Stat435 Final Project

Anthony Chelf, Hansol Lee

12/10/2021

Dataset

Our dataset is based on energy efficiency with regards to various building types. There are 8 attributes covering 768 instances and two individual response variables. The instances were generated in a simulation utilizing Ecotect. Lastly, this data is non-gaussian in nature.

Attribute/Response Details:

- X1: Relative Compactness, 12 potential values
- X2: Surface Area, 12 potential values, unit: m^2
- X3: Wall Area, 7 potential values, unit: m^2
- X4: Roof Area, 4 potential values, unit: m^2
- X5: Overall Height, 2 potential values, unit: m
- X6: Orientation, 4 potential values, values: 2(North), 3(East), 4(South), 5(West)
- X7: Glazing area (ratio), 4 potential values, values: 0.00, 0.10, 0.25, 0.40
- X8: Glazing area distribution, 6 potential values, values: 0(None), 1(Uniform), 2(North), 3(East), 4(South), 5(West)
- Y1: Heating Load - kWh/m^2
- Y2: Cooling Load - kWh/m^2

More information can be found at: https://archive.ics.uci.edu/ml/datasets/Energy+efficiency

What we want to know

- How is each predictor variable related to each response variable?
  - We believe that on their own, each predictor would have a significant relationship with each response. Our intent is to see what happens when they aren't alone through multiple linear regression modeling.
  - H0: Relative Compactness has no effect on Heating/Cooling Load.
  - H0: Surface Area has no effect on Heating/Cooling Load.
  - H0: Wall Area has no effect on Heating/Cooling Load.
  - H0: Roof Area has no effect on Heating/Cooling Load.
  - H0: OVerall Height has no effect on Heating/Cooling Load.

  - H0: Orientation has no effect on Heating/Cooling Load.
  - H0: Glazing Area has no effect on Heating/Cooling Load.
  - H0: Glazing Area Distribution has no effect on Heating/Cooling Load.
- Would each predictor variable have a similar relationship with both individual response variables?
  - Do any variables affect one response positively and the other negatively?
  - Due to the dataset being about energy efficiency and how modern architecture is created with thermodynamics in mind, we expect the variables to interact in a similar manner with both heating/cooling load.
- Which attribute impacts energy efficiency the most? Are they the same for heating and cooling loads?

Loading our dataset and adding the libraries to be used

```
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.0.5
```

```
library(corrplot)
```

```
## corrplot 0.90 loaded
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'tibble' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

```
## Warning: package 'readr' was built under R version 4.0.5
```

```
## Warning: package 'purrr' was built under R version 4.0.5
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## Warning: package 'stringr' was built under R version 4.0.5
```

```
## Warning: package 'forcats' was built under R version 4.0.5
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.0.5
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.5
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(grid)
data <- read_xlsx("ENB2012_data.xlsx")
datay1 <- subset(data, select = -c(Y2))
datay2 <- subset(data, select = -c(Y1))
```

```
# setting the names of our variables for ease of use/identification later on
RelativeCompactness <- data$X1
SurfaceArea <- data$X2
WallArea <- data$X3
RoofArea <- data$X4
OverallHeight <- data$X5
Orientation <- data$X6
GlazingArea <- data$X7
GlazingAreaDist <- data$X8
HeatingLoad <- data$Y1
CoolingLoad <- data$Y2
```

Performing some initial data checking/cleaning up if necessary

The repository for the data says that there aren't missing values, but we are checking just to be sure that there aren't any NA or Null values present ahead of inspecting the data or building any models.

```
sum(is.na(data))
```

```
## [1] 0
```

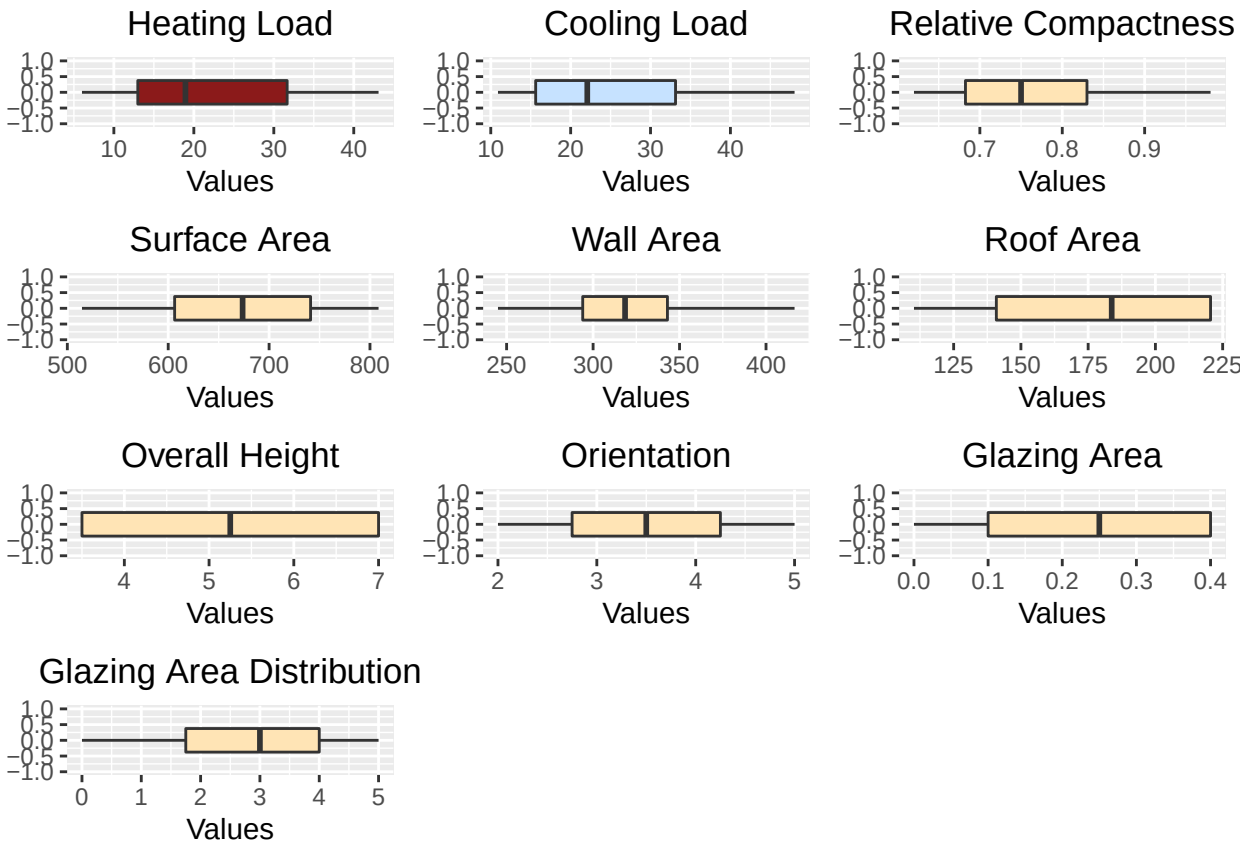
```
is.null(data)
```

```
## [1] FALSE
```

As expected, there are no NAs or null values in the dataset. Next we'll check for outliers, although we don't expect any due to the nature of the attributes having a set number of possible values.

```
box1 = ggplot(data, aes(y=Y1)) +
  geom_boxplot(fill="firebrick4") + xlim(-1, 1) + coord_flip() +
  labs(title="Heating Load", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

box2 = ggplot(data, aes(y=Y2)) +
  geom_boxplot(fill="slategray1") + xlim(-1, 1) + coord_flip() +
  labs(title="Cooling Load", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))
```

```
box3 = ggplot(data, aes(y=X1)) +
  geom_boxplot(fill="moccasin") + xlim(-1, 1) + coord_flip() +
  labs(title="Relative Compactness", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

box4 = ggplot(data, aes(y=X2)) +
  geom_boxplot(fill="moccasin") + xlim(-1, 1) + coord_flip() +
  labs(title="Surface Area", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

box5 = ggplot(data, aes(y=X3)) +
  geom_boxplot(fill="moccasin") + xlim(-1, 1) + coord_flip() +
  labs(title="Wall Area", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

box6 = ggplot(data, aes(y=X4)) +
  geom_boxplot(fill="moccasin") + xlim(-1, 1) + coord_flip() +
  labs(title="Roof Area", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

box7 = ggplot(data, aes(y=X5)) +
  geom_boxplot(fill="moccasin") + xlim(-1, 1) + coord_flip() +
  labs(title="Overall Height", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

box8 = ggplot(data, aes(y=X6)) +
  geom_boxplot(fill="moccasin") + xlim(-1, 1) + coord_flip() +
  labs(title="Orientation", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

box9 = ggplot(data, aes(y=X7)) +
  geom_boxplot(fill="moccasin") + xlim(-1, 1) + coord_flip() +
  labs(title="Glazing Area", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

box10 = ggplot(data, aes(y=X8)) +
  geom_boxplot(fill="moccasin") + xlim(-1, 1) + coord_flip() +
  labs(title="Glazing Area Distribution", y="Values") +
  theme(plot.title = element_text(hjust = 0.5))

grid.arrange(box1, box2, box3, box4, box5, box6, box7, box8, box9, box10)
```

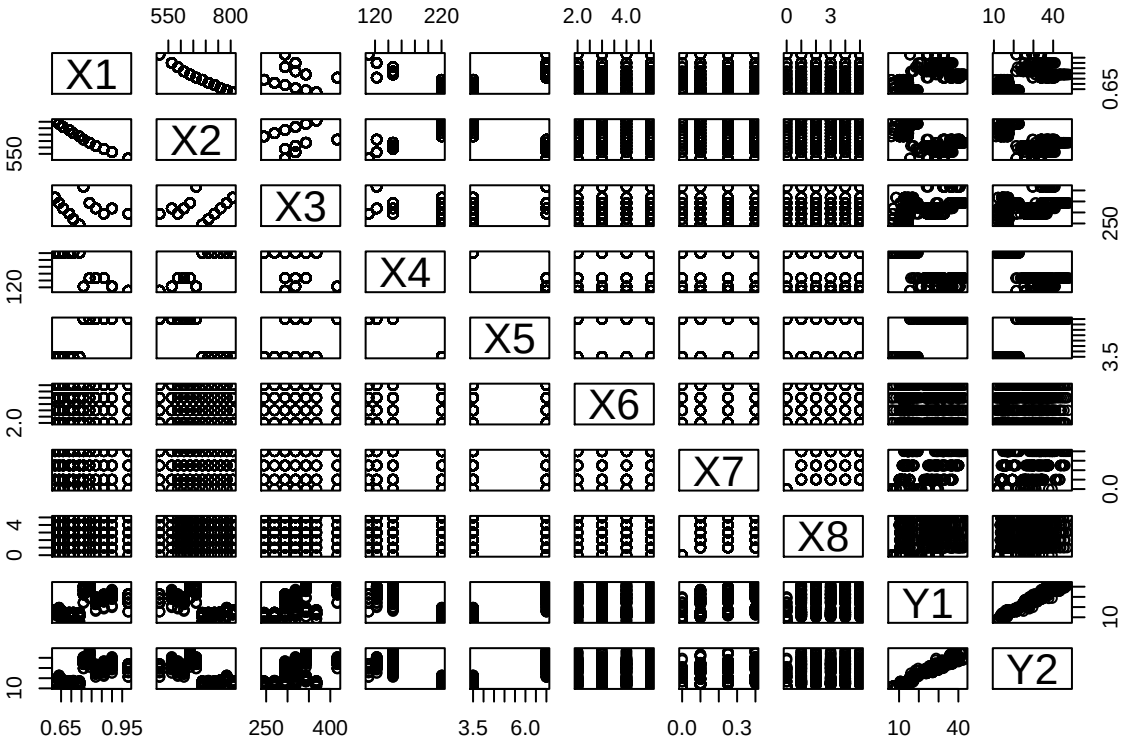


Again, as expected, we can see from the above plots that there are no outliers.

Initial exploratory data analysis

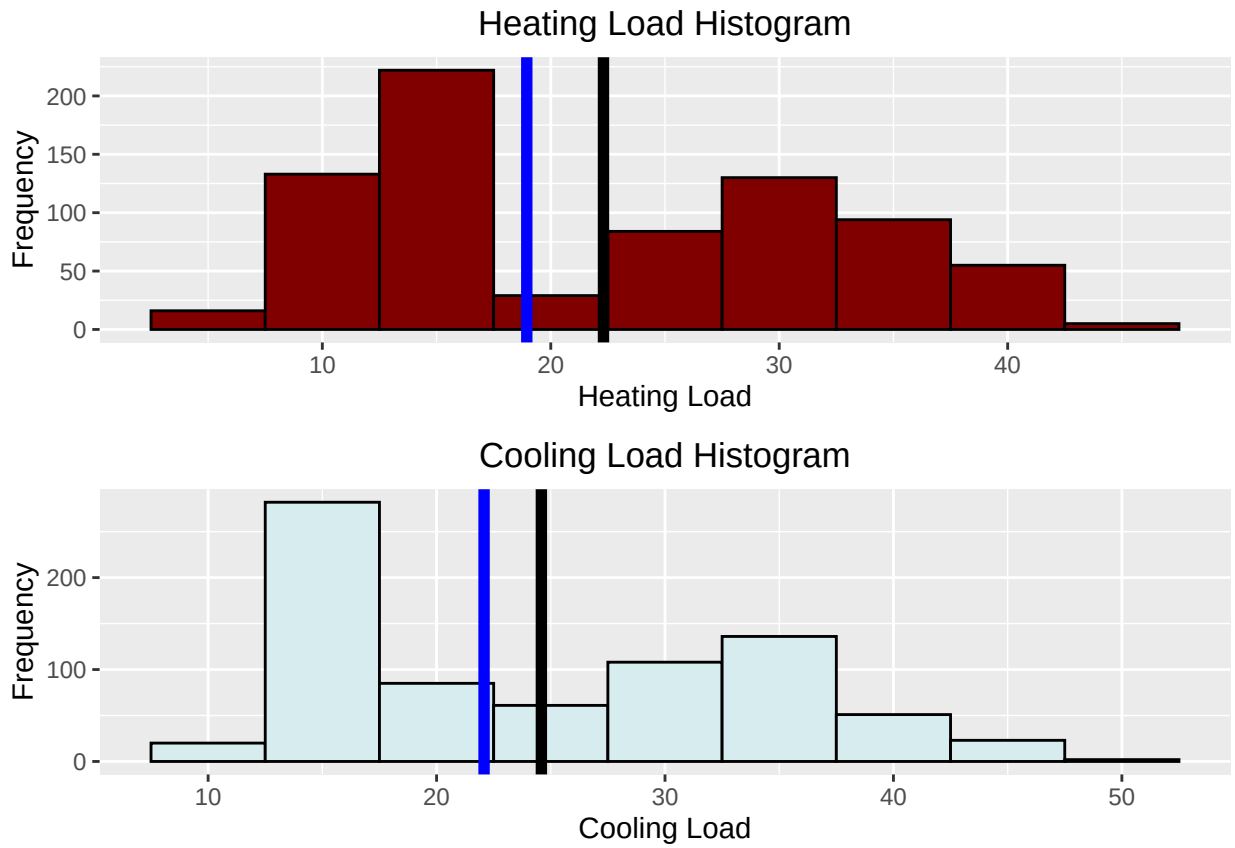
We are going to start by just looking at the dataset.

```
pairs(data)
```




```r
# generating a histogram for Heating Load
hist.hl <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = HeatingLoad),
                 fill  = "#800000",
                 color = "black",
                 binwidth = 5) +
  ggtitle("Heating Load Histogram") +
  xlab("Heating Load") +
  ylab("Frequency") +
  geom_vline(xintercept = mean(HeatingLoad), color="black", lwd=2) +
  geom_vline(xintercept = median(HeatingLoad), color = "blue",  lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))

# generating a histogram for Cooling Load
hist.cl <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = CoolingLoad),
                 fill  = "#d6ecef",
                 color = "black",
                 binwidth = 5) +
  ggtitle("Cooling Load Histogram") +
  xlab("Cooling Load") +
  ylab("Frequency") +
  geom_vline(xintercept = mean(CoolingLoad), color="black", lwd=2) +
  geom_vline(xintercept = median(CoolingLoad), color = "blue",  lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))
```

```
grid.arrange(hist.hl, hist.cl)
```

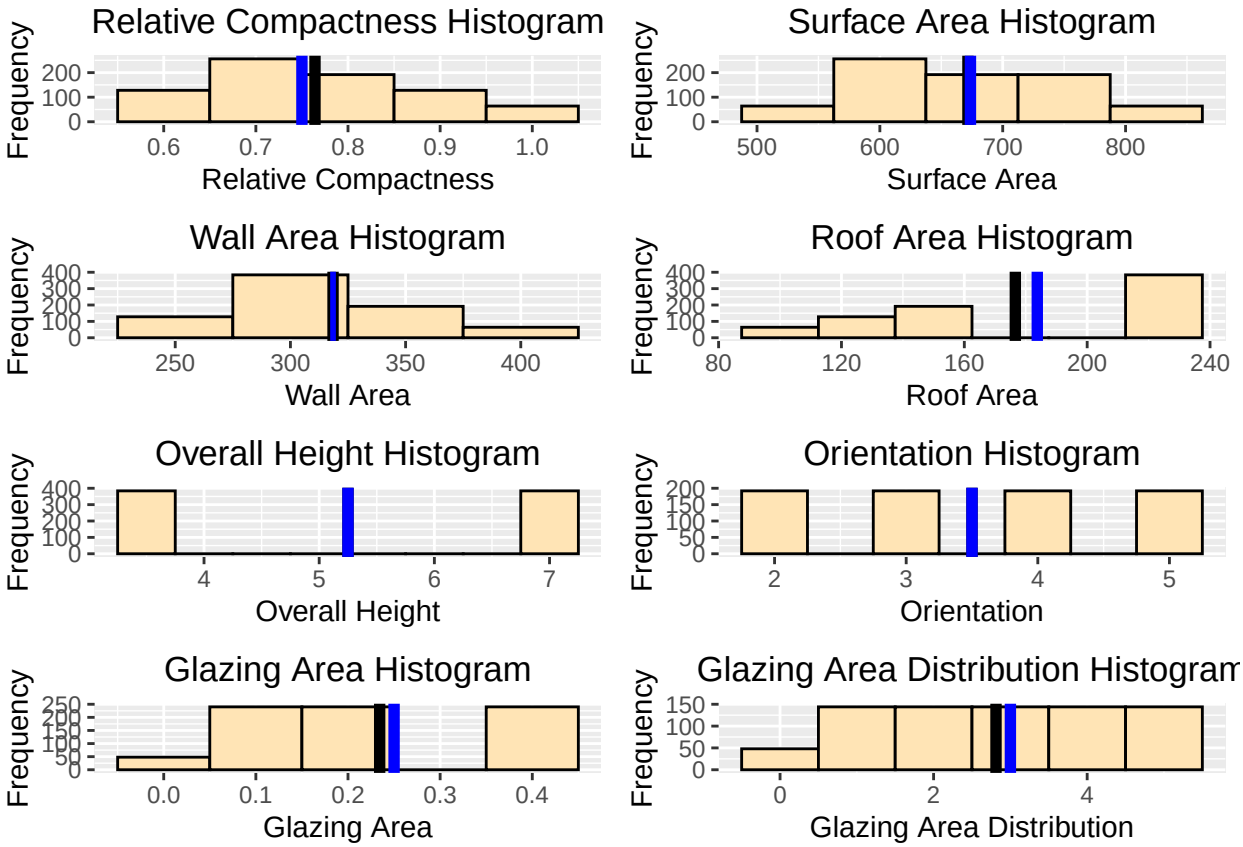


For both these plots, the mean is denoted by the vertical blue line and the median by the vertical blue line. These histograms show that both Heating Load and Cooling Load are right skewed and that the means are greater than the medians.

```
# generating a histogram for Relative Compactness
hist.rc <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = RelativeCompactness),
                 fill  = "moccasin",
                 color = "black",
                 binwidth = .1) +
  ggtitle("Relative Compactness Histogram") +
  xlab("Relative Compactness") +
  ylab("Frequency") +
  geom_vline(xintercept = mean(RelativeCompactness), color="black", lwd=2) +
  geom_vline(xintercept = median(RelativeCompactness), color="blue", lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))

# generating a histogram for Surface Area
hist.sa <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = SurfaceArea),
                 fill  = "moccasin",
                 color = "black",
                 binwidth = 75) +
  ggtitle("Surface Area Histogram") +
  xlab("Surface Area") +
```

```
  ylab("Frequency") +
  geom_vline(xintercept = mean(SurfaceArea), color="black", lwd=2) +
  geom_vline(xintercept = median(SurfaceArea), color="blue", lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))

# generating a histogram for Wall Area
hist.wa <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = WallArea),
                 fill  = "moccasin",
                 color = "black",
                 binwidth = 50) +
  ggtitle("Wall Area Histogram") +
  xlab("Wall Area") +
  ylab("Frequency") +
  geom_vline(xintercept = mean(WallArea), color="black",lwd=2) +
  geom_vline(xintercept = median(WallArea), color="blue", lwd=1) +
  theme(plot.title = element_text(hjust = 0.5))

# generating a histogram for Roof Area
hist.ra <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = RoofArea),
                 fill  = "moccasin",
                 color = "black",
                 binwidth = 25) +
  ggtitle("Roof Area Histogram") +
  xlab("Roof Area") +
  ylab("Frequency") +
  geom_vline(xintercept = mean(RoofArea), color="black", lwd=2) +
  geom_vline(xintercept = median(RoofArea), color="blue", lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))

# generating a histogram for Overall Height
hist.oh <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = OverallHeight),
                 fill  = "moccasin",
                 color = "black",
                 binwidth = .5) +
  ggtitle("Overall Height Histogram") +
  xlab("Overall Height") +
  ylab("Frequency") +
  geom_vline(xintercept = mean(OverallHeight), color="black", lwd=2) +
  geom_vline(xintercept = median(OverallHeight), color="blue", lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))

# generating a histogram for Orientation
hist.or <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = Orientation),
                 fill  = "moccasin",
                 color = "black",
                 binwidth = .5) +
  ggtitle("Orientation Histogram") +
  xlab("Orientation") +
  ylab("Frequency") +
```

```
  geom_vline(xintercept = mean(Orientation), color="black", lwd=2) +
  geom_vline(xintercept = median(Orientation), color="blue", lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))

# generating a histogram for Glazing Area
hist.ga <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = GlazingArea),
                 fill  = "moccasin",
                 color = "black",
                 binwidth = .1) +
  ggtitle("Glazing Area Histogram") +
  xlab("Glazing Area") +
  ylab("Frequency") +
  geom_vline(xintercept = mean(GlazingArea), color="black", lwd=2) +
  geom_vline(xintercept = median(GlazingArea), color="blue", lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))

# generating a histogram for Glazing Area Distribution
hist.gd <- ggplot(data = data) +
  geom_histogram(mapping = aes(x = GlazingAreaDist),
                 fill  = "moccasin",
                 color = "black",
                 binwidth = 1) +
  ggtitle("Glazing Area Distribution Histogram") +
  xlab("Glazing Area Distribution") +
  ylab("Frequency") +
  geom_vline(xintercept = mean(GlazingAreaDist), color="black", lwd=2) +
  geom_vline(xintercept = median(GlazingAreaDist), color="blue", lwd=2) +
  theme(plot.title = element_text(hjust = 0.5))

# arranging the 8 attribute histograms nicely
grid.arrange(hist.rc, hist.sa, hist.wa, hist.ra, hist.oh, hist.or, hist.ga, hist.gd, nrow=4)
```

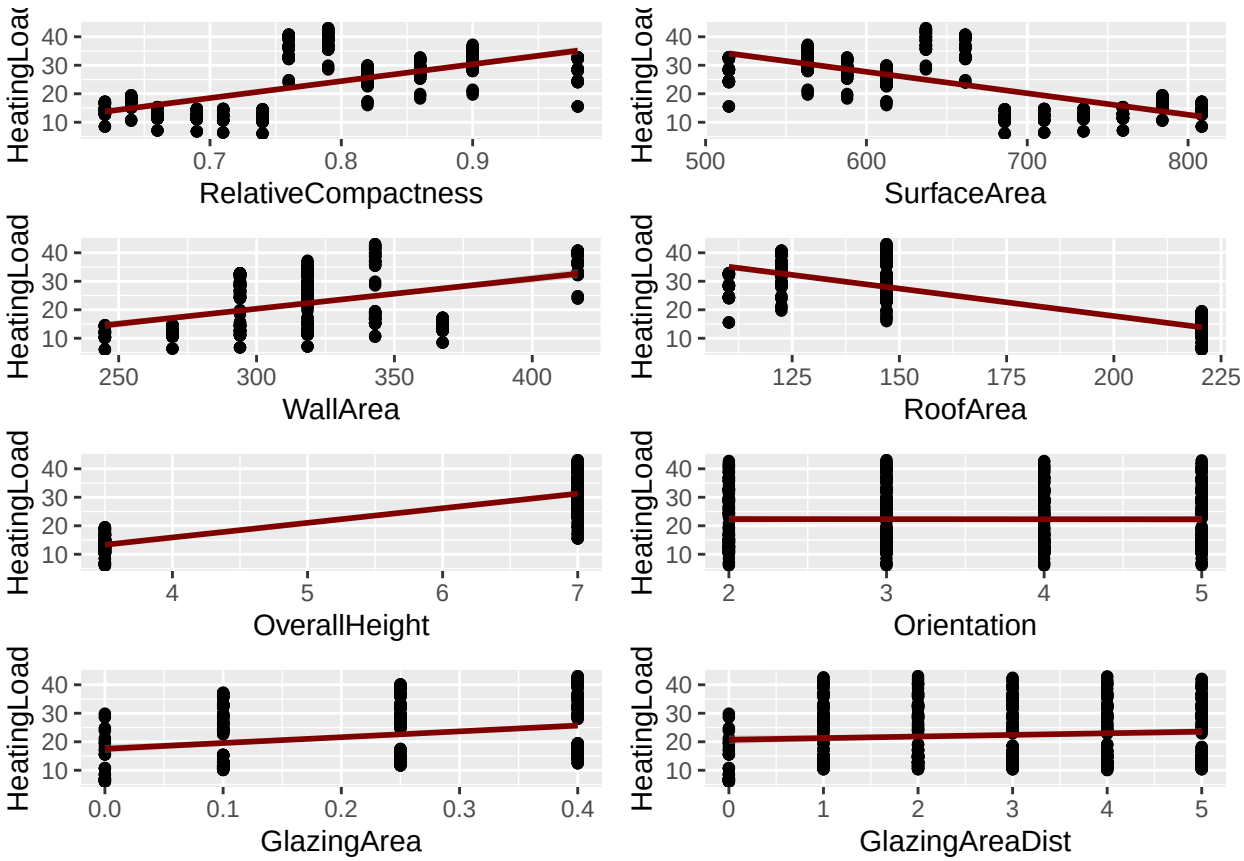


Based on the small number of variations for values in each predictor and there being just 768 instances, these histograms can be considered expected/normally distributed.

```
h1 <- ggplot(data, aes(x = RelativeCompactness, y = HeatingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
h2 <- ggplot(data, aes(x = SurfaceArea, y = HeatingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
h3 <- ggplot(data, aes(x = WallArea, y = HeatingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
h4 <- ggplot(data, aes(x = RoofArea, y = HeatingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
h5 <- ggplot(data, aes(x = OverallHeight, y = HeatingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
h6 <- ggplot(data, aes(x = Orientation, y = HeatingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
h7 <- ggplot(data, aes(x = GlazingArea, y = HeatingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
h8 <- ggplot(data, aes(x = GlazingAreaDist, y = HeatingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
```

```
grid.arrange(h1, h2, h3, h4, h5, h6, h7, h8, nrow=4)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

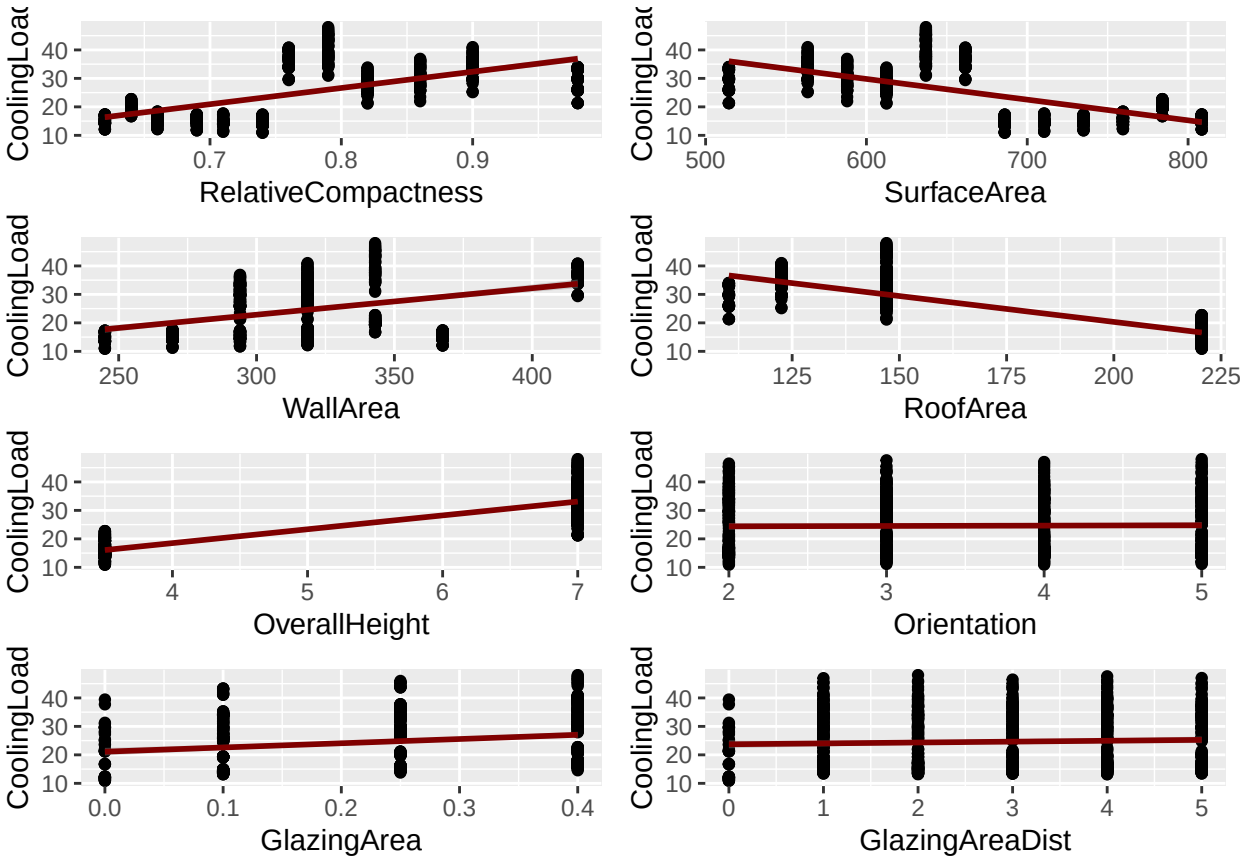


Like histograms above, these plots are to be expected due to the low amount of variance in values for each predictor.

```
c1 <- ggplot(data, aes(x = RelativeCompactness, y = CoolingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
c2 <- ggplot(data, aes(x = SurfaceArea, y = CoolingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
c3 <- ggplot(data, aes(x = WallArea, y = CoolingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
c4 <- ggplot(data, aes(x = RoofArea, y = CoolingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
c5 <- ggplot(data, aes(x = OverallHeight, y = CoolingLoad)) +
```

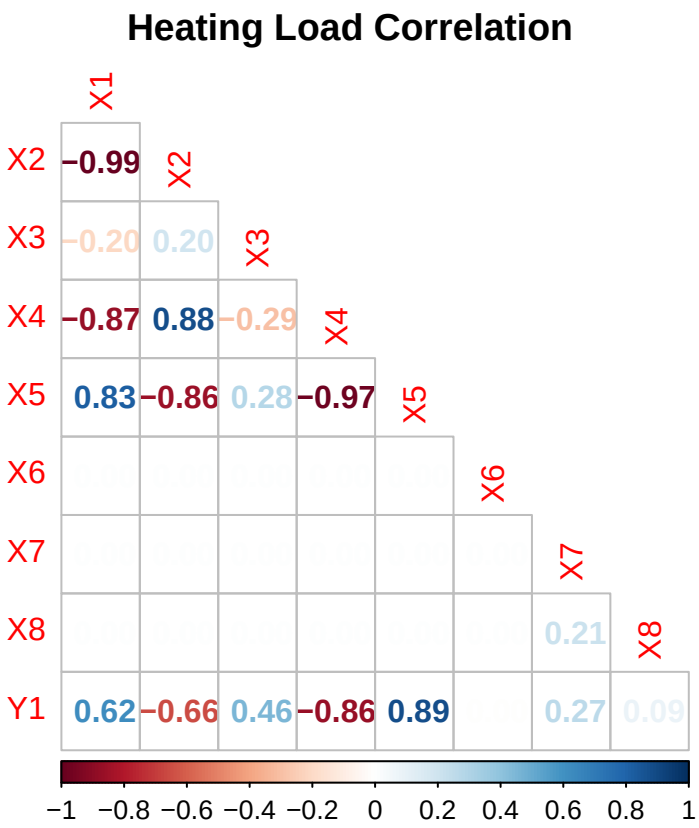
```
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
c6 <- ggplot(data, aes(x = Orientation, y = CoolingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
c7 <- ggplot(data, aes(x = GlazingArea, y = CoolingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")
c8 <- ggplot(data, aes(x = GlazingAreaDist, y = CoolingLoad)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, color = "#800000")

grid.arrange(c1, c2, c3, c4, c5, c6, c7, c8, nrow=4)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



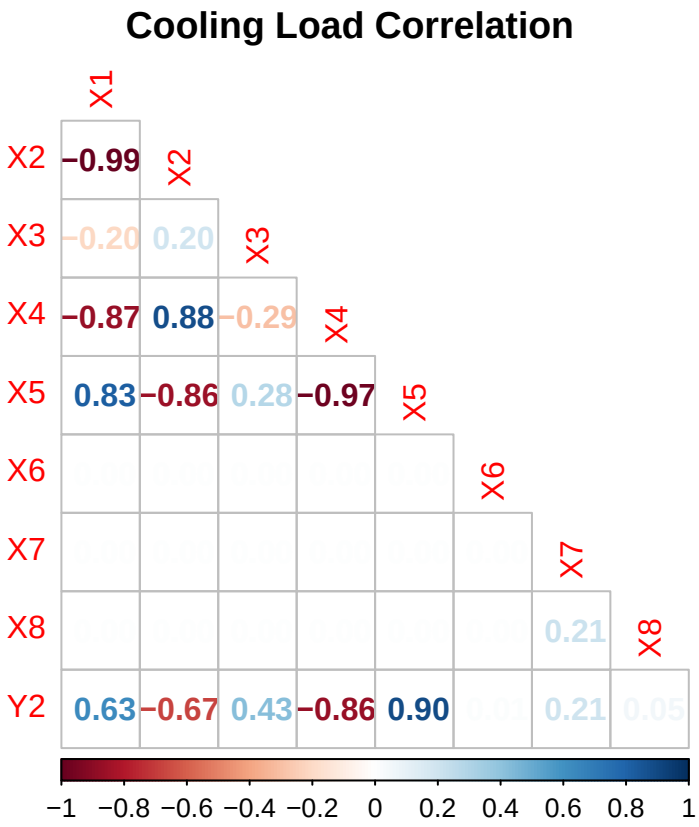
Like histograms above, these plots are to be expected due to the low amount of variance in values for each predictor.

```
# generating a plot for correlation that will return a colorful numbered output to
# help the reader interpret the results
cor.datay1 <- cor(datay1)
corrplot(cor.datay1, method = 'number', title = "Heating Load Correlation",
         type = 'lower', mar=c(0,0,2,0), diag = FALSE)
```



Here we see the existing correlations to heating load. There is a moderate positive correlation to relative compactness, a moderate negative correlation to surface area, a weak correlation to wall area, a strong negative correlation to roof area, a strong positive correlation to overall height, no real correlation to orientation, a weak correlation to glazing area, and a weaker correlation to glazing area distribution.

```
# generating a plot for correlation that will return a colorful numbered output to
# help the reader interpret the results
cor.datay2 <- cor(datay2)
corrplot(cor.datay2, method = 'number', title = "Cooling Load Correlation",
         type = 'lower', mar=c(0,0,2,0), diag = FALSE)
```



Here we see the existing correlations to cooling load. There is a moderate positive correlation to relative compactness, a moderate negative correlation to surface area, a weak correlation to wall area, a strong negative correlation to roof area, a strong positive correlation to overall height, no real correlation to orientation, a weak correlation to glazing area, and a weaker correlation to glazing area distribution.

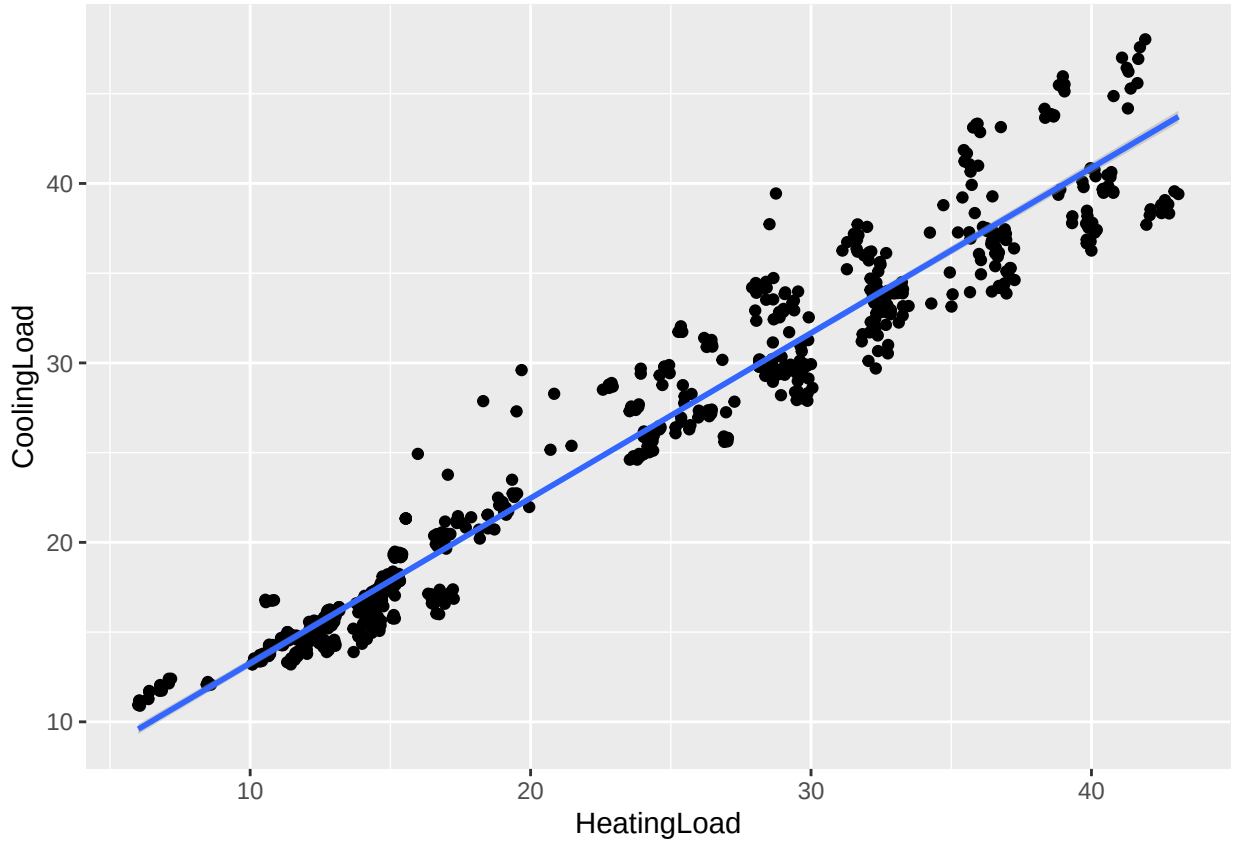
```
cor(data$Y1, data$Y2)
```

```
## [1] 0.9758617
```

After observing the correlations the predictors had to the response and them being fairly identical and yielding the same effective observation by us, we knew that there would be a strong positive correlation (~.976) between Heating Load and Cooling Load.

```
ggplot(data, aes(x=HeatingLoad, y=CoolingLoad)) + geom_point() + stat_smooth(method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Here we are going to create linear models for both our response variables. After that we'll utilize forward and backwards step-wise selection to help us in selecting the best predictors for our linear regression model. We are electing to go with multiple linear regression because after testing models where there was a single independent variable, it was clearly evident that both heating load and cooling load had significant statistical dependence on each of the predictors.

```
# creating a linear model for heating load using all predictors
lm.y1 <- lm(Y1 ~ ., data = datay1)
summary(lm.y1)
```

```
##
## Call:
## lm(formula = Y1 ~ ., data = datay1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8965 -1.3196 -0.0252  1.3532  7.7052
##
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  84.013418  19.033613   4.414 1.16e-05 ***
## X1          -64.773432  10.289448  -6.295 5.19e-10 ***
## X2           -0.087289   0.017075  -5.112 4.04e-07 ***
## X3            0.060813   0.006648   9.148  < 2e-16 ***
## X4                  NA         NA      NA       NA
## X5            4.169954   0.337990  12.338  < 2e-16 ***
## X6           -0.023330   0.094705  -0.246  0.80548
```

```
## X7           19.932736   0.813986  24.488  < 2e-16 ***
## X8            0.203777   0.069918   2.915  0.00367 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.934 on 760 degrees of freedom
## Multiple R-squared:  0.9162, Adjusted R-squared:  0.9154
## F-statistic:  1187 on 7 and 760 DF,  p-value: < 2.2e-16
```

Notice the p-value of the F-statistic is highly significant with value $< 2.2e\text{-}16$. This tells us that at least one of the predictor variables is significantly related to the response variable. And to find out which predictor variable(s) is/are significant, we take a look at the t-statistic p-values of each predictor variable. We can observe that X3, X5, and X7 are significantly associated with the response variable. And while not as significant, X1 and X2 are still very small for a p-value, so we'll maintain them for now. The rest are considered insignificant in the multiple regression model, so we will likely exclude them from the model.

```
# getting confidence intervals for our heating load lm fit
confint(lm.y1)
```

```
##                    2.5 %       97.5 %
## (Intercept)  46.64871750 121.37811881
## X1          -84.97254762 -44.57431616
## X2           -0.12081010  -0.05376864
## X3            0.04776273   0.07386372
## X4                    NA           NA
## X5            3.50644865   4.83345877
## X6           -0.20924412   0.16258370
## X7           18.33480805  21.53066492
## X8            0.06652187   0.34103181
```

```
step(lm.y1, direction = "forward")
```

```
## Start:  AIC=1661.42
## Y1 ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8
```

```
##
## Call:
## lm(formula = Y1 ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8, data = datay1)
##
## Coefficients:
## (Intercept)           X1           X2           X3           X4           X5
##    84.01342    -64.77343     -0.08729      0.06081           NA      4.16995
##          X6           X7           X8
##    -0.02333     19.93274      0.20378
```

```
step(lm.y1, direction = "backward")
```

```
## Start:  AIC=1661.42
## Y1 ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8
##
##
## Step:  AIC=1661.42
## Y1 ~ X1 + X2 + X3 + X5 + X6 + X7 + X8
##
##        Df Sum of Sq     RSS    AIC
## - X6    1       0.5  6544.3 1659.5
## <none>               6543.8 1661.4
```

```
## - X8    1      73.1  6616.9 1668.0
## - X2    1     225.0  6768.8 1685.4
## - X1    1     341.2  6885.0 1698.5
## - X3    1     720.5  7264.3 1739.7
## - X5    1    1310.6  7854.4 1799.6
## - X7    1    5163.1 11706.9 2106.1
##
## Step:  AIC=1659.49
## Y1 ~ X1 + X2 + X3 + X5 + X7 + X8
##
##        Df Sum of Sq     RSS    AIC
## <none>               6544.3 1659.5
## - X8    1      73.1  6617.4 1666.0
## - X2    1     225.0  6769.3 1683.5
## - X1    1     341.2  6885.5 1696.5
## - X3    1     720.5  7264.8 1737.7
## - X5    1    1310.6  7854.9 1797.7
## - X7    1    5163.1 11707.4 2104.2
##
## Call:
## lm(formula = Y1 ~ X1 + X2 + X3 + X5 + X7 + X8, data = datay1)
##
## Coefficients:
## (Intercept)           X1           X2           X3           X5           X7
##    83.93176    -64.77343     -0.08729      0.06081      4.16995     19.93274
##          X8
##     0.20378
```

```r
# creating a linear model for cooling load using all predictors
lm.y2 <- lm(Y2 ~ ., data = datay2)
summary(lm.y2)
```

```
##
## Call:
## lm(formula = Y2 ~ ., data = datay2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.6940 -1.5606 -0.2668  1.3968 11.1775
##
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  97.245749  20.764711   4.683 3.34e-06 ***
## X1          -70.787707  11.225269  -6.306 4.85e-10 ***
## X2           -0.088245   0.018628  -4.737 2.59e-06 ***
## X3            0.044682   0.007253   6.161 1.17e-09 ***
## X4                  NA         NA      NA       NA
## X5            4.283843   0.368730  11.618  < 2e-16 ***
## X6            0.121510   0.103318   1.176    0.240
## X7           14.717068   0.888018  16.573  < 2e-16 ***
## X8            0.040697   0.076277   0.534    0.594
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 3.201 on 760 degrees of freedom
## Multiple R-squared:  0.8878, Adjusted R-squared:  0.8868
## F-statistic: 859.1 on 7 and 760 DF,  p-value: < 2.2e-16
```

```
# getting confidence intervals for our cooling load lm fit
confint(lm.y2)
```

```
##                    2.5 %       97.5 %
## (Intercept)  56.48274688 138.00875181
## X1          -92.82392274 -48.75149129
## X2           -0.12481438  -0.05167553
## X3            0.03044467   0.05891954
## X4                    NA           NA
## X5            3.55999279   5.00769386
## X6           -0.08131228   0.32433311
## X7           12.97380905  16.46032759
## X8           -0.10904099   0.19043551
```

```
step(lm.y2, direction = "forward")
```

```
## Start:  AIC=1795.13
## Y2 ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8
```

```
##
## Call:
## lm(formula = Y2 ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8, data = datay2)
##
## Coefficients:
## (Intercept)           X1           X2           X3           X4           X5
##    97.24575    -70.78771     -0.08824      0.04468           NA      4.28384
##          X6           X7           X8
##     0.12151     14.71707      0.04070
```

```
step(lm.y2, direction = "backward")
```

```
## Start:  AIC=1795.13
## Y2 ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8
##
##
## Step:  AIC=1795.13
## Y2 ~ X1 + X2 + X3 + X5 + X6 + X7 + X8
##
##        Df Sum of Sq     RSS    AIC
## - X8    1      2.92  7791.1 1793.4
## - X6    1     14.17  7802.4 1794.5
## <none>               7788.2 1795.1
## - X2    1    229.96  8018.2 1815.5
## - X3    1    388.96  8177.2 1830.6
## - X1    1    407.52  8195.7 1832.3
## - X5    1   1383.16  9171.4 1918.7
## - X7    1   2814.64 10602.8 2030.1
##
## Step:  AIC=1793.42
## Y2 ~ X1 + X2 + X3 + X5 + X6 + X7
##
##        Df Sum of Sq     RSS    AIC
## - X6    1     14.17  7805.3 1792.8
```

```
## <none>               7791.1 1793.4
## - X2    1    229.96  8021.1 1813.8
## - X3    1    388.96  8180.1 1828.8
## - X1    1    407.52  8198.6 1830.6
## - X5    1   1383.16  9174.3 1916.9
## - X7    1   2988.93 10780.0 2040.8
##
## Step:  AIC=1792.81
## Y2 ~ X1 + X2 + X3 + X5 + X7
##
##        Df Sum of Sq     RSS    AIC
## <none>               7805.3 1792.8
## - X2    1    229.96  8035.3 1813.1
## - X3    1    388.96  8194.3 1828.2
## - X1    1    407.52  8212.8 1829.9
## - X5    1   1383.16  9188.5 1916.1
## - X7    1   2988.93 10794.2 2039.8
##
## Call:
## lm(formula = Y2 ~ X1 + X2 + X3 + X5 + X7, data = datay2)
##
## Coefficients:
## (Intercept)           X1           X2           X3           X5           X7
##    97.76185    -70.78771     -0.08824      0.04468      4.28384     14.81797
```

After performing step-wise selection and reviewing the summary on both of our lm fits, we've elected to drop X4 (Roof Area) from the future models. Further, Roof Area itself appears to be dependent on some of the other predictors. Going to run a model again after dropping Roof Area to see if anything changes. It is also clear that X6 (orientation) has no statistical significant relationship with Heating Load or Cooling Load. Additionally, glazing area distribution has no statistical significance with Cooling Load and a fairly low relationship with Heating Load. We will drop that as well.

```
# dropping X4 since it both produces NAs and doesn't make the cut for two different
# step-wise selections dropping X6 due to its statistical irrelevance and X8
# due to its almost completely statistical irrelevance
datay1 <- subset(datay1, select = -c(X4, X6, X8))
datay2 <- subset(datay2, select = -c(X4, X6, X8))
```

```
# updating our linear model
lm.y1 <- lm(Y1 ~ ., data = datay1)
summary(lm.y1)
```

```
##
## Call:
## lm(formula = Y1 ~ ., data = datay1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.3862  -1.3667  -0.0142   1.3162   7.5555
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  84.386471  19.111765   4.415 1.15e-05 ***
## X1          -64.773432  10.333611  -6.268 6.11e-10 ***
## X2           -0.087289   0.017149  -5.090 4.51e-07 ***
```

```
## X3            0.060813   0.006676   9.109  < 2e-16 ***
## X5            4.169954   0.339441  12.285  < 2e-16 ***
## X7           20.437968   0.798727  25.588  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.947 on 762 degrees of freedom
## Multiple R-squared:  0.9153, Adjusted R-squared:  0.9147
## F-statistic:  1646 on 5 and 762 DF,  p-value: < 2.2e-16
```

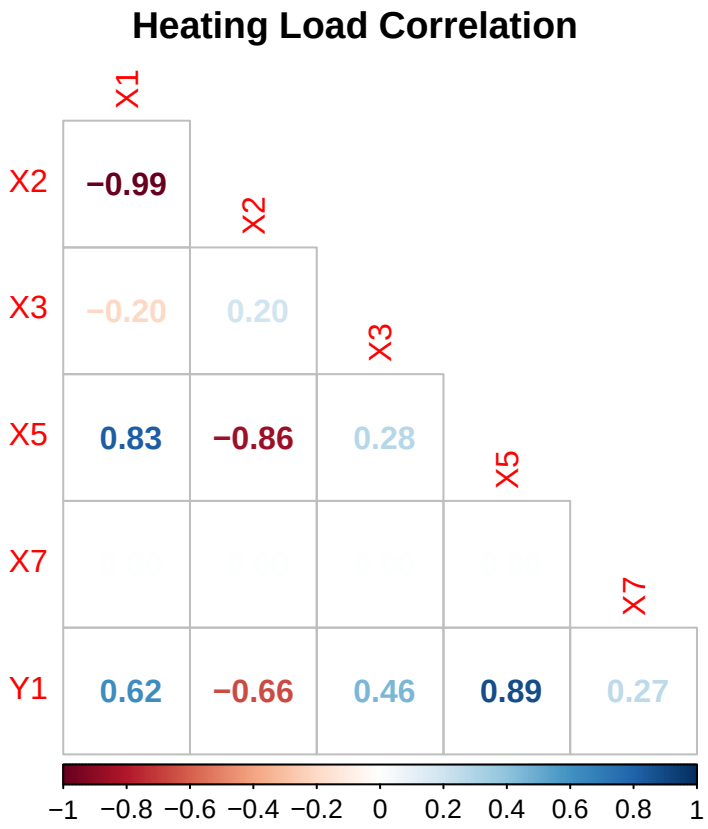
```
# updating our linear model
lm.y2 <- lm(Y2 ~ ., data = datay2)
summary(lm.y2)
```

```
##
## Call:
## lm(formula = Y2 ~ ., data = datay2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.7240 -1.6017 -0.2631  1.3417 11.3251
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  97.761848  20.756339   4.710 2.94e-06 ***
## X1          -70.787707  11.222822  -6.307 4.80e-10 ***
## X2           -0.088245   0.018624  -4.738 2.57e-06 ***
## X3            0.044682   0.007251   6.162 1.16e-09 ***
## X5            4.283843   0.368650  11.620  < 2e-16 ***
## X7           14.817971   0.867458  17.082  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.2 on 762 degrees of freedom
## Multiple R-squared:  0.8876, Adjusted R-squared:  0.8868
## F-statistic:  1203 on 5 and 762 DF,  p-value: < 2.2e-16
```

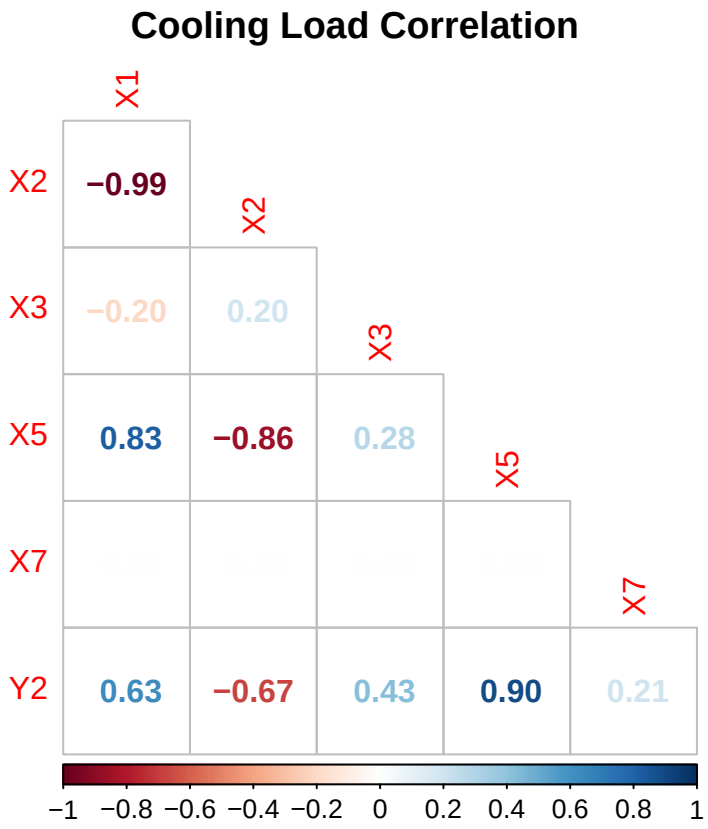
Now that we've updated our models, everything that remains in both of them has statistical significance.

We brought updated versions of the correlation plots down here for easy reference.

```
# generating a plot for correlation that will return a colorful numbered output to
# help the reader interpret the results
cor.datay1 <- cor(datay1)
corrplot(cor.datay1, method = 'number', title = "Heating Load Correlation",
         type = 'lower', mar=c(0,0,2,0), diag = FALSE)
```




```
# generating a plot for correlation that will return a colorful numbered output to
# help the reader interpret the results
cor.datay2 <- cor(datay2)
corrplot(cor.datay2, method = 'number', title = "Cooling Load Correlation",
         type = 'lower', mar=c(0,0,2,0), diag = FALSE)
```



Next we will take our trimmed down Heating Load data set and split it into training and testing sets.

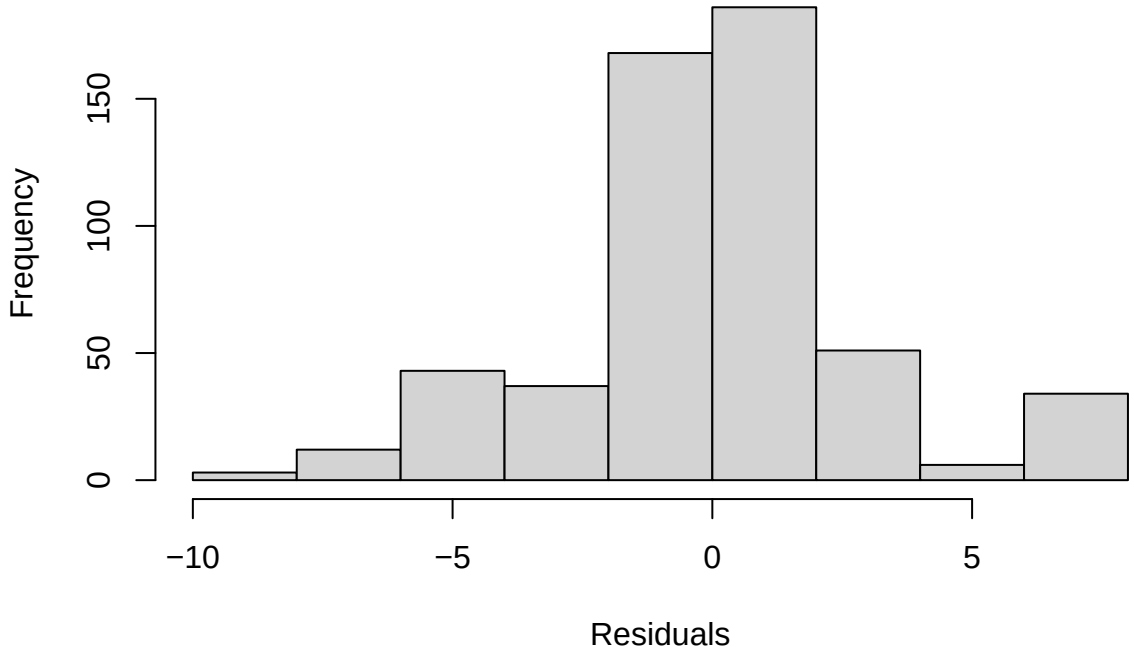
```
set.seed(1)
# splitting the data into smaller training and test datasets at a 70:30 split
sample <- sample(2, nrow(datay1), replace = TRUE, prob = c(0.7,0.3))
train_datay1 <- data[sample == 1,]
test_datay1 <- data[sample == 2,]
```

```
# generating a linear model with our statistically significant predictors
lm.y1_1 <- lm(Y1 ~ X1 + X2 + X3 + X5 + X7, data = train_datay1)
summary(lm.y1_1)
```

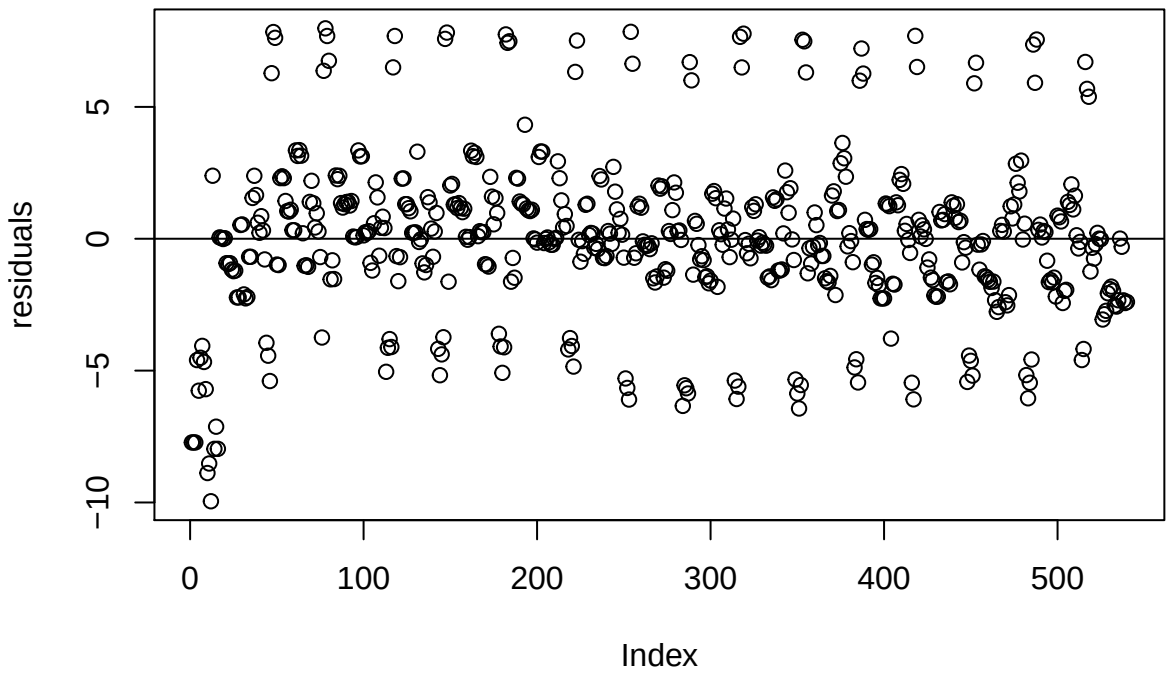
```
##
## Call:
## lm(formula = Y1 ~ X1 + X2 + X3 + X5 + X7, data = train_datay1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.9535 -1.4655  0.0474  1.3121  7.9782
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  83.45820   23.57064   3.541 0.000434 ***
## X1          -61.42587   12.77992  -4.806  2.0e-06 ***
## X2           -0.09024    0.02102  -4.293  2.1e-05 ***
## X3            0.06836    0.00794   8.610  < 2e-16 ***
## X5            3.76317    0.40832   9.216  < 2e-16 ***
## X7           20.41559    0.97244  20.994  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.01 on 534 degrees of freedom
## Multiple R-squared:  0.9106, Adjusted R-squared:  0.9098
## F-statistic:  1088 on 5 and 534 DF,  p-value: < 2.2e-16
```

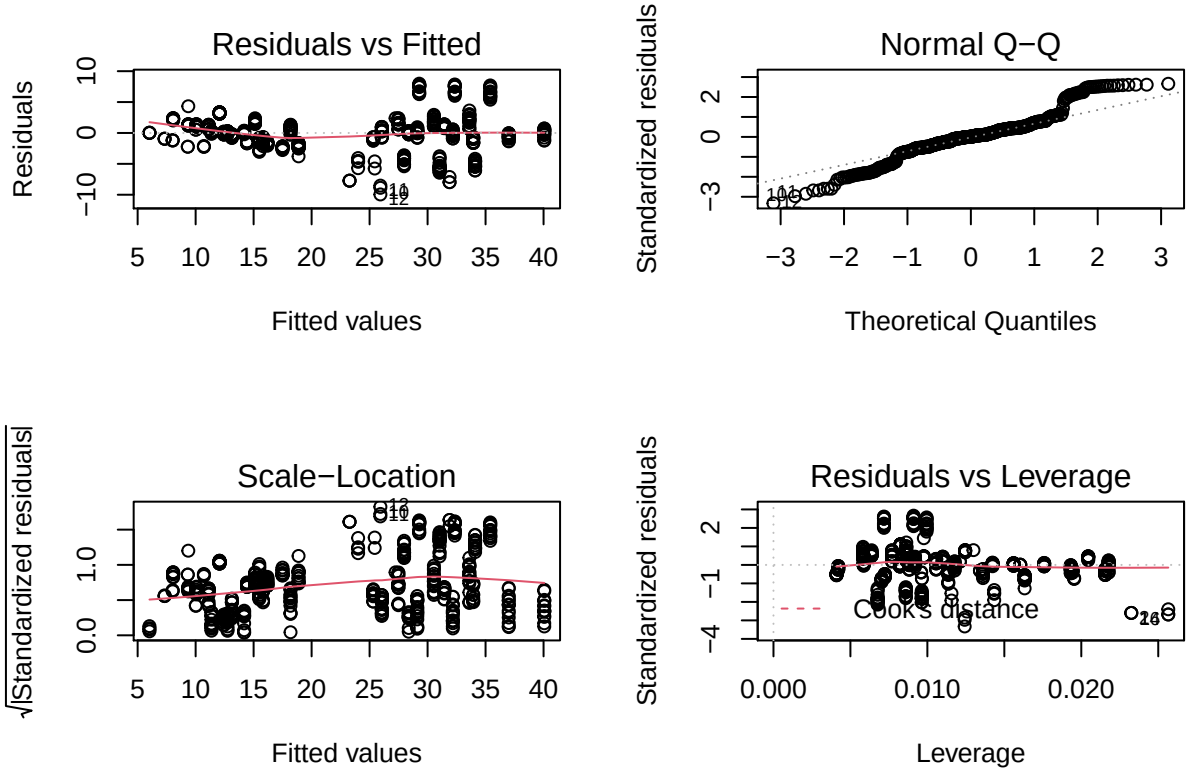
```r
# Checking the residuals of lm.fit1
hist(lm.y1_1$residuals, xlab = 'Residuals', main = '')
```




```r
# Checking the plot of the residuals to ensure there aren't any cone shaped formations
plot(lm.y1_1$residuals, ylab = 'residuals')
abline(a = 0, b = 0)
```




```
par(mfrow = c(2, 2))
plot(lm.y1_1)
```




```
lm.predy1 <- predict(lm.y1_1, test_datay1)
pred1.R2 <- R2(lm.predy1, test_datay1$Y1)
pred1.RMSE <- RMSE(lm.predy1, test_datay1$Y1)
pred1.error <- pred1.RMSE/mean(test_datay1$Y1)
paste("R:", pred1.R2)
```

```
## [1] "R: 0.925747914608117"
```

```
paste("RMSE:", pred1.RMSE)
```

```
## [1] "RMSE: 2.82338002492399"
```

```
paste("Error:", pred1.error)
```

```
## [1] "Error: 0.129908813013"
```

The R^2 at ~.93 shows us the observations are highly correlated to the predicted values. The RMSE is ~2.82 and there is an error rate of ~12.99%. The significant variables in our model are X1, X2, X3, X5, and X7.

```
# generating a quick RSE for reference against our next linear models
sigma(lm.y1_1)/mean(datay1$Y1)
```

```
## [1] 0.1349469
```

Now, to confirm that multiple regression is the preferred model, we'll take a look at linear regression models for each predictor variables: X1, X2, X3, X5, and X7.

```
# linear regression models for X1, X2, X3, X5, X7 and their residual standard error (RSE)
lm.y1_2 = lm(Y1 ~ X1, data = datay1)
lm.y1_3 = lm(Y1 ~ X2, data = datay1)
```

```
lm.y1_4 = lm(Y1 ~ X3, data = datay1)
lm.y1_5 = lm(Y1 ~ X5, data = datay1)
lm.y1_6 = lm(Y1 ~ X7, data = datay1)

sigma(lm.y1_2)/mean(data$Y1)
```

```
## [1] 0.3543152
```

```
sigma(lm.y1_3)/mean(data$Y1)
```

```
## [1] 0.3407871
```

```
sigma(lm.y1_4)/mean(data$Y1)
```

```
## [1] 0.402903
```

```
sigma(lm.y1_5)/mean(data$Y1)
```

```
## [1] 0.2068814
```

```
sigma(lm.y1_6)/mean(data$Y1)
```

```
## [1] 0.4358345
```

RSEs of simple linear regression models are significantly higher than of the multiple regression model. Here, we conclude that multiple regression model is most definitely the better model. To further confirm that multiple regression of all five predictor variables is the best model, we try different combinations of variables in a few additional models.

```
# multiple regression models using a few different combinations
lm.y1_7 = lm(Y1 ~ X1 + X5, data = datay1)
lm.y1_8 = lm(Y1 ~ X3 + X7, data = datay1)
lm.y1_9 = lm(Y1 ~ X2 + X7, data = datay1)
lm.y1_10 = lm(Y1 ~ X3 + X5, data = datay1)
sigma(lm.y1_7)/mean(datay1$Y1)
```

```
## [1] 0.1854595
```

```
sigma(lm.y1_8)/mean(datay1$Y1)
```

```
## [1] 0.3841954
```

```
sigma(lm.y1_9)/mean(datay1$Y1)
```

```
## [1] 0.3183563
```

```
sigma(lm.y1_10)/mean(datay1$Y1)
```

```
## [1] 0.1828285
```

Upon comparing our original model, our models with a single predictor, and our models with random combinations of our predictors, our original model (lm.y1_1) is our best multiple regression model by taking into account all of the predictors. Because Heating Load and Cooling Load behave so similarly with the predictors, we won't run all of these comparisons again and will trust the initial model we create.

Just like we did with Heating Load, we will now take our trimmed down data set for Cooling Load and split it into training and testing sets.

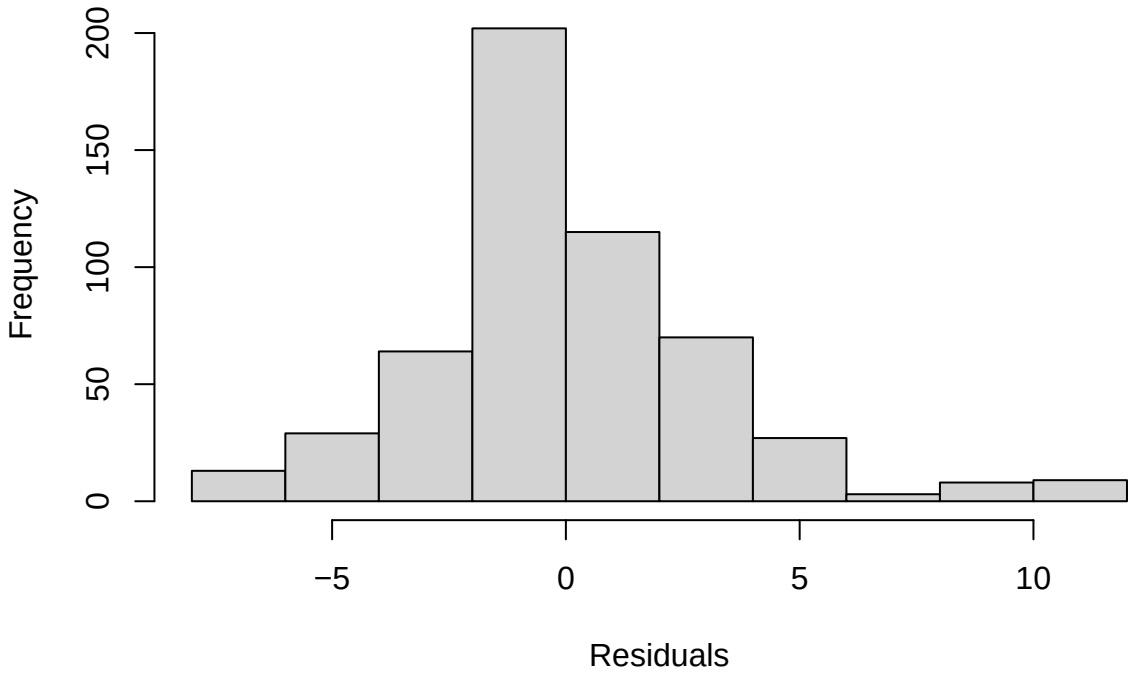
```
set.seed(1)
# splitting the data into smaller training and test datasets at a 70:30 split
sample2 <- sample(2, nrow(datay2), replace = TRUE, prob = c(0.7, 0.3))
```

```r
train_datay2 <- data[sample == 1,]
test_datay2 <- data[sample == 2,]
```

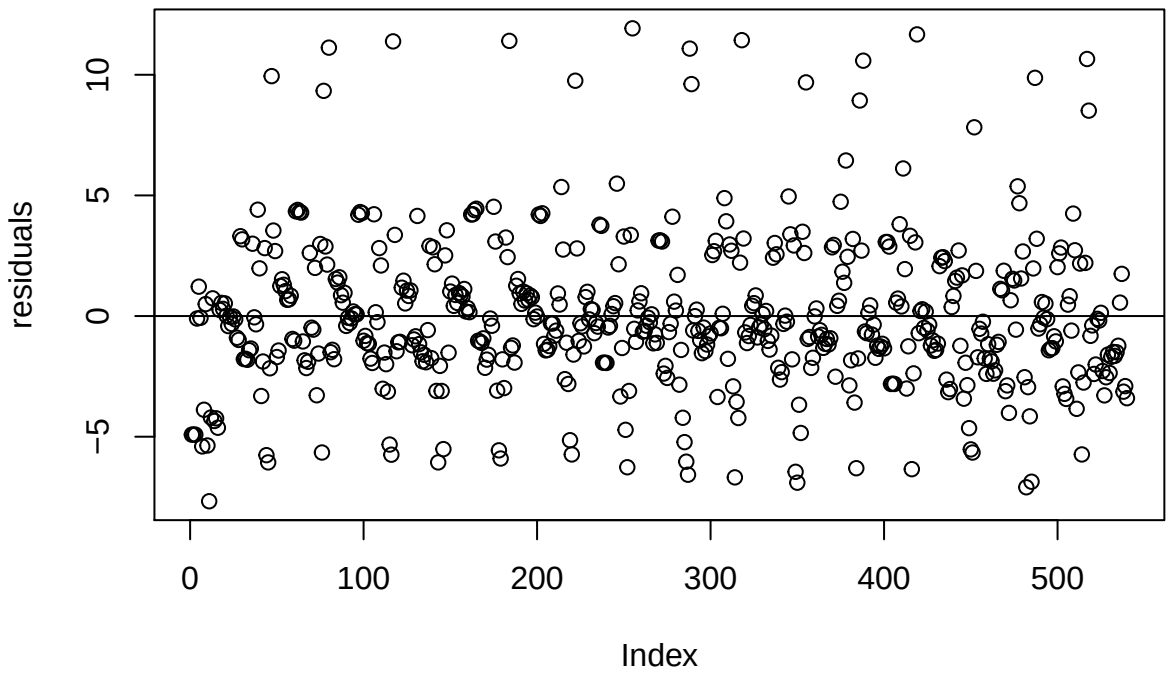
```r
# generating a linear model with our statistically significant predictors
lm.y2_1 <- lm(Y2 ~ X1 + X2 + X3 + X5 + X7, data = train_datay2)
summary(lm.y2_1)
```

```
##
## Call:
## lm(formula = Y2 ~ X1 + X2 + X3 + X5 + X7, data = train_datay2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6742 -1.7139 -0.3935  1.4680 11.9234
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 103.74852   24.75729   4.191 3.26e-05 ***
## X1          -69.97495   13.42332  -5.213 2.66e-07 ***
## X2           -0.09889    0.02208  -4.479 9.18e-06 ***
## X3            0.05654    0.00834   6.779 3.20e-11 ***
## X5            3.61840    0.42888   8.437 3.07e-16 ***
## X7           15.40323    1.02139  15.081  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.162 on 534 degrees of freedom
## Multiple R-squared:  0.8875, Adjusted R-squared:  0.8865
## F-statistic: 842.8 on 5 and 534 DF,  p-value: < 2.2e-16
```

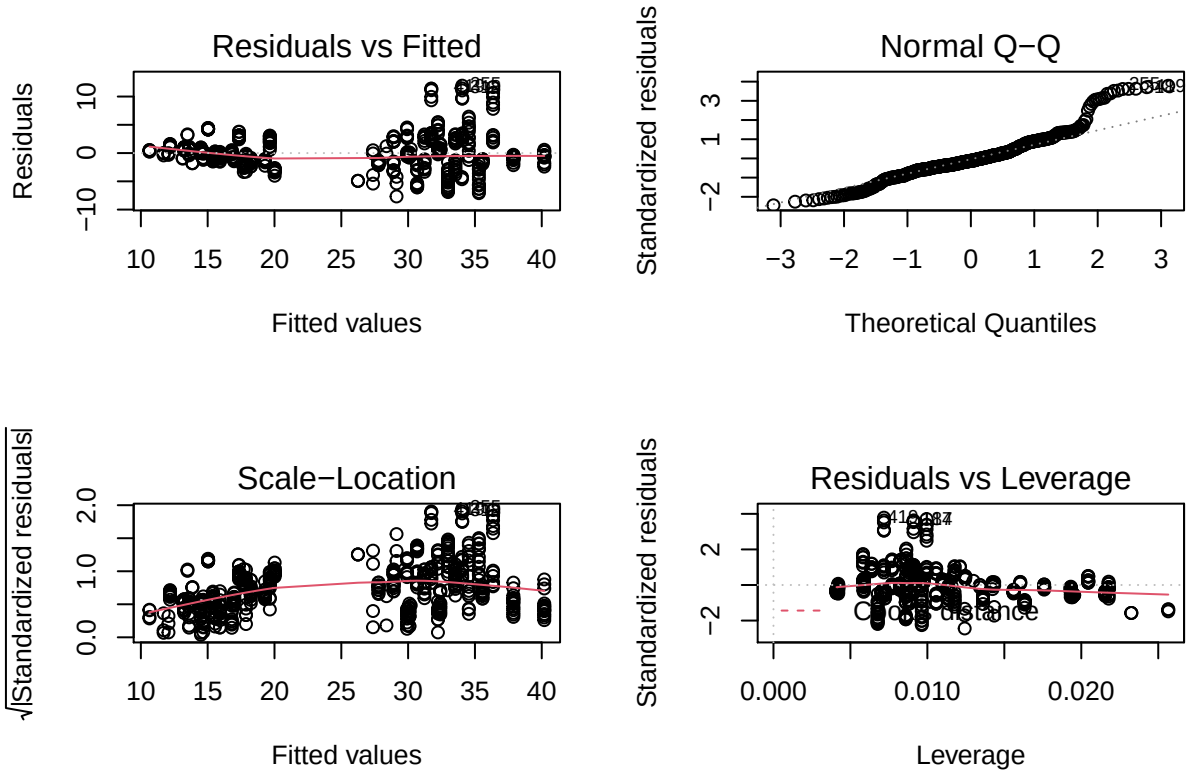
```r
# Checking the residuals of lm.fit1
hist(lm.y2_1$residuals, xlab = 'Residuals', main = '')
```




```
# Checking the plot of the residuals to ensure there aren't any cone shaped formations
plot(lm.y2_1$residuals, ylab = 'residuals')
abline(a = 0, b = 0)
```




```
par(mfrow = c(2, 2))
plot(lm.y2_1)
```




```
lm.predy2 <- predict(lm.y2_1, test_datay2)
pred2.R2 <- R2(lm.predy2,test_datay2$Y2)
pred2.RMSE <- RMSE(lm.predy2,test_datay2$Y2)
pred2.error <- pred2.RMSE/mean(test_datay2$Y2)
paste("R:", pred2.R2)
```

```
## [1] "R: 0.887406056190173"
```

```
paste("RMSE:", pred2.RMSE)
```

```
## [1] "RMSE: 3.3429013740876"
```

```
paste("Error:", pred2.error)
```

```
## [1] "Error: 0.138519019710083"
```

The R^2 at ~.89 shows us the observations are highly correlated to the predicted values. The RMSE is ~3.34 and there is an error rate of ~13.85%. The significant variables in our model are X1, X2, X3, X5, and X7.

```
# generating an RSE for reference
sigma(lm.y2_1)/mean(datay2$Y2)
```

```
## [1] 0.128594
```

Conclusion

As we believed initially, on their own each variable has a significant relationship with each response. However the models aren't very reliable on their own, also like we suspected. Going through multiple linear regression modeling we were able to identify which variables rejected/failed to reject the null hypothesis. They are

stated in the above document, but to reinforce our conclusion we'll also include them here. The results for Heating and Cooling Loads are the same. They are:

- For both Heating and Cooling Load, in a multiple linear regression model, we fail to reject H0 for X6(Orientation) and X8(Glazing Area Distribution). Further, we fail to reject H0 for X4(Roof Area) due to lack of evidence to reject the null hypothesis.
- For both Heating and Cooling Load, in a multiple linear regression model, there is sufficient evidence to reject H0 for X1(Relative Compactness), X2(Surface Area), X3(Wall Area), X5(Overall Height), and X7(Glazing Area).

Moving on, each predictor variable had a similar relationship with both individual response, that can be seen in our correlation plots, their correlation with one another, and a plot that shows how closely they fit. There are no opposite effects for variables, our rationale about modern architecture/thermodynamics seems to hold.

The top positive and negative attributes that impact energy efficiency are:

- Heating Load:
  - Positively: Overall Height at .89
  - Negatively: Surface Area at -.66
- Cooling Load:
  - Positively: Overall Height at .90
  - Negatively: Surface Area at -.67

Note, there is a very slight variation on how these predictors relate to the responses.

Contributions

Anthony Chelf researched for the dataset to work with, examining whether the data is appropriate for this project and if it is of our interest to study. Most work were done mutually: we performed EDA, tested various models and study their fit, and also generate relevant plots. After finalizing analyses and drawing conclusions, Hansol Lee formatted the overall code to ensure readability and organization of work.

Bibliography

A. Tsanas, A. Xifara: 'Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools', Energy and Buildings, Vol. 49, pp. 560-567, 2012