**Jobathon Nov 2022 Time Series Forecasting Approach**

Table of Contents

[Model(s)’ Performance 1](#__RefHeading___Toc539_3937304071)

[Exploratory Data Analysis 2](#__RefHeading___Toc541_3937304071)

[Libraries Used 2](#__RefHeading___Toc543_3937304071)

[Visualizing Data 2](#__RefHeading___Toc545_3937304071)

[Check for Seasonality 3](#__RefHeading___Toc547_3937304071)

[Check for Yearly seasonality 3](#__RefHeading___Toc549_3937304071)

[Check for 24 hours seasonality 3](#__RefHeading___Toc551_3937304071)

[Check for Weekly seasonality 5](#__RefHeading___Toc553_3937304071)

[Seasonal Decomposition 6](#__RefHeading___Toc555_3937304071)

[Determination of order of Auto Regression and Moving Average 8](#__RefHeading___Toc557_3937304071)

[Model Development 10](#__RefHeading___Toc559_3937304071)

[Model Training and Evaluation 10](#__RefHeading___Toc561_3937304071)

[Model Finalization 10](#__RefHeading___Toc563_3937304071)

[Baseline Performance 10](#__RefHeading___Toc565_3937304071)

[Why specific model is chosen ? 10](#__RefHeading___Toc567_3937304071)

[Libraries Used 10](#__RefHeading___Toc569_3937304071)

[Constraints with direct ARIMA / SARIMA models 10](#__RefHeading___Toc571_3937304071)

[Faster Alternative to statstools ARIMA / SARIMA 11](#__RefHeading___Toc573_3937304071)

[Why the model is chosen ? 11](#__RefHeading___Toc575_3937304071)

[Configuration for Unobserved components model 11](#__RefHeading___Toc577_3937304071)

[Checking Model Performance from Result Statistics 12](#__RefHeading___Toc579_3937304071)

[Feature Engineering 13](#__RefHeading___Toc581_3937304071)

[Software / Packages Used: 13](#__RefHeading___Toc593_3937304071)

[Monthly-Hourly Average Energy 13](#__RefHeading___Toc583_3937304071)

[Quarterly-Hourly Average Energy 15](#__RefHeading___Toc585_3937304071)

[Week of Year-Hourly Average Energy 15](#__RefHeading___Toc587_3937304071)

[Week Day-Hourly Average Energy 17](#__RefHeading___Toc589_3937304071)

[Week Day Group-Hourly Average Energy 19](#__RefHeading___Toc591_3937304071)

[Code that generate the Features 20](#__RefHeading___Toc595_3937304071)

# Model(s)’ Performance

Various models used in this competition and its performance on Leaderboard and validation set are as below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Private LB** | **Public LB** | **Validation Score** |
| Final (UC Model with aggregated mean features) | 436.4 | 264.9 | 210.1 |
| UC Model with OHE Hour Features | 386.6 | 294.3 | 259.21 |
| UC Model with Holiday Features | 389.3 | 304.5 | 259.17 |
| UC Model without exogenous Features | 404.5 | 302.1 | 279.2 |
| Baseline Model | 617.9 | 941.4 | 431.7 |

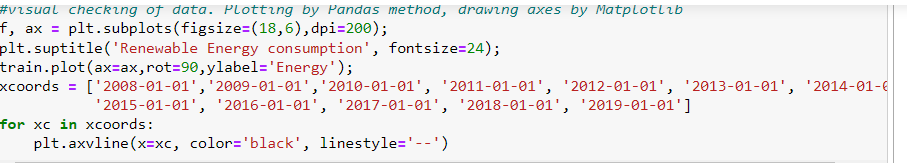
# **Exploratory Data Analysis**

## **Libraries Used**

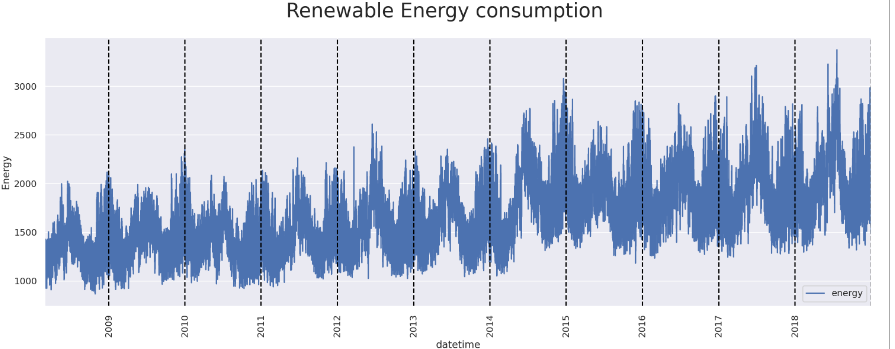
* statsmodels.tsa.seasonal\_decompose
* statsmodels.graphics.tsaplots.plot\_acf, statsmodels.graphics.tsaplots.plot\_pacf
* pandas,numpy,matplotlib,datetime

## Visualizing Data

Code to visualize



Let us visualize the data where each vertical dashed line corresponds to the start of the year.



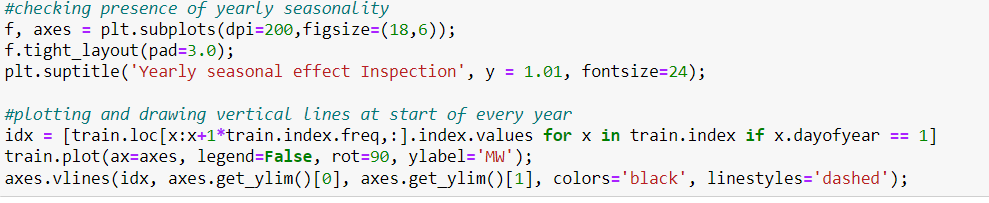
The data seem to have a seasonal variation. Also, the energy seem to increase, but not significantly over time since the mean energy has almost the same value regardless of the year.

Also, within each point (I.e) each day, energy has major variation and probably could be due to different hour and this pattern could be understood when we dig further at hour level analysis subsequently.

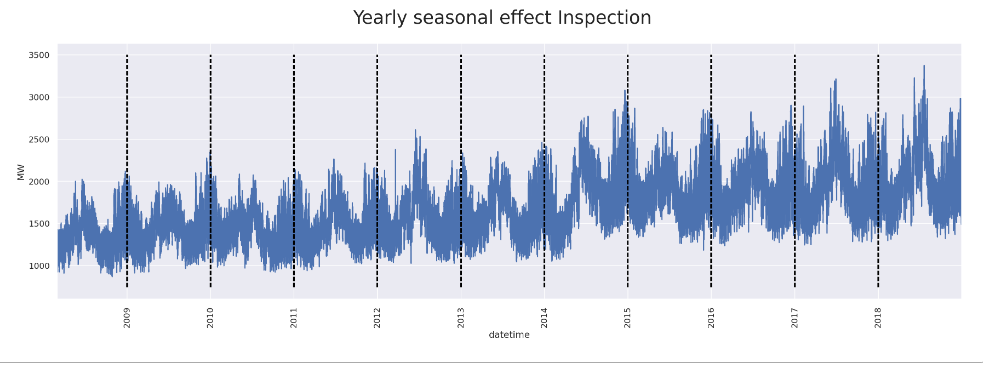
## Check for Seasonality

### Check for Yearly seasonality

code to check yearly seasonality

****

Let us visualize the data where each vertical dashed line corresponds to the start of the year.

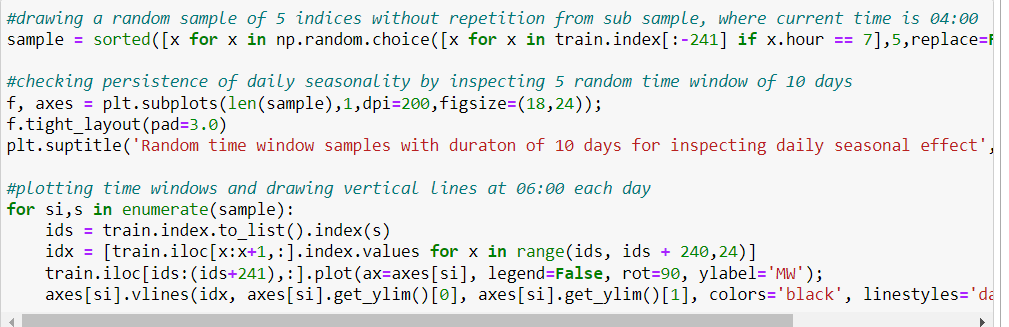
****

From the above chart we can clearly see that pattern repeats every year and hence yearly seasonal effect is present.

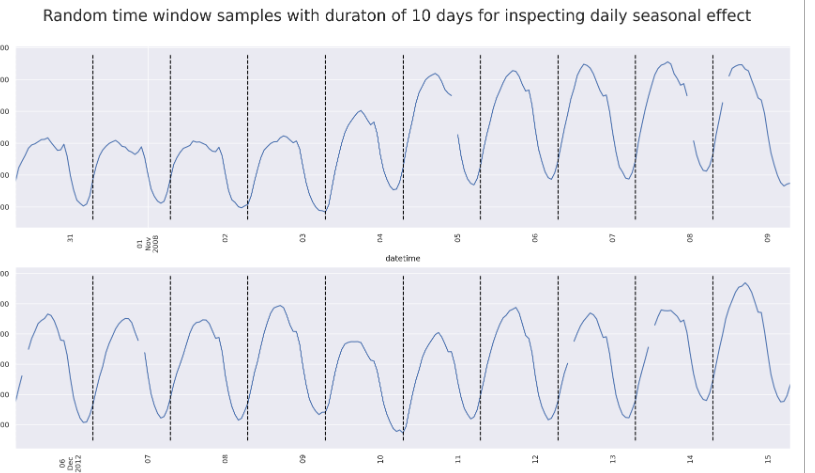
### Check for 24 hours seasonality

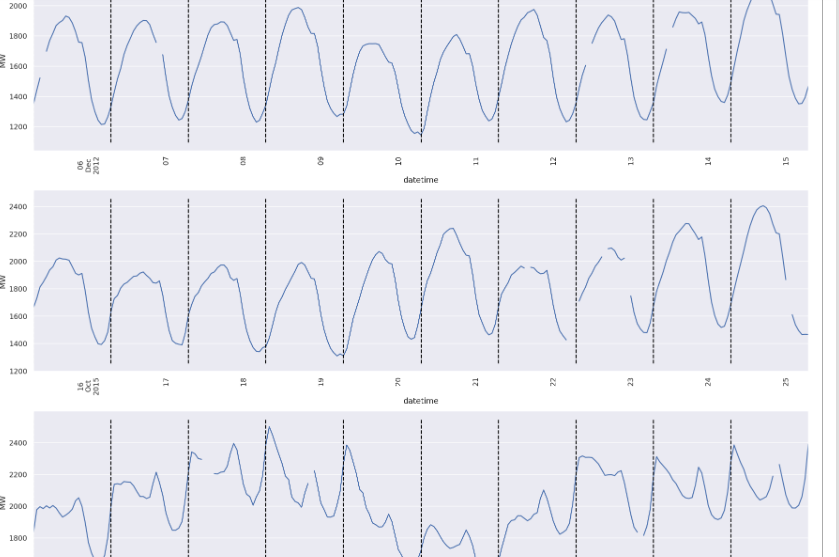
For visualizing 24 hours data, let us select five random time windows and plot at different scales for checking any seasonal patterns:

code for visualizing 24 hours data

****

Let us visualize the data where each vertical dashed line corresponds to 7:00 AM every day

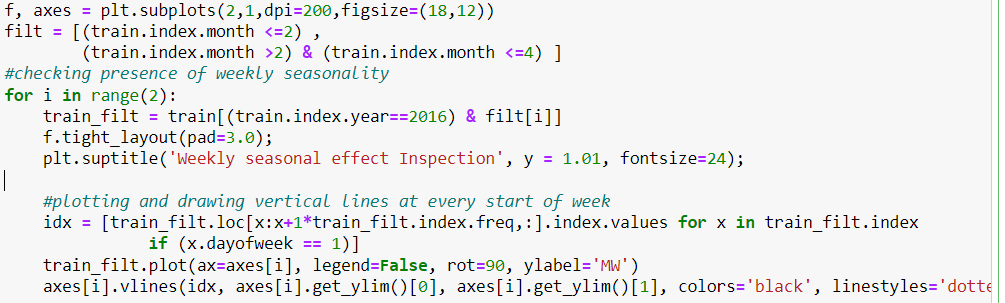




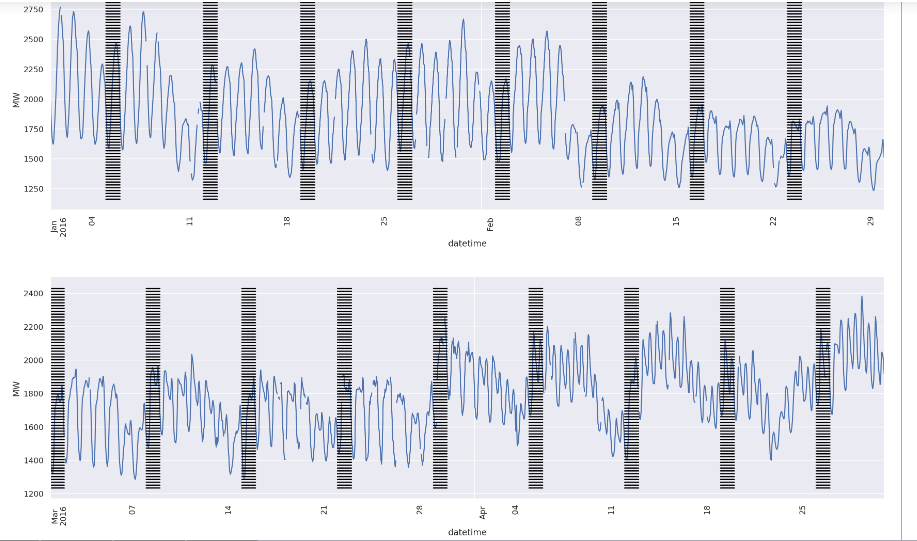
From the above chart we can clearly see that pattern repeats every day for most of the cases (i.e) every 24 hours and hence daily seasonal effect is present

### Check for Weekly seasonality

To check weekly seasonality, let us consider 4 different months from one of the year (Jan -Feb 2016 and Mar-April 2016) and plot.

****

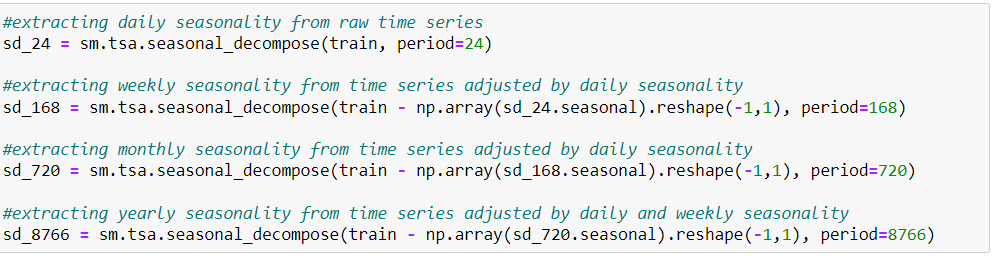
Let us visualize the data where each vertical dashed line always corresponds to fixed day of the week



From the above chart we can see that seasonality pattern exists for week where last 2 days (weekends) have lower energy consumption compared to other days.

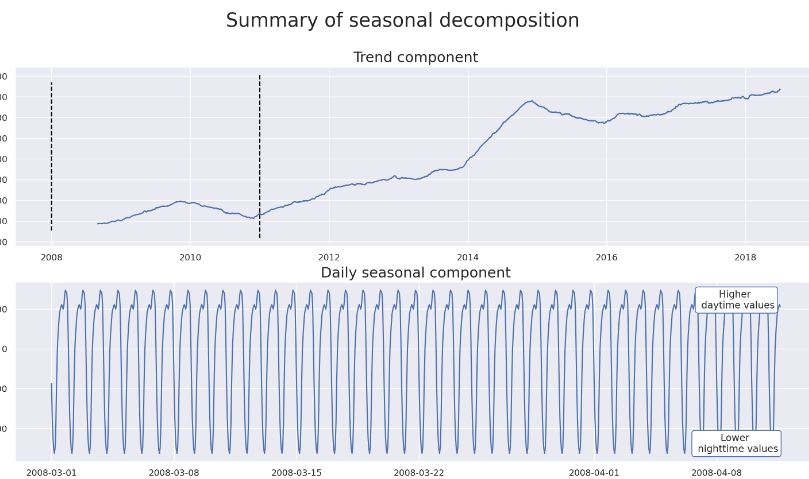
## Seasonal Decomposition

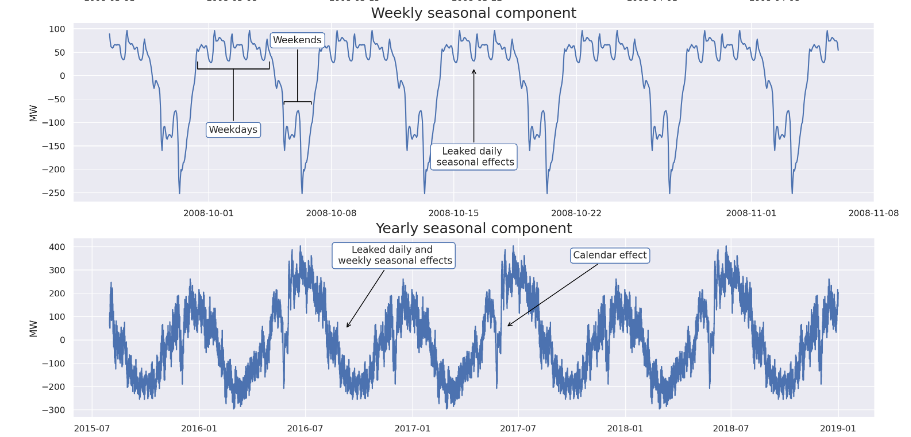
Using seasonal decomposition (from statsmodel api), one can get the seasonal and trend components of the energy data. Using this method, we can also confirm all above seasonality pattterns.

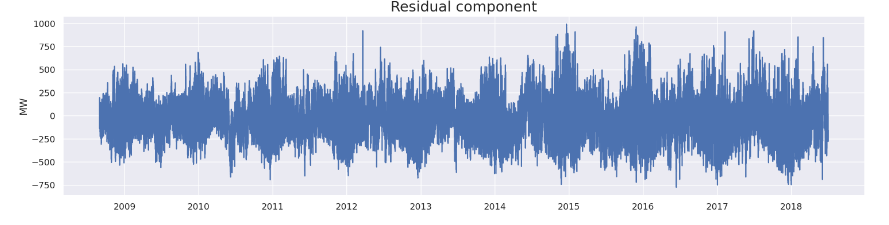




Let us plot trend component and seasonal decomposition component for different subset of data for different seasonality.







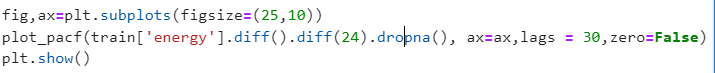
From the above charts the following are the **inferences**:

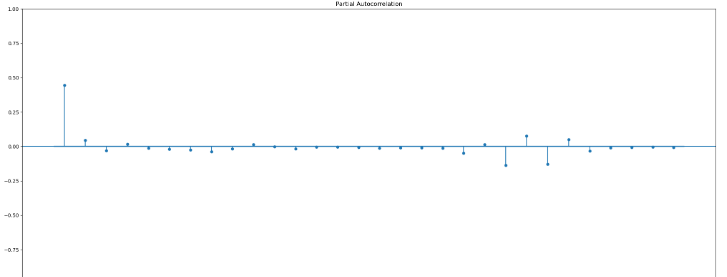
* Seasonality pattern confirmed for
  + **Daily** (24 hour)
  + **Weekly**
  + **Yearly**
* Trend is **linear**. Trend could be either stochastic or deterministic. In this case, the slope of the trend is almost same except during 2015 where there is sudden increase, but in later years, it rolled back to the earlier slope. Hence it would be better to treat the data using **deterministic** trend.

## Determination of order of Auto Regression and Moving Average

ACF and PACF plots are used to determine order of AR and MA.

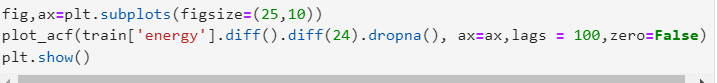
PACF Plot

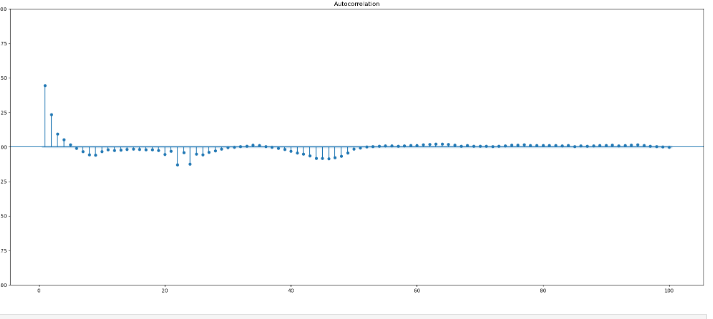
****

****

PACF plot, there are only 2 significant lags and this plot determines AR component of ARIMA model.So AR order (p) could be atmost 2.

ACF Plot





Here ACF plot tails off while PACF plot cutsoff at certain point.

**So the MA component would be zero (i.e q=0) and AR component would be 2 (p=2).**

# Model Development

## Model Training and Evaluation

This process would be used to decide which model and features would perform optimal forecasting of the given timeseries data.

To have validation set similar to forecast horizon (ie test data has 3 years of data), train test split is performed as following:

* Validation Set: 3 years period with data from 01-Jan-2016 to 31-Dec-2018
* Train Set: Remaining 8 years with data from 01-Mar-2008 to 31-Dec-2015

## Model Finalization

In this process, the model with entire train dataset is fitted and perform predictions on test dataset using this model.

## Baseline Performance

* Before choosing model, a simple baseline prediction has been carried out and then the model benchmark would be compared against this baseline
  + All values in validation set predictions are simply set as single fixed value using mean of the train set values.
  + Validation Baseline RMSE: 431.73

Hence machine learning models need to perform much better on validation set compared to this baseline prediction.

## Why specific model is chosen ?

### Libraries Used

* sm.tsa.UnobservedComponents
* pmdarima
* RAPIDS AI cudf, cuml.tsa.arima

### Constraints with direct ARIMA / SARIMA models

SARIMA model can provide state of the art solutions to time series forecasting. But it has two major drawbacks:

(1) can model only a single seasonal effect,

(2) season length cannot be too long.

But SARIMA models can still be used for multiple seasonality indirectly using manually computed Fourier transformed exogenous features. But even if it has been done, the other constraint with these models is that the computation time of the corresponding statstools API for the given data is long.

### Faster Alternative to statstools ARIMA / SARIMA

So another alternative could be to use GPU enabled API from RAPIDS AI ARIMA package. It performs much faster than statstools ARIMA packages, but still it consumed in terms of hours when high seasonality such as 24 hours or more (for yearly seasonality) is used.

And of course, the performance of ARIMA model with single seasonality (just 24 hours) also poor (RMSE 422.02 just closer to baseline performance)

Model Requirement for given data

So we required a reliable model which support multiple seasonalities and at the sametime also execute faster. So the model that satisfies these constraints is Unobserved Components package.

### Why the model is chosen ?

**Unobserved Components** package supports both requirements of supporting multiple seasonalities at relatively much faster execution time

Brief Intro on Unobserved Components Model

|  |
| --- |
| *Unobserved Components Model (UCM) performs a time series decomposition into components such as trend, seasonal, cycle, and the regression effects due to predictor series.* |

The execution time comparison of ARIMA packages and the Unobserved Components package is given below.

|  |  |
| --- | --- |
| **Model** | **Execution Time** |
| RAPIDS ARIMA | 123 min (with GPU) |
| Unobserved Component | 3 min |

### **Configuration for Unobserved components model**

* Seasonality period are 24 hours, 168 hours (weekly) and 8766 hours (yearly) respectively
* **Trend** has been set as **Local Linear Deterministic Trend** which is captured using level parameter value as *lldtrend*.

This has been chosen because the trend is deterministic based upon the inference from Exploratory Analysis.

Also ”Local Linear” implies that we would like to capture the ARMA process in prediction so that the model would perform better.

* ARMA regression value of “**2**” is selected from one of the best value from PACF and ACF plots during Exploratory Analysis

Code to fit Unobserved components model for given time-series data



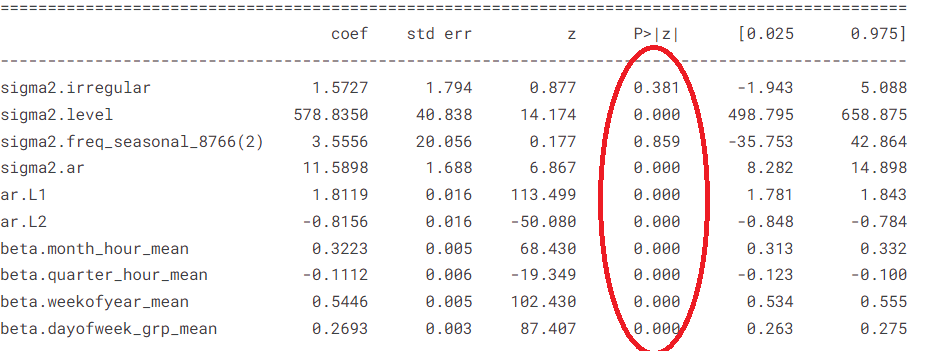
### Checking Model Performance from Result Statistics

Result Summary can be checked using below code after performing model fit.



The results summary of the Unobserved Components model on validation set is as below.

From the below summary, it can be seen that the p-value for all predictors are below 0.05 and hence null hypothesis of coefficient=0 can be rejected. So it can be clearly confirmed that all predictors are very good predictors (I.e) these predictors have strong relation with the target value according to this model.



|  |
| --- |
| RMSE of the final model on validation set: 210.1 |

This model performed much better than baseline model performance of 431.33

# Feature Engineering

## Software / Packages Used:

Tableau

Though the time series model is capable of performing independent forecasting without any features, adding exogenous features would enforce the model to perform a dynamic regression and thereby improve the model performance.

The exploration of mean energy for each hour is done for each of the calendar group for month-wise, quarter-wise, week of year wise and day of week wise.

**Tableau** Visualization tool is used for exploration since it can quickly generate charts for timestamp related aggregation features.

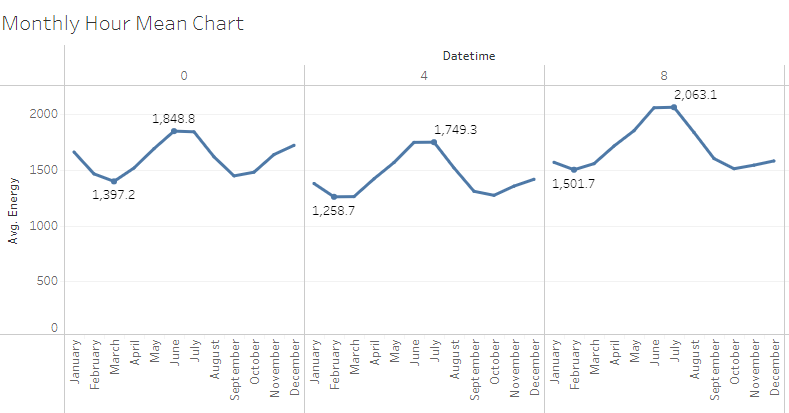
## Monthly-Hourly Average Energy

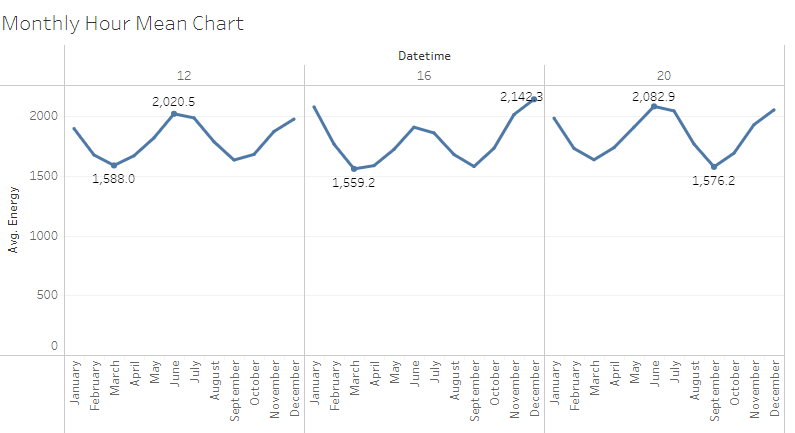
The below chart contains mean energy for month and hour combination.

Tableau selection options to generate the chart



Let us visualize the data for some of random hours for every month . Here labels corresponds to minimum and maximum energy values for the specific hour





From the above charts, though at high level it looks like similar pattern, but if one spends time to look into these charts, there are considerable differentiation of the values of the target (energy) as well as the patterns for certain hours. So this feature could be useful for forecasting.

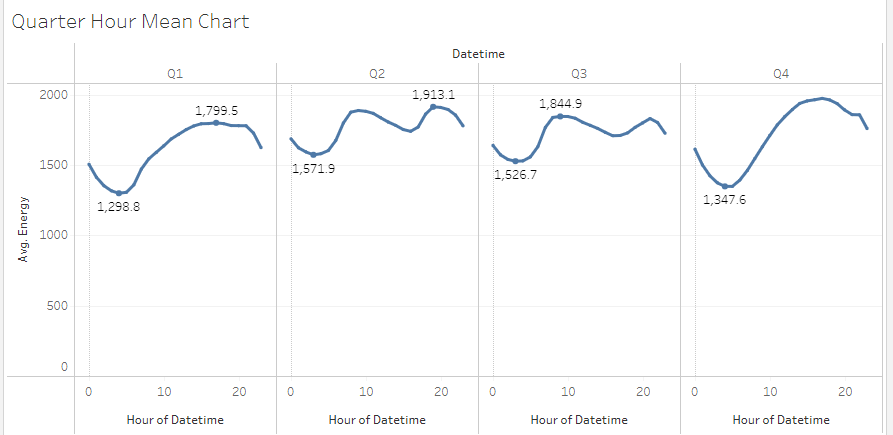
## Quarterly-Hourly Average Energy

The below chart contains mean energy for quarter and hour combination.

Tableau selection options to generate the chart



Let us visualize the data for some of random hours for every quarter. Here labels corresponds to minimum and maximum energy values for the specific quarter.

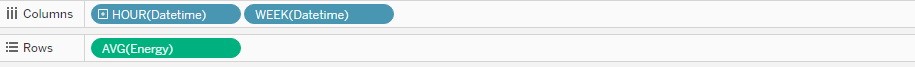


From the above charts, there is clear differentiation of the pattern between different quarters and also values of the target (energy) for different hours. So this feature could be useful for forecasting.

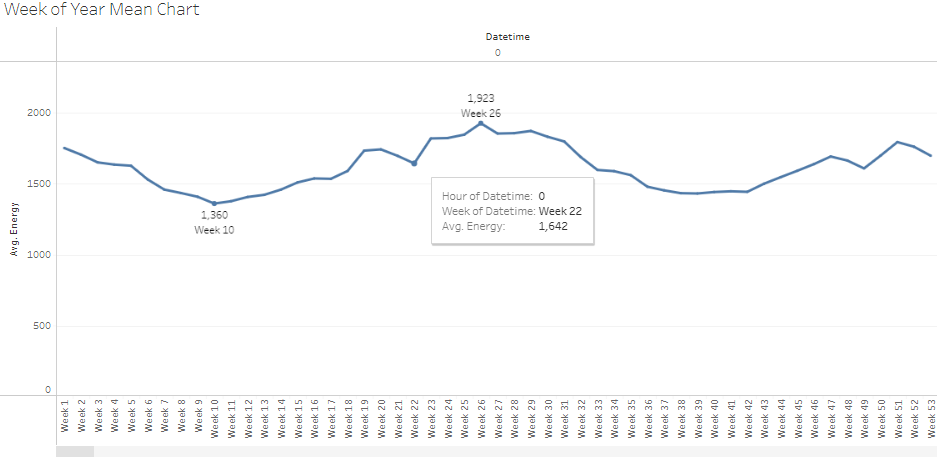
## Week of Year-Hourly Average Energy

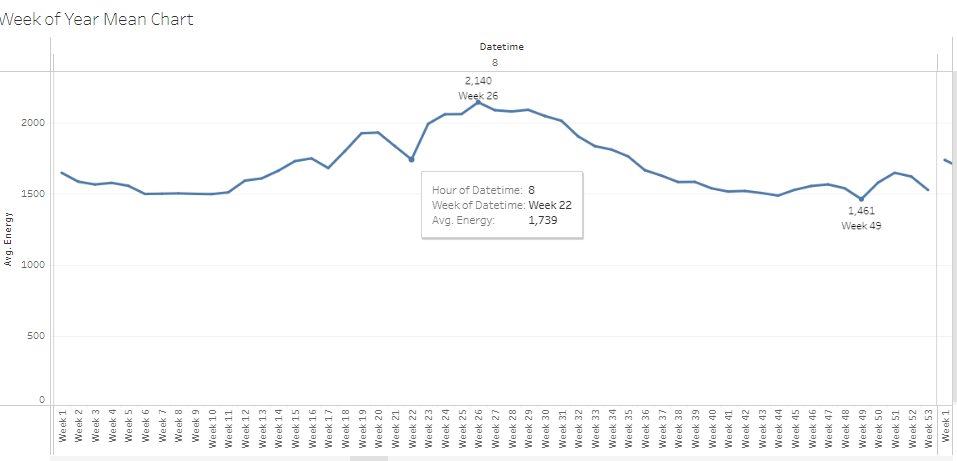
The below chart contains mean energy for week of year and hour combination.

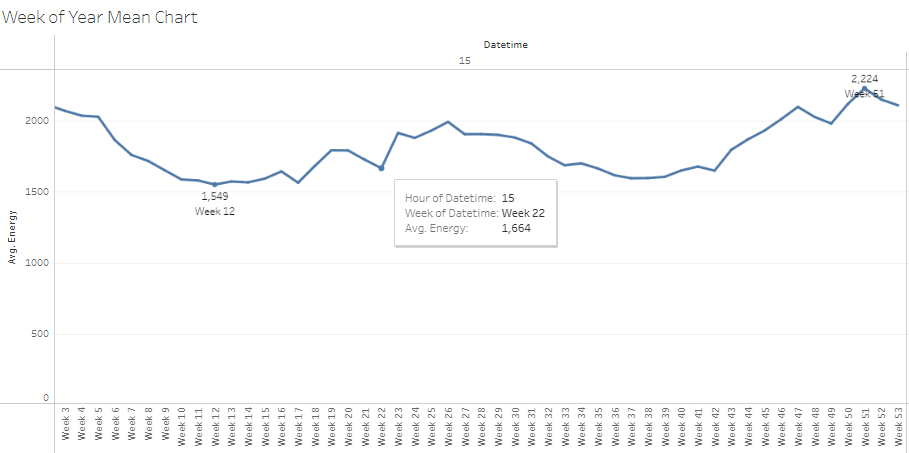
Tableau selection options to generate the chart

****

Let us visualize the data for some of random hours (0,8 and 15) for every week. Here labels corresponds to minimum and maximum energy values for the specific hour.







From the above charts, though at high level it looks like similar pattern, but if one spends time to look into these charts, there are considerable differentiation of the values of the target (energy) as well as some part of the pattern (especially at ends of the pattern). So this feature could be useful for forecasting.

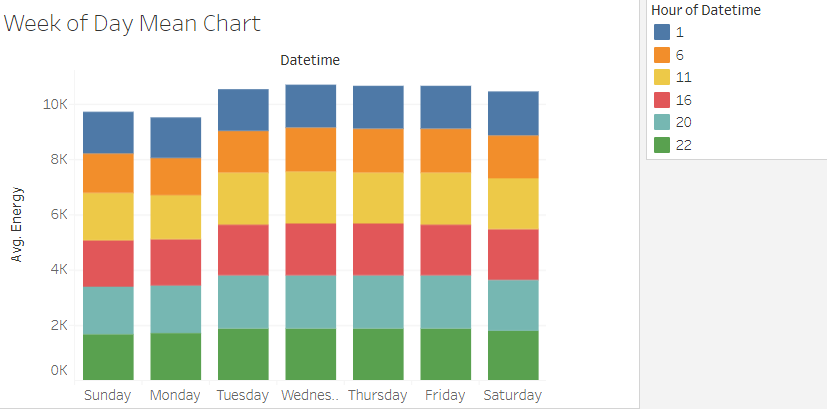
## Week Day-Hourly Average Energy

The below chart contains mean energy for week day and hour combination.

Tableau selection options to generate the chart

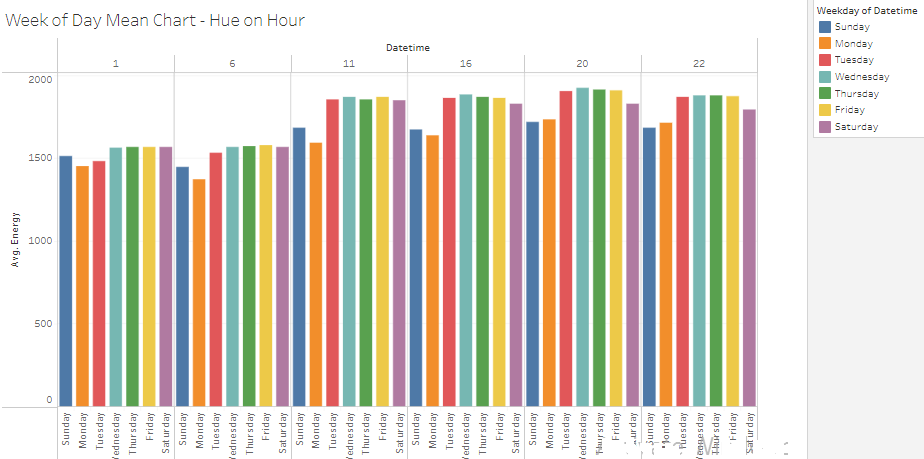


Let us visualize the data for every week day with colored hue on random hours.



Here from above chart, it can be seen that the days Sunday, Monday have lowest energy values, Saturday have next low energy values. So there is a clear differentiation of energy values among week day.

Let us also visualize the data for every random hour but with colored hue on week day.

From the above pattern, it can be confirmed that different hours exhibit different patterns when week day is used. So the week day feature could also be useful for forecasting.

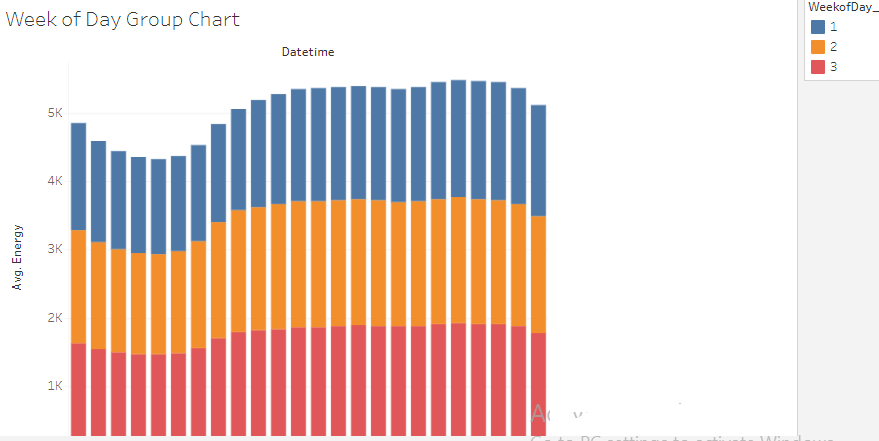
Here we could notice that other week days have almost same energy values and hence these week days may not be differentiator among themselves. To fine tune this pattern, week day can be grouped into a new feature as mentioned in next section.

## Week Day Group-Hourly Average Energy

Week day can be grouped as follows:

* Group 1: Monday and Sunday
* Group 2: Saturday
* Group 3: Remaining days

With this grouping, visualization of hourly mean chart is as below with x-axis having hour starting from 0 and color hue for week day group.



Here it can be clearly seen that lot of pattern differences for various hours can be seen. So week day group hour mean feature is preferable over just week day hour mean.

## Code that generate the Features

