Data605_Project3

Angus Huang

December 17, 2017

```
options(warn = 0)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(MASS)
```

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

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

```
library(ggplot2)
```

1.Download the dataset from the source. The original datasets are available from Kaggle.com https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

```
mydata <- read.table ("https://github.com/angus001/Data605/raw/master/train.csv", header = T,
sep =",")
testdata <- read.table ("https://github.com/angus001/Data605/raw/master/test.csv",header =T, s
ep =",")

#subset quantitative & a few other variables
mydata2 <-mydata[,which(names(mydata)%in% c("LotFrontage","LotArea","OverallQual","MasVnrArea"
, "BsmtFinSF1", "BsmtFinSF2","X1stFlrSF", "TotRmsAbvGrd", "GarageCars", "SalePrice","Neighborh
ood"))]
```

2 Clean up data

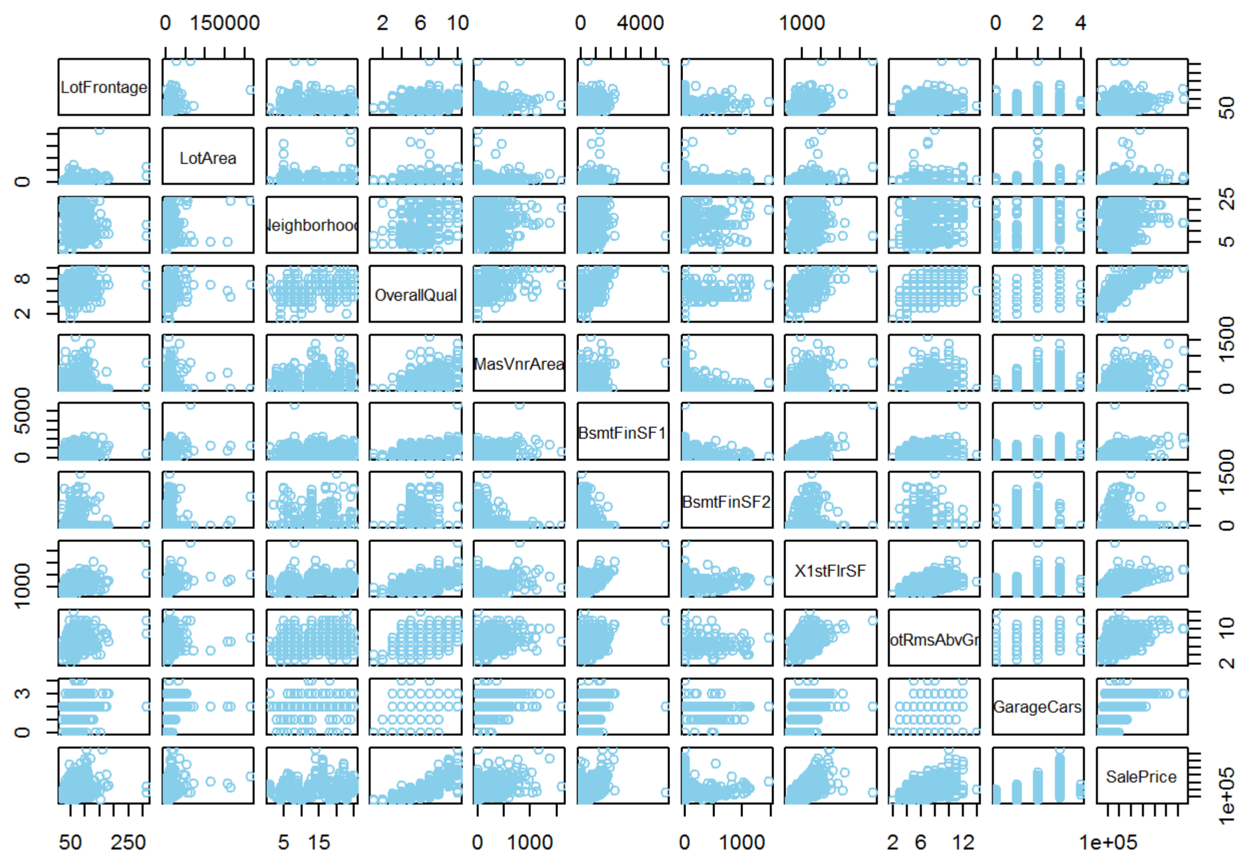
```
#check how many na in each variable/column
na_count <-sapply(mydata, function(x) sum(length(which(is.na(x)))))
na_count <- data.frame(na_count)
na_count$col_names <- rownames(na_count)

df1 <-filter(na_count, na_count > 0) #filter the dataframe into one that shows columns with na
 values
head( df1[order(-df1$na_count),]) #sort descending and show column with most na
```

```
##    na_count   col_names
## 17     1453      PoolQC
## 19     1406 MiscFeature
## 2      1369       Alley
## 18     1179       Fence
## 11      690 FireplaceQu
## 1       259 LotFrontage
```

2(a). Produce pair charts to see if any variable might have better relationship with the output-“SalePrice”. “X1stFlrSF” seems to be a great candidate.

```
pairs(mydata2,gap=0.5, col = 'skyblue')
```




```
#assigned zero to na values
mydata2[is.na(mydata2)]<-0
```

Picking a quantitative value and caculate the probabilty P(X>x&Y>y)

```
X <- mydata2$X1stFlrSF

Y <- mydata2$SalePrice

x <-quantile(X, 0.75)
y<-quantile(Y,0.5)
x
```

```
##     75%
## 1391.25
```

```
y
```

```
##    50%
## 163000
```

```
dfpb <- data.frame(X,Y)
dfpb$'P(X>x&Y>y)' <-ifelse (dfpb$X > x & dfpb$Y>y, 1.00,0)

dfpb$'Y>y' <-ifelse(dfpb$Y > y, 1.00,0)

#P(X,Y)/P(Y)
round((sum(dfpb$'P(X>x&Y>y)')/nrow(dfpb))/(sum(dfpb$'Y>y')/nrow(dfpb)), digits = 4)
```

```
## [1] 0.4299
```

```
sum(dfpb$'P(X>x&Y>y)')/nrow(dfpb)
```

```
## [1] 0.2143836
```

```
dfpb$'P(X<x & Y>y)' <- ifelse( X<x &Y> y, 1,0)
(sum(dfpb$'P(X<x & Y>y)')/nrow(dfpb))/(sum(dfpb$'Y>y')/nrow(dfpb))
```

```
## [1] 0.5700549
```

Building a regression model

1. Perform simple linear regression

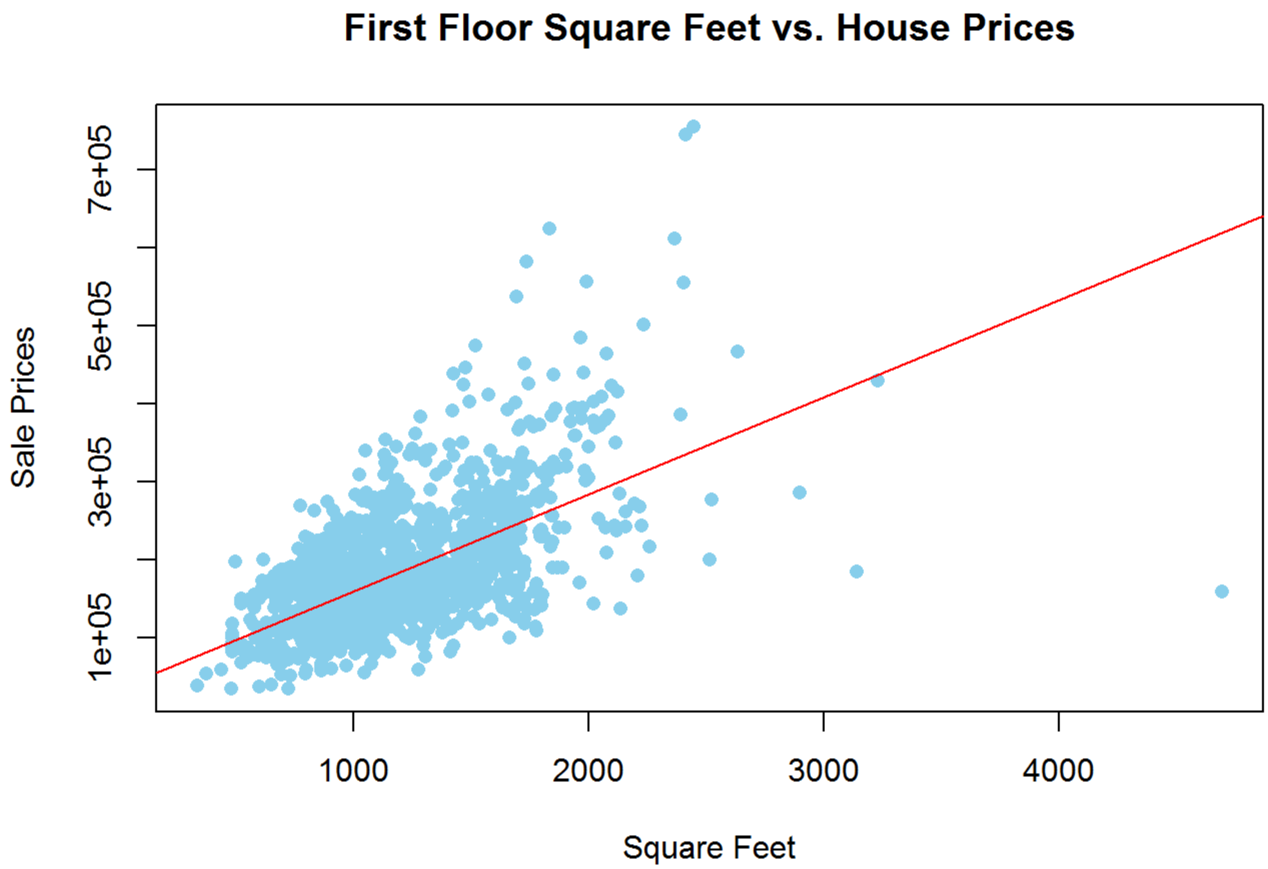
The R-Squared of 0.3671 indicates the 1st floor square feet ('X1stFlrSF) alone explain about 36% of the variance in saleprice across the selected neighborhoods. The P value is very small and less than 0.05, therefore the model is valid. F-statistic is used for additional check on the validity of R-Sqaured value. R-Squared value explains the strength of the relationship between the (input vs. output) variables. F-statistic then check if the R-sqaured value is valid or not. Low F-value means close similarity between groups while the high F-value means the opposite.

```
housepricelm <- lm(SalePrice ~ X1stFlrSF,  data = mydata2 )
summary(housepricelm)
```

```
##
## Call:
## lm(formula = SalePrice ~ X1stFlrSF, data = mydata2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -460330  -36494  -13164   36291  414547
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 36173.447   5245.728   6.896 7.95e-12 ***
## X1stFlrSF     124.501      4.282  29.078  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 63220 on 1458 degrees of freedom
## Multiple R-squared:  0.3671, Adjusted R-squared:  0.3666
## F-statistic: 845.5 on 1 and 1458 DF,  p-value: < 2.2e-16
```

Plotting a scatter plot with regression line.

```
{plot(mydata2$X1stFlrSF,mydata2$SalePrice,main = "First Floor Square Feet vs. House Prices",
      col = 'skyblue', pch = 16, xlab = "Square Feet", ylab = "Sale Prices")
abline(housepricelm, col = "red")}
```



With Multivariate regression, the 76% of the variance can be explained by the variables. The variables “LotFrontage” and “BsmtFinSF2” have has large P values (>0.05), and were removed in subsequent backward elimination.

```
#Subset data and select variables from pair plot.
housepricelm3 <-lm(SalePrice ~ X1stFlrSF+LotFrontage+LotArea+OverallQual+MasVnrArea+   BsmtFin
SF1+BsmtFinSF2+X1stFlrSF+TotRmsAbvGrd+GarageCars,  data = mydata2 )
summary(housepricelm3)
```

```
##
## Call:
## lm(formula = SalePrice ~ X1stFlrSF + LotFrontage + LotArea +
##     OverallQual + MasVnrArea + BsmtFinSF1 + BsmtFinSF2 + X1stFlrSF +
##     TotRmsAbvGrd + GarageCars, data = mydata2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -448667  -19676   -1053   16209  335340
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.261e+05  5.513e+03 -22.865  < 2e-16 ***
## X1stFlrSF     2.349e+01  3.571e+00   6.577 6.67e-11 ***
## LotFrontage   6.399e+00  3.030e+01   0.211   0.8328
## LotArea       6.296e-01  1.078e-01   5.843 6.33e-09 ***
## OverallQual   2.827e+04  9.986e+02  28.309  < 2e-16 ***
```

```
## MasVnrArea    3.729e+01  6.346e+00   5.876 5.19e-09 ***
## BsmtFinSF1    2.371e+01  2.575e+00   9.209  < 2e-16 ***
## BsmtFinSF2    1.129e+01  6.416e+00   1.760   0.0786 .
## TotRmsAbvGrd  8.534e+03  7.406e+02  11.523  < 2e-16 ***
## GarageCars    1.682e+04  1.751e+03   9.610  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 38470 on 1450 degrees of freedom
## Multiple R-squared:  0.767,  Adjusted R-squared:  0.7655
## F-statistic: 530.3 on 9 and 1450 DF,  p-value: < 2.2e-16
```

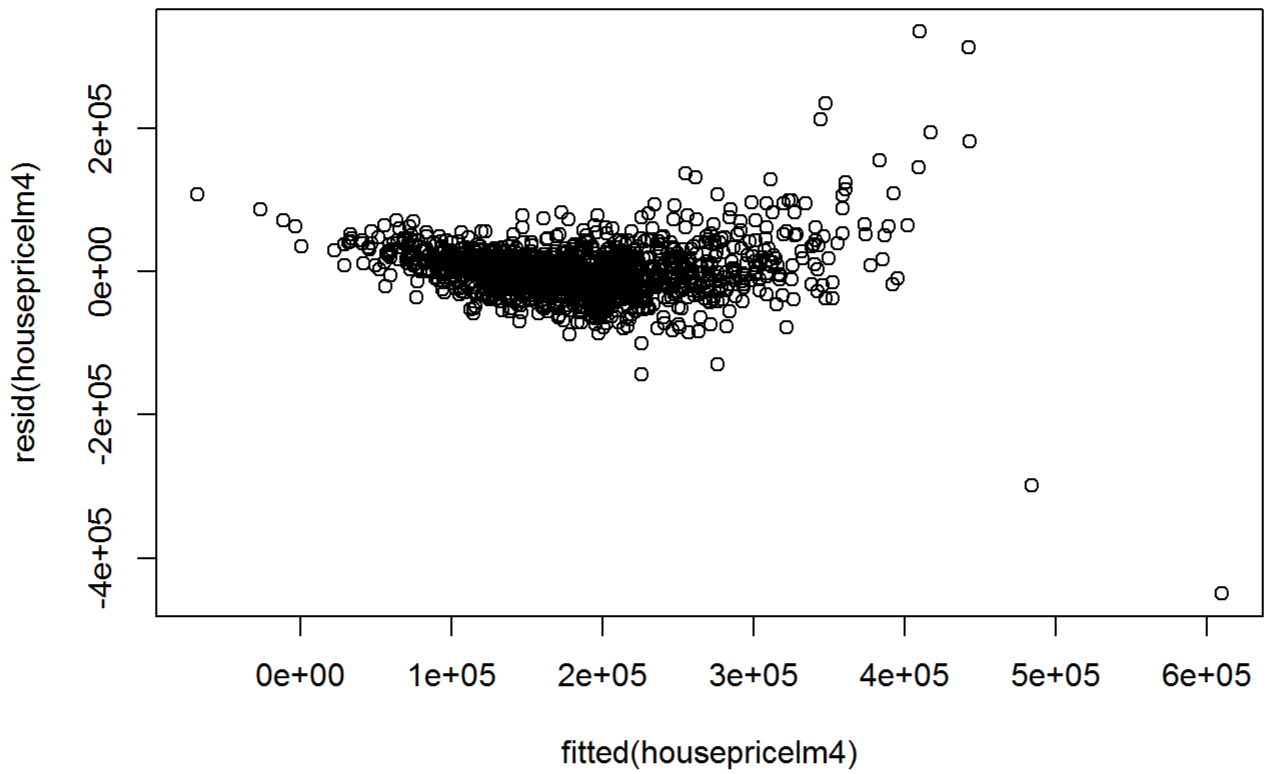
Remove variables (LotFrontage & BsmtFinSF2) with large p value (p value > 0.005)

```
housepricelm4 <- update(housepricelm3, .~. -LotFrontage, data = mydata2)
housepricelm4 <- update(housepricelm3, .~. -BsmtFinSF2, data = mydata2)
summary(housepricelm4)
```

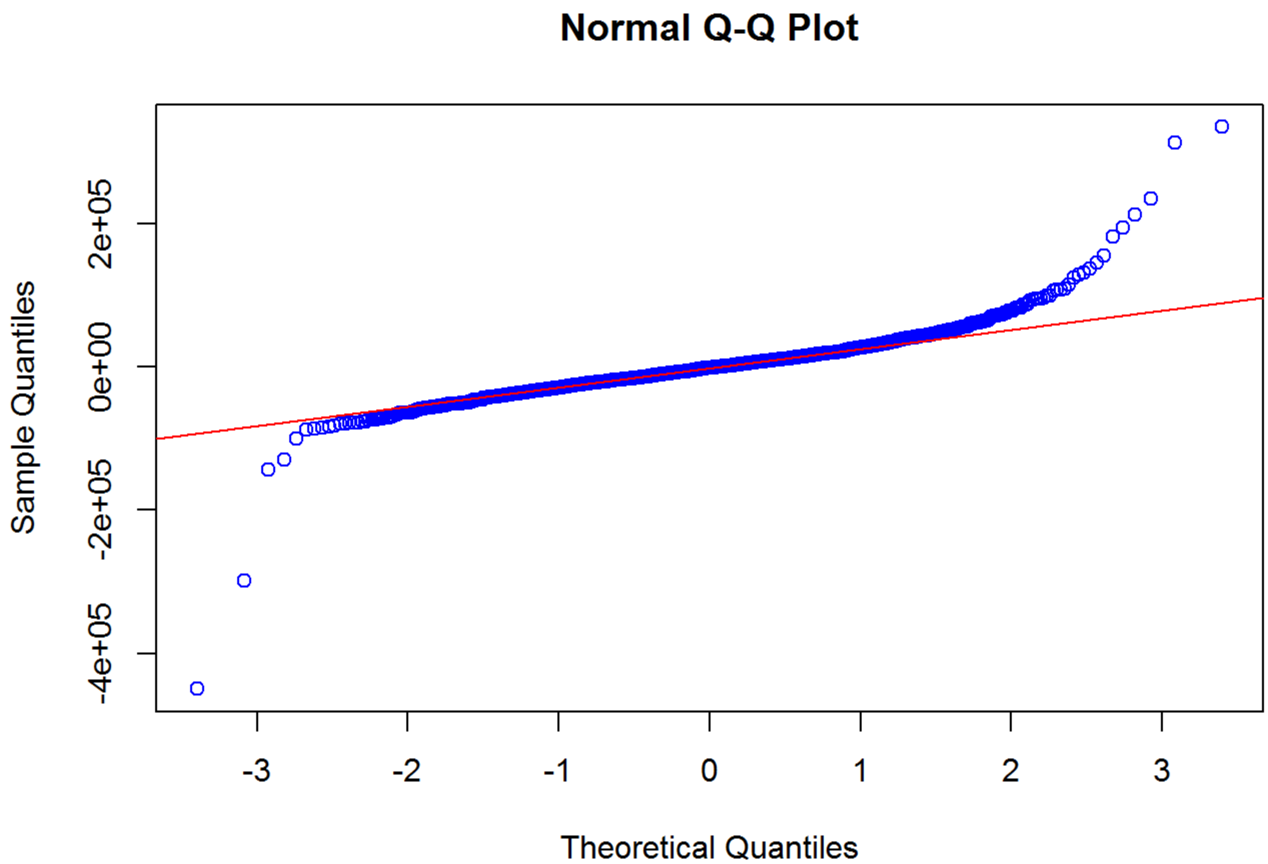
```
##
## Call:
## lm(formula = SalePrice ~ X1stFlrSF + LotFrontage + LotArea +
##     OverallQual + MasVnrArea + BsmtFinSF1 + TotRmsAbvGrd + GarageCars,
##     data = mydata2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -449697  -19708   -1112   16271  335042
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.254e+05  5.503e+03 -22.781  < 2e-16 ***
## X1stFlrSF     2.457e+01  3.521e+00   6.977 4.57e-12 ***
## LotFrontage   5.088e+00  3.031e+01   0.168    0.867
## LotArea       6.491e-01  1.073e-01   6.051 1.83e-09 ***
## OverallQual   2.819e+04  9.982e+02  28.237  < 2e-16 ***
## MasVnrArea    3.666e+01  6.340e+00   5.781 9.06e-09 ***
## BsmtFinSF1    2.319e+01  2.560e+00   9.060  < 2e-16 ***
## TotRmsAbvGrd  8.443e+03  7.394e+02  11.419  < 2e-16 ***
## GarageCars    1.675e+04  1.751e+03   9.563  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 38490 on 1451 degrees of freedom
## Multiple R-squared:  0.7665, Adjusted R-squared:  0.7652
## F-statistic: 595.4 on 8 and 1451 DF,  p-value: < 2.2e-16
```

Residual analysis shows the plot is slightltly curvilinear.

```
plot(fitted(housepricelm4),resid(housepricelm4))
```




```
qqnorm(resid(housepricelm4), col = "blue")
qqline(resid(housepricelm4), col = "red")
```



3 Predict the price with the model build above

3(a) Subset the data into a smaller data frame with variables used in the above model

```
testdata2 <-testdata[,which(names(testdata)%in% c("LotFrontage","LotArea","OverallQual","MasVnrArea", "BsmtFinSF1","Id", "BsmtFinSF2","X1stFlrSF", "TotRmsAbvGrd", "GarageCars","Neighborhood"))]

#check the number of na in data column
na_count2 <-sapply(testdata2, function(x) sum(length(which(is.na(x)))))
na_count2 <- data.frame(na_count2)
na_count2$col_names <- rownames(na_count2)
na_count2 <-filter(na_count2, na_count2 > 0) #filter the dataframe into one that shows columns with na values
na_count2[order(-na_count2$na_count2),] #sort descending and show column with most na
```

```
##   na_count2   col_names
## 1       227 LotFrontage
## 2        15  MasVnrArea
## 3         1  BsmtFinSF1
## 4         1  BsmtFinSF2
## 5         1  GarageCars
```

```
# Assign zero for na values
```

```
testdata2[is.na(testdata2)]<-0
```

```
#predict the "SalePrice"
results <-predict(housepricelm4,testdata2)
testdata2$SalePrice <-c(abs(results))
testdata3<-data.frame(testdata2[,c("Id","SalePrice")])

head(testdata3)
```

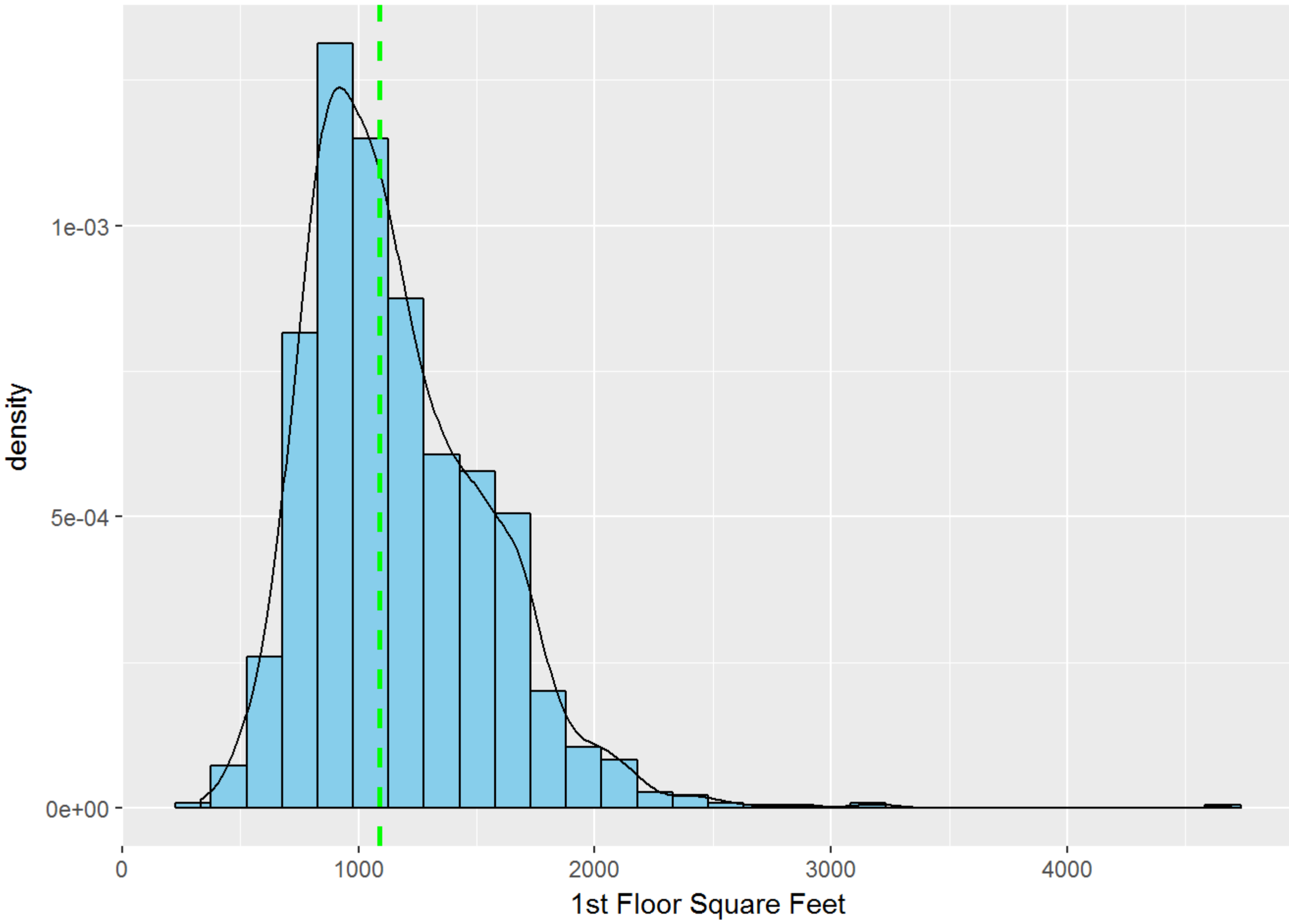
```
##     Id SalePrice
## 1 1461  115339.1
## 2 1462  178837.6
## 3 1463  150210.7
## 4 1464  180660.1
## 5 1465  216841.4
## 6 1466  161960.7
```

```
#Below line is writing the result to csv file for submission.
#write.csv(testdata3, file= "Myresult.csv", sep=",")
```

```
X<-mydata$X1stFlrSF
Y<-mydata$SalePrice
df<-data.frame(X,Y)
```

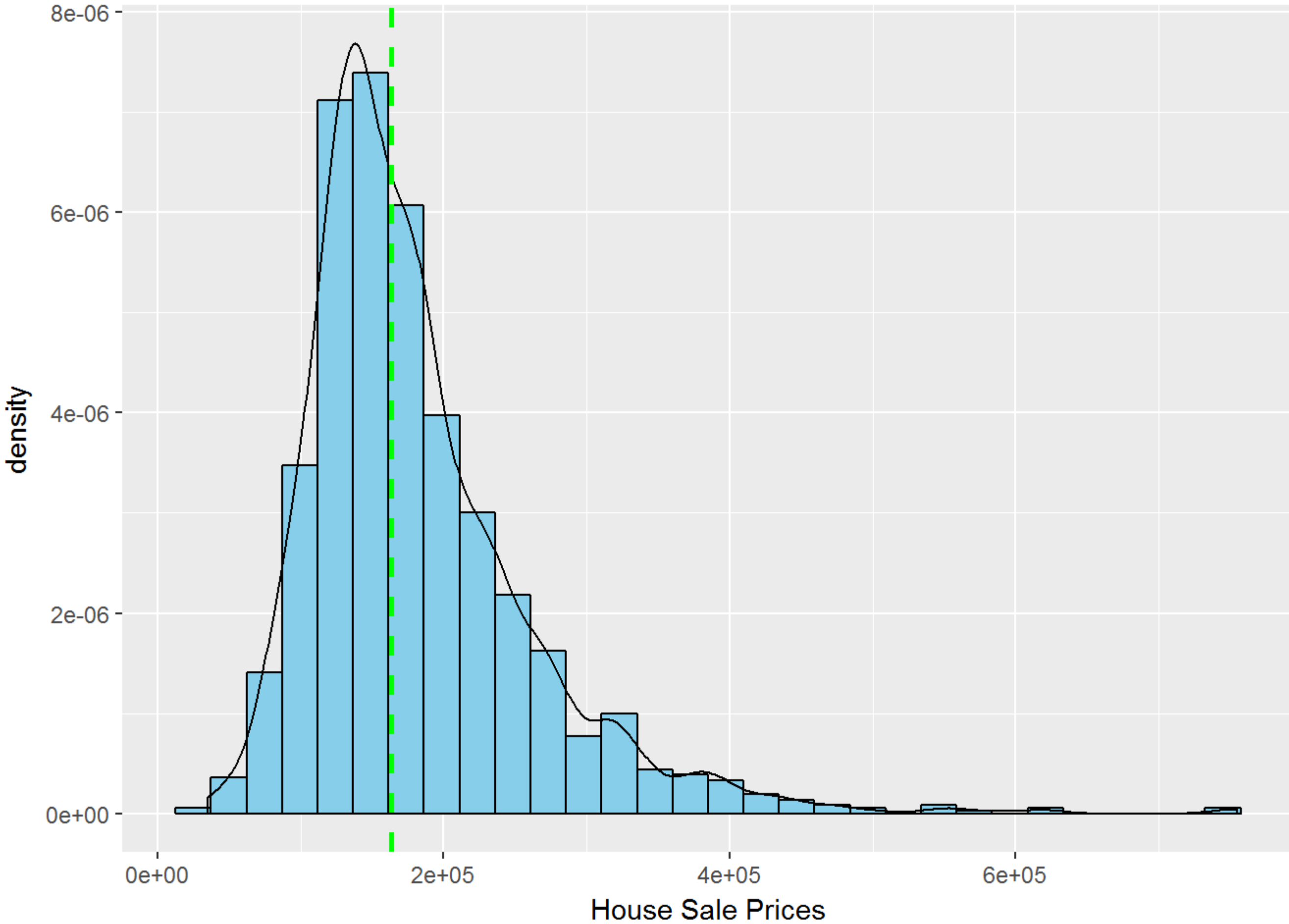
```
ggplot(df, aes(x=X)) +
  geom_histogram(aes(y=..density..), colour="black", fill="skyblue")+
  geom_density(alpha=0.5) +
  xlab("1st Floor Square Feet") +
  geom_vline(aes(xintercept=median(df$X)),
             color="green", linetype="dashed", size=1)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



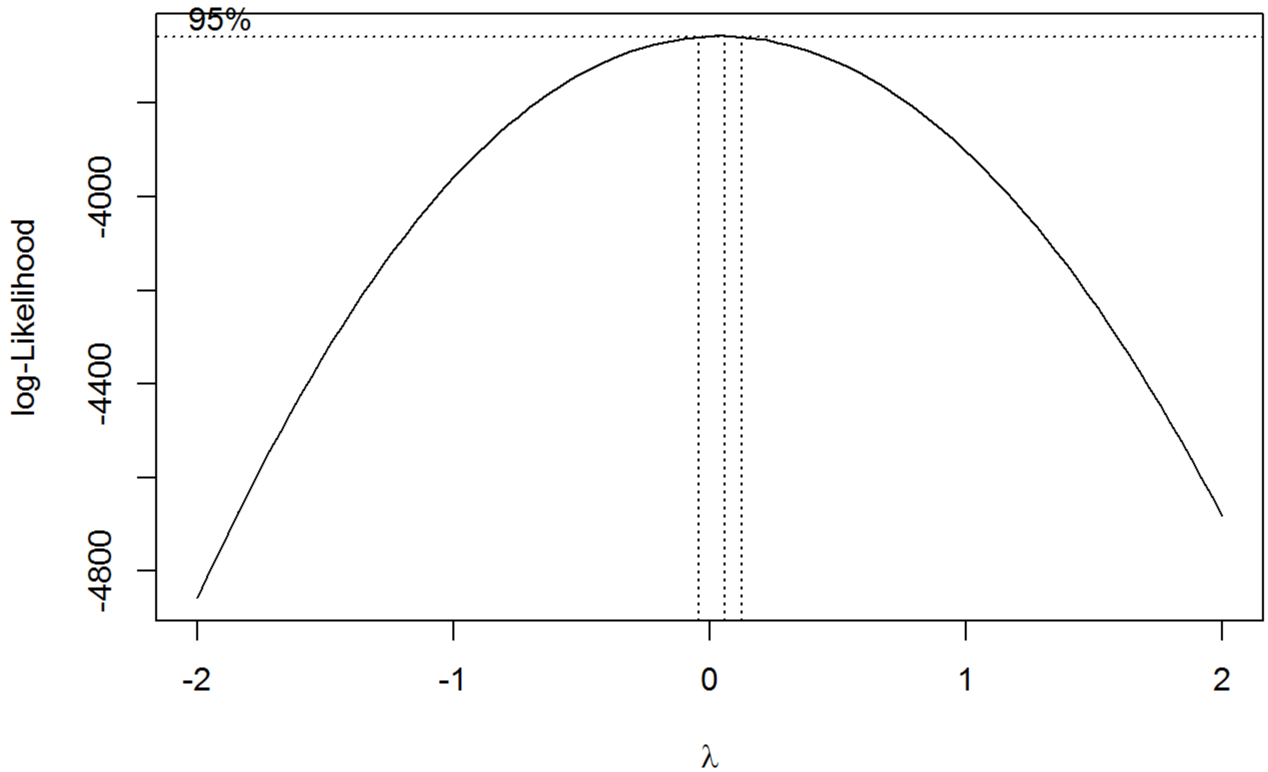

```
ggplot(df, aes(x=Y)) +
  geom_histogram(aes(y=..density..), colour="black", fill="skyblue")+
  geom_density(alpha=0.5) +
  xlab("House Sale Prices") +
  geom_vline(aes(xintercept=median(df$Y)),
             color="green", linetype="dashed", size=1)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Perform boxcox analysis to find log-Likelihood. Look for Lambda value with max likelihood. THe max likelihood is at 0.06 power.

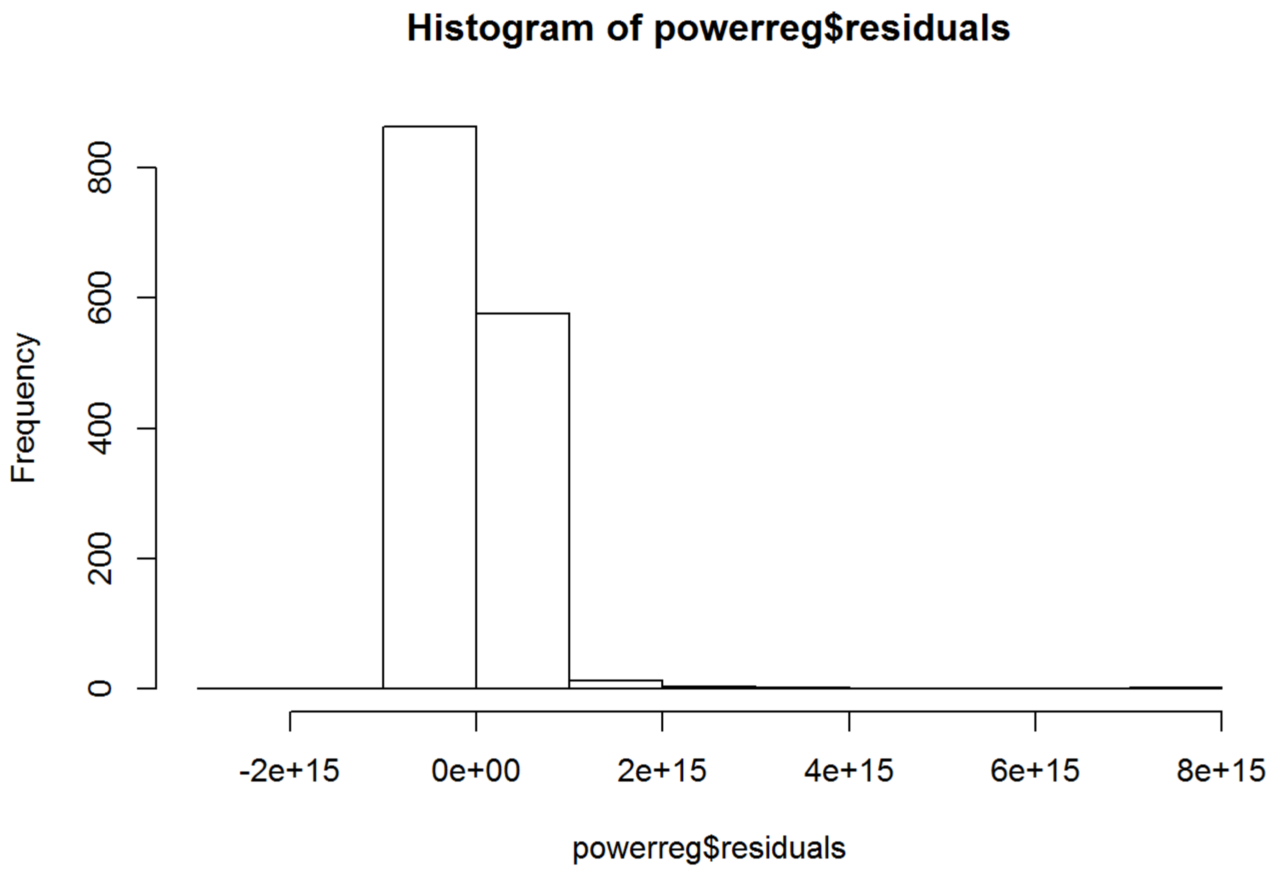
```
bc = boxcox(Y~X, data = df)
```



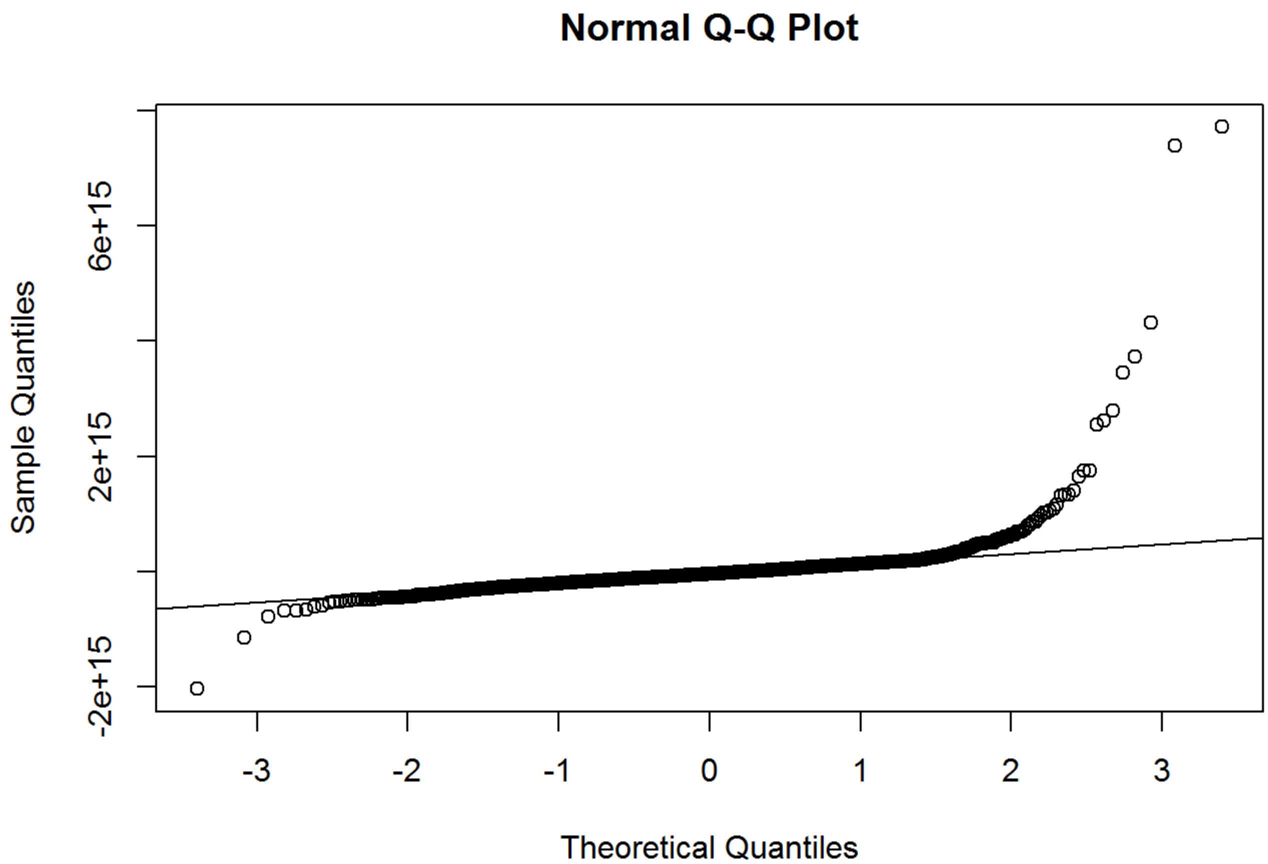

```
lamda =bc$x
likelihood = bc$y
bc1=cbind(lamda,likelihood)
head(bc1[order(-likelihood),])
```

```
##             lamda likelihood
## [1,]  0.06060606  -3656.618
## [2,]  0.02020202  -3656.622
## [3,]  0.10101010  -3657.490
## [4,] -0.02020202  -3657.505
## [5,]  0.14141414  -3659.237
## [6,] -0.06060606  -3659.270
```

```
df$Ypower = (df$Y)^3/50
powerreg <- lm(Ypower~X, df)
hist(powerreg$residuals)
```




```
qqnorm(powerreg$residuals)
qqline(powerreg$residuals)
```



Perform a correlation test between variables. The correlation test shows a correlation of 0.605 without tranforming the variable. The correlation actually become less to 0.441 after transforming the variable.

```
cor(df)
```

```
##                X         Y    Ypower
## X      1.0000000 0.6058522 0.4411002
## Y      0.6058522 1.0000000 0.8019417
## Ypower 0.4411002 0.8019417 1.0000000
```

```
cor.test(df$X,df$Y, conf.level = 0.99)
```

```
## 
##  Pearson's product-moment correlation
## 
## data:  df$X and df$Y
## t = 29.078, df = 1458, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 99 percent confidence interval:
##  0.5613896 0.6468270
## sample estimates:
##       cor 
## 0.6058522
```

Fitting the data point into different distribution to understand the underlying spread of the data.

```
fit <- fitdistr(df$X, densfun = 'cauchy')
fit
```

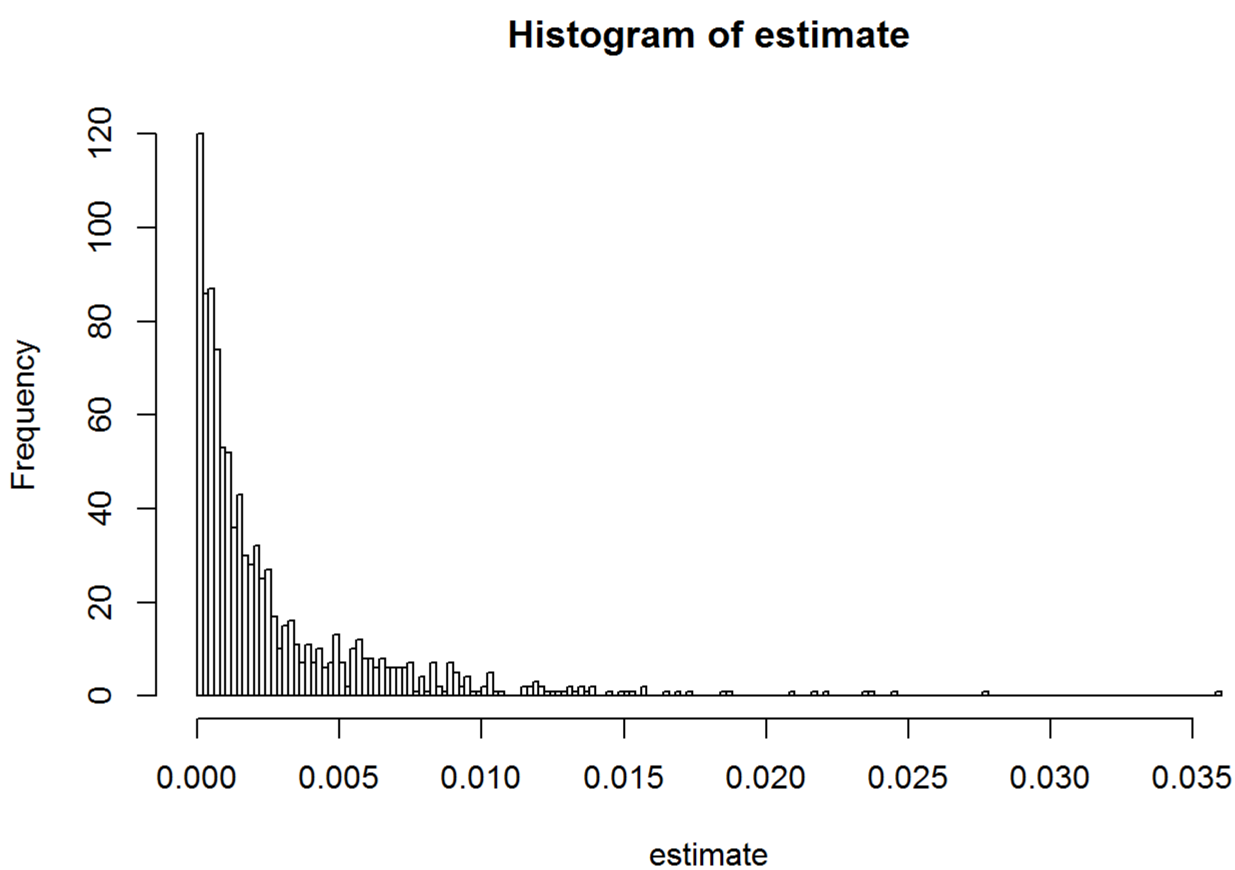
```
##      location        scale
##   1059.239655    212.473032
##  (   9.090705) (   7.210534)
```

```
lamda2 <- fit$estimate
lamda2
```

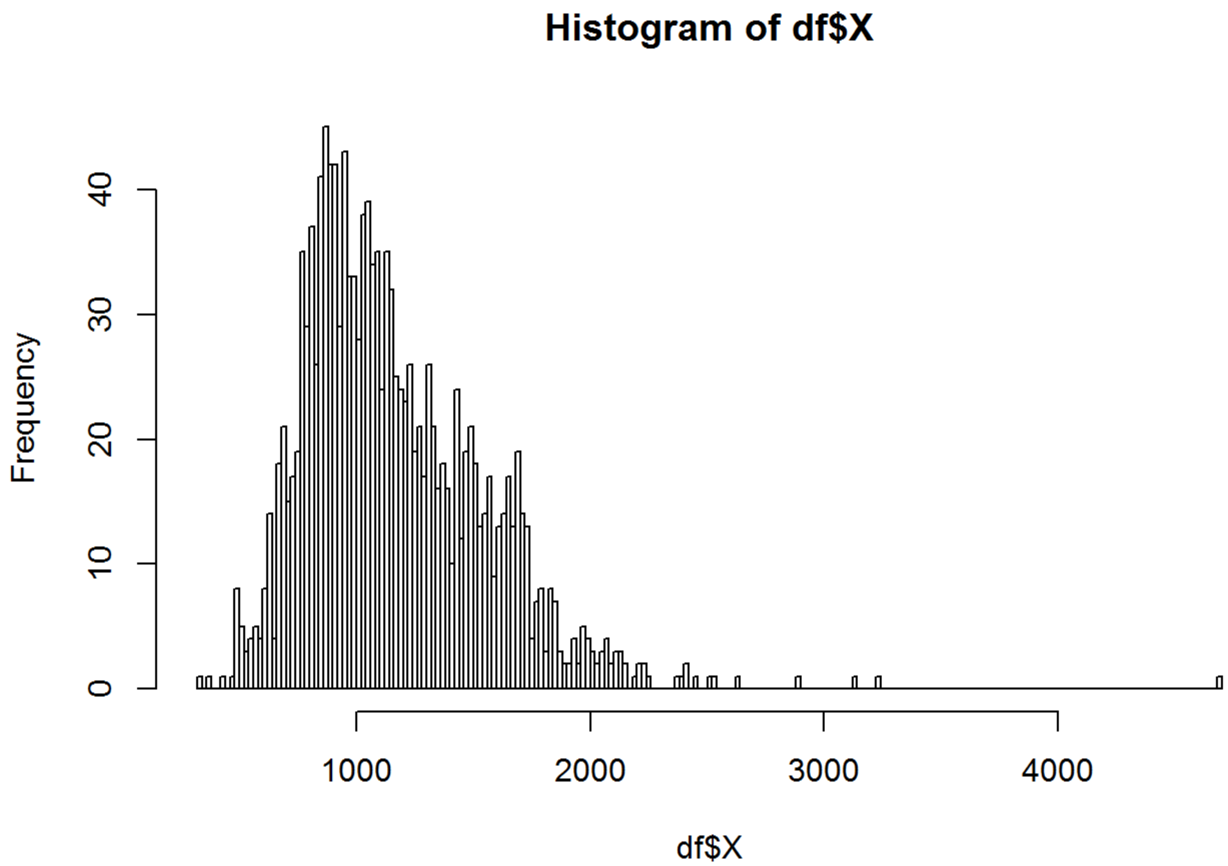
```
## location    scale
## 1059.240  212.473
```

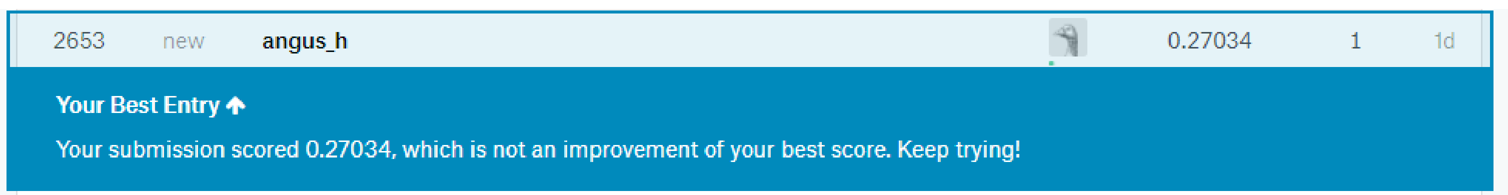
Take 1000 samples from the distribution, plot a histogram and compare with the non-transformed original values.

```
estimate <- rexp(1000,lamda2)
hist(estimate, breaks = 200)
```




```
hist(df$X, breaks = 200)
```





Kaggle Result

```
#![Kaggle Result](https://github.com/angus001/Data605/blob/master/kagglefirsttry.PNG?raw=true)
```