\*\* Analysis Write Up :\*\*

1. Data Engineering

2. Solution

3. Problem methodology

4. Code

5. Analysis write-up

I have tried to solve this problem in a given procedure:

Some of the rules that a regression problem should follow and be careful about , which also applies to this dataset and problem given are the following :

* Correlation between features < Threshold value - check for mulit-collinearity between the features
* Correlation between features and y should be high - checked that by taking only top N features with highest correlation with ‘kwh’ amongst all columns/features
* Features to be standardised and scaled - categorical features not scaled/standardized, only numeric features like area, etc. scaled
* Check and combine features if they can be combined - could have combined some features like area of house, total area, room area - but decided to keep it as separate features in order to not lose any relevant information - also because the check for collinearity is already eliminating any similar features

Some important points observed and taken care of while coding :

**SELECTING FEATURES AND DATA ENGINEERING**

**UNDERSTANDING THE DATA SOURCE**

First understanding the dataset given and the data source helps in forming the base for further analysis and probing into the dataset.

We do visual analysis, statistical analysis and other common methods to understand the features space, datatypes in dataset, categorical and numerical features, correlation between features.

Also, we can use feature engineering methods and also ways to visualize the features.

# initially selected/shortlisted 79 features from all columns to have good correlation score with output column to be predicted - KWH. These were the top features with highest correlation to the column needed to be predicted -> ‘kwh’ or y value in this case.

**FEATURE ANALYSIS AND FEATURE ENGINEERING**

# then some feature analysis and dataset evaluation is done, once we get the features to understand the nature of these features, how to move forward with dropping and keeping features.

We drop features with correlation = 1 as that suggests , after dataset analysis, that it is just a change of unit type of KWH , thus should not be included as a feature.

Also some features may have been similar in nature and a method is to combine these features so as to reduce dimensionality and hence help the model to perform better on given number of datapoints. [Curse of dimensionality should be avoided].

This also meant that the features given have to be selected with various filters so as to keep them minimum in number, so that the model can predict better.

I make a two stage filter for this purpose :

# check for mulit-collinearity and avoid similar features to be fed to ML model

# I compare the correlation amongst themselves for all the selected features and remove any one of two features that have a correlation higher than 0.8 (threshold that can be 0.9 as well - but seemed to work best for 0.8 value- not much statistical basis, more of trial and error and heuristical value). So the redundancy in information is stemmed and information gain is improved by filtering these features given to the model later.

# Also to verify this , we do a visual inspection of the feature set by plotting a heat map of the correlation matrix of features dataframe. On comparing it with earlier heatmap, it suggests that the collinearity is significantly reduced amongst the feature space now.

# Only 25 features are selected now , as per collinearity conditions and 2 stage filtering techniques applied.

**SELECTING BEST COMBINATION OF FEATURES**

# Now selecting the best combination of features to be fed to the ML model by trying out various combinations of the selected features based on p-value test

# This stage of filtering helps us choose the best set of features which , when fed to the ML model together, would result in best prediction for the ML model. Since some ML models, given a particular feature combination/feature space, tend to perform better than some other feature combination, it may be a good idea to choose the best feature combination that can work best for this regression problem for the ML model.

# For this, we use the p-value hypothesis test method , where the null hypothesis is - The selected combination of dependent variables does not have any effect on the independent variable”.

And then we build a small regression model to calculate the p-values.

And if the p values is higher than the threshold, we discard that combination of features.

In the end, we come up with a list of features to be selected from the previously filtered feature set, and filter out the rest of the features for next steps.

# Standardize and normalize the dataset for the ML models

#for now not scaling the data

# But scale only the numerical features, not the categorical features - so scaling columns like area, dollarel etc.

**Building the ML model with selected features :**

# For the purpose of this type of problem, I will try to test various families of models and analyze the results for each of them. After careful error analysis and result evaluation, I can declare a champion model amongst the models , for this problem.

I choose 3 ML models very diverse from each other belonging to different families -

linear regression,

xg boost - ensemble learner with base model being a tree -> Gradient Boosting Tree

Unlike Random Forest, which is a parallel growth of deep decision trees (low bias, high variance) being averaged together, in case of GB Tree used here, we have weak learners (or shallow trees- high bias, low variance) being boosted sequentially and then averaged (to reduce variance).

Neural Network - Another classical example model to be explored as it may seem like a non-linear solution space with many variables and thus the function can be a non-convex one. In this case, to converge to an optimal minima, we can depend on a neural network. Gradient descent or Adam as the optimizer and relu as the activation function can help converge to the minima faster. But the prediction function y = f(x) itself may be a simple linear one, in which case NN may fail even after using more computation power.

Thus we include two subcategories in this model ->

(i)with small number of neurons - which is to see if the model is not overfit

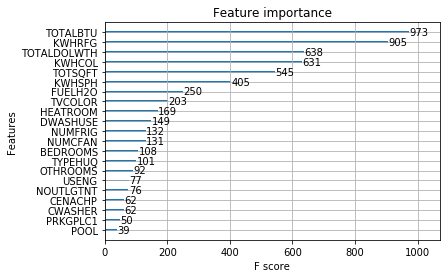
(ii)another with large number of neurons - which is to see if there are more data points compared to feature space, and hence overcoming curse of dimensionality

# Then split dataset into test and train dataset for cross validation

Since there is no test dataset given to check the validity of the model and it’s prediction accuracy, we need to divide the dataset given itself into test and train sets. Exposing the model only to the training set while training, we can then predict on the test set and see accuracy of the predicted values.

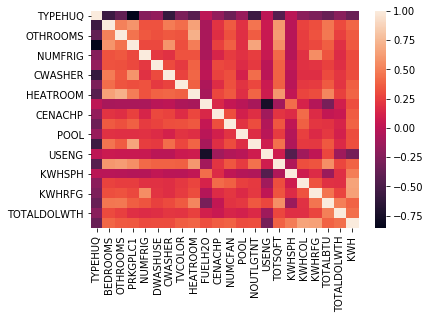
# to indicate good features and their importance, plot feature importance scores or F1 scores for features input to ML model

Also the feature importance score is important to see the relevance of the features being given to the ML model, and how the features compare to each other in terms of importance to the information gain it provides the model for decision making.

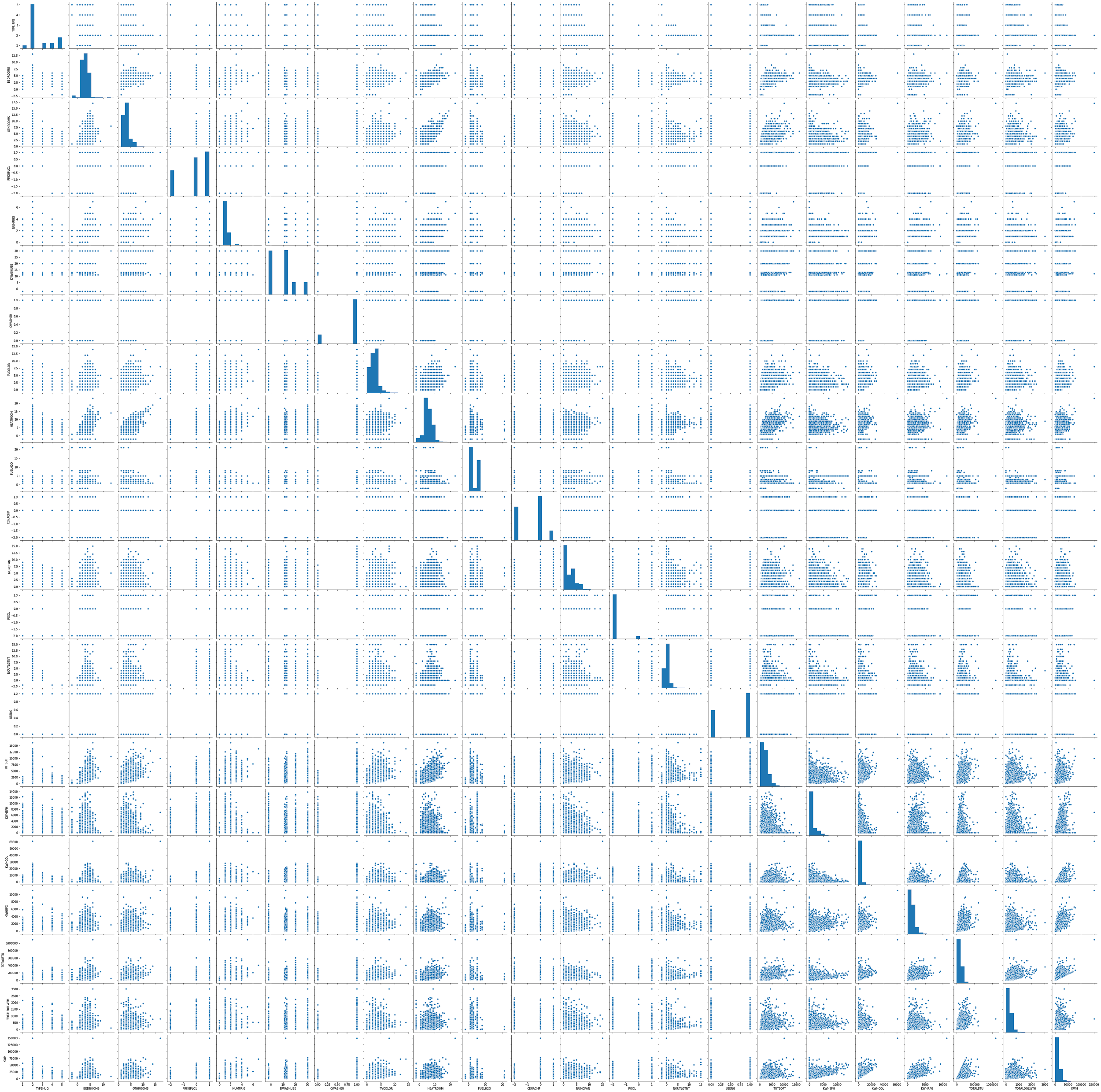


**Feature analysis/ Selection and Data Engineering :**

Seaborn heatmap for correlation matrix between features :



Also part of feature engineering and Model selection is the distribution of data between features :

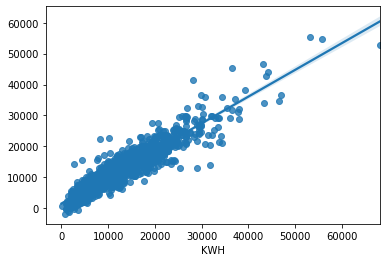


In the above plots, we see some of the values are categorical and some are numerical. Also some graphs have a regression and patterns that can be extracted from an ML model. This is a visual inspection, which helps in choosing a good ML model that would fit the dataset and the distribution of feature space observed in above plots.

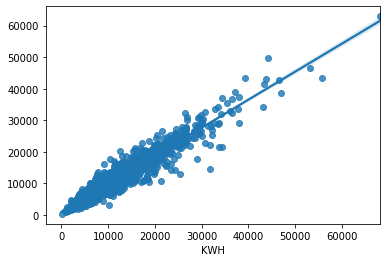
**Outputs for ML models :**

This gives the RMSE score for predicted values of consumption on one axis and actual values on another axis

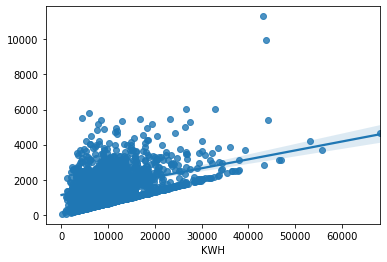
-- FOr Linear Model --



-- For XG boost model --



-- For Neural Network model --



print("The minimum RMSE goes to: %.2f" % min([rmse\_linear, rmse\_nn, rmse\_xgb]))

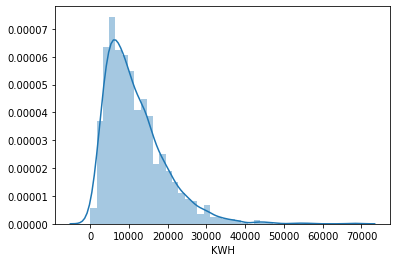
# XG boost model seems to be the champion model here with RMSE :

The minimum RMSE goes to: 2228.23 XG boost model

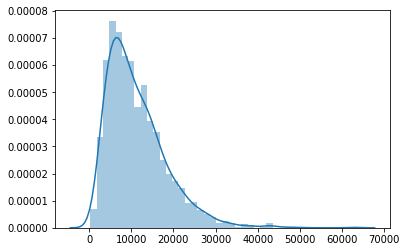
So the champion model in this case is the xGBoost model, but the linear model is also very close in terms of error.

Using these top 3 three methods for regression and comparing the results by using RMSE method for error analysis, we can then choose the champion model, which best fits the dataset for prediction with least error.

# Plot the Final dataset test histogram - y\_test



# Plot the FInal dataset predicted by Best mode - xgb



Thus the histogram plot assumes a similar bell curve shape with peak on around 10,000 value of the 'KWH' bins on x-axis with test and the predicted values. It can also give a hint that the prediction values are following the same trend as the real values , and it can be gauged as another confirmation of the model’s accuracy on one level.