RESEARCH

Introduction of Profile Areas of Data Science :Project 6

Suresh Kumar Choudhary, Frenny Macwan and Aman Jain

Full list of author information is available at the end of the article

Abstract

Goal of the Project: We need to forecast numbers of covid-19 cases in Germany (time-series data) using decomposition model and an auto-regressive model with visualization and analysis.

Main Result of the Project: Successfully build the decomposition model(Prophet library), an auto-regressive model and LSTM to predict the number of Covid-19 case in Germany with analysis.

Personal Key Leanings: We understood the vital aspect of decomposition model(Prophet library), an auto-regressive model(SARIMA) and LSTM to predict the time series data.

Estimate working hours: 8 (24 Hours in Total)

Project Evaluation: 2

Number of Words: 1904

Keywords: Time Series data trend, seasonality and noise; Prophet library; SARIMA; LSTM; Covid-19 cases in Germany

Scientific Background

Generally, all the supervised algorithms we have an independent variable which is being used to predict the dependant variable. But if the independent variable is time, and we need to predict or forecast any outcome based on a certain time frame, time series analysis comes into picture. Time series is a set of observations or data points taken at specified times, usually at equal intervals,through which we can predict the future values based on the previous observed values. Time series is really important when it comes to retail, finance and other domains. Through time series analysis we can understand the past behavior, like every weekend the business in restaurants are the highest whereas Mondays have the worst sales, number of flight ticket purchases increases during festival season etc. Using Time series analysis we analyze the past and forecast the future. Generally the pattern is trend, seasonal or irregular which are called as Components of Time Series. Trend could either be a up trend, a down trend or a horizontal trend. Seasonality is a repeating pattern in a fixed time period, where the graph changes on a certain time period always. Irregular are nonrepeating and unsual happenings which changes the dependant variable, like suddenly the claims of Personal Vehicle have dropped during Covid Times because of lockdowns. In this project we are using the decomposition model(prophet library), seasonal ARIMA(Auto regression integrated moving average) and LSTM (Long Short-Term Memory networks) for time series data prediction.

LSTM:

The Long Short-Term Memory network, or LSTM network[1], is a recurrent neural network that is trained using Backpropagation Through Time and overcomes the vanishing gradient problem. As such, it can be used to create large recurrent networks that in turn can be used to address difficult sequence problems in machine learning and achieve state-of-the-art results. Instead of neurons, LSTM networks have memory blocks that are connected through layers. A block has components that make it smarter than a classical neuron and a memory for recent sequences. A block contains gates that manage the block's state and output. A block operates upon an input sequence and each gate within a block uses the sigmoid activation units to control whether they are triggered or not, making the change of state and addition of information flowing through the block conditional. There are three types of gates within a unit:

- **Forget Gate:** Conditionally decides what information to throw away from the block.
- **Input Gate:** Conditionally decides which values from the input to update the memory state.
- **Output Gate:** Conditionally decides what to output based on input and the memory of the block.

Each unit is like a mini-state machine where the gates of the units have weights that are learned during the training procedure.

Goal

We've been provided with the ongoing pandemic Covid-19 sum of cases and deaths for the whole of Germany and for each state. Our goal of the project is to forecast the cases using decomposition and autoregressive models.

Data

We have longitudinal data from the current Covid-19 case numbers for Germany. It consists of time series data for individual Bundesländer and Landkreise (states and counties). The column names use the ISO 3166 code for individual states. The points in time are encoded using localized ISO 8601 time string notation.

Result

Task 1 : Visualize and analyze the given time-series data.

To perform this task we have used data.csv which contains time, state wise cases and deaths, total cases, and total deaths. We have used Prophet lib to implement the model and visualize the results. Prophet is nothing but library which follows the sklearn model API. The input to Prophet is always a dataframe with two columns: ds and y. So we converted our dataframe into two columns dataframe: 'time_iso8601' column in YYYY-MM-DD format as 'ds' and 'sum_cases' column as 'y'. We have used the Prophet.plot_components method to see the trend, yearly seasonality, and weekly seasonality of the time series as shown in Figure 1 and 2.

We have used another approach to analyze time series using seasonal_decompose() method from statsmodels.tsa.seasonal. The general trend of dataset can be seen in Figure 3. Also since not all autoregressive models work on nonstationary data, for

Figure 1 Time series data analysis month wise

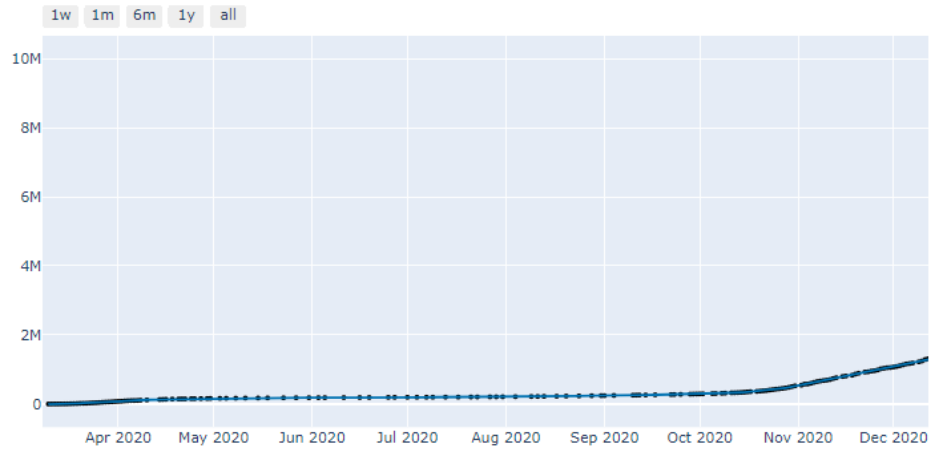
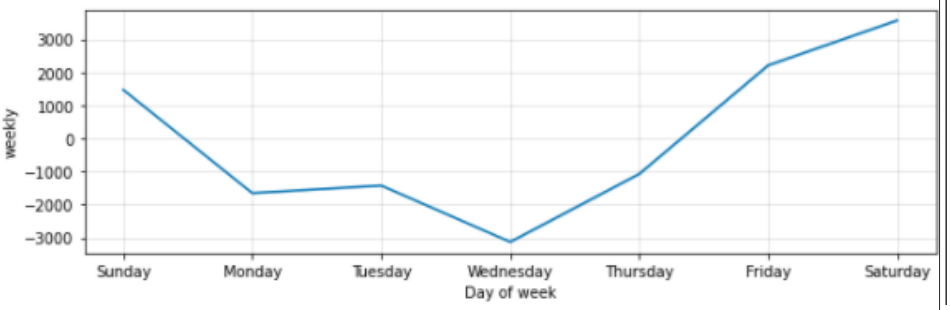


Figure 2 Time series data analysis weekdays wise



that we have used Augmented Dickey-Fuller unit root test, which will let us know if the data is stationary or not. After performing the AD Fuller Test, we got a pvalue of 0.9942 which states that the data is not stationary (PValue more than 0.05 will denote it as a non stationary data), hence not all autoregressive models will work. That is why we have used SARIMA model which makes the nonstationary data as stationary and also gets us the seasonal components. The pvalue which has been derived can be seen in Figure 4.

Task 2 : Perform a forecasting analysis. To perform forecasting analysis, we have distributed the data into train and test set. Train set contains data before '2020-11-27', and test set contains data after '2020-11-27'. We have used fit() method of Prophet to fit the model on train set and predict() method to predict the outcome of test set. Then we compared the predicted values with the actual values and calculated RMSE(root mean squared error) 13927.60 and R2 score 0.97 5. We further tried to forecast outcome of next year using make_future_dataframe(periods=365) and tied to find out the trend that you can see in 6.

Figure 3

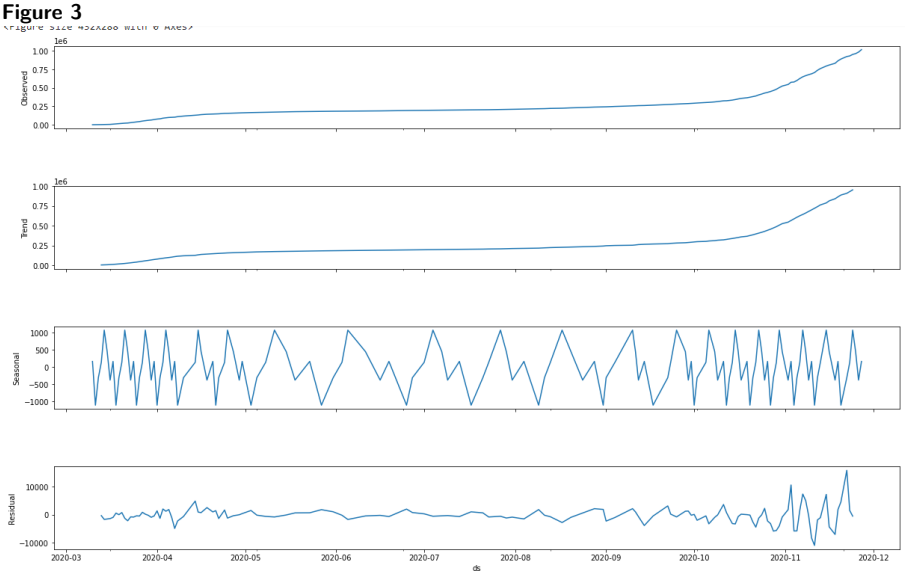
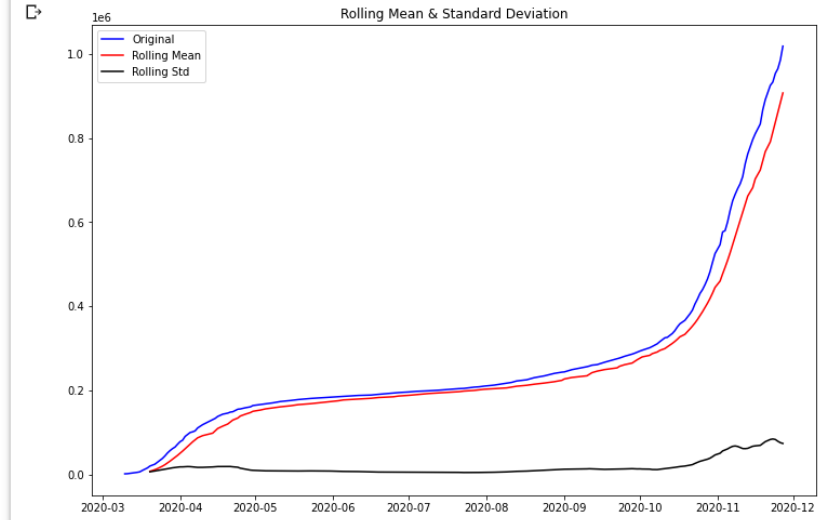


Figure 4




```
Results of Dickey-Fuller Test:
p-value = 0.9942. The series is likely non-stationary.
Test Statistic                  0.991656
p-value                         0.994177
#Lags Used                     11.000000
Number of Observations Used    129.000000
Critical Value (1%)             -3.482088
Critical Value (5%)             -2.884219
Critical Value (10%)            -2.578864
dtype: float64
```

For auto regressive approach, we have used Seasonal Autoregressive Integrated Moving Average model, which is an extension of ARIMA that explicitly supports univariate time series data with a seasonal component. The SARIMA model is specified $\mathbf{(p,d,q)\times(P,D,Q)m}$.

Figure 5 RMSE and R2 score

```
import sklearn.metrics as sm

print("Explain variance score =", round(sm.explained_variance_score(data_test['y'], y_pred['yhat']), 2))
print("R2 score =", round(sm.r2_score(data_test['y'], y_pred['yhat']), 2))
```

```
Explain variance score = 0.98
R2 score = 0.97
```

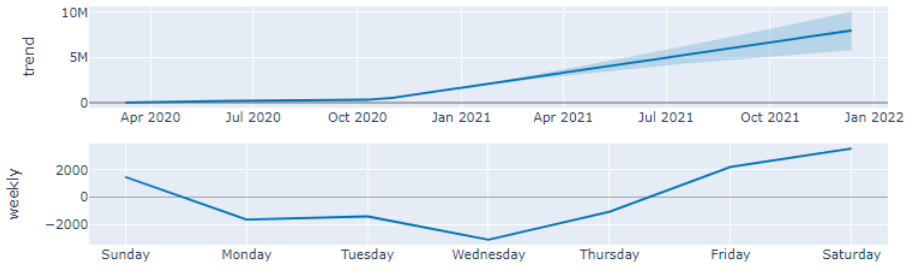
```
import math
mse = sm.mean_squared_error(data_test['y'], y_pred['yhat'])

rmse = math.sqrt(mse)

print("RMSE:",rmse)
```

```
RMSE: 13927.602007044028
```

Figure 6 Forecast trend analysis

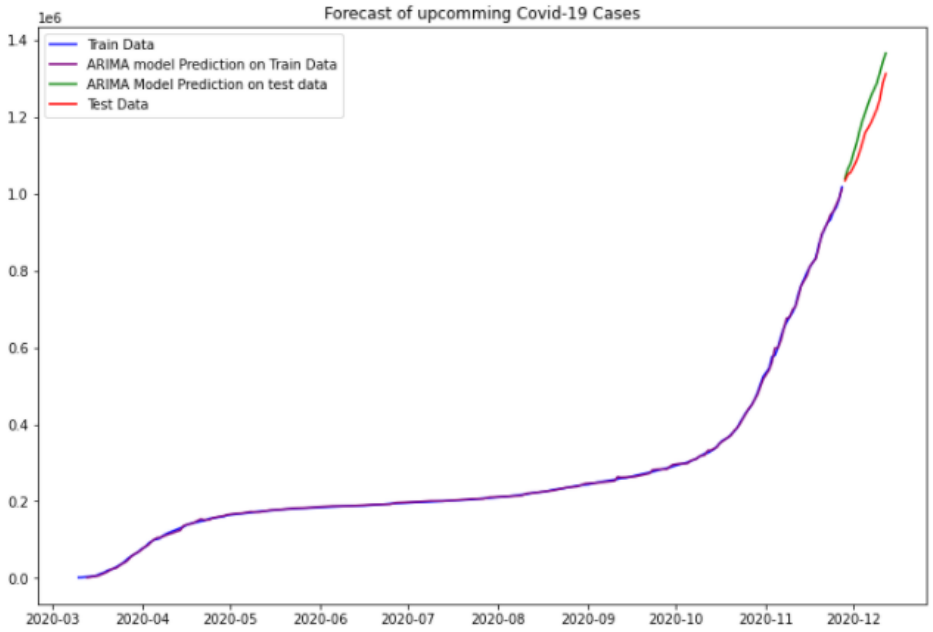


- p: Trend autoregression order.
- d: Trend difference order.
- q: Trend moving average order
- P: Seasonal autoregressive order.
- D: Seasonal difference order.
- Q: Seasonal moving average order.
- m: The number of time steps for a single seasonal period.

Task 3 :Describe, compare and visualize the results of (1) and (2). The trend which we can see in both the approaches are linear, and the number of Covid-19 cases are keeping on increasing till the next year. Using the decomposition model (Facebook's prophet) we saw a weekly seasonal trend where the cases are the least on Wednesdays and shoot up as we reach the weekends and gradually goes down again, and follows this seasonal trend (Weekly) Figure 6. While training the auto regressive model we have used order=(11, 1, 0), seasonal_order=(0, 0, 0, 0), trend='ct' parameters. Using this model we have predicted last two weeks data and calculated RMSE 51513.64 and plotted the trend which can be seen in Figure 7. The comparison of all the models will be found in Task 6.

Task 4 : Perform a change-point analysis. We used the prophet library for the change point analysis, where there were already some change points automatically

Figure 7 ARIMA forecast analysis



assigned (Figure : 8). Since prophet library allows us to manually add the change points we detected two change points and added in the plot.(Figure 9)

- Change Point 1 - 12th April - We could see the number of Covid Cases had stopped rising here, and the graph was flatten because the lockdown was becoming effective,less people were making contacts with each other and hence the rise in Covid Cases reduced and was controlled, but only for time being.
- Change Point 2 - 27th October - The resurgence of Covid-19 occured, and the Wave 2 of Coronavirus has begun infecting more people than Wave 1. It has been linked to to young people ignoring health guidelines - particularly at parties and social events in German cities. Germany had to contain the Wave 2, and hence the Chancellor Angela Merkel decided to put more stricter lock-downs in the end of 2020.

Task 5 : Result Improvement

As a part of result improvement, we used the grid search to find the hyperparameters of SARIMA model for trend ,order and seasonality as dataset was not stationary. We tested the stationary through Augmented Dickey-Fuller Test and rolling statistics. The general auto regression methods can't be used with this stationary dataset. So we used the seasonal ARIMA which takes input with seasonal, trend and noise components and make dataset as stationary.

Task 6 : Forecasting analysis using a data-driven approach(LSTM)

As a part of data driven approach, We are using LSTM for time series data prediction using the window method and the size of the window is tuned for this dataset that is 3. For the current time (t) we are predicting the value at the next time in the sequence (t+1), so we are using the current time (t), as well as the two prior times (t-1 and t-2) as input variables. When phrased as a regression problem,

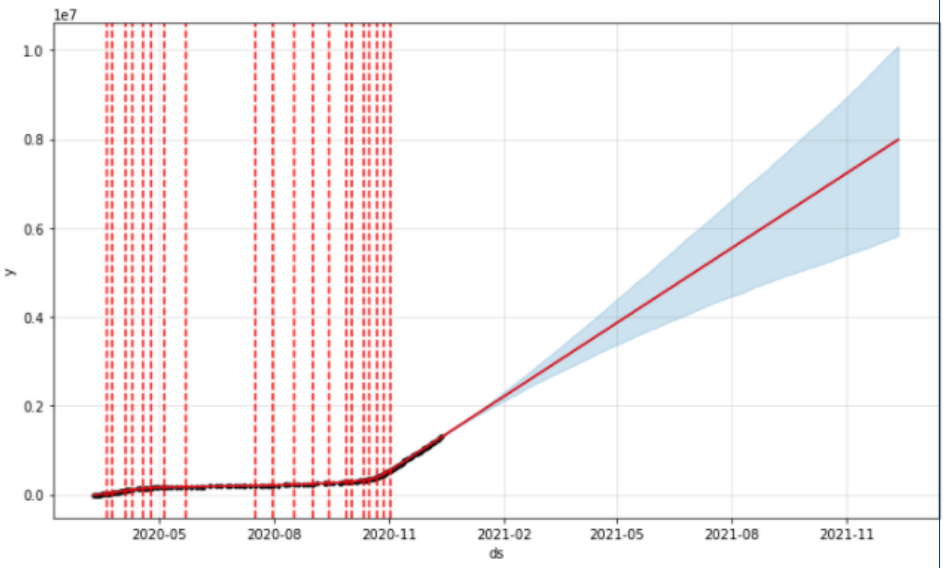
the input variables are t-2, t-1, t and the output variable is t+1. We are using one LSTM layer with 4 nodes and one dense layer (output layer) with loss metric as $'mean\_squared\_error'$, activation function as $'tanh'$and optimizer as $'adam'$. We splitted the dataset as 8:2, so initial 80% rows as trainset and last remaining 20% as test set for LSTM model. We got root mean squared error(RMSE) on test set as 54020.93 which is around 5% of mean of test set(1076665.54). So we can infer from it as our LSTM model is perfectly learned this time series data and predicting well on unseen data. The RMSE value on test set, trainset model prediction and test set model prediction can be seen in figure 10 and 11 respectively.

As a part of comparative analysis of all models, we calculated the RMSE and proportion of RMSE with respect to mean value on test set for decomposition model(prophet library), Auto-regression model(SARIMA) and Data Driven model(LSTM), which can be found in table 1. So we can conclude that decompotion model is performing better than the other two models as it has lowest RMSE and proportion of RMSE with mean value on test set.

Table 1 Comparative analysis of RMSE of all models

Model/algorithm	RMSE on test set	Mean value of test data	Proportion/error
Decomposition model	13927.60	1152499	1.21%
Seasonal ARIMA	51513.64	1152499	4.7%
LSTM	54020.93	1076665.54	5%

Figure 8 Automatic changepoint detection in Prophet



Discussion

Time series allows us to analyze major patterns such as trends, seasonality, cyclicity, and irregularity. There are so many prediction problems that involve a time component. These problems are neglected because it is this time component that makes time series problems more difficult to handle. Through this project we are

Figure 9 Specifying the locations of the changepoints

```
m = Prophet(changepoints=['2020-04-12','2020-10-27']) # specifying change points
forecast = m.fit(df).predict(future)
fig = m.plot(forecast)
a = add_changepoints_to_plot(fig.gca(), m, forecast)
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
```

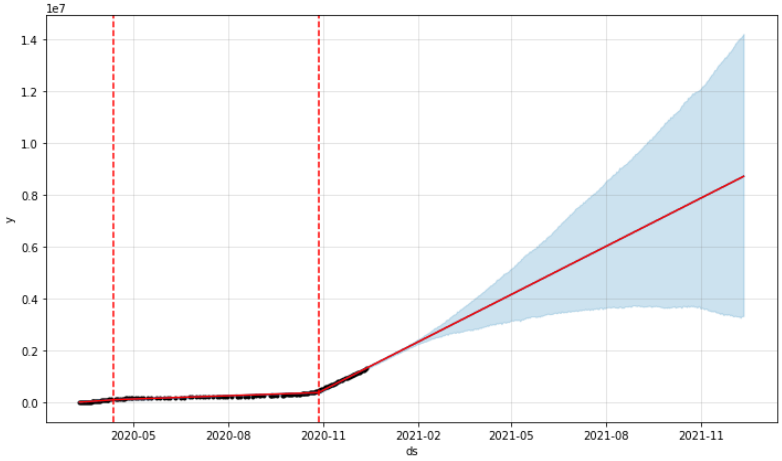


Figure 10 LSTM RMSE on Train and Test Set

```
from sklearn.metrics import mean_squared_error

trainPredict = model.predict(X_tr_t, batch_size=batch_size)
model.reset_states()

testPredict = model.predict(X_tst_t, batch_size=batch_size)
# invert predictions
trainPredict = scaler.inverse_transform(trainPredict)
y_train = scaler.inverse_transform([y_train])
testPredict = scaler.inverse_transform(testPredict)
y_test = scaler.inverse_transform([y_test])
# calculate root mean squared error
trainScore = math.sqrt(mean_squared_error(y_train[0], trainPredict[:,0]))
print('Train Score: %.2f RMSE' % (trainScore))
testScore = math.sqrt(mean_squared_error(y_test[0], testPredict[:,0]))
print('Test Score: %.2f RMSE' % (testScore))
```

```
Train Score: 12533.51 RMSE
Test Score: 54020.93 RMSE
```

able to find the trends, seasonality, cyclicity, irregularity in time series data and can build the time series forecasting using various methods including deep learning that would help in various applications such as stock market analysis, pattern recognition, earthquake prediction, economic forecasting, census analysis, medicine, biology, supply chain management and so on. So this is a typical project for a data-scientist.

Appendix

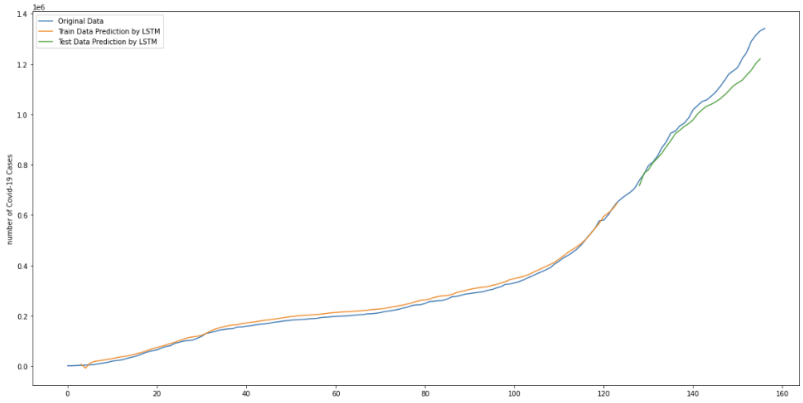
Code -

- Aman - Task3, Task4
- Suresh - Task5, Task6, Auto Regression Model
- Frenny - Decomposition Model, Analysis of Decomposition model

Report -

Figure 11 LSTM Model Prediction plot for Train, Test Set

```
plt.figure(figsize=(20,10))
plt.plot(scaler.inverse_transform(dataset ), label='Original Data')
plt.plot(trainPredictPlot, label='Train Data Prediction by LSTM')
plt.plot(testPredictPlot,label='Test Data Prediction by LSTM')
plt.ylabel('number of Covid-19 Cases')
plt.legend()
plt.show()
```




```
In [579]: def GridSearch(dataset):
```

- Aman - Scientific background, Task2, Task4
- Suresh - Abstract, Discussion, Task5, Task6, some part of scientific background
- Frenny - Abstract, Result (Task 1 and Task 2).

Author details

References

1. https://machinelearningmastery.com/how-to-develop-lstm-models-for-time-series-forecasting/