DATA 605 Final project

Vishal Arora

12/15/2019

Computational Mathematics

Solutions should be provided in a format that can be shared on R Pubs and Git hub and You are also expected to make a short presentation via YouTube and post that recording to the board.

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 ????????????(N+1)/2.

Solution:

Generate a random variable X

```
# set seed value
set.seed(1)
N <- 6
X <- runif(10000, min = 1, max = N)
```

Generate a random variable Y

```
# mean
mu <- (N+1)/2
Y <- rnorm(10000 , mean = mu)
```

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.

5 points

a. P(X>x | X>y) = 0.7875256

Solution:

```
# first calculate x and y
x <- median(X)
y <- summary(Y)[2][[1]]


#p(A|B) = P(AB)/P(B)
sum(X>x & X > y)/sum(X>y)
```

```
## [1] 0.7875256
```

The probability of X greater than median value of X given that X is greater than first quartile of y is 0.78.

b. P(X>x, Y>y) = 0.3754

Solution:

```r
#P(AB)
pab <- sum(X>x & Y>y)/length(X)
```

The probability of X greater than median value of X and Y is greater than first quartile of y is 0.3754.

c. P(Xy) = 0.2124744

Solution:

```r
#p(A|B) = P(AB)/P(B)
sum(X<x & X > y)/sum(X>y)
```

```
## [1] 0.2124744
```

The probability of X less than median value of X given that X is greater than first quartile of y is 0.2124744.

5 points.

Investigate whether P(X>x and Y>y)=P(X>x)P(Y>y) by building a table and evaluating the marginal and joint probabilities.

Answer:

```r
tab <- c(sum(X<x & Y < y),
       sum(X < x & Y == y),
       sum(X < x & Y > y))
tab <- rbind(tab,
              c(sum(X==x & Y < y),
       sum(X == x & Y == y),
       sum(X == x & Y > y))

             )
tab <- rbind(tab,
              c(sum(X>x & Y < y),
       sum(X > x & Y == y),
       sum(X > x & Y > y))
             )
tab <- cbind(tab, tab[,1] + tab[,2] + tab[,3])
tab <- rbind(tab, tab[1,] + tab[2,] + tab[3,])
colnames(tab) <- c("Y<y", "Y=y", "Y>y", "Total")
rownames(tab) <- c("X<x", "X=x", "X>x", "Total")
knitr::kable(tab)
```

	Y<y	Y=y	Y>y	Total
X<x	1254	0	3746	5000
X=x	0	0	0	0
X>x	1246	0	3754	5000
Total	2500	0	7500	10000

Joint and marginal probability table. Now test the condition

```r
# P(X>x and Y>y)
3754/10000
```

```
## [1] 0.3754
```

```r
#P(X>x)P(Y>y)
((5000)/10000)*(7500/10000)
```

```
## [1] 0.375
```

we can see that the condition holds since P(X>x and Y>y) = 0.3754 and P(X>x)P(Y>y) = 0.375 are approximately equal.

5 points. 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?

Solution:

Fisher's Exact Test

```r
fisher.test(table(X>x,Y>y))
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  table(X > x, Y > y)
## p-value = 0.8716
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.9202847 1.1052820
## sample estimates:
## odds ratio
##    1.00857
```

The p-value is greater than zero we don't reject the null hypothesis. Two events are independent.

The Chi Square Test

```r
chisq.test(table(X>x,Y>y))
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table(X > x, Y > y)
## X-squared = 0.026133, df = 1, p-value = 0.8716
```

The p-value is greeter than zero we don't reject the null hypothesis. Two events are independent.

Fisher's exact test the null of independence of rows and columns in a contingency table with fixed marginals.

Chi-squared test tests contingency table tests and goodness-of-fit tests.

Fisher's exact test is appropriate here. Since the contingency table are fixed here in the table.

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 . I want you to do the following.

Reading the data and construct train n test dataframes.

```r
train = read.csv("train.csv", header = TRUE)

test= read.csv("test.csv", header = TRUE)
```

Check data types and no of observations n columns using the str function.

```
str(train)
```

```
## 'data.frame':    1460 obs. of  81 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : Factor w/ 5 levels "C (all)","FV",..: 4 4 4 4 4 4 4 4 5 4 ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
##  $ Street       : Factor w/ 2 levels "Grvl","Pave": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Alley        : Factor w/ 2 levels "Grvl","Pave": NA NA NA NA NA NA NA NA NA NA ...
##  $ LotShape     : Factor w/ 4 levels "IR1","IR2","IR3",..: 4 4 1 1 1 1 4 1 4 4 ...
##  $ LandContour  : Factor w/ 4 levels "Bnk","HLS","Low",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Utilities    : Factor w/ 2 levels "AllPub","NoSeWa": 1 1 1 1 1 1 1 1 1 1 ...
##  $ LotConfig    : Factor w/ 5 levels "Corner","CulDSac",..: 5 3 5 1 3 5 5 1 5 1 ...
##  $ LandSlope    : Factor w/ 3 levels "Gtl","Mod","Sev": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Neighborhood : Factor w/ 25 levels "Blmngtn","Blueste",..: 6 25 6 7 14 12 21 17 18 4 ...
##  $ Condition1   : Factor w/ 9 levels "Artery","Feedr",..: 3 2 3 3 3 3 3 5 1 1 ...
##  $ Condition2   : Factor w/ 8 levels "Artery","Feedr",..: 3 3 3 3 3 3 3 3 3 1 ...
##  $ BldgType     : Factor w/ 5 levels "1Fam","2fmCon",..: 1 1 1 1 1 1 1 1 1 2 ...
##  $ HouseStyle   : Factor w/ 8 levels "1.5Fin","1.5Unf",..: 6 3 6 6 6 1 3 6 1 2 ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
##  $ RoofStyle    : Factor w/ 6 levels "Flat","Gable",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ RoofMatl     : Factor w/ 8 levels "ClyTile","CompShg",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Exterior1st  : Factor w/ 15 levels "AsbShng","AsphShn",..: 13 9 13 14 13 13 13 7 4 9 ...
##  $ Exterior2nd  : Factor w/ 16 levels "AsbShng","AsphShn",..: 14 9 14 16 14 14 14 7 16 9 ...
##  $ MasVnrType   : Factor w/ 4 levels "BrkCmn","BrkFace",..: 2 3 2 3 2 3 4 4 3 3 ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 4 3 4 3 4 4 4 ...
##  $ ExterCond    : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ Foundation   : Factor w/ 6 levels "BrkTil","CBlock",..: 3 2 3 1 3 6 3 2 1 1 ...
##  $ BsmtQual     : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 3 3 4 3 3 1 3 4 4 ...
##  $ BsmtCond     : Factor w/ 4 levels "Fa","Gd","Po",..: 4 4 4 2 4 4 4 4 4 4 ...
##  $ BsmtExposure : Factor w/ 4 levels "Av","Gd","Mn",..: 4 2 3 4 1 4 1 3 4 4 ...
##  $ BsmtFinType1 : Factor w/ 6 levels "ALQ","BLQ","GLQ",..: 3 1 3 1 3 3 3 1 6 3 ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : Factor w/ 6 levels "ALQ","BLQ","GLQ",..: 6 6 6 6 6 6 6 2 6 6 ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : Factor w/ 6 levels "Floor","GasA",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ HeatingQC    : Factor w/ 5 levels "Ex","Fa","Gd",..: 1 1 1 3 1 1 1 1 3 1 ...
##  $ CentralAir   : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",..: 5 5 5 5 5 5 5 5 2 5 ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
```

```
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 3 3 4 3 4 4 4 ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : Factor w/ 7 levels "Maj1","Maj2",..: 7 7 7 7 7 7 7 7 3 7 ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : Factor w/ 5 levels "Ex","Fa","Gd",..: NA 5 5 3 5 NA 3 5 5 5 ...
##  $ GarageType   : Factor w/ 6 levels "2Types","Attchd",..: 2 2 2 6 2 2 2 2 6 2 ...
##  $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
##  $ GarageFinish : Factor w/ 3 levels "Fin","RFn","Unf": 2 2 2 3 2 3 2 2 3 2 ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 2 3 ...
##  $ GarageCond   : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ PavedDrive   : Factor w/ 3 levels "N","P","Y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : Factor w/ 3 levels "Ex","Fa","Gd": NA NA NA NA NA NA NA NA NA NA ...
##  $ Fence        : Factor w/ 4 levels "GdPrv","GdWo",..: NA NA NA NA NA 3 NA NA NA NA ...
##  $ MiscFeature  : Factor w/ 4 levels "Gar2","Othr",..: NA NA NA NA NA 3 NA 3 NA NA ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
##  $ SaleType     : Factor w/ 9 levels "COD","Con","ConLD",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ SaleCondition: Factor w/ 6 levels "Abnorml","AdjLand",..: 5 5 5 1 5 5 5 5 1 5 ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...
```

checking the data for missing values in columns, using the sapply function.

```r
colSums(sapply(train, is.na))
```

```
##            Id    MSSubClass      MSZoning   LotFrontage       LotArea 
##             0             0             0           259             0 
##        Street         Alley      LotShape   LandContour     Utilities 
##             0          1369             0             0             0 
##     LotConfig     LandSlope  Neighborhood    Condition1    Condition2 
##             0             0             0             0             0 
##      BldgType    HouseStyle   OverallQual   OverallCond     YearBuilt 
##             0             0             0             0             0 
##  YearRemodAdd     RoofStyle      RoofMatl   Exterior1st   Exterior2nd 
##             0             0             0             0             0 
##    MasVnrType    MasVnrArea     ExterQual     ExterCond    Foundation 
##             8             8             0             0             0 
##      BsmtQual      BsmtCond  BsmtExposure  BsmtFinType1    BsmtFinSF1 
##            37            37            38            37             0 
##  BsmtFinType2    BsmtFinSF2     BsmtUnfSF   TotalBsmtSF       Heating 
##            38             0             0             0             0 
##     HeatingQC    CentralAir    Electrical     X1stFlrSF     X2ndFlrSF 
##             0             0             1             0             0 
##  LowQualFinSF     GrLivArea  BsmtFullBath  BsmtHalfBath      FullBath 
```

```
##            0            0            0            0            0
##     HalfBath BedroomAbvGr KitchenAbvGr  KitchenQual TotRmsAbvGrd
##            0            0            0            0            0
##   Functional   Fireplaces  FireplaceQu   GarageType  GarageYrBlt
##            0            0          690           81           81
##  GarageFinish   GarageCars   GarageArea   GarageQual   GarageCond
##           81            0            0           81           81
##   PavedDrive   WoodDeckSF  OpenPorchSF EnclosedPorch   X3SsnPorch
##            0            0            0            0            0
##  ScreenPorch     PoolArea       PoolQC        Fence  MiscFeature
##            0            0         1453         1179         1406
##      MiscVal       MoSold       YrSold     SaleType SaleCondition
##            0            0            0            0            0
##    SalePrice
##            0
```

As per the missing value function above removing Right off the bat, MiscFeature, PoolQC, and Alley columns , as these columns have more than 50% data missing.

5 points.

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 a 80% confidence interval. Discuss the meaning of your analysis. Would you be worried about familywise error? Why or why not?

Rearranging the data and removing the id column from the data frame as it is not required, then using the summary function to have statistics about each individual column.

```r
nums <- unlist(lapply(train, is.numeric))
trainb<-train[ , nums]
trainc<-subset(trainb, select=-c(Id))
summary(trainc)
```

```
##    MSSubClass     LotFrontage        LotArea        OverallQual
##  Min.   : 20.0   Min.   : 21.00   Min.   :  1300   Min.   : 1.000
##  1st Qu.: 20.0   1st Qu.: 59.00   1st Qu.:  7554   1st Qu.: 5.000
##  Median : 50.0   Median : 69.00   Median :  9478   Median : 6.000
##  Mean   : 56.9   Mean   : 70.05   Mean   : 10517   Mean   : 6.099
##  3rd Qu.: 70.0   3rd Qu.: 80.00   3rd Qu.: 11602   3rd Qu.: 7.000
##  Max.   :190.0   Max.   :313.00   Max.   :215245   Max.   :10.000
##                  NA's   :259
##   OverallCond      YearBuilt     YearRemodAdd    MasVnrArea
##  Min.   :1.000   Min.   :1872   Min.   :1950   Min.   :   0.0
##  1st Qu.:5.000   1st Qu.:1954   1st Qu.:1967   1st Qu.:   0.0
##  Median :5.000   Median :1973   Median :1994   Median :   0.0
##  Mean   :5.575   Mean   :1971   Mean   :1985   Mean   : 103.7
##  3rd Qu.:6.000   3rd Qu.:2000   3rd Qu.:2004   3rd Qu.: 166.0
##  Max.   :9.000   Max.   :2010   Max.   :2010   Max.   :1600.0
##                                                NA's   :8
##    BsmtFinSF1       BsmtFinSF2        BsmtUnfSF       TotalBsmtSF
##  Min.   :   0.0   Min.   :   0.00   Min.   :   0.0   Min.   :   0.0
##  1st Qu.:   0.0   1st Qu.:   0.00   1st Qu.: 223.0   1st Qu.: 795.8
##  Median : 383.5   Median :   0.00   Median : 477.5   Median : 991.5
##  Mean   : 443.6   Mean   :  46.55   Mean   : 567.2   Mean   :1057.4
```

```
##  3rd Qu.: 712.2   3rd Qu.:   0.00   3rd Qu.: 808.0   3rd Qu.:1298.2
##  Max.   :5644.0   Max.   :1474.00   Max.   :2336.0   Max.   :6110.0
##
##    X1stFlrSF      X2ndFlrSF     LowQualFinSF       GrLivArea
##  Min.   : 334   Min.   :   0   Min.   :  0.000   Min.   : 334
##  1st Qu.: 882   1st Qu.:   0   1st Qu.:  0.000   1st Qu.:1130
##  Median :1087   Median :   0   Median :  0.000   Median :1464
##  Mean   :1163   Mean   : 347   Mean   :  5.845   Mean   :1515
##  3rd Qu.:1391   3rd Qu.: 728   3rd Qu.:  0.000   3rd Qu.:1777
##  Max.   :4692   Max.   :2065   Max.   :572.000   Max.   :5642
##
##   BsmtFullBath     BsmtHalfBath       FullBath        HalfBath
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.0000
##  Median :0.0000   Median :0.00000   Median :2.000   Median :0.0000
##  Mean   :0.4253   Mean   :0.05753   Mean   :1.565   Mean   :0.3829
##  3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:2.000   3rd Qu.:1.0000
##  Max.   :3.0000   Max.   :2.00000   Max.   :3.000   Max.   :2.0000
##
##   BedroomAbvGr    KitchenAbvGr    TotRmsAbvGrd     Fireplaces
##  Min.   :0.000   Min.   :0.000   Min.   : 2.000   Min.   :0.000
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.: 5.000   1st Qu.:0.000
##  Median :3.000   Median :1.000   Median : 6.000   Median :1.000
##  Mean   :2.866   Mean   :1.047   Mean   : 6.518   Mean   :0.613
##  3rd Qu.:3.000   3rd Qu.:1.000   3rd Qu.: 7.000   3rd Qu.:1.000
##  Max.   :8.000   Max.   :3.000   Max.   :14.000   Max.   :3.000
##
##   GarageYrBlt     GarageCars      GarageArea       WoodDeckSF
##  Min.   :1900   Min.   :0.000   Min.   :   0.0   Min.   :  0.00
##  1st Qu.:1961   1st Qu.:1.000   1st Qu.: 334.5   1st Qu.:  0.00
##  Median :1980   Median :2.000   Median : 480.0   Median :  0.00
##  Mean   :1979   Mean   :1.767   Mean   : 473.0   Mean   : 94.24
##  3rd Qu.:2002   3rd Qu.:2.000   3rd Qu.: 576.0   3rd Qu.:168.00
##  Max.   :2010   Max.   :4.000   Max.   :1418.0   Max.   :857.00
##  NA's   :81
##   OpenPorchSF     EnclosedPorch      X3SsnPorch      ScreenPorch
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00   1st Qu.:  0.00
##  Median : 25.00   Median :  0.00   Median :  0.00   Median :  0.00
##  Mean   : 46.66   Mean   : 21.95   Mean   :  3.41   Mean   : 15.06
##  3rd Qu.: 68.00   3rd Qu.:  0.00   3rd Qu.:  0.00   3rd Qu.:  0.00
##  Max.   :547.00   Max.   :552.00   Max.   :508.00   Max.   :480.00
##
##     PoolArea          MiscVal             MoSold           YrSold
##  Min.   :  0.000   Min.   :    0.00   Min.   : 1.000   Min.   :2006
##  1st Qu.:  0.000   1st Qu.:    0.00   1st Qu.: 5.000   1st Qu.:2007
##  Median :  0.000   Median :    0.00   Median : 6.000   Median :2008
##  Mean   :  2.759   Mean   :   43.49   Mean   : 6.322   Mean   :2008
##  3rd Qu.:  0.000   3rd Qu.:    0.00   3rd Qu.: 8.000   3rd Qu.:2009
##  Max.   :738.000   Max.   :15500.00   Max.   :12.000   Max.   :2010
##
##    SalePrice
##  Min.   : 34900
##  1st Qu.:129975
```

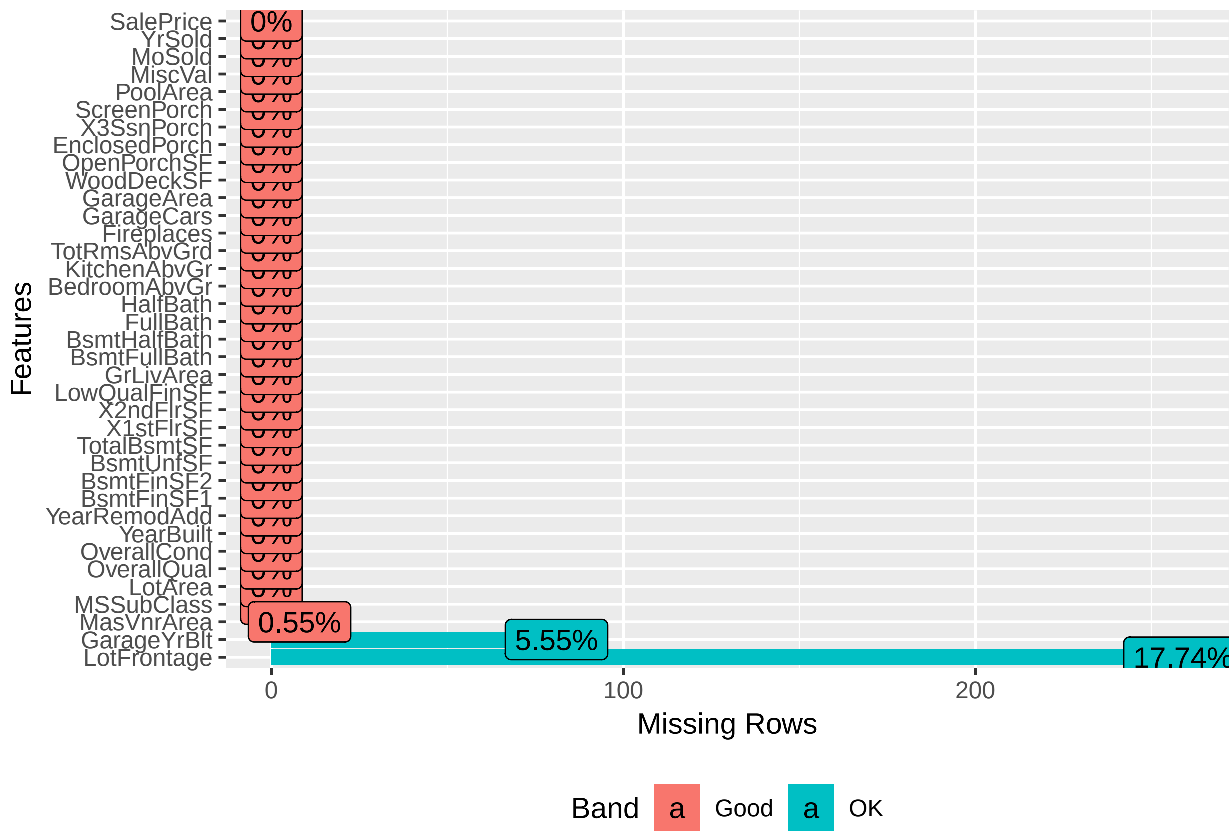
```
##  Median :163000
##  Mean   :180921
##  3rd Qu.:214000
##  Max.   :755000
##
```

Using plot_missing function from DataExplorer package to see the missing values for each individual columns and what is the missing data profile.

```
library(DataExplorer)
```

```
## Warning: package 'DataExplorer' was built under R version 3.5.3
```

```
plot_missing(trainc)[1]
```

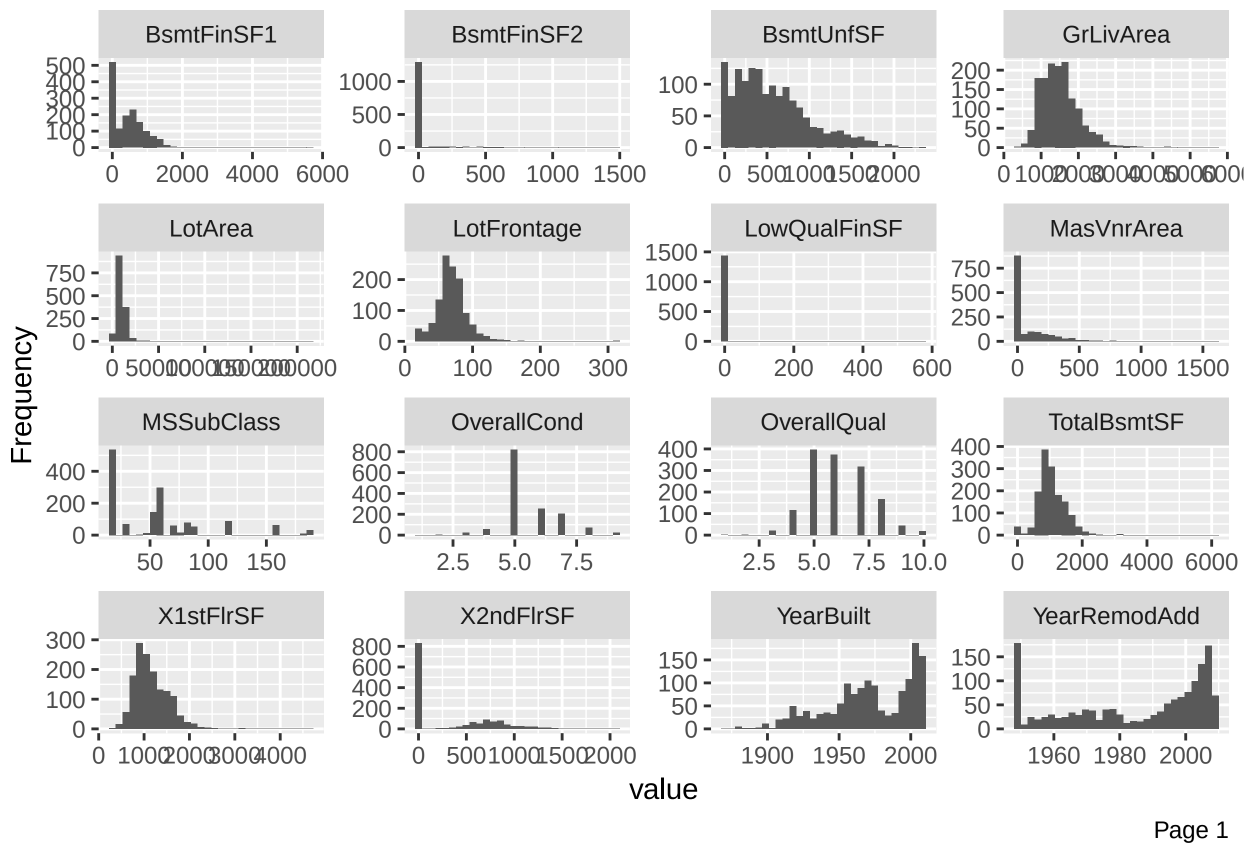



```
## $data
##          feature num_missing pct_missing Band
##  1:   MSSubClass           0 0.000000000 Good
##  2:  LotFrontage         259 0.177397260   OK
##  3:      LotArea           0 0.000000000 Good
##  4:  OverallQual           0 0.000000000 Good
##  5:  OverallCond           0 0.000000000 Good
##  6:    YearBuilt           0 0.000000000 Good
##  7: YearRemodAdd           0 0.000000000 Good
##  8:   MasVnrArea           8 0.005479452 Good
##  9:   BsmtFinSF1           0 0.000000000 Good
## 10:   BsmtFinSF2           0 0.000000000 Good
## 11:    BsmtUnfSF           0 0.000000000 Good
```

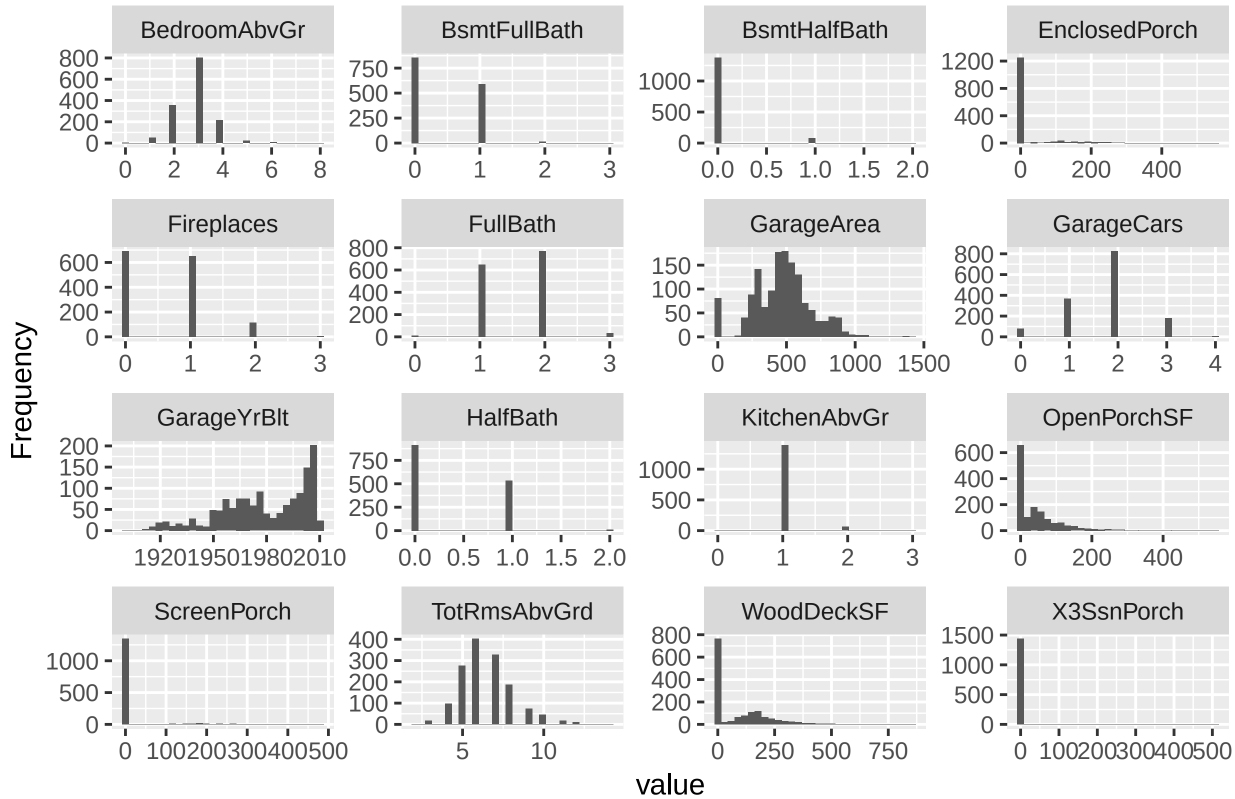
```
## 12:   TotalBsmtSF           0 0.000000000 Good
## 13:     X1stFlrSF           0 0.000000000 Good
## 14:     X2ndFlrSF           0 0.000000000 Good
## 15:  LowQualFinSF           0 0.000000000 Good
## 16:     GrLivArea           0 0.000000000 Good
## 17:  BsmtFullBath           0 0.000000000 Good
## 18:  BsmtHalfBath           0 0.000000000 Good
## 19:      FullBath           0 0.000000000 Good
## 20:      HalfBath           0 0.000000000 Good
## 21:  BedroomAbvGr           0 0.000000000 Good
## 22:  KitchenAbvGr           0 0.000000000 Good
## 23:  TotRmsAbvGrd           0 0.000000000 Good
## 24:    Fireplaces           0 0.000000000 Good
## 25:   GarageYrBlt          81 0.055479452   OK
## 26:    GarageCars           0 0.000000000 Good
## 27:    GarageArea           0 0.000000000 Good
## 28:    WoodDeckSF           0 0.000000000 Good
## 29:   OpenPorchSF           0 0.000000000 Good
## 30: EnclosedPorch           0 0.000000000 Good
## 31:    X3SsnPorch           0 0.000000000 Good
## 32:   ScreenPorch           0 0.000000000 Good
## 33:      PoolArea           0 0.000000000 Good
## 34:       MiscVal           0 0.000000000 Good
## 35:        MoSold           0 0.000000000 Good
## 36:        YrSold           0 0.000000000 Good
## 37:     SalePrice           0 0.000000000 Good
##           feature num_missing pct_missing Band
```

Out of the selected numerical variables, they all have less than 20% missing data. we will try to impute the missing values with the mean or mode. While there are not many interesting insights from plot_missing , below is the output from plot_histogram.

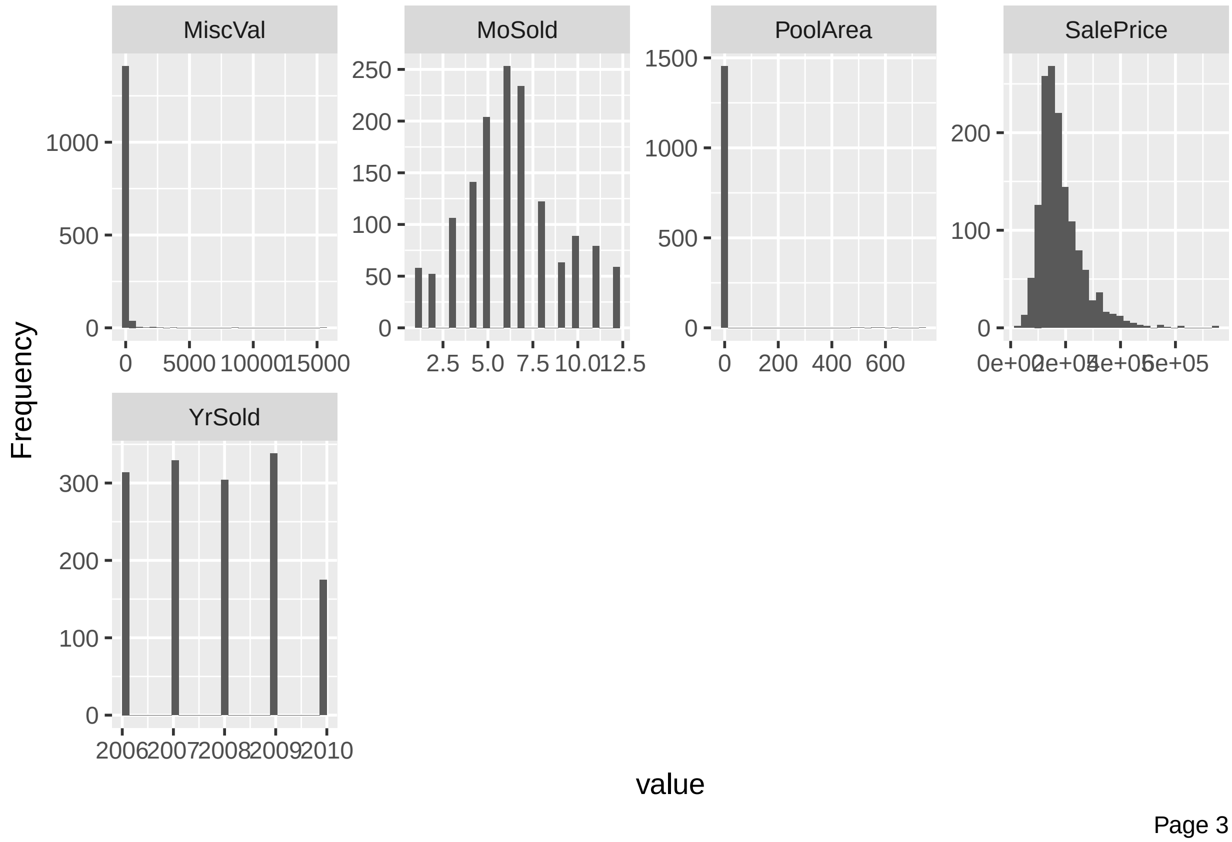
```
plot_histogram(trainc)
```



BsmtFinSF1
BsmtFinSF2
BsmtUnfSF
GrLivArea
LotArea
LotFrontage
LowQualFinSF
MasVnrArea
MSSubClass
OverallCond
OverallQual
TotalBsmtSF
X1stFlrSF
X2ndFlrSF
YearBuilt
YearRemodAdd
Frequency
value
Page 1


BedroomAbvGr
BsmtFullBath
BsmtHalfBath
EnclosedPorch
Fireplaces
FullBath
GarageArea
GarageCars
GarageYrBlt
HalfBath
KitchenAbvGr
OpenPorchSF
ScreenPorch
TotRmsAbvGrd
WoodDeckSF
X3SsnPorch
Frequency
value

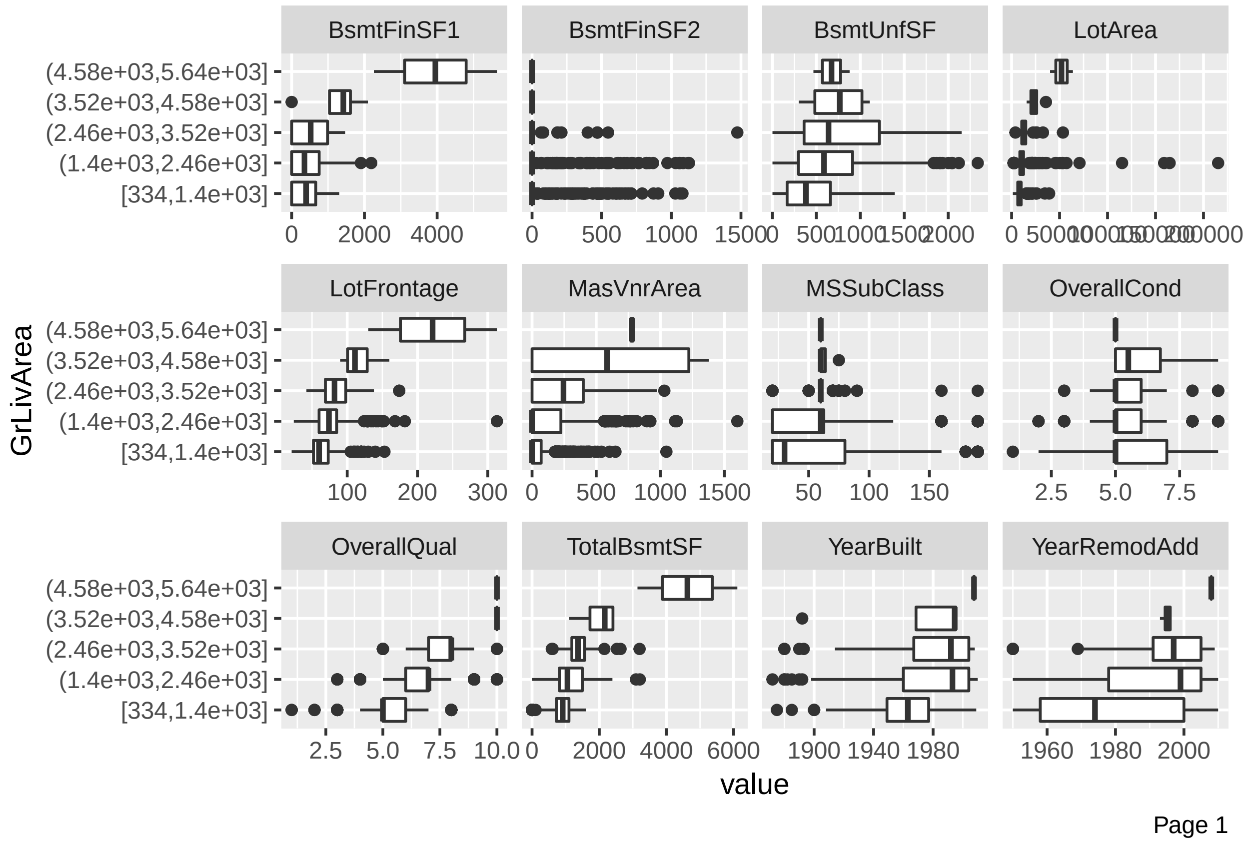
Page 2



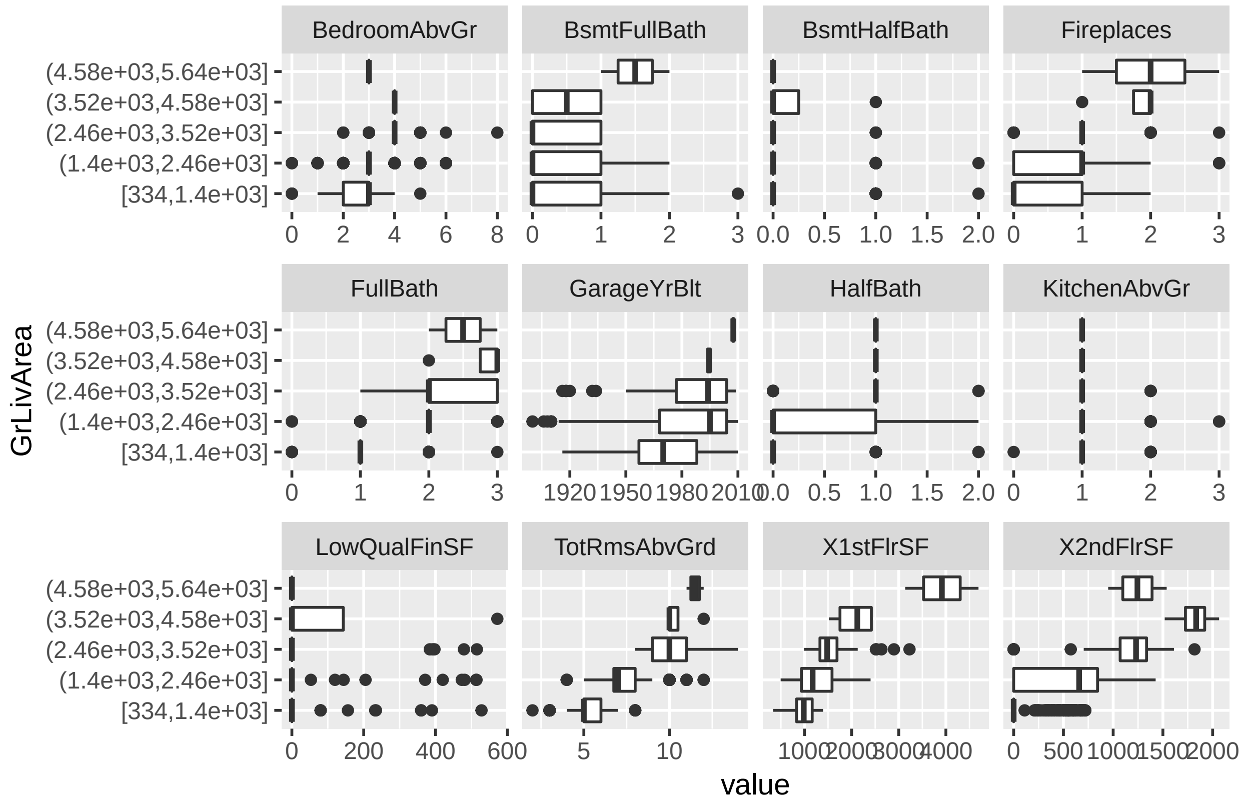
At this point, we have much better understanding of the data distribution.Now assume we are interested in GrLivArea, and would like to build a model to predict it. Let's plot it against all other variables.

```
plot_boxplot(trainc, by = "GrLivArea")
```

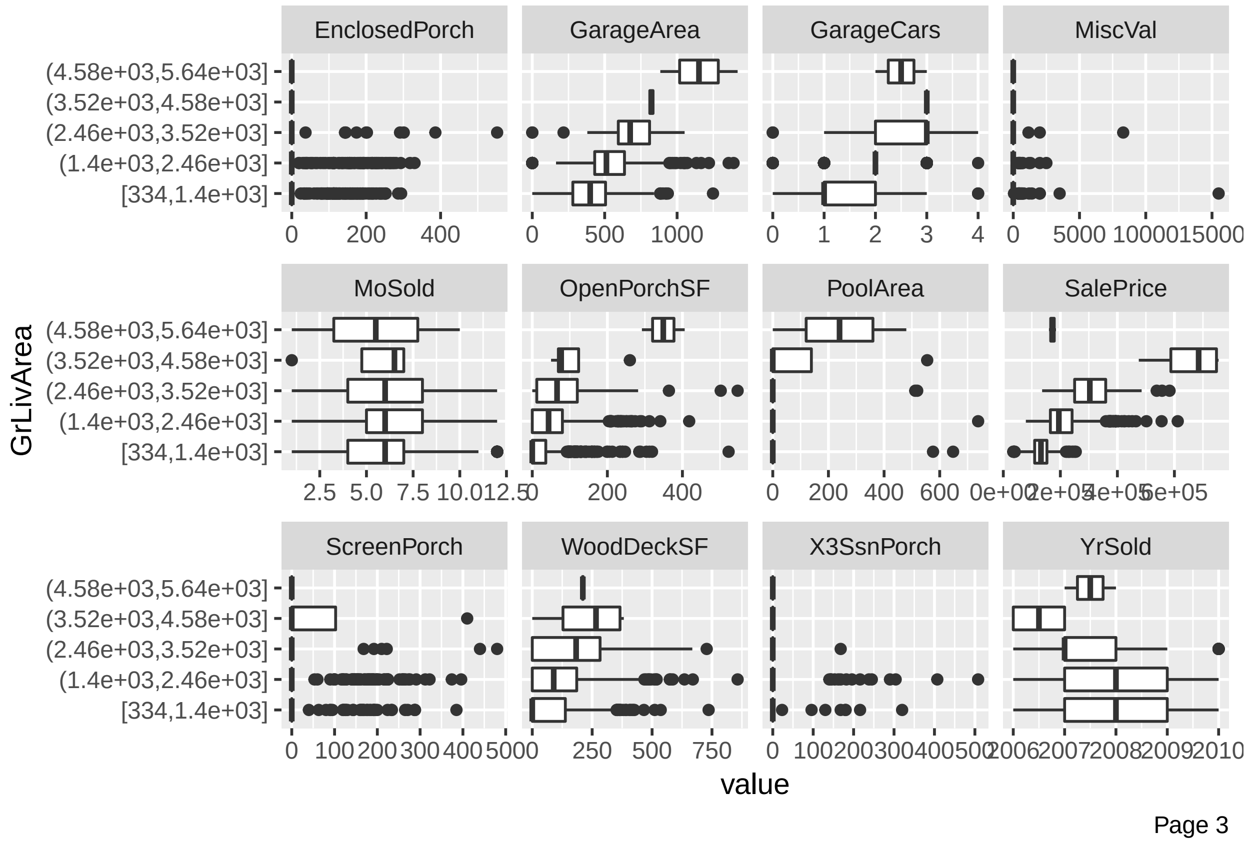
```
## Warning: Removed 267 rows containing non-finite values (stat_boxplot).
```




```
## Warning: Removed 81 rows containing non-finite values (stat_boxplot).
```


BedroomAbvGr
BsmtFullBath
BsmtHalfBath
Fireplaces
FullBath
GarageYrBlt
HalfBath
KitchenAbvGr
LowQualFinSF
TotRmsAbvGrd
X1stFlrSF
X2ndFlrSF
GrLivArea
(4.58e+03,5.64e+03]
(3.52e+03,4.58e+03]
(2.46e+03,3.52e+03]
(1.4e+03,2.46e+03]
[334,1.4e+03]
value

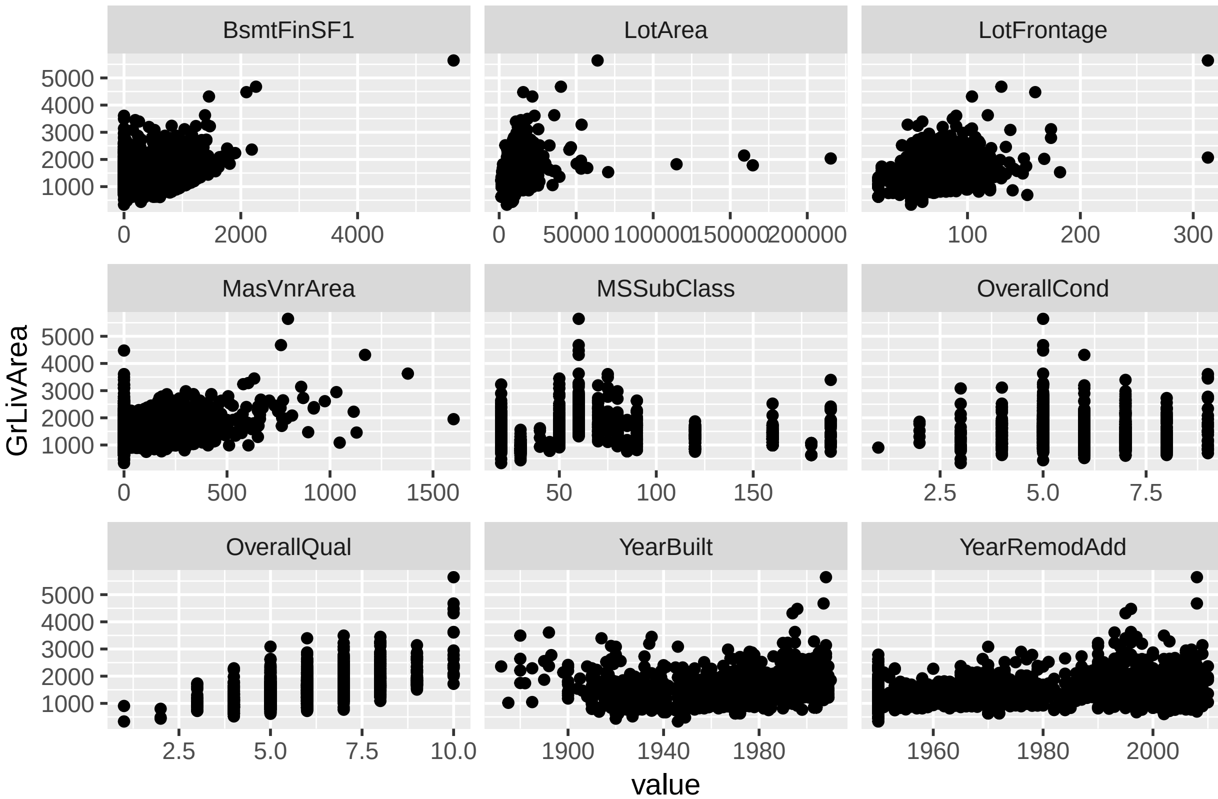
Page 2



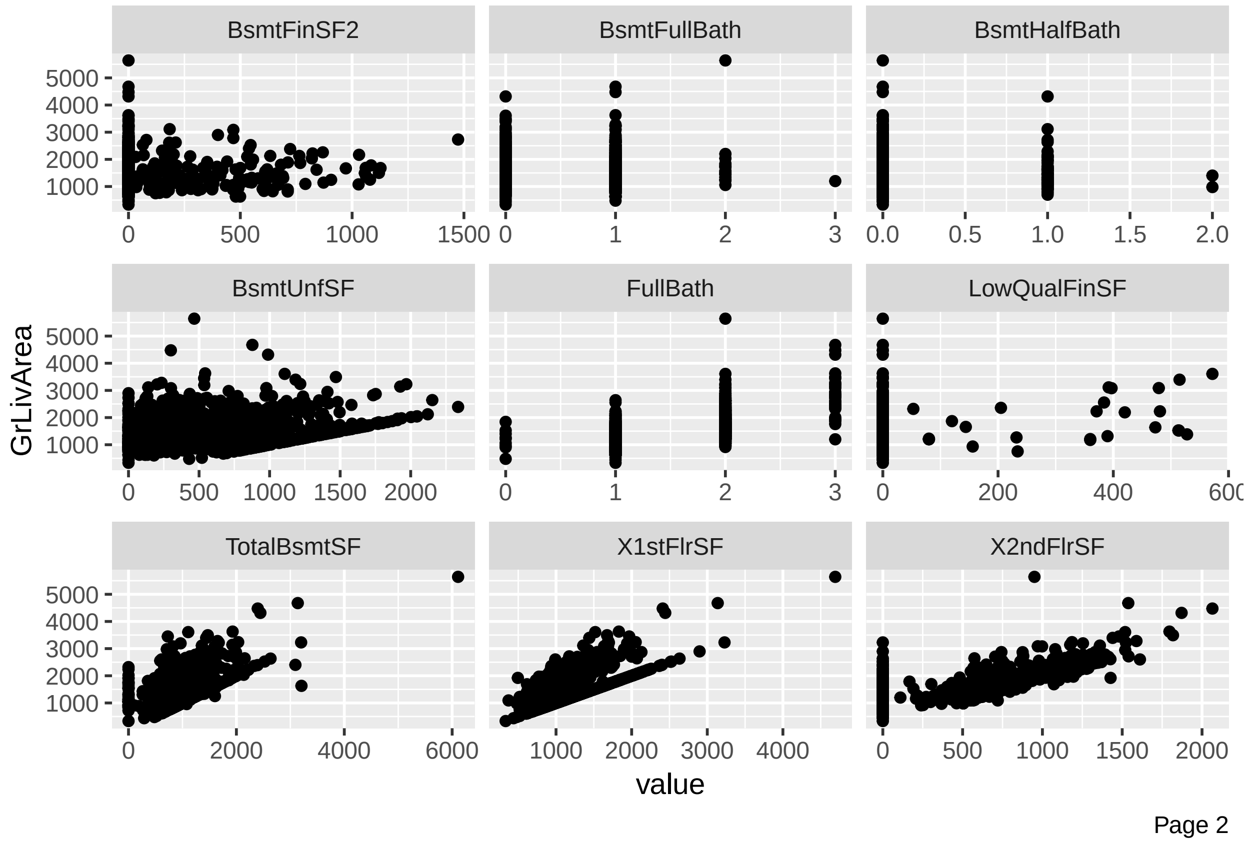
Plotting scatter plots for all variables against the response variable

```
plot_scatterplot(trainc,  by = "GrLivArea")
```

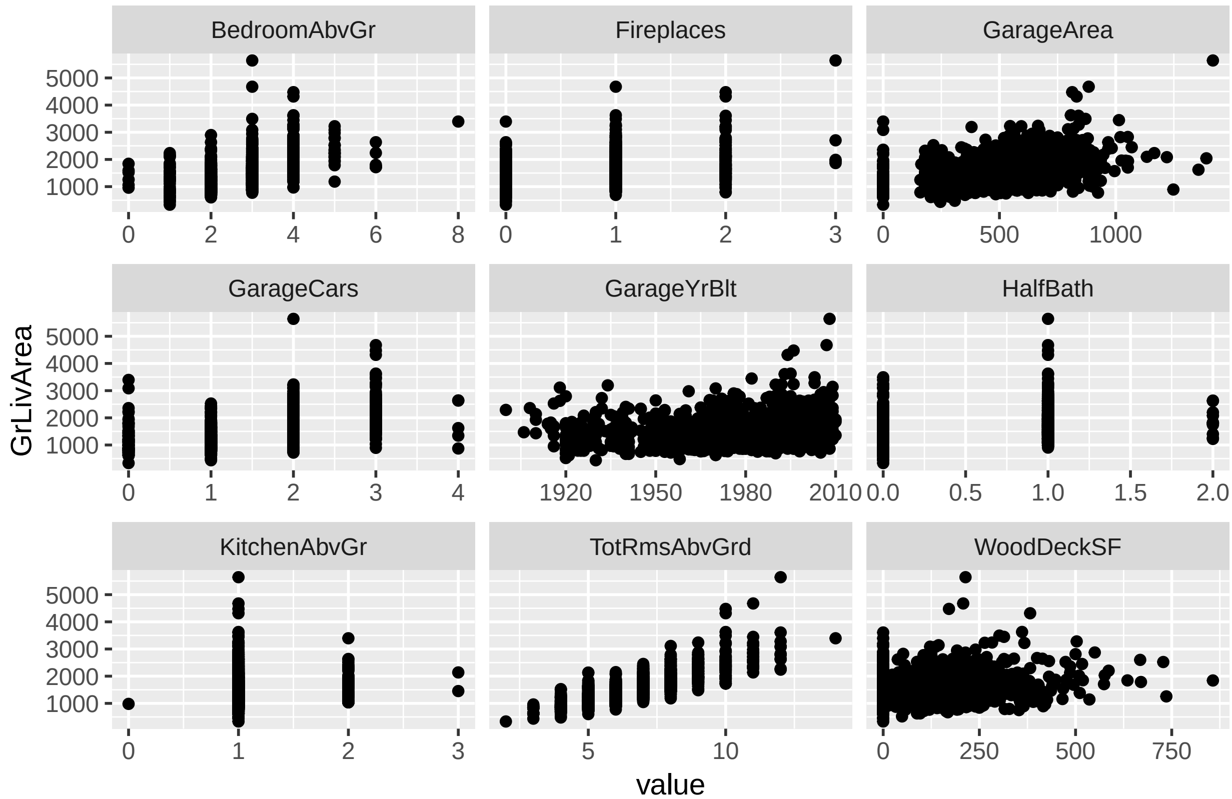
```
## Warning: Removed 267 rows containing missing values (geom_point).
```


BsmtFinSF1
LotArea
LotFrontage
MasVnrArea
MSSubClass
OverallCond
OverallQual
YearBuilt
YearRemodAdd
GrLivArea
value
1000
2000
3000
4000
5000
0
2000
4000
50000
100000
150000
200000
100
200
300
500
1000
1500
50
150
2.5
5.0
7.5
10.0
1900
1940
1980
1960
2000

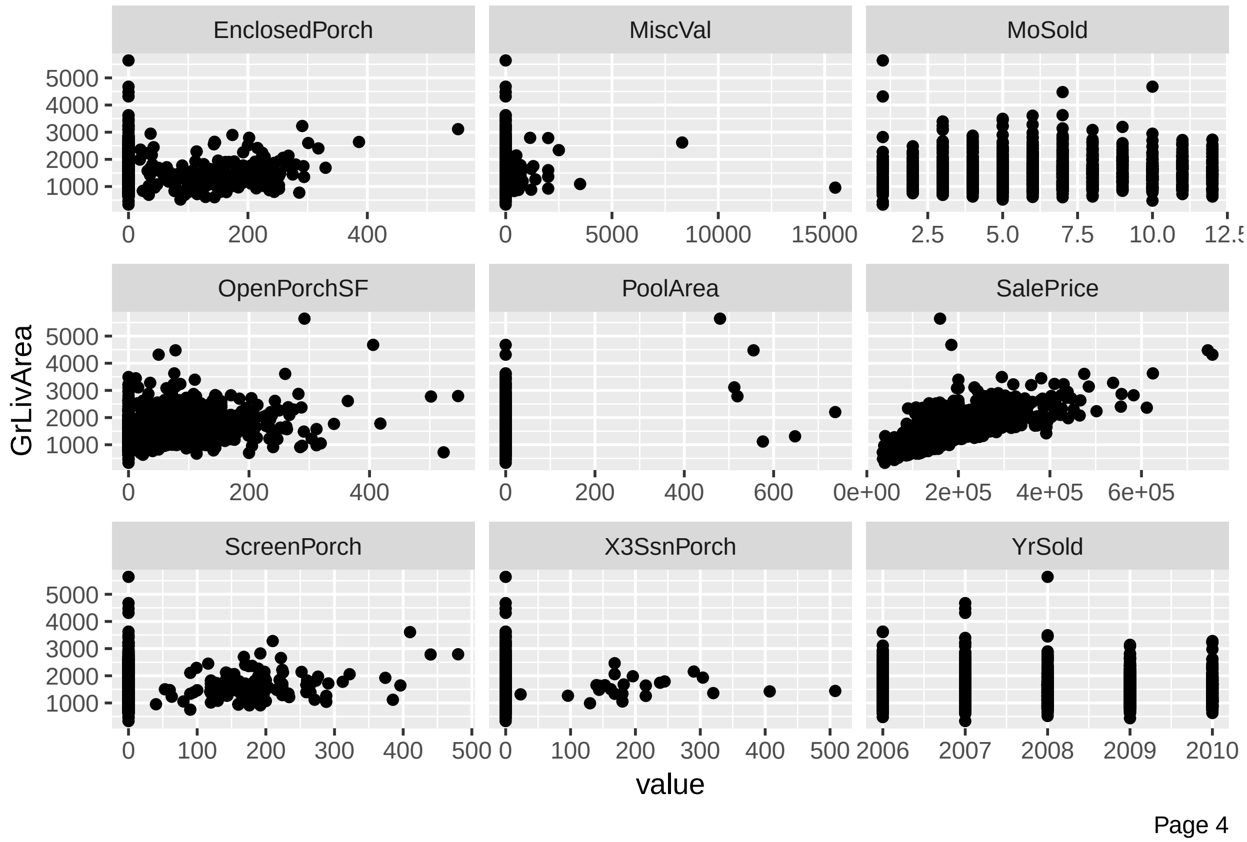
Page 1




```
## Warning: Removed 81 rows containing missing values (geom_point).
```

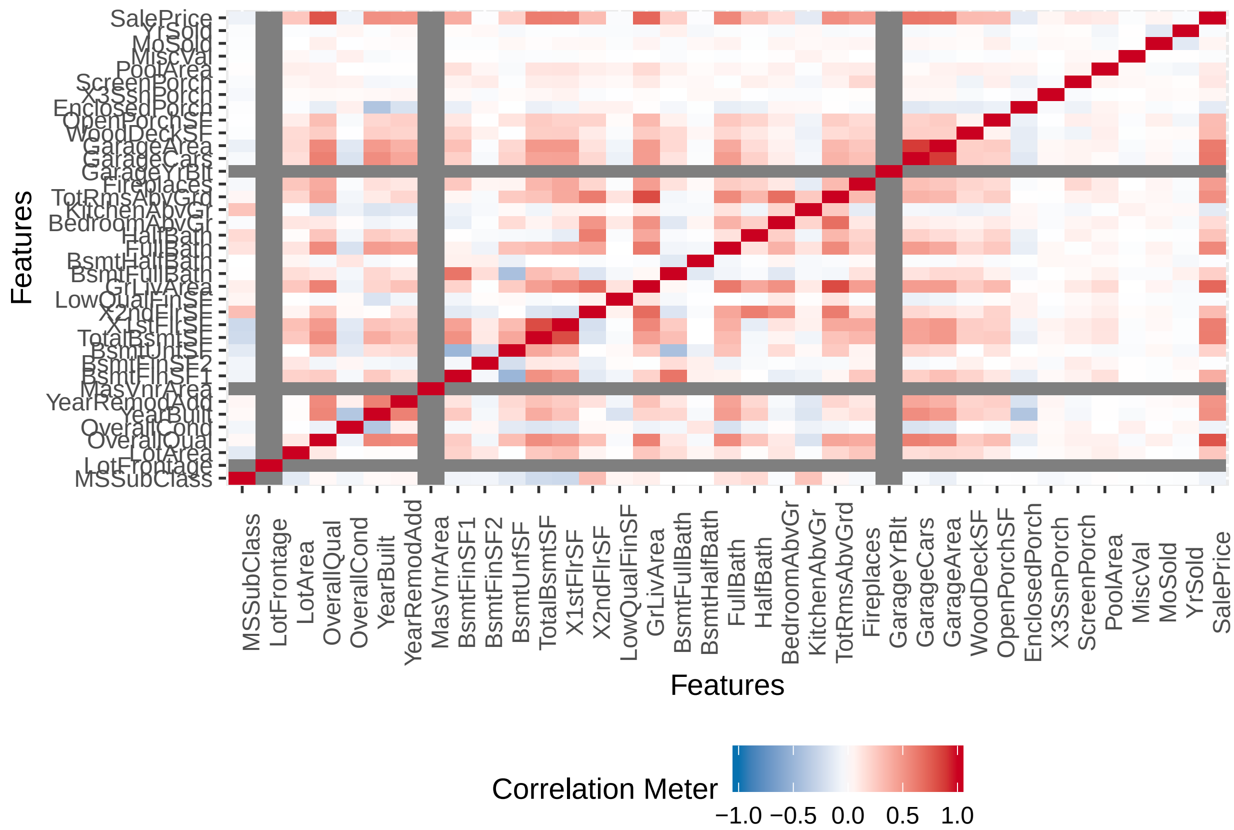

BedroomAbvGr
Fireplaces
GarageArea
GarageCars
GarageYrBlt
HalfBath
KitchenAbvGr
TotRmsAbvGrd
WoodDeckSF
GrLivArea
value
1000
2000
3000
4000
5000
0
2
4
6
8
1
3
500
1920
1950
1980
2010
0.0
0.5
1.0
1.5
2.0
10
250
750

Page 3



Plotting the correlation

```
plot_correlation(trainc)
```



Derive a correlation matrix for any THREE variables .Lets pick the two variables from the scatter plot and the response variable.

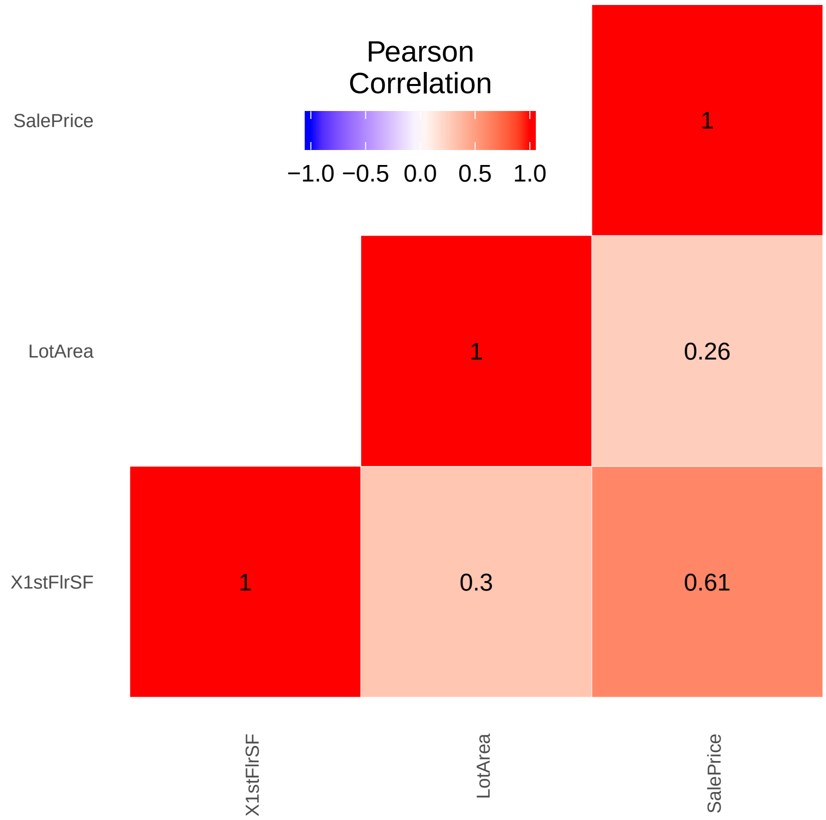
```
corr_data<-subset(trainc,select=c("X1stFlrSF","LotArea", "SalePrice"))


correlation_matrix <- round(cor(corr_data),2)

# Get lower triangle of the correlation matrix
  get_lower_tri<-function(correlation_matrix){
    correlation_matrix[upper.tri(correlation_matrix)] <- NA
    return(correlation_matrix)
  }
  # Get upper triangle of the correlation matrix
  get_upper_tri <- function(correlation_matrix){
    correlation_matrix[lower.tri(correlation_matrix)]<- NA
    return(correlation_matrix)
  }

  upper_tri <- get_upper_tri(correlation_matrix)



library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 3.5.2
```

```r
# Melt the correlation matrix
melted_correlation_matrix <- melt(upper_tri, na.rm = TRUE)

# Heatmap
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```r
ggheatmap <- ggplot(data = melted_correlation_matrix, aes(Var2, Var1, fill = value))+
 geom_tile(color = "white")+
 scale_fill_gradient2(low = "blue", high = "red", mid = "white",
   midpoint = 0, limit = c(-1,1), space = "Lab",
   name="Pearson\nCorrelation") +
  theme_minimal()+
 theme(axis.text.x = element_text(angle = 45, vjust = 1,
    size = 15, hjust = 1))+
 coord_fixed()


#add labels
ggheatmap +
geom_text(aes(Var2, Var1, label = value), color = "black", size = 3) +
theme(
  axis.title.x = element_blank(),
  axis.title.y = element_blank(),
  axis.text.x=element_text(size=rel(0.8), angle=90),
  axis.text.y=element_text(size=rel(0.8)),
  panel.grid.major = element_blank(),
  panel.border = element_blank(),
  panel.background = element_blank(),
  axis.ticks = element_blank(),
  legend.justification = c(1, 0),
  legend.position = c(0.6, 0.7),
  legend.direction = "horizontal")+
  guides(fill = guide_colorbar(barwicrash_training2h = 7, barheight = 1,
               title.position = "top", title.hjust = 0.5))
```



Test the hypotheses that the correlations between each pairwise set of variables is 0 and provide a 80% confidence interval.

```
cor.test(corr_data$X1stFlrSF, corr_data$SalePrice, method = c("pearson", "kendall", "spearman"), conf.le
```

```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$X1stFlrSF and corr_data$SalePrice
## t = 29.078, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.5841687 0.6266715
## sample estimates:
##       cor
## 0.6058522
```

```
cor.test(corr_data$LotArea, corr_data$SalePrice, method = c("pearson", "kendall", "spearman"), conf.leve
```

```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$LotArea and corr_data$SalePrice
## t = 10.445, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.2323391 0.2947946
## sample estimates:
```

```
##       cor
## 0.2638434
```

```
cor.test(corr_data$X1stFlrSF, corr_data$LotArea, method = c("pearson", "kendall", "spearman"), conf.leve
```

```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$X1stFlrSF and corr_data$LotArea
## t = 11.985, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
##  0.2686127 0.3297222
## sample estimates:
##       cor
## 0.2994746
```

In all three instances, we have generated an 80 percent confidence interval. We should also note the small p value. Hence for the three iterations of testing, we can reject the the null hypothesis and conclude that the true correlation is not 0 for the selected variables.

Discuss the meaning of your analysis. Would you be worried about familywise error? Why or why not?

What is a family wise error? The familywise error rate (FWE or FWER) is the probability of a coming to at least one false conclusion in a series of hypothesis tests . In other words, it's the probability of making at least one Type I Error. The term "familywise" error rate comes from family of tests, which is the technical definition for a series of tests on data.The FWER is also called alpha inflation or cumulative Type I error.

```
n=3

alpha=(0.5)/n

print(paste0("Familywise error rate is ", 1-alpha))
```

```
## [1] "Familywise error rate is 0.833333333333333"
```

5 points. Linear Algebra and Correlation. Invert your 3 x 3 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.

Correlation matrix

```
print(correlation_matrix)
```

```
##           X1stFlrSF LotArea SalePrice
## X1stFlrSF      1.00    0.30      0.61
## LotArea        0.30    1.00      0.26
## SalePrice      0.61    0.26      1.00
```

Invert correlation matrix

```
require(Matrix)
```

```
## Loading required package: Matrix
```

```
my_mat <- solve(correlation_matrix)
print(my_mat)
```

```
##            X1stFlrSF    LotArea  SalePrice
## X1stFlrSF  1.6489230 -0.2500619 -0.9408269
```

```
## LotArea   -0.2500619  1.1104234 -0.1361723
## SalePrice -0.9408269 -0.1361723  1.6093092
```

Multiply the correlation matrix by the precision matrix

```r
p_mat <- correlation_matrix%*%my_mat
print(p_mat)
```

```
##                X1stFlrSF     LotArea     SalePrice
## X1stFlrSF  1.000000e+00 1.387779e-17  0.000000e+00
## LotArea   -2.775558e-17 1.000000e+00 -5.551115e-17
## SalePrice -1.110223e-16 0.000000e+00  1.000000e+00
```

multiply the precision matrix by the correlation matrix.

```r
x_mat <- p_mat%*%correlation_matrix
print("We have derived our original correlation matrix")
```

```
## [1] "We have derived our original correlation matrix"
```

```r
print( x_mat)
```

```
##           X1stFlrSF LotArea SalePrice
## X1stFlrSF      1.00    0.30      0.61
## LotArea        0.30    1.00      0.26
## SalePrice      0.61    0.26      1.00
```

Conduct LU decomposition on the matrix.

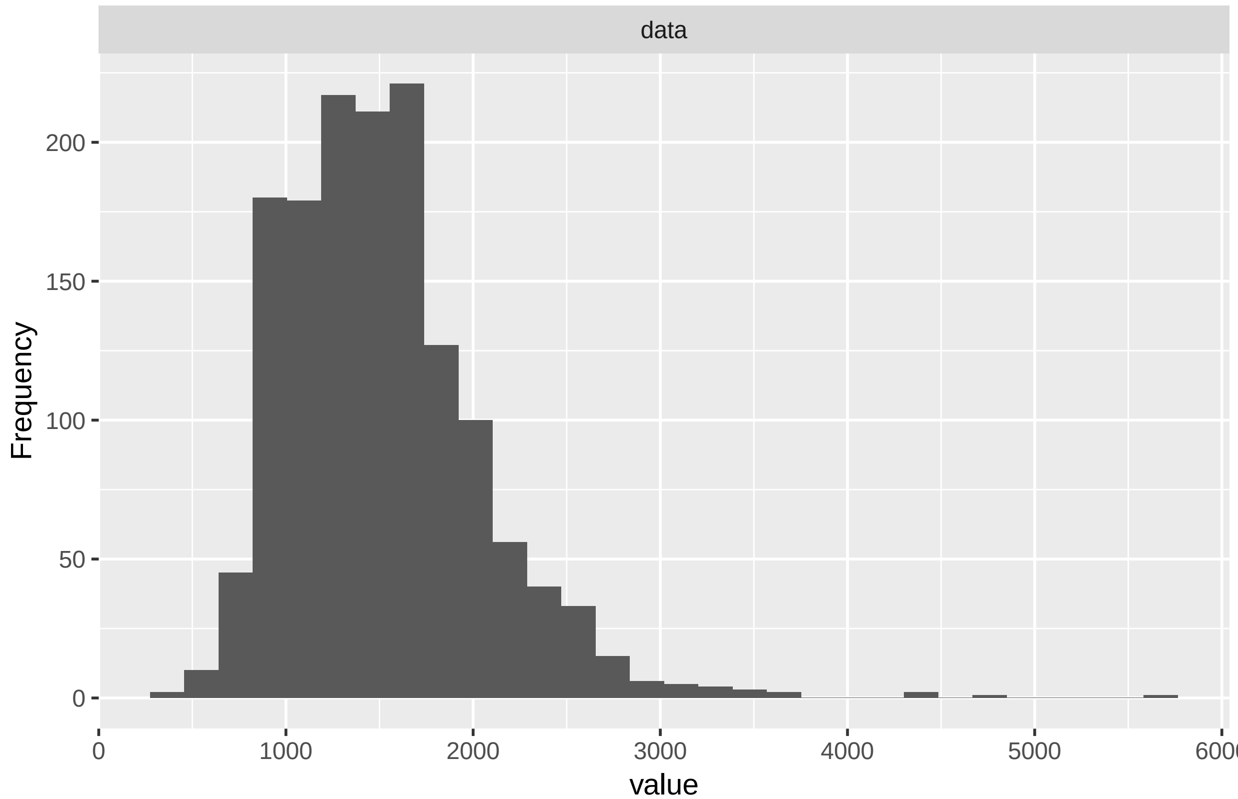
```r
lu_mat<-lu(correlation_matrix)
lu_mat2<-expand(lu_mat)
print(lu_mat2$L %*% lu_mat2$U)
```

```
## 3 x 3 Matrix of class "dgeMatrix"
##      [,1] [,2] [,3]
## [1,] 1.00 0.30 0.61
## [2,] 0.30 1.00 0.26
## [3,] 0.61 0.26 1.00
```

5 points. 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.

Gr Living Area is a variable with a right skew

```r
plot_histogram(trainc$GrLivArea);
```




```
summary(trainc$GrLivArea)
```

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

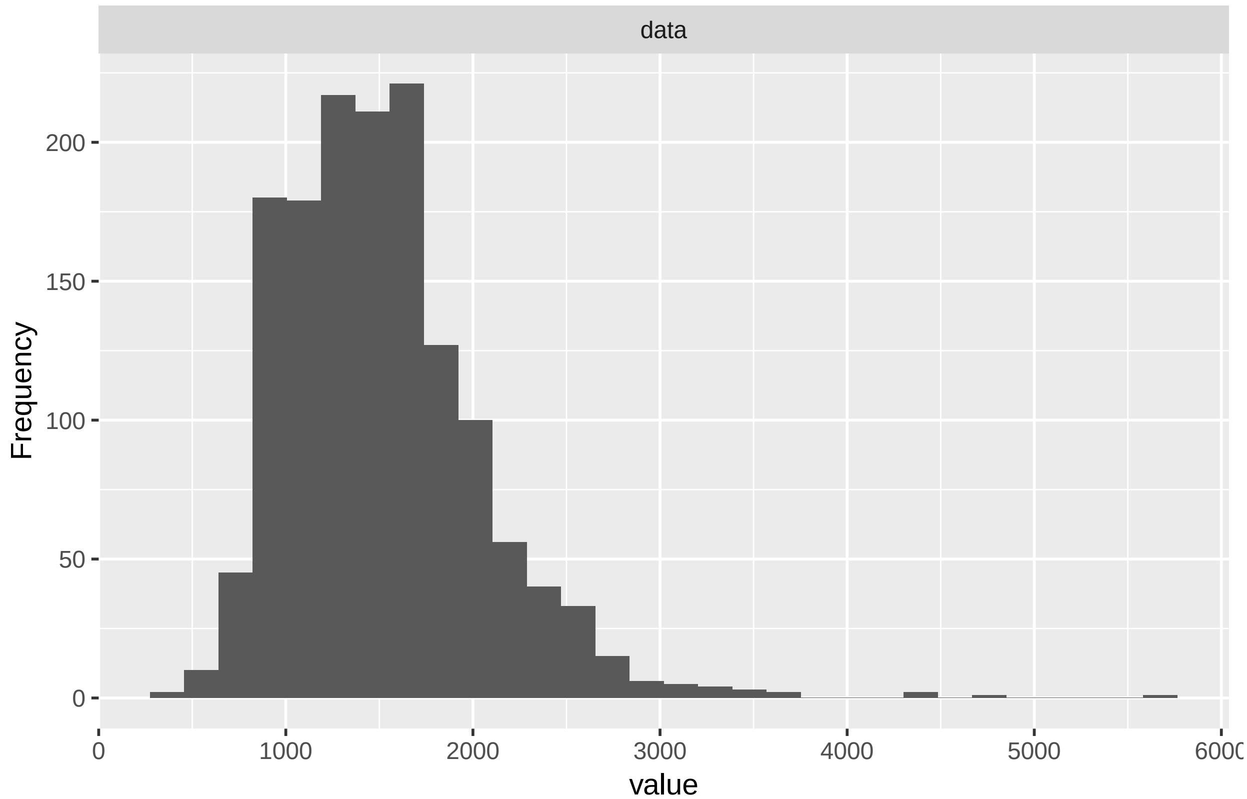
Gr Living Area does not have a minimum of zero, therefore we do not need to shift the variable.

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 Î» for this distribution, and then take 1000 samples from this exponential distribution using this value (e.g., rexp(1000, Î»)). Plot a histogram and compare it with a histogram of your original variable.

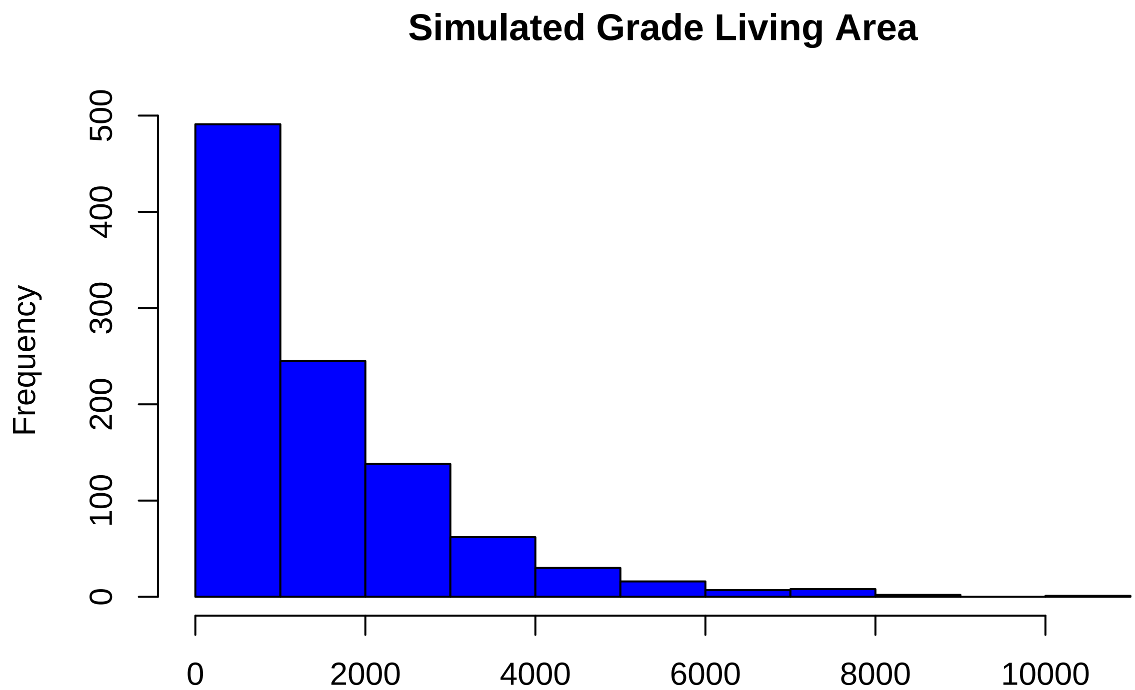
```
library(MASS)

dist<-fitdistr(trainc$GrLivArea, densfun = 'exponential')
lamda <- dist$estimate
exp_distibution <- rexp(1000, lamda)

summary(exp_distibution);
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##     1.486   452.588  1025.358  1482.973  2087.005 10081.691
```

```
plot_histogram(trainc$GrLivArea);
```




```
hist(exp_distibution, main = "Simulated Grade Living Area", xlab="", col = "blue")
```



Using the exponential pdf, find the 5th and 95th percentiles using the cumulative distribution function (CDF).

```
quantile(exp_distibution, c(.05, .95))
```

```
##         5%        95%
##   58.66712 4350.01900
```

Also generate a 95% confidence interval from the empirical data, assuming normality.

```
library(Rmisc)
```

```
## Warning: package 'Rmisc' was built under R version 3.5.3

## Loading required package: lattice

## Loading required package: plyr
```

```
CI(trainc$GrLivArea, ci=0.95)
```

```
##    upper     mean    lower
## 1542.440 1515.464 1488.487
```

Finally, provide the empirical 5th percentile and 95th percentile of the data. Discuss.

```
quantile(trainc$GrLivArea, c(.05, .95))
```

```
##     5%    95%
##  848.0 2466.1
```

10 points. 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.

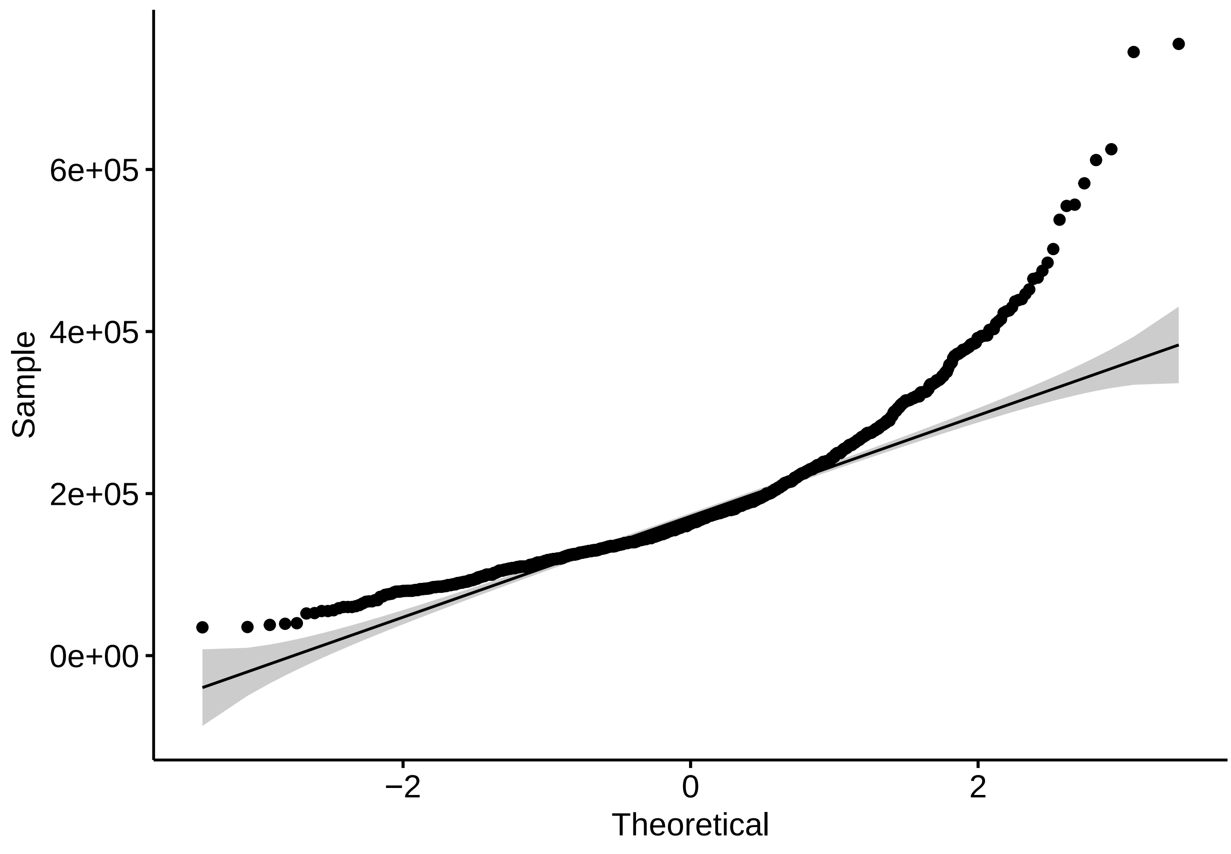
When it comes to modeling, there are numerous things that can be done with some more involved than others. I could do PCA decomposition to reduce the dimension of the data. I could also create multiple dummy variables with a degrees of freedom trade off. For the sake of this course, I will keep the model simple and limit them to variables that had decent correlation with Sales Price.

Lets see how close to the normal distribution our response variable is

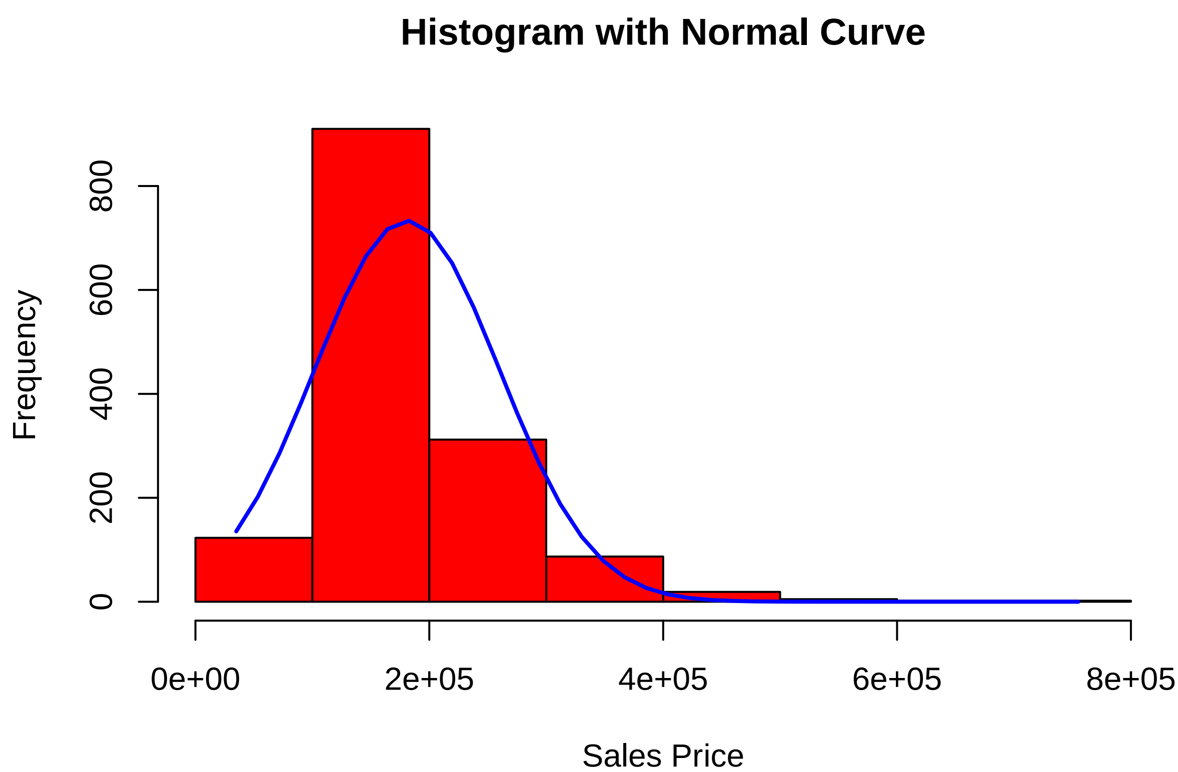
```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 3.5.3

## Loading required package: magrittr

##
## Attaching package: 'ggpubr'

## The following object is masked from 'package:plyr':
##
##     mutate
```

```
ggqqplot(trainc$SalePrice);
```




```
x <- trainc$SalePrice
h<-hist(x, breaks=10, col="red", xlab="Sales Price",
    main="Histogram with Normal Curve")
xfit<-seq(min(x),max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
```



It seems that no major transformation needs to be done on the response variable, however we will confirm with diagnostics.

We examined a heat map based off the correlation matrix. We can use that to our advantage. We can actually systematically go through a process that can identify what predictors have significant correlations with the response variables.

```
cor(trainc[-37], trainc$SalePrice)
```

```
##                    [,1]
## MSSubClass    -0.08428414
## LotFrontage            NA
## LotArea        0.26384335
## OverallQual    0.79098160
## OverallCond   -0.07785589
## YearBuilt      0.52289733
## YearRemodAdd   0.50710097
## MasVnrArea             NA
## BsmtFinSF1     0.38641981
## BsmtFinSF2    -0.01137812
## BsmtUnfSF      0.21447911
## TotalBsmtSF    0.61358055
## X1stFlrSF      0.60585218
## X2ndFlrSF      0.31933380
## LowQualFinSF  -0.02560613
## GrLivArea      0.70862448
## BsmtFullBath   0.22712223
```

```
## BsmtHalfBath  -0.01684415
## FullBath       0.56066376
## HalfBath       0.28410768
## BedroomAbvGr   0.16821315
## KitchenAbvGr  -0.13590737
## TotRmsAbvGrd   0.53372316
## Fireplaces     0.46692884
## GarageYrBlt            NA
## GarageCars     0.64040920
## GarageArea     0.62343144
## WoodDeckSF     0.32441344
## OpenPorchSF    0.31585623
## EnclosedPorch -0.12857796
## X3SsnPorch     0.04458367
## ScreenPorch    0.11144657
## PoolArea       0.09240355
## MiscVal       -0.02118958
## MoSold         0.04643225
## YrSold        -0.02892259
```

We will take variables with strong positive correlations greater than .5. Using backward elimination technique to arrive at the optimal features to be included to our model.

```r
mod <- lm(SalePrice~GarageArea+GarageCars+TotRmsAbvGrd+FullBath+GrLivArea+X1stFlrSF+TotalBsmtSF+YearRem

summary(mod)
```

```
##
## Call:
## lm(formula = SalePrice ~ GarageArea + GarageCars + TotRmsAbvGrd +
##     FullBath + GrLivArea + X1stFlrSF + TotalBsmtSF + YearRemodAdd +
##     YearBuilt + OverallQual, data = trainc)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -489958  -19316   -1948   16020  290558
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.186e+06  1.291e+05  -9.187  < 2e-16 ***
## GarageArea    1.495e+01  1.031e+01   1.450 0.147384
## GarageCars    1.042e+04  3.044e+03   3.422 0.000639 ***
## TotRmsAbvGrd  3.310e+01  1.119e+03   0.030 0.976404
## FullBath     -6.791e+03  2.682e+03  -2.532 0.011457 *
## GrLivArea     5.130e+01  4.233e+00  12.119  < 2e-16 ***
## X1stFlrSF     1.417e+01  4.930e+00   2.875 0.004097 **
## TotalBsmtSF   1.986e+01  4.295e+00   4.625 4.09e-06 ***
## YearRemodAdd  2.965e+02  6.363e+01   4.659 3.47e-06 ***
## YearBuilt     2.682e+02  5.035e+01   5.328 1.15e-07 ***
## OverallQual   1.960e+04  1.190e+03  16.472  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 37920 on 1449 degrees of freedom
## Multiple R-squared:  0.7737, Adjusted R-squared:  0.7721
```

```
## F-statistic: 495.4 on 10 and 1449 DF,  p-value: < 2.2e-16
```

Garage area and Total Rooms Above grade do not appear to be significant as p-value is more than 0.05 hence we can remove this variable,and adjusted R squared value of .77 meaning, 77% of the variability in the data is accounted for.

Remove non significant predictors and re-fit model

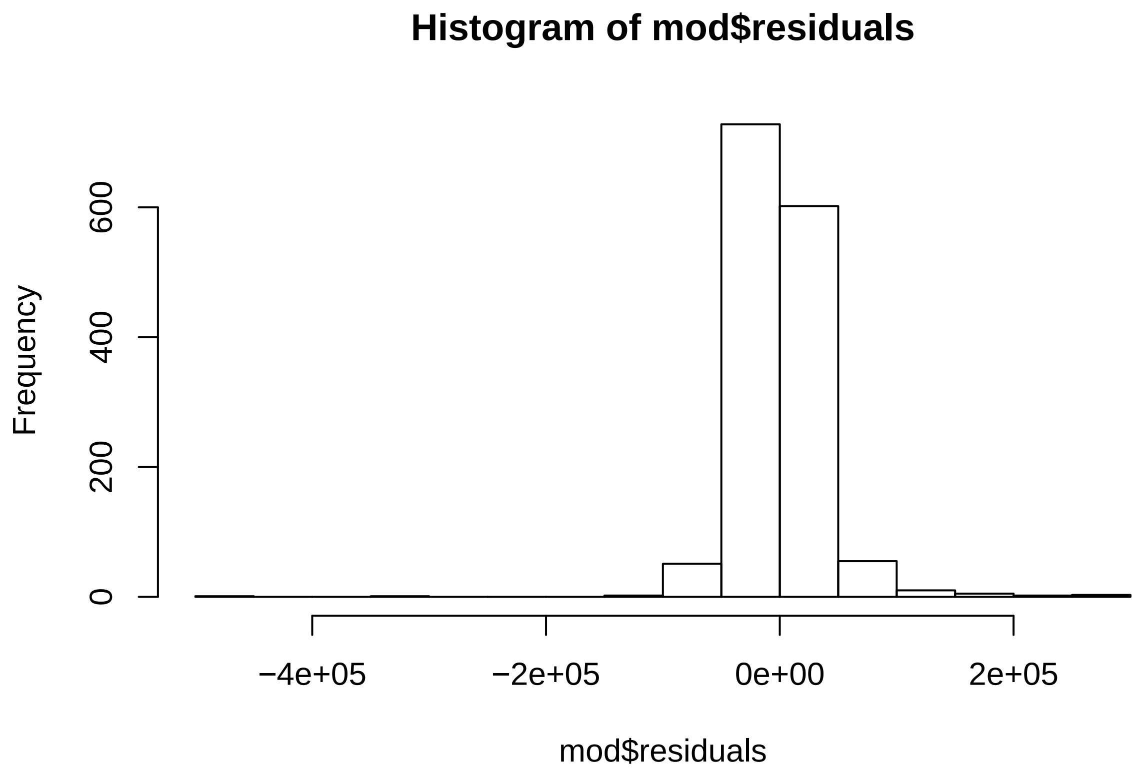
```
mod <- lm(SalePrice~GarageCars+FullBath+GrLivArea+X1stFlrSF+TotalBsmtSF+YearRemodAdd+YearBuilt+OverallQu

summary(mod)
```

```
##
## Call:
## lm(formula = SalePrice ~ GarageCars + FullBath + GrLivArea +
##     X1stFlrSF + TotalBsmtSF + YearRemodAdd + YearBuilt + OverallQual,
##     data = trainc)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -482525  -19191   -1801   16208  289639
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.188e+06  1.284e+05  -9.255  < 2e-16 ***
## GarageCars    1.395e+04  1.817e+03   7.680 2.92e-14 ***
## FullBath     -7.184e+03  2.644e+03  -2.717  0.00666 **
## GrLivArea     5.177e+01  3.097e+00  16.714  < 2e-16 ***
## X1stFlrSF     1.465e+01  4.919e+00   2.979  0.00294 **
## TotalBsmtSF   2.039e+01  4.269e+00   4.775 1.98e-06 ***
## YearRemodAdd  2.957e+02  6.362e+01   4.649 3.64e-06 ***
## YearBuilt     2.699e+02  5.017e+01   5.380 8.69e-08 ***
## OverallQual   1.959e+04  1.188e+03  16.486  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 37920 on 1451 degrees of freedom
## Multiple R-squared:  0.7734, Adjusted R-squared:  0.7721
## F-statistic: 618.9 on 8 and 1451 DF,  p-value: < 2.2e-16
```

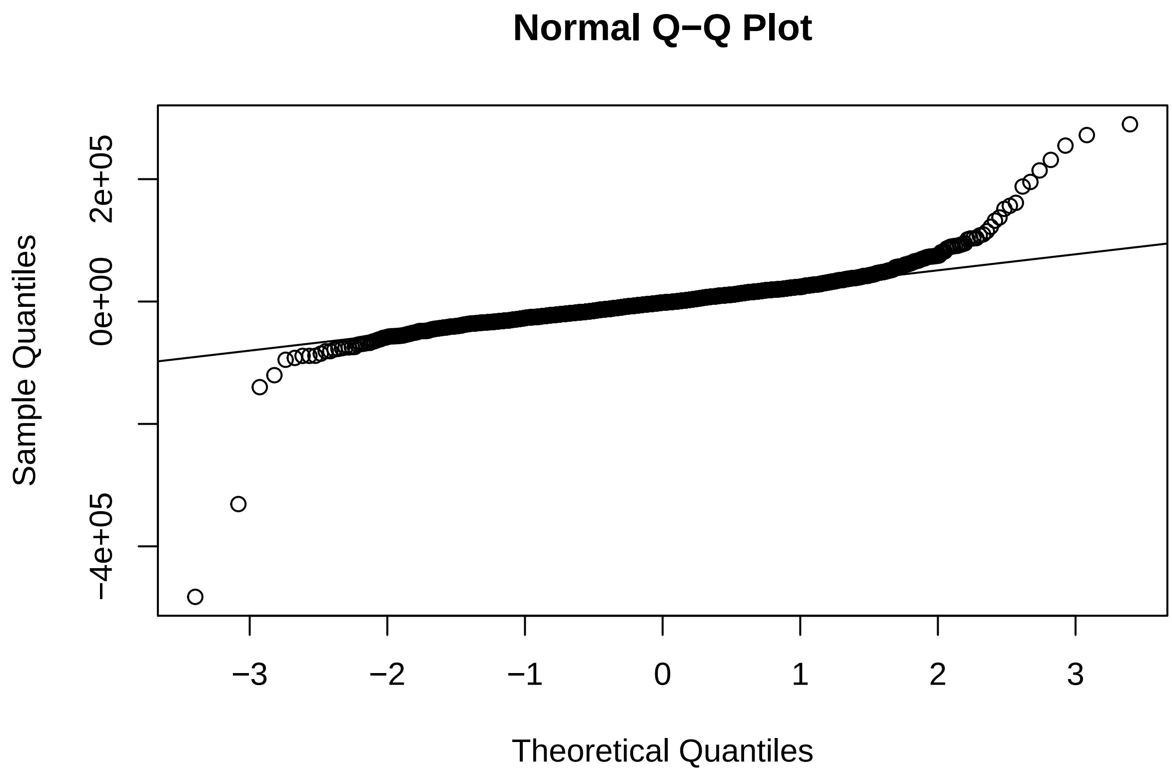
We still retain almost identical adjusted r square with fewer predictors.

Diagnostics

```
hist(mod$residuals);
```




```
qqnorm(mod$residuals);
qqline(mod$residuals)
```



The residuals seem to follow a close to normal distribution. We need to check constant variance.

```r
library('olsrr')
```

```
## Warning: package 'olsrr' was built under R version 3.5.3

##
## Attaching package: 'olsrr'

## The following object is masked from 'package:MASS':
##
##     cement

## The following object is masked from 'package:datasets':
##
##     rivers
```

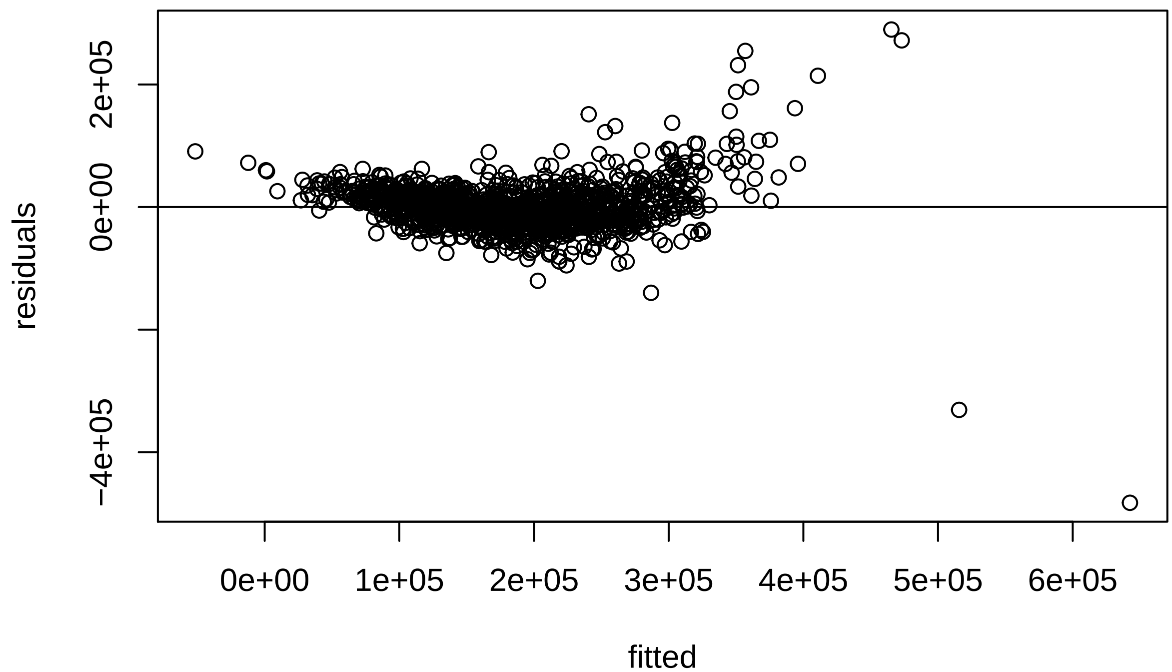
```r
olsrr::ols_test_breusch_pagan(mod)
```

```
##
##  Breusch Pagan Test for Heteroskedasticity
##  -----------------------------------------
##  Ho: the variance is constant
##  Ha: the variance is not constant
##
##                Data
##  -------------------------------------
##  Response : SalePrice
##  Variables: fitted values of SalePrice
```

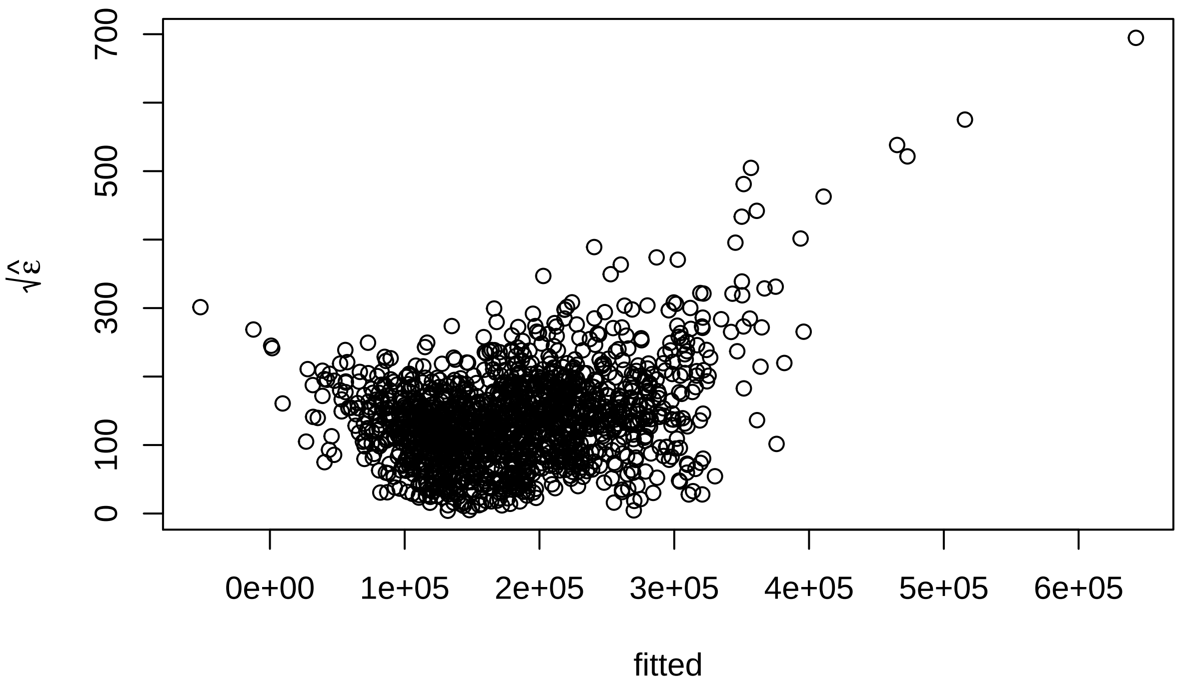
```
##
##         Test Summary
##  ----------------------------
##  DF              =    1
##  Chi2            =    2921.5080
##  Prob > Chi2     =    0.0000
```

A small p value in the Breusch Pagan Test for Heteroskedasticity indicates strong evidence against the null value. We can say that constant variance is not met. Let us check visually.

```r
plot(fitted(mod), residuals(mod), xlab="fitted", ylab="residuals")
abline(h=0);
```



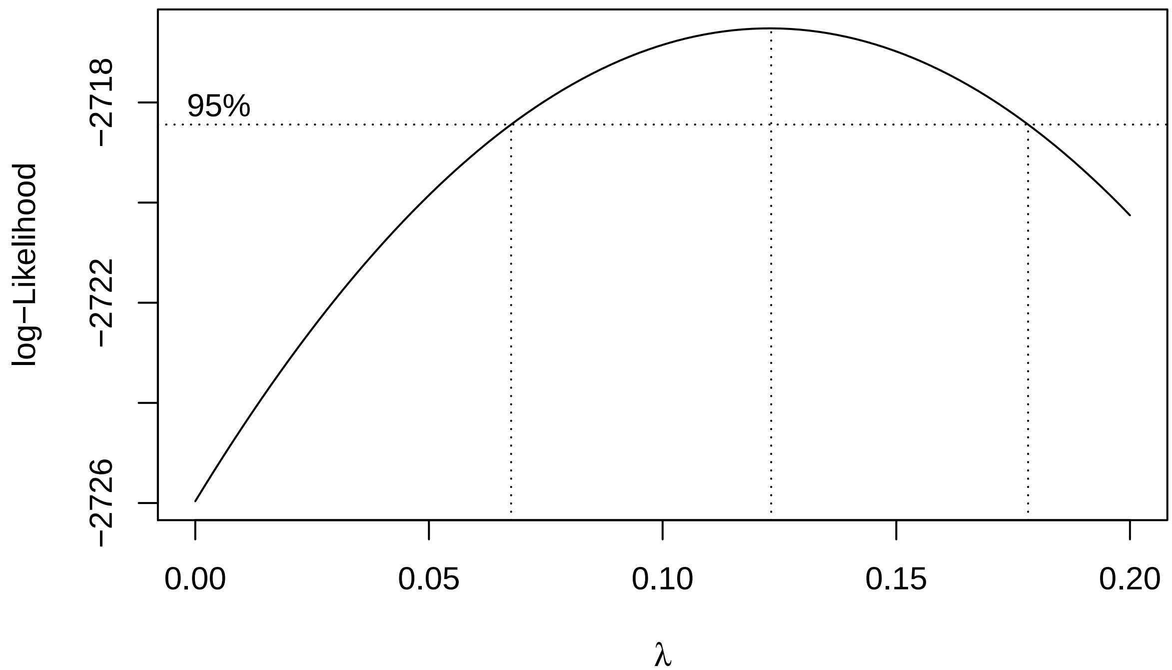

```r
plot(fitted(mod), sqrt(abs(residuals(mod))), xlab="fitted", ylab=expression(sqrt(hat(epsilon))))
```



If we look at the residuals, we can see a parabolic shape indicating that some transform needs to be done on the response variable. If we look at the square root of the residuals, the parabolic pattern becomes much more prominant.

We can apply the Box-Cox transform. This worlflow is highlighted in detail in Julian Farayws linear model in r book.

```
library(MASS)

boxcox(mod, plotit=T, lambda=seq(0, 0.2, by=0.01))
```



According to the transform, the max log-likelihood happens around -2700. We can estimate a parameter lambda by using the center line bounded by the interval roughly (0.07, 0.17). It looks like our power transform is going to be 0.13

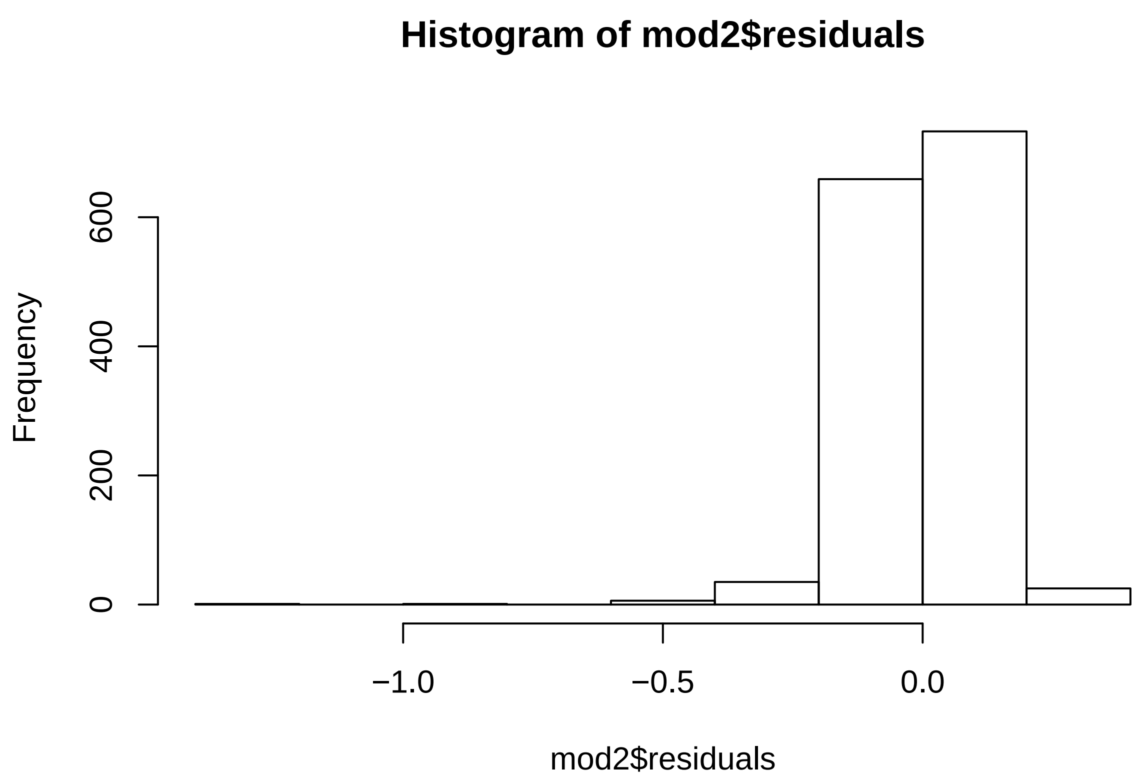
```
traind<-trainc

mod2 <- lm(SalePrice^(0.13)~GarageCars+FullBath+GrLivArea+X1stFlrSF+TotalBsmtSF+YearRemodAdd+YearBuilt+(

summary(mod2)
```

```
##
## Call:
## lm(formula = SalePrice^(0.13) ~ GarageCars + FullBath + GrLivArea +
##     X1stFlrSF + TotalBsmtSF + YearRemodAdd + YearBuilt + OverallQual,
##     data = trainc)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.38745 -0.04679  0.00302  0.05812  0.34836
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -8.760e-01  3.544e-01  -2.471   0.0136 *
## GarageCars    5.209e-02  5.016e-03  10.383  < 2e-16 ***
## FullBath     -1.291e-02  7.301e-03  -1.769   0.0771 .
## GrLivArea     1.500e-04  8.552e-06  17.542  < 2e-16 ***
## X1stFlrSF     3.827e-05  1.358e-05   2.818   0.0049 **
```

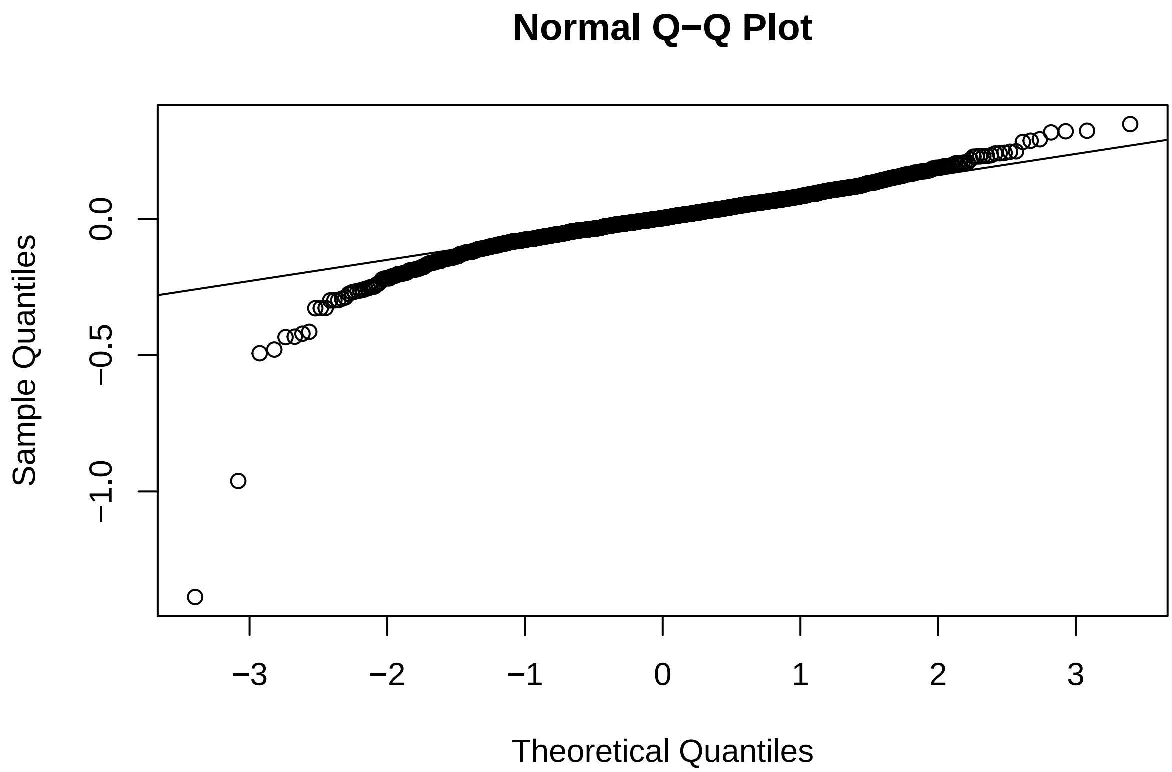
```
## TotalBsmtSF   5.718e-05  1.179e-05   4.850 1.36e-06 ***
## YearRemodAdd  1.331e-03  1.757e-04   7.579 6.19e-14 ***
## YearBuilt     1.139e-03  1.385e-04   8.219 4.50e-16 ***
## OverallQual   5.982e-02  3.281e-03  18.234  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1047 on 1451 degrees of freedom
## Multiple R-squared:  0.8246, Adjusted R-squared:  0.8236
## F-statistic: 852.4 on 8 and 1451 DF,  p-value: < 2.2e-16
```

Residual standard error has decreased significantly while out adjusted r square has increased from .77 to .82.

```r
hist(mod2$residuals);
```

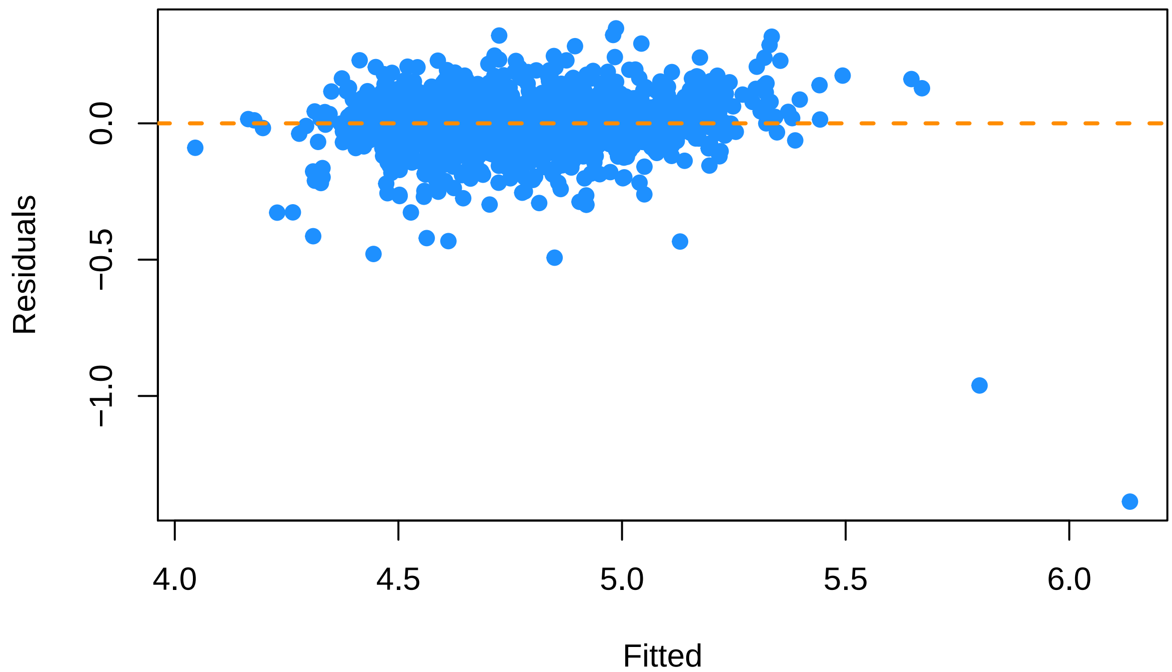



```r
qqnorm(mod2$residuals);
qqline(mod2$residuals)
```



There is a slight skew introduced into the residuals but it does not appear to be much. Lets visually check constant variance.

```
plot(fitted(mod2), resid(mod2), col = "dodgerblue",
     pch = 20, cex = 1.5, xlab = "Fitted", ylab = "Residuals")
abline(h = 0, lty = 2, col = "darkorange", lwd = 2)
```



Lets examine outliers on a top level

```
library(car)
```

```
## Warning: package 'car' was built under R version 3.5.3
```

```
## Loading required package: carData
```

```
## Warning: package 'carData' was built under R version 3.5.2
```

```
outlierTest(mod2)
```

```
##        rstudent unadjusted p-value Bonferonni p
## 1299 -15.538912         1.7562e-50   2.5640e-47
## 524   -9.634839         2.4441e-21   3.5683e-18
## 633   -4.748411         2.2531e-06   3.2895e-03
## 31    -4.617930         4.2190e-06   6.1597e-03
## 1325  -4.171794         3.2015e-05   4.6742e-02
```

We have identified several outliers however the low p values indicates that they do not seem to be significant.

Can we reduce our predictors even more by using variance inflation numbers?

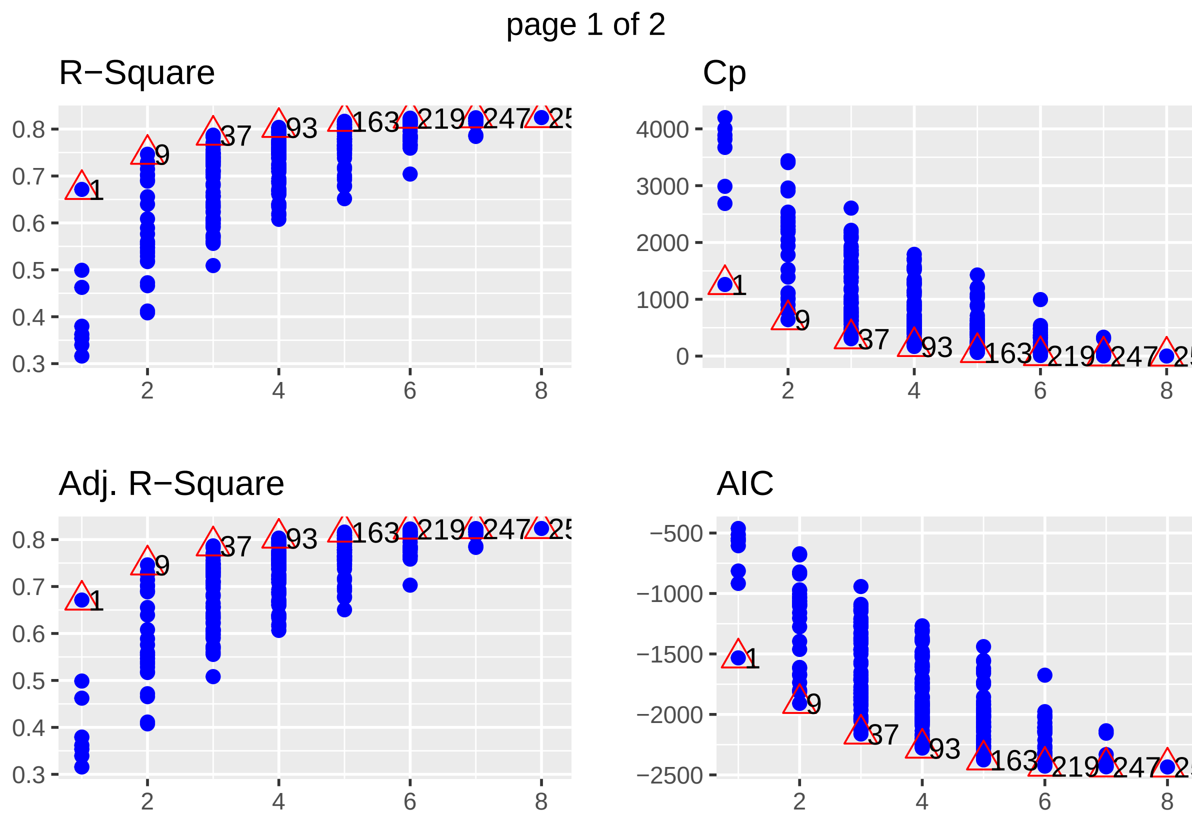
```
vif(mod2)
```

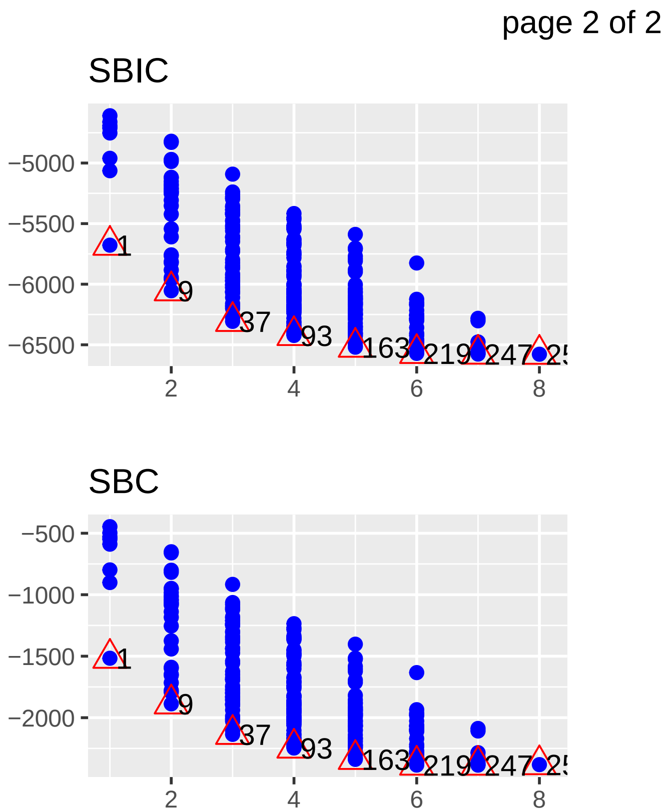
```
##   GarageCars     FullBath    GrLivArea     X1stFlrSF  TotalBsmtSF
##     1.869688     2.152343     2.687048      3.668313     3.558638
## YearRemodAdd    YearBuilt  OverallQual
##     1.750023     2.329177     2.738752
```

VIF numbers indicate that we do not need to remove additional predictors since there is no VIF number that is unsually large.

Before we conclude modeling, lets examine all possible permutations of predictors and see their performance based on KPI such as adjusted r square and mallows CP.

```
k<-ols_step_all_possible(mod2)
plot(k)
```





From a top level, using all 8 predictors yields the better adjusted r square and reduced the AIC.

Feature selection at a more detailed level

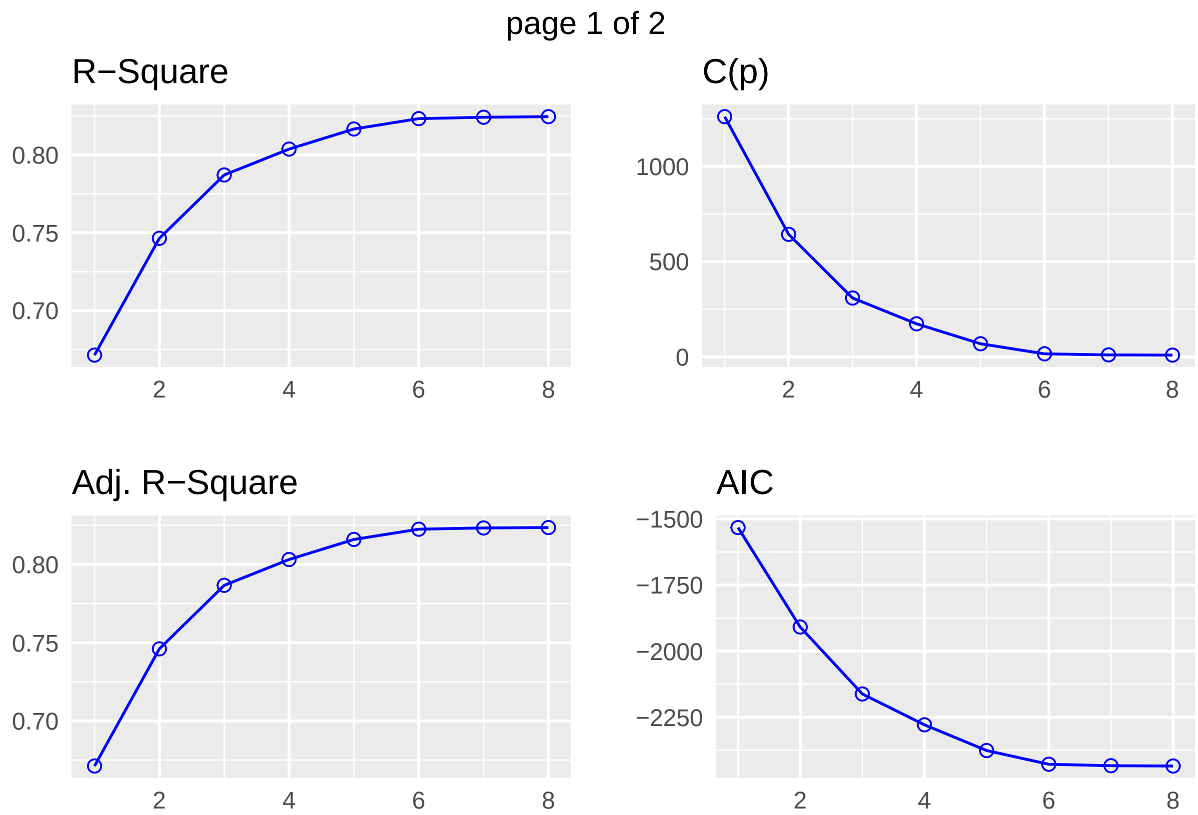
```r
h<-ols_step_best_subset(mod2)
h
```

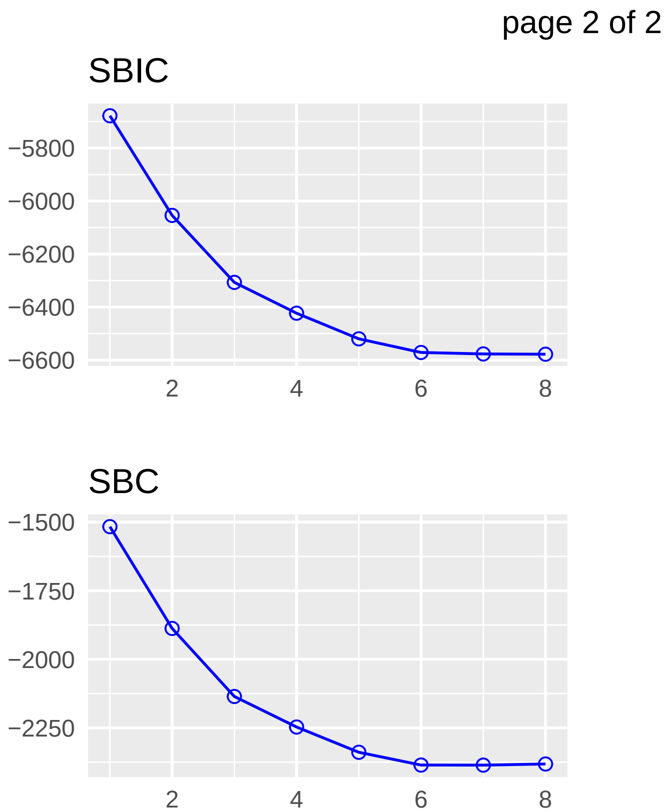
```
##                                                Best Subsets Regression
## ----------------------------------------------------------------------------------------------------------
## Model Index    Predictors
## ----------------------------------------------------------------------------------------------------------
##      1         OverallQual
##      2         GrLivArea OverallQual
##      3         GrLivArea YearBuilt OverallQual
##      4         GarageCars GrLivArea YearBuilt OverallQual
##      5         GarageCars GrLivArea TotalBsmtSF YearBuilt OverallQual
##      6         GarageCars GrLivArea TotalBsmtSF YearRemodAdd YearBuilt OverallQual
##      7         GarageCars GrLivArea X1stFlrSF TotalBsmtSF YearRemodAdd YearBuilt OverallQual
##      8         GarageCars FullBath GrLivArea X1stFlrSF TotalBsmtSF YearRemodAdd YearBuilt OverallQual
## ----------------------------------------------------------------------------------------------------------
##
##                                                                Subsets Regression Summary
## ----------------------------------------------------------------------------------------------------------
##                        Adj.        Pred
## Model     R-Square   R-Square    R-Square      C(p)          AIC           SBIC           SBC
## ----------------------------------------------------------------------------------------------------------
##   1        0.6714      0.6712      0.6704    1261.4740    -1532.4749    -5678.0546    -1516.6163
##   2        0.7464      0.7461      0.7435     643.2617    -1908.7174    -6054.0257    -1887.5727
```

```
##   3        0.7871       0.7867       0.7839      308.7680      -2162.0458      -6306.7712      -2135.6149     (
##   4        0.8037       0.8032       0.8003      173.3529      -2278.6801      -6423.0072      -2246.9630     (
##   5        0.8166       0.8160       0.8073       68.6192      -2375.9748      -6519.7261      -2338.9715     (
##   6        0.8233       0.8225       0.8137       15.6517      -2427.8729      -6571.1893      -2385.5834     (
##   7        0.8242       0.8233       0.8144       10.1290      -2433.4066      -6576.6425      -2385.8308     (
##   8        0.8246       0.8236       0.8134        9.0000      -2434.5516      -6577.7405      -2381.6897     (
## -----------------------------------------------------------------------------------------------------
## AIC: Akaike Information Criteria
##  SBIC: Sawa's Bayesian Information Criteria
##  SBC: Schwarz Bayesian Criteria
##  MSEP: Estimated error of prediction, assuming multivariate normality
##  FPE: Final Prediction Error
##  HSP: Hocking's Sp
##  APC: Amemiya Prediction Criteria
```

It seems that model 4-8 are pretty close in adjusted r square but if any more predictors get removed, there is a sharp drop off.

```r
plot(h)
```





It is easy to keep looking for methods to optimize the model. I would even go as far as saying that a GLM should be considered here but that is outside the scope of the class.

Lets apply to our test data and make some predictions

```
test_results <- predict(mod2, test)

prediction <- data.frame(Id = test[,"Id"],  SalePrice = test_results)

prediction[prediction<0] <- 0

prediction <- replace(prediction,is.na(prediction),0)

prediction$SalePrice <- prediction$SalePrice^(1/.13)

head(test_results)
```

```
##        1        2        3        4        5        6
## 4.525119 4.684621 4.768366 4.825383 4.915384 4.806456
```

```
write.csv(prediction, "SPpredictions.csv")
```

Kaggle results Number 5761, posted under vishal0229 : 1 submission score 0.47026

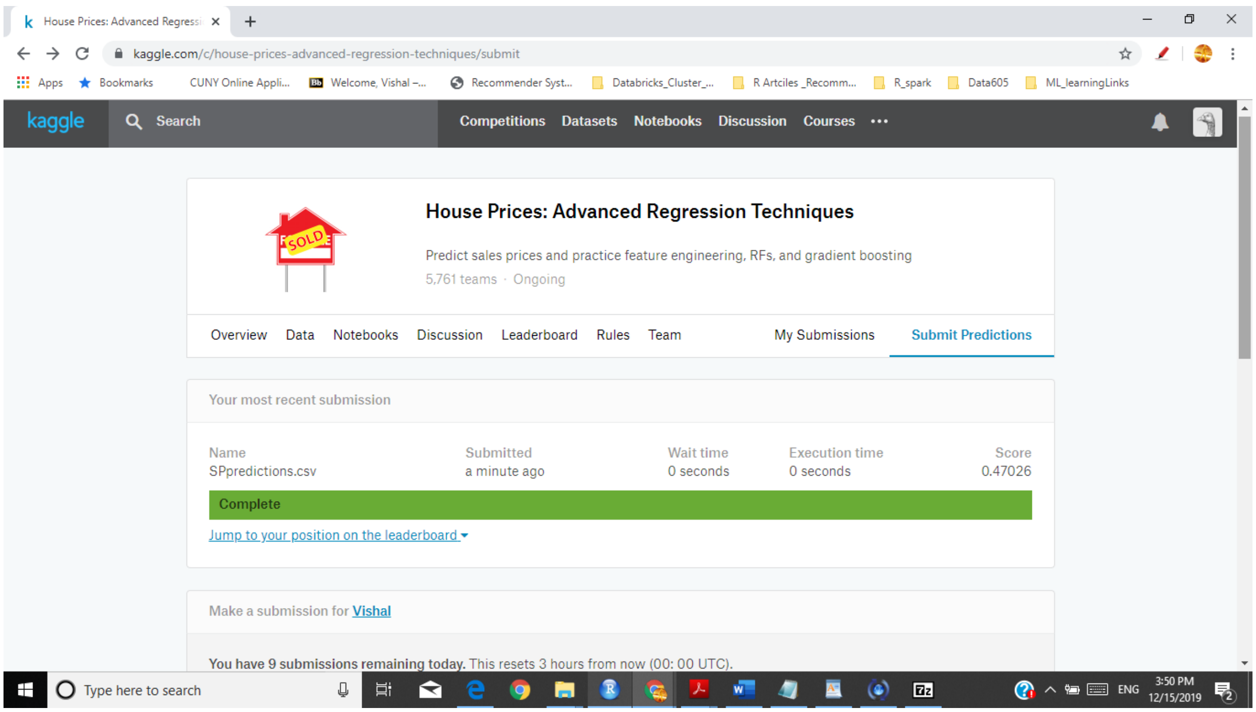

House Prices: Advanced Regression Techniques
Predict sales prices and practice feature engineering, RFs, and gradient boosting
5,761 teams · Ongoing
Overview Data Notebooks Discussion Leaderboard Rules Team My Submissions Submit Predictions
Your most recent submission
Name SPpredictions.csv
Submitted a minute ago
Wait time 0 seconds
Execution time 0 seconds
Score 0.47026
Complete
Jump to your position on the leaderboard
Make a submission for Vishal
You have 9 submissions remaining today. This resets 3 hours from now (00: 00 UTC).

Figure 1: