Title:

COVITA - Covid19 Text Analyzer

Participants:

Kaniska Mandal (kaniska.mandal@gmail.com)

Anil Berry (anilberry@gmail.com)

Description:

The goal for project COVITA (Covid19 Text Analyzer) is to analyze the Covid19 texts like medical research documents and tweets in order to find the relevant topics, understand user intents, find geospatial outbreaks, identify and heal mental anxieties , find availability of medical kits. Once we showcase the capability of our text analysis, we would like to extend it further to perform medical document recommendation, identify sensitive information, spread positive uplifting messages , predict outbreaks and create a marketplace for medical kit providers.

Codebase:

https://github.com/hacking-for-humanity/COVITA

Setup:

- First we created cluster and executed are notebooks in **Databricks** cluster **for all types of initial Exploratory Analysis and NLP** (**Appendix-A)**
- Since we have **a storage quota limitation inside Databricks**, we decided to **store the data inside Azure and access from Databricks**.
- Access the data from Azure Storage inside Databricks (see **Appendix-C**)
- Azure offers 1 month of free subscription and limited CPU and Memory
- Overall Databricks Instances helped us getting started quickly
- Since **Google Cloud offers 1 year of free account with $300 credit** we decided to store 100 million tweets in Google Cloud Storage
- We connected with google cloud from Databricks Notebook.
- But **eventually moved to Free Notebook Instance (400G Disk, 60 G RAM) offered by Google so that we don't need to worry about shutting down the instances and we don't face memory issues.** (**Appendix-B)**
- Once the Notebook Instance is setup with Http / Https access enabled, we can connect to the instance using local port forwarding: **gcloud compute ssh --project <my_project> --zone us-west1-b <my_vm> -- -L 8080:localhost:8080**

Data Source:

- Cord-19 Research Paper data: https://www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge
- Covid 19 Twitter Data: https://github.com/echen102/COVID-19-TweetIDs
    - Generate compressed json files using utility https://github.com/DocNow/twarc/tree/master/utils

    - We experimented with mapByPartition to speed up tweet hydration process
    - Link to our initial code:
        - https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/2963169389322382/714053717136170/1585028396443806/latest.html
        - https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/Utils.py

    - Next upload the compressed json files either manually to Databricks notebooks or use following script to store in Google Bucket
      gsutil-cp -r <hydrated_compressed_tweets>
      gs:/bucket-covid/TweetData/COVID-19-TweetIDs-master/2020-03/

Features:

Common Steps:

- Initialize Spark Session

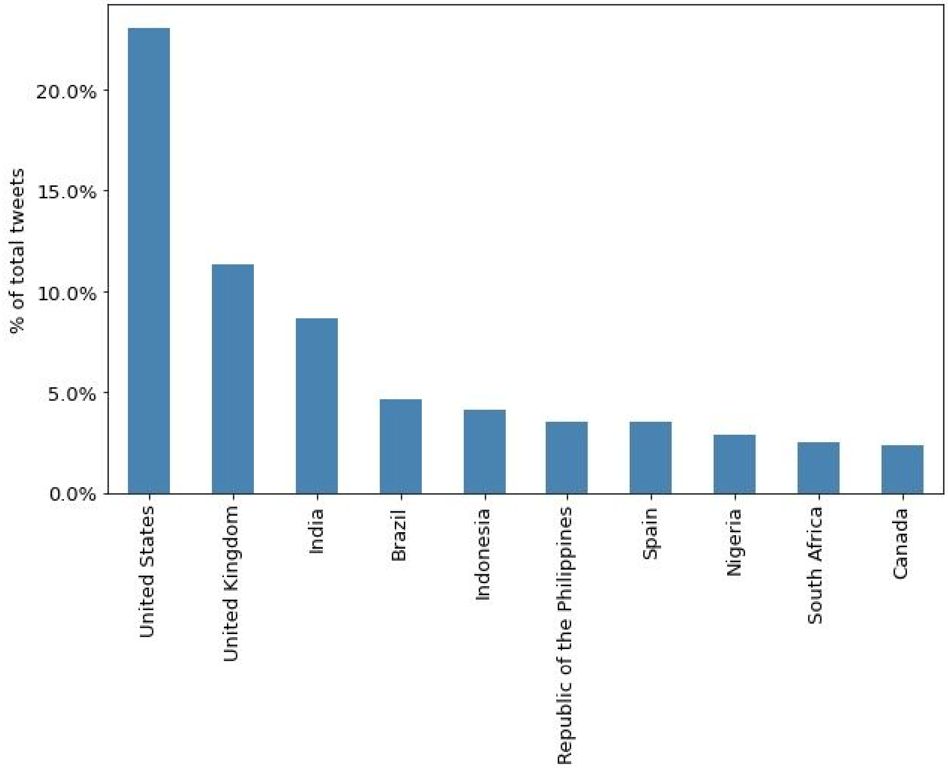
```
 builder = SparkSession.builder \
        .appName("Spark-COVITA") \
        .master("local[*]") \
        .config("spark.driver.memory", "54G") \
        .config("spark.serializer", "org.apache.spark.serializer.KryoSerializer") \
        .config("spark.kryoserializer.buffer.max", "2040M") \
        .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")
```

- Load Sample Data
    - sourceSampleData = spark.read.format("json").load("gs://covid19-tweets/2020-04/coronavirus-tweet-id-2020-04-15*.jsonl.gz")
    -
- Use parquet and local view for faster query
    - sourceSampleData.repartition(100).write.save("sampleParquet.parquet")
    - sampleParquet = spark.read.parquet("sampleParquet.parquet")
    - sampleParquet.createOrReplaceTempView("tweetView")
- Tweet text cleanup and extracting tokens

Analyze the Hash-tags, User-mentions, Follower and Favourite count data

- **Basic Analysis - Overall Tweet Distribution**

```
[298]: # plot tweets counts by country of origin
top_10_countries = pdf.country.value_counts(1).head(10)
ax = top_10_countries.plot(kind='bar',figsize=(10,6), fontsize=13, color='steelblue')
plt.ylabel('% of total tweets', fontsize=13)
ax.yaxis.set_major_formatter(mtick.PercentFormatter(1))
```



Link to code:
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/MentalAnxietyAnalysisV2.ipynb

- Find most important user mentions
    - Observations
        - **Head of WHO is mentioned most Tedros Adhanom**
        - **Other top influencers are Bill Gates , Saint Laurent Don , Nancy Pelosi , CNN , Donald J. Trump**

- Find the possibly sensitive tweets and its relationship with other metrics (retweet_count, favourite_count, followers_count)
        - **On average Sensitive tweets have very low retweet count**
        - **A possibly sensitive tweet may actually come from a trusted handle if it has high followers_count and high favourite_count compared to others sending sensitive tweets.**
        - **Few Fake News Sources also got captured in this analysis**

        - **Example:**

username	sensitive_count	retweet_count	favourite_count	followers_count
Somsirsa Chatterjee	19	0	0	796
Against Ignorance	10	0	0	157
Prince Neal_Agniv...	9	0	0	318
ดกนปหด🇻🇳🇺🇸J?...	8	9	0	2775
Francesca BaiMuDa...	8	12	0	4926
George	8	0	0	434
Kim Kardashian	8	0	0	1030
uMbhali Wodumo	7	0	0	3645
James Wu	7	7	7	2
The Daily Lafayette	7	0	2	693
Birmingham Live	7	11	28	297097
Miles to Go	7	76	2	816
King Lee	6	0	0	7
Servelan, reclaim...	6	1	0	2584
Sir Gary The Econ...	6	120	0	1391
CATHERINE STEVENS	6	0	0	67
CafeNetAmerica	5	1	0	5551
Nectes Gospel Med...	5	0	0	284
Sioux Falls News	5	0	0	90
News365.co.za	5	0	0	11904

- As the next step we want to find how the most influential hashtags , handles with quality tweets and true source of information actually helping people i.e. getting re-tweeted and marked as favorite not just for few days but over a longer duration of the pandemic

  Link to Code:
  https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/TweetUserAnalysisV1.ipynb

- We have planned to analyze all the public state Twitter Handles to extract major announcements and important news
  https://covidtracking.com/data#state-CA

- The goal is to build a recommendation model after creating clusters of topics and assigning handles and hashtags to such clusters.

Extract & Analyze the Entities like (Persons, Organizations, Events, National Groups, Locations) from Tweets

- **Extract entities using Spacy both using NER and Rule-based-matching**

nlp = spacy.load('en_core_web_sm') ⇒ English multi-task CNN trained on OntoNotes. Assigns context-specific token vectors, POS tags, dependency parse and named entities.
Example of basic NLP output

TEXT	LEMMA	POS	TAG	DEP	SHAPE	ALPHA	STOP
U.K.	u.k.	PROPN	NNP	compound	X.X.	False	False

- We extract many different types of entities - people,organisations, nationalgroups ,events, facilities , products,locations

- We analyze the frequency and relative sentiment (using vaderSentiment.SentimentIntensityAnalyzer) of each Entity
- We feed both clean tokens and raw texts to Spacy Named-Entity-Recognizer
- We also apply a Custom Entity-based Ruler to focus on a set of given Entities

```
persons = [('boris johnson','johnson'),('boris','johnson'),('johnson','johnson'),('prime minister','johnson'),
           ('primeminister','johnson'),('matt','hancock'),('hancock','hancock'),('matt hancock','hancock'),
           ('health secretary','hancock'),('deborah birx ','birx'),('deborah','birx'),('birx','birx'),
          ('anthony stephen fauci','fauci'),('anthony fauci','fauci'),('POTUS','trump'),('trump','trump'),
           ('president of united states','trump'),('donald trump','trump')]
orgs = [('nhs','nhs'),('cdc','cdc'),('who','who'),('world health organization','who'),('fda','fda'),
        ('government','government')]

ruler_persons = EntityRuler(nlp, overwrite_ents=True)
ruler_orgs = EntityRuler(nlp, overwrite_ents=True)

for (p,i) in persons:
    ruler_persons.add_patterns([{"label": "PERSON", "pattern": [{"LOWER": p}], "id": i}])
for (o,i) in orgs:
    ruler_orgs.add_patterns([{"label": "ORG", "pattern": [{"LOWER": o}], 'id': i}])

ruler = EntityRuler(nlp)
ruler.add_patterns(ruler_orgs.patterns)
ruler.add_patterns(ruler_persons.patterns)

nlp.add_pipe(ruler)
```

Custom Entity Rules show some interesting trends on overall sentiments and popularity of selected Persons and Organizations !

https://spacy.io/api/entityruler

Link to Code:
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/EntityDetectorV1.ipynb

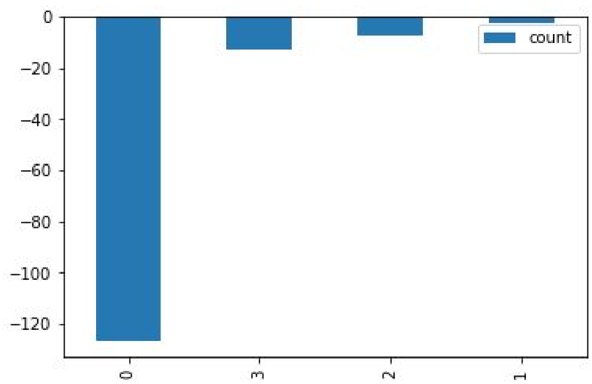
```
persons_sentiment_v2,orgs_sentiment_v2 = find_custom_entity_sentiments(pdf2.sample(800000)['text'])
```

800000/? [2:05:30<00:00, 106.24it/s]

```
dfa = pd.DataFrame(list(persons_sentiment_v2.items()),columns = ['person','count'])
dfa = dfa.sort_values('count',ascending = True)
print (dfa)
dfa.plot(kind='bar')
```

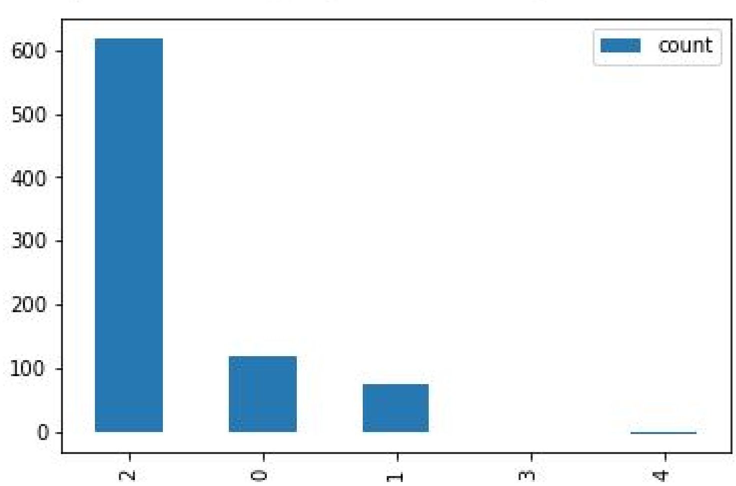
```
    person     count
0    trump -126.7111
3     birx  -12.9179
2  hancock   -7.3268
1  johnson   -2.3247
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f5444bd7290>




```
          org     count
2         who  618.7265
0  government  118.5657
1         nhs   72.9029
3         cdc   -1.8540
4         fda   -3.0997
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f5443c4da50>



We shouldn't draw any type of conclusion as this is just a random sample from a specific day's tweets ! It just shows the possibilities of different types of NER and EntityRules !

We also want to use different types of models like en_core_web_lg and en_core_web_md

Next we want to cluster the different entities based on different metrics like sentiment, followers, frequency with different statistical variations (rate_of_change , moving average, std dev etc.)

We shall create a time-series data and store it in inside a Time Series database like Elastic Search so that we can quickly perform some analysis using tools like Kibana

Analyze the Biological Named Entities in Tweet and Research Literature

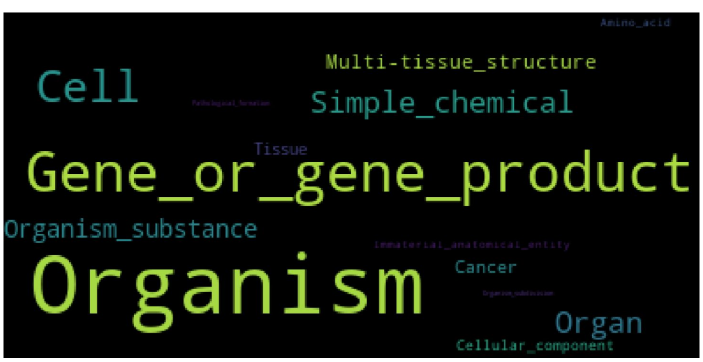
```
cord19PaperRawDF = spark.read.json("gs://covid19-papers/document_parses/pdf_json/0*",
schema=generate_schema(), multiLine=True)
cord19PaperRawDF.repartition(5).write.save("datajson.parquet")
parquetFile = spark.read.parquet("datajson.parquet")
parquetFile.createOrReplaceTempView("parquetFile")
```

Create a Clinical NER Model - Appendix B

Read pdf document

```
cord19PaperRawDF = spark.read.json("gs://covid19-papers/document_parses/pdf_json/0*",......)

embeddings = WordEmbeddingsModel.pretrained("embeddings_clinical", "en", "clinical/models")
clinical_pos = PerceptronModel.pretrained("pos_clinical", "en", "clinical/models")
bio_ner = NerDLModel.pretrained('ner_bionlp', 'en', 'clinical/models')
converter = NerConverter()
```



The Goal is develop a topic cluster from medical terms and build a recommendation model so that given a title in metadata file corresponding research papers can be suggested.

We also explored Gensim Medical Entity Extraction model and need to perform more comparison between Spark-NLP BioI NLP NER.

https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/MedicalEntityRecognition.ipynb

We shall create a Graph model to show the connection of the actual 'raw medical terms' with the clean topic i.e. the Biological Entity

Link to Code:
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/CovidLiteratureAnalysisV1.ipynb

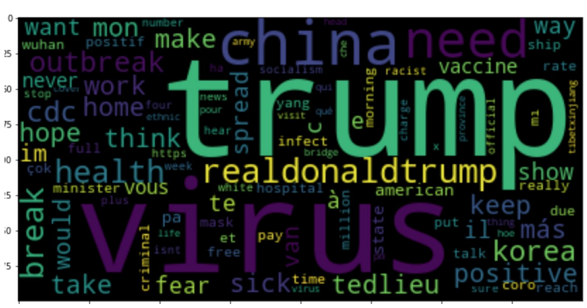
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/CovidLiteratureAnalysisV1.html

Create the Topic clusters over a period of time

- We have initially taken multiple approaches for creating clusters
  Code:
  https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/TweetTopicClusterV1.ipynb
- Final Approach: leverage Spark-NLP pretrained model
  - DocumentAssembler => Tokenizer => Normalizer => Lemmatizer => StopWordsCleaner => Finisher
- Observe that **WordClouds show some unique topics over a period of time (ignoring the common words) signifying how the crisis slowly unfolded and engulfed the society !!!**

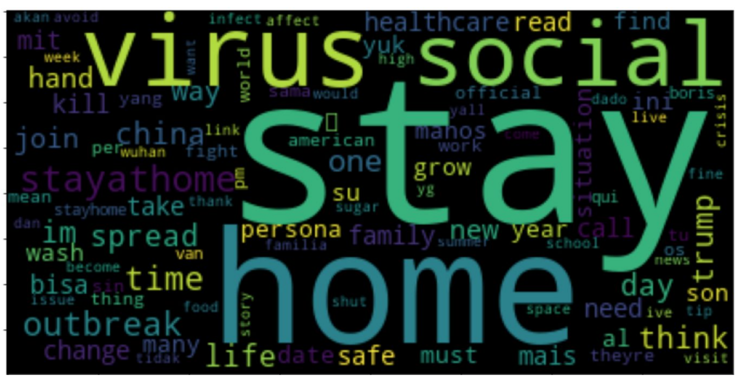
(realdonaldtrump , hopeful, korea, china , virus, spread, vaccine, work from home) ⇒ (panic , covid positive news, health)
⇒ (outbreak, cdc, stay home, hand wash, social) => (move, auction, hospitals, crisis)

March (01-09)

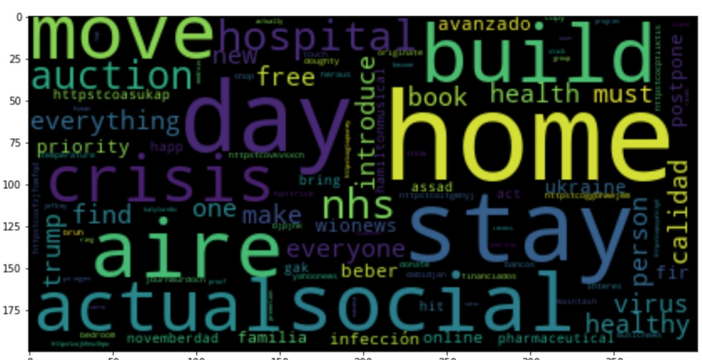


March (10-19)

March (20-29)



March (30-31)

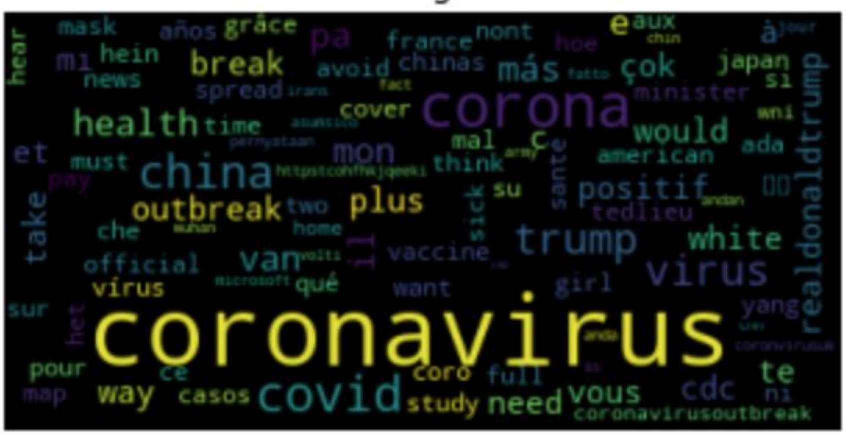


Link to Code:
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/TweetTopicClusterV1.ipynb

Analyze the sentiments associated with the Topics

High Negative

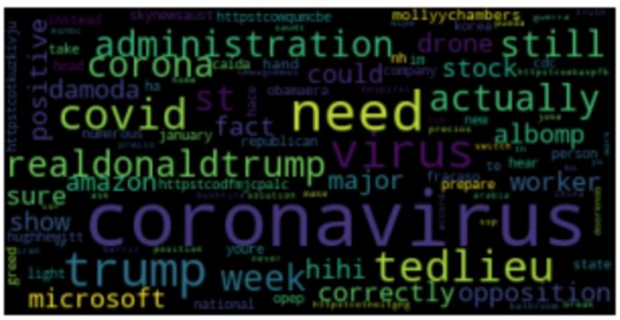
Low Negative



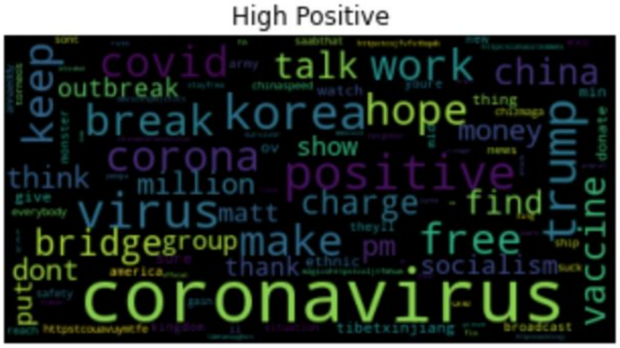
While people were certainly concerned about health , infection spread, criminal rate and outbreak, there were significant positive sentiments due to donation of millions of dollars , hope for vaccine, socialism , thanking people , amazon deliveries , recovery in korea (in the month of March)

High Positive

Low Positive

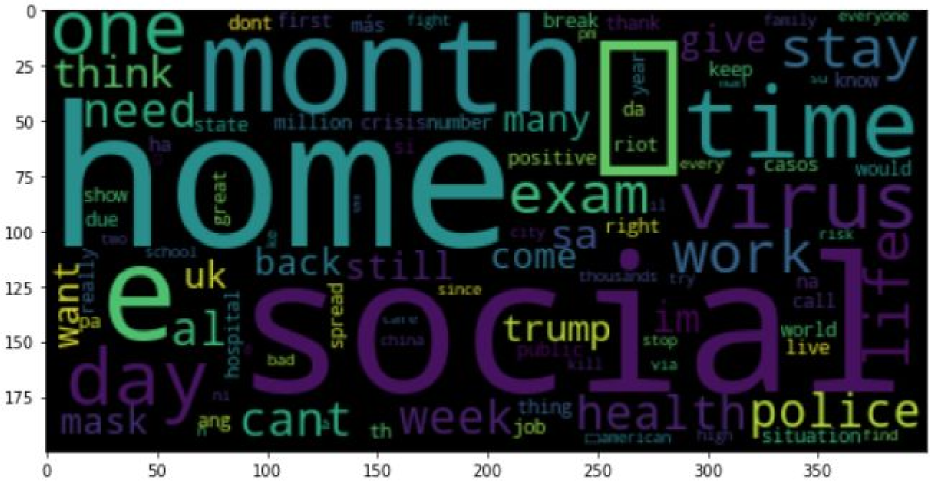


High Positive



we created the Word Cloud for June where police and riot are some of the new topics of discussions

```
# Show June WordCloud
create_show_wordcloud(6)
```

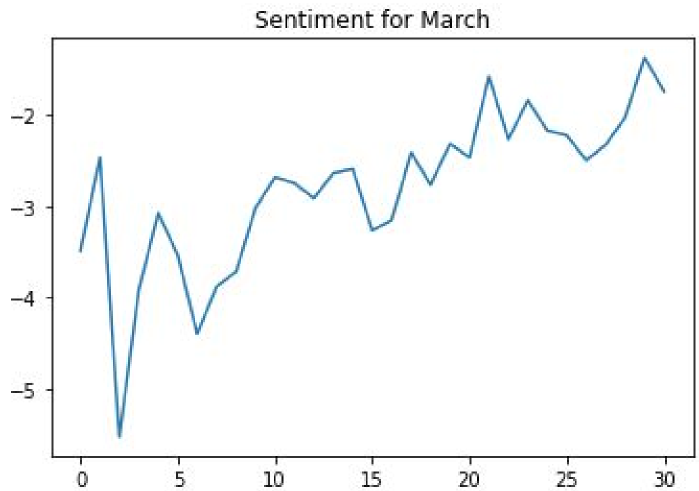


Link to Code:
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/TweetTopicClustersV2.ipynb

Analyze the Mental Anxiety Pattern

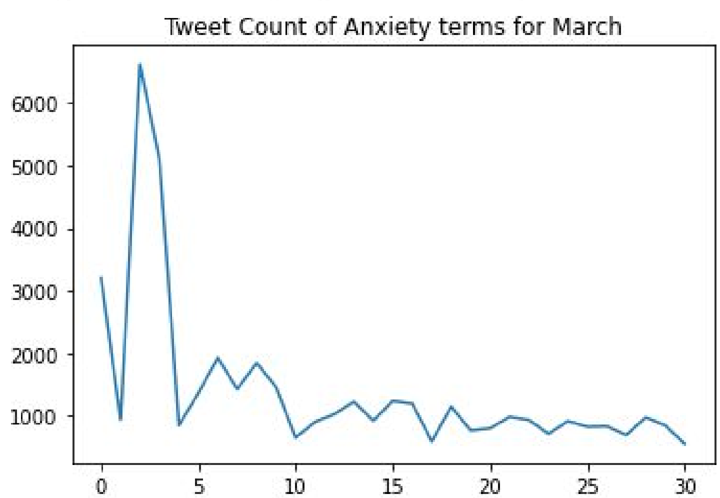
```
df.toPandas()["sentiment"].plot(title="Sentiment for March", legend=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9b02035590>
```

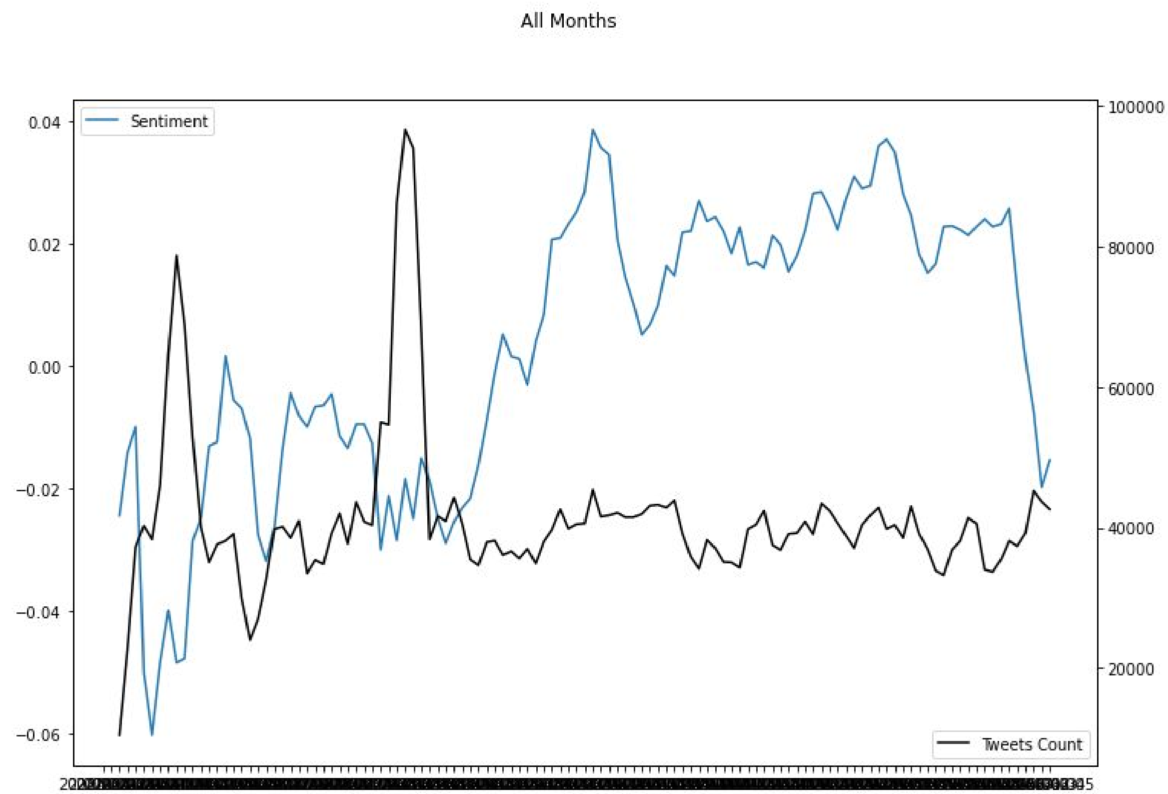



```
df.toPandas()["total_tweet"].plot(title="Tweet Count of Anxiety terms for March",
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9b028ee190>
```

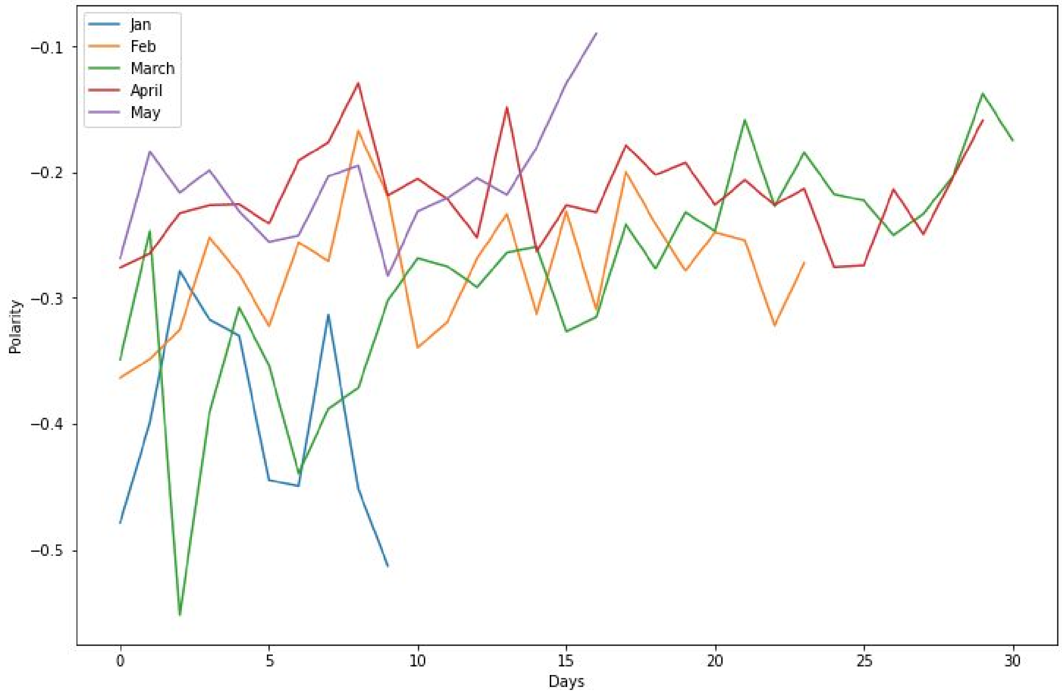


In the initial time-period of March when the lock-down started, there seems to be high concern with strong anxiety and eventually the degree of negativity reduced , but still mental anxiety prevails the entire month of March.



We observe high peaks in mental anxieties during certain time periods of Jan, Feb, March and then it improved a lot and again started showing negative trend (more mental concerns) in the month of June

```
[282]: Text(0, 0.5, 'Polarity')
```



Mental anxiety shows improvement over period of time from Jan to May as shown above in monthly distribution

Our goal is to forward uplifting tweets (e.g. #EmotionalConnection) to the tweets showing mental health crises over a prolonged period.

Link to code:
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/MentalAnxietyAnalysisV1.ipynb

https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/MentalAnxietyAnalysisV2.ipynb

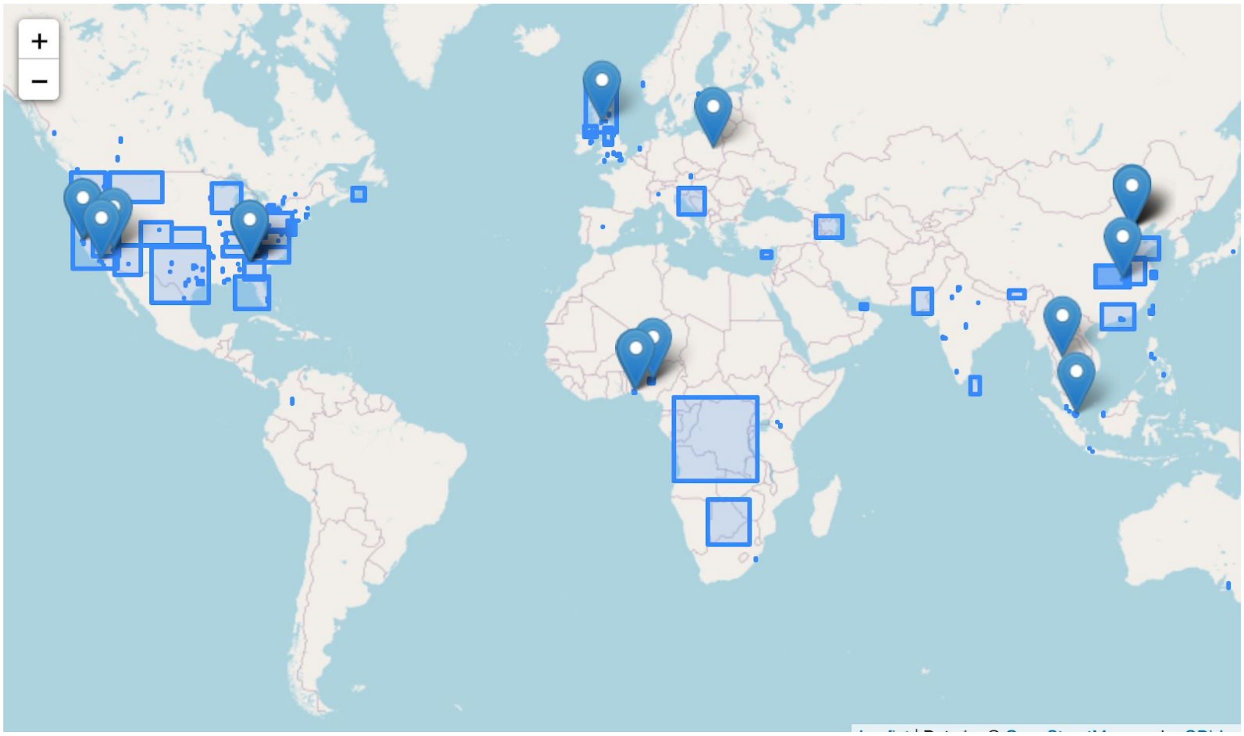
We shall Analyze Well-being of Humanity by levering DLATK (http://dlatk.wwbp.org/) which is an end to end human text analysis

Mental Anxiety terms reference: Appendix

GeoSpatial Analysis of Outbreaks

First we created the GeoJson from Tweets and then generated plots using Folium

Result of Analyzing January Data already shows that how the Virus started spreading to UK and USA from China



We want to show the actual sentiments in geo locations and correlate with infections using color codes

Link to Code:
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/GeoSpatialAnalysisV1.ipynb

https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/GeoSpatialAnalysisV2.html

For large amount of data we want to explore
https://datasystemslab.github.io/GeoSpark/tutorial/viz/

We created a Notebook but hit some issues while converting rdd to dataframe and couldn't visualize the data using GeoSpark
For reference:
https://github.com/hacking-for-humanity/COVITA/blob/master/Advanced/GeoSpatialAnalysisV2.ipynb

Challenges:

We faced challenges in terms of storing large data, processing massive volume of tweets. But we relentlessly fixed issues and leveraged multiple cloud platforms like Azure and Google and used multiple development environments like Colab Notebook , Databricks , Dataproc and Google VMs in order to store data and utilize the free compute power as much as possible.

Future Work:

- We would like to extend our work further to perform medical document recommendation, identify sensitive information, spread positive uplifting messages , predict outbreaks and create a marketplace for medical kit providers.
- We shall build a recommendation model based on the creating clusters of topics for recommending hashtags.
- We also want to use different types of models like en_core_web_lg and en_core_web_md
- Next we want to cluster the different entities based on different metrics like sentiment, followers, frequency with different statistical variations (rate_of_change , moving average, std dev etc.)
- We shall create a time-series data of the above statistical variations to detect anomaly and patterns.
- We need to create more sophisticated geospatial maps by correlating infection rate with outbreak locations

References:

Databricks

https://docs.databricks.com/data/tables.html#create-a-partitioned-table
https://docs.databricks.com/spark/latest/spark-sql/udf-python.html
https://docs.databricks.com/notebooks/notebooks-use.html

Google Cloud VM & Dataproc Cluster

https://cloud.google.com/ai-platform/notebooks/docs/create-new
https://cloud.google.com/dataproc/docs/concepts/components/jupyter

Spark

Spark-SQL Tricks

https://sparkbyexamples.com/spark/spark-sql-window-functions/
https://supergloo.com/spark-sql/spark-sql-json-examples/
https://docs.databricks.com/spark/latest/spark-sql/spark-pandas.html

Spark-Data-Munging
https://mungingdata.com/apache-spark/advanced-string-matching-with-rlike/

Spark-Multi-Processing
https://towardsdatascience.com/speeding-up-and-perfecting-your-work-using-parallel-computing-8bc2f0c073f8

https://github.com/mahmoudparsian/pyspark-algorithms/blob/master/code/chap05/rdd_transformation_mappartitions.py
https://github.com/alreadyexists/somedemos/blob/master/mapPartitions.ipynb

Spark-NLP
https://johnsnowlabs.github.io/spark-nlp-workshop/databricks/index.html#python/annotation/Spark%20NLP%20start.html

https://johnsnowlabs.github.io/spark-nlp-workshop/databricks/scala/annotation/2-%20Pre-trained%20Pipelines%20-%20onto_recognize_entities_sm.html

https://johnsnowlabs.github.io/spark-nlp-workshop/databricks/scala/annotation/1-%20Pre-trained%20Pipelines%20-%20recognize_entities_dl.html

Bio-NLP
https://github.com/JohnSnowLabs/spark-nlp-workshop/blob/master/jupyter/enterprise/healthcare/Clinical-Text-Analysis.ipynb

https://github.com/JohnSnowLabs/spark-nlp-workshop/blob/master/jupyter/enterprise/healthcare/BioNLP-NER.ipynb

Geo-Spark
https://towardsdatascience.com/interactive-geospatial-data-visualization-with-geoviews-in-python-7d5335c8efd1

Pandas

https://pbpython.com/simple-graphing-pandas.html
https://github.com/GoogleCloudDataproc/cloud-dataproc/blob/master/notebooks/python/3.1.%20Spark%20DataFrame%20%26%20Pandas%20Plotting%20-%20Python.ipynb

Appendix

Appendix-A

Install Basic Spark-NLP library

https://nlp.johnsnowlabs.com/docs/en/install#databricks

Setup Spark-NLP Licensed Version in Databricks

python3 -m pip install --upgrade spark-nlp-jsl==2.5.0 --user --extra-index-url https://pypi.johnsnowlabs.com/<KEY>

Install Libraries
spark-nlp==2.5.0
com.johnsnowlabs.nlp:spark-nlp_2.11:2.5.1

Set Configuration Setting
spark.serializer org.apache.spark.serializer.KryoSerializer
spark.kryoserializer.buffer.max 2000M
spark.databricks.delta.preview.enabled true
spark.jars.packages com.johnsnowlabs.nlp:spark-nlp_2.11:2.5.1
spark.jars https://pypi.johnsnowlabs.com/<KEY>/spark-nlp-jsl-2.5.0.jar

PYSPARK_PYTHON=/databricks/python3/bin/python3
AWS_ACCESS_KEY_ID=aaa
AWS_SECRET_ACCESS_KEY=bbb
SPARK_NLP_SECRET_KEY=ccc
secret=ddd
SPARK_NLP_LICENSE=eee

spark.conf.set("spark.jars.packages", "JohnSnowLabs:spark-nlp:2.5.0")
spark.conf.set("spark.jars","https://pypi.johnsnowlabs.com/<KEY>/spark-nlp-jsl-2.5.0.jar")
spark.conf.set("spark.jars",
"https://s3.amazonaws.com/auxdata.johnsnowlabs.com/public/spark-nlp-assembly-2.5.0.jar")

Appendix (B)

Inside Google Cloud VM , after installing pyspark copy the gcs-hadoop connector jar

gsutil cp gs://hadoop-lib/gcs/gcs-connector-hadoop2-latest.jar
/opt/conda/lib/python3.7/site-packages/pyspark/jars/

Setup Spark-NLP in Google Cloud Jupyter VM

```
license_keys = {'secret':"xyz",
'SPARK_NLP_LICENSE': 'aaa',
'JSL_OCR_LICENSE': 'bbb',
'AWS_ACCESS_KEY_ID':"ccc",
'AWS_SECRET_ACCESS_KEY':"ddd",
'JSL_OCR_SECRET':"eee"}
```

```
import os
secret = license_keys['secret']
os.environ['AWS_ACCESS_KEY_ID']= license_keys['AWS_ACCESS_KEY_ID']
os.environ['AWS_SECRET_ACCESS_KEY'] = license_keys['AWS_SECRET_ACCESS_KEY']
os.environ['SPARK_NLP_LICENSE'] = license_keys['SPARK_NLP_LICENSE']
```

```
.config("spark.driver.memory", "22G") \
        .config("spark.serializer", "org.apache.spark.serializer.KryoSerializer") \
.config("spark.kryoserializer.buffer.max", "2000M") \
.config("spark.jars.packages", "com.johnsnowlabs.nlp:spark-nlp_2.11:2.5.1") \
.config("spark.jars", "https://pypi.johnsnowlabs.com/"+secret+"/spark-nlp-jsl-2.5.0.jar") \
.config("fs.gs.impl", "com.google.cloud.hadoop.fs.gcs.GoogleHadoopFileSystem") \
.config("fs.AbstractFileSystem.gs.impl", "com.google.cloud.hadoop.fs.gcs.GoogleHadoopFS")
```

Setup Google Cloud Proc
https://cloud.google.com/dataproc/docs/tutorials/jupyter-notebook

Appendix (C)

Access Azure from Databricks

https://docs.databricks.com/data/data-sources/azure/azure-storage.html#language-python

```
spark.conf.set("fs.azure.account.key.[XYZ].blob.core.windows.net",  "ABC")
file_location =
"wasbs://<container-name>@<storage-account-name>.blob.core.windows.net/<directory-name
>"
file_type = "json"
spark.conf.set("spark.sql.files.ignoreCorruptFiles", "true")
```

```
df = spark.read.option("badRecordsPath", "/tmp/badRecords/").format(file_type).load(file_location)
```

Install Databricks CLI

```
pip install --index-url=https://pypi.python.org/simple/ --upgrade pip
pip install --index-url=https://pypi.python.org/simple/ databricks-cli
```

Install Azure CLI