Chapter 2

Forecasting Using LSTM

Forecasting models have many applications in the fields of economics, business, and finance. Traditionally, there are various stochastic, or randomly determined, forecasting models existing in literature such as Moving Average, AutoRegressive Integrated Moving Average (ARIMA), AutoRegression (AR) and many more. Recently, with advancements in computational power and the development of advanced machine learning algorithms, like deep learning, aspirations for forecasting models have increased. Forecasting models like ARIMA and AutoRegression have limitations due to overly-simple assumptions and the inability to model nonlinear relationship among variables. Overcoming these issues was one of the incentives to develop new algorithms, like RNN-LSTM. The LSTM model has repeatedly outperformed models like ARIMA, random forest and other traditional forecasting algorithms. There are many reasons why the RNN-LSTM model has become the de-facto model in time-series forecasting. However, there are a few reasons more prevalent than others:

- LSTM has a gated architecture that enables it to exploit the memory state of a cell; meaning LSTM has the capability to extract complex patterns and remember the values from earlier stages for a future purpose. Since additional pieces of previous information may affect the accuracy of model, LSTMs become a natural choice of use.

- LSTM can be used to build multivariate input models. For example, in the case of 3D data, input data needs to be converted into a 3D vector and feed it into LSTM model whereas, traditional linear algorithms do not produce satisfactory results in similar circumstances.

- On comparing with linear models, LSTM provides more flexibility while designing the model for a problem, meaning more parameters and hyper parameters can be tuned.

- LSTM provides multiple models like 'many to one' (predicting the next time step value using previous values), 'many to many' (predicting more than one time-step value using previous values) as well as several others. It is possible to configure models for LSTM as per each need. For example, the 'many to many' model can be used to predict only next time steps for input with multiple features. It is also possible to configure how much past data is needed to predict future values using the 'window method' and many more.

But, like other deep learning models, LSTM also requires a good amount of data to provide accurate results and also requires the tuning of multiple parameters and hyper parameters. If these requirements are handled properly then the LSTM model provides satisfactory result.

In this chapter, the application of forecasting LSTM models is discussed.

An architecture using the LSTM model to forecast demand of a products is shown in fig. 1.

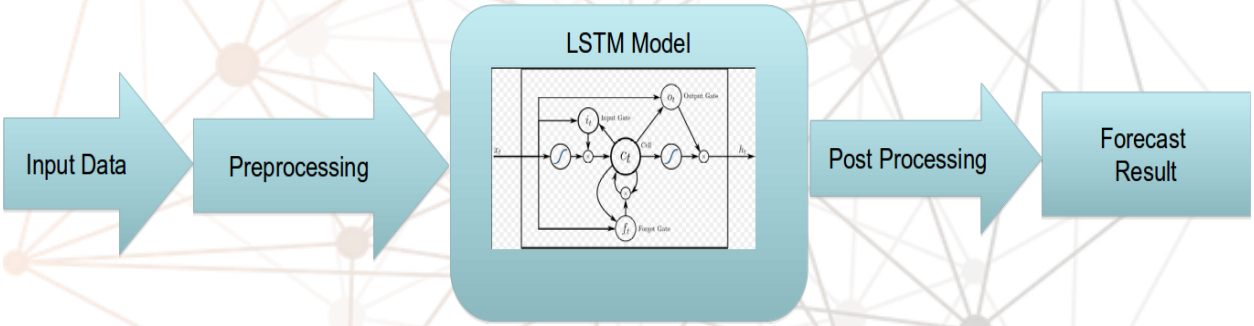

Fig. 1

Fig. 2, represents actual data of demand value. If raw input data is fed into the LSTM model, it will result in an erroneous output; meaning more deviation from the expected output. Time-series forecasting is different from classification and regression problems. In time-series forecasting, temporal structure is encountered that needs to be observed and addressed specifically. In general, during time-series modelling it is assumed that data would be stationary. But in practical problems, this assumption is violated frequently due to trend and seasonality in the data. Data can be said to be stationary if it doesn't have trends and seasonality; meaning the mean and variance of the data will not be time dependent. If the mean and variance changes over time, the data is non stationary. It is easier to work on stationary data.

That said, what do ‘trend’ and ‘seasonality’ mean? ‘Trend’ exists in data if there is a long term increase and decrease in data. It does not have to be linear. Fig.4 shows trend for data shown in Fig. 2. Seasonality exists in the data when similar data patters arise after fixed intervals (i.e., similar pattern after every quarter or 6 months or year and so on).

Fig. 5 shows seasonality for data shown in Fig. 2.

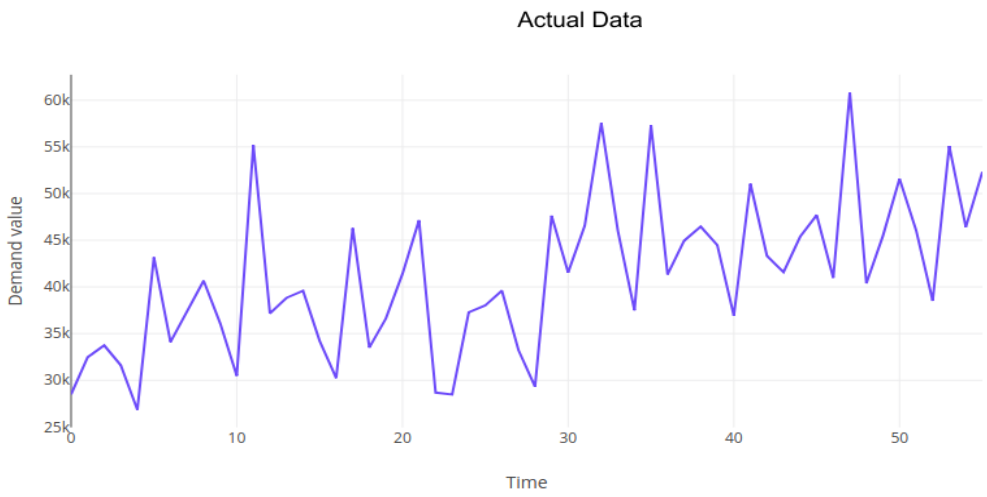

Fig. 2

In Fig. 3, decomposition of data in Trend, Seasonality, and Residual is shown. In this example, during decomposition it is assumed that the actual data follows an additive property; meaning data is composed of trend + seasonality + residual. Data can also follow the multiplicative property. If a variation in the seasonality component changes over time then the data follows the multiplicative property. But if variation is relatively constant then it can be said that data follows the additive property.

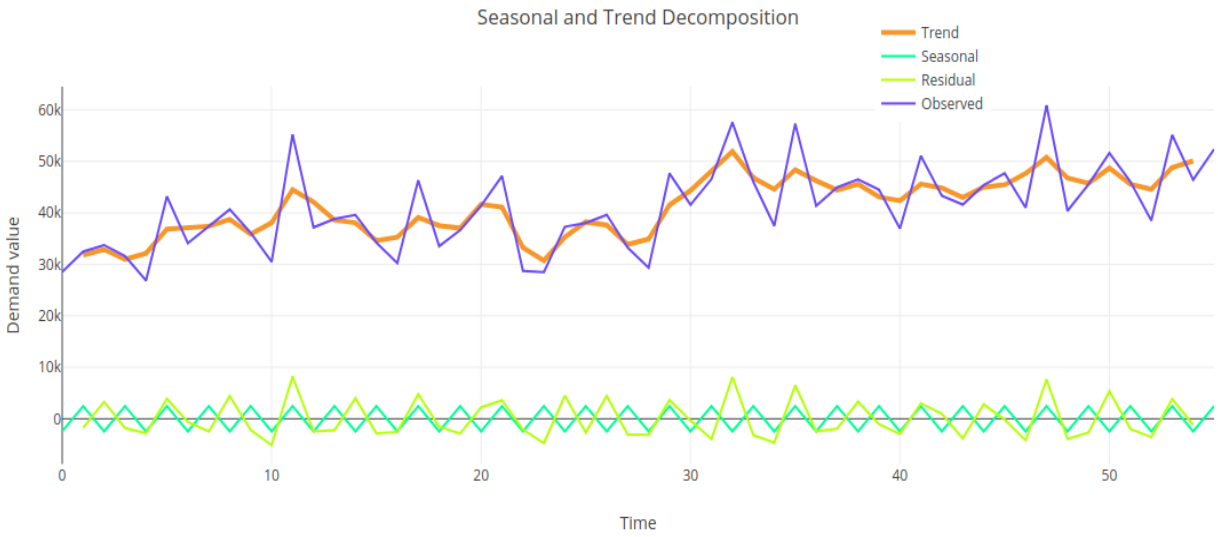

Fig.3: Decomposition of Input data in Trend, Seasonality and Residual

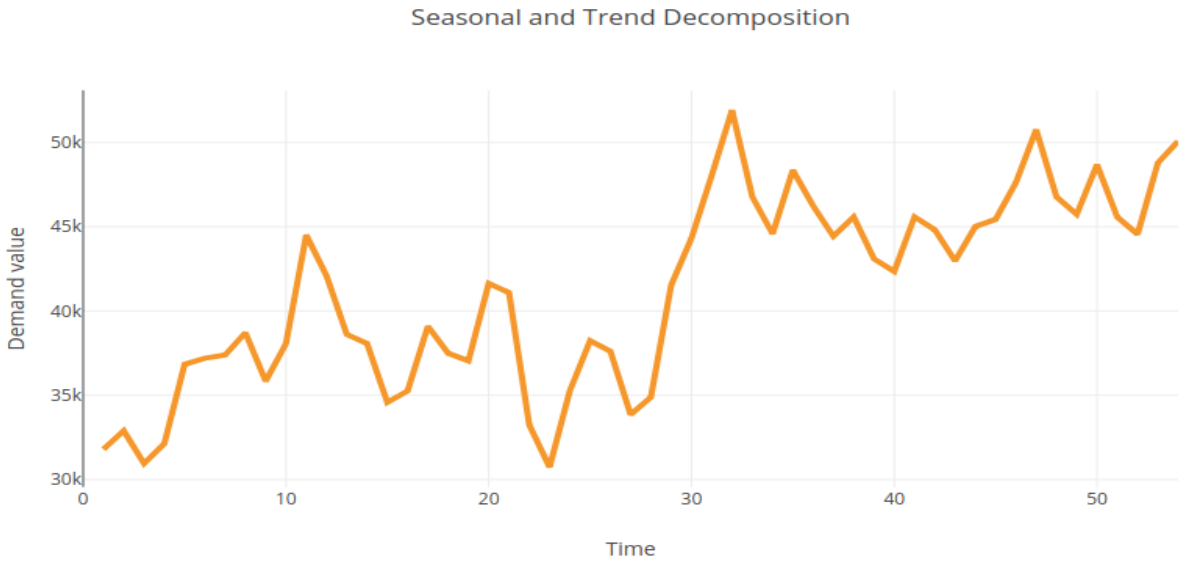

Fig. 4: Trend component in actual data

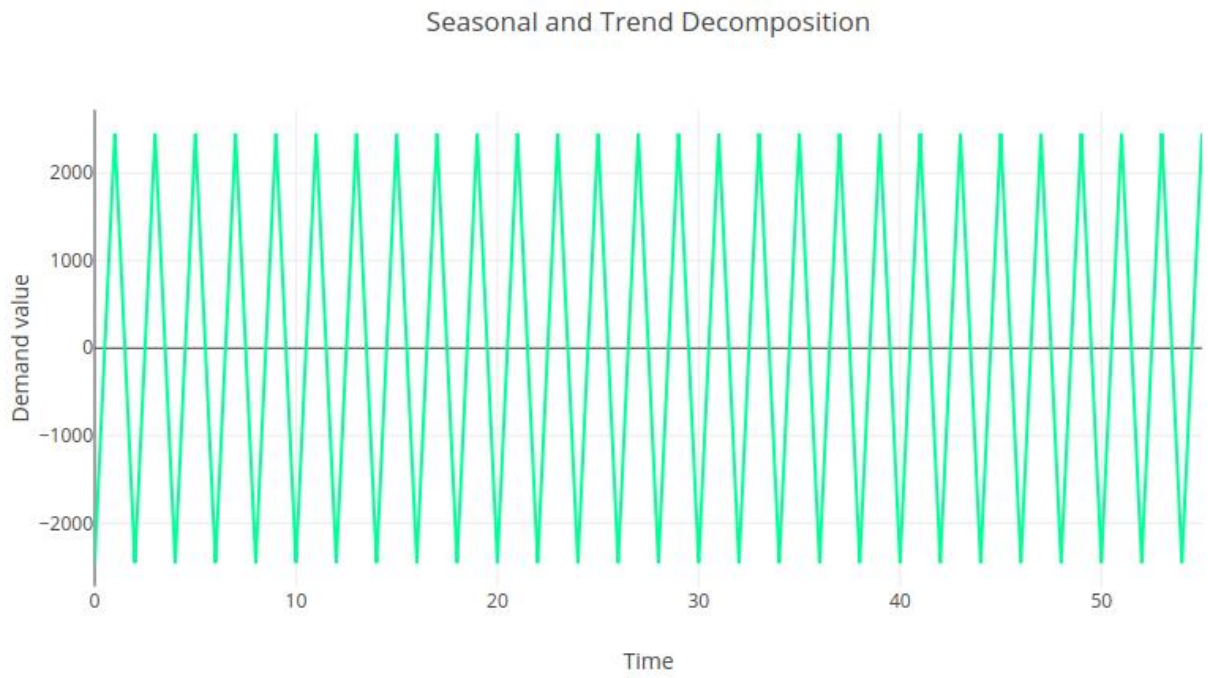

Fig. 5: Seasonal component in actual data

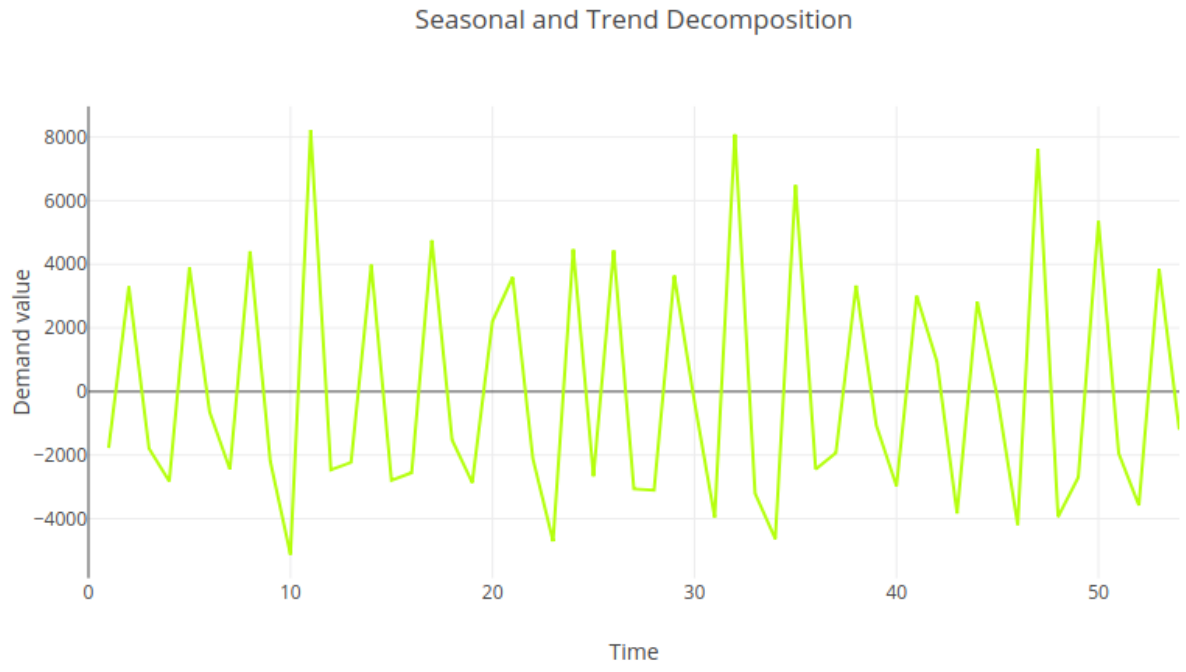

Fig. 6: Residual component

Fig. 6, shows the residual component of data or noise in data. Residual can be discovered by by subtracting trend and seasonality from actual data. If actual data is denoted by Y, trend by T, seasonality by S and Residual by R then Y = T + S + R and so R = Y - S - R.

It is also possible to detrend and de-seasonalize data by using Difference Transform and Log Transformation. In this example following points are done as pre-processing steps:

1. Normalization using z-score or min-max. (to bring data in same scale)
2. Log transformation
3. Difference Transform

In difference method, generally subtract previous observation with current observation i.e.:

D(t) = observation(t) - observation(t-1). Predicted value must be converted into original scale using equation I(t) = D(t) + observation(t-1). If there is still a trend in data then Difference can be repeated once more. To remove seasonality, first find the cycle of pattern in the data and subtract the cyclic pattern. The results of Log Transformation and Difference Transformation on input data are shown in Fig. 7 and 8 respectively.

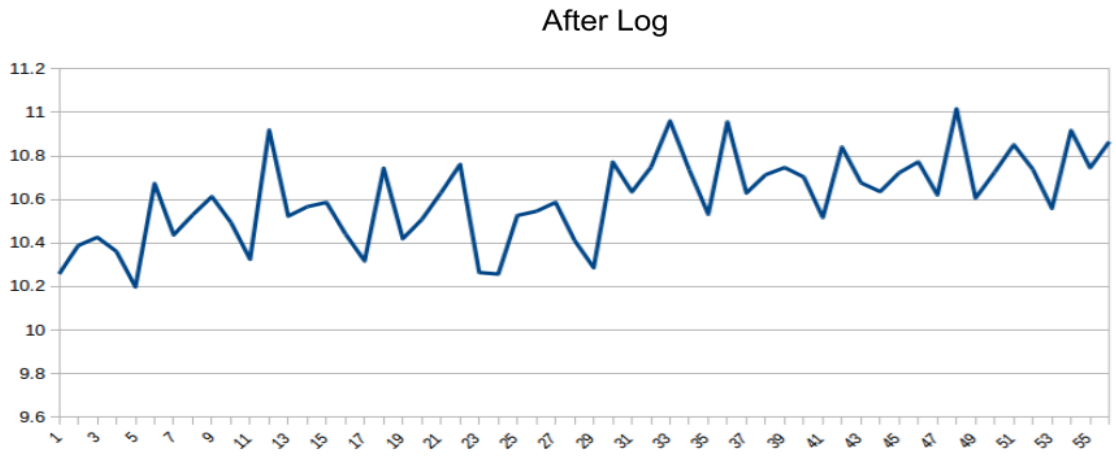

Fig.7: Normalized Log Transformation

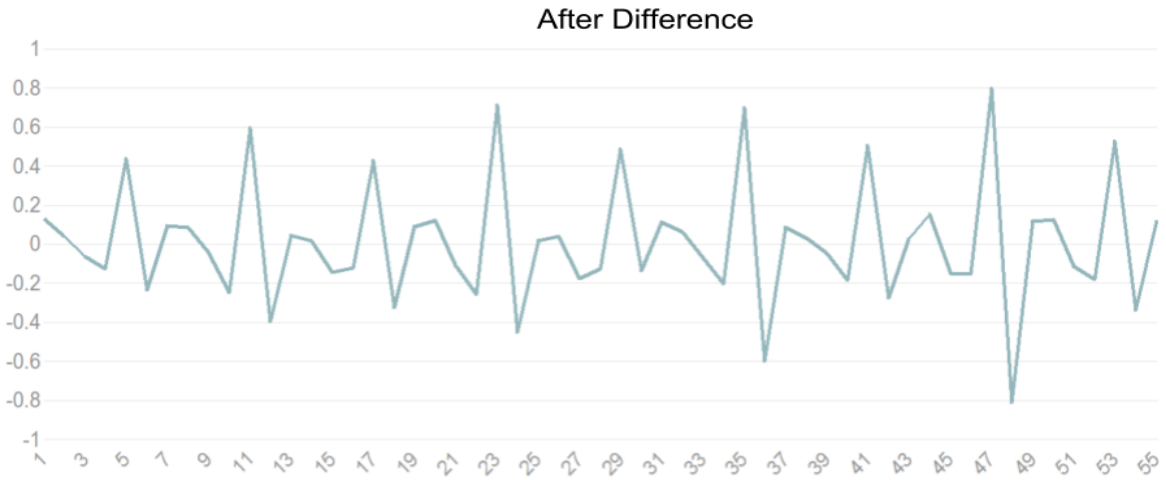

Fig. 8: Difference Transform

After applying Log and Difference transformations on input data, it gets fed into the LSTM model as input. The output from the LSTM model is again post processed (inverse Difference Transformation and inverse Log Transformation) to convert the data in the original scale.

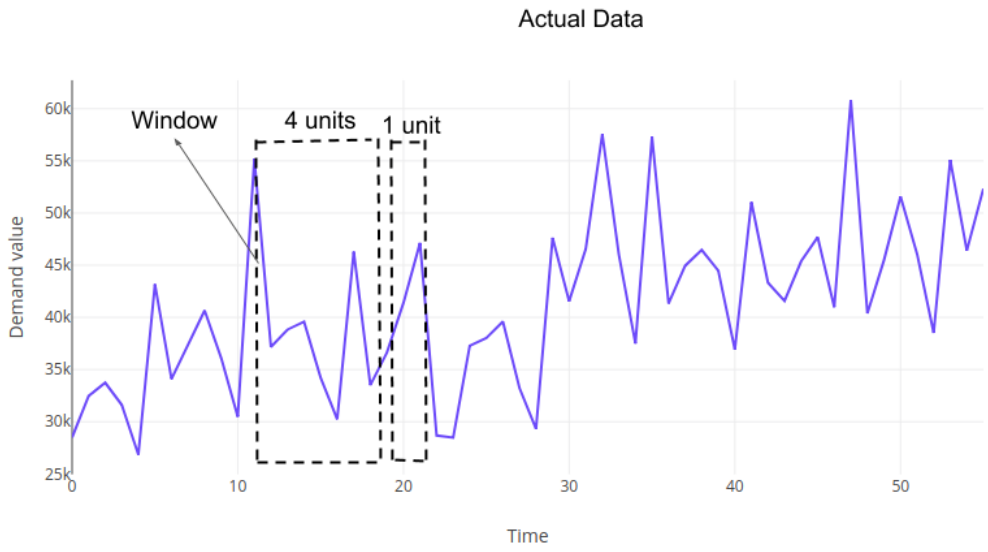

Fig. 9: Window method used to convert data into supervised learning

Input data for the LSTM model must be in a supervised form. In this example, Window method is used. As shown in Fig. 9, the window size for input is 4 so the first four values are selected as the 1st input sample. Similarly, window size for the output is 1. So the next value is selected as output. For all data, both windows are shifted by 1 unit to create a supervised dataset. For example, if the sequence of data is 2, 4, 7, 8, 9, 24, 12, 15, 32, 18; then the 1st sample pair of input and output would be:

Input	Output
2, 4, 7, 8	9
4, 7, 8, 9	24

...and so on. The data set is divided in the ratio of 70:30 for training and testing respectively.

Fig. 10 shows the result after using traditional forecasting techniques like a combination of Moving Average and ARIMA models. Fig. 10 shows the result from the LSTM model. MAPE is used to measure prediction accuracy. Traditional techniques result in a MAPE value around 25% whereas, the LSTM model results in a MAPE value around 8%.

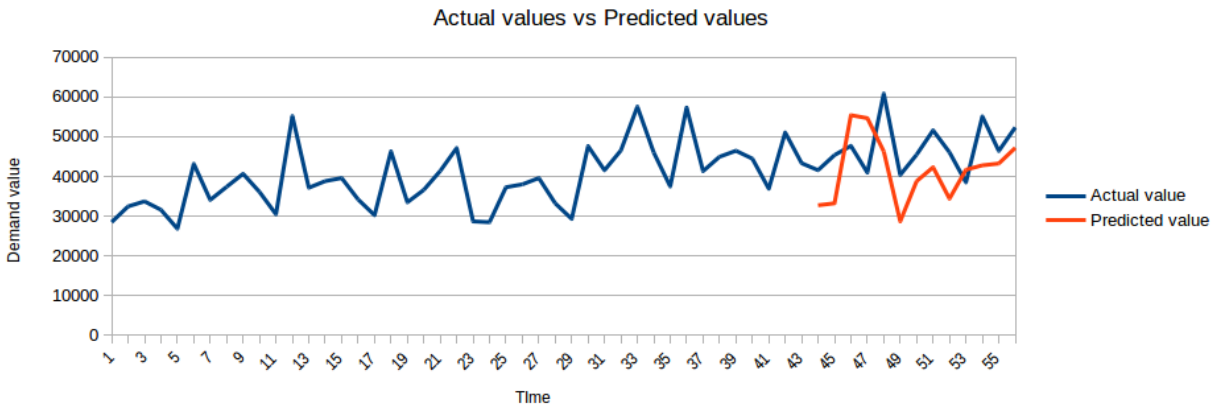

Fig. 10: Output from traditional forecasting techniques

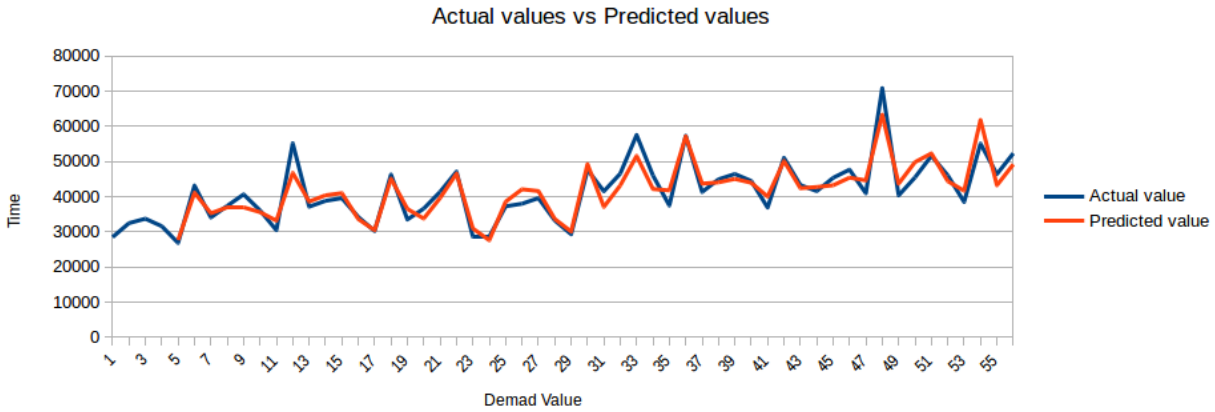

Fig.11: Output from LSTM model