ANALYSIS AND PREDICTION OF THE SPREAD OF COVID-19 IN INDIA

submitted as a part of partial fulfillment of the course

Machine Learning (BITS F464)

Prepared By

Arohi Dureja	**(2017B3A70531G)**
Siddharth Sharma	**(2018A7PS0199G)**
Shreyas Mali	**(2018A7PS0239G)**
Arjun Bajpai	**(2018A7PS0182G)**

INDEX

ACKNOWLEDGEMENT

We would like to express our sincerest gratitude to Prof. Ashwin Srinivasan and Prof. Tirtharaj Dash for giving us this opportunity which helped us discover the topic with deep insights. The constant motivation to explore new ideas, guidance and valuable inputs for our work helped us to learn many new concepts. Moreover, working on this was tremendous exposure to the real-world applications and will surely help with our future ideas and research works in the field.

We would like to acknowledge our team members and our classmates who have provided extensive personal and professional guidance both in our economic research and life in general.

ABSTRACT

Since the beginning of Coronavirus disease i.e. COVID-19 there have been several papers on predicting the intensity of the pandemic with varying degrees of accuracy. These predictions help the government in planning the required hospital resources in the future. This project is one such attempt to do an information-led short-term projection of the number of new cases and deaths coming per day in India and their implications in the future using models such as LSTM, ARIMA and Random Forest model.We have used raw data set in time series frame from 15th February, 2020 to 31st dec 2020. A data-driven forecasting method has been used to approximate the new cases coming per day and the new deaths for the subsequent days. The LSTM model gave very promising results compared to the other model. We were able to achieve predictions with approximately 9% error from the expected outcomes.

INTRODUCTION

From Dec 2019 , when the first series of pneumonia cases were confirmed in Wuhan, that was later named coronavirus disease 2019 (COVID-19). This virus has spread very quickly around the globe infecting millions of people and killing millions. Countries were forced to take extreme steps such as complete lockdown, quarantine, stop international flights, preventing movement to save lives. Moreover, the economic and social impact of this virus is far more disastrous for developing and under-developed countries.

India reported its first case of COVID-19 on January 30,2020 which initially grew very slowly till 100th case on March 14, 2020 but thereafter the cases increased steadily making the situation tough to control.

India's government has proposed and implemented several lockdowns over this one year to control the situation. Lockdown was successful in controlling situations all over the country with some delay as lockdown does not have immediate effect. But lockdown cannot be continued forever as it impacts the economy drastically. So, a more practical solution is quarantining the critical zones and people infected.

Now that people are being vaccinated it has changed the situation but the coming of the second wave has changed everything for the country.

Since the beginning of this pandemic, people have been trying to make mathematical and statistical models that could predict the global and national with varied accuracy and reliability. The reliability depends on the assumptions, values and input features taken into account. As time passed, the quality of data available has improved and this has helped to build better models which can predict with higher accuracy. Also, as pathogens behaviour cannot be controlled so very high accuracy cannot be achieved through these models.

If we have an idea about the future it becomes easier for the system to prepare all kinds of resources that will be needed to control the situation, this makes predicting of great importance.

Here we are proposing machine learning-based models for predicting the number of new cases approximately for 10 days ahead for India. We have used a long short-term memory (LSTM) based model, ARIMA model and Random Forest model. We have tested the working of these 3 models to decide which predicts better.

RELATED WORK

The literature on predicting the COVID-19 cases in India is limited as the database is still building and as of now the available data is limited. Still there is a good amount of research available for different countries.

Arora[1] has used RNN based LSTM models like stacked LSTM and Bi-LSTM which have given less that 3% error, and they have predicted cases for some states individually. According to their statistical research Bi-LSTM gives the best results while convolutional LSTM gives the worst results.

Siami-Namini[2] has done a comparative study on ARIMA, LSTM and Bi-LSTM models for the stock market prices, which has revealed that Bi-LSTM models give better predictions compared to LSTM and ARIMA models. They have also concluded that Bi-LSTM models reach the equilibrium much slower than LSTM based models.

Arti[3] In his paper has discussed the problems faced in modelling and predicting COVID-19 cases in India. He has also emphasized on the importance of lockdown and isolation in controlling the situation. In the paper a tree based model is used to predict the cases and determine the effect of lockdown.

[1] "Prediction and analysis of COVID-19 positive ... - ScienceDirect.com." https://www.sciencedirect.com/science/article/pii/S096007792030415X. Accessed 30 Apr. 2021.
[2] "A Comparative Analysis of Forecasting Financial Time Series Using ...." 21 Nov. 2019, https://arxiv.org/abs/1911.09512. Accessed 30 Apr. 2021.
[3] "Modeling and Predictions for COVID 19 Spread in ... - ResearchGate." https://www.researchgate.net/profile/Arti-Mk/publication/340362418_Modeling_and_Predictions_for_COVID_19_Spread_in_India/links/5e84d25f92851c2f52735ad0/Modeling-and-Predictions-for-COVID-19-Spread-in-India.pdf. Accessed 30 Apr. 2021.

DATASET

DATA COLLECTION

We have looked at related papers and understood that time series features affecting the number of covid cases are divided into 3 parts - **testing & vaccination, public policy implementation, mobility data**[4][5]. The first section includes the number of tests being done per day which is a very important factor to determine the number of new cases confirmed.Number of people vaccinated is a lagged feature as it takes time to show its effect (around 8 weeks). Public policy implementation refers to the various restrictions imposed like no public gatherings, no internal movement, stay at home, international travel control, etc., market closing, school closing, fiscal measures, economic measures and healthcare measures like testing policies, contact tracing , protection of elderly people, etc. Other features like Stringency index, Government response index and Economic support index are also part of the public policy implementations. Mobility data covers restrictions on market, public places, parks, grocery and pharmacy, interstate transport, etc. Output features include number of cases confirmed and number of deaths. Data for all these features were available for 321 days starting from 15th February, 2020. Using this data to train our model and predict for the next 10 days.

DATA PRE-PROCESSING

The collected data is put together as raw data. Initial steps of preprocessing the data have been done manually on excel which required referring to different sources to cross check the daily values and filling in the few values which were missing in a particular source. Further, in the initial period of 15th Feb, 2020 to 15th March, 2020 most of the public policy features were not implemented so we had to handle their values appropriately. Feature naming was also made consistent to make the dataset easily understandable. This dataset is available for reference under the name “Processed data”.

We had taken a lot of features according to our understanding and what was referred to as important features in other related works. To understand the dependence of our output feature i.e *new_cases_smoothed* and *new_deaths_smoothed* on the input features correlation values have

[4] "Coronavirus Pandemic (COVID-19 ...." https://ourworldindata.org/coronavirus. Accessed 30 Apr. 2021.
[5] "• India: COVID-19 tests per day | Statista." https://www.statista.com/statistics/1107370/india-coronavirus-covid-19-tests-per-day/. Accessed 30 Apr. 2021.

been calculated. This shows us that 8 features were such that they had NaN correlation with our output. So, we dropped there features namely *new_vaccinations*, *new_vaccinations_smoothed*, *new_vaccinations_smoothed_per_million*, *international_travel_control*, *public_information_campaigns* , *testing_policy*, *contact_tracing*, *vaccination_policy*, this is reliable also as the vaccination drive had not begun in the time period we have taken so it's value is constant and will not affect the *new_cases_smoothed*.

We also decided that in our model we will be predicting the *new_cases_smoothed* and *new_deaths_smoothed* so we removed the other features which were kept on the output side namely *total_cases, new_cases, total_deaths, new_deaths, total_cases_per_million, new_cases_per_million, new_cases_smoothed_per_million, total_deaths_per_million, new_deaths_per_million, new_deaths_smoothed_per_million*.

Further we observed through scatterplots that some features appear to be constant over time with very few instances different from that. Such features are expected to have very little to no impact on our output feature. These are also those features which showed very small correlation with *new_cases_smoothed* in our previous step. So, we have dropped these features too, namely *international_support*, *emergency_investment_in_healthcare*, *investment_in_vaccines*.

Now the only aspect left to check is if the features in itself have high correlation, very highly correlated features would mean their individual impact on the *new_cases_smoothed* is not very different and hence considering one out of the two highly correlated features should be enough. We have calculated a correlation matrix of all the features against all of them to check which of them are highly correlated. On doing so we have found that some of the features had correlation > 0.9. We have selected one of these pairs of highly correlated feature, so we have dropped the features namely new_tests, *new_tests_smoothed_per_thousand*, *tests_per_case*, *StringencyLegacyIndex*, *ContainmentHealthIndex*, *GovernmentResponseIndex*. Removing all these features has helped us ensure that the features now available in our dataset are only those which are independent and help us predict the output in a clear way without any overfitting. Finally while giving input to the model we have also dropped the *date* column. Scatterplots of all the features are available in our code file for better understanding of the input features and their behaviour with respect to time.

Fig 1 : Correlation matrix for all features

METHODOLOGY

I. ARIMA MODEL

ARIMA model is a very popular and widely used statistical method for time series forecasting. ARIMA stands for **AutoRegressive Integrated Moving Averag**e. It is a class of models that captures a suite of different standard temporal structures in time series data.
It is a generalization of the simpler AutoRegressive Moving Average. It explicitly caters to a suite of standard structures in time series data, and as such provides a simple yet powerful method for making skillful time series forecasts.
ARIMA is Auto Regressive — (AR)
Past time points of time series data can impact current and future time points. ARIMA models take this concept into account when forecasting current and future values. ARIMA uses a number of lagged observations of time series to forecast observations. A weight is applied to each of the past terms and the weights can vary based on how recent they are.
AR(x) means x lagged error terms are going to be used in the ARIMA model.

ARIMA is Integrated — (I)
If a trend exists then time series is considered non stationary and shows seasonality. Integrated is a property that reduces seasonality from a time series. ARIMA models have a degree of differencing which eliminates seasonality. D property of ARIMA represents degree of differencing.

ARIMA is Moving Average — (MA)
Error terms of previous time points are used to predict current and future point's observation. Moving average (MA) removes non-determinism or random movements from a time series. The property Q represents Moving Average in ARIMA [6].

Our model uses new_cases features for prediction purpose and uses data over a period of 10 months for predicting results for the next 10 days.

[6] "How to Create an ARIMA Model for Time ...." 9 Jan. 2017, https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/. Accessed 30 Apr. 2021.

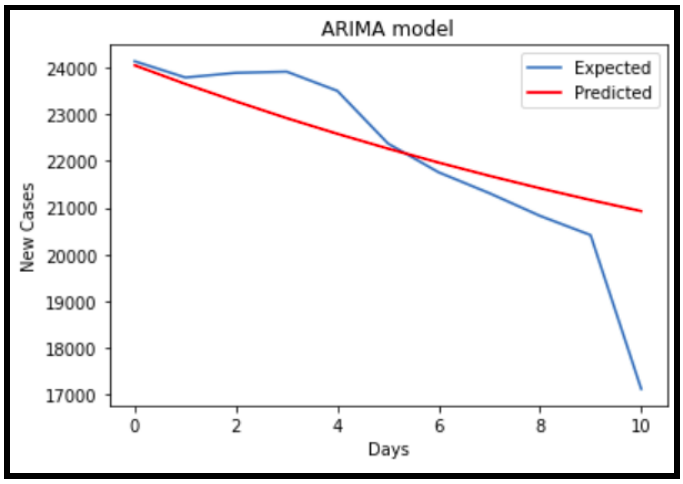


Fig 2 : Cross Validation (ARIMA Model)

Mean Absolute Error = 1273 cases

Although this result looks considerably good, there are few reasons why this value could give us a false impression about the correctness of our model. For example, this model fails to predict any ups and downs in the actual values and simply describes values with almost constant slope. This is because of the fact that our model is simply learning on patterns of new_cases and does not consider any other factor into account. Another reason is that if the patterns after five days are considerably different than the ones at the beginning of the prediction then this model will fail to predict them. This same fact is evident for the less difference between predicted and expected values at the beginning of the prediction as compared to later days.

II. RANDOM FOREST MODEL

Data science provides a plethora of regression algorithms such as linear regression, logistic regression, support vector machines and decision trees. But near the top of the classifier hierarchy is the random forest regressor. The Decision Tree algorithm has a major disadvantage in that it causes overfitting. Random Forest algorithm being very fast and robust than other regression models also helps in limiting over-fitting problems. In short, The Random Forest Algorithm merges the output of multiple Decision Trees to generate the final output.

Random forest algorithm belongs to supervised learning technique which can be used for both classification and regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of a model.

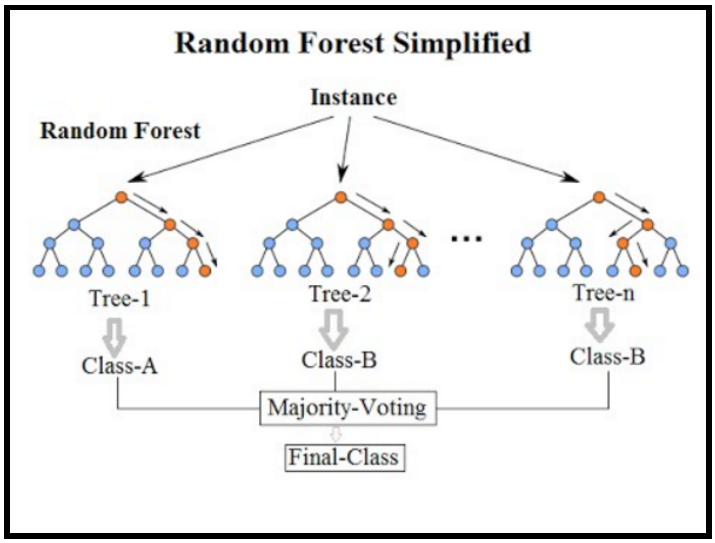


Fig 3 : Random Forest

Our problem statement urges us to predict for more than 1 time instances in the future based on present training data.

But, Random Forest can't predict for more than t+1 right?

Yes. We can't use Random Forest to forecast into more than 1 value or indefinitely into the future as one can with the linear model. So, when we are trying to predict for 1 month ahead, we'll need previous input data of at least 1 month for training purposes.

Thus, we can use it for time series forecasting, although it requires a time series dataset to be transformed into a supervised learning problem first . It also requires a specialized technique for evaluating the model called ***walk-forward validation***, as the *k-fold cross validation* model would result in optimistically biased results.

Now, Given a sequence of numbers for a time series dataset, we can restructure the data to look like a supervised learning problem using ***sliding window*** representation. Furthermore, we not only want our model to be multivariate but also multistep and hence random forest model is run n number of times (where n is the number of time instances we want to predict in future) while adding a new row of predicted output to the current dataset[7].

Because of the multivariate nature of the problem, the only appropriate feature value in the model is the one which is forecasted from previous data and used as input for forecasting further outputs. The remaining features (total 21) either have to be kept the same as the last one or mapped to the appropriate time instances back.

Here, our model trains on the data from Feb 15th to Nov 30th and output results are forecasted for next 10 days such that data from 20th Nov is mapped to the data for 1st Dec, 21st Nov is mapped to the data for 2nd Dec and so on.

[7] "Multivariate Time Series Forecasting Using Random Forest | by ...." https://towardsdatascience.com/multivariate-time-series-forecasting-using-random-forest-2372f3ecbad1. Accessed 30 Apr. 2021.

Following is the graph denoting expected and predicted values of output feature

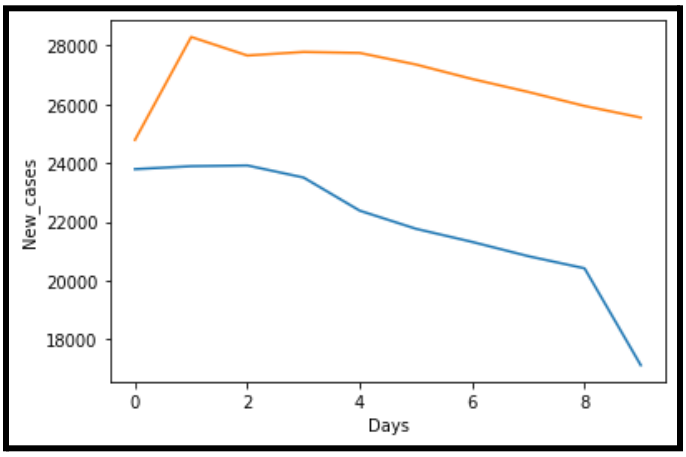


Fig 4 : Cross-Validation (Random Forest Model)

Mean Absolute Error = 4994 cases

The reason why this model couldn't improve beyond a certain point is that the 21 features which are used for prediction are merely the approximation of what could be actual features in the next 10 days. So any sudden spikes in those features may lead to drastic changes in the output which becomes very difficult to predict based on present data.

III. LONG SHORT-TERM MEMORY MODEL

Why LSTMs over RNNs

In Recurrent Neural Networks, we observe a problem called short-term memory, i.e., in case of long sequences, the RNN model will find it difficult to remember and retain the information learned in earlier time steps, for a very long time. This is because during back propagation, RNNs suffer from the Vanishing Gradient Problem, in which the gradients' values are reduced to tiny values, which further do not update the weights' values in the Neural Networks. These small gradient values furthermore do not contribute significantly to the learning process. This phenomenon can be observed generally in the earlier layers. Since these layers have stopped learning, RNNs start forgetting that information, and thus the short-term memory.

How can this short-term memory problem be solved? By using LSTM units/blocks in place of regular neurons in the RNN model. These LSTM blocks have internal mechanisms called gates that actually regulate the flow of information, i.e., which information to keep or forget and throw away. They also have memory that can store the recent and/or the older information. Most of the state-of-the-art results, which require the usage of RNNs, are achieved using LSTMs.

LSTM

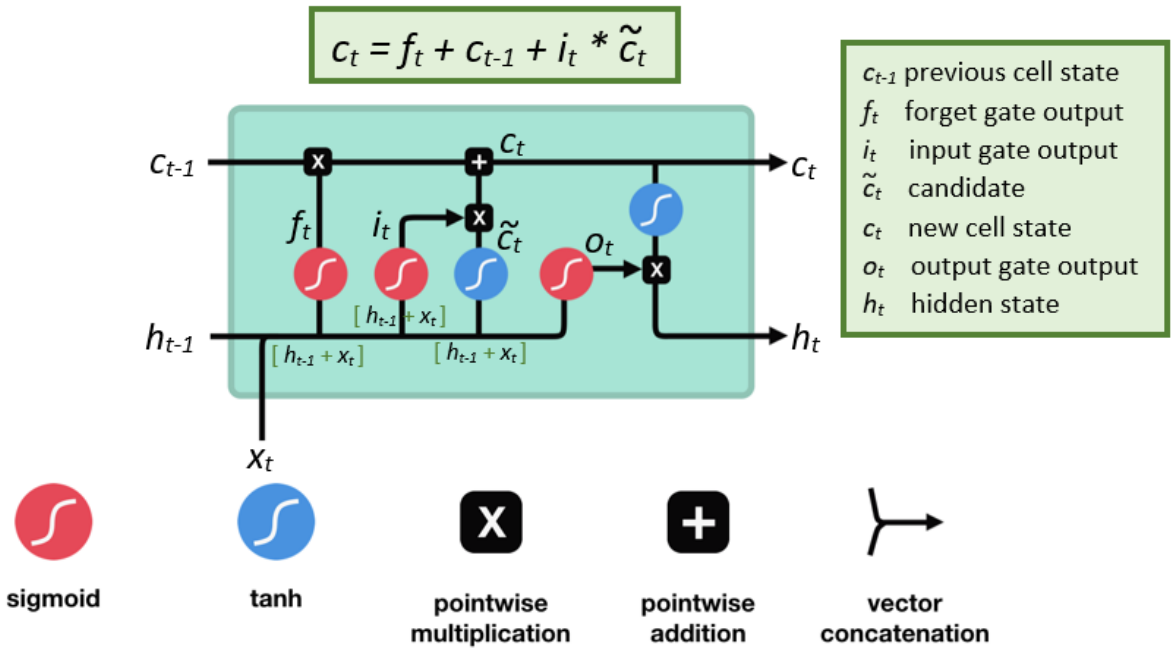


Fig 5: LSTM Unit/Block

- **Forget Gate:** This gate shall decide which information to keep or throw away. The hidden state from the previous cell, along with the information from the current input, are passed through the sigmoid function. If the output from this gate (f_t) is closer to 0, it means to forget it, and if it is closer to 1, it means to keep the information.

  Output from the Forget Gate: $f_t = \sigma([h_{t-1} + x_t])$
  where
  h_{t-1} is hidden state from previous cell
  x_t is current cell input
  [a + b] is vector concatenation of a and b, and
  σ is the sigmoid function

- **Input Gate:** This gate is to update the cell state. The output from this gate gives us new cell state value, which is computed using the below formula:
  $c_t = f_t + c_{t-1} + i_t * \check{c}_t$
  where
  $i_t = \sigma([h_{t-1} + x_t])$
  $\check{c}_t = \tanh([h_{t-1} + x_t])$

- **Output Gate:** The output gate decides what the next hidden state should be. The hidden state contains information about the previous inputs. This is also used for making the predictions. The hidden state is calculated using the below formula:
  $h_t = o_t * \tanh(c_t)$
  where
  $o_t = \sigma([h_{t-1} + x_t])$
  In short, the forget gate is responsible for deciding what all information is relevant from the prior steps. The input gate decides the relevant information to be added from the current input, and the output gate determines the next hidden state.

LSTM Model for our problem (Implementation)[8]

Building the LSTM: Our Model has an input layer, three hidden layers and an output layer that gives us the single predicted value of the number of new cases. The input layer as well as the all the hidden layers comprise of 32 LSTM blocks each. The output layer is of size 2 (for prediction of new cases as well as new deaths) and is the result of dense connection with the second last layer.

In addition to the above mentioned layers, we are also adding Dropout layers in which 10% of the layers will be dropped in random, in the course of learning. We are compiling the model using the adam optimizer and mean squared error function as our loss function.

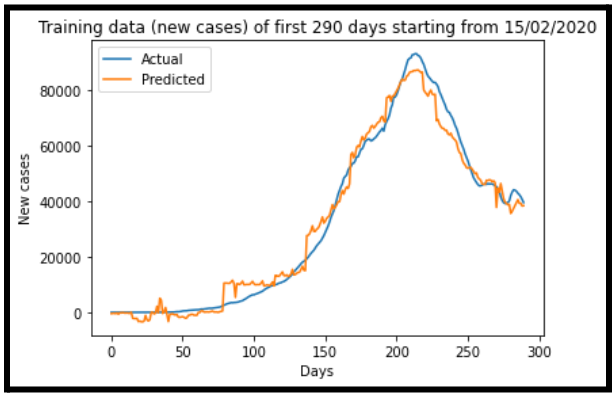
Fitting our data to the Model: We are fitting the model to run on 70 epochs with the batch size of 20.

[8] "Illustrated Guide to LSTM's and GRU's: A step by step explanation ...." https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21. Accessed 30 Apr. 2021.

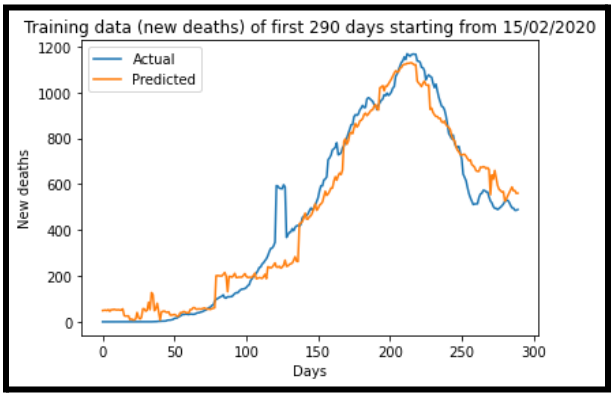
RESULTS

As can be seen from the below graphs, the predicted values are matching very well with the actual smoothed values, both for the number of new cases per day, as well as the number of death cases per day.

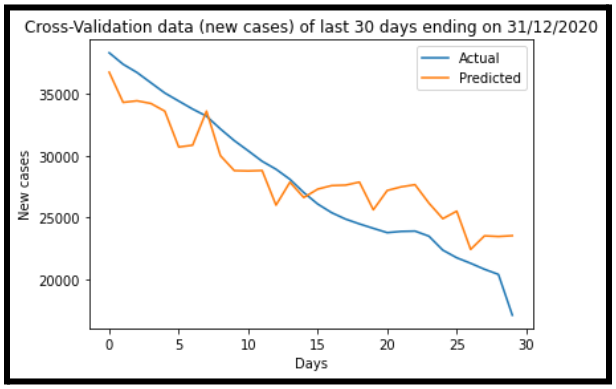
The performance of our model on both the train data as well as the test data is shown below: -



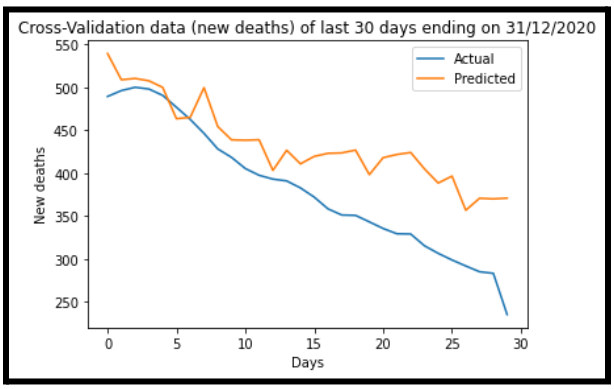
Train RMSE (new cases): 4243.963



Train RMSE (new deaths): 88.534



Test RMSE (new cases): 2708.759



Test RMSE (new deaths): 62.903

Fig 6 : Training and Cross-Validation (LSTM Model)

NOTE: For the forecast part, we are assuming that the conditions that prevailed in the month of December will be the same for the month of January as well.

The below graphs are appearing to be similar in shapes since according to the training data, the graphs for new cases per day as well as new deaths per day, are also similar in shape to each other ,i.e, the number of deaths and number of new cases are strongly correlated to each other.

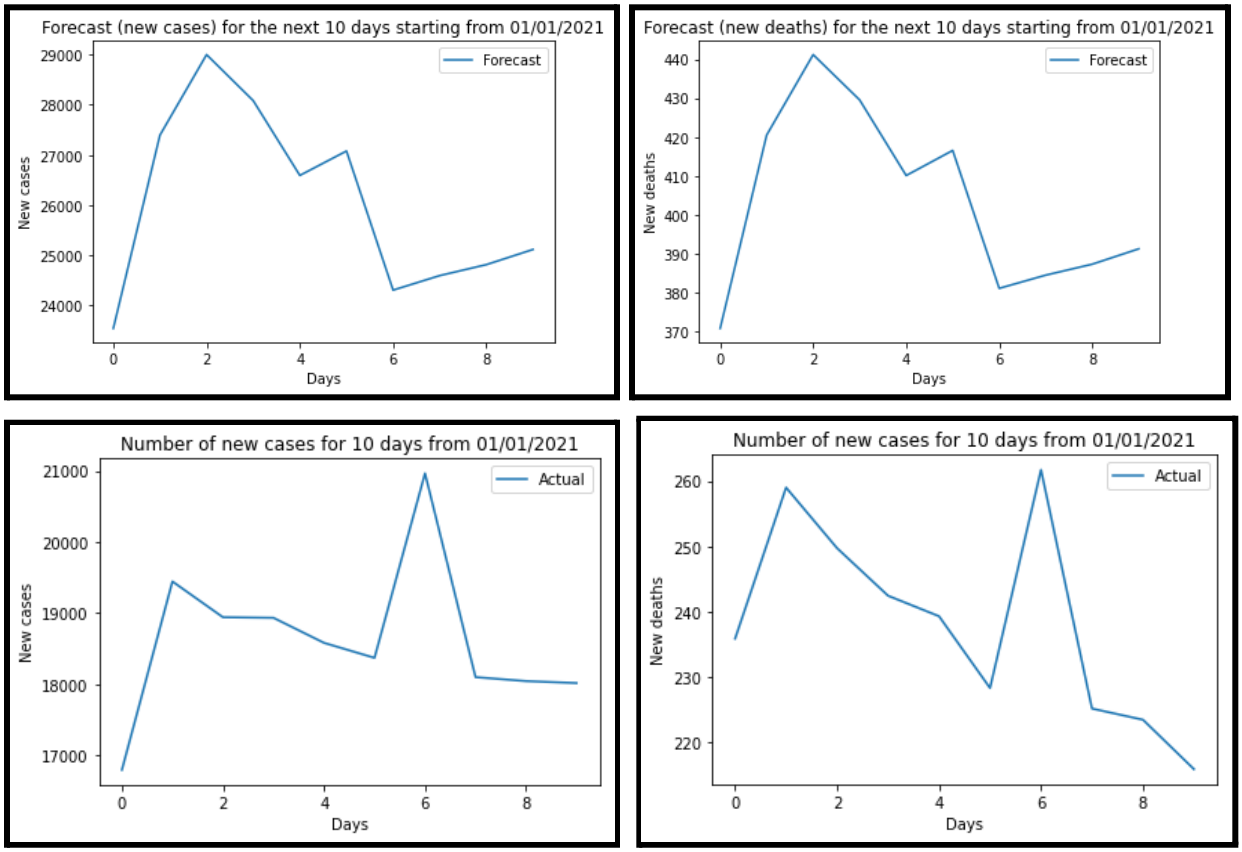


Fig 7 : Forecast (by LSTM Model) V/S the actual data

CONCLUSION

In this paper, we have tried to tackle the problem of prediction of new COVID19 cases per day as well as new death cases per day by using mainly 3 machine learning models - ARIMA, Random Forest, LSTM.

Although ARIMA is a widely used time series forecasting model, because of the univariate nature of the model it is not well suited for our problem statement which requires consideration of multiple factors such as testing and vaccination, public policy implementation, etc. Random Forest is another very powerful and robust supervised learning technique used for complex problem statements. It is very useful for multivariate dataset like ours for emphasizing on each and every feature as per requirement. The only drawback in this approach is that multi step prediction is very difficult to tackle in Random Forest models. We have to assume that trends of data in the prediction period are very similar to the one we use for training reference.

LSTM has edge over it's other other counterparts because of its ability to keep or forget the relevant old information thus enabling effective multi step time series forecasting with multivariate input data. The requirements of our problem statement precisely matches with LSTM properties making it the best of all the models we've used so far. We were able to achieve cross validation results with RMSE = 2708 which is approximately 9% error from the expected outcomes.

Since we are using completely different datasets for training and forecasting purposes, RMSE value may not be the effective criteria to judge the success of the model. All of these results are applicable under certain assumptions made throughout the design of the model like similar trends in the input features over the time period of forecasting, no sudden change in testing and vaccination, public and international policies, etc. This makes it very difficult to achieve better results beyond a certain point for our problem statement. Nonetheless, we tried considering minimum assumptions and achieving results by not merely stressing upon numbers but also focussing on the practical aspect of the problem statement. We believe good research in this area would definitely help improve the current economic situation as well as health care facilities.