DATA 605 Final Exam

CUNY Spring 2021

Philip Tanofsky

24 May 2021

Final Exam: Computational Mathematics

Problem 1

Using R, generate a random variable X that has 10,000 random uniform numbers from 1 to N, where N can be any number of your choosing greater than or equal to 6. Then generate a random variable Y that has 10,000 random normal numbers with a mean of $\mu = \sigma = (N + 1)/2$.

```r
set.seed(1234)

N <- 10
n <- 10000
mu <- (N + 1) / 2
sigma <- mu

X <- runif(n, min=1, max=N)
Y <- rnorm(n, mean=mu, sd=mu)

x <- quantile(Y, 0.5)
y <- quantile(Y, 0.25)
```

Probability

Calculate as a minimum the below probabilities a through c. Assume the small letter "x" is estimated as the median of the X variable, and the small letter "y" is estimated as the 1st quartile of the Y variable. Interpret the meaning of all probabilities.

A.

$$P(X > x \mid X > y)$$

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

The probability of the intersection of events A and B divided by the probability of B

```
p.int.a.b <- length(which(X > x & X > y)) / n
p.b <- length(which(X > y)) / n
#p.int.a.b
#p.b

result.a <- p.int.a.b / p.b
#result.a
```

Answer: 0.548337.

B.

$$P(X > x, Y > y)$$

The probability of events A and B.

```
p.a.b <- length(which(X > x & Y > y)) / n
result.b <- p.a.b
#result.b
```

Answer: 0.3695.

C.

$$P(X < x \mid X > y)$$

The probability of the intersection of events A and B divided by the probability of B. (See part A for definition.)

```
p.int.a.b <- length(which(X < x & X > y)) / n
p.b <- length(which(X > y)) / n
#p.int.a.b
#p.b

result.c <- p.int.a.b / p.b
#result.c
```

Answer: 0.451663.

Build a Table

Investigate whether $P(X > x \text{ } and \text{ } Y > y) = P(X > x) * P(Y > y)$ by building a table and evaluating the marginal and joint probabilities.

$P(A \text{ } and \text{ } B) = P(A) * P(B)$ for independent events

```
p.x.y <- length(which(X > x & Y > y)) / n
p.x.not.y <- length(which(X > x & Y < y)) / n
p.not.x.y <- length(which(X < x & Y > y)) / n
p.not.x.not.y <- length(which(X < x & Y < y)) / n
```

```
table <- matrix(c(p.x.y, p.not.x.y, sum(p.x.y + p.not.x.y),
                   p.x.not.y, p.not.x.not.y, sum(p.x.not.y + p.not.x.not.y),
                   sum(p.x.y + p.x.not.y), sum(p.not.x.y + p.not.x.not.y), sum(p.x.y + p.not.x.y + p.x.
                nrow=3, ncol=3, byrow=T,
                dimnames=list(c('Y>y', 'Y<y', 'Marginal'), c('X>x', 'X<x', 'Marginal')))
table
```

```
##             X>x    X<x Marginal
## Y>y      0.3695 0.3805     0.75
## Y<y      0.1251 0.1249     0.25
## Marginal 0.4946 0.5054     1.00
```

```
x.and.y <- sum(p.x.y + p.x.not.y) * sum(p.x.y + p.not.x.y)
x.and.y
```

```
## [1] 0.37095
```

The multiplication of marginal probabilities results in 0.37095 for $P(A) * P(B)$ which is quite close to 0.3695, the joint probability for $P(A\ and\ B)$, a difference of 0.00145. Thus, in practice the definition of probability stands anecdotally.

Check Independence

Check to see if independence holds by using Fisher's Exact Test and the Chi Square Test. What is the difference between the two? Which is most appropriate?

Fisher's Exact Test

```
tab.ind <- table(X > x, Y > y)
fisher.test(tab.ind)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  tab.ind
## p-value = 0.5031
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.8846876 1.0625460
## sample estimates:
## odds ratio
##  0.9695309
```

The Fisher test results in p-value of 0.5031, thus we cannot reject the null hypothesis. The null hypothesis states that the variables are independent.

Chi Square Test

```
chisq.test(tab.ind)
```

```
## 
##  Pearson's Chi-squared test with Yates' continuity correction
## 
## data:  tab.ind
## X-squared = 0.41818, df = 1, p-value = 0.5178
```

The Chi-squared test results in p-value of 0.5178, thus we cannot reject the null hypothesis. The null hypothesis states that the variables are independent.

The Fisher test is exact and typically used when the sample size is small. The Chi-squared test is an approximation for when the samples becomes infinite, and thus more applicable for larger sample sizes. A rule of thumb, the Chi-squared test is not applicable when the expected values in the one of the contingency table cells is less than 5. That scenario does not apply to this contingency table.

Given the size of the sample size, I recommend the Chi-squared test for independence. The p-value results for both tests were quite close, so in this particular scenario both tests of independence are acceptable.

Problem 2

You are to register for Kaggle.com (free) and compete in the House Prices: Advanced Regression Techniques competition. https://www.kaggle.com/c/house-prices-advanced-regression-techniques.

```
library(ggplot2)
library(tidyverse)
df_train <- read_csv("KaggleCompetition/house-prices-advanced-regression-techniques/train.csv")
#df_test <- read_csv("KaggleCompetition/house-prices-advanced-regression-techniques/test.csv")
```

Descriptive and Inferential Statistics

Provide univariate descriptive statistics and appropriate plots for the training data set. Provide a scatterplot matrix for at least two of the independent variables and the dependent variable. Derive a correlation matrix for any three quantitative variables in the dataset. Test the hypotheses that the correlations between each pairwise set of variables is 0 and provide an 80% confidence interval. Discuss the meaning of your analysis. Would you be worried about familywise error? Why or why not?

```
summary(df_train)
```

```
##        Id           MSSubClass      MSZoning          LotFrontage
##  Min.   :   1.0   Min.   : 20.0   Length:1460        Min.   : 21.00
##  1st Qu.: 365.8   1st Qu.: 20.0   Class :character   1st Qu.: 59.00
##  Median : 730.5   Median : 50.0   Mode  :character   Median : 69.00
##  Mean   : 730.5   Mean   : 56.9                      Mean   : 70.05
##  3rd Qu.:1095.2   3rd Qu.: 70.0                      3rd Qu.: 80.00
##  Max.   :1460.0   Max.   :190.0                      Max.   :313.00
##                                                      NA's   :259
##     LotArea          Street             Alley             LotShape
##  Min.   :  1300   Length:1460        Length:1460        Length:1460
##  1st Qu.:  7554   Class :character   Class :character   Class :character
##  Median :  9478   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 10517
##  3rd Qu.: 11602
##  Max.   :215245
##
##  LandContour         Utilities          LotConfig          LandSlope
##  Length:1460        Length:1460        Length:1460        Length:1460
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  Neighborhood        Condition1         Condition2          BldgType
##  Length:1460        Length:1460        Length:1460        Length:1460
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   HouseStyle         OverallQual      OverallCond      YearBuilt
##  Length:1460        Min.   : 1.000   Min.   :1.000   Min.   :1872
##  Class :character   1st Qu.: 5.000   1st Qu.:5.000   1st Qu.:1954
##  Mode  :character   Median : 6.000   Median :5.000   Median :1973
##                     Mean   : 6.099   Mean   :5.575   Mean   :1971
##                     3rd Qu.: 7.000   3rd Qu.:6.000   3rd Qu.:2000
##                     Max.   :10.000   Max.   :9.000   Max.   :2010
##
##   YearRemodAdd   RoofStyle           RoofMatl         Exterior1st
##  Min.   :1950   Length:1460        Length:1460        Length:1460
##  1st Qu.:1967   Class :character   Class :character   Class :character
##  Median :1994   Mode  :character   Mode  :character   Mode  :character
##  Mean   :1985
##  3rd Qu.:2004
##  Max.   :2010
##
##  Exterior2nd         MasVnrType          MasVnrArea      ExterQual
##  Length:1460        Length:1460        Min.   :   0.0   Length:1460
##  Class :character   Class :character   1st Qu.:   0.0   Class :character
##  Mode  :character   Mode  :character   Median :   0.0   Mode  :character
##                                        Mean   : 103.7
##                                        3rd Qu.: 166.0
##                                        Max.   :1600.0
##                                        NA's   :8
##   ExterCond          Foundation          BsmtQual           BsmtCond
##  Length:1460        Length:1460        Length:1460        Length:1460
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  BsmtExposure       BsmtFinType1         BsmtFinSF1     BsmtFinType2
##  Length:1460        Length:1460        Min.   :   0.0   Length:1460
##  Class :character   Class :character   1st Qu.:   0.0   Class :character
##  Mode  :character   Mode  :character   Median : 383.5   Mode  :character
##                                        Mean   : 443.6
##                                        3rd Qu.: 712.2
##                                        Max.   :5644.0
##
##    BsmtFinSF2        BsmtUnfSF       TotalBsmtSF       Heating
##  Min.   :   0.00   Min.   :   0.0   Min.   :   0.0   Length:1460
##  1st Qu.:   0.00   1st Qu.: 223.0   1st Qu.: 795.8   Class :character
##  Median :   0.00   Median : 477.5   Median : 991.5   Mode  :character
##  Mean   :  46.55   Mean   : 567.2   Mean   :1057.4
##  3rd Qu.:   0.00   3rd Qu.: 808.0   3rd Qu.:1298.2
##  Max.   :1474.00   Max.   :2336.0   Max.   :6110.0
##
##   HeatingQC          CentralAir         Electrical           1stFlrSF
##  Length:1460        Length:1460        Length:1460        Min.   : 334
##  Class :character   Class :character   Class :character   1st Qu.: 882
##  Mode  :character   Mode  :character   Mode  :character   Median :1087
##                                                           Mean   :1163
##                                                           3rd Qu.:1391
##                                                           Max.   :4692
##
##     2ndFlrSF     LowQualFinSF       GrLivArea     BsmtFullBath
##  Min.   :   0   Min.   :  0.000   Min.   : 334   Min.   :0.0000
##  1st Qu.:   0   1st Qu.:  0.000   1st Qu.:1130   1st Qu.:0.0000
##  Median :   0   Median :  0.000   Median :1464   Median :0.0000
##  Mean   : 347   Mean   :  5.845   Mean   :1515   Mean   :0.4253
##  3rd Qu.: 728   3rd Qu.:  0.000   3rd Qu.:1777   3rd Qu.:1.0000
##  Max.   :2065   Max.   :572.000   Max.   :5642   Max.   :3.0000
##
##   BsmtHalfBath        FullBath        HalfBath       BedroomAbvGr
##  Min.   :0.00000   Min.   :0.000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:2.000
##  Median :0.00000   Median :2.000   Median :0.0000   Median :3.000
##  Mean   :0.05753   Mean   :1.565   Mean   :0.3829   Mean   :2.866
##  3rd Qu.:0.00000   3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :2.00000   Max.   :3.000   Max.   :2.0000   Max.   :8.000
##
##   KitchenAbvGr   KitchenQual         TotRmsAbvGrd     Functional
##  Min.   :0.000   Length:1460        Min.   : 2.000   Length:1460
##  1st Qu.:1.000   Class :character   1st Qu.: 5.000   Class :character
##  Median :1.000   Mode  :character   Median : 6.000   Mode  :character
##  Mean   :1.047                      Mean   : 6.518
##  3rd Qu.:1.000                      3rd Qu.: 7.000
##  Max.   :3.000                      Max.   :14.000
##
##    Fireplaces    FireplaceQu         GarageType         GarageYrBlt
##  Min.   :0.000   Length:1460        Length:1460        Min.   :1900
##  1st Qu.:0.000   Class :character   Class :character   1st Qu.:1961
##  Median :1.000   Mode  :character   Mode  :character   Median :1980
##  Mean   :0.613                                         Mean   :1979
##  3rd Qu.:1.000                                         3rd Qu.:2002
##  Max.   :3.000                                         Max.   :2010
##                                                        NA's   :81
##  GarageFinish         GarageCars      GarageArea      GarageQual
##  Length:1460        Min.   :0.000   Min.   :   0.0   Length:1460
##  Class :character   1st Qu.:1.000   1st Qu.: 334.5   Class :character
##  Mode  :character   Median :2.000   Median : 480.0   Mode  :character
##                     Mean   :1.767   Mean   : 473.0
##                     3rd Qu.:2.000   3rd Qu.: 576.0
##                     Max.   :4.000   Max.   :1418.0
##
##   GarageCond         PavedDrive          WoodDeckSF      OpenPorchSF
##  Length:1460        Length:1460        Min.   :  0.00   Min.   :  0.00
##  Class :character   Class :character   1st Qu.:  0.00   1st Qu.:  0.00
##  Mode  :character   Mode  :character   Median :  0.00   Median : 25.00
##                                        Mean   : 94.24   Mean   : 46.66
##                                        3rd Qu.:168.00   3rd Qu.: 68.00
##                                        Max.   :857.00   Max.   :547.00
##
##  EnclosedPorch      3SsnPorch       ScreenPorch        PoolArea
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   :  0.000
##  1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.000
##  Median :  0.00   Median :  0.00   Median :  0.00   Median :  0.000
##  Mean   : 21.95   Mean   :  3.41   Mean   : 15.06   Mean   :  2.759
##  3rd Qu.:  0.00   3rd Qu.:  0.00   3rd Qu.:  0.00   3rd Qu.:  0.000
##  Max.   :552.00   Max.   :508.00   Max.   :480.00   Max.   :738.000
##
##     PoolQC             Fence           MiscFeature          MiscVal
##  Length:1460        Length:1460        Length:1460        Min.   :    0.00
##  Class :character   Class :character   Class :character   1st Qu.:    0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :    0.00
##                                                           Mean   :   43.49
##                                                           3rd Qu.:    0.00
##                                                           Max.   :15500.00
##
##      MoSold           YrSold       SaleType         SaleCondition
##  Min.   : 1.000   Min.   :2006   Length:1460        Length:1460
##  1st Qu.: 5.000   1st Qu.:2007   Class :character   Class :character
##  Median : 6.000   Median :2008   Mode  :character   Mode  :character
##  Mean   : 6.322   Mean   :2008
##  3rd Qu.: 8.000   3rd Qu.:2009
##  Max.   :12.000   Max.   :2010
##
##    SalePrice
##  Min.   : 34900
##  1st Qu.:129975
##  Median :163000
##  Mean   :180921
##  3rd Qu.:214000
##  Max.   :755000
##
```

Above is the initial, high-level summary of the training dataset from Kaggle. Later, I will transform character variables to factor and also account for missing data.

Univariate statistics and plots

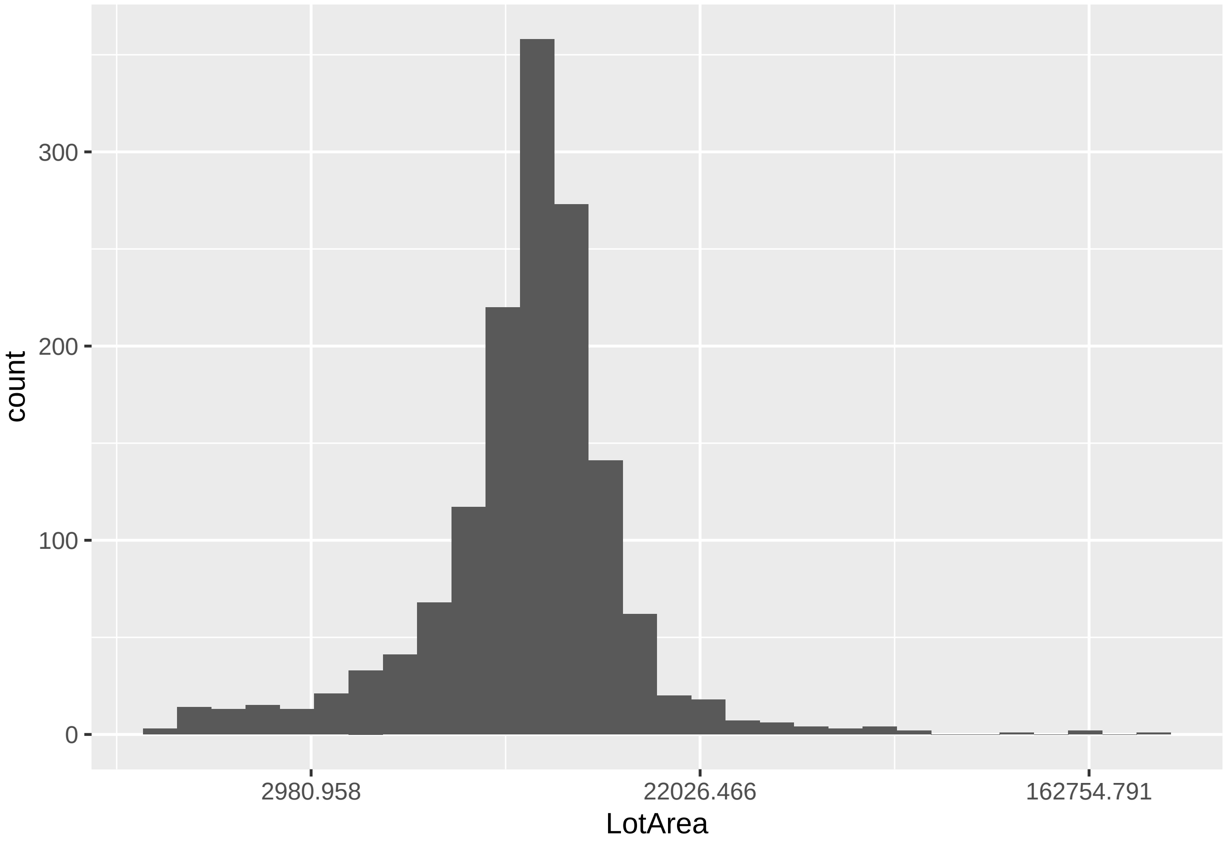
Selecting 4 popular variables in describing houses for sale: lot area (LotArea), above ground living area square feet (GrLivArea), total rooms above grade (TotRmsAbvGrd), and year built (YearBuilt) along with the dependent variable sale price (SalePrice).

Lot area shows a minimum of 1300 square feet and over 200,000 square feet with a mean of 10,517. The histogram shows a near normal distribution when applying log to the lot area values.

```
summary(df_train$LotArea)
```

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

```
# Lot Area
ggplot(df_train, aes(LotArea)) +
  geom_histogram() +
  scale_x_continuous(trans="log")
```

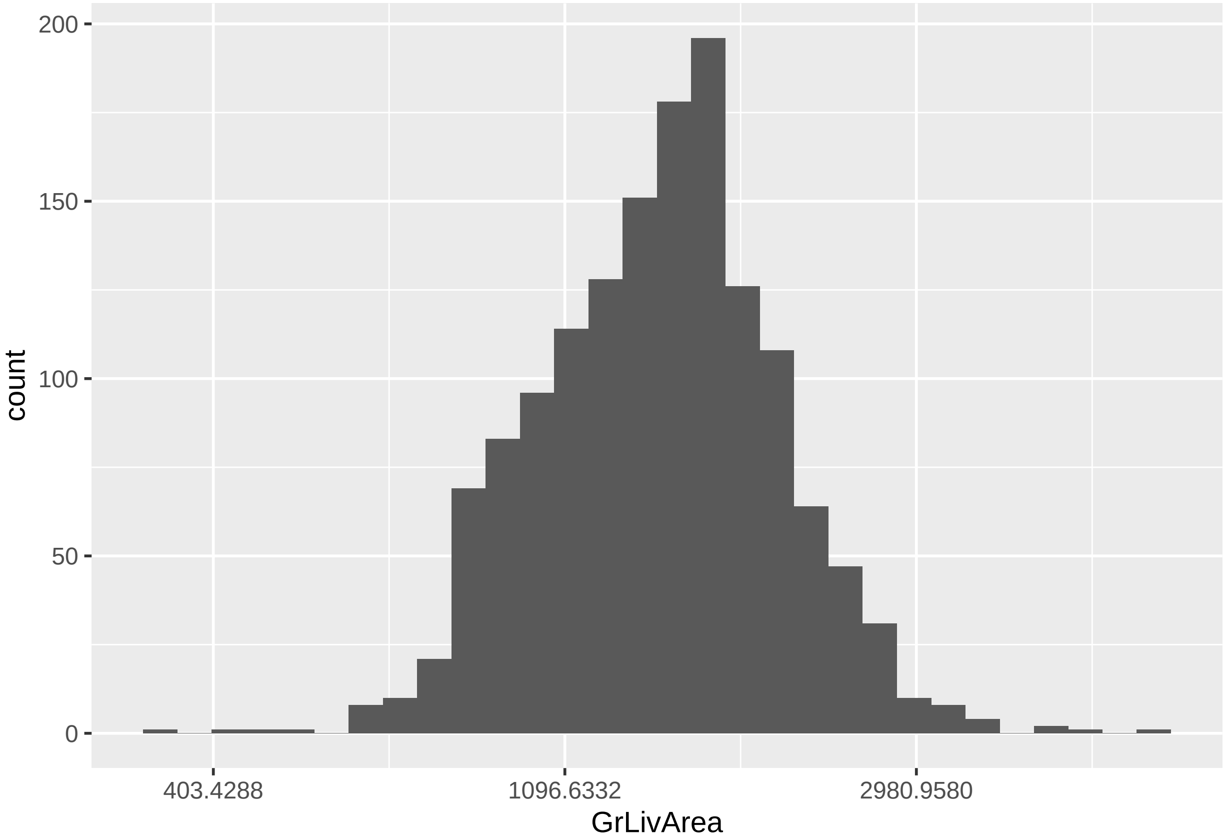


The above ground living area as measured in square feet indicates a minimum of 334 and a maximum of 5,642 and mean of 1,515. The histogram also shows a near normal distribution once applying log to the ground living area values.

```
summary(df_train$GrLivArea)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     334    1130    1464    1515    1777    5642
```

```
# Above ground living area
ggplot(df_train, aes(GrLivArea)) +
  geom_histogram() +
  scale_x_continuous(trans="log")
```

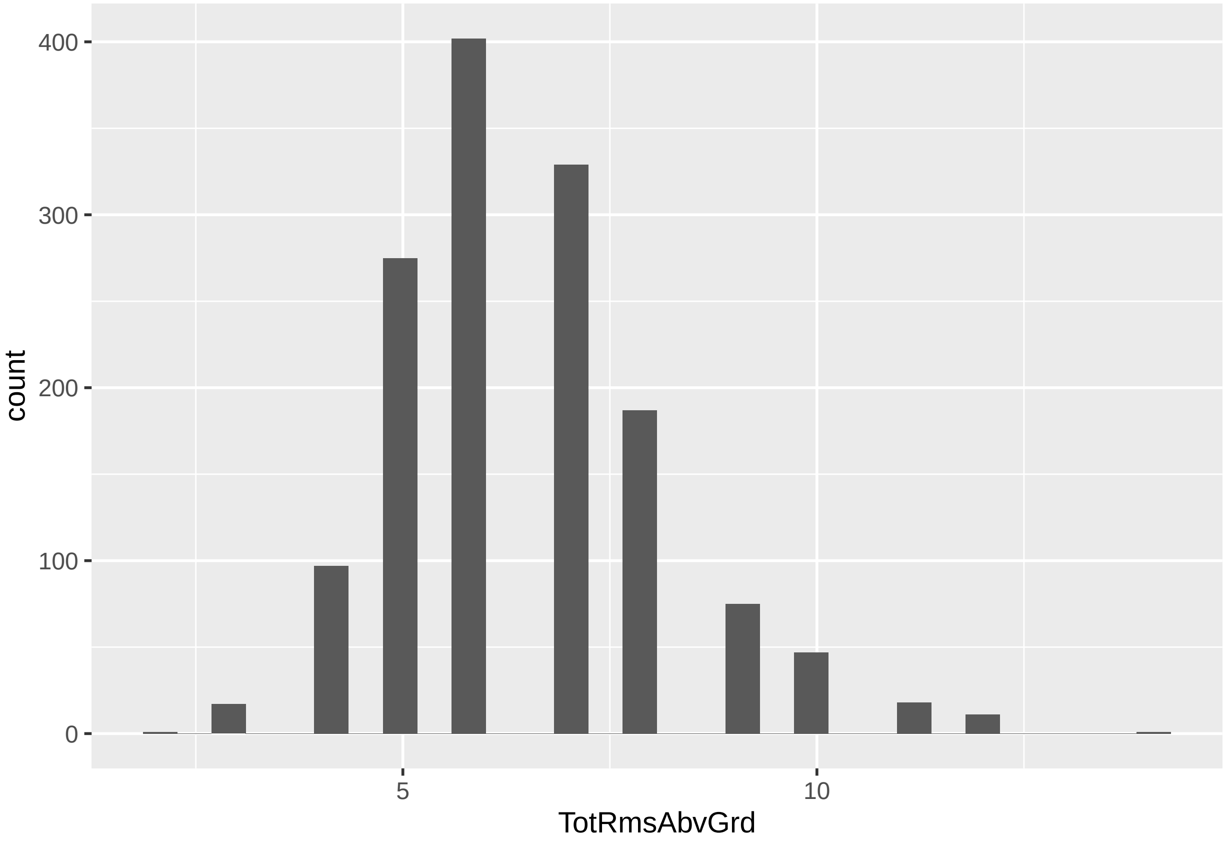


Total rooms above grade indicates a minimum of 2 and a maximum of 14 (must be nice) along with a mean of 6.5. The histogram shows a near normal distribution, interestingly the right skew of this plot doesn't match the left skew of the total living area plot above.

```
summary(df_train$TotRmsAbvGrd)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   5.000   6.000   6.518   7.000  14.000
```

```
# Bedroom Above Ground
ggplot(df_train, aes(TotRmsAbvGrd)) +
  geom_histogram()
```



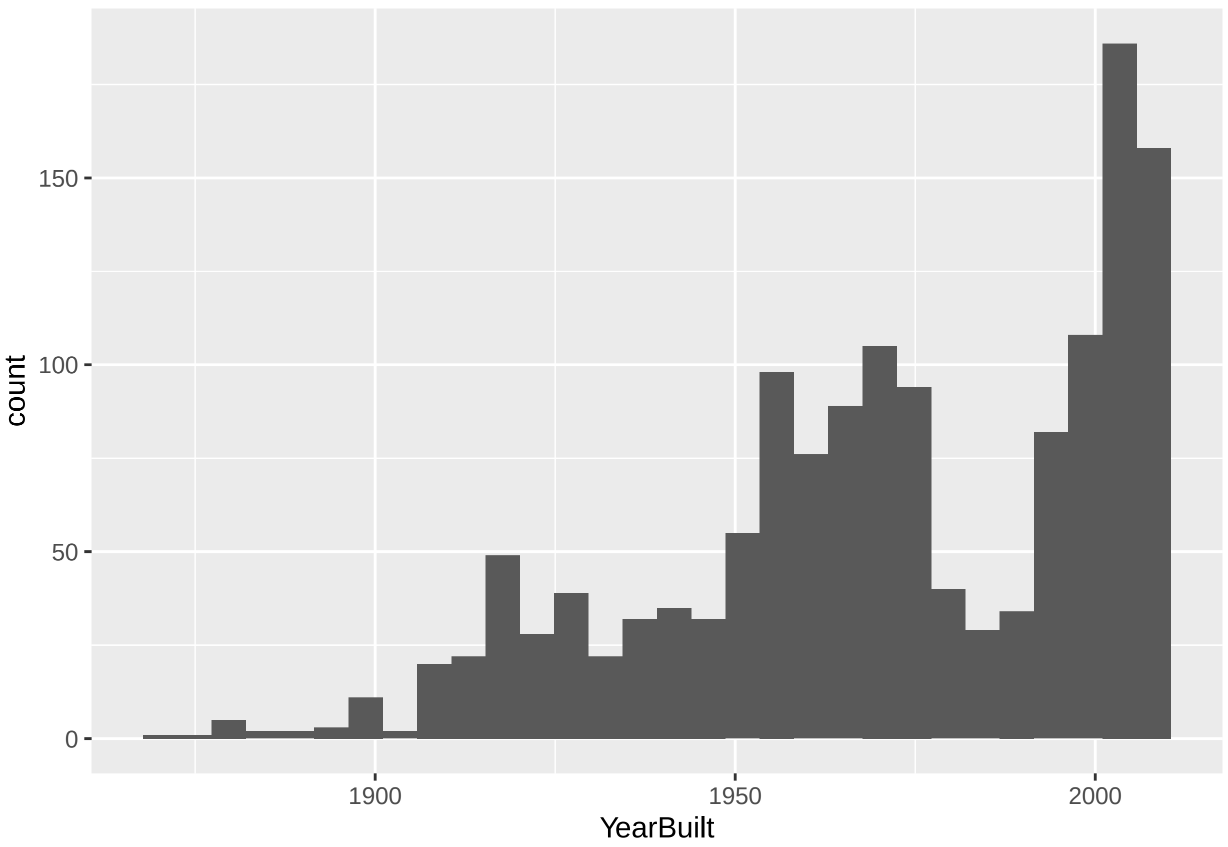
```
#  scale_x_continuous(trans="log")
```

Year built is numerical and can be considered quantitative. With that in mind, the minimum year is 1872 with a maximum of 2010. The median year is 1973. The histogram show a spike in the 1960s and 1970s, followed by a dip in the 1980s and then another large spike in the 1990s and 2000s. Not expecting a normal distribution given the nature of the year variable.

```
summary(df_train$YearBuilt)
```

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

```
# Year built
ggplot(df_train, aes(YearBuilt)) +
  geom_histogram()
```



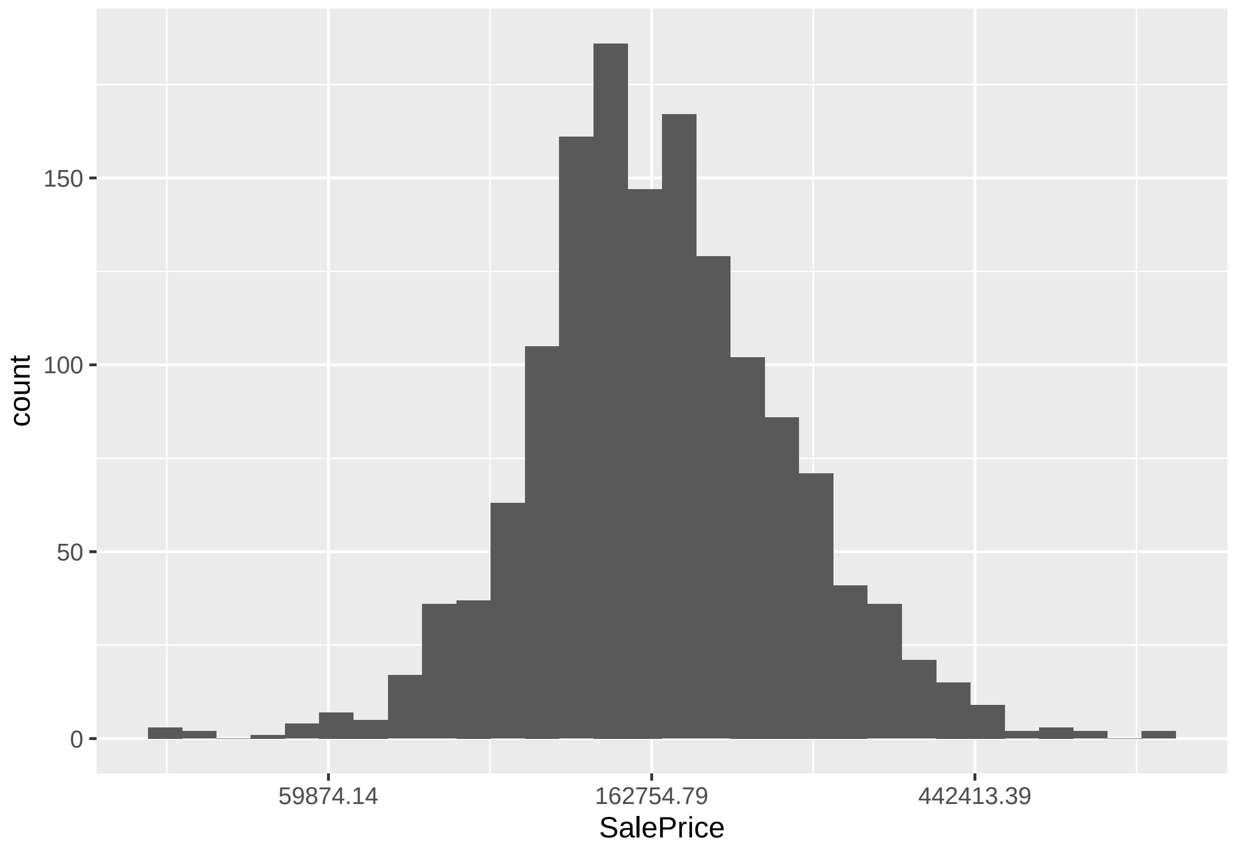
```
#  scale_x_continuous(trans="log")
```

Finally, in assessing the dependent variable SalePrice, the minimum value is 34,900 and the maximum of 755,000. I had to look it up, the dataset represents houses sold between 2006 and 2010 in Ames, Iowa. With the current housing market, I couldn't imagine a maximum house for less than one million dollars. The dataset is 11 years old, so I'll allow it. Overall, the histogram shows an almost normal distribution for the sale prices. A good sign for the later regression modeling to be performed.

```
summary(df_train$SalePrice)
```

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

```
# Sale Price
ggplot(df_train, aes(SalePrice)) +
  geom_histogram() +
  scale_x_continuous(trans="log")
```

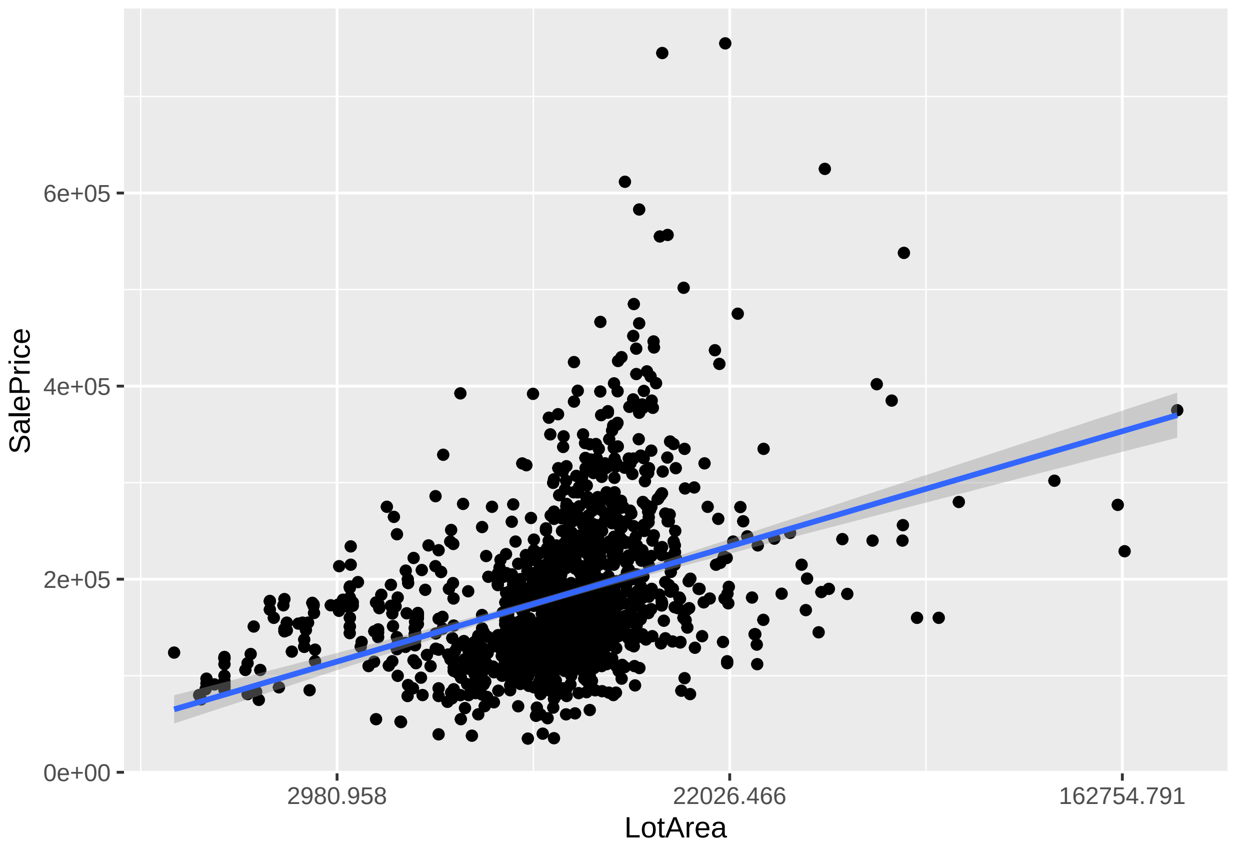


Scatterplots

Scatterplots compare the 4 selected independent variables against the dependent variable of SalePrice.

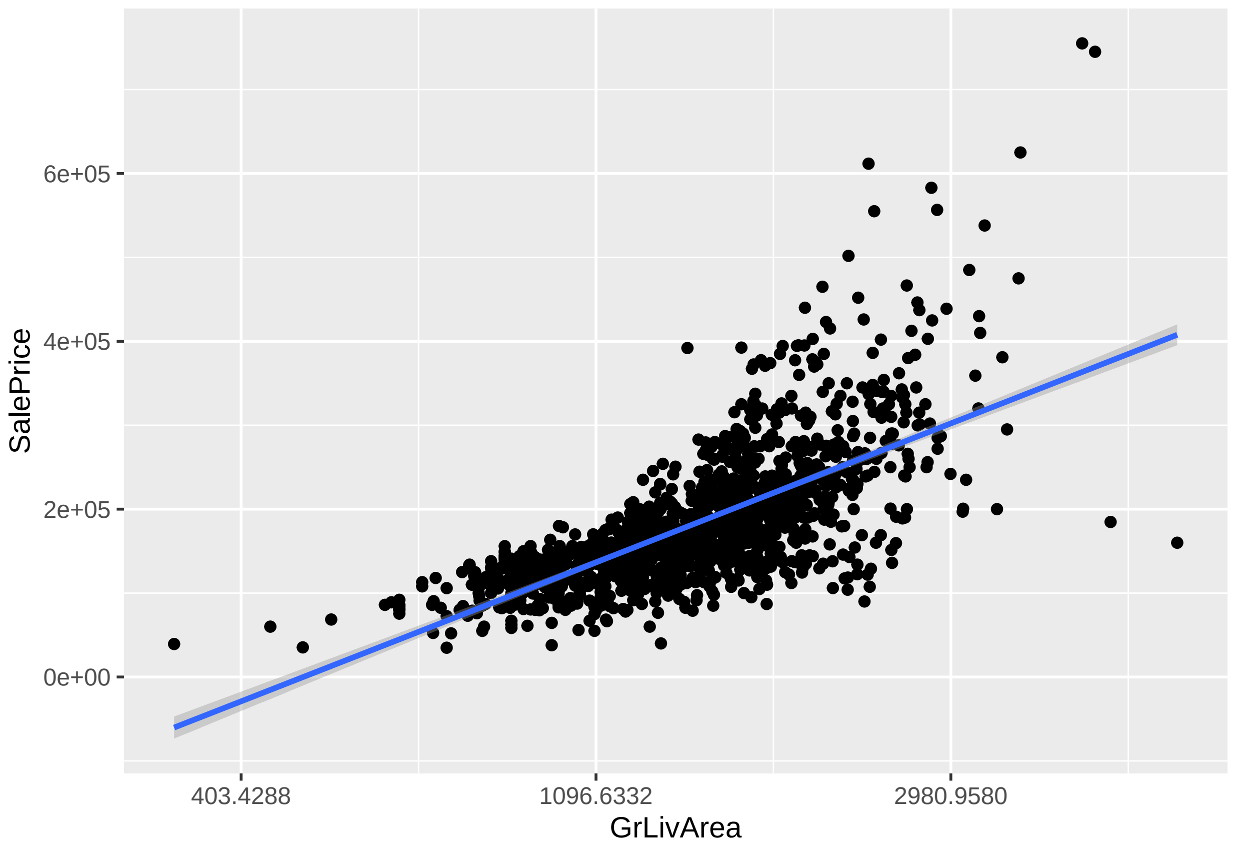
The SalePrice by LotArea scatterplot does show a positive linear relationship with the log applied to the LotArea. But visual inspection does show quite a cluster of instances, so even though the plot shows a linear relationship, this variable may not be the most valuable.

```r
ggplot(df_train, aes(x=LotArea, y=SalePrice)) +
  geom_point() +
  scale_x_continuous(trans="log") +
  geom_smooth(method=lm)
```



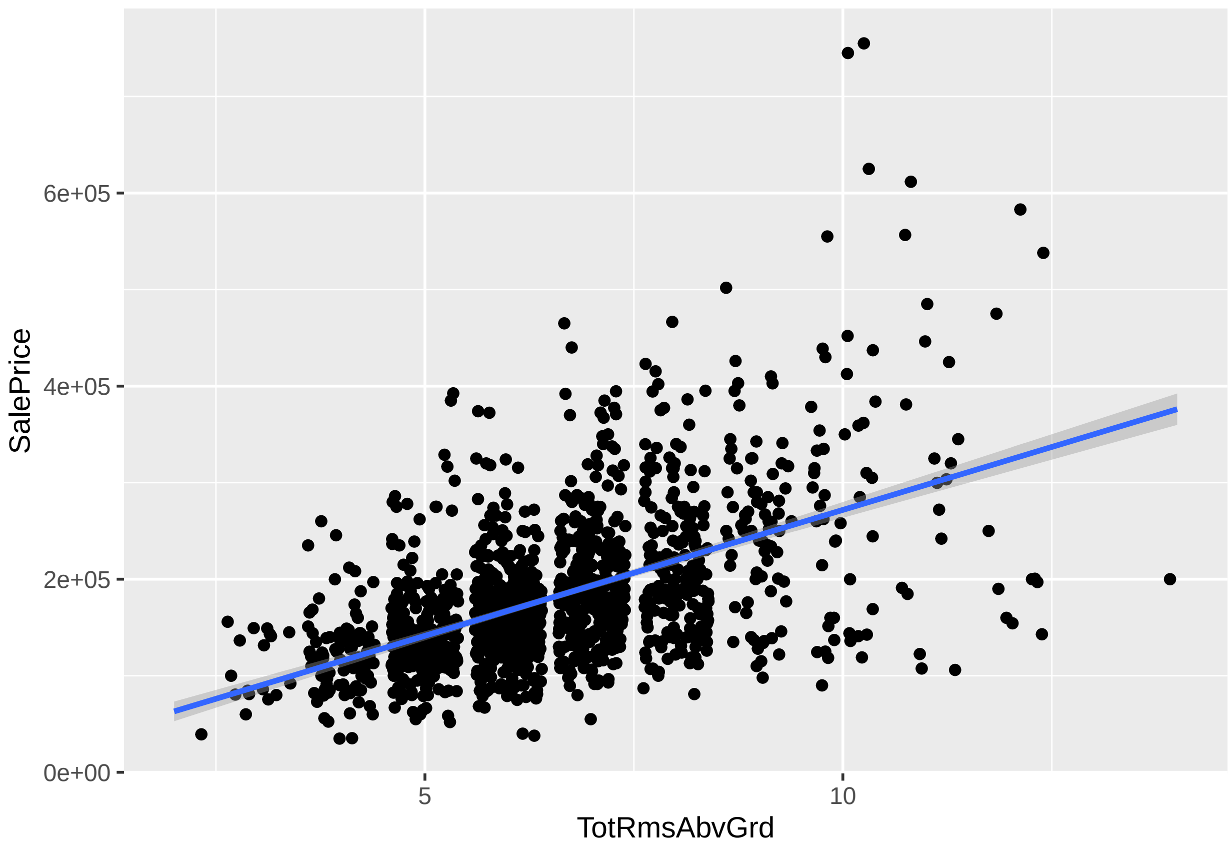
The SalePrice by GrLivArea scatterplot shows a very clear positive linear relationship between the values with the log applied to the GrLivArea variable. Using total above ground living area as a predictor of sale prices appears to be a good hypothesis.

```r
ggplot(df_train, aes(x=GrLivArea, y=SalePrice)) +
  geom_point() +
  scale_x_continuous(trans="log") +
  geom_smooth(method=lm)
```



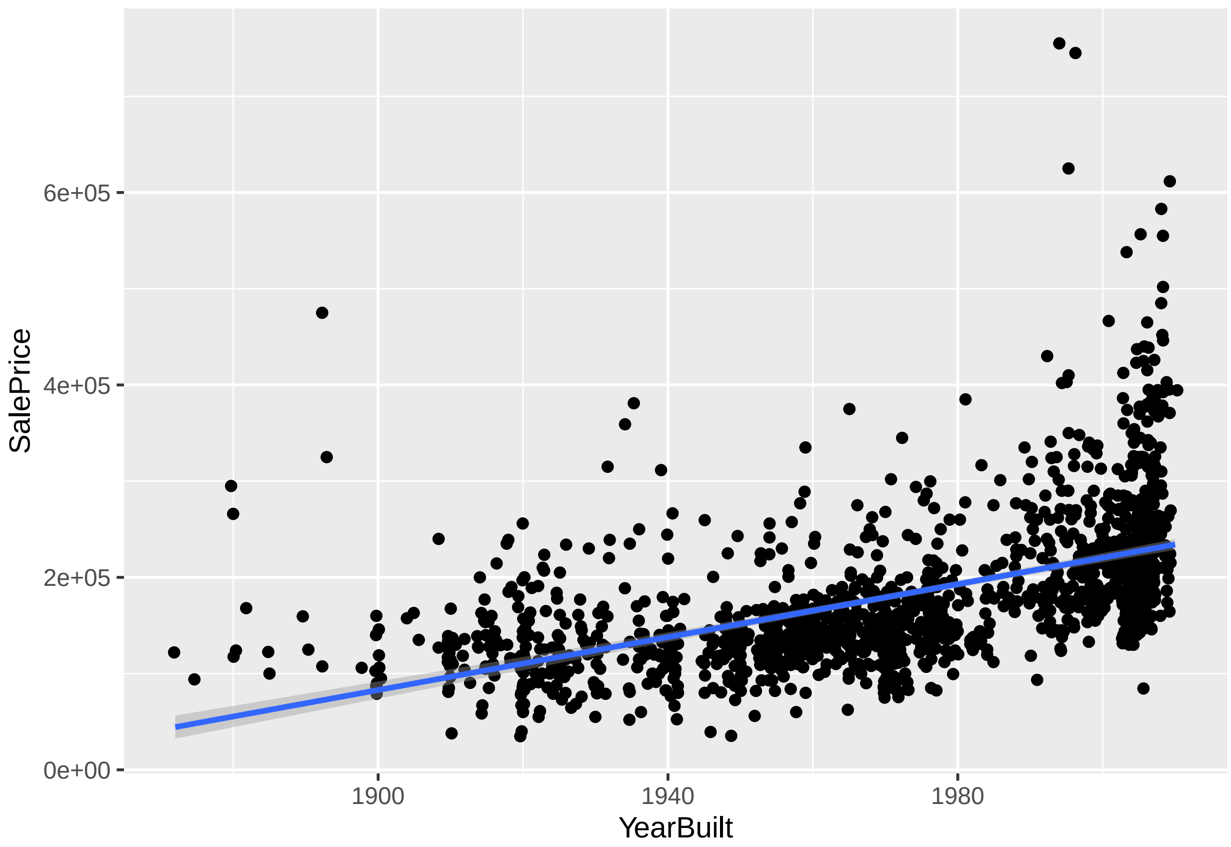
The SalePrice by TotRmsAbvGrd appears very similar to the above plot for GrLivArea with a positive linear relationship. I would expect the variables of TotRmsAbvGrd and GrLivArea to be highly correlated variables.

```r
ggplot(df_train, aes(x=TotRmsAbvGrd, y=SalePrice)) +
  geom_jitter() +
  geom_smooth(method=lm)
```



Finally, a scatterplot of SalePrice by YearBuilt show a slightly positive linear relationship with outliers more likely to appear above the linear regression line. Given the shape of the scatterplot, YearBuilt may be valuable in predicting SalePrice.

```r
ggplot(df_train, aes(x=YearBuilt, y=SalePrice)) +
  geom_jitter() +
  geom_smooth(method=lm)
```



Correlation matrix

Using 3 of the 4 selected variables above, a correlation matrix is derived with GrLivArea, LotArea, and TotRmsAbvGrd.

```
cols <- c("GrLivArea", "LotArea", "TotRmsAbvGrd")

cor.mat <- cor(df_train[cols])
cor.mat
```

```
##              GrLivArea   LotArea TotRmsAbvGrd
## GrLivArea    1.0000000 0.2631162    0.8254894
## LotArea      0.2631162 1.0000000    0.1900148
## TotRmsAbvGrd 0.8254894 0.1900148    1.0000000
```

Using Pearson's algorithm for correlation, the correlation between GrLivArea and LotArea is not equal to 0, and the 80 percent confidence interval is 0.2315997 to 0.2940809. This would appear to be a moderate correlation.

```
cor.test.1 <- cor.test(df_train$GrLivArea, df_train$LotArea, method="pearson", conf.level=0.8)
cor.test.1
```

```
##
##  Pearson's product-moment correlation
##
```

```
## data:  df_train$GrLivArea and df_train$LotArea
## t = 10.414, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.2315997 0.2940809
## sample estimates:
##       cor
## 0.2631162
```

Using Pearson’s algorithm for correlation, the correlation between GrLivArea and TotRmsAbvGrd is not equal to 0, and the 80 percent confidence interval is 0.8144931 to 0.8358928. As expected, this would appear to be a strong correlation.

```
cor.test.2 <- cor.test(df_train$GrLivArea, df_train$TotRmsAbvGrd, method="pearson", conf.level=0.8)
cor.test.2
```

```
##
##  Pearson's product-moment correlation
##
## data:  df_train$GrLivArea and df_train$TotRmsAbvGrd
## t = 55.846, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.8144931 0.8358928
## sample estimates:
##       cor
## 0.8254894
```

Using Pearson’s algorithm for correlation, the correlation between LotArea and TotRmsAbvGrd is not equal to 0, and the 80 percent confidence interval is 0.1574573 to 0.2221597. This would appear to be a moderate to weak correlation.

```
cor.test.3 <- cor.test(df_train$LotArea, df_train$TotRmsAbvGrd, method="pearson", conf.level=0.8)
cor.test.3
```

```
##
##  Pearson's product-moment correlation
##
## data:  df_train$LotArea and df_train$TotRmsAbvGrd
## t = 7.3901, df = 1458, p-value = 2.461e-13
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.1574573 0.2221597
## sample estimates:
##       cor
## 0.1900148
```

As the results of the 3 correlation tests above indicate, the p-value is quite small indicating correlation between each set of variables, rejecting the null hypothesis that true correlation is zero. Given the very small values for each p-value, one should not worry about familywise error.

Linear Algebra and Correlation

Invert your correlation matrix from above. (This is known as the precision matrix and contains variance inflation factors on the diagonal.) Multiply the correlation matrix by the precision matrix, and then multiply the precision matrix by the correlation matrix. Conduct LU decomposition on the matrix.

Create precision matrix by inverting correlation matrix.

```
# Invert your correlation matrix from above
inv.cor.mat <- solve(cor.mat)
inv.cor.mat
```

```
##               GrLivArea     LotArea TotRmsAbvGrd
## GrLivArea     3.2588901 -0.35926433  -2.62191358
## LotArea      -0.3592643  1.07706384   0.09191085
## TotRmsAbvGrd -2.6219136  0.09191085   3.14689738
```

Multiply the correlation matrix by the precision matrix. Apply the zapsmall to allow for interpretability. Results in the identity matrix.

```
# Multiply the correlation matrix by the precision matrix
cor.by.prec <- cor.mat %*% inv.cor.mat
#cor.by.prec

cor.by.prec <- zapsmall(cor.by.prec)
cor.by.prec
```

```
##              GrLivArea LotArea TotRmsAbvGrd
## GrLivArea            1       0            0
## LotArea              0       1            0
## TotRmsAbvGrd         0       0            1
```

Multiply the precision matrix by the correlation matrix, and again apply the zapsmall function. Again, results in identity matrix.

```
# multiply the precision matrix by the correlation matrix
prec.by.cor <- inv.cor.mat %*% cor.mat
#prec.by.cor

prec.by.cor <- zapsmall(prec.by.cor)
prec.by.cor
```

```
##              GrLivArea LotArea TotRmsAbvGrd
## GrLivArea            1       0            0
## LotArea              0       1            0
## TotRmsAbvGrd         0       0            1
```

Conduct LU decomposition on the matrix. Output the lower triangular matrix.

```
library(matrixcalc)
# Conduct LU decomposition on the matrix
lu.de <- lu.decomposition(cor.mat)

lower <- lu.de$L
lower
```

```
##           [,1]        [,2] [,3]
## [1,] 1.0000000  0.00000000    0
## [2,] 0.2631162  1.00000000    0
## [3,] 0.8254894 -0.02920681    1
```

Output the upper triangular matrix.

```
upper <- lu.de$U
upper
```

```
##      [,1]      [,2]        [,3]
## [1,]    1 0.2631162  0.82548937
## [2,]    0 0.9307699 -0.02718482
## [3,]    0 0.0000000  0.31777331
```

To confirm accuracy of LU decomposition, multiply to the lower and upper triangular matrices to return to initial correlation matrix.

```
res <- lower %*% upper
res
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.2631162 0.8254894
## [2,] 0.2631162 1.0000000 0.1900148
## [3,] 0.8254894 0.1900148 1.0000000
```

```
cor.mat
```

```
##              GrLivArea   LotArea TotRmsAbvGrd
## GrLivArea    1.0000000 0.2631162    0.8254894
## LotArea      0.2631162 1.0000000    0.1900148
## TotRmsAbvGrd 0.8254894 0.1900148    1.0000000
```

Calculus-Based Probability & Statistics

Many times, it makes sense to fit a closed form distribution to data. Select a variable in the Kaggle.com training dataset that is skewed to the right, shift it so that the minimum value is absolutely above zero if necessary. Then load the MASS package and run $fitdistr$ to fit an exponential probability density function. (See https://stat.ethz.ch/R-manual/R-devel/library/MASS/html/fitdistr.html). Find the optimal value of λ for this distribution, and then take 1000 samples from this exponential distribution using this value (e.g., $rexp(1000, \lambda)$). Plot a histogram and compare it with a histogram of your original variable. Using the exponential pdf, find the 5th and 95th percentiles using the cumulative distribution function (CDF). Also generate a 95% confidence interval from the empirical data, assuming normality. Finally, provide the empirical 5th percentile and 95th percentile of the data. Discuss.

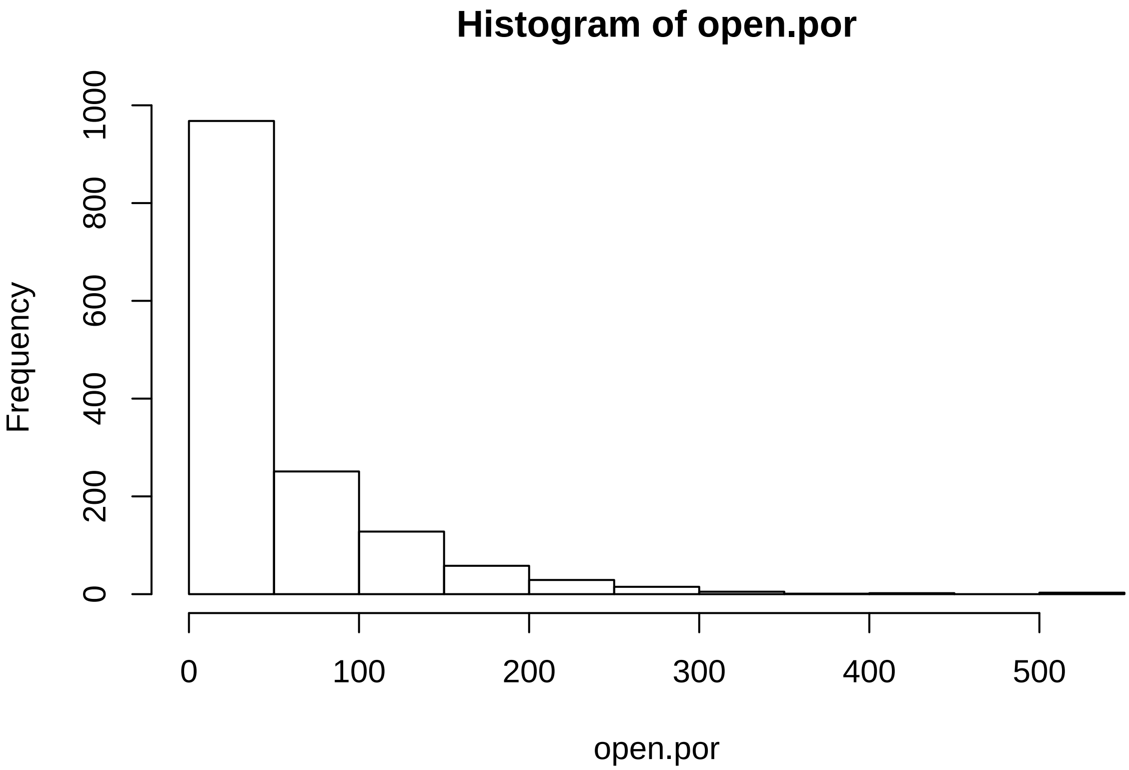
Select variable OpenPorchSF which is skewed to the right. The values are shifted by positive one to ensure that the minimum value is absolutely above zero. The shift of +1 results in the mean and median values being one greater than the initial data and no change to the standard deviation of the variable values.

The histogram of the shifted variable confirms the right skew. The summary of the shifted variable shows the minimum of 1 and the maximum of 548.

```
# Select a variable in the Kaggle.com training dataset that is skewed to the right
# OpenPorchSF

open.por <- df_train$OpenPorchSF
open.por <- open.por + 1

hist(open.por)
```



```
summary(open.por)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    1.00   26.00   47.66   69.00  548.00
```

Run function fitdistr from the MASS package to fit an exponential probability density function (PDF)

```
# Then load the MASS package and run fitdistr to fit an exponential probability density function (PDF)
library(MASS)
set.seed(1234)
exp.pdf <- fitdistr(open.por, densfun="exponential")
```

The optimal value of lambda for the distribution is the estimate attribute from the fitdistr response. The value is output below.

```
# Find the optimal value of lambda for this distribution
exp.pdf$estimate
```

```
##       rate
## 0.02098183
```

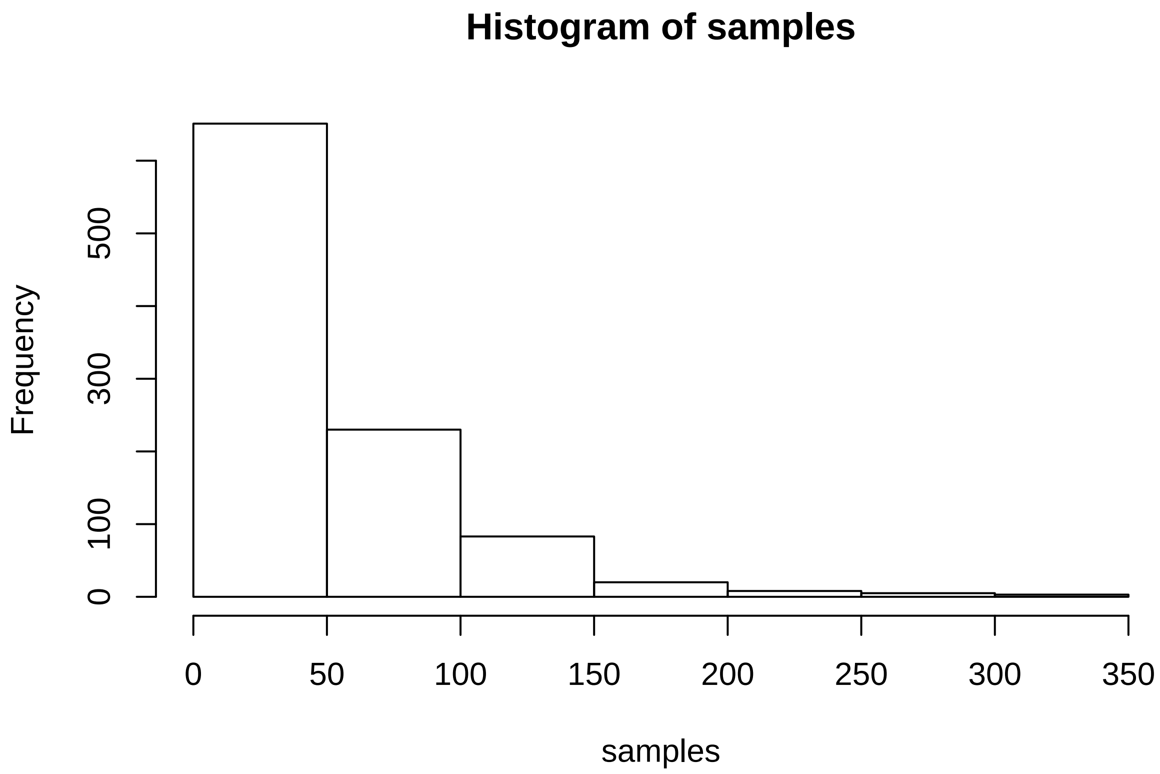
```
lambda <- exp.pdf$estimate
```

Generate 1000 samples using the lambda value.

```
# then take 1000 samples from this exponential distribution using this value
samples <- rexp(1000, lambda)
```

The below histogram based on the 1000 samples shows a decreased range of values across the x-axis along with a less concentrated count along the y-axis. Yes, the new histogram is still right skewed, but not to same the degree. From visual inspection, the second bucket of the below histogram is much closer to half of the first bucket as compared to the initial histogram. Overall the data is a bit more uniformly distributed, though not all completely uniform, nor normal, and the range of values has decreased by almost half.

```
# Plot a histogram and compare it with a histogram of your original variable.
hist(samples)
```



Given the lambda of the exponential PDF, the 5th percentile is 2.444652 and the 95th percentile is 142.7774.

```
# Using the exponential pdf, find the 5th and 95th percentiles using the cumulative distribution functi
qexp(.05, rate=lambda, lower.tail=T)
```

```
## [1] 2.444652
```

```
qexp(.95, rate=lambda, lower.tail=T)
```

```
## [1] 142.7774
```

Using the initial data, generating the 95% confidence interval assuming normality using the function qnorm along with the t.test function. For function qnorm the 95% confidence interval would be 44.2617 to 51.05885, quite close to result from t.test.

```
# Generate a 95% confidence interval from the empirical data, assuming normality.
mean <- mean(open.por)
sd <- sd(open.por)
error <- qnorm(0.975)*sd/sqrt(length(open.por))
left <- mean - error
right <- mean + error

left
```

```
## [1] 44.2617
```

```
right
```

```
## [1] 51.05885
```

```
t.test(open.por)
```

```
##
##  One Sample t-test
##
## data:  open.por
## t = 27.486, df = 1459, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  44.25888 51.06167
## sample estimates:
## mean of x
##  47.66027
```

The 5th percentile for the shifted empirical data is 1.00 and the 95th percentile is 176.05. Much different compared to the 95% confidence interval if the data were normal. With the exponential distribution, the skewed data provided realistic results while the normal data clearly did not match the skew of the original variable values.

```
# Provide the empirical 5th percentile and 95th percentile of the data.
quantile(open.por, c(0.05, 0.95))
```

```
##     5%    95%
##   1.00 176.05
```

Modeling

Build some type of multiple regression model and submit your model to the competition board. Provide your complete model summary and results with analysis. Report your Kaggle.com user name and score.

Approach

To find the best multiple regression model, I used an abbreviated backward elimination step approach. I started with every variable and upon the first model, removed all the variables with a significance greater than 0 and less than 0.001. Then, I continued to remove all variables that did not meet the same significance criteria until all independent variables had the same significance. But, for the final model below, I did add the variable Neighborhood back into the model as an attempt to improve my score, which did it. In the end, the below model sure isn't the simplest model, but fared well according to my standards and did pare down the overall list of variables.

To prepare the training and test data, I did convert all character variables to factor data type and set all missing numeric entries to zero, except for missing GarageYrBlt in which I just use the YearBuilt value as a replacement. As noted below, the test dataset had many more columns with missing data requiring data imputation

```
# First submitted to Kaggle
# Adjusted R-squared:  0.5018
# Naive approach based soly on the ground living area
#mod_lm <- lm(SalePrice ~ GrLivArea, data=df_train)

# Read in Train and Test again
df_train <- read_csv("KaggleCompetition/house-prices-advanced-regression-techniques/train.csv")
df_test <- read_csv("KaggleCompetition/house-prices-advanced-regression-techniques/test.csv")

# For Train
df_train$LotFrontage <- replace(df_train$LotFrontage, is.na(df_train$LotFrontage), 0)
df_train$MasVnrArea <- replace(df_train$MasVnrArea, is.na(df_train$MasVnrArea), 0)
df_train$GarageYrBlt <- replace(df_train$GarageYrBlt, is.na(df_train$GarageYrBlt), df_train$YearBuilt)

# For Test
df_test$LotFrontage <- replace(df_test$LotFrontage, is.na(df_test$LotFrontage), 0)
df_test$MasVnrArea <- replace(df_test$MasVnrArea, is.na(df_test$MasVnrArea), 0)
df_test$BsmtFinSF1 <- replace(df_test$BsmtFinSF1, is.na(df_test$BsmtFinSF1), 0)
df_test$BsmtFinSF2 <- replace(df_test$BsmtFinSF2, is.na(df_test$BsmtFinSF2), 0)
df_test$BsmtUnfSF <- replace(df_test$BsmtUnfSF, is.na(df_test$BsmtUnfSF), 0)
df_test$TotalBsmtSF <- replace(df_test$TotalBsmtSF, is.na(df_test$TotalBsmtSF), 0)
df_test$BsmtFullBath <- replace(df_test$BsmtFullBath, is.na(df_test$BsmtFullBath), 0)
df_test$BsmtHalfBath <- replace(df_test$BsmtHalfBath, is.na(df_test$BsmtHalfBath), 0)
df_test$GarageCars <- replace(df_test$GarageCars, is.na(df_test$GarageCars), 0)
df_test$GarageArea <- replace(df_test$GarageArea, is.na(df_test$GarageArea), 0)
df_test$GarageYrBlt <- replace(df_test$GarageYrBlt, is.na(df_test$GarageYrBlt), df_test$YearBuilt)

# Convert all character fields to factor
df_train[is.na(df_train)] <- "Not Found"
df_train <- df_train %>% mutate_if(is.character,as.factor)

df_test[is.na(df_test)] <- "Not Found"
df_test <- df_test %>% mutate_if(is.character,as.factor)
df_test$KitchenQual[df_test$KitchenQual == "Not Found"] <- "TA"
```

```r
#mod_lm.all <- lm(SalePrice ~ ., data=df_train)
# Adjusted R-squared:  0.9192

#mod_lm <- lm(SalePrice ~ LotArea + LandSlope + Neighborhood + Condition1 + Condition2 + OverallQual +
# Adjusted R-squared:  0.8914
#PST_052321_03.csv
# Kaggle score: 0.16268 (Best score)

# mod_lm <- lm(SalePrice ~ LotArea + LandSlope + Neighborhood + OverallQual + OverallCond + RoofMatl +
# Adjusted R-squared:  0.8802
```

The above commented out models were used during the development, but for the final model summary were not selected even though my best Kaggle score is noted in one of the models above.

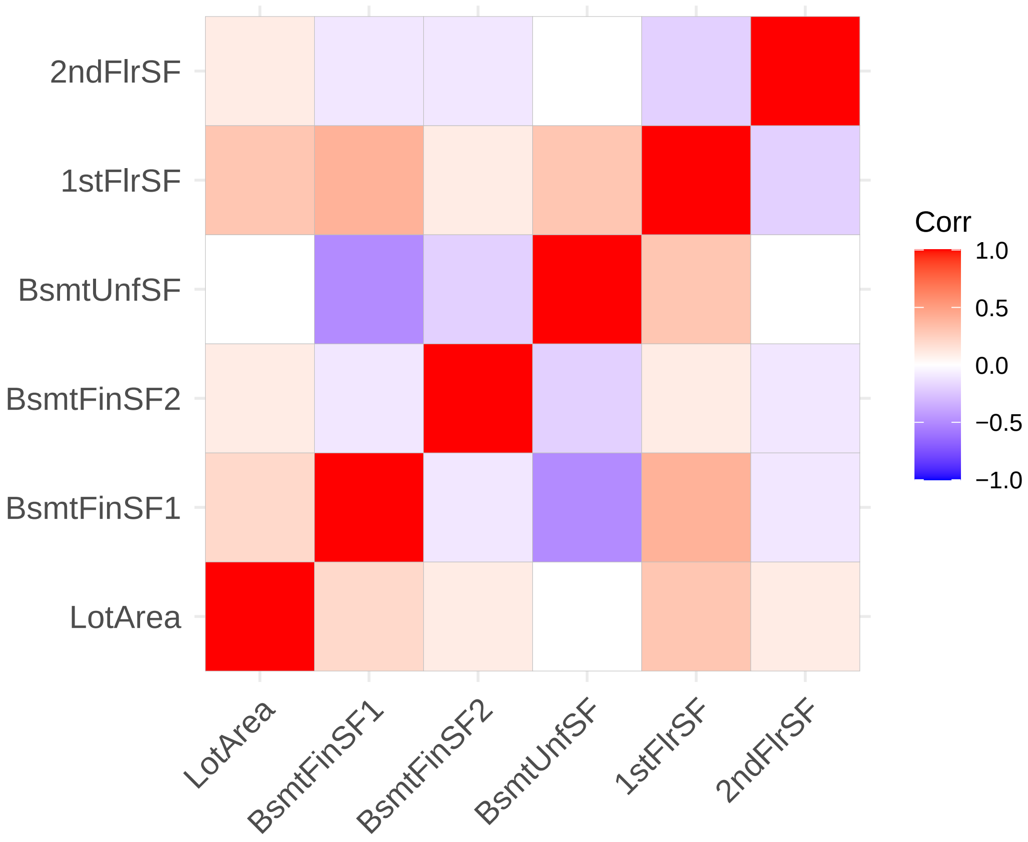
Based on the final model formula, I wanted to check the correlation between all the numeric values. Overall, the correlation values are near zero except for BsmtUnfSF and BsmtFinSF1, which score a correlation of -0.5. Based on the low correlation of all the variables below, they are included in the final formula.

```r
library(dplyr)
library(ggcorrplot)
df_train.abbr <- df_train %>% dplyr::select(LotArea, BsmtFinSF1, BsmtFinSF2, BsmtUnfSF, `1stFlrSF`, `2n

corr <- round(cor(df_train.abbr), 1)
head(corr)
```

```
##            LotArea BsmtFinSF1 BsmtFinSF2 BsmtUnfSF 1stFlrSF 2ndFlrSF
## LotArea        1.0        0.2        0.1       0.0      0.3      0.1
## BsmtFinSF1     0.2        1.0       -0.1      -0.5      0.4     -0.1
## BsmtFinSF2     0.1       -0.1        1.0      -0.2      0.1     -0.1
## BsmtUnfSF      0.0       -0.5       -0.2       1.0      0.3      0.0
## 1stFlrSF       0.3        0.4        0.1       0.3      1.0     -0.2
## 2ndFlrSF       0.1       -0.1       -0.1       0.0     -0.2      1.0
```

```r
ggcorrplot(corr)
```



The final linear regression model is generated and displayed in summary. The model contains both numeric and factor variables. As the model summary shows, the factor variables are converted into dummy variables by the lm function.

```
mod_lm <- lm(SalePrice ~ Neighborhood + LotArea + OverallQual + OverallCond + RoofMatl + ExterQual + Bs
# Adjusted R-squared:  0.8664

summary(mod_lm)
```

```
##
## Call:
## lm(formula = SalePrice ~ Neighborhood + LotArea + OverallQual +
##     OverallCond + RoofMatl + ExterQual + BsmtFinSF1 + BsmtFinSF2 +
##     BsmtUnfSF + `1stFlrSF` + `2ndFlrSF` + KitchenQual, data = df_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -362426  -12603    -143   12599  207530
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -5.524e+05  3.562e+04 -15.509  < 2e-16 ***
## NeighborhoodBlueste -1.816e+04  2.206e+04  -0.823 0.410471
## NeighborhoodBrDale  -3.018e+04  1.064e+04  -2.835 0.004642 **
## NeighborhoodBrkSide -1.715e+04  8.597e+03  -1.995 0.046247 *
## NeighborhoodClearCr -1.037e+04  9.791e+03  -1.059 0.289868
## NeighborhoodCollgCr  3.134e+03  7.570e+03   0.414 0.678951
## NeighborhoodCrawfor  2.663e+02  8.605e+03   0.031 0.975311
## NeighborhoodEdwards -2.491e+04  8.162e+03  -3.052 0.002312 **
## NeighborhoodGilbert  5.086e+03  8.071e+03   0.630 0.528674
## NeighborhoodIDOTRR  -2.940e+04  9.105e+03  -3.229 0.001273 **
## NeighborhoodMeadowV -2.652e+04  1.057e+04  -2.510 0.012184 *
## NeighborhoodMitchel -1.295e+04  8.619e+03  -1.502 0.133298
## NeighborhoodNAmes   -1.940e+04  7.858e+03  -2.468 0.013700 *
## NeighborhoodNoRidge  4.535e+04  8.794e+03   5.157 2.87e-07 ***
## NeighborhoodNPkVill -1.597e+04  1.234e+04  -1.294 0.195746
## NeighborhoodNridgHt  3.488e+04  8.100e+03   4.306 1.78e-05 ***
## NeighborhoodNWAmes  -1.744e+04  8.292e+03  -2.103 0.035598 *
## NeighborhoodOldTown -3.640e+04  8.169e+03  -4.456 9.01e-06 ***
## NeighborhoodSawyer  -1.750e+04  8.396e+03  -2.084 0.037313 *
## NeighborhoodSawyerW -7.431e+03  8.227e+03  -0.903 0.366575
## NeighborhoodSomerst  1.619e+04  7.818e+03   2.071 0.038571 *
## NeighborhoodStoneBr  4.255e+04  9.318e+03   4.566 5.39e-06 ***
## NeighborhoodSWISU   -3.194e+04  9.684e+03  -3.298 0.000997 ***
## NeighborhoodTimber   6.072e+03  8.807e+03   0.689 0.490675
## NeighborhoodVeenker  7.378e+03  1.157e+04   0.638 0.523874
## LotArea              5.186e-01  8.897e-02   5.829 6.88e-09 ***
## OverallQual          1.147e+04  1.056e+03  10.865  < 2e-16 ***
## OverallCond          5.397e+03  7.954e+02   6.785 1.70e-11 ***
## RoofMatlCompShg      5.904e+05  3.263e+04  18.094  < 2e-16 ***
## RoofMatlMembran      6.286e+05  4.464e+04  14.082  < 2e-16 ***
## RoofMatlMetal        6.190e+05  4.435e+04  13.955  < 2e-16 ***
## RoofMatlRoll         5.695e+05  4.365e+04  13.048  < 2e-16 ***
## RoofMatlTar&Grv      5.915e+05  3.376e+04  17.522  < 2e-16 ***
## RoofMatlWdShake      5.854e+05  3.532e+04  16.576  < 2e-16 ***
## RoofMatlWdShngl      6.660e+05  3.424e+04  19.451  < 2e-16 ***
## ExterQualFa         -3.249e+04  1.054e+04  -3.082 0.002097 **
## ExterQualGd         -3.577e+04  5.305e+03  -6.742 2.27e-11 ***
## ExterQualTA         -4.057e+04  5.897e+03  -6.880 8.98e-12 ***
## BsmtFinSF1           4.132e+01  3.657e+00  11.300  < 2e-16 ***
## BsmtFinSF2           2.494e+01  5.795e+00   4.303 1.80e-05 ***
## BsmtUnfSF            1.837e+01  3.588e+00   5.119 3.50e-07 ***
## `1stFlrSF`           5.766e+01  3.844e+00  15.000  < 2e-16 ***
## `2ndFlrSF`           5.276e+01  2.298e+00  22.957  < 2e-16 ***
## KitchenQualFa       -4.030e+04  6.836e+03  -5.896 4.66e-09 ***
## KitchenQualGd       -3.143e+04  3.964e+03  -7.929 4.44e-15 ***
## KitchenQualTA       -3.834e+04  4.448e+03  -8.620  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29030 on 1414 degrees of freedom
## Multiple R-squared:  0.8705, Adjusted R-squared:  0.8664
## F-statistic: 211.3 on 45 and 1414 DF,  p-value: < 2.2e-16
```

Assessing the residuals: With a goal of matching a normal distribution, the median isn't quite at 0, so the center of the distribution is off. The 1st and 3rd quartile are almost the same, so a good sign for the normal nature. The min and max values are not close, but given the large range, I do believe they are similar in magnitude.

Assessing the standard error and the corresponding t-value: Except for many of the Neighborhood dummy variables, the results show good t-values with absolute values of at least 5. The p-value reflects the same understanding as the t-value. The p-value of all the variables expect for some of the Neighborhood dummy variables indicate a high level of significance to the model.

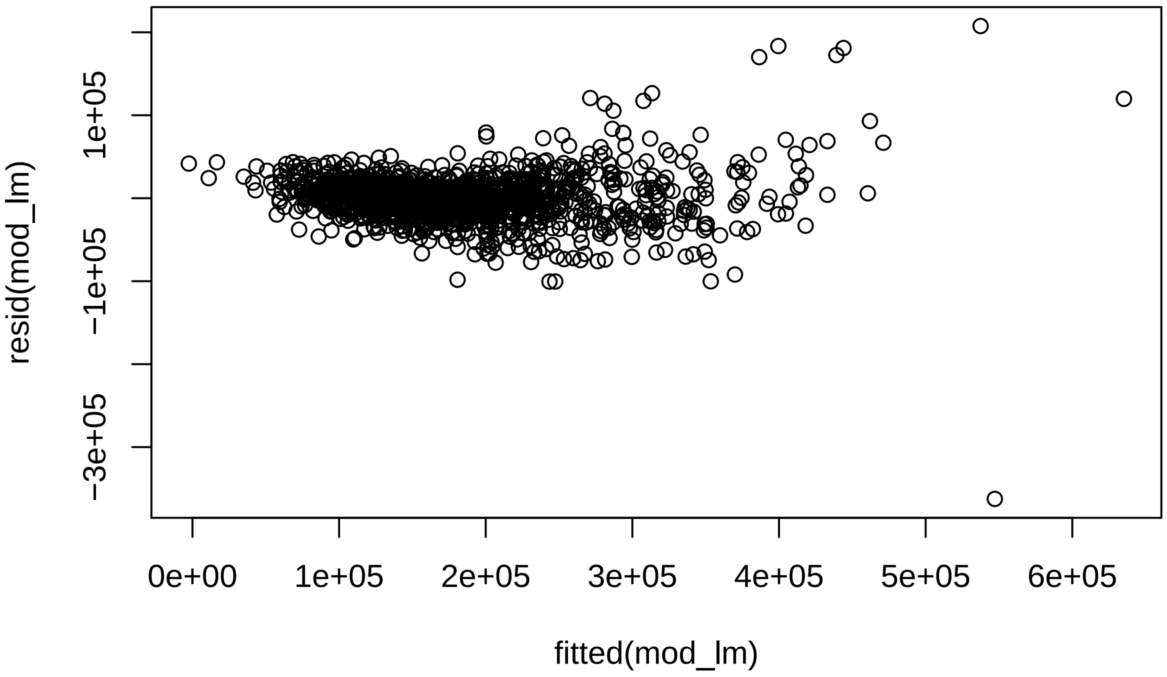
The Adjusted R-squared value of 0.8664 indicates the model can explain over 86% of the data's variation.

Just for the sake of completeness, I did attempt to use the stepAIC function, but the result was not better than the final model presented.

```
# DO NOT RUN, THIS HAS STEPAIC, NOT WORTH IT
mod_lm.all <- lm(SalePrice ~ ., data=df_train)
mod_lm.aic <- stepAIC(mod_lm.all, direction="both")
summary(mod_lm.aic)
```

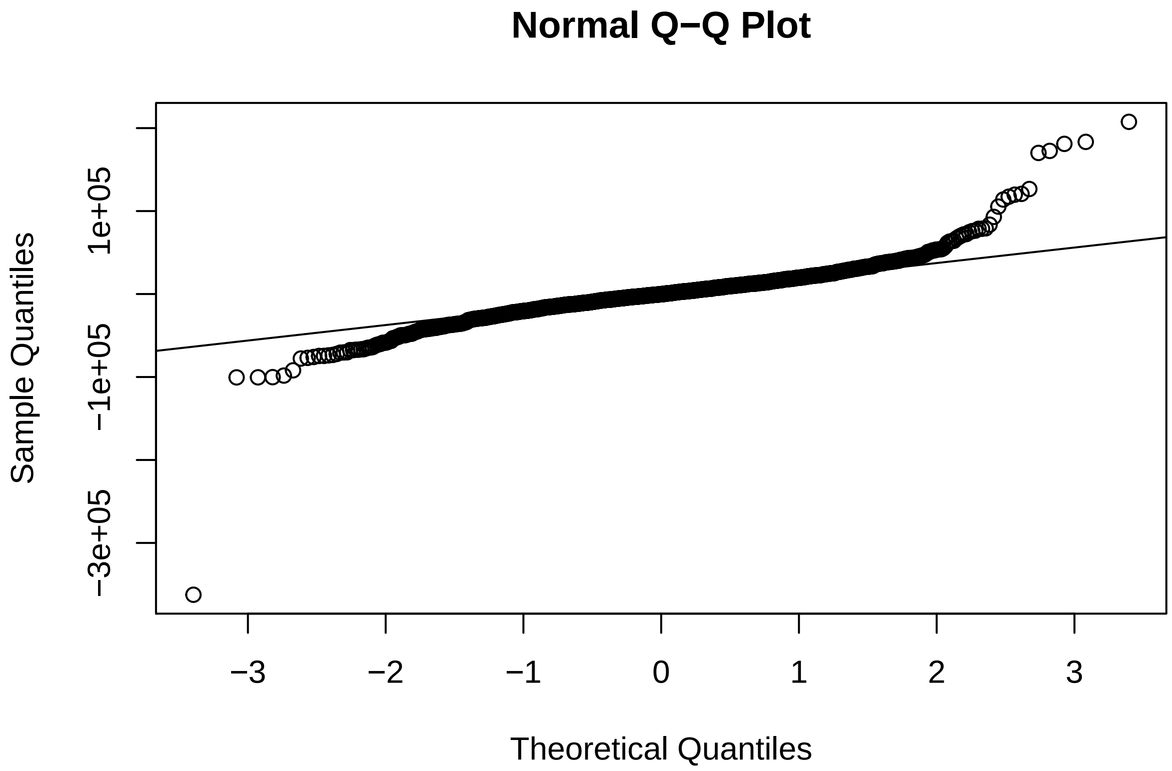
The residuals plot shows most of the points along the zero x-axis, a good sign.

```
plot(fitted(mod_lm),resid(mod_lm))
```



The normal Q-Q plot does indicate points diverging a bit at the two ends. Certainly, a better model exists, but the Normal Q-Q plot shows most of the points on or near the straight line.

```
qqnorm(resid(mod_lm))
qqline(resid(mod_lm))
```



Finally, the predictions are made with the test dataset based on the final regression model.

```
preds <- predict(mod_lm, newdata=df_test)
```

```
submission <- data.frame(Id=df_test$Id, SalePrice=preds)

submission %>% write_csv("PST_052321_03.csv")
```

kaggle.com username: philiptanofsky (link: https://www.kaggle.com/philiptanofsky)

Top score: 0.16268 (Above model score: 0.17720)