**OR 568 Final Project Report**

**Predictive Analytics on Appliances Energy**

**Aishwarya Maruthappan | Asmitha Rao Annamaneni | Manish Reddy Kunchala |**

**Tejasri Surapaneni | Venkata Rama Sai Anumula | Vikas Marupadiga**

**1.0 Abstract**

As demand for cost-effective energy and resource management continues to grow, intelligent automated building solutions are necessary to reduce energy consumption, increase alternative energy sources, reduce operational costs and find interoperable solutions that integrate with legacy equipment without massive investments in new equipment and tools. The ability to analyze, understand and predict building behavior offers tremendous opportunities to demonstrate and validate increased energy efficiencies, which may ease many exorbitant pressures taxing the grid. This Dataset is for data-driven predictive models for the energy use of appliances. Data used include measurements of temperature and humidity sensors from a wireless network, weather from a nearby airport station and recorded energy use of lighting fixtures.

**2.0 Data**

The dataset chosen is Appliance’s Energy Prediction dataset from the UCI Machine Learning Repository. This dataset helps in predicting the energy utilization of appliances. Data is collected by measuring the temperature and humidity using wireless sensor networks. The data was captured every ten minutes in order to record any quick changes in energy consumption. The house temperature and humidity conditions were monitored with a ZigBee wireless sensor network.

Each wireless node transmitted the temperature and humidity conditions around 3.3 min and the wireless data was averaged for 10 minutes periods. The weather data from the nearest airport weather station is merged with this to evaluate its impact on the energy consumption of appliances. Two random variables are used in this dataset to test the regression model and to eliminate the non-predictive attributes (UCI Machine Learning Repository, n.d.).

**Source:** Dataset has been chosen from UCI Machine Learning Repository - [Center for Machine Learning and Intelligent Systems](http://cml.ics.uci.edu/).

**Website Link:** <https://archive.ics.uci.edu/ml/>.

**2.1 Predictors**

 1. Date time – represents the date and time when the temperature was calculated

2. Appliances - shows the energy utilization in Watts

3. lights - shows the energy utilization of light fixtures in the house in Watts

4. T1 - Temperature in the kitchen area, in Celsius

5. RH\_1- Humidity in the kitchen area, in %

6. T2 - Temperature in the living room area, in Celsius

7. RH\_2- Humidity in living room area, in %

8. T3 - Temperature in the laundry room area

9. RH\_3 - Humidity in laundry room area, in %

10. T4 - Temperature in office room, in Celsius

11. RH\_4 - Humidity in office room, in %

12. T5 - Temperature in the bathroom, in Celsius

13. RH\_5 - Humidity in the bathroom, in %

14. T6 - Temperature outside the building (north side), in Celsius

15. RH\_6 - Humidity outside the building (north side), in %

16. T7 - Temperature in ironing room, in Celsius

17. RH\_7 - Humidity in ironing room, in %

18. T8 - Temperature in teenager room 2, in Celsius

19. RH\_8 - Humidity in teenager room 2, in %

20. T9 - Temperature in parent’s room, in Celsius

21. RH\_9 - Humidity in parent’s room, in %

22. To - Temperature outside (from Chievres weather station), in Celsius

23. Pressure (from Chievres weather station) - in mmHg

24. RH\_out - Humidity outside (from Chievres weather station), in %

25. Wind speed (from Chievres weather station), in m/s

26. Visibility (from Chievres weather station), in km

27. Tdewpoint (from Chievres weather station), Â°C

The above predictors help in knowing the appliances used in which room or environment consumes less energy and the factors behind it to inculcate these methods and methodologies in our day to day life to contribute to the energy-saving resources.

**Raw Dataset**

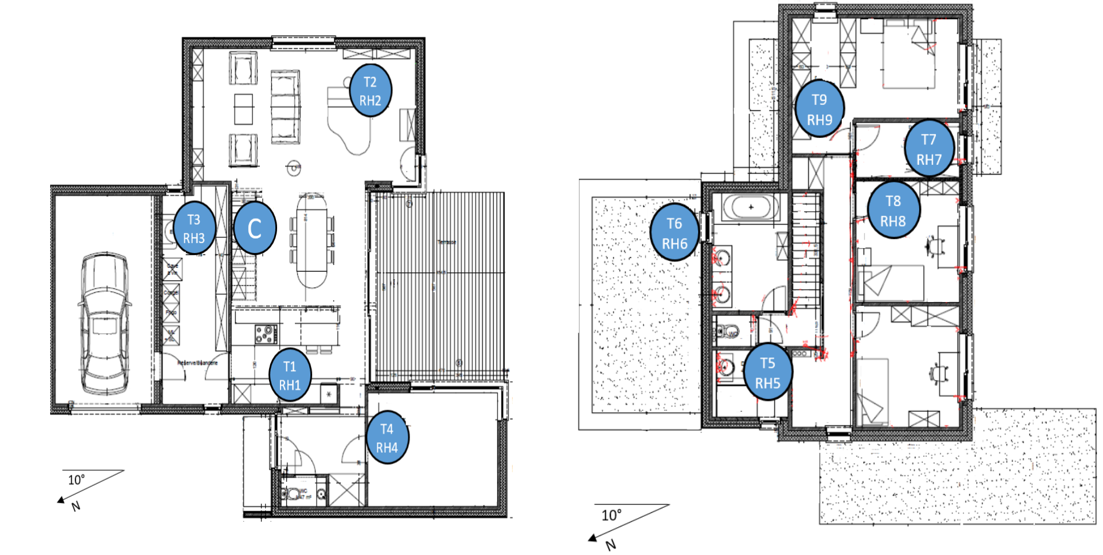
A close up of a window

Description automatically generated

**Figure 2.1: The snapshot includes the entire values of predictors and response variable from the raw dataset.**

**2.2 Floor Dimensions**

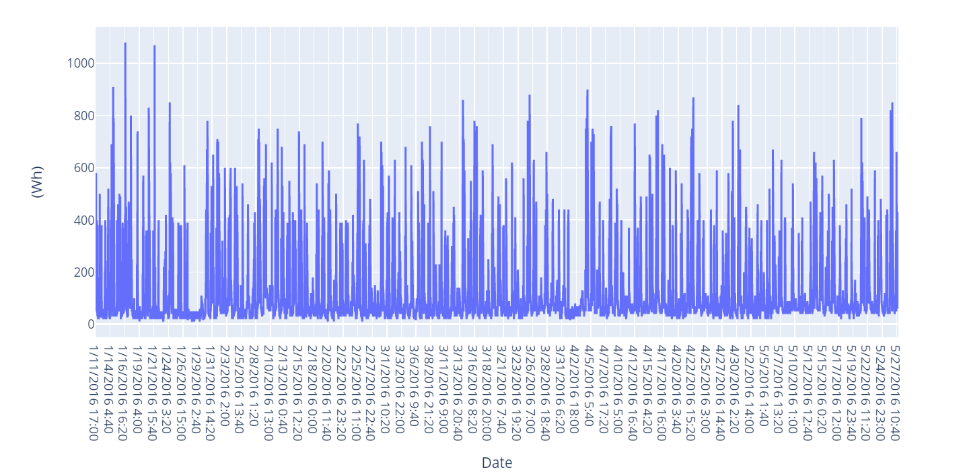
Figure 2.2 depicts the dimensions of the building and also where the sensors are placed and in which areas to record the temperature and humidity readings for even 10 minutes interval. The total building which is selected for this experiment is about 1,196 square foot. There are sensors which are also placed outside the building as there is an airport nearby the building and the sensors outside the building records the visibility, tdewpoint, Windspeed, temperature, humidity and pressure outside the building which are also considered as crucial to conduct this predictive analytics.

****

**Figure 2.2: Showing the floor dimensions of the building on which the sensor recordings are taken.**

**3.0 Pre-Processing:**

Our Pre-Processing steps included identifying the missing values and imputing them with the centralized measures and have removed the unnecessary data as our dataset has the time series one where the first predictor is the date with a recording for every 10 minutes. Few of the recording was missing as there were some technical issues while recording and the values aren’t recorded properly, and those recordings were skipped. Such cells are considered to be removed and that’s one of the reasons why we haven’t considered the date column for further analysis and also our main aim and goal is to predict the energy consumed by the appliances. The below picture helps in knowing the time series representation of the entire dataset.

**  
 Figure 3.1: Time series representation of the raw dataset.**

**3.1 Unnormalized data**

Detecting the unnormalized data using the histogram density plot to identify the unnormalized distributions and used several pre-processing methods like center and scaling which helped to normalize the data and deal with the outlier values. As we have skewness to the response variable, we split the entire dataset into 2 parts and dealt with them separately to make the models work efficiently. In the below figure we can clearly identify few predictor variables whose distributions isn’t normal but unnormalized with irregular distributions like RH\_6, Visibility, RH\_out, Windspeed. These variables are to be further normalized and the relation between them can be identified using the correlation plot.

A close up of a logo

Description automatically generated

**Figure 3.1.1: Histogram distribution showing the normalized and unnormalized data.**

**3.2 Feature Selection**

Feature selection algorithms were used to identify and select the features which are significant. The feature selection and scaling algorithms like Boruta importance and relative importance are implemented to check the significance of the predictor variable which can help to identify the importance of each predictor variable in the process of analysis.

In machine learning and statistics, feature selection, also known as variable selection, attribute selection or variable subset selection, is the process of selecting a subset of relevant features(variables, predictors) for use in model construction. Feature selection techniques are used for simplification of models to make them easier to interpret by researchers/users, shorter training times, to avoid the curse of dimensionality, enhanced generalization by reducing overfitting (formally, reduction of variance). We have worked on Boruta Algorithm for feature selection in our dataset.

* + 1. **The Boruta Algorithm:**

The Boruta algorithm is a wrapper built around the random forest classification algorithm. It tries to capture all the important, interesting features you might have in your dataset with respect to an outcome variable. It works well for both classification and regression problem.

A screenshot of a cell phone

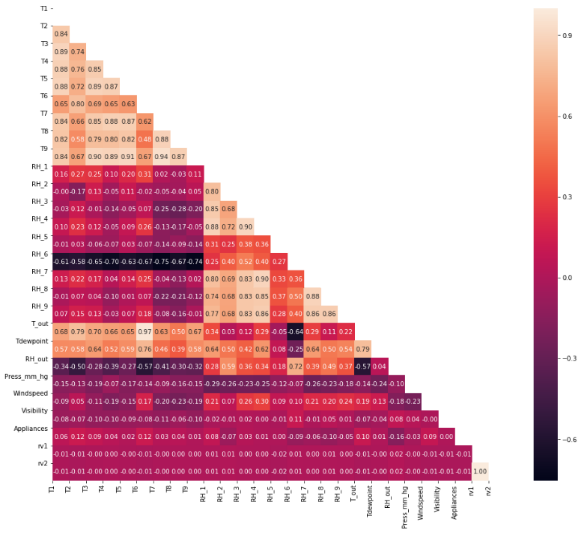
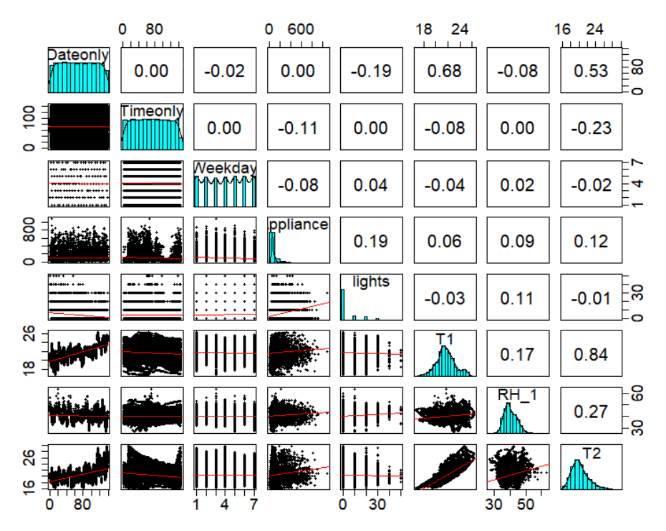
Description automatically generated

**Figure 3.2.1: Identifying the important variables using the Boruta variable importance methods.**

The Boruta Algorithm includes plot for all the predictor variables against the response variable. As we see, lights, Pressure, Humidity in Bathroom and Parent’s Room are the most important predictor variables.

**3.3 Correlation – Remove Highly correlated variables**

We have taken additional steps to normalize the data where we have removed the highly correlated variables with a cut off at 0.9 and 0.8 varying to get the optimized results. We found the predictor T9 to be highly correlated than compared to the rest of the predictors and the predictor T9 is removed from the dataset. The predictor variable T9 is highly correlating with all the predictor variables relatively highly and had a huge effect on the response variable than compared to others. The other predictors like visibility and wind speed are also removed as they are also highly influencing the other predictor variables and they are also considered to remove from the dataset on which the analysis must be performed.

 ****

**Figure 3.3.1: Correlation plots helping to identify the highly correlated variables**

In the above figure 3.3.1, the first plot shows the relation between the predictor variales and got the variables from the upper traingles whose values are greater than 0.8 and 0.9 and they are removed. Out of all the predictor variables the variables like T9 which is the temperature in the parent room and the To temperature outside the building seems to be highly correlated with every other variable in the dataset and thus they are removed.

**4.0 Modeling Techniques**

**4.1 Gradient Boosting Methods**

Boosting is one of the best ensemble methods where each new tree is fitted on a modified version of the original data set. This gradient boosting method very much helped to improve the model efficiency than compared to other models and ensemble models are advantageous in this aspect than compared to other models as they combine different models and retrieves the best values out of it.

Once the values are retrieved to tune the model finely K-fold cross validation is used which is helpful to tune the models across different tuning parameters with shrinkage values 0.01 and 0.10 and then with max tree depth of 1 and 5 the optimum values are retrieved. The optimal solution is found for number of trees = 500, learning rate = 0.10, interaction depth = 5 with least RMSE (**0.0041**) and high Rsquared (**0.998**).

A close up of a map

Description automatically generated

**Figure 4.1: Showing the Plot which is plotted with the tuning parameters and RMSE on Y axis.**

GBM plot includes boosting iterations on X-axis and RMSE on Y-axis. As number of trees increase, RMSE drastically drops from 0.06 to 0.01 for learning rate 0.01 when tree depth is 1. As number of trees increase, RMSE gradually drops from 0.02 to 0.01(approx.) for learning rate 0.10 when tree depth is 5. Using the variable importance, GBM predicts that **Pressure as highly important** and **Temperature Outside as a least important predictor.**

**4.2 Extra Tree Regressor**

Extra Tree Regressor can be very well explained when compared with the random forest. This is also one of the ensemble models we have implemented in this project. In random forest, we chose to sample with a replacement for bootstrap datasets and consider best split for each iteration whereas in extra tree regressor it takes the samples without replacement where duplications aren’t allowed, and the random split is considered for each iteration of the subset of the dataset.

Fine tuning is applied to the models after the initial values are obtained. Randomized CV and grid search CV are used to tune this model with different tuning parameters with the estimators ranging from 200 – 2000 and the depth from 80 – 250 in the parameter grid. Grid search evaluates all possible combinations of the parameter values where it combines the hyperparameter values of a model that gives us the best values. Grid search CV obtained the best values when tuned the model which is the best values of prediction compared to remaining models with the best RMSE and R squared values.

A close up of a piece of paper

Description automatically generated

**Figure 4.2: Plot for Extra tree regressor showing a branch with depth of 4 and 10 estimators**

The obtained RMSE and R-Squared values we received on our dataset with this Extra Tree Regressor are **0.004** and **0.999**for tuning parameters number of estimators at 250 and maximum depth of tree at 80. Using the variable importance, Extra Tree Regressor predicts that **Humidity Outside as highly important** and **Temperature in Kitchen Area as a least important predictor.**

**4.3 Random Forest**

Random forest helps in reducing the overfitting by bootstrap aggregation and significantly lowering the risk of overfitting. Also, it has it traces in reducing the variance whereby using the multiple trees, the chance of stumbling across the classifier is reduced.

A screenshot of a cell phone

Description automatically generated**A close up of a map

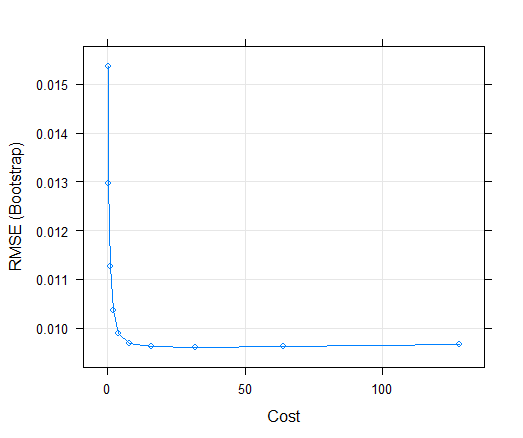
Description automatically generated  
Figure 4.3: Showing the plot for random forest with the trees on X axis and the error rate on y axis and also the random forest tree.**

The tuning parameter for Random Forest is the number of trees. The Random Forest plot is plotted against on number of trees on X-axis and error rate on Y-axis. The trees vary from 0 to 500. The error rate decreases as the no. of trees increase until it reaches its optimal value and remains constant as trees are increased. The tuning to the model is done by giving the estimators and or trees and mtry values.

The obtained RMSE and R-Squared values we received on our dataset with Random Forest are **0.006** and **0.996**for tuning parameters number of trees = 500 and mtry value = 2.

**4.4 Support Vector Machine - SVM**

SVM is a supervised machine learning algorithm that can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs. The kernel trick takes the data you give it and transforms it. In goes some great features which you think are going to make a great classifier, and out comes some data that you don't recognize anymore.

**  
Figure 4.4: Showing the SVM plot with the tuning parameter cost on X axis and the error rate on Y axis.**

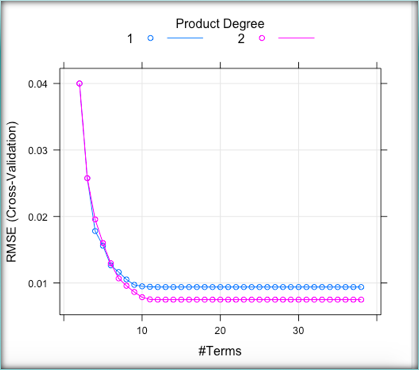
The tuning parameter for the Support Vector Machine is the Cost Function. The SVM model plot is shown below, the cost function varies from 0 to 128. The plot shows that the RMSE value decreases and is almost constant from C=4. The obtained RMSE and R-Squared values we received on our dataset with this SVM model are **0.0096** and **0.993**for the optimal value at **C=32.**

Using the variable importance, SVM predicts that **Pressure as highly important** and **T\_out as the least important predictor.** SVM works relatively well when there is a clear margin of separation between classes. Also, SVM is effective in cases where the number of dimensions is greater than the number of samples. The main disadvantage of this model is, the SVM model is not suitable for large datasets as it takes a longer time to run and does not perform very well when the dataset has more noise. This is one of the reasons we couldn’t tune the model and get optimized values apart from the initial values.

**4.5 Multivariate Adaptive Regression Splines (MARS)**

Multivariate Adaptive Regression Splines (MARS) is a non-parametric regression method that models multiple nonlinearities and analyzes the data point as a knot for every feature and creates a linear regression model with the feature(s) of the candidate. Therefore, once the broad set of knots has been recognized, we can sequentially delete knots that do not contribute significantly to predictive accuracy. This process is called "pruning" (Boehmke, n.d.). We can fit a direct engine MARS model with the **earth** package. There are two essential tuning parameters connected with our MARS model: **the maximum degree of interactions** and **the number of terms retained in the final model.** The obtained RMSE and R-Squared values we received on our dataset with this model are **0.007** and **0.995(98%)** for the optimal value at **nprune = 12** and **degree =2,** these are the values obtained after tuning the model with repeated CV and other cross validation methods also including the grid parameters**.**

Using the variable importance, MARS predicts that **Pressure as highly important** and **RH\_5 as a least important predictor.** There are numerous benefits to MARS.First, MARS naturally handles both quantitative and qualitative types of predictors. MARS requires minimal scaling of features as well as an automated selection of features.

****

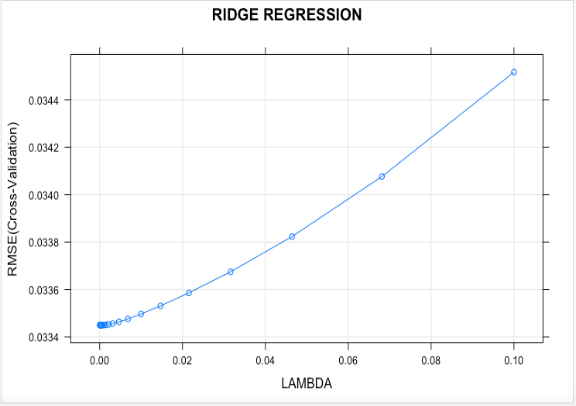
**Figure 4.5: Showing the plot of tuning parameters for MARS model.**

The plot shows us the optimal model which retains 38 terms and includes up to two-degree interactions. Number of terms is plotted on X-axis and RMSE on Y-axis.

**4.6 Ridge Regression**

Ridge Regression is a multi-regression data analysis technique where Most square estimates are objective when multicollinearity occurs, but their variances are huge so they may be distant from the true value. The ridge regression excludes standard errors by applying a degree of bias to the regression estimates. Ridge regression is a parsimonious structure that controls L2 ( realdataweb, 2017).

The L2 regularization concerns an equivalent penalty to the square of the regression coefficients and aims to minimize them. The strength of the penalty term in ridge regression is controlled by a **tuning parameter (λ)**, called as penalty parameter. It is essentially the sum of shrinkage in which data values are reduced to a central point, such as the mean. The obtained RMSE and R-Squared values we received on our dataset with this model are **0.0342** and **0.900 ( 81%)** for the optimal value at **λ = 0.0004** these values are retrieved after performing the tuning to the model with the K-fold CV**.**



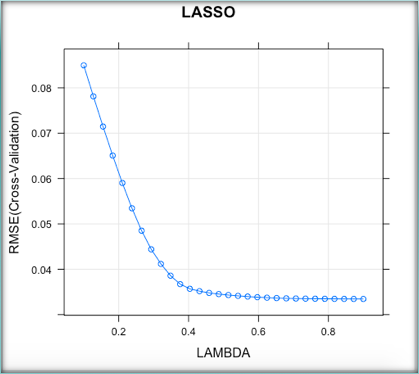
**Figure 4.6: Showing the plot of tuning parameters for Ridge model.**

The plot below shows us the optimal model which retains **20 lambda values** with the increasing RMSE values as the lambda value increases. Using the variable importance, RIDGE predicts that **Pressure as highly important** and **T\_out as the least important predictor.** A major advantage of ridge regression is that it still works well in a situation where you have large multivariate data with the number of predictors (p) greater than the number of observations (n) relative to the ordinary least square process.

**4.7 Lasso Regression**

Lasso regression is a parsimonious model that controls L1. The L1 regularization includes and purposes to minimize a penalty equal to the absolute magnitude of the coefficients of regression. The obtained RMSE and R-Squared values we received on our dataset with this model are **0.0342** and **0.9007 (81%)** for the optimal value at **λ = 0.9** these are the values obtained when tuning is done to the model after running the basic model, fine tuning is done to the model specifying the tune length and using the k-fold CV method which evaluates with different test and train dataset**.**

The plot below shows us the optimal model which retains **30 lambda values** with the decreasing RMSE values as the lambda value increases. Using the variable importance, Lasso predicts that **Pressure as highly important** and **T\_out as a least important predictor.** It provides a very suitable prediction accuracy, because shrinking and eliminating the coefficients can reduce variance without a substantial increase of the bias, this is especially useful when you have a small number of observations and many features. As far as the tuning parameter π is concerned, we know that bias increases and variance decreases as π increases, a trade-off between bias and variance is indeed to be found (Oleszak, 2019).

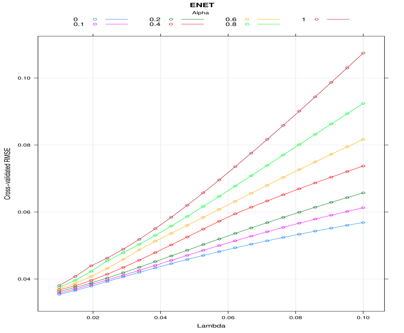


**Figure 4.7: Showing the plot of tuning parameters for Lasso model.**

**4.8 Elastic Net Regularization**

Elastic net regularization method includes both LASSO (L1) and Ridge (L2) regularization methods. Also, elastic net is computationally more expensive than LASSO or ridge as the relative weight of LASSO versus ridge must be selected using cross validation. If a reasonable grid of alpha values is [0,1] with a step size of 0.1, that would mean elastic net is roughly 11 times as computationally expensive as LASSO or ridge. (Since LASSO and ridge do not have quite the same computational complexity, the result is just a rough guess) (Arunava, n.d.).

The obtained RMSE and R-Squared values we received on our dataset with this model are **0.0361** and **0.893 (79%)** for the optimal value at **λ = 0.01 and alpha = 0.**

****

**Figure 4.8: Showing the plot of tuning parameters for ENET model.**

The plot shows us the optimal model which retains **125 tuning values for 5 interactions** with the increasing RMSE values as the lambda value increases.

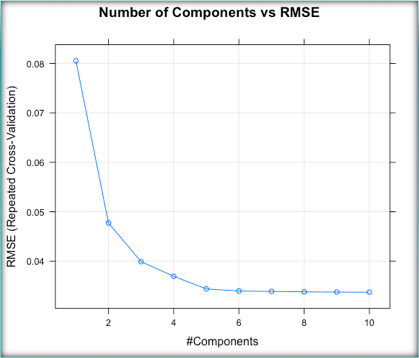
Tuning is performed using the k-fold CV by defining the grid parameters with lambda and alpha values. Using the variable importance, ENET predicts that **Pressure as highly important** and **Humidity in Bathroom as a least important predictor.**

**4.9 Partial Least Squares Regression**

The PLS family of methods is known as bilinear variable models because both the X and Y data are projected into new spaces. Discriminant Analysis (PLS-DA) partial least squares is a form used when the Y is categorical. The predictors we use in many data sets may correlate with the answer and with each other. If too many variables of predictors are associated, then the variance will make the regression unstable (Kassambara, 2018).

One way to correct predictor correlation is to use PCA or Principle Component Analysis to find a linear predictor combination to obtain the most variance. Unlike PCA, PLS regression seeks to find components that optimize predictor variance but vary from PCA because PLS allows the components to have full response correlation. PLS is supervised, while PCA is unmonitored.

The obtained RMSE and R-Squared values we received on our dataset with this model are **0.0337** and **0.907 ( 81%)** for the optimal value at **ncomp=10** obtained using the repeated CV method in the process of tuning and also by specifying the length of tuning**.**

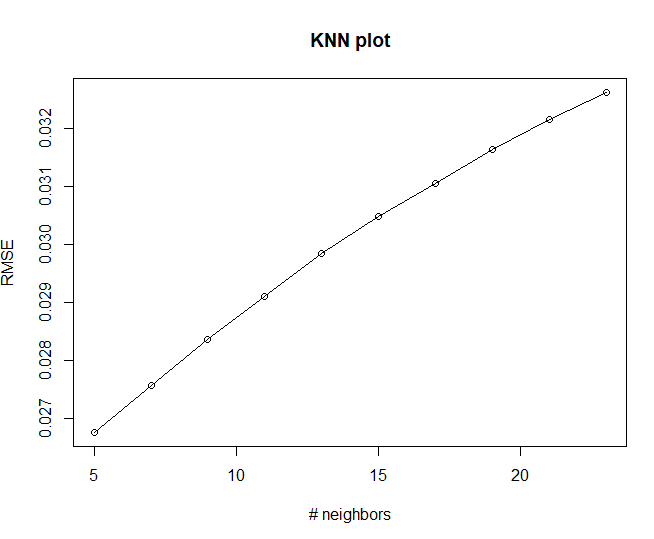


**Figure 4.9: Showing the plot of tuning parameters for PLS model.**

The tuning parameter for PLS regression is number of components. Number of components are plotted on X-axis and RMSE on Y-axis. The plot shows us the optimal model which retains **10 tuning values** with the decreasing RMSE values as the ncomp value increases. Using the variable importance, RIDGE predicts that **Pressure as highly important** and **Wind Speed as a least important predictor.**

**4.10 K- Nearest Neighbor**

The k-Nearest Neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. The tuning parameter is the resulting optimal K value along with the 10-fold cross validation method. The result contains the change of tuning parameter values ranging from 5 to 23 along with their RMSE, Rsquared, RMSESD, Rsquared SD values. The obtained RMSE and R-Squared values we received on our dataset with K – Nearest Neighbor are 0.02671164 and 0.9427125 for tuning parameters K = 5.

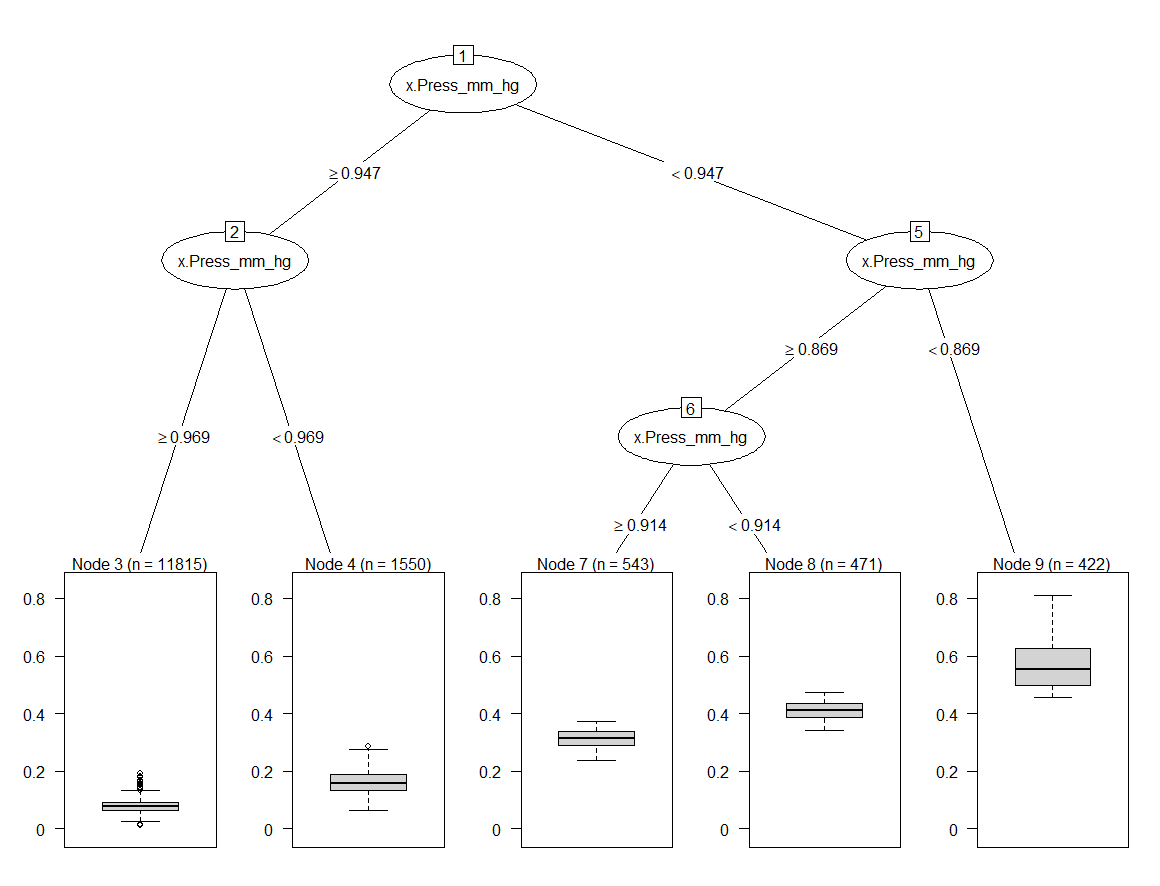


**Figure 4.10: Showing the plot of tuning parameters for KNN model.**

For KNN, number of neighbors is on X-axis and RMSE on Y-axis. We can see that RMSE value increase as number of neighbors increase.

**4.11 Decision Tree**

Decision trees have a natural “if … then … else …” construction that makes it fit easily into a programmatic structure. To identify the variable and the split, decision tree uses various algorithms. The tuning parameters are the maximum depth of the tree and the complexity parameter which is used to select the size of the tree. The obtained R-Squared and RMSE values are 0.908 and 0.032 for the tuning parameters cp = 0.01 and maximum depth = 30.



**Figure 4.11: Showing the plot of tuning parameters for Decision Tree model.**

In the decision tree plot we can see that Pressure outside the house is taken as one of the important variables.

**5.0 Future Work**

Future work includes adding more predictive parameters such as occupancy information, the area of the house, day to day activities performed by the occupant, more indoor and outdoor environmental conditions, by using different datasets can also improve the prediction. Training other machine models such as ANN (Artificial Neural Network) can further boost the predictive capacity, energy consumption is a vast domain and has scope in future to implement the hierarchical and hybrid models to predict the appliances energy on a larger scale.

**6.0 Conclusion**

**A screenshot of a cell phone

Description automatically generated**

**Fig 6.1 Bar graph showing the RMSE, Rsquared values of the predictive models.**

The bar graph above is depicting the RMSE, Rsquared values of all the predictive models implemented in this project to predict the energy utilization of the Appliances, we see that on the test set, Support Vector Machine, Mars, Random Forest, Gradient Boosting Method, Extra Tree Regressor have highest R-squared values. Least RMSE values are shown for all the ensemble methods like Extra Tree Regressor, Gradient Boosting Method and Random Forest. Out of all the methods including the ensemble methods, Extra tree regressor has the least RMSE value and the highest R2 value which is considered to be the best model for this dataset and the application of predicting the energy usage of the appliances.

The optimal predictive model for Predicting Appliances Energy Consumption is Extra Tree Regressor with RMSE (**0.004**) and Rsquared (**0.999**).

**REFERENCES**

[1] Realdataweb. (2017, May 23). *Ridge Regression and the Lasso*. Retrieved from RBlogggers: <https://www.r-bloggers.com/ridge-regression-and-the-lasso/>.

[2] Arunava. (n.d.). *ENet — A Deep Neural Architecture for Real-Time Semantic Segmentation*. Retrieved from Towards Data Science: <https://towardsdatascience.com/enet-a-deep-neural-architecture-for-real-time-semantic-segmentation-2baa59cf97e9>.

[3] Boehmke, B. (n.d.). *Multivariate Adaptive Regression Splines*. Retrieved from <https://bradleyboehmke.github.io/HOML/mars.html#fitting-a-basic-mars-model>.

[4] Kassambara. (2018, November 3). *Statistical tools for high-throughput data analysis*. Retrieved from STDHA: <http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/152-principal-component-and-partial-least-squares-regression-essentials/>.

[5] Oleszak, M. (2019, November 12). *Regularization: Ridge, Lasso and Elastic Net*. Retrieved from DataCamp: <https://www.datacamp.com/community/tutorials/tutorial-ridge-lasso-elastic-net>.

[6] Xu, P. J. (2019, November 25). Classification Trees and Rule-based Models. Retrieved from <https://mymasonportal.gmu.edu/webapps/blackboard/content/listContent.jsp?course_id=_375504_1&content_id=_9047730_1&mode=reset>.

**7.0 Appendix:**

**7.1 Gradient Boosting Method:**

A screenshot of a cell phone

Description automatically generated

**Figure 7.1: Showing the results of Gradient Boosting Model**

The image shows the results of Gradient Boosting Model on train set that includes RMSE, Rsquared, Rsquared, RMSESD values against tuning parameters, learning rate(0.01, 0.10), interaction depth(1,5) and number of trees(100,500).

A screenshot of a cell phone

Description automatically generated

**Figure 7.1.1: Showing Variable Importance and RMSE, Rsquared values on test set.**

The image shows the variable importance by Gradient Boosting Model and predicted RMSE, Rsquared values on test set.

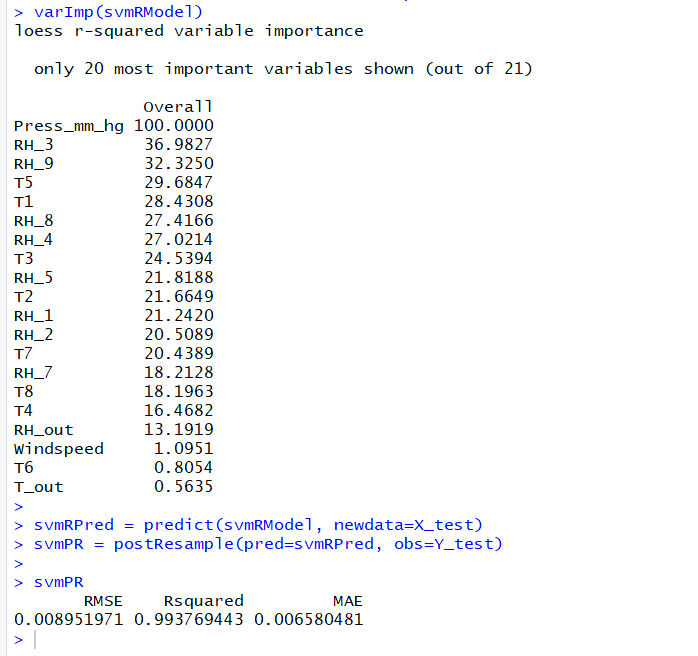
**7.2 Support Vector Machines:**

**A close up of text on a black background

Description automatically generated**

**Figure 7.2: Showing the results of Support Vector Regression**

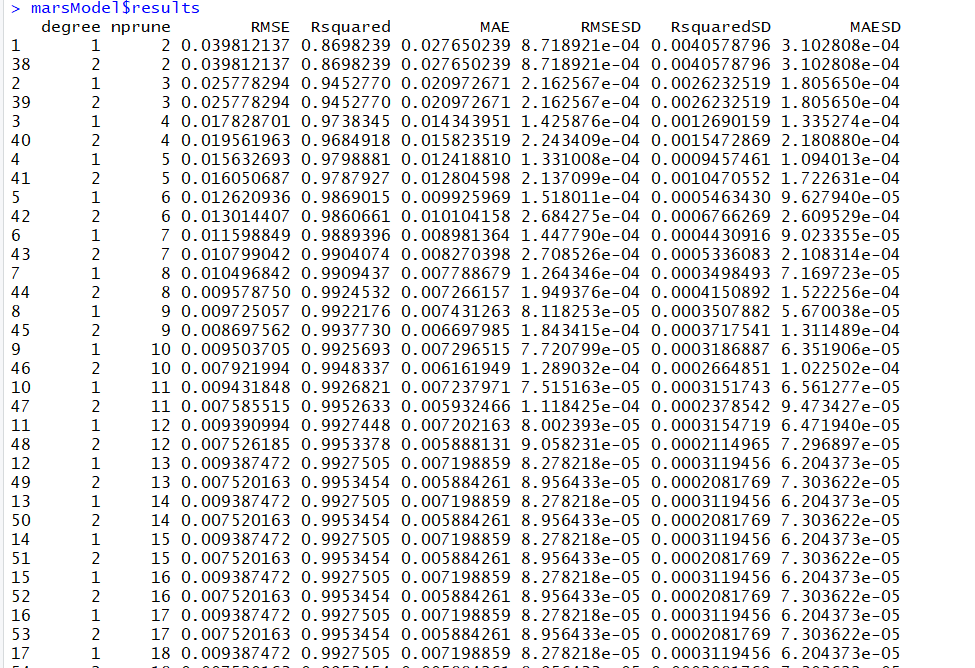
The image shows the results of Support Vector Regression on train set that includes RMSE, Rsquared, Rsquared, RMSESD values against tuning parameters, cost function(0.25, 0.50 1.00….128).



**Figure 7.2.1: Showing Variable Importance and RMSE, Rsquared values on test set.**

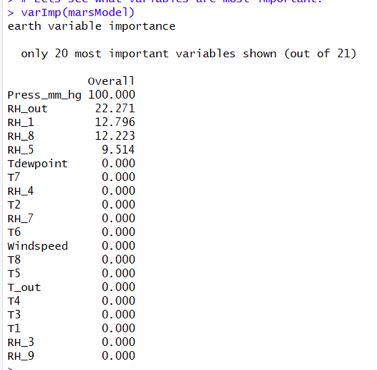
The image shows the variable importance by Support Vector Regression and predicted RMSE, Rsquared values on test set.

**7.3 MARS:**



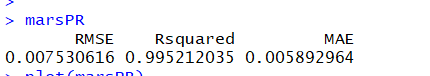
**Figure 7.3: Showing the results of MARS model**

The image shows the results of MARS model on train set that includes RMSE, Rsquared, Rsquared, RMSESD values against tuning parameters, degree (1,2) and nprune (2,30).

****

**Figure 7.3.1: Showing Variable Importance for MARS model**

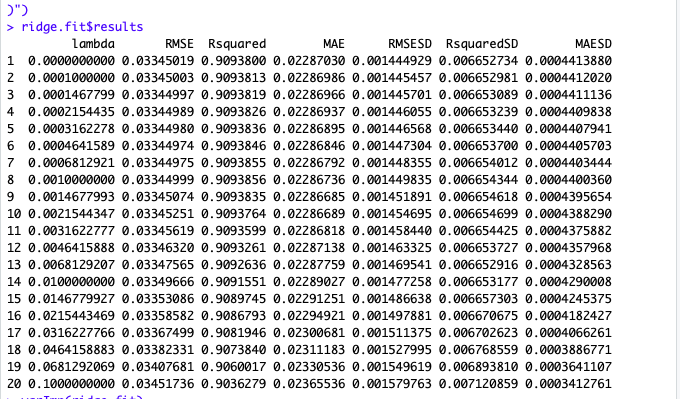
The image shows the variable importance by MARS model and predicted RMSE, Rsquared values on test set.



**Figure 7.3.2: Showing RMSE, Rsquared on test set.**

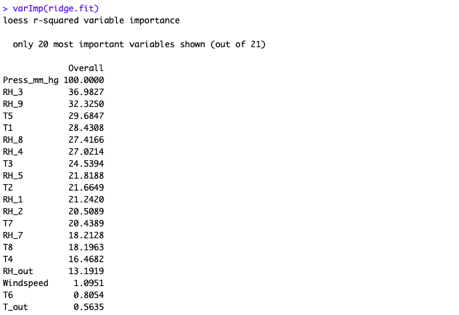
The image shows the RMSE, Rsqaured by MARS model on test set.

**7.4 RIDGE:**

****

**Figure 7.4: Showing the results of Ridge Regression**

The image shows the results of Ridge Regression on train set that includes RMSE, Rsquared, Rsquared, RMSESD values against tuning parameters, lambda (0,0.1)



**Figure 7.4.1: Showing Variable Importance for Ridge model**

The image shows the variable importance by Ridge model.

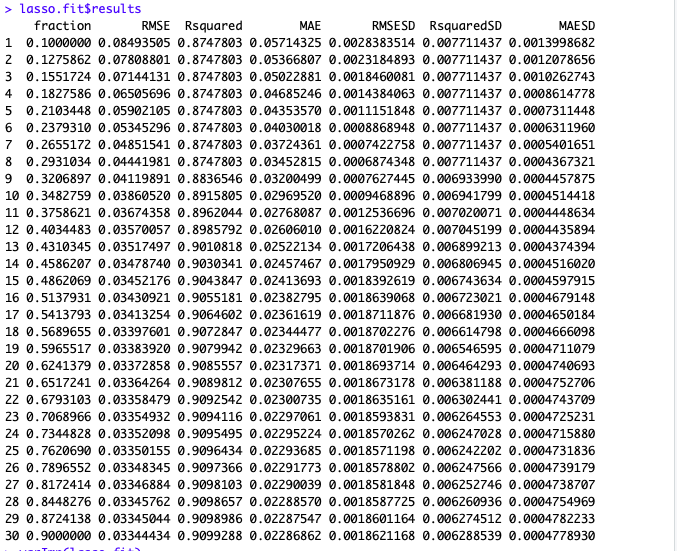
A screenshot of a cell phone

Description automatically generated

**Figure 7.4.2: Showing RMSE, Rsquared on test set.**

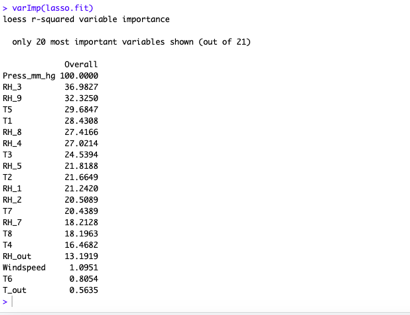
The image shows the RMSE, Rsqaured by Ridge model on test set.

**7.5 LASSO:**

****

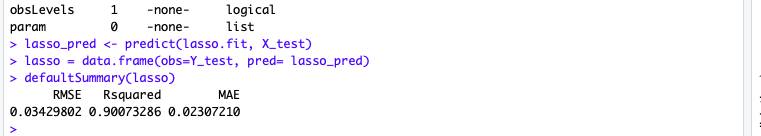
**Figure 7.5: Showing the results of Lasso Regression**

The image shows the results of Lasso Regression on train set that includes RMSE, Rsquared, Rsquared, RMSESD values against tuning parameters, lambda (0,0.9)



**Figure 7.5.1: Showing Variable Importance for Lasso model**

The image shows the variable importance by Lasso model.



**Figure 7.5.2: Showing RMSE, Rsquared on test set.**

The image shows the RMSE, Rsqaured by Lasso model on test set.

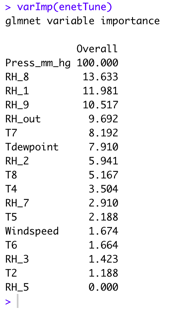
**7.6 ENET:**

A picture containing window, large, white

Description automatically generated

**Figure 7.6: Showing the results of ENET Regression**

The image shows the results of ENET Regression on train set that includes RMSE, Rsquared, Rsquared, RMSESD values against tuning parameters, alpha(0,1) and lambda(0.01, 0.02).

****

**Figure 7.6.1: Showing Variable Importance for ENET model**

The image shows the variable importance by ENET model.

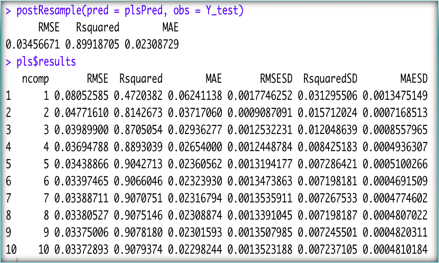
A screenshot of a cell phone

Description automatically generated

**Figure 7.6.2: Showing RMSE, Rsquared on test set.**

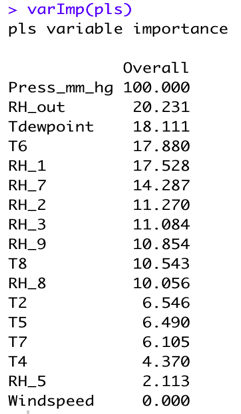
The image shows the RMSE, Rsqaured by ENET model on test set.

**7.7 PLS:**

****

**Figure 7.7: Showing the results of PLS Regression and RMSE, Rsquared on test set.**

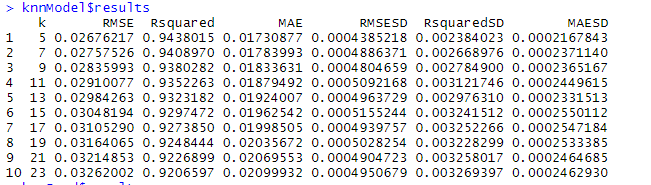
The image shows the results of PLS Regression on train set that includes RMSE, Rsquared, Rsquared, RMSESD values against tuning parameters, number of components, ncomp(1,10).



**Figure 7.7.1: Showing Variable Importance for PLS model**

The image shows the variable importance by PLS model.

**7.8 KNN:**



**Figure 7.8: Showing the results of K-Nearest Neighbors.**

The image shows the results of K-Nearest Neighbors on train set that includes RMSE, Rsquared, Rsquared, RMSESD values against tuning parameter, K, number of neighbors.



**Figure 7.8.1: Showing RMSE, Rsquared on test set.**

The image shows the RMSE, Rsqaured by K-Nearest Neighborson test set.

A close up of a map

Description automatically generated

**Figure 7.8.2: Showing the plot of tuning parameters for KNN model.**

For KNN, number of neighbors is on X-axis and RMSE on Y-axis. We can see that RMSE value increase as number of neighbors increase.

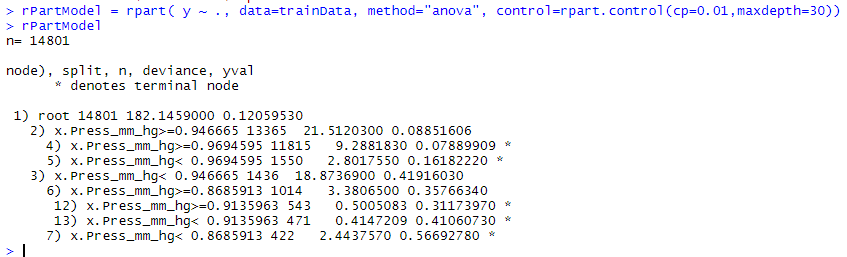
A screenshot of a cell phone

Description automatically generated

**Figure 7.8.3: Showing the results of K-Nearest Neighbors.**

The image shows the results of K-Nearest Neighbors on train set that includes RMSE, Rsquared, values against tuning parameter, K, number of neighbors.

**7.9 Decision tree:**



**Figure 7.9: Showing the results of Decision Trees.**

The image shows the results of Decision Trees on train against tuning parameters, cost complexity parameter (0.01) and maximum node depth(30).



**Figure 7.9.1: Showing RMSE, Rsquared on test set.**

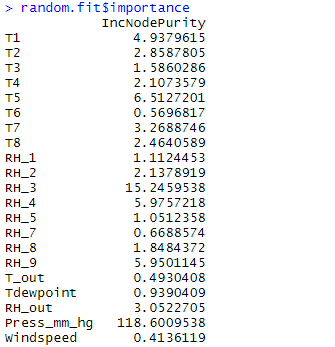
The image shows the RMSE, Rsqaured by Decision Treeson test set.

**7.10 Random Forest:**



**Figure 7.10: Showing RMSE, Rsquared on test set.**

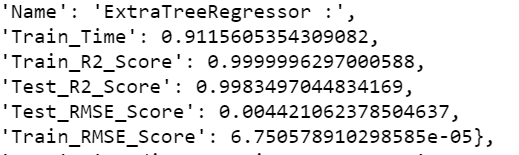
The image shows the RMSE, Rsqaured by Decision Treeson test set.



**Figure 7.7.1: Showing Variable Importance for Random Forest**

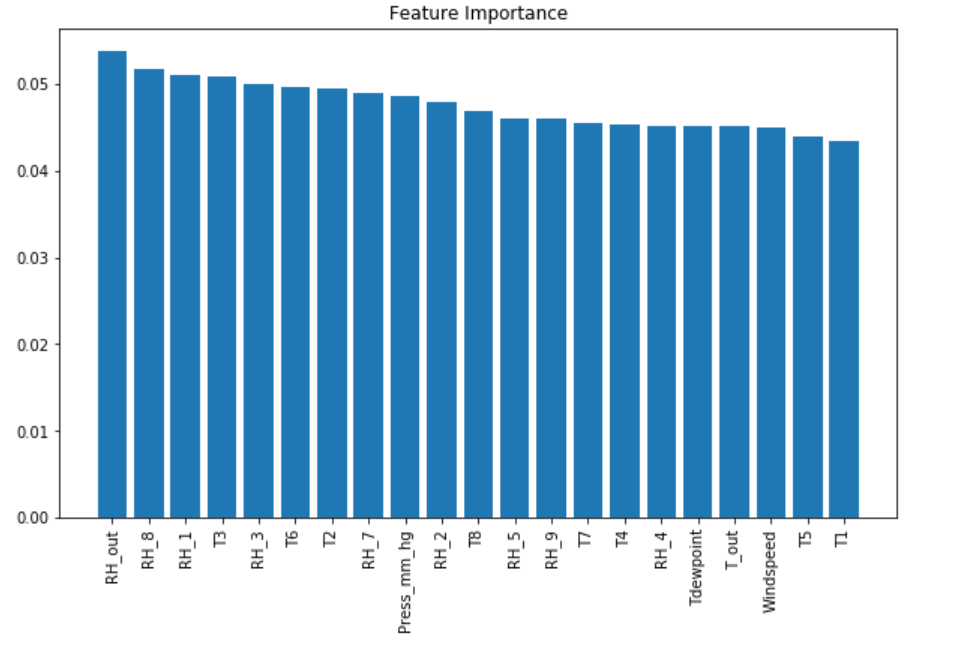
The image shows the variable importance by Random Forest.

**7.11 Extra Tree Regressor:**

****

**Figure 7.11: Showing RMSE, Rsquared on test and train set.**

The image shows the RMSE, Rsqaured by Extra Tree Regressoron test and train set.

****

**Figure 7.11.1: Showing Variable Importance for Extra Tree Regressor**

The image shows the variable importance by Extra Tree Regressor.

**8.0 How to run code:**

**Step 1:** The raw data is included in the data folder within the zip folder and link to the dataset is also provided in the main body of document under dataset section.

**Step 2:** The python code is attached where it includes all the pre-processing steps, normalizing and standardizing and removing the highly correlated variables etc.

**Step 3:** After running the raw dataset through the python code first we have saved the data which is pre-processed and splitted as train\_x, train\_y, test\_x and test\_y in the local directory.

**Step 4:** These 4 datasets where train\_x and test\_x contains the data of the predictors where as train\_y and test\_y contains the data regarding the response variable. This data is shared among the other people in the team to run their models on.

**Step 5:** The datasets which ae pre-processed and splitted, which are used to run the R code is also included in the Zip file with the same names.

**Step 6:** After loading those 4 datasets which are extracted from the raw dataset after pre-processing and normalizing into the R studio then please run the rest of the models.

**Step 7:** Most of the models took so much of time to run them. We have also tuned the models using random search CV, grid search CV and K fold CV and repeated CV using number of repeats.