Internship Report

Participant: Arkajyoti Roy

College: University of California San Diego, Fall Quarter Sophomore

Date: September 30, 2020

Objective

Find actionable insights by analyzing tweets related to Covid-19.

Tools & Technologies

Development Environment

Jupyter Notebook

- Jupyter Notebook is a web application that allows you to run live code, embed visualization and explanatory text all in one place.
- The reason for using this over an IDE is the visualization aspect of Jupyter notebook. Running on pycharm or a similar IDE, you can't utilize graphs, tables and charts like you can in Jupyter Notebook.
- It is also easy to share and present your research by using Jupyter.

Programming Technology

Spark for fast Big Data processing and analytics

- Apache Spark is an open-source processing engine used to store and process data.
- We use Spark because it has optimized query execution for fast queries against data.
- Spark also runs on memory(RAM) so it makes the processing faster than on disk drives.
    - It wouldn't make sense to accessing storage (hard drive) every time, so now there is memory (RAM) or in-memory view which enables fast access

Pandas for statistical analysis and visualization

- We use Pandas DataFrame because it allows us to utilize two-dimensional labeled data structures with columns of different types (potentially)
    - We can label the indices
    - Dictionary based NumPy arrays
- A DataFrame is a set of records

Hosting the application in Cloud

Setup the above Development Environment in Google Cloud and use the above mentioned Technologies to create the application and run in Cloud.

Google Cloud offers $300 credit for 1 year so I stored the tweets in Google Cloud Storage. Google Cloud allows you to process, store and analyze your data all in one place. Cloud hosting gives you the advantage because it is too expensive to do these tasks with traditional database systems.

Setup Steps

1. Get access to the bucket storing data
    a. Covid19-internship was given read access to fetch bucket.
2. On the project selector page, select or create a Cloud project.
    a. Press “Create Project”
    b. Enter “covid19-internship” in Project name
3. Once in the project, select Billing from the Navigation dropdown
    a. Enable billing on the project by linking billing account
4. Register your application for Compute Engine API in Google Cloud Platform in order to manage our application and monitor API usage
5. Once the API has been enabled, check the credentials in order to use the API
6. Create a new instance of the AI Platform Notebooks
    a. Select “Python 3”
    b. Select Machine type to be “n1-standard-16”
    c. Set instance name as “covid-19 instance”

Demonstration

Environment Setup and Library installation

[4]:
```
from google.cloud import storage
```

[5]:
```
client = storage.Client()
print("Client created using default project: {}".format(client.project))
```

```
Client created using default project: covid19-internship
```

[6]:
```
buckets = client.list_buckets()

print("Buckets in {}:".format(client.project))
for item in buckets:
    print("\t" + item.name)
```

```
Buckets in covid19-internship:
```

[7]:
```
!java -version
```

```
openjdk version "1.8.0_252"
OpenJDK Runtime Environment (build 1.8.0_252-8u252-b09-1~deb9u1-b09)
OpenJDK 64-Bit Server VM (build 25.252-b09, mixed mode)
```

[8]:
```
#! sudo apt-get install -y openjdk-8-jdk-headless -qq > /dev/null
```

[9]:
```
import os
```

[10]:
```
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["PATH"] = os.environ["JAVA_HOME"] + "/bin:" + os.environ["PATH"]
```

[1]:
```
from pyspark.sql import SparkSession
```

[2]:
```
def start():
    builder = SparkSession.builder \
        .appName("Spark NLP Licensed") \
        .master("local[*]") \
        .config("spark.driver.memory", "24G") \
        .config("spark.serializer", "org.apache.spark.serializer.KryoSerializer") \
        .config("spark.kryoserializer.buffer.max", "2040M") \
        .config("spark.jars.packages", "com.johnsnowlabs.nlp:spark-nlp_2.11:2.5.1") \
        .config("fs.gs.impl", "com.google.cloud.hadoop.fs.gcs.GoogleHadoopFileSystem") \
        .config("fs.AbstractFileSystem.gs.impl", "com.google.cloud.hadoop.fs.gcs.GoogleHadoopFS")
    return builder.getOrCreate()
spark = start()
spark.version
!ps -ef | grep spark
```

```
jupyter  31866 31814 98 09:14 ?        00:00:08 /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java -cp
on3.7/site-packages/pyspark/jars/* -Xmx24G org.apache.spark.deploy.SparkSubmit --conf spark.master=
fer.max=2040M --conf spark.jars.packages=com.johnsnowlabs.nlp:spark-nlp_2.11:2.5.1 --conf spark.ser
e=Spark NLP Licensed --conf fs.AbstractFileSystem.gs.impl=com.google.cloud.hadoop.fs.gcs.GoogleHadc
ystem pyspark-shell
jupyter  32039 31814  0 09:14 pts/0    00:00:00 /bin/bash -c ps -ef | grep spark
jupyter  32041 32039  0 09:14 pts/0    00:00:00 grep spark
```

- Data is in “cloud storage” -- held in buckets
- From pyspark.sql import SparkSession is the entry point library
- !ps -ef will check and print processes -> “jupyter”
- grep spark
- Spark is the main controller
- Hands instruction over to spark

Sentiment Analyzer

```
[26]: from pyspark.sql.functions import udf

def sentiment_analyzer_scores(text):
    score = analyser.polarity_scores(text)['compound']
    #print("{:-<40} {}".format(text, str(score)))
    return score

sentiment_analyzer_scores = udf(sentiment_analyzer_scores)
spark.udf.register("sentiment_analyzer_scores", sentiment_analyzer_scores)
```

- Used VADER Sentiment Analysis to find sentiment polarity scores for various queries
- It is necessary to register the sentiment_analyzer_scores() function because I was unable to create my own custom function and run that against the tweets database
- A compound polarity score lies between -1 (most extreme negative) and +1 (most extreme positive)
- A score between:
  - -1 and -0.4 is HIGH negative
  - -0.4 and 0 is LOW negative
  - 0 and 0.4 is LOW positive
  - 0.4 and 1 is HIGH positive

```
[56]: sentiment_analyzer_scores("The phone is horrible.")
The phone is horrible.------------------ {'neg': 0.538, 'neu': 0.462, 'pos': 0.0, 'compound': -0.5423}

[57]: sentiment_analyzer_scores("The phone is kinda horrible.")
The phone is kinda horrible.------------ {'neg': 0.445, 'neu': 0.555, 'pos': 0.0, 'compound': -0.4951}

[60]: sentiment_analyzer_scores("The phone is kinda horrible. 😭")
The phone is kinda horrible. 😭---------- {'neg': 0.348, 'neu': 0.652, 'pos': 0.0, 'compound': -0.4951}

[61]: sentiment_analyzer_scores("The phone is kinda horrible! 😭")
The phone is kinda horrible! 😭---------- {'neg': 0.368, 'neu': 0.632, 'pos': 0.0, 'compound': -0.5422}

[62]: sentiment_analyzer_scores("The phone is kinda HORRIBLE! 😭")
The phone is kinda HORRIBLE! 😭---------- {'neg': 0.414, 'neu': 0.586, 'pos': 0.0, 'compound': -0.6407}
```

As shown in the image above, the polarity scores take into account Punctuation (!), Capitalization to emphasize a sentiment relevant word (“HORRIBLE”), Intensifiers and slang(“kinda”) and even emojis (😭).

Sentiment towards President's involvement in Coronavirus Pandemic

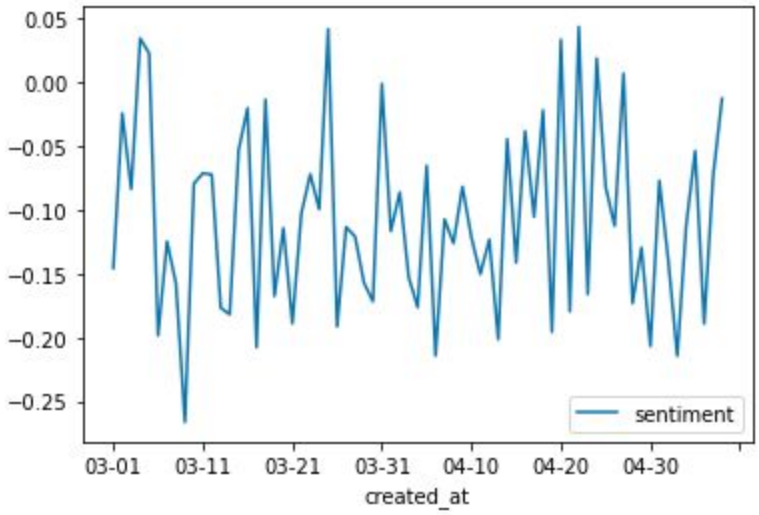
[5]:
```
from pyspark.sql.functions import from_unixtime
from pyspark.sql.functions import unix_timestamp
from pyspark.sql.functions import col

tempDF20 = spark.sql("""SELECT full_text AS text, created_at FROM tweetViewV11 WHERE
                        full_text rLIKE 'trump|donald trump|president of united states|POTUS'""")
convertedDateV4 = tempDF20.withColumn('created_at', from_unixtime(unix_timestamp(col("created_at"),
                                      "EEE MMM dd HH:mm:ss ZZZZ "), "MM-dd"))
convertedDateV4.show()
```

```
tempDF40 = spark.sql("""SELECT text, created_at FROM president_data_table""")
tempDF41 = tempDF40.withColumn("Sentiment", sentiment_analyzer_scores('text'))
tempDF41.show()
```

[38]:
```
monthly_Trump_Sentiment = spark.sql("""SELECT created_at, AVG(Sentiment) AS sentiment FROM
                                    trump_sentiment GROUP BY created_at ORDER BY created_at ASC""")
monthly_Trump_Sentiment.show(100, False)
```

[36]:
```
import numpy as np
import pandas as pdf
```

[39]:
```
pdf = monthly_Trump_Sentiment.toPandas()
pdf.plot.line(x = "created_at", y= "sentiment")
```



From the initial time-period of March when the lockdown started to the beginning of May, the sentiment towards President Donald Trump handling of the pandemic fluctuates between 0.05 and -0.25. The overall sentiment can be classified as "LOW negative".

GeoSpatial Analysis of the Pandemic

[16]:
```
def getGeoJson(pdf):
    features = []
    centroid = False
    fuzz = 0.01
    for index, row in pdf.iterrows():
        geodata = {
            "type": "Feature",
            "properties": {
                "name": row["user"]["name"],
                "screen_name": row["user"]["screen_name"]
            }
        }

        if row["geo"]:
            geodata['geometry'] = {
                "type": "Point",
                "coordinates": [
                    row["geo"]["coordinates"][1],
                    row["geo"]["coordinates"][0]
                ]}
        elif row["place"] and any(row["place"]["bounding_box"]):
            bbox = row["place"]["bounding_box"]["coordinates"][0]

            if centroid:
                min_x = bbox[0][0]
                min_y = bbox[0][1]
                max_x = bbox[2][0]
                max_y = bbox[2][1]

                fuzz_x = fuzz * random.uniform(-1,1)
                fuzz_y = fuzz * random.uniform(-1,1)

                center_x = ((max_x + min_x) / 2.0) + fuzz_x
                center_y = ((max_y + min_y) / 2.0) + fuzz_y

                geodata['geometry'] = {
                    "type": "Point",
                    "coordinates": [
                        center_x,
                        center_y
                    ]
                }

            else:
                geodata['geometry'] = {
                    "type": "Polygon",
                    "coordinates": [
                        [
                            bbox[0],
                            bbox[1],
                            bbox[2],
                            bbox[3],
```

```
                            bbox[0]
                        ]
                    ],
                }

        if 'geometry' in geodata:
            features.append(geodata)

    geojson = {"type" : "FeatureCollection", "features": features}
    return geojson
```

[19]:
```
import json

pdf = outbreakDF1.toPandas()
json_object = json.dumps(getGeoJson(pdf), indent = 2)
with open("outbreak1.geojson", "w") as outfile:
    outfile.write(json_object)
```

[20]:
```
!pip install folium
```

```
Collecting folium
  Downloading folium-0.11.0-py2.py3-none-any.whl (93 kB)
     |████████████████████████████████| 93 kB 1.8 MB/s eta 0:00:01
Collecting branca>=0.3.0
  Downloading branca-0.4.1-py3-none-any.whl (24 kB)
Requirement already satisfied: jinja2>=2.9 in /opt/conda/lib/pytho
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/s
Requirement already satisfied: requests in /opt/conda/lib/python3.
Requirement already satisfied: MarkupSafe>=0.23 in /opt/conda/lib/
Requirement already satisfied: chardet<4,>=3.0.2 in /opt/conda/lib
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/lib/pyth
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/li
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.
Installing collected packages: branca, folium
Successfully installed branca-0.4.1 folium-0.11.0
```

[23]:
```
import folium
from folium import GeoJson
from folium import plugins

kw = {'location': [48, -102],'zoom_start': 2,'max_bounds': True}
m1 = folium.Map(tiles="OpenStreetMap", **kw)
GeoJson(getGeoJson(pdf)).add_to(m1)
#m.save('GeoJSONWithoutTitles_0.html')
m1
```

Results of the March through Beginning of May data shows that the coronavirus pandemic has reached the US and South America. Tweets related to the coronavirus are concentrated very high in the United States compared to other nations.

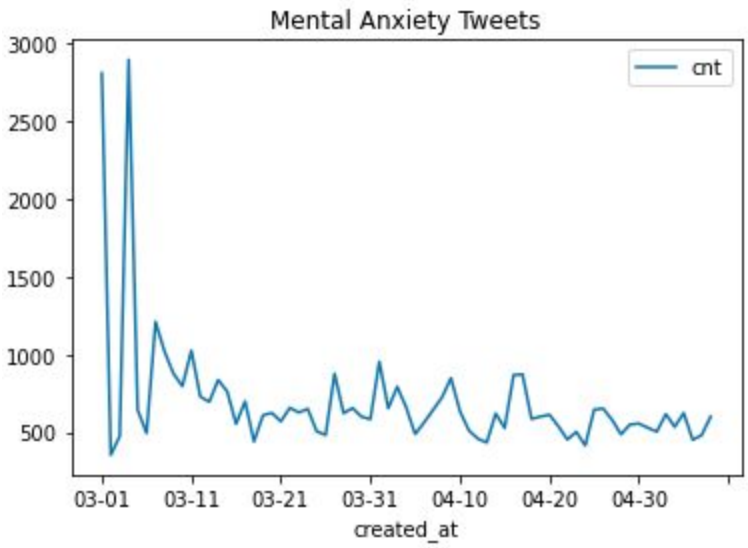
Analyze Mental Anxiety

[88]:
```
from pyspark.sql.functions import from_unixtime
from pyspark.sql.functions import unix_timestamp
from pyspark.sql.functions import col

tempDF14 = spark.sql("""SELECT full_text AS text, created_at FROM tweetViewV11 WHERE full_text
                        rLIKE 'fear|scared|depression|sad|stress|worried|restless|mental health|
                        toxic|depressed|anxiety'""")
convertedDateV2 = tempDF14.withColumn('created_at', from_unixtime(unix_timestamp(col("created_at"),
                                                                  "EEE MMM dd HH:mm:ss ZZZZ "), "MM-dd"))
convertedDateV2.show()
```

[89]:
```
convertedDateV2.write.saveAsTable("mentalHealth_data_table")
```

[90]:
```
monthlyCount = spark.sql("""SELECT created_at, COUNT(*) AS cnt
                            FROM mentalHealth_data_table
                            GROUP BY created_at ORDER BY created_at ASC""")
monthlyCount.show(100,False)
```

[93]:
```
import numpy as np
import pandas as pdf

pdf = monthlyCount.toPandas()
pdf.plot.line(x = "created_at", y = "cnt", title="Mental Anxiety Tweets")
```



At the start of March, there is a sharp spike in the graph that indicated a large number of tweets that mention one of the mental anxiety terms. There seems to be high concern and anxiety surrounding the coronavirus as well as the lockdown that starts at the beginning of May. Gradually, the degree of negativity reduces greatly from the beginning influx of tweets but still remains throughout March, April and May.

[87]:
```
tempDF54 = spark.sql("""SELECT User.name, User.screen_name, full_text, place.full_name
                        FROM tweetViewV11 WHERE full_text rLIKE 'fear|panic|mental health|
                        insomnia|depression|uncertainty|sick|anxiety|loneliness|sad|stress|
                        worried|nervous|restless'""")
tempDF54.show()
tempDF54.write.saveAsTable("users_mental_data")
```

```
+--------------------+---------------+--------------------+---------+
|                name|    screen_name|           full_text|full_name|
+--------------------+---------------+--------------------+---------+
|               chloé|       jnsxchlo|RT @disposableteе...|     null|
|                MONA|    _ramonnnaaa|RT @BiancaXaviera...|     null|
|Bebe 😎 Casey #Sc...|  1958_BestYear|RT @kimbakit: Tha...|     null|
|                birb| brittneymmarks|RT @TygrrTV: Star...|     null|
|      🖤💛Marsha🖤💛|_marisakathryn_|These college kid...|     null|
|            D. $mith|      DLSmith24|RT @korndiddy: Pl...|     null|
|                   J|          jx12_|RT @BiancaXaviera...|     null|
|       Niles Niemuth|  niles_niemuth|RT @gmarlowe1917:...|     null|
|      Marisol Toledo|         Pichol|RT @YoYo_Ma: In t...|     null|
|       Taylor Shears|  taylor_shears|One of my favorit...|     null|
|   𝔈𝔩 𝔐𝔲𝔢𝔯𝔱𝔢|       hawthrn_|RT @eyy_bby: "Sap...|     null|
|            ClutchAs|       ClutchAs|@bballbreakdown "...|     null|
|        Barbara Doan|    Doanziegirl|RT @megynkelly: I...|     null|
|      Hillary Nelson|cgardenwkitchen|RT @JessieNYC: If...|     null|
|Build the Wall - ...|wrdennis56_bill|RT @CDCgov: If yo...|     null|
|      Patricia Haley|patriciahaley62|RT @AmbassadorRic...|     null|
|Dennis J.S.S 🇱🇷...|    denjsalazar|RT @nowthisnews: ...|     null|
|              Aliento|      AlientoAZ|"During the COVID...|     null|
|Lisa 💋🇺🇸...|RLiberalskiddin|RT @CDCgov: If yo...|     null|
|              Vince|    seriousserb|RT @CDCgov: If yo...|     null|
+--------------------+---------------+--------------------+---------+
only showing top 20 rows
```

[84]:
```
tempDF56 = spark.sql("""SELECT screen_name, COUNT(full_text) AS cnt
                        FROM users_mental_data GROUP BY screen_name
                        ORDER BY cnt DESC""")
tempDF56.show()
```

```
+---------------+---+
|    screen_name|cnt|
+---------------+---+
|   SimoneSS1971| 16|
|  varaprasadnik| 12|
| world_news_eng| 10|
|trailblazer1408|  9|
|LadyBookworm117|  8|
|    VIKBataille|  8|
| Lorena35316563|  8|
|     davidr8203|  8|
|Masky_Jay_Hoody|  8|
|   jazmasigan_2|  8|
|   Bot_Corona_V|  7|
|   OverFlowless|  6|
|TomthunkitsMind|  6|
|  chargersfann1|  6|
|     All435Reps|  6|
|   newsfilterio|  6|
|       PulpNews|  6|
| Shawan_J_Singh|  6|
|Alfamal61368204|  6|
|        Caixa2E|  6|
+---------------+---+
only showing top 20 rows
```

I found usernames that tweeted multiple tweets that included one of the mental anxiety terms. We should find a way to send these users uplifting messages. Furthermore, it might be critical to keep an eye on such users to make sure their mental health problems don't escalate. In the future, we should look into extracting ethnicity from such users in order to find whether the coronavirus is affecting certain ethnicities more than others.