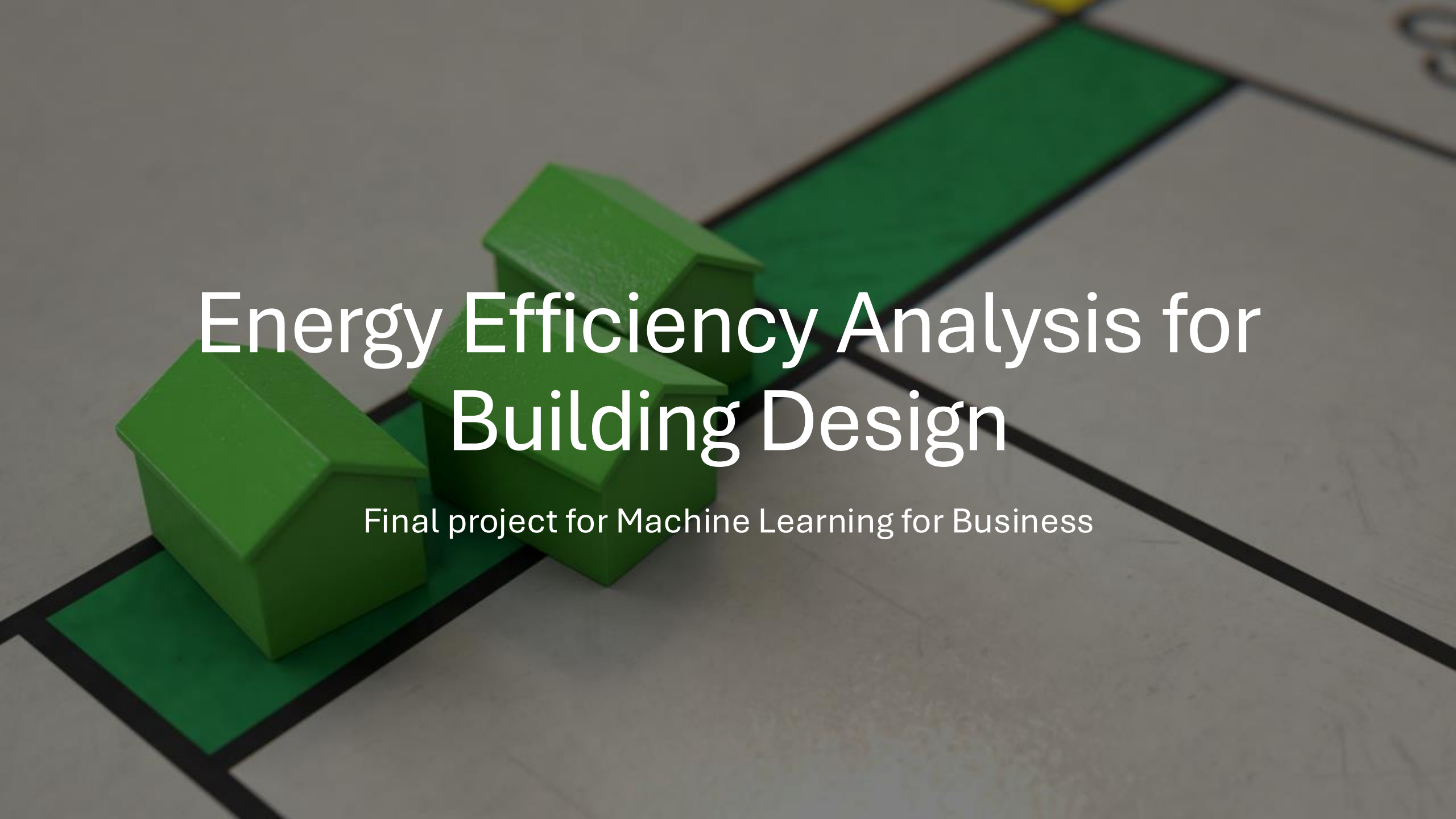


Energy Efficiency Analysis for Building Design

Final project for Machine Learning for Business

TEAM 2

Shuping Wang

Yeshwanth Chalapathi Sureshkumar

Bhavana Yachenahalli Channakeshava Murthy

Varun Sham Kumar Kannappanahalli Shamanna

Dataset overview

Dataset Description

- Buildings vary in terms of **glazing area**, **glazing area distribution**, **orientation**, and other structural parameters.
- The objective is to utilize the **eight features** to accurately predict the response variables.

VARIABLE	DATA	DATA TYPE
X1	Relative Compactness	Continuous
X2	Surface Area	Continuous
X3	Wall Area	Continuous
X4	Roof Area	Continuous
X5	Overall Height	Continuous
X6	Orientation	Categorical and Discrete
X7	Glazing Area	Continuous
X8	Glazing Area Distribution	Categorical and Discrete
Y1	Heating Load	Continuous
Y2	Cooling Load	Continuous

Data cleaning

- **Handling X6: Orientation**

- The original variable had four categories:

  2 = North, 3 = East, 4 = South, 5 = West.

- Following the rule for dummy variable creation *(n categories → n-1 dummies)*, I excluded **West (5)** as the base category and created dummy variables for **North, East, and South using Excel.**

- **Handling X8: Glazing Area Distribution**

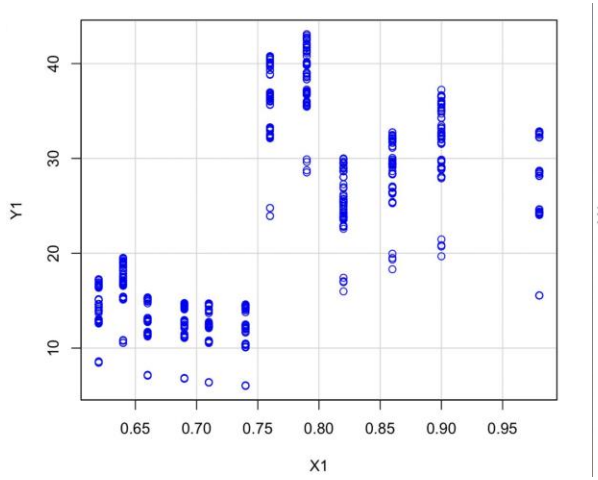
- The original variable had six categories:

  0 = Unknown, 1 = Uniform, 2 = North, 3 = East, 4 = South, 5 = West.

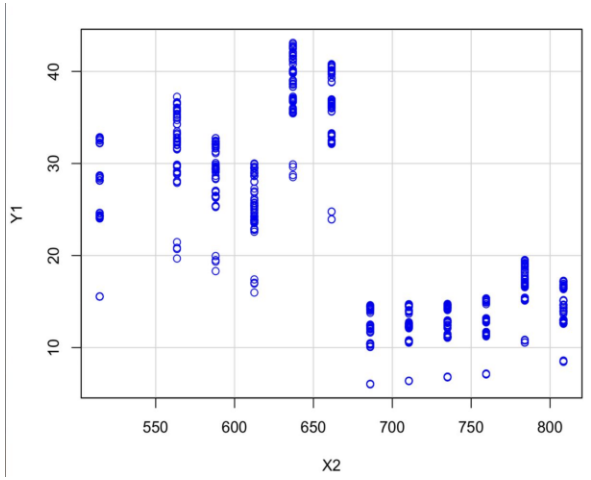
- Similarly, I chose **Unknown (0)** as the base category and created dummy variables for **Uniform, North, East, South, and West.**

1. Visualization

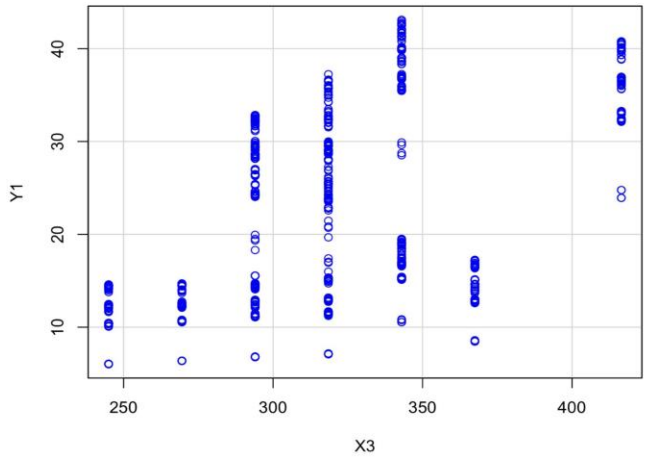
Scatter Plot relationships between the predictors (X1–X8) vs Heating Pad(y1)



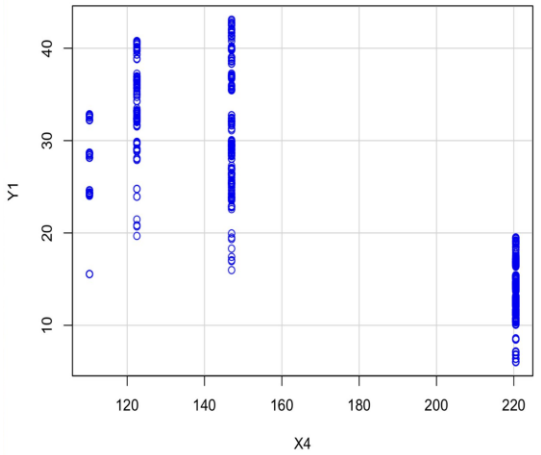
We observe an upward trend



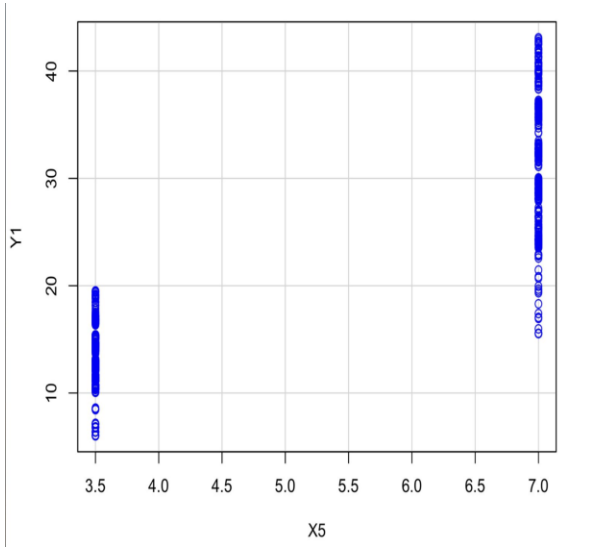
We observe a downward trend



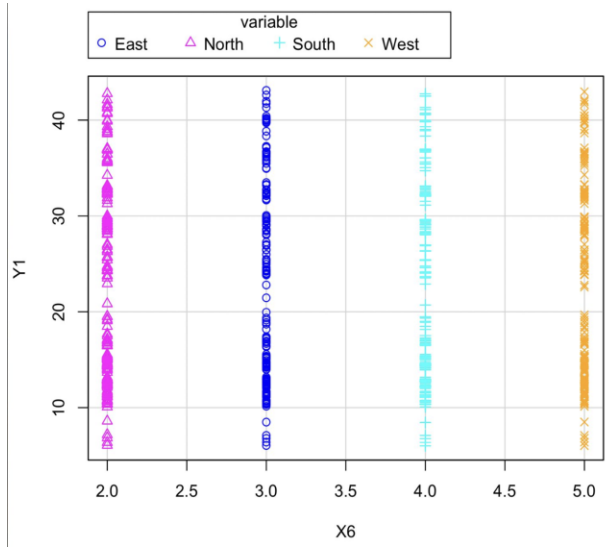
We observe an upward trend



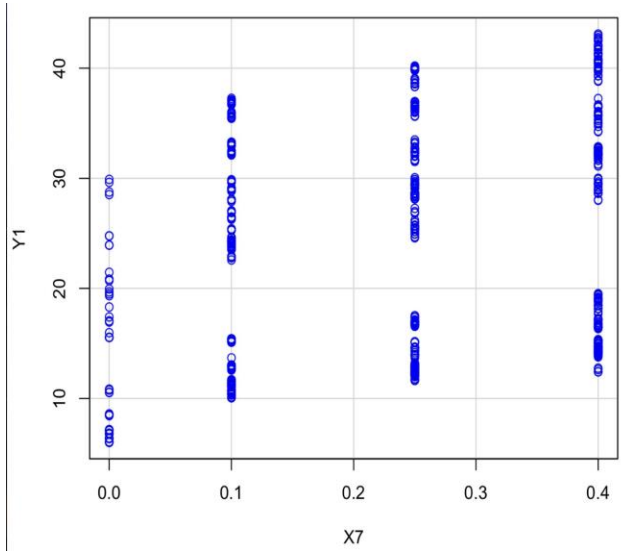
We observe a downard trend



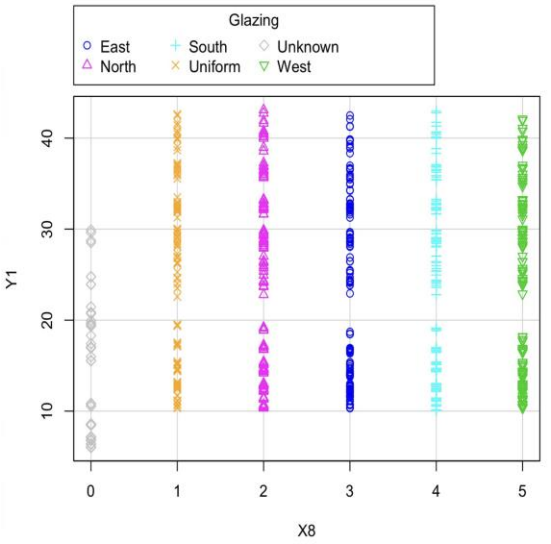
We observe an upward trend



Change in y does not affect x for each of the variable

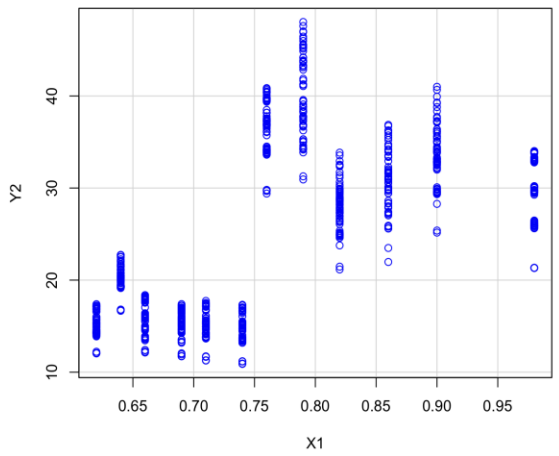


We observe an upward trend

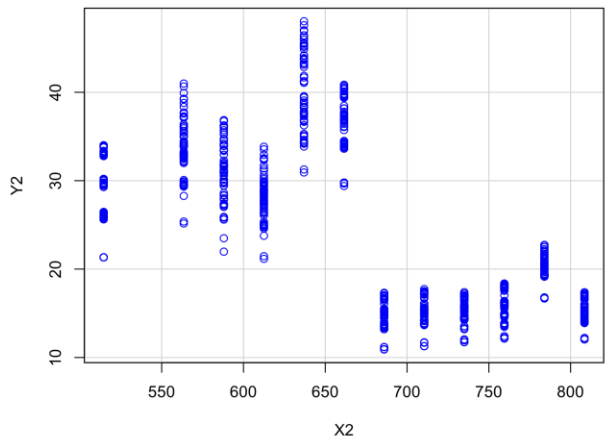


Change in y does not affect x for each of the variable

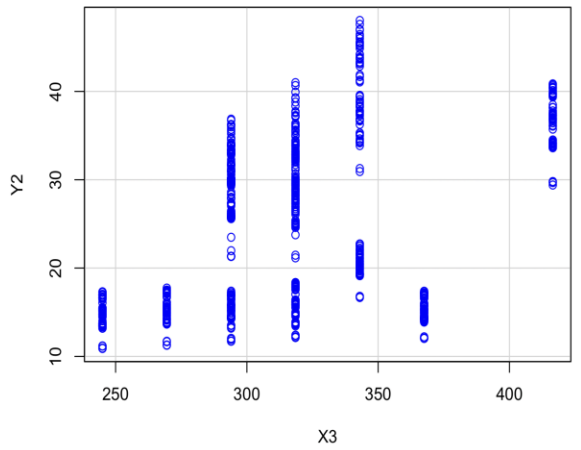
Scatter Plot relationships between the predictors (X1–X8) vs Cooling Pad(y2)



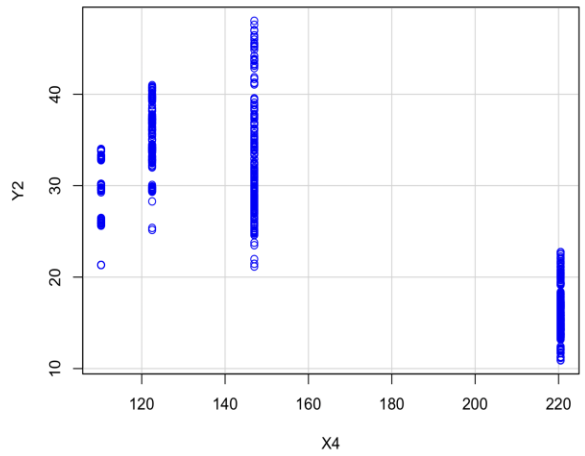
We observe an upward trend



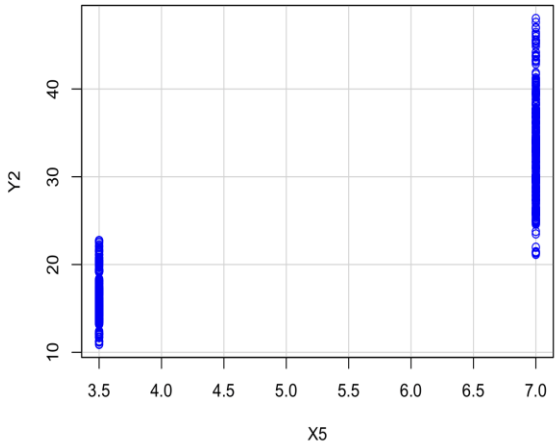
We observe a downard trend



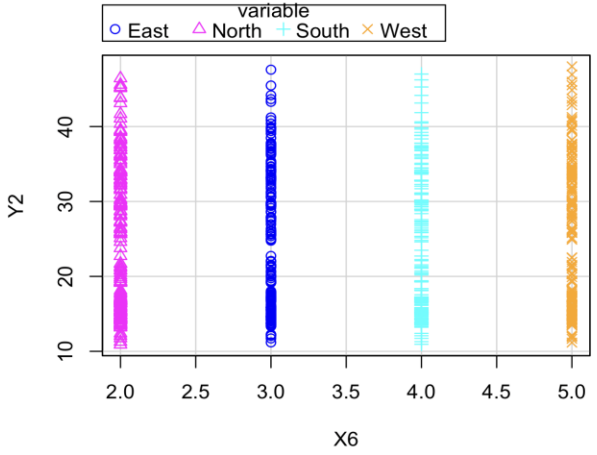
We observe an upward trend



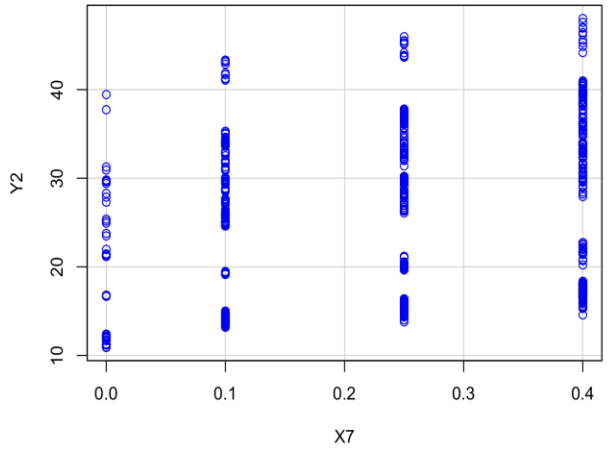
We observe a downard trend



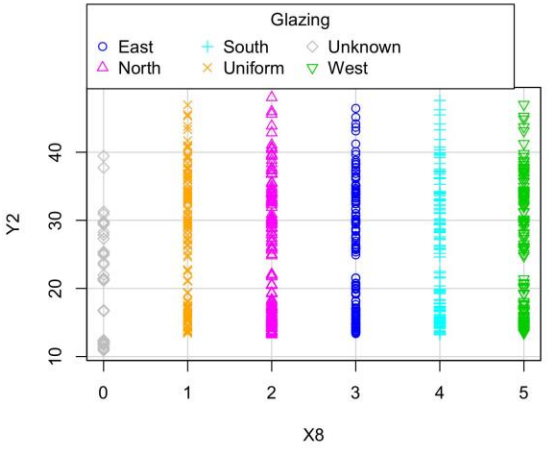
We observe an upward trend



Change in y does not affect x for each of the variable

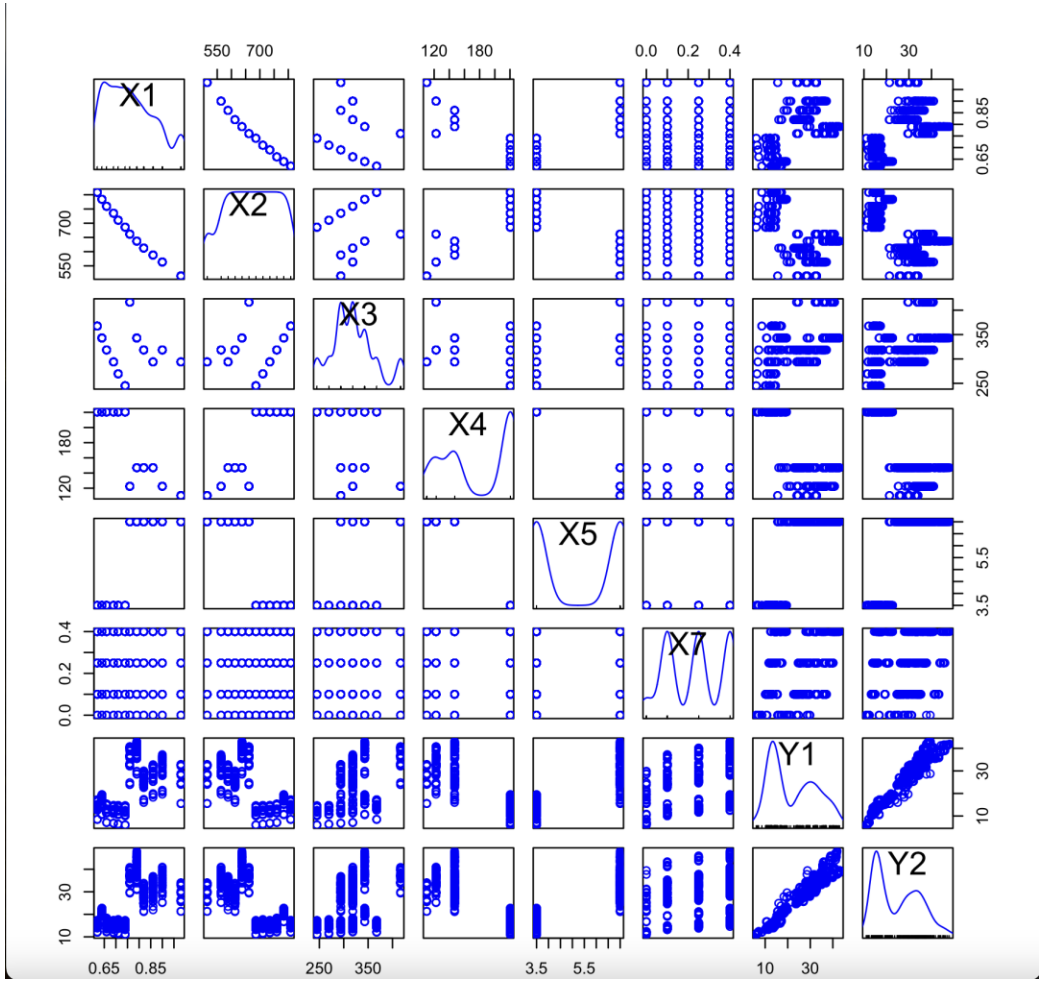


We observe an upward trend



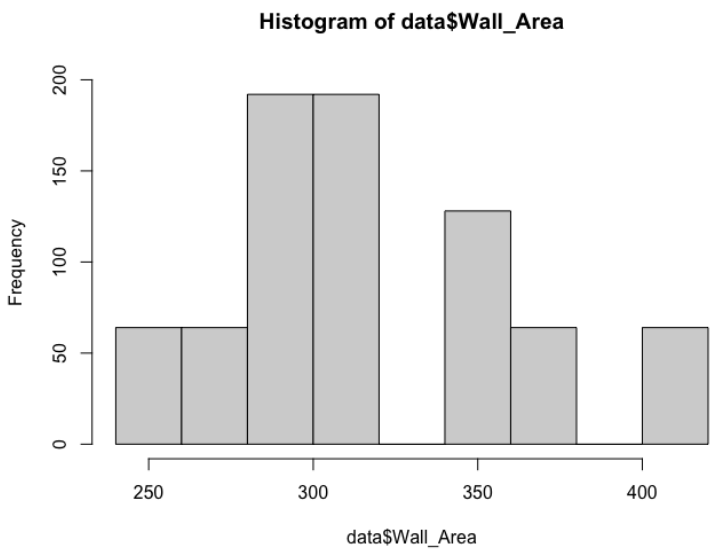
Change in y does not affect x for each of the variable

Scatter Plot Matrix

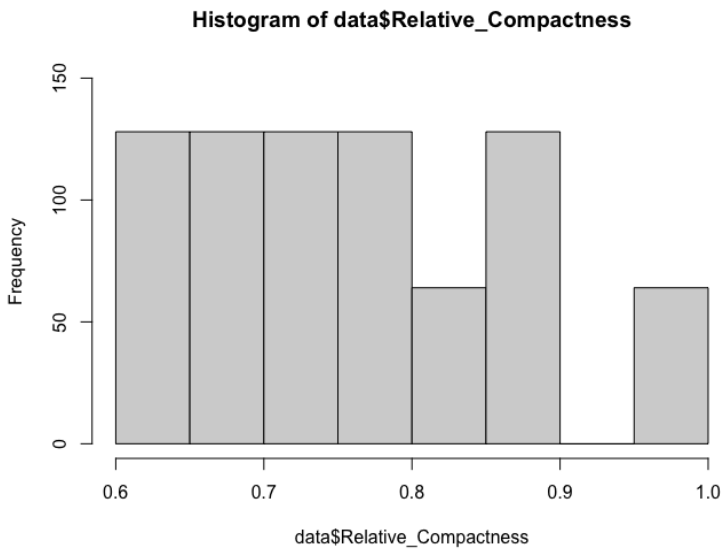


Scatter Plot matrix for variables x1,x2,x3,x4,x5,x7,x8 and y1 and y2

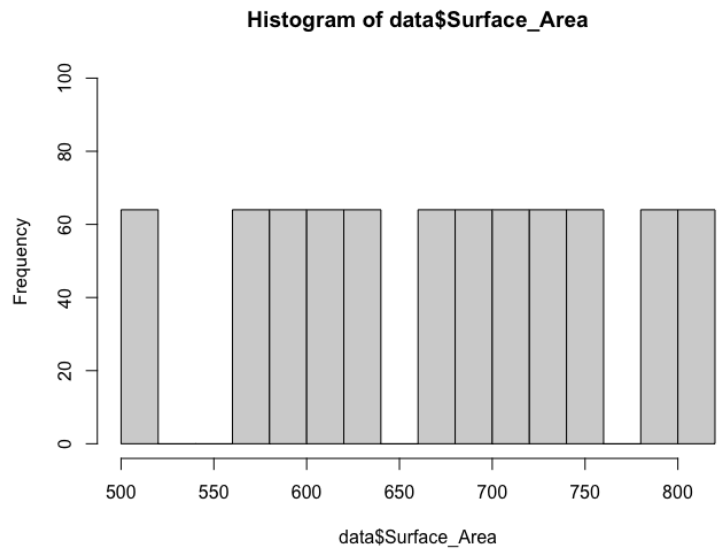
Histograms



Majority of the values are concentrated around 300-350 freq units

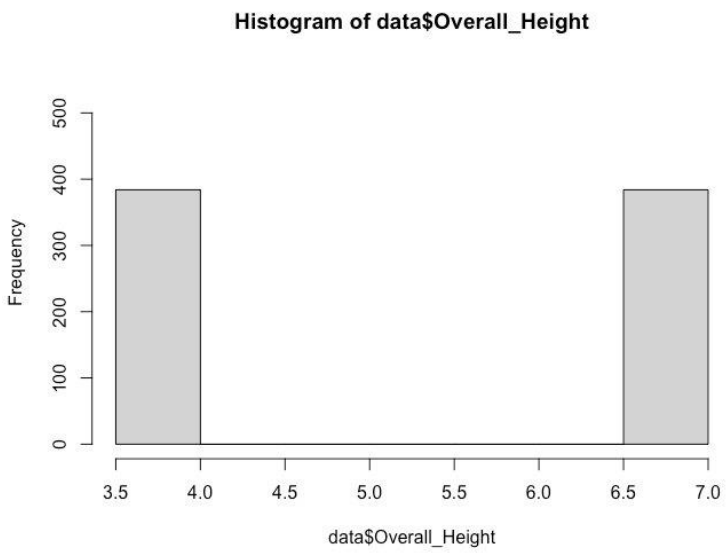


Most values are at 125 freq units

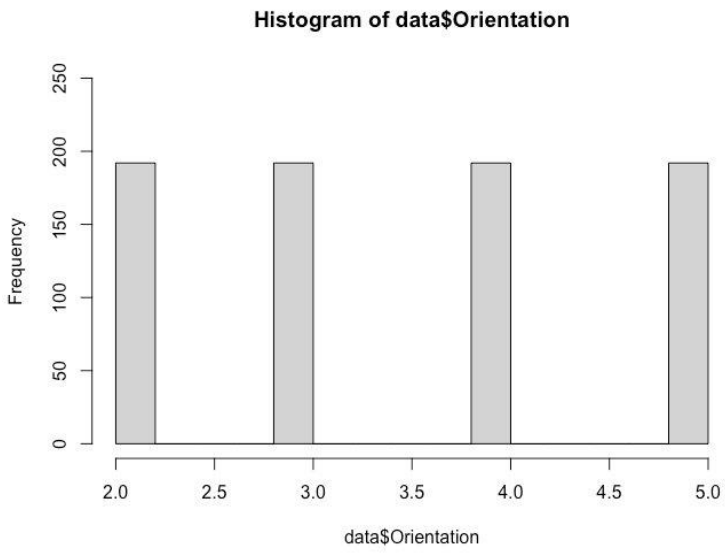


All values are slightly above 60 freq units

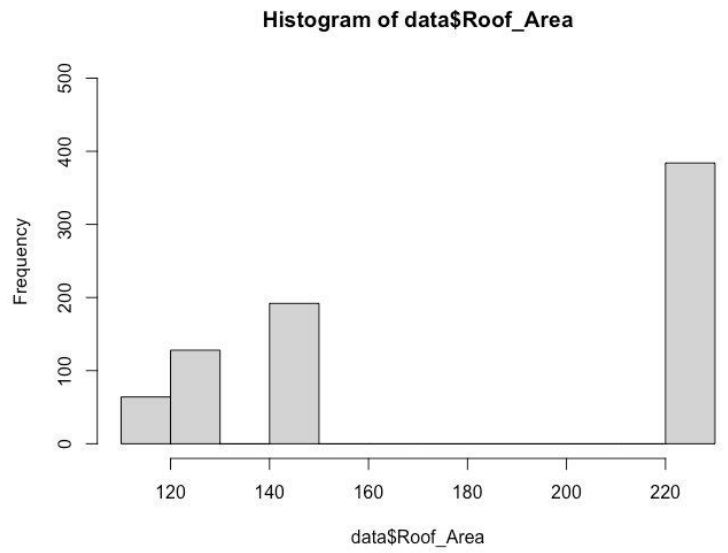
Histograms



Both values are at around 400 freq units

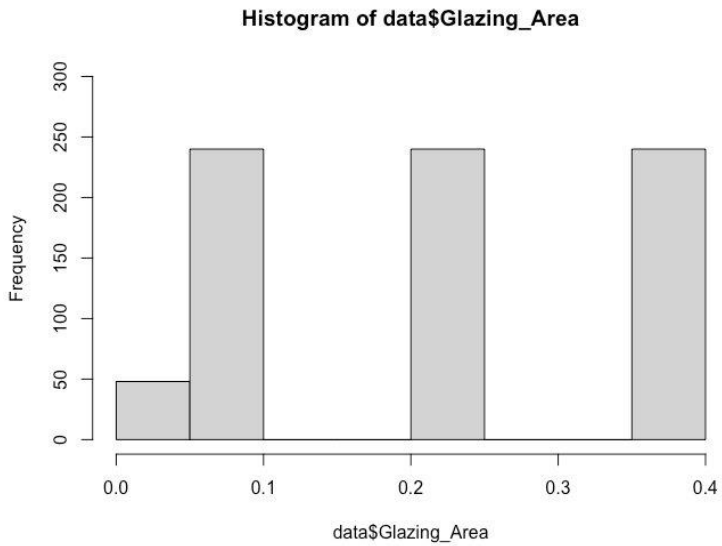


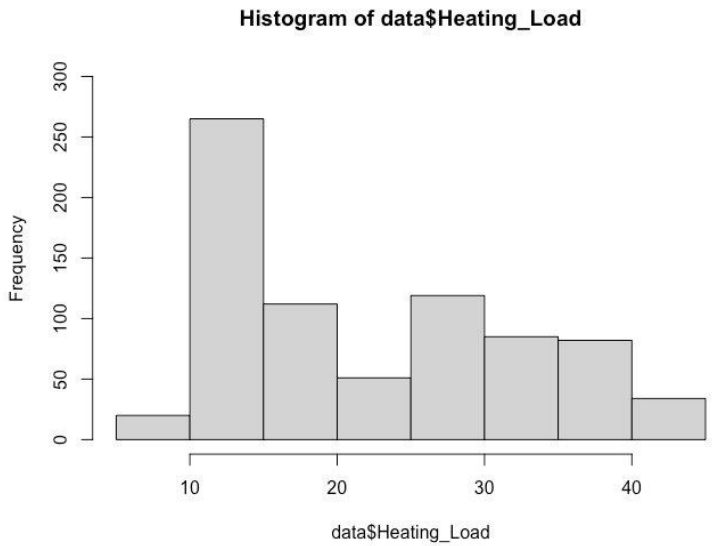
All values are around 400 freq units

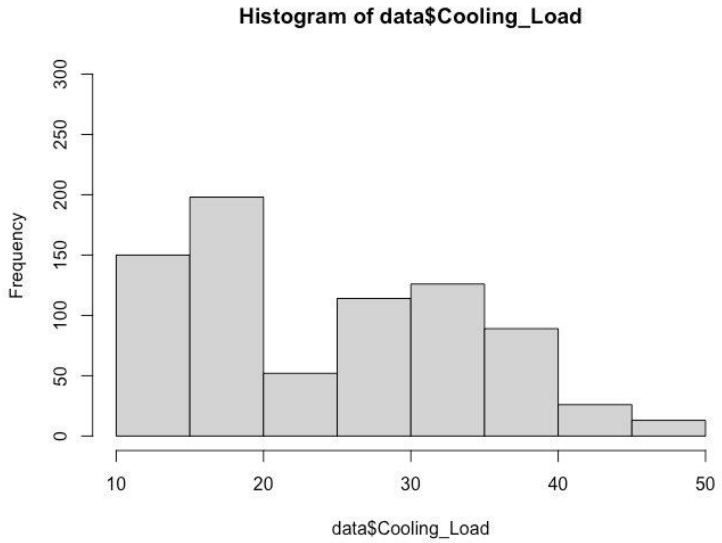


The freq increases gradually

Histograms


Histogram of data$Glazing_Area
Frequency
0
50
100
150
200
250
300
0.0
0.1
0.2
0.3
0.4
data$Glazing_Area


Histogram of data$Heating_Load
Frequency
0
50
100
150
200
250
300
10
20
30
40
data$Heating_Load


Histogram of data$Cooling_Load
Frequency
0
50
100
150
200
250
300
10
20
30
40
50
data$Cooling_Load


Histogram of data$Glazing_Area_Distribution
Frequency
0
50
100
150
200
0
1
2
3
4
5
data$Glazing_Area_Distribution

2. Statistical Analysis

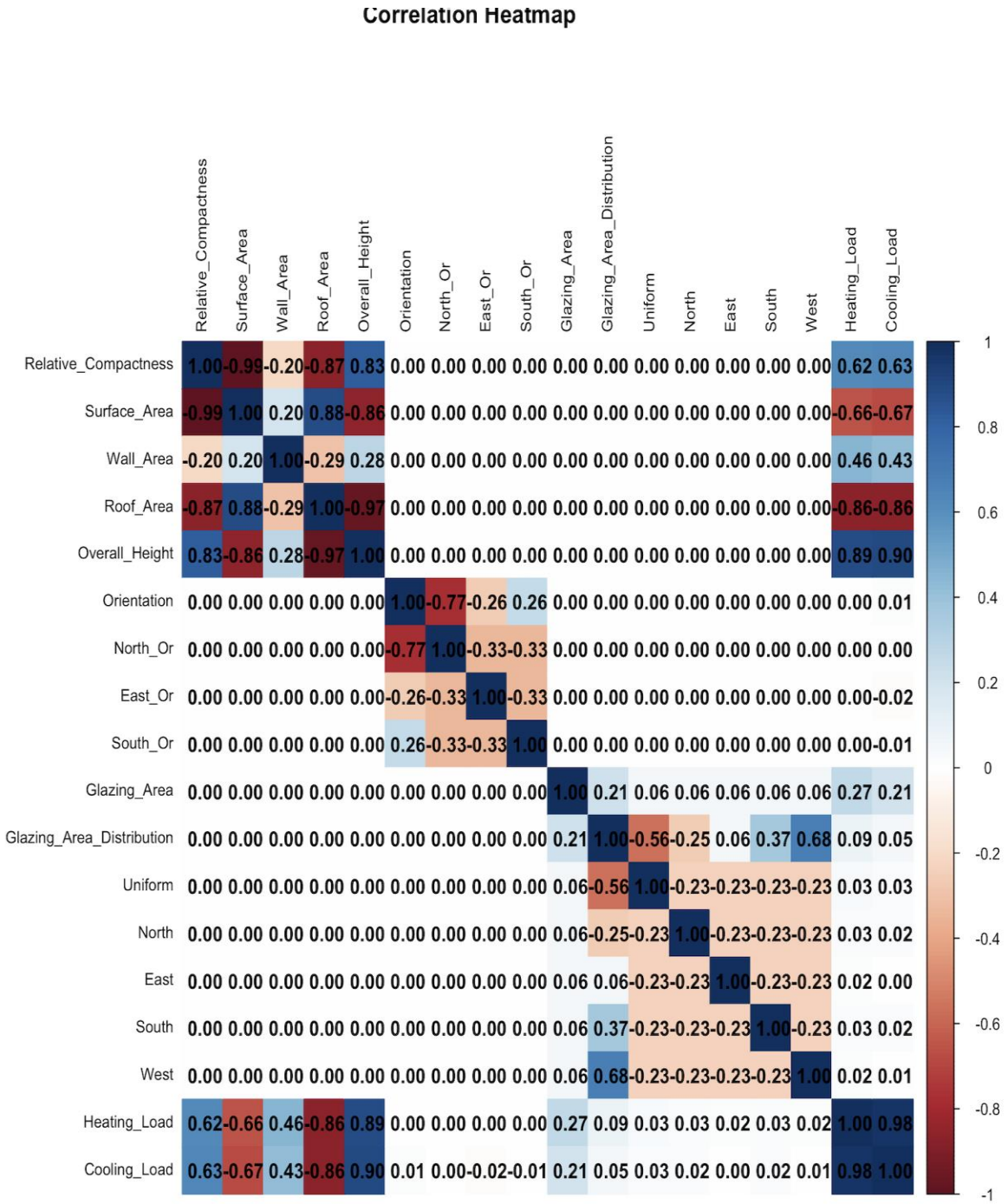
Descriptive Statistics

- **Let's interpret some of the key variables:**

  - **<u>Relative Compactness:</u>** Ranges from **0.62 to 0.98**, with a mean of approximately **0.76.**
  - ➢ Indicates variability in building compactness.
  - **<u>Surface Area:</u>** Varies from **514.5 to 808.5** with a mean of approximately **671.7**
  - ➢ Highlights diverse building sizes.
  - **<u>Cooling Load:</u>** The dependent variable of interest, ranging from **10.90 to 48.03** with a mean of about **24.59**
  - ➢ Reflects variability in energy requirements for heating across buildings.

```
> library(readxl)
> EnergyEfficiencyfinal <- read_excel("Desktop/Humana /EnergyEfficiencyfinal.xlsx")
> View(EnergyEfficiencyfinal)
> summary(EnergyEfficiencyfinal)
 Relative_Compactness  Surface_Area     Wall_Area       Roof_Area     Overall_Height  Orientation
 Min.   :0.6200       Min.   :514.5   Min.   :245.0   Min.   :110.2   Min.   :3.50   Min.   :2.00
 1st Qu.:0.6825       1st Qu.:606.4   1st Qu.:294.0   1st Qu.:140.9   1st Qu.:3.50   1st Qu.:2.75
 Median :0.7500       Median :673.8   Median :318.5   Median :183.8   Median :5.25   Median :3.50
 Mean   :0.7642       Mean   :671.7   Mean   :318.5   Mean   :176.6   Mean   :5.25   Mean   :3.50
 3rd Qu.:0.8300       3rd Qu.:741.1   3rd Qu.:343.0   3rd Qu.:220.5   3rd Qu.:7.00   3rd Qu.:4.25
 Max.   :0.9800       Max.   :808.5   Max.   :416.5   Max.   :220.5   Max.   :7.00   Max.   :5.00
    North_Or       East_Or        South_Or     Glazing_Area    Glazing_Area_Distribution
 Min.   :0.00   Min.   :0.00   Min.   :0.00   Min.   :0.0000   Min.   :0.000
 1st Qu.:0.00   1st Qu.:0.00   1st Qu.:0.00   1st Qu.:0.1000   1st Qu.:1.750
 Median :0.00   Median :0.00   Median :0.00   Median :0.2500   Median :3.000
 Mean   :0.25   Mean   :0.25   Mean   :0.25   Mean   :0.2344   Mean   :2.812
 3rd Qu.:0.25   3rd Qu.:0.25   3rd Qu.:0.25   3rd Qu.:0.4000   3rd Qu.:4.000
 Max.   :1.00   Max.   :1.00   Max.   :1.00   Max.   :0.4000   Max.   :5.000
    Uniform           North             East             South             West          Heating_Load
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   : 6.01
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:12.99
 Median :0.0000   Median :0.0000   Median :0.0000   Median :0.0000   Median :0.0000   Median :18.95
 Mean   :0.1875   Mean   :0.1875   Mean   :0.1875   Mean   :0.1875   Mean   :0.1875   Mean   :22.31
 3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:31.67
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :43.10
  Cooling_Load
 Min.   :10.90
 1st Qu.:15.62
 Median :22.08
 Mean   :24.59
 3rd Qu.:33.13
 Max.   :48.03
>
```

2(b) Correlation



- **Relative_Compactness and Surface_Area**:
- **Strong negative correlation** (-0.99): As the relative compactness of a building increases, its surface area decreases, and vice versa.
- **Overall_Height and Roof_Area**:
- **Very strong negative correlation** (-0.97): Taller buildings generally have less roof area.
- **Heating_Load**:
- **Strong positive correlation** with Relative_Compactness (0.62): Buildings with higher relative compactness are associated with higher heating loads.
- **Very Strong positive correlation** with Overall_Height (0.89): Taller buildings tend to have higher heating loads.
- **Strong negative correlation** with Surface_Area (-0.66): Buildings with larger surface areas tend to have lower heating loads.
- **Cooling_Load**:
- **Strong positive correlation** with Relative_Compactness (0.63): More compact buildings are linked to higher cooling loads.
- **Moderate positive correlation** with Wall_Area (0.43): A larger wall area contributes to higher cooling loads.
- **Very strong positive correlation** with Overall_Height (0.90): Taller buildings have significantly higher cooling loads.
- **Strong negative correlation** with Surface_Area (-0.67): Larger surface areas correspond to lower cooling loads.

Linear Regression Heating_Load

- **R-Squared: 0.9228**
- Indicates that approximately **92.28%** of the variance in the Heating load can be explained by the model's predictors.

- **Positive Correlation**
- East, North, South , West
- Glazing Area
- Overal Height
- Uniform

As these variables increases, heating load also increases.
Example: Larger glazing areas lead to higher heating loads.

- **Negative Correlation**
- Relative Compactness
- Roof Area
- Surface Area

As the variable increases, heating load decreases.
Example: Higher relative compactness reduces the heating load due to better energy efficiency.

```
> summary(RegModel.2)

Call:
lm(formula = Heating_Load ~ East + East_Or + Glazing_Area + North +
    North_Or + Overall_Height + Relative_Compactness + Roof_Area +
    South + South_Or + Surface_Area + Uniform + Wall_Area + West,
    data = EnergyEfficiencyfinal)

Residuals:
    Min      1Q  Median      3Q     Max
-7.1237 -1.4299 -0.2555  1.2972  7.4342

Coefficients: (1 not defined because of singularities)
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         81.12397   18.18928   4.460 9.44e-06 ***
East                 4.18300    0.51372   8.143 1.60e-15 ***
East_Or              0.10531    0.28618   0.368   0.7130
Glazing_Area        16.84833    0.85324  19.746  < 2e-16 ***
North                4.43599    0.51372   8.635  < 2e-16 ***
North_Or             0.03750    0.28618   0.131   0.8958
Overall_Height       4.16995    0.32298  12.911  < 2e-16 ***
Relative_Compactness -64.77343   9.83256  -6.588 8.38e-11 ***
Roof_Area           -0.12163    0.01271  -9.573  < 2e-16 ***
South                4.38821    0.51372   8.542  < 2e-16 ***
South_Or            -0.01549    0.28618  -0.054   0.9569
Surface_Area        -0.02648    0.01221  -2.169   0.0304 *
Uniform              4.52765    0.51372   8.814  < 2e-16 ***
Wall_Area                 NA         NA      NA       NA
West                 4.18244    0.51372   8.142 1.61e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.804 on 754 degrees of freedom
Multiple R-squared:  0.9241,    Adjusted R-squared:  0.9228
F-statistic:   706 on 13 and 754 DF,  p-value: < 2.2e-16
```

Significant Predictors:

Linear Regression Cooling_Load

- **R-Squared: 0.8885** Indicates that approximately **88.85%** of the variance in the cooling load can be explained by the model's predictors.

- **Positive Correlation**
- East, North, South , West
- Glazing Area
- Overal Height
- Uniform

As the variable increases, cooling load also increases.
Example: Larger glazing areas lead to higher cooling loads.

- **Negative Correlation**
- Relative Compactness
- East_Or
- Roof Area
- Surface Area

As the variable increases, cooling load decreases.
Example: Higher relative compactness reduces the cooling load due to better energy efficiency.

```
> summary(RegModel.3)

Call:
lm(formula = Cooling_Load ~ East + East_Or + Glazing_Area + North +
    North_Or + Overall_Height + Relative_Compactness + Roof_Area +
    South + South_Or + Surface_Area + Uniform + Wall_Area + West,
    data = EnergyEfficiencyfinal)

Residuals:
    Min      1Q  Median      3Q     Max
-7.7082 -1.6996 -0.2814  1.4302 11.0365

Coefficients: (1 not defined because of singularities)
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          96.71918   20.60428   4.694 3.18e-06 ***
East                  1.63997    0.58192   2.818 0.004956 **
East_Or              -0.64109    0.32418  -1.978 0.048340 *
Glazing_Area         13.25292    0.96652  13.712  < 2e-16 ***
North                 1.97740    0.58192   3.398 0.000714 ***
North_Or             -0.34911    0.32418  -1.077 0.281865
Overall_Height        4.28384    0.36586  11.709  < 2e-16 ***
Relative_Compactness -70.78771   11.13804  -6.355 3.59e-10 ***
Roof_Area            -0.08936    0.01439  -6.209 8.80e-10 ***
South                 1.99566    0.58192   3.429 0.000638 ***
South_Or             -0.47333    0.32418  -1.460 0.144683
Surface_Area         -0.04356    0.01383  -3.151 0.001694 **
Uniform               2.16003    0.58192   3.712 0.000221 ***
Wall_Area                  NA         NA      NA       NA
West                  1.69552    0.58192   2.914 0.003678 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.176 on 754 degrees of freedom
Multiple R-squared:  0.8904,    Adjusted R-squared:  0.8885
F-statistic: 471.3 on 13 and 754 DF,  p-value: < 2.2e-16
```

Significant Predictors:

VIF Analysis

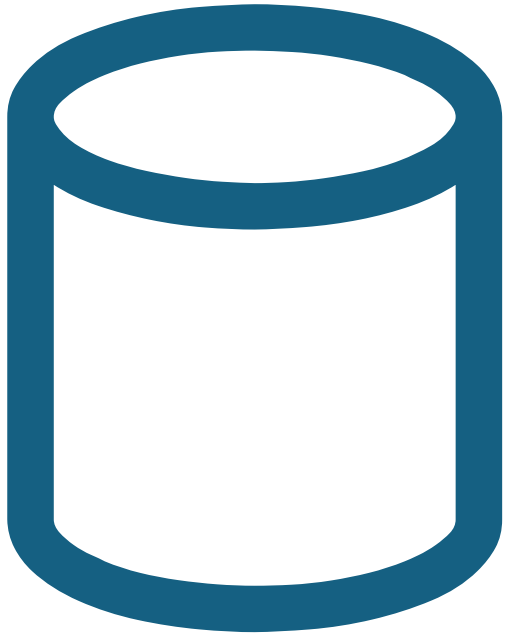
➢ **High Multicollinearity**:

•**Roof Area**: VIF = **32.12**.
•**Overall Height**: VIF = **31.20**.
•**Surface Area**: VIF = **112.77**.
•**Relative Compactness**: VIF = **105.5**.
•all have VIF values above **10** Indicates strong multicollinearity, requiring attention during model adjustment.

➢ **Low Multicollinearity**:

•Directional variables (**East**, **South**, **West**, and **North**) and **Glazing Area** all have VIF values below **5**, indicating minimal concern for multicollinearity.

```
> library(car)
> vif_values <- vif(RegModel.2_updated)
> print(vif_values)
                East         Glazing_Area                North       Overall_Height
            1.456897             1.043103             1.456897            31.205474
Relative_Compactness            Roof_Area                South          Surface_Area
          105.524054            32.124528             1.456897           112.774196
                West
            1.456897
>
>
```



3. Modeling Analysis

Assign categories for each Y-variable

A for highest quartile, B for second highest quartile, C for third highest quartile, D for lowest quartile

Assigning Categories for Heating Load

```
> data$HeatingLoadCategory <- cut(
+     data$Y1,
+     breaks = quantile(data$Y1, probs = seq(0, 1, 0.25), na.rm = TRUE),
+     labels = c("D", "C", "B", "A"),
+     include.lowest = TRUE
+ )
>
```

Assigning Categories for Cooling Load

```
> data$CoolingLoadCategory <- cut(
+     data$Y2,
+     breaks = quantile(data$Y2, probs = seq(0, 1, 0.25), na.rm = TRUE),
+     labels = c("D", "C", "B", "A"),
+     include.lowest = TRUE
+ )
>
```

Perceptrons

```r
# Define the perceptron algorithm
perceptron <- function(X, y, numEpochs) {
  w <- runif(ncol(X), -10, 10)  # Initialize weights
  for (epoch in 1:numEpochs) {
    for (i in 1:length(y)) {
      xi <- as.numeric(unlist(X[i, ]))
      if (sign(w %*% xi) != y[i]) {
        w <- w + y[i] * xi
      }
    }
  }
  return(w)
}

# Run the process 5 times for both heating and cooling
set.seed(42)  # For reproducibility
numRuns <- 5
numEpochs <- 10
heating_accuracies <- numeric(numRuns)
cooling_accuracies <- numeric(numRuns)

for (run in 1:numRuns) {
  # Split data into training (70%) and testing (30%)
  train_indices <- sample(nrow(data), 0.7 * nrow(data))
  train_data <- data[train_indices, ]
  test_data <- data[-train_indices, ]

  # For Heating
  X_heating <- train_data[, c("Relative_Compactness", "Surface_Area", "Wall_Area",
                              "Roof_Area", "Overall_Height", "North_Or", "East_Or",
                              "South_Or", "Glazing_Area", "Uniform", "North",
                              "East", "South", "West")]
  y_heating <- train_data$Heating_Class

  w_heating <- perceptron(X_heating, y_heating, numEpochs)

  X_heating_test <- test_data[, c("Relative_Compactness", "Surface_Area", "Wall_Area",
                                  "Roof_Area", "Overall_Height", "North_Or", "East_Or",
                                  "South_Or", "Glazing_Area", "Uniform", "North",
                                  "East", "South", "West")]
  y_heating_test <- test_data$Heating_Class
  y_heating_pred <- sign(as.matrix(X_heating_test) %*% w_heating)

  heating_accuracies[run] <- mean(y_heating_pred == y_heating_test)
```

```r
  # For Cooling
  X_cooling <- train_data[, c("Relative_Compactness", "Surface_Area", "Wall_Area",
                              "Roof_Area", "Overall_Height", "North_Or", "East_Or",
                              "South_Or", "Glazing_Area", "Uniform", "North",
                              "East", "South", "West")]
  y_cooling <- train_data$Cooling_Class

  w_cooling <- perceptron(X_cooling, y_cooling, numEpochs)

  X_cooling_test <- test_data[, c("Relative_Compactness", "Surface_Area", "Wall_Area",
                                  "Roof_Area", "Overall_Height", "North_Or", "East_Or",
                                  "South_Or", "Glazing_Area", "Uniform", "North",
                                  "East", "South", "West")]
  y_cooling_test <- test_data$Cooling_Class
  y_cooling_pred <- sign(as.matrix(X_cooling_test) %*% w_cooling)

  cooling_accuracies[run] <- mean(y_cooling_pred == y_cooling_test)
}

# Print accuracies
cat("Heating Accuracies over 5 runs:\n", heating_accuracies, "\n")
cat("Cooling Accuracies over 5 runs:\n", cooling_accuracies, "\n")
```

```
+ }
>
> # Print accuracies
> cat("Heating Accuracies over 5 runs:\n", heating_accuracies, "\n")
Heating Accuracies over 5 runs:
 0.978355 0.965368 0.969697 0.974026 0.982684
> cat("Cooling Accuracies over 5 runs:\n", cooling_accuracies, "\n")
Cooling Accuracies over 5 runs:
 0.982684 0.974026 0.969697 0.982684 0.987013
```

Mode Type	Accuracy-Model 1	Accuracy-Model 2	Accuracy-Model 3	Accuracy-Model 4	Accuracy-Model 5	NIR	P-value
Heating Load	0.978355	0.965368	0.969697	0.974026	0.982684	0.5368	0.00
Cooling Load	0.982684	0.974026	0.969697	0.982684	0.987013	0.5368	0.00

Support Vector Machines

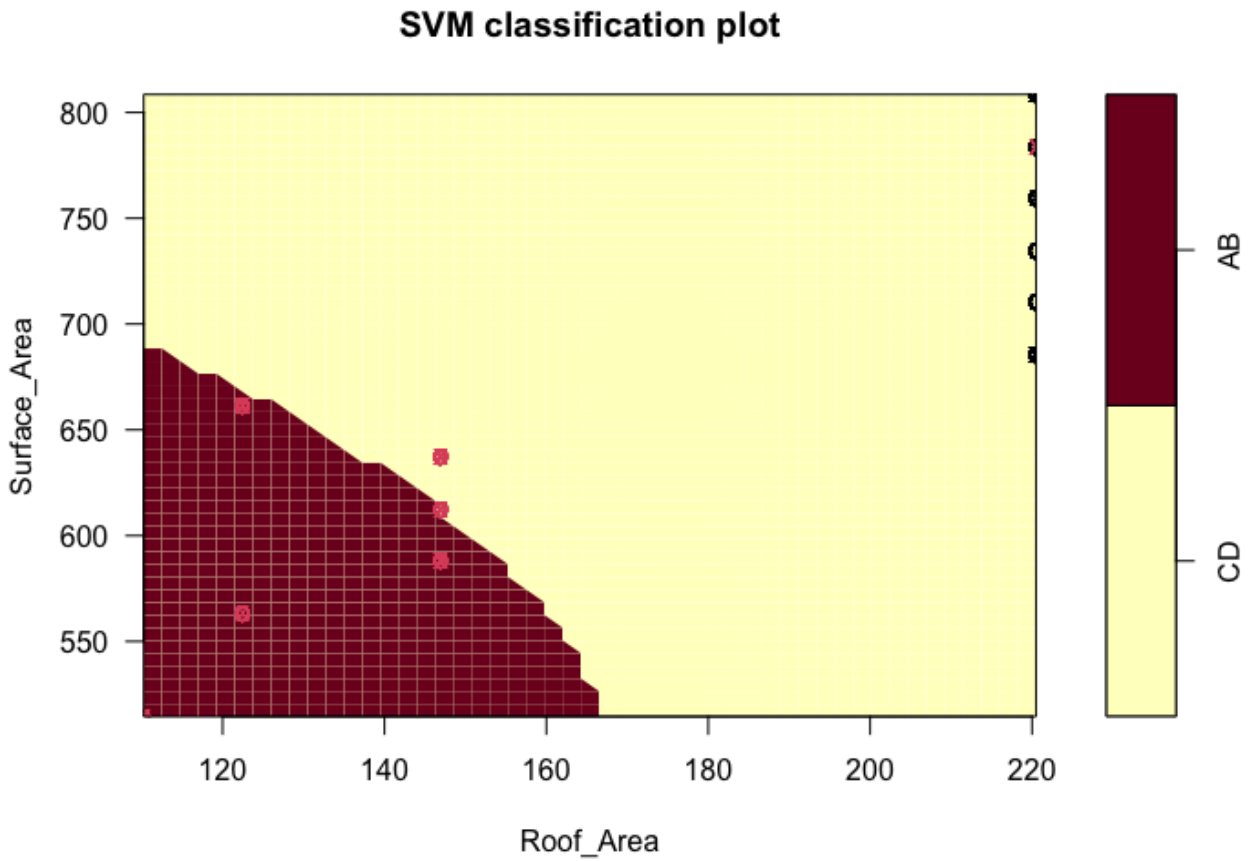
DV: Cooling

```r
#Run the svm for Cooling_Class
Cooling_Class_datasvm <- svm(Cooling_Class ~ Relative_Compactness + Surface_Area + Wall_Area
                             + Roof_Area + Overall_Height + North_Or + East_Or + South_Or +
                               Glazing_Area + Uniform + North + East + South + West,
                             data = data_train)
#Print Cooling_Class_datasvm
print(Cooling_Class_datasvm)

# Plot the SVM model for Cooling_Class
plot(Cooling_Class_datasvm,
     data_train,
     Surface_Area ~ Roof_Area,
     slice = list(Relative_Compactness = 0.8,
                  Wall_Area = 500,
                  Overall_Height = 2,
                  North_Or = 1,
                  East_Or = 0,
                  South_Or = 1,
                  Glazing_Area = 0.03,
                  Uniform = 1,
                  North =  1,
                  East = 0,
                  South = 1,
                  West = 1
     ))

# Predict Cooling_Class on the test dataset
cooling_predictions <- predict(Cooling_Class_datasvm, newdata = data_test[,c("Relative_Compactness
cooling_predictions
predicttable <- table(data_test$Cooling_Class == 'AB', cooling_predictions =='AB')
predicttable
sum(diag(predicttable))/sum(predicttable)
library(caret)
confusionMatrix(cooling_predictions,data_test$Cooling_Class)
```

```
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  113
```

```
> predicttable <- table(data_test$Cooling_Class == 'AB', cooling_predictions =='AB')
> predicttable

       FALSE TRUE
  FALSE   104    4
  TRUE      5  118
> sum(diag(predicttable))/sum(predicttable)
[1] 0.961039
> library(caret)
> confusionMatrix(cooling_predictions,data_test$Cooling_Class)
Confusion Matrix and Statistics

          Reference
Prediction  CD  AB
        CD 104   5
        AB   4 118

               Accuracy : 0.961
                 95% CI : (0.9273, 0.982)
    No Information Rate : 0.5325
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9218

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9630
            Specificity : 0.9593
         Pos Pred Value : 0.9541
         Neg Pred Value : 0.9672
             Prevalence : 0.4675
         Detection Rate : 0.4502
   Detection Prevalence : 0.4719
      Balanced Accuracy : 0.9612

       'Positive' Class : CD
```

DV: Cooling


SVM classification plot
Surface_Area
800
750
700
650
600
550
120
140
160
180
200
220
Roof_Area
AB
CD

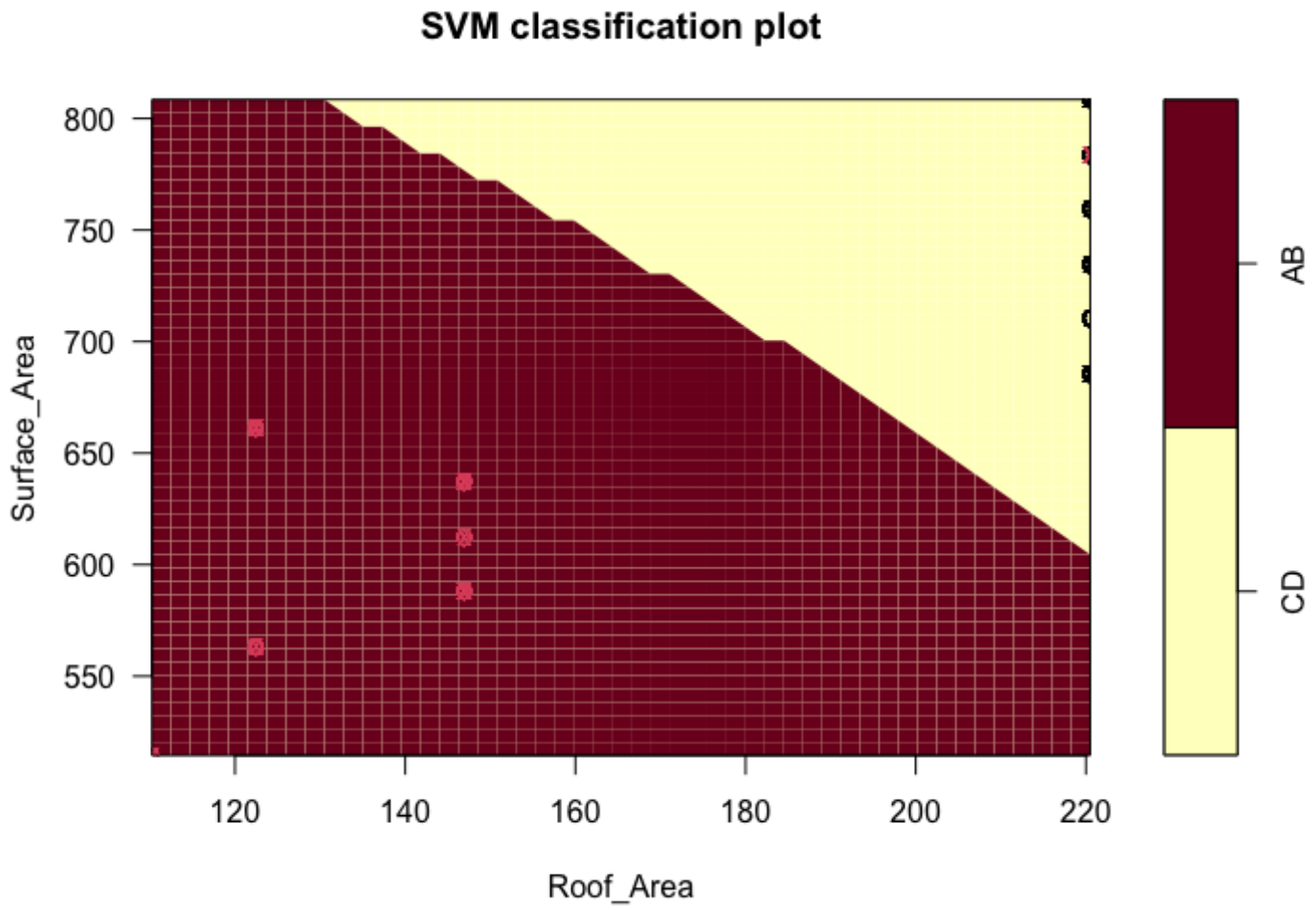
DV: Heating

```r
#Run the svm for Heating_Class
Heating_Class_datasvm <- svm(Heating_Class ~ Relative_Compactness + Surface_Area + Wall_Area +
                               Roof_Area + Overall_Height + North_Or + East_Or + South_Or +
                               Glazing_Area + Uniform + North + East + South + West,
                             data = data_train)

#Print Heating_Class_datasvm
print(Heating_Class_datasvm)


# plot heating svm
plot(Heating_Class_datasvm,
     data_train,
     Surface_Area ~ Roof_Area,
     slice = list(Relative_Compactness = 1,
                  Wall_Area = 400,
                  Overall_Height = 2,
                  North_Or = 1,
                  East_Or = 0,
                  South_Or = 1,
                  Glazing_Area = 0.00,
                  Uniform = 1,
                  North =  1,
                  East = 0,
                  South = 0,
                  West = 0
     ))

# Predict Cooling_Class on the test dataset
heating_predictions <- predict(Heating_Class_datasvm, newdata = data_test[,c("Relative_Compactness
heating_predictions
predicttable2 <- table(data_test$Heating_Class == 'AB', heating_predictions =='AB')
predicttable2
sum(diag(predicttable2))/sum(predicttable2)
library(caret)
confusionMatrix(heating_predictions,data_test$Heating_Class)
```

```
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  127
```

```
> predicttable2 <- table(data_test$Heating_Class == 'AB', heating_predictions =='AB')
> predicttable2

       FALSE TRUE
  FALSE   104    4
  TRUE      5  118
> sum(diag(predicttable2))/sum(predicttable2)
[1] 0.961039
> library(caret)
> confusionMatrix(heating_predictions,data_test$Heating_Class)
Confusion Matrix and Statistics

          Reference
Prediction  CD  AB
        CD 104   5
        AB   4 118

               Accuracy : 0.961
                 95% CI : (0.9273, 0.982)
    No Information Rate : 0.5325
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9218

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9630
            Specificity : 0.9593
         Pos Pred Value : 0.9541
         Neg Pred Value : 0.9672
             Prevalence : 0.4675
         Detection Rate : 0.4502
   Detection Prevalence : 0.4719
      Balanced Accuracy : 0.9612

       'Positive' Class : CD
```

DV: Heating


SVM classification plot
800
750
700
650
600
550
Surface_Area
120
140
160
180
200
220
Roof_Area
AB
CD

Neural Networks

Using a loop for 1 to 5 hidden nodes, create a neural network for each Y-variable cooling load and heating load) (use actual numeric values for loads, continuous output neural network). Report the accuracies for all ten (10) models (5 cooling and 5 heating

```
install.packages("neuralnet", dependencies = TRUE)

# Load necessary libraries
library(neuralnet)
library(readxl)


data <-read_excel("/Users/bhvvnaa/Downloads/EnergyEfficiency_final (1).xlsx")

cooling_net <- neuralnet(Cooling_Load ~ Relative_Compactness + Surface_Area + Wall_Area + Roof_Area +
                           Overall_Height + North_Or + East_Or + South_Or + Glazing_Area + Uniform + North + East + South + West, data, hidden=5,
                         lifesign="minimal", linear.output=TRUE, threshold=0.01)

normalize <- function(x) {return((x-min(x))/(max(x)-min(x)))}
denormalize <- function(y,x){return(y*(max(x)-min(x))+min(x))}

cooling_norm <- as.data.frame(lapply(data,normalize))



set.seed(123)  # For reproducibility
data_index <- sample(nrow(cooling_norm), 0.7 * nrow(cooling_norm), replace = FALSE)
cooling_train <- cooling_norm[data_index, ]
cooling_test <- cooling_norm[-data_index, ]

head(data)
head(cooling_norm)
```

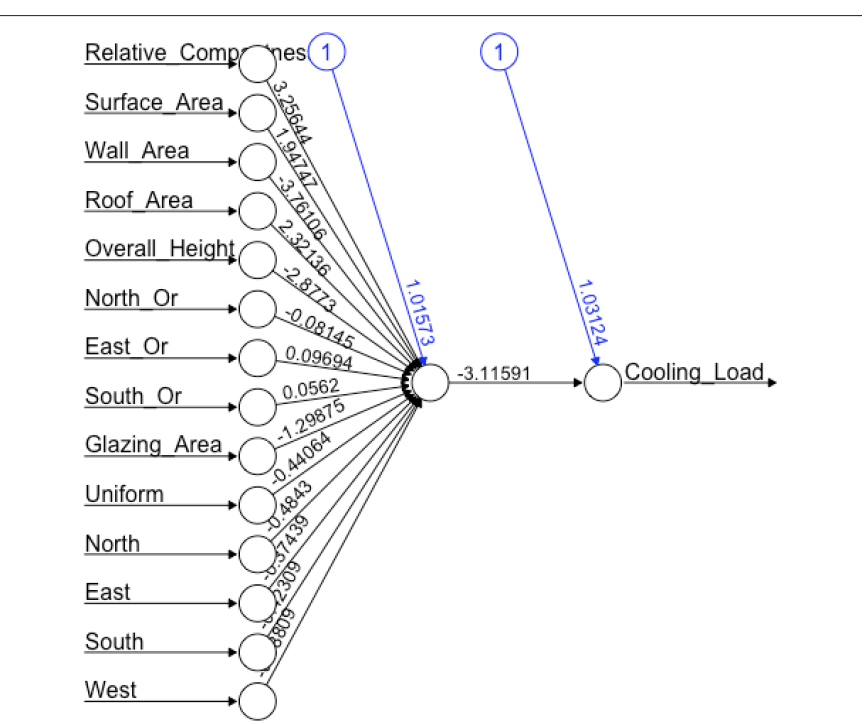
	Relative_Compactness	Surface_Area	Wall_Area	Roof_Area	Overall_Height	Orientation	North_Or	East_Or	South_Or	Glazing_Area	Glazing_Area_Distribution	Uniform	North
1	1.0000000	0.0000000	0.2857143	0.0000000	1	0.0000000	1	0	0	0	0	0	0
2	1.0000000	0.0000000	0.2857143	0.0000000	1	0.3333333	0	1	0	0	0	0	0
3	1.0000000	0.0000000	0.2857143	0.0000000	1	0.6666667	0	0	1	0	0	0	0
4	1.0000000	0.0000000	0.2857143	0.0000000	1	1.0000000	0	0	0	0	0	0	0
5	0.7777778	0.1666667	0.4285714	0.1111111	1	0.0000000	1	0	0	0	0	0	0
6	0.7777778	0.1666667	0.4285714	0.1111111	1	0.3333333	0	1	0	0	0	0	0

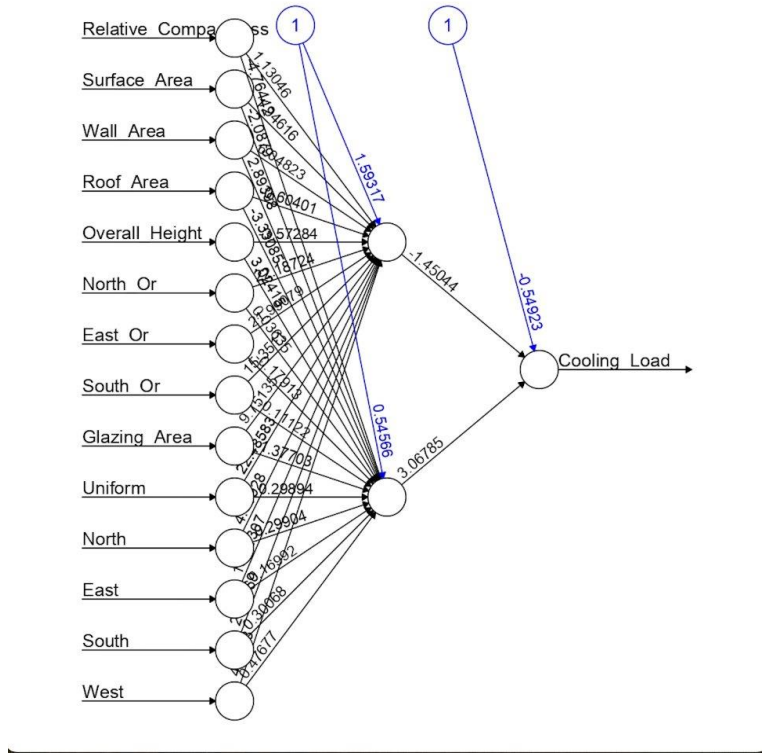
	East	South	West	Heating_Load	Cooling_Load
1	0	0	0	0.2572122	0.2809049
2	0	0	0	0.2572122	0.2809049
3	0	0	0	0.2572122	0.2809049
4	0	0	0	0.2572122	0.2809049
5	0	0	0	0.3998382	0.4680851
6	0	0	0	0.4165543	0.3899811

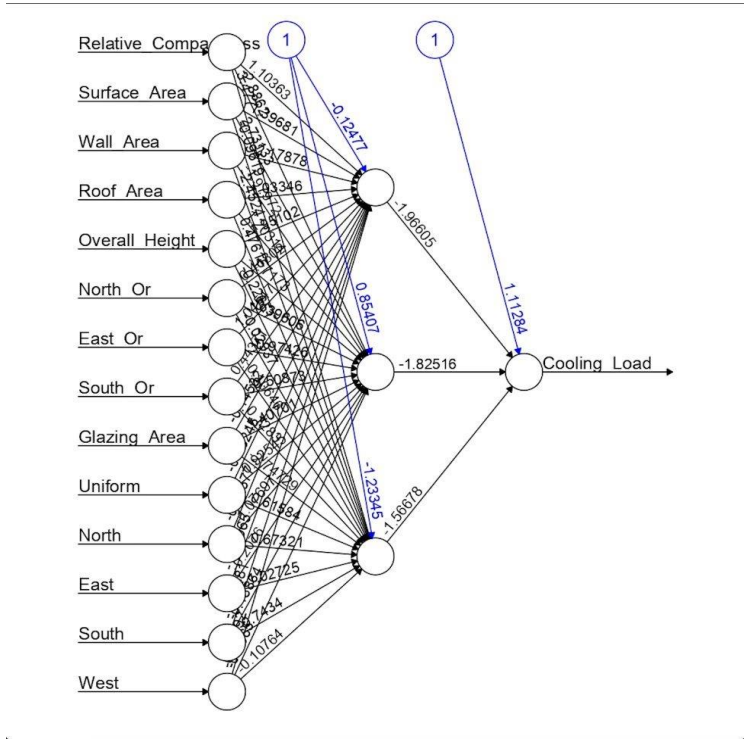
DV: Cooling

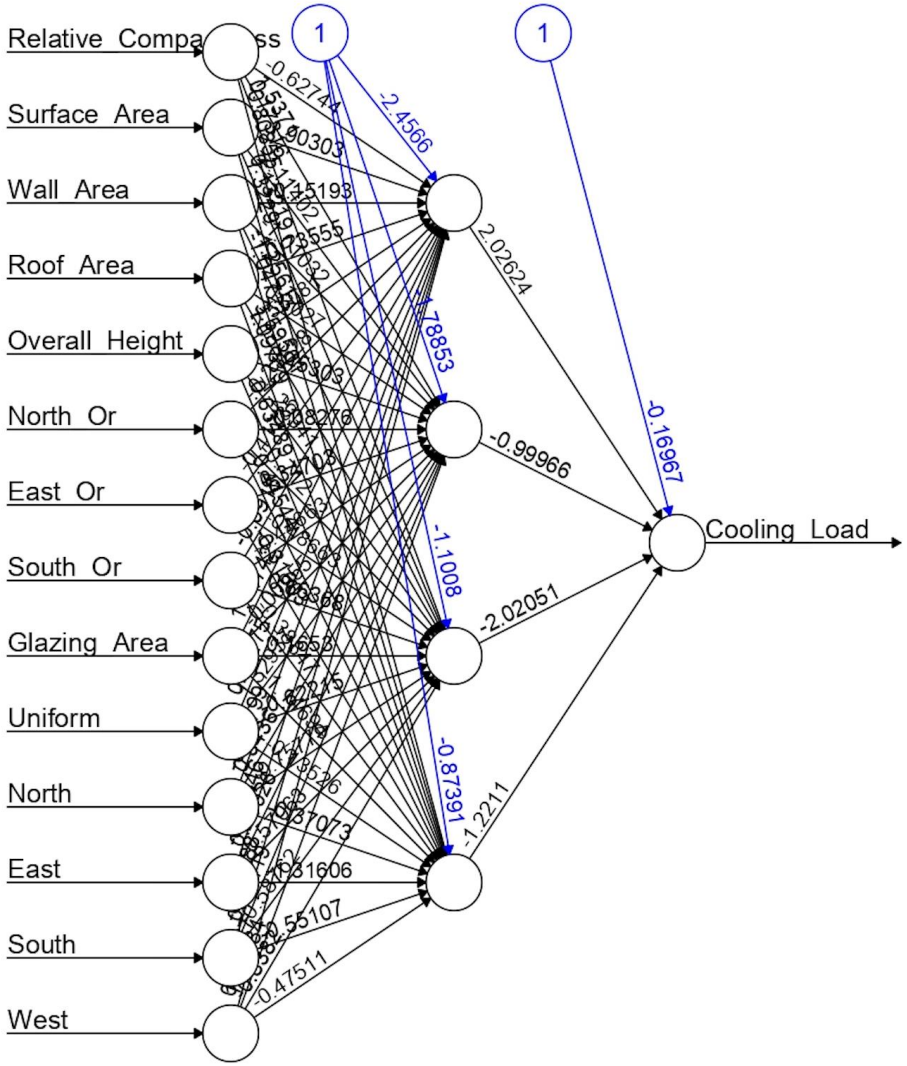
```
corr_max_list <- list()
# Loop through 1 to 5 hidden nodes
for (hidden_nodes in 1:5){
  # Cooling_Load Model
  cooling_model <- neuralnet(Cooling_Load ~ Relative_Compactness + Surface_Area + Wall_Area + Roof_Area +
                               Overall_Height + North_Or + East_Or + South_Or + Glazing_Area + Uniform + North + East + South + West,
                             data = cooling_train, hidden = hidden_nodes, lifesign = "minimal", linear.output = FALSE, threshold = 0.1, stepmax = 1e7)
  cooling_results <- compute(cooling_model, cooling_test[, c("Relative_Compactness", "Surface_Area", "Wall_Area", "Roof_Area",
                                                             "Overall_Height", "North_Or", "East_Or", "South_Or", "Glazing_Area", "Uniform", "North", "East", "South", "
  cooling_denorm <- denormalize(cooling_results$net.result,data$Cooling_Load[data_index])
  cooling_correlation <- cor(cooling_denorm, actualCooling)
  print(cooling_correlation)
  corr_max_list[paste(hidden_nodes)] <- cooling_correlation
  plot(cooling_model)
  }
corr_max_list[which.max(sapply(corr_max_list,max))]
```

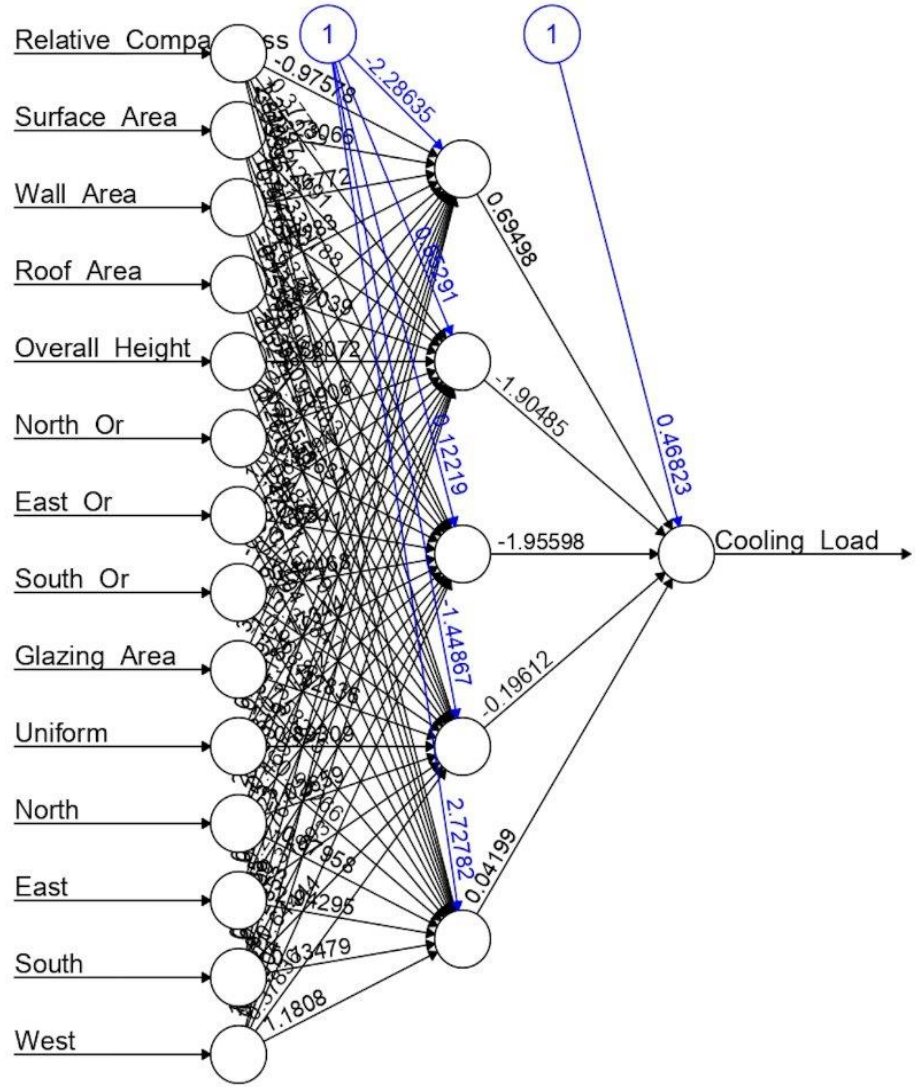
```
hidden: 1    thresh: 0.1    rep: 1/1    steps:     159  error: 2.11538  time: 0.01 secs
          [,1]
[1,] 0.9421985
hidden: 2    thresh: 0.1    rep: 1/1    steps:      70  error: 2.11147  time: 0.01 secs
          [,1]
[1,] 0.942914
hidden: 3    thresh: 0.1    rep: 1/1    steps:      48  error: 1.9949   time: 0 secs
          [,1]
[1,] 0.9470624
hidden: 4    thresh: 0.1    rep: 1/1    steps:      45  error: 1.97789  time: 0.03 secs
          [,1]
[1,] 0.947677
hidden: 5    thresh: 0.1    rep: 1/1    steps:      72  error: 1.76572  time: 0.01 secs
          [,1]
[1,] 0.9447179
> corr_max_list[which.max(sapply(corr_max_list,max))]
$`4`
[1] 0.947677
```


Relative_Compactness
Surface_Area
Wall_Area
Roof_Area
Overall_Height
North_Or
East_Or
South_Or
Glazing_Area
Uniform
North
East
South
West
1
1
1.01573
1.03124
-3.11591
Cooling_Load


Relative_Compactness
Surface_Area
Wall_Area
Roof_Area
Overall_Height
North_Or
East_Or
South_Or
Glazing_Area
Uniform
North
East
South
West
1
1
1.59317
0.54566
-1.45044
-0.54923
3.06785
Cooling_Load


Relative_Compactness
Surface_Area
Wall_Area
Roof_Area
Overall_Height
North_Or
East_Or
South_Or
Glazing_Area
Uniform
North
East
South
West
1
1
-0.12477
0.85407
-1.23345
-1.96605
1.11284
-1.56678
-1.82516
Cooling_Load


Relative Compa ss
Surface Area
Wall Area
Roof Area
Overall Height
North Or
East Or
South Or
Glazing Area
Uniform
North
East
South
West
1
1
-0.62744
-2.4566
2.02624
-1.78853
-0.99966
-1.1008
-2.02051
-0.87391
-1.2211
-0.16967
Cooling Load

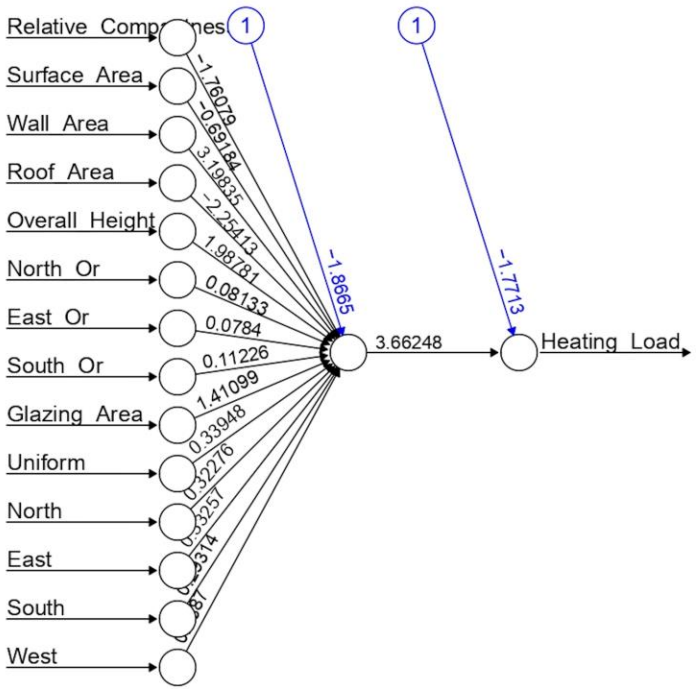

Relative Compa ss
Surface Area
Wall Area
Roof Area
Overall Height
North Or
East Or
South Or
Glazing Area
Uniform
North
East
South
West
1
1
-0.97578
-2.28635
0.69498
0.65291
-1.90485
0.12219
-1.95598
-1.44867
-0.19612
2.72782
0.04199
0.46823
Cooling Load

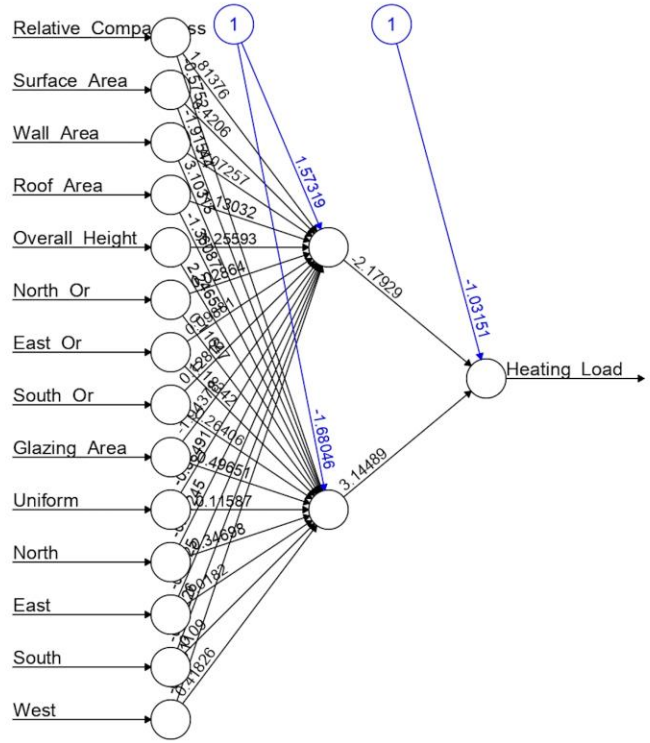
Method		Cooling			
		Accuracy	NIR	P-value	Significant?
Neural network	Hidden = 1	0.7662	0.2771	<2.2e-16	YES
	Hidden = 2	0.7576	0.2771	<2.2e-16	YES
	Hidden = 3	0.7403	0.2771	<2.2e-16	YES
	Hidden = 4	0.7403	0.2771	<2.2e-16	YES
	Hidden = 5	0.7835	0.2771	<2.2e-16	YES

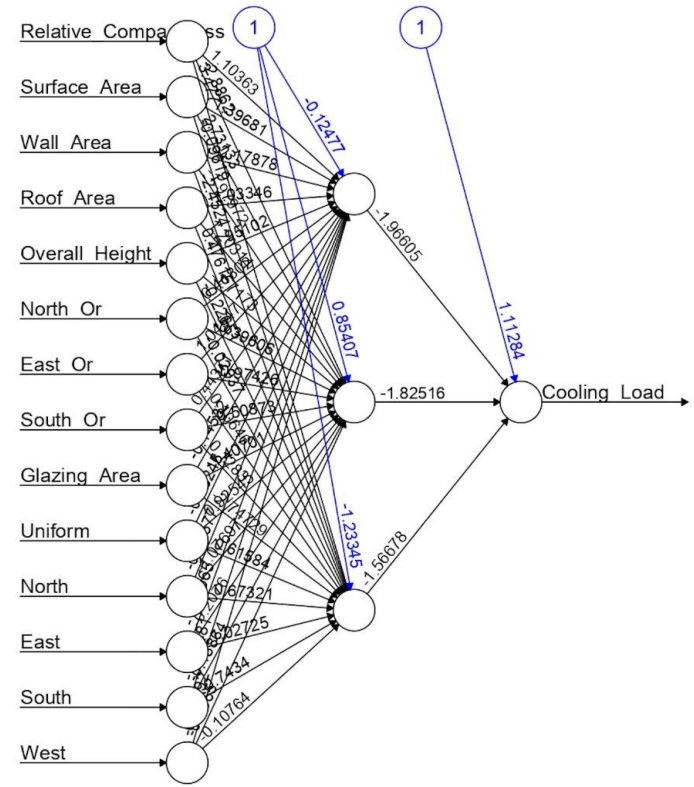
DV: Heating

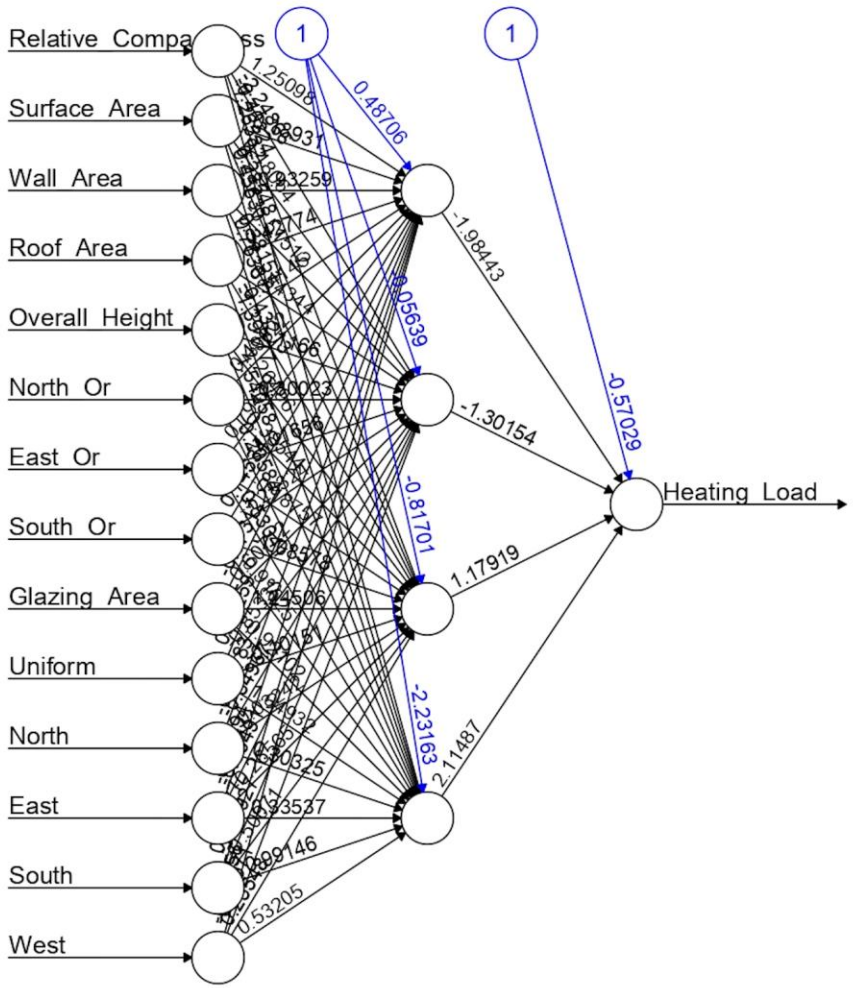
```
heating_net <- neuralnet(Heating_Load ~ Relative_Compactness + Surface_Area + Wall_Area + Roof_Area +
                           Overall_Height + North_Or + East_Or + South_Or + Glazing_Area + Uniform + North + East + South + West, data, hidden=5,
                         lifesign="minimal", linear.output=TRUE, threshold=0.01)

normalize <- function(x) {return((x-min(x))/(max(x)-min(x)))}
denormalize <- function(y,x){return(y*(max(x)-min(x))+min(x))}

heating_norm <- as.data.frame(lapply(data,normalize))


set.seed(123)  # For reproducibility
data_index <- sample(nrow(heating_norm), 0.7 * nrow(heating_norm), replace = FALSE)
heating_train <- heating_norm[data_index, ]
heating_test <- heating_norm[-data_index, ]

head(data)
head(heating_norm)
actualHeating <- data$Heating_Load[-data_index]

corr_max_list2 <- list()
# Loop through 1 to 5 hidden nodes
for (hidden_nodes in 1:5){
  # Cooling_Load Model
  heating_model <- neuralnet(Heating_Load ~ Relative_Compactness + Surface_Area + Wall_Area + Roof_Area +
                               Overall_Height + North_Or + East_Or + South_Or + Glazing_Area + Uniform + North + East + South + West,
                             data = heating_train, hidden = hidden_nodes, lifesign = "minimal", linear.output = FALSE, threshold = 0.1, stepmax = 1e7)
  heating_results <- compute(heating_model, heating_test[, c("Relative_Compactness", "Surface_Area", "Wall_Area", "Roof_Area",
                                                             "Overall_Height", "North_Or", "East_Or", "South_Or", "Glazing_Area", "Uniform", "North", "East", "South", "Wes
  heating_denorm <- denormalize(heating_results$net.result,data$Heating_Load[data_index])
  heating_correlation <- cor(heating_denorm, actualHeating)
  print(heating_correlation)
  corr_max_list2[paste(hidden_nodes)] <- heating_correlation
  plot(heating_model)
}
corr_max_list2[which.max(sapply(corr_max_list2,max))]
```

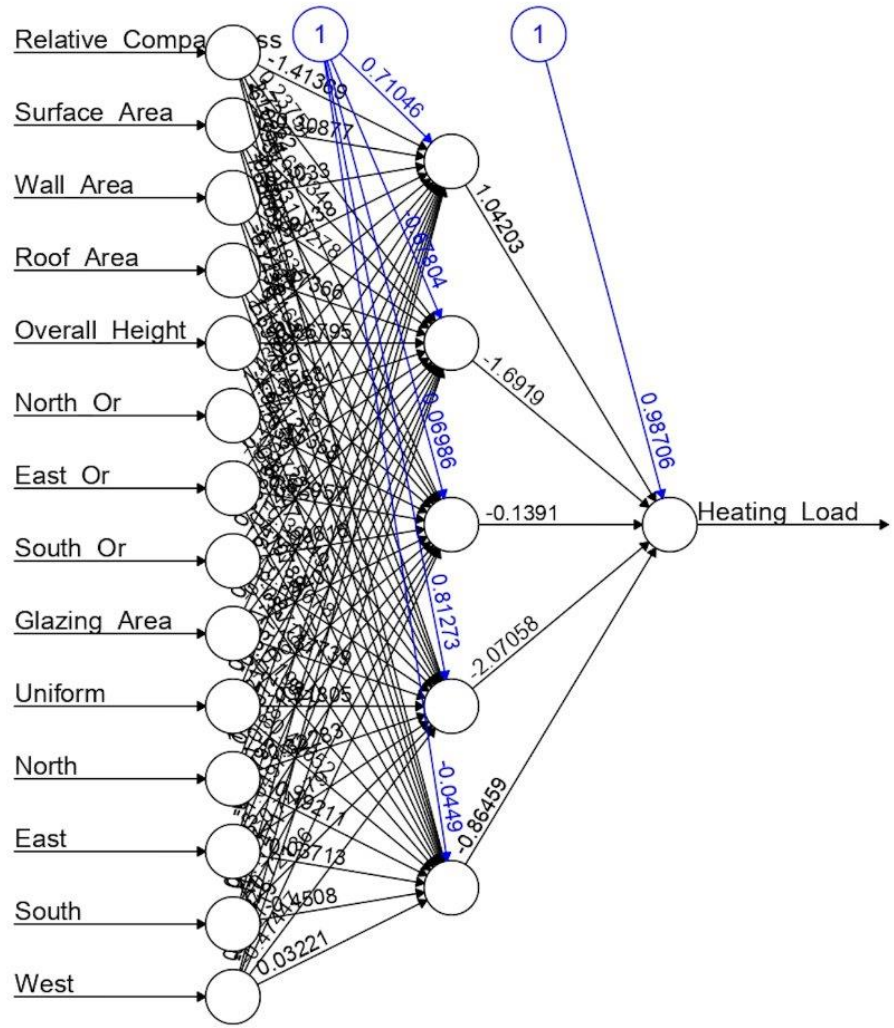
```
hidden: 1    thresh: 0.1    rep: 1/1    steps:     133  error: 1.72166  time: 0.01 secs
          [,1]
[1,] 0.9559848
hidden: 2    thresh: 0.1    rep: 1/1    steps:     107  error: 1.35973  time: 0.01 secs
          [,1]
[1,] 0.9645049
hidden: 3    thresh: 0.1    rep: 1/1    steps:      56  error: 1.53629  time: 0.01 secs
          [,1]
[1,] 0.963374
hidden: 4    thresh: 0.1    rep: 1/1    steps:      55  error: 1.49456  time: 0.01 secs
          [,1]
[1,] 0.9639266
hidden: 5    thresh: 0.1    rep: 1/1    steps:      60  error: 1.53034  time: 0.01 secs
          [,1]
[1,] 0.9609335
> corr_max_list2[which.max(sapply(corr_max_list2,max))]
$`2`
[1] 0.9645049
```


Relative Compactness
Surface Area
Wall Area
Roof Area
Overall Height
North Or
East Or
South Or
Glazing Area
Uniform
North
East
South
West
1
1
-1.8665
3.66248
-1.7713
Heating Load


Relative Compactness
Surface Area
Wall Area
Roof Area
Overall Height
North Or
East Or
South Or
Glazing Area
Uniform
North
East
South
West
1
1
1.57319
-1.68046
-2.17929
3.14489
-1.03151
Heating Load


Relative Compactness
Surface Area
Wall Area
Roof Area
Overall Height
North Or
East Or
South Or
Glazing Area
Uniform
North
East
South
West
1
1
-0.12477
0.85407
-1.96605
-1.23345
-1.82516
1.11284
-1.56678
Cooling Load


Relative Compa ss
Surface Area
Wall Area
Roof Area
Overall Height
North Or
East Or
South Or
Glazing Area
Uniform
North
East
South
West
1
1
0.48706
-1.98443
-0.05639
-1.30154
-0.81701
1.17919
-2.23163
2.11487
-0.57029
Heating Load


Relative Compa ss
Surface Area
Wall Area
Roof Area
Overall Height
North Or
East Or
South Or
Glazing Area
Uniform
North
East
South
West
1
1
0.71046
1.04203
-0.61904
-1.6919
-0.06986
-0.1391
0.81273
-2.07058
-0.0449
-0.86459
0.98706
Heating Load

Method		Heating			
		Accuracy	NIR	P-value	Significant?
Neural network	Hidden = 1	0.7835	0.2684	<2.2e-16	YES
	Hidden = 2	0.7835	0.2684	<2.2e-16	YES
	Hidden = 3	0.8268	0.2684	<2.2e-16	YES
	Hidden = 4	0.8095	0.2684	<2.2e-16	YES
	Hidden = 5	0.7965	0.2684	<2.2e-16	YES

3 (e) K-nearest neighbor

3(e) DV: cooling; k=5

```
> set.seed(7)
> energy_index <- sample(nrow(energy_norm), 0.7 * nrow(energy_norm))
> energy_train <- energy_norm[energy_index, ]
> energy_test <- energy_norm[-energy_index, ]
> energy_train_labels <- energy[energy_index,17,drop=TRUE]
> energy_test_labels <- energy[-energy_index,17,drop=TRUE]
> library(class)
> energy_test_pred <- knn(train = energy_train, test = energy_test,
+                         cl=energy_train_labels, k=5)
> library(gmodels)
> energytable <- CrossTable(x=energy_test_labels, y=energy_test_pred,
+                           prop.chisq=FALSE)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  231

                 | energy_test_pred
energy_test_labels |         A |         B |         C |         D | Row Total |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 A |        36 |        33 |         0 |         0 |        69 |
                   |     0.522 |     0.478 |     0.000 |     0.000 |     0.299 |
                   |     0.692 |     0.452 |     0.000 |     0.000 |           |
                   |     0.156 |     0.143 |     0.000 |     0.000 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 B |        16 |        40 |         1 |         0 |        57 |
                   |     0.281 |     0.702 |     0.018 |     0.000 |     0.247 |
                   |     0.308 |     0.548 |     0.017 |     0.000 |           |
                   |     0.069 |     0.173 |     0.004 |     0.000 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 C |         0 |         0 |        41 |        13 |        54 |
                   |     0.000 |     0.000 |     0.759 |     0.241 |     0.234 |
                   |     0.000 |     0.000 |     0.695 |     0.277 |           |
                   |     0.000 |     0.000 |     0.177 |     0.056 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 D |         0 |         0 |        17 |        34 |        51 |
                   |     0.000 |     0.000 |     0.333 |     0.667 |     0.221 |
                   |     0.000 |     0.000 |     0.288 |     0.723 |           |
                   |     0.000 |     0.000 |     0.074 |     0.147 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
      Column Total |        52 |        73 |        59 |        47 |       231 |
                   |     0.225 |     0.316 |     0.255 |     0.203 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
```

```
> sum(diag(energytable$prop.tbl))
[1] 0.6536797
> confusionMatrix(energy_test_pred,energy_test_labels)
Confusion Matrix and Statistics

          Reference
Prediction  A  B  C  D
         A 36 16  0  0
         B 33 40  0  0
         C  0  1 41 17
         D  0  0 13 34

Overall Statistics

               Accuracy : 0.6537
                 95% CI : (0.5885, 0.7149)
    No Information Rate : 0.2987
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5383

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D
Sensitivity            0.5217   0.7018   0.7593   0.6667
Specificity            0.9012   0.8103   0.8983   0.9278
Pos Pred Value         0.6923   0.5479   0.6949   0.7234
Neg Pred Value         0.8156   0.8924   0.9244   0.9076
Prevalence             0.2987   0.2468   0.2338   0.2208
Detection Rate         0.1558   0.1732   0.1775   0.1472
Detection Prevalence   0.2251   0.3160   0.2554   0.2035
Balanced Accuracy      0.7115   0.7560   0.8288   0.7972
```

Accuracy: 0.6537

3(e) DV: cooling; k=7

```
> energy_test_pred <- knn(train = energy_train, test = energy_test,
+                         cl=energy_train_labels, k=7)
> library(gmodels)
> energytable <- CrossTable(x=energy_test_labels, y=energy_test_pred,
+                           prop.chisq=FALSE)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  231


                 | energy_test_pred
energy_test_labels |         A |         B |         C |         D | Row Total |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 A |        35 |        34 |         0 |         0 |        69 |
                   |     0.507 |     0.493 |     0.000 |     0.000 |     0.299 |
                   |     0.673 |     0.466 |     0.000 |     0.000 |           |
                   |     0.152 |     0.147 |     0.000 |     0.000 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 B |        17 |        39 |         1 |         0 |        57 |
                   |     0.298 |     0.684 |     0.018 |     0.000 |     0.247 |
                   |     0.327 |     0.534 |     0.017 |     0.000 |           |
                   |     0.074 |     0.169 |     0.004 |     0.000 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 C |         0 |         0 |        42 |        12 |        54 |
                   |     0.000 |     0.000 |     0.778 |     0.222 |     0.234 |
                   |     0.000 |     0.000 |     0.700 |     0.261 |           |
                   |     0.000 |     0.000 |     0.182 |     0.052 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 D |         0 |         0 |        17 |        34 |        51 |
                   |     0.000 |     0.000 |     0.333 |     0.667 |     0.221 |
                   |     0.000 |     0.000 |     0.283 |     0.739 |           |
                   |     0.000 |     0.000 |     0.074 |     0.147 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
      Column Total |        52 |        73 |        60 |        46 |       231 |
                   |     0.225 |     0.316 |     0.260 |     0.199 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
```

```
> confusionMatrix(energy_test_pred,energy_test_labels)
Confusion Matrix and Statistics

          Reference
Prediction  A  B  C  D
         A 35 17  0  0
         B 34 39  0  0
         C  0  1 42 17
         D  0  0 12 34

Overall Statistics

               Accuracy : 0.6494
                 95% CI : (0.584, 0.7108)
    No Information Rate : 0.2987
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5325

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D
Sensitivity            0.5072   0.6842   0.7778   0.6667
Specificity            0.8951   0.8046   0.8983   0.9333
Pos Pred Value         0.6731   0.5342   0.7000   0.7391
Neg Pred Value         0.8101   0.8861   0.9298   0.9081
Prevalence             0.2987   0.2468   0.2338   0.2208
Detection Rate         0.1515   0.1688   0.1818   0.1472
Detection Prevalence   0.2251   0.3160   0.2597   0.1991
Balanced Accuracy      0.7012   0.7444   0.8380   0.8000
```

Accuracy: 0.6494

3(e) DV: heating; k=5

```
> set.seed(7)
> energy_index <- sample(nrow(energy_norm), 0.7 * nrow(energy_norm))
> energy_train <- energy_norm[energy_index, ]
> energy_test <- energy_norm[-energy_index, ]
> energy_train_labels <- energy[energy_index,18,drop=TRUE]
> energy_test_labels <- energy[-energy_index,18,drop=TRUE]
> library(class)
> energy_test_pred <- knn(train = energy_train, test = energy_test,
+                         cl=energy_train_labels, k=5)
> library(gmodels)
> energytable <- CrossTable(x=energy_test_labels, y=energy_test_pred,
+                           prop.chisq=FALSE)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  231


                 | energy_test_pred
energy_test_labels |         D |         C |         B |         A | Row Total |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 D |        42 |        11 |         0 |         0 |        53 |
                   |     0.792 |     0.208 |     0.000 |     0.000 |     0.229 |
                   |     0.764 |     0.212 |     0.000 |     0.000 |           |
                   |     0.182 |     0.048 |     0.000 |     0.000 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 C |        13 |        37 |         1 |         0 |        51 |
                   |     0.255 |     0.725 |     0.020 |     0.000 |     0.221 |
                   |     0.236 |     0.712 |     0.012 |     0.000 |           |
                   |     0.056 |     0.160 |     0.004 |     0.000 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 B |         0 |         4 |        45 |         8 |        57 |
                   |     0.000 |     0.070 |     0.789 |     0.140 |     0.247 |
                   |     0.000 |     0.077 |     0.536 |     0.200 |           |
                   |     0.000 |     0.017 |     0.195 |     0.035 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
                 A |         0 |         0 |        38 |        32 |        70 |
                   |     0.000 |     0.000 |     0.543 |     0.457 |     0.303 |
                   |     0.000 |     0.000 |     0.452 |     0.800 |           |
                   |     0.000 |     0.000 |     0.165 |     0.139 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
      Column Total |        55 |        52 |        84 |        40 |       231 |
                   |     0.238 |     0.225 |     0.364 |     0.173 |           |
-------------------|-----------|-----------|-----------|-----------|-----------|
```

```
> confusionMatrix(energy_test_pred,energy_test_labels)
Confusion Matrix and Statistics

          Reference
Prediction  D  C  B  A
         D 42 13  0  0
         C 11 37  4  0
         B  0  1 45 38
         A  0  0  8 32

Overall Statistics

               Accuracy : 0.6753
                 95% CI : (0.6108, 0.7353)
    No Information Rate : 0.303
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5691

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: D Class: C Class: B Class: A
Sensitivity            0.7925   0.7255   0.7895   0.4571
Specificity            0.9270   0.9167   0.7759   0.9503
Pos Pred Value         0.7636   0.7115   0.5357   0.8000
Neg Pred Value         0.9375   0.9218   0.9184   0.8010
Prevalence             0.2294   0.2208   0.2468   0.3030
Detection Rate         0.1818   0.1602   0.1948   0.1385
Detection Prevalence   0.2381   0.2251   0.3636   0.1732
Balanced Accuracy      0.8597   0.8211   0.7827   0.7037
```

Accuracy: 0.6753

3(e) DV: heating; k=7

```
> energy_test_pred <- knn(train = energy_train, test = energy_test,
+                         cl=energy_train_labels, k=7)
> library(gmodels)
> energytable <- CrossTable(x=energy_test_labels, y=energy_test_pred,
+                           prop.chisq=FALSE)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  231
```

energy_test_labels	energy_test_pred D	C	B	A	Row Total
D	42	11	0	0	53
	0.792	0.208	0.000	0.000	0.229
	0.764	0.208	0.000	0.000	
	0.182	0.048	0.000	0.000	
C	13	38	0	0	51
	0.255	0.745	0.000	0.000	0.221
	0.236	0.717	0.000	0.000	
	0.056	0.165	0.000	0.000	
B	0	4	41	12	57
	0.000	0.070	0.719	0.211	0.247
	0.000	0.075	0.539	0.255	
	0.000	0.017	0.177	0.052	
A	0	0	35	35	70
	0.000	0.000	0.500	0.500	0.303
	0.000	0.000	0.461	0.745	
	0.000	0.000	0.152	0.152	
Column Total	55	53	76	47	231
	0.238	0.229	0.329	0.203	

```
> confusionMatrix(energy_test_pred,energy_test_labels)
Confusion Matrix and Statistics

          Reference
Prediction  D  C  B  A
         D 42 13  0  0
         C 11 38  4  0
         B  0  0 41 35
         A  0  0 12 35

Overall Statistics

               Accuracy : 0.6753
                 95% CI : (0.6108, 0.7353)
    No Information Rate : 0.303
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5682

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: D Class: C Class: B Class: A
Sensitivity            0.7925   0.7451   0.7193   0.5000
Specificity            0.9270   0.9167   0.7989   0.9255
Pos Pred Value         0.7636   0.7170   0.5395   0.7447
Neg Pred Value         0.9375   0.9270   0.8968   0.8098
Prevalence             0.2294   0.2208   0.2468   0.3030
Detection Rate         0.1818   0.1645   0.1775   0.1515
Detection Prevalence   0.2381   0.2294   0.3290   0.2035
Balanced Accuracy      0.8597   0.8309   0.7591   0.7127
```

Accuracy: 0.6753

3(f) Naïve bayes

3(f) DV: cooling

```
> ##Naive bayes -- for cooling
> #install.packages("e1071",dependencies=TRUE)
> library(e1071) ## in SVM
> energy <- read_xlsx("energy.xlsx")
> convert_to_quartile_categories <- function(x) {
+   quartiles <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))
+   categories <- cut(x,
+                     breaks = quartiles,
+                     labels = c("D", "C", "B", "A"),
+                     include.lowest = TRUE)
+   return(categories)
+ }
> energy$cooling_category <- convert_to_quartile_categories(energy$Cooling_Load)
> energy$heating_category <- convert_to_quartile_categories(energy$Heating_Load)
> energy <- subset(energy, select = -c(Orientation, Glazing_Area_Distribution, h
eating_category, Heating_Load, Cooling_Load ))
> set.seed(7)
> energy_index <- sample(nrow(energy), 0.7 * nrow(energy))
> energy_train <- energy[energy_index, ]
> energy_test <- energy[-energy_index, ]
> #Run the Naïve Bayes classifier
> energy_model <- naiveBayes(cooling_category ~ ., data=energy_train, laplace=1)
> #Print the results of the Naïve Bayes
> energy_model
```

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
       D        C        B        A
0.2625698 0.2569832 0.2513966 0.2290503

Conditional probabilities:
   Relative_Compactness
Y        [,1]       [,2]
  D 0.6898582 0.04218981
  C 0.6786232 0.06867029
  B 0.8673333 0.08102146
  A 0.8282114 0.07184948

   Surface_Area
Y       [,1]     [,2]
  D 734.3050 43.13106
  C 747.6051 59.73703
  B 586.5481 58.55900
  A 612.8984 48.49370

   Wall_Area
Y       [,1]     [,2]
  D 293.3050 43.13106
  C 316.1920 38.01149
  B 310.6963 21.51155
  A 349.5732 46.53554

   Roof_Area
Y       [,1]     [,2]
  D 220.5000  0.00000
  C 215.7065 21.21426
  B 137.9259 23.89666
  A 131.6626 13.63683

   Overall_Height
Y       [,1]      [,2]
  D 3.500000 0.0000000
  C 3.677536 0.7708198
  B 6.844444 0.7239678
  A 7.000000 0.0000000

   North_Or
Y        [,1]      [,2]
  D 0.2765957 0.4489095
  C 0.2246377 0.4188639
  B 0.2666667 0.4438636
  A 0.2357724 0.4262167

   East_Or
Y        [,1]      [,2]
  D 0.2695035 0.4452837
  C 0.2681159 0.4445921
  B 0.2888889 0.4549343
  A 0.2357724 0.4262167

   South_Or
Y        [,1]      [,2]
  D 0.1985816 0.4003545
  C 0.2608696 0.4407086
  B 0.2222222 0.4172881
  A 0.2357724 0.4262167

   Glazing_Area
Y        [,1]       [,2]
  D 0.1514184 0.09616639
  C 0.2916667 0.13202291
  B 0.2096296 0.13008268
  A 0.2894309 0.12318386

   Uniform
Y        [,1]      [,2]
  D 0.1347518 0.3426756
  C 0.1811594 0.3865533
  B 0.1925926 0.3958044
  A 0.1707317 0.3778133

   North
Y        [,1]      [,2]
  D 0.1560284 0.3641759
  C 0.2028986 0.4036227
  B 0.1925926 0.3958044
  A 0.1788618 0.3848040

   East
Y        [,1]      [,2]
  D 0.1985816 0.4003545
  C 0.1594203 0.3674011
  B 0.1925926 0.3958044
  A 0.2113821 0.4099587

   South
Y        [,1]      [,2]
  D 0.1843972 0.3891903
  C 0.2173913 0.4139736
  B 0.1925926 0.3958044
  A 0.1951220 0.3979154

   West
Y        [,1]      [,2]
  D 0.2127660 0.4107225
  C 0.1666667 0.3740357
  B 0.1481481 0.3565699
  A 0.2357724 0.4262167
```

```
> energy_pred <- predict(energy_model, energy_test, type="class")
> energy_pred_table <- table(energy_test$cooling_category, energy_pred)
> energy_pred_table
   energy_pred
     D  C  B  A
  D 51  0  0  0
  C 54  0  0  0
  B  1  0  0 56
  A  0  0  0 69
```

3(f) DV: heating

```
> library(e1071) ## in SVM
> energy <- read_xlsx("energy.xlsx")
> convert_to_quartile_categories <- function(x) {
+   quartiles <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))
+   categories <- cut(x,
+                     breaks = quartiles,
+                     labels = c("D", "C", "B", "A"),
+                     include.lowest = TRUE)
+   return(categories)
+ }
> energy$cooling_category <- convert_to_quartile_categories(energy$Cooling
_Load)
> energy$heating_category <- convert_to_quartile_categories(energy$Heating
_Load)
> energy <- subset(energy, select = -c(Orientation, Glazing_Area_Distribut
ion, cooling_category, Heating_Load, Cooling_Load ))
> set.seed(7)
> energy_index <- sample(nrow(energy), 0.7 * nrow(energy))
> energy_train <- energy[energy_index, ]
> energy_test <- energy[-energy_index, ]
> #Run the Naïve Bayes classifier
> energy_model <- naiveBayes(heating_category ~ ., data=energy_train, lapl
ace=1)
> #Print the results of the Naïve Bayes
> energy_model
```

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
        D         C         B         A
0.2588454 0.2625698 0.2513966 0.2271881

Conditional probabilities:
   Relative_Compactness
Y         [,1]       [,2]
  D 0.6957554 0.03593429
  C 0.6736879 0.07059352
  B 0.8635556 0.07934872
  A 0.8327869 0.07617143

   Surface_Area
Y       [,1]     [,2]
  D 727.7734 35.81531
  C 753.0709 62.64431
  B 588.9074 57.60611
  A 610.0902 51.21854

   Wall_Area
Y       [,1]     [,2]
  D 286.7734 35.81531
  C 322.4965 40.43021
  B 310.6963 21.71878
  A 349.6270 46.72358

   Roof_Area
Y       [,1]     [,2]
  D 220.5000  0.00000
  C 215.2872 21.77775
  B 139.1056 23.49513
  A 130.2316 13.65745

   Overall_Height
Y       [,1]      [,2]
  D 3.500000 0.0000000
  C 3.698582 0.8125784
  B 6.844444 0.7239678
  A 7.000000 0.0000000

   North_Or
Y         [,1]      [,2]
  D 0.2661871 0.4435617
  C 0.2340426 0.4249084
  B 0.2814815 0.4513967
  A 0.2213115 0.4168416

   East_Or
Y         [,1]      [,2]
  D 0.2589928 0.4396660
  C 0.2624113 0.4415135
  B 0.2814815 0.4513967
  A 0.2622951 0.4416962

   South_Or
Y         [,1]      [,2]
  D 0.2158273 0.4128829
  C 0.2482270 0.4335242
  B 0.2000000 0.4014898
  A 0.2540984 0.4371484

   Glazing_Area
Y         [,1]      [,2]
  D 0.1500000 0.09724465
  C 0.2890071 0.13216497
  B 0.2040741 0.12623300
  A 0.2979508 0.12084372

   Uniform
Y         [,1]      [,2]
  D 0.1366906 0.3447629
  C 0.1773050 0.3832882
  B 0.2000000 0.4014898
  A 0.1639344 0.3717427

   North
Y         [,1]      [,2]
  D 0.1654676 0.3729460
  C 0.1914894 0.3948760
  B 0.1851852 0.3898945
  A 0.1885246 0.3927434

   East
Y         [,1]      [,2]
  D 0.1942446 0.3970489
  C 0.1631206 0.3707928
  B 0.1777778 0.3837495
  A 0.2295082 0.4222507

   South
Y         [,1]      [,2]
  D 0.1726619 0.3793216
  C 0.2269504 0.4203535
  B 0.1851852 0.3898945
  A 0.2049180 0.4053062

   West
Y         [,1]      [,2]
  D 0.1942446 0.3970489
  C 0.1843972 0.3891903
  B 0.1703704 0.3773581
  A 0.2131148 0.4111968
```

```
> energy_pred <- predict(energy_model, energy_test, type="class")
> energy_pred_table <- table(energy_test$heating_category, energy_pred)
> energy_pred_table
   energy_pred
     D  C  B  A
  D 53  0  0  0
  C 50  0  0  1
  B  3  0  0 54
  A  0  0  0 70
```

3(f) comparison

DV: cooling

```
> confusionMatrix(energy_pred,energy_test$cooling_category)
Confusion Matrix and Statistics

          Reference
Prediction  D  C  B  A
         D 51 54  1  0
         C  0  0  0  0
         B  0  0  0  0
         A  0  0 56 69

Overall Statistics

               Accuracy : 0.5195
                 95% CI : (0.453, 0.5855)
    No Information Rate : 0.2987
    P-Value [Acc > NIR] : 2.078e-12

                  Kappa : 0.3481

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: D Class: C Class: B Class: A
Sensitivity            1.0000   0.0000   0.0000   1.0000
Specificity            0.6944   1.0000   1.0000   0.6543
Pos Pred Value         0.4811      NaN      NaN   0.5520
Neg Pred Value         1.0000   0.7662   0.7532   1.0000
Prevalence             0.2208   0.2338   0.2468   0.2987
Detection Rate         0.2208   0.0000   0.0000   0.2987
Detection Prevalence   0.4589   0.0000   0.0000   0.5411
Balanced Accuracy      0.8472   0.5000   0.5000   0.8272
```

Accuracy: 0.5195

DV: heating

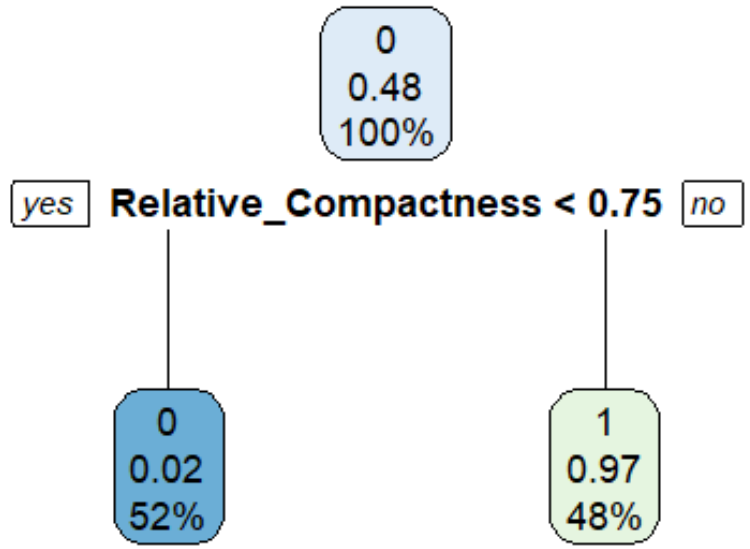
```
> confusionMatrix(energy_pred,energy_test$heating_category)
Confusion Matrix and Statistics

          Reference
Prediction  D  C  B  A
         D 53 50  3  0
         C  0  0  0  0
         B  0  0  0  0
         A  0  1 54 70

Overall Statistics

               Accuracy : 0.5325
                 95% CI : (0.4659, 0.5982)
    No Information Rate : 0.303
    P-Value [Acc > NIR] : 3.474e-13

                  Kappa : 0.3602

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: D Class: C Class: B
Sensitivity            1.0000   0.0000   0.0000
Specificity            0.7022   1.0000   1.0000
Pos Pred Value         0.5000      NaN      NaN
Neg Pred Value         1.0000   0.7792   0.7532
Prevalence             0.2294   0.2208   0.2468
Detection Rate         0.2294   0.0000   0.0000
Detection Prevalence   0.4589   0.0000   0.0000
Balanced Accuracy      0.8511   0.5000   0.5000
                     Class: A
Sensitivity            1.0000
Specificity            0.6584
Pos Pred Value         0.5600
Neg Pred Value         1.0000
Prevalence             0.3030
Detection Rate         0.3030
Detection Prevalence   0.5411
Balanced Accuracy      0.8292
> ### 0.5324675
```

Accuracy: 0.5325

3(g) Decision tree

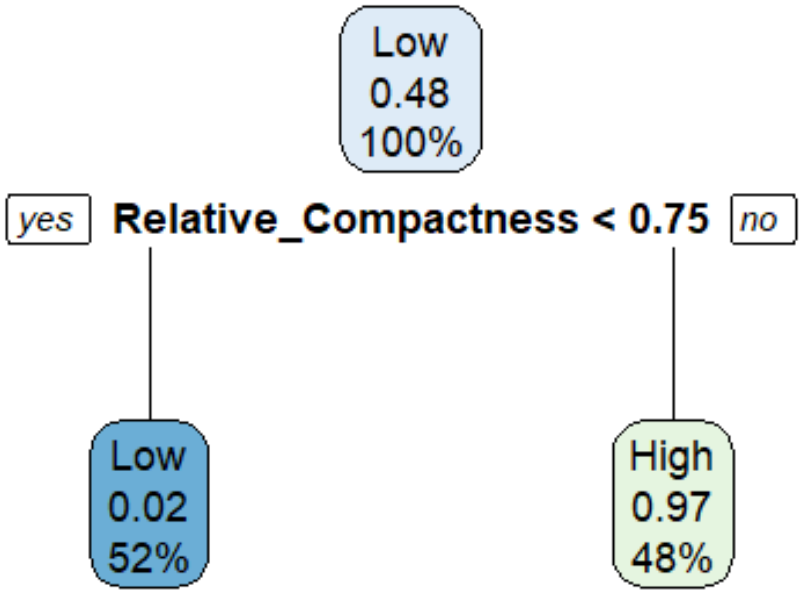
3(g) cooling

```
> energy <- read_xlsx("energy.xlsx")
> convert_to_quartile_categories <- function(x) {
+   quartiles <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))
+   categories <- cut(x,
+                     breaks = quartiles,
+                     labels = c("D", "C", "B", "A"),
+                     include.lowest = TRUE)
+   return(categories)
+ }
> energy$cooling_category <- convert_to_quartile_categories(energy$Cooling_Load)
> energy$heating_category <- convert_to_quartile_categories(energy$Heating_Load)
> energy$cooling_binary <- ifelse(energy$cooling_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- ifelse(energy$heating_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- factor(energy$heating_binary,
+                                 levels = c(0, 1),
+                                 labels = c("Low", "High"))
> energy$cooling_binary <- factor(energy$cooling_binary,
+                                 levels = c(0, 1),
+                                 labels = c("Low", "High"))
> energy <- subset(energy, select = -c(Orientation, Glazing_Area_Distribution, he
ating_category, Heating_Load, Cooling_Load, heating_binary, cooling_category ))
> set.seed(7)
> energy_index <- sample(nrow(energy), 0.7 * nrow(energy))
> energy_train <- energy[energy_index, ]
> energy_test <- energy[-energy_index, ]
> prop.table(table(energy_train$cooling_binary))

      Low      High
0.5195531 0.4804469
> library(rpart)
> library(rpart.plot)
> energytree <- rpart(cooling_binary~ ., data = energy_train, method = 'class')
> rpart.plot(energytree)
```



3(g) heating

```
> energy <- read_xlsx("energy.xlsx")
> convert_to_quartile_categories <- function(x) {
+   quartiles <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))
+   categories <- cut(x,
+                     breaks = quartiles,
+                     labels = c("D", "C", "B", "A"),
+                     include.lowest = TRUE)
+   return(categories)
+ }
> energy$cooling_category <- convert_to_quartile_categories(energy$Cooling_Load)
> energy$heating_category <- convert_to_quartile_categories(energy$Heating_Load)
> energy$cooling_binary <- ifelse(energy$cooling_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- ifelse(energy$heating_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- factor(energy$heating_binary,
+                                 levels = c(0, 1),
+                                 labels = c("Low", "High"))
> energy$cooling_binary <- factor(energy$cooling_binary,
+                                 levels = c(0, 1),
+                                 labels = c("Low", "High"))
> energy <- subset(energy, select = -c(Orientation, Glazing_Area_Distribution, he
ating_category, Heating_Load, Cooling_Load, cooling_binary, cooling_category ))
> set.seed(7)
> energy_index <- sample(nrow(energy), 0.7 * nrow(energy))
> energy_train <- energy[energy_index, ]
> energy_test <- energy[-energy_index, ]
> prop.table(table(energy_train$heating_binary))

      Low      High
0.5214153 0.4785847
> library(rpart)
> library(rpart.plot)
> energytree <- rpart(heating_binary~ ., data = energy_train, method = 'class')
> rpart.plot(energytree)
```



3(g) comparison

DV: cooling

```
> energypredict <- predict(energytree, energy_test[,1:14],
+                          type = 'class')
> #install.packages("caret")
> library("caret")
> confusionMatrix(energypredict, energy_test$cooling_binary)
Confusion Matrix and Statistics

          Reference
Prediction Low High
      Low  105    1
      High   0  125

               Accuracy : 0.9957
                 95% CI : (0.9761, 0.9999)
    No Information Rate : 0.5455
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9913

 Mcnemar's Test P-Value : 1

            Sensitivity : 1.0000
            Specificity : 0.9921
         Pos Pred Value : 0.9906
         Neg Pred Value : 1.0000
             Prevalence : 0.4545
         Detection Rate : 0.4545
   Detection Prevalence : 0.4589
      Balanced Accuracy : 0.9960

       'Positive' Class : Low
```

Accuracy: 0.9957

DV: heating

```
> energypredict <- predict(energytree, energy_test[,1:14],
+                          type = 'class')
> #install.packages("caret")
> library("caret")
> confusionMatrix(energypredict, energy_test$heating_binary)
Confusion Matrix and Statistics

          Reference
Prediction Low High
      Low  103    3
      High   1  124

               Accuracy : 0.9827
                 95% CI : (0.9563, 0.9953)
    No Information Rate : 0.5498
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9651

 Mcnemar's Test P-Value : 0.6171

            Sensitivity : 0.9904
            Specificity : 0.9764
         Pos Pred Value : 0.9717
         Neg Pred Value : 0.9920
             Prevalence : 0.4502
         Detection Rate : 0.4459
   Detection Prevalence : 0.4589
      Balanced Accuracy : 0.9834

       'Positive' Class : Low
```

Accuracy: 0.9827

3(h) Random forest

3(h) DV: cooling

```
> library("randomForest")
> energy <- read_xlsx("energy.xlsx")
> convert_to_quartile_categories <- function(x) {
+   quartiles <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))
+   categories <- cut(x,
+                     breaks = quartiles,
+                     labels = c("D", "C", "B", "A"),
+                     include.lowest = TRUE)
+   return(categories)
+ }
> energy$cooling_category <- convert_to_quartile_categories(energy$Cooling_Load)
> energy$heating_category <- convert_to_quartile_categories(energy$Heating_Load)
> energy$cooling_binary <- ifelse(energy$cooling_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- ifelse(energy$heating_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- factor(energy$heating_binary,
+                                 levels = c(0, 1),
+                                 labels = c("Low", "High"))
> energy$cooling_binary <- factor(energy$cooling_binary,
+                                 levels = c(0, 1),
+                                 labels = c("Low", "High"))
> energy <- subset(energy, select = -c(Orientation, Glazing_Area_Distribution, heating
_category, Heating_Load, Cooling_Load, heating_binary, cooling_category ))
> set.seed(7)
> energy_index <- sample(nrow(energy), 0.7 * nrow(energy))
> energy_train <- energy[energy_index, ]
> energy_test <- energy[-energy_index, ]
> energyforest <- randomForest(cooling_binary~ ., data = energy_train, ntree=500,
+                              proximity=TRUE, importance=TRUE)
> energyforest

Call:
 randomForest(formula = cooling_binary ~ ., data = energy_train,
ntree = 500, proximity = TRUE, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 1.86%
Confusion matrix:
     Low High class.error
Low  275    4  0.01433692
High   6  252  0.02325581
```

```
> energypredict <- predict(energyforest, newdata = energy_test
[,1:14], type = 'class')
> confusionMatrix(energypredict, energy_test$cooling_binary)
Confusion Matrix and Statistics

          Reference
Prediction Low High
      Low  105    1
      High   0  125

               Accuracy : 0.9957
                 95% CI : (0.9761, 0.9999)
    No Information Rate : 0.5455
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9913

 Mcnemar's Test P-Value : 1

            Sensitivity : 1.0000
            Specificity : 0.9921
         Pos Pred Value : 0.9906
         Neg Pred Value : 1.0000
             Prevalence : 0.4545
         Detection Rate : 0.4545
   Detection Prevalence : 0.4589
      Balanced Accuracy : 0.9960

       'Positive' Class : Low

> ##0.9957
```

3(h) DV: heating

```
> library("randomForest")
> energy <- read_xlsx("energy.xlsx")
> convert_to_quartile_categories <- function(x) {
+   quartiles <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))
+   categories <- cut(x,
+                     breaks = quartiles,
+                     labels = c("D", "C", "B", "A"),
+                     include.lowest = TRUE)
+   return(categories)
+ }
> energy$cooling_category <- convert_to_quartile_categories(energy$Cooling_Load)
> energy$heating_category <- convert_to_quartile_categories(energy$Heating_Load)
> energy$cooling_binary <- ifelse(energy$cooling_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- ifelse(energy$heating_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- factor(energy$heating_binary,
+                                 levels = c(0, 1),
+                                 labels = c("Low", "High"))
> energy$cooling_binary <- factor(energy$cooling_binary,
+                                 levels = c(0, 1),
+                                 labels = c("Low", "High"))
> energy <- subset(energy, select = -c(Orientation, Glazing_Area_Distribution, he
ating_category, Heating_Load, Cooling_Load, cooling_binary, cooling_category ))
> set.seed(7)
> energy_index <- sample(nrow(energy), 0.7 * nrow(energy))
> energy_train <- energy[energy_index, ]
> energy_test <- energy[-energy_index, ]
> energyforest <- randomForest(heating_binary~ ., data = energy_train, ntree=500,
+                              proximity=TRUE, importance=TRUE)
> energyforest
```

```
Call:
 randomForest(formula = heating_binary ~ ., data = energy_train,      ntree = 50
0, proximity = TRUE, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 1.68%
Confusion matrix:
     Low High class.error
Low  277    3  0.01071429
High   6  251  0.02334630
```

```
> energypredict <- predict(energyforest, newdata = energy_test[,1:14],
type = 'class')
> confusionMatrix(energypredict, energy_test$heating_binary)
Confusion Matrix and Statistics

          Reference
Prediction Low High
      Low  103    3
      High   1  124

               Accuracy : 0.9827
                 95% CI : (0.9563, 0.9953)
    No Information Rate : 0.5498
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9651

 Mcnemar's Test P-Value : 0.6171

            Sensitivity : 0.9904
            Specificity : 0.9764
         Pos Pred Value : 0.9717
         Neg Pred Value : 0.9920
             Prevalence : 0.4502
         Detection Rate : 0.4459
   Detection Prevalence : 0.4589
      Balanced Accuracy : 0.9834

       'Positive' Class : Low

> ##0.0.9827
```

3(i) Boosting

3(i) DV: cooling

```
> library("randomForest")
> energy <- read_xlsx("energy.xlsx")
> convert_to_quartile_categories <- function(x) {
+   quartiles <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))
+   categories <- cut(x,
+                     breaks = quartiles,
+                     labels = c("D", "C", "B", "A"),
+                     include.lowest = TRUE)
+   return(categories)
+ }
> energy$cooling_category <- convert_to_quartile_categories(energy$Cooling_Load)
> energy$heating_category <- convert_to_quartile_categories(energy$Heating_Load)
> energy$cooling_binary <- ifelse(energy$cooling_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- ifelse(energy$heating_category %in% c("A", "B"), 1, 0)
> energy <- subset(energy, select = -c(Orientation, Glazing_Area_Distribution, he
ating_category, Heating_Load, Cooling_Load, heating_binary, cooling_category ))
> set.seed(7)
> energy_index <- sample(nrow(energy), 0.7 * nrow(energy))
> x_train <- as.matrix(energy[energy_index,1:14])
> y_train <- energy[energy_index,15,drop=TRUE]
> x_test <- as.matrix(energy[-energy_index,1:14])
> y_test <- energy[-energy_index,15,drop=TRUE]
> library(xgboost)
> dtrain <- xgb.DMatrix(data = x_train, label = y_train)
> energyxgb <- xgboost(data = dtrain, max.depth = 5, eta = 1, nthread = 2,
+                      nrounds = 1000,objective = "binary:logistic",verbose = 0)
>
> energypredict <- predict(energyxgb, x_test)
> head(energypredict)
[1] 9.938075e-01 6.822284e-01 9.947772e-01 9.997812e-01 9.932132e-01 1.143190e-09
> energypredict <- as.numeric(energypredict > 0.5)
> head(energypredict)
[1] 1 1 1 1 1 0
```

```
> confusionMatrix(factor(energypredict),factor(y_test))
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 105   1
         1   0 125

               Accuracy : 0.9957
                 95% CI : (0.9761, 0.9999)
    No Information Rate : 0.5455
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9913

 Mcnemar's Test P-Value : 1

            Sensitivity : 1.0000
            Specificity : 0.9921
         Pos Pred Value : 0.9906
         Neg Pred Value : 1.0000
             Prevalence : 0.4545
         Detection Rate : 0.4545
   Detection Prevalence : 0.4589
      Balanced Accuracy : 0.9960

       'Positive' Class : 0

> ###0.0.9957
```

3(i) DV: heating

```
> library("randomForest")
> energy <- read_xlsx("energy.xlsx")
> convert_to_quartile_categories <- function(x) {
+   quartiles <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))
+   categories <- cut(x,
+                     breaks = quartiles,
+                     labels = c("D", "C", "B", "A"),
+                     include.lowest = TRUE)
+   return(categories)
+ }
> energy$cooling_category <- convert_to_quartile_categories(energy$Cooling_Load)
> energy$heating_category <- convert_to_quartile_categories(energy$Heating_Load)
> energy$cooling_binary <- ifelse(energy$cooling_category %in% c("A", "B"), 1, 0)
> energy$heating_binary <- ifelse(energy$heating_category %in% c("A", "B"), 1, 0)
> energy <- subset(energy, select = -c(Orientation, Glazing_Area_Distribution, heat
ing_category, Heating_Load, Cooling_Load, cooling_binary, cooling_category ))
> set.seed(7)
> energy_index <- sample(nrow(energy), 0.7 * nrow(energy))
> x_train <- as.matrix(energy[energy_index,1:14])
> y_train <- energy[energy_index,15,drop=TRUE]
> x_test <- as.matrix(energy[-energy_index,1:14])
> y_test <- energy[-energy_index,15,drop=TRUE]
> library(xgboost)
> dtrain <- xgb.DMatrix(data = x_train, label = y_train)
> energyxgb <- xgboost(data = dtrain, max.depth = 5, eta = 1, nthread = 2,
+                      nrounds = 1000,objective = "binary:logistic",verbose = 0)
>
> energypredict <- predict(energyxgb, x_test)
> head(energypredict)
[1] 8.769583e-01 8.858487e-01 9.960819e-01 9.960819e-01 9.970587e-01 4.586229e-11
> energypredict <- as.numeric(energypredict > 0.5)
> head(energypredict)
[1] 1 1 1 1 1 0
```

```
> confusionMatrix(factor(energypredict),factor(y_test))
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0  99   3
         1   5 124

               Accuracy : 0.9654
                 95% CI : (0.9329, 0.9849)
    No Information Rate : 0.5498
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9299

 Mcnemar's Test P-Value : 0.7237

            Sensitivity : 0.9519
            Specificity : 0.9764
         Pos Pred Value : 0.9706
         Neg Pred Value : 0.9612
             Prevalence : 0.4502
         Detection Rate : 0.4286
   Detection Prevalence : 0.4416
      Balanced Accuracy : 0.9642

       'Positive' Class : 0

> ###0.9654
```

Method		Cooling				Heating			
		Accuracy	NIR	P-value	Significant?	Accuracy	NIR	P-value	Significant?
Perceptrons (best model)		0.9812	0.5368	<2.2e-16	YES	0.9766	0.5368	<2.2e-16	YES
SVM		0.9610	0.5325	<2.2e-16	YES	0.9610	0.5325	<2.2e-16	YES
Neural Network	Node = 1	0.7662	0.2771	<2.2e-16	YES	0.7835	0.2684	<2.2e-16	YES
	Node = 2	0.7576	0.2771	<2.2e-16	YES	0.7835	0.2684	<2.2e-16	YES
	Node = 3	0.7403	0.2771	<2.2e-16	YES	0.8268	0.2684	<2.2e-16	YES
	Node = 4	0.7403	0.2771	<2.2e-16	YES	0.8095	0.2684	<2.2e-16	YES
	Node = 5	0.7835	0.2771	<2.2e-16	YES	0.7965	0.2684	<2.2e-16	YES
K-nearest	K=5	0.6523	0.2987	<2.2e-16	YES	0.6759	0.3030	<2.2e-16	YES
	K=7	0.6494	0.2987	<2.2e-16	YES	0.6753	0.3030	<2.2e-16	YES
Naïve bayes		0.5195	0.2987	2.078e-12	YES	0.5325	0.3030	3.474e-13	YES
Decision tree		0.9957	0.5455	<2e-16	YES	0.9827	0.5498	<2e-16	YES
Random forest		0.9957	0.5455	<2e-16	YES	0.9827	0.5498	<2e-16	YES
Boosting		0.9957	0.5455	<2e-16	YES	0.9654	0.5498	<2e-16	YES

4. Lessons Learned

Which technique worked best? Justify by using the accuracies of your models when possible.

- If only looking at accuracy rate, decision tree, random forest, boosting worked the best. They all have the highest accuracy rate of 0.9957 for predicting cooling categories, and 0.9827 for predicting heating categories. The no information rates are 0.5455 and 0.5498 relatively. The p-values are all equals to 0, which shows that the model is better than random guessing
- However, these methods perform better partially because we reduce the variations of dependent variables for those models. (e.g., from continuous --> 4 --> 2 categories).
  - If we take account into the variation of dependent variables, neural network did a good job too in predicting continuous variables.

What insights do you have for builders of energy efficient buildings? What should they do to reduce cooling loads? What should they do to reduce heating loads?

Heating

	Low	High	MeanDecreaseAccuracy	MeanDecreaseGini
Relative_Compactness	14.62596053	14.545571	14.649960	65.5104663
Surface_Area	13.73628743	13.632912	13.738306	58.5011722
Wall_Area	10.23626977	9.587888	10.773045	8.1961694
Roof_Area	13.39019704	13.374925	13.459355	57.2016905
Overall_Height	13.15143550	13.297356	13.285478	57.1495503
North_Or	-6.75988526	-4.077085	-7.667284	0.4740749
East_Or	-3.72958387	-3.370329	-4.852650	0.5180595
South_Or	-3.90651474	-4.611873	-6.386411	0.4173750
Glazing_Area	25.25637593	18.435475	28.551884	9.1125303
Uniform	0.68833757	5.941626	5.311090	1.1400686
North	-0.07292229	3.136873	2.619627	0.5351335
East	2.69350070	4.910656	5.598635	0.3780917
South	-1.93889218	3.646664	1.558124	0.3221363
West	4.88090809	5.034044	6.731009	0.4660417

Cooling

	Low	High	MeanDecreaseAccuracy	MeanDecreaseGini
Relative_Compactness	14.0498493	13.5749081	13.864109	61.6504884
Surface_Area	13.9696652	13.7566835	13.936819	59.9742842
Wall_Area	9.9960372	9.9904452	11.072411	9.3966730
Roof_Area	13.2103420	13.3501541	13.367688	56.6515229
Overall_Height	13.4038388	13.5535182	13.556726	59.4447912
North_Or	-2.8915394	0.3426788	-2.072015	0.4970665
East_Or	-1.7256062	-3.9433085	-4.036582	0.6374922
South_Or	-5.1823686	-2.0607331	-5.211837	0.5728588
Glazing_Area	21.2868525	13.9867456	23.106493	8.1620201
Uniform	-2.2652941	5.8303410	2.559019	1.1714747
North	-2.1724762	2.5803805	1.014268	0.3659726
East	1.6671445	4.3526616	4.534058	0.3452947
South	-0.4350009	2.5966100	1.974207	0.3347015
West	7.3956345	4.9540937	7.961014	0.4592505

- Glazing area
- Relative compactness
- Surface area
- Roof area
- Overall height

What insights do you have for builders of energy efficient buildings? What should they do to reduce cooling loads? What should they do to reduce heating loads?

- Correlation matrix shows that relative compactness has strong negative correlation with cooling/heating load, which is verified in the decision tree.

TAKEAWAY 1:
Larger relative compactness of a building will use less energy for heating and cooling
- Larger surface area

- Linear regression shows that Overall height has strong positive correlation with cooling/heating load

TAKEAWAY 2:
Build lower-height structure is better for energy efficiency

What insights do you have for builders of energy efficient buildings? What should they do to reduce cooling loads? What should they do to reduce heating loads?

Build energy efficient building for both heating & cooling:

- Reduce overall height
- Minimize roof area
- Minimize surface area
- Optimize relative compactness

International Renewable Energy Agency headquarters
Abu Dhabi


Thanks!