

# Final Exam

Subhalaxmi Rout

2020-12-07

**YouTube Link:**

## 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  $\mu = \sigma = \frac{(N+1)}{2}$ .

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.  $P(X>x \mid X>y)$  b.  $P(X>x, Y>y)$  c.  $P(X< x \mid X>y)$

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

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?

*I. 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  $\mu = \sigma = \frac{(N+1)}{2}$ .*

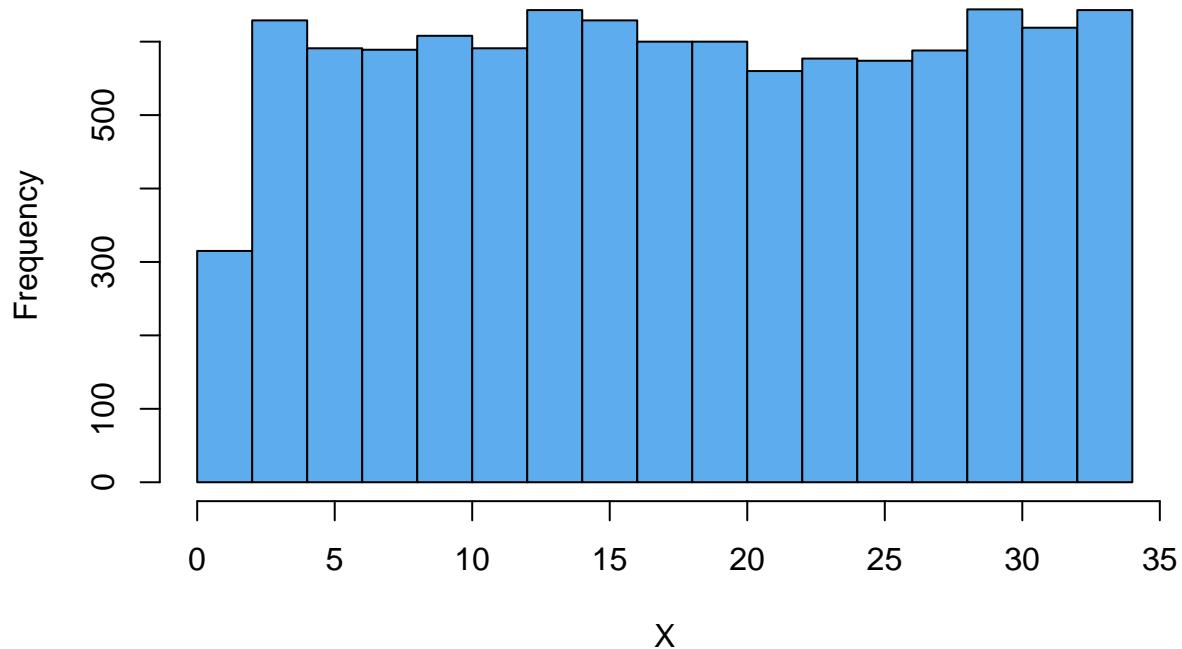
## Solution

```
set.seed(1)
N <- round(runif(1, 10, 100))
mu <- (N+1)/2
sd <- (N+1)/2
X <- runif(10000,min=1,max=N)
summary(X)

##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
##    1.004   9.331  17.359  17.505  25.977  33.998

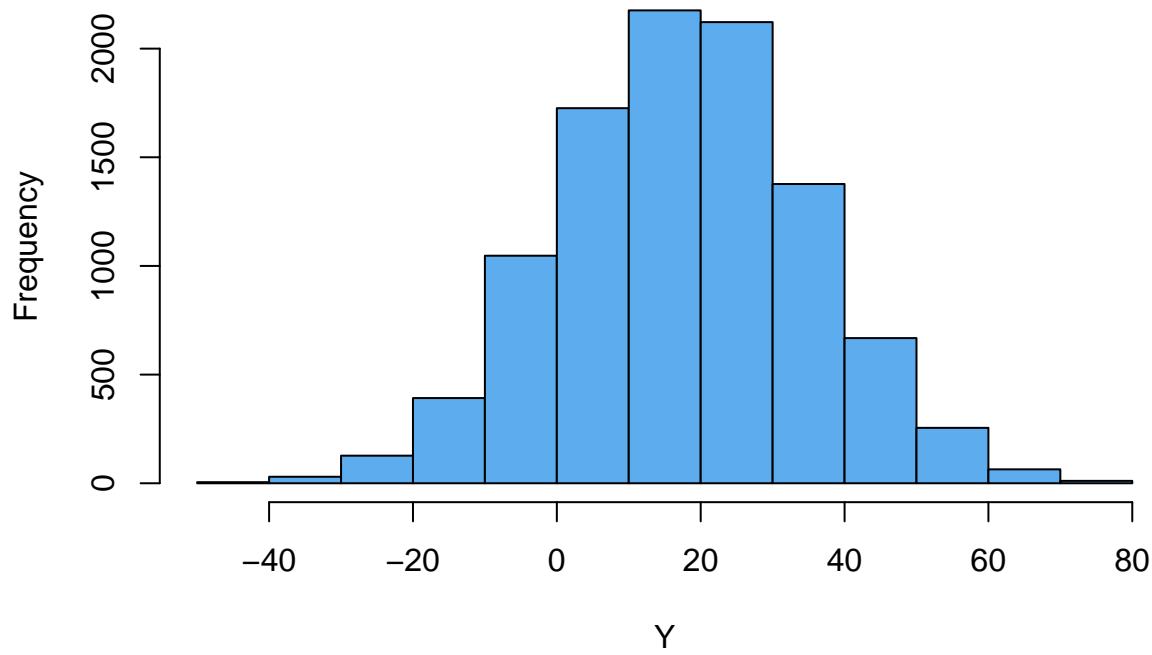
hist(X, col = "steelblue2")
```

### Histogram of X



```
Y <- rnorm(10000, mean = mu, sd = sd)  
hist(Y, col = "steelblue2")
```

### Histogram of Y



```
summary(Y)
```

```
##      Min. 1st Qu. Median     Mean 3rd Qu.    Max.  
## -44.017   5.652 17.684 17.572 29.359 79.942
```

Distribution of X showing almost uniform and distribution of Y showing mostly normal.

**II. 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.

**Solution**

```
x<-median(X)  
round(x,2)
```

```
## [1] 17.36
```

```
y<-quantile(Y,0.25)[[1]]  
round(y,2)
```

```
## [1] 5.65
```

Interpret the meaning of all probabilities.

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

**Solution**

$$P(X > x \mid X > y) = \frac{P(X > x, X > y)}{P(X > y)}$$

```
Pxxandxy <- sum(X>x & X>y)/10000  
Pxy <- sum(X>y)/10000  
Pxx_xy <- Pxxandxy/Pxy  
round(Pxx_xy,2)
```

```
## [1] 0.58
```

The probability of a random number uniformly ranging from 1 to 34 being greater than 17.36 given that it is greater than 5.65 is 0.58

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

**Solution**

```
Pxxyy<-sum(X>x & Y>y)/10000
round(Pxxyy,2)
```

```
## [1] 0.38
```

The probability of a random number uniformly ranging from 1 to 34 being greater than 17.36 and greater than 5.65 is 0.38

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

**Solution**

```
Pxx_and_xy<-sum(X<x & X>y)/10000
round(Pxx_and_xy,2)
```

```
## [1] 0.36
```

The probability of a random number uniformly ranging from 1 to 34 being less than 17.36 given that it is greater than 5.65 is 0.36

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

**Solution**

```
library(kableExtra)
Prob_X_x <- (length((X > x)[(X > x) == TRUE]))/(length((X > x)))
Prob_Y_y <- (length((Y > y)[(Y > y) == TRUE]))/(length((Y > y)))

X_x = c(Prob_X_x, Prob_Y_y, Prob_X_x*Prob_Y_y, Prob_X_x*Prob_Y_y)
Y_y = c(Prob_Y_y, Prob_X_x, Prob_X_x*Prob_Y_y, Prob_X_x*Prob_Y_y)

contingency_p <- as.data.frame(cbind(X_x, Y_y))
rownames(contingency_p) <- c("(X>x)", "(Y>y)", "(X>x)*(Y>y)", "(X>x and Y>y)")

kable(contingency_p) %>% kable_styling(bootstrap_options = c("striped", "hover", "condensed", "responsive"))
```

	X_x	Y_y
(X>x)	0.500	0.750
(Y>y)	0.750	0.500
(X>x)*(Y>y)	0.375	0.375
(X>x and Y>y)	0.375	0.375

Above table shows,  $P(X>x \text{ and } Y>y)$  and  $P(X>x)P(Y>y)$  are equal.

**IV. 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

We will use above created contingency table to perform these 2 tests - Fisher's and Chi Square test.

Null Hypothesis,  $H_0$ :  $X>x$  and  $Y>y$  are independent events

Alternate Hypothesis,  $H_A$ : Both of these are dependent events

```
fisher.test(contingency_p)
```

```
##  
## Fisher's Exact Test for Count Data  
##  
## data: contingency_p  
## p-value = 1  
## alternative hypothesis: two.sided
```

```
chisq.test(contingency_p)
```

```
##  
## Pearson's Chi-squared test  
##  
## data: contingency_p  
## X-squared = 0.1, df = 3, p-value = 0.9918
```

Both test showing high p-value so, we cannot reject null hypothesis that  $X>x$  and  $Y>y$  are independent events.

### 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.**

**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?

**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.

**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  $\lambda$  for this distribution, and then take 1000 samples from this exponential distribution using this value (e.g., `rexp(1000, λ)`). 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.

**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.

### Solution

Download train and test csv file from kaggle and load these in R. The goal of this analysis is predict SalesPrice. We will do below steps.

- Load data and perform EDA (Exploratory Data Analysis)
  - Descriptive and Inferential Statistics
  - Linear Algebra and Correlation
  - Calculus-Based Probability & Statistics
- Clean data
- Build Models
- Select best model to predict salesprice on test dataset

**Data Load and EDA** Load all necessary libraries.

```
library(ggplot2)
library(kableExtra)
library(visdat)
library(corrplot)
library(MASS)
library(purrr)
```

Train dataset consists of 1460 obsevations and 81 features. Test dataset consists of 1459 obsevations and 80 features.

```
train <-
  read.csv("https://raw.githubusercontent.com/SubhalaxmiRout002/DATA-605/master/Final%20Project/train.csv")
dim(train)

## [1] 1460    81

test <-
  read.csv("https://raw.githubusercontent.com/SubhalaxmiRout002/DATA-605/master/Final%20Project/test.csv")
dim(test)

## [1] 1459    80
```

```
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  : chr "RL" "RL" "RL" "RL" ...
##   $ 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    : chr "Pave" "Pave" "Pave" "Pave" ...
##   $ Alley     : chr NA NA NA NA ...
##   $ LotShape  : chr "Reg" "Reg" "IR1" "IR1" ...
##   $ LandContour: chr "Lvl" "Lvl" "Lvl" "Lvl" ...
##   $ Utilities : chr "AllPub" "AllPub" "AllPub" "AllPub" ...
##   $ LotConfig  : chr "Inside" "FR2" "Inside" "Corner" ...
##   $ LandSlope  : chr "Gtl" "Gtl" "Gtl" "Gtl" ...
##   $ Neighborhood: chr "CollgCr" "Veenker" "CollgCr" "Crawfor" ...
##   $ Condition1 : chr "Norm" "Feedr" "Norm" "Norm" ...
##   $ Condition2 : chr "Norm" "Norm" "Norm" "Norm" ...
##   $ BldgType   : chr "1Fam" "1Fam" "1Fam" "1Fam" ...
##   $ HouseStyle : chr "2Story" "1Story" "2Story" "2Story" ...
##   $ 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  : chr "Gable" "Gable" "Gable" "Gable" ...
##   $ RoofMatl   : chr "CompShg" "CompShg" "CompShg" "CompShg" ...
##   $ Exterior1st : chr "VinylSd" "MetalSd" "VinylSd" "Wd Sdng" ...
##   $ Exterior2nd : chr "VinylSd" "MetalSd" "VinylSd" "Wd Shng" ...
##   $ MasVnrType : chr "BrkFace" "None" "BrkFace" "None" ...
##   $ MasVnrArea : int 196 0 162 0 350 0 186 240 0 0 ...
##   $ ExterQual  : chr "Gd" "TA" "Gd" "TA" ...
##   $ ExterCond  : chr "TA" "TA" "TA" "TA" ...
##   $ Foundation : chr "PConc" "CBlock" "PConc" "BrkTil" ...
##   $ BsmtQual   : chr "Gd" "Gd" "Gd" "TA" ...
##   $ BsmtCond   : chr "TA" "TA" "TA" "Gd" ...
##   $ BsmtExposure: chr "No" "Gd" "Mn" "No" ...
##   $ BsmtFinType1: chr "GLQ" "ALQ" "GLQ" "ALQ" ...
##   $ BsmtFinSF1  : int 706 978 486 216 655 732 1369 859 0 851 ...
##   $ BsmtFinType2: chr "Unf" "Unf" "Unf" "Unf" ...
##   $ BsmtFinSF2  : int 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     : chr "GasA" "GasA" "GasA" "GasA" ...
##   $ HeatingQC   : chr "Ex" "Ex" "Ex" "Gd" ...
##   $ CentralAir  : chr "Y" "Y" "Y" "Y" ...
##   $ Electrical  : chr "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
##   $ 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 0 1 ...
##   $ BsmtHalfBath: int 0 1 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   : chr "Gd" "TA" "Gd" "Gd" ...
## $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
## $ Functional    : chr "Typ" "Typ" "Typ" "Typ" ...
## $ Fireplaces    : int  0 1 1 1 1 0 1 2 2 2 ...
## $ FireplaceQu   : chr NA "TA" "TA" "Gd" ...
## $ GarageType    : chr "Attchd" "Attchd" "Attchd" "Detchd" ...
## $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
## $ GarageFinish   : chr "RFn" "RFn" "RFn" "Unf" ...
## $ 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    : chr "TA" "TA" "TA" "TA" ...
## $ GarageCond    : chr "TA" "TA" "TA" "TA" ...
## $ PavedDrive    : chr "Y" "Y" "Y" "Y" ...
## $ 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        : chr NA NA NA NA ...
## $ Fence          : chr NA NA NA NA ...
## $ MiscFeature   : chr NA NA 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       : chr "WD" "WD" "WD" "WD" ...
## $ SaleCondition : chr "Normal" "Normal" "Normal" "Abnorml" ...
## $ SalePrice     : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...

```

```
summary(train)
```

## Descriptive and Inferential Statistics

```

##      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   NA's   :259       Mean   : 70.05
## 3rd Qu.:1095.2 3rd Qu.:70.0   NA's   :259       3rd Qu.: 80.00
## Max.   :1460.0  Max.   :190.0  NA's   :259       Max.   :313.00
## 
##      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        X1stFlrSF
##  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
##
##      X2ndFlrSF          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
##  3rd Qu.:1.000
##  Max.   :3.000
##
##      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
##  3rd Qu.:1.000
##  Max.   :3.000
##
##      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 X3SsnPorch 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
##

```

Provide univariate descriptive statistics and appropriate plots for the training data set.

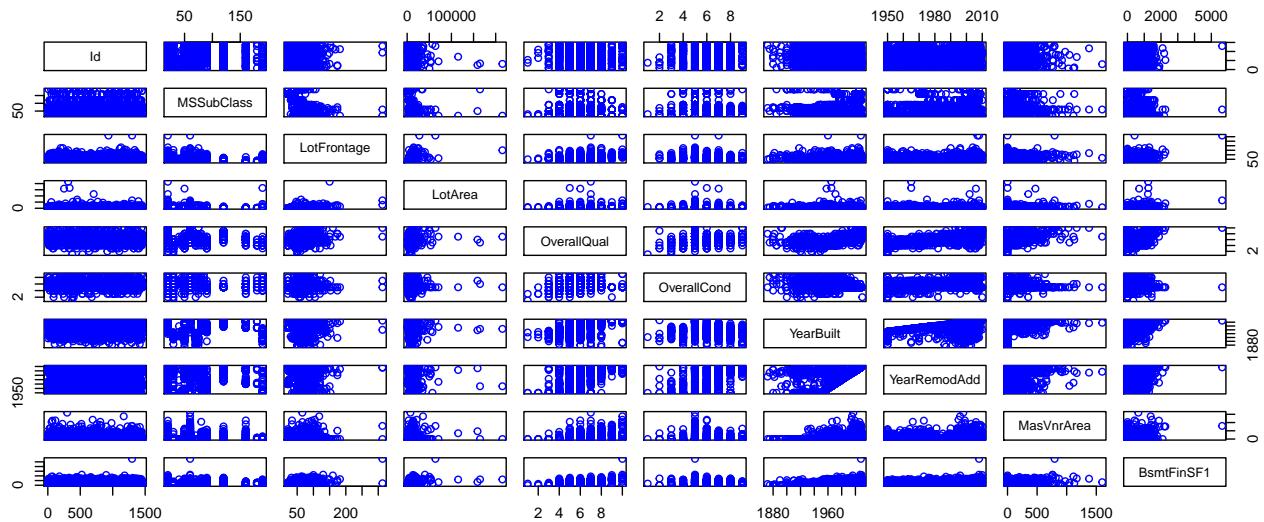
Pairplot shows the relationship between variables.

```

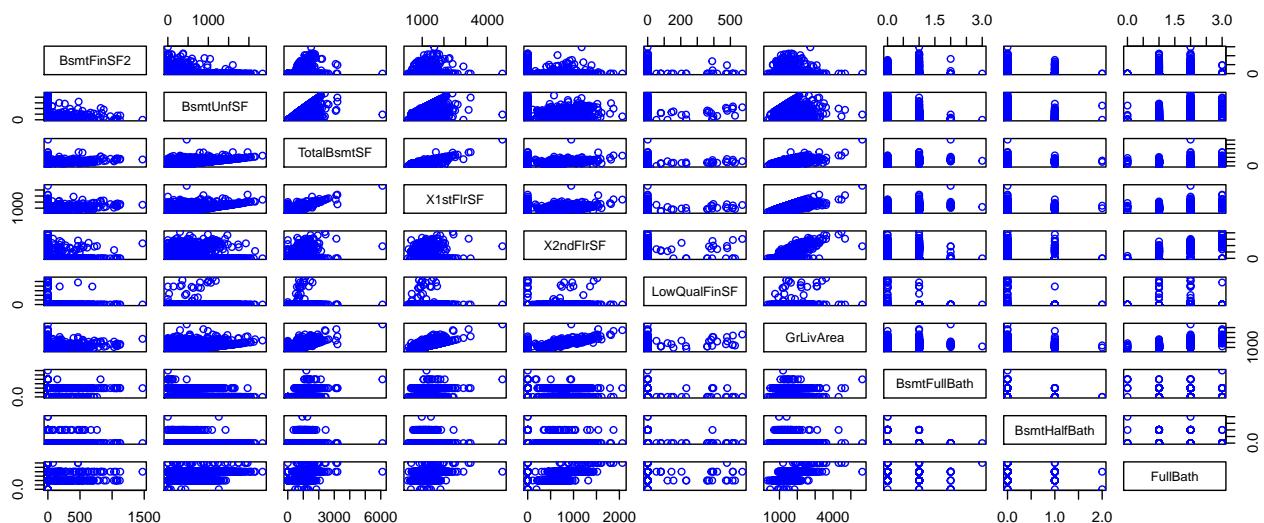
train_2 <- dplyr::select_if(train, is.numeric) %>% keep(is.numeric)

plot(train_2[1:10], col="blue")

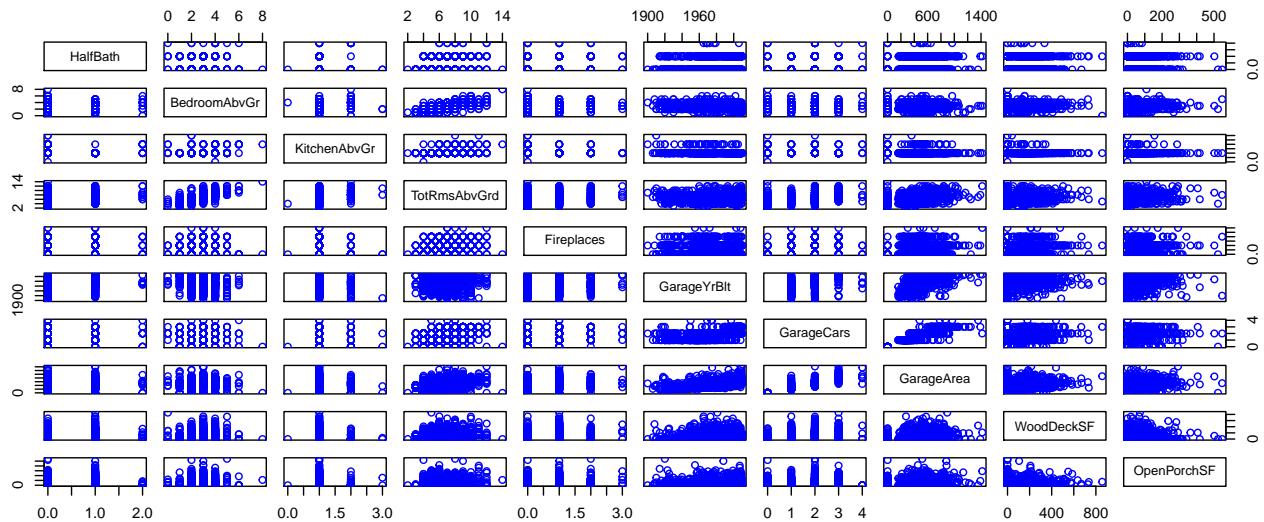
```



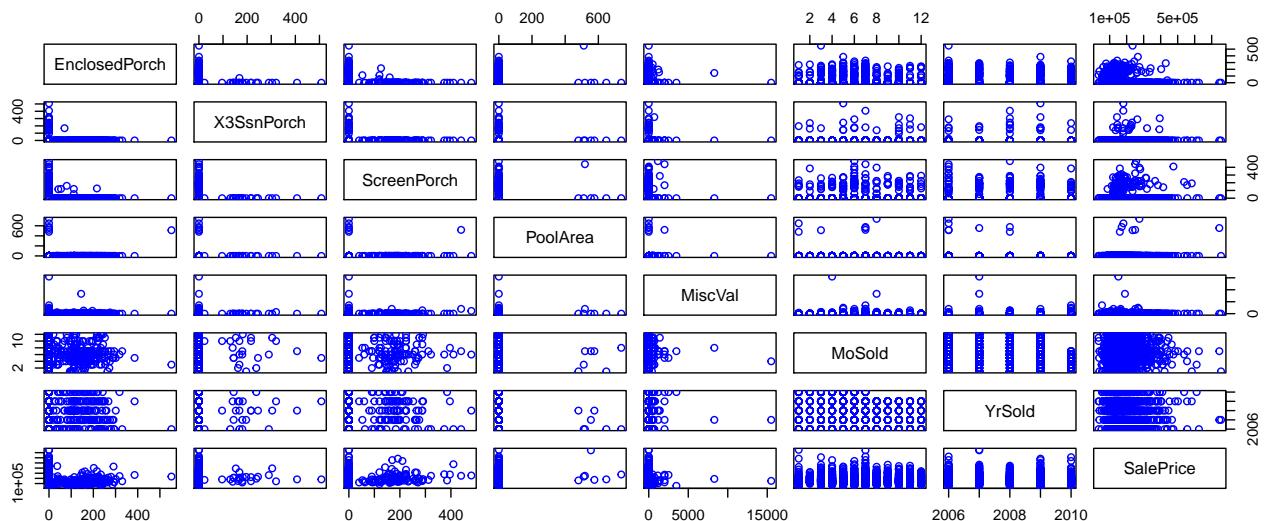
```
plot(train_2[11:20], col="blue")
```



```
plot(train_2[21:30], col="blue")
```

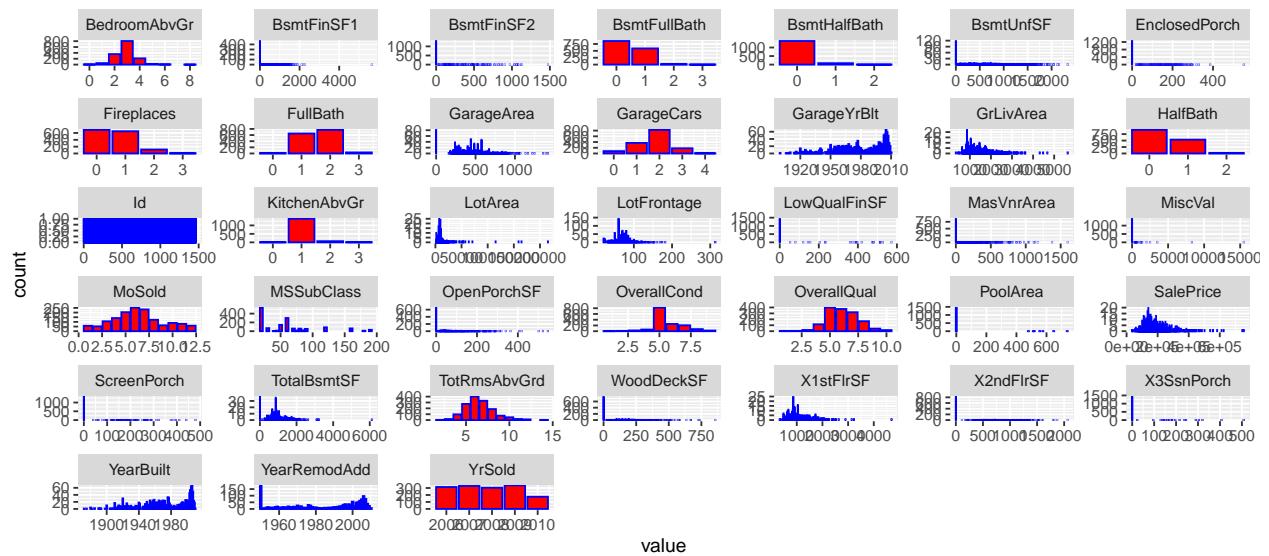


```
plot(train_2[31:38], col="blue")
```



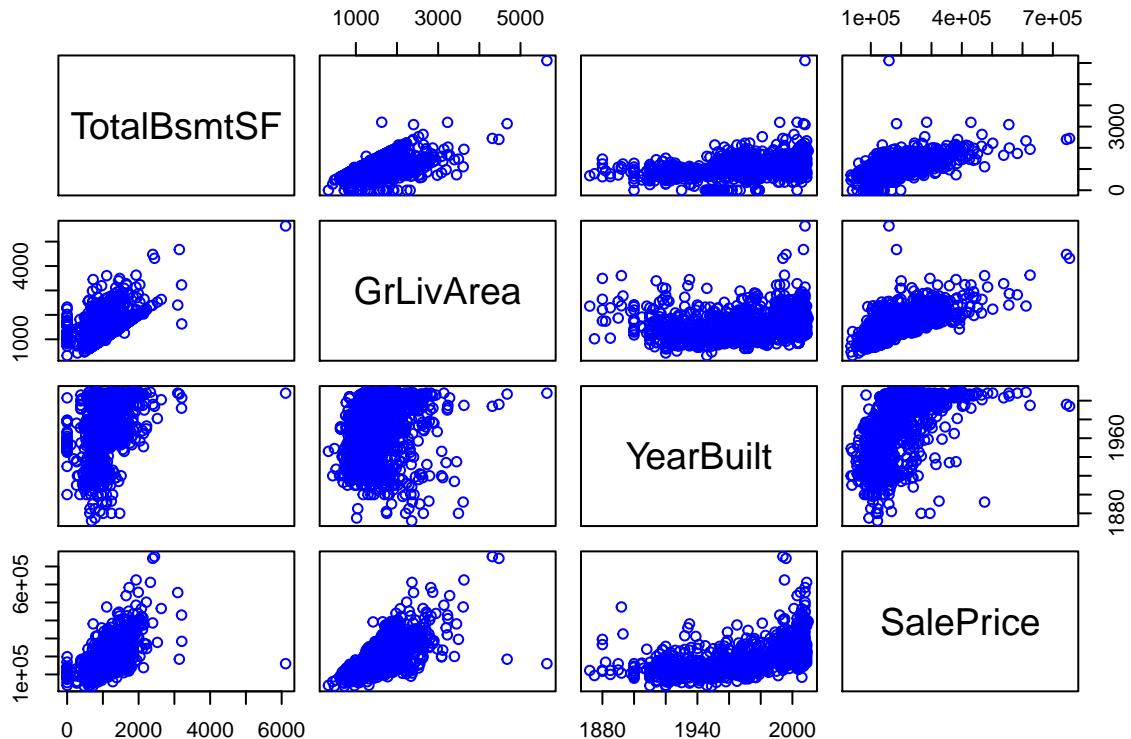
Below plot shows the distribution of data.

```
ggplot(dplyr::select_if(train, is.numeric) %>% keep(is.numeric) %>% tidyrr::gather() , aes(value)) +
  facet_wrap(~ key, scales = "free") +
  geom_bar(color = "blue", fill = "red")
```



Provide a scatterplot matrix for at least two of the independent variables and the dependent variable.

```
plot(train %>%
  dplyr::select(TotalBsmtSF, GrLivArea, YearBuilt, SalePrice), col="blue")
```



Derive a correlation matrix for any three quantitative variables in the dataset.

```
correlation_matrix <- train %>%
  dplyr::select(TotalBsmtSF, GrLivArea, YearBuilt) %>%
  cor() %>%
```

```

as.matrix()

kable(correlation_matrix) %>% kable_styling(bootstrap_options = c("striped", "hover", "condensed", "respo

```

	TotalBsmtSF	GrLivArea	YearBuilt
TotalBsmtSF	1.0000000	0.4548682	0.3914520
GrLivArea	0.4548682	1.0000000	0.1990097
YearBuilt	0.3914520	0.1990097	1.0000000

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

```
cor.test(train$TotalBsmtSF, train$SalePrice, method = "pearson", conf.level = 0.8)
```

```
##
## Pearson's product-moment correlation
##
## data: train$TotalBsmtSF and train$SalePrice
## t = 29.671, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
## 0.5922142 0.6340846
## sample estimates:
## cor
## 0.6135806
```

```
cor.test(train$GrLivArea, train$SalePrice, method = "pearson", conf.level = 0.8)
```

```
##
## Pearson's product-moment correlation
##
## data: train$GrLivArea and train$SalePrice
## t = 38.348, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
## 0.6915087 0.7249450
## sample estimates:
## cor
## 0.7086245
```

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

```
##
## Pearson's product-moment correlation
##
## data: train$YearBuilt and train$SalePrice
```

```

## t = 23.424, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 80 percent confidence interval:
## 0.4980766 0.5468619
## sample estimates:
## cor
## 0.5228973

```

The p-values for all three variables are less than 0.05 and the correlation coefficient between TotalBsmtSF, GrLivArea, YearBuilt with respect to SalePrice is 0.6135806, 0.7086245 , 0.5228973 respectively. The correlation falls within the 80% CI.

**Would you be worried about familywise error? Why or why not?**

Familywise error:

The familywise error rate is the probability of coming to at least one false conclusion in a series of hypothesis tests. Familywise error is that of making at least one “type I error”, or a false positive, rejection of a true null.

In our above 3 cases, the P value is extremely small (is less than 0.05) and rejects null hypothesis. So, I would not be worried about committing a type I 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.)

```

library(pracma)

precision_matrix <- pracma::inv(correlation_matrix)
kable(precision_matrix) %>% kable_styling(bootstrap_options = c("striped", "hover", "condensed", "responsive"))

```

	TotalBsmtSF	GrLivArea	YearBuilt
TotalBsmtSF	1.4310199	-0.5616907	-0.4483937
GrLivArea	-0.5616907	1.2617078	-0.0312171
YearBuilt	-0.4483937	-0.0312171	1.1817371

Multiply the correlation matrix by the precision matrix, and then multiply the precision matrix by the correlation matrix.

```
kable(round(correlation_matrix %*% precision_matrix)) %>% kable_styling(bootstrap_options = c("striped"))
```

	TotalBsmtSF	GrLivArea	YearBuilt
TotalBsmtSF	1	0	0
GrLivArea	0	1	0
YearBuilt	0	0	1

```
kable(round(precision_matrix %*% correlation_matrix)) %>% kable_styling(bootstrap_options = c("striped"))
```

	TotalBsmtSF	GrLivArea	YearBuilt
TotalBsmtSF	1	0	0
GrLivArea	0	1	0
YearBuilt	0	0	1

### Conduct LU decomposition on the matrix.

Below find out L and U matrix, and as per the LU decomposition,  $LU = A$  Here A is the correlation matrix we created above. So, if we multiply L and U above, it should give correlation matrix.

```
correlation_L <- lu(correlation_matrix)$L
correlation_U <- lu(correlation_matrix)$U

kable(correlation_L %*% correlation_U) %>%
  kable_styling(bootstrap_options =
    c("striped", "hover", "condensed", "responsive"),
    latex_options="scale_down")
```

	TotalBsmtSF	GrLivArea	YearBuilt
TotalBsmtSF	1.0000000	0.4548682	0.3914520
GrLivArea	0.4548682	1.0000000	0.1990097
YearBuilt	0.3914520	0.1990097	1.0000000

```
correlation_matrix == correlation_L %*% correlation_U
```

```
##          TotalBsmtSF GrLivArea YearBuilt
## TotalBsmtSF      TRUE      TRUE      TRUE
## GrLivArea        TRUE      TRUE      TRUE
## YearBuilt        TRUE      TRUE      TRUE
```

### Calculus-Based Probability & Statistics

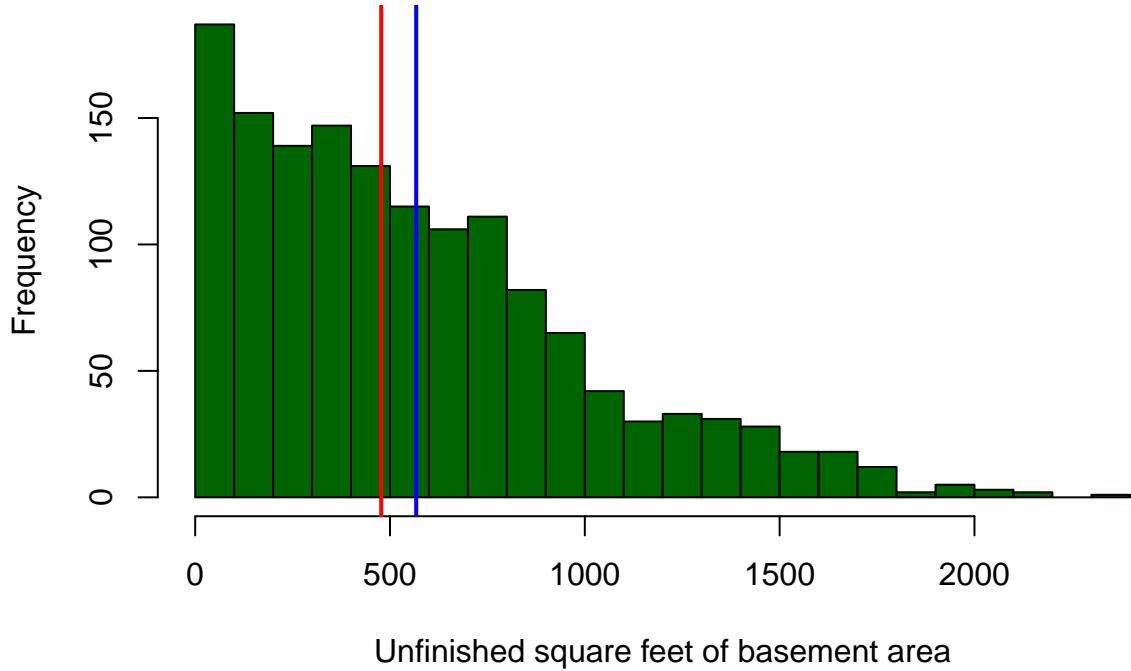
Many times, it makes sense to fit a closed form distribution to data. Select a variable in the Kaggle.com training dataset that is skewed to the right, shift it so that the minimum value is absolutely above zero if necessary. Then load the MASS package and run fitdistr to fit an exponential probability density function. (See <https://stat.ethz.ch/R-manual/R-devel/library/MASS/html/fitdistr.html> ). Find the optimal value of  $\lambda$  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.

```

hist(train$BsmtUnfSF, main = "Histogram of BsmtUnfSF"
  , col = "darkgreen",
  xlab = "Unfinished square feet of basement area", breaks = 20)
abline(v = mean(train$BsmtUnfSF), col = "blue", lwd = 2)
abline(v = median(train$BsmtUnfSF), col = "red", lwd = 2)

```

## Histogram of BsmtUnfSF

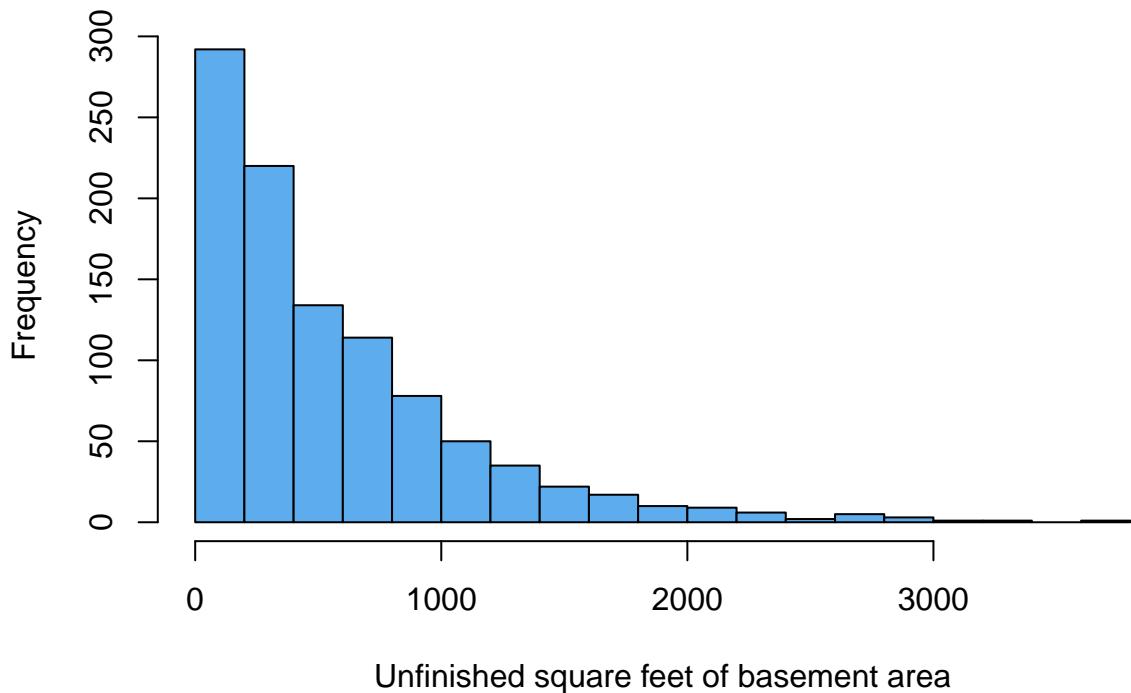


```

lambda <- fitdistr(train$BsmtUnfSF,"exponential")[[1]]
sample <- rexp(1000, lambda)
hist(sample , main="Histogram",xlab="Unfinished square feet of basement area",
  breaks = 20, col = "steelblue2")

```

## Histogram



```
summary(sample)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
## 0.556 169.404 383.793 556.042 782.351 3773.592
```

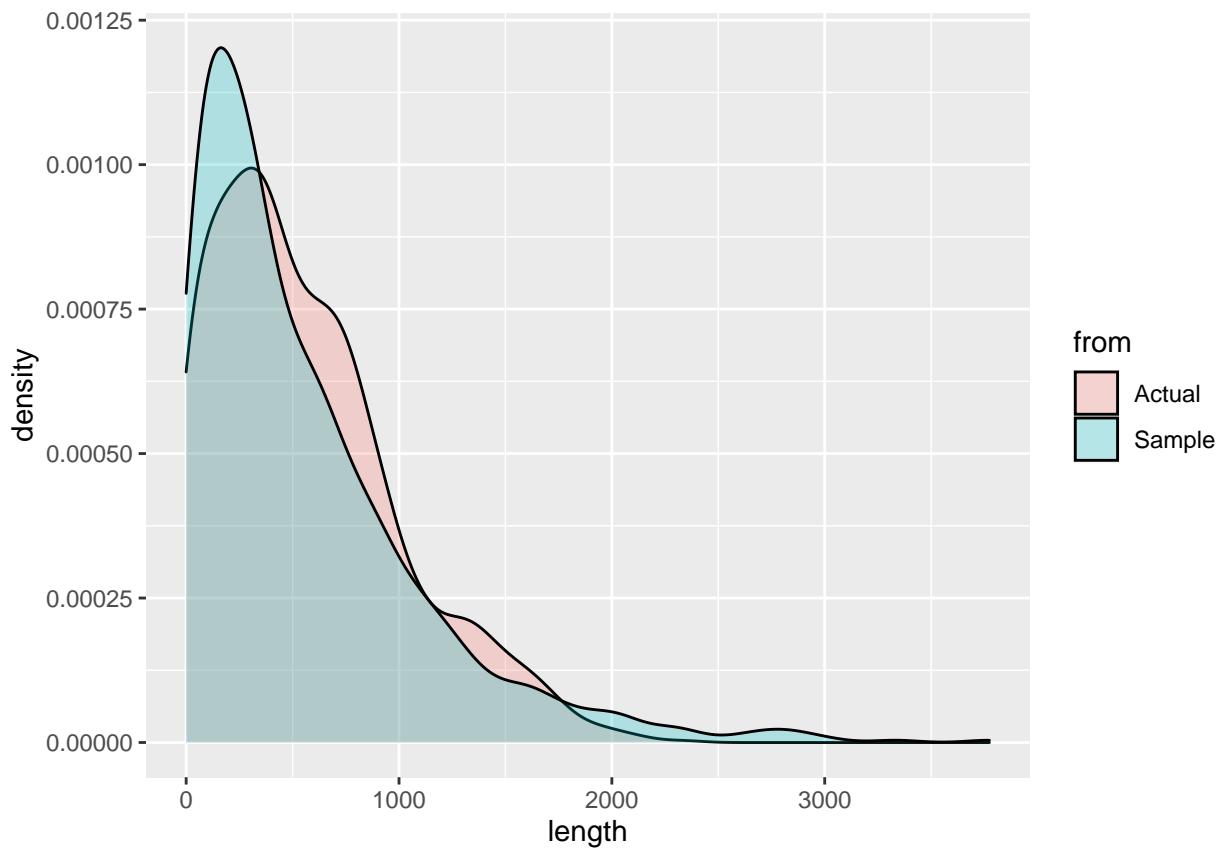
The optimal value for  $\lambda$  is 0.0017629.

```
sample.df <- data.frame(length = sample)
Actual.df <- data.frame(length = train$BsmtUnfSF)

sample.df$from <- 'Sample'
Actual.df$from <- 'Actual'

both.df <- rbind(sample.df, Actual.df)

ggplot(both.df, aes(length, fill = from)) + geom_density(alpha = 0.25)
```



The exponential distribution data has a longer tail than the basement square footage data.

**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.**

Generating the 5th and 95th percentiles

```
qexp(c(0.05,0.95), lambda)
```

```
## [1] 29.09563 1699.30041
```

Generating 95% confidence level from the data, assuming that the distribution is normal.

```
qnorm(c(0.05, 0.95), mean = mean(train$BsmtUnfSF), sd = sd(train$BsmtUnfSF))
```

```
## [1] -159.5661 1294.0469
```

Now using the actual data to get 5th and 95th percentile

```
quantile(train$BsmtUnfSF, c(0.05, 0.95))
```

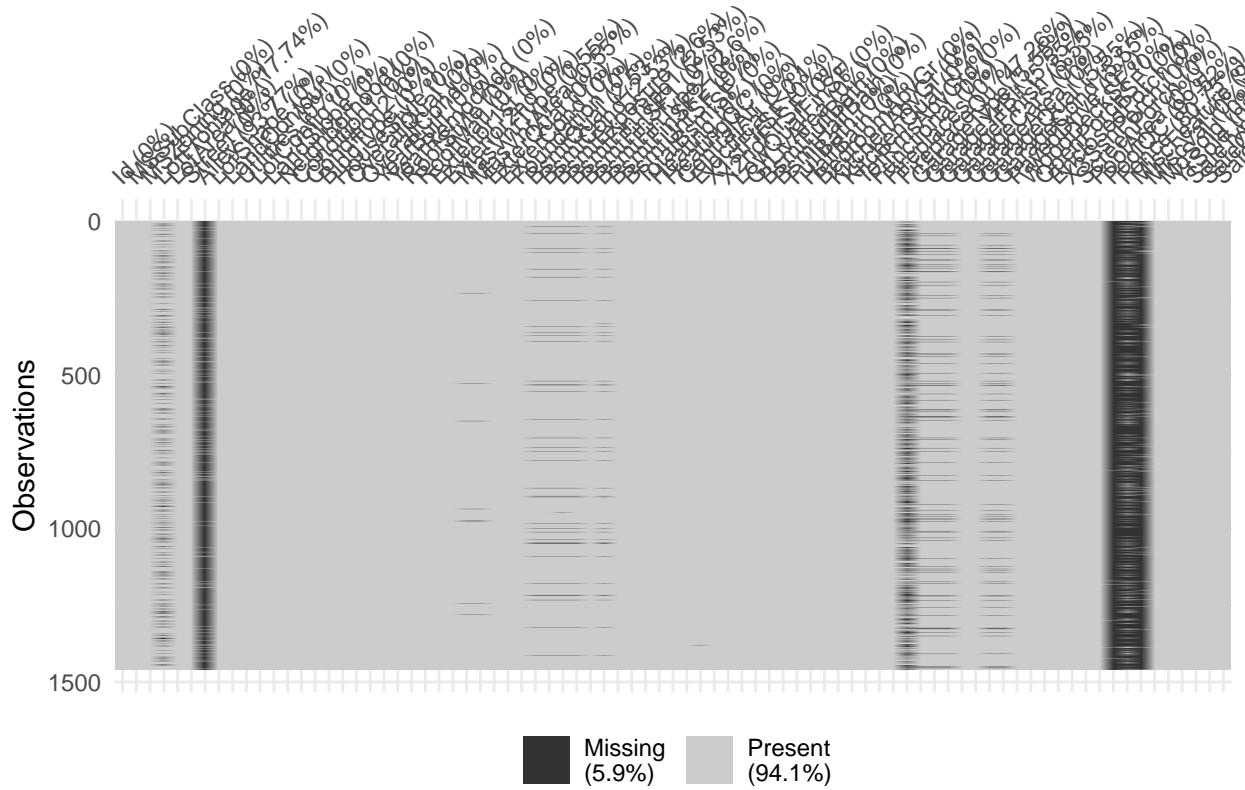
```
##      5%    95%
##      0 1468
```

If we model the data as exponentially distributed the 5th percentile is 29.0956294. If we model it as normally distributed the 5th is at -159.5660531. Actual data distributed the 5th percentile is 0. If we look at the 95th percentile we have 1699.300406 if the data are exponentially distributed, 1294.046875 if it is normally distributed and 1468 in actual data.

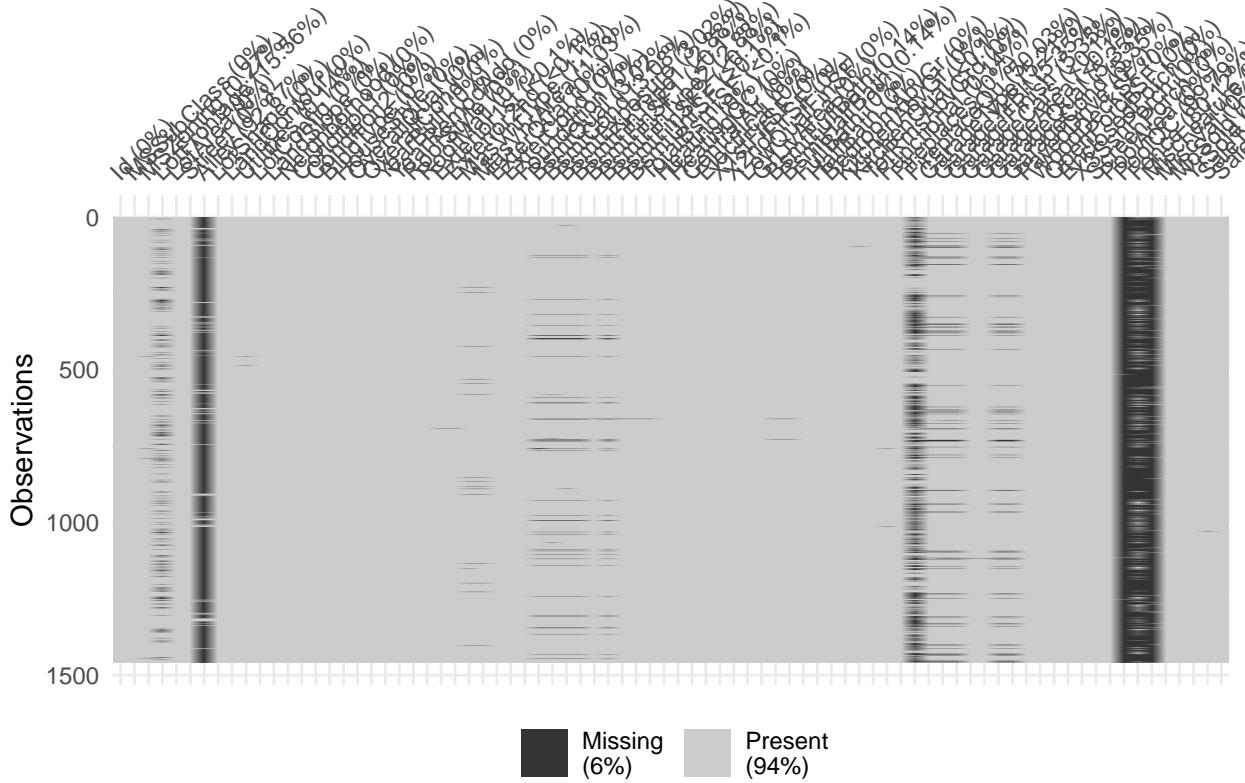
## Data Cleaning

To check null values, if available replace with 0.

```
visdat::vis_miss(train)
```



```
visdat::vis_miss(test)
```



```

train$LotFrontage[is.na(train$LotFrontage)] <- median(train$LotFrontage,na.rm=T)
train$GarageYrBlt[is.na(train$GarageYrBlt)] <- median(train$GarageYrBlt,na.rm=T)
train$MasVnrArea[is.na(train$MasVnrArea)] <- median(train$MasVnrArea,na.rm=T)

train$LotFrontage <- as.integer(train$LotFrontage)
train$MasVnrArea <- as.integer(train$MasVnrArea)
train$TotalBsmtSF <- as.integer(train$TotalBsmtSF)
train$GrLivArea <- as.integer(train$GrLivArea)
train$GarageYrBlt <- as.integer(train$GarageYrBlt)

test$LotFrontage[is.na(test$LotFrontage)] <- median(test$LotFrontage,na.rm=T)
test$GarageYrBlt[is.na(test$GarageYrBlt)] <- median(test$GarageYrBlt,na.rm=T)
test$MasVnrArea[is.na(test$MasVnrArea)] <- median(test$MasVnrArea,na.rm=T)
test$LotFrontage[is.na(test$LotFrontage)] <- median(test$LotFrontage,na.rm=T)
test$BsmtFinSF1[is.na(test$BsmtFinSF1)] <- median(test$BsmtFinSF1,na.rm=T)
test$BsmtFullBath[is.na(test$BsmtFullBath)] <- median(test$BsmtFullBath,na.rm=T)
test$BsmtHalfBath[is.na(test$BsmtHalfBath)] <- median(test$BsmtHalfBath,na.rm=T)
test$GarageYrBlt[is.na(test$GarageYrBlt)] <- median(test$GarageYrBlt,na.rm=T)
test$GarageCars[is.na(test$GarageCars)] <- median(test$GarageCars,na.rm=T)
test$BsmtFinSF2[is.na(test$BsmtFinSF2)] <- median(test$BsmtFinSF2,na.rm=T)
test$BsmtUnfSF[is.na(test$BsmtUnfSF)] <- median(test$BsmtUnfSF,na.rm=T)
test$TotalBsmtSF[is.na(test$TotalBsmtSF)] <- median(test$TotalBsmtSF,na.rm=T)
test$GarageArea[is.na(test$GarageArea)] <- median(test$GarageArea,na.rm=T)

test$LotFrontage <- as.integer(test$LotFrontage)
test$MasVnrArea <- as.integer(test$MasVnrArea)

```

```

test$TotalBsmtSF <- as.integer(test$TotalBsmtSF)
test$GrLivArea <- as.integer(test$GrLivArea)
test$GarageYrBlt <- as.integer(test$GarageYrBlt)

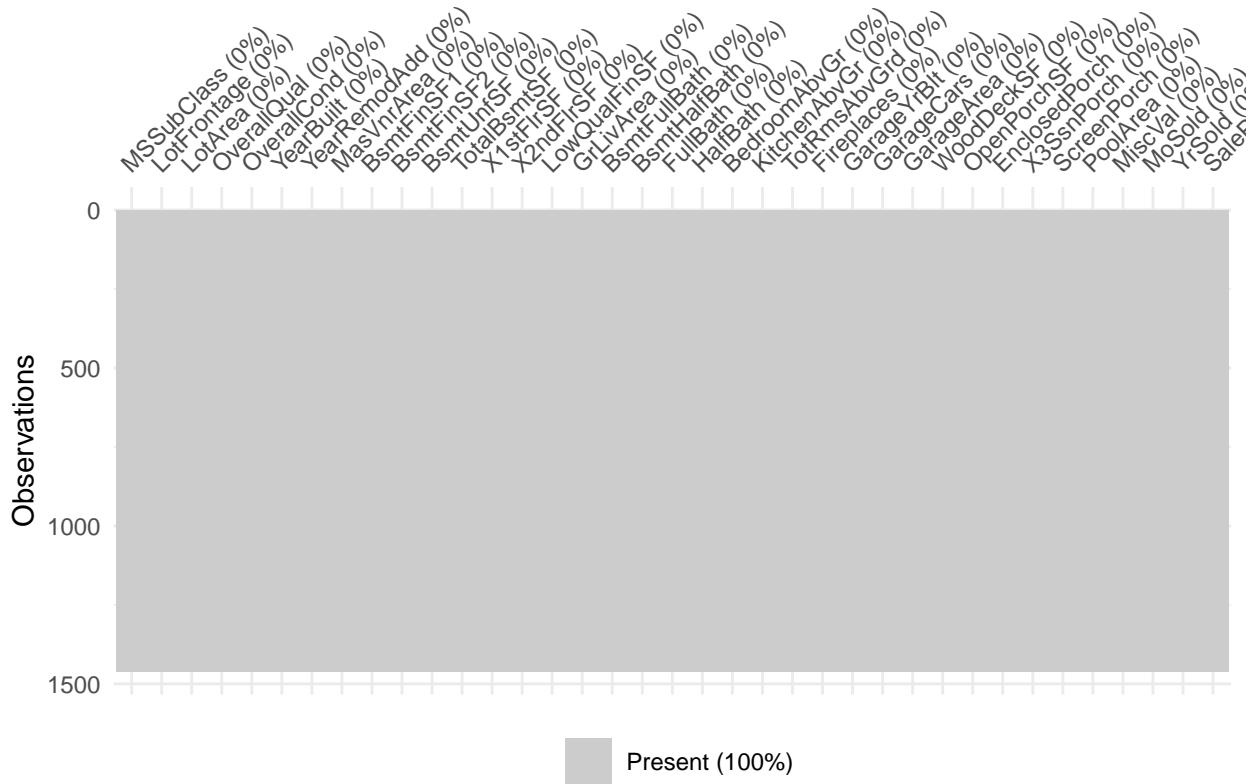
```

Dropping variables with character types.

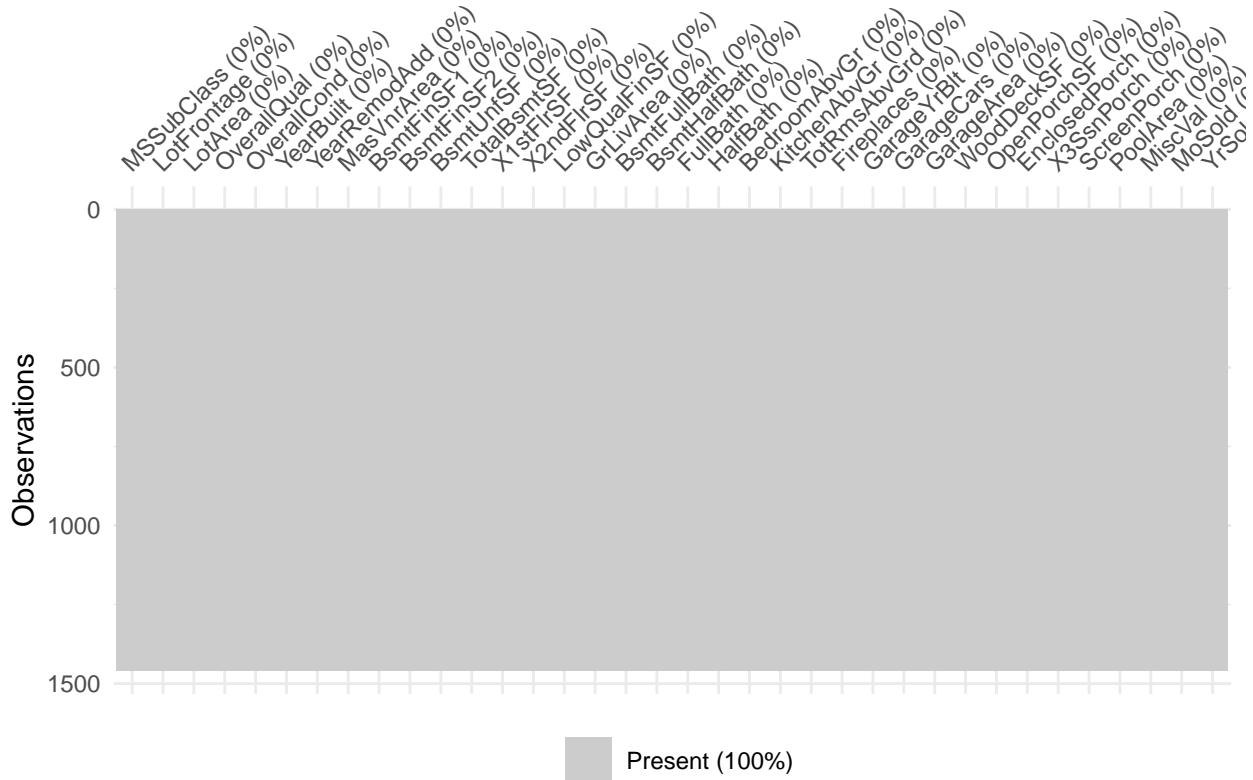
```

train_4 <- train %>%
  dplyr::select(-Id, -Alley, -MasVnrType, -BsmtQual, -BsmtCond, -BsmtExposure, -BsmtFinType1, -BsmtFinT
  test_4 <- test %>%
  dplyr::select(-Id, -Alley, -MasVnrType, -BsmtQual, -BsmtCond, -BsmtExposure, -BsmtFinType1, -BsmtFinT
  visdat::vis_miss(train_4)

```



```
visdat::vis_miss(test_4)
```



Model1 with all variables

```
model1 <- lm(SalePrice ~ ., data = train_4)
summary(model1)
```

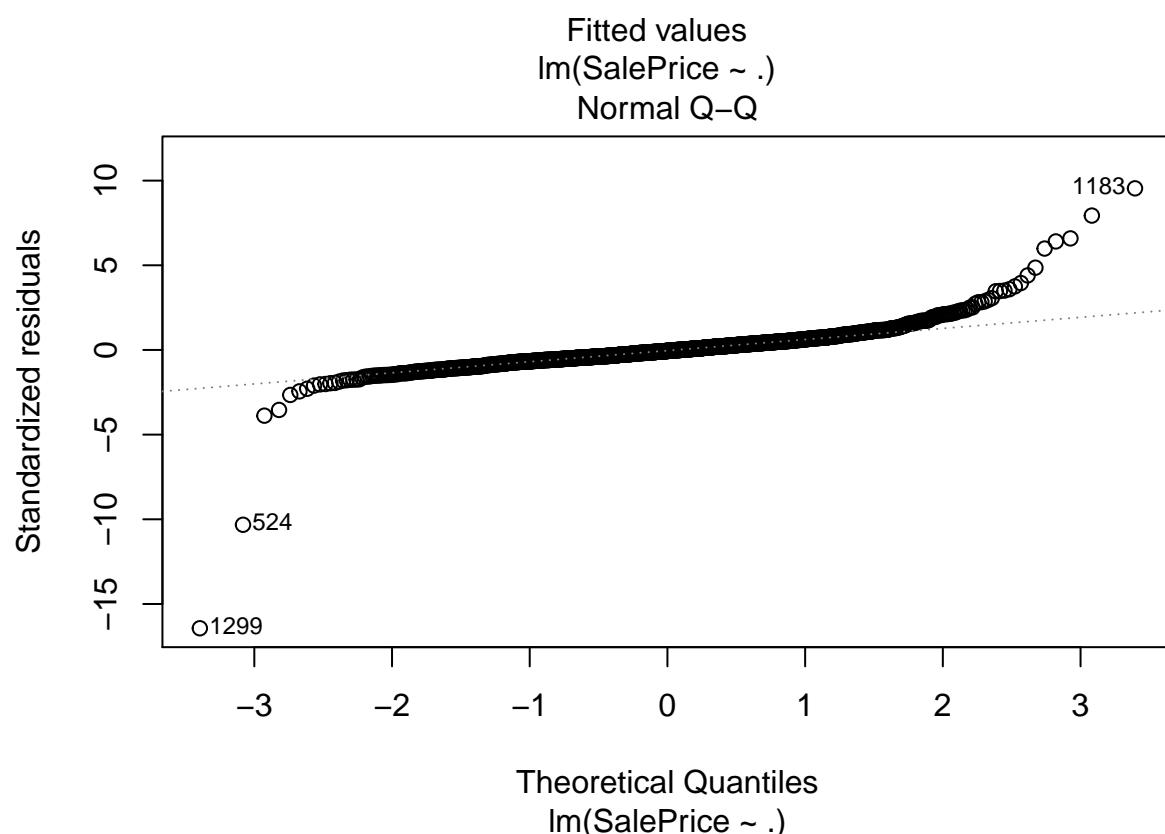
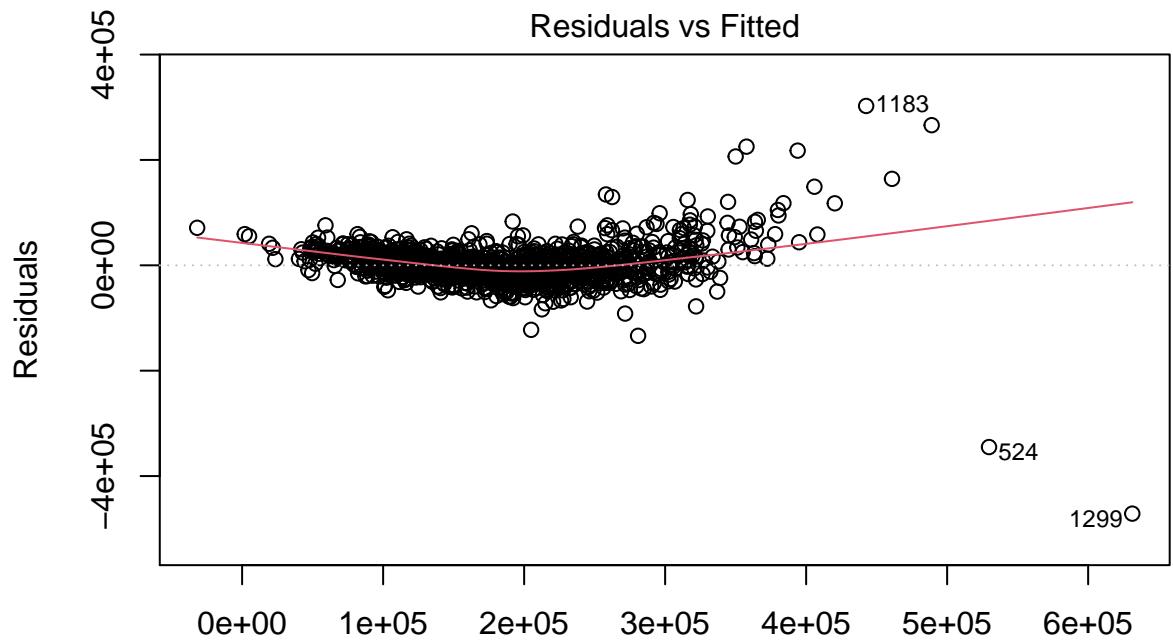
```
##
## Call:
## lm(formula = SalePrice ~ ., data = train_4)
##
## Residuals:
##     Min      1Q      Median      3Q      Max
## -471312 -16510   -2011    13793  302440
##
## Coefficients: (2 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.628e+05 1.414e+06  0.327 0.743435
## MSSubClass -1.817e+02 2.767e+01 -6.566 7.22e-11 ***
## LotFrontage -5.444e+01 5.171e+01 -1.053 0.292680
## LotArea     4.297e-01 1.021e-01  4.209 2.72e-05 ***
## OverallQual 1.733e+04 1.187e+03 14.597 < 2e-16 ***
## OverallCond 4.674e+03 1.033e+03  4.526 6.51e-06 ***
## YearBuilt    2.692e+02 6.741e+01  3.993 6.85e-05 ***
## YearRemodAdd 1.346e+02 6.859e+01  1.962 0.049921 *
## MasVnrArea   3.134e+01 5.933e+00  5.283 1.47e-07 ***
## BsmtFinSF1   1.921e+01 4.667e+00  4.116 4.07e-05 ***
## BsmtFinSF2   8.274e+00 7.057e+00  1.172 0.241220
## BsmtUnfSF   9.304e+00 4.194e+00  2.218 0.026687 *
## TotalBsmtSF          NA          NA          NA          NA
```

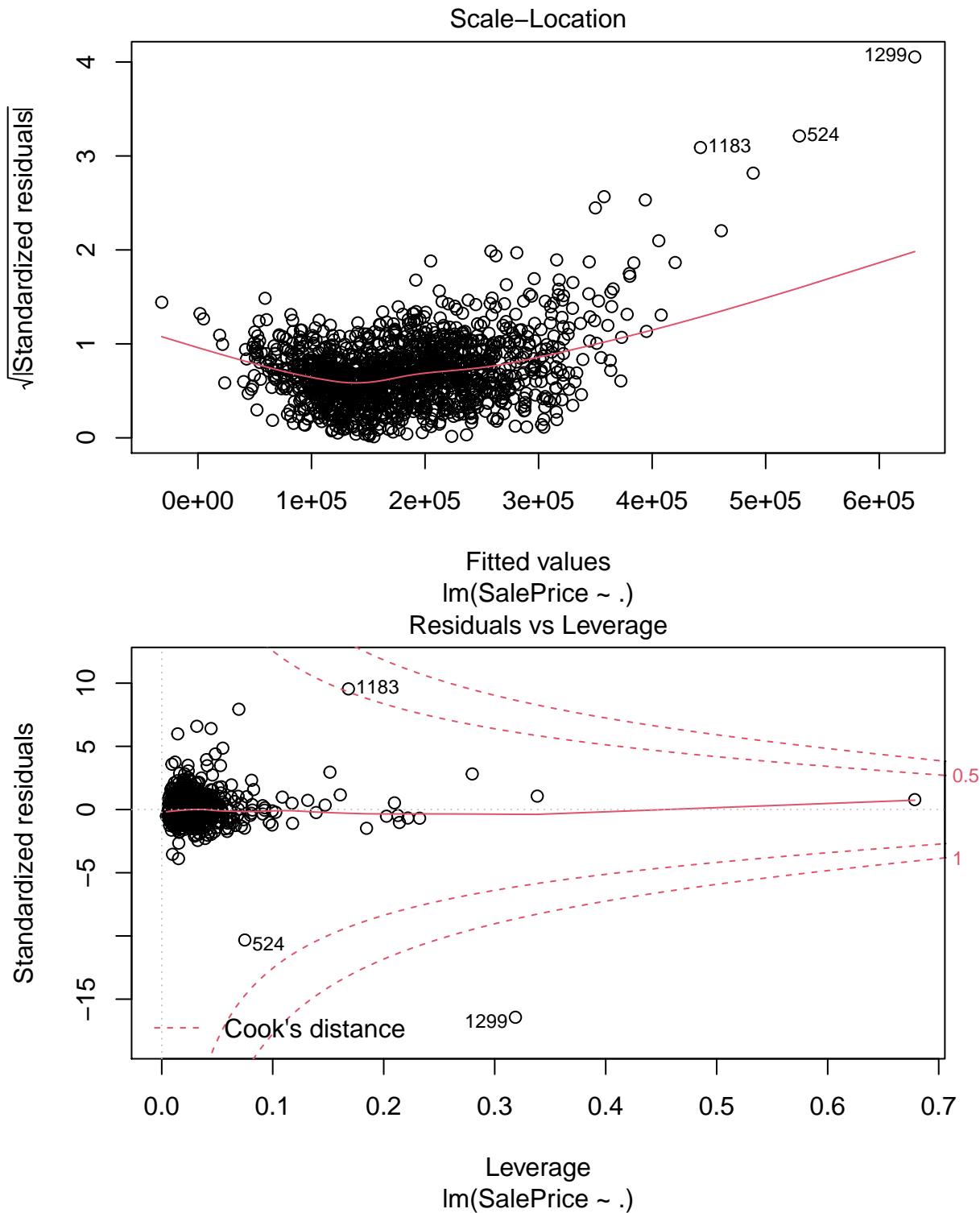
```

## X1stFlrSF      4.898e+01  5.809e+00   8.432 < 2e-16 ***
## X2ndFlrSF      4.901e+01  4.983e+00   9.835 < 2e-16 ***
## LowQualFinSF   2.530e+01  1.997e+01   1.267 0.205442
## GrLivArea       NA          NA          NA          NA
## BsmtFullBath   9.372e+03  2.612e+03   3.588 0.000344 ***
## BsmtHalfBath   2.055e+03  4.091e+03   0.502 0.615546
## FullBath        3.447e+03  2.837e+03   1.215 0.224513
## HalfBath        -1.872e+03 2.663e+03  -0.703 0.482212
## BedroomAbvGr   -1.009e+04 1.702e+03  -5.931 3.77e-09 ***
## KitchenAbvGr   -1.217e+04 5.212e+03  -2.335 0.019676 *
## TotRmsAbvGrd   5.045e+03  1.237e+03   4.078 4.79e-05 ***
## Fireplaces      3.979e+03  1.777e+03   2.239 0.025278 *
## GarageYrBlt    1.269e+02  6.898e+01   1.840 0.065965 .
## GarageCars      1.129e+04  2.876e+03   3.927 9.03e-05 ***
## GarageArea      -4.410e+00 9.941e+00  -0.444 0.657435
## WoodDeckSF     2.389e+01  8.012e+00   2.982 0.002914 **
## OpenPorchSF    -2.891e+00 1.518e+01  -0.190 0.848996
## EnclosedPorch   1.191e+01  1.686e+01   0.706 0.480195
## X3SsnPorch     2.032e+01  3.139e+01   0.647 0.517617
## ScreenPorch     5.599e+01  1.719e+01   3.257 0.001153 **
## PoolArea        -2.919e+01 2.381e+01  -1.226 0.220457
## MiscVal         -7.336e-01 1.855e+00  -0.395 0.692545
## MoSold          -4.840e+01 3.448e+02  -0.140 0.888392
## YrSold          -7.807e+02 7.025e+02  -1.111 0.266589
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34750 on 1425 degrees of freedom
## Multiple R-squared:  0.8132, Adjusted R-squared:  0.8087
## F-statistic: 182.4 on 34 and 1425 DF,  p-value: < 2.2e-16

```

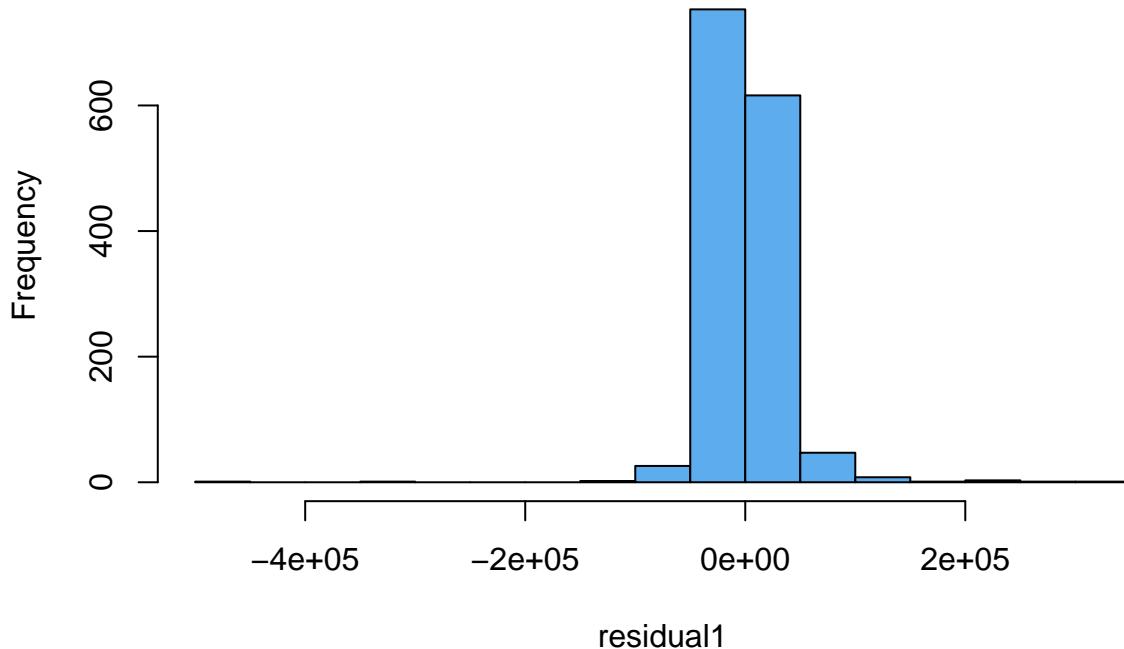
```
plot(model1)
```



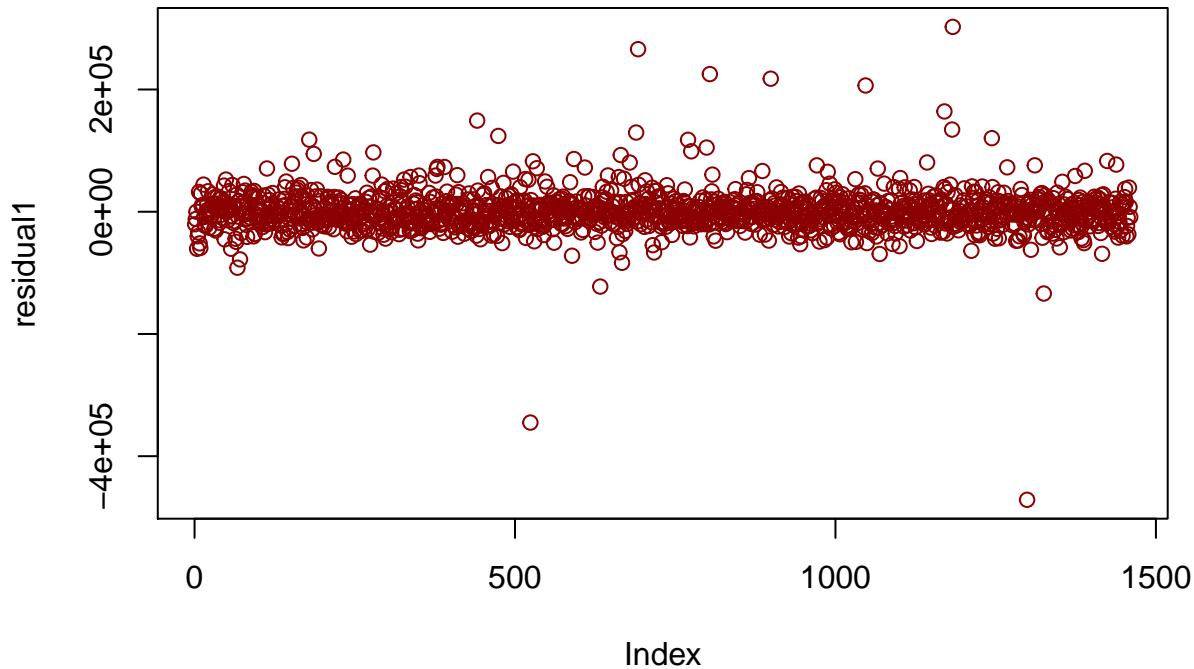


```
residual1 <- resid(model1)
hist(residual1, col = "steelblue2")
```

### Histogram of residual1



```
plot(residual1, col = 'dark red')
```



Model2 with significant variables

```
model2 <- lm(SalePrice ~ MSSubClass + LotArea + OverallQual + OverallCond +  
YearBuilt + YearRemodAdd + MasVnrArea + BsmtFinSF1 + BsmtUnfSF +  
X1stFlrSF + X2ndFlrSF + BsmtFullBath + BedroomAbvGr + KitchenAbvGr +
```

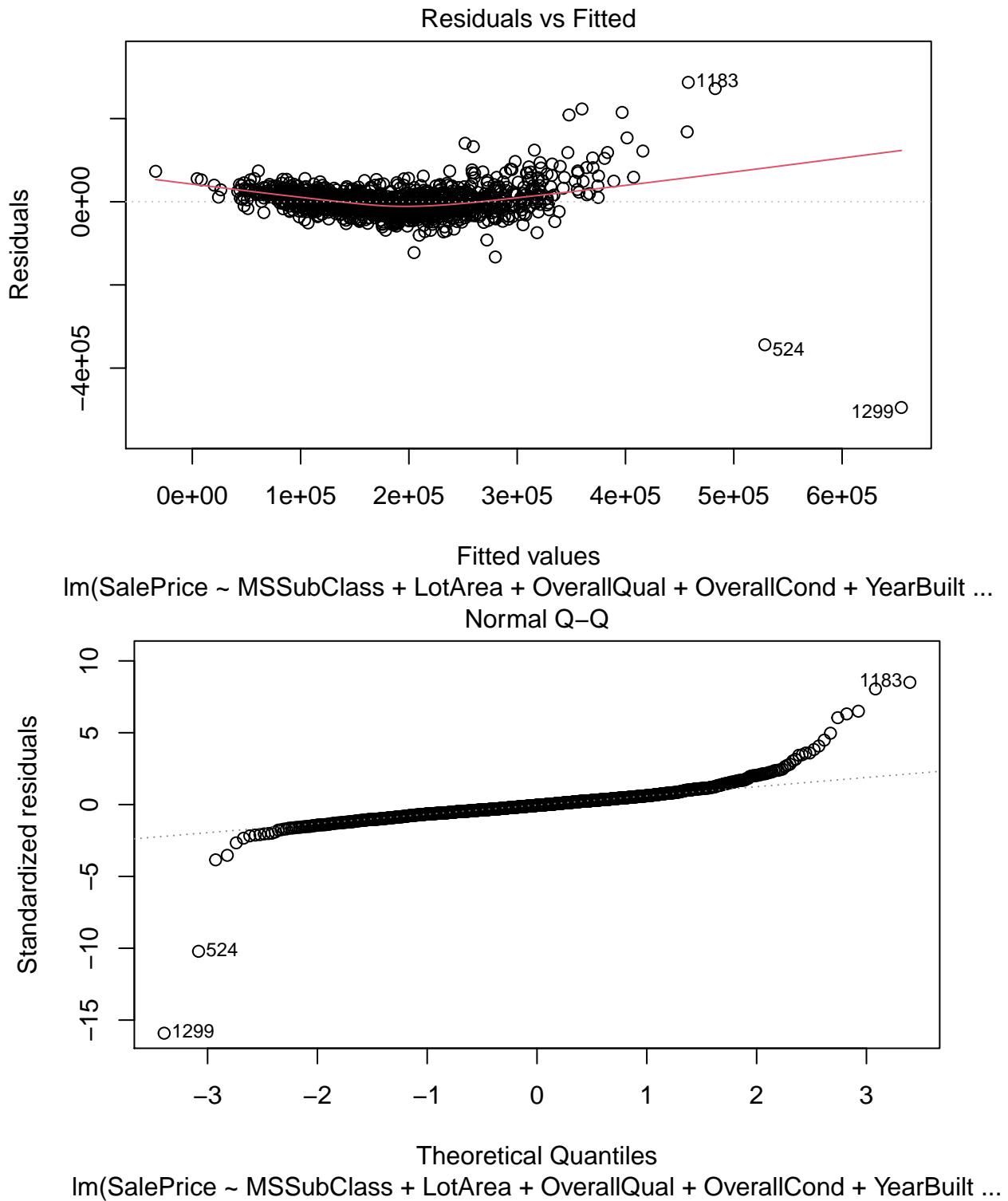
```

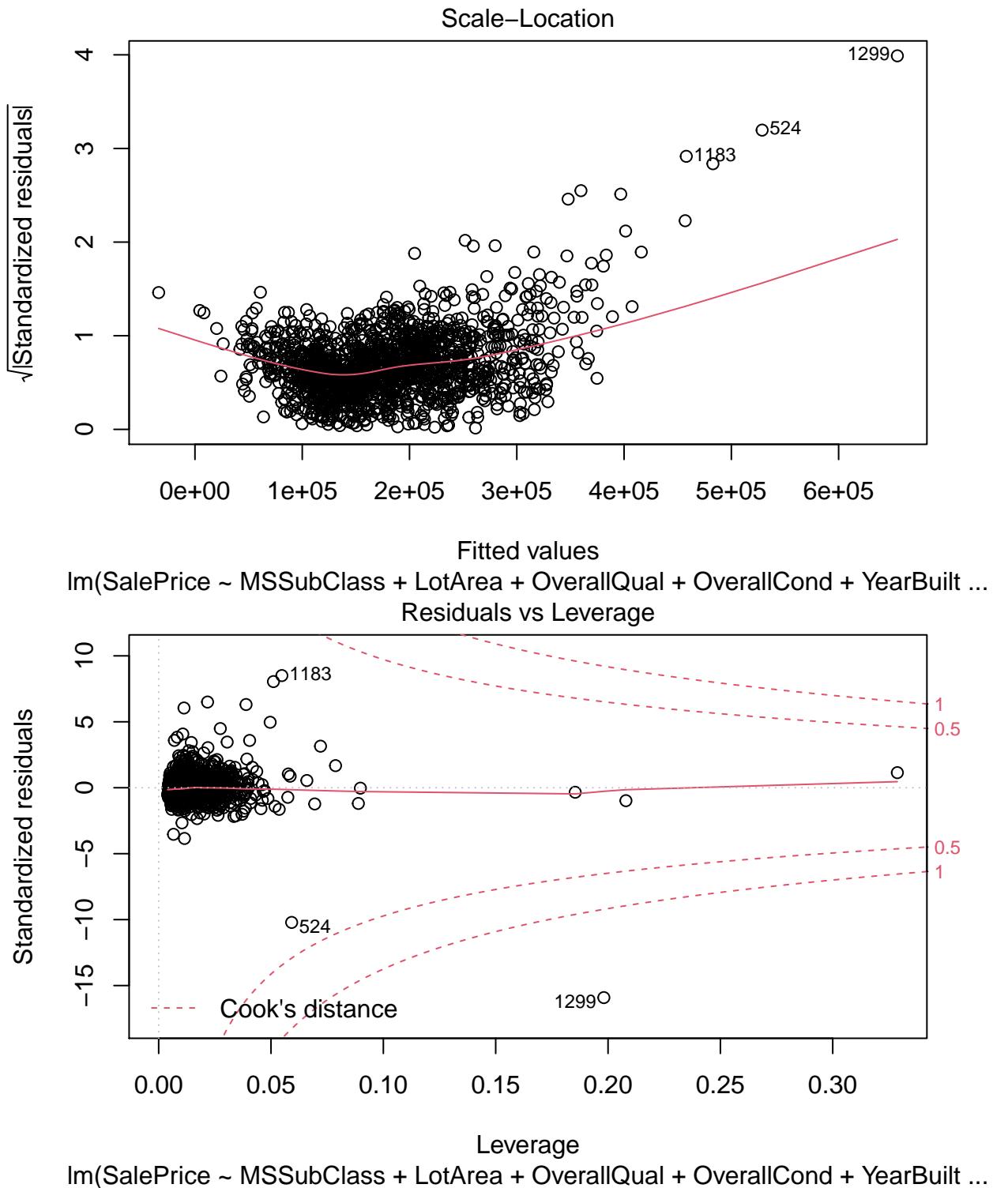
    TotRmsAbvGrd + Fireplaces + GarageYrBlt + GarageCars + WoodDeckSF +
ScreenPorch, data = train_4)
summary(model2)

##
## Call:
## lm(formula = SalePrice ~ MSSubClass + LotArea + OverallQual +
##     OverallCond + YearBuilt + YearRemodAdd + MasVnrArea + BsmtFinSF1 +
##     BsmtUnfSF + X1stFlrSF + X2ndFlrSF + BsmtFullBath + BedroomAbvGr +
##     KitchenAbvGr + TotRmsAbvGrd + Fireplaces + GarageYrBlt +
##     GarageCars + WoodDeckSF + ScreenPorch, data = train_4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -494744 -16147 -1821 13648 287029
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.121e+06 1.217e+05 -9.208 < 2e-16 ***
## MSSubClass   -1.648e+02 2.581e+01 -6.382 2.34e-10 ***
## LotArea       4.291e-01 9.987e-02  4.296 1.85e-05 ***
## OverallQual  1.774e+04 1.167e+03 15.195 < 2e-16 ***
## OverallCond  4.452e+03 1.014e+03  4.390 1.22e-05 ***
## YearBuilt     2.485e+02 5.874e+01  4.230 2.48e-05 ***
## YearRemodAdd 1.475e+02 6.746e+01  2.186 0.028980 *
## MasVnrArea   3.055e+01 5.868e+00  5.206 2.21e-07 ***
## BsmtFinSF1   1.571e+01 3.905e+00  4.024 6.02e-05 ***
## BsmtUnfSF    6.632e+00 3.630e+00  1.827 0.067874 .
## X1stFlrSF    5.147e+01 5.094e+00 10.104 < 2e-16 ***
## X2ndFlrSF    4.722e+01 4.089e+00 11.548 < 2e-16 ***
## BsmtFullBath 9.087e+03 2.399e+03  3.788 0.000158 ***
## BedroomAbvGr -9.634e+03 1.663e+03 -5.795 8.40e-09 ***
## KitchenAbvGr -1.298e+04 5.056e+03 -2.567 0.010365 *
## TotRmsAbvGrd 5.217e+03 1.211e+03  4.306 1.77e-05 ***
## Fireplaces   3.808e+03 1.758e+03  2.167 0.030428 *
## GarageYrBlt  1.419e+02 6.592e+01  2.152 0.031530 *
## GarageCars   1.030e+04 1.691e+03  6.088 1.46e-09 ***
## WoodDeckSF   2.389e+01 7.894e+00  3.027 0.002517 **
## ScreenPorch   5.369e+01 1.685e+01  3.186 0.001475 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34720 on 1439 degrees of freedom
## Multiple R-squared:  0.8116, Adjusted R-squared:  0.809
## F-statistic:  310 on 20 and 1439 DF,  p-value: < 2.2e-16

```

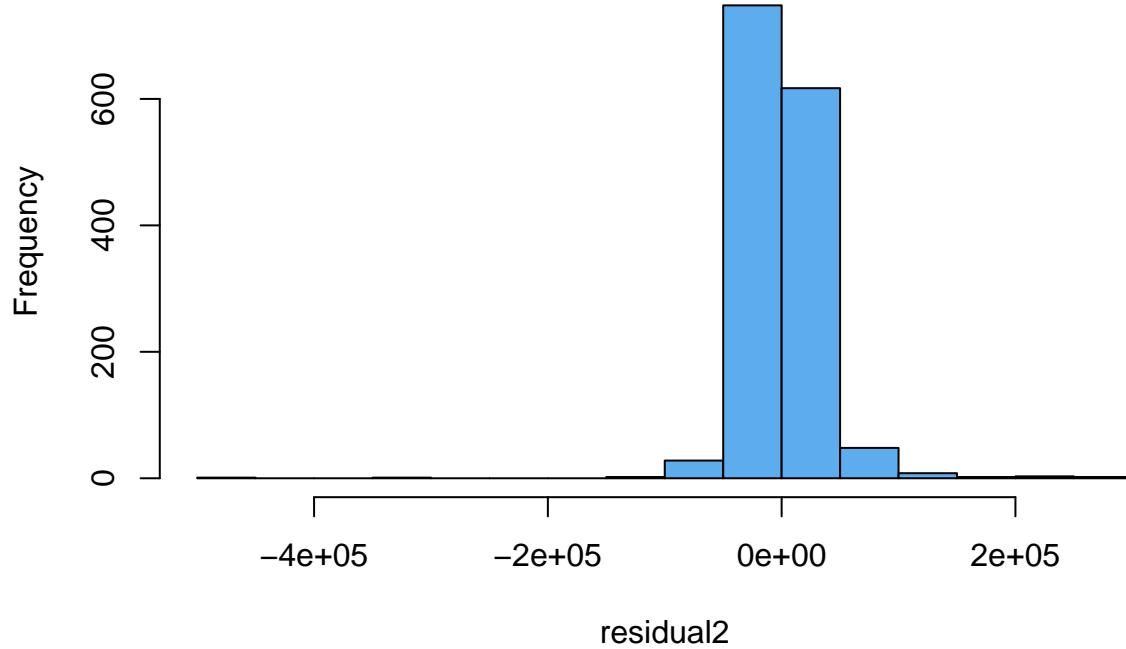
```
plot(model2)
```



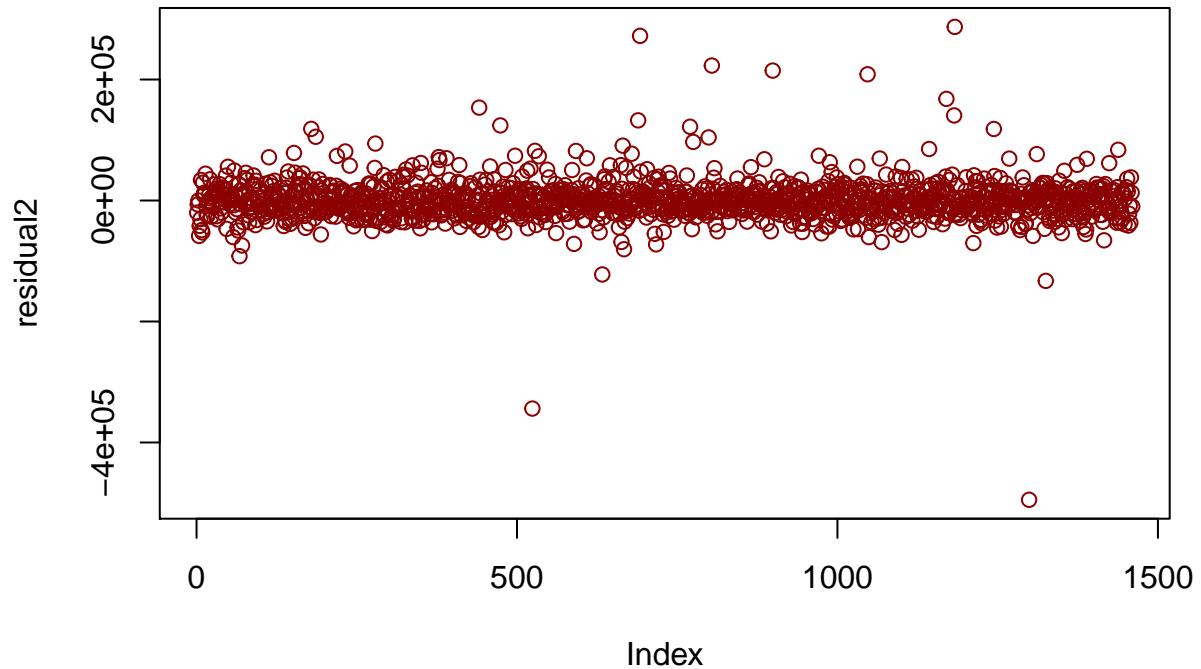


```
residual2 <- resid(model2)
hist(residual2, col = "steelblue2")
```

### Histogram of residual2



```
plot(residual2, col = 'dark red')
```



Select best model to predict salesprice on test dataset

Select model1 to predict the salesPrice on testdata.

```

pred1<-predict(model1, test_4)
#export data for Kaggle
train_5 <- as.data.frame(cbind(test$Id, pred1))
colnames(train_5) <- c("Id", "SalePrice")
write.csv(train_5, file = "Kaggle_Submission1.csv", quote=FALSE, row.names=FALSE)

```

**Prediction Submission Scores** Kaggle username: Subhalaxmi Rout



**Future work** Model can be improved using Random forest, Decision Tree algorithms. In future I will work more data cleaning and model building to get optimal prediction.

## Reference

[https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data?select=sample\\_submission.csv](https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data?select=sample_submission.csv)