# Practical Machine Learning

## ML Algorithm Selection Report

Sanjeeb Humagain

April 18, 2024

**Introduction:**

The main goal of this exercise is to use Machine Learning (ML) approach to select the appropriate model for any specific task and dataset. In this report, the methods and its performance of the model is presented which will help to analyze and compare model performance. There are a lot of methods which could be used to create a model in ML. However, selection of appropriate model for a particular dataset can be quite challenging task. To get the best model, different methods should be examined and compared with each other.

To select an appropriate ML algorithm, first we need to understand the problem. For instance, if the problem is related to classification we have to choose a different algorithm than an algorithm for prediction. For instance, Linear Regression is used for forecasting and Logistic Regression algorithm is used for classification problems.

Correlation of features were checked and there were some negative correlations as well. A negative correlation is a relationship between two variables that move in opposite directions. From the correlation plot of heatmap, it was fond that some features have negative correlation with system load indicating that system load decreases as the particular feature increases and vice versa. The one advantage of checking the correlation is that we could remove the features which has very less correlation or zero correlation with the data we want to forecast. Removing such features would reduce the complexity and might ultimately help to increase the model performance.

**Dataset Description:**

In this experiment, the dataset has 9 features which are used to predict the system load. The data set contains 87648 samples of data and the target system load from 2006 to 2011. In this exercise, the system load data is being used to compare machine learning algorithms and compare their performance. One important thing to remember is that we cannot entirely be sure which model is best in this exercise because the hyper parameter optimization has not yet been used in this exercise. The performance of the models cannot be compared when we have suboptimal solutions. However, we will try to summarize the model performance from the available data and information. In all ML algorithms used, roughly same steps are followed. They are, checking for the missing value, data preprocessing, clean up, scaling, split up, creating model, training the model and running the test etc.

**Experimental Setup:**

Python is used for this experiment because of the availability of crucial libraries for instance Pandas (for data manipulation), Scikit-learn (for ML) etc. The dataset imported from the csv file and split into training and testing sets in ratio of 80:20.

Six ML algorithms are used for evaluation namely, Random Forest (RF), Support Vector Machine (SVM), Decision Tree (DT), Gradient Boosting (GB), Neural Network (NN), and Multiple Linear Regression (MLR).The performance of each model was assessed using r2 score. Finally, the most appropriate algorithm was selected based on the available information.

**Results:**

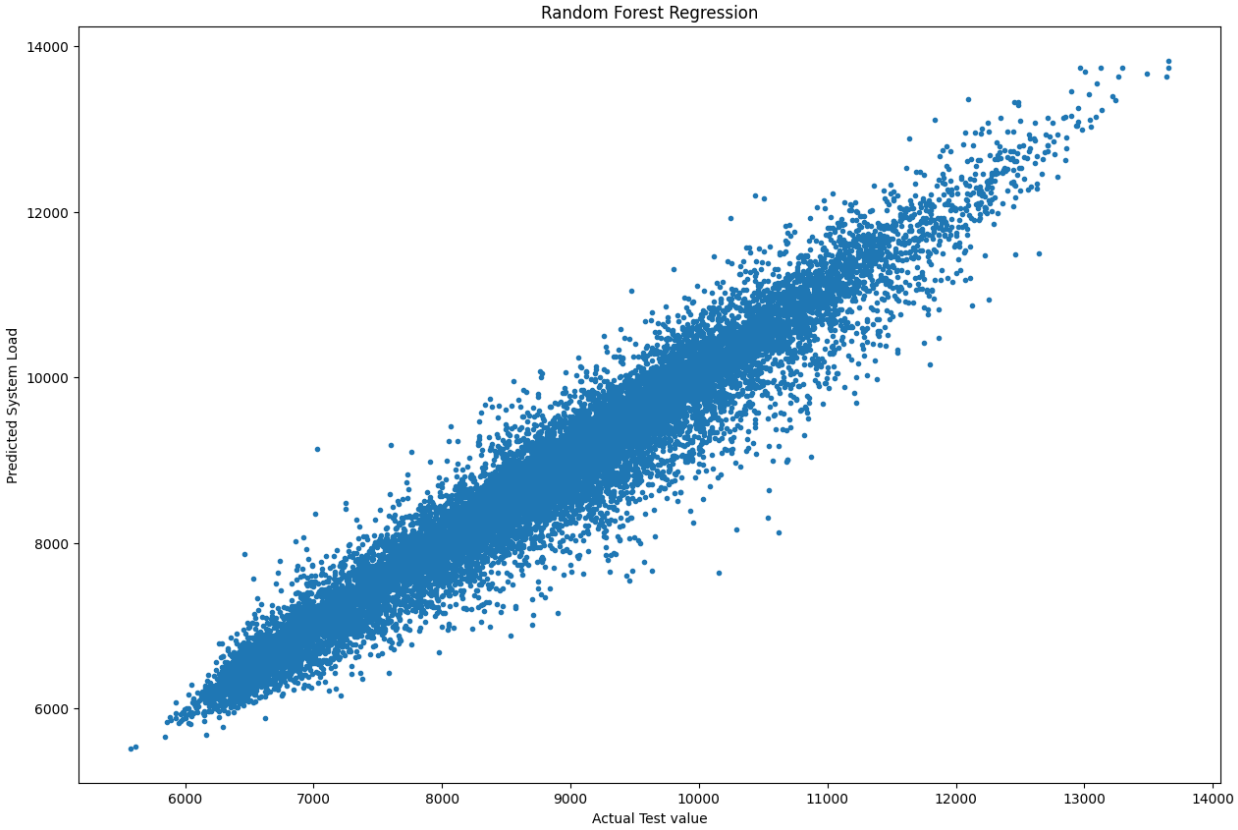
The R2 Score result of the ML algorithm used for this exercise is tabulated below in a descending oreder.

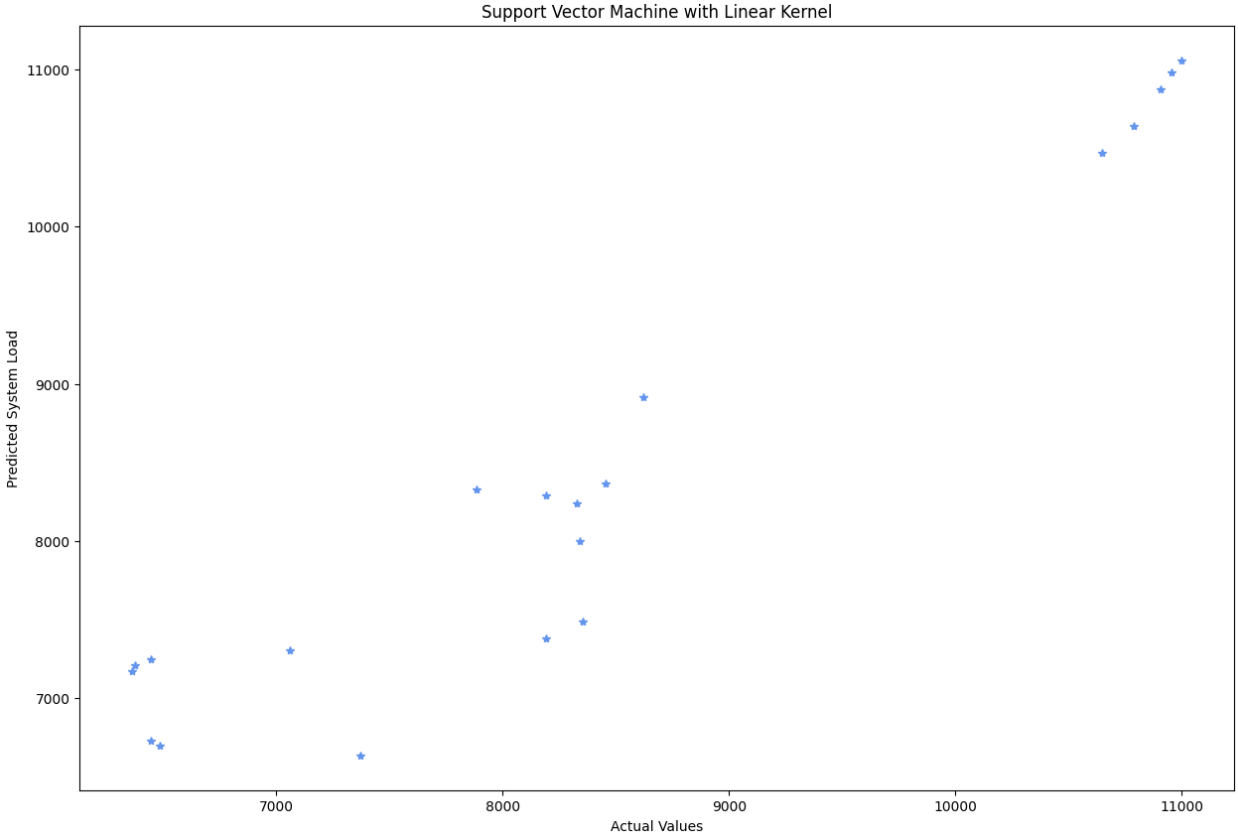
| S.N. | ML Algorithm | R2 Score |
|---|---|---|
| 1 | Random Forest (RF) | 0.9412 |
| 2 | Support Vector Machine (SVM) | 0.9128 |
| 3 | Decision Tree (DT) | 0.9004 |
| 4 | Neural Network (NN) | 0.7595 |
| 5 | Multiple Linear Regression (MLR) | 0.6185 |
| 6 | Gradient Boosting (GB) | 0.4733 |

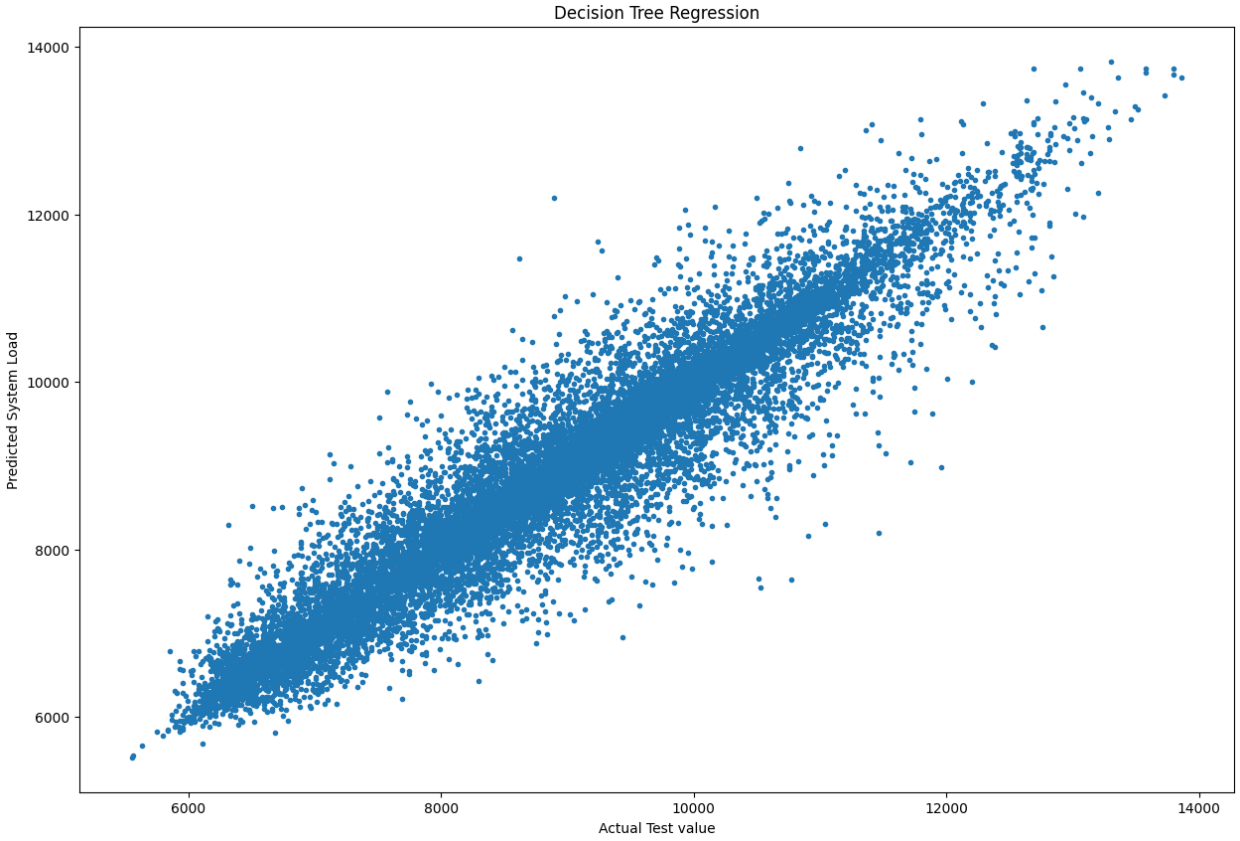
According to the observation Random Forest (RF) stands as the best model for this data with maximum R2 Score of 0.9412. However, the hyper parameters used in each of the algorithm are not optimal. The next exercise is of Hyper Parameter Optimization which might be used to find the optimal solution for each algorithm. There were some observations which I would like to mention here. Multiple Linear Regression had better performance for when the dataset size was 3525. The performance decreased when tried to include more data. Additionally, in Support Vector Machine algorithm, the model worked well for small number of data but when tried to input all the data, it would take forever to give the result. The number of data required and the execution time was very less for SVM.

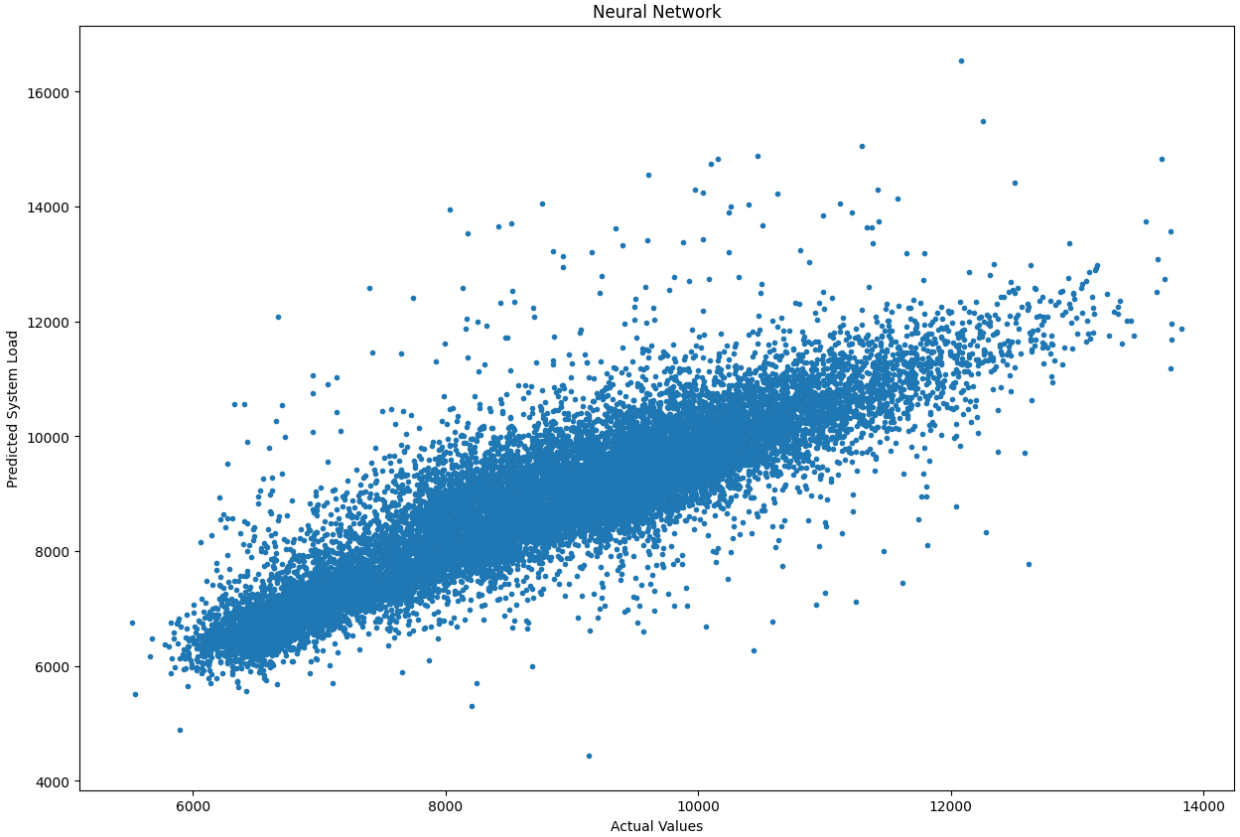
In conclusion, after the HPO the order of the performance as listed in above table might change as we might be working in local optimum regions.

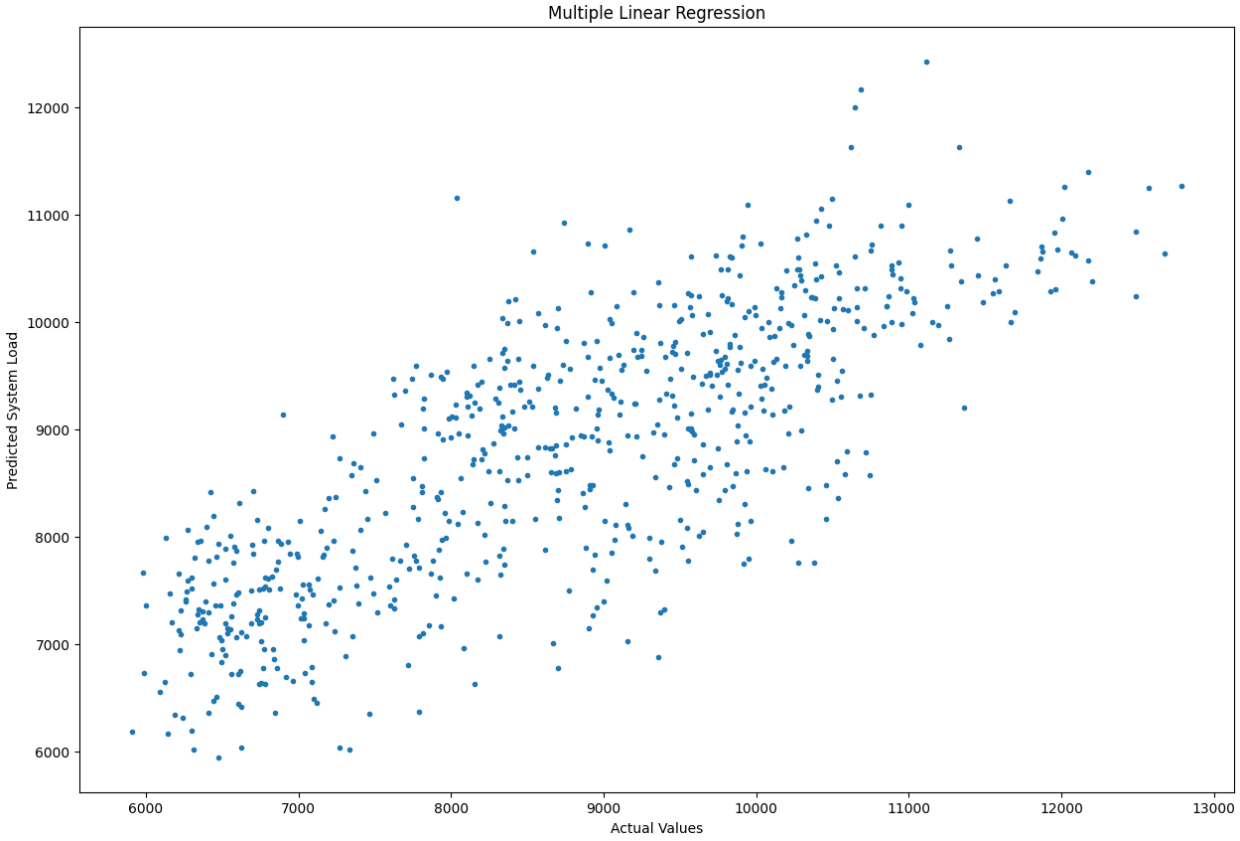
The Plots of all the algorithms used are listed below.

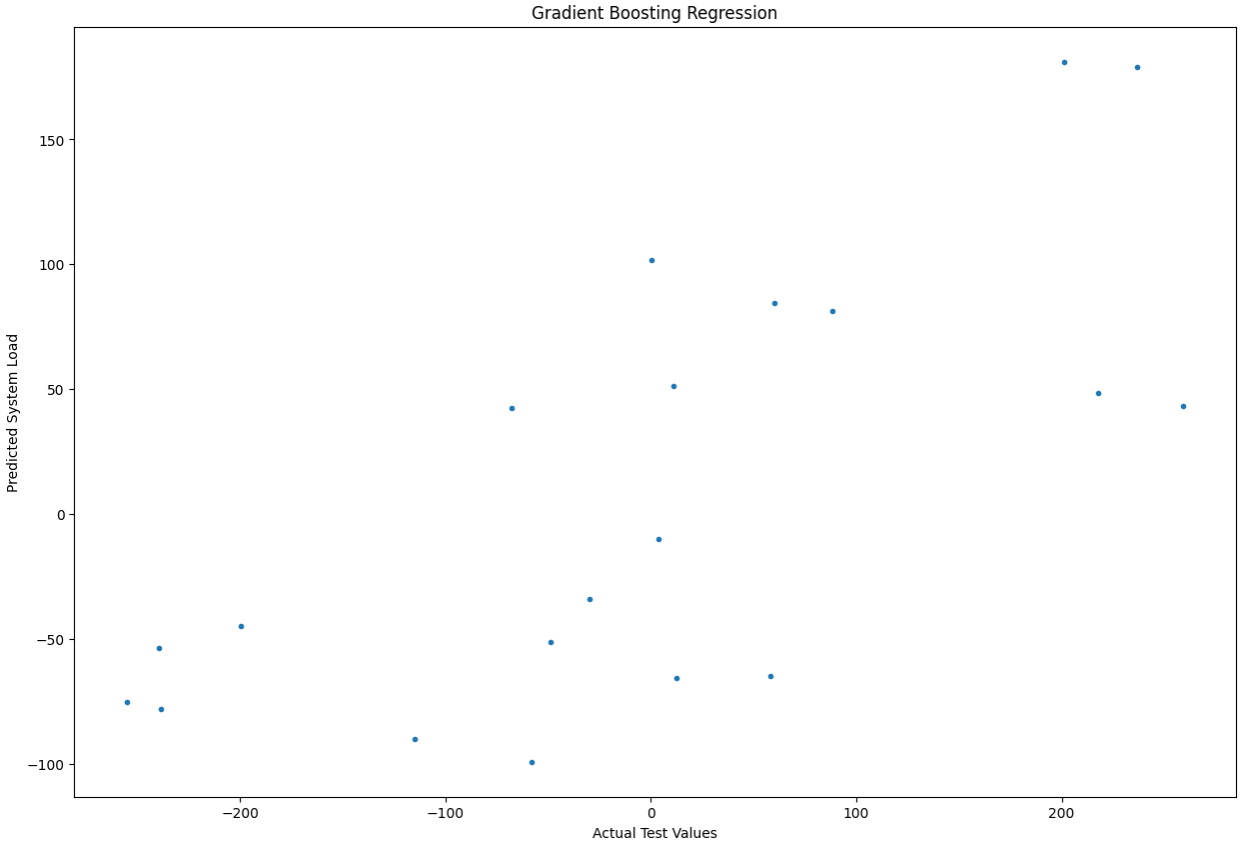

Random Forest Regression
Predicted System Load
Actual Test value
14000
12000
10000
8000
6000
6000
7000
8000
9000
10000
11000
12000
13000
14000


Support Vector Machine with Linear Kernel
Predicted System Load
Actual Values
11000
10000
9000
8000
7000
7000
8000
9000
10000
11000


Decision Tree Regression
Predicted System Load
Actual Test value
14000
12000
10000
8000
6000
6000
8000
10000
12000
14000


Neural Network
Predicted System Load
Actual Values
16000
14000
12000
10000
8000
6000
4000
6000
8000
10000
12000
14000


Multiple Linear Regression
Predicted System Load
Actual Values
6000
7000
8000
9000
10000
11000
12000
13000


Gradient Boosting Regression
Predicted System Load
Actual Test Values
150
100
50
0
−50
−100
−200
−100
0
100
200