# Pandemia: A COVID-19 Predictor

Alaukika Diwanji
*San Jose State University*
014547013
alaukika.diwanji@sjsu.edu

Amit Garg
*San Jose State University*
014541072
amit.garg@sjsu.edu

Nachiket Trivedi
*San Jose State University*
014544933
nachiket.trivedi@sjsu.edu

***Abstract*—In this project we present and address two problem statements which are one of the most important and have been one of the gravest concerns in the post COVID-19 outbreak world. In our first challenge we try to predict the global turnout of the pandemic pertaining to each and every country, for certain range of dates in April and May, specific to the parameters of confirmed cases and fatalities. In our second challenge, we try to predict the same outcomes, for the same period, but for each individual state and territory of the United States. We evaluate and discuss our outcomes, present different implementations and the reasoning behind the chosen approaches, accompanied by a graphical representation of our results and an apt conclusion.**

***Index Terms*—COVID-19, Prediction, Traditional Machine Learning, Deep Learning, LSTM, Dense Neural Network, Multiple Linear Regression, Support Vector Regression.**

## I. Introduction

Coronavirus disease 2019 (COVID-19) is an infectious disease caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). The disease was first identified in December 2019 in Wuhan, the capital of China's Hubei province, and has since spread globally. As of 29 April 2020, more than 3.17 million cases have been reported all around the world, resulting in more than 224,000 deaths. (Source: WHO)

We're living through a global crisis whose outreach and impact is going to be insurmountable. A designated pandemic, COVID-19 has affected every single individual in one way or the other. Along with such massive influence, there's also a looming clout of uncertainty, whose mitigation itself might prove useful. This was the major motivation for us to take this as our final project for the course CMPE 256 Large Scale Analytics. We intend to be a part of the huge global helping community to assist in whichever way we can, and hence for the same, we participated in Kaggle's COVID-19 Global Forecasting Competition. We plan to use the skills we learned in our coursework to formulate a prediction for the month of April and May of 2020 which might hopefully shed some light on this uncertainty, and thereby do our 'good-part' in this crisis.

In this project, we intend to target two specific outcomes. The first one being a global predictor for each individual country (explained in detail in Section II). This was the problem statement of the aforementioned Kaggle competition, where we made a successful submission too. The second one, which we composed specifically for the coursework, is a COVID-19 predictor for all the American states and territories (explained in detail in Section III). We try our best to make our predictions as accurate as possible and discuss our findings in a conclusion at the end of this report.

## II. COVID-19 Global Prediction

### A. Goal

The primary goal for this section is to predict confirmed cases as well as fatalities pertaining to COVID-19 for all the countries, starting from April 1st to May 14th, 2020. We shall take into consideration various datasets that can be fed to our machine learning/deep learning models for training, which in turn will help make our predictions.

### B. Pre-processing

For accomplishing the intended goal of predicting the cases related to COVID-19, we'll be needing a large amount of data to begin with and preprocess it. We've divided this into 3 substeps: Data Gathering, Data Clean-up and Data Editing.

*1) Data Gathering:* We gathered 5 different datasets for consideration from various sources, primary one being Kaggle, which in turn referenced data from Johns Hopkins University, WHO, CDC. The datasets and their brief content outline are as follows:

1) train.csv
   Source: Kaggle
   Content: Id, Province_State, Country_Region, Date, ConfirmedCases, Fatalities
2) covid19countryInfo.csv
   Source: Kaggle
   Content: https://www.kaggle.com/koryto/countryinfo
3) share-of-adults-who-smoke.csv
   Source: Kaggle
   Content: Entity, Code, Year, Smoking prevalence, total (ages 15+) (
4) WPP2019_PopulationByAgeSex_Medium.csv
   Source: Kaggle
   Content: LocID, Location, VarID, Variant, Time, MidPeriod, AgeGrp, AgeGrpStart, AgeGrpSpan, PopMale, PopFemale, PopTotal

*2) Data Clean-up:*

- In this step, we'll be using the mentioned datasets and merging them to form a single tangible dataset such that it contains all the information that is bound to be useful to us in the subsequent steps, at the same time omitting all the clutter which is either redundant or not useful to us in training our model.
- We'll use the covid19countryInfo, share-of-adults-who-smoke, WPP2019_PopulationByAgeSex_Medium datasets as our primary datasets, cleanse and merge them as specified in order to form our final dataset of this step- enhanced.csv
- After removing redundant fields and merging the three datasets, our new dataset, enhanced.csv contains these fields: Country, Date, restrictions, quarantine, schools, hospibed(hospital bed availability per 1000 people), lung(death rate due to lung disease per 100k people), total_pop, density(population density), age_65+(proportion of people aged more than 65 years), smokers_perc.

*3) Data Editing:* After the data clean-up step, we're left with two datasets: train.csv, provided by Kaggle, and enhanced.csv, generated by processing and merging multiple datasets in the clean-up step. We'll have to process these two datasets further such that to make them tangible for our model training. We used the following approach to do the same:

- The train.csv dataset contains fields such as State, Country, Date, Confirmed Cases and Fatalities; while the enhanced.csv dataset contains fields viz. Country, Date, restrictions, quarantine, schools, hospibed(hospital bed availability per 1000 people), lung(death rate due to lung disease per 100k people), total_pop, density(population density), age_65+(proportion of people aged more than 65 years), smokers_perc. These are all the metrics we considered for training our model and thereby effective predicting our output.
- First of all, we merge these two datasets into a single dataframe. Now, we'll bifurcate this data into parts: (1) which shows no dependency on date, like demographic factors such as population, and (2) set which does show a dependency with date, like past infections, fatalities, quarantine days, schools closed days etc.
- Making the final dataframe: We put all the data pertaining to (1) of the previous point in an array called demographic_inputs, and the data pertaining to (2) in an array called temporal_inputs, and generated these arrays for each respective date.
- Taking care of null values and excessive zeroes: In the newly formed final dataframe, there are certain values which are zeroes for a larger span of the set. These zeroes, though redundant, will later skew our model for the wrong, hence we delimited to a predefined count, to maintain balance while model training. In addition to that, there were certain null values in the dataset, whose field values were unknown. We replaced these nulls with the mean of all the other values that comprise the field.
- Making the training and testing data: We'll use the first 90% of our data as the training data as well as the remaining 10% as out testing data, and will use the generated training data to train our model. For model training, the starting set of the training data, i.e. our x values, will comprise the aforementioned formations of demographic and temporal sets. Our target points for model training, i.e. our y values, will be the confirmed cases and the fatalities. With this, we're ready to train our model.

### *C. System Design and Implementation: Model Architecture*

Our final data consists of two major sections: demographic data: which consists of fields that aren't date dependent, and temporal data: which consists of fields dependent on date. As our temporal data changes with the date, it is bound to be sequential and hence cannot be fed to a model which doesn't consider sequentiality. This means we cannot use a dense neural network for this data, and hence to combat this, we'll use LSTM for temporal data.

On the other hand, our demographic data doesn't have any sort of dependency on dates, and hence can be fed into a dense neural network model. We use the library provided by keras to accomplish the said goals.

*1) Dense Neural Network:*

- We used the dense neural network model for the demographic data for training. The input of the model was the demographic part of the training dataset, while the target for the model were the y values of confirmed cases and fatalities from the training dataset.
- The demographic set contains 5 fields, which'll be then passed to the network. The network comprises a mono layered structure with 16 neurons. Each single neuron will receive all the demographic values as input and train based on the values. After processing the dense neural network, each neuron will've set and determined a weight, which will hence be our model itself.

*2) LSTM:*

- LSTM(Long short term memory) are a type of recurrent neural networks used in deep learning, which primarily focuses on feedback connections. As our temporal data is data dependent, we cannot entertain the option of feeding all the data to each neuron irrespective of its order, and hence for this data, the dense neural network renders useless.
- Hence to preserve sequentiality, and to use the property for enhancing the model, we use the LSTM recurrent neural network.
- The input will be the temporal subset of our training dataset, while our target will be the y values of confirmed cases and fatalities. We considered the dimensionality of

the output space as 64, and for a linear transformation of the recurrent state we opted 0.25 as the fraction of the units to drop.

Model: "model_3"

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| input_7 (InputLayer) | [(None, 4, 4)] | 0 | |
| lstm_11 (LSTM) | (None, 4, 64) | 17664 | input_7[0][0] |
| input_8 (InputLayer) | [(None, 6)] | 0 | |
| lstm_12 (LSTM) | (None, 4, 64) | 33024 | lstm_11[0][0] |
| dense_9 (Dense) | (None, 16) | 112 | input_8[0][0] |
| lstm_13 (LSTM) | (None, 32) | 12416 | lstm_12[0][0] |
| dropout_9 (Dropout) | (None, 16) | 0 | dense_9[0][0] |
| lstm_14 (LSTM) | (None, 32) | 12416 | lstm_12[0][0] |
| concatenate_6 (Concatenate) | (None, 48) | 0 | lstm_13[0][0]<br>dropout_9[0][0] |
| concatenate_7 (Concatenate) | (None, 48) | 0 | lstm_14[0][0]<br>dropout_9[0][0] |
| dense_10 (Dense) | (None, 128) | 6272 | concatenate_6[0][0] |
| dense_11 (Dense) | (None, 128) | 6272 | concatenate_7[0][0] |
| dropout_10 (Dropout) | (None, 128) | 0 | dense_10[0][0] |
| dropout_11 (Dropout) | (None, 128) | 0 | dense_11[0][0] |
| cases (Dense) | (None, 1) | 129 | dropout_10[0][0] |
| fatalities (Dense) | (None, 1) | 129 | dropout_11[0][0] |

Total params: 88,434
Trainable params: 88,434
Non-trainable params: 0

Fig. 1. LSTM model architecture

### D. System Design and Implementation: Model Training

- After defining our model architecture, we've now formulated our model, ready for training. The training parameters consist of the demographic data as well as temporal data, with 90% of confirmed cases and fatalities being our target fields, and the remainder of the 10% of the whole dataset acting as our testing data.
- We train our model for 250 epochs, for minimizing loss and thereby increasing the performance metric. The model will continue to be executed for 250 times, and will keep on enhancing itself.
- We formulated the correlation of epochs with loss for this scenario and displayed the same in the form of a graph:

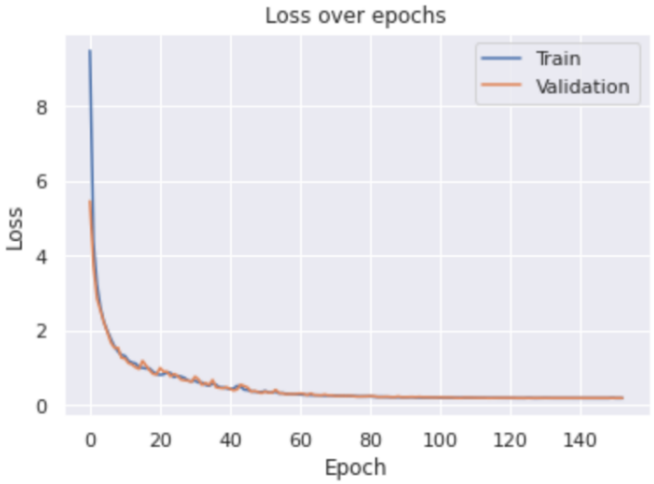


Fig. 2. Correlation of loss with respect to epoch

- This delineates the slump in loss when the epoch is increased, thereby justifying our use of 250 epochs.
- Henceforth, our model is ready for prediction.

### E. Prediction

- We'll use our trained model on our testing data, and predict the casualties and fatalities from 17$^{th}$ April, 2020 to 15$^{th}$ May, 2020, for every country.
- The output we get in this step is our final output, which successfully satisfies our intended goal.
- We dumped the output of this step into a csv file, and submitted the same for the Kaggle COVID-19 Global Forecasting (Week 4). On the private leader board, we received a rank of 259 out of 313 total participants.

### F. Experiments

For global prediction we considered past data trends for each data row for enhancing our results. Taking the record of past x days, we get more accurate predictions for the next date. For this, we considered and ran our algorithm for 5, 7, 10, 13 days and calculated the RMSE for Confirmed cases as well as fatalities. As we decreased our days, our training size increased, and hence we get the best results for past trend day set to 5. Hence we consider 5 during our model training. The figure describes our results.

| Index | Past Trend Days | Train Size | Test Size | Confirmed RMSE | Deaths RMSE |
|---|---|---|---|---|---|
| 1 | 10 | 1810 | 648 | 23525.30 | 2309.61 |
| 2 | 13 | 1621 | 331 | 24652.91 | 2463.53 |
| 3 | 7 | 3118 | 976 | 21248.53 | 2090.36 |
| 4 | 5 | 4920 | 1933 | 19989.60 | 1932.21 |

Fig. 3. RMSE values of confirmed cases and deaths corresponding with past trend days value

## III. COVID-19 US States Prediction

### A. Goal

The primary goal for this section is to predict confirmed cases as well as fatalities pertaining to COVID-19 for all the states and territories of the United States, starting from April 1st to May 14th, 2020. We shall take into consideration various datasets that can be fed to our machine learning/deep learning models for training, which in turn will help make our predictions

### B. Pre-processing

For accomplishing the intended goal of predicting the cases related to COVID-19 for each state and territory of the United States, we'll be needing a large amount of data to begin with and preprocess it. We've divided this into 3 substeps: Data Gathering, Data Clean-up and Data Editing.

*1) Data Gathering:* We gathered 3 different datasets for consideration from various sources, primary one being Kaggle, which in turn referenced data from Johns Hopkins University, WHO, CDC. The datasets and their brief content outline are as follows:

1) train.csv
   Source: Kaggle
   Content: Id, Province_State, Country_Region, Date, ConfirmedCases, Fatalities.
2) USA-COVID19LockdownData.csv
   Source: CDC, Forbes, The New York Times
   Content: 'Date_index', 'State','Confirmed', 'Deaths', 'Cumulative Confirmed', 'Cumulative Deaths', 'State of Emergency Declared', 'Stay at home ordered', 'Gatherings banned', 'Out-of-state Travel Restrictions', 'Schools closed', 'Daycares Closed', 'Bars and Restaurants Closed', 'Non-essential retails closed'.
3) us-states-demographic.csv
   Source: Kaggle, US govt. census
   Content: Latitude, Longitude, Total Population, Area, Population Density

*2) Data Clean-up:*

- In this step, we'll be using the mentioned datasets and merging them to form a single tangible dataset such that it contains all the information that is bound to be useful to us in the subsequent steps, at the same time omitting all the clutter which is either redundant or not useful to us in training our model.
- We'll use the USA-COVID19LockdownData and us-states-demographic as our primary datasets, cleanse and merge them with the train.csv in order to form our final dataset of this step- final_dataset.csv
- We found and formulated the USA-COVID19LockdownData.csv from sources: CDC, Forbes and informative analysis of The new York Times. The data gathered also contained details of other countries, which filtered it for the US, and got state specific values. The field values were majorly macro and in form of dates, like for example stay at home ordered in the state on Arkansas on the date 24th March. We converted such macro elements to numerical format by issuing them a number by correlating it with state of emergency declared date for each respective state.
- We considered the demographic elements like population, area, population density, latitude and longitude for each state and territory from the last U.S. Government Census of 2010. We then merged this data with the previous dataset. We also merged these with the American specifc values of the train.csv, thereby generating our final dataset (final_dataset.csv), which contains the following parameters: *Date_index, State, Confirmed, Deaths, Cumulative Confirmed, Cumulative Deaths, State of Emergency Declared, Stay at home ordered, Gatherings banned, Out-of-state Travel Restrictions, Schools closed, Daycares Closed, Bars and Restaurants Closed, Non-essential retails closed, Lat, Long, Population, Area, Density*.

*3) Data Editing:*

- After the clean up step, we're left with a single dataset, the final_dataset.csv, our final one. Each row will be date and state specific, containing parameters mentioned above. In the next steps, we'll be applying machine learning models on this dataset but before we do that, we'll modify this such that to increase our accuracy of prediction.
- For every row entry, if there were existing data of previous records, we would have a prediction trend which can help us during prediction. The parameters necessary for this step being deaths and confirmed cases.
- Hence, to enforce this, we grouped deaths and confirmed cases of previous three row entries, and appended it with the same row. For example: if considering row entry of any state for date x, we grouped the deaths and confirmed cases pertaining to dates x-1, x-2 and x-3, and appended the same with the considered row.
- Our dataset is now good for the next step.
- Making the training and testing data: The data we considered starts from January 22$^{nd}$ and stretches till April 17$^{th}$, for each US state and territory. That marks a total of 87 days. We considered the first 70 days i.e. till March 31$^{st}$ as our training dataset and the subsequent 17 dates till April 17$^{th}$ as our testing data. We can now proceed to model training.

### *C. System Design and Implementation: Model Architecture*

This section delineates the models we considered as well as the reasoning behind it.
Our training data, as well as our testing data both will have x values and y values. x values, being the input to the models and y being the target. The y values we considered are deaths and the confirmed cases, while all the other parameters being the x values.

*1) Multiple Linear Regression:*

- The first model that we used was multiple linear regression. We use multiple linear regression when our x values are multi parametered. As here we have multiple fields such as state at home ordered, schools closed etc., we can aptly use this model.
- In linear regression, we plot each data entry on a plane, with the output corresponding to the y values. Then a line is plotted such that the distance from each point is minimum with the line.
- The line must always be linear and shouldn't have polynomial powers. The equation of the line can be used to determine y values corresponding to new x values.

*2) Support Vector Regression (SVR):*

- SVR, being a type of Support Vector Machine maps the x values of the training set on a plane of it's respective dimensionality.
- Each data entry will act as a vector, on the high dimensional plane. The model will form a plane on this called the hyperplane such that the extremities of each vectors has a maximum distance from the plane.
- We use SVR when the data entries have high similarities. In our case, as each data rows doesn't vary to a larger extent, SVR implementation seems apt.

### *D. System Design and Implementation: Model Training*

- As formulated above, we considered two traditional machine learning models: Multiple Linear Regression and Support Vector Regression. For each state and union territory of the United States, we trained both the models, and opted the one presenting better results.
- During SVR, there might arise a situation when the datapoints on the dimensionality plane intersect a lot, which makes it difficult for the dividing hyperplane to form. In this case, one group can be raised such that the hyperplane can pass from between, thereby rendering the division feasible.
- This scenario can be handled by setting RBF Kernel in the SVR model. We implemented the same, only to find out that the aforementioned situation doesn't happen in our case. For that reason, we used our RBF kernel as linear.

### *E. Prediction*

- We'll use our trained model on our testing data, and predict the casualties and fatalities from 17$^{th}$ April, 2020 to 15$^{th}$ May, 2020, for every country.
- The output we get in this step is our final output, which successfully satisfies our intended goal, and dump the same into a csv file, which presents the final result of this sub-part of the project.

### *F. Experiments*

For US states', prediction we considered past data trends for each data row for enhancing our results. We exercised this for both the models: Multiple Linear Regression and SVR. Taking the record of past x days, we get more accurate predictions for the next date. For this, we considered and ran both our models for 3, 5, 7, 10 days and calculated the RMSE for confirmed cases as well as fatalities, for the both of those. As we decreased our days, our training size increased, and hence we get the best results for past trend day set to 3. Hence we consider 3 during our model training. The following table describes our results.

| Index | Past Trend Days | Train Size | Test Size | Confirmed RMSE(Mul. Linear) | Deaths Avg. RMSE(Mul. Linear) | Confirmed Avg. RMSE(SVR) | Deaths Avg. RMSE(SVR) |
|---|---|---|---|---|---|---|---|
| 1 | 3 | 67 | 17 | 3390.04 | 86.50 | 2457.02 | 176.69 |
| 2 | 5 | 67 | 17 | 8012.93 | 238.22 | 3063.87 | 221.42 |
| 3 | 7 | 67 | 17 | 16201.31 | 239.85 | 3441.01 | 135.42 |
| 4 | 10 | 67 | 17 | 922325.66 | 4787.45 | 7514.03 | 256.88 |

Fig. 4. RMSE values of confirmed cases and deaths corresponding with past trend days value, for Multiple Linear Regression and SVR.

We trained both our models for each states, and predicted the output(confirmed cases and fatalities) for the the dates same as the dates in our testing dataset, i.e. from 1$^{st}$ April, 2020 to 17$^{th}$ April, 2020. As we already have the actual cases and fatalities information for the said dates, we can get a good idea by mapping our predicted values to the actual values. In this step, we obtained some interesting results, where in majority of states, there seems a high similarity. We've presented the predicted confirmed cases as well as deaths of four of our best outcomes in the trailing figure. The states considered here are California, Texas, Pennsylvania and South Dakota. The red lines indicate the actual data while the blue depicts predicted outcomes.

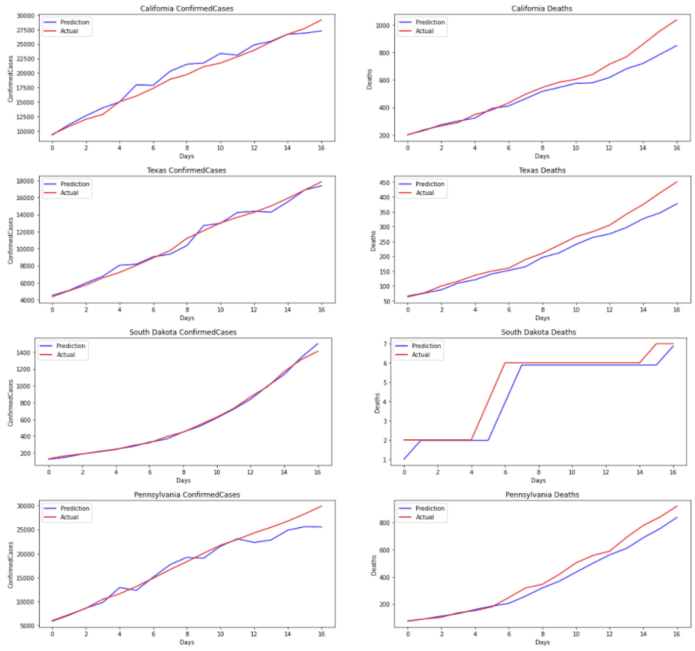


Fig. 5. Comparison of actual confirmed cases and fatalities with predicted confirmed cases and fatalities for the states California, Texas, Pennsylvania and South Dakota.

As we obtained our model, as well as a fairly decent prediction for the testing dataset comprising dates from 1$^{st}$ April, 2020 to 17$^{th}$ April, 2020, we can use the same model, for predicting data for future. We did exactly the same. We used our established models for predicting confirmed cases as well as fatalities for the all states pertaining to dates starting 18$^{th}$

April, 2020 to 15th May, 2020. The following figure represents the predicted values for California and Texas.

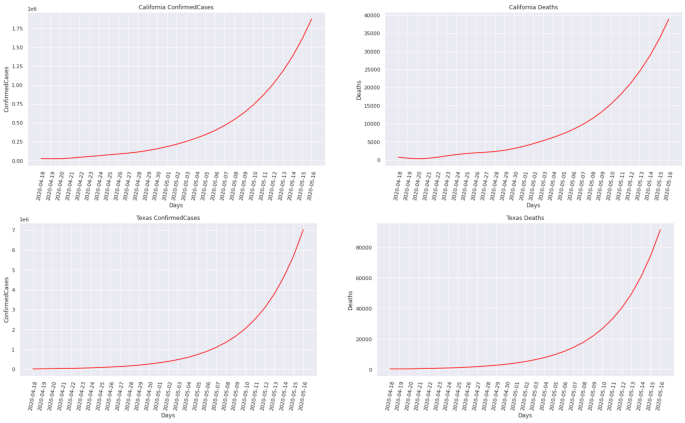


Fig. 6. Prediction of Confirmed Cases and Deaths for California and Texas pertaining to the dates starting 18th April, 2020 to 15th May, 2020.

## Conclusion

In this project we tried to address two problem statements of grave importance in this post-covid-19 world. First being the prediction of confirmed cases and fatalities for each countries, which was also our Kaggle COVID-19 Global Forecasting competition submission. The second challenge we addressed was prediction of confirmed cases and fatalities for each US states and territories. The predicted values were specific to dates from 17th April, 2020 to 15th May, 2020. We got some interesting results in both the predictions, and we displayed the same in a graphical presentation.

However, we did face certain difficulties while completing this project. We generated a fairly large dataset for our global predictor because of the availability of data for all the countries and their states. This allowed us to use deep learning models like LSTM and Dense Neural Network. However in the state-wise predictor, the available data was pertaining to only the 50 states and even less territories, which renders the dataset size less than the one computed for global predictor. We tried applying the aforementioned deep learning models here also, but due to the compact size of the dataset, we weren't able to produce good results. Furthermore, traditional machine learning algorithms: Random Forest and Decision Trees didn't display a decent convergence of loss, hence we used machine learning models like multiple linear regression and SVR, which in turn gave us commendable results.

In our future work, we intend to apply diligent and calculated domain knowledge to gather a large amount of data (specific to each US states and territories), and thereby apply deep learning models to get better results in the state-wise prediction.

## Task Distribution

1) Alaukika Diwanji: Data exploration and data preparation of global and US states' forecast, traditional machine learning like Support Vector Regression for US states' forecast, report and PPT.
2) Amit Garg: Global and US states' model architecture which includes LSTM, Dense Neural Network etc., traditional machine learning algorithm like Random Forest for US states' forecast and evaluation metrics.
3) Nachiket Trivedi: Data Pre-processing for global forecast as well US states' forecast, traditional machine learning algorithms like Decision Tree and Multiple Linear Regression for US states' forecast.

## Acknowledgment

The project couldn't have been possible without the unwavering guidance and novel lectures of Prof. Magdalini Eirinaki for the course CMPE 256, Large Scale Analytics. We humbly thank the Professor for the same.