COVID-19 Final Report

Kyle W. Brown

The COVID-19 Final Project for DSE 6300 was selected based primarily on the timing of the pandemic and its impact to us all and proposed by me. This section follows a top down format and structured to indicate my work contributed to the overall Project Schedule, goals, deadlines, and success. Words highlighted in blue are hyperlinked to their respective model, markdown file, and Jupyter notebook located in the COVID-19 project GitHub repo.

DSE 6300 COVID-19 GitHub Repo

My initial contribution with creating a GitHub repo to track work and progress. The COVID19-Project repo was created on April 1st and collaborator requests were sent to the entire group including the professor. Notices were also sent to the group immediately via a text, email, and/or discussion board post.

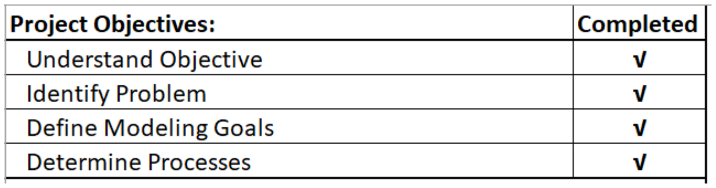
Project Proposal

Initially the Project Proposal was somewhat collaborated on from behalf of the team. Could not agree on any form of communication for several weeks. I addressed each question individually to the group then proceed to answer them myself which is my meaning of somewhat collaboration. After communicating roles with the team, I suggested: Derick: Pipeline, Jacqueline: Machine Learning, Ishaan: Project Manager, Kyle: Dashboard and Visualization, Nishchitha: Storage and Database.

Project Schedule

Laid out the Project Schedule by proposing to the group to follow a shortened agile framework focusing on 2 - (1) week iterations followed by, (1) week of either model integration and dashboard development, and (1) of dashboard deployment. With guidelines of the assignment we needed a Checkpoint 1 and Checkpoint 2 to indicate progress. The Checkpoints were identified by the Project Schedule and was adjusted to reflect the midterm Checkpoint at week (3). Updated the team prior to the first Checkpoint with the Initial Task list that lead to the development of the Agile Schedule. As advised by professor created a final week project schedule which was supposed to work back from the deadline to the day.

Screenshot of Initial Task List



Project Objectives:	Completed
Understand Objective	√
Identify Problem	√
Define Modeling Goals	√
Determine Processes	√

Final Week Project Schedule

Thu	Fri	Sat	Sun	Mon	Tue
4/16/2020	4/17/2020	4/18/2020	4/19/2020	4/20/2020	4/21/2020
Final Modeling	Model update, submit models by 7:00 p.m.	• Group model selection • Kafka setup	• Start ppt • Kafka topics, records, cluster	• Finish ppt parts •Finish pipeline	Finalize ppt

Checkpoint 1

1. What did you do this week?

- Setup GitHub repo.
- Combined several datasets from state level economic, environmental, and health factors. Also added a SARS dataset.
- Commit to repo with linear regression using R.
- Ran a time series analysis using Tensorflow and received an initial score of 18.5%.
- Using AutoML to explore potential deep learning, machine learning, and time series models then will export the model via an API.
    - Exploring different ways to train and test the model against SARS data.
- Standardized data from the initial dataset posted by John Hopkins University Github into 1 dataset based on Date, Province State, Country Region, Confirmed, Deaths, Recovered.
- Developed Task List.
- Attempting to configure SparkR, RStudio, and the Grid with the HPC team.

2. What are you planning to do next week?

- Finalize datasets and data sources.
- Make a commit to GitHub to complete linear regression from Regression_R file in GitHub.
- Complete a time series model expressing a metric like Mean Absolute Error using AutoML as well as the increase the Tensorflow notebook up from 18.5%
- Configure SparkR, shinyjs, Shiny, and the Grid correctly.
- Explore dashboard layouts and style.
- Prepare the data and setup Spark Stream using the Grid.

2. What problems are you having?

- Initially, my biggest problem was overall project organization and understanding what tools could be used.
- Lack of complete data for factors at the city/county/zip code level.
- Cleansing the data for a standardized dataset.
- X11 Forwarding gave me issues for display. Had to export dbus, first.
- Time needed for model training, development, and selection.

- SparkR was not an installed package on my node through the Grid.
- Issues launching RStudio because of X11 Forwarding errors.

Checkpoint 2

1. **What did you do this week?**
   - Configured and installed the remaining components on my Grid RStudio notebook & server to use SparkR, sparklr, RStudio, Shiny, and Spark properly.
   - Completed linear, non-linear, and started a possible exponential regression with varying results.
   - Completed AutoML GBM Multinomial model that provided exceptional results.
   - Created a baseline time series with minor results.
   - Configured dashboard UI, global, server, and CSS folders in dashboard.
   - Made 5 commits this week to the GitHub.

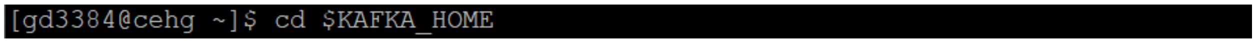
2. **What are you planning to do next week?**
   - Combine all models for presenting purposes.
   - Complete PowerPoint for final presentation.
   - Possibly adjust models for more complexity in operational dashboard.
   - Finalize dashboard CSS fonts, colors and style.
   - Complete final dashboard development.
   - Make final commit to COVID19 GitHub repo with final project accompanied with code, project schedule, data, and findings.

3. **What problems are you having?**
   - Difficulty communicating with group members.
   - No one follows up with updates or deadlines.

Pipeline

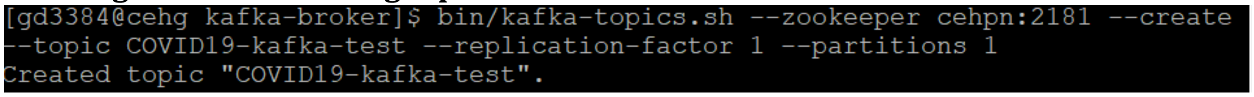
The goal of the original pipeline was to use Kafka and stream to the Grid using SparkR and Spark. Attention was given since the project objective centered around building a functional dashboard in Shiny presenting COVID19 virus predictions and streaming COVID data using Kafka or Spark Streaming. Unfortunately, time constraints played the biggest factor in the pipeline development for me.

Kafka directory to the Grid:



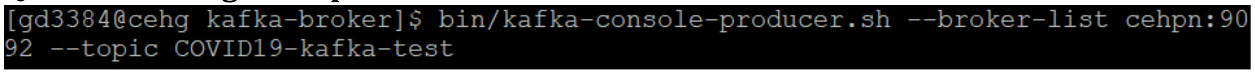
```
[gd3384@cehg ~]$ cd $KAFKA_HOME
```

Creating a COVID19 testing topic:



```
[gd3384@cehg kafka-broker]$ bin/kafka-topics.sh --zookeeper cehpn:2181 --create
--topic COVID19-kafka-test --replication-factor 1 --partitions 1
Created topic "COVID19-kafka-test".
```

Quick test using the topic:



```
[gd3384@cehg kafka-broker]$ bin/kafka-console-producer.sh --broker-list cehpn:90
92 --topic COVID19-kafka-test
```

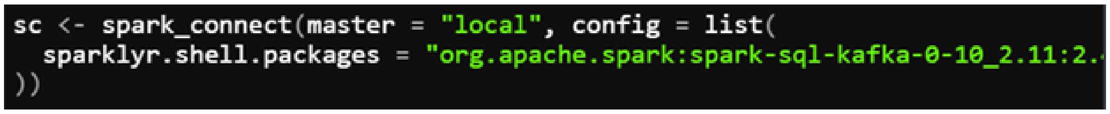
Consumer contents for the COVID19 topic:

```
>
bin/kafka-console-consumer.sh --zookeeper cehpn:2181 --topic COVID19-kafka-test
--from-beginning>
```

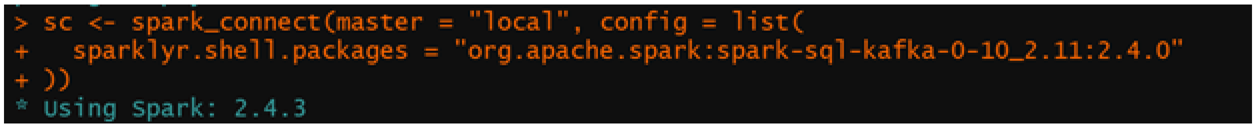
Connected SparkR and Kafka on the Grid

Connected Grid with SparkR and Kafka using WSU Ondemand HPC RStudio Server. Installed Spark into my RStudio environment to use SparkR on the Grid and Kafka for streaming but came up short of integration due to timing.

Connected SparkR and Kafka to the Grid

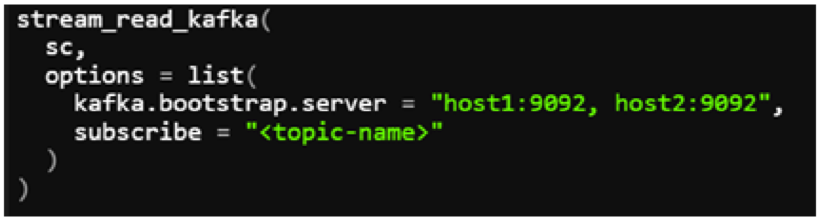


```
sc <- spark_connect(master = "local", config = list(
  sparklyr.shell.packages = "org.apache.spark:spark-sql-kafka-0-10_2.11:2.
))
```



```
> sc <- spark_connect(master = "local", config = list(
+   sparklyr.shell.packages = "org.apache.spark:spark-sql-kafka-0-10_2.11:2.4.0"
+ ))
* Using Spark: 2.4.3
```

Example setup to read Kafka stream.



```
stream_read_kafka(
  sc,
  options = list(
    kafka.bootstrap.server = "host1:9092, host2:9092",
    subscribe = "<topic-name>"
  )
)
```

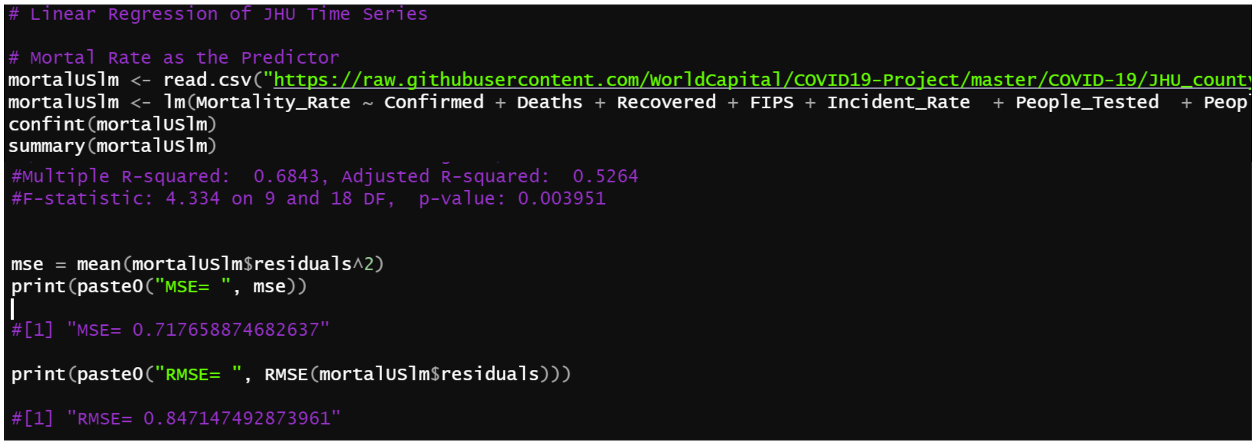
Models

The modeling process began with a linear regression and progressed into AutoML GBM Multinomial model. The initial Linear Regression was posted on 4/4/2020 as Regression_R to the project GitHub with 20+ models compiled using CDC data. The remaining models were created from either STATWORX COVID19 API and John Hopkins University GitHub (both files are included in repo). On 4/11/2020, a Models Update R Markdown file containing several linear, non-linear, and principal component analysis models were added to the repo.

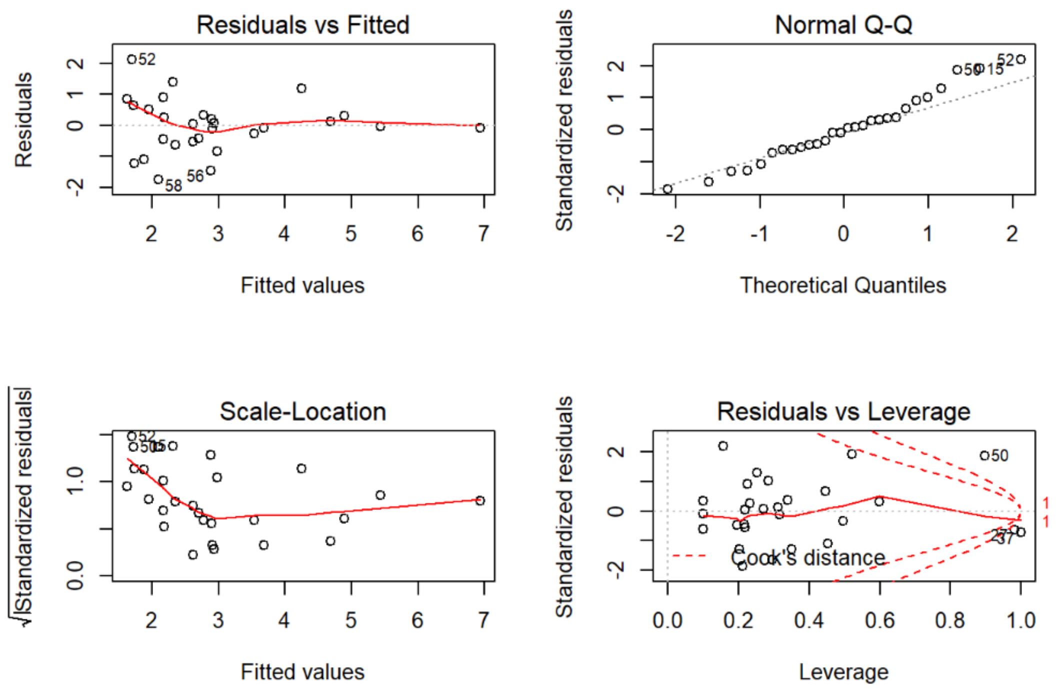
The models used were:

- Linear Regression
- Multivariate Regression
- Non-Linear Regression
- Principal Component Analysis
- Tensorflow Time Series
- AutoML Deep Learning

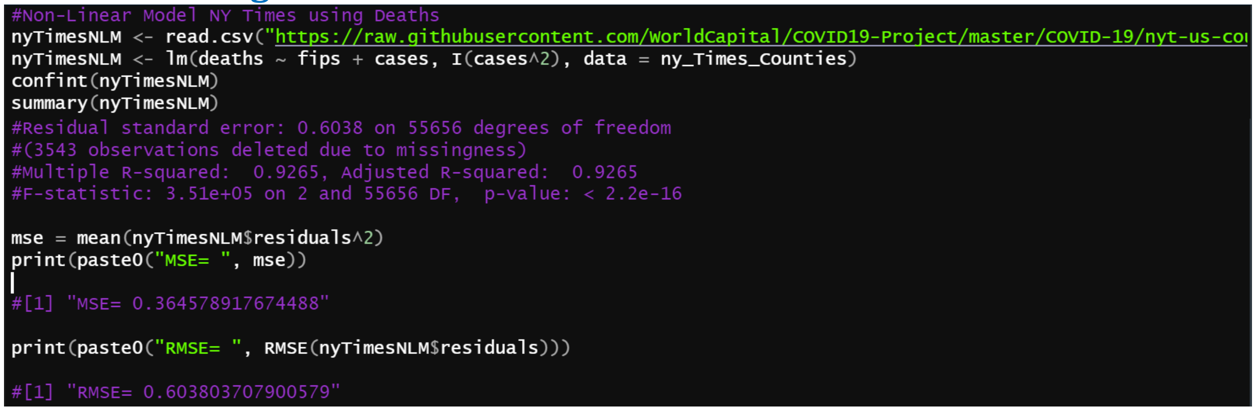
Linear Regressions



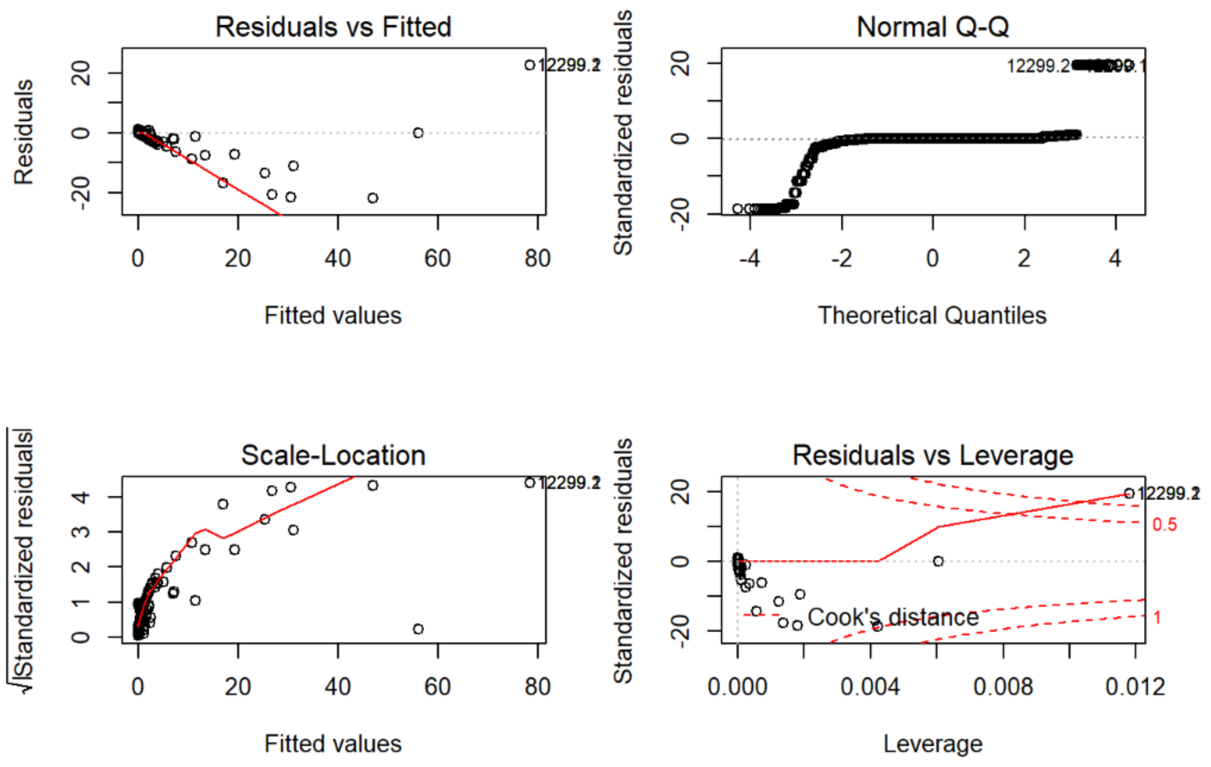
```
# Linear Regression of JHU Time Series

# Mortal Rate as the Predictor
mortalUSlm <- read.csv("https://raw.githubusercontent.com/WorldCapital/COVID19-Project/master/COVID-19/JHU_county
mortalUSlm <- lm(Mortality_Rate ~ Confirmed + Deaths + Recovered + FIPS + Incident_Rate  + People_Tested  + Peop
confint(mortalUSlm)
summary(mortalUSlm)
#Multiple R-squared:  0.6843, Adjusted R-squared:  0.5264
#F-statistic: 4.334 on 9 and 18 DF,  p-value: 0.003951


mse = mean(mortalUSlm$residuals^2)
print(paste0("MSE= ", mse))
|
#[1] "MSE= 0.717658874682637"

print(paste0("RMSE= ", RMSE(mortalUSlm$residuals)))

#[1] "RMSE= 0.847147492873961"
```



Non-Linear Regressions



```
#Non-Linear Model NY Times using Deaths
nyTimesNLM <- read.csv("https://raw.githubusercontent.com/WorldCapital/COVID19-Project/master/COVID-19/nyt-us-co
nyTimesNLM <- lm(deaths ~ fips + cases, I(cases^2), data = ny_Times_Counties)
confint(nyTimesNLM)
summary(nyTimesNLM)
#Residual standard error: 0.6038 on 55656 degrees of freedom
#(3543 observations deleted due to missingness)
#Multiple R-squared:  0.9265, Adjusted R-squared:  0.9265
#F-statistic: 3.51e+05 on 2 and 55656 DF,  p-value: < 2.2e-16

mse = mean(nyTimesNLM$residuals^2)
print(paste0("MSE= ", mse))
|
#[1] "MSE= 0.364578917674488"

print(paste0("RMSE= ", RMSE(nyTimesNLM$residuals)))

#[1] "RMSE= 0.603803707900579"
```

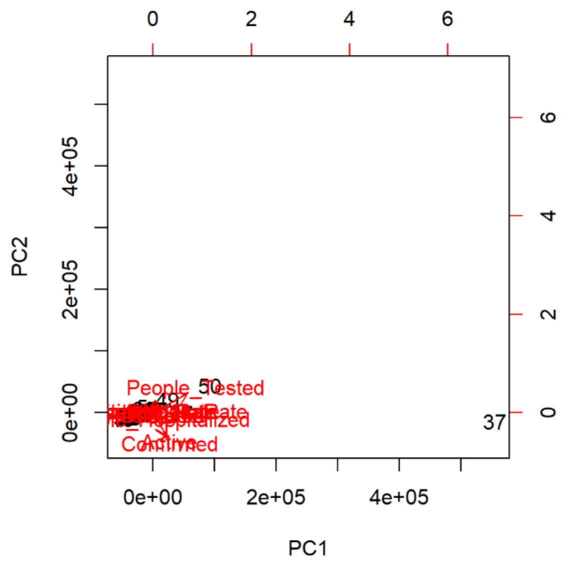


Principal Component Analysis (PCA)

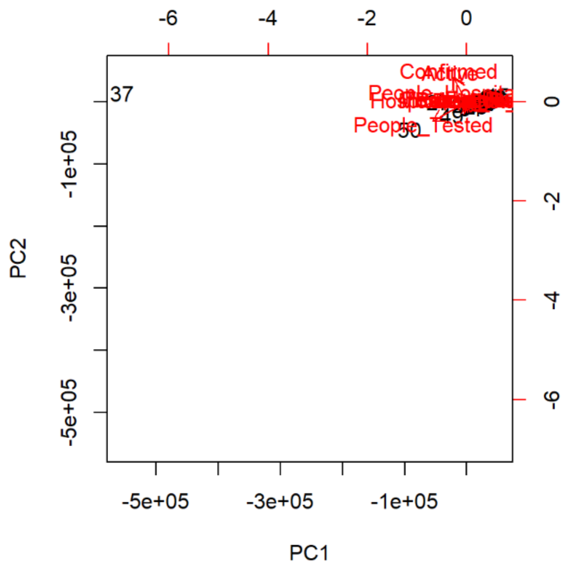
```
# Principal Component Analysis

countUS <- subset(countyUS, select = -c(Province_State, Last_Update,Country_Region,UID, Lat, Long_, ISO3))

countUS <- na.omit(countUS)

countUS <- transform(countUS, Confirmed = as.numeric(Confirmed),
          Deaths = as.numeric(Deaths),
          Recovered = as.numeric(Recovered),
          Active = as.numeric(Active),
          FIPS = as.numeric(FIPS),
          People_Tested = as.numeric(People_Tested),
          People_Hospitalized = as.numeric(People_Hospitalized))

apply(countUS, 2, mean)
```

```
##           Confirmed              Deaths           Recovered
##        11165.321429          553.250000         1405.750000
##              Active                FIPS       Incident_Rate
##        10612.071429           32.500000          150.369843
##       People_Tested People_Hospitalized      Mortality_Rate
##        49492.428571         2158.714286            2.949897
##        Testing_Rate Hospitalization_Rate
##         1274.784932           14.246080
```

```
biplot(pr.out, scale =0)
```



```
pr.out$rotation = -pr.out$rotation
pr.out$x = -pr.out$x
biplot(pr.out, scale =0)
```



```
pve = pr.var/sum(pr.var)
pve
```

```
##  [1] 9.900833e-01 9.854313e-03 4.000426e-05 1.641233e-05 5.064630e-06
##  [6] 6.897524e-07 1.881403e-07 2.168554e-08 1.291613e-09 5.738525e-11
## [11] 5.100606e-33
```

pve shows variability >1.

Tensorflow Time Series

The Tensorflow Time Series performed

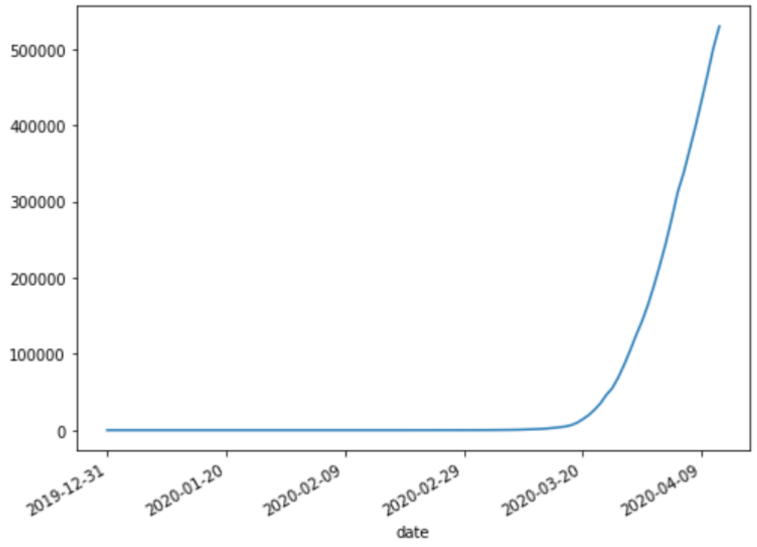
In [0]:
```
import tensorflow as tf
tf.enable_eager_execution()

import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd
import requests
import json

mpl.rcParams['figure.figsize'] = (8, 6)
mpl.rcParams['axes.grid'] = False

# POST to API
payload = {'code': 'US'} # or {'code': 'DE'}
URL = 'https://api.statworx.com/covid'
response = requests.post(url=URL, data=json.dumps(payload))

# Convert to data frame
df = pd.DataFrame.from_dict(json.loads(response.text))
```

In [0]:
```
df.head()
```

Out[0]:

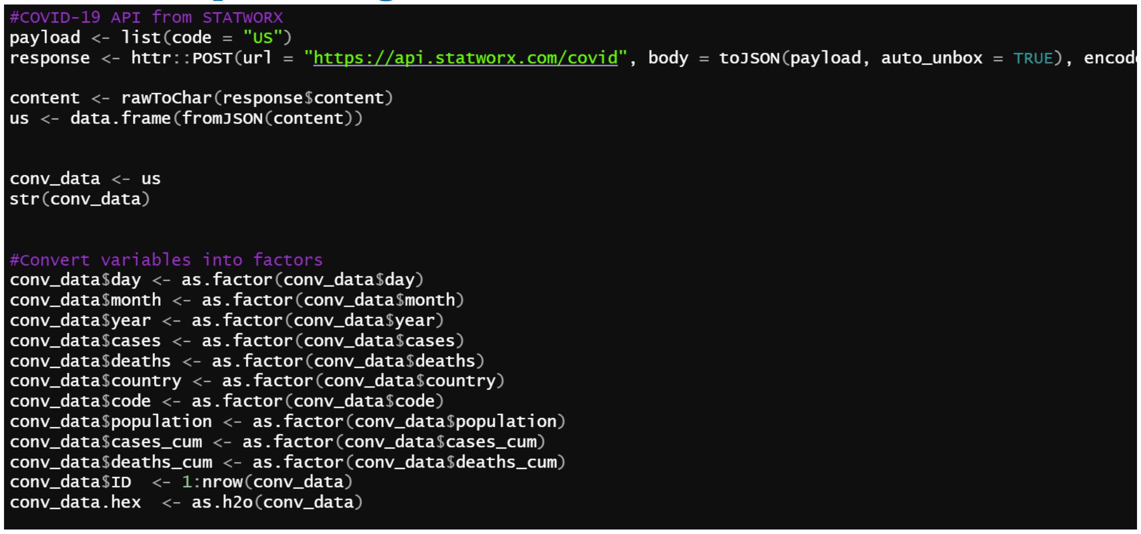
	date	day	month	year	cases	deaths	country	code	population	cases_cum	deaths_cum
0	2019-12-31	31	12	2019	0	0	United_States_of_America	US	327167434	0	0
1	2020-01-01	1	1	2020	0	0	United_States_of_America	US	327167434	0	0
2	2020-01-02	2	1	2020	0	0	United_States_of_America	US	327167434	0	0
3	2020-01-03	3	1	2020	0	0	United_States_of_America	US	327167434	0	0
4	2020-01-04	4	1	2020	0	0	United_States_of_America	US	327167434	0	0
...	...	...	...	...	...	...	...	...	...	...	...
99	2020-04-08	8	4	2020	30613	1906	United_States_of_America	US	327167434	398809	12895

```
In [0]: uni_data.plot(subplots=True)

Out[0]: array([<matplotlib.axes._subplots.AxesSubplot object at 0x7ff79e3c4cf8>],
              dtype=object)
```

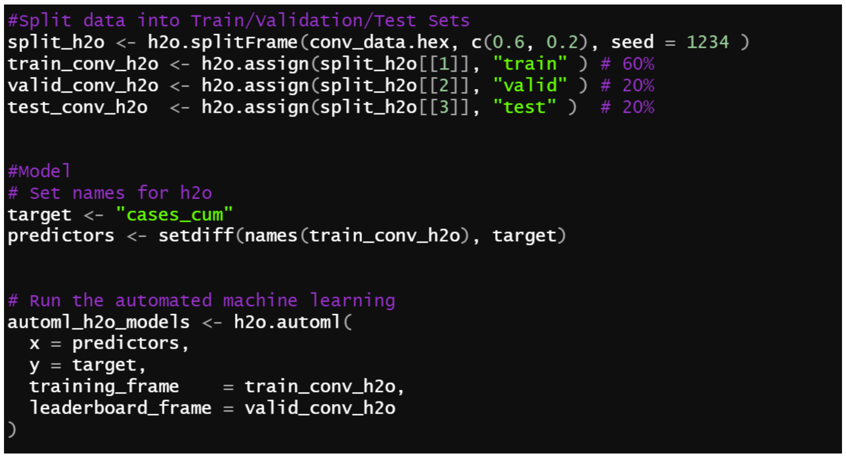


```
In [0]: print ('Single window of past history')
        print (x_train_uni[0])
        print ('\n Target cumulative confirmed to predict')
        print (y_train_uni[0])

Single window of past history
[[-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]
 [-0.35604638]]

 Target cumulative confirmed to predict
-0.3560463830477833
```

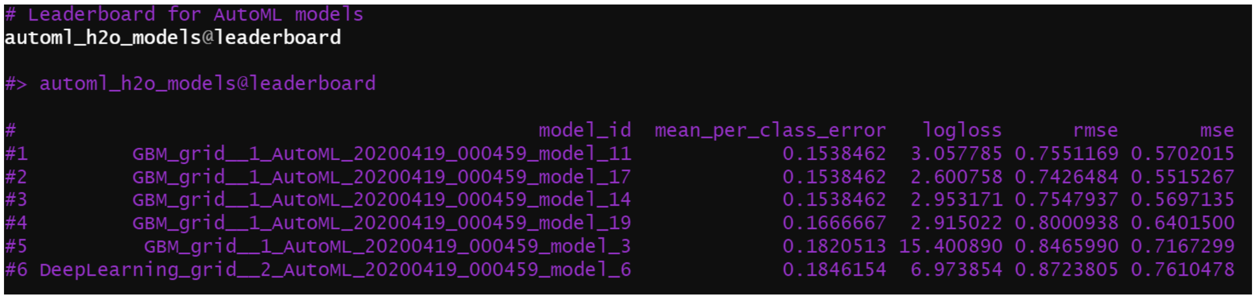
AutoML Deep Learning Models



```
#COVID-19 API from STATWORX
payload <- list(code = "US")
response <- httr::POST(url = "https://api.statworx.com/covid", body = toJSON(payload, auto_unbox = TRUE), encod

content <- rawToChar(response$content)
us <- data.frame(fromJSON(content))


conv_data <- us
str(conv_data)


#Convert variables into factors
conv_data$day <- as.factor(conv_data$day)
conv_data$month <- as.factor(conv_data$month)
conv_data$year <- as.factor(conv_data$year)
conv_data$cases <- as.factor(conv_data$cases)
conv_data$deaths <- as.factor(conv_data$deaths)
conv_data$country <- as.factor(conv_data$country)
conv_data$code <- as.factor(conv_data$code)
conv_data$population <- as.factor(conv_data$population)
conv_data$cases_cum <- as.factor(conv_data$cases_cum)
conv_data$deaths_cum <- as.factor(conv_data$deaths_cum)
conv_data$ID  <- 1:nrow(conv_data)
conv_data.hex  <- as.h2o(conv_data)
```



```
#Split data into Train/Validation/Test Sets
split_h2o <- h2o.splitFrame(conv_data.hex, c(0.6, 0.2), seed = 1234 )
train_conv_h2o <- h2o.assign(split_h2o[[1]], "train" ) # 60%
valid_conv_h2o <- h2o.assign(split_h2o[[2]], "valid" ) # 20%
test_conv_h2o  <- h2o.assign(split_h2o[[3]], "test" )  # 20%


#Model
# Set names for h2o
target <- "cases_cum"
predictors <- setdiff(names(train_conv_h2o), target)


# Run the automated machine learning
automl_h2o_models <- h2o.automl(
  x = predictors,
  y = target,
  training_frame    = train_conv_h2o,
  leaderboard_frame = valid_conv_h2o
)
```

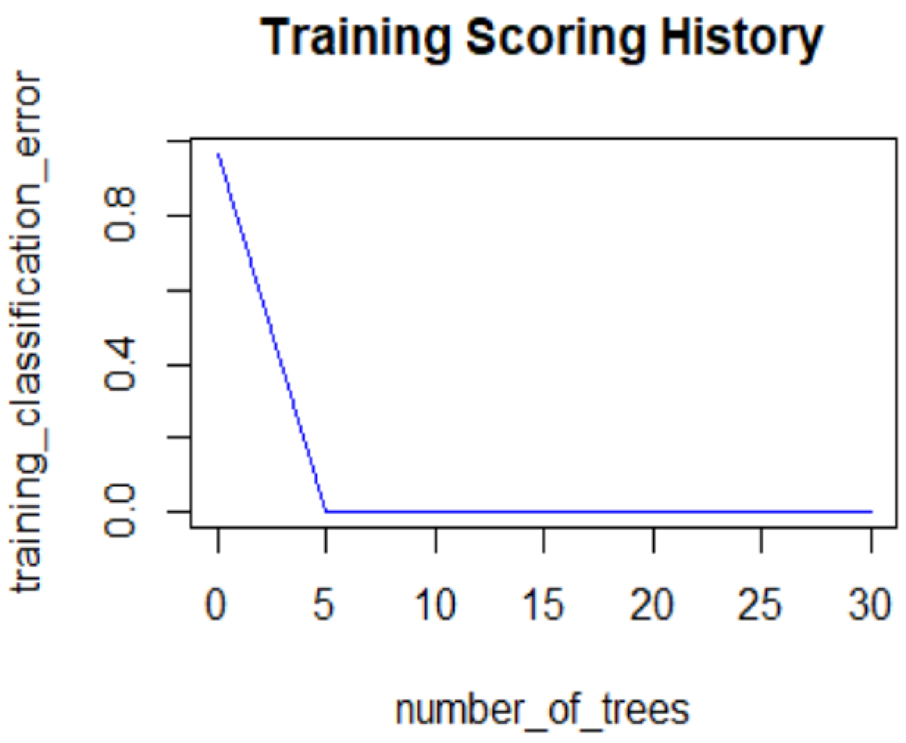


```
# Leaderboard for AutoML models
automl_h2o_models@leaderboard

#> automl_h2o_models@leaderboard

#                                              model_id  mean_per_class_error   logloss      rmse       mse
#1         GBM_grid__1_AutoML_20200419_000459_model_11              0.1538462  3.057785 0.7551169 0.5702015
#2         GBM_grid__1_AutoML_20200419_000459_model_17              0.1538462  2.600758 0.7426484 0.5515267
#3         GBM_grid__1_AutoML_20200419_000459_model_14              0.1538462  2.953171 0.7547937 0.5697135
#4         GBM_grid__1_AutoML_20200419_000459_model_19              0.1666667  2.915022 0.8000938 0.6401500
#5          GBM_grid__1_AutoML_20200419_000459_model_3              0.1820513 15.400890 0.8465990 0.7167299
#6 DeepLearning_grid__2_AutoML_20200419_000459_model_6              0.1846154  6.973854 0.8723805 0.7610478
```

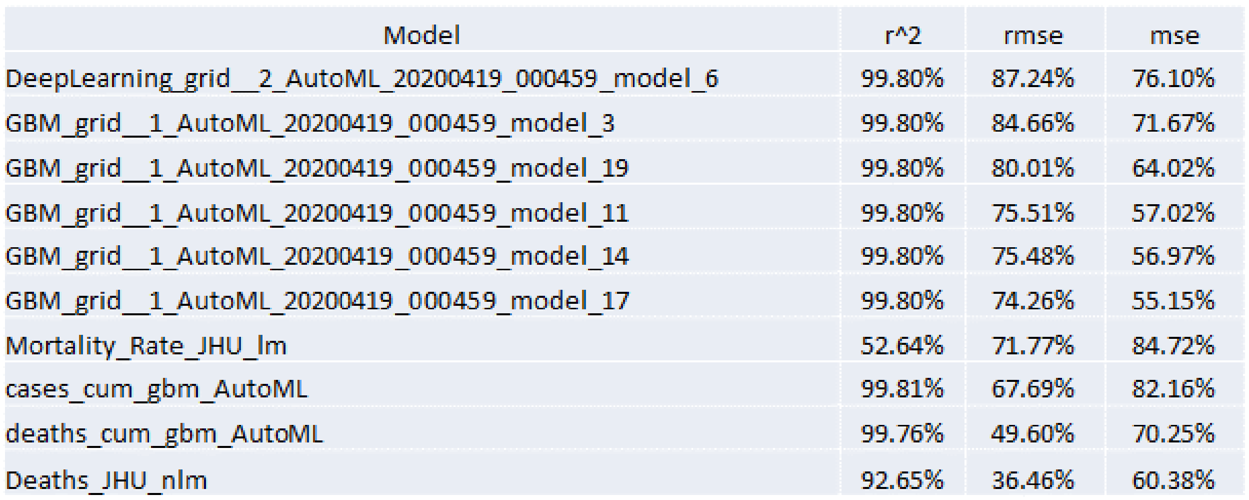
```
# Extract leader model
automl_leader <- automl_h2o_models@leader
automl_leader


#> automl_leader
#Model Details:
#  ==============

#  H2OMultinomialModel: gbm
#Model ID:  GBM_grid__1_AutoML_20200419_000459_model_11
#Model Summary:
#  number_of_trees number_of_internal_trees model_size_in_bytes min_depth max_depth mean_depth min_leaves
#1              30                     1440              246515         1         4    3.63472          2
#max_leaves mean_leaves
#1        16     6.77708
```

```
#Cross-Validation Metrics Summary:
#                              mean          sd cv_1_valid cv_2_valid cv_3_valid cv_4_valid cv_5_valid
#accuracy                0.30769232 0.094211146 0.46153846 0.23076923 0.23076923 0.30769232 0.30769232
#err                      0.6923077 0.094211146 0.53846157  0.7692308  0.7692308  0.6923077  0.6923077
#err_count                      9.0   1.2247449        7.0       10.0       10.0        9.0        9.0
#logloss                  3.4635184   0.4602315  2.7115996  3.6106849  3.9662728   3.548047  3.4809875
#max_per_class_error            1.0         0.0        1.0        1.0        1.0        1.0        1.0
#mean_per_class_accuracy 0.86153847  0.01884223  0.8923077 0.84615386 0.84615386 0.86153847 0.86153847
#mean_per_class_error    0.13846155  0.01884223 0.10769231 0.15384616 0.15384616 0.13846155 0.13846155
#mse                      0.6727001  0.08535113 0.53059614 0.71209335  0.7577791  0.6815926 0.68143946
#r2                       0.9982088 6.2457175E-4  0.9988421 0.99843067 0.99719256 0.99811006  0.9984687
#rmse                    0.81877226 0.053760055 0.72842026  0.8438563  0.8705051  0.8255862 0.82549345
```

```
# AutoML Leader plot
plot(automl_leader)
```



Top Models Results



Model	r^2	rmse	mse
DeepLearning_grid__2_AutoML_20200419_000459_model_6	99.80%	87.24%	76.10%
GBM_grid__1_AutoML_20200419_000459_model_3	99.80%	84.66%	71.67%
GBM_grid__1_AutoML_20200419_000459_model_19	99.80%	80.01%	64.02%
GBM_grid__1_AutoML_20200419_000459_model_11	99.80%	75.51%	57.02%
GBM_grid__1_AutoML_20200419_000459_model_14	99.80%	75.48%	56.97%
GBM_grid__1_AutoML_20200419_000459_model_17	99.80%	74.26%	55.15%
Mortality_Rate_JHU_lm	52.64%	71.77%	84.72%
cases_cum_gbm_AutoML	99.81%	67.69%	82.16%
deaths_cum_gbm_AutoML	99.76%	49.60%	70.25%
Deaths_JHU_nlm	92.65%	36.46%	60.38%

Top Performer Based on AutoML Leader

	R^2	MSE	RMSE
cases_cum_gbm_model	99.81%	67.69%	82.16%

Models Summary

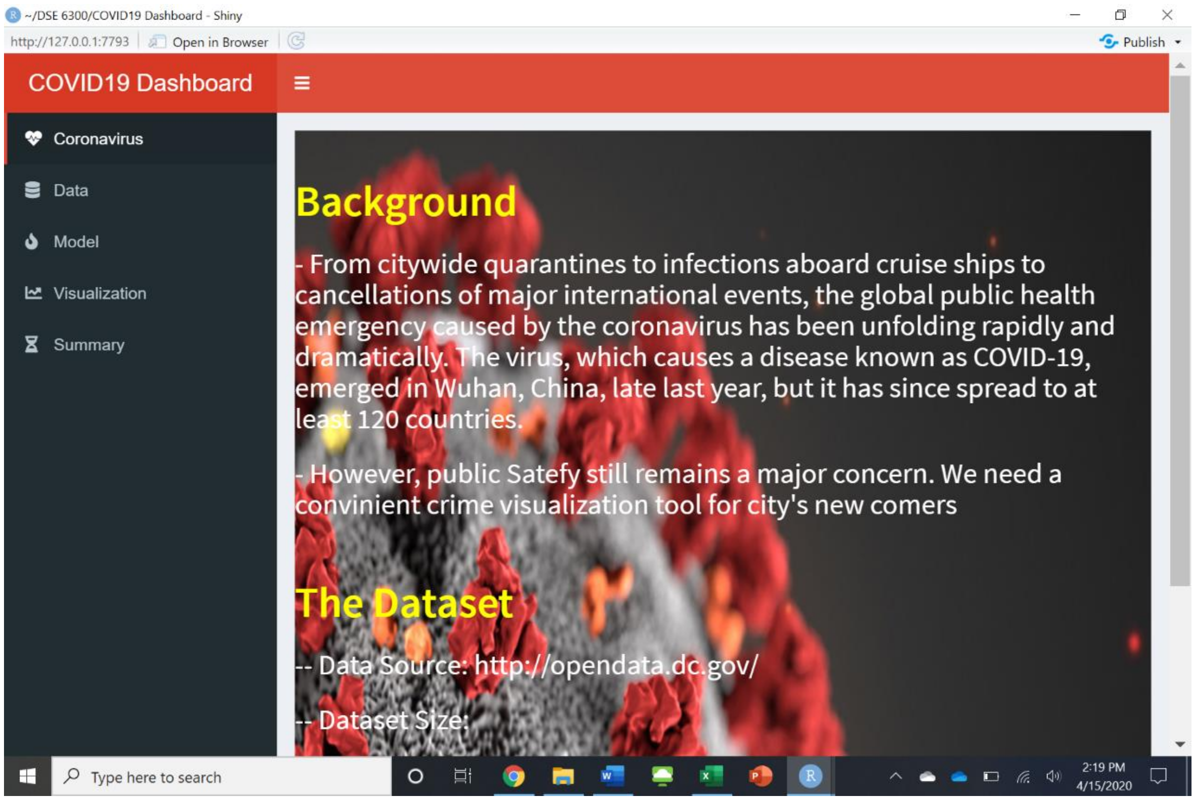
In summary, the modeling needed another week to deploy on real data. AutoML provided the best models with results ranging from 99% R-Squared, 74%-87% RMSE, and 55%-76% MSE. These models far outperform any of the R linear, non-linear, and exponential regressions, PCA, and Tensorflow Time Series. It is worth noting that Deaths JHU non-linear model outperformed Mortality Rate JHU linear model in R-Squared 92.65% to 52.64%. The Tensorflow Time Series only baselined 36.5% accuracy but not much better. The PCA PVE was greater than 1 rendering it useless.

Dashboards

Shiny Dashboard

The COVID19 Shiny Dashboard went into development once the topic for the project was selected. The COVID19 Dashboard directories, UI, server, CSS, and deployment was configured. The project ran out of time to incorporate the Grid and SparkR due to lack of pipeline and streams throughout the project. Attached below is a screenshot of the Shiny dashboard testing I used for initial testing using the STATWORX API. Once Shiny and SparkR became unfeasible, the project migrated to Tableau for dashboard development.

Shiny Dashboard Screenshots:



Background

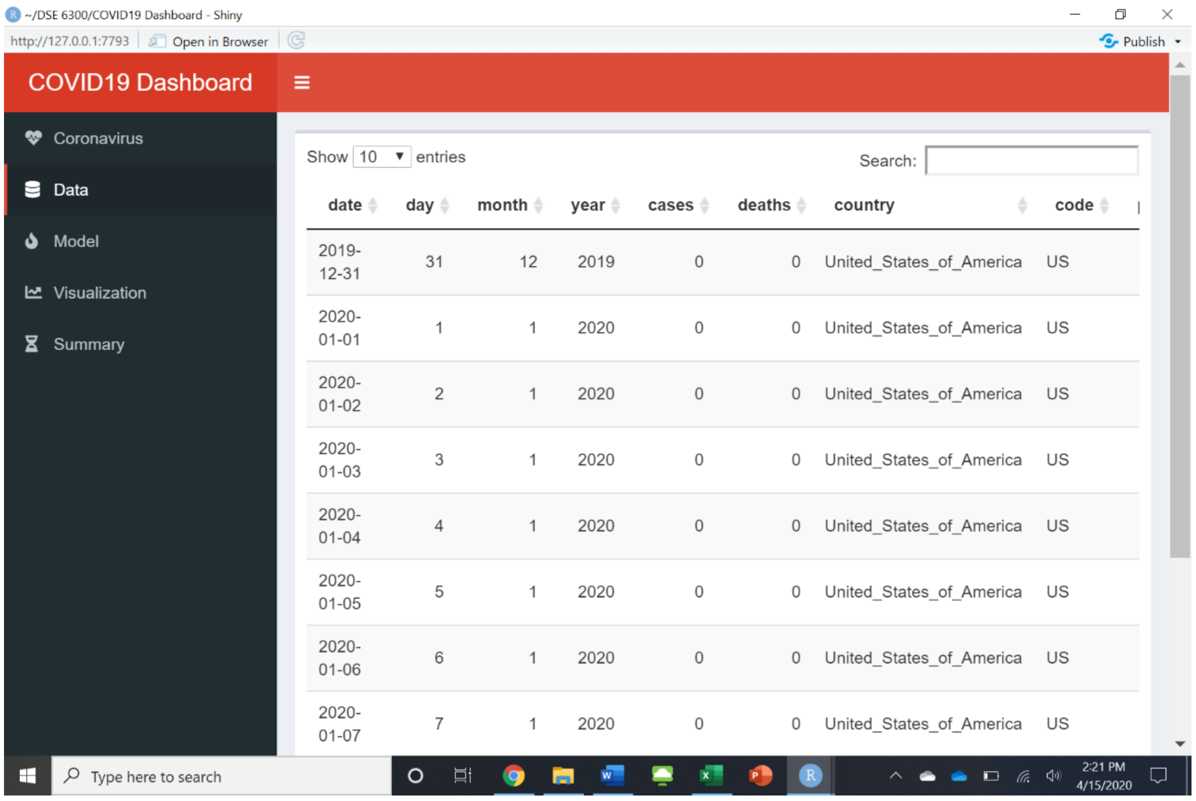
- From citywide quarantines to infections aboard cruise ships to cancellations of major international events, the global public health emergency caused by the coronavirus has been unfolding rapidly and dramatically. The virus, which causes a disease known as COVID-19, emerged in Wuhan, China, late last year, but it has since spread to at least 120 countries.

- However, public Satefy still remains a major concern. We need a convinient crime visualization tool for city's new comers

The Dataset

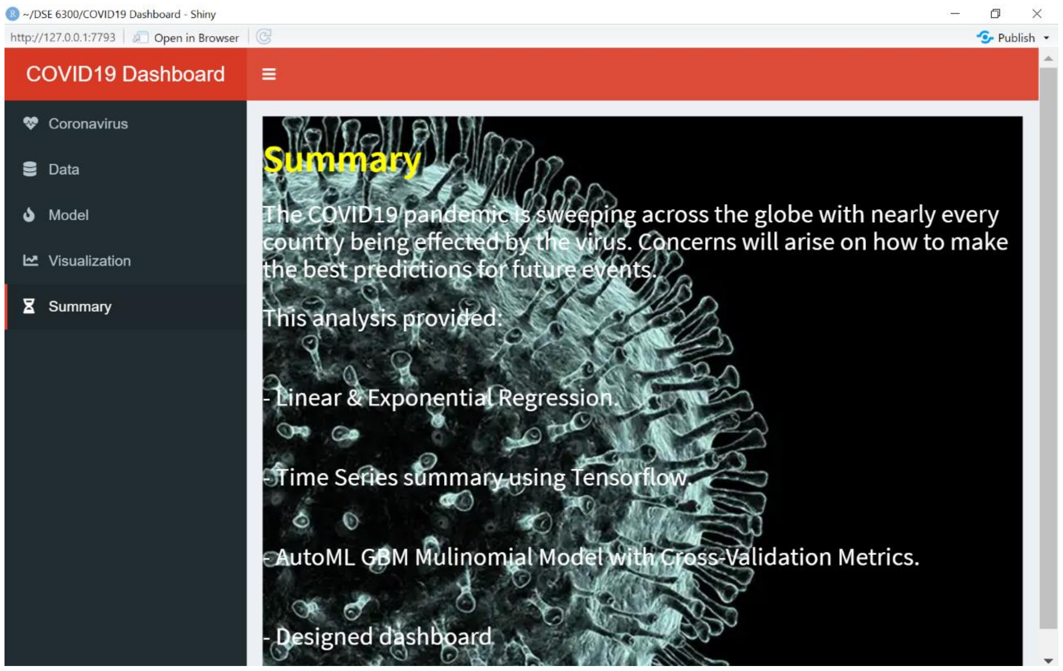
-- Data Source: http://opendata.dc.gov/

-- Dataset Size:



Show 10 entries Search:

date	day	month	year	cases	deaths	country	code
2019-12-31	31	12	2019	0	0	United_States_of_America	US
2020-01-01	1	1	2020	0	0	United_States_of_America	US
2020-01-02	2	1	2020	0	0	United_States_of_America	US
2020-01-03	3	1	2020	0	0	United_States_of_America	US
2020-01-04	4	1	2020	0	0	United_States_of_America	US
2020-01-05	5	1	2020	0	0	United_States_of_America	US
2020-01-06	6	1	2020	0	0	United_States_of_America	US
2020-01-07	7	1	2020	0	0	United_States_of_America	US



Summary

The COVID19 pandemic is sweeping across the globe with nearly every country being effected by the virus. Concerns will arise on how to make the best predictions for future events.

This analysis provided:

- Linear & Exponential Regression.

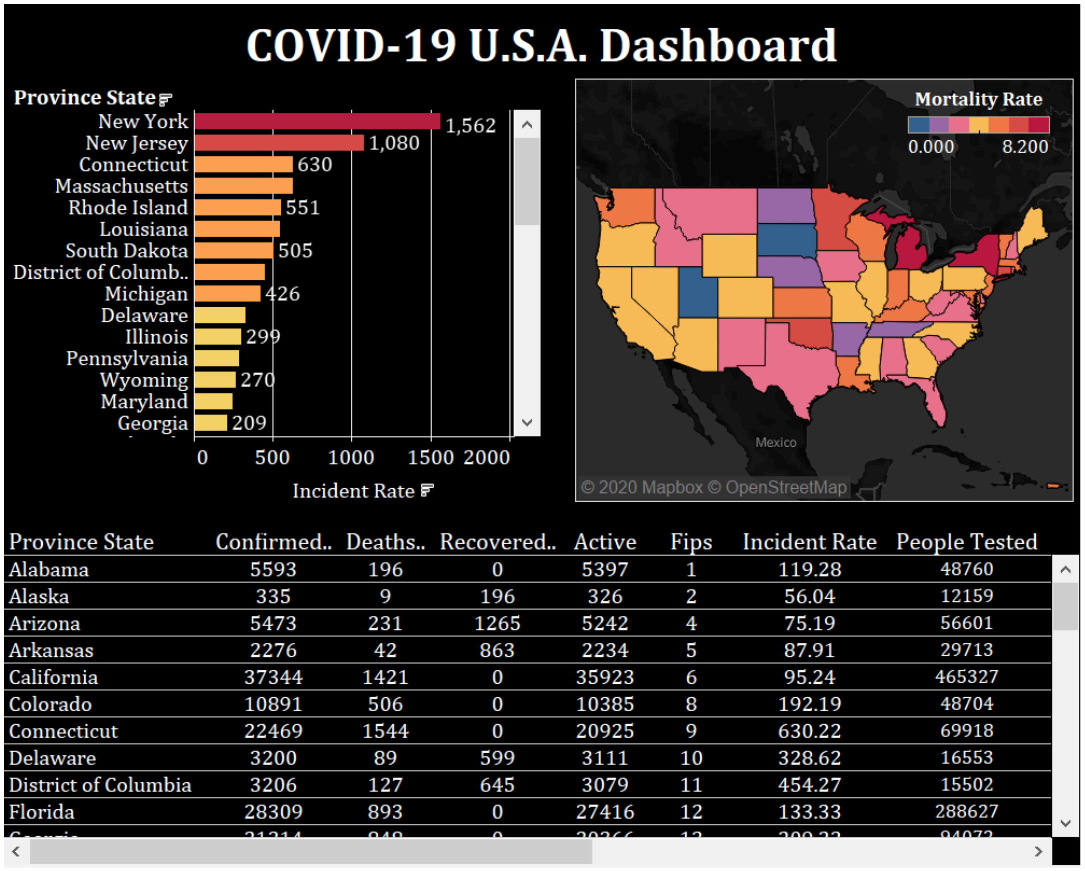
- Time Series summary using Tensorflow.

- AutoML GBM Mulinomial Model with Cross-Validation Metrics.

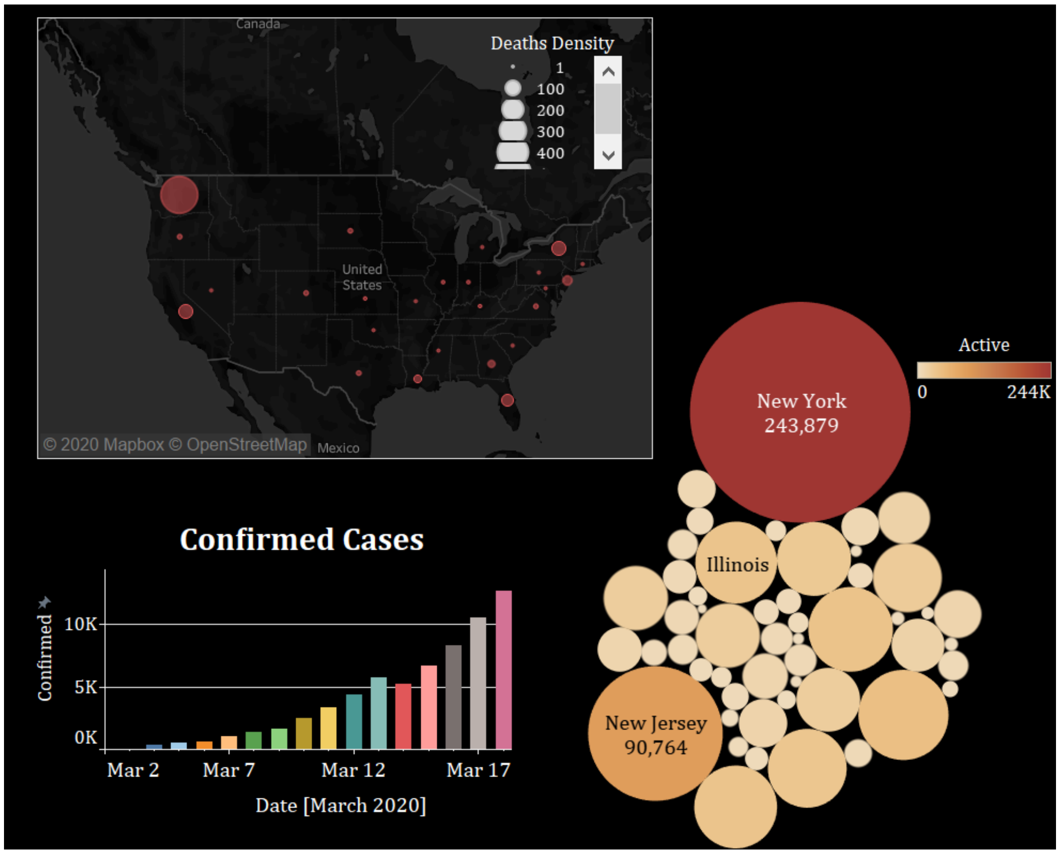
- Designed dashboard.

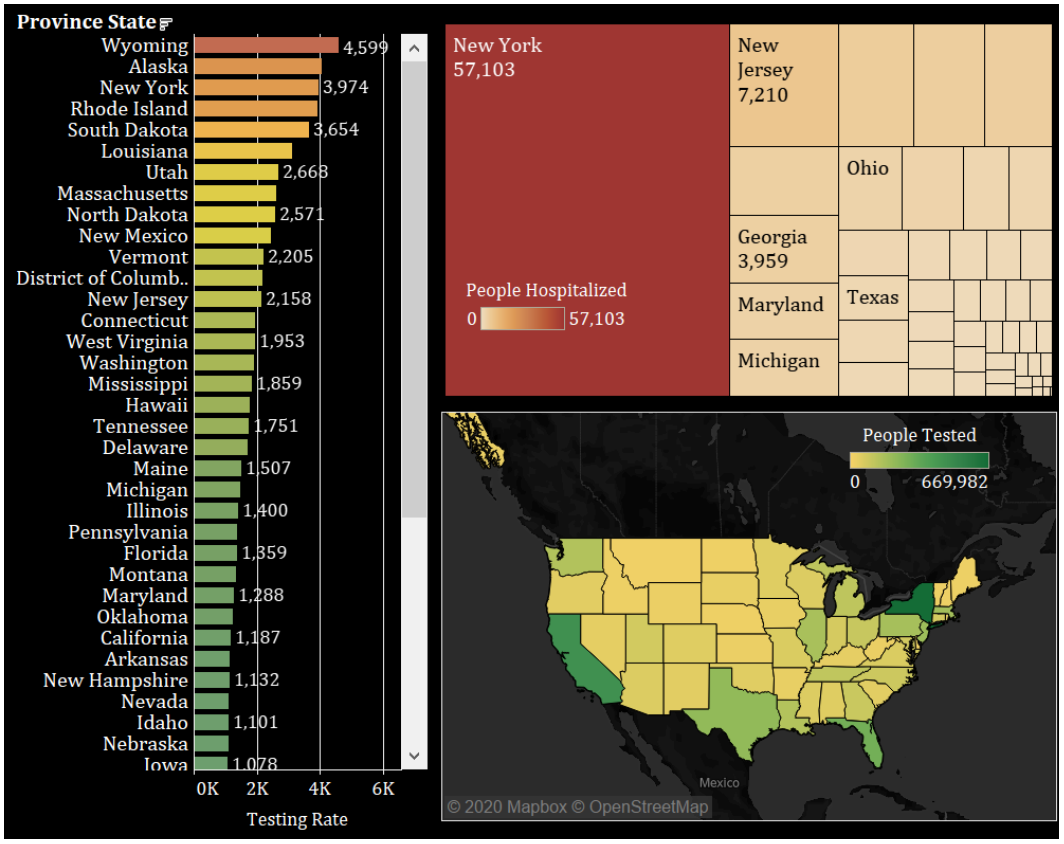
Tableau Dashboard

Tableau Dashboard Screenshots:



Province State	Confirmed..	Deaths..	Recovered..	Active	Fips	Incident Rate	People Tested
Alabama	5593	196	0	5397	1	119.28	48760
Alaska	335	9	196	326	2	56.04	12159
Arizona	5473	231	1265	5242	4	75.19	56601
Arkansas	2276	42	863	2234	5	87.91	29713
California	37344	1421	0	35923	6	95.24	465327
Colorado	10891	506	0	10385	8	192.19	48704
Connecticut	22469	1544	0	20925	9	630.22	69918
Delaware	3200	89	599	3111	10	328.62	16553
District of Columbia	3206	127	645	3079	11	454.27	15502
Florida	28309	893	0	27416	12	133.33	288627


Deaths Density
1
100
200
300
400
Canada
United States
Mexico
© 2020 Mapbox © OpenStreetMap
Active
0
244K
New York 243,879
Illinois
New Jersey 90,764
Confirmed Cases
Confirmed
10K
5K
0K
Mar 2
Mar 7
Mar 12
Mar 17
Date [March 2020]


Province State
Wyoming 4,599
Alaska
New York 3,974
Rhode Island
South Dakota 3,654
Louisiana
Utah 2,668
Massachusetts
North Dakota 2,571
New Mexico
Vermont 2,205
District of Columb..
New Jersey 2,158
Connecticut
West Virginia 1,953
Washington
Mississippi 1,859
Hawaii
Tennessee 1,751
Delaware
Maine 1,507
Michigan
Illinois 1,400
Pennsylvania
Florida 1,359
Montana
Maryland 1,288
Oklahoma
California 1,187
Arkansas
New Hampshire 1,132
Nevada
Idaho 1,101
Nebraska
Iowa 1,078
0K
2K
4K
6K
Testing Rate
New York 57,103
New Jersey 7,210
Ohio
Georgia 3,959
Maryland
Texas
Michigan
People Hospitalized
0
57,103
People Tested
0
669,982
Mexico
© 2020 Mapbox © OpenStreetMap

Summary

Still more conclusions needed to be drawn from the entire project. The GBM Multinomial model provided exceptional results based on time constraints with 99.81% R-Square, 67.69% MSE, and 82.16% RSME. Collaboration should have been the focus but due to external circumstance, communication time was limited (i.e. corona). The COVID19-Project repo will be updated with the final project and results. As an investor and customer, I would politely pass on funding this product/project.