```
EnergyRating<-read.csv('C:/Users/Kanishk/Downloads/IE Courses/Data Mining/Project/Combine.csv')

EnergyRating<- EnergyRating[ , -c(1 , 2 , 3 , 4 , 5 , 6, 10, 11, 12 ,18, 20, 21 ,22 ,23 ,24 ,25)]#Removing unwanted
columns

library(dplyr)
#Filtering of Datasets
EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!Energy.Star.Score=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$Gross.Area..sq.ft.=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$Site.EUI..kBTU.sf.=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$GHG.Emissions..MTCO2e.=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$GHG.Intensity..kgCO2.sf.=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$Total.Site.Energy..kBTU.=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$X..Electricity=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$X..Gas=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$Water.Intensity..gal.sf.=='Not Available')

EnergyRating<-EnergyRating %>%
select(Gross.Area..sq.ft.,Site.EUI..kBTU.sf.,Energy.Star.Score,GHG.Emissions..MTCO2e.,GHG.Intensity..kgCO2.sf.,
                                    Total.Site.Energy..kBTU.,X..Electricity,X..Gas,Water.Intensity..gal.sf.,) %>%
filter(!EnergyRating$Gross.Area..sq.ft.=='Not Available')
#Converting Datasets to numeric data type
EnergyRating$Gross.Area..sq.ft.<-as.numeric(as.character(EnergyRating$Gross.Area..sq.ft.))
EnergyRating$Site.EUI..kBTU.sf.<-as.numeric(as.character(EnergyRating$Site.EUI..kBTU.sf.))
EnergyRating$Energy.Star.Score<-as.numeric(as.character(EnergyRating$Energy.Star.Score))
EnergyRating$GHG.Emissions..MTCO2e.<-as.numeric(as.character(EnergyRating$GHG.Emissions..MTCO2e.))
EnergyRating$GHG.Intensity..kgCO2.sf.<-as.numeric(as.character(EnergyRating$GHG.Intensity..kgCO2.sf.))
EnergyRating$Total.Site.Energy..kBTU.<-as.numeric(as.character(EnergyRating$Total.Site.Energy..kBTU.))
EnergyRating$X..Electricity<-as.numeric(as.character(EnergyRating$X..Electricity))
EnergyRating$X..Gas<-as.numeric(as.character(EnergyRating$X..Gas))
EnergyRating$Water.Intensity..gal.sf.<-as.numeric(as.character(EnergyRating$Water.Intensity..gal.sf.))
summary(EnergyRating)


#Visualize Missing Value in Matrix
library(dplyr)
library(wakefield)


library(Amelia)

missmap(EnergyRating)
## Visualize propotion Missing datasets
library(naniar)
gg_miss_var(EnergyRating)
#Removing all the na values
EnergyRating<-EnergyRating %>% filter(!is.na(Energy.Star.Score))
EnergyRating<-EnergyRating %>% filter(!is.na(Gross.Area..sq.ft.))
EnergyRating<-EnergyRating %>% filter(!is.na(Site.EUI..kBTU.sf.))
EnergyRating<-EnergyRating %>% filter(!is.na(GHG.Emissions..MTCO2e.))
EnergyRating<-EnergyRating %>% filter(!is.na(GHG.Intensity..kgCO2.sf.))
EnergyRating<-EnergyRating %>% filter(!is.na(Total.Site.Energy..kBTU.))
```

```
EnergyRating<-EnergyRating %>% filter(!is.na(X..Electricity))
EnergyRating<-EnergyRating %>% filter(!is.na(X..Gas))
EnergyRating<-EnergyRating %>% filter(!is.na(Water.Intensity..gal.sf.))
#Visualizing HeatMap of correlation Matrix
library(ggcorrplot)
library(reshape2)

qplot(x=Var1, y=Var2, data=melt(cor(EnergyRating)), fill=value, geom="tile")+
  geom_tile(color = "white")+
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name="Pearson\nCorrelation") +
  theme_minimal()+ # minimal theme
  theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                   size = 12, hjust = 1))+
  coord_fixed()
#Pre-Processing
StandardScale <- function(x){
  return((x-mean(x))/sd(x))
}

EnergyRating.norm<-EnergyRating
EnergyRating.norm[,c(1:2,4:9)]<-data.frame(lapply(EnergyRating[,c(1:2,4:9)],FUN =StandardScale))
train.index <- sample(c(1:dim(EnergyRating.norm)[1]), dim(EnergyRating.norm)[1]*0.6)
train.df <- EnergyRating.norm[train.index, ]
valid.index <- sample(c(1:dim(EnergyRating.norm)[1]), dim(EnergyRating.norm)[1]*0.4)
valid.df<-EnergyRating.norm[valid.index,]
summary(EnergyRating.norm)

#use k-fold cross validation and Random Forest Regression
library(randomForest)
set.seed(131)
library(caret)
k_10_fold<-trainControl(method = "repeatedcv",number=10,savePredictions = TRUE)
#Tunning the parameters for Random Forest Algorithm
model_fitted <-train(Energy.Star.Score ~Gross.Area..sq.ft.+Site.EUI..kBTU.sf.+GHG.Emissions..MTCO2e.
+GHG.Intensity..kgCO2.sf.+
                       Total.Site.Energy..kBTU.+X..Electricity+X..Gas+Water.Intensity..gal.sf., data=train.df,
family
                     = identity,trControl = k_10_fold, tuneLength =5)

print(model_fitted)


#XG Boosting Algorithm
set.seed(123)
model <- train(
  Energy.Star.Score ~Gross.Area..sq.ft.+Site.EUI..kBTU.sf.+GHG.Emissions..MTCO2e.+GHG.Intensity..kgCO2.sf.
+Total.Site.Energy..kBTU.
  +X..Electricity+X..Gas+Water.Intensity..gal.sf., data = train.df, method = "xgbTree",
  trControl = trainControl("cv", number = 10)
)
plot(varImp(model))
plot(model)
#Predicting the model
Predict_valid_rf<-predict(model_fitted,valid.df)
Predict_valid_xgb<-predict(model,valid.df)
#Result Interpretation
library(forecast)
accuracy(Predict_valid_rf,valid.df$Energy.Star.Score) #Random_Forest_Regression
accuracy(Predict_valid_xgb,valid.df$Energy.Star.Score) #XG Gradient Boosting Algorithm
#Lift Charts
library(gains)
gain <- gains(valid.df$Energy.Star.Score[!is.na(Predict_valid_rf)], Predict_valid_rf[!is.na(Predict_valid_rf)])

rating <- valid.df$Energy.Star.Score[!is.na(valid.df$Energy.Star.Score)]
plot(c(0,gain$cume.pct.of.total*sum(rating))~c(0,gain$cume.obs),
     xlab="# cases", ylab="Cumulative Price", main="Lift Chart for Random Forest", type="l")
lines(c(0,sum(rating))~c(0,dim(valid.df)[1]), col="red", lty=3)
##Decile Chart
barplot(gain$mean.resp/mean(rating), names.arg = gain$depth,
        xlab = "Percentile", ylab = "Mean Response", main = "Decile-wise lift chart",col=c("red"))
#Lift Charts
library(gains)
gain <- gains(valid.df$Energy.Star.Score[!is.na(Predict_valid_xgb)], Predict_valid_xgb[!is.na(Predict_valid_xgb)])
rating <- valid.df$Energy.Star.Score[!is.na(valid.df$Energy.Star.Score)]
plot(c(0,gain$cume.pct.of.total*sum(rating))~c(0,gain$cume.obs),
     xlab="# cases", ylab="Cumulative Price", main="Lift Chart for Boosted Tree", type="l")
lines(c(0,sum(rating))~c(0,dim(valid.df)[1]), col="gray", lty=2)

barplot(gain$mean.resp/mean(rating), names.arg = gain$depth,
        xlab = "Percentile", ylab = "Mean Response", main = "Decile-wise lift chart",col=c("red"))
accuracy(Predict_valid_xgb,valid.df$Energy.Star.Score)
```