Final Project Report: COVID-19 Analysis

ITCS 6155 - Knowledge-Based Systems - (University of North Carolina at Charlotte)

Github Link:

https://github.com/mkrish14/itcs6155_group14

App Engine:

https://kbs-covid.ue.r.appspot.com/

Team Members (Project Group 14):

- Aniruddha Sudhindra Shirahatti
- Bharadwaj Aryasomayajula
- Manoj Krishna Mohan
- Ravi Teja Kolla
- Sai Kumar Thallada

Introduction:

COVID 19 Pandemic: Situation in the US

Different parts of the country are seeing different levels of COVID-19 activity. The United States nationally is in the acceleration phase of the pandemic. The duration and severity of each pandemic phase can vary depending on the characteristics of the virus and the public health response.

- CDC and state and local public health laboratories are testing for the virus that causes COVID-19. For more details, view CDC's Public Health Laboratory Testing map.
- All 50 states have reported cases of COVID-19 to CDC.
- U.S. COVID-19 cases include:
    - People who were infected while travelling, before returning to the United States
    - People who were infected after having close contact with someone known to be infected with the virus
    - People in a community who were infected with the virus but don't know how or where they were infected
- All U.S. states are reporting community spread of COVID-19.

With Coronavirus on everyone's mind and forcing almost all of us indoors, many in the ML community are wondering how they might help. While there have been other articles on fighting coronavirus with AI, few have offered a truly comprehensive view. Therefore, we decided to import a COVID-19 dataset and use cases of machine learning applied to coronavirus. We believe that machine learning and data analytics can help accelerate solutions and minimize the impacts of the virus in conjunction with all the other great research and planning going.

Research Question:

With the upcoming rise in the global pandemic, we try to predict the upcoming number of cases which could be confirmed for a specific date. We also plan to predict the region across which the number of cases can increase. With our knowledge, we take on a challenging problem to solve a global case of emergency.

Major Steps Involved In Development:

- Data Collection
- Data Preprocessing
- Exploratory Data Analysis
- Data Modeling and Evaluation

Domain and Data:

The domain of the data is Healthcare or more precisely diseases classification. The dataset contains cumulative and non-cumulative count of confirmed, death and recovered global cases of COVID-19 (upto March 14, 2020), we expect the data to be populated again. The dataset consists of 500,000 rows approximately and 11 columns for each CSV:

1. Location
2. County
3. Ship
4. Case_type
5. Cases
6. Country
7. Date
8. Difference
9. Latitude
10. Longitude
11. state

Since the dataset we have taken contains the global data of the pandemic COVID 19 affected, we have filtered the data based on the Country and have taken into account the cases only in the United States.

We have also used US Air Quality data to check if there is any relation between AQI and COVID-19 cases. The dataset contains following columns:

- State
- County
- Date
- Ozone

Links to dataset:

- https://www.tableau.com/covid-19-coronavirus-data-resources
- https://www.airnow.gov/state/?name=alabama

Data Preprocessing:

We have done preprocessing and included as a Jupyter Notebook which can be viewed in the GitHub site URL provided: https://github.com/mkrish14/itcs6155_group14/blob/master/Deliverable_3/deliverable3.ipynb

- We dropped a few columns which are least correlated with the predictor variables and had only kept those columns which actually contribute in determining the target and help in the prediction analysis.
- We have renamed a few columns for our convenience

- Since the column by name "County" had a preceding label "|County" we cleaned the data by removing the preceding labels.
- Since the date column was a string, we converted the entire column to a Pandas DateTime object.
- We performed label encoding on the categorical data type objects to convert them into int data type.
- As a last step of preprocessing we converted all the NaN values of the "difference" columns to zeroes.

Exploratory Data Analysis:

We have performed the exploratory data analysis by creating dashboards and gaining insights necessary for understanding the data using Google Data Studio.

Screenshots are maintained on the Github.
https://github.com/mkrish14/itcs6155_group14/blob/master/Deliverable_3/ExploratoryDataAnalysis.pdf

The data for plotting is stored in the Google Cloud Storage Bucket, which is later on imported to the Google DataStudio.
We have plotted three different plot to visualize the data:

- Line plot (along with a trendline which is exponential). We plotted a line plot against the cases vs Data which depicts an exponential growth of the increase in number of COVID-19 cases. From the graph, it can be inferred that there was a spike/increase in the number of positive cases in the 3rd week of March following an exponential trend line
- A GeoPlot/GeoMap Plot which indicates which regions of the United States are affected more. We plotted a geo Map which shows the maximum affected states in the United States. This is the contrast of the red color in the geo map. From the graph, it is clearly evident how the states of the US are affected based on the color contrast. As we can see states like North Carolina, Wisconsin, Arizona, Alabama to be in light contrast when compared to mid range affected states like California, Florida and Washington. New York is the most affected state with no other state matching the color contrast.
- A Pie Chart which shows how much percentage of People with Confirmed Active COVID-19 Cases are present and number of Deaths in the US. From the chart, it can be inferred that 97.9% (https://www.worldometers.info/coronavirus/country/us/) are the confirmed cases and the rest 2.1 % are the total number of deaths occurring across the nation.

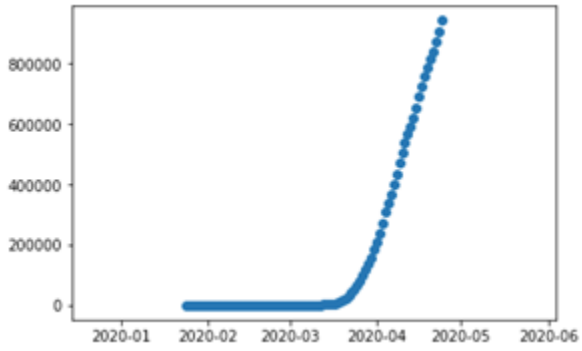
Retrieval of Air quality index data for all the states in Unites States

The recent studies determine that a person residing for decades in a county with high stages of best particulate count number is 15 percent more likely to die from the coronavirus than someone in a region with one unit less of the pleasant particulate pollutants.

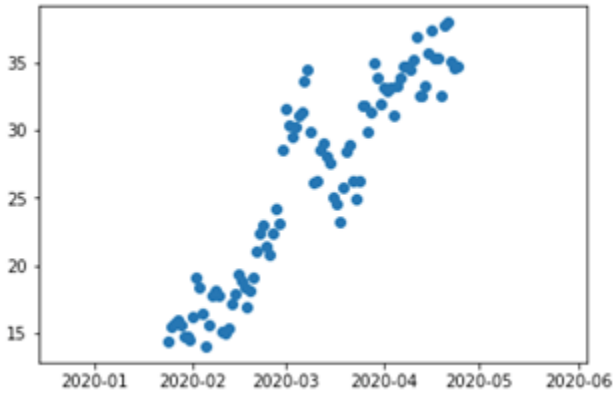
The COVID-19 lockdown has prompted cleaner air, however, will do little to address the issue of air contamination over the long haul. Individuals living with poor air quality might be increasingly defenseless to this infection, and airborne particulate matter may assist with spreading the infection. It is determined that a person residing for decades in a county with high stages of best particulate count is 15 percent more likely to die from the coronavirus than someone in a region with one unit less of the pleasant particulate pollutants.

We have made an attempt to analyze the Air Quality index and use it for the prediction of COVID-19 for all the states in the US. For this, we referred to the data corresponding to the air quality index provided by AirNow.gov - Home of the U.S. Air Quality Index.

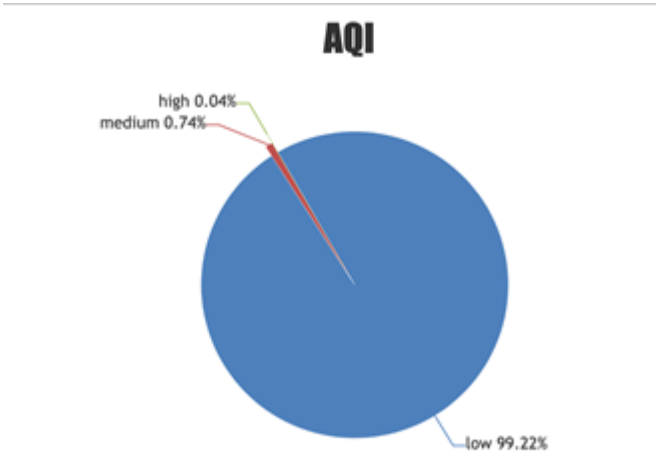
The website provides a detailed description of all the historical data of Air quality index segregated according to all the states with their respective counties. We managed to get the data for all the states from January 24th , 2020 to April 24th ,2020 by developing a Python script which retrieved the data from the same source from where the website is getting the data from since there was no access to the AirNow.gov API. Since we observed that there were a few missing values for the County's , Date and AQI data we replaced all the missing values with NaN. The obtained dataset has been merged with the COVID-19 dataset grouping it by Date and State. After merging the data, we had dropped the rows in the "Date" column from the date range of 22nd January 2020 and 23rd January, 2020. We grouped the total number of cases on each date from January 24th to April 24th, 2020 and plotted it in a scatter plot with Date ranges in the X-axis and the total number of cases on the Y-Axis.



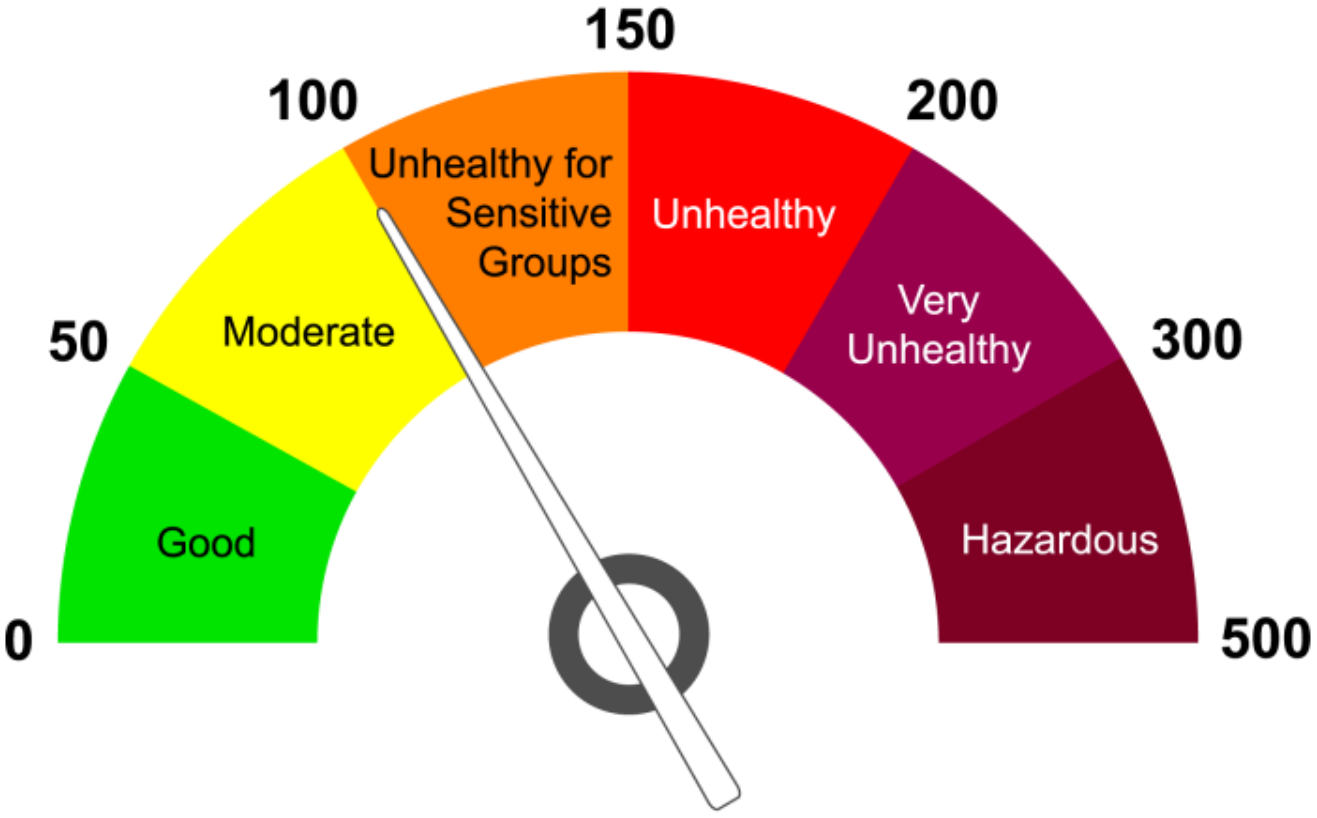
From the graph, it can be observed there was an exponential trend in the total number of cases in the above time period. We made an attempt to group the states based on three different categories of AQI. For this, we referred to the EPA index provided by Airnow which gives as a metric for reporting the air quality. According to this, the higher the AQI value, the greater the level of air pollution and the greater the health concern. For example, an AQI value of 50 or below represents good air quality, while an AQI value over 300 represents hazardous air quality. For each pollutant an AQI value of 100 generally corresponds to an ambient air concentration that equals the level of the short-term national ambient air quality standard for protection of public health. AQI values at or below 100 are generally thought of as satisfactory. When AQI values are above 100, air quality is unhealthy: at first for certain sensitive groups of people, then for everyone as AQI values get higher. The AQI is divided into six categories. Each category corresponds to a different level of health concern. For our convenience, we had binned the AQI index into a total of three different categories namely Low, Medium and High. We combined all the air quality indexes in the range of 0 to 51 into Low Air quality index, 51 to 101 as "Medium Air Quality index" and 101 to 150 as "High Air Quality Index". We plotted a scatter plot by grouping all the date ranges with respect to the average of the air quality index.



We had encoded all the state names to get the numerical representation by identifying distinct values for each state. We converted the date value to Unix timestamp for giving it as an input feature to the model. Below is a pie chart which categorizes the AQI values into three different categories. As most of the states in the US are in the low-quality index it is difficult to implement a ML on the Air Quality Index data.



As most of the states in the US are in the low-quality index it is difficult to implement a ML on the Air Quality Index data since it is difficult to predict the number of cases based on the AQI



Data Modeling and Evaluation:

We are working with the time series data. ‘A time series is simply a series of data points ordered in time’. In this time series data, time is the independent variable and our goal is to make a forecast for the number of COVID-19 cases on a given date in the future.

We started with the simple linear regression model. A linear regression model is a linear approximation of the relationship between two or more variables. With the help of the sample data we came up with the model that explains the data and we make predictions based on the model we have developed.

Then, we converted the regular date format to a unix timestamp as cannot use the regular date format as an input to the linear regression model. We separated data for each state so that we can perform state wise predictions on the date. As the growth in the number of cases is exponential, we converted the data into logarithmic form before training the linear regression model. We used sklearn FunctionTransformer for taking the log of frequencies.

The next step was to input this data to the linear regression model and try fitting the exponential curve with the help of historical data. We created a python function ‘modeller’ which takes the state name as input, trains the linear model, predicts the results, and outputs the R2 score. R2 score is one of the evaluation metrics that helps us determine how well the model fits our data. R2 score is always between 0 and 100%.

- 0% indicates that the model explains none of the variability of the response data around its mean.
- 100% indicates that the model explains all the variability of the response data around its mean.

In general, the higher the R-squared, the better the model fits your data.

Curve fitting for Exponential Data using LinearRegressor:

We trained the models separately for each state and tried to best fit a line to this exponential data. Post training the linear regression model, we plotted the graph to observe the fit. We observed that the R2 scores for the trained models were not that great to make the best predictions on the unseen data.

In [332]:
```
index=0
for states,dataframe in allstates_df.items():
    index+=1
    print(index,'The r^2 score for ',states,' is:',modeller(states))
```

```
1 The r^2 score for  Alabama  is: 34.92678066934507
2 The r^2 score for  Arizona  is: 34.76525293539214
3 The r^2 score for  Arkansas  is: 32.486538464318016
4 The r^2 score for  California  is: 36.091938071093914
5 The r^2 score for  Colorado  is: 36.46952843479563
6 The r^2 score for  Connecticut  is: 29.076234234854237
7 The r^2 score for  Delaware  is: 26.930324055059906
8 The r^2 score for  Florida  is: 35.45357457159639
9 The r^2 score for  Georgia  is: 33.229996970646205
10 The r^2 score for  Hawaii  is: 37.673954807382025
11 The r^2 score for  Idaho  is: 36.58141531198973
12 The r^2 score for  Illinois  is: 31.795656641994064
13 The r^2 score for  Indiana  is: 31.954838557769726
14 The r^2 score for  Kansas  is: 30.881422177295526
15 The r^2 score for  Kentucky  is: 28.821018554167523
16 The r^2 score for  Louisiana  is: 36.52389723028273
17 The r^2 score for  Maine  is: 38.99470610798763
18 The r^2 score for  Maryland  is: 28.077516177110006
19 The r^2 score for  Massachusetts  is: 30.074441975504417
20 The r^2 score for  Michigan  is: 35.54514195476395
21 The r^2 score for  Minnesota  is: 34.2883810482971
22 The r^2 score for  Mississippi  is: 33.2174116141904 96
23 The r^2 score for  Missouri  is: 33.83628734277641
24 The r^2 score for  Montana  is: 39.884394705027134
25 The r^2 score for  Nebraska  is: 24.12154347941968
26 The r^2 score for  Nevada  is: 36.17963590113635
27 The r^2 score for  New Hampshire  is: 33.734877362601 37
28 The r^2 score for  New Jersey  is: 31.958129944583913
29 The r^2 score for  New Mexico  is: 29.101554980902055
30 The r^2 score for  New York  is: 38.51125426221462
31 The r^2 score for  North Carolina  is: 34.09412971012262
32 The r^2 score for  North Dakota  is: 28.975779934363356
33 The r^2 score for  Ohio  is: 30.106716352362994
34 The r^2 score for  Oklahoma  is: 34.2139834737696
35 The r^2 score for  Oregon  is: 43.36982361095225
36 The r^2 score for  Pennsylvania  is: 30.568132409378237
37 The r^2 score for  Rhode Island  is: 20.487727041029657
38 The r^2 score for  South Carolina  is: 36.5916852963742 44
39 The r^2 score for  South Dakota  is: 22.05385086729608
40 The r^2 score for  Tennessee  is: 37.30917421830229
41 The r^2 score for  Texas  is: 32.82883487704633
42 The r^2 score for  Utah  is: 36.7540573614515
43 The r^2 score for  Vermont  is: 39.11314134674213
44 The r^2 score for  Virginia  is: 28.52169688576829
45 The r^2 score for  Washington  is: 49.18440422913355
46 The r^2 score for  West Virginia  is: 31.49516316165315
47 The r^2 score for  Wisconsin  is: 39.28431805411125
48 The r^2 score for  Wyoming  is: 34.21124478436561
49 The r^2 score for  Alaska  is: 40.00167543804618
50 The r^2 score for  Iowa  is: 54.577260181141106
```

Figure 1. As we can observe from the R2 scores, the trained models have more loss due to the almost asymptotic exponentially increasing data.

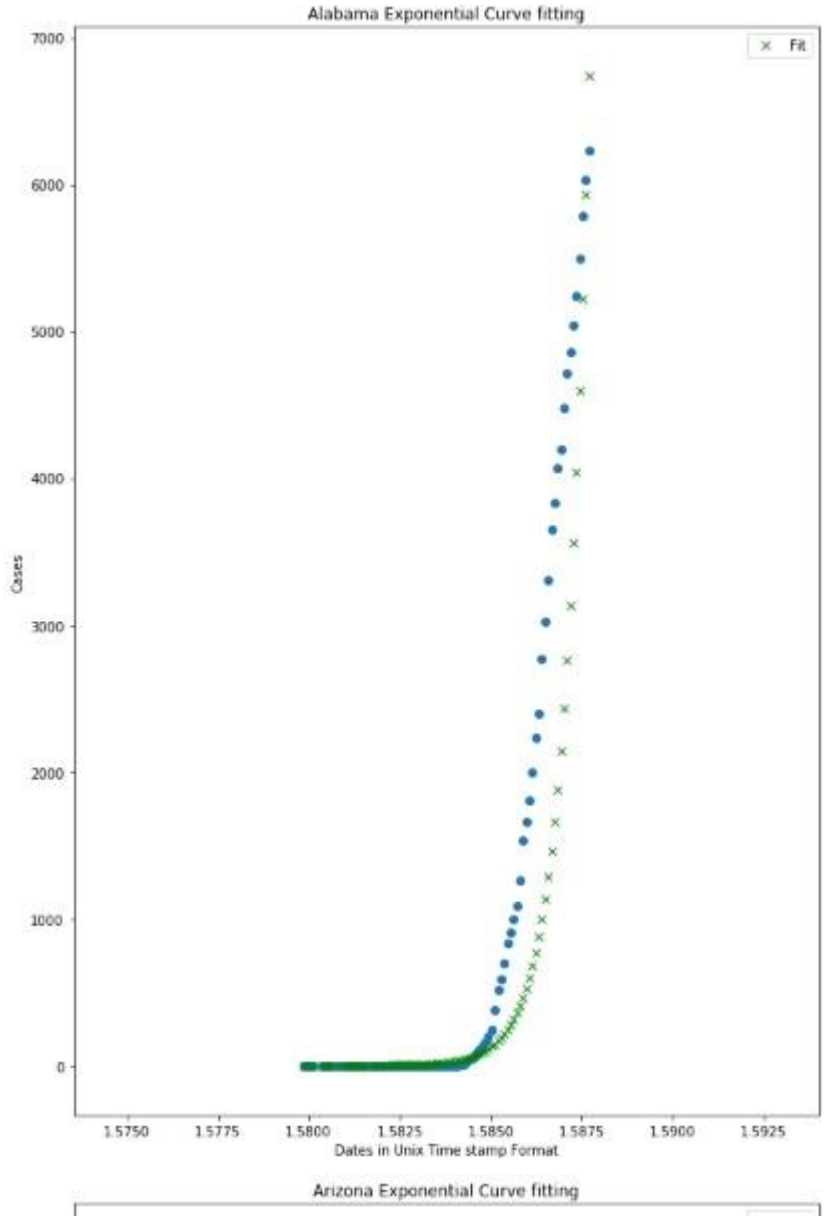


Figure 2. We are trying to fit a line for sample data input for Alabama state.

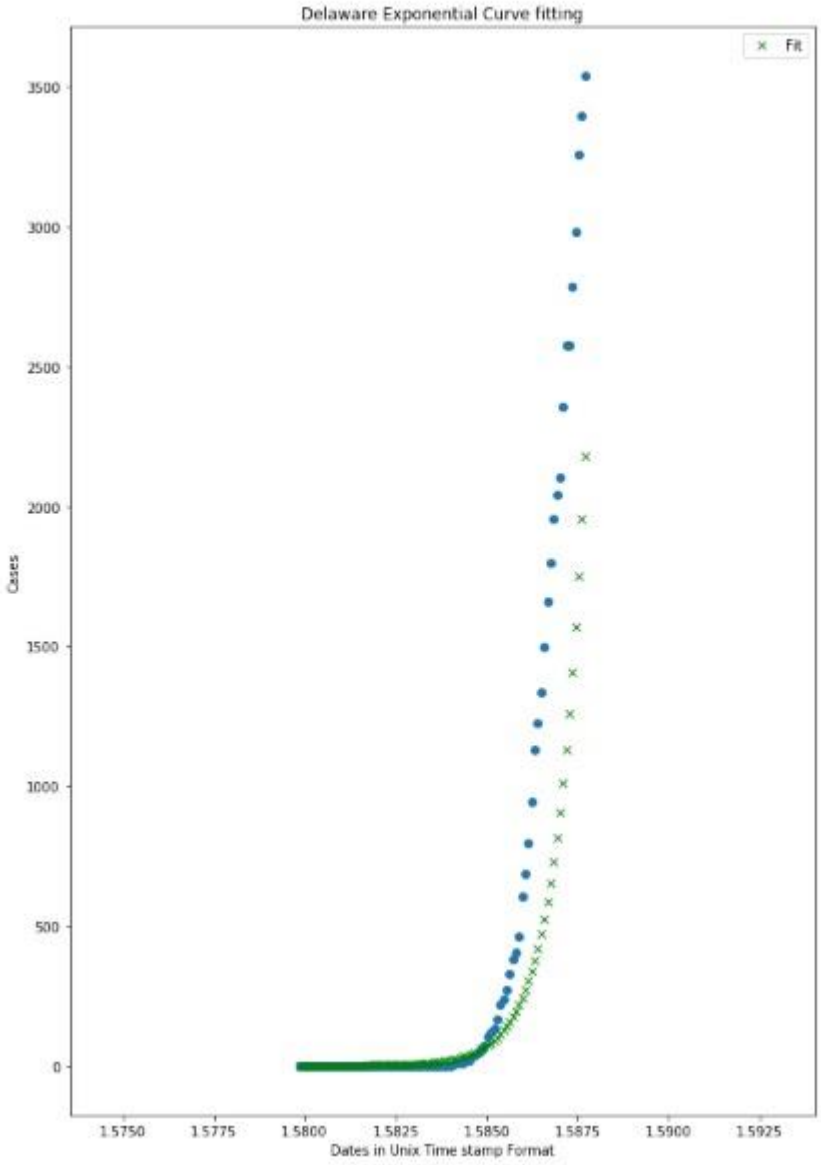


Figure 3. We are trying to fit a line for sample data input for Delaware state.

Similarly, we tried fitting line for each state in the United States.

Implementation with Tensorflow and LSTM:

Since Neural Nets are known for Learning better than a normal Linear Regression Model, we implemented a LSTM model by training the data on the previous 10 days so that it can learn from the history.
But since the dataset has a lot of features, and the type of data is repetitive, the Deep Learning model fails to achieve a Good Rate of accuracy.
The LSTM consists of 128 Units and an Adam Optimizer.
We have used the metrics as Mean Square Error and ran the model for ***500 EPOCHS.***
Adding more layers resulted in increased loss.
Basically, the Model was trying to learn from the data in the lower region of the graph as seen above and trying to predict the Cases for a Future data which is very high (Exponential). Hence, we discarded the idea for Deep Neural Nets. And moved on to statistically powerful libraries which are more statistical.

```
    Xs, ys = [], []
    for i in range(len(X) - time_steps):
        v = X.iloc[i:(i + time_steps)].values
        Xs.append(v)
        ys.append(y.iloc[i + time_steps])
    return np.array(Xs).reshape(-1,np.array(Xs).shape[-1], np.array(Xs).shape[-2]), np.array(ys)
```

```
2071 518
```

In [185]:
```
time_steps = 48
X_train, y_train = create_dataset(train, train.cases, time_steps)
X_test, y_test = create_dataset(test, test.cases, time_steps)

print(X_train.shape, y_train.shape)
model = keras.Sequential()
model.add(keras.layers.LSTM(
  units=128,
  input_shape=(X_train.shape[1], X_train.shape[2])
))
# model.add(keras.layers.Dense(units=512 , input_shape=(X_train.shape[1], X_train.shape[2])))
# model.add(keras.layers.Dense(units=256))

model.add(keras.layers.Dense(units=1))
model.compile(
  loss="MSE",
  metrics=['accuracy'],
  optimizer=keras.optimizers.Adam(0.001)
)
```

In [199]:
```
history = model.fit(
    X_train, y_train,
    epochs=500,
    validation_split=0.2,
    verbose=1,
    shuffle=False
)
```

```
Train on 1618 samples, validate on 405 samples
Epoch 1/500
1618/1618 [==============================] - 4s 3ms/sample - loss: 0.0000e+00 - accuracy: 0.0890 - val_loss: 0.0000e+00 - val_a
ccuracy: 0.0000e+00
Epoch 2/500
1618/1618 [==============================] - 1s 733us/sample - loss: 0.0000e+00 - accuracy: 0.0890 - val_loss: 0.0000e+00 - val
_accuracy: 0.0000e+00
Epoch 3/500
1618/1618 [==============================] - 1s 795us/sample - loss: 0.0000e+00 - accuracy: 0.0890 - val_loss: 0.0000e+00 - val
_accuracy: 0.0000e+00
Epoch 4/500
 512/1618 [========>                     ] - ETA: 0s - loss: 0.0000e+00 - accuracy: 0.2875
```

Forecasting using FacebookProphet:

Then, we started thinking about implementing a model that can work best on this exponential data and also help in making accurate predictions which users can trust and accept. We researched about Prophet - an open source library published by Facebook which provides us the ability to make time series predictions with good accuracy. We used the Prophet to predict the number of COVID-19 cases on any given date for each state in the United States separately.

As the predictions made using Facebook Prophet were close to the current trend, we finalized implementing our system using the powerful library from Facebook.

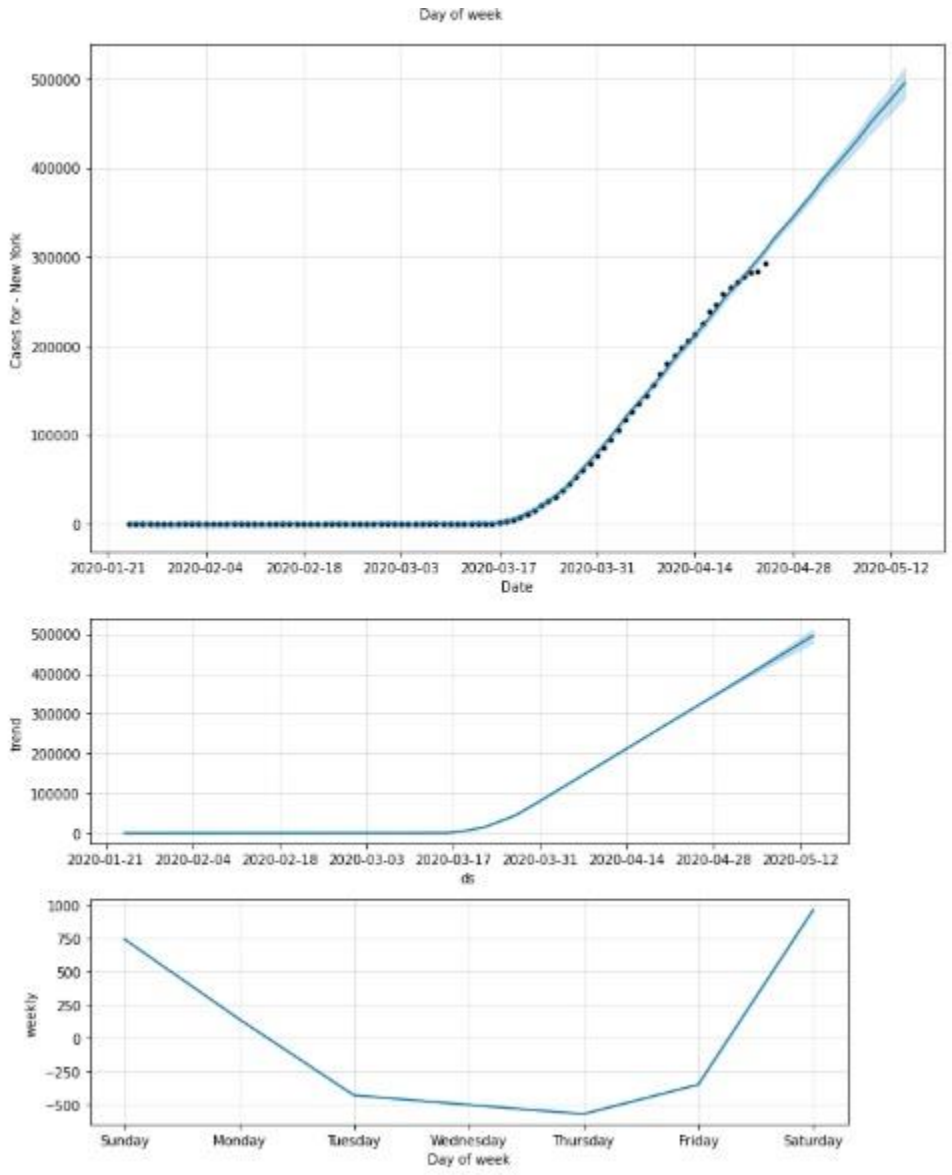


Figure 4: We can observe from the graphs how the extended blue line explains the forecasted number of cases and explain the overall trend in the growth of the number of cases.

Consider one of the graphs we plotted using predictions obtained using Facebook Prophet.

- The first graph shows the date v/s number of cases for New York state. The black dots are the current observations from the dataset and the extended blue line is the forcasting. We can observe from the graph that the cases have grown exponentially and we can observe the exponential growth starting March 17, 2020.
- The second graph shows the trend line of the growth of the number of cases in the New York state each day.
- The third graph shows the trend line of the growth of the number of cases in the New York state weekly.

We are leveraging the power of machine learning for predictions. We have carefully referred to the information shared by Google and implemented the below mentioned rules to develop a good machine learning solution for our project on ‘Prediction of COVID-19 Cases’.

Before Machine Learning:

- **Rule #1:** Don't be afraid to launch a product without machine learning.
  We have carefully checked the usability of our dataset which consists of 500,000 records and 11 features. Therefore, using machine learning will give us a 100% boost in making predictions instead of a heuristic approach.
- **Rule #2:** First, Design and implement metrics.
  We have come up with some strong metrics like R2 score, Mean Squared Error (MSE), and Mean Absolute Error (MAE) after carefully assessing the data quality for making predictions.

ML Phase I: Your First Pipeline:

- **Rule #4:** Keep the first model simple and get the infrastructure right.
  This project is a classic regression problem i.e. predicting the number of confirmed Coronavirus cases on a given date. Therefore, we first implemented a basic linear regression model on simple features to perform predictions and then improved our forecasting system by implementing Facebook Prophet.

ML Phase II: Feature Engineering:

- **Rule #16:** Plan to launch and iterate.
  We have built a system which will simplify incorporating new changes and model tuning that may arise as a future requirement for improvising our machine learning solution.
- **Rule #33:** If you produce a model based on the data until January 5th, test the model on the data from January 6th and after.
  To make sure to better understand the performance of the developed machine learning model, we have used this approach of testing the model on data with dates after the date in the training data.

App Engine:

We have deployed our app on Google App Engine using Google Cloud Platform.
Here is the link to Dataset:
https://kbs-covid.ue.r.appspot.com/

Research Citations:

[1] https://www.thelancet.com/journals/laninf/article/PIIS1473-3099(20)30243-7/fulltext
[2] https://console.cloud.google.com/marketplace/details/johnshopkins/covid19_jhu_global_cases
[3] https://www.fredhutch.org/en/news/center-news/2020/03/coronavirus-latest-scientific-research.html
[4]https://www.accuweather.com/en/weather-blogs/weathermatrix/analysis-of-new-research-paper-tying-coronavirus-to-weather/703270
[5] https://towardsdatascience.com/the-complete-guide-to-time-series-analysis-and-forecasting-70d476bfe775
[6] https://365datascience.com/explainer-video/simple-linear-regression-model/
[7] https://www.tensorflow.org/tutorials/structured_data/time_series
[8] https://www.coursera.org/specializations/gcp-architecture