**Advances in Data Sciences**

**Report – Assignment 2**

**Part 1:**

1. Data Cleansing: Code and outputs from the three approaches to filling data.

“**NewData.csv**” file has the raw data with zero values. First the file is cleaned by using various functions like aggregation, melt, cbind, ifelse, as.Date etc.

After this, we get the “Assignment2\_WithZeroesDataset.csv” which has data in the proper format but with a lot of zero KWh values.

1. Remove all the zero-entries and use only the non-zero entries for KWH to build a regression model (KWH = function of (temperature, day of week, month etc.).

**1\_Removing all the zero entries:**

**2\_Building the model with non-zero dataset:**

* The data is “Assignment2\_NonZeroDataset” after removing all the rows with zero KWh values in the

*data1aa<- read.csv(file.choose(),header = T)*

*head(data1aa)*

*summary(data1aa)*

*View(data1aa)*

* 75% of the sample size is used for train and rest for test

*smp\_size <- floor(0.75 \* nrow(data1aa))*

* Set the seed to make your partition reproductible

*set.seed(123)*

*train\_ind <- sample(seq\_len(nrow(data1aa)), size = smp\_size)*

#Split the data into training and testing

*train <- data1aa[train\_ind, ]*

*test <- data1aa[-train\_ind, ]*

* Here the regression model is built on the train data of non-zero file which will be used for the prediction on Non-zero dataset and the raw data set.

*fit<-with(train,lm(KWh ~ Temperature+Peakhour+DayofWeek+WeekDay+hour+month+day))*

*summary(fit)*

*files<-summary(fit)*

*a<-files$coefficients[,1]*

*coef<-coefficients(files)*

*View(coef)*

*write.csv(coef, file="Coefficient\_Dataset\_1aa.csv")*

*install.packages("forecast")*

*library(forecast)*

*KWh<-predict.lm(fit, test)*

*newKWh<-data.frame(KWh)*

*View(newKWh)*

*write.csv(newKWh, file="Predicted\_Dataset\_1aa.csv")*

* The RMSE, MAPE and MAE values will be shown here

*abc<-accuracy(KWh, train$KWh)*

*View(abc)*

*write.csv(abc, file="Regression\_Dataset\_1aa.csv")*

**Applying the above model to the Non-Zero Dataset:**

*install.packages("forecast")*

*library(forecast)*

*KWh<-predict.lm(fit, test)*

*newKWh<-data.frame(KWh)*

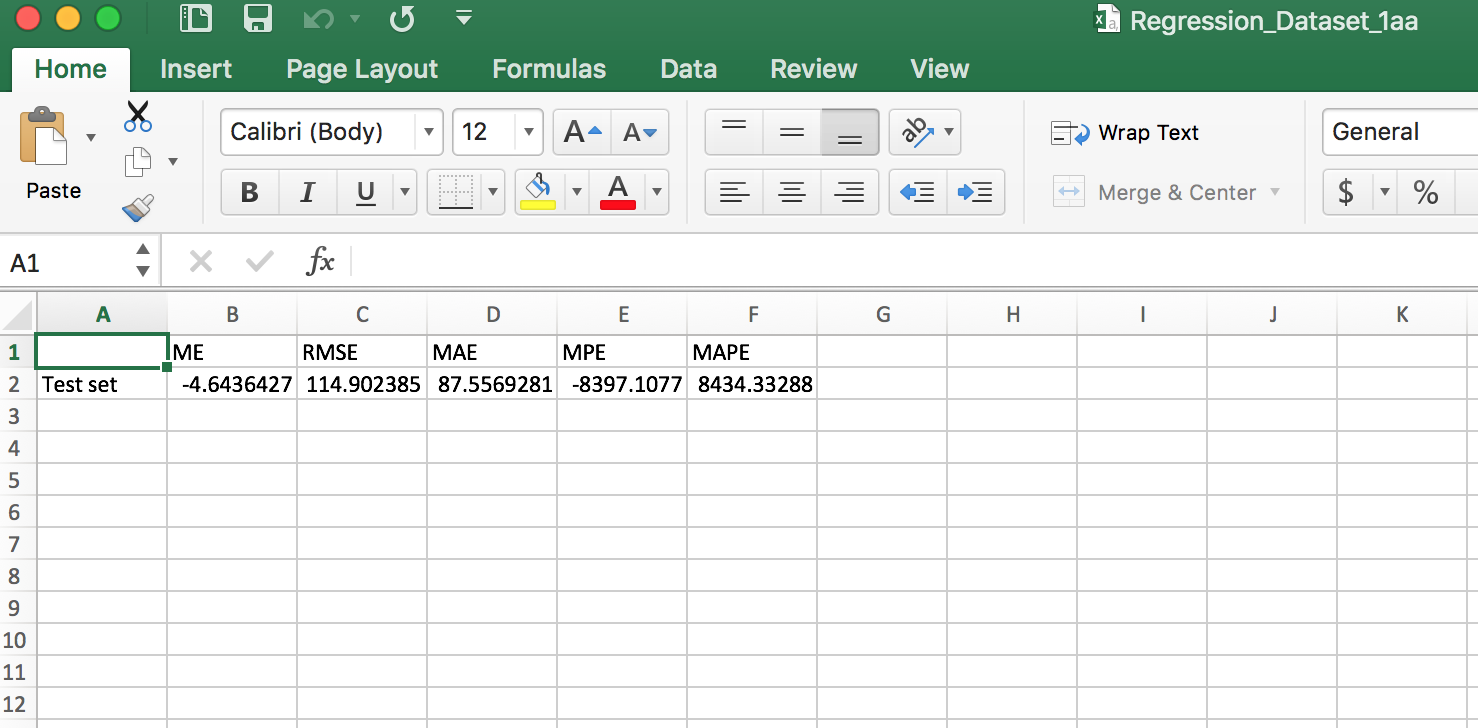
*View(newKWh)*

*write.csv(newKWh, file="Predicted\_Dataset\_1aa.csv")*

*abc<-accuracy(KWh, train$KWh)*

*View(abc)*

*write.csv(abc, file="Regression\_Dataset\_1aa.csv")*

**

**Applying the above model to the Raw (With Zeroes) Dataset:**

*data2<-read.csv(file.choose(),header = T)*

*prediction<-predict.lm(fit, data2)*

*View(data2)*

*head(prediction)*

*summary(prediction)*

*dataframe<-data.frame(prediction)*

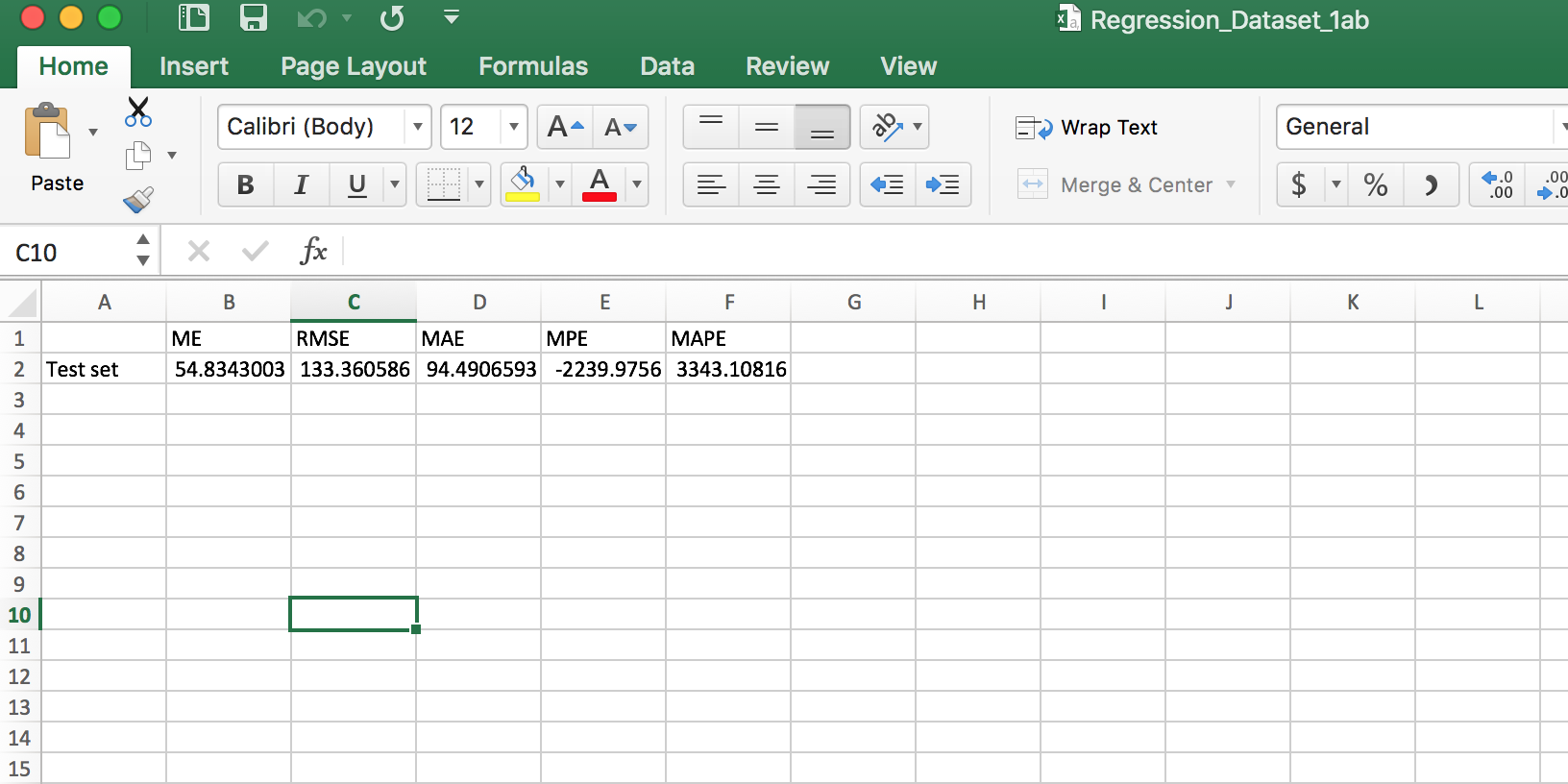
*View(dataframe)*

*write.csv(dataframe, file="Predicted\_Dataset\_1ab.csv")*

*abc<-accuracy(prediction, data1$KWh)*

*View(abc)*

*write.csv(abc, "Regression\_Dataset\_1ab.csv")*

**

1. You build a model to fill the data. For this, you build a regression model (KWH = function of (day of week, month etc.) without temperature) with non-zero data. You use this model to replace the zeros with the model generated values. You use this data build a regression model (KWH = function of (temperature, day of week, month etc..).

**Regression Model without Temperature by using the non-zero dataset:**

*library(MASS)*

*install.packages("ISLR")*

*library(ISLR)*

* Using the file that was cleansed with zeroes removed- Assignment2, Part1, a

*data<- read.csv(file.choose(),header = T)*

*df <- data.frame(data)*

*View(df)*

*names(df)*

* 75% of the sample size

*smp\_size <- floor(0.75 \* nrow(df))*

* Set the seed to make your partition reproductible

*set.seed(123)*

*train\_ind <- sample(seq\_len(nrow(df)), size = smp\_size)*

* Split the data into training and testing

*train <- df[train\_ind, ]*

*test <- df[-train\_ind, ]*

* Fit a linear regression model without Temperature

*lm.fit = lm(KWh ~ Peakhour+WeekDay+month+hour+DayofWeek+Date, data = train)*

*summary(lm.fit)*

* Making the csv file for the coefficients

*files<-summary(lm.fit)*

*a<-files$coefficients[,1]*

*coef<-coefficients(files)*

*View(coef)*

*write.csv(abc, file="Coefficient\_Assign2Part1ba\_newData.csv")*

* Measures of predictive accuracy

*install.packages("forecast")*

*library(forecast)*

*pred = predict(lm.fit, test)*

*newKWh<-data.frame(pred)*

*View(newKWh)*

*abc<-accuracy(pred, train$KWh)*

*View(abc)*

*write.csv(abc, file="Regression\_Assign2Part1ba\_newData.csv")*

**Building the model to replace the zeros with the model generated values:**

* Assignment2, Part1: Algorithm Implementation, 1: Data wrangling and cleansing and Multiple linear regression, a, b
* Assignment2\_NonZeroDataset.csv

*data1<- read.csv(file.choose(),header = T)*

*head(data1)*

*summary(data1)*

*fit<-with(data1,lm(KWh~DayofWeek+WeekDay+hour+month+day))*

*summary(fit)*

* Assignment2\_WithZeroesDataset.csv

*data2<-read.csv(file.choose(),header = T)*

*prediction<-predict.lm(fit, data2)*

*View(data2)*

*head(prediction)*

*summary(prediction)*

*dataframe<-data.frame(prediction)*

*View(dataframe)*

*write.csv(dataframe, file="Predicted\_KWh.csv", row.names=FALSE)*

* Predicted\_KWh.csv

*data3<-read.csv(file.choose(),header = T)*

*predicted\_KWh<-data.frame(data3$prediction)*

*View(predicted\_KWh)*

*View(data3)*

* dataset with only KWh values from zeroes dataset

*KWhValues<-data.frame(KWh=data2$KWh)*

* Replacing the zero KWh values with the model generated values

*zero\_locations <- which(KWhValues == 0, arr.ind=TRUE)*

*KWhValues[zero\_locations] <- predicted\_KWh[zero\_locations]*

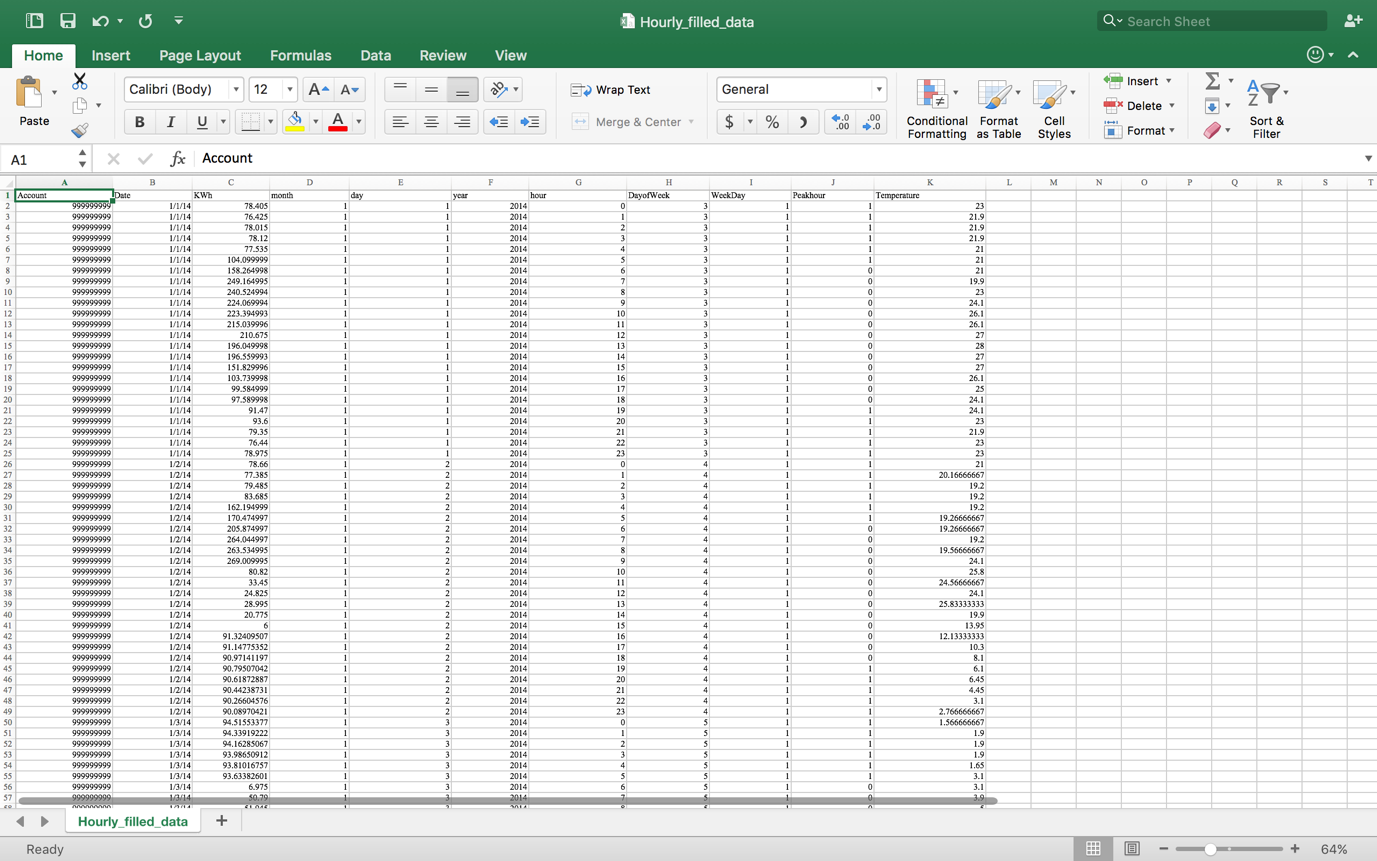
*View(KWhValues)*

*final\_replaced<- cbind(data2[1:2],kWh=KWhValues,data2[4:11])*

* The final dataset received has 8016 rows and is completely non-zero with model generated replaced values

*View(final\_replaced)*

*write.csv(final\_replaced, file="Hourly\_filled\_data.csv", row.names=FALSE)*



**Applying the above model for “filled” dataset:**

* Assignment2, Part1: Algorithm Implementation, 1: Data wrangling and cleansing and Multiple linear regression, b, a
* Model Building for part a and b using the file created by model made without temperature
* Hourly\_filled\_data.csv

*data2ba<-read.csv(file.choose(),header = T)*

*prediction<-predict.lm(fit, data2ba)*

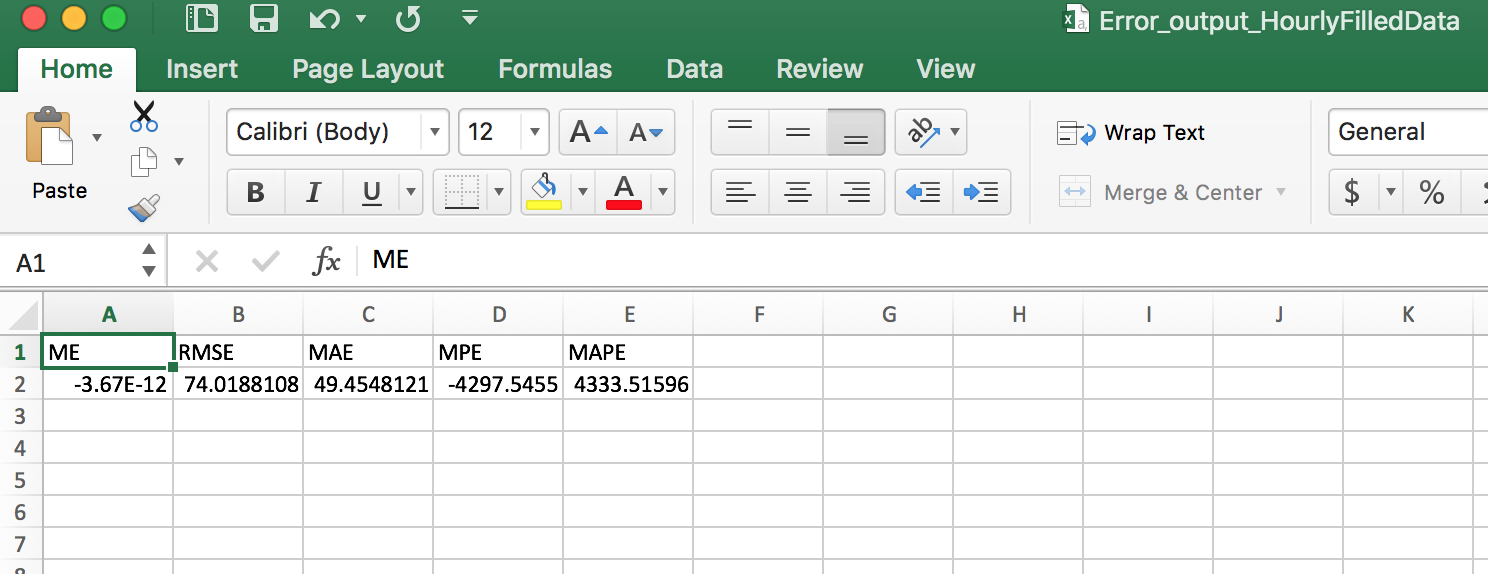
*View(data2ba)*

* Regression Coefficients - MAPE, RMSE etc.

*abc<-accuracy(prediction, data1ba$KWh)*

*View(abc)*

*write.csv(abc, "Error\_output\_HourlyFilledData.csv", row.names=FALSE)*

****

**Applying the above model for “Raw data” dataset:**

* Assignment2, Part1: Algorithm Implementation, 1: Data wrangling and cleansing and Multiple linear regression, b, b
* Model Building for part a and b using the file created by model made without temperature
* Applying model to Part1\_Q1\_rawdata.csv

*data2bb<-read.csv(file.choose(),header = T)*

*prediction<-predict.lm(fit, data2bb)*

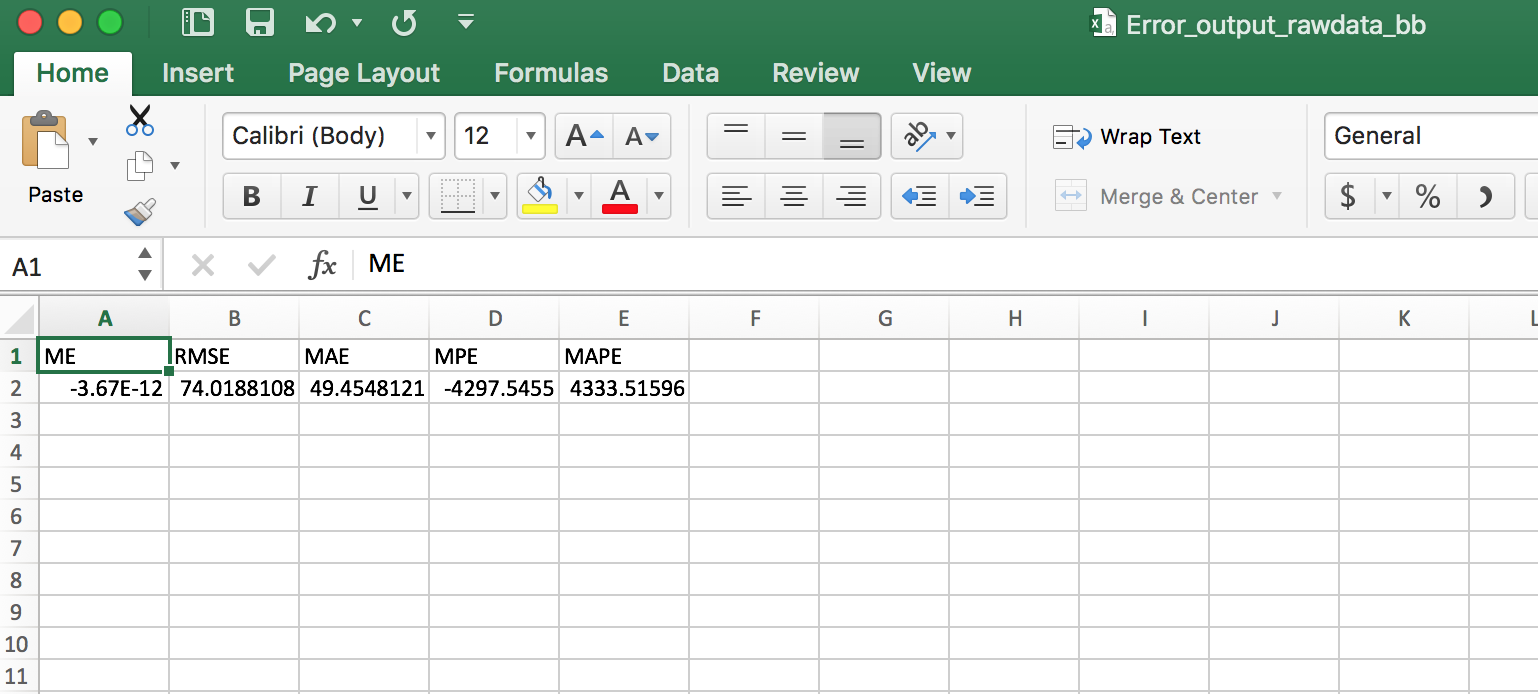
*View(data2bb)*

* *Regression Coefficients - MAPE, RMSE etc.*

*abc<-accuracy(prediction, data1bb$KWh)*

*View(abc)*

*write.csv(abc, "Error\_output\_rawdata\_bb.csv", row.names=FALSE)*

**

1. You come across a package zoo(https://cran.r-project.org/web/packages/zoo/zoo.pdf) that offers functions na.approx, na.fill, na.locf to fill NAs. You replace the zeros with NA and try using this package to fill the NAs. You use this data build a regression model (KWH = function of (temperature, day of week, month etc..).

* We used the na.locf to replace the zero dataset with the predicted values.

*data<- read.csv(file.choose(),header = T)*

*df <- data.frame(data)*

*data\_rawdata<-df[df$Units!= 'Power Factor' & df$Units!= 'kVARh', ]*

*attach(data\_rawdata)*

*aggregate\_12<-sapply(seq(5,292,by=12),function(i) rowSums(data\_rawdata[,i:(i+11)]))*

*View(aggregate\_12)*

*nrow(aggregate\_12)*

*data\_rawdata\_dframe<-cbind(data\_rawdata[1:4],aggregate\_12)*

*View(data\_rawdata\_dframe)*

*install.packages("reshape")*

*library(reshape)*

*data\_rawdata\_meltdata <- melt(data\_rawdata\_dframe, id=c("Account","Date","Channel","Units"))*

*View(data\_rawdata\_meltdata)*

*install.packages("doBy")*

*library(doBy)*

*ordereddataframe <- data\_rawdata\_meltdata[order(data\_rawdata\_meltdata$Date),]*

*View(ordereddataframe)*

*ordereddata<-data.frame(ordereddataframe)*

*attach(ordereddata)*

*library(data.table)*

*install.packages("lubridate")*

*library(lubridate)*

*ordereddata\_year<-year(as.Date(ordereddata$Date, format="%m/%d/%Y"))*

*ordereddata\_month<-month(as.Date(ordereddata$Date, format="%m/%d/%Y"))*

*ordereddata\_day<-day(as.Date(ordereddata$Date, format="%m/%d/%Y"))*

*ordereddata\_weekday <-(wday(as.Date(ordereddata$Date, format="%m/%d/%Y")))-1*

*ordereddata1<-cbind(ordereddata[1:6],ordereddata\_year,ordereddata\_month,ordereddata\_day,ordereddata\_weekday)*

*View(ordereddata1)*

*ordereddata\_weekend<-NA*

*ordereddata2<-cbind(ordereddata1,ordereddata\_weekend)*

*ordereddata\_weekend<- ifelse( ordereddata1$ordereddata\_weekday == 0 | ordereddata1$ordereddata\_weekday == 6 , 0 , 1)*

*ordereddata2<-cbind(ordereddata1,ordereddata\_weekend)*

*View(ordereddata2)*

*ordereddata\_peakhours <-NA*

*ordereddata3<-cbind(ordereddata1,ordereddata\_weekend,ordereddata\_peakhours)*

*ordereddata\_peakhours <- ifelse( as.numeric(ordereddata1$variable) > 6 & as.numeric(ordereddata1$variable) < 20 , 0 , 1)*

*##Dates <- as.Date(newdata$Date, "%m/%d/%Y")*

*FinalDate <- as.Date(ordereddata3$Date, format="%m/%d/%Y")*

*Hour<-as.numeric(ordereddata3$variable)-1*

*ordereddata3<-cbind(ordereddata2,ordereddata\_peakhours,FinalDate,Hour)*

*View(ordereddata3)*

*install.packages("weatherData")*

*library(weatherData)*

*weather\_data<-getWeatherForDate("KBOS","2014-1-1","2014-12-31", opt\_detailed = TRUE,opt\_all\_columns=TRUE)*

*View(weather\_data)*

*weather\_data\_aggregate<-aggregate(list(temperature = weather\_data$TemperatureF), list(Time = cut(weather\_data$Time, "1 hour")), mean)*

*View(weather\_data\_aggregate)*

*weather\_hour <- format(as.POSIXct(weather\_data\_aggregate$Time),"%H")*

*Hour<- as.numeric(weather\_hour)*

*View(Hour)*

*finalweatherdata = cbind(weather\_data\_aggregate,Hour)*

*View(finalweatherdata)*

*final\_rawdata<-data.frame(ordereddata3)*

*View(final\_rawdata)*

*FinalDate <- as.Date(finalweatherdata$Time)*

*final\_weatherdata<-cbind(finalweatherdata,FinalDate)*

*colnames(final\_weatherdata)<-c("Time","Temperature","Hour","FinalDate")*

*View(final\_weatherdata)*

*install.packages("zoo")*

*library(zoo)*

*mergeddata<-merge(x = final\_rawdata,final\_weatherdata,by=c("FinalDate","Hour"),all.x=TRUE)*

*mergeddata$value[mergeddata$value == 0] <- NA*

*mergesdata<- zoo(mergeddata$value)*

*View(mergeddata)*

*X<-na.locf(mergesdata, na.rm=TRUE)*

*y<-data.frame(X)*

*View(y)*

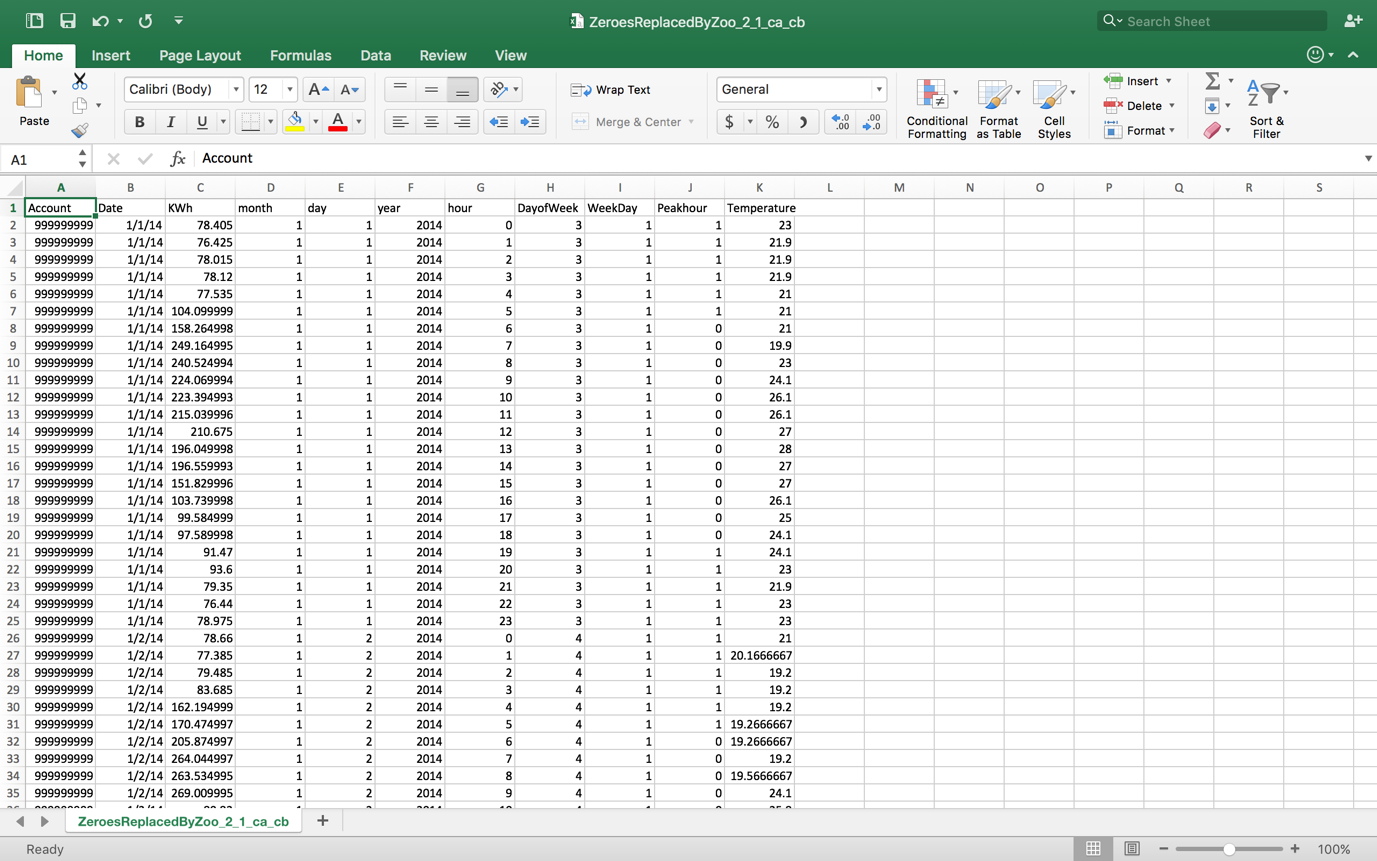
*final<-cbind(mergeddata[2:14],mergeddata$Temperature)*

*View(final)*

*newdataformat<-cbind(final[2:3],KWh=y$X,month=final$ordereddata\_month,day = final$ordereddata\_day,year= final$ordereddata\_year,hour= final$Hour,DayofWeek = final$ordereddata\_weekday, WeekDay = final$ordereddata\_weekend, Peakhour = final$ordereddata\_peakhours,Temperature = mergeddata$Temperature)*

*View(newdataformat)*

*write.csv(newdataformat,file="* *ZeroesReplacedByZoo\_2\_1\_ca\_cb",row.names = FALSE)*

****

**Applying the above model for the “filled” dataset:**

* Assignment2, Part1: Algorithm Implementation, 1: Data wrangling and cleansing and Multiple linear regression, c, a
* Using file made by zoo file (Applying to filled dataset)
* ZeroesReplacedByZoo\_2\_1\_ca\_cb.csv

*data1ca<- read.csv(file.choose(),header = T)*

*head(data1ca)*

*summary(data1ca)*

*fit<-with(data1ca,lm(KWh~Peakhour+Temperature+DayofWeek+WeekDay+hour+month+day))*

*summary(fit)*

*files<-summary(fit)*

*a<-files$coefficients[,1]*

*coef<-coefficients(files)*

*View(coef)*

*write.csv(coef, file="Coefficient\_Dataset\_1ca\_filled.csv")*

* ZeroesReplacedByZoo\_2\_1\_ca\_cb.csv

*data2ca<-read.csv(file.choose(),header = T)*

*prediction<-predict.lm(fit, data2ca)*

*View(data2ca)*

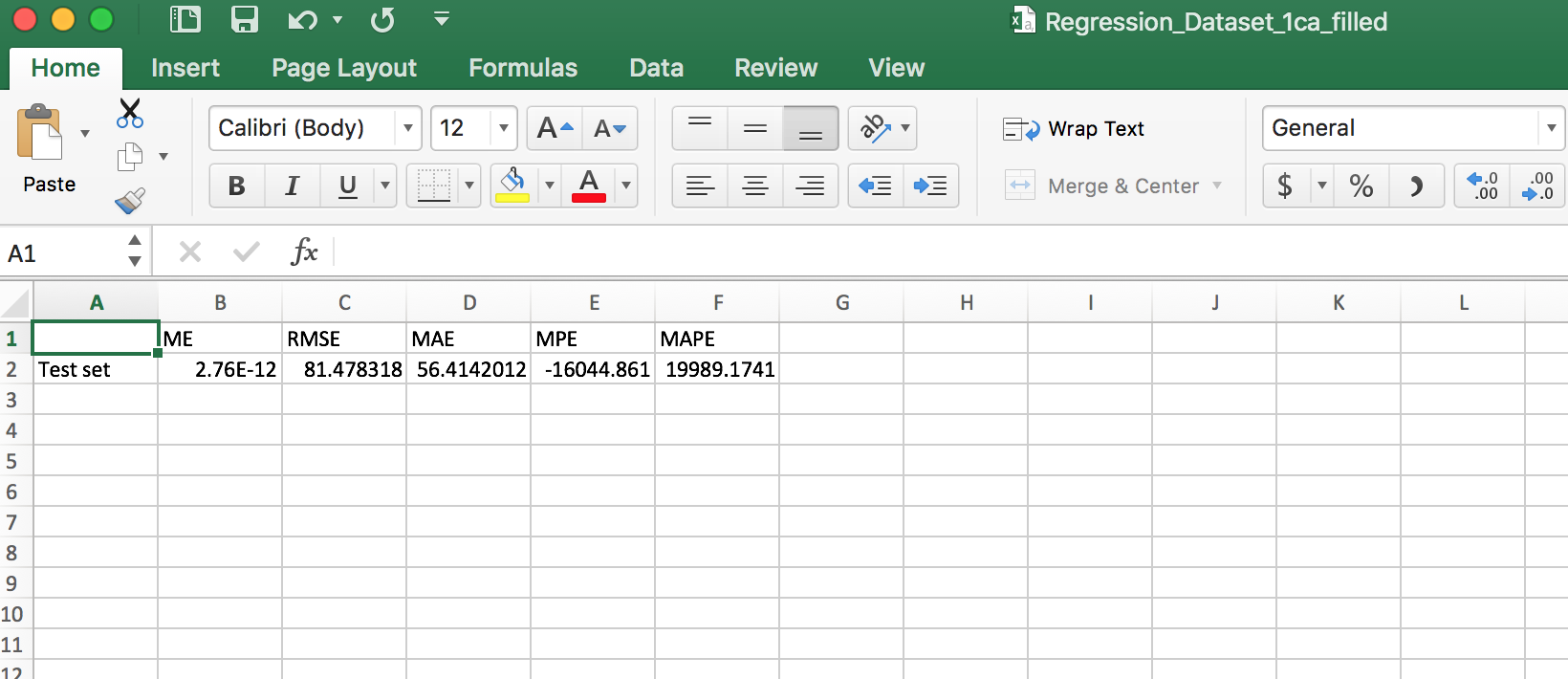
* Regression Coefficients - MAPE, RMSE etc.

*library("forecast")*

*abc<-accuracy(prediction, data1ca$KWh)*

*View(abc)*

*write.csv(abc, "Regression\_Dataset\_1ca\_filled.csv")*



**Applying the above model to the “raw” dataset**

* Assignment2, Part1: Algorithm Implementation, 1: Data wrangling and cleansing and Multiple linear regression, c, b
* Using file made by zoo file (Applying to raw/zeroes dataset)
* ZeroesReplacedByZoo\_1\_ca\_cb.csv

*data1cb<- read.csv(file.choose(),header = T)*

*head(data1cb)*

*summary(data1cb)*

*fit<-with(data1bb,lm(KWh~Peakhour+Temperature+DayofWeek+WeekDay+hour+month+day))*

*summary(fit)*

*files<-summary(fit)*

*a<-files$coefficients[,1]*

*coef<-coefficients(files)*

*View(coef)*

*write.csv(coef, file="Coefficient\_Dataset\_1cb\_zeroes.csv")*

* Using Assignment2\_WithZeroesDataset.csv

*data2cb<-read.csv(file.choose(),header = T)*

*prediction<-predict.lm(fit, data2cb)*

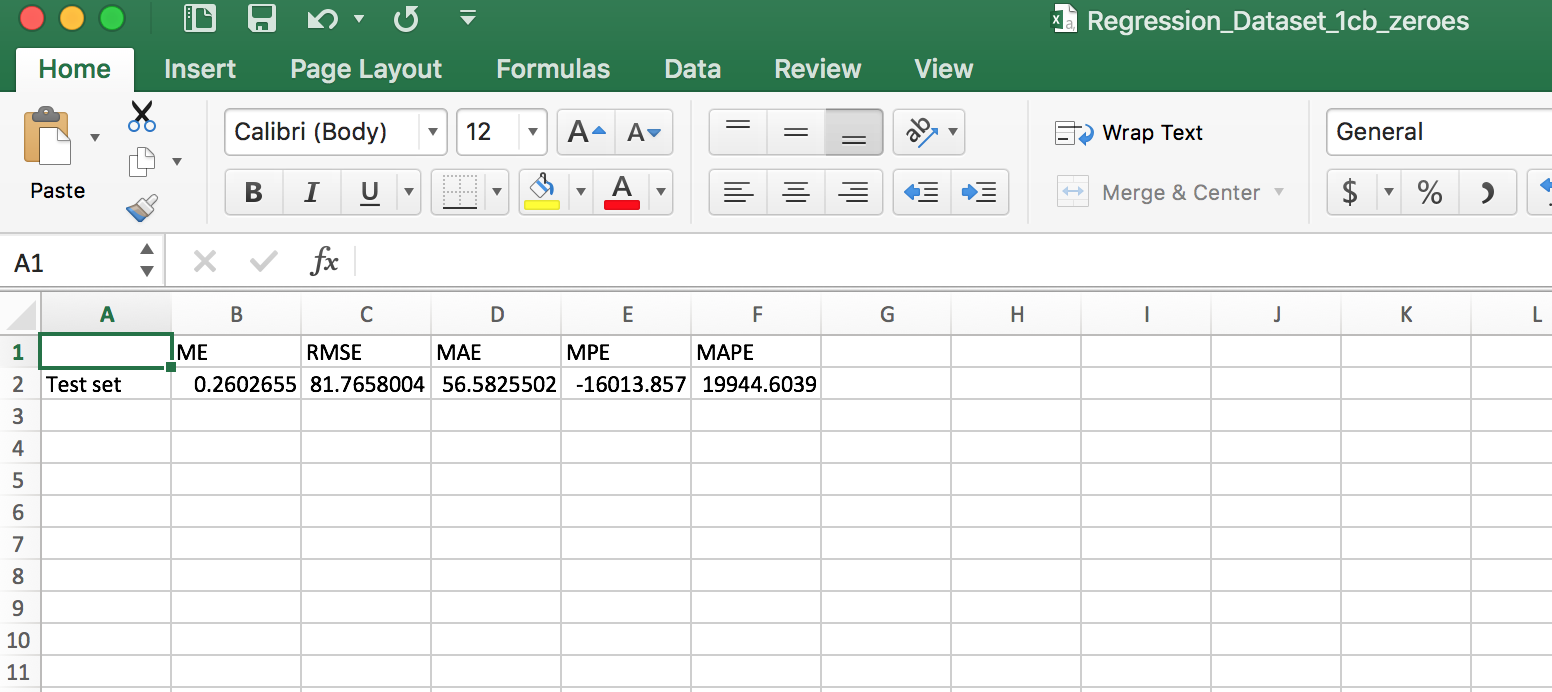
*View(data2cb)*

* Regression Coefficients - MAPE, RMSE etc.

*abc<-accuracy(prediction, data1cb$KWh)*

*View(abc)*

*write.csv(abc, "Regression\_Dataset\_1cb\_zeroes.csv")*



1. Prediction
2. Code and prediction output files in the format **PredictionPerformanceMetrics.csv**

#Assignment 2, Part 1, Question 2, Prediction

# Regression Trees

*install.packages("tree")*

*library(tree)*

*library(MASS)*

*library(ISLR)*

#Hourly\_filled\_data.csv

*data<-read.csv(file.choose(),header=TRUE)*

*Tree\_dataframe<-data.frame(data)*

*View(Tree\_dataframe)*

#Set the seed to make your partition reproductible

*set.seed(1)*

*train\_ind <- sample(1:nrow(Tree\_dataframe), nrow(Tree\_dataframe)/2)*

*#Split the data into training and testing*

*train <- Tree\_dataframe[train\_ind, ]*

*test <- Tree\_dataframe[-train\_ind, ]*

*fit <- tree(KWh~Temperature+Peakhour+hour+month+DayofWeek+WeekDay, train)*

*summary(fit)*

*KWh\_tree<-predict(fit, test)*

*newKWh\_tree<-data.frame(KWh\_tree)*

*View(newKWh\_tree)*

*library(forecast)*

*abc<-accuracy(KWh\_tree, train$KWh)*

*View(abc)*

- Plotting and pruning

*plot (fit)*

*text(fit, pretty = 0)*

* Cross-validation

*cv\_tree <- cv.tree(fit)*

*plot(cv\_tree$size, cv\_tree$dev, type="b")*

*prune\_tree <- prune.tree(fit, best = 5)*

*plot(prune\_tree)*

*text(prune\_tree, pretty = 0)*

* Unpruned tree on test data

*unpruned\_test <- predict(fit, test)*

*testdata1 <- Tree\_dataframe [-train\_ind,"KWh"]*

*plot(unpruned\_test, testdata1)*

*abline(0,1)*

*mean((unpruned\_test -testdata1)^2)*

* Prune tree on test data

*pruned\_test = predict (prune\_tree, test)*

*testdata2=Tree\_dataframe [-train\_ind,"KWh"]*

*plot(pruned\_test,testdata2)*

*abline (0,1)*

*mean((pruned\_test -testdata2)^2)*

*testdata<-data.frame(pruned\_test)*

*View(testdata)*

* Artificial Neural Network

*set.seed(500)*

*library(MASS)*

*data<-read.csv(file.choose(),header=TRUE)*

* Just to check the missing data

*apply(data,2,function(x) sum(is.na(x)))*

*index <- sample(1:nrow(data),round(0.75\*nrow(data)))*

*#train <- data[index,]*

*#test <- data[-index,]*

#deleting the account, date and year columns

*data <- data[,-c(1,2,6)]*

*View(data)*

*maxs <- apply(data, 2, max)*

*mins <- apply(data, 2, min)*

*head(data)*

*data$WeekDay <- as.numeric(data$WeekDay)*

*data$Peakhour <- as.numeric(data$Peakhour)*

*data$day<-as.numeric(data$day)*

*data$hour<-as.numeric(data$hour)*

*data$month<-as.numeric(data$month)*

*data$DayofWeek<-as.numeric(data$DayofWeek)*

*scaled <- as.data.frame(scale(data, center = mins, scale =maxs - mins))*

*str(scaled)*

*train\_ <- scaled[index,]*

*test\_ <- scaled[-index,]*

#model for neural network

*library(neuralnet)*

*n <- names(train\_)*

*f <- as.formula(paste("KWh ~", paste(n[!n %in% "KWh"], collapse = " + ")))*

*nn <- neuralnet(f,data=train\_, hidden=5, threshold= 0.6, linear.output=F)*

*plot(nn)*

*pr.nn <- compute(nn,test\_[,2:8])*

*View(pr.nn)*

*pr.nn\_ <- pr.nn$net.result\*(max(data$KWh)-min(data$KWh))+min(data$KWh)*

*View(pr.nn\_)*

*#The predicted output KWh values*

*test.r <- (test\_$KWh)\*(max(data$KWh)-min(data$KWh))+min(data$KWh)*

*#test.r\_frame<-data.frame(test.r)*

*#View(test.r\_frame)*

*MSE.nn <- sum((test.r - pr.nn\_)^2)/nrow(test\_)*

*View(MSE.nn)*

*RMSE.nn<-sqrt(MSE.nn)*

*#error*

*error<- test.r - pr.nn\_*

*#rmse*

*rmse <- function(error)*

*{*

*sqrt(mean(error^2))*

*}*

*RMSE.nn<-rmse(error)*

*#mae*

*mae <- function(error)*

*{*

*mean(abs(error))*

*}*

*MAE.nn<-mae(error)*

*#mape*

*mape<-function(error)*

*mean(abs((error/test.r)))*

*MAPE.nn<-mape(error)*

*#Binding the values into one csv*

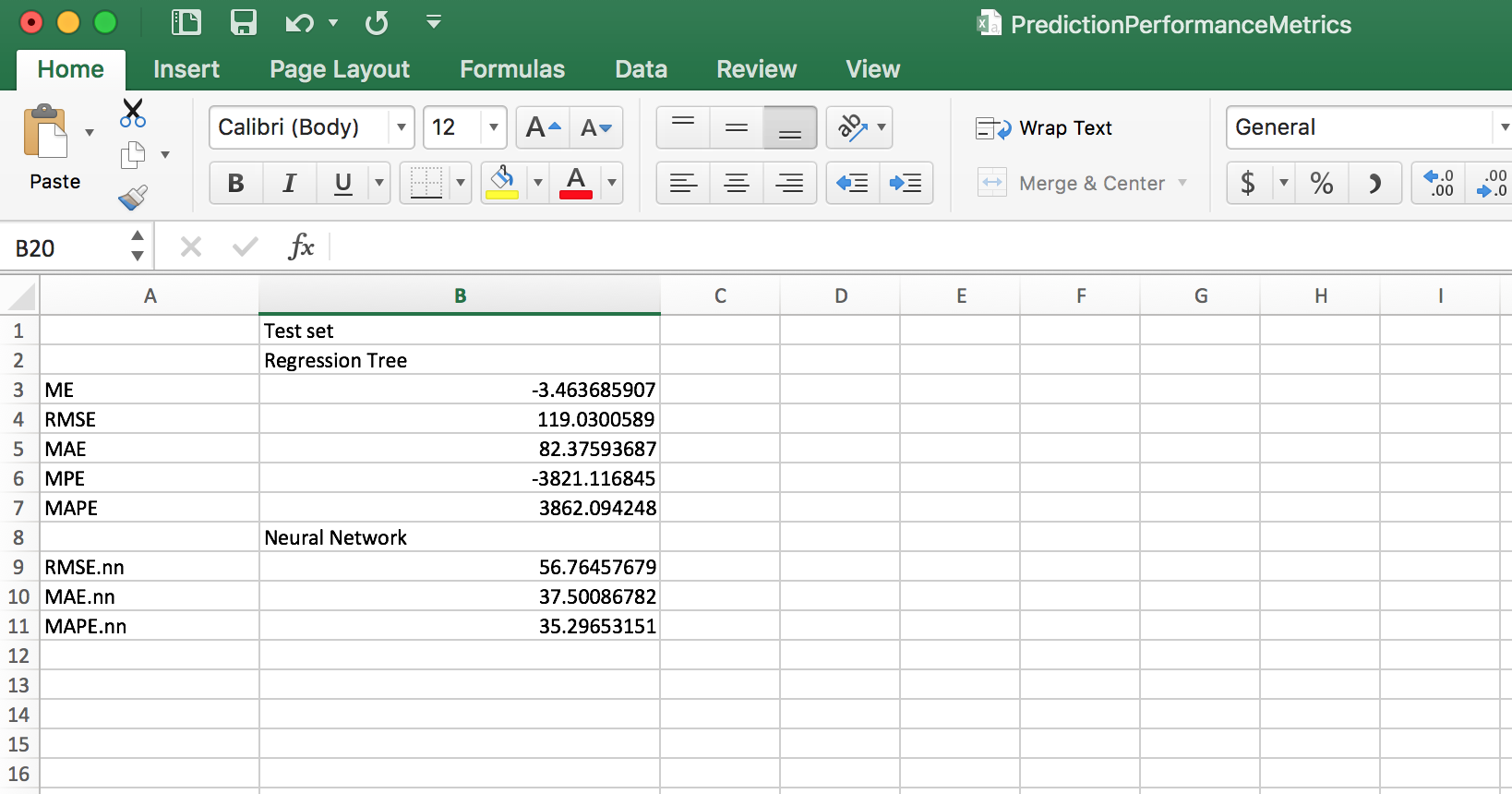
*error\_rt<-cbind("Regression Tree",abc)*

*error\_nn<- cbind("Neural Network",RMSE.nn,MAE.nn,MAPE.nn)*

*final\_error<-cbind(error\_rt,error\_nn)*

*View(t(final\_error))*

*write.csv(t(final\_error),"PredictionPerformanceMetrics.csv")*



1. Output from your forecast script in the format **forecastOutput\_<Account No>\_neuralNetwork.csv** and **forecastOutput\_<Account No>\_regressionTree.csv**

* Artificial Neural Network on multiple models
* **ForecastNewData.csv**

*set.seed(500)*

*library(MASS)*

* Hourly\_filled\_dataset

*data<-read.csv(file.choose(),header=TRUE)*

* Just to check the missing data

*apply(data,2,function(x) sum(is.na(x)))*

*index <- sample(1:nrow(data),round(0.75\*nrow(data)))*

* Deleting the account, date and year columns

*data <- data[,-c(1,2,6)]*

*View(data)*

* Standardizing data in maxs and mins so the whole data can be scaled

*maxs <- apply(data, 2, max)*

*mins <- apply(data, 2, min)*

*head(data)*

* Converting the column attributes to the numeric values because neural networks works for the numeric data

*data$WeekDay <- as.numeric(data$WeekDay)*

*data$Peakhour <- as.numeric(data$Peakhour)*

*data$day<-as.numeric(data$day)*

*data$hour<-as.numeric(data$hour)*

*data$month<-as.numeric(data$month)*

*data$DayofWeek<-as.numeric(data$DayofWeek)*

*scaled <- as.data.frame(scale(data, center = mins, scale =maxs - mins))*

*str(scaled)*

*train\_ <- scaled[index,]*

*test\_ <- scaled[-index,]*

* Building the model for neural network

*library(neuralnet)*

*n <- names(train\_)*

*f <- as.formula(paste("KWh ~", paste(n[!n %in% "KWh"], collapse = " + ")))*

*nn <- neuralnet(f,data=train\_,hidden=5, threshold= 0.6,linear.output=F)*

*plot(nn)*

* Using ForecastNewData.csv
* ForecastNewData.csv

*firstnewforecast<-read.csv(file.choose(),header=TRUE)*

*View(firstnewforecast)*

*newforecast <- firstnewforecast[,-c(1,4)]*

*View(newforecast)*

*str(newforecast)*

*newforecast$WeekDay <- as.numeric(newforecast$WeekDay)*

*newforecast$Peakhour <- as.numeric(newforecast$Peakhour)*

*newforecast$Day<-as.numeric(newforecast$Day)*

*newforecast$Hour<-as.numeric(newforecast$Hour)*

*newforecast$month<-as.numeric(newforecast$month)*

*newforecast$Day.of.Week<-as.numeric(newforecast$Day.of.Week)*

*newforecast$Temperature<-as.numeric(newforecast$Temperature)*

* Using compute function to predict the neuron columns

*predictforecast.nn <- compute(nn,newforecast[,1:7])*

*View(predictforecast.nn)*

* Using it to forecast the KWh value for ForecastNewData.csv

*pr.nn\_forecast <- predictforecast.nn$net.result\*(max(data$KWh)-min(data$KWh))+min(data$KWh)*

*View(pr.nn\_forecast)*

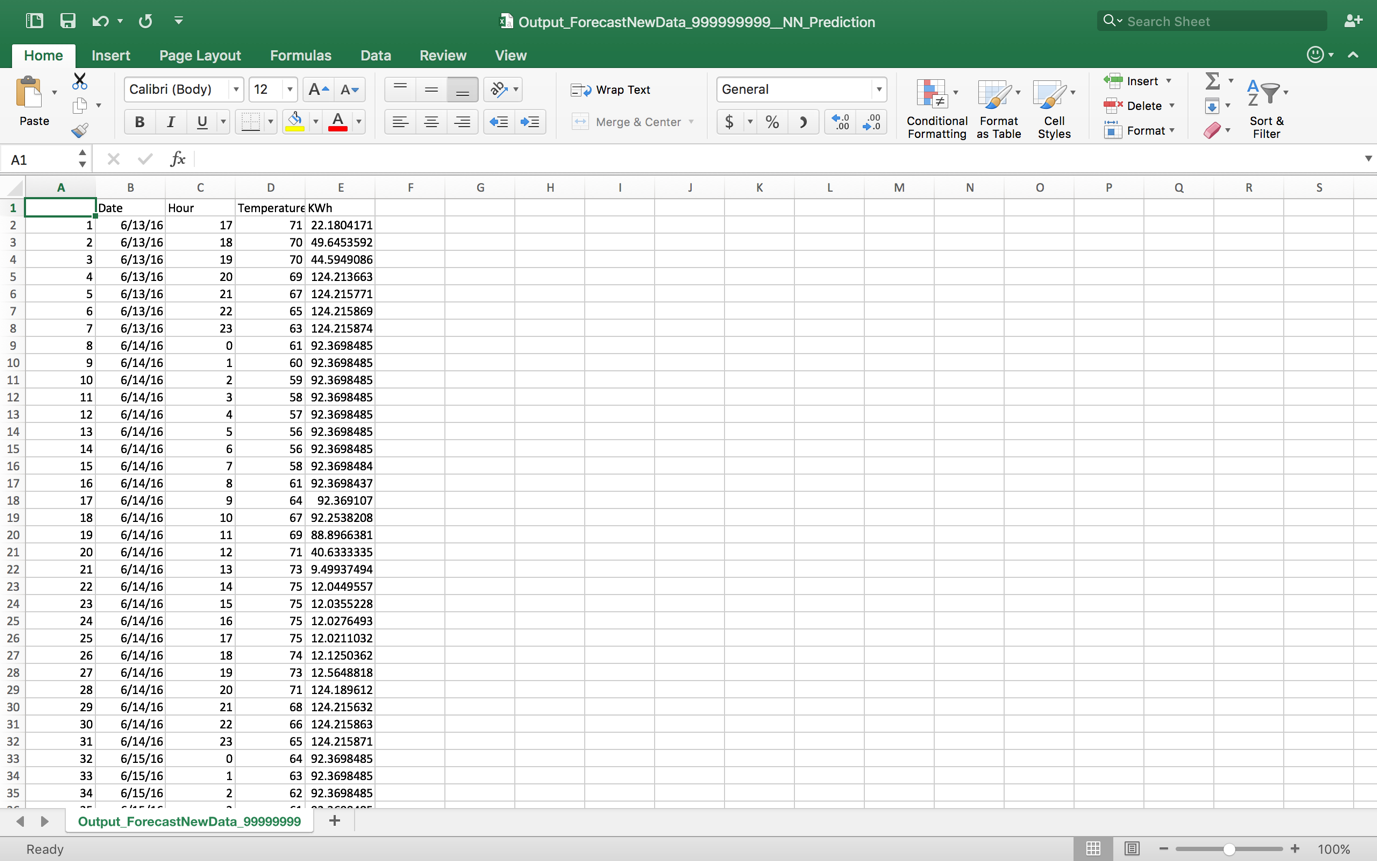
* Combining the Forecast file by adding the KWh predicted column to the forecast file’s three columns – Date, Hour and Temperature

*View(firstnewforecast)*

*Forecast\_KWh<-cbind(firstnewforecast[1],firstnewforecast[5],firstnewforecast[9],KWh= pr.nn\_forecast)*

*View(Forecast\_KWh)*

*write.csv(Forecast\_KWh, file="ForecastNewData\_999999999\_NN\_Prediction.csv")*



* **Using ForecastNewData2.csv**
* *Artificial Neural Network on multiple models*

*set.seed(500)*

*library(MASS)*

* *Hourly\_filled\_dataset*

*data<-read.csv(file.choose(),header=TRUE)*

* *Just to check the missing data*

*apply(data,2,function(x) sum(is.na(x)))*

*index <- sample(1:nrow(data),round(0.75\*nrow(data)))*

* *deleting the account, date and year columns*

*data <- data[,-c(1,2,6)]*

*View(data)*

*maxs <- apply(data, 2, max)*

*mins <- apply(data, 2, min)*

*head(data)*

*data$WeekDay <- as.numeric(data$WeekDay)*

*data$Peakhour <- as.numeric(data$Peakhour)*

*data$day<-as.numeric(data$day)*

*data$hour<-as.numeric(data$hour)*

*data$month<-as.numeric(data$month)*

*data$DayofWeek<-as.numeric(data$DayofWeek)*

*scaled <- as.data.frame(scale(data, center = mins, scale =maxs - mins))*

*str(scaled)*

*train\_ <- scaled[index,]*

*test\_ <- scaled[-index,]*

* *model for neural network*

*library(neuralnet)*

*n <- names(train\_)*

*f <- as.formula(paste("KWh ~", paste(n[!n %in% "KWh"], collapse = " + ")))*

*nn <- neuralnet(f,data=train\_,hidden=5, threshold= 0.6,linear.output=F)*

*plot(nn)*

* *----ForecastNewData2.csv----*

*firstnewforecast<-read.csv(file.choose(),header=TRUE)*

*View(firstnewforecast)*

*newforecast <- firstnewforecast[,-c(1,4)]*

*View(newforecast)*

*str(newforecast)*

*newforecast$WeekDay <- as.numeric(newforecast$WeekDay)*

*newforecast$Peakhour <- as.numeric(newforecast$Peakhour)*

*newforecast$Day<-as.numeric(newforecast$Day)*

*newforecast$Hour<-as.numeric(newforecast$Hour)*

*newforecast$month<-as.numeric(newforecast$month)*

*newforecast$Day.of.Week<-as.numeric(newforecast$Day.of.Week)*

*newforecast$Temperature<-as.numeric(newforecast$Temperature)*

*predictforecast.nn <- compute(nn,newforecast[,1:7])*

*View(predictforecast.nn)*

*pr.nn\_forecast <- predictforecast.nn$net.result\*(max(data$KWh)-min(data$KWh))+min(data$KWh)*

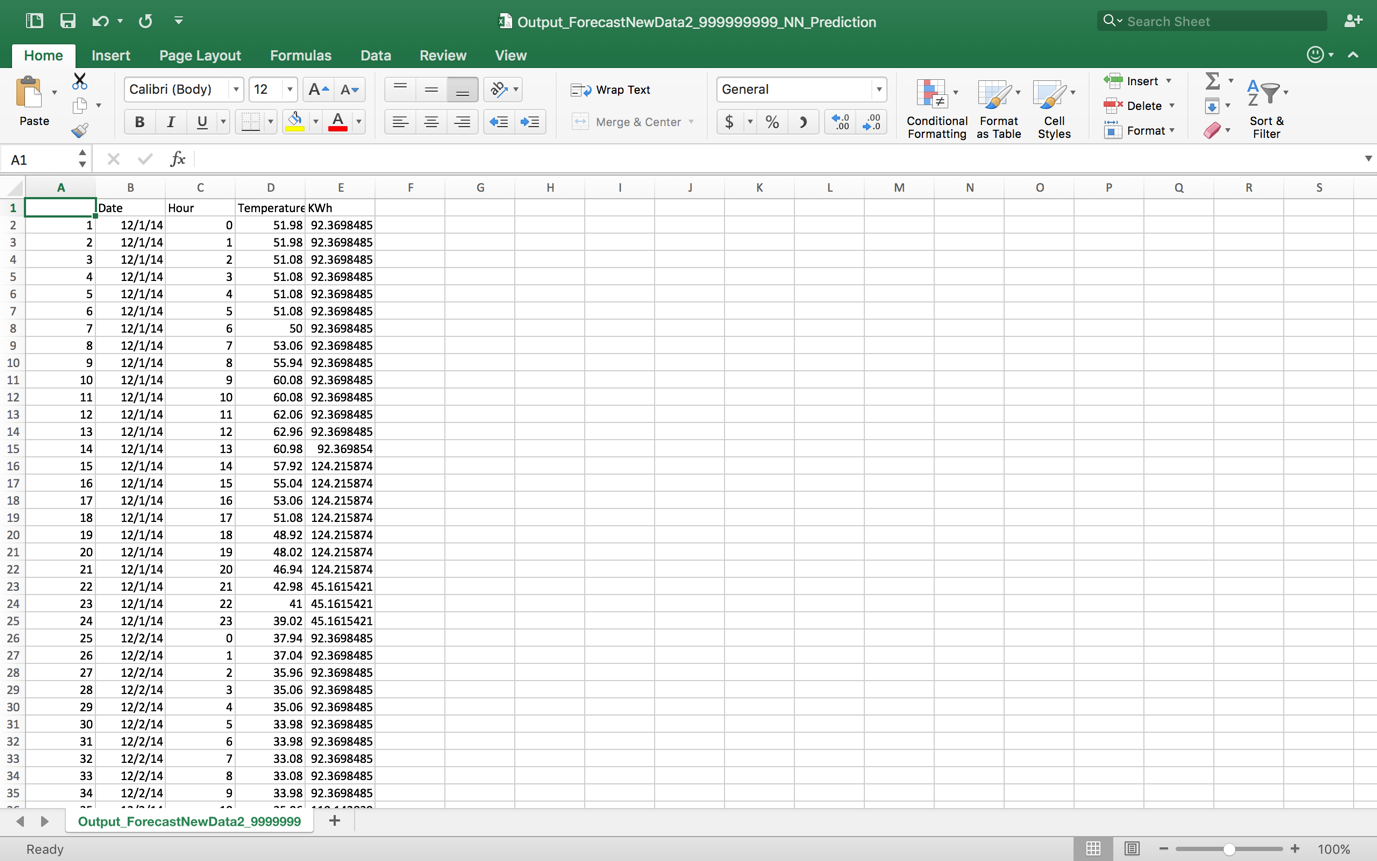
*View(pr.nn\_forecast)*

*View(firstnewforecast)*

*Forecast\_KWh<-cbind(firstnewforecast[1],firstnewforecast[5],firstnewforecast[9],KWh= pr.nn\_forecast)*

*View(Forecast\_KWh)*

*write.csv(Forecast\_KWh, file="ForecastNewData2\_NN\_Prediction.csv")*



3. Classification

a) Code and Regression output files in the format **ClassificationPerformanceMetrics.csv**

b) Output from your forecast script in the format **forecastOutput\_<Account No>\_neuralNetworkClassification.csv** and **forecastOutput\_<Account No>\_ClassificationTree.csv**

**LOGISTIC REGRESSION:**

1. Read hourly data file

**data<- read.csv(file.choose(),header = T)**

**head(data)**

**summary(data)**

**df<-data.frame(data)**

1. Mean of KWh column

**mean\_KWh<- mean(df$KWh)**

**View(mean\_KWh)**

1. Create new column KWh\_Class on the basis of condition df$KWh>mean

**KWh\_Class<-ifelse(df$KWh>mean\_KWh,"Above\_Normal","Optimal")**

**df\_final<-data.frame(df,KWh\_Class)**

1. Create new column abs with 1 and 0 values

**View(df\_final)**

**a <-data.frame(df\_final)**

**data(a,package="AER")**

**df\_final["abs"]<-NA**

**df\_final$abs[df\_final$KWh\_Class == "Above\_Normal"]<-1**

**df\_final$abs[df\_final$KWh\_Class == "Optimal"]<-0**

**df\_final$abs<-factor(df\_final$abs,level=c(0,1),labels=c("No","Yes"))**

**table(df\_final$abs)**

1. Divide the data into train and test

**smp\_size<-floor(0.75\*nrow(df\_final))**

**set.seed(123)**

**train\_ind<-sample(seq\_len(nrow(df\_final)),size=smp\_size)**

**train <- df\_final[train\_ind,]**

**test <- df\_final[-train\_ind,]**

1. Build the model

**fit<-glm(abs~Temperature+hour+month+day+year+Peakhour+DayofWeek+WeekDay,data=train,family=binomial(link="logit"))**

**summary(fit)**

1. Predict the values using predict function.

**test.probs<-predict(fit,test,type='response')**

**pred<- rep("No",length(test.probs))**

**pred[test.probs>=0.5]<-"Yes"**

**install.packages("caret")**

**library(caret)**

1. Create the confusion matrix and calculate the error percentage

**logisticregression\_table<-table(pred,test$KWh\_Class)**

**Error <- (((logisticregression\_table[1,2]) + (logisticregression\_table[2,1])) /((logisticregression\_table[2,1]) + (logisticregression\_table[1,2]) + (logisticregression\_table[1,1])+(logisticregression\_table[2,2])))**

**ErrorFin <- Error\*100**

**logisticregression <-cbind("Error",ErrorFin)**

**Classification Tree**

1. Read hourly filled data file

**data<- read.csv(file.choose(),header = T)**

**head(data)**

**summary(data)**

**df<-data.frame(data)**

1. Same as step 2 and 3 of logistic regression

**mean\_KWh<- mean(df$KWh)**

**View(mean\_KWh)**

**KWh\_Class<-ifelse(df$KWh>mean\_KWh,"Above\_Normal","Optimal")**

**df\_final<-data.frame(df,KWh\_Class)**

**View(df\_final)**

1. Build the tree model

**tree <- tree(KWh\_Class~Temperature+hour+month+day+year+Peakhour+DayofWeek+WeekDay,df\_final)**

**summary(tree)**

**plot(tree)**

**text(tree,pretty=0)**

1. Partition data into training and testing data and predict the values

**set.seed(2)**

**train <- sample(1:nrow(df\_final),200)**

**df\_final.test<-df\_final[-train,]**

**KWh\_Class.test <- KWh\_Class[-train]**

**tree.train<- tree(KWh\_Class~Temperature+hour+month+day+year+Peakhour+DayofWeek+WeekDay,df\_final,subset=train)**

**summary(tree.train)**

**tree.pred = predict(tree.train,df\_final.test,type="class")**

**classification\_tree<-table(tree.pred,KWh\_Class.test)**

**View(classification\_tree)**

1. Create the table and find the error

**classification\_tree<-table(tree.pred,KWh\_Class.test)**

**View(classification\_tree)**

**Error <- (((classification\_tree[1,2]) + (classification\_tree[2,1])) /((classification\_tree[2,1]) + (classification\_tree[1,2]) + (classification\_tree[1,1])+(classification\_tree[2,2])))**

**ErrorFin <- Error\*100**

**classification\_tree\_fin <-cbind("Error",ErrorFin)**

**Neural Network**

1. Follow first three steps of Logistic Regression

**data<- read.csv(file.choose(),header = T)**

**head(data)**

**summary(data)**

**df<-data.frame(data)**

**library(nnet)**

**mean\_KWh<- mean(df$KWh)**

**View(mean\_KWh)**

**KWh\_Class<-ifelse(df$KWh>mean\_KWh,"Above\_Normal","Optimal")**

**df\_final<-data.frame(df,KWh\_Class)**

**View(df\_final)**

1. Divide the data into train and test

**train<- sample(1:nrow(df\_final),5359)**

**test<-setdiff(1:nrow(df\_final),train)**

1. Build the model

**seedsANN = nnet(class.ind(KWh\_Class)~Temperature+hour+month+day+year+Peakhour+DayofWeek+WeekDay, df\_final[train,], size=10, softmax=TRUE)**

1. Predict the values and create the confusion matrix

**predict(seedsANN,df\_final[train,-12], type="class")**

**output<-table(predict(seedsANN, df\_final[test,-12], type="class"),df\_final[test,]$KWh\_Class)**

**REGRESSION MODEL**

Write into csv file

Run the model

Measure of predictive accuracy

**Build the linear regression model**

Split the data into train and test

Use the Regression Model in prediction Module

Split the data into training and testing

Use the logistic Regression Model in classification

Create a confusion matrix

Cutoff value=0.5

Build a logistic regression model

**CLASSIFICATION MODEL**