Assignment - 1 : Electricity Demand Prediction

(AMS 559)

Rohan Vaish (111447435)
Jatin Sood (112079000)
Harsha Chandwani (111481387) (Team Leader)

Date - September 21, 2018

Data Preprocessing

While loading data from the respective csv files, a column of DateTime object type was added to identitfy each row of energy consumption (in kW) with a date and time.

Feature Engineering

Features are needed to train Linear regression, KNearest regression, Random Forest regression and AdaBoost regression models. Since there were no features in the data as it was, new features had to be discovered in the data. Some of the features which were discovered were:

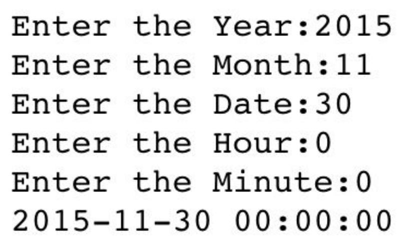
1. Seasons (Fall, Winter, Spring, Summer)
2. Holidays (Saturday/Sunday, any public holidays)
3. Peak hours

All the features were discovered after carefully observing the data for unsual hike and decline in energy consumption due to various factors.

```
for i, row in df.iterrows():
#handling seasons
    df.at[i, 'Year'] = row[1].year
    df.at[i, 'Date'] = row[1].day
    df.at[i, 'Month'] = row[1].month
    df.at[i, 'Hour'] = row[1].hour
    df.at[i, 'Minute'] = row[1].minute
    if row[1].month in range(1,4) :
        df.at[i, 'Summer'] = 1
    elif row[1].month in range(4,7):
        df.at[i, 'Spring'] = 1
    elif row[1].month in range(7,10):
        df.at[i, 'Fall'] = 1
    elif row[1].month in range(10,13):
        df.at[i, 'Winter'] = 1

#handling peak hours
    if row[1].hour in [0,1,2,3,4,5,6,7,20,21,22,23]:
        df.at[i, 'Peak Hour'] = 1
    else:
        df.at[i, 'Peak Hour'] = 0

#handling holidays & weekends
    if row[1].weekday() in [5,6] :
        df.at[i, 'Holiday'] = 1
    else:
        df.at[i, 'Holiday'] = 0

df['Spring'].fillna(0,inplace=True)
df['Fall'].fillna(0,inplace=True)
df['Summer'].fillna(0,inplace=True)
df['Winter'].fillna(0,inplace=True)
```

Training and Testing

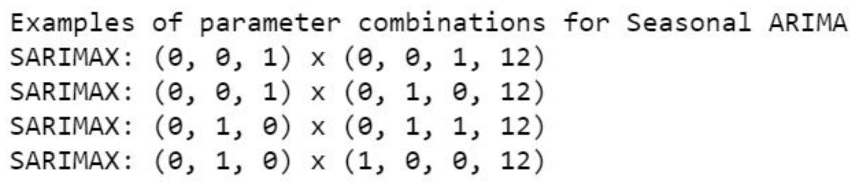
For the purpose of training, date/time input is taken from the user. Energy consumption before this date/time is used for training and energy consumption for next 96 timeslots is predicted.



```
Enter the Year:2015
Enter the Month:11
Enter the Date:30
Enter the Hour:0
Enter the Minute:0
2015-11-30 00:00:00
```

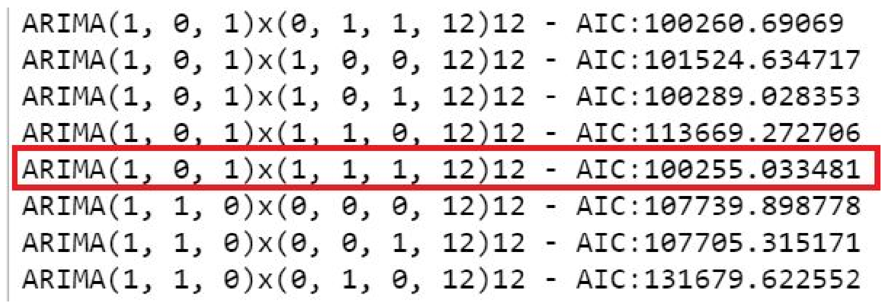
Linear regression, KNearest regression, Random Forest regression and AdaBoost regression models took the data before the user entered date/time as training set and predict energy consumption for next 96 timeslots. Using the predicted 96 timeslots, mean absolute error (MAE) is calculated.

Arima model has three parameters (p, d, q) which account for seasonality, trend and noise in data, where each of the parameter can assume either a value of 0 or 1.



```
Examples of parameter combinations for Seasonal ARIMA
SARIMAX: (0, 0, 1) x (0, 0, 1, 12)
SARIMAX: (0, 0, 1) x (0, 1, 0, 12)
SARIMAX: (0, 1, 0) x (0, 1, 1, 12)
SARIMAX: (0, 1, 0) x (1, 0, 0, 12)
```

We need to select the optimal parameters (p, d, q) for our Seasonal ARIMA model which yield us the best performance, i.e. the lowest possible AIC value.



```
ARIMA(1, 0, 1)x(0, 1, 1, 12)12 - AIC:100260.69069
ARIMA(1, 0, 1)x(1, 0, 0, 12)12 - AIC:101524.634717
ARIMA(1, 0, 1)x(1, 0, 1, 12)12 - AIC:100289.028353
ARIMA(1, 0, 1)x(1, 1, 0, 12)12 - AIC:113669.272706
ARIMA(1, 0, 1)x(1, 1, 1, 12)12 - AIC:100255.033481
ARIMA(1, 1, 0)x(0, 0, 0, 12)12 - AIC:107739.898778
ARIMA(1, 1, 0)x(0, 0, 1, 12)12 - AIC:107705.315171
ARIMA(1, 1, 0)x(0, 1, 0, 12)12 - AIC:131679.622552
```

For home1, we see the highlighted parameters yield us the least AIC value, therefore it is the optimal combination of model parameters. We store this combination in a data tuple “pdq”. Hence we train our model using “pdq” combination of parameters.

```
mod = sm.tsa.statespace.SARIMAX(dft,
                                order=(pdq[1], pdq[2], pdq[3]),
                                seasonal_order=(pdq[4], pdq[5], pdq[6], 12),
                                enforce_stationarity=False,
                                enforce_invertibility=False)
results = mod.fit()
```

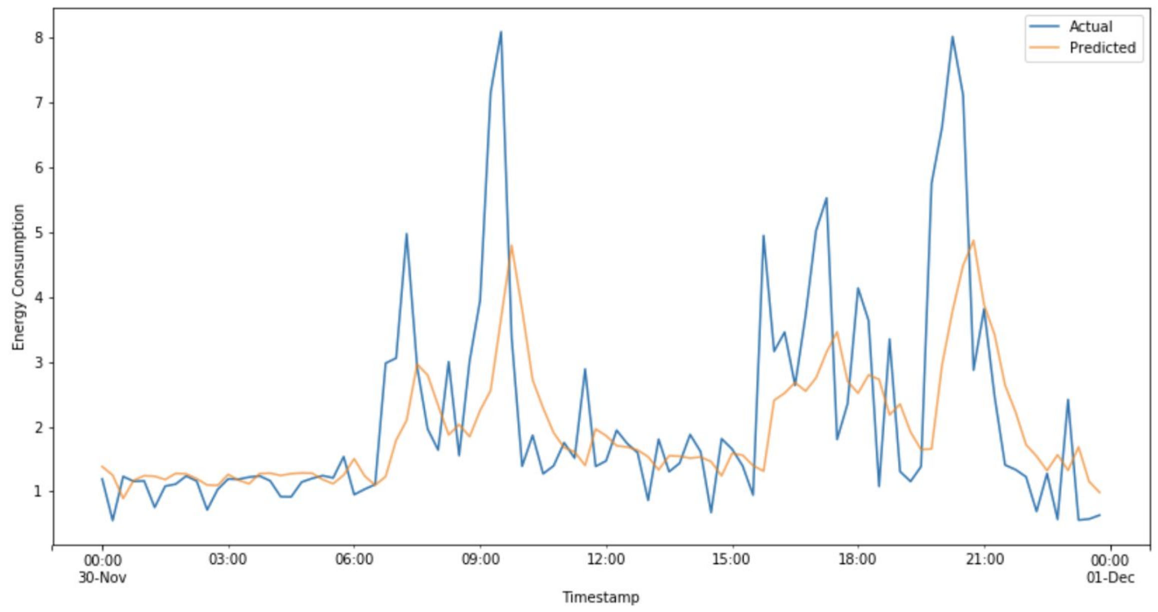
Models, Plots and Mean Absolute Errors for Home 1:

1. SARIMA Time Series Forecasting (kW)

   (Best Model according to our experiment)

   SARIMA stands for “Seasonal Autoregressive integrated moving average” which takes into account the seasonality in the observed data. Seasonality means repeatition of a particular pattern in data after a certain time interval.

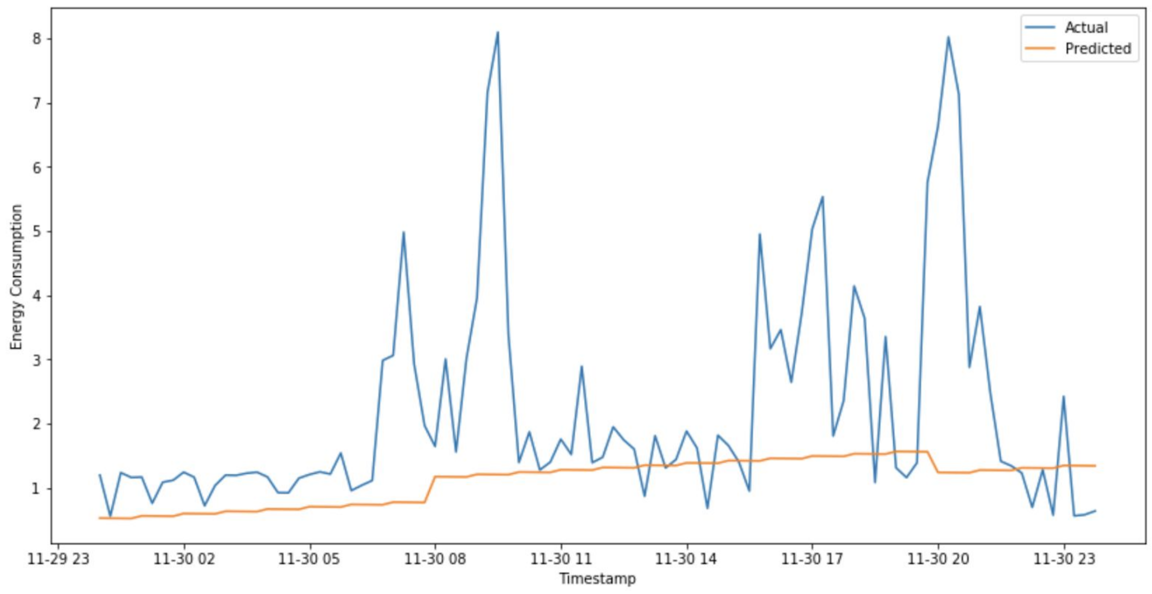
```
Mean absolute error SAR: 0.885857739
```



2. Linear Regression (kW)

   Using Linear regression, we study the relationship between the observed veariable and mutiple predictor variables.

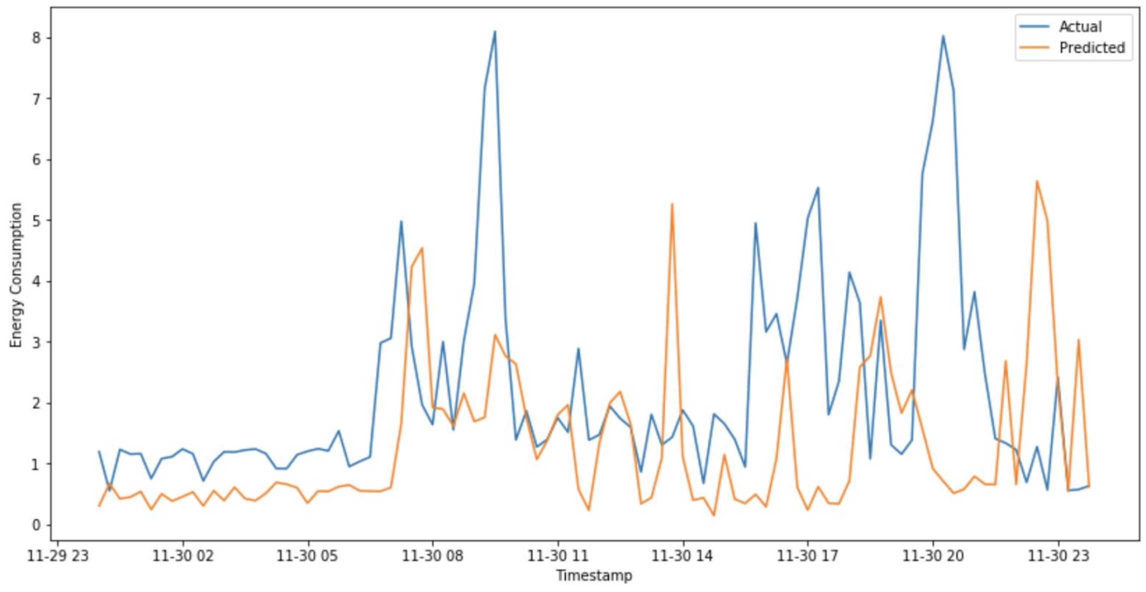
```
Mean absolute error LR: 1.206663472
```



3. KNeighbors Regression

   KN regressor observes all the cases, stores them and predicts the target value by using a similarity measure (like distance function).

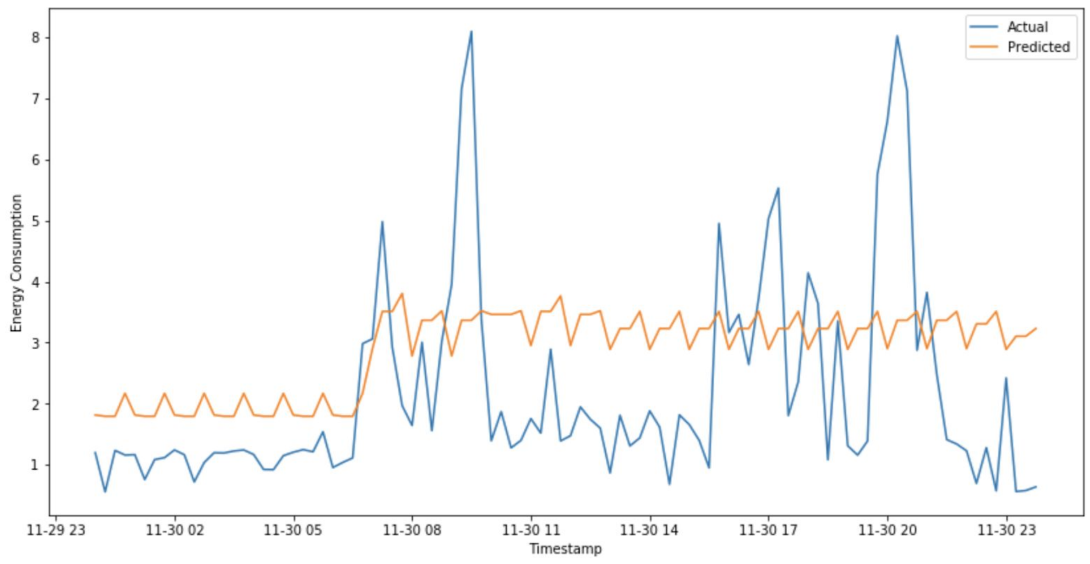
```
Mean absolute error KN: 1.504697500
```



4. AdaBoost Regression (kW)

Adaboost regressor is a meta-estimator which tries to fit the original data to a regressor and then fits the data on the more copies of the regressor while adjusting the weights of instance being adjusted according to error on the existing prediction.

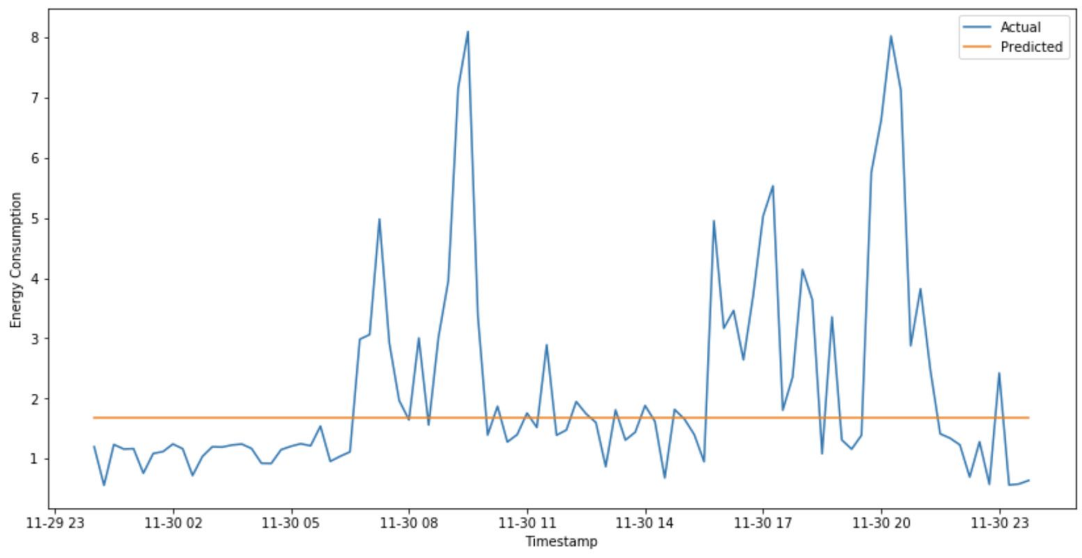
```
Mean absolute error AB: 1.433756127
```



5. Random Forest Regression (kW)

Random forest regressor builds various decision trees and combines together to get more accurate prediction.

```
Mean absolute error RF: 1.116673598
```



Comparison of results

Mean Absolute Error (kW):

Home ID	Naive	SARIMA	Linear	KNeighbor	Adaboost	Random Forest
1	1.18	**0.886**	1.206	1.504	1.433	1.116
2	1.32	**1.183**	1.409	1.842	1.244	1.411
3	1.24	**1.069**	1.519	1.764	1.155	1.193
4	1.21	0.846	0.793	1.098	0.872	**0.745**
5	1.15	**1.076**	1.433	1.636	1.114	1.287
6	0.72	**0.964**	0.973	1.358	1.029	1.063
7	1.13	**0.645**	1.255	1.291	0.803	1.118
8	1.62	**1.072**	1.383	1.482	1.116	1.219
9	0.44	**0.654**	0.829	1.207	0.669	0.885
10	1.57	**0.972**	1.261	1.439	0.996	1.018