Part B – Context Based Sentiment Analysis

Team Details 1B

Student	Email	Github ID	Class ID
Attaluri, Lalith Chandra	la4kf@mail.umkc.edu	LalithChandraAttaluri	4
Karumanchi, Pranitha Saroj	pkt59@mail.umkc.edu	pranithakarumanchi99	7

Goals and Objectives:

Motivation:

In this data-driven competitive environment, in many sectors, such as Telecom, Banking, Financial and Health service sectors, data handling has become vital in the decision taking process. The key factors will be to handle the humongous amounts of data and get benefits from it. One of the impressive systems, use Apache Spark, which can manage big data in real time and perform various analyses.

Objectives:

The main idea of our project is to use Spark Streaming to do the ETL process and to implement the machine learning concepts on that data in real time. Our device source is Twitter data and we'd be using Streaming Content, which is real-time data processing, and using streaming API we'd be collecting the data for a collection of specified keywords in a near-real-time process. Then we'd do the transformations on the RDD's streaming set and load the data into the Hive framework which is similar to the basic ETL method. We will also be executing the EDA on twitter data while capturing the data background. Our project would also highlight the Sentiment Analysis System where the tweets are populated with real-time sentiments. It also describes the keyword reasons for categorizing a tweet into positive or negative sentiment.

Significance:

For sentiment analysis, we use an predefined ML (TextBlob) method to predict sentiment of the tweets, and based on that, we will analyze keywords that are critical to sentiment prediction, and we will train a new ML model with that analysis to predict tweet feeling that will improve prediction accuracy.

Features:

The characteristics of the project include collecting real-time tweets from twitter streaming APIs, performing ETL (pre-processing data, extracting required information and loading data into the Hive), and using TextBlob to predict the sentiment for each tweet and thereby train a new ML model with tweets as input and type of sentiment as output.

Description of Dataset:

We use twitter streaming API to capture tweets on a particular collection of keywords, almost in real time. Each tweet is in JSON format, which is pairs with key value. It has different details about a tweet such as tweet text, who tweeted it, user information, location where the tweet originated, tweet source such as Android, Iphone etc. Using Spark, we capture real time data from streaming API The streaming background with a 20-second window and the streaming API was able to download 500 kb of data from which we search keywords such as **Coronavirus, COVID-19, Pandemic, COVID19, covid19, covid-19, coronavirus, pandemic** and extract nearly 80 kb from that.

Design Flow:

We originally built an account in the Twitter Developers API. Using the Spark Streaming program, we downloaded the tweets from the provided API tokens and credentials. Our project has a sort of client / server model where we got tweets using our application's Tweepy PY library that served as the server level. Now we used the program Spark Streaming to submit the request and, the application will receive the server tweets on a window-based model upon completion.

If the customer receives the tweets on a window-based setup, then two operations will be performed. First, we'd show the feeling on fly for any tweet that's streamed on the stream info. Second, after carrying out the series of transformations on the streaming tweets, we will be storing the tweets into the HIVE system. Having saved the data in HIVE, by training on stored tweets we would be building a classification model. If the model is trained then we'd run the model on tweets in real time and predict the feeling.

The next stage of our project is to compare the scores on the model and the direct sentiment scores on the tweets. This also describes the keyword reasons for categorizing a tweet into positive or negative sentiment.

Step by Step Implementation of the design:

Data Extraction:

We have created a twitter developer account for downloading twitter info, and have used tweepy python library and spark streaming background to download tweets. Every tweet is a raw json data, which also has complex json formats, we pre-processed the data and converted

the complex json to a simple text format that can be managed and sent by socket to the client side system.

We transformed them from the client side into a table structure where we could write hive queries for some research. Results of the study are briefly clarified in the preliminary report.

Server Client Implementation:

We adopted the client-serve approach to accessing the streaming twitter tweets. On the server side, we used tweepy library to connect to the twitter streaming API with the appropriate credentials that we got from the twitter developer account. To pass data from server to client we used socket link. In browser, the data we received from streaming twitter.

The API is in JSON format, which we have pre-processed on the server side, and sends simple text data to the client side with appropriate fields. The tweepy will start to stream twitter data and put the data in the socket.

On the client side, we used spark streaming context to collect data from the socket, configuring the spark streaming context with a window length of 20 time units and a sliding interval of 20 seconds, which means that the next 20 windows of data will be read by the spark reader for every 20 seconds.

Data Pre-processing:

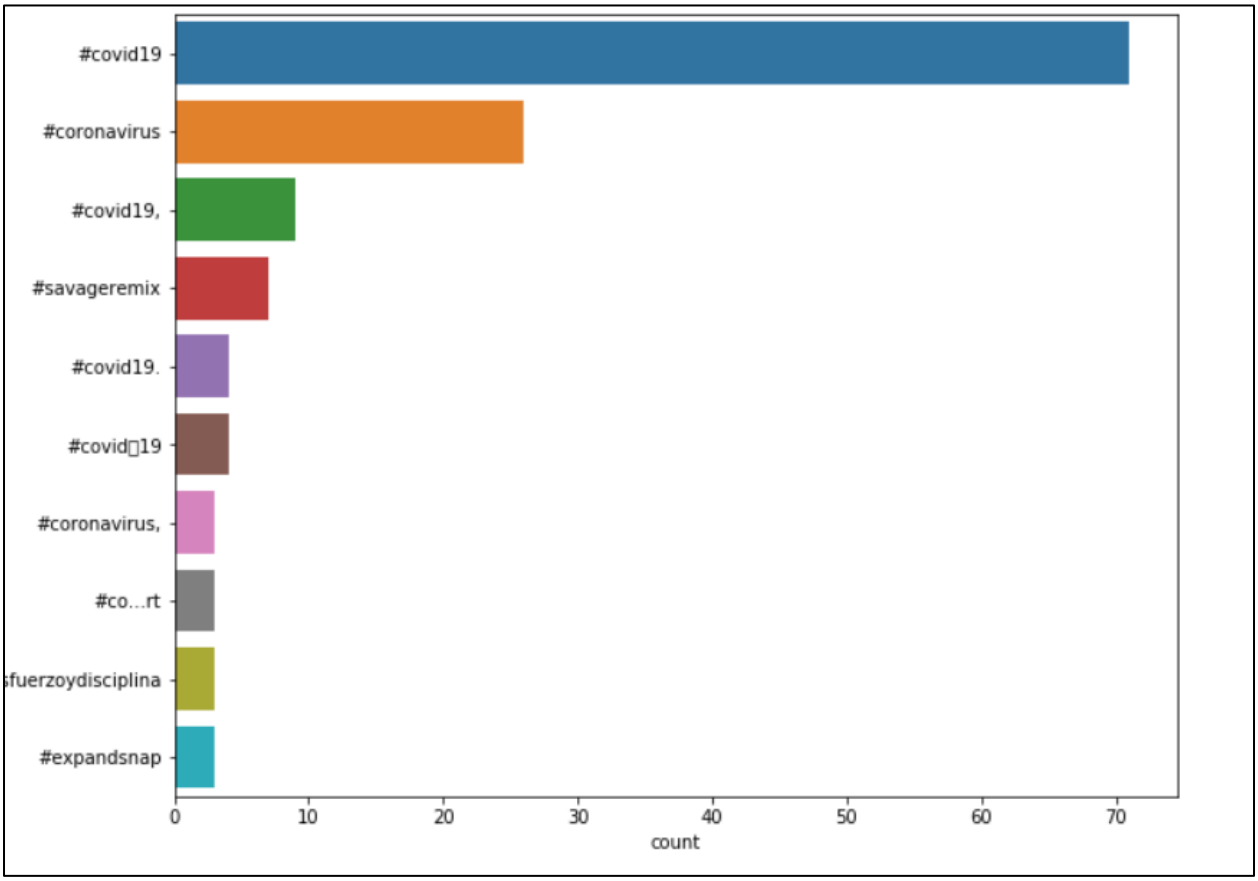
If the client observes the data, we apply map and lower it to turn the data into table form and store it in a hive database. It's easier to write Spark SQL queries from the hive, and to do some data analysis.

Data Visualization:

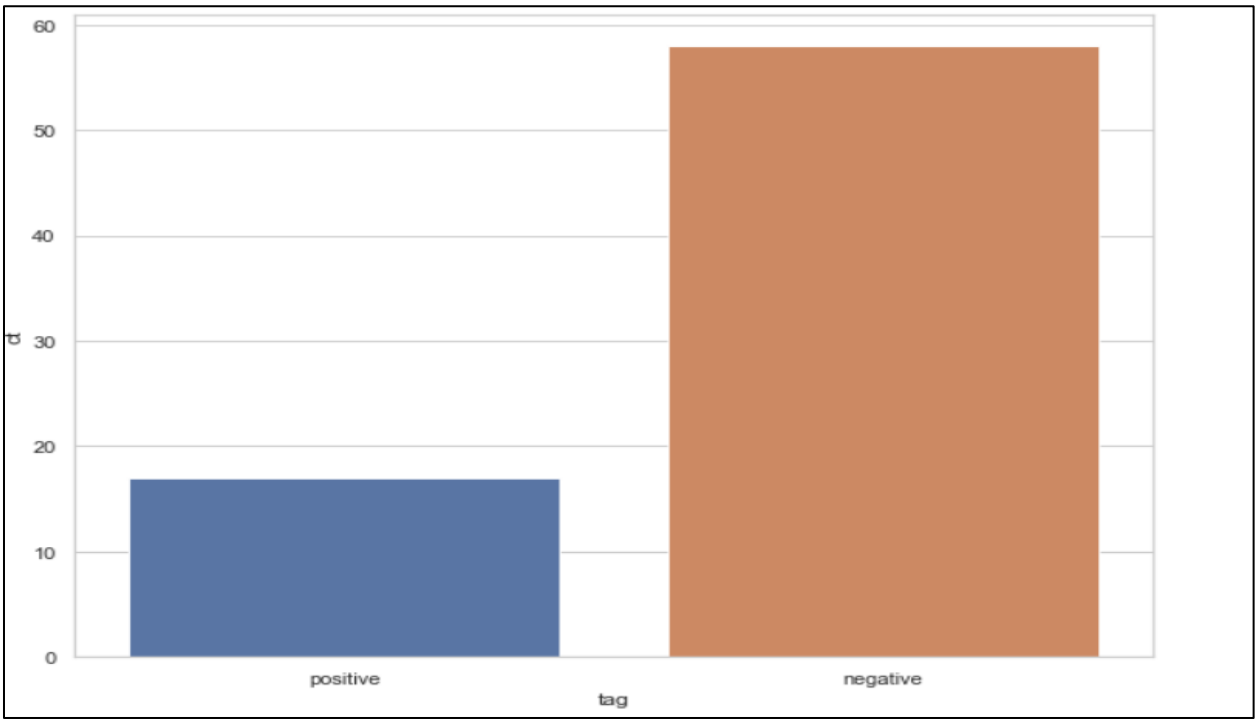
- **Most Widely used URL's:** The below figure shows the distinct URL's present in the time frame

```
+-------------------------------------------------+---+
|url                                              |ct |
+-------------------------------------------------+---+
|https://t.co/kq6pjltosert                        |1  |
|https://t.co/kzvxb8svpd                          |1  |
|https://t.co/awn5ffzyzyel                        |1  |
|https://t.co/toyotopm39rt                        |1  |
|https://t.co/oruil2ueidrt                        |1  |
|https://t.co/c9czei681vhttps://t.co/r9vwnanp4trt|1  |
|https://t.co/kwzortp3q1rt                        |1  |
|https://t.co/fcgg0o5oxbrt                        |1  |
|https://t.co/jy8jtyqftart                        |1  |
|https://t.co/dvuzqxewbr@maddow                   |1  |
+-------------------------------------------------+---+
```

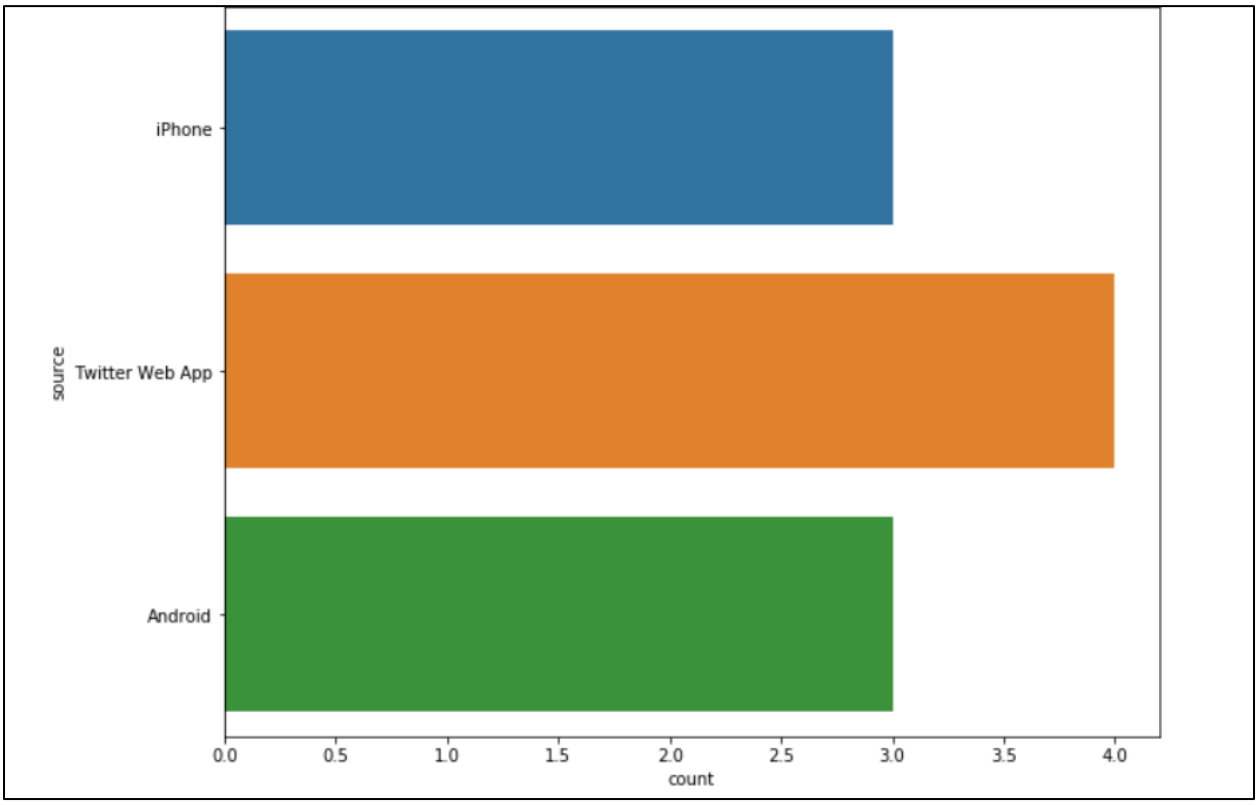
- **Top 10 hashtags:** The below figure shows the top 10 hashtags used in the particular timeframe



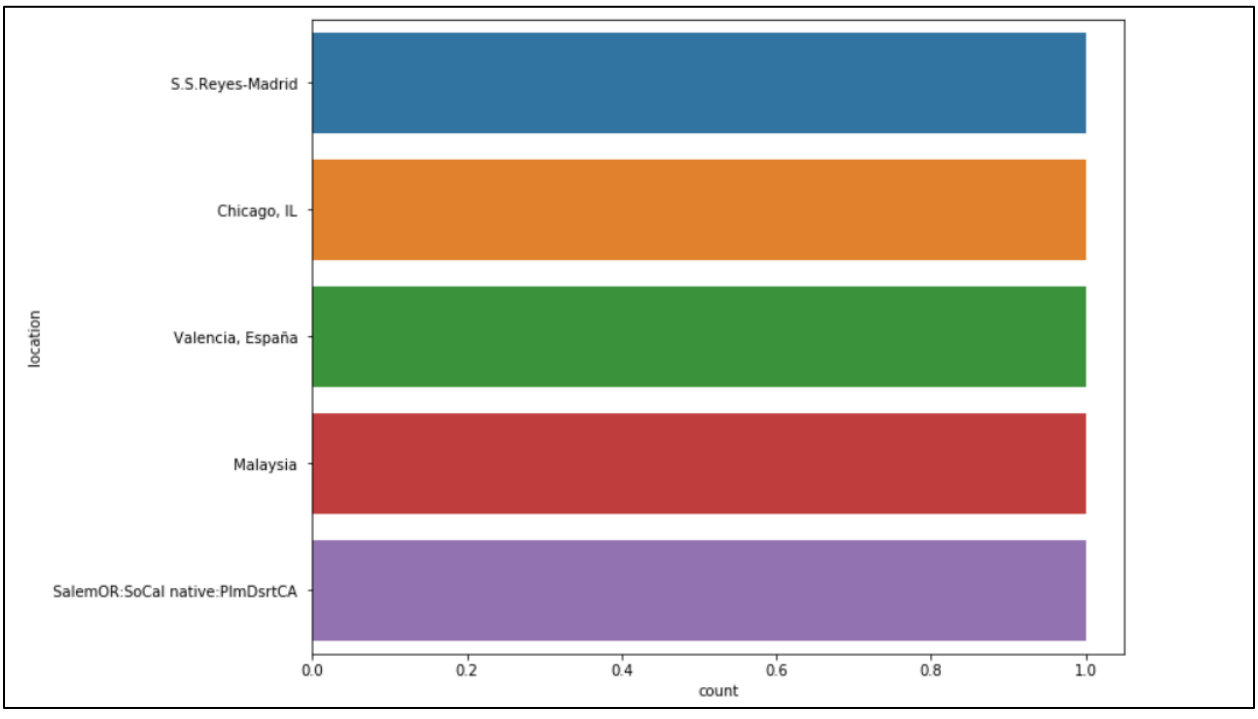
- **Positive vs Negative tweets:** As the tweets we have collected are based on Covid-19 pandemic negative tweets are higher than positive tweets.



- **Tweets from different sources:**

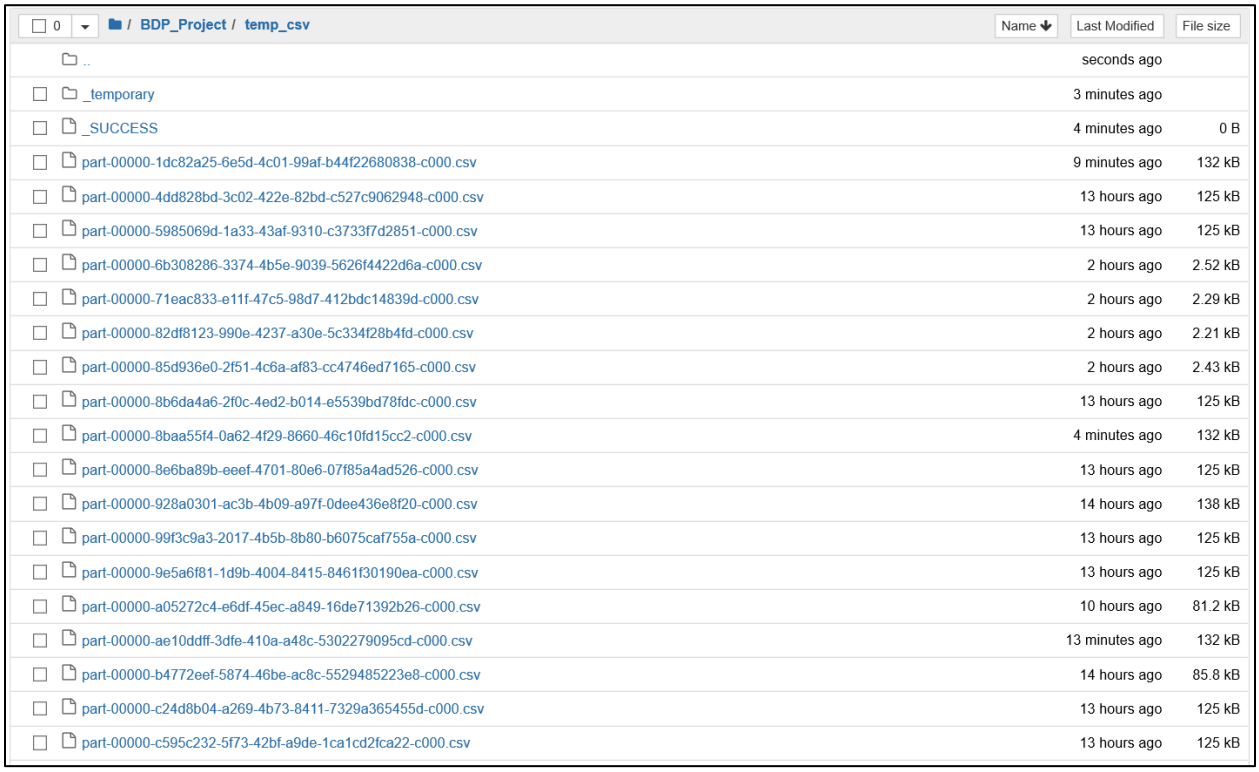


- **Tweets from different locations:**



Saving and loading the streamed data:

Using Spark Streaming Background, we gather the tweets every 20 seconds and save them to a local folder with the following options: coalesce=1, which will store the data in a single file; headers = true, store the file with headers. After we have collected enough tweets to train a model, we merge all the csv files using Globalize and load the merged file into the Spark SQL sense, and create external hive tables. We've gathered about 50k tweets. We used TextBlob on this data to predict feelings for each tweet and with this data we are training a machine learning model to predict feelings about the meaning of sports. Below are the csv files that are saved.



0 / BDP_Project / temp_csv

	Name	Last Modified	File size
..		seconds ago	
_temporary		3 minutes ago	
_SUCCESS		4 minutes ago	0 B
part-00000-1dc82a25-6e5d-4c01-99af-b44f22680838-c000.csv		9 minutes ago	132 kB
part-00000-4dd828bd-3c02-422e-82bd-c527c9062948-c000.csv		13 hours ago	125 kB
part-00000-5985069d-1a33-43af-9310-c3733f7d2851-c000.csv		13 hours ago	125 kB
part-00000-6b308286-3374-4b5e-9039-5626f4422d6a-c000.csv		2 hours ago	2.52 kB
part-00000-71eac833-e11f-47c5-98d7-412bdc14839d-c000.csv		2 hours ago	2.29 kB
part-00000-82df8123-990e-4237-a30e-5c334f28b4fd-c000.csv		2 hours ago	2.21 kB
part-00000-85d936e0-2f51-4c6a-af83-cc4746ed7165-c000.csv		2 hours ago	2.43 kB
part-00000-8b6da4a6-2f0c-4ed2-b014-e5539bd78fdc-c000.csv		13 hours ago	125 kB
part-00000-8baa55f4-0a62-4f29-8660-46c10fd15cc2-c000.csv		4 minutes ago	132 kB
part-00000-8e6ba89b-eeef-4701-80e6-07f85a4ad526-c000.csv		13 hours ago	125 kB
part-00000-928a0301-ac3b-4b09-a97f-0dee436e8f20-c000.csv		14 hours ago	138 kB
part-00000-99f3c9a3-2017-4b5b-8b80-b6075caf755a-c000.csv		13 hours ago	125 kB
part-00000-9e5a6f81-1d9b-4004-8415-8461f30190ea-c000.csv		13 hours ago	125 kB
part-00000-a05272c4-e6df-45ec-a849-16de71392b26-c000.csv		10 hours ago	81.2 kB
part-00000-ae10ddff-3dfe-410a-a48c-5302279095cd-c000.csv		13 minutes ago	132 kB
part-00000-b4772eef-5874-46be-ac8c-5529485223e8-c000.csv		14 hours ago	85.8 kB
part-00000-c24d8b04-a269-4b73-8411-7329a365455d-c000.csv		13 hours ago	125 kB
part-00000-c595c232-5f73-42bf-a9de-1ca1cd2fca22-c000.csv		13 hours ago	125 kB

Model Training:

To train our ML sentiment model we followed the below procedure

- Data Processing
- Initialization of model variables / Data division / Fit and Validation of mode

Data Processing:

- Using the GLOB library we loaded all the files into one single DF.
- Because tweets are free text, we only tested the percentage of non-alpha tweets that was about 6%.

- We used a feature to clean up the tweets. As we feed the tweets into a model of ML. We need the needless words removed from the tweets. We have implemented the approach using regex.

- Used the Textblob model in -order to obtain the tweets' polarity scores and allocated a new column in the pandas dataframe
- Created 2 new columns to obtain the Sentiment score and definition that is described based on the polarity score. If < = 0 we tagged it to the negative feeling and scored it to 0, while if > 0 we tagged it to the positive feeling and scored it to 1

Initialization of model variables / Data division / Fit and Validation of mode:

- We have divided the data into train and test splits using the test train split
- Initialized the TFIDF vector and performed the fit and transformed the train data in a single step.
- We have train features and test features separately in various TFIDF variables
- Logistic Regression, Decision Tree Classification, Random Forest Classification and Gradient initialized
- Generated a pipeline structure for each model for respective hyperparameters
- Now we fit the TFIDF train features for each classifier along with the positive and negative sentiment definition
- Next we predicted the test features of the ML models
- All models observed the accuracy.
- Now we have downloaded the code TFIDF and ML(LR) using the jsonl

Sentiment Comparison Model (User Defined Functions):

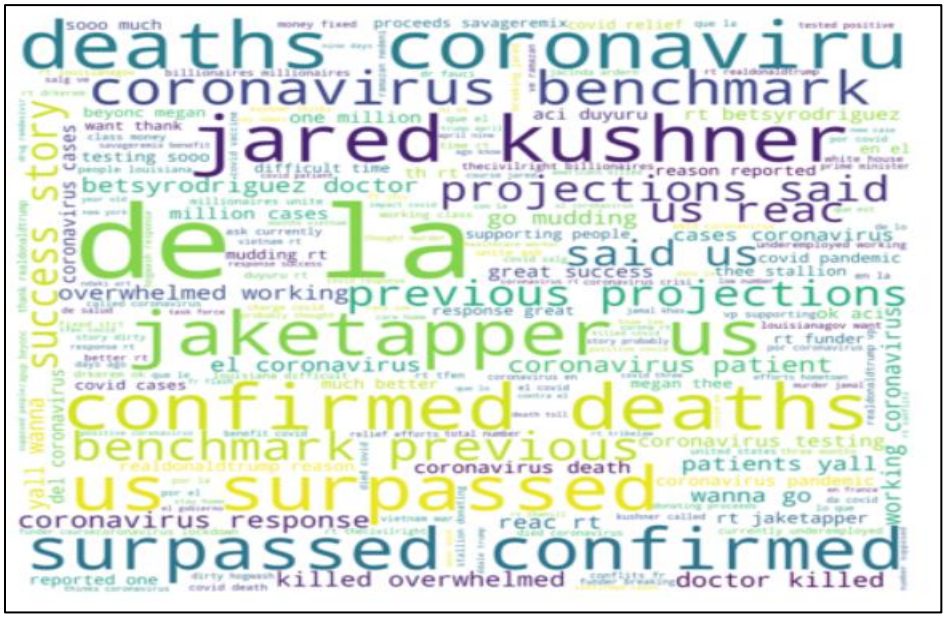
3 User Defined Functions have been created to serve the purpose

- **Data Cleaning** – Removing junk characters such as smileys, other language characters, etc. that that affect the output of the ML model being trained.
- **TextBlob Prediction** – This UDF will take the parameter of tweeting and call the TextBlob feature with the tweet, while the TextBlob will give the tweet a polarity of feel. We defined a threshold of 0 i.e. When polarity is less than 0, the feeling would be 'Negative' then 'Positive.'
- **Model Prediction** – We store the trained ML model in a folder using jsonlib after we have trained an ML model. In this UDF we load the model using jsonlib and predict feeling for a tweet and give the feeling back.

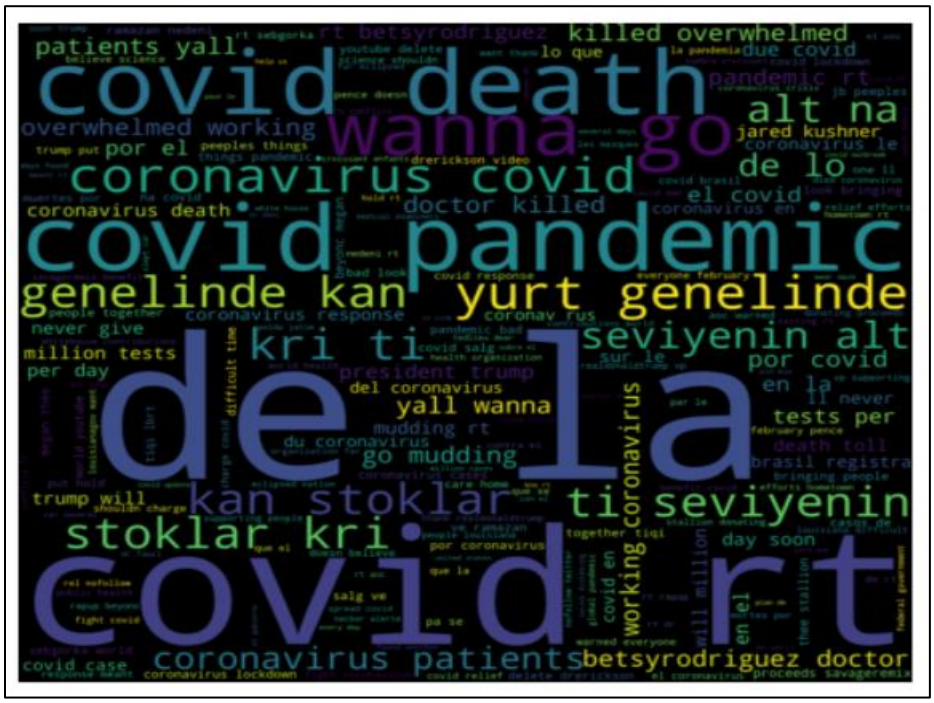
Stop Words:

```
stopwords.add('co')
stopwords.add('https')
stopwords.add('hey')
stopwords.add('hello')
stopwords.add('school')
```

Positive Words:



Negative Words:



Words with more importance:

	Word	Importance
17154	rt	0.032737
8915	https	0.021289
4067	coronavirus	0.018957
13310	new	0.015749
4186	covid	0.008859
3793	confirmed	0.008136
6164	en	0.008105
2868	cases	0.008035
11351	live	0.006637
14353	pandemic	0.006565
15184	positive	0.006190
10353	just	0.004882
4686	deaths	0.004159
5808	earth	0.004084
19001	surpassed	0.004070
15953	que	0.003942
8156	great	0.003768
9252	important	0.003670
1995	best	0.003613
6199	energy	0.003557

Words with no importance:

	Word	Importance
0	aa	0.0
1	aad	0.0
3	aafaizli	0.0
7	aaionwgdte	0.0
8	aajkamrankhanrt	0.0
...	...	...
22204	zxq	0.0
22208	zyyoif	0.0
22209	zyywanedy	0.0
22210	zz	0.0
22213	zzoeiqdmoh	0.0

Project Management:

Lalith Chandra Attaluri(50%):

- Server Implementation
- Data Visualisation: Tweets from different sources, Tweets from different counties
- Data Pre processing & UDF's creation
- Building the model
- Documentation

Pranitha Karumanchi(50%):

- Client Implementation
- Data Visualisation: Top 10 hash tags & Top 10 URL's
- Training and Testing the model
- Word Cloud Implementation

Work to be completed:

- Predicting the score by training the model with real time tweets
- Comparing the accuracy and differences between UDF and Custom model.

Remaining work percentage (15%)