Assignment 1: Report

Smart Energy for Info Age

Author: Rohit Rawat

SBU ID: 112963417

Dataset:

We have Time series data of three homes named **B, C,** and **F**. For all the homes, we have two types of datasets i.e. **Meter data** and **Weather data**. Meter dataset contains power consumption by all the electrical appliances in the house for every half an hour or/and every minute of a year. Also, Weather dataset contains feature columns about the weather-related data like temperature, humidity, summary of weather, icon, visibility, pressure, cloud Cover, etc. Weather dataset contains per hour records. All the features have numerical values except two features i.e. **"icon"** and **"summary"** which are categorical features.

Dataset Challenges:

We have encountered following challenges with given datasets:

1. Meter dataset contains index feature i.e. ***"Date & Time"*** in Timestamp format and Weather dataset contains index feature i.e. ***"time"*** in frequency format.
2. Meter dataset records start from 5 hours late as compare to Weather dataset i.e. For home B, weather dataset starts from 00:00 hours of Jan-01-2015 to 23:00 of Dec-31-2015 and but meter dataset records start from 05:00 of Jan-01-2015 to 04:00 of Jan-01-2016.
3. Mostly, all the meter datasets contain per half an hour records, but weather datasets contain per hour records.
4. For Home C, we have per minute records from Dec-16-2015 to Dec-31-2015 but rest is per half an hour record.
5. For Home F, meter dataset records are not available for last 15 days of Dec 2016, but they are present in weather dataset.
6. There are two categorical features (**"icon"** and **"summary"**) which need to be handled before making prediction models as prediction models processes numerical features only.
7. There are Non defined (NaN) values in few features of weather dataset, which are **'visibility','pressure','windSpeed','windBearing',** and **'cloudCover'**.
8. All the attributes collectively are not standardized or normalized i.e. they contains different range of values in different features.

Data Preprocessing:

We have handled above mentioned dataset challenges as follows:

1. Using ***'fromtimestamp(time)'*** of *datetime* library, We have converted frequency format values of 'time' feature of weather dataset into timestamp format like '***Date & Time***' of Meter dataset.

```
def fromtimestamp_timestamp(weather):
  dt_object = []
  for timestamp in weather.time:
    dt_object.append(datetime.fromtimestamp(timestamp))
  weather.time = dt_object
```

2. We have calculated total power consumption per record for Meter dataset.

3. For 5 hours late starting of Meter dataset records, I have inserted the last 5 hours records into the beginning assuming if first 5 hours of records will be similar for two consecutive years. So, Weather and Meter dataset starts from same time.

```
#Covering for 5 hours data
def reset_timestamp(weather):
  fyear = weather.time[0].year
  weather.time = weather.time.apply(lambda time: time.replace(year=fyear))
  weather = weather.sort_values(by = 'time')
  weather = weather.reset_index(drop=True)
  return weather
```

4. Under function ***collaborate()***, we have collaborated home and weather data by calculating per hour consumption for meter datasets and appending that value in weather datasets as a new column ***'Total_consumption'***.

5. Considering the size of dataset enough, we have dropped few records i.e. last 15 days of records of home C and home F due to challenges 4 and 5.

6. We have replaced NaN values with the mean of that feature columns.

```
#Replacing NaN values
col_mean = ['visibility','pressure','windSpeed','windBearing','cloudCover']
for col in col_mean:
  if col in weather.columns:
    weather[col] = weather[col].fillna(weather[col].mean())
```

7. Label encoded the two categorical feature columns i.e. ***'icon'*** and ***'summary'***.

```
#label Encoding
col_cat = ['icon','summary']
for i in col_cat:
  lb=le()
  lb.fit(weather[i])
  temp=lb.transform(weather[i])
  weather[i]=temp
```

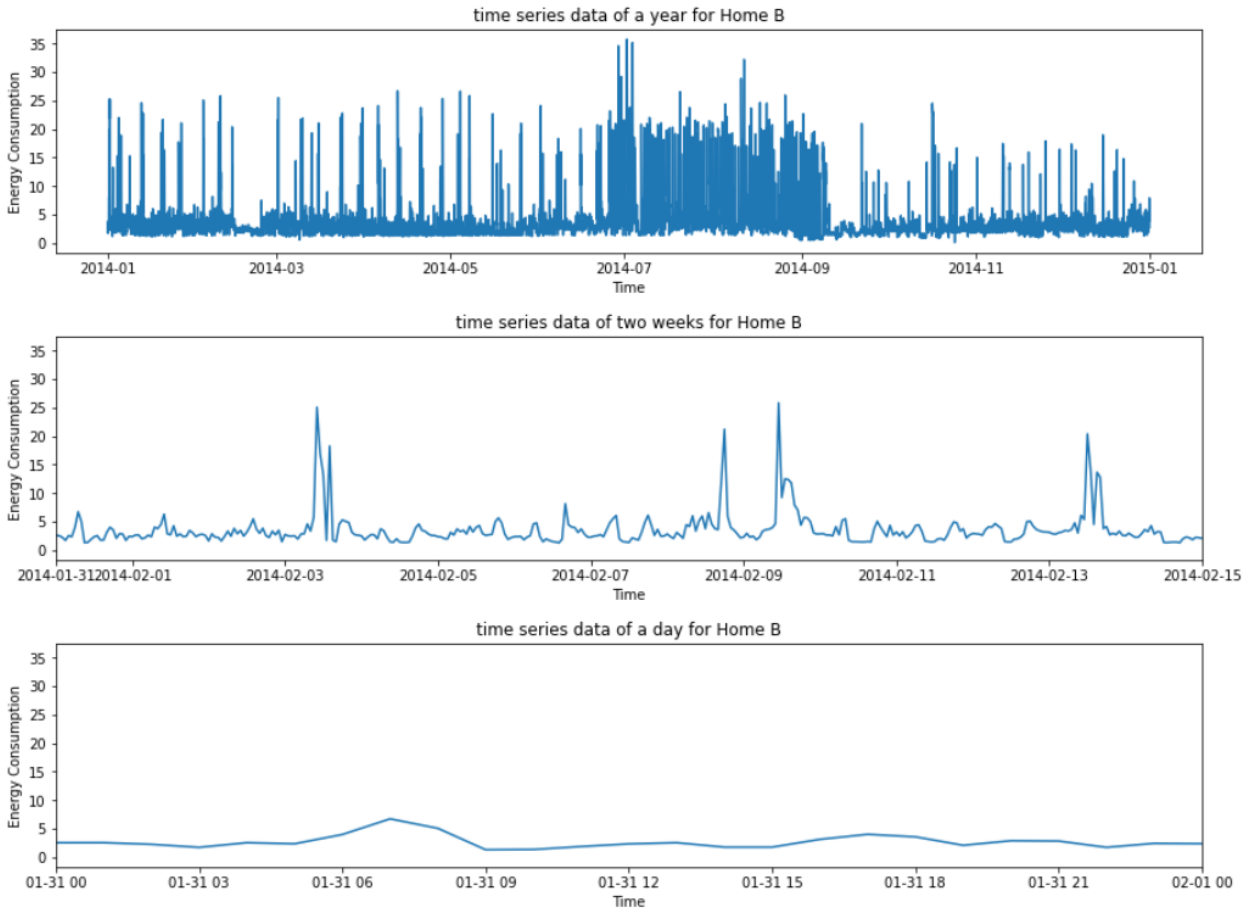
8. Under function Min_max_scaling(), datasets are Scaled in range of (0,1) using ***MinMaxScaler()*** function of ***sklearn.preprocessing*** library.

```
def Min_max_scaling(home):
  scaler = MinMaxScaler()
  scaler.fit(home)
  home = scaler.transform(home)
  return home
```

9. We have to predict values of next one hour and next whole day power consumption. So, I have shifted '***Total_consumption'*** by one row and appended it as true value of next hour power consumption i.e. ***'Next_hour'*** column. Similarly, for next day power consumption I have shifted it by 24 records and appended as ***'Next_day'*** column.
10. Wrote a function ***Calc_MAE()*** to calculate Mean Absolute Error between predicted and true values to compare the models' efficiencies.
11. Wrote a function ***plot_performance()*** to depict the true and predicted values in a line graph for any 15 days.

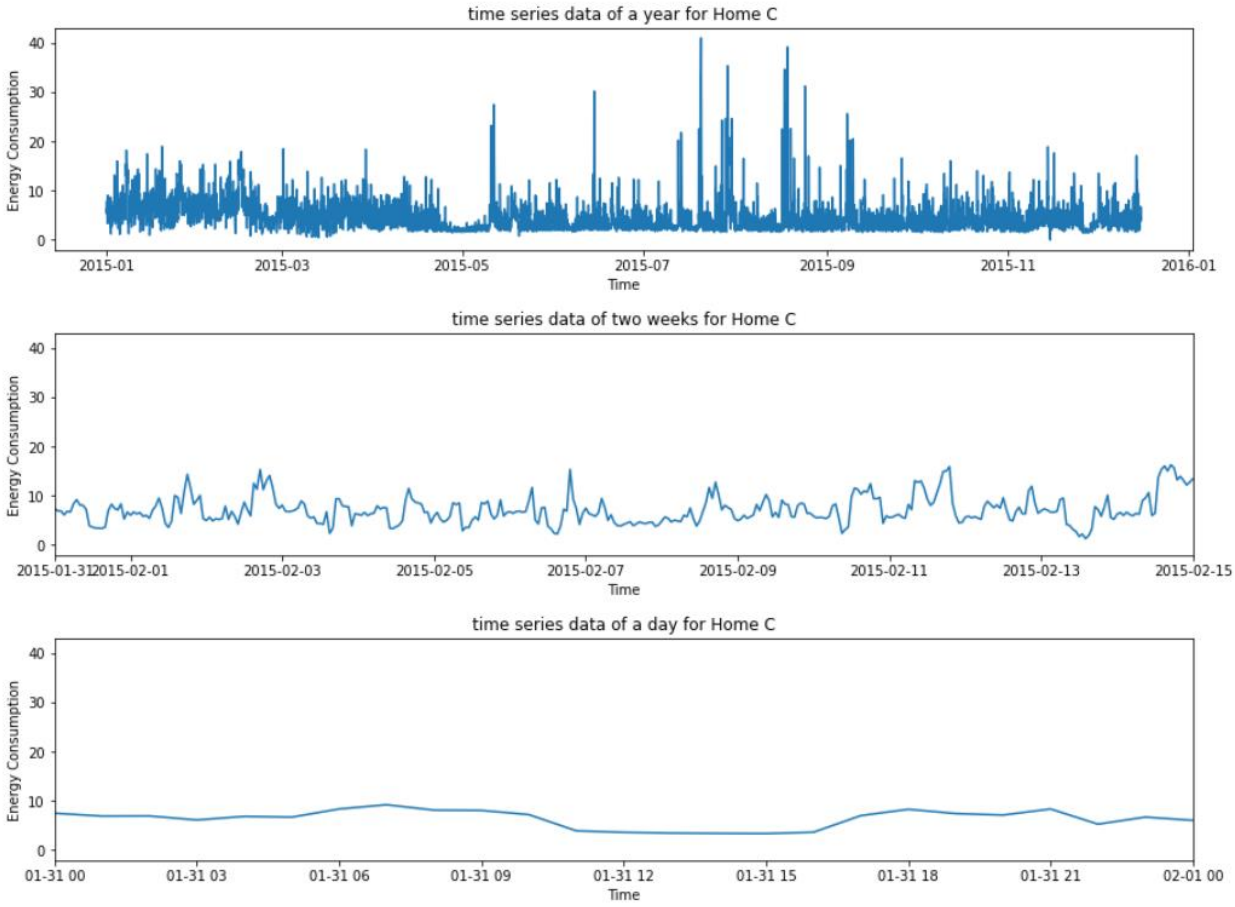
Data Visualization and Analysis:

- **Time Series plot for Home B:**



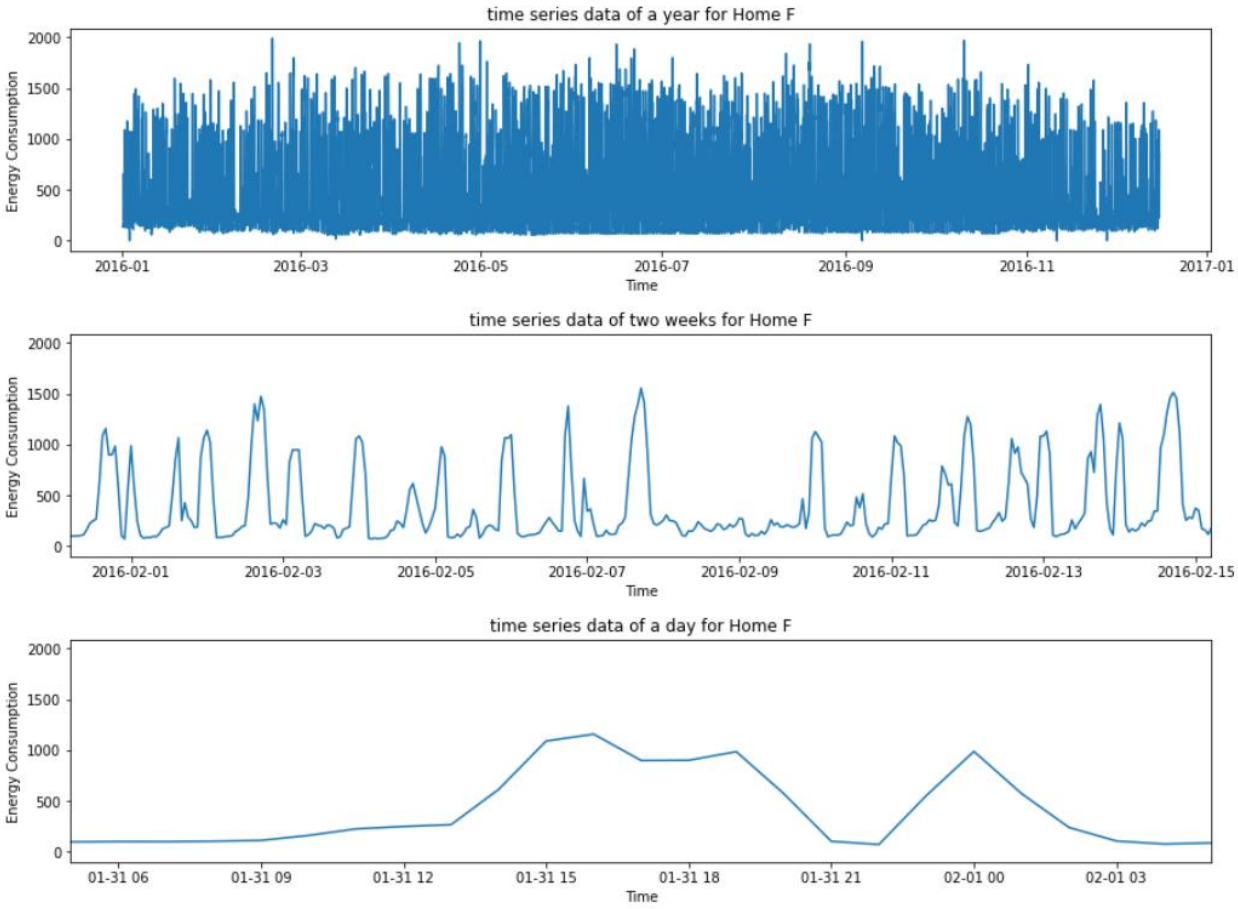
  - From year graph, we can observe that power consumption becomes dense from 2014-07 to 2014-09 which is the duration of summer holiday season. Also, for first 6 months power consumption is more than the last 6 months because heater appliances are in used in winter season and cooling appliances in summer.
  - From two weeks graph, we can observe, in almost every 7 days power consumption is peaked up probably because of the weekends.
  - From day graph, we can observe between 06:00 to 09:00 power consumption is maximum which is probably because of the breakfast cooking time and other morning activities.

- **Time Series plot for Home C:**



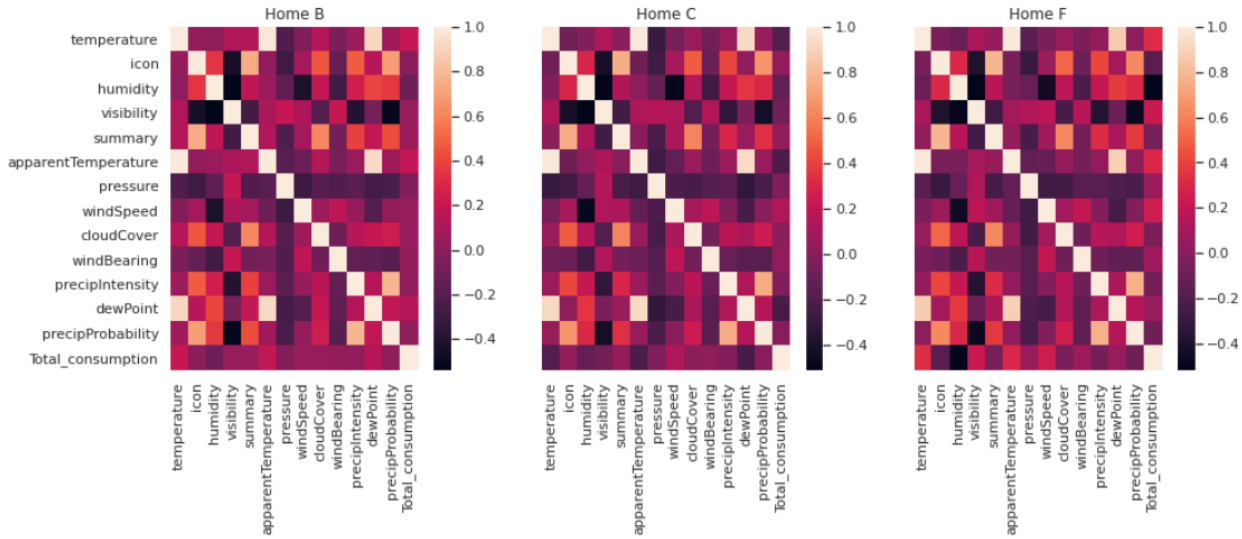
  - From year graph, we can observe that power consumption becomes peaked from 2014-08 to 2014-09 which is the duration of summer holiday season. Also, as like home B, for first 6 months power consumption is more than the last 6 months because heater appliances are in used in winter season and cooling appliances in summer.
  - From two weeks graph, we can observe, unlike home B power consumption doesn't varying much in a week.
  - From day graph, we can observe between 06:00 to 10:00 and 16:00 to 21:30 power consumption is high which is probably because of the breakfast and dinner cooking time respectively.

- **Time Series plot for Home F:**



  - From year graph, we can observe that power consumption through put the year is consistent and dense. And unlike home B and home C, power consumption is very large.
  - From month graph, we can observe consistency every day except around 2016-02-09. Also, usage is consistently high in fixed hours of a day.
  - From day graph, we can observe peaked value from 12:30 to 21:00 and at mid night. Those peaked values are probably the cyclic on these hours as we have seen in month graph.

- **Features Correaltion matrix with Heatmap:**



Home B:

- Total_Consumption is comparatively strongly correlated with temperature and apparentTemperature than other attributes.
- Temperature is strongly correlated with dewpoint. And visibility is strongly inversely correlated with icon, humidity.

Home C:

- Unlike Home B, Total_Consumption is moderately but inversely correlated with temperature and apparentTemperature.
- Like Home B, Temperature is strongly correlated with dewpoint. And visibility is strongly inversely correlated with icon, humidity.

Home F:

- Like other homes, Total_Consumption is correlated with temperature and apparentTemperature and as well as with windspeed. Also, Total_consumption is strongly but inversely correlated to humidity.
- Like other homes, Temperature is showing good correlation with dewpoint. And visibility is strongly inversely correlated with icon, humidity.

Prediction Models:

- **Model 0: Naive Model (Persistence Algorithm)**
  This is the Naïve Model which is one hour and 24 hours record shifting algorithm which shifts **"Total_consumption"** values.

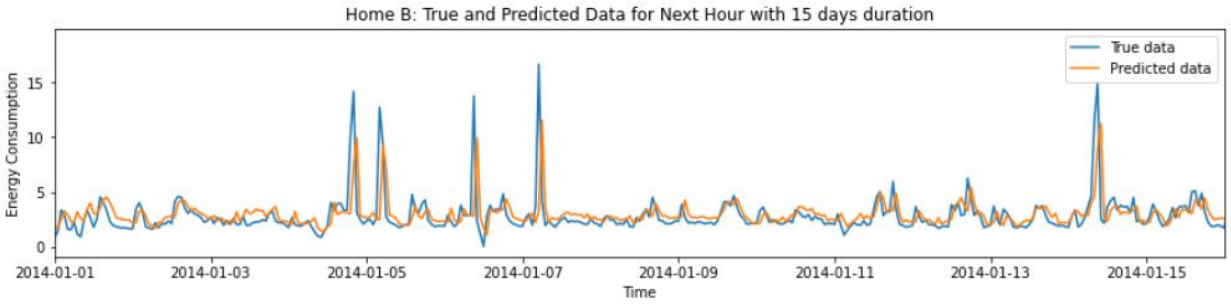
  Results for Home B:
  Mean Absolute Error for Naive Model for next hour= .948038331438215
  Mean Absolute Error for Naive Model for next Day= 1.4336747638306637

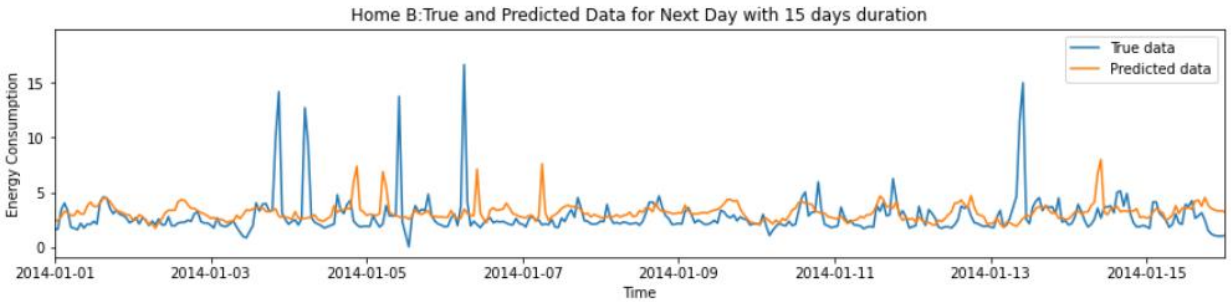
  Results for Home C:
  Mean Absolute Error for Naive Model for next hour=1.2184935316642729
  Mean Absolute Error for Naive Model for next Day= 1.8159641086379412

  Results for Home F:
  Mean Absolute Error for Naive Model for next hour= 171.5334679809642
  Mean Absolute Error for Naive Model for next Day= 276.514154977129

- **Model 1: Linear Regression Model**
  Used ***LinearRegression()*** of sklearn.linear_model library.**Results for Home B**:

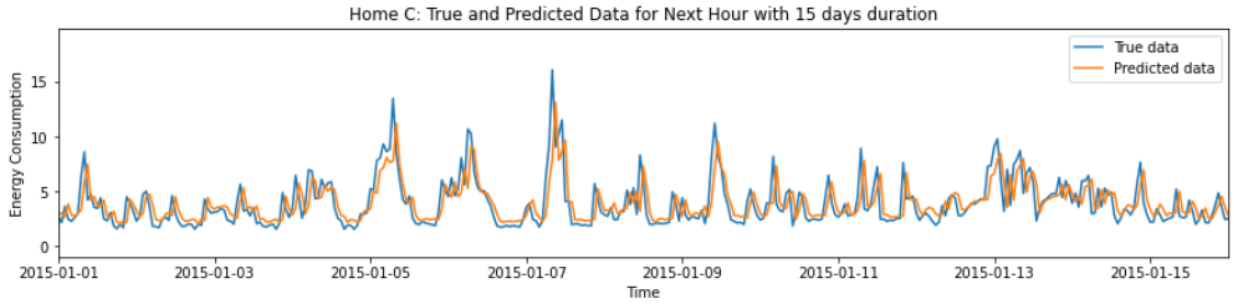
```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Linear Regression for next hour =  0.9680266850349996
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Linear Regression for next day =  1.2209428742678372
```

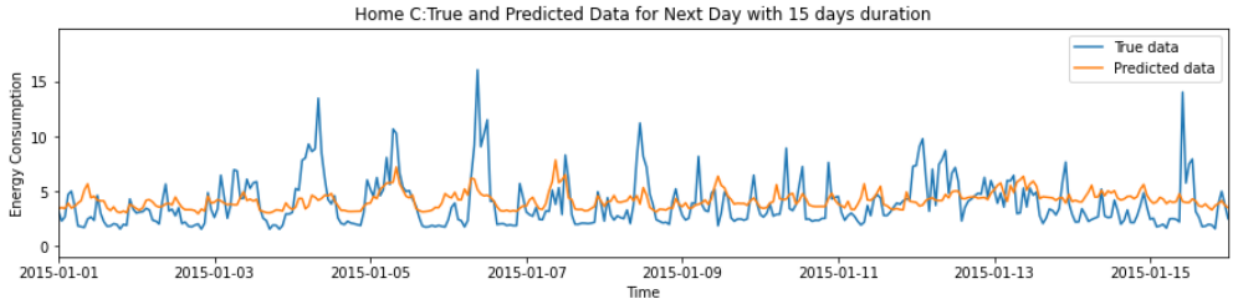




Results for Home C:

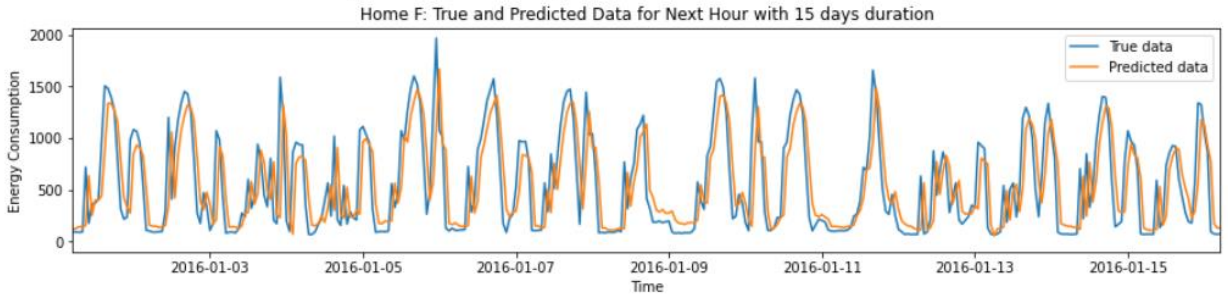
```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Linear Regression for next hour =  1.159536898139981
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Linear Regression for next day =  1.5823510223243737
```

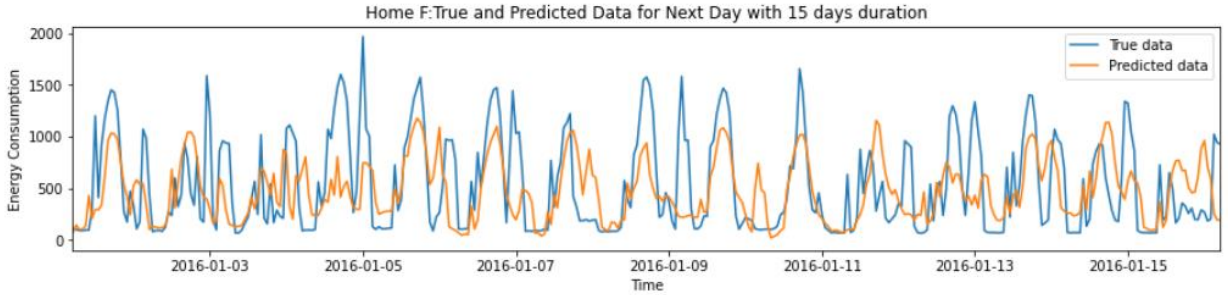




Results for Home F:

```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Linear Regression for next hour =  179.386910233394
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Linear Regression for next day =  272.2058246234367
```



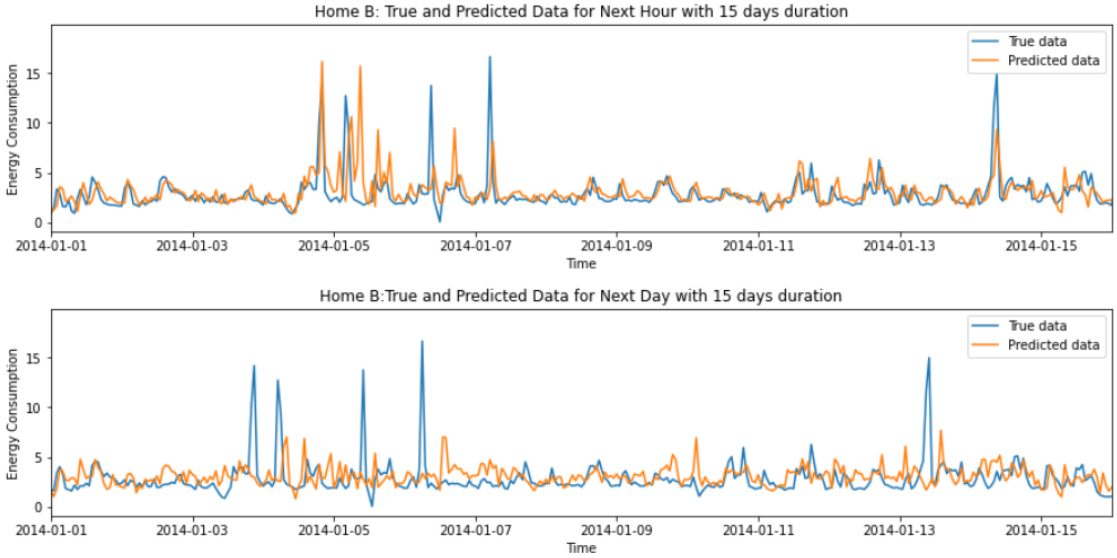


- **Model 2: XGBRegressor Model**

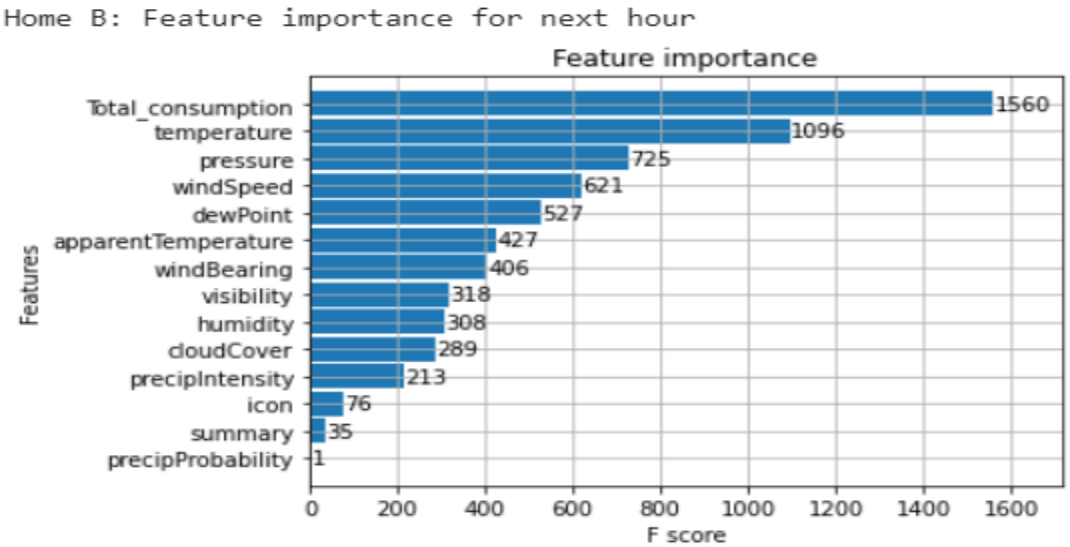
  Used ***XGBRegressor()*** of xgboost library as prediction model.

  Results for Home B:

```
Spliting Dataset in training = 80% and testing = 20% ratio.
[16:29:07] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
Mean Absolute Error for XGBRegressor model for next hour =  1.0341071344754775
Spliting Dataset in training = 80% and testing = 20% ratio.
[16:29:12] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
Mean Absolute Error for XGBRegressor model for next day =  1.2917336991199668
```

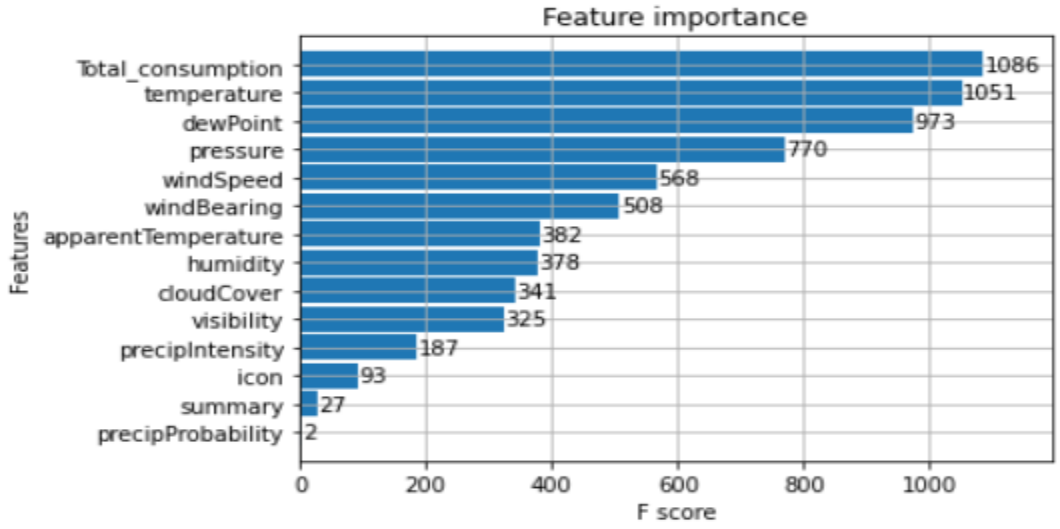


  Plot for Feature Importance for Home B:



  - So, we can see current hour ***Total_consumption*** attribute affects mostly in predicting the next hour power consumption for home B. After that ***temperature*** feature is second most important and ***precipProbability*** is least important.

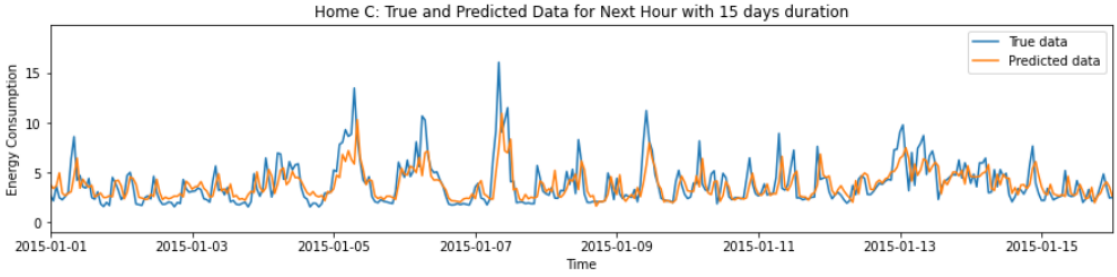
```
Home B: Feature importance for next day
```

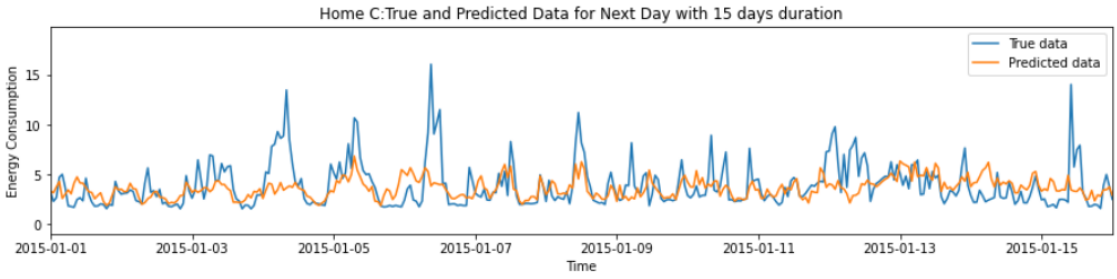


- Similarly, we can see current hour ***Total_consumption*** attribute affects mostly in predicting the Next Day power consumption for home B. After that ***temperature*** feature is second most important and ***precipProbability*** is least important.

Results for Home C:

```
Spliting Dataset in training = 80% and testing = 20% ratio.
[16:29:18] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
Mean Absolute Error for XGBRegressor model for next hour =  1.2355908529835187
Spliting Dataset in training = 80% and testing = 20% ratio.
[16:29:23] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
Mean Absolute Error for XGBRegressor model for next day =  1.7851372667727115
```

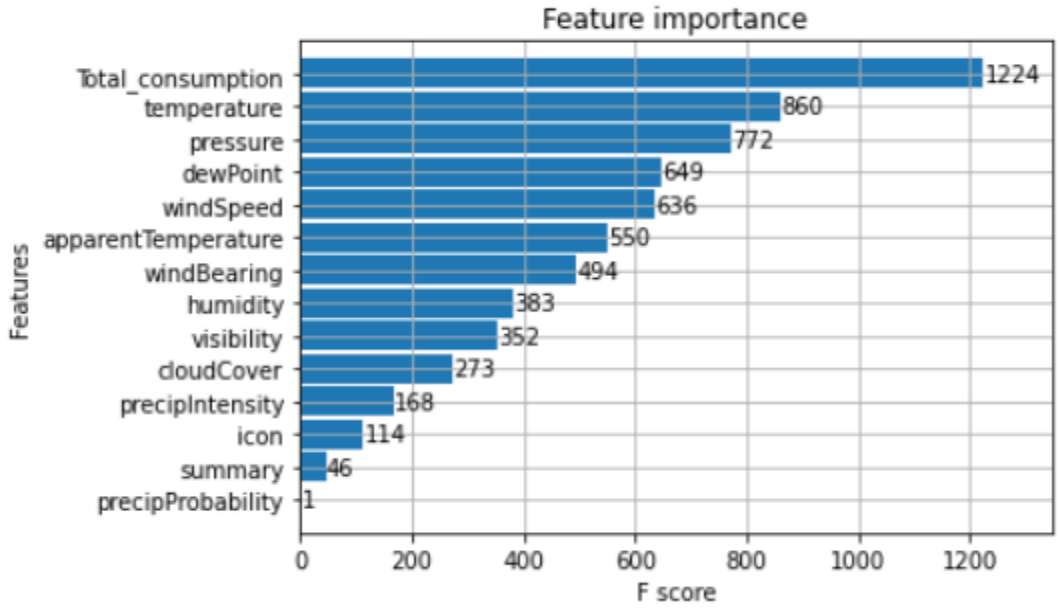




Plot for Feature Importance for Home C:

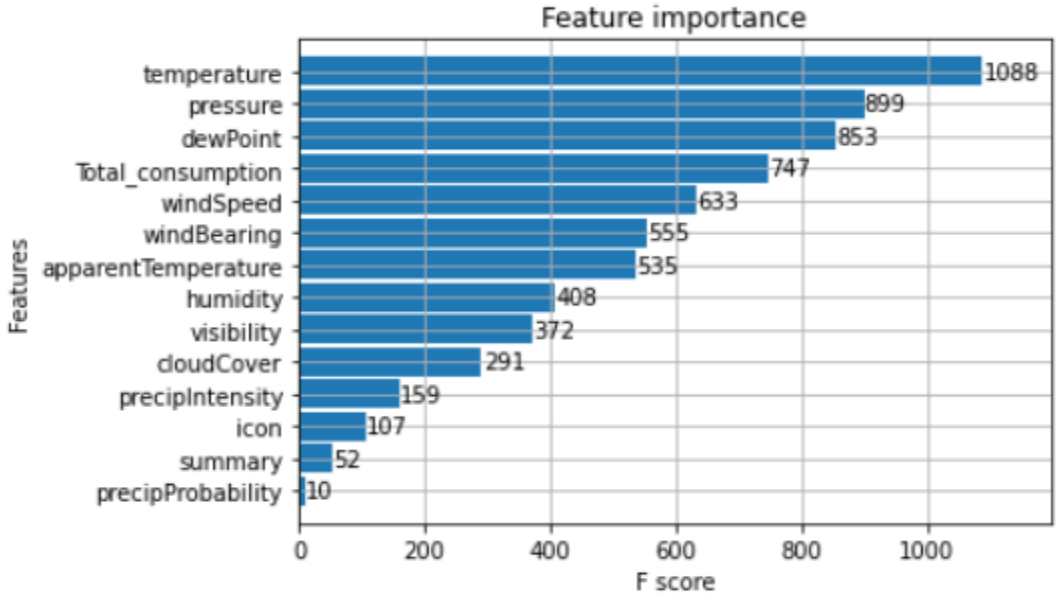
- So, we can see current hour ***Total_consumption*** attribute affects mostly in predicting the next hour power consumption for home C. After that ***temperature*** feature is second most important and ***precipProbability*** is least important.

Home C: Feature importance for next hour



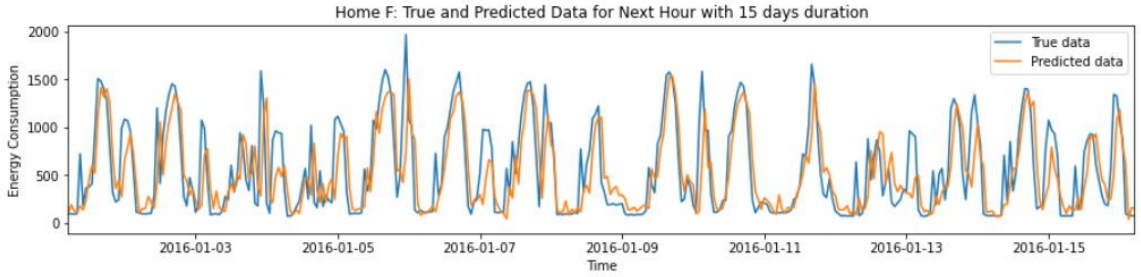
- But we can see current hour ***temperature*** attribute affects mostly in predicting the Next Day power consumption for home C. After that ***pressure*** feature is second most important and ***precipProbability*** is least important.

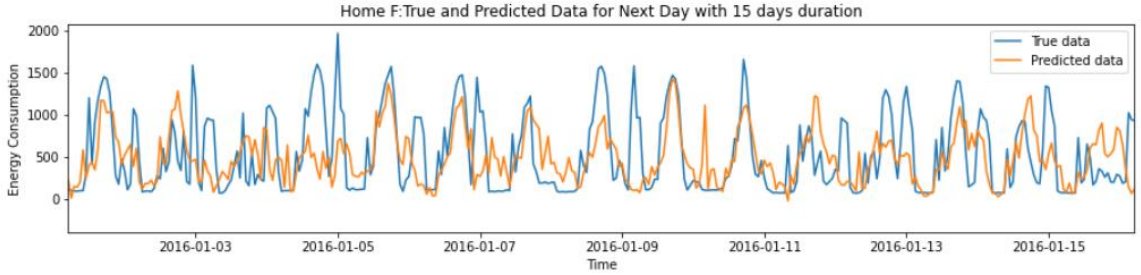
Home C: Feature importance for next day



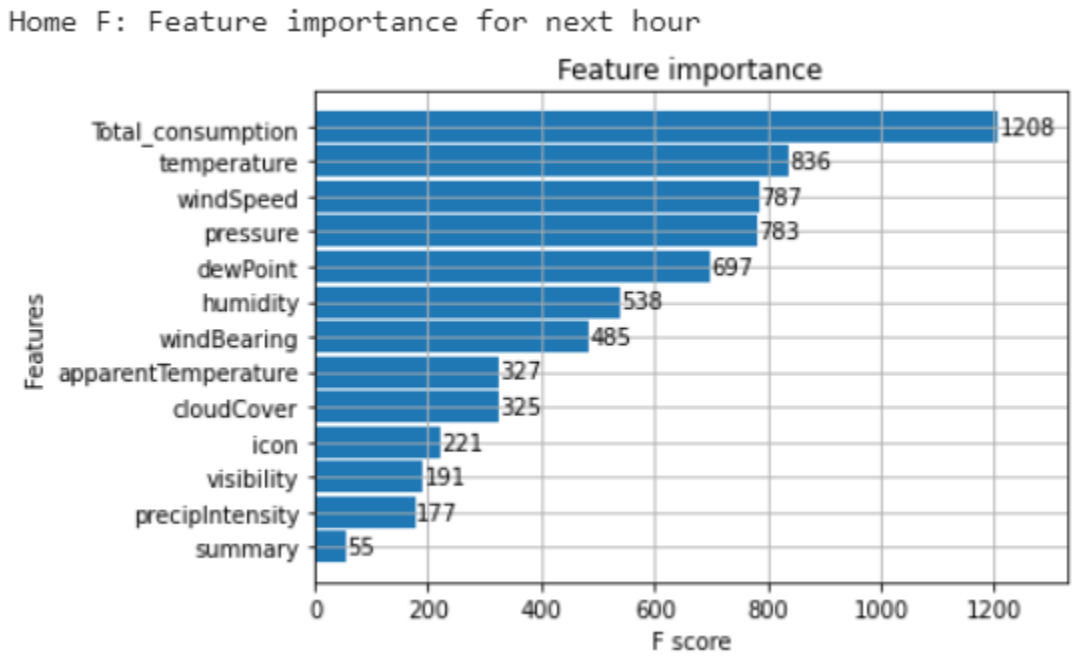
Results for Home F:

```
Spliting Dataset in training = 80% and testing = 20% ratio.
[16:29:29] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
Mean Absolute Error for XGBRegressor model for next hour =  184.51261092195384
Spliting Dataset in training = 80% and testing = 20% ratio.
[16:29:35] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
Mean Absolute Error for XGBRegressor model for next day =  270.6780752994049
```



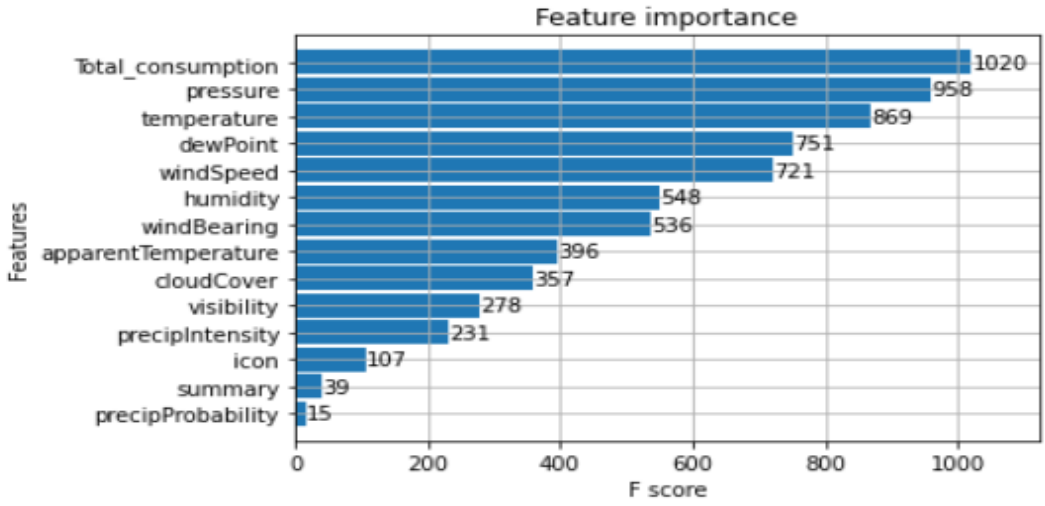


Plot for Feature Importance for Home F:



- So, we can see current hour ***Total_consumption*** attribute affects mostly in predicting the next hour power consumption for home F. After that ***temperature*** feature is second most important and ***summary*** is least important.

Home F: Feature importance for next day

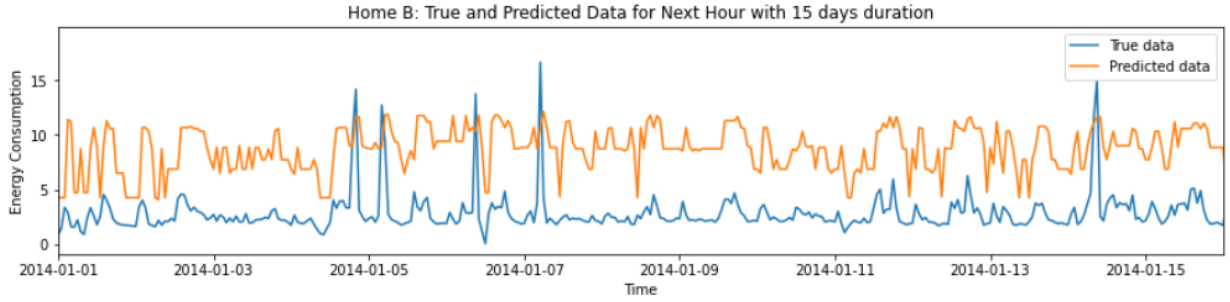


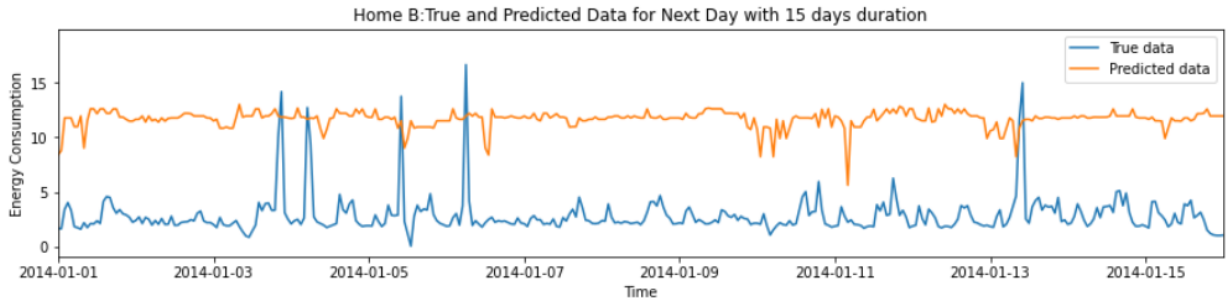
- Similarly, we can see current hour ***Total_consumption*** attribute affects mostly in predicting the Next Day power consumption for home F. After that ***pressure*** feature is second most important and ***precipProbability*** is least important.

- **Model 3: AdaBoost Regressor Model**
  Used ***AdaBoostRegressor()*** of sklearn.ensemble library as prediction model.

  Results for Home B:

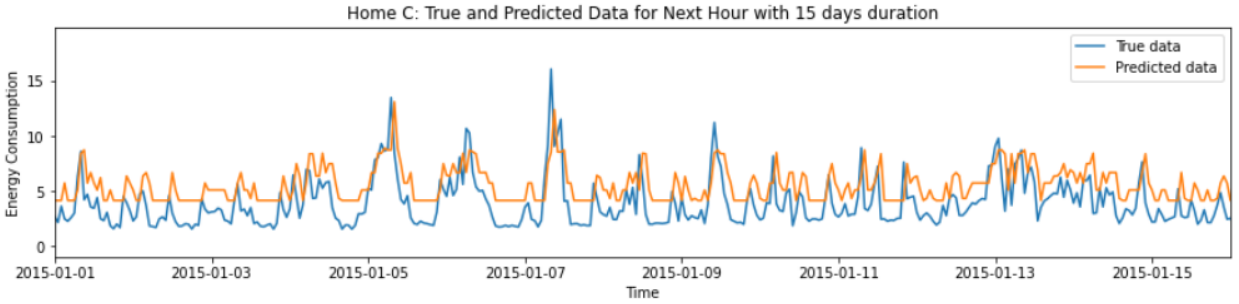
```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next hour =  6.061154551597378
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next day =  8.355118608259469
```

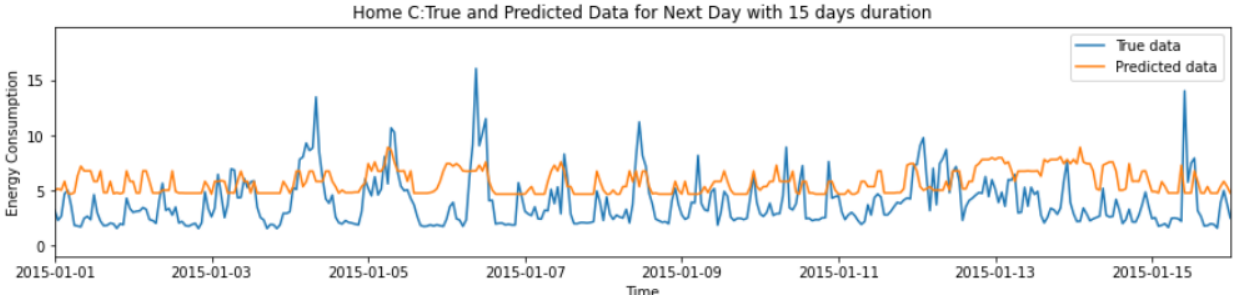




Results for Home C:

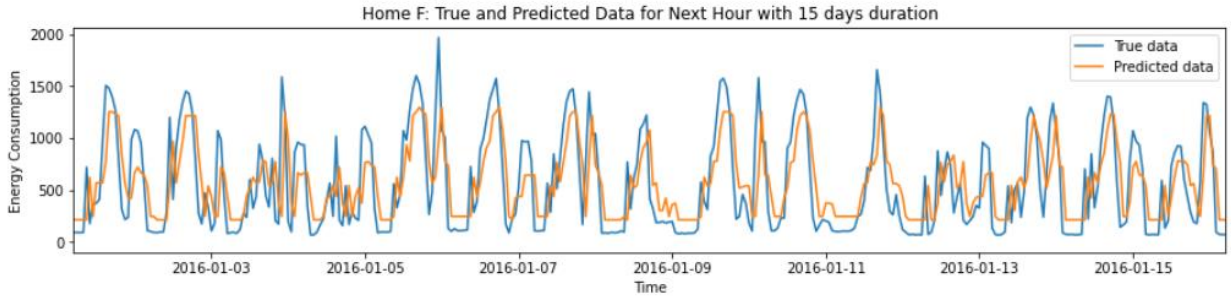
```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next hour =  2.0225948068990167
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next day =  2.609695907433157
```

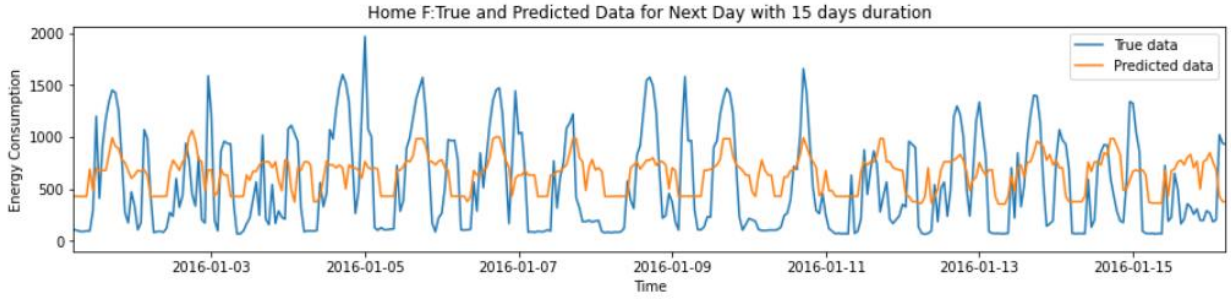




Results for Home F:

```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next hour =  222.11236983420932
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next day =  367.94131115214356
```



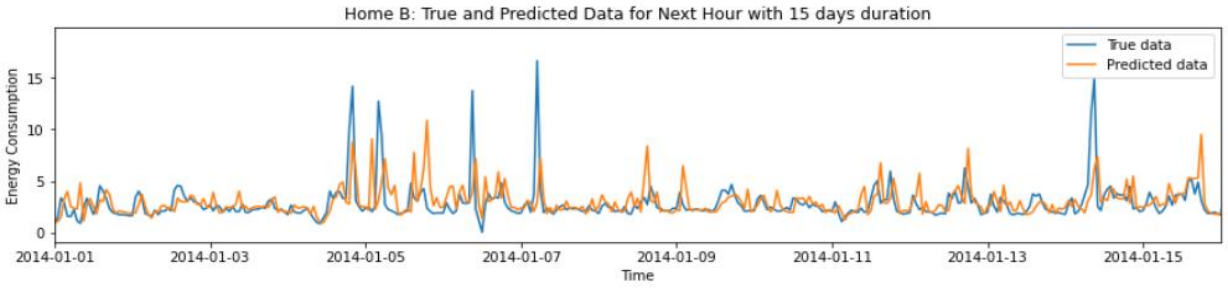


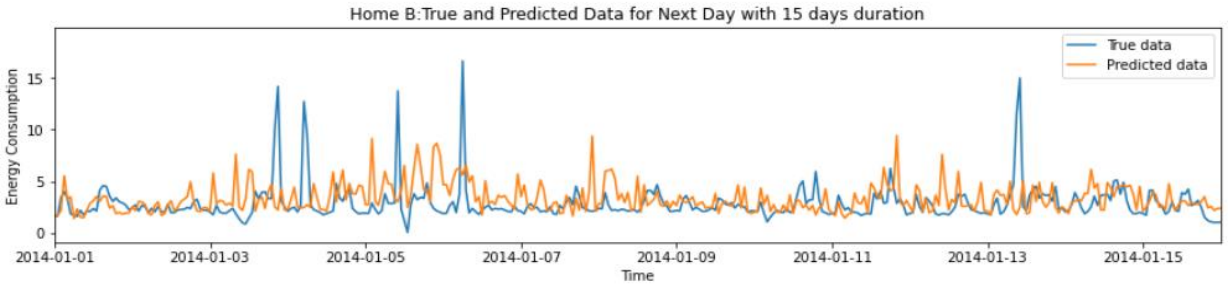
- **Model 4: Bagging Regressor Model**

  Used ***BaggingRegressor()*** of sklearn.ensemble library as prediction model.

  Results for Home B:

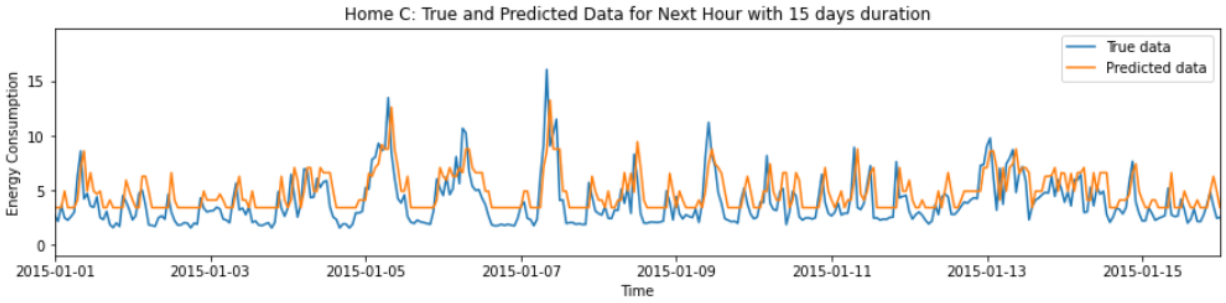
```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Bagging Regressor model for next hour =  1.1738039311881883
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Bagging Regressor model for next day =  1.4478147987636063
```

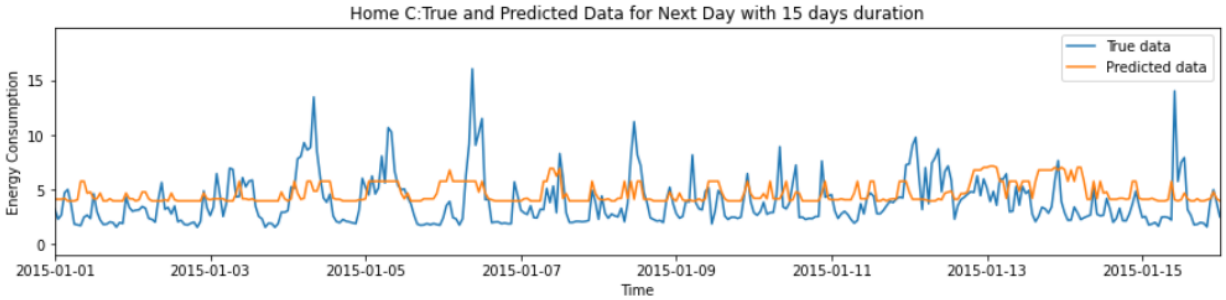




  Results for Home C:

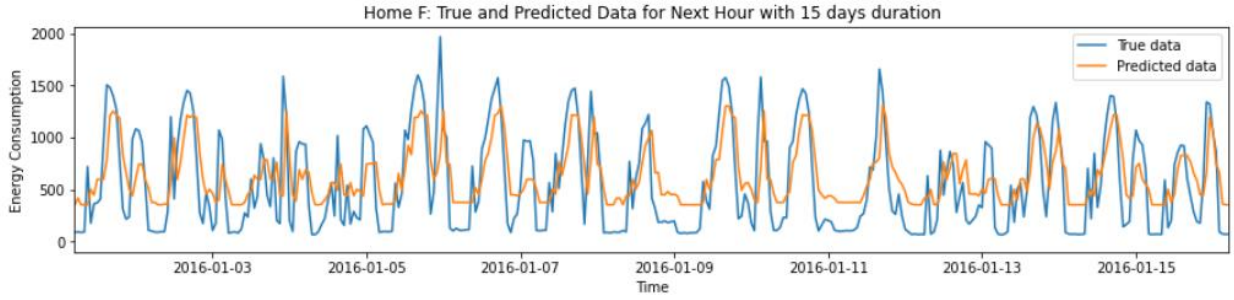
```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next hour =  1.5468287924125725
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next day =  1.8673422391283687
```

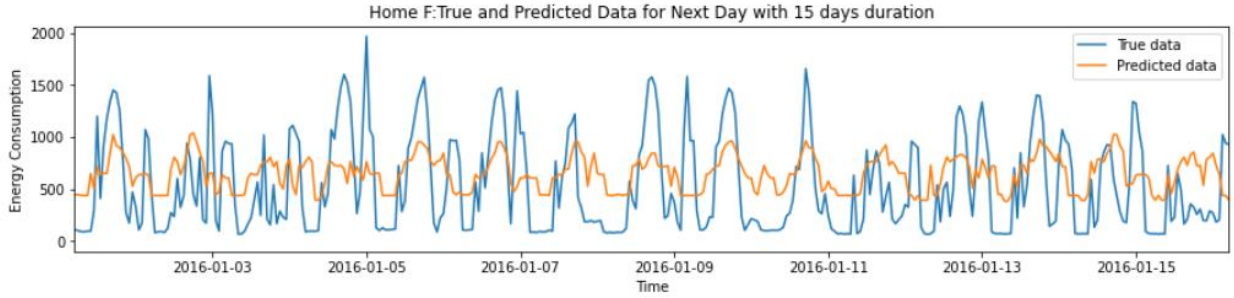




Results for Home F:

```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next hour =  277.215546650058
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for AdaBoostRegressor_model for next day =  370.8933062412737
```



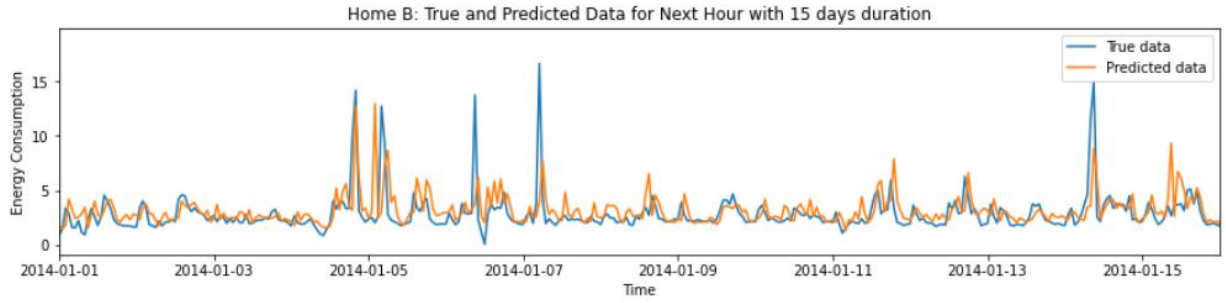


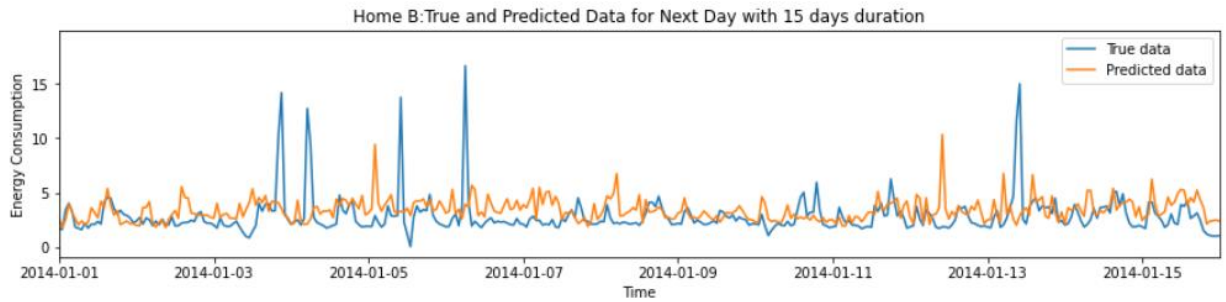
- **Model 5: Extra Trees Regressor Model**

  Used ***ExtraTreesRegressor()*** of sklearn.ensemble library as prediction model.

Results for Home B:

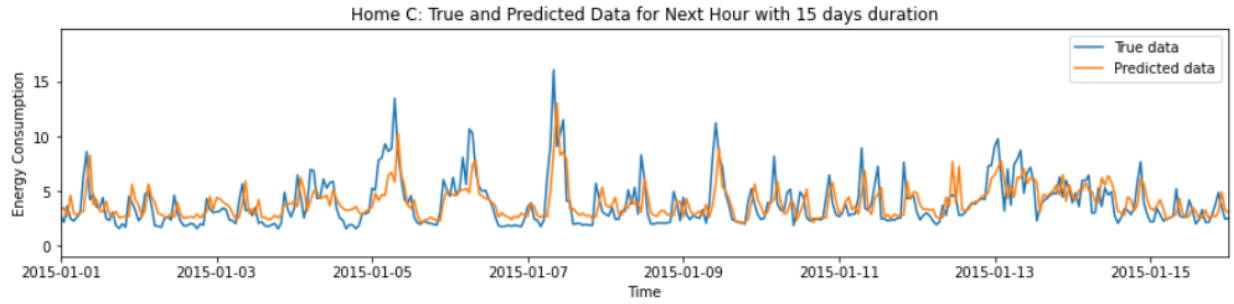
```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Extra Trees Regressor model for next hour =  1.1187040317962356
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Extra Trees Regressor model for next day =  1.3287182185263673
```

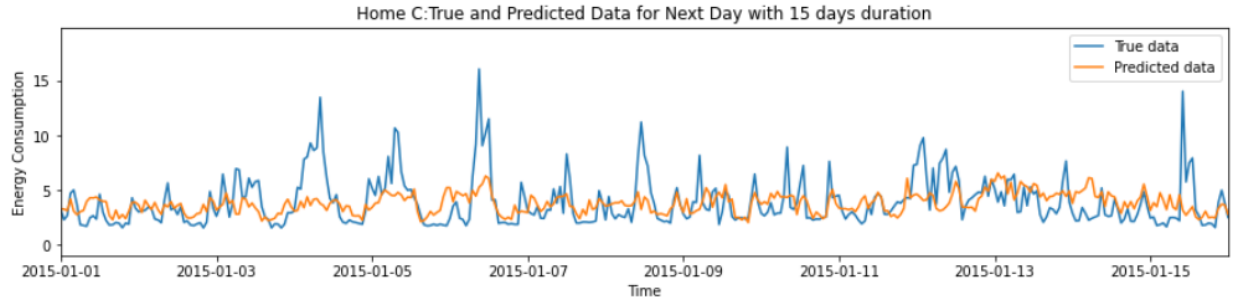




Results for Home C:

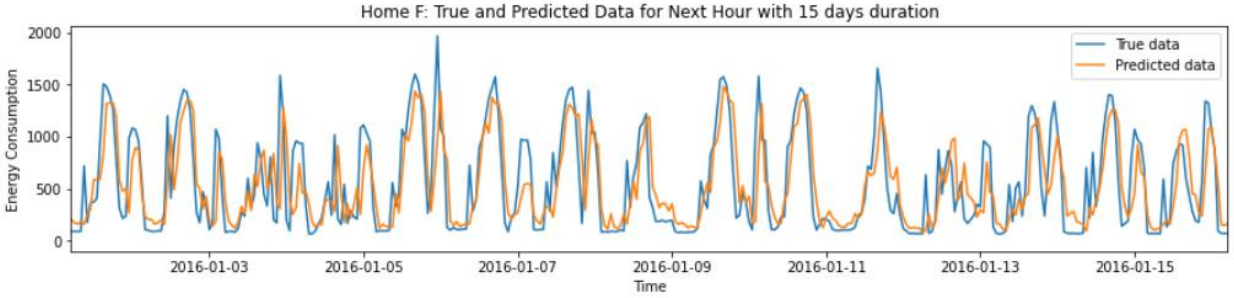
```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Extra Trees Regressor model for next hour =  1.2483299220008945
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Extra Trees Regressor model for next day =  1.6124561810150566
```

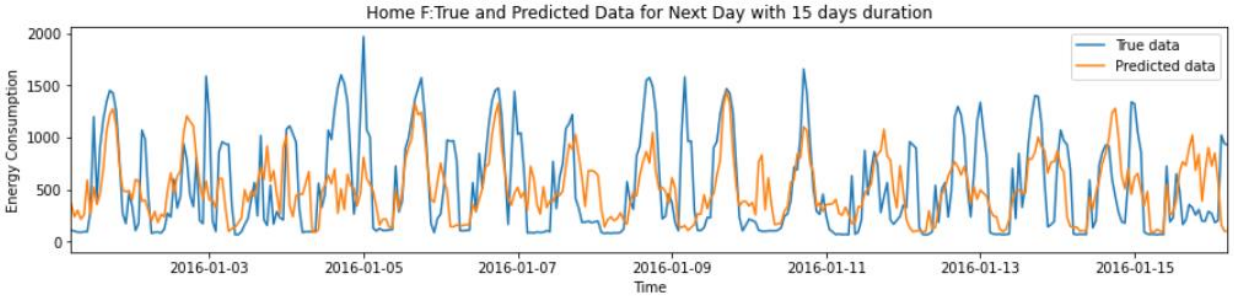




Results for Home F:

```
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Extra Trees Regressor model for next hour =  184.6792075919848
Spliting Dataset in training = 80% and testing = 20% ratio.
Mean Absolute Error for Extra Trees Regressor model for next day =  274.17772444614207
```

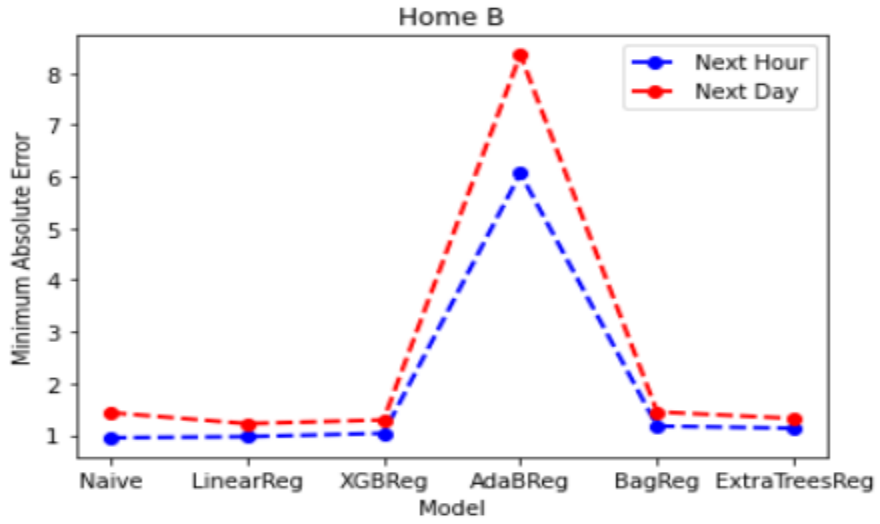




Comparison and Conclusion:

Home B:

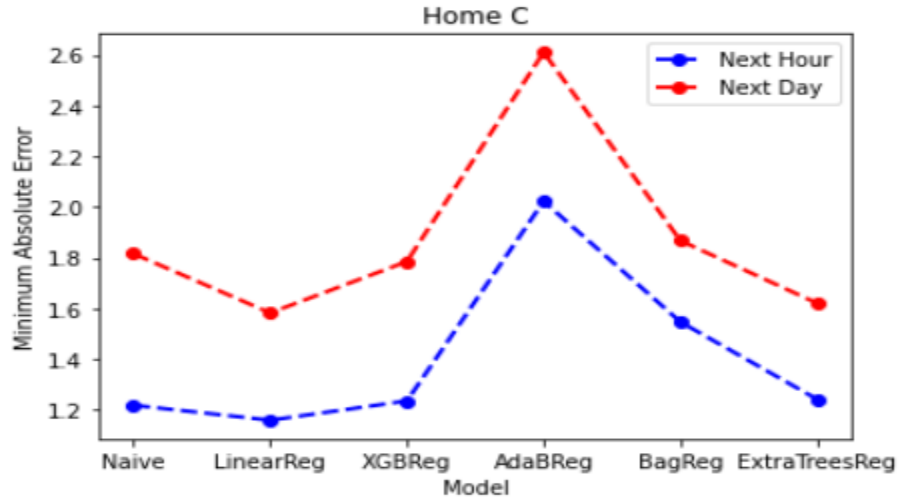
```
For Home B, Best Model for MAE of Next Hour: Naive Model
For Home B, Best Model for MAE of Next Day: LinearReg Model
```



- For Next hour, best model is Model 0: Naïve Model with MAE of .948038331438215.
- For Next Day, best model is Model 1: Linear Regression with MAE of .02209428742678372.

Home C:

```
For Home C, Best Model for MAE of Next Hour: LinearReg Model
For Home C, Best Model for MAE of Next Day: LinearReg Model
```

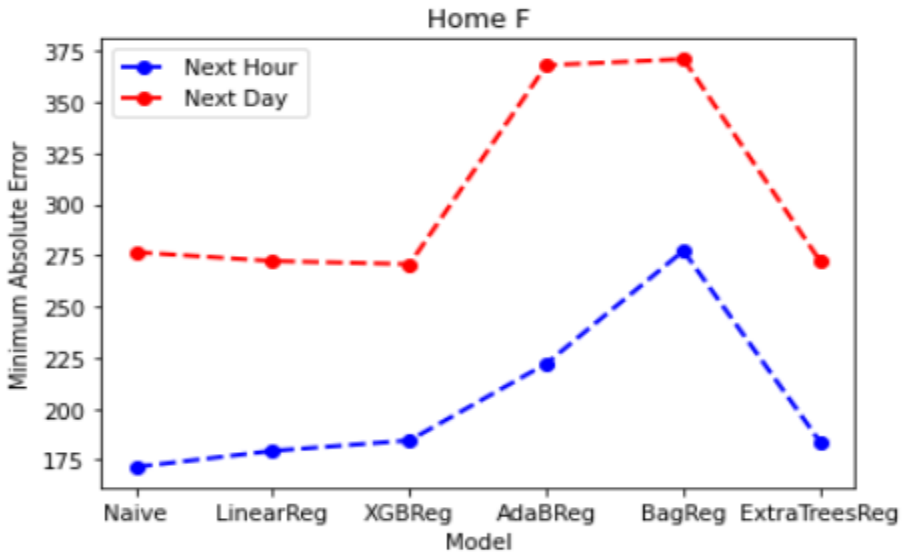


- For Next hour, best model is Model 1: Linear Regression with MAE of 1.159536898139981.

- For Next Day, best model is Model 1: Linear Regression with MAE of `1.5823510223243737.`

Home F:

```
For Home F, Best Model for MAE of Next Hour: Naive Model
For Home F, Best Model for MAE of Next Day: XGBReg Model
```



- For Next hour, best model is Model 0: Naive Regression with MAE of `171.5334679809642.`
- For Next Day, best model is Model 2: XGBRegressor with MAE of `270.6780752994049.`

We have handled different kind of challenges under data preprocessing in all the datasets of different homes. Then we used different model as prediction model for these time series datasets, considering the need of prediction of Next hour and Next Day.

After calculating Miniumum Absolute Error of all the prediction models, we have chosen the best models for all the different datasets of different homes for Next hour and Next Day.