Final

David Moste

12/14/2020

Problem 1

I started by creating all four sets: X, Y, x, and y. I then used R to calculate all the listed probabilities.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

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

```
set.seed(12345)
N <- 10
mu <- (N+1)/2
sigma <- (N+1)/2
X <- runif(10000,1,N)
Y <- rnorm(10000,mean = mu,sd = sigma)
df <- data.frame(X,Y)

x <- median(X)
y <- summary(Y)[2]

df_Xy <- df %>% filter(X > y)
df_Yy <- df %>% filter(Y > y)

# Part a
df_a <- df_Xy %>% filter(X > x)
prob_a <- nrow(df_a)/nrow(df_Xy)
prob_a
```

```
## [1] 0.5512679
```

```r
# Part b
df_b <- df_Yy %>% filter(X > x)
prob_b <- nrow(df_b)/nrow(df)
prob_b
```

```
## [1] 0.3808
```

```r
# Part c
df_c <- df_Xy %>% filter(X < x)
prob_c <- nrow(df_c)/nrow(df_Xy)
prob_c
```

```
## [1] 0.4487321
```

$P_A = 0.55$
$P_B = 0.38$
$P_C = 0.45$

The next step was to create the marginal and joint probability table.

```r
# Probability table
XY <- df %>% filter(X > x, Y > y) %>% nrow()
Xy <- df %>% filter(X > x, Y < y) %>% nrow()
Xx_total <- XY/nrow(df) + Xy/nrow(df)

xY <- df %>% filter(X < x, Y > y) %>% nrow()
xy <- df %>% filter(X < x, Y < y) %>% nrow()
xX_total <- xY/nrow(df) + xy/nrow(df)

Yy_total <- XY/nrow(df) + xY/nrow(df)
yY_total <- Xy/nrow(df) + xy/nrow(df)
total <- Xx_total + xX_total

table <- data.frame(rbind(XY/nrow(df),Xy/nrow(df),XY/nrow(df) + Xy/nrow(df)),
                    rbind(xY/nrow(df),xy/nrow(df),xY/nrow(df) + xy/nrow(df)),
                    rbind(Yy_total,yY_total,total))
names(table) <- c("X > x","X < x","Total")
row.names(table) <- c("Y > y","Y < y","Total")
table
```

```
##         X > x  X < x Total
## Y > y 0.3808 0.3692  0.75
## Y < y 0.1192 0.1308  0.25
## Total 0.5000 0.5000  1.00
```

As can be seen, it is clear that the two probabilities are equal.
$P(X > x, Y > y) = P(X > x)\,P(Y > y)$

The nextr step is to check for independence. Fisher's Test works for small datasets while Chi Squared is more appropriate for larger datasets. Therefore I would choose Chi Squared for this problem.

```r
# Check independence
count_table <- matrix(c(XY,Xy,xY,xy),nrow = 2, ncol = 2, byrow = FALSE)
fisher.test(count_table)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  count_table
## p-value = 0.007904
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  1.032680 1.240419
## sample estimates:
## odds ratio
##   1.131777
```

```r
chisq.test(count_table)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  count_table
## X-squared = 7.0533, df = 1, p-value = 0.007912
```

Problem 2

Descriptive and Inferential Statistics

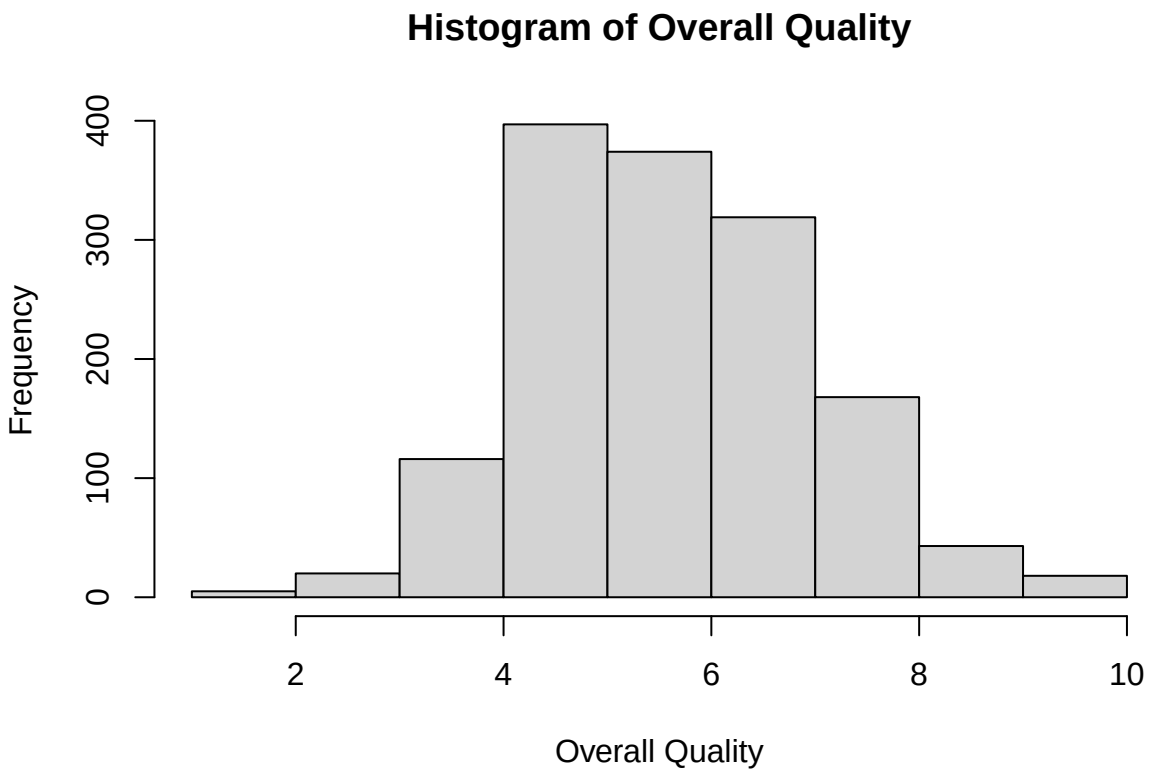
```r
train <- read.csv('https://raw.githubusercontent.com/dmoste/DATA605/master/train.csv')
test <- read.csv('https://raw.githubusercontent.com/dmoste/DATA605/master/test.csv')

train[is.na(train)] = 0

# Summary and histogram of Overall Quality
summary(train$OverallQual)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   5.000   6.000   6.099   7.000  10.000
```

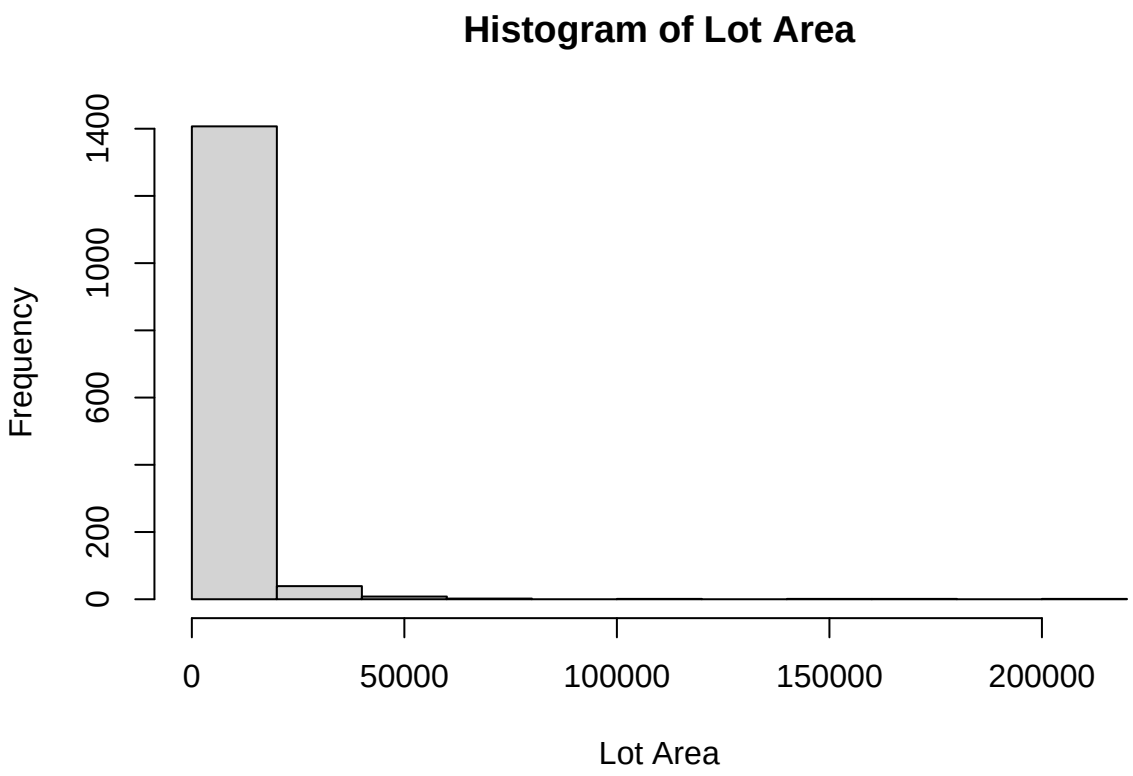
```r
hist(train$OverallQual,
     xlab = "Overall Quality",
     main = "Histogram of Overall Quality")
```




```
# Summary and histogram of Lot Area
summary(train$LotArea)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1300    7554    9478   10517   11602  215245
```

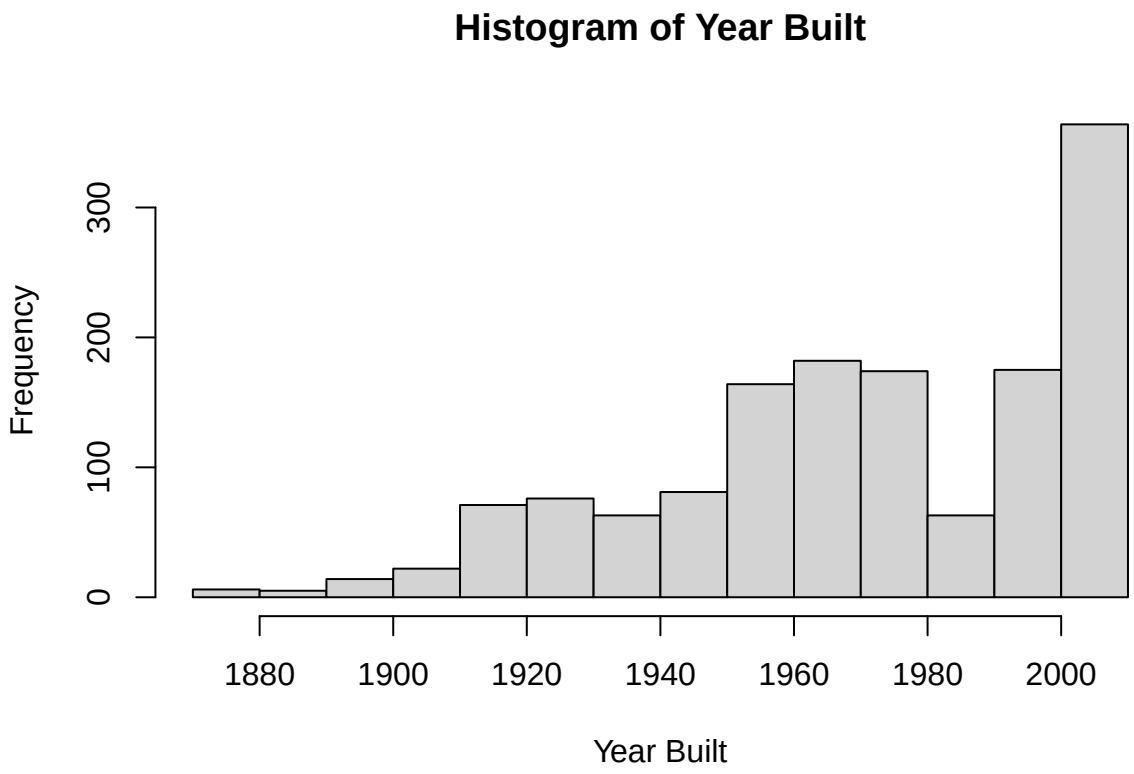
```
hist(train$LotArea,
     xlab = "Lot Area",
     main = "Histogram of Lot Area")
```



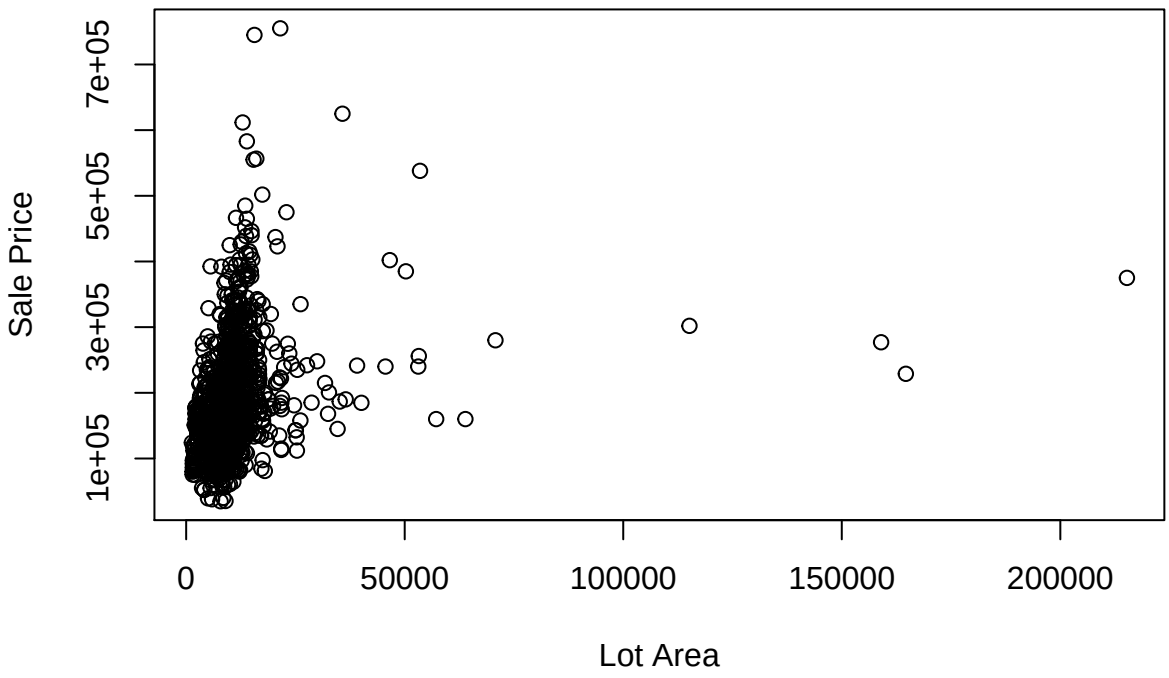

```
# Summary and histogram of Year Built
summary(train$YearBuilt)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1872    1954    1973    1971    2000    2010
```

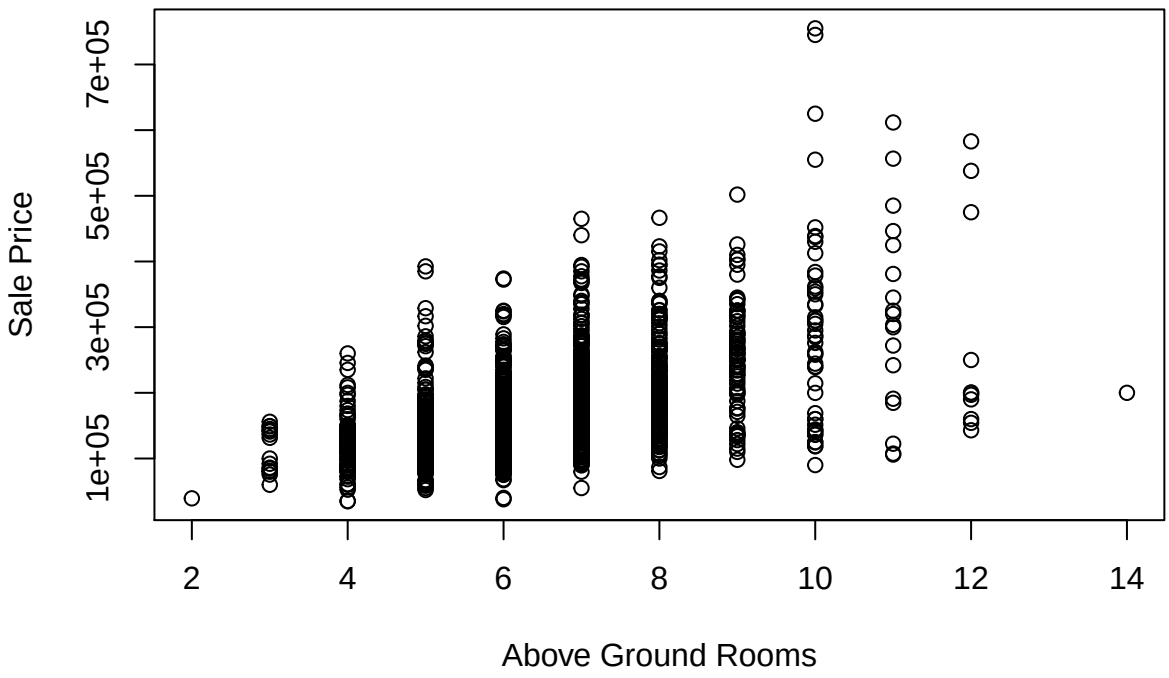
```
hist(train$YearBuilt,
     xlab = "Year Built",
     main = "Histogram of Year Built")
```




```r
# Plot Lot Area vs Sale Price
plot(train$LotArea,
     train$SalePrice,
     xlab = "Lot Area",
     ylab = "Sale Price")
```




```
# Plot Total above ground rooms vs Sale Price
plot(train$TotRmsAbvGrd,
     train$SalePrice,
     xlab = "Above Ground Rooms",
     ylab = "Sale Price")
```




```
# Remove outliers
train <- train %>% filter(LotArea < 100000)

# Create a correlation matrix for Lot Area, Year Built, and Sale Price
correlation <- cor(train %>% dplyr::select(LotArea, YearBuilt, SalePrice) %>% as.matrix())
correlation
```

```
##              LotArea  YearBuilt SalePrice
## LotArea   1.00000000 0.04230918 0.3544944
## YearBuilt 0.04230918 1.00000000 0.5255868
## SalePrice 0.35449443 0.52558678 1.0000000
```

```
# Run a correlation test with 80% confidence interval
cor.test(train$LotArea, train$SalePrice, conf.level = 0.8)
```

```
##
##  Pearson's product-moment correlation
##
## data:  train$LotArea and train$SalePrice
## t = 14.456, df = 1454, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.3247557 0.3835329
## sample estimates:
##       cor
## 0.3544944
```

```
cor.test(train$YearBuilt, train$SalePrice, conf.level = 0.8)
```

```
##
##  Pearson's product-moment correlation
##
## data:  train$YearBuilt and train$SalePrice
## t = 23.558, df = 1454, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.5008255 0.5494885
## sample estimates:
##       cor
## 0.5255868
```

The correlation between Lot Area and Sale Price is 0.35, which indicates they are weakly correlated at best. The correlation between Year Built and Sale Price is 0.52, which is another weak correlation. The p-value is less than 0.05 though for both, which suggests the correlation is real.

Linear Algebra and Correlation

```
# Invert the correlation matrix
precision = solve(correlation)

# Multiply the correlation and precision matrices
round(correlation %*% precision)
```

```
##           LotArea YearBuilt SalePrice
## LotArea         1         0         0
## YearBuilt       0         1         0
## SalePrice       0         0         1
```

```
round(precision %*% correlation)
```

```
##           LotArea YearBuilt SalePrice
## LotArea         1         0         0
## YearBuilt       0         1         0
## SalePrice       0         0         1
```

```
# Conduct LU decomposition
library(matrixcalc)
```

```
## Warning: package 'matrixcalc' was built under R version 4.0.3
```

```
decomp <- matrixcalc::lu.decomposition(correlation)
decomp
```

```
## $L
##            [,1]     [,2] [,3]
```

```
## [1,] 1.00000000 0.000000    0
## [2,] 0.04230918 1.000000    0
## [3,] 0.35449443 0.511504    1
##
## $U
##      [,1]       [,2]      [,3]
## [1,]    1 0.04230918 0.3544944
## [2,]    0 0.99820993 0.5105884
## [3,]    0 0.00000000 0.6131657
```

Calculus-Based Probability & Statistics

```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.0.3
```

```
##
## Attaching package: 'MASS'
```

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

```
# Fit Exponential Probability Density
ep = MASS::fitdistr(train$LotArea, "exponential")
ep
```

```
##        rate
##   9.904415e-05
##  (2.595662e-06)
```

```
# Find the optimal lambda
lambda = ep$estimate
lambda
```

```
##         rate
## 9.904415e-05
```

```
# Get 1000 values
samples = rexp(1000, lambda)
samples
```

```
##    [1] 12759.353408  6758.703443  1646.158875 14978.258847  4125.725679
##    [6]  3972.972632 19076.061572  2597.712978 14967.294877 14016.218359
##   [11]   920.934735  2680.105893  5310.812064  8079.675692 11522.474613
##   [16] 11538.935490  2144.935351  1592.732518 13517.936092 40556.932542
##   [21]  8722.787067 23959.528560  2228.853826 14166.285138  1043.722041
##   [26]  9975.405026 12082.506550  9471.273655 36416.087021  3684.155675
##   [31]  1955.995685 30477.861548 20337.597183  3186.218267 20151.545872
```

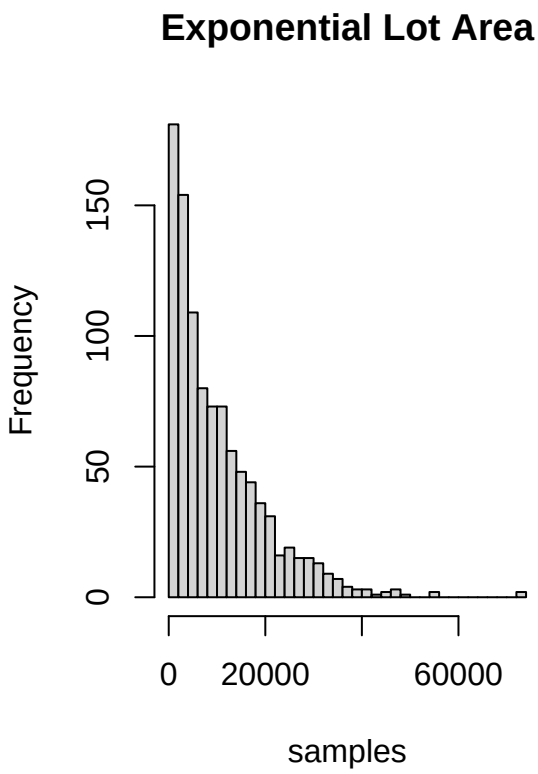
```
##   [36]  3862.866029  4148.178165  3170.693961   854.263960 25973.790525
##   [41] 33215.740557  6435.568467  1268.136506 21244.180496  3506.718879
##   [46] 10517.833929 23917.661910 20544.206145  4320.607816 15634.186493
##   [51] 21595.271765 17636.750649  4012.054044  2656.947318 19024.572725
##   [56]  2315.103519  2266.475401  9248.438920 14811.978680 10338.222691
##   [61]  7256.065942 24665.272543 17787.576267  7653.117254  1025.728318
##   [66]  4636.119880  2730.942597  5159.999017  2486.115937  3305.584220
##   [71]  1554.227525  4242.922685  2006.430690  5128.508375 29286.313772
##   [76]   669.389030 11192.717587 18585.455663   505.045197 21641.270356
##   [81] 15726.146180   674.594378   790.706901   765.385516  2240.183303
##   [86]  5624.937003   317.842570 12740.188440 11702.790121 31830.531486
##   [91] 10212.736952 28385.222982  1234.941766 13969.032500  4408.898826
##   [96]  9436.529802 21843.985821 10434.139435  5800.810703  9046.358672
##  [101]  1710.525182  5966.001130 20110.185241    41.587773  7709.612614
##  [106] 21375.947748  2148.600290 20482.141794  1029.798217   779.470567
##  [111] 16266.434663 13966.209077 12936.387347   767.215722  6809.852777
##  [116]  5269.500506  2980.571350  6515.954087  6920.203856  7347.088553
##  [121]  5361.034571  6034.129449  1231.155835 17982.337561  3839.503910
##  [126]   832.605173 17902.241234  4607.888469 34588.925162 24282.721304
##  [131] 30635.238730  3846.878728    76.164533  4003.398989  1883.077241
##  [136]  2003.089191 72717.001619 14456.087671  3726.302270  4421.688987
##  [141]   488.420939  8233.621744  4349.327389  3995.937408  8782.380507
##  [146]  3392.573755 14473.475721  7075.375824  2212.433047  1017.627837
##  [151] 13495.780204  1650.628242 33946.433913  2234.193764 21669.244477
##  [156]  9051.528384  1183.831393  4886.524305 27250.576675  4715.335810
##  [161]  9182.750055  3972.403302  2824.462743 12650.070319  8733.167108
##  [166]  1107.402786  6640.017136  8740.675327 17456.928835 19887.623402
##  [171]  1056.588745  3387.398805 21989.390478  4300.540984  9606.933931
##  [176]  2083.854043  8179.737860  7694.539605 16094.486431  4152.863113
##  [181]  4852.189899  4441.958216  4661.281292 33006.888784  4957.485248
##  [186]  1528.799629   791.319055  5020.780851    36.087572  8718.373760
##  [191] 11497.603228 12334.624432 20361.657096 12189.876059   841.612223
##  [196]  1880.603489 25010.466711 14696.607560  3333.462496 10833.266722
##  [201]  2660.485721  2073.717864 28869.849284  1816.209693  4616.066930
##  [206] 25782.775616  2439.775720  5164.194678 22944.772994  2741.843063
##  [211]  2490.977094 14602.894970 11073.511436 12248.674383  7848.279222
##  [216]  1719.334834 13011.007157 15201.494783  3711.463130  9961.034742
##  [221]   715.659455  5203.420198 13077.511824  2653.915969    77.237987
##  [226]  4087.488920  4524.220688  1300.886121 28618.648905 42612.728791
##  [231] 16640.285622  2208.775004  3453.553777  2938.458950   955.370639
##  [236]  3220.677287 34719.903048 10220.668791  6543.585215 12746.226583
##  [241]  6960.042285 10209.163591  6838.752977  2713.817831  6681.737135
##  [246] 31686.565128  6043.739277  2049.190092   465.035219  2225.499654
##  [251] 11770.824109  2552.104860 10589.996434    54.398733  1877.614905
##  [256]  4462.848034  2709.700045 10009.304067  7227.212387 14386.197844
##  [261]   103.089174  5869.357099  6873.651672 18606.600825 10729.253641
##  [266]  2221.760285  6577.071206  2778.509689 29206.282889 11333.419751
##  [271] 14823.799587 25941.275489  1744.361414 18814.791587  4756.039278
##  [276] 10723.743113  5849.094270 16869.421191 19466.617393  5947.830033
##  [281] 16677.841516 11617.831945 28514.718554   221.425539  8463.517170
##  [286]   262.429168  6342.918978 13974.125448 11095.223350 16820.157203
##  [291]  5170.241707  7857.933534  1898.726611  8506.388756 19149.505580
##  [296] 11494.700677  3696.098611 19830.463142   715.424479  5049.832540
##  [301] 33714.331646 22927.100910  3387.857923  4764.888591   186.585200
```

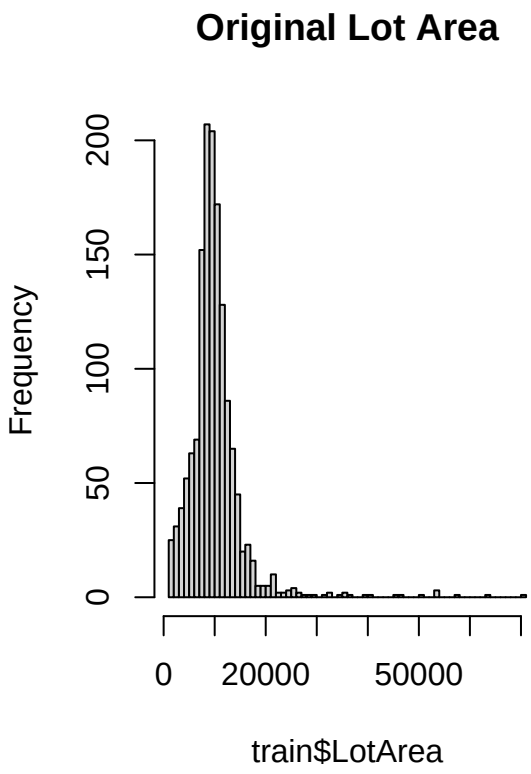
```
##  [306] 27949.671309   807.183373   784.677409  3507.483751 37642.631091
##  [311]  2048.173861 12117.436237 10674.458804  2901.917274 19674.273466
##  [316] 19768.534558  7194.941397  7571.269426  1583.961861 12304.356122
##  [321]  4848.450533 10814.814764 13718.570444 10753.812954 11474.882401
##  [326]  8743.813285 13658.346741  1367.188850  9373.933075  9115.087934
##  [331]   522.904369  2552.715451 24052.054569  1433.880588  7531.587612
##  [336]  8484.520189  4557.828693  3656.147052  6277.329558  2291.013936
##  [341]  6872.484780   493.894884  3015.094283  7897.151778  8634.307531
##  [346] 28155.212728  1012.265574 17764.608471 26924.839300 34062.041408
##  [351]   645.394815 22204.836896 19499.320096 23216.333512 15469.182391
##  [356]  7265.617215   956.550174 26779.794481  8443.273409  8808.448412
##  [361] 47428.130991 15693.018330 14719.114570  2762.416921  5450.403453
##  [366]  3253.830639 19378.963694  1635.363842  7254.351007 14050.668661
##  [371]   875.828921  2387.939955  7637.550651  2398.031123 19892.629484
##  [376] 10437.826402 31937.762695  6600.681591 26002.300967  9734.618677
##  [381] 27555.061302   634.432158 21191.957355 10749.119093  9129.594932
##  [386] 35400.521769  2209.572834 24223.245534  3042.545661  1025.267978
##  [391] 12402.126123  1385.405191 28566.035963 10638.992097 29271.173719
##  [396] 16844.100039 35085.423025  5904.078888 11749.955169 15913.839835
##  [401]  1754.612622 12730.313515 23837.108139  5985.088766 11408.800197
##  [406]  5938.296239 32725.791072 10435.290899 10282.548776   930.341632
##  [411] 16581.755918  7069.410528   310.577905  3207.181640  2775.471627
##  [416] 25479.885681    63.659437  5089.107378 11973.659869  2759.855872
##  [421]  3507.778195  6649.658577  2212.281342 10875.744980 20658.422695
##  [426]  6814.335778  4068.525080 12650.720789  9361.456893 10390.958457
##  [431]  2030.950648   611.370501 20506.047723  1396.989642  4461.314401
##  [436]  2206.387345  4930.306832  6143.639872 19060.817916  1119.846535
##  [441]  1276.532322 18202.651814 13730.568054  8243.136505  4416.221222
##  [446] 12534.611164 30233.155184 14137.894538 17267.424818  6125.916457
##  [451]  2008.825927  3564.113864   664.553774  2704.981183  6044.094997
##  [456] 15675.250550 30293.837786  1504.340746  1481.125097  6840.403279
##  [461] 40031.076585 11200.981863  5714.159680   511.863739  3337.978742
##  [466] 29386.779481 10353.723382  2524.603683  4857.879582 22515.860206
##  [471]  3727.165724  9736.827193 14264.048817  1479.265870 18122.865260
##  [476]  8892.012935  4320.213757   628.340525   416.298462 16849.932108
##  [481] 25293.633144  9759.032311  9310.422658  3462.314634 15085.505019
##  [486]  3409.496354  4212.448444  3371.852110 10100.478909 18146.576647
##  [491] 11668.748755   584.916406  3266.908426  4140.868840 28249.242568
##  [496]  3418.984693  2160.849900 12188.204356 45624.935639 28525.780829
##  [501] 15291.718028   213.273318 18549.265323 54356.939633 23888.293486
##  [506] 27686.955803 14400.149467 14429.034675 11858.964823   352.182070
##  [511]   937.436836  9070.657528 14752.097570 11328.281462  3284.316075
##  [516] 20734.949398  4269.710153  1974.145934 15515.961972  4597.355921
##  [521]  7890.627678  7159.567112   527.924661  1596.066075  1909.702017
##  [526]  6048.585055  3986.179693 11066.356588 27079.869239 37231.436941
##  [531]   292.890899  6457.729282 15542.461838  9102.932164  2860.762383
##  [536]  7437.031323 10406.754056 23929.568356  3058.351210 16442.823283
##  [541] 25207.956862 22024.019370  4688.554160  3123.872483  1150.617233
##  [546]  1445.827957  4847.352194    32.131666    41.953278 12857.940088
##  [551]  8597.401363 35682.037726  8510.870606 10372.542841  2329.747784
##  [556] 20884.213241 32329.234660 18595.174951  2518.883750 12817.296538
##  [561]  6285.986875  4121.294513 20795.035336  3895.243332  1054.885729
##  [566] 13185.506254 20784.527328  3180.835205  1685.340558 14849.322116
##  [571] 25002.300716  8871.562399 10253.808165  1842.937291  9604.905944
```

```
##  [576]  5169.952750 21460.272424   145.774810 24274.111633  2035.826812
##  [581]  3169.189328   525.543633 17797.112802  7905.057548 10172.551897
##  [586] 15441.912055 25968.949115 18657.797170   961.333750   582.410625
##  [591]  7581.989690 20122.879285 10471.654478  4379.169582  5573.332347
##  [596]   432.456674 11141.475176  4723.006422  4456.055831   109.590691
##  [601] 15961.549893 11367.261866 14557.707745 10854.381293  9859.720754
##  [606] 30888.043412 39396.256390  4267.743498  3038.644468 16502.078008
##  [611] 22106.019113 38221.339873  2886.041296   280.641102  6228.980858
##  [616]  4182.643894   230.397611   132.937428 29891.593774  4617.431325
##  [621]  3593.382377 47473.801498 25654.102920  3513.717164 21056.071797
##  [626]  8731.443970  7070.301373  2266.728068  9715.412574 14545.852855
##  [631]   839.357829 34404.194085  3082.362130  1104.226377  3735.500796
##  [636]  1151.453761 11286.924454  9336.885745  5577.999249  2122.859488
##  [641]  7316.372865 12752.315804  3418.688285  1439.528762 28676.331165
##  [646] 31214.759500  4555.051096  4598.942991  4556.082523 17195.591617
##  [651]  1343.252719 16128.430512   580.857212   969.558882  2767.502619
##  [656]  6315.161406  1583.309556 17259.156026 16283.188364 12915.619479
##  [661] 11843.703780  8881.092643  4881.556315 17619.725749  8501.749738
##  [666]  4856.600896     2.911364  5046.311392  2332.627936  7426.744026
##  [671] 11926.111778 15321.980815 14466.485535 12552.411745   779.646904
##  [676]  4478.838560 18748.082534 19019.056620 13070.373992 17278.044946
##  [681] 18322.813871 22635.580646  8948.071462 10979.083141  3096.925392
##  [686]  5193.259649 15579.340810  3024.716998  9571.714507 18435.146453
##  [691] 16480.255018  5087.693387   120.246009   178.514045 10802.717650
##  [696] 18459.186131  6597.060018  8513.922694  3587.560764 12126.023322
##  [701] 17694.996431  2938.394793   950.896817 13247.096773 20373.894881
##  [706]  5178.766812 15272.573192   301.345414 10684.385570  7065.900588
##  [711] 14970.625787 12601.051367 18815.045207 12519.580012  1416.591695
##  [716]  3073.618144  8617.932564 12467.393741  7926.797834 13327.189915
##  [721]  8882.203827  3101.915562  1933.480141 38004.972763  5482.761977
##  [726] 10693.552414  4521.717473 17461.485485  1168.490829  6969.097734
##  [731]   208.900800  4954.955769  8616.033620  8275.692480 17849.116282
##  [736]  1867.267262 30137.003165  1709.512964  2153.543932 18773.086670
##  [741] 16314.069761 13750.959121   275.013705  7029.379923 13753.425142
##  [746]  3585.834138     1.586600  4621.954548 24180.794951 10589.006559
##  [751] 31699.762048 73437.226397  1014.720572  8268.486899  6392.931115
##  [756] 29936.270610  8405.116192  4531.588439 17023.508672  1285.091553
##  [761]  4444.785987  5360.671090 10592.690847   892.674683  5460.578925
##  [766]  1118.332614   561.104135  1407.046838 20761.447255 14364.669702
##  [771] 54119.344072  1411.943890 12517.124128  1728.615848   165.268369
##  [776] 15141.414672 17869.620637  6672.821515 18712.423570  7780.916923
##  [781] 26283.493800  2449.484568  8516.893361  8618.388938 24779.255670
##  [786]  1473.913482  6432.769463  1116.148640   556.425791  2927.925506
##  [791]  2198.784859  2587.054083 16023.730000  1012.590991 11961.781800
##  [796]  1968.475175  1565.745108  6362.282280  2075.313162  5321.446647
##  [801]   773.033255 14764.661576  2506.478131 13212.812524  1109.514049
##  [806]  1392.721043  8420.662271  3603.200593  2801.796365  3875.457040
##  [811]   183.752519 24166.483876  2841.846378  5295.602282 16937.724502
##  [816]  1352.773561 21319.562588  5721.908413  7094.807690 10675.368802
##  [821] 13911.707117  1043.289915 10601.815949   334.580850   954.449267
##  [826] 16964.339184  4495.326125  6688.084758  2412.930947  2701.402732
##  [831]   199.976920  9554.287608  7430.985615 13588.927263  3138.215206
##  [836]  9912.594294 21256.511314  1322.030306 13771.886616  3853.706654
##  [841] 10206.481802  2854.539248 10560.178697  9973.180646 14116.044204
```

```
##  [846]  6468.035697 19150.509982 33101.187081  1210.440558  2779.834853
##  [851]   276.752890 18029.593601  9022.090357  2675.694459 30684.362082
##  [856]  9705.833556   162.440016  7889.661516 12805.442475  7463.632228
##  [861]  1168.946221 17359.059433 11118.713380 16437.663840 13188.256916
##  [866]  3292.694491  5717.583784  2537.258824   343.419191 12862.025169
##  [871] 21967.475676  7385.886516 11036.181430 22030.804237 12115.082523
##  [876] 20299.975922 18803.358577 21887.991023 33886.359503 13398.315192
##  [881]  2571.171738 36469.019575 18533.860291  1635.371059 16455.316158
##  [886]  6240.058332 16831.139325 41497.564895 14948.806706  6720.791142
##  [891] 10272.720368 15554.134545  3145.449345  7013.390277   440.769955
##  [896]  5586.995825  1185.429967 16762.909013   106.540696  8763.591278
##  [901]  1553.421129 19818.313582 12017.297919 16516.567667  7235.586085
##  [906] 17637.960504 16247.287439 14162.653648  6614.473516    80.966591
##  [911]  3747.529014   871.149600  5981.543438  1412.705316 27668.575491
##  [916] 18726.208141 11440.247407  5661.233474  2541.202250  1886.261211
##  [921]   147.503558  2039.480723 11261.454739 14183.088459  1143.534333
##  [926]  5676.761640  4689.618007   325.489020 12224.663393  3118.352817
##  [931]  4188.295635 32245.914947  7032.485374 26245.669997  9658.412941
##  [936]   944.598658  3827.160688  8607.047364  2667.189787 12183.824982
##  [941]  9108.130478 27005.701493 26832.882533   325.080742  3052.047260
##  [946]  8684.330930 12800.381836 47342.924714  6257.082340  2331.766080
##  [951]  8311.747013  4202.287169  3700.057921 10287.866055  1494.454753
##  [956]  5775.005742  5470.158014   500.304736 16079.600450  2640.780587
##  [961]  7302.092585 10655.770648  1553.983571  2932.010000  1452.289621
##  [966] 18072.862949  4380.741420  2040.133562   437.396406 20798.786536
##  [971]  4233.884768 23780.180272  8332.681846 45807.021602  2610.010970
##  [976]  1475.611905  2661.326989  1272.384199  3443.817813  3998.445513
##  [981]  9128.095889  2709.427494 26546.407486  2083.494454 49552.561789
##  [986] 23216.518979 25374.126515  8078.335234  5605.701484  5473.094571
##  [991]  5700.056701 26616.571202 13093.457779  5956.223062    60.493708
##  [996] 10885.181493  8598.745918 15722.304956 30105.551627  7389.696597
```

```r
# Plot histogram of original and exponential
par(mfrow = c(1, 2))
hist(samples, breaks = 50, main = "Exponential Lot Area")
hist(train$LotArea, breaks = 50, main = "Original Lot Area")
```






```
# Percentiles based on exponential
qexp(0.05, rate = lambda)
```

```
## [1] 517.8831
```

```
qexp(0.95, rate = lambda)
```

```
## [1] 30246.43
```

```
# Empirical 95% confidence interval assuming normailty
me <- qnorm(0.975)*nrow(train)/sqrt(sd(train$LotArea))
upper <- mean(train$LotArea) + me
lower <- mean(train$LotArea) - me

# Empirical percentiles
quantile(train$LotArea, 0.05)
```

```
##     5%
## 3294.5
```

```
quantile(train$LotArea, 0.95)
```

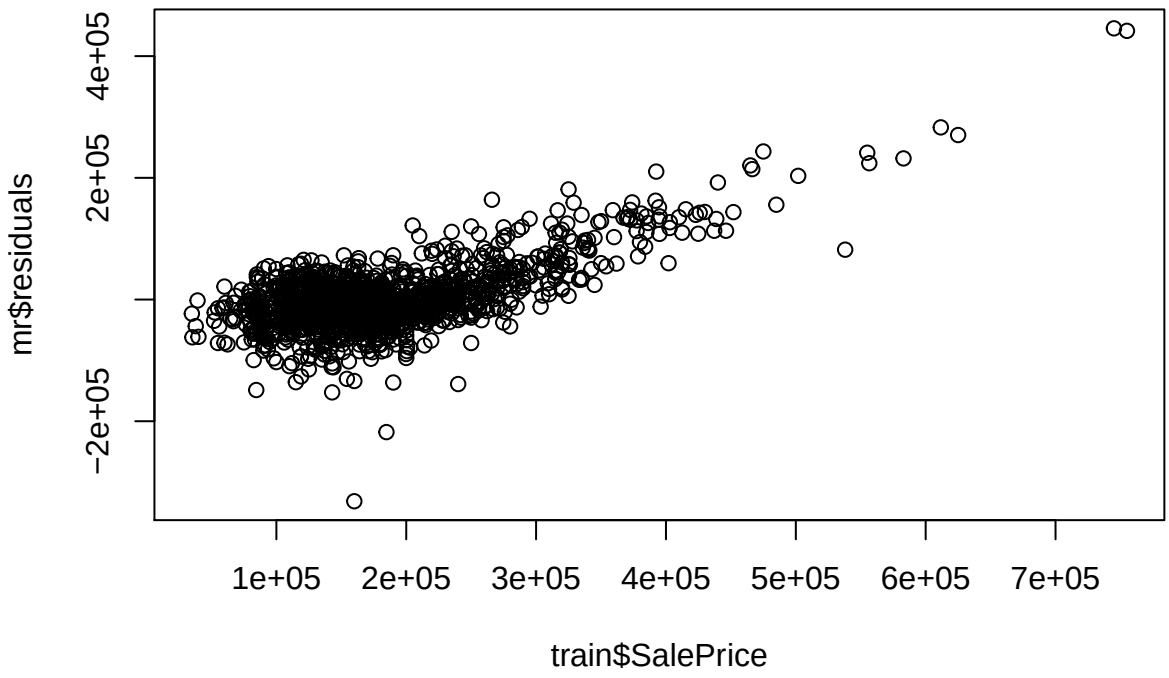
```
##   95%
## 17108
```

Modeling

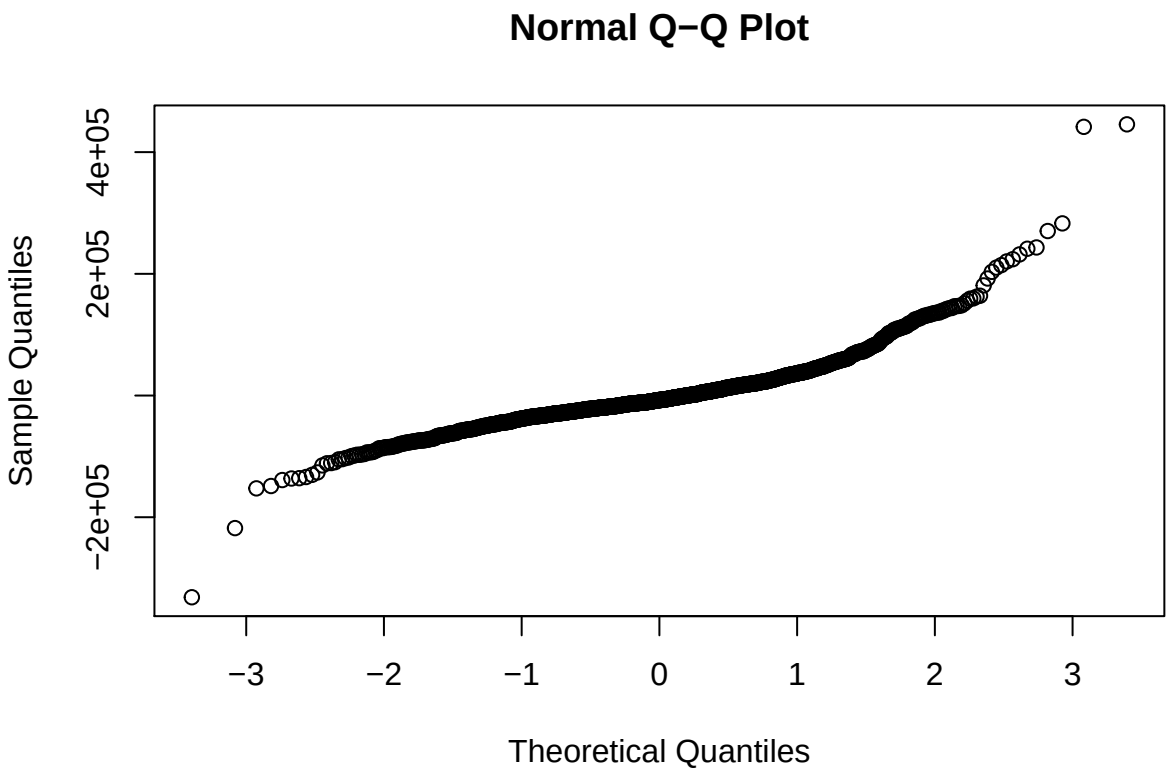
```
# Create multiple regression
mr <- lm(SalePrice ~ LotArea + YearBuilt + TotRmsAbvGrd, data = train)
summary(mr)
```

```
##
## Call:
## lm(formula = SalePrice ~ LotArea + YearBuilt + TotRmsAbvGrd,
##     data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -331486  -27636   -7406   19341  445648
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.444e+06  9.110e+04  -26.83   <2e-16 ***
## LotArea       2.809e+00  2.605e-01   10.79   <2e-16 ***
## YearBuilt     1.248e+03  4.640e+01   26.91   <2e-16 ***
## TotRmsAbvGrd  2.078e+04  9.080e+02   22.88   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 53270 on 1452 degrees of freedom
## Multiple R-squared:  0.5494, Adjusted R-squared:  0.5484
## F-statistic:   590 on 3 and 1452 DF,  p-value: < 2.2e-16
```

```
plot(mr$residuals ~ train$SalePrice)
```




```
qqnorm(mr$residuals)
```




```
submission <- (2.809*test$LotArea)+(1248*test$YearBuilt)+(20780*test$TotRmsAbvGrd)-2444000

#submission <- cbind(test,submission) %>% select(Id, submission)
#names(submission) <- c("Id","SalePrice")
#write.csv(submission, file = "DMosteSubmission.csv")
```

My username is davidmoste and I hade a score of 0.285.