Sentiment Analysis of COVID-19 Tweets - Visualization Dashboard

A Project Submitted
in fulfilment of Requirements
for the Challenge

By
Pranav Bhaskar

Team Name
0xHEISENBERG

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Overview

Social media has always provided a platform for people to express their opinion and emotions. COVID-19 also known as the novel coronavirus has had a substantial effect on the routine as well as the mental state of the people. In this situation people are turning towards social media platfoWHOrms like twitter to express their sentiments towards the events taking place all across the country. Any data analyst will not miss this chance to analyse this flood of emotions.

1.2 Purpose

The purpose of the project is to analyse the sentiments of Indians and make a dashboard with visualization of people reaction to the government announcements on COVID-19 and it is lock-downs.

2. LITERATURE SURVEY

2.1 Existing Problem

It is not wrong to say that there is an abundance of data present on social networks like twitter and preprocessing the data and analysing it is not an easy task. Some of the problems are as follows:

1. Filtering the data
2. Hyderating tweet_id
3. Storing and processing such a huge amount of data
4. Building an UI to display the results

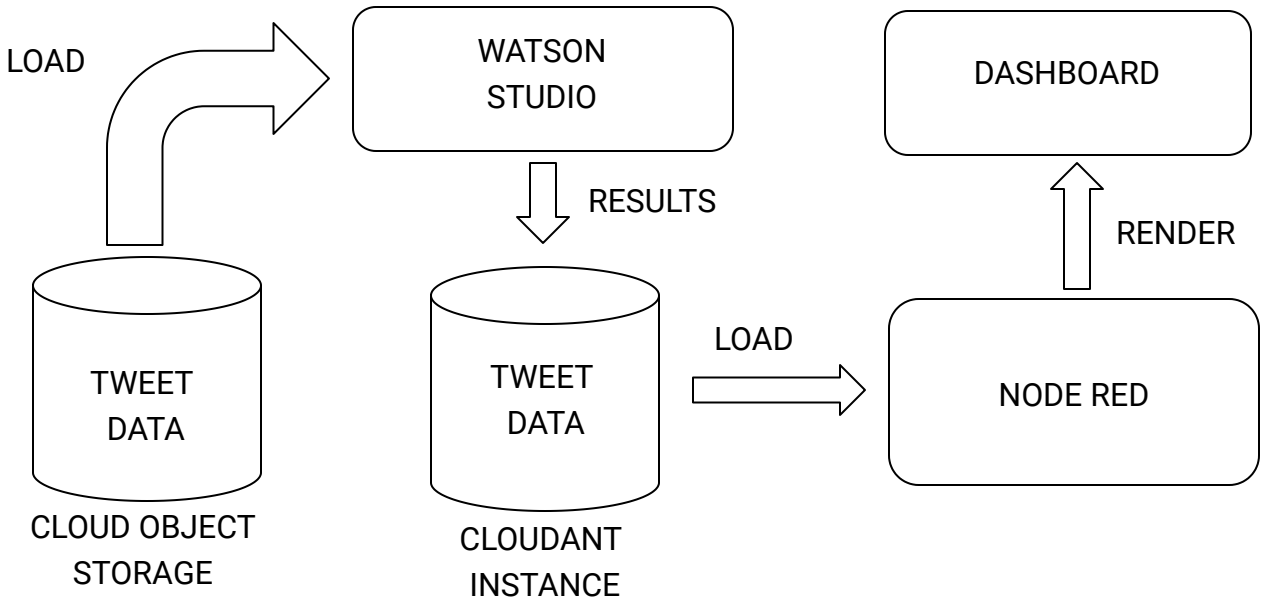
2.2 Proposed Solution

Most of the problems faced by us can easily be solved by using IBMCloud to build our project. Since IBM is providing us 5GB of Cloud Object Storage which can be connected to Watson Studio, where we can spawn notebooks with 8GB RAM which is more than sufficient to bear the load of the data. This notebook would be responsible for processing the data and storing the results to a Cloudant instance.

We would then be using Node-RED for hosting the UI and retrieving data from the Cloudant instance.

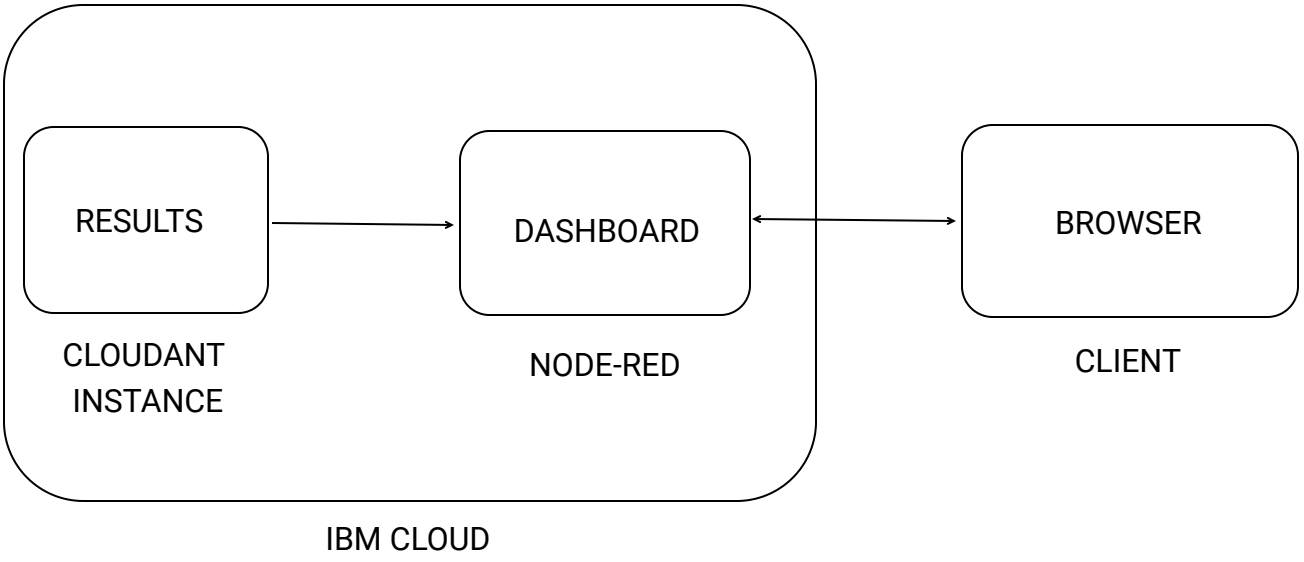
3. THEORETICAL ANALYSIS

3.1 Block Diagram



BLOCK DIAGRAM

3.2 Hardware/Software Design



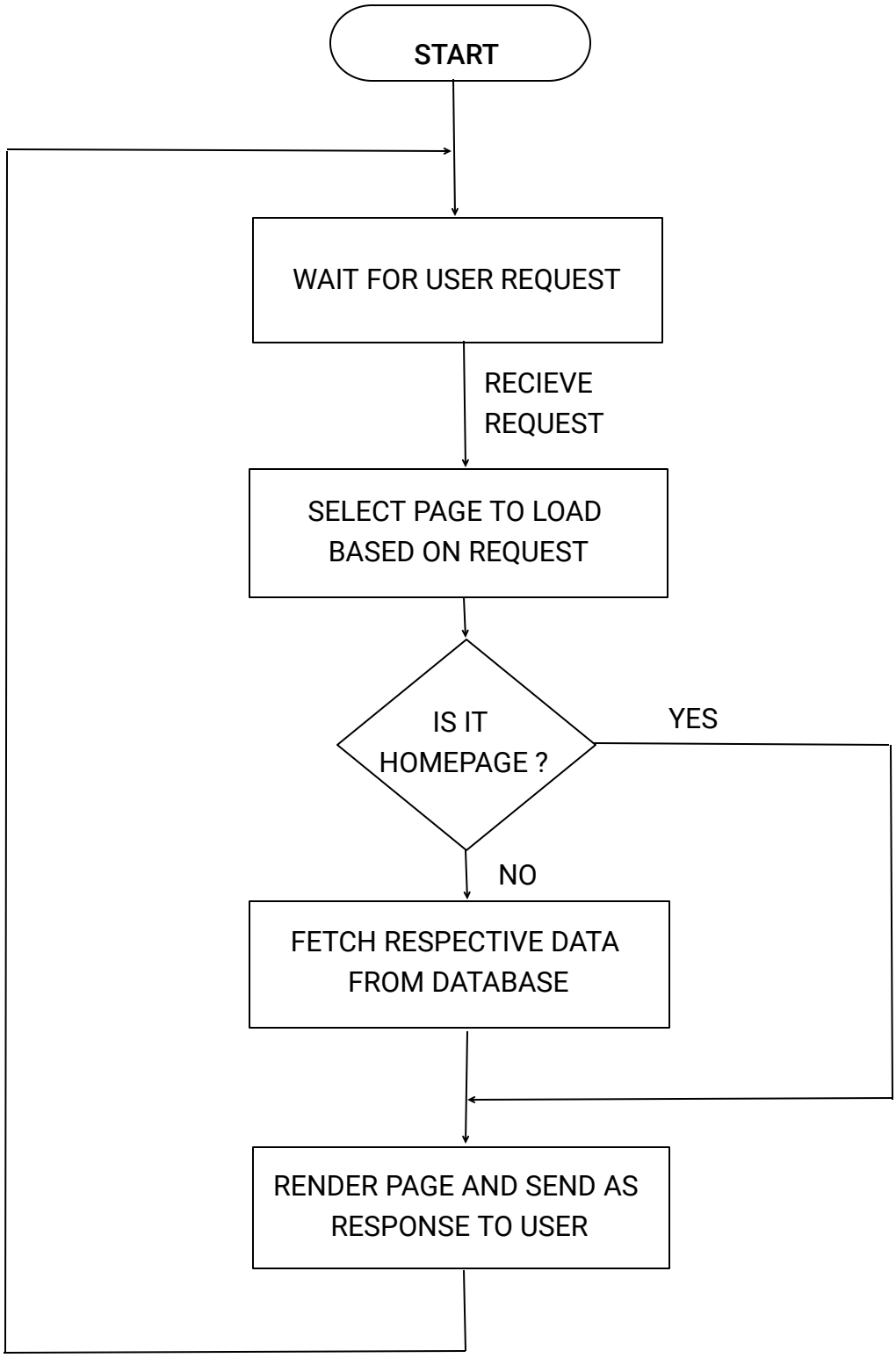
4. EXPERIMENTAL INVESTIGATIONS

There were a substantial number of datasets which were considered for this project, one of which was the IEEE dataset named "CORONAVIRUS (COVID-19) TWEETS DATASET." It was a good dataset for this project, but because not all tweet_ids in it were geotagged it had to be skipped as the twitter API provides only 900 hydrations of tweet_id per 15 mins.

To avoid this problem I went through the "CORONAVIRUS (COVID-19) GEO-TAGGED TWEETS DATASET" it consisted of tweet_id of only those tweets which are geo-tagged. This was a very generous gesture by the one who made this dataset (Rabindra Lamsal), but since we could only do 900 hydrations per 15 mins it was not used.

Later a dataset was used which was already hydrated, geo-tagged and made available for public use. From that dataset all the Indian tweets related to COVID-19 were then filtered. This was then used as the input for this project.

5. FLOWCHART



FLOW CHART

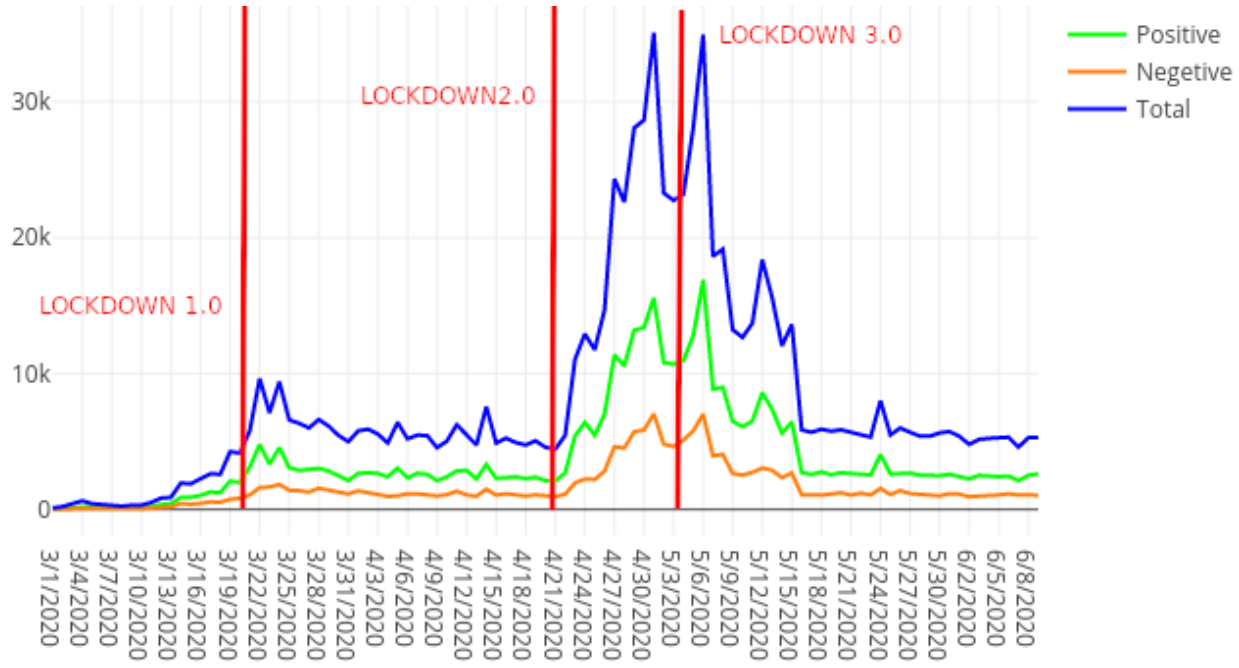
6. RESULT

A dashboard was created which consisted of interactive views (charts and tables) to get the abstract of sentiments of the people of India towards various government decisions along with some other interesting details like hashtags, geotags and user mentions.
Currently the site is hosted at: https://node-red-gqeir.eu-gb.mybluemix.net/

Some results I find interesting are:

UserMention	Positive	Negetive	Total
@narendramodi	16298	7219	32246
@PMOIndia	13864	7193	28864
@CMOMaharashtra	3263	1926	6867
@MoHFW_INDIA	3344	1301	6634
@AmitShah	2522	1580	5477
@ArvindKejriwal	2598	1431	5373
@KTRTRS	2364	1208	4640
@WHO	1896	858	4168

@CMOMaharashtra was mentioned more than @MoHFW_INDIA



Lockdown 2.0 caused an exponential growth in number of tweets

7. ADVANTAGES AND DISADVANTAGES

Some advantages of the project are listed bellow:

1. We have an interactive dashboard for visualizing the analysed data.
2. The same project can be run on a different situation to analyse the sentiments of people with just changing the input dataset.
3. The result of the project is quite lightweight and the dashboard hardly uses any computation.
4. The same dashboard can also be used to analyse other trends (hashtags and user mentions) which took place during the COVID-19.

Some disadvantages of the project are listed bellow:

1. No real time tweet data is being processed, since I did not have a premium or enterprise plan on twitter API, I could not run it on real time data without exhausting the API limit.

8. APPLICATIONS

Some applications of this project are:

1. Analyse the people’s sentiment towards the epidemic.
2. Understand the sentiments of people on government's decisions during the epidemic.
3. Analysing the sentiments of people based on their region.
4. Analysing the sentiments of people based on the hashtags used by them.
5. Analysing the sentiments of people based on the users mentioned by them.
6. Analysing the sentiments of people on other circumstances by just changing the input dataset of the project.

9. CONCLUSION

We have developed a dashboard for visualizing the twitter data collected during the COVID-19.

10. FUTURE SCOPE

We can add real time tweet processing to it which will keep the data updated on the site. However, for it to happen we would have to acquire a premium or enterprise level API from twitter.

11. BIBLIOGRAPHY

1. https://ieee-dataport.org/open-access/coronavirus-covid-19-tweets-dataset
2. https://ieee-dataport.org/open-access/coronavirus-covid-19-geo-tagged-tweets-dataset
3. https://www.kaggle.com/smid80/coronavirus-covid19-tweets
4. https://www.kaggle.com/smid80/coronavirus-covid19-tweets-early-april
5. https://www.kaggle.com/smid80/coronavirus-covid19-tweets-late-april
6. https://www.kaggle.com/vagavoludheeraj/twitter-dateset-collected-during-covid19-pandemic

APPENDIX

A. Source Code

The complete source code of the project along with the final result database (in json format) can be found at the github repository located at : https://github.com/SmartPracticeschool/SBSPS-Challenge-511-Sentiment-Analysis-of-COVID-19-Tweets---Visualization-Dashboard

The source code for the generation of resultdb from the dataset in the current directory on to Cloudant in a concise form is given bellow:

```
import json
import datetime
import pandas as pd
import re
from textblob import TextBlob
from cloudant.client import Cloudant
from cloudant.error import CloudantException
from cloudant.result import Result, ResultByKey
from cloudant.database import CloudantDatabase

username = "XXX"  #for cloudant
apikey = "XXX"  #for cloudant

def fetcher(dfs, date):
    file_name = 'data_' + date + '.csv'
    dfs.append(pd.read_csv(file_name, usecols=['Tweet Posted Time (UTC)', 'Tweet Content', 'Tweet Location']))
date = datetime.datetime(2020, 3, 1)
dfs = []
for _ in range(101):
    fetcher(dfs, date.strftime('%Y-%m-%d'))
    date += datetime.timedelta(days=1)
df = pd.concat(dfs, ignore_index=True)
df.columns = ['Date', 'Text', 'Location']

```

```
25 def dateFormater(date):
26     date = datetime.datetime.strptime(date, '%Y-%m-%d')
27     return date.strftime('%d/%m/%Y')
28
29 epoch = datetime.datetime.utcfromtimestamp(0)
30
31 def timeStamp(date):
32     date = datetime.datetime.strptime(date, '%d/%m/%Y')
33     return (date - epoch).total_seconds() * 1000.0
34
35 df['Date'] = df['Date'].apply(dateFormater)
36
37 df['TimeStamp'] = df.apply(lambda row:
   timeStamp(row['Date']), axis=1)
38
39 def hash_extractor(text):
40     hash_tags = re.findall('(#[^\s]*)', text)
41     return hash_tags
42
43 def mention_extractor(text):
44     user_mentions = re.findall('(@[^\s]*)', text)
45     return user_mentions
46
47 df['HashTags'] = df.apply(lambda row:
   json.dumps(hash_extractor(row['Text'])), axis=1)
48
49 df['UserMentions'] = df.apply(lambda row:
   json.dumps(mention_extractor(row['Text'])), axis=1)
50
51
52 df['SentimentScore'] = df['Text'].apply(lambda text:
   TextBlob(text).sentiment.polarity)
53
54 client = Cloudant.iam(username, apikey)
55 client.connect()
56 db = CloudantDatabase(client, 'resultdb')
```

```
if db.exists():
    print('DB already exists')
else:
    db.create()
    print('DB created')
dates = {}
location = {}
hashtags = {}
usermention = {}
for _, row in df.iterrows():
    # based on date
    ts = str(int(row['TimeStamp']))
    if ts not in dates:
        dates[ts] = {}
        dates[ts]['positive'] = 0
        dates[ts]['negetive'] = 0
        dates[ts]['neutral'] = 0
    if row['SentimentScore'] > 0:
        dates[ts]['positive'] += 1
    elif row['SentimentScore'] < 0:
        dates[ts]['negetive'] += 1
    else:
        dates[ts]['neutral'] += 1
    # based on date ends
    # on location
    loc = row['Location']
    if loc not in location:
        location[loc] = {}
        location[loc]['total'] = {}
        location[loc]['total']['positive'] = 0
        location[loc]['total']['negetive'] = 0
        location[loc]['total']['neutral'] = 0
    if ts not in location[loc]:
        location[loc][ts] = {}
        location[loc][ts]['positive'] = 0
        location[loc][ts]['negetive'] = 0
```

```
93          location[loc][ts]['neutral'] = 0
94      if row['SentimentScore'] > 0:
95          location[loc][ts]['positive'] += 1
96          location[loc]['total']['positive'] += 1
97      elif row['SentimentScore'] < 0:
98          location[loc][ts]['negetive'] += 1
99          location[loc]['total']['negetive'] += 1
100      else:
101          location[loc][ts]['neutral'] += 1
102          location[loc]['total']['neutral'] += 1
103      #location ends
104      # based on hashtag
105      for ht in json.loads(row['HashTags']):
106          if ht not in hashtags:
107              hashtags[ht] = {}
108              hashtags[ht]['positive'] = 0
109              hashtags[ht]['negetive'] = 0
110              hashtags[ht]['neutral'] = 0
111              hashtags[ht]['total'] = 0
112          hashtags[ht]['total'] += 1
113          if row['SentimentScore'] > 0:
114              hashtags[ht]['positive'] += 1
115          elif row['SentimentScore'] < 0:
116              hashtags[ht]['negetive'] += 1
117          else:
118              hashtags[ht]['neutral'] += 1
119      # end of hashtag
120      # based on usersmentioned
121      for um in json.loads(row['UserMentions']):
122          if um not in usermention:
123              usermention[um] = {}
124              usermention[um]['positive'] = 0
125              usermention[um]['negetive'] = 0
126              usermention[um]['neutral'] = 0
127              usermention[um]['total'] = 0
128          usermention[um]['total'] += 1
```

```
129         if row['SentimentScore'] > 0:
130             usermention[um]['positive'] += 1
131         elif row['SentimentScore'] < 0:
132             usermention[um]['negetive'] += 1
133         else:
134             usermention[um]['neutral'] += 1
135     # end of usermentions
136 
137 #removing extra hashtags
138 for k in list(hashtags):
139     if hashtags[k]['total'] < 20:
140         del hashtags[k]
141 #removing extra usermentions
142 for k in list(usermention):
143     if usermention[k]['total'] < 10:
144         del usermention[k]
145 json_doc = {
146     '_id': 'date',
147     'date': dates
148 }
149 db.create_document(json_doc)
150 
151 json_doc = {
152     '_id': 'location',
153     'location': location
154 }
155 db.create_document(json_doc)
156 
157 json_doc = {
158     '_id': 'hashtag',
159     'hashtag': hashtags
160 }
161 db.create_document(json_doc)
162 
163 json_doc = {
164     '_id': 'usermention',
```

```
165     'usermention': usermention
166 }
167 db.create_document(json_doc)
```