Home Sales Statistics and Regression

Mustafa Telab

5/16/2021

Contents

Intro

This markdown will utilize a dataset made available by kaggle.com. The context is a competition to predict the home sale prices of the test set using regression techniques. We will ultimately be making such submission after we have navigated through the data set, selected our independent variables, and built our model.

```
#Import Libraries
library(tidyverse)
library(ggcorrplot)
library(pastecs)
library(modelr)
library(MASS, exclude = 'select')
```

```r
#Import Training Set
train <- read_csv("house-prices-advanced-regression-techniques/train.csv")
```

Explore

Lets begin by retrieving some plots to explore our dependent variable.

```r
hist(train$SalePrice)
```

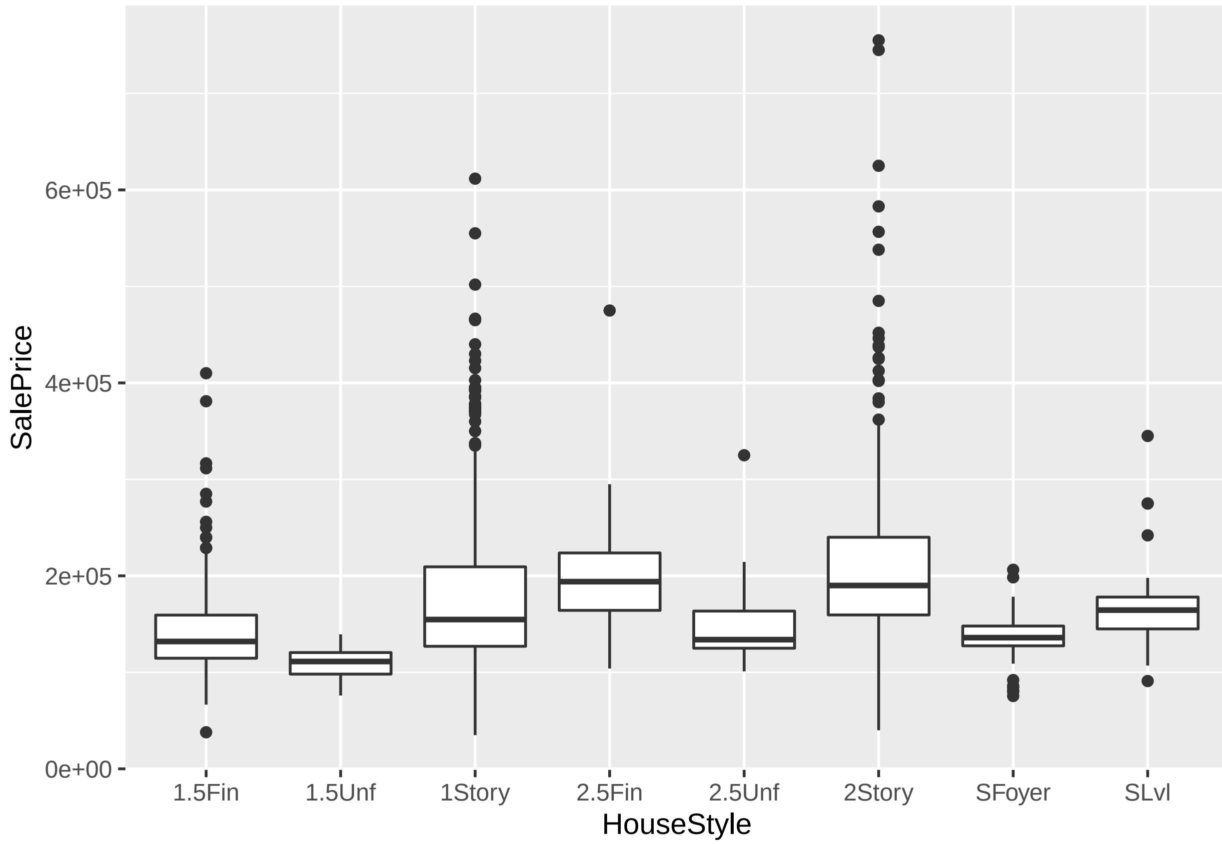

```r
summary(train$SalePrice)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   34900  129975  163000  180921  214000  755000
```

Visualize

We now move onto the possible explanatory variables. The first selections involve the structure and type of the home.

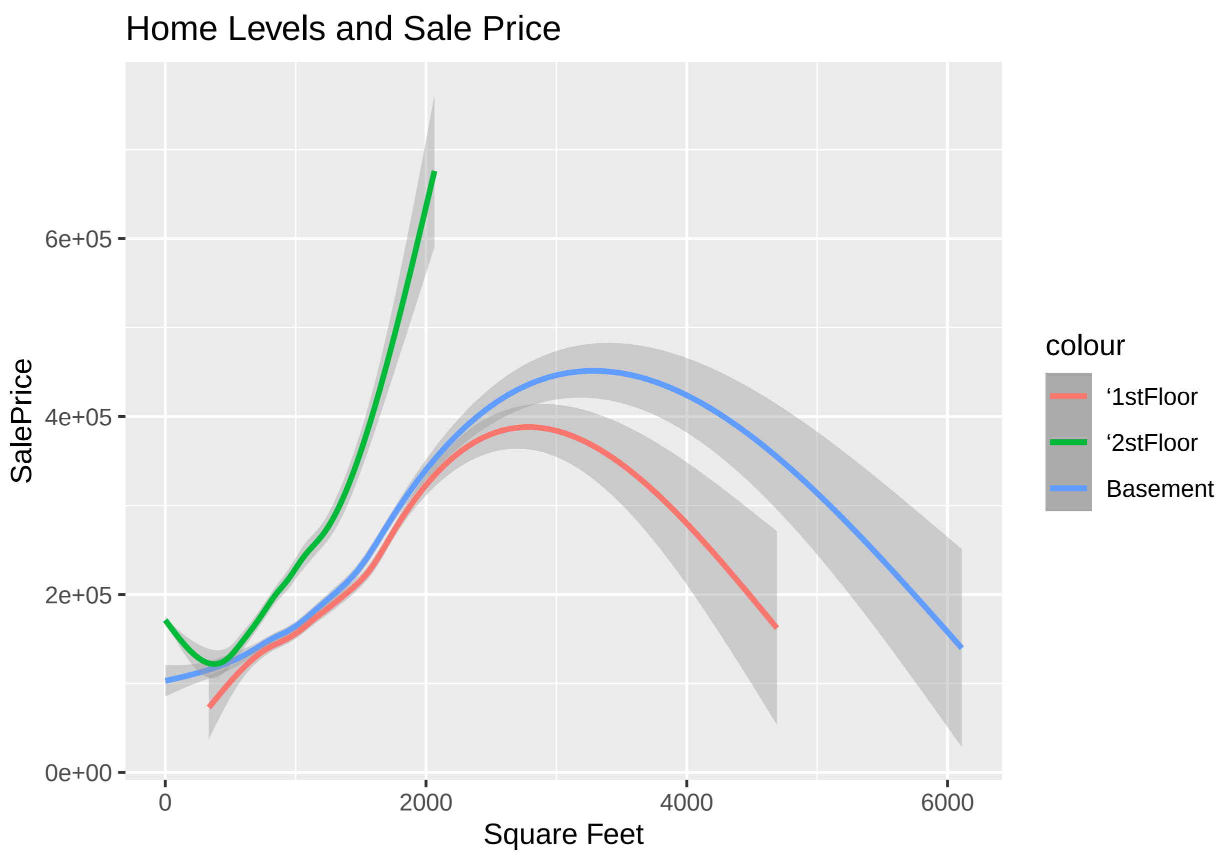
```r
train%>%
ggplot(aes(x = HouseStyle, y = SalePrice)) + geom_boxplot()
```



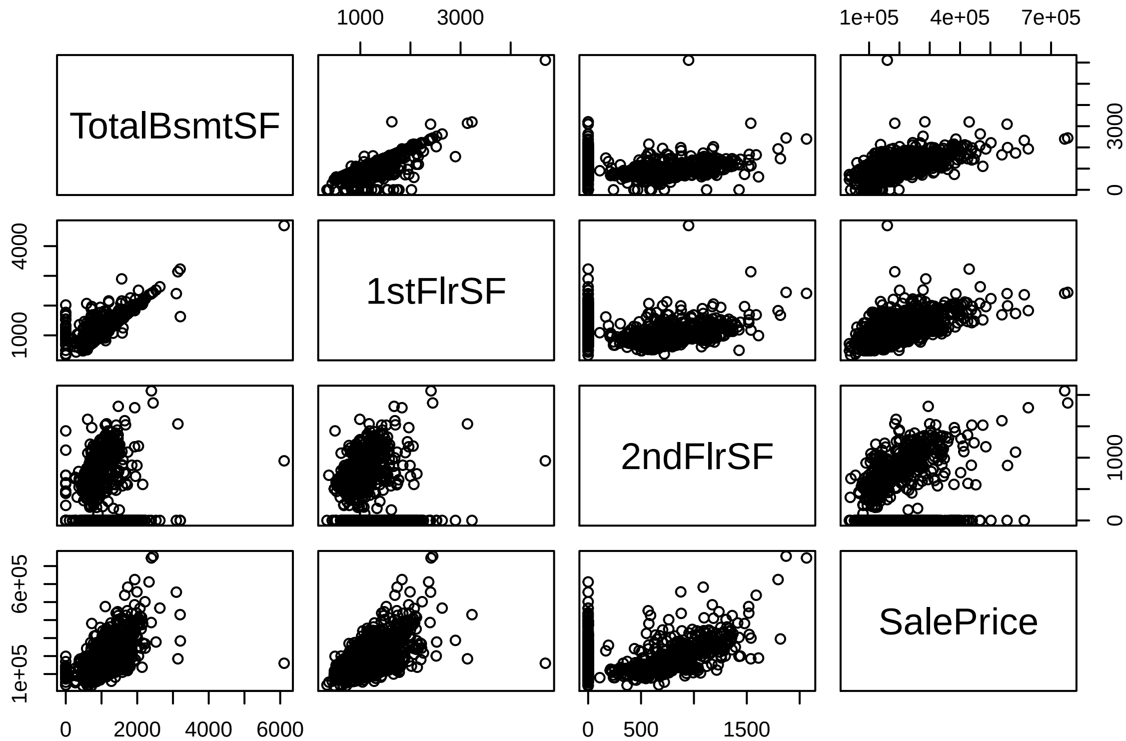
Following still with home configuration, we find an interesting reaction with the home level square footage. These plots with prove important further down for our regression analysis.

```
sf_plot = train%>%
ggplot() +
  geom_smooth(aes(x = TotalBsmtSF, y = SalePrice, col = "Basement")) +
  geom_smooth(aes(x = `1stFlrSF`, y = SalePrice, col = "`1stFloor"))+
  geom_smooth(aes(x = `2ndFlrSF`, y = SalePrice, col = "`2stFloor"))+
  labs(title = "Home Levels and Sale Price", x = "Square Feet")

sf_plot
```



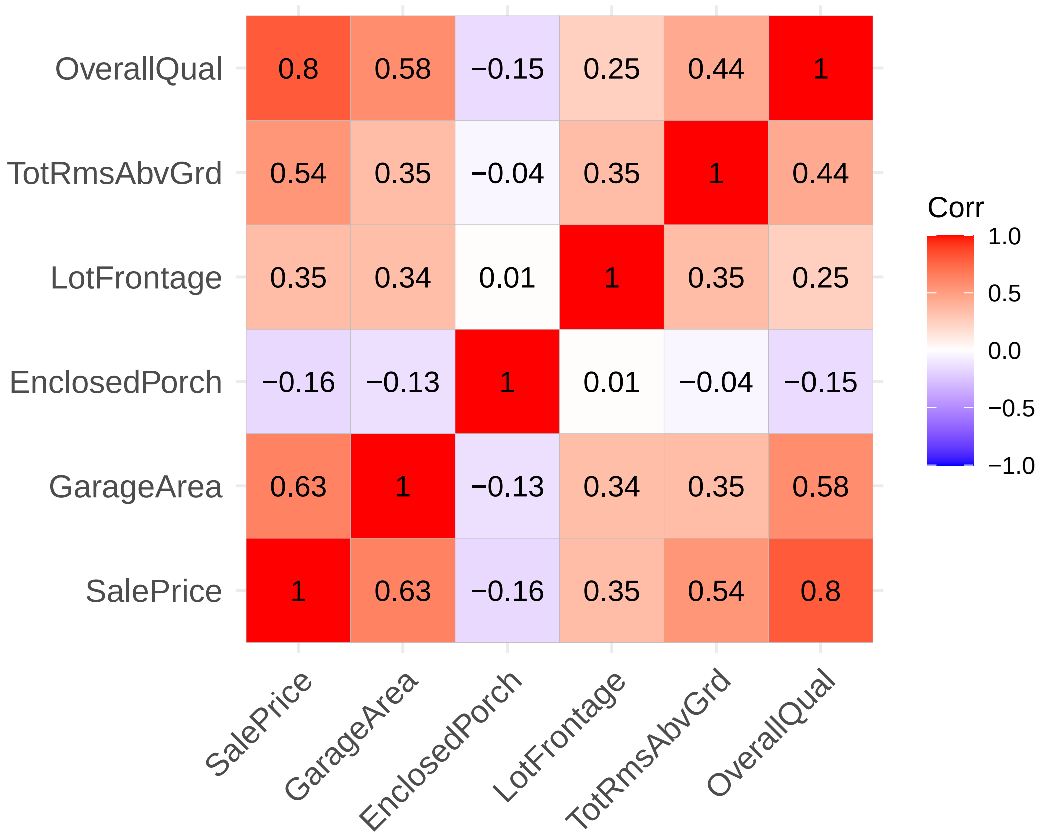

```r
vars = c('TotalBsmtSF','1stFlrSF','2ndFlrSF')
train%>%
select(vars , SalePrice)%>%
pairs(c(vars, 'SalePrice'))
```



Correlation

Next we are taking another set of quantitative variables to build a correlation matrix next with the sale price. The below matrix was landed on after cycling through a series of random samples of pairs.

```
set.seed(997)
var_pairs = select_if(train,is.numeric)%>%
  select(SalePrice, sample(1:length(colnames(select_if(train, is.numeric))), 5, replace=F))%>%
  data.frame()
var_pairs%>%
  cor(use = 'na.or.complete') %>%
  round(2)%>%
  ggcorrplot( lab = TRUE)
```



Correlation Test

Proceeding with a loop through all of the pairs for the above matrix, we collect testing data on the null hypothesis that the correlation is zero.

```
rowcounter = 1
pair_cortest = data.frame(pair = character(),
                          cor = numeric(),
                          confidence_80 =numeric(),
                          p_value =numeric(),
                          t_Statistic =numeric(),
                          observations=numeric() )
for (i in seq(1,length(colnames(var_pairs)))) {
  for (v in seq(1,length(colnames(var_pairs))-i)) {
    if (i + v <= 6){
         cort = cor.test(var_pairs[,i],var_pairs[,i+v])
    pair_cortest[rowcounter,1] = paste0(colnames(var_pairs)[i],"~",colnames(var_pairs)[i+v])
    pair_cortest[rowcounter,2] = round(cort[["estimate"]][["cor"]],2)
    pair_cortest[rowcounter,3] = paste0(round(cort[["conf.int"]][1],2)," - ",round(cort[["conf.int"]][2
    pair_cortest[rowcounter,4] = round(cort[["p.value"]],5)
    pair_cortest[rowcounter,5] = round(cort[["statistic"]][["t"]],2)
    pair_cortest[rowcounter,6] = cort[["parameter"]][["df"]]
    rowcounter = rowcounter +1
    }
```

```
  }
}
```

From the list of correlations below, we can reject the null hypothesis that the correlation between the pairs are zero, for all but the top two pairs in the list. The top two do not have much of a linear relationship, and the P-value supports this observation.

```
pair_cortest[order(-pair_cortest$p_value),]
```

```
##                          pair   cor confidence_80 p_value   t_Statistic
## 11 EnclosedPorch~TotRmsAbvGrd  0.00   -0.05 - 0.06 0.87407          0.16
## 10  EnclosedPorch~LotFrontage  0.01   -0.05 - 0.07 0.71105          0.37
## 12  EnclosedPorch~OverallQual -0.11 -0.16 - -0.06 0.00001         -4.38
## 1         SalePrice~GarageArea  0.62    0.59 - 0.65 0.00000         30.45
## 2      SalePrice~EnclosedPorch -0.13 -0.18 - -0.08 0.00000         -4.95
## 3        SalePrice~LotFrontage  0.35      0.3 - 0.4 0.00000         13.01
## 4       SalePrice~TotRmsAbvGrd  0.53     0.5 - 0.57 0.00000         24.10
## 5        SalePrice~OverallQual  0.79    0.77 - 0.81 0.00000         49.36
## 6     GarageArea~EnclosedPorch -0.12 -0.17 - -0.07 0.00000         -4.68
## 7       GarageArea~LotFrontage  0.34    0.29 - 0.39 0.00000         12.73
## 8      GarageArea~TotRmsAbvGrd  0.34    0.29 - 0.38 0.00000         13.71
## 9       GarageArea~OverallQual  0.56     0.53 - 0.6 0.00000         25.95
## 13    LotFrontage~TotRmsAbvGrd  0.35      0.3 - 0.4 0.00000         13.03
## 14     LotFrontage~OverallQual  0.25      0.2 - 0.3 0.00000          9.00
## 15    TotRmsAbvGrd~OverallQual  0.43    0.38 - 0.47 0.00000         18.05
## 16     OverallQual~OverallQual  1.00          1 - 1 0.00000 2562469171.85
##    observations
## 11         1458
## 10         1199
## 12         1458
## 1          1458
## 2          1458
## 3          1199
## 4          1458
## 5          1458
## 6          1458
## 7          1199
## 8          1458
## 9          1458
## 13         1199
## 14         1199
## 15         1458
## 16         1458
```

Matrix Decomposition

```
cor_matrix = var_pairs%>%
  cor(use = 'na.or.complete')%>%
  matrix(nrow = length(colnames(var_pairs)), ncol = length(colnames(var_pairs)))
```

```r
cor_matrix_P = solve(cor_matrix)
cor_matrix%*%(cor_matrix_P%*%cor_matrix)
```

```
##            [,1]       [,2]        [,3]       [,4]        [,5]       [,6]
## [1,]  1.0000000  0.6317615 -0.16400433 0.35179910  0.53721530  0.8022875
## [2,]  0.6317615  1.0000000 -0.12836568 0.34499672  0.35049561  0.5836331
## [3,] -0.1640043 -0.1283657  1.00000000 0.01070034 -0.03506514 -0.1516277
## [4,]  0.3517991  0.3449967  0.01070034 1.00000000  0.35209595  0.2516458
## [5,]  0.5372153  0.3504956 -0.03506514 0.35209595  1.00000000  0.4447412
## [6,]  0.8022875  0.5836331 -0.15162766 0.25164578  0.44474116  1.0000000
```

Matrix LU Decomposition of the correlation matrix

```r
cor_matrix_L = diag(length(colnames(var_pairs)))
cor_matrix_U = cor_matrix
for (j in seq(1,length(colnames(var_pairs)),1)){
  for (i in seq(1,length(colnames(var_pairs)),1)){
        if (i > j){
          term = cor_matrix_U[i,j]/cor_matrix_U[j,j]
          cor_matrix_U[i,]= cor_matrix_U[i,] - (term* cor_matrix_U[j,])
          cor_matrix_L[i,j]= term
    }
  }
}
print(cor_matrix_L)
```

```
##            [,1]        [,2]        [,3]        [,4]      [,5] [,6]
## [1,]  1.0000000  0.00000000  0.00000000  0.00000000 0.0000000    0
## [2,]  0.6317615  1.00000000  0.00000000  0.00000000 0.0000000    0
## [3,] -0.1640043 -0.04119653  1.00000000  0.00000000 0.0000000    0
## [4,]  0.3517991  0.20427395  0.07556303  1.00000000 0.0000000    0
## [5,]  0.5372153  0.01847911  0.05503433  0.18541969 1.0000000    0
## [6,]  0.8022875  0.12777780 -0.01737098 -0.05322309 0.0317967    1
```

```r
print(cor_matrix_U)
```

```
##      [,1]          [,2]        [,3]       [,4]       [,5]        [,6]
## [1,]    1  6.317615e-01 -0.16400433 0.35179910 0.53721530  0.80228746
## [2,]    0  6.008774e-01 -0.02475406 0.12274360 0.01110368  0.07677880
## [3,]    0  0.000000e+00  0.97208280 0.07345352 0.05349793 -0.01688603
## [4,]    0  0.000000e+00  0.00000000 0.84561370 0.15679343 -0.04500618
## [5,]    0  0.000000e+00  0.00000000 0.00000000 0.67917773  0.02159561
## [6,]    0 -1.387779e-17  0.00000000 0.00000000 0.00000000  0.34314885
```

We can check that A = LU

```r
print(cor_matrix)
```

```
##            [,1]       [,2]        [,3]       [,4]        [,5]       [,6]
## [1,]  1.0000000  0.6317615 -0.16400433 0.35179910  0.53721530  0.8022875
```

```
## [2,]  0.6317615  1.0000000 -0.12836568 0.34499672  0.35049561  0.5836331
## [3,] -0.1640043 -0.1283657  1.00000000 0.01070034 -0.03506514 -0.1516277
## [4,]  0.3517991  0.3449967  0.01070034 1.00000000  0.35209595  0.2516458
## [5,]  0.5372153  0.3504956 -0.03506514 0.35209595  1.00000000  0.4447412
## [6,]  0.8022875  0.5836331 -0.15162766 0.25164578  0.44474116  1.0000000
```

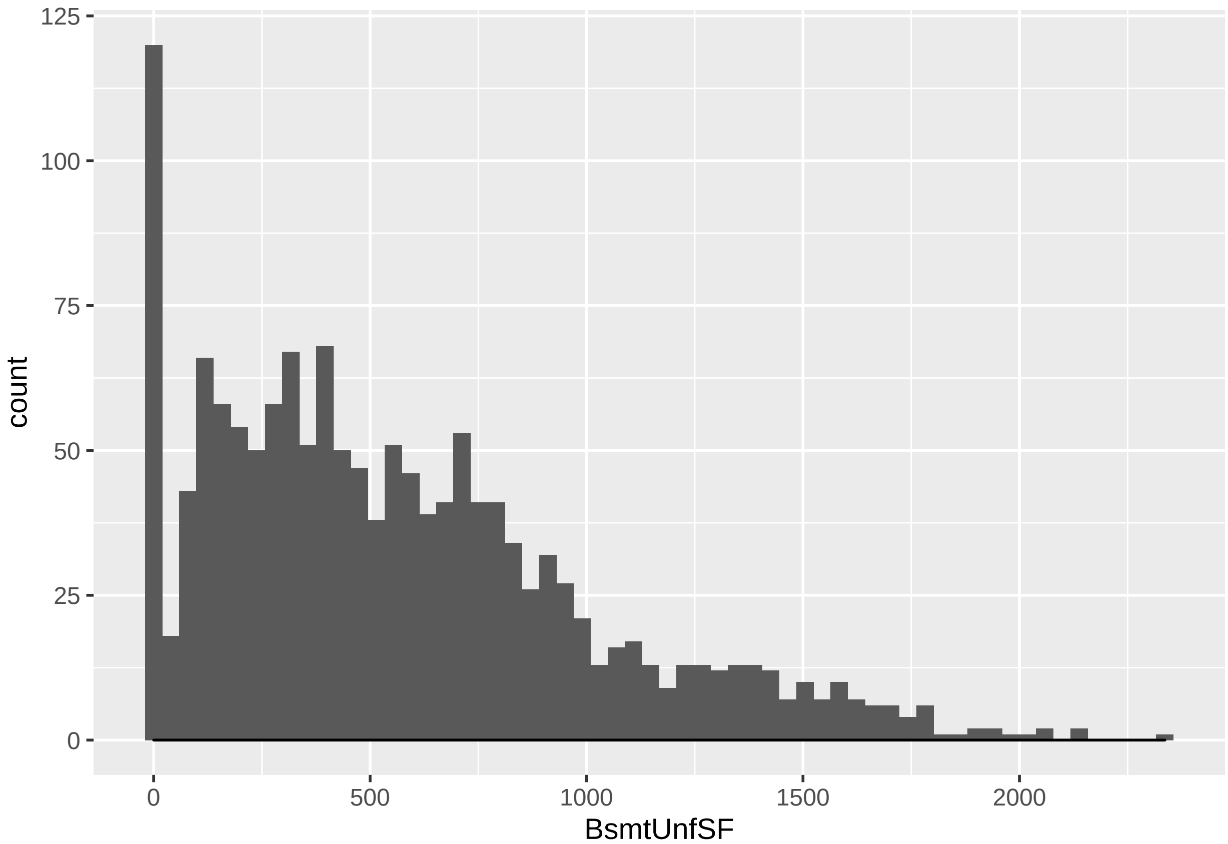
```
print(cor_matrix_L%*%cor_matrix_U)
```

```
##            [,1]       [,2]        [,3]       [,4]        [,5]       [,6]
## [1,]  1.0000000  0.6317615 -0.16400433 0.35179910  0.53721530  0.8022875
## [2,]  0.6317615  1.0000000 -0.12836568 0.34499672  0.35049561  0.5836331
## [3,] -0.1640043 -0.1283657  1.00000000 0.01070034 -0.03506514 -0.1516277
## [4,]  0.3517991  0.3449967  0.01070034 1.00000000  0.35209595  0.2516458
## [5,]  0.5372153  0.3504956 -0.03506514 0.35209595  1.00000000  0.4447412
## [6,]  0.8022875  0.5836331 -0.15162766 0.25164578  0.44474116  1.0000000
```

Variable Simulation

Looking for a right skewed variable, we stumble upon BsmtUnfSF. The histogram, along with the difference between the median and mean, make this a good example for this exercise.

```
train%>%
ggplot(aes(x = BsmtUnfSF))+
  geom_histogram( bins = 60)+
  geom_density()
```



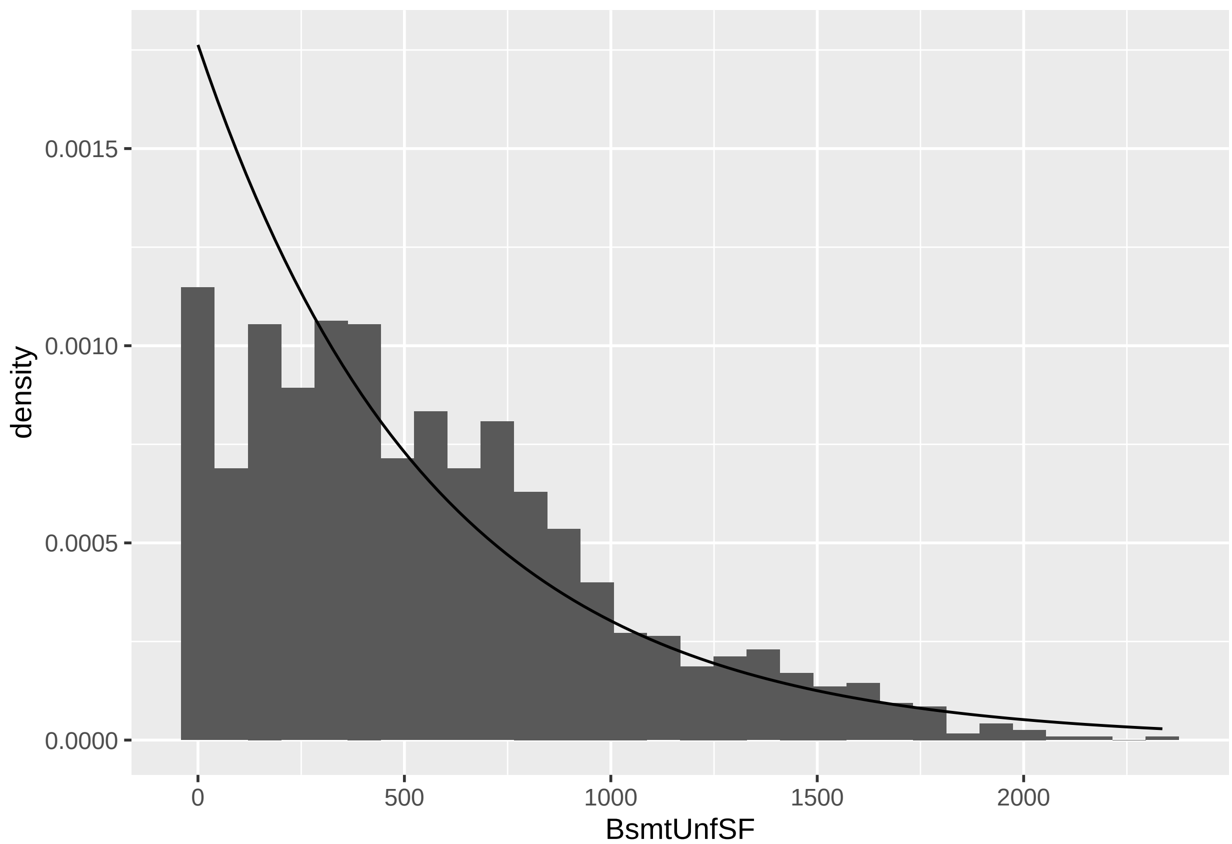

```
print(summary(train$BsmtUnfSF))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   223.0   477.5   567.2   808.0  2336.0
```

```
BsmtUnfSFexp =
fitdistr(train$BsmtUnfSF+.001, "exponential")
```

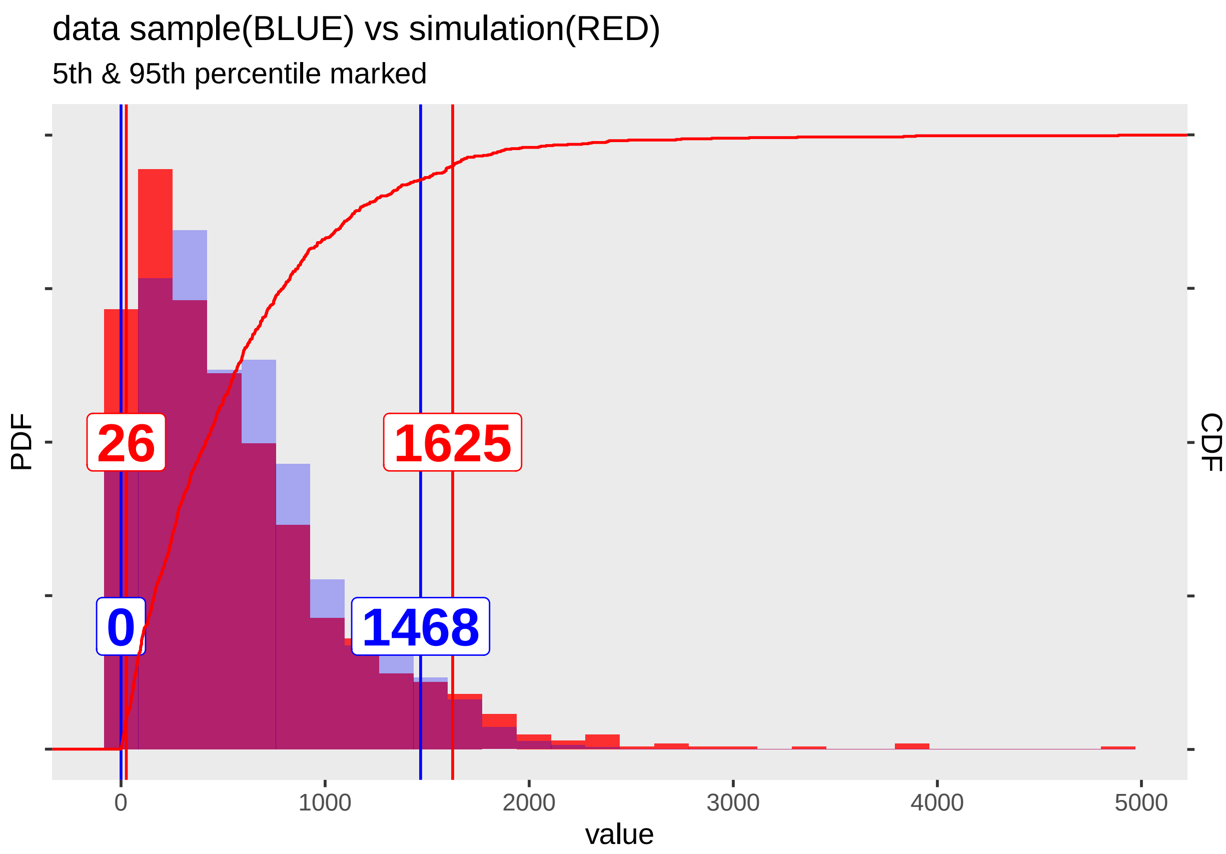
Our first look at the exponential distribution line on top of the actual histogram.

```
basegg = ggplot(train, aes(x = BsmtUnfSF)) + geom_histogram(aes(y=..density..), bins = 30)
basegg + stat_function(aes(x = train$BsmtUnfSF),fun = dexp, args = list(rate = BsmtUnfSFexp$estimate["ra
```



Below we simulate a sample of 1000, plucked from the exponential distribution that we fitted to the data above. The plot below features an overlay of two histograms. The blue represents the data that we pulled from the data set, while the red represents the simulated sample. The middle 90%(5th and 95th intervals) is marked by vertical lines. We first notice that the simulation has a wider spread. Also worth noting that the raw data's 5th percentile is 0, which makes sense considering there are a set of observations at value zero. However the simulation does not cluster at zero like the empirical data does.

```r
set.seed(92929)
c_int = t.test(train$BsmtUnfSF)
expo_sample = rexp(1000, BsmtUnfSFexp$estimate["rate"])
ggplot()+
  geom_histogram(aes(x = expo_sample, y=..density..*800),  fill = "red", alpha =.8) +
  geom_histogram(aes(x = train$BsmtUnfSF, y=..density..*800),  fill = "blue", alpha = .3) +
  geom_vline(xintercept = quantile(expo_sample,.05), color = "red", show.legend = TRUE)+
  geom_vline(xintercept = quantile(expo_sample,.95), color = "red", show.legend = TRUE)+
  geom_vline(xintercept = quantile(train$BsmtUnfSF,.05), color = "blue", show.legend = TRUE)+
  geom_vline(xintercept = quantile(train$BsmtUnfSF,.95), color = "blue", show.legend = TRUE)+
  geom_label(aes( x=quantile(expo_sample,.05), y=.50, label=round(quantile(expo_sample,.05),)),color="r
  geom_label(aes( x=quantile(expo_sample,.95), y=.50, label=round(quantile(expo_sample,.95),)),color="r
  geom_label(aes( x=quantile(train$BsmtUnfSF,.05), y=.200, label=round(quantile(train$BsmtUnfSF,.05),))
  geom_label(aes( x=quantile(train$BsmtUnfSF,.95), y=.200, label=round(quantile(train$BsmtUnfSF,.95),))
  stat_ecdf(aes(x=expo_sample), geom = "step", color = "red")+
  scale_y_continuous("PDF",sec.axis = sec_axis(~ (. - 0), name = "CDF"))+
  labs(title = "data sample(BLUE) vs simulation(RED)", subtitle = "5th & 95th percentile marked", x = "
  theme(legend.position = "none",panel.grid = element_blank(), axis.text.y = element_blank())
```




```
print(
  paste0("The 95% confidence interval for the mean of the empirical data is ", round(c_int[["conf.int"]]
)
```

```
## [1] "The 95% confidence interval for the mean of the empirical data is 545 to 590"
```

Modeling

Backwards Elimination

To begin our model we first select our variables. These will include the variables we explored earlier and more. The independent variables chosen here are mostly quantitative, with the exception of 'KitchenQual,' which is included based on the popular adage that "kitchens sell homes."

We will move along using the Backward Elimination method, to trim off what appear to be poor "performing" variables.

```
mylm = lm(SalePrice ~ GrLivArea + LotArea + YearBuilt + WoodDeckSF + PoolArea + KitchenQual + TotalBsmtS
summary(mylm)
```

```
##
## Call:
## lm(formula = SalePrice ~ GrLivArea + LotArea + YearBuilt + WoodDeckSF +
```

```
##     PoolArea + KitchenQual + TotalBsmtSF + BsmtUnfSF + BsmtFinSF2 +
##     MasVnrArea + OverallCond + OverallQual + GarageArea + EnclosedPorch +
##     LotFrontage + TotRmsAbvGrd + `1stFlrSF` + `2ndFlrSF` + TotRmsAbvGrd +
##     TotalBsmtSF, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -509162  -15267   -1721   12392  307145
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -8.233e+05  1.168e+05  -7.051 3.03e-12 ***
## GrLivArea      2.053e+01  2.144e+01   0.957  0.33869
## LotArea        7.989e-01  1.535e-01   5.204 2.30e-07 ***
## YearBuilt      4.048e+02  5.834e+01   6.939 6.53e-12 ***
## WoodDeckSF     2.884e+01  9.476e+00   3.044  0.00239 **
## PoolArea      -8.140e+01  2.869e+01  -2.838  0.00462 **
## KitchenQualFa -4.456e+04  8.608e+03  -5.176 2.66e-07 ***
## KitchenQualGd -4.658e+04  4.562e+03 -10.210  < 2e-16 ***
## KitchenQualTA -5.768e+04  5.250e+03 -10.988  < 2e-16 ***
## TotalBsmtSF    2.152e+01  4.917e+00   4.377 1.31e-05 ***
## BsmtUnfSF     -1.498e+01  2.866e+00  -5.226 2.05e-07 ***
## BsmtFinSF2    -7.841e+00  7.274e+00  -1.078  0.28128
## MasVnrArea     2.847e+01  6.795e+00   4.189 3.01e-05 ***
## OverallCond    5.641e+03  1.136e+03   4.966 7.83e-07 ***
## OverallQual    1.690e+04  1.354e+03  12.482  < 2e-16 ***
## GarageArea     3.443e+01  6.570e+00   5.241 1.89e-07 ***
## EnclosedPorch  8.198e+00  1.932e+01   0.424  0.67133
## LotFrontage   -2.037e+01  5.386e+01  -0.378  0.70537
## TotRmsAbvGrd   1.550e+03  1.220e+03   1.271  0.20408
## `1stFlrSF`     2.573e+01  2.198e+01   1.170  0.24205
## `2ndFlrSF`     2.061e+01  2.157e+01   0.956  0.33952
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36400 on 1174 degrees of freedom
##   (265 observations deleted due to missingness)
## Multiple R-squared:  0.8117, Adjusted R-squared:  0.8085
## F-statistic:   253 on 20 and 1174 DF,  p-value: < 2.2e-16
```

Some variables under the gun are `1stFlrSF` ,`2ndFlrSF`, and TotalBsmtSF. These variables refer to the square footage of the different home levels. GrLivArea, which is a combination of the former two, is also on the chopping block. From our earlier plots on these variables we know the 1st and 2nd floor footgage follow a similar pattern; so we can move forward with the assumption that the aggregated variable can be used in leu of the separated top floors. Leaving basement alone for now.

```
print(stat.desc(train[c('GrLivArea','1stFlrSF','2ndFlrSF')]))
```

```
##                 GrLivArea     1stFlrSF     2ndFlrSF
## nbr.val      1.460000e+03 1.460000e+03 1.460000e+03
## nbr.null     0.000000e+00 0.000000e+00 8.290000e+02
## nbr.na       0.000000e+00 0.000000e+00 0.000000e+00
## min          3.340000e+02 3.340000e+02 0.000000e+00
```

```
## max          5.642000e+03 4.692000e+03 2.065000e+03
## range        5.308000e+03 4.358000e+03 2.065000e+03
## sum          2.212577e+06 1.697435e+06 5.066090e+05
## median       1.464000e+03 1.087000e+03 0.000000e+00
## mean         1.515464e+03 1.162627e+03 3.469925e+02
## SE.mean      1.375245e+01 1.011746e+01 1.142447e+01
## CI.mean.0.95 2.697669e+01 1.984633e+01 2.241014e+01
## var          2.761296e+05 1.494501e+05 1.905571e+05
## std.dev      5.254804e+02 3.865877e+02 4.365284e+02
## coef.var     3.467456e-01 3.325123e-01 1.258034e+00
```

```r
mylm = update(mylm, .~. -`2ndFlrSF` -`1stFlrSF`, data = train)
summary(mylm)
```

```
##
## Call:
## lm(formula = SalePrice ~ GrLivArea + LotArea + YearBuilt + WoodDeckSF +
##     PoolArea + KitchenQual + TotalBsmtSF + BsmtUnfSF + BsmtFinSF2 +
##     MasVnrArea + OverallCond + OverallQual + GarageArea + EnclosedPorch +
##     LotFrontage + TotRmsAbvGrd, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -510158  -15234   -1539   12653  309237
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -8.289e+05  1.157e+05  -7.167 1.35e-12 ***
## GrLivArea      4.130e+01  4.489e+00   9.201  < 2e-16 ***
## LotArea        8.062e-01  1.534e-01   5.256 1.75e-07 ***
## YearBuilt      4.084e+02  5.783e+01   7.063 2.78e-12 ***
## WoodDeckSF     2.934e+01  9.469e+00   3.098  0.00199 **
## PoolArea      -8.406e+01  2.862e+01  -2.938  0.00337 **
## KitchenQualFa -4.492e+04  8.600e+03  -5.224 2.08e-07 ***
## KitchenQualGd -4.674e+04  4.548e+03 -10.279  < 2e-16 ***
## KitchenQualTA -5.781e+04  5.238e+03 -11.038  < 2e-16 ***
## TotalBsmtSF    2.474e+01  3.619e+00   6.838 1.29e-11 ***
## BsmtUnfSF     -1.521e+01  2.861e+00  -5.317 1.26e-07 ***
## BsmtFinSF2    -8.093e+00  7.269e+00  -1.113  0.26579
## MasVnrArea     2.883e+01  6.773e+00   4.256 2.25e-05 ***
## OverallCond    5.642e+03  1.134e+03   4.977 7.41e-07 ***
## OverallQual    1.682e+04  1.352e+03  12.446  < 2e-16 ***
## GarageArea     3.536e+01  6.534e+00   5.412 7.57e-08 ***
## EnclosedPorch  7.556e+00  1.927e+01   0.392  0.69501
## LotFrontage   -1.305e+01  5.323e+01  -0.245  0.80637
## TotRmsAbvGrd   1.575e+03  1.220e+03   1.291  0.19695
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36400 on 1176 degrees of freedom
##   (265 observations deleted due to missingness)
## Multiple R-squared:  0.8114, Adjusted R-squared:  0.8085
## F-statistic:   281 on 18 and 1176 DF,  p-value: < 2.2e-16
```

Continue again dropping the values with the highest p-values.(results combined below for readability)

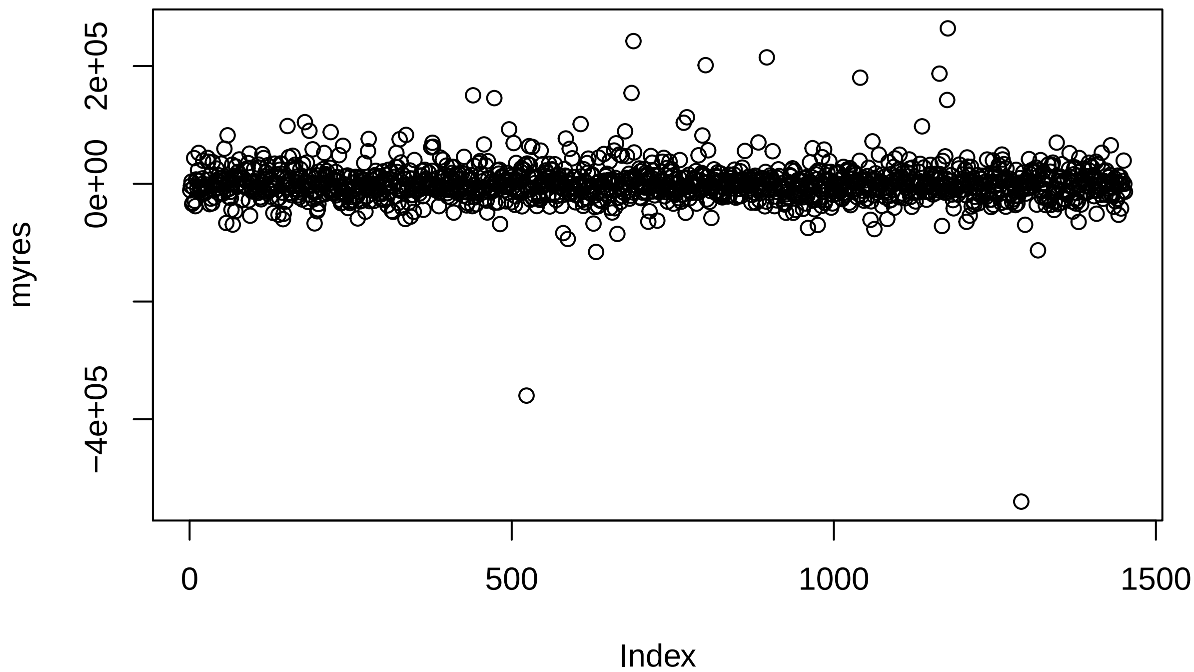
```r
mylm = update(mylm, .~. -LotFrontage - EnclosedPorch -PoolArea -BsmtFinSF2, data = train)
summary(mylm)
```

```
##
## Call:
## lm(formula = SalePrice ~ GrLivArea + LotArea + YearBuilt + WoodDeckSF +
##     KitchenQual + TotalBsmtSF + BsmtUnfSF + MasVnrArea + OverallCond +
##     OverallQual + GarageArea + TotRmsAbvGrd, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -539932  -14714   -1602   12212  264103
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -8.077e+05  9.266e+04  -8.717  < 2e-16 ***
## GrLivArea      4.230e+01  3.792e+00  11.156  < 2e-16 ***
## LotArea        5.854e-01  9.812e-02   5.965 3.07e-09 ***
## YearBuilt      3.991e+02  4.656e+01   8.572  < 2e-16 ***
## WoodDeckSF     2.763e+01  7.778e+00   3.552 0.000395 ***
## KitchenQualFa -4.487e+04  7.656e+03  -5.861 5.71e-09 ***
## KitchenQualGd -4.617e+04  4.104e+03 -11.249  < 2e-16 ***
## KitchenQualTA -5.672e+04  4.643e+03 -12.217  < 2e-16 ***
## TotalBsmtSF    2.487e+01  2.935e+00   8.473  < 2e-16 ***
## BsmtUnfSF     -1.311e+01  2.396e+00  -5.470 5.29e-08 ***
## MasVnrArea     2.712e+01  5.827e+00   4.655 3.54e-06 ***
## OverallCond    5.375e+03  9.361e+02   5.742 1.14e-08 ***
## OverallQual    1.636e+04  1.154e+03  14.172  < 2e-16 ***
## GarageArea     3.430e+01  5.665e+00   6.055 1.79e-09 ***
## TotRmsAbvGrd   1.473e+03  1.035e+03   1.422 0.155140
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34620 on 1437 degrees of freedom
##   (8 observations deleted due to missingness)
## Multiple R-squared:  0.8112, Adjusted R-squared:  0.8094
## F-statistic:   441 on 14 and 1437 DF,  p-value: < 2.2e-16
```

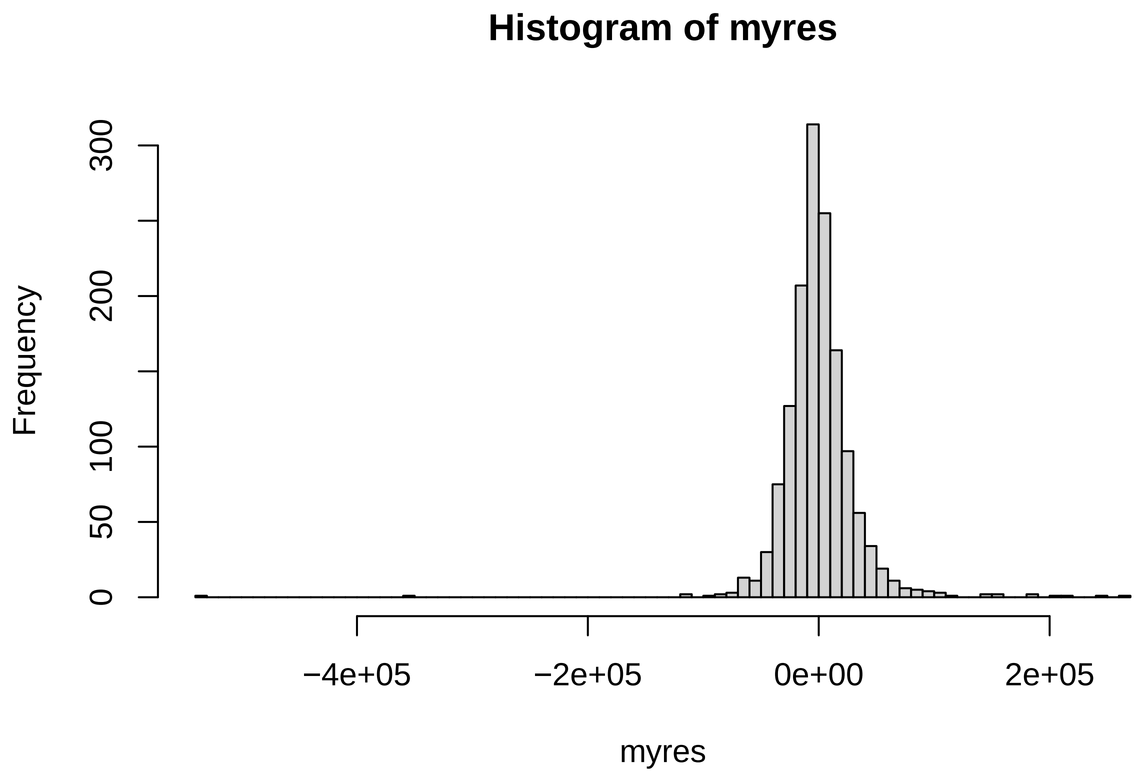
Residuals

Now that we have what appears to be a good collection of predictor variable, we can review the residuals. We already see from the summary above that the residuals are not centered around zero which is not ideal. However, we can see that aside from a few outliers. Thus, the residuals have a fairly normal distribution and can be considered a good sign for our model.

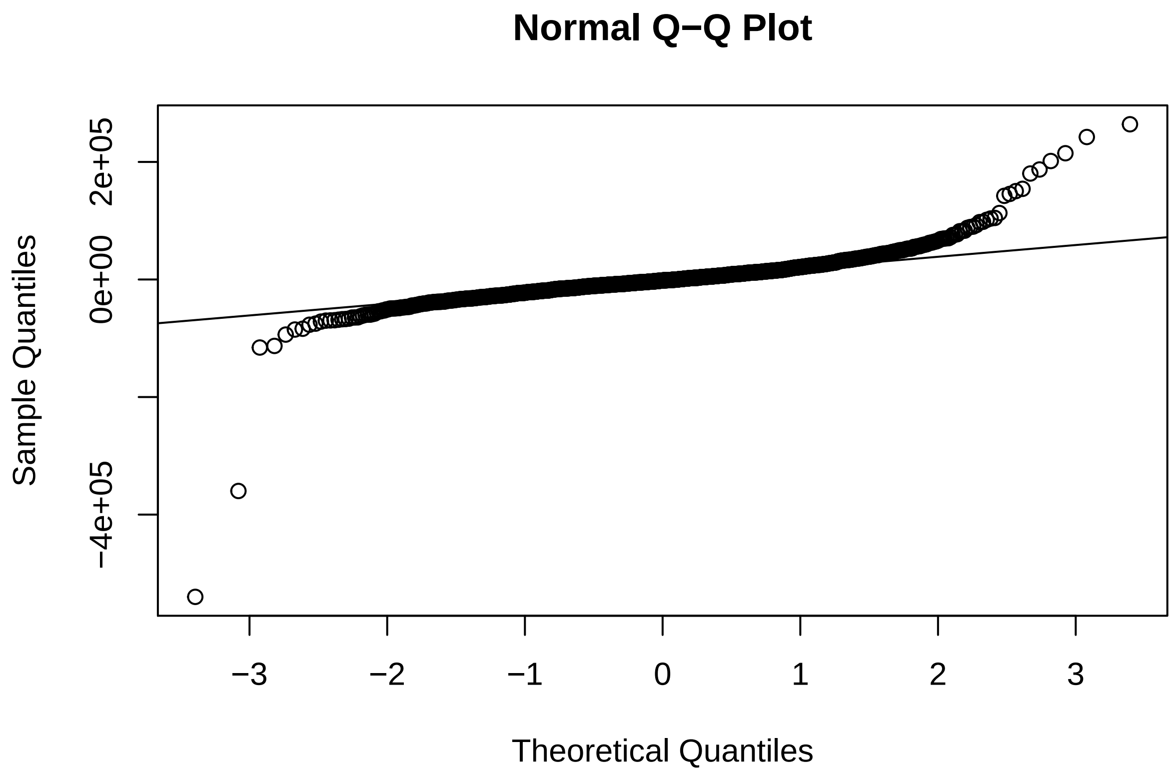
```r
myres = resid(mylm)
scplot = plot(myres, type = 'p')
```




```
hplot = hist(myres, breaks = 100)
```




```r
qqnorm(myres)
qqline(myres)
```



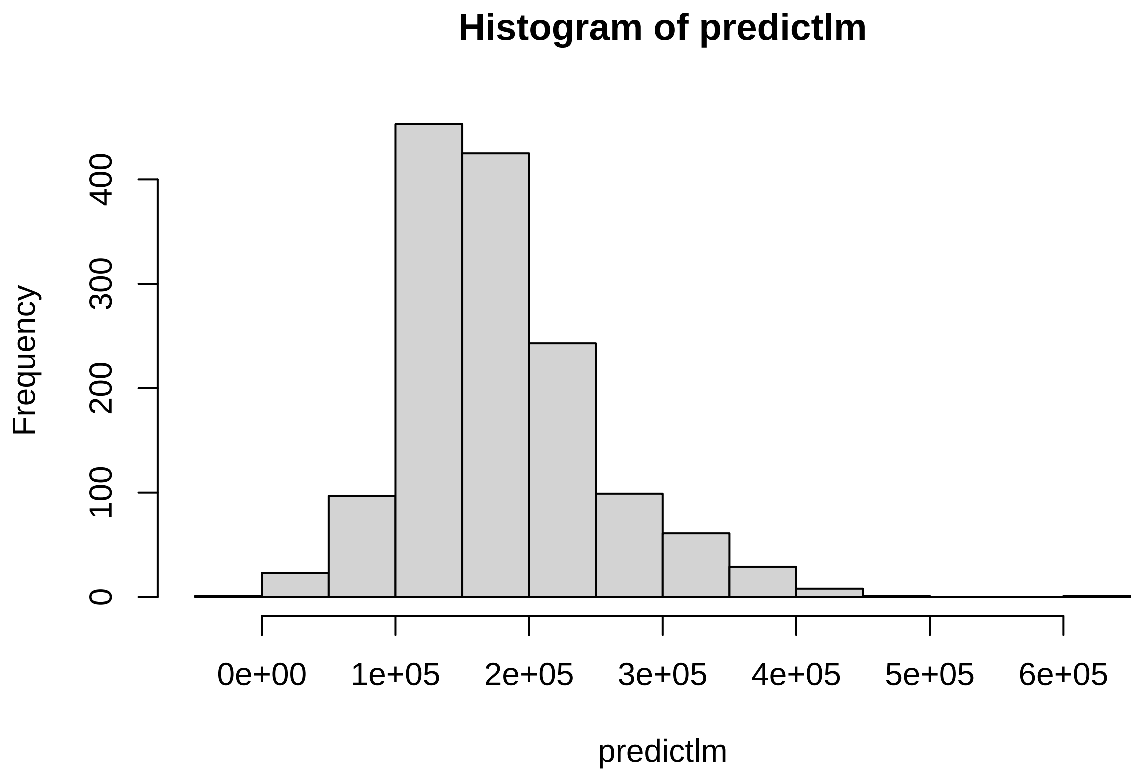
Prediction

Now that we are satisfied with the predictive variables and the residuals thus far; we can see how this model reacts to the test data.

```
test <- read_csv("house-prices-advanced-regression-techniques/test.csv")
predictlm = predict(mylm, newdata = test)
```

A look at the prediction results.

```
hist(predictlm)
```



Missing Values

We notice that we have less predictions than observations; which signals that some NAs must have been dropped. To avoid generating a prediction of NA, we want to first identify the predictive variables that have NA in the test data, and impute over with a reasonable value.

```
pred_vars = c('Id', colnames(mylm[["model"]])[-1])
print(summary(select(test,all_of(pred_vars))))
```

```
##        Id         GrLivArea       LotArea        YearBuilt      WoodDeckSF
##  Min.   :1461   Min.   : 407   Min.   : 1470   Min.   :1879   Min.   :   0.00
##  1st Qu.:1826   1st Qu.:1118   1st Qu.: 7391   1st Qu.:1953   1st Qu.:   0.00
##  Median :2190   Median :1432   Median : 9399   Median :1973   Median :   0.00
##  Mean   :2190   Mean   :1486   Mean   : 9819   Mean   :1971   Mean   :  93.17
##  3rd Qu.:2554   3rd Qu.:1721   3rd Qu.:11518   3rd Qu.:2001   3rd Qu.: 168.00
##  Max.   :2919   Max.   :5095   Max.   :56600   Max.   :2010   Max.   :1424.00
##
##  KitchenQual          TotalBsmtSF     BsmtUnfSF        MasVnrArea
##  Length:1459         Min.   :   0   Min.   :   0.0   Min.   :   0.0
##  Class :character    1st Qu.: 784   1st Qu.: 219.2   1st Qu.:   0.0
##  Mode  :character    Median : 988   Median : 460.0   Median :   0.0
##                      Mean   :1046   Mean   : 554.3   Mean   : 100.7
##                      3rd Qu.:1305   3rd Qu.: 797.8   3rd Qu.: 164.0
##                      Max.   :5095   Max.   :2140.0   Max.   :1290.0
##                      NA's   :1      NA's   :1        NA's   :15
##   OverallCond      OverallQual       GarageArea      TotRmsAbvGrd
##  Min.   :1.000   Min.   : 1.000   Min.   :   0.0   Min.   : 3.000
```

```
##  1st Qu.:5.000   1st Qu.: 5.000   1st Qu.: 318.0   1st Qu.: 5.000
##  Median :5.000   Median : 6.000   Median : 480.0   Median : 6.000
##  Mean   :5.554   Mean   : 6.079   Mean   : 472.8   Mean   : 6.385
##  3rd Qu.:6.000   3rd Qu.: 7.000   3rd Qu.: 576.0   3rd Qu.: 7.000
##  Max.   :9.000   Max.   :10.000   Max.   :1488.0   Max.   :15.000
##                                   NA's   :1
```

After seeing that the NAs appear for variables that could be considered optional; it is reasonable to set these values to 0. (Most likely a recording error in the data)

```
test = data.frame(test[pred_vars]) %>%
  replace_na(list(TotalBsmtSF = 0, GarageArea = 0, MasVnrArea = 0,BsmtUnfSF = 0, KitchenQual = 'Gd'))
```

```
submission = add_predictions(test, mylm, var = 'SalePrice')%>%
  select('Id', 'SalePrice')%>%
  data_frame()
write_csv( submission, "SalePrice_Predictions.csv")
```

Summary and Kaggle Competition Results

lets recap how we landed on our results. We first explored a few explanatory variables, by following our intuition and testing with plots. We delved even deeper by putting a correlation matrix together and testing the null hypothesis that the was no correlation. A portion of the analysis was devoted to a single right skewed variable and attempting to mimic the distribution by sampling from an exponential function. Then we began our regression with a chunk of variables until it was trimmed down through backwards elimination. After reviewing the residuals and cleaning the test data, we finaly earned our score below.

User name: Mustafa Telab

Score: 0.33501