# Using machine learning to predict the next covid wave using ARIMA, LSTM and FBProphet models

Viraj Sonavane
Computer Science
California State University
Chico California, USA
vdsonavane@mail.csuchico.edu

## ABSTRACT

The Coronavirus Disease-2019 (COVID-19) pandemic continues to have a humiliating sway on the wellbeing and prosperity of the worldwide populace. Even after vaccination in several part of the world we can still see rise in new COVID cases. The second COVID-19 wave was deadlier than the first wave and countries struggled to keep up with it. Even after imposing lockdown and following COVID-19 safety protocol, the countries public health system collapsed, which further increased the new cases and deaths globally. Hence, there is need of global predictor that accurately predict the development of the next COVID-19 wave. In this paper, we discuss three different machine learning models that are used to predict next COVID-19 wave. The models are LSTM (Long-Short-Term-Memory) model, FBProphet (Facebook Prophet) model and, ARIMA (Autoregressive Integrated Moving Average) model. For a real-time forecast, live data is used to train the models and then predict the forecast for the upcoming days. The paper also describes the process of tuning up the hyperparameter of the models to achieve lower RMSE (Root Mean Squared Error) values which help to achieve higher accuracy in forecasting the next wave. The paper concludes that LSTM model provided the least RMSE value followed by FBProphet and then ARIMA. Hence, LSTM can be used predict the next covid wave accurately.

## KEYWORDS

COVID-19, ARIMA, LSTM, FBProphet, RMSE.

## 1 Introduction

It's been two years since the time we heard about the first COVID-19 case in the world. The World Health Organization (WHO) declared COVID-19 as a global pandemic by March 2020. This was the time many countries suffered from the first COVID-19 peak wave. The introduction of vaccines did give hope to the world. Numerous, vaccines were administered in the latter half of 2020 which helped the nations to control the rising COVID-19 cases and deaths. Along with COVID-19 vaccines, countries also followed the Lockdown protocol and followed the COVID-19 safety protocol to further limit the new cases. Due to this, the COVID-19 cases started reducing throughout the world and eventually, countries started easing the Lockdown imposed during COVID-19 peak time. This helped the general populous to relax and provided the freedom to roam around. Everything was going well till late 2020 and start of 2021 world came to know about a DELTA version COVID-19 which had more infection rate than the primary one. Again COVID-19 cases started rising throughout the world even after vaccination the cases remain prominent. Vaccinations and following COVID-19 protocol did not help that much to keep a low count of new cases. This gave rise to the second wave or DELTA wave throughout the world. This wave was uglier than the first one because the public health systems of the countries had just gone through the first wave, and they were hit by the second wave. The public health systems could not handle the new cases load and eventually, the system collapsed, and the general population suffered enormously during this time wave. Now a new variant known as OMNICRON of COVID-19 is spreading in the world. This variant spreads twice as fast as the DELTA variant and four times as fast as the primary COVID-19 virus. From Figure 1, we can countries have eased up the restriction protocol. From Figure 2 and Figure 3 we can see countries in which new cases are rising and these rising cases may be due to OMNICRON variant or ease of restriction. These countries may see a third covid wave if precautions are not taken. Figure 4 and Figure 5 show the effect of Covid restriction on new cases. As we can the tighter Covid restriction did help to reduce the count of new cases in the past but only after the new case count had gone up. This shows that countries were not prepared and were not informed about the upcoming COVID-19 wave. Hence, there is a need of a COVID-19 peak wave detector that can help countries to ready their public health system for impeding waves and tackle the upcoming wave effectively using restrictions and lockdown.

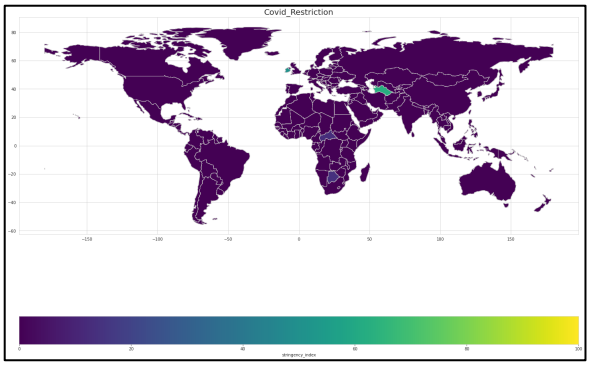



**Figure 1: Covid Restriction in different countries**

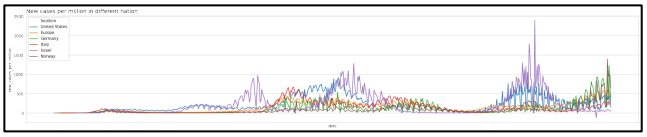


**Figure 2: New cases per million for countries**

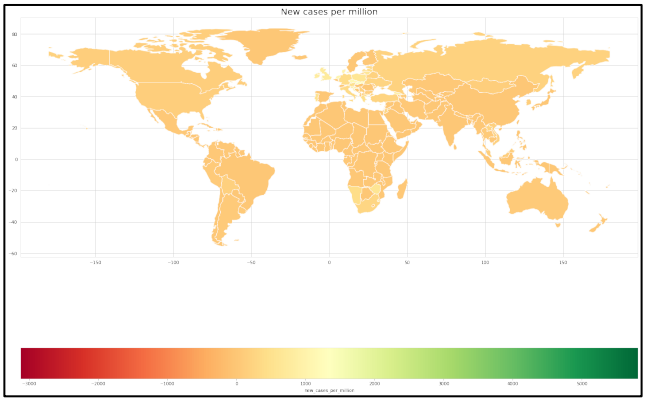



**Figure 3: New cases per million in world**

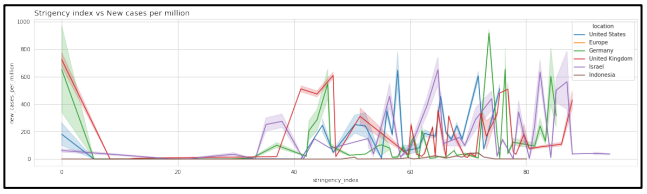



**Figure 4: New cases per million vs Stringency index**

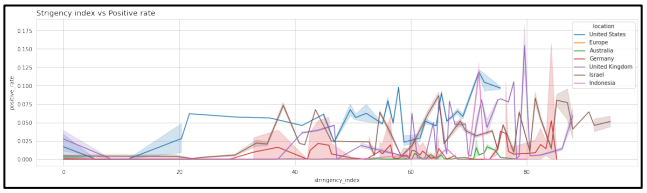



**Figure 5: Positive rate vs Stringency index**

The purpose of this paper is to study and design three machine learning models namely LSTM (Long-Short-Term-Memory), ARIMA (Autoregressive Integrated Moving Average) and, FBProphet (Facebook). Compare the findings of these models using RMSE (Root Mean Square Error) and narrow down one model that has high accuracy to forecast the next COVID-19 wave for all the countries. The paper also discusses how to improve and tune up these models using hyper-parameters to increase the accuracy of forecasts.

## 2 Literature Review

The need of COVID-19 peak detector is very high right now that can help countries to protect themselves from upcoming COVID-19 wave.

Author [1] had used Johns Hopkins University dataset to predict COVID-19 cases for a period of 7 days by using apply machine learning models using ARIMA, ANN, LSTM and CNN. The author has tune up the hyperparameters like p, q and q for ARIMA to increase its accuracy. The author has used data from a time frame of January 22, 2020, till June 30, 3033. The author has used 1024 neuron, dropout layer of 0.2 to avoid over fitting and ReLu activation function to fine tune the LSTM model. The compiler used for LSTM model is Adam optimizer.

Author [2] had used SIR model and FBProphet to predict peak infectives for three different States of India. They have considered a time frame of June 1, 2020, to July 30, 2020, for training the models. The author has used dataset gathered from government of India website. He has used FBProphet model provided by Facebook. The author converted raw data into ds and y before suppling to the model.

Author [3] has used FBProphet to predict the COVID-19 effect on four widely affected nation USA, Brazil, India, and Russia for a period of 60 days. The dataset used by the author is Johns Hopkins University dataset which updated daily. The time frame for training the data is from January 20, 2020, to July 30,2020. The author has set yearly and daily as false for the model.

Author [4] has used ARIMA to predict the daily number of COVID-19 cases in Saudi Arabia for next 4 week or next 1 month. The author had tried and tested the ARIMA model with ARMA, AR and MA and concluded that ARIMA is better. The dataset used by the author is from the website of Saudi Arabia. The ARIMA model is trained from time frame of March 2, 2020, and April 20, 2020. The author compared the different models using RMSE, MAPE and RMSRE values.

Author [5] has used ARIMA and ARMA to develop an outbreak prediction system that predicts COVID-19 cases for ten densely populated countries. The author achieved an accuracy of 89 %. The author has used dataset form respective country. He has split the dataset as 94 % and 6 % before supping to the model. The model was tune up using full autocorrelation. The author has set the range for p and q till 6. The author has used same tune up process to train the ARIMA model also. The Author achieved accuracy of 99% for ARMA and 85% for ARIMA.

In this paper we unlike other authors, we have used ARIMA, FBProphet and LSTM together to predict next COVID wave. We have compared these models over RMSE values and tune them up for better accuracy. Comparison over RMSE gives us

the perfect model that can be used to predict COVID-19 wave for all the countries.

# 3 Methodology

## 3.1 Data Sources

We are using COVID-19 dataset provided by the Our World in Data (OWID) team [1]. This is live data, and it gets updated daily by the OWID team. The OWID team provides a dashboard where we can see global vaccinations, new cases, and deaths. The dataset has 138941 rows and 67 columns. Figure 5 shows the sample of the dataset where first column is the iso_code for every country. The dataset covers a period from as early as 01/01/2020 till the current date for different countries. The OWID dataset collects data from a reliable source like Johns Hopkins University, which provides daily confirmed cases and deaths globally [2]. The dataset does have missing values and early preprocessing of the raw data is needed.

| | iso_code | continent | location | date | total_cases | new_cases | new_cases_smoothed | total_deaths | new_deaths | new_deaths_smoothed | total_cases_per_million | new_cases_per_million |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | AFG | Asia | Afghanistan | 2020-02-24 | 5.0 | 5.0 | NaN | NaN | NaN | NaN | 0.126 | 0.126 |
| 1 | AFG | Asia | Afghanistan | 2020-02-25 | 5.0 | 0.0 | NaN | NaN | NaN | NaN | 0.126 | 0.000 |
| 2 | AFG | Asia | Afghanistan | 2020-02-26 | 5.0 | 0.0 | NaN | NaN | NaN | NaN | 0.126 | 0.000 |
| 3 | AFG | Asia | Afghanistan | 2020-02-27 | 5.0 | 0.0 | NaN | NaN | NaN | NaN | 0.126 | 0.000 |
| 4 | AFG | Asia | Afghanistan | 2020-02-28 | 5.0 | 0.0 | NaN | NaN | NaN | NaN | 0.126 | 0.000 |

**Figure 5: Sample dataset**

## 3.2 Data Preprocessing

The OWID dataset does have a lot of missing values that can be seen from Figure 6 and hence, preprocessing is required. So, to clean the data, we have dropped the columns that are not required
for this paper. Then we have checked the columns where 80% of the dataset are NULL dataset and then dropped these columns. Now for the remaining columns that have less than 80% of NULL dataset as shown in Figure 7, we have replaced the NULL values with zeroes. Finally, we have a clean dataset with no NULL values as shown in Figure 8 and Figure 9.

| | |
|---|---|
| iso_code | 0 |
| continent | 8838 |
| location | 0 |
| date | 0 |
| total_cases | 7735 |
| new_cases | 7740 |
| new_cases_smoothed | 8780 |
| total_deaths | 19076 |
| new_deaths | 18880 |
| new_deaths_smoothed | 8780 |
| total_cases_per_million | 8407 |
| new_cases_per_million | 8412 |
| new_cases_smoothed_per_million | 9447 |
| total_deaths_per_million | 19735 |
| new_deaths_per_million | 19539 |
| new_deaths_smoothed_per_million | 9447 |
| reproduction_rate | 29336 |
| icu_patients | 121838 |
| icu_patients_per_million | 121838 |
| hosp_patients | 119386 |
| hosp_patients_per_million | 119386 |
| weekly_icu_admissions | 137615 |
| weekly_icu_admissions_per_million | 137615 |
| weekly_hosp_admissions | 136713 |
| weekly_hosp_admissions_per_million | 136713 |
| new_tests | 80684 |
| total_tests | 80565 |
| total_tests_per_thousand | 80565 |
| new_tests_per_thousand | 80684 |
| new_tests_smoothed | 68804 |
| new_tests_smoothed_per_thousand | 68804 |
| positive_rate | 72877 |
| tests_per_case | 73538 |
| tests_units | 66722 |
| total_vaccinations | 102280 |
| people_vaccinated | 103904 |
| people_fully_vaccinated | 106849 |
| total_boosters | 129635 |
| new_vaccinations | 108441 |
| new_vaccinations_smoothed | 73268 |
| total_vaccinations_per_hundred | 102280 |
| people_vaccinated_per_hundred | 103904 |
| people_fully_vaccinated_per_hundred | 106849 |
| total_boosters_per_hundred | 129635 |
| new_vaccinations_smoothed_per_million | 73268 |
| new_people_vaccinated_smoothed | 74440 |
| new_people_vaccinated_smoothed_per_hundred | 74440 |
| stringency_index | 25811 |
| population | 989 |
| population_density | 13187 |
| median_age | 19181 |
| aged_65_older | 20503 |
| aged_70_older | 19834 |
| gdp_per_capita | 18411 |
| extreme_poverty | 58419 |
| cardiovasc_death_rate | 19001 |
| diabetes_prevalence | 14962 |
| female_smokers | 45523 |
| male_smokers | 46871 |
| handwashing_facilities | 78568 |
| hospital_beds_per_thousand | 29972 |
| life_expectancy | 9774 |
| human_development_index | 18877 |
| excess_mortality_cumulative_absolute | 134047 |
| excess_mortality_cumulative | 134047 |
| excess_mortality | 134047 |
| excess_mortality_cumulative_per_million | 134047 |

**Figure 6: Null values in dataset**

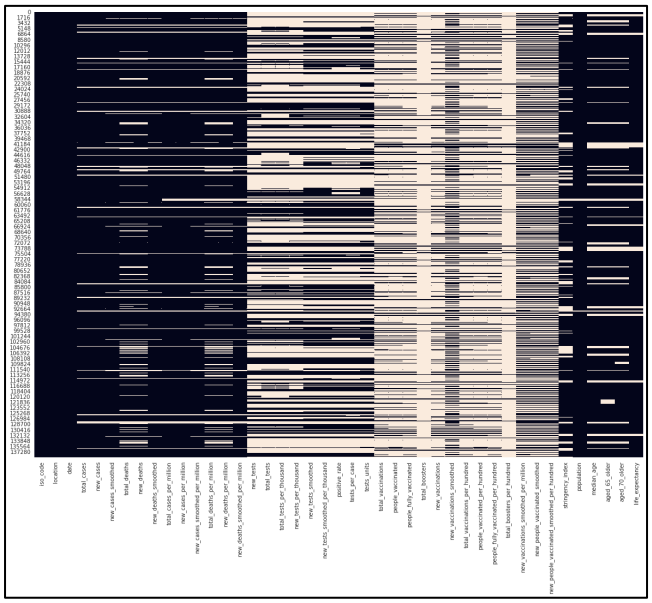


**Figure 7: Missing values in dataset**

```
iso_code                                    0
location                                    0
date                                        0
total_cases                                 0
new_cases                                   0
new_cases_smoothed                          0
total_deaths                                0
new_deaths                                  0
new_deaths_smoothed                         0
total_cases_per_million                     0
new_cases_per_million                       0
new_cases_smoothed_per_million              0
total_deaths_per_million                    0
new_deaths_per_million                      0
new_deaths_smoothed_per_million             0
new_tests                                   0
total_tests                                 0
total_tests_per_thousand                    0
new_tests_per_thousand                      0
new_tests_smoothed                          0
new_tests_smoothed_per_thousand             0
positive_rate                               0
tests_per_case                              0
tests_units                                 0
total_vaccinations                          0
people_vaccinated                           0
people_fully_vaccinated                     0
total_boosters                              0
new_vaccinations                            0
new_vaccinations_smoothed                   0
total_vaccinations_per_hundred              0
people_vaccinated_per_hundred               0
people_fully_vaccinated_per_hundred         0
total_boosters_per_hundred                  0
new_vaccinations_smoothed_per_million       0
new_people_vaccinated_smoothed              0
new_people_vaccinated_smoothed_per_hundred  0
stringency_index                            0
population                                  0
median_age                                  0
aged_65_older                               0
aged_70_older                               0
life_expectancy                             0
dtype: int64
```

**Figure 8: Zero NULL values in dataset**

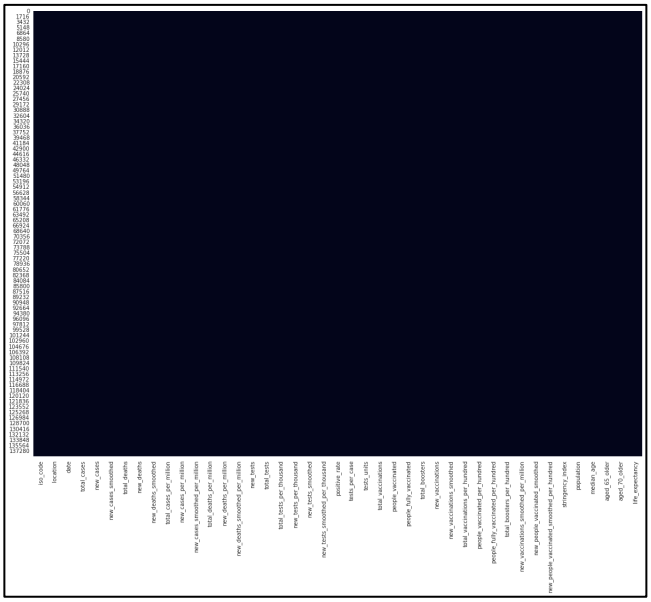


**Figure 9: Clean dataset**

## 3.3 Environmental Setup

We are using python as a programming language and a Google Colab environment to train and run the machine learning models. As said earlier we are using the COVID-19 dataset provided by the OWID team. For this paper, we are using only the 'date' and 'new cases per million' columns from the OWID dataset for training the models. Other columns like 'people fully vaccinated', 'stringency index' and 'positive rate' are used for exploratory data analysis only.

## 3.4 Proposed Approach

This section emphasizes on the working and experimental setup of the models. We have used three different models, LSTM, ARIMA and, FBProphet.

A. Long-Short-Term-Memory (LSTM):

There are different types of deep learning (DL) techniques like convolution neural network (CNN), recurrent neural network (RNN) which is an upgrade over the deep neural network (DNN). The drawback of DNN is that it is a feed-forward neural network (FFNN) which allows data only to move forward from one input layer to the next output layer in a forward direction without ever going in a backward direction or ever touching the node from the last layer once again. The DL techniques were designed by observing the working of the human brain and the DNN does not seem to follow the human brain protocol [8]. To overcome this drawback RNN technique was introduced. The RNN solves this issue by using the previous layer to compute the next layer hence, it allows data to flow in the backward direction. As RNN uses the nodes from the previous layer to compute the next layer, it can use information effectively as each node in the previous layer gets a chance to analyze the data once again and use that analysis to determine the next layer. This effectively increases the efficiency of the RNN technique. Hence, RNN is used in speech recognition and handwriting recognition [9]. But RNN also suffers from drawbacks like vanishing/exploding gradient [10] or long-term dependency. When input data is fed to an RNN network, it breaks the incoming input and assigns different weights to these inputs. The problem is that by the time RNN receives the last input it forgets the first input that it has received previously because the weights of that input have been reduced to zero or irrelevant. To solve this problem LSTM networks are used over RNN. This is a different type of RNN network that has the capacity of learning and understanding the long-term dependencies of the input data. The efficiency of LSTM is better than RNN because it incorporates math functions which helps LSTM to have a better memory than RNN. LSTM has three gates Input Gate, Forget Gate and, Output Gate as shown in Figure 17. The LSTM has the following equations in its cells.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(z_i - \tilde{z}_i)^2}$$

**Figure 10: Equation 1 [4]**

$$i_t = \sigma\left(W_i \cdot \left[u_{t-1}, x_t\right] + b_i\right),$$

**Figure 11: Equation 1 [7]**

$$\tilde{C}_t = \tanh\left(W_C \cdot [u_{t-1}, x_t] + b_C\right),$$

**Figure 12: Equation 2 [7]**

$$f_t = \sigma\left(W_f \cdot [u_{t-1}, x_t] + b_f\right),$$

**Figure 13: Equation 3 [7]**

$$o_t = \sigma\left(W_o \cdot [u_{t-1}, x_t] + b_o\right),$$

**Figure 14: Equation 8 [7]**

$$C_t = f_t C_{t-1} + i_t \tilde{C}_t,$$

**Figure 15: Equation 9 [7]**

$$u_t = \tanh C_t \times o_t.$$

**Figure 16: Equation 10 [7]**

where the weights are $W_i$, $W_c$, $W_f$, $W_o$ and the biases are $b_i$, $b_c$, $b_f$, $b_o$. As we can see from the equations, tanh activation functions are used in all cells of LSTM. This helps to distribute the gradients in the LSTM network while training it. We are using LSTM because it works better on time-series data and covid data is time-series data [9].

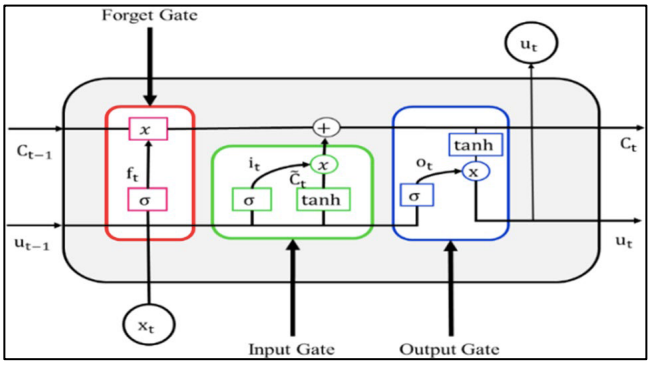



**Figure 17: LSTM cell [7]**

Experimental Setup:

The LSTM model was imported using Keras API. The dataset was split into 70:30 ratio as training data and testing data. For this paper, we are using a time-series generator provided by the Keras module. The neural network requires formatted time series data for forecasting future values. The time-series generator inserts the time series of the dataset into a time-series generator object and that object is used by the neural network to generate the forecast. We are feeding our training and testing data to the time-series generator with a batch size of 64 and a length of 18. The length is the number of months on which the model is trained. So, the last 18-month of data is used to train the LSTM model. For this paper, we are using a univariate sequential LSTM model. The LSTM model has a single hidden layer and, one output layer. The hidden layer has 85 neurons, ReLu or rectifier activation function and input shape (18,1). The output layer is a dense layer that connects the previous neurons to the next neurons. The LSTM model uses Adam optimizer and mean squared error loss (MSE) with 500 epochs. Figure 18 shows the summary of layers in the LSTM model. Then, the model is fitted, and we can see the loss per epoch. The loss per epoch for the country Norway is shown in Figure 19. The RMSE value for Norway is 11.12 which is not exactly zero but still better, while the MAPE value for Norway is 85.08. Figure 20 shows actual data vs predicted data for Norway. Finally, we use the trained LSTM model to forecast the upcoming wave over a period of 250 days ahead. Figure 21 shows the future forecast for Norway where we can see that Norway will get a Covid-19 peak wave by January 2022. The future forecast for different countries is shown in the Result section.

```
Model: "sequential_11"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_11 (LSTM)              (None, 85)                29580

 dense_11 (Dense)            (None, 1)                 86

=================================================================
Total params: 29,666
Trainable params: 29,666
Non-trainable params: 0
_________________________________________________________________
```

**Figure 18: LSTM layers with 85 neurons**

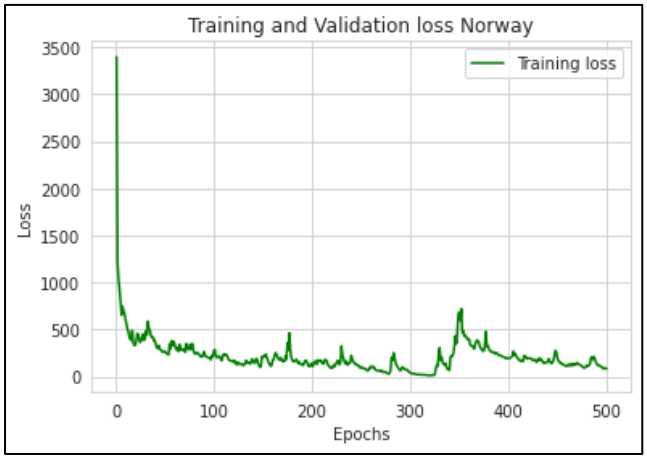



**Figure 19: Loss per epochs for Norway**

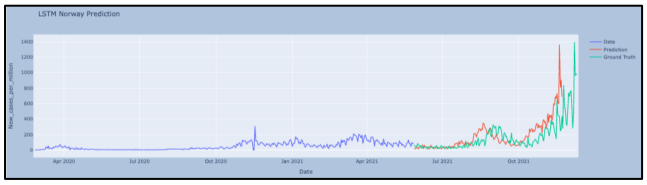


**Figure 20: Norway Actual data vs Predicted data**

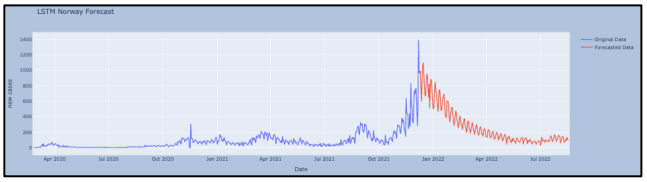


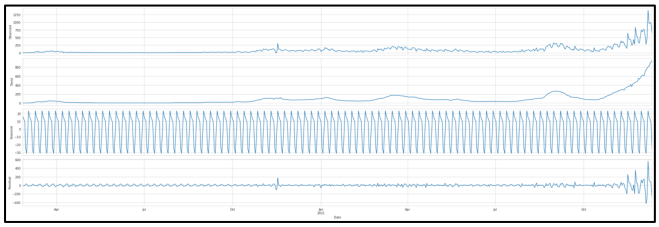

**Figure 21: Norway Future Forecast**

B. Autoregressive Integrated Moving Average (ARIMA)

ARIMA is a special type of statistical model which supports time series forecasting. ARIMA uses past values to predict the future time series. ARIMA can be broken down as AR, I and, MA. The AR (Autoregression) is a type of model that makes use of the relationship between observation and lagged observation. The I (Integrated) work in a way to make the time series stationary by using the technique of differencing the observation. The MA. (Moving Average) is a model that uses the dependency of observation and residual error of the applied model over lagged observation [11]. ARIMA model has three different hyper-parameters namely p, q and, d where p is called lag order represents the autoregression, q is called the moving average and, d is called degree of difference, which represents the required differences [5]. The ARIMA model requires stationary time series data. For this, we have used ADF (Augmented Dicky Fuller) test. If the time series data is not stationary, then the model accuracy will vary at different time points. The time series is made stationary using differencing. Then autocorrelation is used to check for trend and seasonality in the supplied dataset. Then we fine-tune p, q and, d values to train the ARIMA model on the obtained non-stationary data.

Experimental Setup:

To check the seasonality and trend in the dataset we have used seasonal decompose with additive model and extrapolate trend as frequency. Figure 22 shows the seasonal decompose for Norway country. Then we have use the ADF test on the dataset to see if the data is stationary or not. Figure 22 also shows the ADF test summary of Norway. The dataset is then passed through the autocorrelation function to see the seasonal trend in the dataset. Figure 24 shows autocorrelation output for Norway. For this paper, we are using auto-arima provided by the pmdarima API module. This is a strong process that identifies the best and optimal hyper-parameter for the ARIMA model based on the AIC (Akaike Information Criterion) values and delivers a good ARIMA model. The working of auto-arima is such that it has inbuilt tests like ADF to find out perfect d, the range for p and q, stepwise algorithms, seasonal tests, and trends. In this paper we have used auto-arima with ADF test, set the range of p, q as minimum 1 and maximum 3, set the seasonal as true to fit the computed ARIMA model, set the trace as true to print the summary of the fitted model and set the stepwise algorithm as true to find out optimal parameters and avoid overfitting. First, we have train the auto-arima model on partial data and used that trained model to predict the next 2-month prediction. Figure 24 shows the auto-arima summary when we supply partial data of new cases per million for Norway. From Figure 24 we can see that auto-arima tried and tested different p, q and, d parameters to find out one optimal p, q and, d over the smaller AIC value. Figure 25 shows actual data vs predicted data for new cases per million for Norway. We can validate that the model can predict future data properly. After validating the model, we have trained the model for the full dataset and used that trained model to predict a two-month future forecast. Figure 26 shows the auto-arima summary for the full dataset of new cases per million for Norway and Figure 27 shows the future precisions for Norway. We can see that Norway will have a COVID-19 Peak wave by January 2022. The RMSE value for Norway using ARIMA model is 58.21. The future forecast for different countries using ARIMA is shown in the Result section.



```
ADF Test Statistic : 3.251286195763205
p-value : 1.0
#Lags Used : 20
Number of Observations : 634
weak evidence against null hypothesis,indicating it is non-stationary
```

**Figure 22: Seasonal decompose and**

**ADF test for Norway**

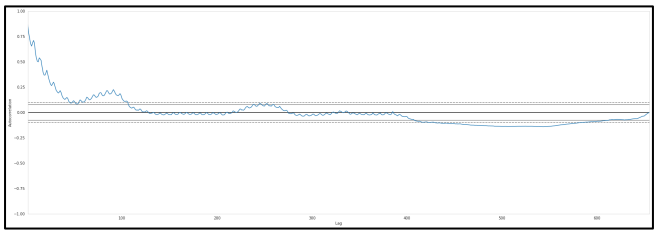


**Figure 23: Norway Future Forecast**

```
ARIMA(0,0,1)(0,1,1)[7] intercept   : AIC=3650.693, Time=0.63 sec
ARIMA(0,0,0)(0,1,0)[7]             : AIC=3697.058, Time=0.02 sec
ARIMA(1,0,1)(0,1,0)[7] intercept   : AIC=3655.941, Time=0.46 sec
ARIMA(1,0,1)(1,1,1)[7] intercept   : AIC=3509.273, Time=2.07 sec
ARIMA(1,0,1)(1,1,0)[7] intercept   : AIC=3581.250, Time=1.03 sec
ARIMA(1,0,1)(2,1,1)[7] intercept   : AIC=3510.378, Time=4.43 sec
ARIMA(1,0,1)(1,1,2)[7] intercept   : AIC=3510.497, Time=5.53 sec
ARIMA(1,0,1)(0,1,2)[7] intercept   : AIC=3509.755, Time=4.46 sec
ARIMA(1,0,1)(2,1,0)[7] intercept   : AIC=3556.795, Time=2.73 sec
ARIMA(1,0,1)(2,1,2)[7] intercept   : AIC=3512.583, Time=6.99 sec
ARIMA(0,0,1)(1,1,1)[7] intercept   : AIC=inf, Time=1.61 sec
ARIMA(1,0,0)(1,1,1)[7] intercept   : AIC=3609.542, Time=1.13 sec
ARIMA(2,0,1)(1,1,1)[7] intercept   : AIC=3510.681, Time=2.98 sec
ARIMA(1,0,2)(1,1,1)[7] intercept   : AIC=3510.315, Time=3.01 sec
ARIMA(0,0,0)(1,1,1)[7] intercept   : AIC=inf, Time=1.73 sec
ARIMA(0,0,2)(1,1,1)[7] intercept   : AIC=3629.691, Time=1.85 sec
ARIMA(2,0,0)(1,1,1)[7] intercept   : AIC=3570.275, Time=1.31 sec
ARIMA(2,0,2)(1,1,1)[7] intercept   : AIC=3512.613, Time=3.27 sec
ARIMA(1,0,1)(1,1,1)[7]             : AIC=3509.050, Time=1.28 sec
ARIMA(1,0,1)(0,1,1)[7]             : AIC=3510.439, Time=0.82 sec
ARIMA(1,0,1)(1,1,0)[7]             : AIC=3579.774, Time=0.57 sec
ARIMA(1,0,1)(2,1,1)[7]             : AIC=3510.317, Time=2.76 sec
ARIMA(1,0,1)(1,1,2)[7]             : AIC=3510.445, Time=3.99 sec
ARIMA(1,0,1)(0,1,0)[7]             : AIC=3654.566, Time=0.24 sec
ARIMA(1,0,1)(0,1,2)[7]             : AIC=3509.455, Time=2.91 sec
ARIMA(1,0,1)(2,1,0)[7]             : AIC=3555.421, Time=1.20 sec
ARIMA(1,0,1)(2,1,2)[7]             : AIC=3512.316, Time=3.68 sec
ARIMA(0,0,1)(1,1,1)[7]             : AIC=3652.810, Time=0.60 sec
ARIMA(1,0,0)(1,1,1)[7]             : AIC=3615.816, Time=0.69 sec
ARIMA(2,0,1)(1,1,1)[7]             : AIC=3510.443, Time=1.41 sec
ARIMA(1,0,2)(1,1,1)[7]             : AIC=3510.069, Time=1.85 sec
ARIMA(0,0,0)(1,1,1)[7]             : AIC=3682.776, Time=0.49 sec
ARIMA(0,0,2)(1,1,1)[7]             : AIC=3640.944, Time=0.87 sec
ARIMA(2,0,0)(1,1,1)[7]             : AIC=3572.787, Time=0.76 sec
ARIMA(2,0,2)(1,1,1)[7]             : AIC=3512.395, Time=2.53 sec

Best model:  ARIMA(1,0,1)(1,1,1)[7]
Total fit time: 74.215 seconds
```

Statespace Model Results

| | | | |
|---|---|---|---|
| **Dep. Variable:** | y | **No. Observations:** | 401 |
| **Model:** | SARIMAX(1, 0, 1)x(1, 1, 1, 7) | **Log Likelihood** | -1749.525 |
| **Date:** | Sun, 12 Dec 2021 | **AIC** | 3509.050 |
| **Time:** | 22:14:46 | **BIC** | 3528.932 |
| **Sample:** | 0 | **HQIC** | 3516.928 |
| | - 401 | | |
| **Covariance Type:** | opg | | |

| | coef | std err | z | P>\|z\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| **ar.L1** | 0.9800 | 0.009 | 107.054 | 0.000 | 0.962 | 0.998 |
| **ma.L1** | -0.7164 | 0.026 | -27.732 | 0.000 | -0.767 | -0.666 |
| **ar.S.L7** | 0.1221 | 0.048 | 2.540 | 0.011 | 0.028 | 0.216 |
| **ma.S.L7** | -0.8682 | 0.038 | -22.708 | 0.000 | -0.943 | -0.793 |
| **sigma2** | 413.2547 | 11.730 | 35.231 | 0.000 | 390.265 | 436.245 |

| | | | |
|---|---|---|---|
| **Ljung-Box (Q):** | 50.52 | **Jarque-Bera (JB):** | 48565.02 |
| **Prob(Q):** | 0.12 | **Prob(JB):** | 0.00 |
| **Heteroskedasticity (H):** | 6.82 | **Skew:** | 3.87 |
| **Prob(H) (two-sided):** | 0.00 | **Kurtosis:** | 56.84 |

**Figure 24: Auto-Arima summary for partial data for Norway**

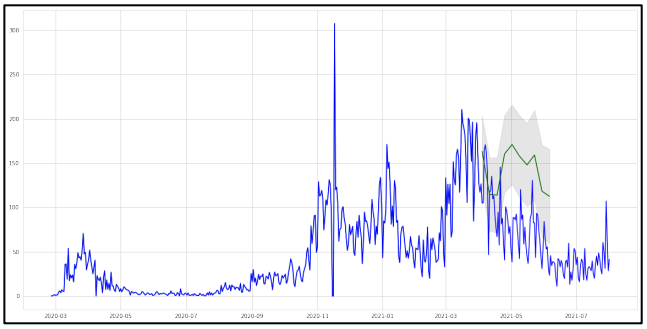


**Figure 25: Actual data vs Predicted data by Arima model for Norway**

```
Performing stepwise search to minimize aic
 ARIMA(1,1,1)(0,1,1)[7]             : AIC=6838.774, Time=0.74 sec
 ARIMA(0,1,0)(0,1,0)[7]             : AIC=7408.515, Time=0.03 sec
 ARIMA(1,1,0)(1,1,0)[7]             : AIC=7087.620, Time=0.28 sec
 ARIMA(0,1,1)(0,1,1)[7]             : AIC=6852.219, Time=0.82 sec
 ARIMA(1,1,1)(0,1,0)[7]             : AIC=6901.921, Time=0.23 sec
 ARIMA(1,1,1)(1,1,1)[7]             : AIC=6840.005, Time=0.99 sec
 ARIMA(1,1,1)(0,1,2)[7]             : AIC=6838.450, Time=1.46 sec
 ARIMA(1,1,1)(1,1,2)[7]             : AIC=6824.477, Time=3.10 sec
 ARIMA(1,1,1)(2,1,2)[7]             : AIC=6799.965, Time=4.53 sec
 ARIMA(1,1,1)(2,1,1)[7]             : AIC=6801.714, Time=2.24 sec
 ARIMA(0,1,1)(2,1,2)[7]             : AIC=6807.369, Time=4.16 sec
 ARIMA(1,1,0)(2,1,2)[7]             : AIC=6941.476, Time=2.99 sec
 ARIMA(2,1,1)(2,1,2)[7]             : AIC=6790.867, Time=5.21 sec
 ARIMA(2,1,1)(1,1,2)[7]             : AIC=6817.705, Time=4.02 sec
 ARIMA(2,1,1)(2,1,1)[7]             : AIC=6796.363, Time=2.98 sec
 ARIMA(2,1,1)(1,1,1)[7]             : AIC=6830.239, Time=1.14 sec
 ARIMA(2,1,0)(2,1,2)[7]             : AIC=6859.213, Time=4.74 sec
 ARIMA(3,1,1)(2,1,2)[7]             : AIC=6791.182, Time=7.98 sec
 ARIMA(2,1,2)(2,1,2)[7]             : AIC=6743.008, Time=8.51 sec
 ARIMA(2,1,2)(1,1,2)[7]             : AIC=6788.347, Time=6.11 sec
 ARIMA(2,1,2)(2,1,1)[7]             : AIC=6743.966, Time=6.12 sec
 ARIMA(2,1,2)(1,1,1)[7]             : AIC=6792.799, Time=1.89 sec
 ARIMA(1,1,2)(2,1,2)[7]             : AIC=6744.473, Time=5.38 sec
 ARIMA(3,1,2)(2,1,2)[7]             : AIC=6741.321, Time=9.98 sec
 ARIMA(3,1,2)(1,1,2)[7]             : AIC=6790.254, Time=11.07 sec
 ARIMA(3,1,2)(2,1,1)[7]             : AIC=6737.897, Time=8.44 sec
 ARIMA(3,1,2)(1,1,1)[7]             : AIC=6794.716, Time=2.95 sec
 ARIMA(3,1,2)(2,1,0)[7]             : AIC=6754.648, Time=5.63 sec
 ARIMA(3,1,2)(1,1,0)[7]             : AIC=6836.927, Time=1.57 sec
 ARIMA(3,1,1)(2,1,1)[7]             : AIC=6796.266, Time=4.35 sec
 ARIMA(3,1,3)(2,1,1)[7]             : AIC=inf, Time=11.42 sec
 ARIMA(2,1,3)(2,1,1)[7]             : AIC=6771.683, Time=8.47 sec
 ARIMA(3,1,2)(2,1,1)[7] intercept   : AIC=6738.434, Time=13.58 sec

Best model:  ARIMA(3,1,2)(2,1,1)[7]
Total fit time: 153.214 seconds
```

**Figure 26: Auto-Arima summary for full dataset for Norway**

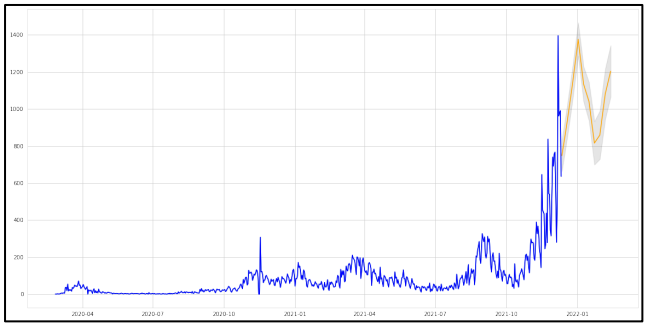


**Figure 27: Future Forecast for Norway by Arima model**

C. FBProphet

FBProphet is an open-source machine learning algorithm introduced in 2017 by Facebook's Core Data Science team. The Prophet predicts the time series forecast for a dataset using an additive model. This is a robust machine learning algorithm that takes care of outlier data or missing data and detects changes in the trend of the dataset hence, it doesn't require that much preprocessing of data to get a good future forecast. The Prophet fits the non-linear trend with daily, weekly, and yearly seasonality along with the holiday effects. This model requires very little tweaking as compared to other models. When the provided dataset has good seasonal effects and variation of seasons over the historical data then the models work perfectly. The Prophet is composed of three models that are holiday, seasonality, and trend this is represented in Figure 28.

$$y(t) = g(t) + s(t) + h(t) + e(t)$$

**Figure 28: FBProphet Equation [3]**

here g(t) represents a trend that can be linear or logistic for modeling the non-periodic changes in the time series, s(t) represents weekly, daily, and yearly changes. The model uses the Fourier series for adjusting the yearly changes. The h(t) represents irregular holiday effects and e(t) is to handle any errors [3]. A Prophet instance class is created then fit and predict methods are called. The input to the model is in form of two columns ds and y where ds require date columns of the dataset and y requires the numeric dataset on which we want the future forecast [12].

Experimental Setup:

We have imported Prophet from the prophet API which follows the sklearn model API. As discussed earlier the FBProphet requires columns in two forms that are ds and y. So, we have renamed our columns as ds and y where ds consist of dates from the COVID-19 dataset and y consists of new cases per million for different countries. FBProphet model has lots of hyperparameters that can be used to optimize the model. For this paper, we have used changepoint prior scale, changepoint range, seasonality mode, growth, and yearly seasonality parameters to tune up the model. To figure out optimal values of the above parameters we have used cross-validation with full dataset, initial as 400 days, period as 90 and, horizon as 180 days. The cross-validation uses MAPE to find out the optimal value of changepoint prior scale, changepoint range, seasonality mode, growth, and yearly seasonality parameters. Figure 29 shows optimal values that cross-validation found out for new cases per million in Norway. After we have obtained the optimal values for the hyper-parameters, we have then fed these values to the model with one more additional feature that is seasonality with the period of 30.5, Fourier order as 5 and, named as monthly. This model is then fitted over the COVID-19 dataset and used for making the future prediction. We have used make the future data frame with the period of 365 and frequency as 'D'. We have first trained the mode on partial data to see if the trained model can predict the future values properly. Figure 30 shows actual values vs precited values of new cases per million for Norway. As we can see the model can forecast the future properly, we will now train the same model over the entire dataset and use the trained model to forecast future values over a period of 365 days. Figure 31 shows the future forecast of new cases per million for Norway. We can see that Norway will have a COVID-19 peak wave by end of December 2021 and another wave by April 2021. The RMSE plot for Norway using FBProphet is shown in Figure 32. The future forecast for different countries is shown in the Result section

```
changepoint_prior_scale 0.005
changepoint_range 0.8
seasonality_prior_scale 0.1
seasonality_mode multiplicative
growth linear
yearly_seasonality 10
```

**Figure 29: Optimal values for hyper-parameters**

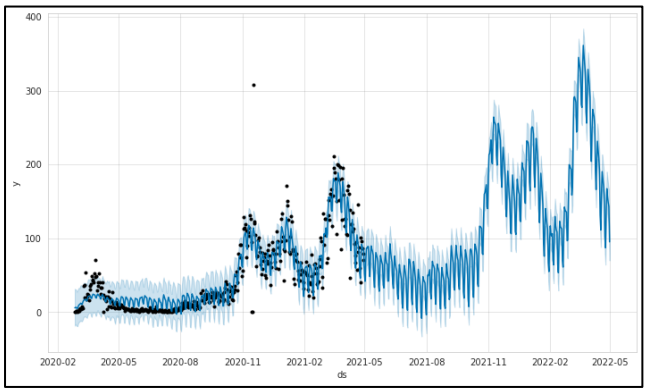



**Figure 30: Actual values vs Predicted Values for Norway**

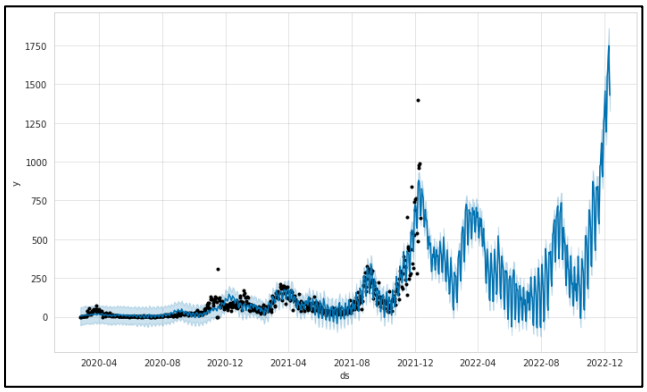



**Figure 31: Future forecast for Norway**

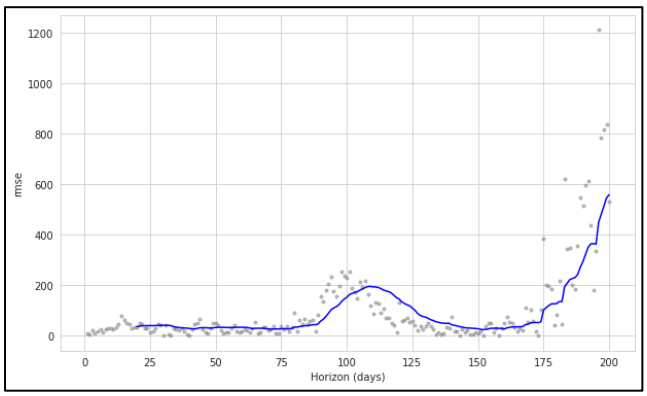



**Figure 32: RMSE values for Norway**

## 4. Results

Here we can see forecast of different countries using LSTM, ARIMA and FBProphet. The forecast of all the three models for some countries are very similar. The countries that are used here for forecasting are the ones that have currently high new cases per million overall. The only exception is Indonesia which currently have low new cases per million

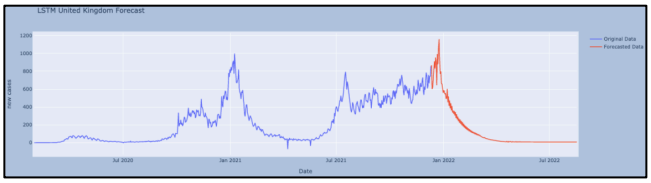



**Figure 33: United Kingdom Future Forecast using LSTM**

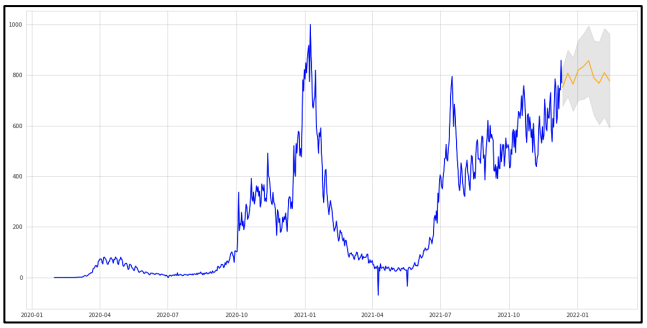


**Figure 34: United Kingdom Future Forecast using ARIMA**

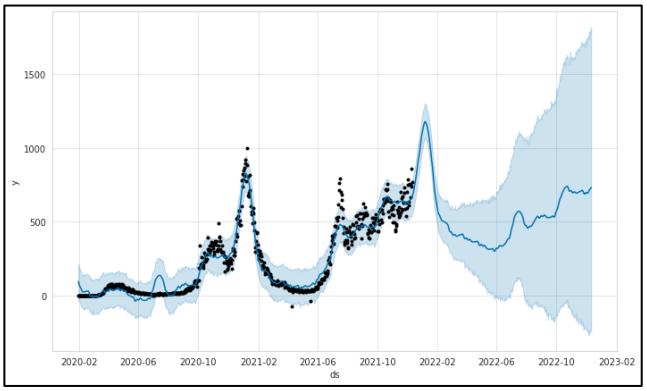



**Figure 35: United Kingdom Future Forecast using FBProphet**

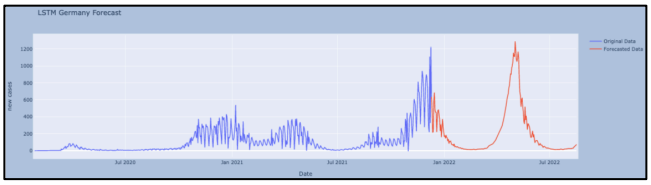



**Figure 36: Germany Future Forecast using LSTM**

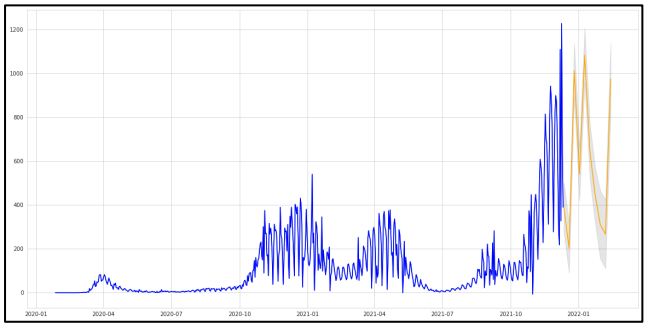


**Figure 37: Germany Future Forecast using ARIMA**

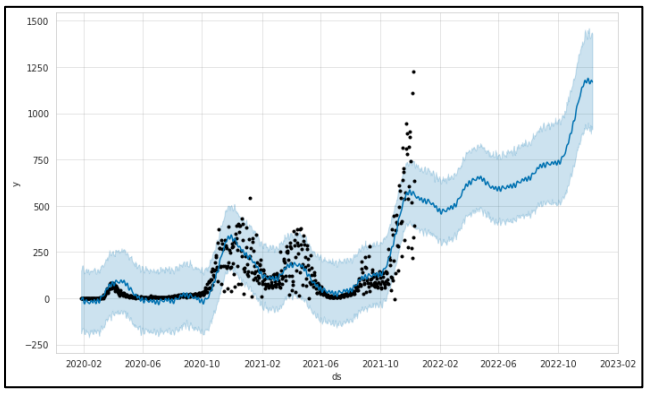



**Figure 38: Germany Future Forecast using FBProphet**

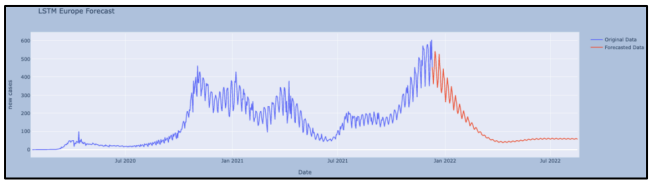



**Figure 39: Europe Future Forecast using LSTM**

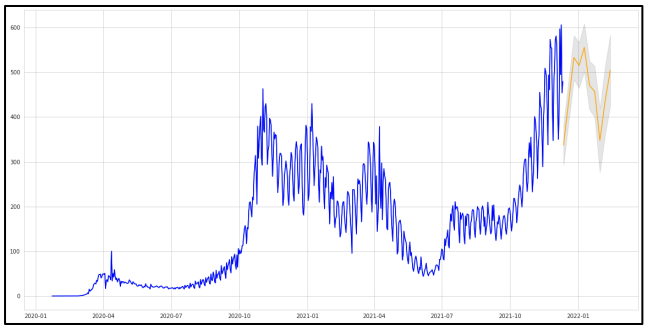


**Figure 40: Europe Future Forecast using ARIMA**

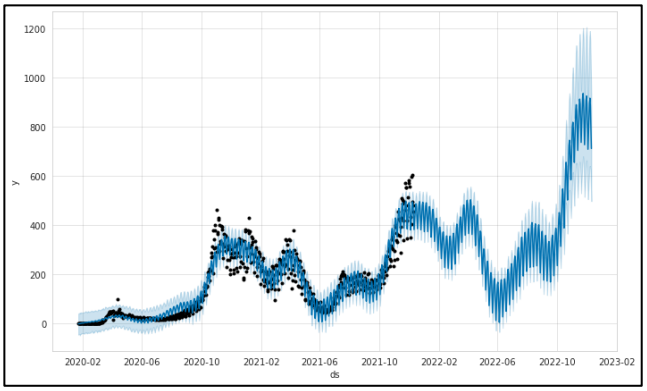



**Figure 41: Europe Future Forecast using FBProphet**

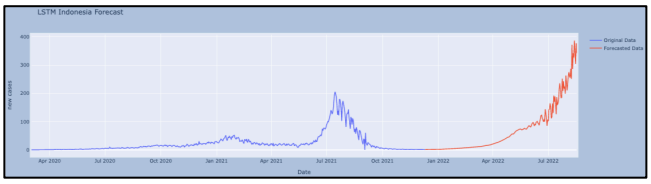



**Figure 42: Indonesia Future Forecast using LSTM**

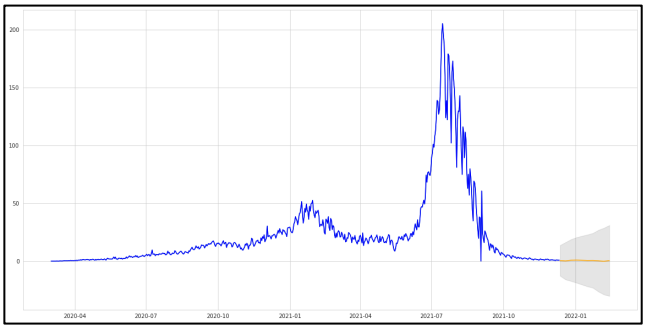


**Figure 43: Indonesia Future Forecast using ARIMA**

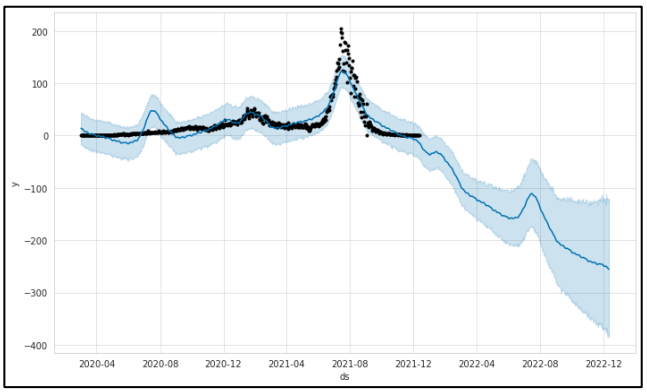



**Figure 44: Indonesia Future Forecast using FBProphet**

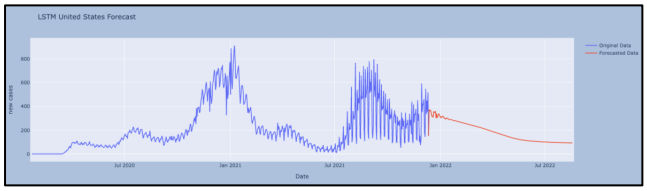



**Figure 45: USA Future Forecast using LSTM.**

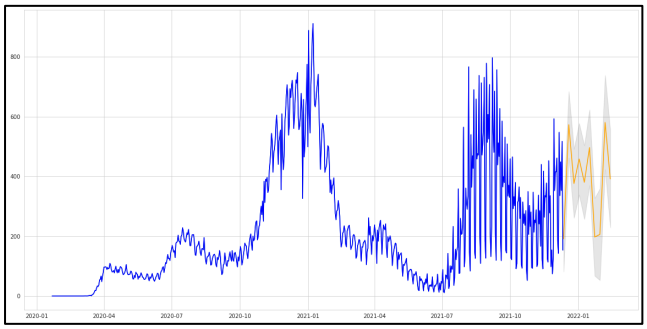


**Figure 46: USA Future Forecast using ARIMA.**

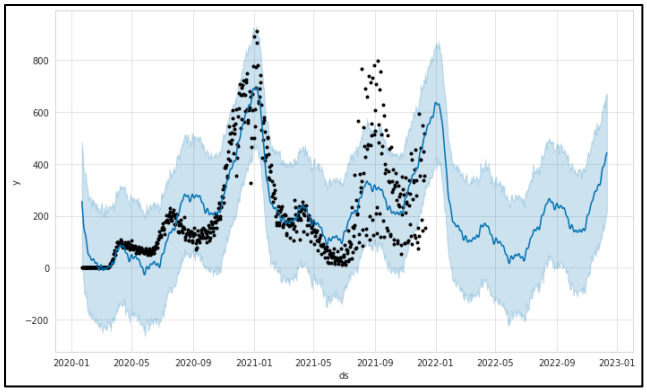



**Figure 47: USA Future Forecast using FBProphet.**

## 5 Conclusion and Future work

With increasing COVID-19 cases even after vaccination, it has become an important matter for all the countries to investigate. This paper aims to use three machine learning models LSTM, ARIMA and, FBProphet to predict the next peak wave of COVID-19 in different countries. This paper discussed how the three models can be designed to get a good future forecast for different countries, it also focuses on the tuning process of the models where we figure out the best hyper-parameter combination for the model to further improve the accuracy of forecasts. Finally, we compare the result obtained from all the models.

Looking at RMSE values of all the three models, LSTM performs better than the other two models, followed by FBProphet and ARIMA. FBProphet was able to forecast 365 days while LSTM was able to forecast 250 days properly. ARIMA can only predict forecasts for the next 60 days. Both LSTM and ARIMA cannot predict 365 days of forecast because the accuracy decreases with increases in forecast days. So, for long-term forecasting, FBProphet should be preferred and for higher accuracy and medium-term forecasting, LSTM should be used. Currently, we

have a limited amount of data hence, in the future when we have a good amount of data the result might change.

Even though the machine learning models that are discussed in this paper provide a good forecast with good accuracy there are still some problems associated with them. In LSTM even though we are getting a good RMSE value, the loss per epoch for some countries is very irregular and that decreases the accuracy of the predicted forecast. Hence, this should be further studied and a single LSTM model which can provide low loss per epoch with high accuracy of forecast on any country should be developed in the future. The problem with the ARIMA model is that it cannot predict beyond 60 days and have a high RMSE value. This problem can be solved by training the model on a big dataset in the future. The FBProphet provides a good long-term forecast but the model sometimes cannot catch outlier data, and this affects the accuracy of the model. This problem can be solved by turning up the hyper-parameter like holidays, uncertainty sample and holiday prior scale that should be done in the future. While training the models only new cases per millions column was used. In the future, the model should be trained on data like new COVID-19 variants, booster vaccination and holidays. A new machine learning algorithm that can handle multiple datasets should be considered to build a robust model that can predict COVID-19 wave for all countries

## REFERENCES


[1] O. Istaiteh, T. Owais, N. Al-Madi and S. Abu-Soud, "Machine Learning Approaches for COVID-19 Forecasting," 2020 International Conference on Intelligent Data Science Technologies and Applications (IDSTA), 2020, pp. 50-57, doi: 10.1109/IDSTA50958.2020.9264101.

[2] N. Darapaneni, P. Jain, R. Khattar, M. Chawla, R. Vaish and A. R. Paduri, "Analysis and Prediction of COVID-19 Pandemic in India," 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020, pp. 291-296, doi: 10.1109/ICACCCN51052.2020.9362817.

[3] Battineni, G., Chintalapudi, N. and Amenta, F. (2020), "Forecasting of COVID-19 epidemic size in four high hitting nations (USA, Brazil, India and Russia) by Fb-Prophet machine learning model", Applied Computing and Informatics, Vol. ahead-of-print No. ahead-of-print. https://doi.org/10.1108/ACI-09-2020-0059

[4] Saleh I. Alzahrani, Ibrahim A. Aljamaan, Ebrahim A. Al-Fakih, Forecasting the spread of the COVID-19 pandemic in Saudi Arabia using ARIMA prediction model under current public health interventions, Journal of Infection and Public Health, Volume 13, Issue 7,2020, Pages 914-919, ISSN 1876-0341, https://doi.org/10.1016/j.jiph.2020.06.001.

[5] Khakharia A, Shah V, Jain S, et al. Outbreak Prediction of COVID-19 for Dense and Populated Countries Using Machine Learning [published online ahead of print, 2020 Oct 16]. Annals of Data Science. 2020;1-19. doi:10.1007/s40745-020-00314-9.

[6] Wang, Peipei & Zheng, Xin-Qi & Li, Jiayang & Zhu, Bangren. (2020). Prediction of Epidemic Trends in COVID-19 with Logistic Model and Machine Learning Technics.Chaos, Solitons & Fractals. 139. 110058. 10.1016/j.chaos.2020.110058.

[7] Bedi, Punam & Dhiman, Shivani & Gole, Pushkar & Gupta, Neha & Jindal, Vinita. (2021). Prediction of COVID-19 Trend in India and Its Four Worst-Affected States Using Modified SEIRD and LSTM Models. SN Computer Science. 2. 10.1007/s42979-021-00598-5.

[8] Haykin S. Neural networks: a comprehensive foundation. 2nd ed. Upper Saddle River: Prentice Hall PTR; 1998.

[9] Goodfellow I, Bengio Y, Courville A. Deep learning. Cambridge: The MIT Press; 2016.

[10] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9:1735–80. https:// doi. org/ 10. 1162/ neco. 1997.9. 8. 1735.

[11] https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/

[12] https://facebook.github.io/prophet/docs/quick_start.html.