Applied Machine Learning-2
High Frequency Time series

Arkaprava Sinha - MDS201801

Aishee Dasgupta - MDS201815

Rohan Khaitan - MDS201812

Subhasish Basak - MDS201803

Introduction

High frequency data refers to time-series data collected at an extremely fine scale. Largely used in financial analysis and in high frequency trading, high frequency data provides intraday observations that can be used to understand market behaviors, dynamics, and micro-structures.

Generic Time series models do not work well with high frequency data. Therefore we studied the data, visualized it, applied some statistical tools and experimented with various models like some well-known time series models (like ARIMA, SARIMA, Auto ARIMA) and LSTM and got some interesting results.

Data Description

PJM Interconnection LLC (PJM) is a regional transmission organization (RTO) in the United States. It is part of the Eastern Interconnection grid operating an electric transmission system serving all or parts of Delaware, Illinois, Indiana, Kentucky, Maryland, Michigan, New Jersey, North Carolina, Ohio, Pennsylvania, Tennessee, Virginia, West Virginia, and the District of Columbia.

The hourly power consumption data comes from PJM's website and are in megawatts (MW).

For convenience we use the data for Kentucky only. The same algorithm may be replicated for the other states.

East Kentucky Power Cooperative (EKPC) has collected the data for Kentucky. The data is the hourly consumption of electricity from June 2013 to August 2018. There are about 46000 data points in the dataset.
So we need to do a time series analysis. Before describing the implementation part we first give short description of about time series and required methods.

Time series data:

Data observed collected or recorded over time is called Time series data. For this project the collected dataset on energy consumption is an example of time series data. The simplest mode of diagrammatic representation of Time series data is the use of Time series plot. Taking two perpendicular axes of coordinates, the vertical one for the variable under study) & the other for time points, each pair of values is plotted. The resulting set of points constitutes the Time series plot

Components of Time-series:

Stationary Component - If present we say the data, Stationary Data
Non-stationary Component- If present we say the data, Non-Stationary Data

Short Description: Non-stationary as the name suggests, the changes in the values of a time series variable over a period of time is due to the Non-stationary component present in that time series. Analysis of Non-stationary in time series calls for:

Identification of the components of which the time series is made up of.
Isolation and measurement of the effects of these components
separately and independently holding the other effects constant.

To analyse the Non-stationary part of the time series in hand, We use classical decomposition. In the Classical approach of Time series analysis it is assumed that any time series Y t is composed of four major components viz.

- Trend (T_t)
- Seasonal component (S_t)
- Cyclical component (C_t)
- Irregular component (I_t)

Stationary Component is of Two types - 1. Strongly Stationary 2. Weakly Stationary

Test for stationarity : The Augmented DickeyFuller test(ADF), tests the null hypothesis that a unit root is present in a time series sample. A unit root is a feature of some stochastic processes such that, a linear stochastic process has a unit root if 1 is a root of the process's characteristic equation. Such a process is non-stationary but does not always have a trend.

Thus it is equivalent to state the Hypothesis of interest of an ADF test as,

H_0 : The series is non-stationary.

H_1 : The series is stationary.

The methods and tools we used are:

- **Autocorrelation plot (ACF)**: it is the similarity between observations as a function of the time lag between them.
- **Partial autocorrelation function**: It gives the partial correlation of a stationary time series with its own lagged values, regressed the values of the time series at all shorter lags.

  These plots are used to decide the order of the Moving average and Auto-regressive Process.

- **Autoregressive process of order p (AR(p))** : it is used to describe certain time-varying processes in nature, economics, etc. The autoregressive model specifies that the output variable depends linearly on its own previous values and on a stochastic term (an imperfectly predictable term); thus the model is in the form of a stochastic difference equation. In machine learning, an autoregressive model learns from a series of timed steps and takes measurements from previous actions as inputs for a regression model, in order to predict the value of the next time step.
- **Moving Average process of order q (MA(q)) :** It works in same way of Autoregressive process. Together with the autoregressive (AR) model, the moving-average model is a special case and key component of the more general ARMA and ARIMA models of time series, which have a more complicated stochastic structure.

Implementation

Visualisation: First we visualize the raw data.

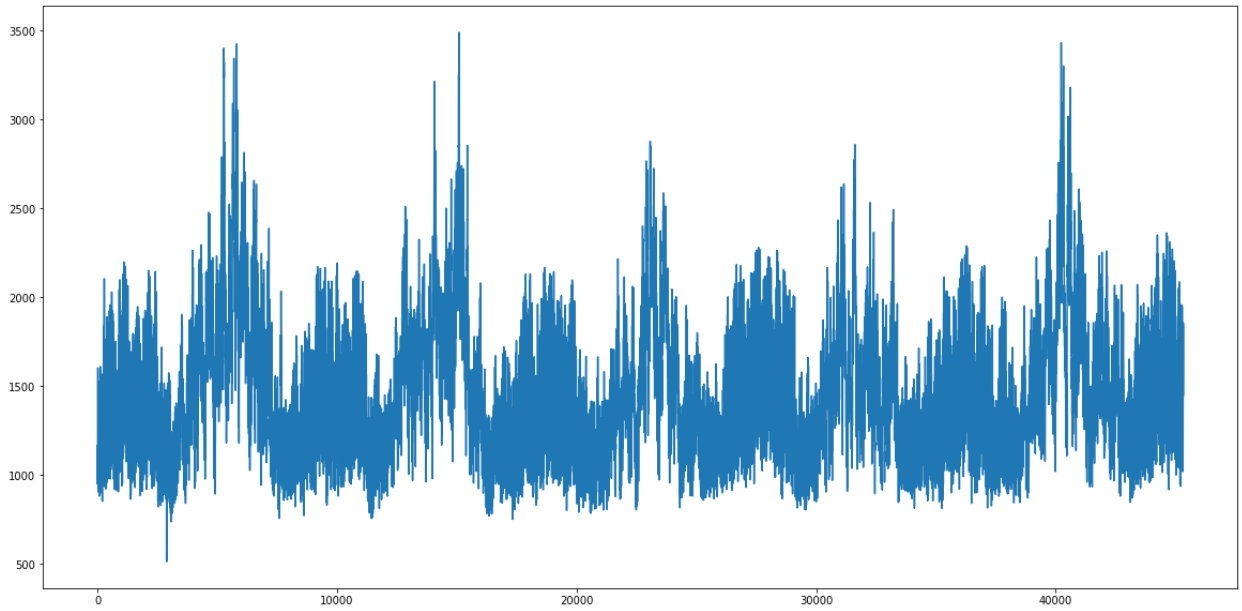


Figure: Hourly Energy Consumption

Though we get a rough idea about the behaviour of the data ,Visualizing doesn't give us a clear idea whether the data is stationary or not.

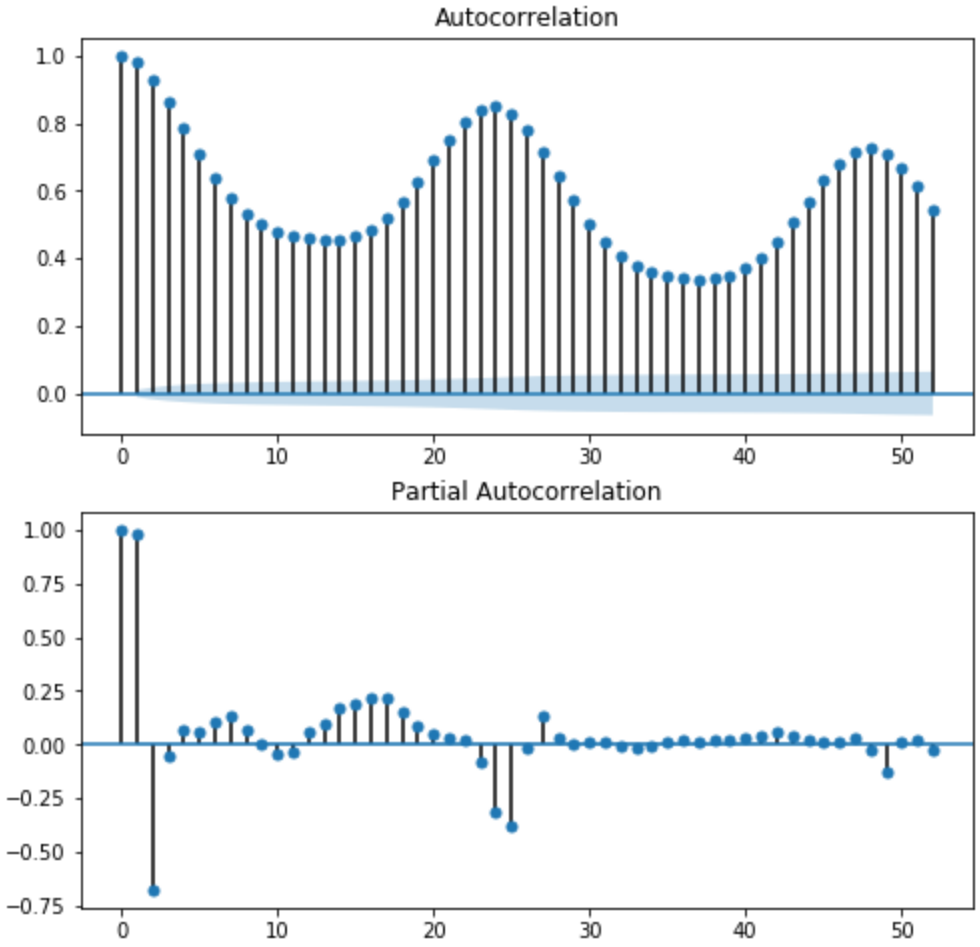
Test for stationarity: So we need to check for stationarity.So we did Augmented-Dickey Fuller test.

- P-Value - 2.895578e-19
- Null Hypothesis Rejected
- Stationary Data

We find that our hourly energy consumption data is stationary.So now we fit the ARIMA model.

Order of the parameters of the model:

As the data is found to be stationary we take the order of differencing as 0.Now we need to decide the order of the AR and MA process.We decide it based on the Autocorrelation plot and Partial Autocorrelation plot.



After deciding the orders of the parameters we fit our ARIMA model.

Visualisation: We plot test data and the prediction on the test data in the same graph .

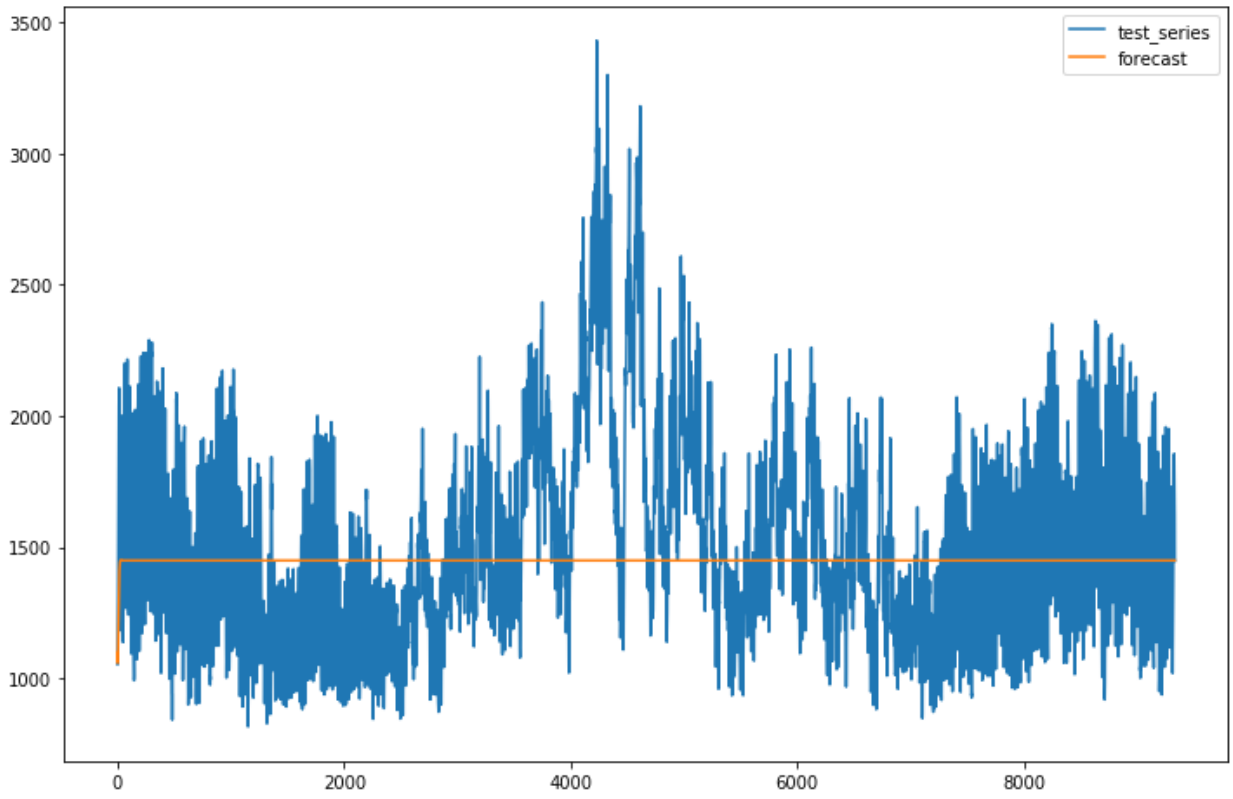


Figure: Prediction on Hourly Test Data

Comment: ARIMA models fits very badly. It is due to the presence of various seasonal factors. So we can see that in the presence of various seasonal factors ARIMA model fails. So we have to modify our model i.e. we need to take care of the seasonal factors so that we can get a better prediction for the future values.

Python Libraries for ARIMA model:

- ARIMA
- SARIMA
- Auto-Arima

ARIMA: It simply takes the parameters-order of AR and MA process and order of differencing.

SARIMA is seasonal ARIMA. It gives us the flexibility to get rid of a seasonal factor by giving the period of seasonality. For eg- For monthly data we can give a period 12 as one year contains 12 months.

Auto-Arima: It gives us the flexibility to provide a range of the order of processes and it computes AIC and BIC score for each combination of the values of the parameters and gives the best possible result within that given range.

It needs very high computation power to run Auto-arima model with huge amount of data. Even in SARIMA model we require high computation power. ARIMA model with parameters (24,0,2) took 10 mins in a system with 16 gb ram and i7 7th gen processor. So running Auto-arima and SARIMA model is very difficult without a system with very high specifications.

Daily Energy Consumption:

We convert our hourly data to daily data.

Visualisation: We then visualize the daily energy consumption data.

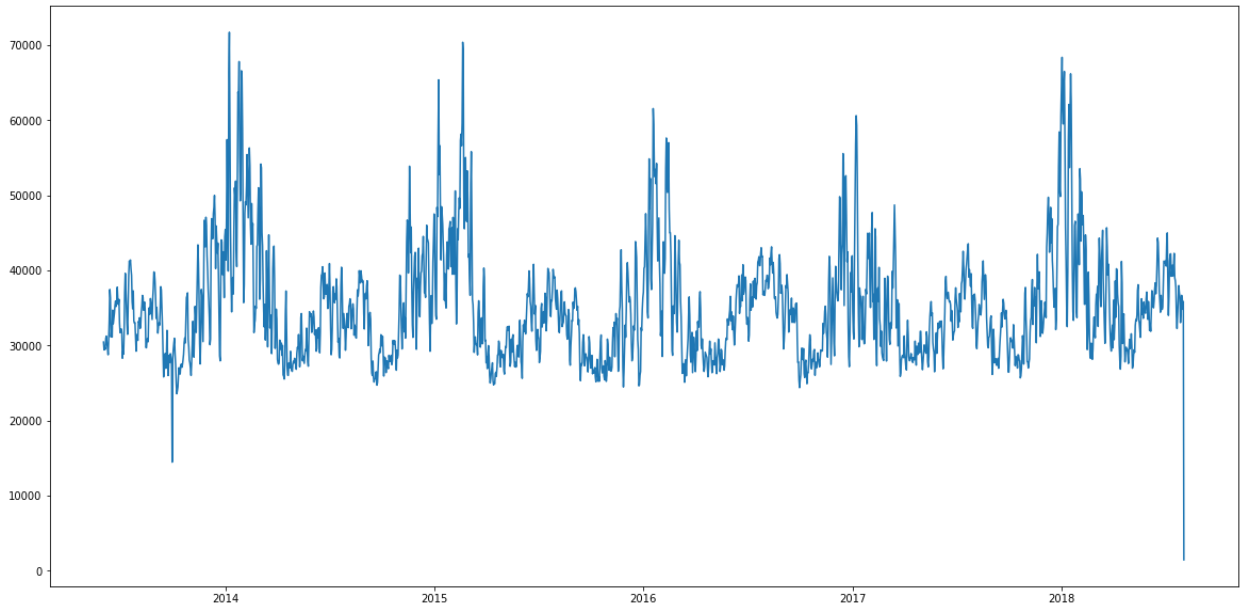


Figure: Daily Energy Consumption

Again we cannot conclude about the stationarity of the data only by visualizing it.
We follow the same steps as we did in case of hourly data.

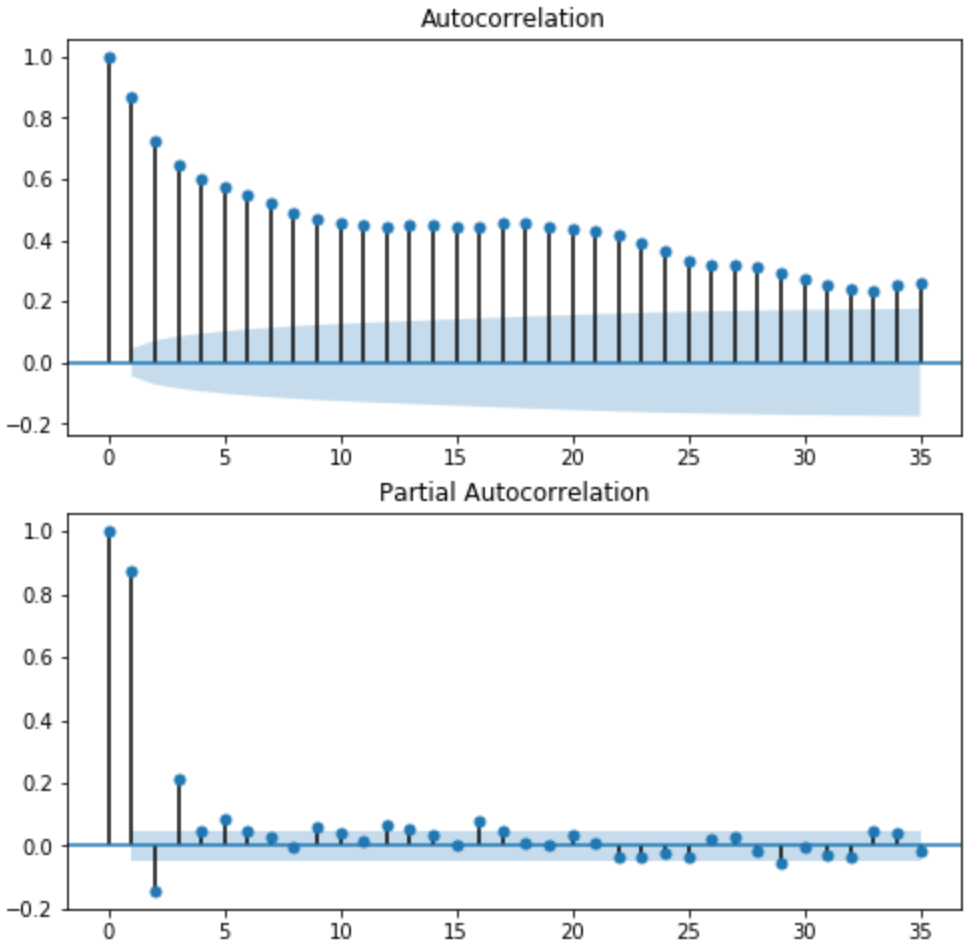
Check for stationarity: We did Augmented Dicky-Fuller Test.

- P-Value - 0.000171
- Null Hypothesis Rejected
- Stationary Data

So again We found the data to be stationary.

Order of the parameters of the model:

Now we need to decide the order of the AR and MA processes.
We decide it based on the Autocorrelation plot and Partial Autocorrelation plot.



After deciding the orders of the parameters we fit our ARIMA model.

Visualisation: We plot test data and the prediction on the test data in the same graph .

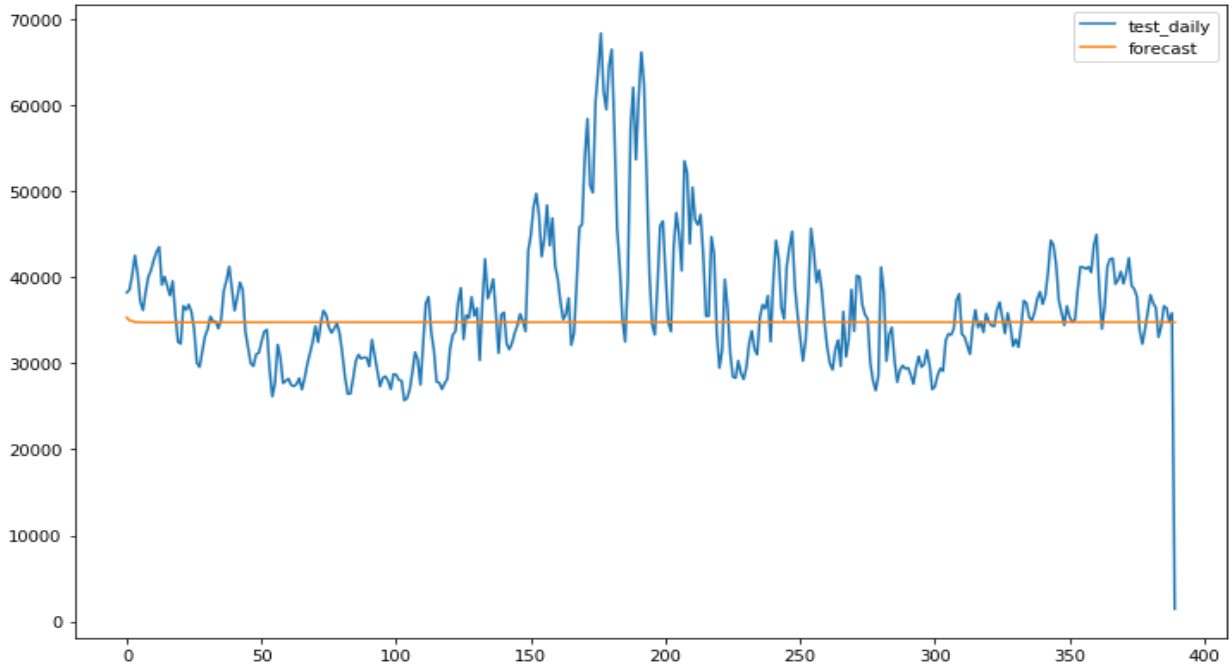


Figure: Prediction on Daily Test Data

Comment : ARIMA model again fit very badly. We removed the hourly fator by transforming the data to daily energy consumption data.It means that the data contains other seasonal factors such as weekly,monthly, yearly etc besides hourly seasonality. So ARIMA model fails for this High-frequency time series data.

Problems in fitting ARIMA model:

It Can not capture all the seasonality present in the data. Again it fails to accurately give a solution for the following cases-

- Monthly Data - No of days in a month,may be 30 or 31 or 28.
- Yearly Data - Leap Year
- Holidays.

Fitting Prophet Model :

Prophet is an open source library published by Facebook that is based on decomposable (trend+seasonality+holidays) models. It provides us with the ability to make time series predictions with good accuracy using simple intuitive parameters and has support for including impact of custom seasonality and holidays!

Advantages of using Prophet model :

Prophet can handle the following situations -

- Hourly, daily, or weekly observations with at least a few months of history
- Important holidays that occur at irregular intervals that are known in advance
- A reasonable number of missing observations or large outliers
- Historical trend changes, for instance due to product launches

Visualisation:

We plot the Train data and prediction on test data-

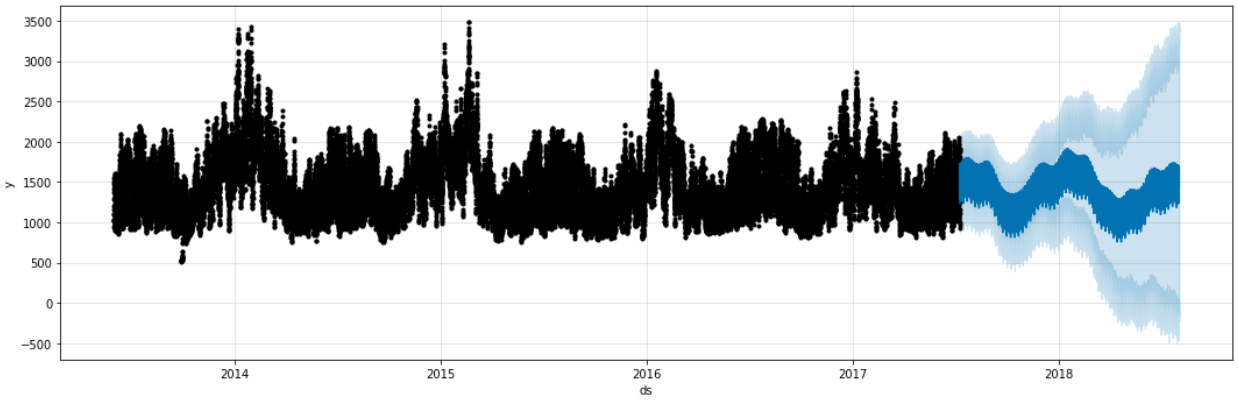
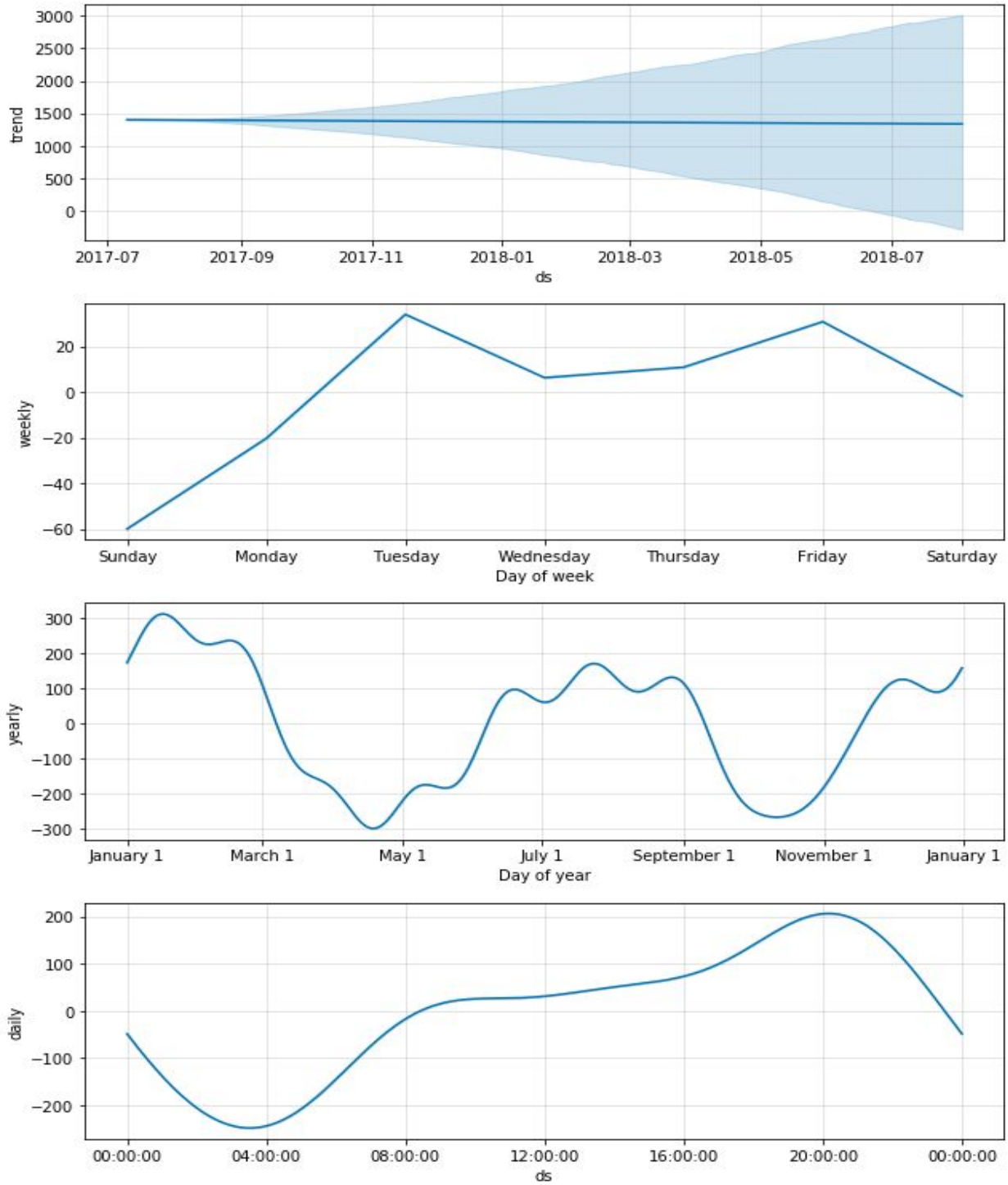


Figure: Train and Predicted Test Data

Various seasonal factors -



So we can see that various seasonal factors are present in the data.

After fitting the Prophet model we now get the following plot -

Plot of test data and the prediction on test data:

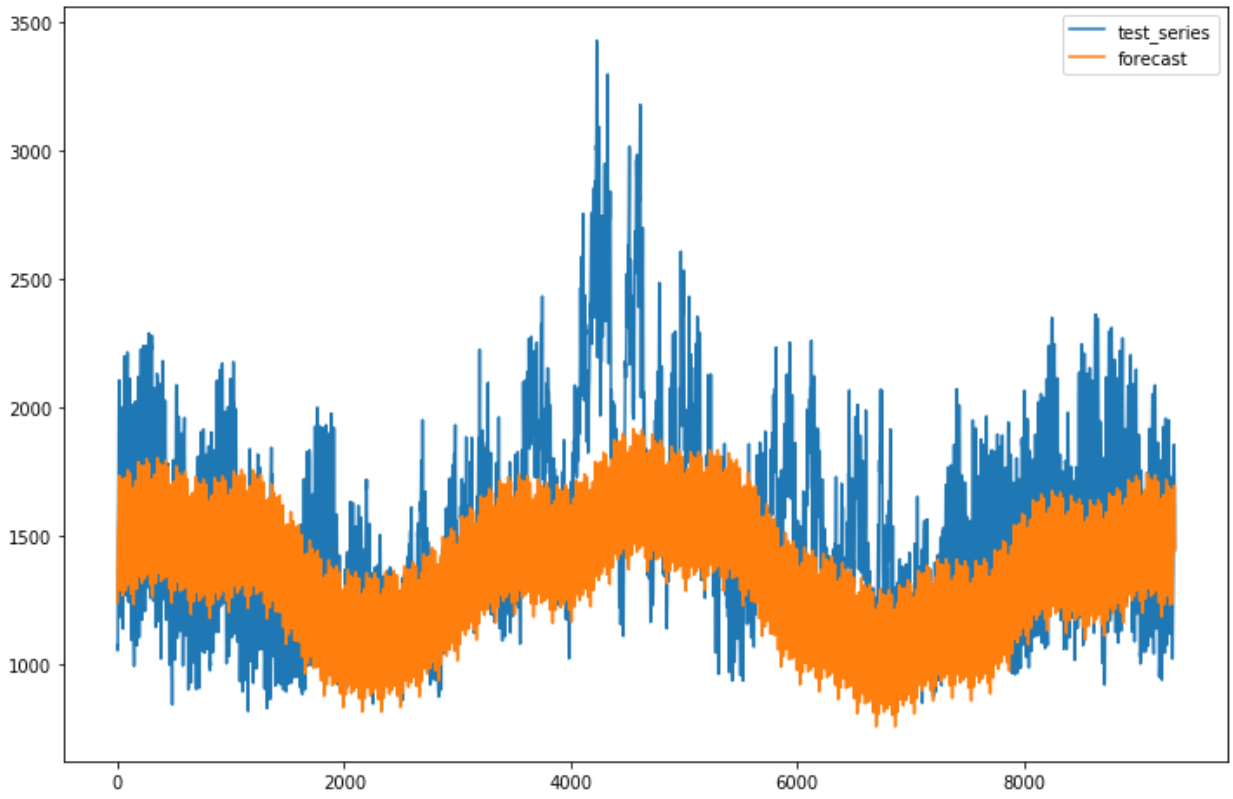


Figure: Test Data and Predicted Values

Comment: Prophet model fit relatively better than the ARIMA model.

LSTM:

Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. They work tremendously well on a large variety of problems, and are now widely used.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior.

Here we see that LSTM gives a good fit to the data and performs exceptionally well on previously unseen test data.

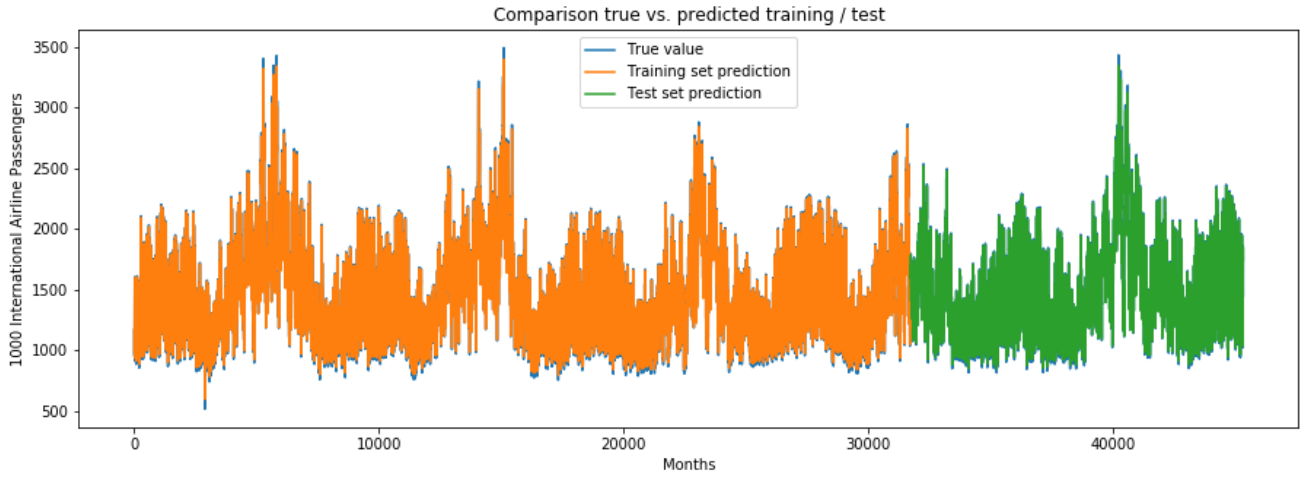


Figure: Actual and Predicted Values

In the figure above the orange part of the plot is the train data and the green part is the test data. The original data is in black. We see that the predictions of the LSTM almost overlaps the original data.

Conclusion:

We fit several models to the high-frequency time series data. Among all the models we see that LSTM outperforms all the other models by a huge margin.

Thus LSTM may be used to fit high frequency time series data. Now with the model we can project the consumption of electricity in the future and make necessary decisions about energy consumption projects like installing solar panels etc.