Time Series Analysis

Hourly Energy Consumption

Rajas Vaidya (MDS202131)
Rishika Tibrewal (MDS202135)

Project Report

Contents

1 Dataset 2

2 Objective 2

3 Data Cleaning and summary 2

4 Exploratory Data Analysis 2

5 Classical Decomposition 3

6 Augmented Dickey Fuller test for stationarity 3

7 Train-Test Split 4

8 Prophet model 4
8.1 Introduction to Prophet model 4
8.2 Prophet model components 5
8.3 Prophet model results 6
8.4 Prophet model evaluation metrics 6

9 XGBoost 6
9.1 Introduction to XGBoost 6
9.2 When can we use XGboost? 6
9.3 Feature Engineering 7
9.4 Patterns in the data 7
9.5 Feature Importance 8
9.6 XGBoost results 8
9.7 XGBoost model evaluation metrics 8
9.8 Best and worst predicted days of XGBoost 9

10 Conclusion 9

11 References 10

1 Dataset

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.

We worked with hourly power consumption data of PJM Interconnection LLC from 1st Jan 2002 to 2nd Aug 2018.

2 Objective

- To explore and visualize the dataset.
- To check for stationarity and any underlying patterns in the data over the years.
- To check whether there is any trend or seasonality in the data.
- To build models on the dataset to predict energy consumption.
- To test the models on the test set and compare the results.

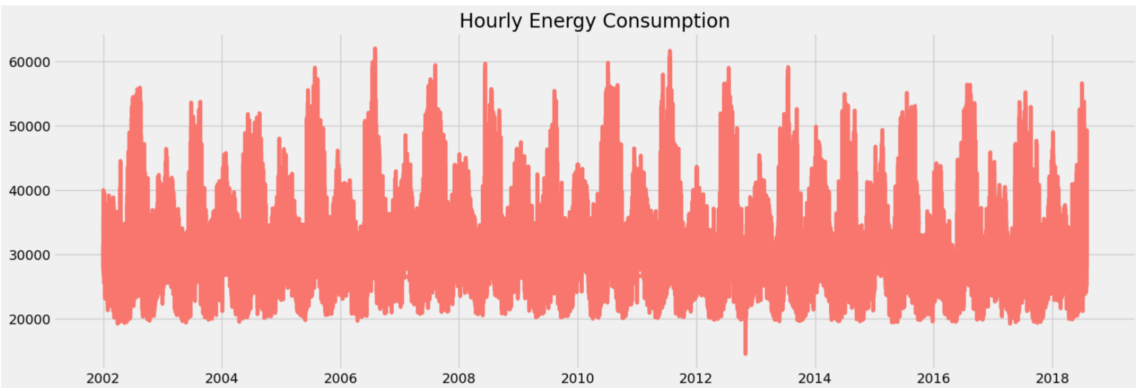
3 Data Cleaning and summary

There are no null or missing values in the dataset.

Statistic	**Energy**
count	145366
Mean	32080.222831
Standard deviation	6464.0122
Min	14544
25%	27573
50%	31421
75%	35650
max	62009

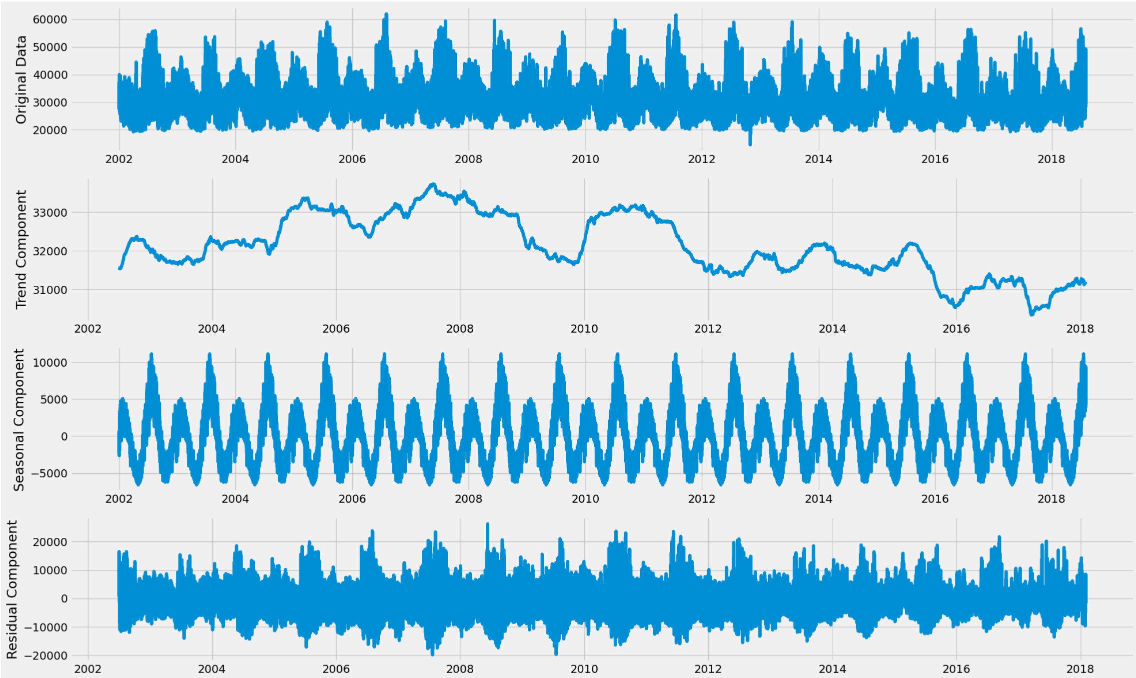
Note: All values are in Megawatts

4 Exploratory Data Analysis



It can be noticed that there is no trend i.e., that is no long term increase or decrease in the values of observations. The data definitely has some seasonality, as there are periodic changes at regular intervals.

5 Classical Decomposition



After carrying out the classical decomposition, we found out that there was no trend but clearly a seasonal component was present.

6 Augmented Dickey Fuller test for stationarity

The Augmented Dickey-Fuller (ADF) test is a statistical test commonly used to test for the presence of a unit root in a time series dataset. A unit root is a characteristic of a time series dataset where the mean and variance of the series are not constant over time.

The ADF test is a type of unit root test that checks whether a time series dataset is stationary or not. If the test indicates the presence of a unit root, then the dataset is non-stationary, which means the data has a trend and does not have a constant mean and variance. In contrast, if the test indicates the absence of a unit root, then the dataset is stationary, which means the data has a constant mean and variance.

The ADF test is widely used in econometrics and finance, where time series data is commonly used. The test can be applied to various types of time series data, such as macroeconomic variables, financial market data, and stock prices. The ADF test is a useful tool for determining the appropriateness of statistical models and for forecasting future values of a time series.

Result of ADF test :

```
Observations of Dickey-fuller test
Test Statistic                  -1.882891e+01
p-value                          2.022125e-30
#lags used                       7.400000e+01
number of observations used      1.452910e+05
critical value (1%)             -3.430395e+00
critical value (5%)             -2.861560e+00
critical value (10%)            -2.566781e+00
dtype: float64
```

As we can see, the p-value is less than 0.05. Hence we rejected the null hypothesis to conclude that the data is stationary.

7 Train-Test Split

For building the models, we split the data into two parts: training data and test data.

The time period of 1st Jan 2002 to 31st Dec 2014 was taken for training the model and the time period of 1st Jan 2015 to 2nd Aug 2018 was taken as the test data.

8 Prophet model

8.1 Introduction to Prophet model

Prophet is a time series forecasting model developed by Facebook's Core Data Science team in 2017. It is an open-source model based on the additive regression model, which is designed to make accurate predictions of time series data with multiple seasonality and changing trends.

Prophet model is easy to use and requires minimal data preparation, making it a popular choice for forecasting in industries such as finance, retail, and manufacturing. It can handle missing values, outliers, and changes in trend and seasonality. Prophet decomposes the time series data into trend, seasonality, and holiday components, allowing it to capture the underlying patterns of the data.

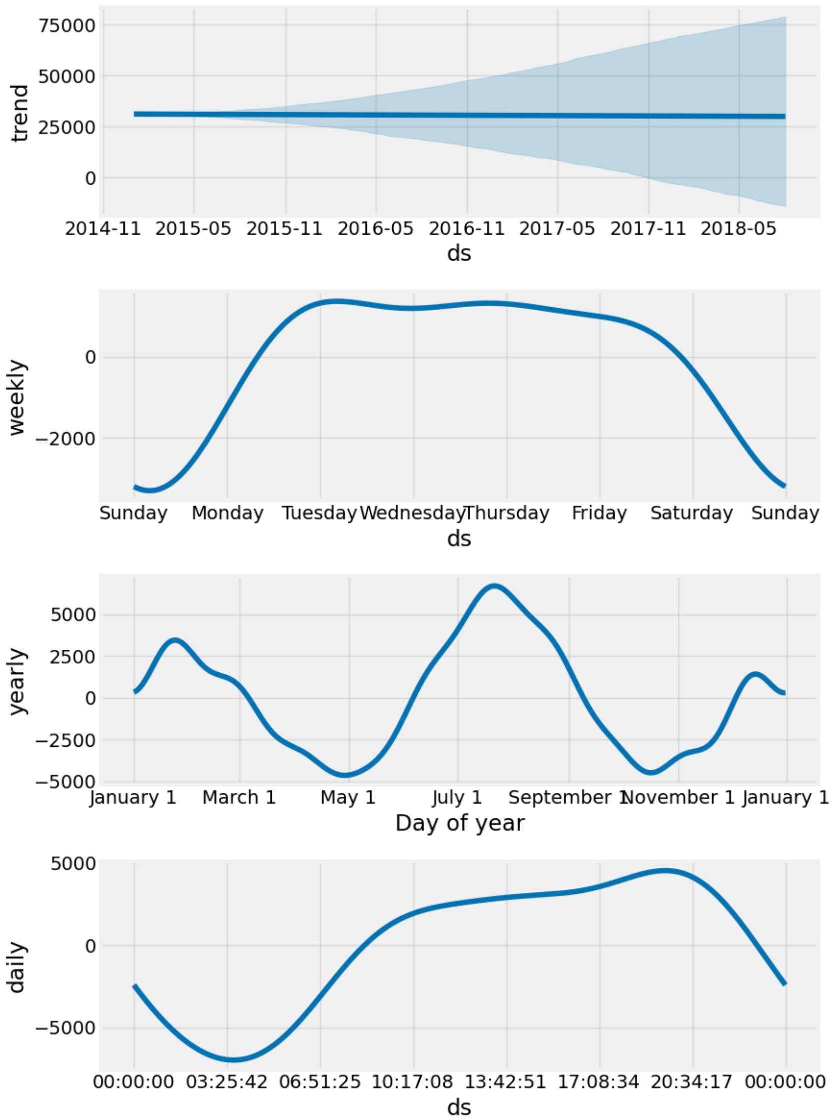
The model uses Bayesian modeling techniques to generate a probabilistic forecast, which provides a range of possible outcomes and their associated probabilities. This allows users to quantify the uncertainty in their forecasts and make informed decisions based on the probability distribution of future values.

Prophet is based on an additive regression model that decomposes the time series into four components: **trend**, **seasonality**, **holidays**, and **an error term**.

$$y(t) = g(t) + h(t) + s(t) + \epsilon(t)$$

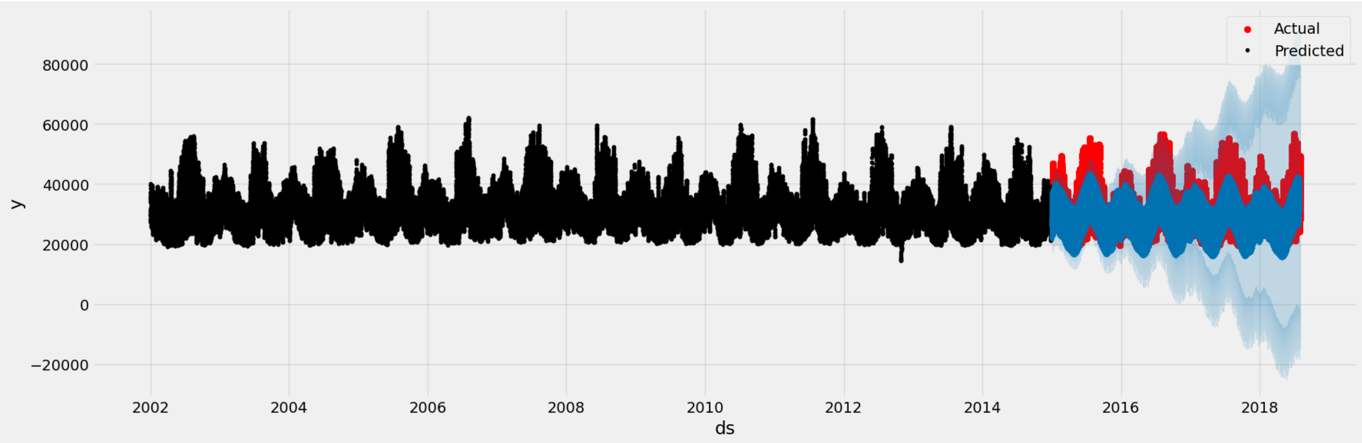
where $g(t)$ describes a piece-wise linear trend (or "growth term"), $s(t)$ describes the various seasonal patterns, $h(t)$ captures the holiday effects, and $\epsilon(t)$ is a white noise error term.

8.2 Prophet model components



- As we can see, there is a flat line in the plot for trend. That confirms that there is no trend in the data.
- In the weekly plot, we notice that the power consumption is higher on weekdays (Monday-Friday) and drops down on weekend. This is perhaps because on weekdays, the offices, factories etc. consume a lot of electricity.
- In the yearly plot, we notice that the electricity consumption decreases during the months March to May and records a big jump during the months June to September. It again shows a big drop in the last 2 months of the year. That might be because, U.S. witnesses spring season in March-May and Summer in June-August, due to usage of Air conditioners, summer witnesses a jump in power consumption. Then again as in Autumn season, the consumption drops down while the winter, December-February records a jump due to usage of heaters.
- In the daily plot, we notice that 10 AM to 9 PM sees a lot of power consumption as these are the working hours whereas the night witnesses a drop in the consumption.

8.3 Prophet model results



Here the black lines represent the training data. The red lines are the actual data points of test data whereas the dark blue lines are the predictions made on test dataset by the Prophet model. The faint blue portion is the confidence interval for the predictions. We can see that our prophet model has managed to capture the general pattern and seasonality of the data.

8.4 Prophet model evaluation metrics

Metric	value
Mean absolute error	3104.6694
Mean squared error	16984051.4039
Mean absolute percentage error	9.62
R-square	0.59

We tried another model to see if we can get better results. Looking at the length of the data and high seasonality, we decided to use the XGBoost model.

9 XGBoost

9.1 Introduction to XGBoost

XGBoost stands for extreme gradient boosting machine. It is a tree based ensemble machine learning algorithm which is a scalable machine learning system for tree boosting. It uses more accurate approximations to find the best tree model. Using XGBoost for time-series analysis is an advanced approach of time series analysis that helps in improving our results and speed of modelling.

9.2 When can we use XGboost?

We can consider using XGBoost for any supervised machine learning task when it satisfies the following criteria:

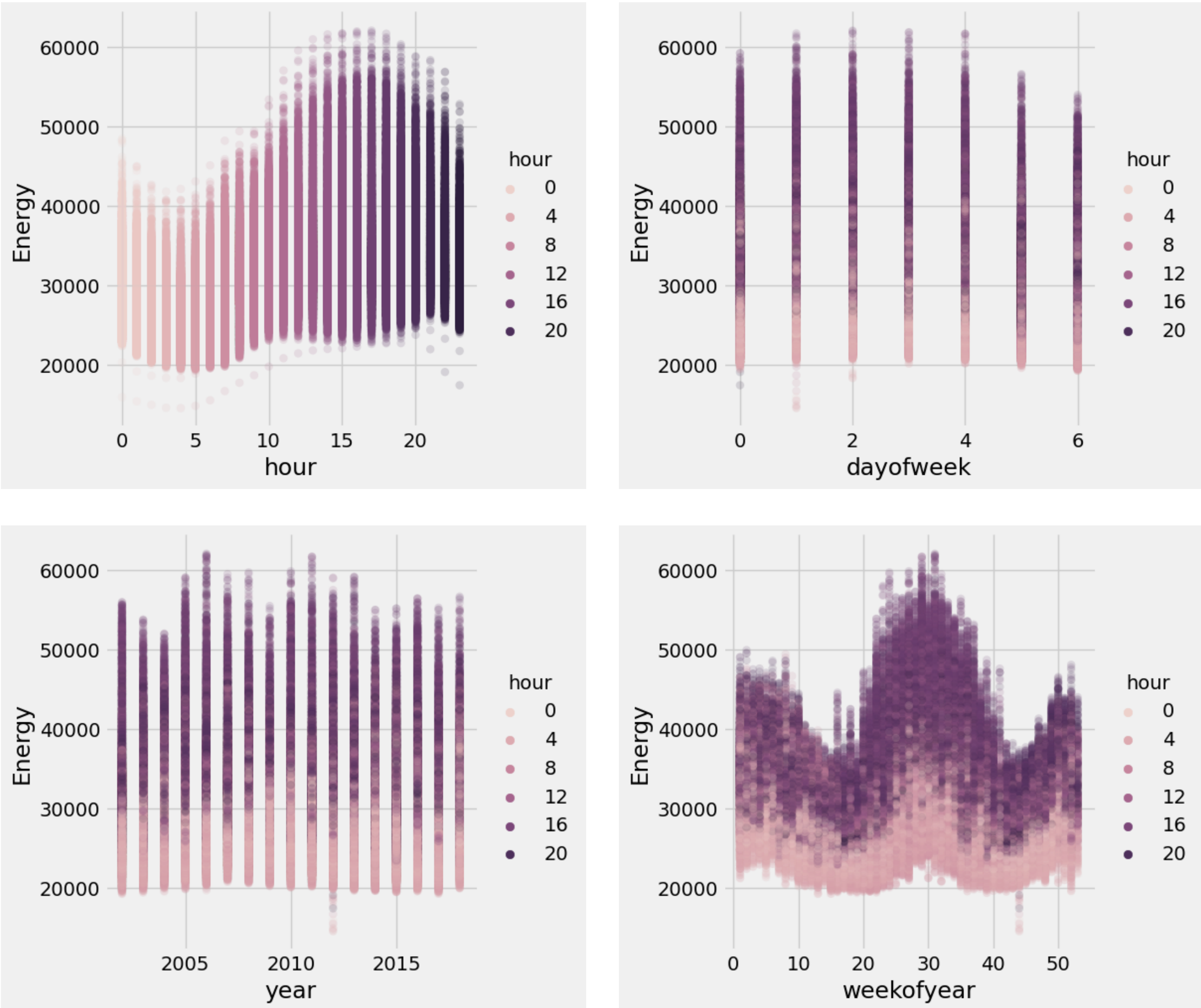
- When we have large number of observations in training data.
- When Number of features is less than the number of observations in training data.
- When the model performance metrics are to be considered.

Since we have a large data and fewer features, we can adopt a XGBoost model for our purpose.

9.3 Feature Engineering

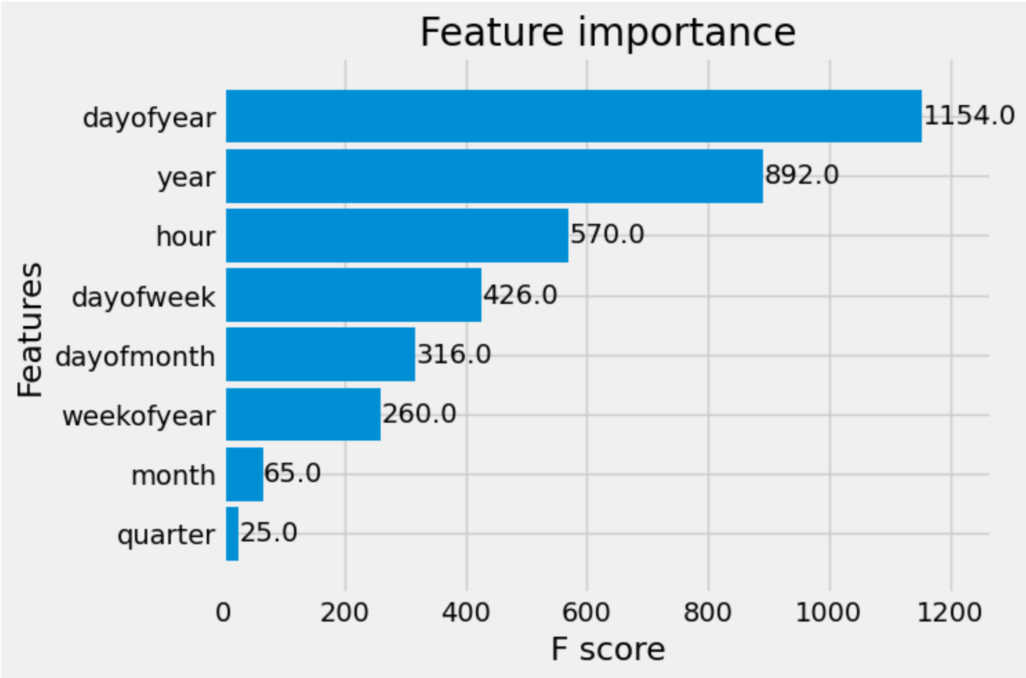
Before building the XGBoost model, we created few features for the observations such as day of week, quarter, month, year, hour of the day etc. which will be used for the model.

9.4 Patterns in the data

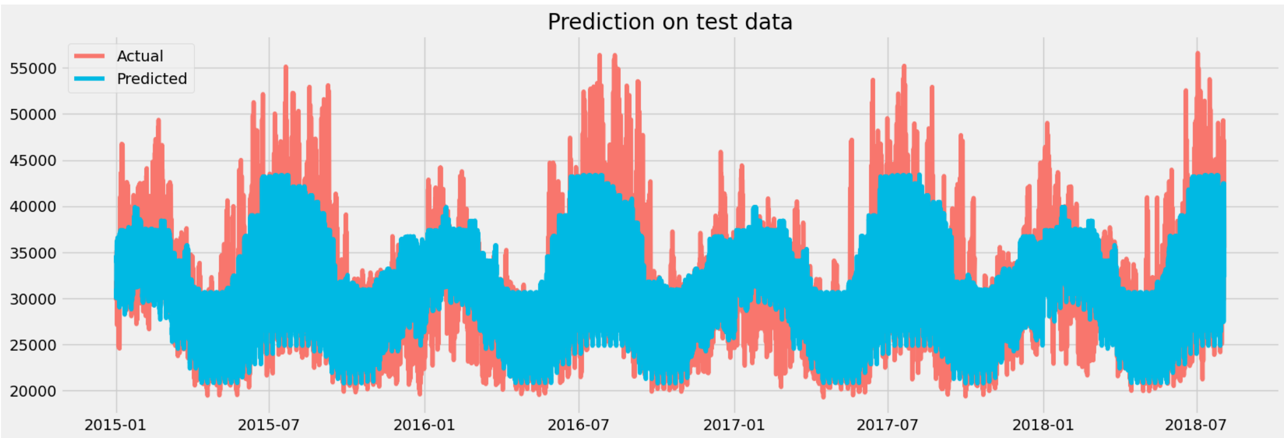


Similar patterns were observed as the Prophet model.

9.5 Feature Importance



9.6 XGBoost results

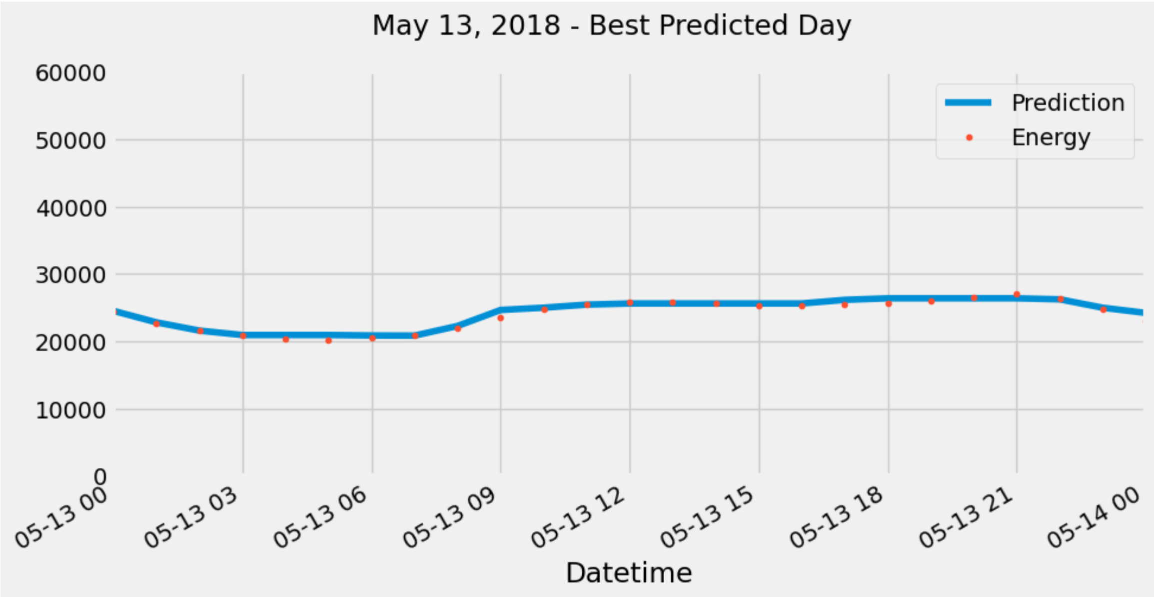


This graph displays the comparison of predicted values and the actual values for the test data. The blue portion shows the predictions made by the model and the red portion shows the actual observations of the test data.

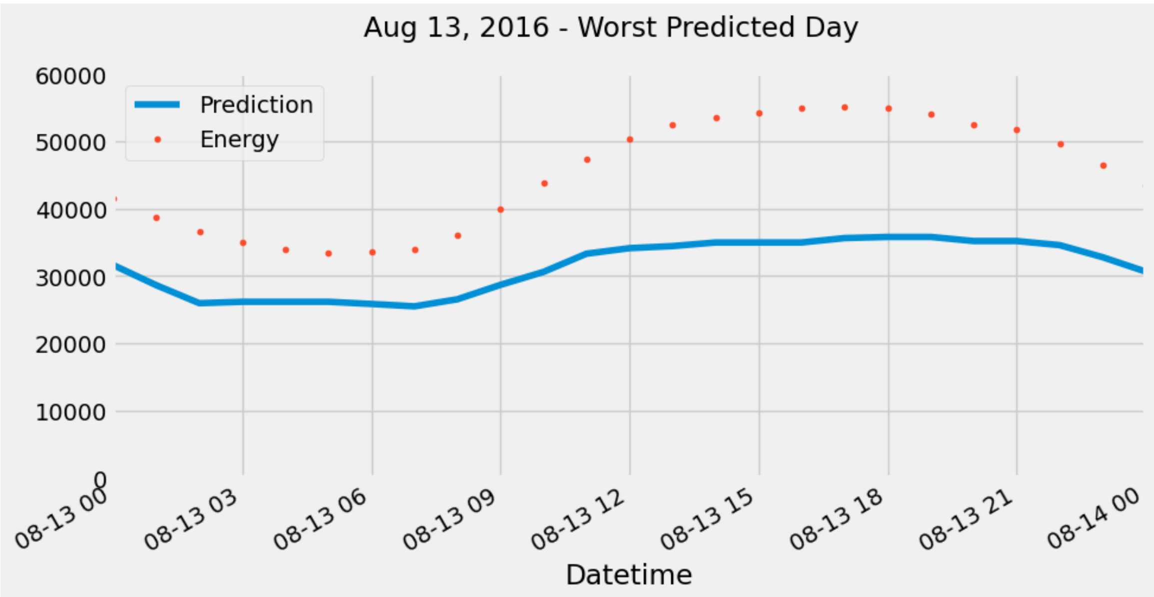
9.7 XGBoost model evaluation metrics

Metric	value
Mean absolute error	2757.2169
Mean squared error	13960913.3056
Mean absolute percentage error	8.5108
R-square	0.6644

9.8 Best and worst predicted days of XGBoost



May 13, 2018 was the best day in terms of predictions made by the XGBoost model.



Aug 13, 2016 was the worst day in terms of predictions made by the XGBoost model.

10 Conclusion

Performance Measure	Prophet	XGBoost
Mean Absolute Error	3105	2757
Mean Square Error	16984051	13960913
Mean Absolute percentage Error	9.62%	8.51%
R Square	0.59	0.66

Both Prophet and XGBoost model worked well on the data. While there was not much difference in the mean absolute percentage percent error (9.62 and 8.51), the R square value increased from 0.59 to 0.66 when we switched to XGBoost from prophet.

11 References

- https://www.kdnuggets.com/2020/12/xgboost-what-when.html
- https://analyticsindiamag.com/how-to-use-xgboost-for-time-series-analysis/
- https://blog.exploratory.io/an-introduction-to-time-series-forecasting-with-prophet