Assignment 2 Report~ Team1

1. Data Cleansing:

i. Regression model with the "raw dataset" including zeros

Building a Regression model with the "raw dataset" including zeros –

A linear regression model was implemented with the following attributes: Peak hour, Temperature, Weekday, Day of Week, Hour, Day and Month.

On fitting the linear regression model, the measures of predictive accuracy are as follows:

	ME	RMSE	MAE	MPE	MAPE
Test set	-1.563999	116.31718	88.09954472	NaN	Inf

ADJUSTED R SQUARE= 0.4075647

ii. Regression model with the "non zero" dataset

Building a Regression model with the "raw dataset" excluding zeros by cleansing the data in Python–

A linear regression model was implemented in R with the following attributes: Peak hour, Temperature, Weekday, Day of Week, Hour, Day and Month.

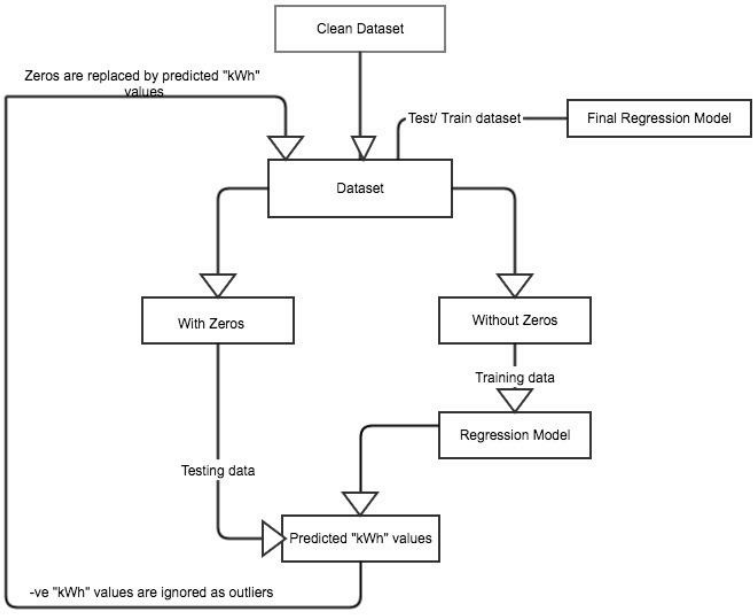
On fitting the linear regression model, the measures of predictive accuracy are as follows:

	ME	RMSE	MAE	MPE	MAPE
Test set	-4.376781	119.477846	92.09790606	-6609.404	6648.9947

ADJUSTED R SQUARE= 0.3484754

iii. Regression model with replaced zeros with predicted values

Building a Regression model with the “raw dataset” excluding zeros with predicted values. The regression model is first trained on a dataset with doesn’t have any zeros then the part of the dataset with have zeros are predicted and finally the regression model is run on the whole dataset with had previously “kWh” values and the also the predicted “kWh”



On fitting the linear regression model, the measures of predictive accuracy are as follows:

	ME	RMSE	MAE	MPE	MAPE
Test set	-0.281915	108.027354	79.95937254	-4195.617	4243.9364

ADJUSTED R SQUARE= 0.4191468

iv. Regression model with replaced zeros by zoo package

Using "**locf function**" of the zoo package, first replaced the zeros in the test data with "NA" and then further used the na.locf function to replace the "NA" values of kWh.

On fitting the linear regression model, the measures of predictive accuracy are as follows:

	ME	RMSE	MAE	MPE	MAPE
Test set	-1.570623	113.21921	85.15383016	-39107.76	41102.4972

ADJUSTED R SQUARE= 0.3973992

Using "**approx. function**" of the zoo package, first replaced the zeros in the test data with "NA" and then further used the na.approx function to replace the "NA" values of kWh.

On fitting the linear regression model, the measures of predictive accuracy are as follows:

	ME	RMSE	MAE	MPE	MAPE
Test set	-1.183909	111.65126	83.69516355	-5257.033	5482.67805

ADJUSTED R SQUARE= 0.3909442

Using "**fill.function**" of the zoo package, first replaced the zeros in the test data with "NA" and then further used the na.approx function to replace the "NA" values of kWh and then use na.fill function to replaced the trailing NA values.

On fitting the linear regression model, the measures of predictive accuracy are as follows:

	ME	RMSE	MAE	MPE	MAPE
Test set	-1.579120	111.67315	83.7589262	-6356.035	6756.73950

ADJUSTED R SQUARE= 0.3917891

The technique that works the best is using predicted values of kWh. This technique uses the dataset without zeros to train the model and then further predicts values for the zero values. Finally, the dataset with new predicted values of kWh and the non-zero values of kWh are used to create the final regression model. This model gives around 300 rows of negative kWh values, we have decided to ignore the negatives values as treating them as outliers as kWh can never be negative and 300 rows is around 3.8% of the whole dataset.

The adjusted R square value = 0.4075647, which is the highest in all the methods.

2. Prediction & Forecast:

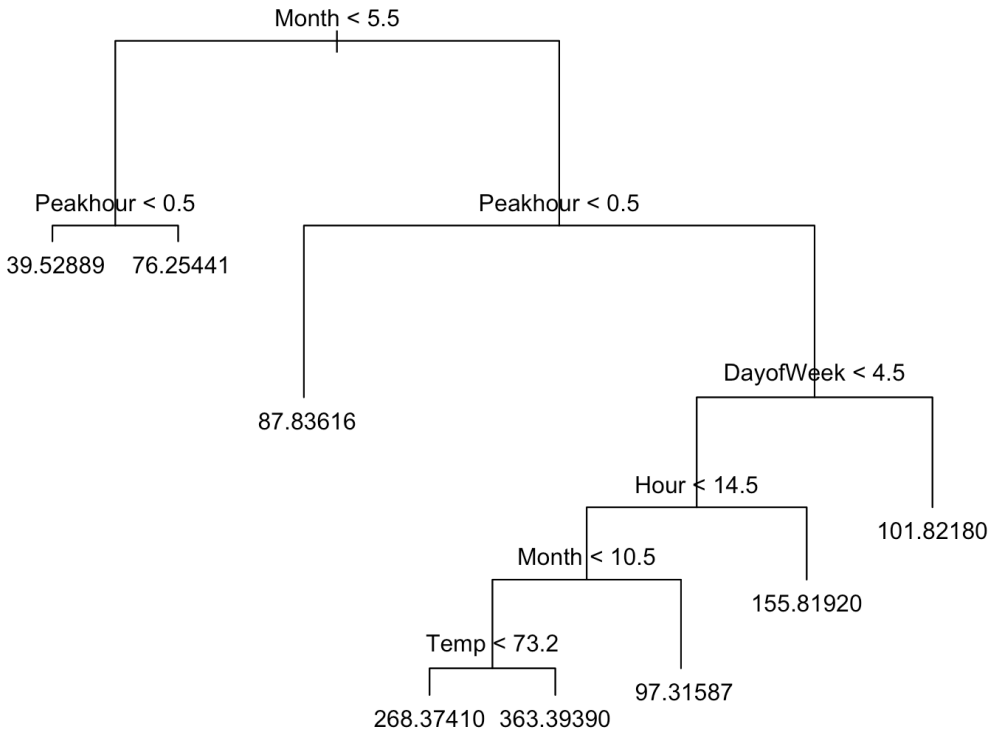
i. Regression trees-

Fitting the Regression Tree Model on to the cleansed dataset without zeros or negative "kWh" values.

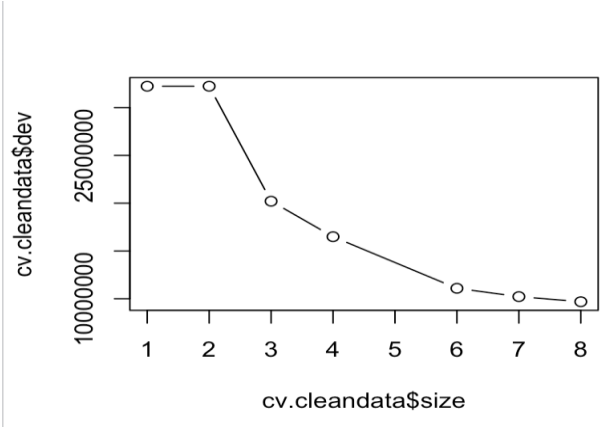
```
> summary(tree.cleandata)

Regression tree:
tree(formula = kWh ~ Peakhour + Temp + Weekday + DayofWeek +
    Hour + Day + Month, data = cleandata, subset = train)
Variables actually used in tree construction:
[1] "Month"     "Peakhour"  "DayofWeek" "Hour"      "Temp"
Number of terminal nodes:  8
Residual mean deviance:  2508.445 = 9579752 / 3819
Distribution of residuals:
       Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
-311.148900  -24.558420   -8.201158    0.000000   25.123950  270.408800
>
```

The model picks only 5 variables namely: Month, Peak hour, Day of the week, Hour, Temperature as these are the variables according to the model that effect the model the most.

On plotting the tree:



Then the tree goes though Cross-Validation to check whether pruning the tree further would improve the performance of the tree model.



After Pruning the tree according to the $dev i.e. deviance produced we use best=7 and grow a tree.

MSE for tree = 2603.957827

MSE for pruned tree = 2762.506921

Therefore the tree without pruning in a better model.

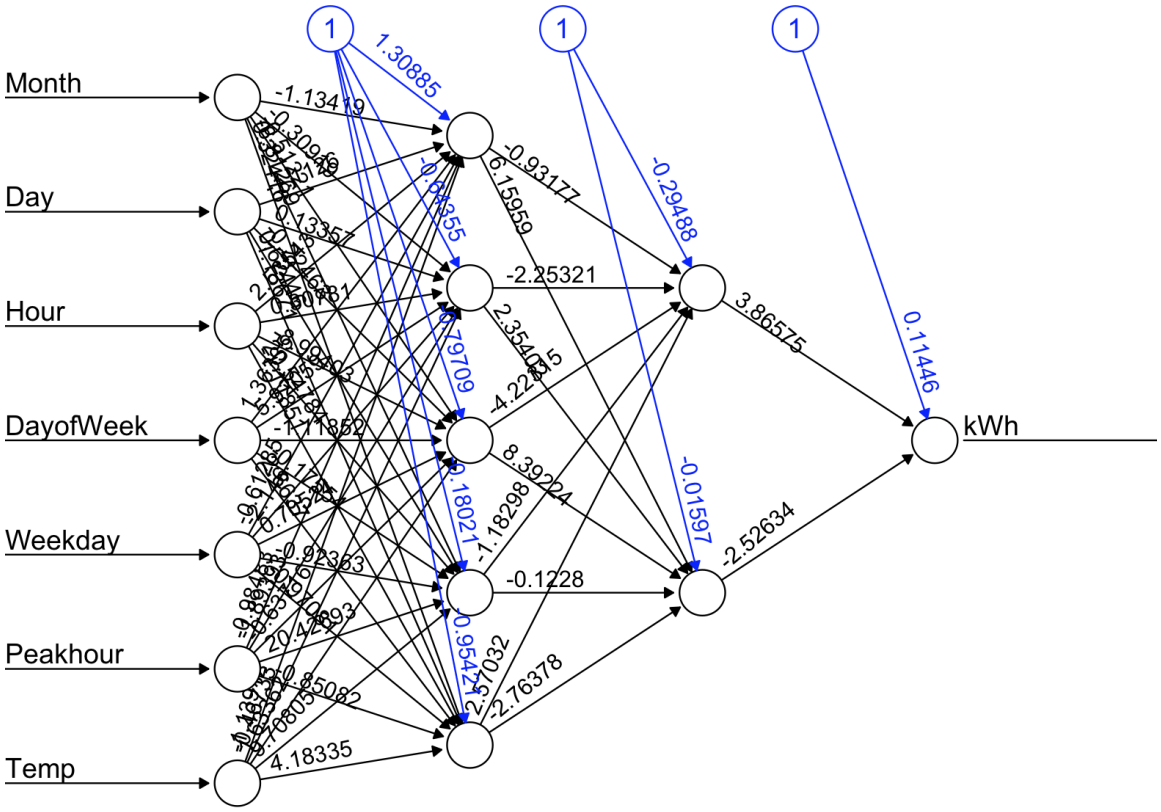
To forecast values for "kWh" first the files forecastNewData.csv and forecastNewData2.csv are cleansed and converted to the sample format. The cleansing of data is done in Python (scripts are provided).

For forecasting the tree model is trained on the dataset hourly_filled_data.csv and the testing dataset is forecastInput.csv. Predicted values of "kWh" is stored in a separated csv file called "forecastNewdataOutput_Tree2" (csv file is provided)

ii. Neural Network-

Fitting the Neural Network Model on to the cleansed dataset without zeros or negative "kWh" values. The dataset is divided into train and test, it is further converted to numeric

After scaling and plotting the neural network the network is as follows:



Error: 40.468995 Steps: 289

The accuracy measures are as follows:

RMSE = 66.68735933

MSE = 4447.203895

MAE = 45.57783088

MAPE = 43.97612948

For forecasting the neural network model is trained on the dataset hourly_filled_data.csv and the testing dataset is forecastInput.csv. Predicted values of "kWh" is stored in a separated csv file called "NeuralNetwork_ForcastOutput" (csv file is provided)

3. Classification:

i.Logistic Regression-

Fitting the Logistic Regression Model on to the cleansed dataset without zeros or negative "kWh" values.

For the purpose of classification a new column is created called "KWH_Class" and the values in this column are either "Optimal" or "Above_Normal"

Condition (code snippet in R):

```
#finding mean of kWh
meanKwh= mean(cleandata$kWh)

# adding the column KWH_Class
cleandata$KWH_Class <- cleandata$kWh

val <- function (x){
  if(x > meanKwh) y <- "Above_Normal"
  if(x <= meanKwh) y <- "Optimal"
  return(y)
}

cleandata$KWH_Class <- sapply(cleandata$kWh,val)
```

On Transforming "KWH_Class" into binary (0,1)

```
Optimal Above_Normal
   2635         5020
```

After splitting the "hourly_filled_data.csv" into train and test data and running logistic regression the Classification matrix obtained is as follows:

```
> #classification matrix
> confusionMatrix(test$nyClass,pred)
Confusion Matrix and Statistics

              Reference
Prediction     Above_Normal Optimal
  Above_Normal          143    1104
  Optimal               448     219

               Accuracy : 0.1891327
                 95% CI : (0.1718139, 0.2074094)
    No Information Rate : 0.6912226
    P-Value [Acc > NIR] : 1

                  Kappa : -0.4533072
 Mcnemar's Test P-Value : <0.0000000000000002

            Sensitivity : 0.24196277
            Specificity : 0.16553288
         Pos Pred Value : 0.11467522
         Neg Pred Value : 0.32833583
             Prevalence : 0.30877743
         Detection Rate : 0.07471264
   Detection Prevalence : 0.65151515
      Balanced Accuracy : 0.20374783

       'Positive' Class : Above_Normal
```

ii. <u>Classification Tree-</u>

Fitting the Classification Tree Model on to the cleansed dataset without zeros or negative "kWh" values.

A new column is added to the dataset called "KWH_Class".

This column contains two values, "Optimal" and "Above_Normal".

Condition:

```
#finding mean of kWh
meanKwh= mean(cleandata$kWh)

# adding the column KWH_Class
cleandata$KWH_Class <- cleandata$kWh

val <- function (x){
  if(x > meanKwh) y <- "Above_Normal"
  if(x <= meanKwh) y <- "Optimal"
  return(y)
}

cleandata$KWH_Class <- sapply(cleandata$kWh,val)
```

To predict the value of "KWH_Class" the tree() function is used.

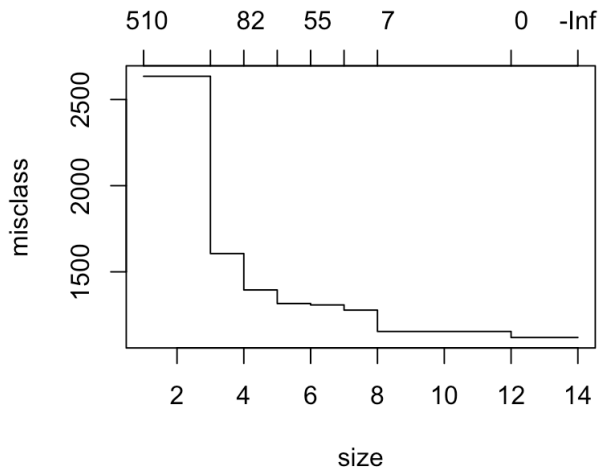
On splitting the dataset into test and train the classification matrix is

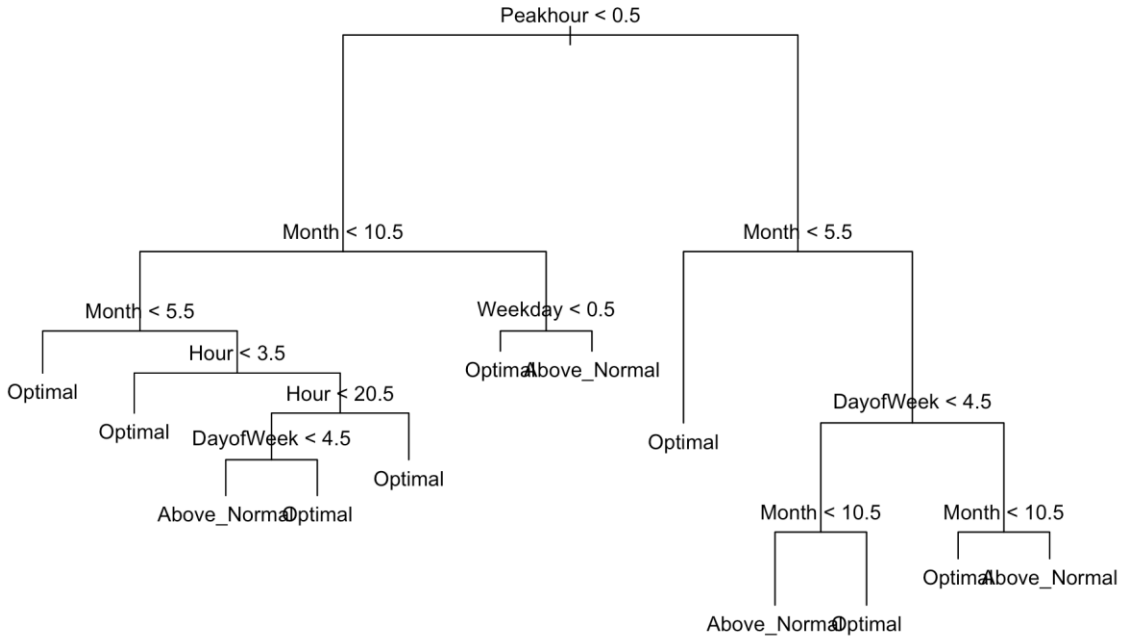
as follows:

```
> #evaluating performance on test data
> tree.pred = predict(tree.train, cleandata.test, type="class")
> #tree.pred<- na.omit(tree.pred)
> table(tree.pred, KWH_Class.test)
             KWH_Class.test
tree.pred      Above_Normal Optimal
  Above_Normal         1862     627
  Optimal               708    4258
>
```

Error rate= (627+708)/(1862+4258)= 21%

On cross validation of the tree and pruning the tree with best=12


510
82
55
7
0
-Inf
misclass
2500
2000
1500
2
4
6
8
10
12
14
size


Peakhour < 0.5
Month < 10.5
Month < 5.5
Month < 5.5
Weekday < 0.5
Hour < 3.5
Optimal
Optimal
Above_Normal
Hour < 20.5
Optimal
DayofWeek < 4.5
Optimal
Above_Normal
Optimal
DayofWeek < 4.5
Optimal
Month < 10.5
Month < 10.5
Above_Normal
Optimal
Above_Normal

The classification matrix of the pruned tree is as follows:

```
> #pruned tree performance
> prune.pred= predict(prune.cleandata,cleandata.test,type="class")
> table(prune.pred,KWH_Class.test)
              KWH_Class.test
prune.pred     Above_Normal Optimal
  Above_Normal         1852     367
  Optimal               718    4518
> 
```

Error rate= (718+367)/(1852+4518)=17.03 %

Therefore, The pruned model is to be selected, as the error rate is lesser.

For forecasting the tree model is trained on the dataset hourly_filled_data.csv and the testing dataset is forecastInput.csv. Predicted values of "kWh" is stored in a separated csv file called "forecastOutput_ClassificationTree.csv" (csv file is provided)

iii. Neural Network-

Fitting the Neural Network Classification Model on to the cleansed dataset without zeros or negative "kWh" values.

A new column is added to the dataset called "KWH_Class".

This column contains two values, "Optimal" and "Above_Normal".

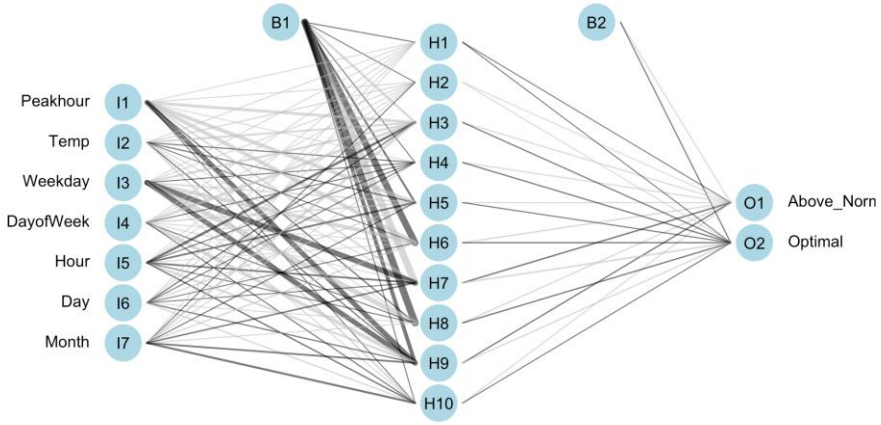
Condition:

```
#finding mean of kWh
meanKwh= mean(cleandata$kWh)

# adding the column KWH_Class
cleandata$KWH_Class <- cleandata$kWh

val <- function (x){
  if(x > meanKwh) y <- "Above_Normal"
  if(x <= meanKwh) y <- "Optimal"
  return(y)
}

cleandata$KWH_Class <- sapply(cleandata$kWh,val)
```

On splitting the dataset into test and train data and removing the "NA" values using na.omit and using the variables:

Peak hour, Temperature, Weekday, Day of Week, Hour, Day and Month. The neural network that was generated is as follows along with its iterations:

```
> seedsANN = nnet(class.ind(KWH_Class)~Peakhour+Temp+Weekday+DayofWeek+Hour+Day+Month, seeds[seedstrain,],size
=10, softmax=TRUE, na.action = na.omit)
# weights:  102
initial  value 4103.414138
iter  10 value 3234.064245
iter  20 value 2787.349843
iter  30 value 2527.880108
iter  40 value 2142.768116
iter  50 value 1983.637335
iter  60 value 1933.973147
iter  70 value 1924.777983
iter  80 value 1911.226350
iter  90 value 1900.385430
iter 100 value 1899.031945
final  value 1899.031945
stopped after 100 iterations
```



The classification Matrix that was generated is as follows:

```
> table(predict(seedsANN, seeds[seedstest,-12], type="class"),seeds[seedstest,]$KWH_Class)

              Above_Normal Optimal
  Above_Normal          635     242
  Optimal               143    1276
```

Error Rate = (143+242)/(635+1276)=20.14%

For forecasting the neural network model is trained on the dataset hourly_filled_data.csv and the testing dataset is forecastInput.csv. Predicted values of “kWh” is stored in a separated csv file called “forcastNewdataOuput_ClassificationNeural.csv” (csv file is provided)