad for these kids
'@JoyKill23 @JoeBiden Again
JustinBoling @JonesStickman @rev_b_wayne @JoeBiden Really? Weird no other 1st world nation has adopted it. In fact… https://t.co/EMMx80Afgc
ove ya @JoeBiden
JoeBiden he's not wrong you should vote early if you can. if not you prob be waiting in the long lines 6 feet apar… https://t.co/50Nn9fQGOk
'RT @JoeBiden: In 8 days
BardsFM @RealCandaceO @PrisonPlanet @JoeBiden Ready. Bracing. Please keep us posted!
'@laurenboebert @JoeBiden Never heard of you! Don’t expect to hear anymore about/from you after November 3
jacobkschneider @JoeBiden He “literally” was out in public all weekend you twat.
T @dudeibadog: @JoeBiden care to comment? #BidenHarris2020
ime taken: 0.191 seconds, Fetched: 20 row(s)
```

3. Tweets that are on Donal Trump



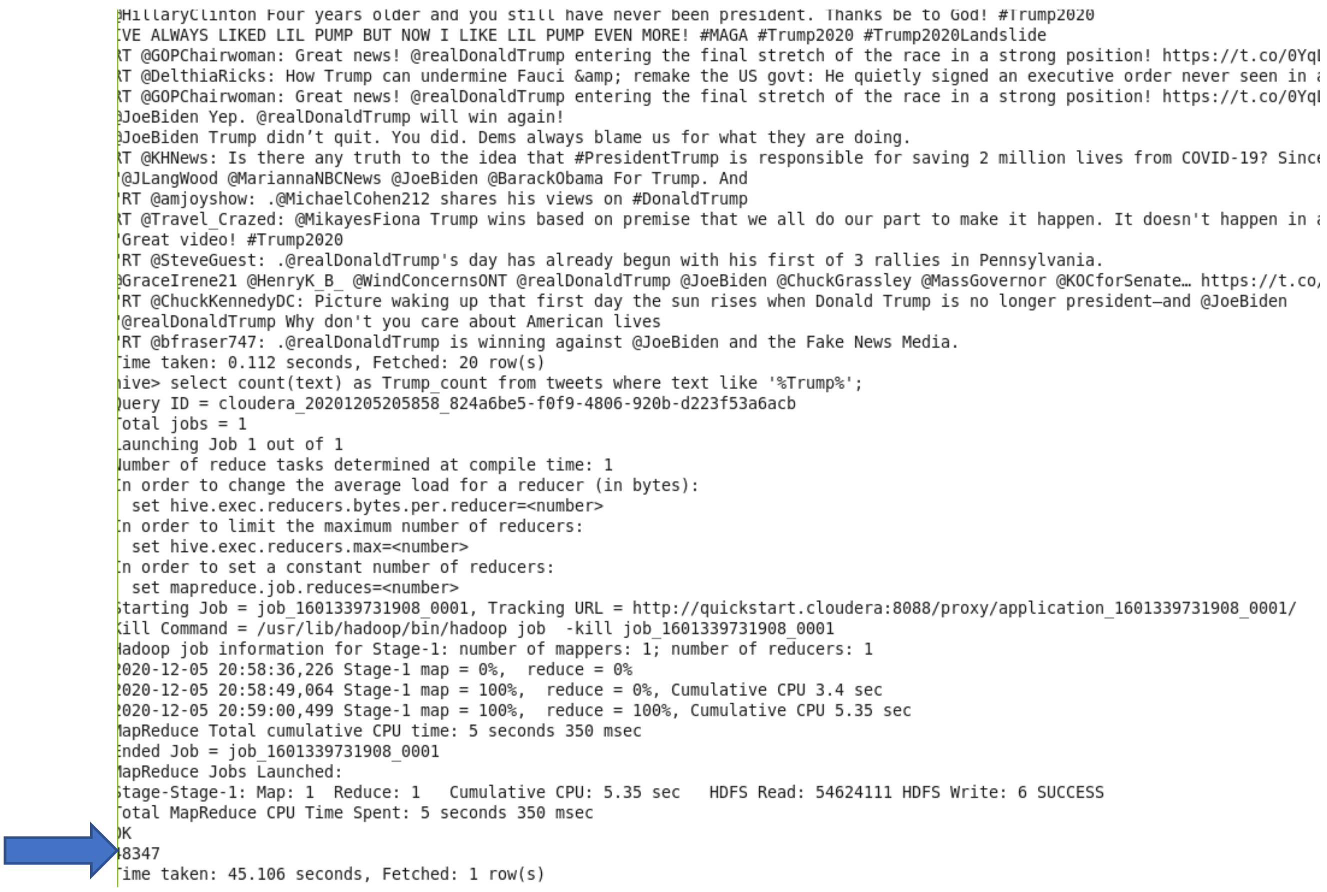
```
ive> select text from tweets where text like '%Trump%' limit 20;
K
'RT @KamalaHarris: The Trump administration’s theory is that the economy is doing well when the stock market is doing well.
appy birthday Killary. Thank you for President Donald Trump and 3 Supreme Court Justices #MAGA2020 #trump2020
'RT @KamalaHarris: The Trump administration’s theory is that the economy is doing well when the stock market is doing well.
HillaryClinton Four years older and you still have never been president. Thanks be to God! #Trump2020
VE ALWAYS LIKED LIL PUMP BUT NOW I LIKE LIL PUMP EVEN MORE! #MAGA #Trump2020 #Trump2020Landslide
T @GOPChairwoman: Great news! @realDonaldTrump entering the final stretch of the race in a strong position! https://t.co/0YqLsvsPa6
T @DelthiaRicks: How Trump can undermine Fauci &amp; remake the US govt: He quietly signed an executive order never seen in a democracy— he di…
T @GOPChairwoman: Great news! @realDonaldTrump entering the final stretch of the race in a strong position! https://t.co/0YqLsvsPa6
JoeBiden Yep. @realDonaldTrump will win again!
JoeBiden Trump didn’t quit. You did. Dems always blame us for what they are doing.
T @KHNews: Is there any truth to the idea that #PresidentTrump is responsible for saving 2 million lives from COVID-19? Since Trump contin…
'@JLangWood @MariannaNBCNews @JoeBiden @BarackObama For Trump. And
'RT @amjoyshow: .@MichaelCohen212 shares his views on #DonaldTrump
T @Travel_Crazed: @MikayesFiona Trump wins based on premise that we all do our part to make it happen. It doesn't happen in a vacuum. He c…
'Great video! #Trump2020
'RT @SteveGuest: .@realDonaldTrump's day has already begun with his first of 3 rallies in Pennsylvania.
GraceIrene21 @HenryK_B_ @WindConcernsONT @realDonaldTrump @JoeBiden @ChuckGrassley @MassGovernor @KOCforSenate… https://t.co/6m8bQeEfPe
'RT @ChuckKennedyDC: Picture waking up that first day the sun rises when Donald Trump is no longer president—and @JoeBiden
'@realDonaldTrump Why don't you care about American lives
'RT @bfraser747: .@realDonaldTrump is winning against @JoeBiden and the Fake News Media.
ime taken: 0.112 seconds, Fetched: 20 row(s)
```

4. Number of Tweets on Joe Biden



```
ime taken: 45.106 seconds, Fetched: 1 row(s)
ive> select count(text) as Biden_count from tweets where text like '%JoeBiden%';
uery ID = cloudera_20201205205959_421ab79f-c964-44aa-99bb-3427407830e7
otal jobs = 1
aunching Job 1 out of 1
umber of reduce tasks determined at compile time: 1
n order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
n order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
n order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
tarting Job = job_1601339731908_0002, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1601339731908_0002/
ill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1601339731908_0002
adoop job information for Stage-1: number of mappers: 1; number of reducers: 1
020-12-05 20:59:46,583 Stage-1 map = 0%,  reduce = 0%
020-12-05 20:59:58,912 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.37 sec
020-12-05 21:00:10,285 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.34 sec
apReduce Total cumulative CPU time: 5 seconds 340 msec
nded Job = job_1601339731908_0002
apReduce Jobs Launched:
tage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.34 sec   HDFS Read: 54624187 HDFS Write: 7 SUCCESS
otal MapReduce CPU Time Spent: 5 seconds 340 msec
K
09351
ime taken: 38.008 seconds, Fetched: 1 row(s)
ive> █
```

5. Number of Tweets on Donald Trump

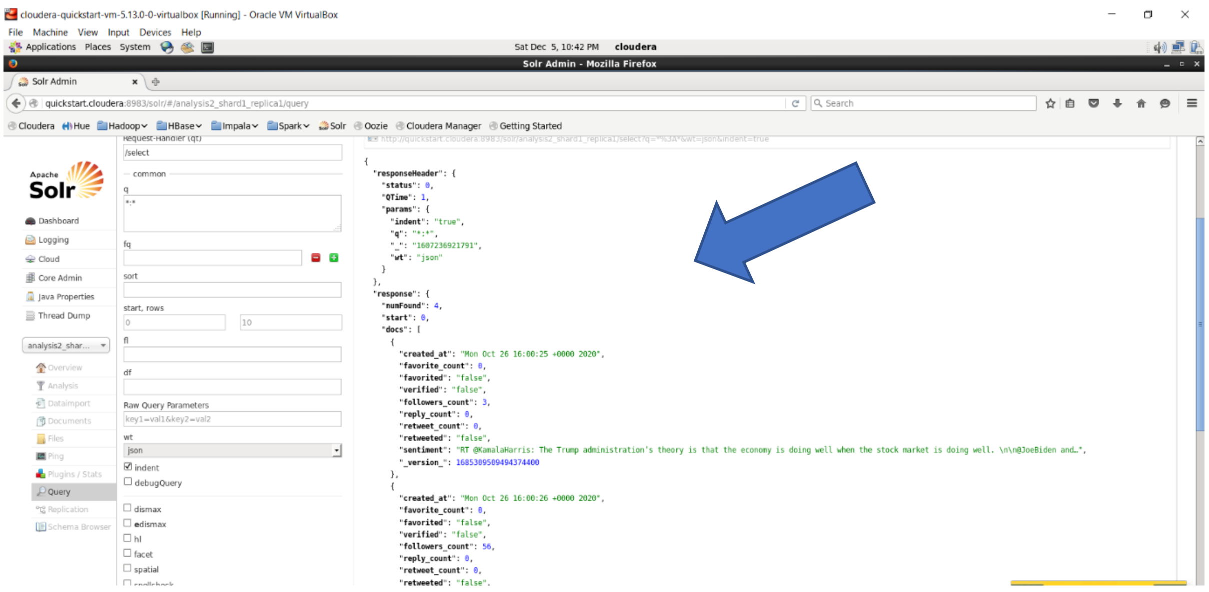


```
@HillaryClinton Four years older and you still have never been president. Thanks be to God! #Trump2020
IVE ALWAYS LIKED LIL PUMP BUT NOW I LIKE LIL PUMP EVEN MORE! #MAGA #Trump2020 #Trump2020Landslide
RT @GOPChairwoman: Great news! @realDonaldTrump entering the final stretch of the race in a strong position! https://t.co/0YqI
RT @DelthiaRicks: How Trump can undermine Fauci &amp; remake the US govt: He quietly signed an executive order never seen in a
RT @GOPChairwoman: Great news! @realDonaldTrump entering the final stretch of the race in a strong position! https://t.co/0YqI
@JoeBiden Yep. @realDonaldTrump will win again!
@JoeBiden Trump didn't quit. You did. Dems always blame us for what they are doing.
RT @KHNews: Is there any truth to the idea that #PresidentTrump is responsible for saving 2 million lives from COVID-19? Since
'@JLangWood @MariannaNBCNews @JoeBiden @BarackObama For Trump. And
'RT @amjoyshow: .@MichaelCohen212 shares his views on #DonaldTrump
RT @Travel_Crazed: @MikayesFiona Trump wins based on premise that we all do our part to make it happen. It doesn't happen in a
'Great video! #Trump2020
'RT @SteveGuest: .@realDonaldTrump's day has already begun with his first of 3 rallies in Pennsylvania.
@GraceIrene21 @HenryK_B_ @WindConcernsONT @realDonaldTrump @JoeBiden @ChuckGrassley @MassGovernor @KOCforSenate… https://t.co,
'RT @ChuckKennedyDC: Picture waking up that first day the sun rises when Donald Trump is no longer president—and @JoeBiden
'@realDonaldTrump Why don't you care about American lives
'RT @bfraser747: .@realDonaldTrump is winning against @JoeBiden and the Fake News Media.
Time taken: 0.112 seconds, Fetched: 20 row(s)
ive> select count(text) as Trump_count from tweets where text like '%Trump%';
Query ID = cloudera_20201205205858_824a6be5-f0f9-4806-920b-d223f53a6acb
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1601339731908_0001, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1601339731908_0001/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1601339731908_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2020-12-05 20:58:36,226 Stage-1 map = 0%,  reduce = 0%
2020-12-05 20:58:49,064 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.4 sec
2020-12-05 20:59:00,499 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.35 sec
MapReduce Total cumulative CPU time: 5 seconds 350 msec
Ended Job = job_1601339731908_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.35 sec   HDFS Read: 54624111 HDFS Write: 6 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 350 msec
OK
8347
Time taken: 45.106 seconds, Fetched: 1 row(s)
```

6.1.2. SOLAR:

Here we have tried to write queries that state whether the data is coming from people who have more followers so that those tweets have more reach. For this, we have created a solar instance with our schema by editing the schema.xml file for taking valuable data out of all collected data so that it could be fast in processing and efficient in terms of finding a vital solution.

1. Looking at all Tweets collected



cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox

Sat Dec 5, 10:42 PM cloudera

Solr Admin - Mozilla Firefox

quickstart.cloudera:8983/solr/#/analysis2_shard1_replica1/query

Request-Handler (qt): /select

q: *:*

start, rows: 0 10

Raw Query Parameters: key1=val1&key2=val2

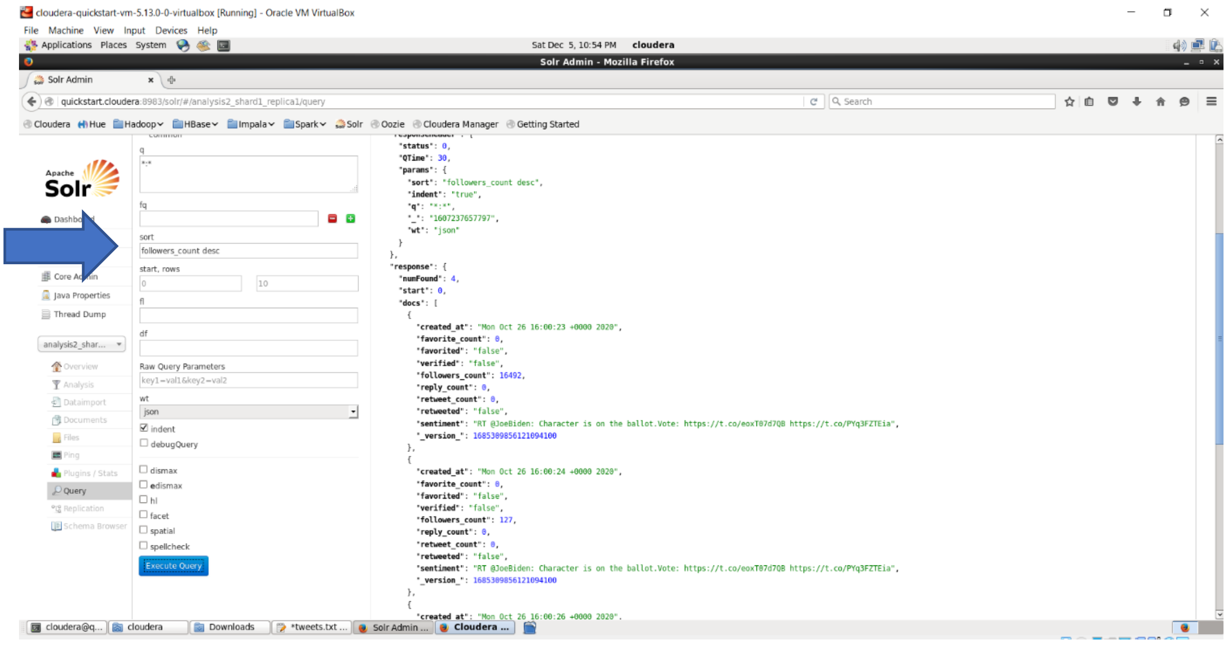
wt: json

☑ indent

http://quickstart.cloudera:8983/solr/analysis2_shard1_replica1/select?q=*%3A*&wt=json&indent=true

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 1,
    "params": {
      "indent": "true",
      "q": "*:*",
      "_": "1607236921791",
      "wt": "json"
    }
  },
  "response": {
    "numFound": 4,
    "start": 0,
    "docs": [
      {
        "created_at": "Mon Oct 26 16:00:25 +0000 2020",
        "favorite_count": 0,
        "favorited": "false",
        "verified": "false",
        "followers_count": 3,
        "reply_count": 0,
        "retweet_count": 0,
        "retweeted": "false",
        "sentiment": "RT @KamalaHarris: The Trump administration's theory is that the economy is doing well when the stock market is doing well. \n\n@JoeBiden and…",
        "_version_": 1685309509494374400
      },
      {
        "created_at": "Mon Oct 26 16:00:26 +0000 2020",
        "favorite_count": 0,
        "favorited": "false",
        "verified": "false",
        "followers_count": 56,
        "reply_count": 0,
        "retweet_count": 0,
        "retweeted": "false",
```

2. Arranging tweets according to followers



cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox

Solr Admin - Mozilla Firefox

quickstart.cloudera:8983/solr/#/analysis2_shard1_replica1/query

q: `*:*`

sort: followers_count desc

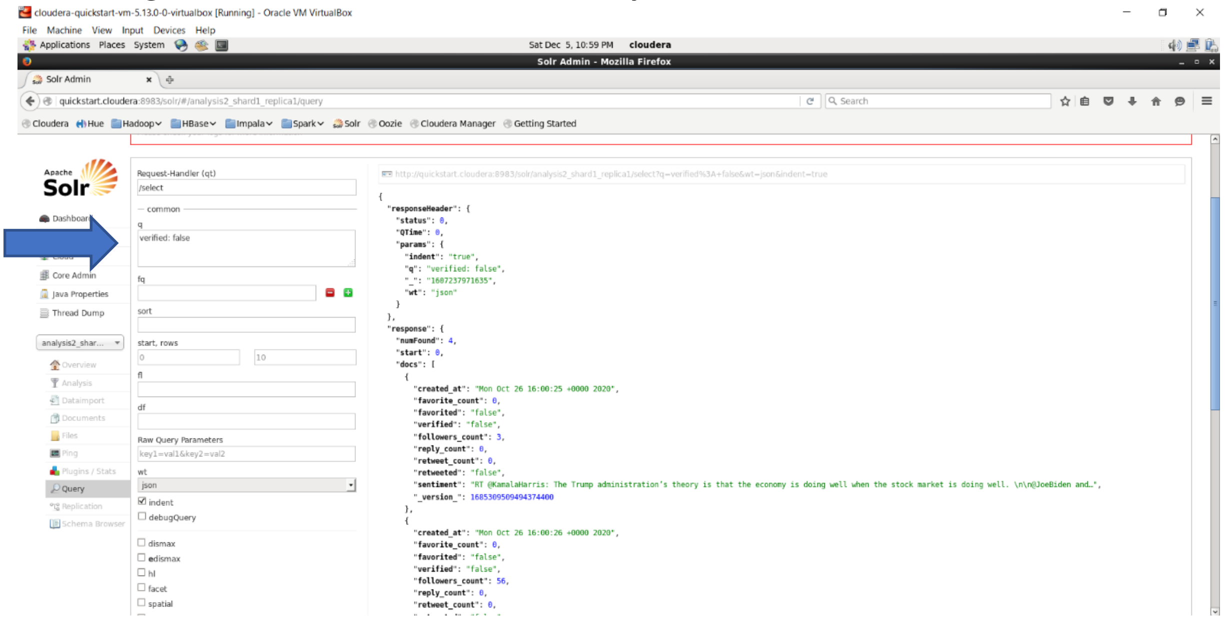
start, rows: 0 10

wt: json

☑ indent

```
  "status": 0,
  "QTime": 30,
  "params": {
    "sort": "followers_count desc",
    "indent": "true",
    "q": "*:*",
    "_": "1607237657797",
    "wt": "json"
  }
},
"response": {
  "numFound": 4,
  "start": 0,
  "docs": [
    {
      "created_at": "Mon Oct 26 16:00:23 +0000 2020",
      "favorite_count": 0,
      "favorited": "false",
      "verified": "false",
      "followers_count": 16492,
      "reply_count": 0,
      "retweet_count": 0,
      "retweeted": "false",
      "sentiment": "RT @JoeBiden: Character is on the ballot.Vote: https://t.co/eoxT07d7QB https://t.co/PYq3FZTEia",
      "_version_": 1685309856121094100
    },
    {
      "created_at": "Mon Oct 26 16:00:24 +0000 2020",
      "favorite_count": 0,
      "favorited": "false",
      "verified": "false",
      "followers_count": 127,
      "reply_count": 0,
      "retweet_count": 0,
      "retweeted": "false",
      "sentiment": "RT @JoeBiden: Character is on the ballot.Vote: https://t.co/eoxT07d7QB https://t.co/PYq3FZTEia",
      "_version_": 1685309856121094100
    },
    {
      "created_at": "Mon Oct 26 16:00:26 +0000 2020",
```

3. Looking for tweets which has been tweeted by non-verified users



cloudera-quickstart-vm-5.13.0-0-virtualbox [Running] - Oracle VM VirtualBox

Solr Admin - Mozilla Firefox

quickstart.cloudera:8983/solr/#/analysis2_shard1_replica1/query

Request-Handler (qt): /select

q: verified: false

start, rows: 0 10

wt: json

☑ indent

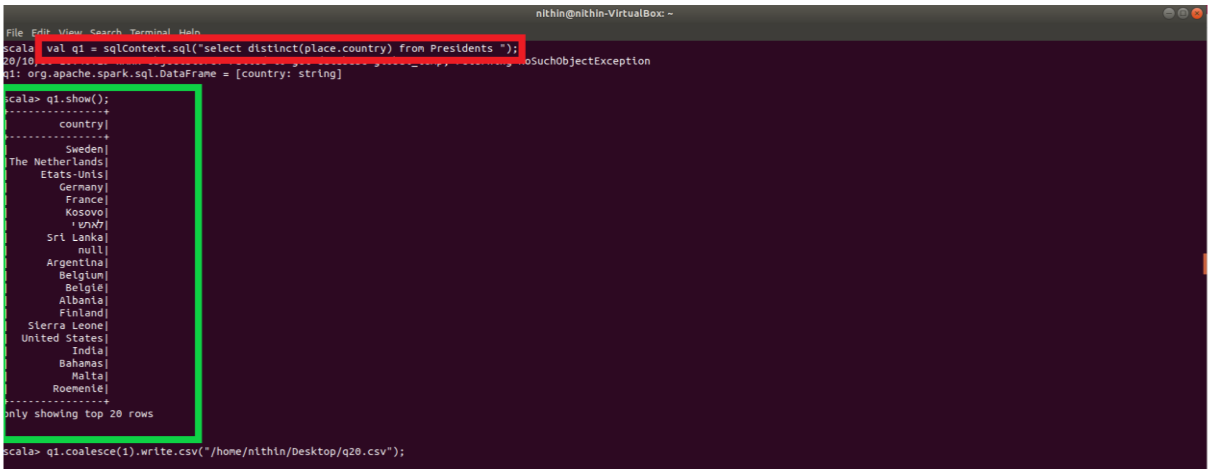
http://quickstart.cloudera:8983/solr/analysis2_shard1_replica1/select?q=verified%3A+false&wt=json&indent=true

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 0,
    "params": {
      "indent": "true",
      "q": "verified: false",
      "_": "1607237971635",
      "wt": "json"
    }
  },
  "response": {
    "numFound": 4,
    "start": 0,
    "docs": [
      {
        "created_at": "Mon Oct 26 16:00:25 +0000 2020",
        "favorite_count": 0,
        "favorited": "false",
        "verified": "false",
        "followers_count": 3,
        "reply_count": 0,
        "retweet_count": 0,
        "retweeted": "false",
        "sentiment": "RT @KamalaHarris: The Trump administration's theory is that the economy is doing well when the stock market is doing well. \n\n@JoeBiden and…",
        "_version_": 1685309509494374400
      },
      {
        "created_at": "Mon Oct 26 16:00:26 +0000 2020",
        "favorite_count": 0,
        "favorited": "false",
        "verified": "false",
        "followers_count": 56,
        "reply_count": 0,
        "retweet_count": 0,
```

6.1.3. SPARK SQL:

To be more sure we have tried writing queries on the JSON data that we collected to check the trend line for the competitors and tried to verify whether we have any anonymous data that have no use. The main point of writing queries is to check whether there is any biased data or not and to check whether we got the right tweets from the right people by verifying their follower's count and friends.

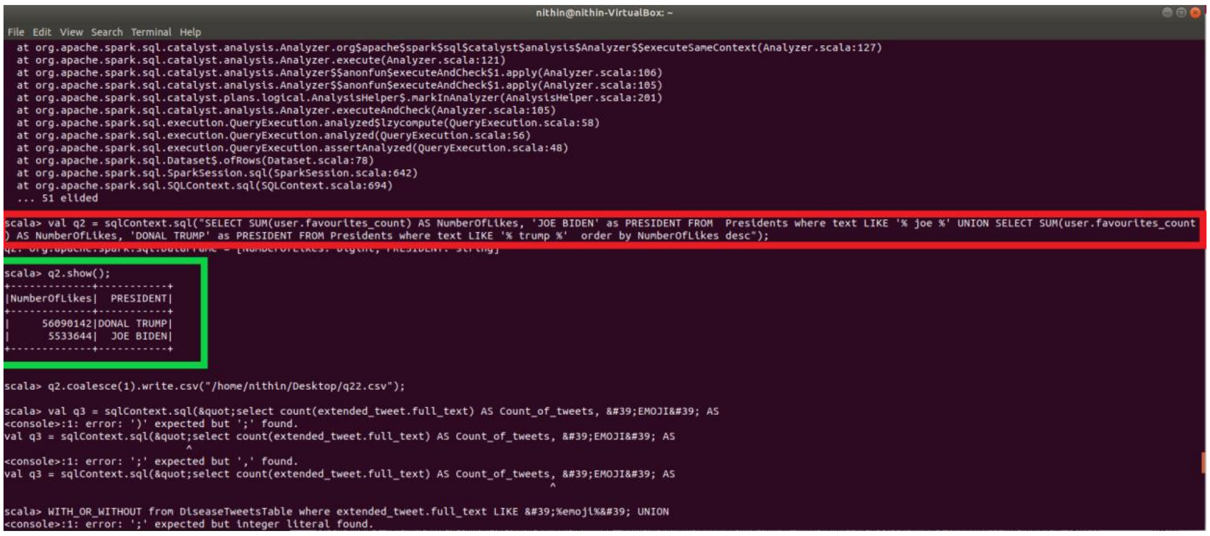
1. EXTRACTING USER LOCATION WHO TWEETED



```
scala> val q1 = sqlContext.sql("select distinct(place.country) from Presidents ");
20/10/[illegible] NoSuchObjectException
q1: org.apache.spark.sql.DataFrame = [country: string]

scala> q1.show();
+---------------+
|        country|
+---------------+
|         Sweden|
|The Netherlands|
|     Etats-Unis|
|        Germany|
|         France|
|         Kosovo|
|         לארשי|
|      Sri Lanka|
|           null|
|      Argentina|
|        Belgium|
|         België|
|        Albania|
|        Finland|
|   Sierra Leone|
|  United States|
|          India|
|        Bahamas|
|          Malta|
|       Roemenië|
+---------------+
only showing top 20 rows

scala> q1.coalesce(1).write.csv("/home/nithin/Desktop/q20.csv");
```

This is information will be useful to know where this topic is going trending.

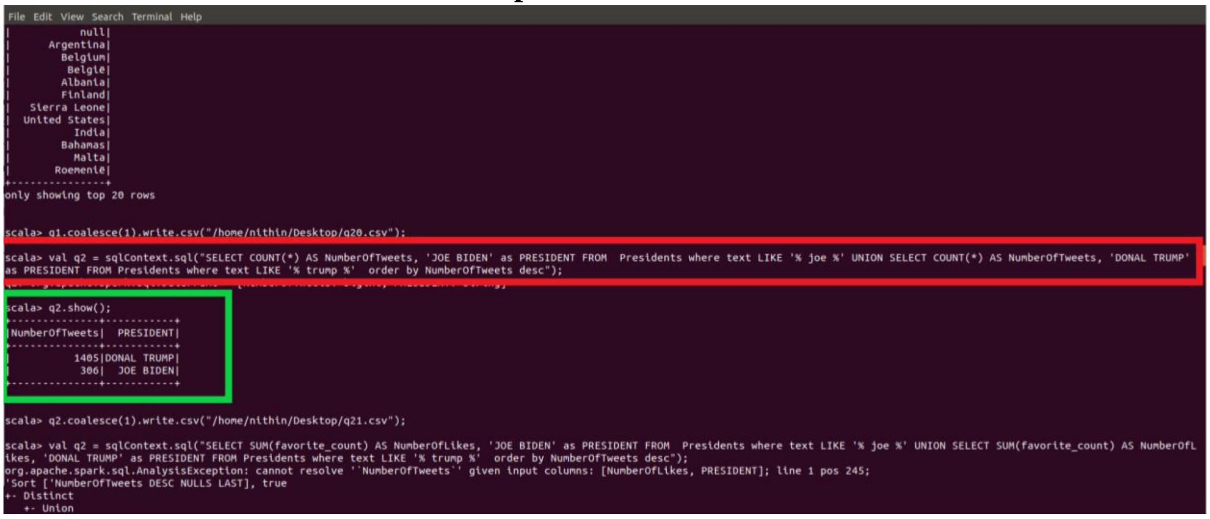
2. NUMBER OF TWEETS FOR PARTICULAR CANDIDATE



```
  at org.apache.spark.sql.catalyst.analysis.Analyzer.org$apache$spark$sql$catalyst$analysis$Analyzer$$executeSameContext(Analyzer.scala:127)
  at org.apache.spark.sql.catalyst.analysis.Analyzer.execute(Analyzer.scala:121)
  at org.apache.spark.sql.catalyst.analysis.Analyzer$$anonfun$executeAndCheck$1.apply(Analyzer.scala:106)
  at org.apache.spark.sql.catalyst.analysis.Analyzer$$anonfun$executeAndCheck$1.apply(Analyzer.scala:105)
  at org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper$.markInAnalyzer(AnalysisHelper.scala:201)
  at org.apache.spark.sql.catalyst.analysis.Analyzer.executeAndCheck(Analyzer.scala:105)
  at org.apache.spark.sql.execution.QueryExecution.analyzed$lzycompute(QueryExecution.scala:58)
  at org.apache.spark.sql.execution.QueryExecution.analyzed(QueryExecution.scala:56)
  at org.apache.spark.sql.execution.QueryExecution.assertAnalyzed(QueryExecution.scala:48)
  at org.apache.spark.sql.Dataset$.ofRows(Dataset.scala:78)
  at org.apache.spark.sql.SparkSession.sql(SparkSession.scala:642)
  at org.apache.spark.sql.SQLContext.sql(SQLContext.scala:694)
  ... 51 elided

scala> val q2 = sqlContext.sql("SELECT SUM(user.favourites_count) AS NumberOfLikes, 'JOE BIDEN' as PRESIDENT FROM  Presidents where text LIKE '% joe %' UNION SELECT SUM(user.favourites_count
) AS NumberOfLikes, 'DONAL TRUMP' as PRESIDENT FROM Presidents where text LIKE '% trump %'  order by NumberOfLikes desc");
q2: org.apache.spark.sql.DataFrame = [NumberOfLikes: bigint, PRESIDENT: string]

scala> q2.show();
+-------------+-----------+
|NumberOfLikes|  PRESIDENT|
+-------------+-----------+
|     56090142|DONAL TRUMP|
|      5533644|  JOE BIDEN|
+-------------+-----------+

scala> q2.coalesce(1).write.csv("/home/nithin/Desktop/q22.csv");

scala> val q3 = sqlContext.sql(&quot;select count(extended_tweet.full_text) AS Count_of_tweets, &#39;EMOJI&#39; AS
<console>:1: error: ')' expected but ';' found.
val q3 = sqlContext.sql(&quot;select count(extended_tweet.full_text) AS Count_of_tweets, &#39;EMOJI&#39; AS
                              ^
<console>:1: error: ';' expected but ',' found.
val q3 = sqlContext.sql(&quot;select count(extended_tweet.full_text) AS Count_of_tweets, &#39;EMOJI&#39; AS
                                                                    ^

scala> WITH_OR_WITHOUT from DiseaseTweetsTable where extended_tweet.full_text LIKE &#39;%emoji%&#39; UNION
<console>:1: error: ';' expected but integer literal found.
```

Through this query we can have a picture who is more trending on twitter, on further analysis we can say it whether it is positive or negative.

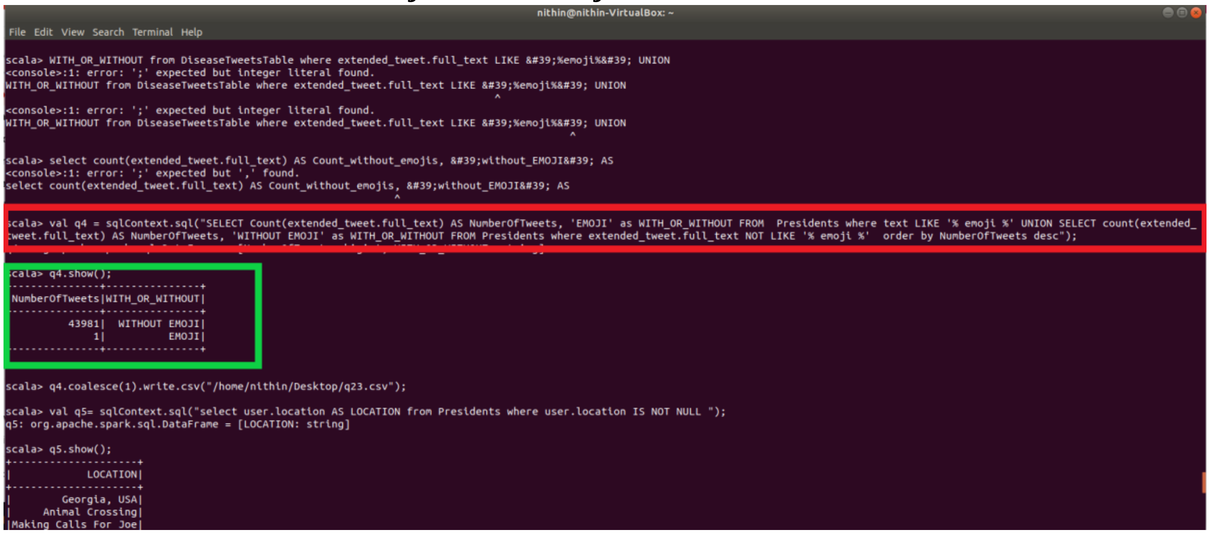
3. Number of Likes for Tweets about particular candidate.



```
|           null|
|      Argentina|
|        Belgium|
|         België|
|        Albania|
|        Finland|
|   Sierra Leone|
|  United States|
|          India|
|        Bahamas|
|          Malta|
|       Roemenië|
+---------------+
only showing top 20 rows


scala> q1.coalesce(1).write.csv("/home/nithin/Desktop/q20.csv");

scala> val q2 = sqlContext.sql("SELECT COUNT(*) AS NumberOfTweets, 'JOE BIDEN' as PRESIDENT FROM  Presidents where text LIKE '% joe %' UNION SELECT COUNT(*) AS NumberOfTweets, 'DONAL TRUMP'
as PRESIDENT FROM Presidents where text LIKE '% trump %'  order by NumberOfTweets desc");

scala> q2.show();
+--------------+-----------+
|NumberOfTweets|  PRESIDENT|
+--------------+-----------+
|          1405|DONAL TRUMP|
|           306|  JOE BIDEN|
+--------------+-----------+

scala> q2.coalesce(1).write.csv("/home/nithin/Desktop/q21.csv");

scala> val q2 = sqlContext.sql("SELECT SUM(favorite_count) AS NumberOfLikes, 'JOE BIDEN' as PRESIDENT FROM  Presidents where text LIKE '% joe %' UNION SELECT SUM(favorite_count) AS NumberOfL
ikes, 'DONAL TRUMP' as PRESIDENT FROM Presidents where text LIKE '% trump %'  order by NumberOfTweets desc");
org.apache.spark.sql.AnalysisException: cannot resolve '`NumberOfTweets`' given input columns: [NumberOfLikes, PRESIDENT]; line 1 pos 245;
'Sort ['NumberOfTweets DESC NULLS LAST], true
+- Distinct
   +- Union
```

After analyzing the data whether we have to know the support of people for that tweet, this query data can be vital in knowing that.

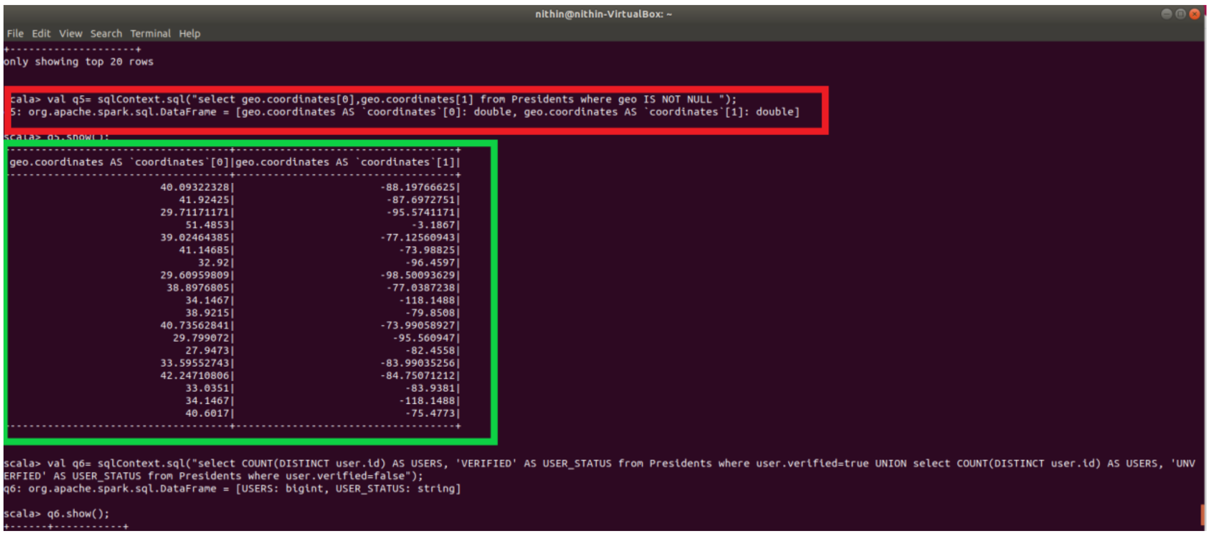
4. Tweets which contains emojis vs nonemoji tweets



```
nithin@nithin-VirtualBox: ~
File  Edit  View  Search  Terminal  Help

scala> WITH_OR_WITHOUT from DiseaseTweetsTable where extended_tweet.full_text LIKE &#39;%emoji%&#39; UNION
<console>:1: error: ';' expected but integer literal found.
WITH_OR_WITHOUT from DiseaseTweetsTable where extended_tweet.full_text LIKE &#39;%emoji%&#39; UNION
                                                                                ^
<console>:1: error: ';' expected but integer literal found.
WITH_OR_WITHOUT from DiseaseTweetsTable where extended_tweet.full_text LIKE &#39;%emoji%&#39; UNION
                                                                                            ^

scala> select count(extended_tweet.full_text) AS Count_without_emojis, &#39;without_EMOJI&#39; AS
<console>:1: error: ';' expected but ',' found.
select count(extended_tweet.full_text) AS Count_without_emojis, &#39;without_EMOJI&#39; AS
                                                              ^

scala> val q4 = sqlContext.sql("SELECT Count(extended_tweet.full_text) AS NumberOfTweets, 'EMOJI' as WITH_OR_WITHOUT FROM  Presidents where text LIKE '% emoji %' UNION SELECT count(extended_
tweet.full_text) AS NumberOfTweets, 'WITHOUT EMOJI' as WITH_OR_WITHOUT FROM Presidents where extended_tweet.full_text NOT LIKE '% emoji %'  order by NumberOfTweets desc");

scala> q4.show();
--------------+--------------+
NumberOfTweets|WITH_OR_WITHOUT|
--------------+--------------+
         43981|  WITHOUT EMOJI|
             1|          EMOJI|
--------------+--------------+

scala> q4.coalesce(1).write.csv("/home/nithin/Desktop/q23.csv");

scala> val q5= sqlContext.sql("select user.location AS LOCATION from Presidents where user.location IS NOT NULL ");
q5: org.apache.spark.sql.DataFrame = [LOCATION: string]

scala> q5.show();
+-------------------+
|           LOCATION|
+-------------------+
|       Georgia, USA|
|    Animal Crossing|
|Making Calls For Joe|
```

This query can be useful while processing the data(during sentimental analysis).

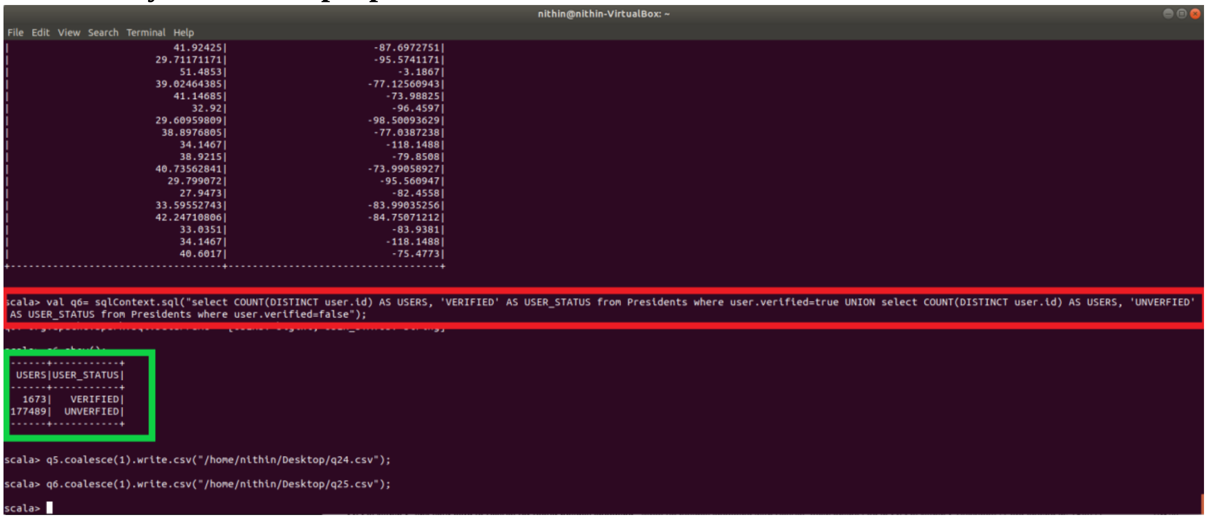
5. Geo location of user



```
nithin@nithin-VirtualBox: ~
File  Edit  View  Search  Terminal  Help
+-------------------+
only showing top 20 rows

cala> val q5= sqlContext.sql("select geo.coordinates[0],geo.coordinates[1] from Presidents where geo IS NOT NULL ");
5: org.apache.spark.sql.DataFrame = [geo.coordinates AS `coordinates`[0]: double, geo.coordinates AS `coordinates`[1]: double]

--------------------------------+--------------------------------+
geo.coordinates AS `coordinates`[0]|geo.coordinates AS `coordinates`[1]|
--------------------------------+--------------------------------+
                     40.09322328|                    -88.19766625|
                        41.92425|                     -87.6972751|
                     29.71171171|                     -95.5741171|
                         51.4853|                         -3.1867|
                     39.02464385|                    -77.12560943|
                        41.14685|                       -73.98825|
                           32.92|                        -96.4597|
                     29.60959809|                    -98.50093629|
                      38.8976805|                     -77.0387238|
                         34.1467|                       -118.1488|
                         38.9215|                         -79.8508|
                     40.73562841|                    -73.99058927|
                       29.799072|                      -95.560947|
                         27.9473|                        -82.4558|
                     33.59552743|                    -83.99035256|
                     42.24710806|                    -84.75071212|
                         33.0351|                        -83.9381|
                         34.1467|                       -118.1488|
                         40.6017|                        -75.4773|
--------------------------------+--------------------------------+

scala> val q6= sqlContext.sql("select COUNT(DISTINCT user.id) AS USERS, 'VERIFIED' AS USER_STATUS from Presidents where user.verified=true UNION select COUNT(DISTINCT user.id) AS USERS, 'UNV
ERFIED' AS USER_STATUS from Presidents where user.verified=false");
q6: org.apache.spark.sql.DataFrame = [USERS: bigint, USER_STATUS: string]

scala> q6.show();
+------+-----------+
```

This can be helpful in knowing the hot spot about the discussion going about candidates.

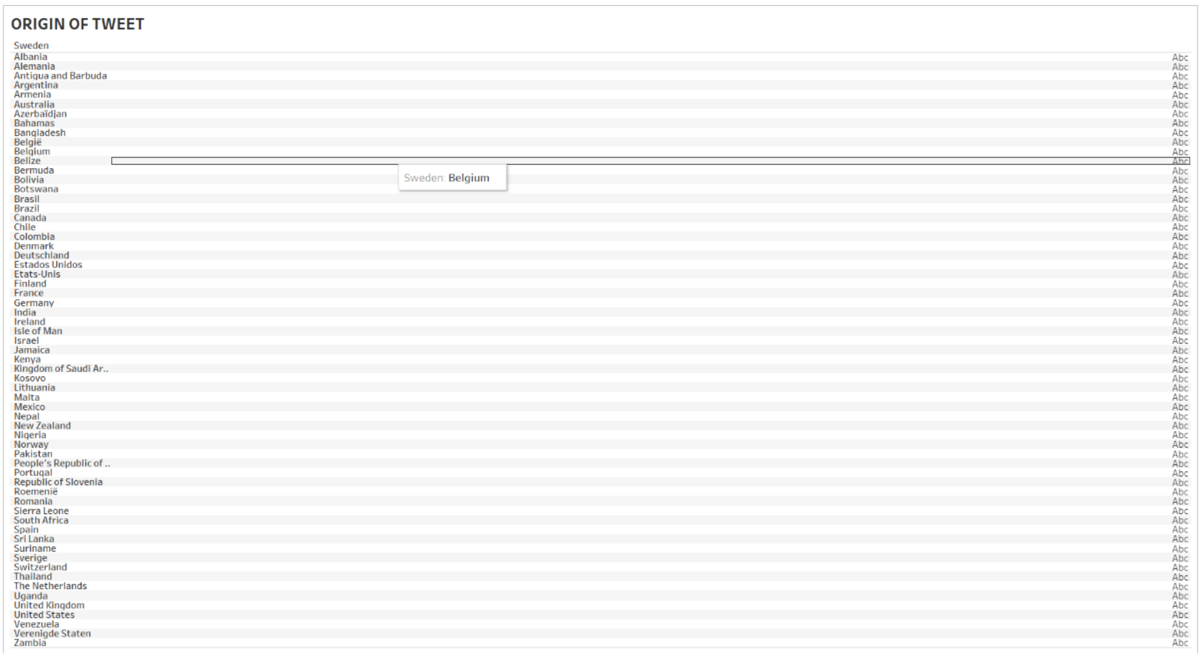
6. Celebrity vs Normal people.



```
nithin@nithin-VirtualBox: ~
File  Edit  View  Search  Terminal  Help
|                       41.92425|                     -87.6972751|
|                    29.71171171|                     -95.5741171|
|                        51.4853|                         -3.1867|
|                    39.02464385|                    -77.12560943|
|                       41.14685|                       -73.98825|
|                          32.92|                        -96.4597|
|                    29.60959809|                    -98.50093629|
|                     38.8976805|                     -77.0387238|
|                        34.1467|                       -118.1488|
|                        38.9215|                         -79.8508|
|                    40.73562841|                    -73.99058927|
|                      29.799072|                      -95.560947|
|                        27.9473|                        -82.4558|
|                    33.59552743|                    -83.99035256|
|                    42.24710806|                    -84.75071212|
|                        33.0351|                        -83.9381|
|                        34.1467|                       -118.1488|
|                        40.6017|                        -75.4773|
+-------------------------------+--------------------------------+

scala> val q6= sqlContext.sql("select COUNT(DISTINCT user.id) AS USERS, 'VERIFIED' AS USER_STATUS from Presidents where user.verified=true UNION select COUNT(DISTINCT user.id) AS USERS, 'UNVERFIED'
 AS USER_STATUS from Presidents where user.verified=false");

------+-----------+
 USERS|USER_STATUS|
------+-----------+
  1673|   VERIFIED|
177489|  UNVERFIED|
------+-----------+

scala> q5.coalesce(1).write.csv("/home/nithin/Desktop/q24.csv");

scala> q6.coalesce(1).write.csv("/home/nithin/Desktop/q25.csv");

scala> 
```

In general case we can trust more on tweets that is been given by celebrity. In order to give weightage to a tweet we can use this information.

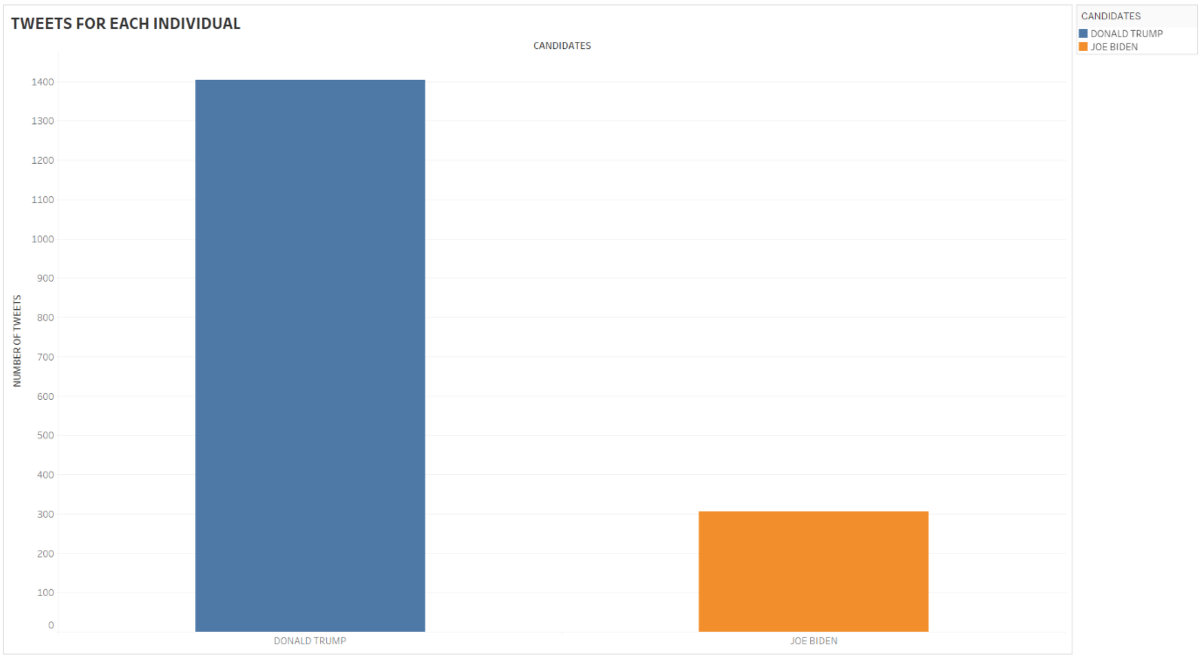
7. Visualization Report:

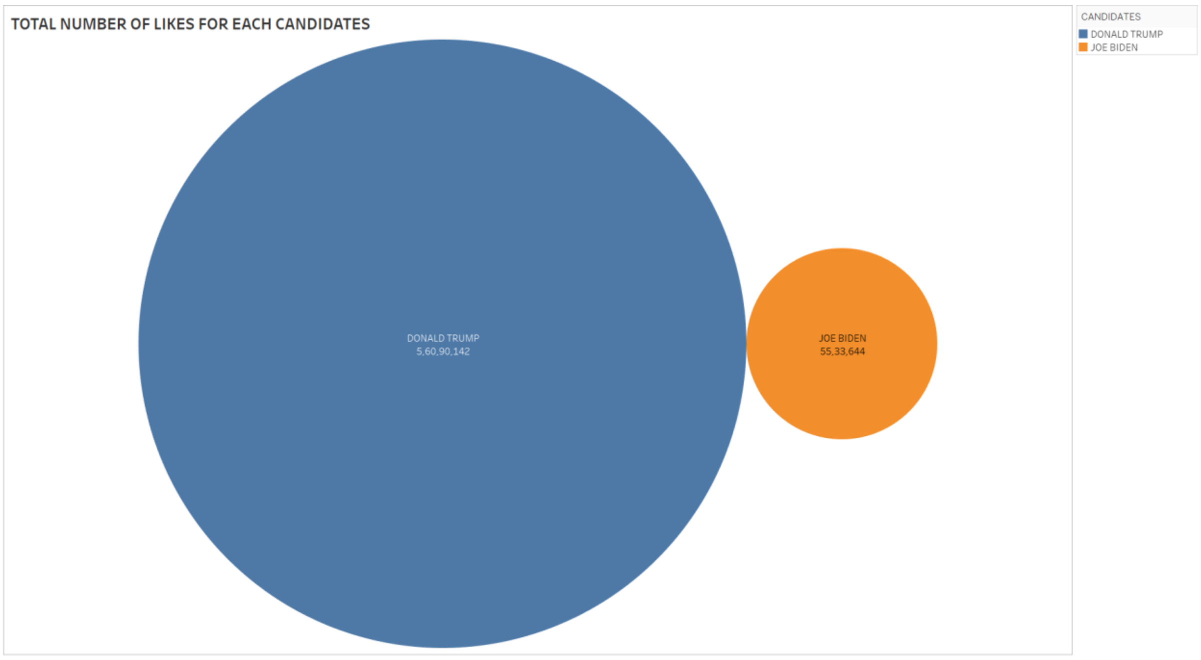


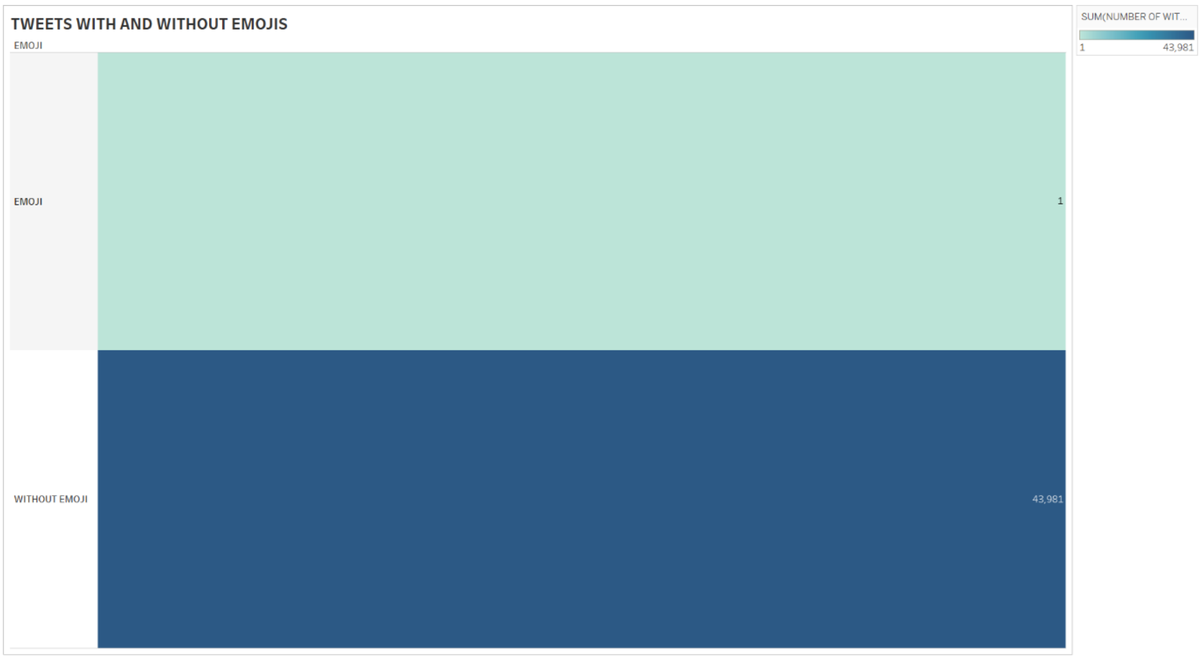
ORIGIN OF TWEET

Sweden	
Albania	Abc
Alemania	Abc
Antigua and Barbuda	Abc
Argentina	Abc
Armenia	Abc
Australia	Abc
Azerbaidjan	Abc
Bahamas	Abc
Bangladesh	Abc
België	Abc
Belgium	Abc
Belize	Abc
Bermuda	Abc
Bolivia	Abc
Botswana	Abc
Brasil	Abc
Brazil	Abc
Canada	Abc
Chile	Abc
Colombia	Abc
Denmark	Abc
Deutschland	Abc
Estados Unidos	Abc
Etats-Unis	Abc
Finland	Abc
France	Abc
Germany	Abc
India	Abc
Ireland	Abc
Isle of Man	Abc
Israel	Abc
Jamaica	Abc
Kenya	Abc
Kingdom of Saudi Ar..	Abc
Kosovo	Abc
Lithuania	Abc
Malta	Abc
Mexico	Abc
Nepal	Abc
New Zealand	Abc
Nigeria	Abc
Norway	Abc
Pakistan	Abc
People's Republic of ..	Abc
Portugal	Abc
Republic of Slovenia	Abc
Roemenië	Abc
Romania	Abc
Sierra Leone	Abc
South Africa	Abc
Spain	Abc
Sri Lanka	Abc
Suriname	Abc
Sverige	Abc
Switzerland	Abc
Thailand	Abc
The Netherlands	Abc
Uganda	Abc
United Kingdom	Abc
United States	Abc
Venezuela	Abc
Verenigde Staten	Abc
Zambia	Abc

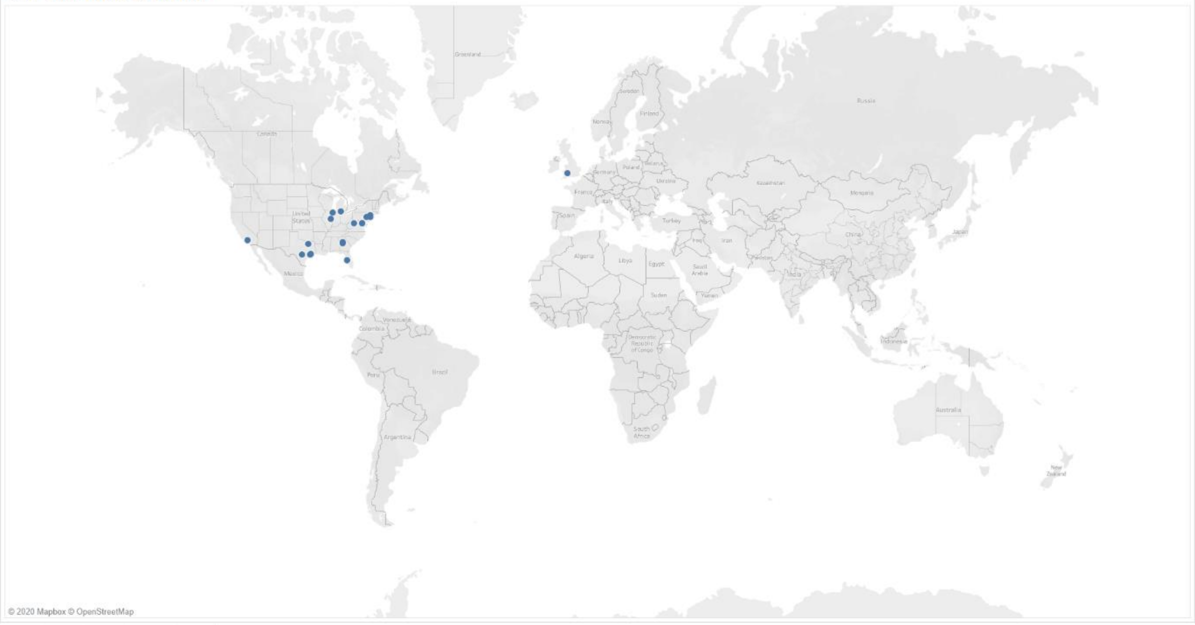
Sweden: Belgium



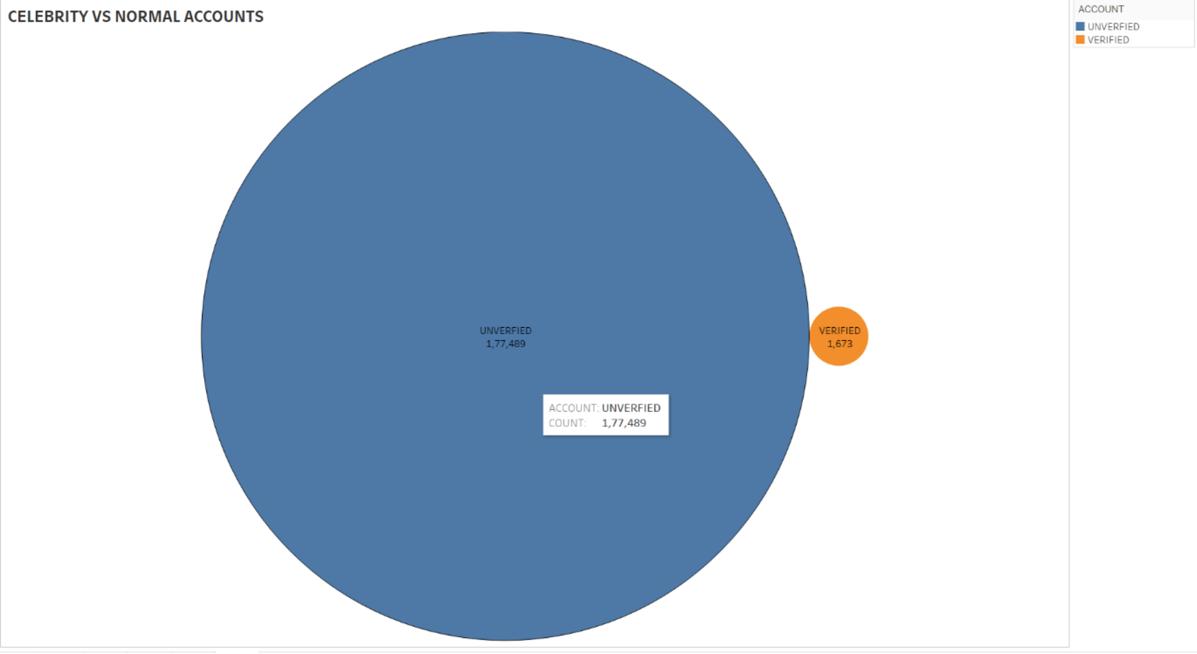

TOTAL NUMBER OF LIKES FOR EACH CANDIDATES
DONALD TRUMP
5,60,90,142
JOE BIDEN
55,33,644
CANDIDATES
DONALD TRUMP
JOE BIDEN


TWEETS WITH AND WITHOUT EMOJIS
EMOJI
EMOJI
1
WITHOUT EMOJI
43,981
SUM(NUMBER OF WIT...
1
43,981

GEO-GRAPHICAL LOCATION

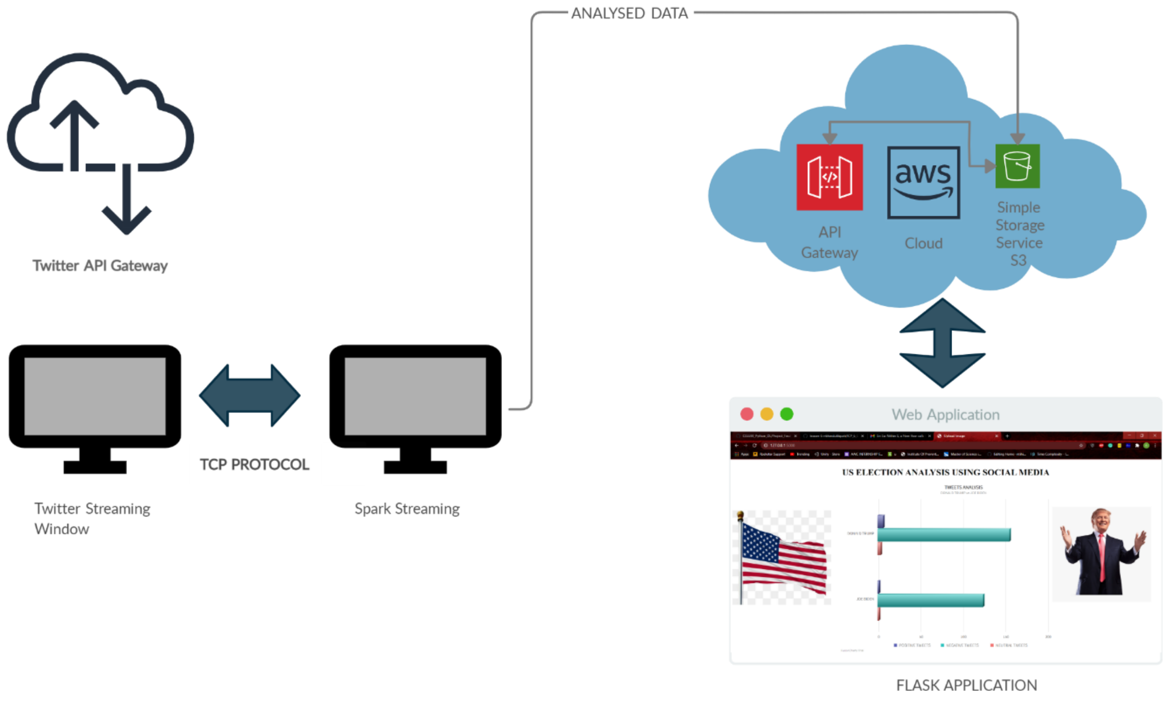


CELEBRITY VS NORMAL ACCOUNTS

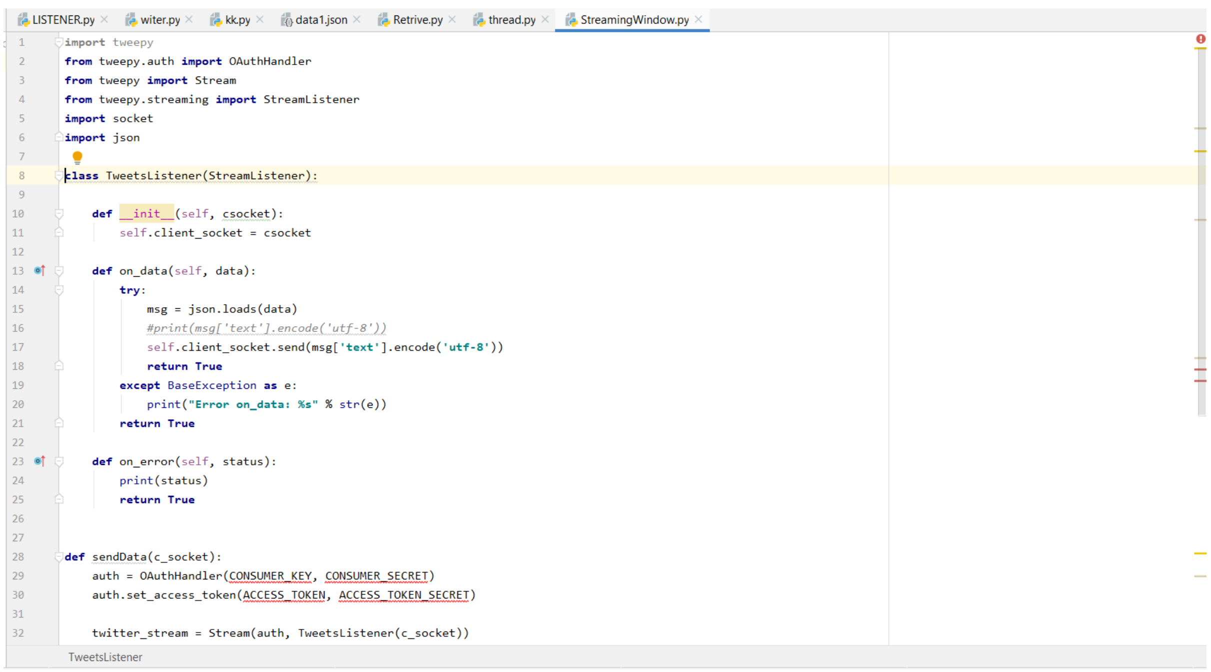


7. IMPLEMENTAION:

PROJECT WORKFLOW:

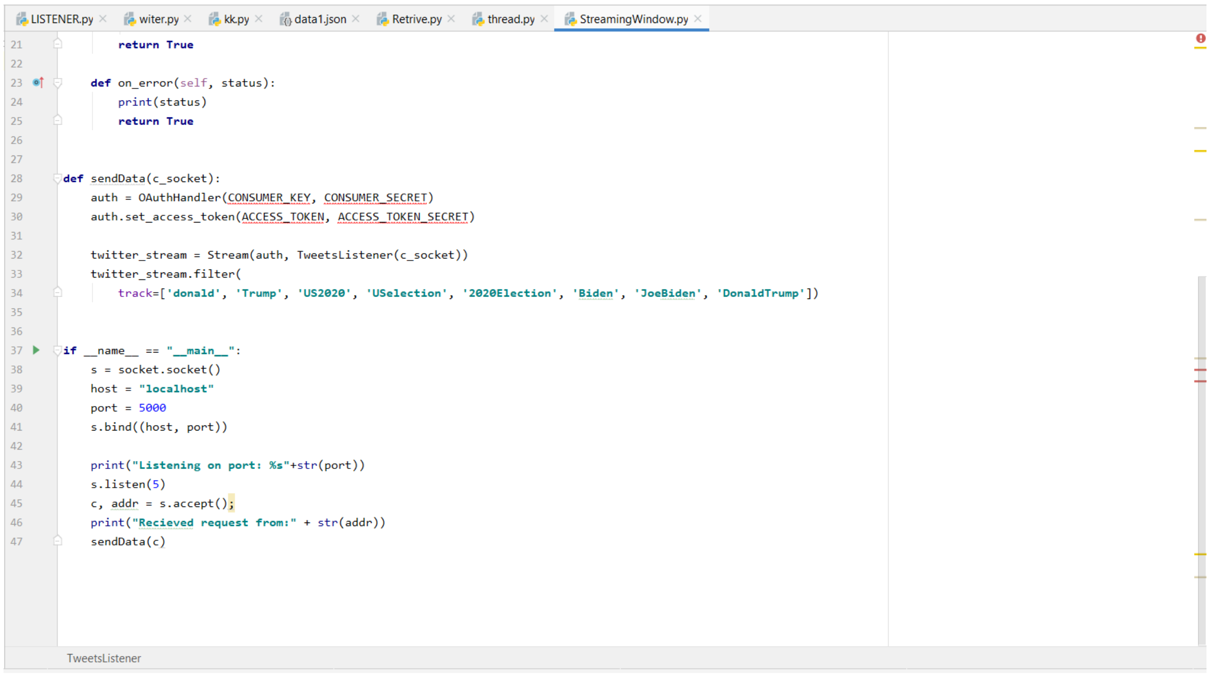


7.1 STREAMING TWITTER DATA:



```python
import tweepy
from tweepy.auth import OAuthHandler
from tweepy import Stream
from tweepy.streaming import StreamListener
import socket
import json

class TweetsListener(StreamListener):

    def __init__(self, csocket):
        self.client_socket = csocket

    def on_data(self, data):
        try:
            msg = json.loads(data)
            #print(msg['text'].encode('utf-8'))
            self.client_socket.send(msg['text'].encode('utf-8'))
            return True
        except BaseException as e:
            print("Error on_data: %s" % str(e))
        return True

    def on_error(self, status):
        print(status)
        return True


def sendData(c_socket):
    auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
    auth.set_access_token(ACCESS_TOKEN, ACCESS_TOKEN_SECRET)

    twitter_stream = Stream(auth, TweetsListener(c_socket))
```

TweetsListener



```python
        return True

    def on_error(self, status):
        print(status)
        return True


def sendData(c_socket):
    auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
    auth.set_access_token(ACCESS_TOKEN, ACCESS_TOKEN_SECRET)

    twitter_stream = Stream(auth, TweetsListener(c_socket))
    twitter_stream.filter(
        track=['donald', 'Trump', 'US2020', 'USelection', '2020Election', 'Biden', 'JoeBiden', 'DonaldTrump'])


if __name__ == "__main__":
    s = socket.socket()
    host = "localhost"
    port = 5000
    s.bind((host, port))

    print("Listening on port: %s"+str(port))
    s.listen(5)
    c, addr = s.accept();
    print("Recieved request from:" + str(addr))
    sendData(c)
```

In here we have tried to retrieve the data for twitter using tweepy library in python, after that we have streamed the data to a port using sockets.

7.2. SPARK STREAMING

In here we are trying to listen to the port where we are trying to stream the data from the previous program. In this program we are listening the port window and performing map reduce operation over the data.

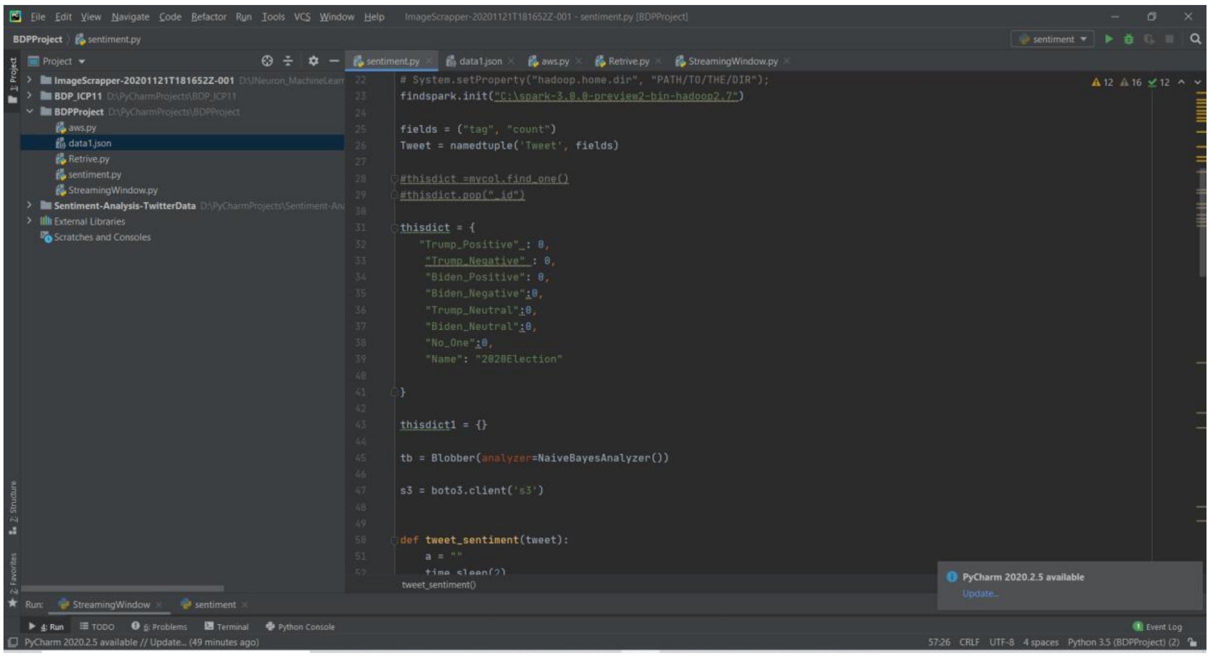
1. Flat map operation, here we are trying to perform sentiment analysis on the tweet that we have received.

2. In next operation we are performing Map operation where are we are trying to map each result.

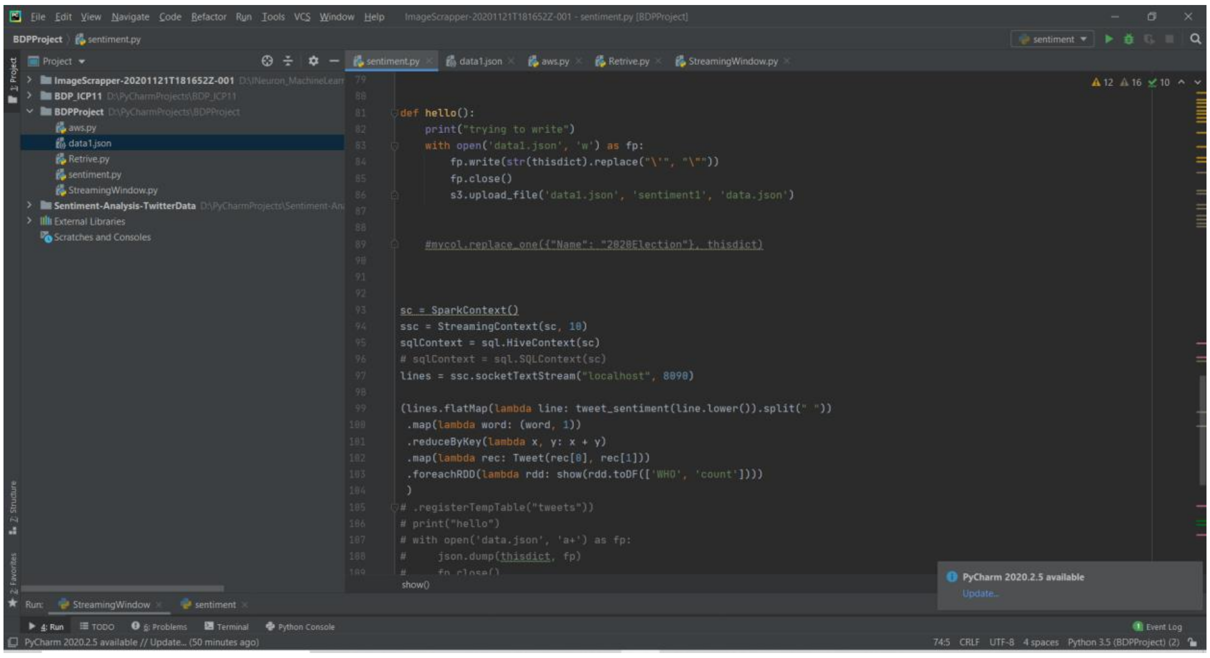
3. In the 3rd step we are performing reduce operation where we are reducing each output of map operation by reducebykey operation where we are aggregating each common result from map phase.

4. In the next step we are trying to perform map operation and trying to map key with their aggregates result.

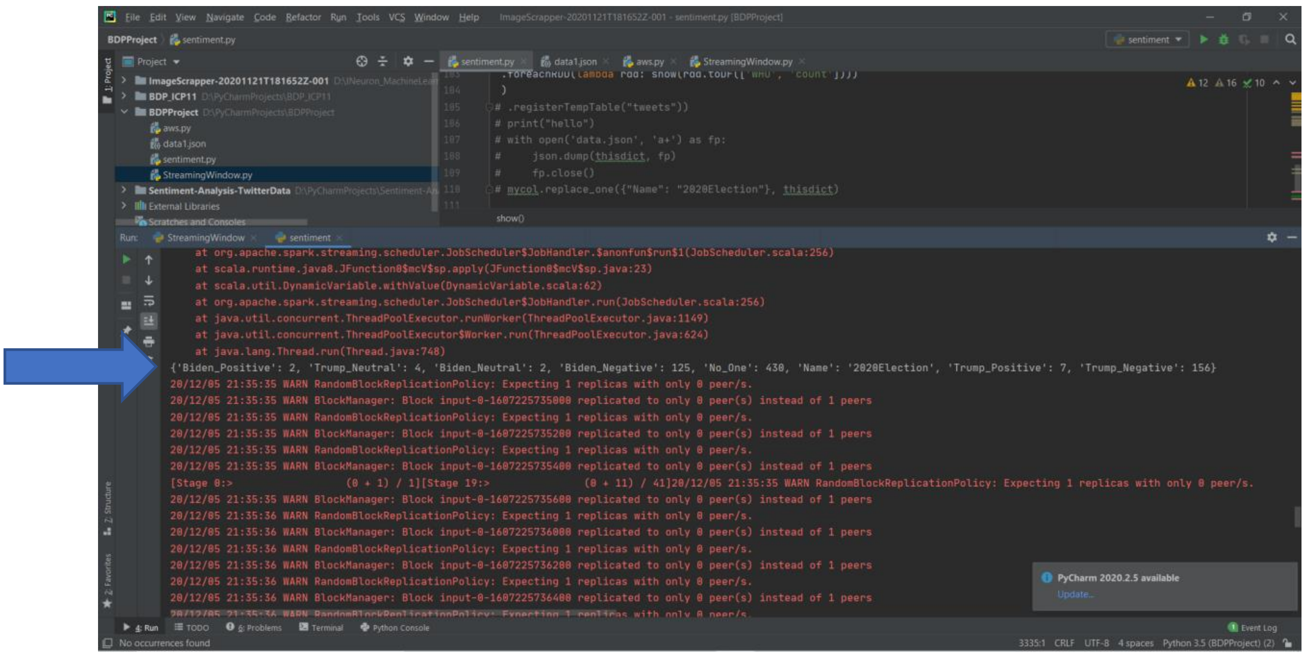
5. In the last step we are trying to store results in each stream into a temporary RDD, and finally we are pushing the data to AWS s3 bucket.



```python
# System.setProperty("hadoop.home.dir", "PATH/TO/THE/DIR");
findspark.init("C:\spark-3.0.0-preview2-bin-hadoop2.7")

fields = ("tag", "count")
Tweet = namedtuple('Tweet', fields)

#thisdict =mycol.find_one()
#thisdict.pop("_id")

thisdict = {
   "Trump_Positive"_: 0,
    "Trump_Negative" : 0,
    "Biden_Positive": 0,
    "Biden_Negative":0,
    "Trump_Neutral":0,
    "Biden_Neutral":0,
    "No_One":0,
    "Name": "2020Election"

}

thisdict1 = {}

tb = Blobber(analyzer=NaiveBayesAnalyzer())

s3 = boto3.client('s3')


def tweet_sentiment(tweet):
    a = ""
    time.sleep(2)
```



```python


def hello():
    print("trying to write")
    with open('data1.json', 'w') as fp:
        fp.write(str(thisdict).replace("\'", "\""))
        fp.close()
        s3.upload_file('data1.json', 'sentiment1', 'data.json')


    #mycol.replace_one({"Name": "2020Election"}, thisdict)



sc = SparkContext()
ssc = StreamingContext(sc, 10)
sqlContext = sql.HiveContext(sc)
# sqlContext = sql.SQLContext(sc)
lines = ssc.socketTextStream("localhost", 8090)

(lines.flatMap(lambda line: tweet_sentiment(line.lower()).split(" "))
 .map(lambda word: (word, 1))
 .reduceByKey(lambda x, y: x + y)
 .map(lambda rec: Tweet(rec[0], rec[1]))
 .foreachRDD(lambda rdd: show(rdd.toDF(['WHO', 'count'])))
 )
# .registerTempTable("tweets"))
# print("hello")
# with open('data.json', 'a+') as fp:
#     json.dump(thisdict, fp)
#     fp.close()
```



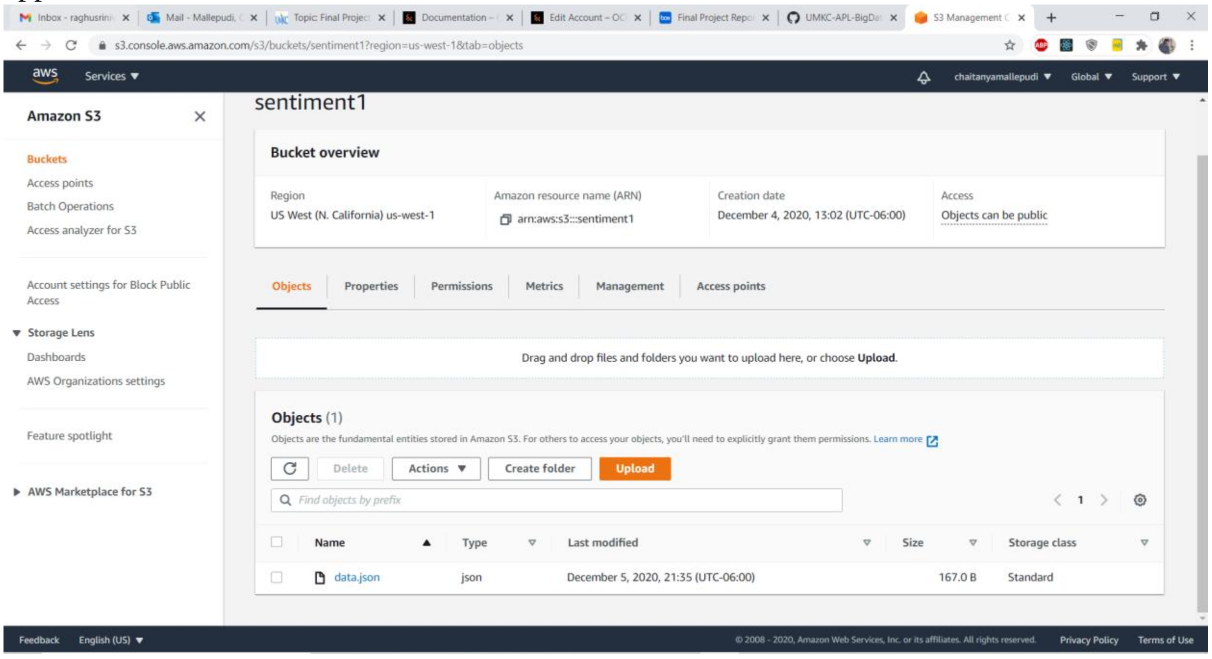
```
    .foreachRDD(lambda rdd: show(rdd.toDF(['WHO', 'count'])))
    )
# .registerTempTable("tweets"))
# print("hello")
# with open('data.json', 'a+') as fp:
#     json.dump(thisdict, fp)
#     fp.close()
# mycol.replace_one({"Name": "2020Election"}, thisdict)
```

```
	at org.apache.spark.streaming.scheduler.JobScheduler$JobHandler.$anonfun$run$1(JobScheduler.scala:256)
	at scala.runtime.java8.JFunction0$mcV$sp.apply(JFunction0$mcV$sp.java:23)
	at scala.util.DynamicVariable.withValue(DynamicVariable.scala:62)
	at org.apache.spark.streaming.scheduler.JobScheduler$JobHandler.run(JobScheduler.scala:256)
	at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
	at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
	at java.lang.Thread.run(Thread.java:748)
{'Biden_Positive': 2, 'Trump_Neutral': 4, 'Biden_Neutral': 2, 'Biden_Negative': 125, 'No_One': 430, 'Name': '2020Election', 'Trump_Positive': 7, 'Trump_Negative': 156}
20/12/05 21:35:35 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
20/12/05 21:35:35 WARN BlockManager: Block input-0-1607225735000 replicated to only 0 peer(s) instead of 1 peers
20/12/05 21:35:35 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
20/12/05 21:35:35 WARN BlockManager: Block input-0-1607225735200 replicated to only 0 peer(s) instead of 1 peers
20/12/05 21:35:35 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
20/12/05 21:35:35 WARN BlockManager: Block input-0-1607225735400 replicated to only 0 peer(s) instead of 1 peers
[Stage 0:>                  (0 + 1) / 1][Stage 19:>                (0 + 11) / 41]20/12/05 21:35:35 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
20/12/05 21:35:35 WARN BlockManager: Block input-0-1607225735600 replicated to only 0 peer(s) instead of 1 peers
20/12/05 21:35:36 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
20/12/05 21:35:36 WARN BlockManager: Block input-0-1607225736000 replicated to only 0 peer(s) instead of 1 peers
20/12/05 21:35:36 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
20/12/05 21:35:36 WARN BlockManager: Block input-0-1607225736200 replicated to only 0 peer(s) instead of 1 peers
20/12/05 21:35:36 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
20/12/05 21:35:36 WARN BlockManager: Block input-0-1607225736400 replicated to only 0 peer(s) instead of 1 peers
20/12/05 21:35:36 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
```

PyCharm 2020.2.5 available
Update...

7.3 API GATEWAY:

In this step we are trying to put a trigger at AWS by writing a code in lambda which triggers the change in s3 bucket and make itself ready to pass the data to the requesting application.



s3.console.aws.amazon.com/s3/buckets/sentiment1?region=us-west-1&tab=objects

Amazon S3

sentiment1

Bucket overview

Region	Amazon resource name (ARN)	Creation date	Access
US West (N. California) us-west-1	arn:aws:s3:::sentiment1	December 4, 2020, 13:02 (UTC-06:00)	Objects can be public

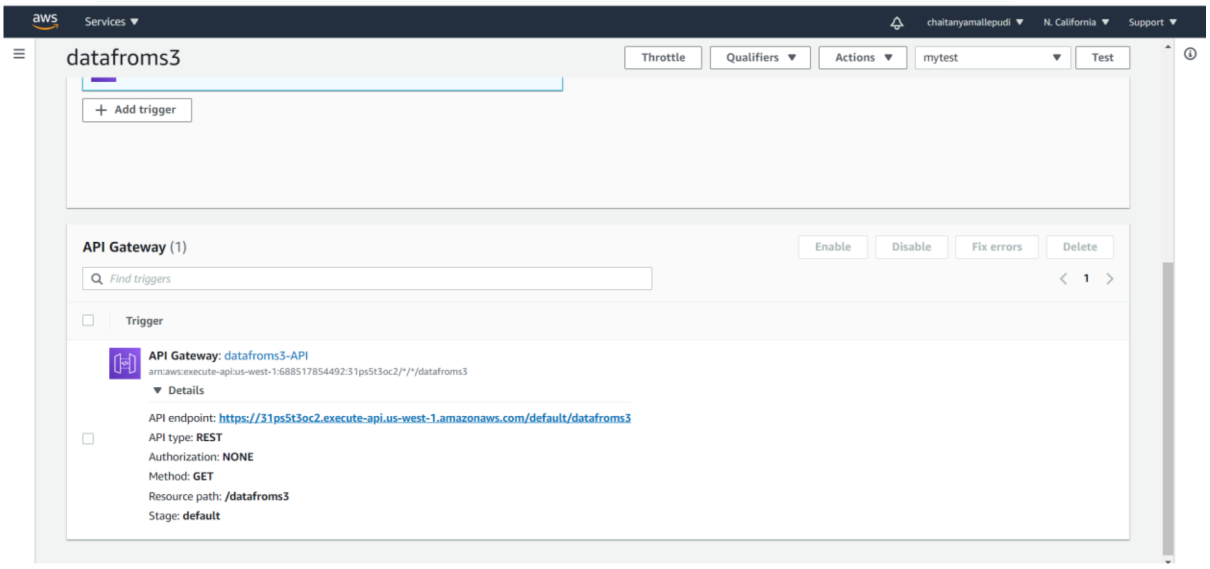
Drag and drop files and folders you want to upload here, or choose **Upload**.

Objects (1)

Objects are the fundamental entities stored in Amazon S3. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
data.json	json	December 5, 2020, 21:35 (UTC-06:00)	167.0 B	Standard

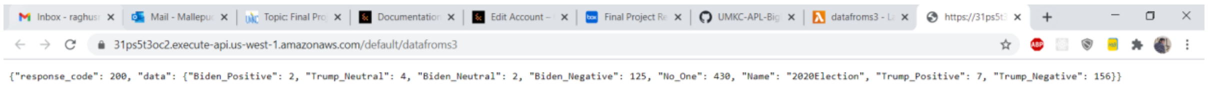
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



API Gateway (1)

API Gateway: dataforms3-API
arn:aws:execute-api:us-west-1:688517854492:31ps5t3oc2/*/*/datafroms3
Details
API endpoint: https://31ps5t3oc2.execute-api.us-west-1.amazonaws.com/default/datafroms3
API type: REST
Authorization: NONE
Method: GET
Resource path: /datafroms3
Stage: default

Requesting API Gateway:



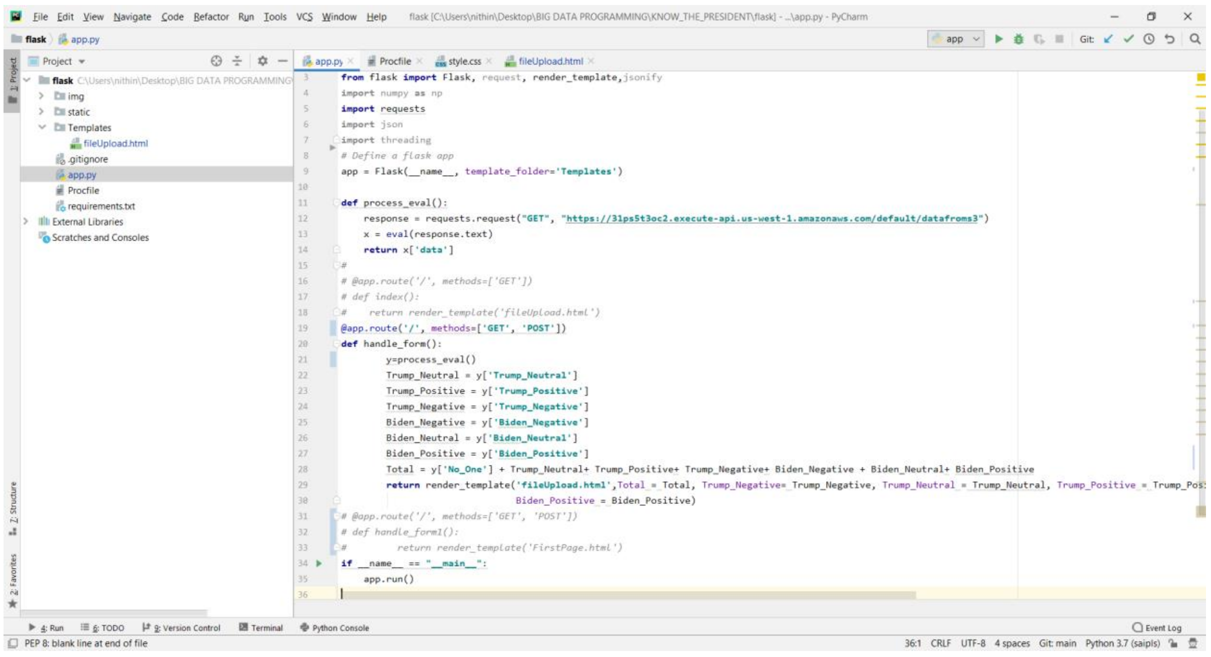
31ps5t3oc2.execute-api.us-west-1.amazonaws.com/default/datafroms3

```
{"response_code": 200, "data": {"Biden_Positive": 2, "Trump_Neutral": 4, "Biden_Neutral": 2, "Biden_Negative": 125, "No_One": 430, "Name": "2020Election", "Trump_Positive": 7, "Trump_Negative": 156}}
```

7.4. WEB APPLICATION:

We have created a web application that gets the data from the API gateway using the URL generated in the above step. We have used fusion charts API to display the result as a bar chart on the web page.

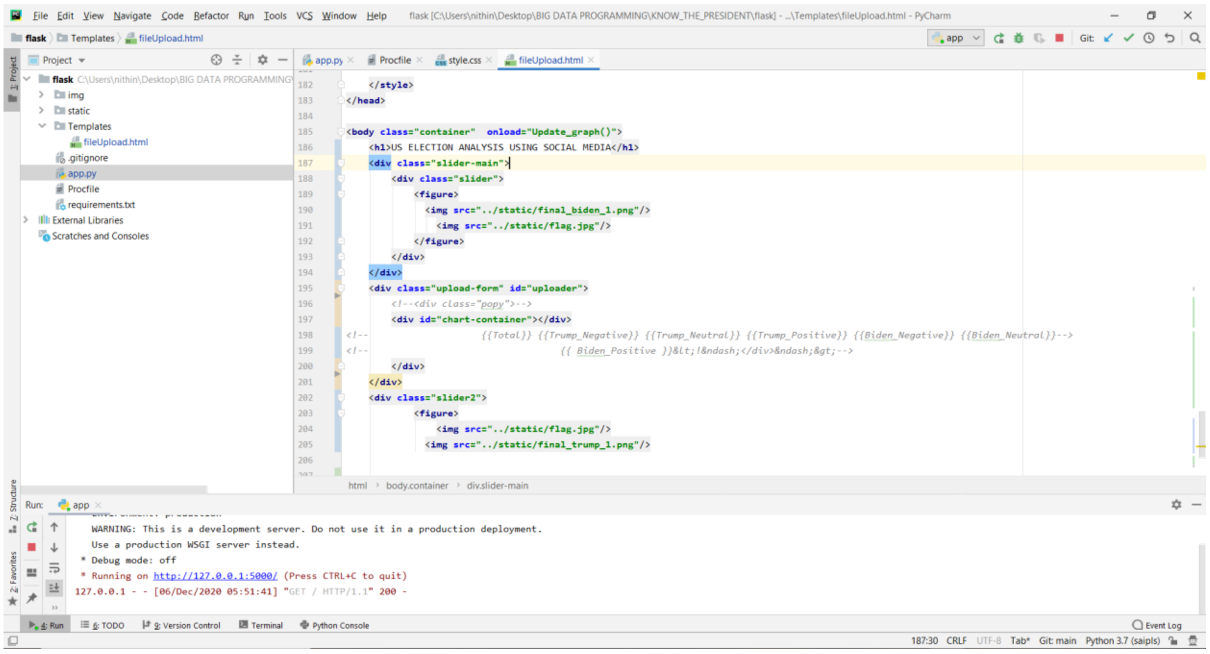
1. BACKEND CODE



flask [C:\Users\nithin\Desktop\BIG DATA PROGRAMMING\KNOW_THE_PRESIDENT\flask] - ...\app.py - PyCharm

```
from flask import Flask, request, render_template,jsonify
import numpy as np
import requests
import json
import threading
# Define a flask app
app = Flask(__name__, template_folder='Templates')

def process_eval():
    response = requests.request("GET", "https://31ps5t3oc2.execute-api.us-west-1.amazonaws.com/default/datafroms3")
    x = eval(response.text)
    return x['data']
#
# @app.route('/', methods=['GET'])
# def index():
#    return render_template('fileUpload.html')
@app.route('/', methods=['GET', 'POST'])
def handle_form():
        y=process_eval()
        Trump_Neutral = y['Trump_Neutral']
        Trump_Positive = y['Trump_Positive']
        Trump_Negative = y['Trump_Negative']
        Biden_Negative = y['Biden_Negative']
        Biden_Neutral = y['Biden_Neutral']
        Biden_Positive = y['Biden_Positive']
        Total = y['No_One'] + Trump_Neutral+ Trump_Positive+ Trump_Negative+ Biden_Negative + Biden_Neutral+ Biden_Positive
        return render_template('fileUpload.html',Total = Total, Trump_Negative= Trump_Negative, Trump_Neutral = Trump_Neutral, Trump_Positive = Trump_Pos
                                Biden_Positive = Biden_Positive)
# @app.route('/', methods=['GET', 'POST'])
# def handle_form1():
#        return render_template('FirstPage.html')
if __name__ == "__main__":
    app.run()
```

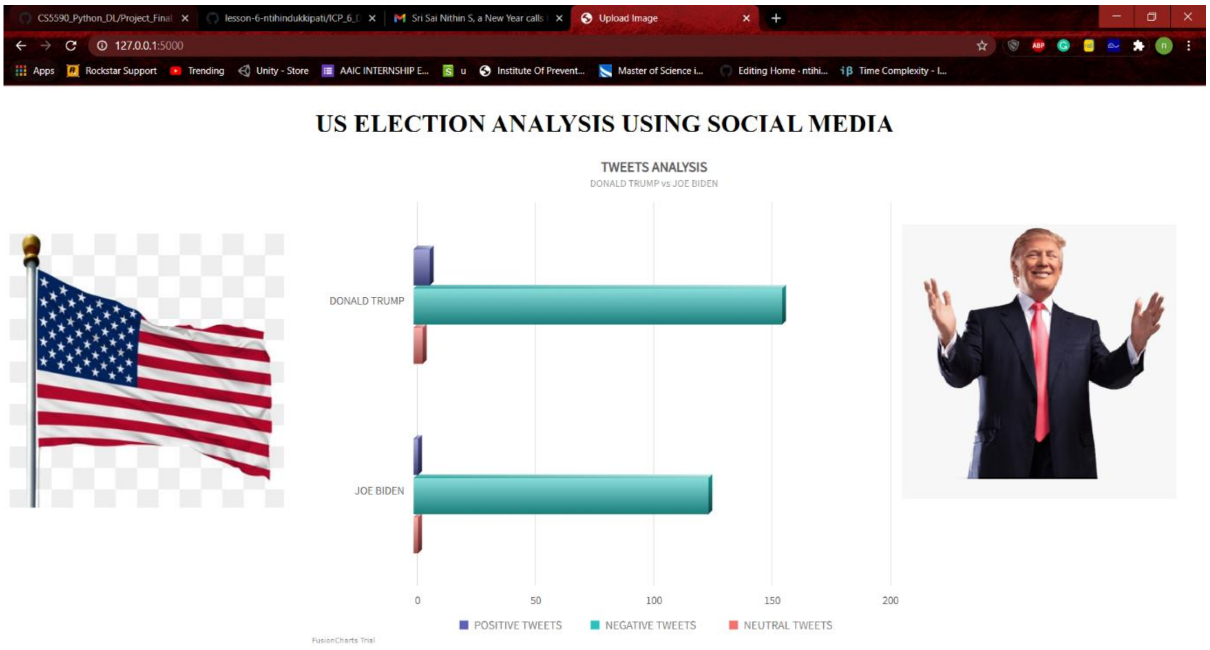
2. FRONT END CODE



```
    </style>
</head>

<body class="container"  onload="Update_graph()">
    <h1>US ELECTION ANALYSIS USING SOCIAL MEDIA</h1>
    <div class="slider-main">
        <div class="slider">
            <figure>
              <img src="../static/final_biden_1.png"/>
                <img src="../static/flag.jpg"/>
            </figure>
        </div>
    </div>
    <div class="upload-form" id="uploader">
        <!--<div class="popy">-->
        <div id="chart-container"></div>
<!--                    {{Total}} {{Trump_Negative}} {{Trump_Neutral}} {{Trump_Positive}} {{Biden_Negative}} {{Biden_Neutral}}-->
<!--                                 {{ Biden_Positive }}&lt;!&ndash;</div>&ndash;&gt;-->
        </div>
    </div>
    <div class="slider2">
            <figure>
                <img src="../static/flag.jpg"/>
              <img src="../static/final_trump_1.png"/>
```

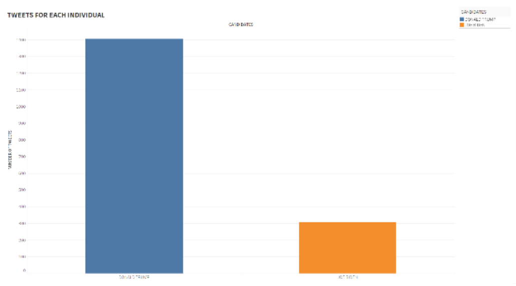
```
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [06/Dec/2020 05:51:41] "GET / HTTP/1.1" 200 -
```

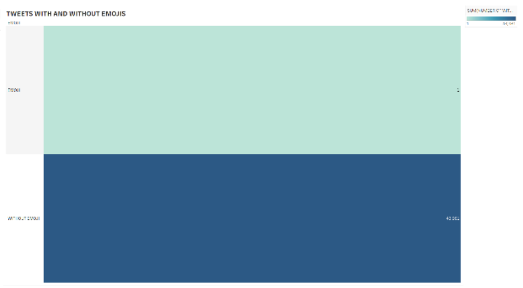
3. FINAL WEBSITE

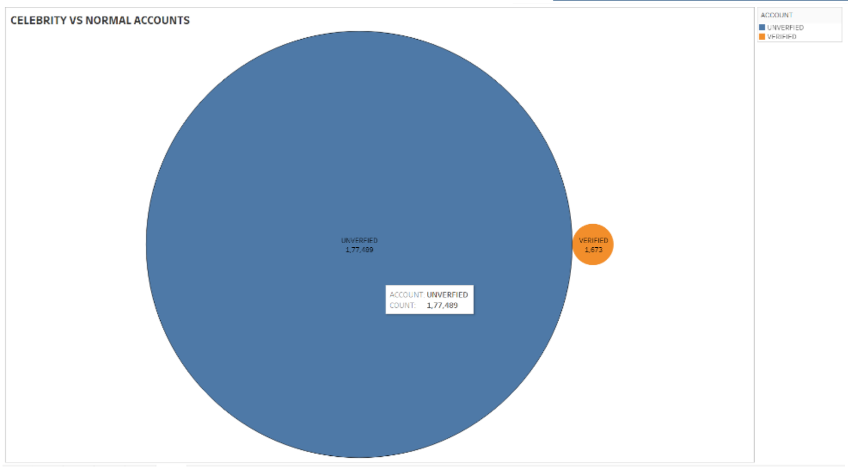


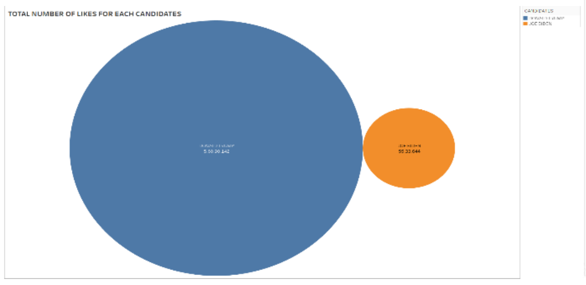
The website gets refreshed whenever there is any data change in the S3 bucket. It shows the real-time analysis of the backend work.

8. RESULT EVALUATION:









GEO-GRAPHICAL LOCATION

Here are some of the evaluation results that we have found while doing the data pre-processing. These results help us know how important is the data that we have received, and it also helps us to find the trend line in the market.

The above graphs have the following information like

1.Number of tweets for Individual

2. Tweets with emoji and without

3. Verified account vs normal accounts

4. Likes for each candidates' tweets

5. Displaying the people's location over the map.

This data can be useful in providing the weight to the tweet by which we can prioritize them to use in our model.

9. CONCLUSION:

We have implemented and developed a real-time application that can get real-time tweets from Twitter and use spark streaming to process the data by doing sentimental analysis over the data to know how the people are thinking over something that exists. This application can be so valuable to the industry who rely on the people opinion on knowing whether a product is being successful in the market or not. This application can also be used for understanding the trends in the market and it can also be used for making major decisions.

10. FUTURE WORK:

In keeping in mind, the scope of our that is could reach we have made our application flexible for future works.

10.1 MULTIPLE RESOURCES:

In future we can add more resources of data by streaming it our port. As of now we are only streaming from only one resource twitter, If possible, we would like add data from YouTube, Facebook etc.

10.2 BETTER SENTIMENTAL ANALYSIS:

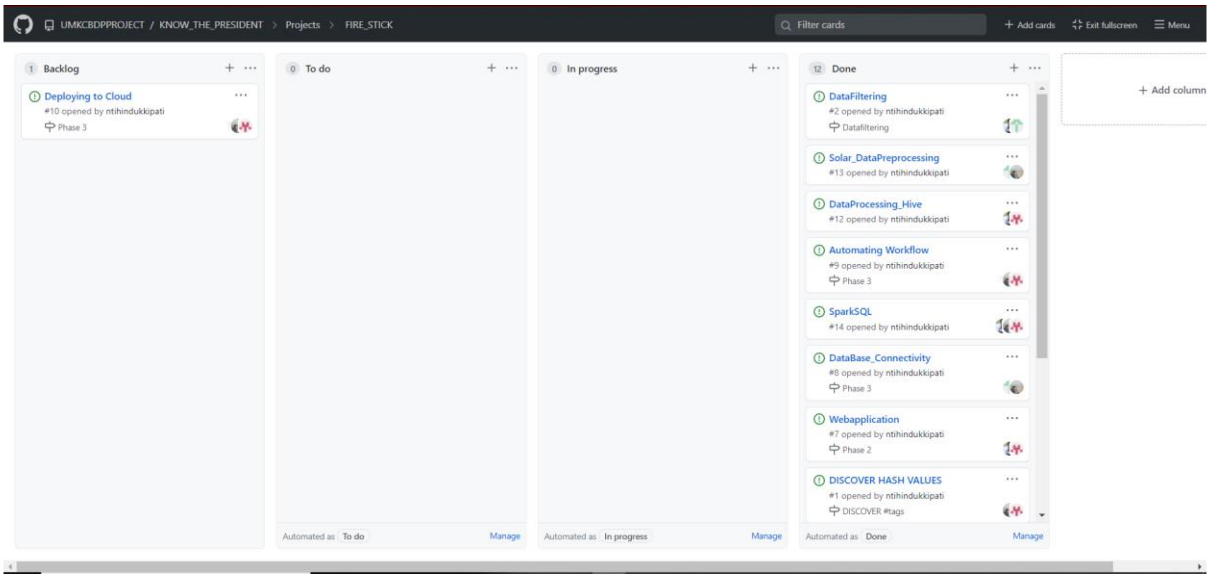
Due some time constraints we were not having much time to develop a deep learning model with recurrent neural network which can be more efficient than current model that we are using.

10.3 MIGRATING TO CLOUD:

As we are Streaming the data using the ports it’s not always optimal to use the ports within cloud services, but we can setup streaming using Kafka in the cloud to stream the data and store the tweets in s3.

11. PROJECT MANAGEMENT:

For Project Management we are using GitHub project Management tool to organize ideas and track our work.


UMKCBDPPROJECT / KNOW_THE_PRESIDENT > Projects > FIRE_STICK

Filter cards | + Add cards | Exit fullscreen | Menu

1 Backlog
- Deploying to Cloud
  #10 opened by ntihindukkipati
  Phase 3

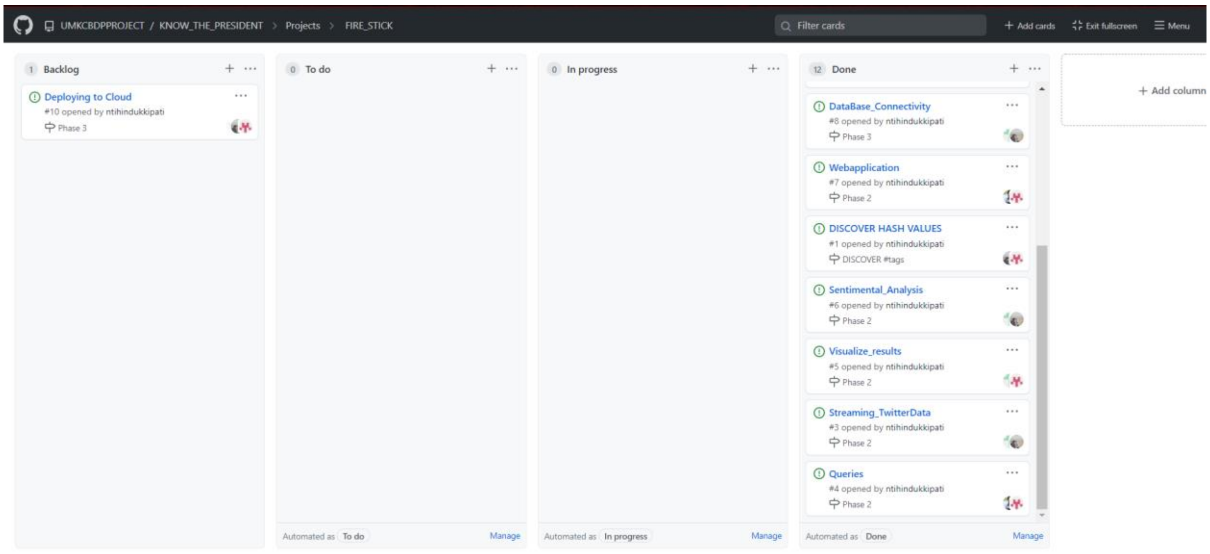
0 To do
Automated as To do | Manage

0 In progress
Automated as In progress | Manage

12 Done
- DataFiltering
  #2 opened by ntihindukkipati
  Datafiltering
- Solar_DataPreprocessing
  #13 opened by ntihindukkipati
- DataProcessing_Hive
  #12 opened by ntihindukkipati
- Automating Workflow
  #9 opened by ntihindukkipati
  Phase 3
- SparkSQL
  #14 opened by ntihindukkipati
- DataBase_Connectivity
  #8 opened by ntihindukkipati
  Phase 3
- Webapplication
  #7 opened by ntihindukkipati
  Phase 2
- DISCOVER HASH VALUES
  #1 opened by ntihindukkipati
  DISCOVER #tags

Automated as Done | Manage

+ Add column


UMKCBDPPROJECT / KNOW_THE_PRESIDENT > Projects > FIRE_STICK

Filter cards | + Add cards | Exit fullscreen | Menu

1 Backlog
- Deploying to Cloud
  #10 opened by ntihindukkipati
  Phase 3

0 To do
Automated as To do | Manage

0 In progress
Automated as In progress | Manage

12 Done
- DataBase_Connectivity
  #8 opened by ntihindukkipati
  Phase 3
- Webapplication
  #7 opened by ntihindukkipati
  Phase 2
- DISCOVER HASH VALUES
  #1 opened by ntihindukkipati
  DISCOVER #tags
- Sentimental_Analysis
  #6 opened by ntihindukkipati
  Phase 2
- Visualize_results
  #5 opened by ntihindukkipati
  Phase 2
- Streaming_TwitterData
  #3 opened by ntihindukkipati
  Phase 2
- Queries
  #4 opened by ntihindukkipati
  Phase 2

Automated as Done | Manage

+ Add column

INDIVIDUAL:

KOLLURI, NIKHITHA

1. In the initial stages I and Eshwar worked on collecting data resource and finding the valuables out of them.

2. Worked on the data pre-processing part where we wrote some queries in understanding and analyzing the data that we collected.

3. I have helped in getting twitter data from tweepy library and wrote spark streaming using Scala which has some incompatible issues which leads to streaming using python.

4. I have developed Web UI with Eshwar.

MALLEPUDI, CHAITANYA

1. I have worked on the querying part in the initial stages where we were trying to pre-process the data

2. worked on the Twitter streaming part and helped Sri in debugging the Spark Streaming

3. Wrote lambda function in AWS, to provide API gateway to the web Application, Tried to migrate the backend services to the cloud.

4. Have helped in sorting out small programming issues in a team

VALLURU, ESWAR

1. I have worked with Nikhitha in finding out the best resources for our application in terms of data to be feed to our application

2. Written some queries to get to know the data for further usage

3. Have helped Sri in solving some errors while using sockets in connecting twitter streamer and spark streaming application.

4. Written code for doing sentimental analysis over the data and helped Nikhitha when we were developing spark streaming using Scala.

DUKKIPATI, SRI SAI NITHIN CHOWDARY:

1. Worked on writing queries where we were more mostly focused to avoid taking data from biased hashtags.

2. Worked on Developing Spark Streaming application with help of all three members.

> **Chaitanya**: Helped in trying different map-reduce techniques which solved our ambiguity in looking for other things.
>
> **Eswar**: Helped me solving some errors when I was working on the sockets part.
>
> **Nikhitha**: Nikhitha helped me finding new resources and material for developing new techniques and she also helped in solving some errors.

3. Helped Chaitanya in writing lambda functions in AWS in setting up API Gateway

4. Helped in migrating backend code to the cloud. (Not achieved).

ISSUES AND CONCERNS:

1. Unable to migrate the entire application to the cloud.

2. As Specified in Increment 2 There are some concerns in streaming data using Scala.

3. The which we collected from Twitter in the initial stages is raw (not well-formed) so we need manually pre-process the data to get it in CSV format which later helped in analyzing the data.

12. STORYTELLING:

12.1. CHAPTER 1:

- **WHO**?
  People who are curious about the 2020 US Election inside as well as outside the United States.
- **WHAT?**
  People want to get awareness on the Elections and since the media is always biased people are unable to get the aggregate information and make the decision.
- **WHEN?**
  When people want to know whether the media and other people across the globe are on the same stand.
- **WHERE?**
  Across the globe wherever the media is highly influenced.
- **WHY?**
  Since the Media is mainly focusing on their ratings instead of the ground truth being represented to the people.

12.2. CHAPTER 2:

- **WHO**?
  The dataset is about Joe Biden and Donald Trump related to US 2020 Elections, there are no undersampled data and oversampled data. There is no risk in disclosing information as there is no sampling issue.
- **WHAT?**
  From the collected data noticeable fields like retweeted, likes, text, timestamp, location, followers count, likes, geo coordinates, friends, dislikes, display name, username, etc. are queried and visualized.
- **WHEN?**
  During US ELECTION 2020 the data was streamed from Twitter mostly real-time data, the data collected is longitudinal as it is only collected at this point of particular time across the world. The generalization can be done whenever required for a different point of events by changing the hashtags while collecting the data.
- **WHERE?**
  The event is taking place in the US, this data is been streamed from Twitter over time, This data is collected across the globe and contains GIS information of the user like geo coordinates. The generalization can be done wherever required at different locations by changing the hashtags while collecting the data.
- **WHY?**
  The data was collected to bring awareness among the people about the US elections by considering various attributes.

12.3. CHAPTER 3: Scientist and AI

- **WHO**?
  People who are going to vote in 2020 United States Presidential Election.
- **WHAT?**
  We have collected the data using Spark Streaming and further performed sentimental analysis using TextBlob, stored the data in cloud and populated it in the user interface.
- **WHEN?**
  During US ELECTION 2020 the data was streamed form the twitter mostly real time data, the data collected is longitudinal data as it is only collected at this point of particular time across the world. The generalisation can be done whenever required for at different point of events by changing the hashtags while collecting the data.
- **WHERE?**
  Spark Environment and sentiment analysis in python which is the part of bigdata course
- **WHY?**

  Collected the data using Spark Streaming and further performed sentimental analysis, stored the data and populated it in the user interface.

12.4. CHAPTER 4: USERS

- **WHO?**
  The main character is the user whoever is viewing the application for gaining knowledge on present situation.
- **WHAT?**
  It can bring awareness to the user of different opinions which are through out the globe. The visualization show about the sentiment(positive/negative) of candidates.
- **WHEN?**
  When ever user needs to know the trends.
- **WHERE?**
  The application cane deployed to the cloud.
- **WHY?**
  Application can be useful in understanding the current trends and get people to know how their opinion stands out when compared with others.
- **HOW?**
  This application can be used to help people in knowing what’s happening around them how they stand among the other people.

12.5. CHAPTER 5: SOCIETY

- **WHO?**
  The general audience. The presidential candidates were sampled. None of them were over or under sampled.
- **WHAT?**
  This will make neutral audience to make sides. Coming to fairness there will not be any influence and since we are collecting data from Twiiter which is a public domain platform there is security concerns.
- **WHEN?**
  Right before elections to know what people are thinking about both the candidates and after the election what people are thinking about the president.
- **WHERE?**
  The social and cultural impact takes at the place of the candidates on whom the data is collected. There will be no data breach and it is unlikely to happen.
- **WHY?**
  These impacts are consequential to the people to refrain themselves from influential media and people around them.
- **HOW?**
  We can collect the data and perform NLP tasks to do the sentiment analysis

13 REFERENCES AND BIBLIOGRAPHY:

https://medium.com/@anicolaspp/spark-streaming-and-twitter-sentiment-analysis-c860938d484

https://www.youtube.com/watch?v=uD_q4Rm4i2Q

https://www.edureka.co/blog/spark-streaming/

https://stackoverflow.com/questions/31466435/how-to-find-source-of-scala-matcherror

https://github.com/PacktPublishing/Hands-On-Deep-Learning-with-Apache-Spark/issues/1

https://docs.aws.amazon.com/s3/index.html?nc2=h_ql_doc_s3

https://docs.aws.amazon.com/apigateway/index.html