KBhave_605FP1

Kumudini Bhave

May 20, 2017

Contents

1 Regression Model : Predicting Sales For City Housing Data

1.1 Summary

This is an R Markdown document for providing documentation for performing **Regression** by practising **Data Exploration, Transformation, Analysis, Modelling and Prediction Of the Housing DataSet**

In the process, we will explore Probability, Descriptive and Inferential Statistics, Linear Algebra and Correlation, Calculus based Probability & Statistics, and Modeling.

For this exploration we will use the **Housing** data set from "The House Prices: Advanced Regression Techniques Competition" on Kaggle.com, see link below.[Kaggle] (https://www.kaggle.com/c/house-prices-advanced-regression-techniques)

1.2 Housing DataSet

The Housing dataset of a major city depicts 1460 observations across 81 variables. The description of these can be found here at [data_description.txt] (https://raw.githubusercontent.com/DataDriven-MSDA/DATA605/master/Final/data_description.txt)

```r
knitr::opts_chunk$set(message = FALSE, echo = TRUE)


# Library for loading CSV data
library(RCurl)

# Library for data display in tabular format

library(DT)
library(dplyr)
# Library for plotting
library(ggplot2)
library(gridExtra)
# Statistical Packages
library(corrplot)
library(e1071)
library(data.table)
library(knitr)
library(caret)
library(pander)
library(car)
# library(bestglm)
library(MASS)
library(Amelia)
library(leaps)
```

1.3 Data Exploration

We load the data pertaining to this competition, and study the same.

Let us performed some basic exploration of the data. This **Housing data set** has **81 variables** and **1460 observations**. Based on the descriptions of the various variables (see data set text documentation), we may conclude that the dependent variable is the SalePrice. The remaining variables are both qualitative or quantitative in nature.

```r
# Getting data

trdata.giturl <- "https://raw.githubusercontent.com/DataDriven-MSDA/DATA605/master/Final/train.csv"

evaldata.giturl <- "https://raw.githubusercontent.com/DataDriven-MSDA/DATA605/master/Final/test.csv"


# Remove the Index Column/variable
traindataorig <- read.csv(url(trdata.giturl))
traindataorig <- dplyr::select(traindataorig, -1)
traindata <- traindataorig

evaldataorig <- read.csv(url(evaldata.giturl))
# evaldataorig <- dplyr::select(evaldataorig, -1)
evaldata <- dplyr::select(evaldataorig, -1)


nrow(traindata)
```

```
## [1] 1460
```

```r
ncol(traindata)
```

```
## [1] 80
```

```r
# View(traindata) View(evaldata)
```

The summary of the Housing data

Out of the 80 variables(after removing the Index), we have one response/ dependant variable **SalePrice**. While 79 others are predictor variables. Below is the summary for all the variables in the dataset

```r
pander(summary(traindata[, seq(1, 25)]))
```

Table 1: Table continues below

MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley
Min. : 20.0	C (all): 10	Min. : 21.00	Min. : 1300	Grvl: 6	Grvl: 50
1st Qu.: 20.0	FV : 65	1st Qu.: 59.00	1st Qu.: 7554	Pave:1454	Pave: 41
Median : 50.0	RH : 16	Median : 69.00	Median : 9478	NA	NA's:1369
Mean : 56.9	RL :1151	Mean : 70.05	Mean : 10517	NA	NA
3rd Qu.: 70.0	RM : 218	3rd Qu.: 80.00	3rd Qu.: 11602	NA	NA
Max. :190.0	NA	Max. :313.00	Max. :215245	NA	NA
NA	NA	NA's :259	NA	NA	NA

Table 2: Table continues below

LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood
IR1:484	Bnk: 63	AllPub:1459	Corner : 263	Gtl:1382	NAmes :225
IR2: 41	HLS: 50	NoSeWa: 1	CulDSac: 94	Mod: 65	CollgCr:150
IR3: 10	Low: 36	NA	FR2 : 47	Sev: 13	OldTown:113
Reg:925	Lvl:1311	NA	FR3 : 4	NA	Edwards:100
NA	NA	NA	Inside :1052	NA	Somerst: 86
NA	NA	NA	NA	NA	Gilbert: 79
NA	NA	NA	NA	NA	(Other):707

Table 3: Table continues below

Condition1	Condition2	BldgType	HouseStyle	OverallQual	OverallCond
Norm :1260	Norm :1445	1Fam :1220	1Story :726	Min. : 1.000	Min. :1.000
Feedr : 81	Feedr : 6	2fmCon: 31	2Story :445	1st Qu.: 5.000	1st Qu.:5.000
Artery : 48	Artery : 2	Duplex: 52	1.5Fin :154	Median : 6.000	Median :5.000
RRAn : 26	PosN : 2	Twnhs : 43	SLvl : 65	Mean : 6.099	Mean :5.575
PosN : 19	RRNn : 2	TwnhsE: 114	SFoyer : 37	3rd Qu.: 7.000	3rd Qu.:6.000
RRAe : 11	PosA : 1	NA	1.5Unf : 14	Max. :10.000	Max. :9.000
(Other): 15	(Other): 2	NA	(Other): 19	NA	NA

Table 4: Table continues below

YearBuilt	YearRemodAdd	RoofStyle	RoofMatl	Exterior1st	Exterior2nd
Min. :1872	Min. :1950	Flat : 13	CompShg:1434	VinylSd:515	VinylSd:504
1st Qu.:1954	1st Qu.:1967	Gable :1141	Tar&Grv: 11	HdBoard:222	MetalSd:214
Median :1973	Median :1994	Gambrel: 11	WdShngl: 6	MetalSd:220	HdBoard:207
Mean :1971	Mean :1985	Hip : 286	WdShake: 5	Wd Sdng:206	Wd Sdng:197
3rd Qu.:2000	3rd Qu.:2004	Mansard: 7	ClyTile: 1	Plywood:108	Plywood:142
Max. :2010	Max. :2010	Shed : 2	Membran: 1	CemntBd: 61	CmentBd: 60
NA	NA	NA	(Other): 2	(Other):128	(Other):136

MasVnrType
BrkCmn : 15
BrkFace:445
None :864
Stone :128
NA's : 8
NA
NA

```
pander(summary(traindata[, seq(26, 50)]))
```

Table 6: Table continues below

MasVnrArea	ExterQual	ExterCond	Foundation	BsmtQual	BsmtCond
Min. : 0.0	Ex: 52	Ex: 3	BrkTil:146	Ex :121	Fa : 45

MasVnrArea	ExterQual	ExterCond	Foundation	BsmtQual	BsmtCond
1st Qu.: 0.0	Fa: 14	Fa: 28	CBlock:634	Fa : 35	Gd : 65
Median : 0.0	Gd:488	Gd: 146	PConc :647	Gd :618	Po : 2
Mean : 103.7	TA:906	Po: 1	Slab : 24	TA :649	TA :1311
3rd Qu.: 166.0	NA	TA:1282	Stone : 6	NA's: 37	NA's: 37
Max. :1600.0	NA	NA	Wood : 3	NA	NA
NA's :8	NA	NA	NA	NA	NA

Table 7: Table continues below

BsmtExposure	BsmtFinType1	BsmtFinSF1	BsmtFinType2	BsmtFinSF2
Av :221	ALQ :220	Min. : 0.0	ALQ : 19	Min. : 0.00
Gd :134	BLQ :148	1st Qu.: 0.0	BLQ : 33	1st Qu.: 0.00
Mn :114	GLQ :418	Median : 383.5	GLQ : 14	Median : 0.00
No :953	LwQ : 74	Mean : 443.6	LwQ : 46	Mean : 46.55
NA's: 38	Rec :133	3rd Qu.: 712.2	Rec : 54	3rd Qu.: 0.00
NA	Unf :430	Max. :5644.0	Unf :1256	Max. :1474.00
NA	NA's: 37	NA	NA's: 38	NA

Table 8: Table continues below

BsmtUnfSF	TotalBsmtSF	Heating	HeatingQC	CentralAir	Electrical
Min. : 0.0	Min. : 0.0	Floor: 1	Ex:741	N: 95	FuseA: 94
1st Qu.: 223.0	1st Qu.: 795.8	GasA :1428	Fa: 49	Y:1365	FuseF: 27
Median : 477.5	Median : 991.5	GasW : 18	Gd:241	NA	FuseP: 3
Mean : 567.2	Mean :1057.4	Grav : 7	Po: 1	NA	Mix : 1
3rd Qu.: 808.0	3rd Qu.:1298.2	OthW : 2	TA:428	NA	SBrkr:1334
Max. :2336.0	Max. :6110.0	Wall : 4	NA	NA	NA's : 1
NA	NA	NA	NA	NA	NA

Table 9: Table continues below

X1stFlrSF	X2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath
Min. : 334	Min. : 0	Min. : 0.000	Min. : 334	Min. :0.0000
1st Qu.: 882	1st Qu.: 0	1st Qu.: 0.000	1st Qu.:1130	1st Qu.:0.0000
Median :1087	Median : 0	Median : 0.000	Median :1464	Median :0.0000
Mean :1163	Mean : 347	Mean : 5.845	Mean :1515	Mean :0.4253
3rd Qu.:1391	3rd Qu.: 728	3rd Qu.: 0.000	3rd Qu.:1777	3rd Qu.:1.0000
Max. :4692	Max. :2065	Max. :572.000	Max. :5642	Max. :3.0000
NA	NA	NA	NA	NA

BsmtHalfBath	FullBath	HalfBath
Min. :0.00000	Min. :0.000	Min. :0.0000
1st Qu.:0.00000	1st Qu.:1.000	1st Qu.:0.0000
Median :0.00000	Median :2.000	Median :0.0000
Mean :0.05753	Mean :1.565	Mean :0.3829
3rd Qu.:0.00000	3rd Qu.:2.000	3rd Qu.:1.0000

BsmtHalfBath	FullBath	HalfBath
Max. :2.00000	Max. :3.000	Max. :2.0000
NA	NA	NA

```
pander(summary(traindata[, seq(51, 80)]))
```

Table 11: Table continues below

BedroomAbvGr	KitchenAbvGr	KitchenQual	TotRmsAbvGrd	Functional
Min. :0.000	Min. :0.000	Ex:100	Min. : 2.000	Maj1: 14
1st Qu.:2.000	1st Qu.:1.000	Fa: 39	1st Qu.: 5.000	Maj2: 5
Median :3.000	Median :1.000	Gd:586	Median : 6.000	Min1: 31
Mean :2.866	Mean :1.047	TA:735	Mean : 6.518	Min2: 34
3rd Qu.:3.000	3rd Qu.:1.000	NA	3rd Qu.: 7.000	Mod : 15
Max. :8.000	Max. :3.000	NA	Max. :14.000	Sev : 1
NA	NA	NA	NA	Typ :1360

Table 12: Table continues below

Fireplaces	FireplaceQu	GarageType	GarageYrBlt	GarageFinish
Min. :0.000	Ex : 24	2Types : 6	Min. :1900	Fin :352
1st Qu.:0.000	Fa : 33	Attchd :870	1st Qu.:1961	RFn :422
Median :1.000	Gd :380	Basment: 19	Median :1980	Unf :605
Mean :0.613	Po : 20	BuiltIn: 88	Mean :1979	NA's: 81
3rd Qu.:1.000	TA :313	CarPort: 9	3rd Qu.:2002	NA
Max. :3.000	NA's:690	Detchd :387	Max. :2010	NA
NA	NA	NA's : 81	NA's :81	NA

Table 13: Table continues below

GarageCars	GarageArea	GarageQual	GarageCond	PavedDrive
Min. :0.000	Min. : 0.0	Ex : 3	Ex : 2	N: 90
1st Qu.:1.000	1st Qu.: 334.5	Fa : 48	Fa : 35	P: 30
Median :2.000	Median : 480.0	Gd : 14	Gd : 9	Y:1340
Mean :1.767	Mean : 473.0	Po : 3	Po : 7	NA
3rd Qu.:2.000	3rd Qu.: 576.0	TA :1311	TA :1326	NA
Max. :4.000	Max. :1418.0	NA's: 81	NA's: 81	NA
NA	NA	NA	NA	NA

Table 14: Table continues below

WoodDeckSF	OpenPorchSF	EnclosedPorch	X3SsnPorch	ScreenPorch
Min. : 0.00	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	1st Qu.: 0.00
Median : 0.00	Median : 25.00	Median : 0.00	Median : 0.00	Median : 0.00
Mean : 94.24	Mean : 46.66	Mean : 21.95	Mean : 3.41	Mean : 15.06
3rd Qu.:168.00	3rd Qu.: 68.00	3rd Qu.: 0.00	3rd Qu.: 0.00	3rd Qu.: 0.00
Max. :857.00	Max. :547.00	Max. :552.00	Max. :508.00	Max. :480.00

WoodDeckSF	OpenPorchSF	EnclosedPorch	X3SsnPorch	ScreenPorch
NA	NA	NA	NA	NA

Table 15: Table continues below

PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold
Min. : 0.000	Ex : 2	GdPrv: 59	Gar2: 2	Min. : 0.00	Min. : 1.000
1st Qu.: 0.000	Fa : 2	GdWo : 54	Othr: 2	1st Qu.: 0.00	1st Qu.: 5.000
Median : 0.000	Gd : 3	MnPrv: 157	Shed: 49	Median : 0.00	Median : 6.000
Mean : 2.759	NA's:1453	MnWw : 11	TenC: 1	Mean : 43.49	Mean : 6.322
3rd Qu.: 0.000	NA	NA's :1179	NA's:1406	3rd Qu.: 0.00	3rd Qu.: 8.000
Max. :738.000	NA	NA	NA	Max. :15500.00	Max. :12.000
NA	NA	NA	NA	NA	NA

YrSold	SaleType	SaleCondition	SalePrice
Min. :2006	WD :1267	Abnorml: 101	Min. : 34900
1st Qu.:2007	New : 122	AdjLand: 4	1st Qu.:129975
Median :2008	COD : 43	Alloca : 12	Median :163000
Mean :2008	ConLD : 9	Family : 20	Mean :180921
3rd Qu.:2009	ConLI : 5	Normal :1198	3rd Qu.:214000
Max. :2010	ConLw : 5	Partial: 125	Max. :755000
NA	(Other): 9	NA	NA

Variables For Study

For the purpose of study we select the predictor variable as *GrLivArea* , as the quantitative Predictor Variable that impacts the Response variable *SalePrice* . The response as well as predictor are continuous variables.

```
# datatable(head(traindata,100), options = list( searching = FALSE, pageLength =
# 5, lengthMenu = c(5, 10, 15, 20) ))

# data.table(head(traindata,100),width = 300)

# setting Y variable and select X variable
Y <- traindata$SalePrice
X <- traindata$GrLivArea
```

1.4 Probability

Lets define the x as 4th quartile (lower bound) of the Predictor Variable of X ie **GrLivArea** and the y as 2nd quartile of the response variable Y ie **SalePrice**

```
x <- quantile(X, 0.75)
x
```

```
##     75%
## 1776.75
```

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

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

(a)

$$P(X > x|Y > y) = \frac{P(X > x \cap Y > y)}{P(Y > y)}$$

```
p_XnY <- filter(traindata, SalePrice > y & GrLivArea > x) %>% tally()/nrow(traindata)

p_Y <- filter(traindata, SalePrice > y) %>% tally()/nrow(traindata)

pa <- (p_XnY/p_Y)
pa
```

```
##           n
## 1 0.4326923
```

43.27% likely that its living area is above the 75th percentile, when it is given that house's sale price is greater than the median house price

(b)

$$P(X > x, Y > y) = P(X > x \ \cap \ Y > y)$$

```
p_XnYonly <- filter(traindata, SalePrice > y & GrLivArea > x) %>% tally()/nrow(traindata)

pb <- p_XnYonly
pb
```

```
##           n
## 1 0.2157534
```

21.58% likely that the living area is above the 75th percentile and the sale price is above the median.

(c)

$$P(X < x|Y > y) = \frac{P(X < x \ \cap \ Y > y)}{P(Y > y)}$$

```
p_XY <- filter(traindata, SalePrice > y & GrLivArea < x) %>% tally()/nrow(traindata)
p_Y <- filter(traindata, SalePrice > y) %>% tally()/nrow(traindata)
```

```
pc <- (p_XY/p_Y)
pc
```

```
##           n
## 1 0.5673077
```

56.73% likeliness that house's living area is below the 75th percentile

1.5 Independance

We have,

x/y	below 2nd Qtr	above 2nd Qtr	Total
below 3rd Qtr	682	413	1095
above 3rd Qtr	50	315	365
Total	732	728	1460

Let X = observations above the 3d quartile for X, Let Y = observations above the 2d quartile for Y.

Checking for

$$P(X|Y) = P(X)P(Y)$$

$P(X)=$365/1460=0.25 ,
$P(Y)=$728/1460=0.4986301

Hence,

$P(X).P(Y)$= 0.25 * 0.4986301 = 0.1246575 $P(X|Y)$ = 0.4327

Since $P(X|Y) \quad \neq \quad P(X)P(Y)$, the variables X and Y are not independent.

Verifying with Chi-Square Test

Let H_0: X and Y are independent Let H_A: X and Y are not independent

```
ctab <- table(traindata$GrLivArea > x, traindata$SalePrice > y)
chitest <- chisq.test(ctab, correct = TRUE)

chitest
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  ctab
## X-squared = 256.53, df = 1, p-value < 2.2e-16
```

Results of Chi-Square test:

The result of the Chi-Square test indicates that the p value is extremely small and much less than 0.05, hence we reject the H_0 hypothesis.

1.6 Descriptive and Inferential Statistics

Summary statistics for Predictor **GrLivArea** X and Response Variable **SalePrice** Y are provided in the table below:

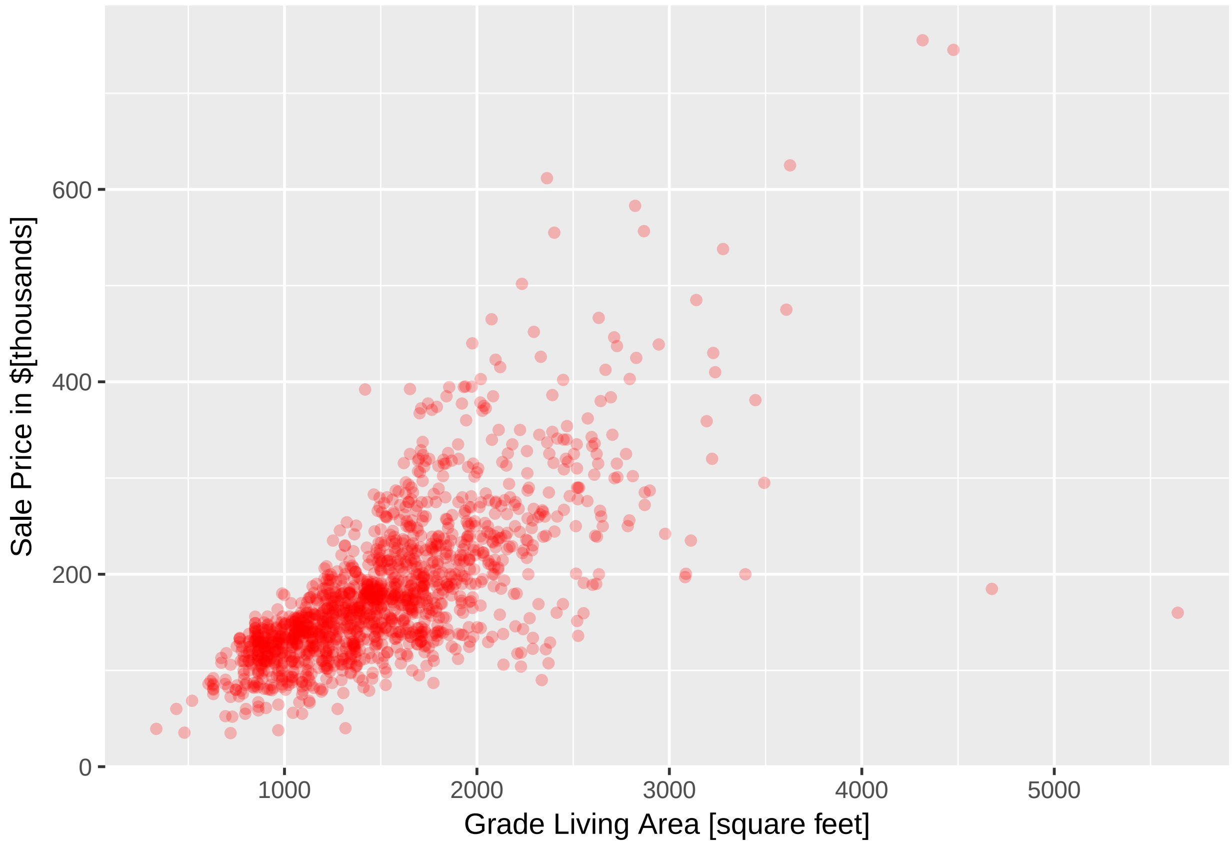
```
# prepare summary table supplemented with standard deviation


sumXY <- rbind(c(summary(X)), c(summary(Y)))
sdXY <- rbind(round(sd(X), 0), round(sd(Y), 0))


sumXYtab <- cbind(sumXY, sdXY)
row.names(sumXYtab) <- c("X (GrLicArea)", "Y (SalePrice)")


pander(sumXYtab)
```

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	
X (GrLicArea)	334	1130	1464	1515	1777	5642	525
Y (SalePrice)	34900	130000	163000	180900	214000	755000	79443

Plot Y Vs X : SalePrice Vs Grade Living Area

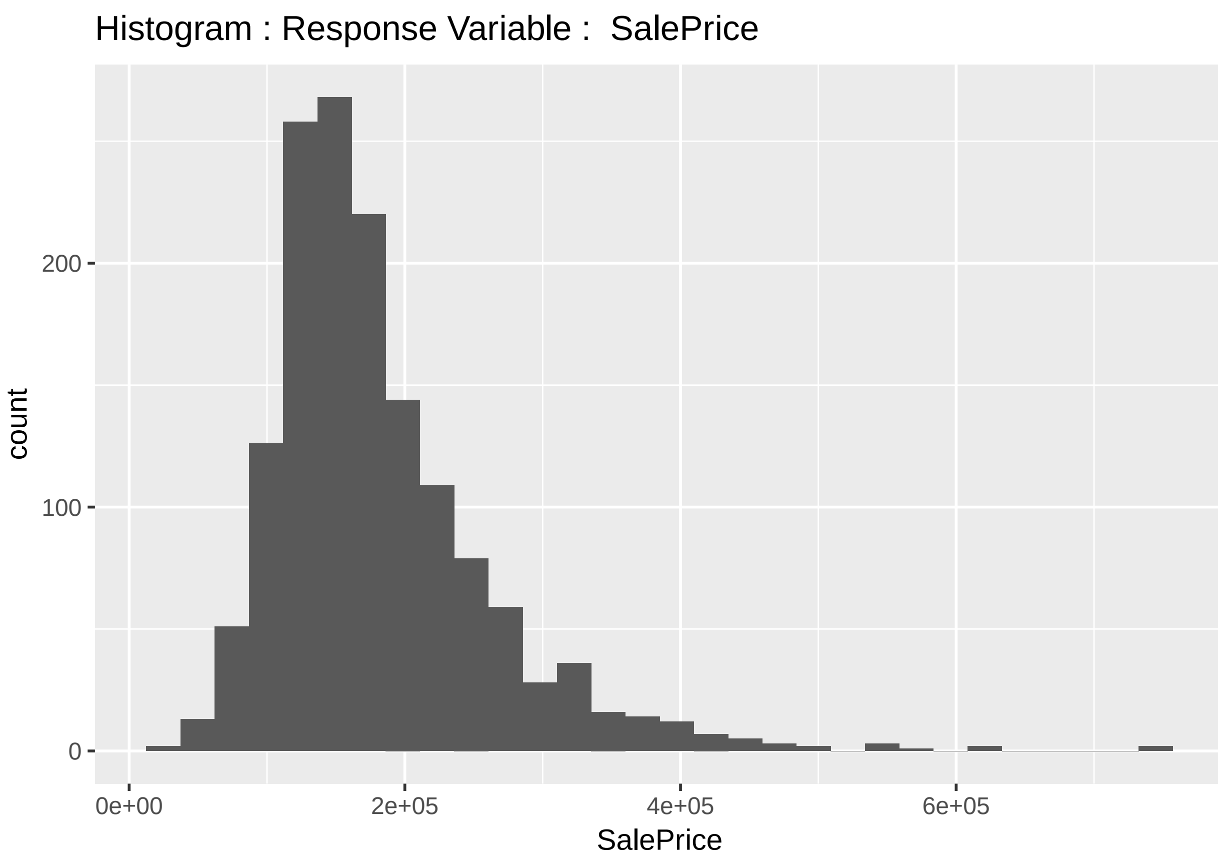
It is intuitive that with an increase in the Living Area , the price of the house would increase. We expect a linear relationship between GrLivArea and SalePrice. We see the same in the plot.

```
ggplot(traindata, aes(x = GrLivArea, y = SalePrice/1000)) + geom_point(alpha = 0.25,
    col = "red") + scale_x_continuous("Grade Living Area [square feet]") + scale_y_continuous("Sale Pri
```

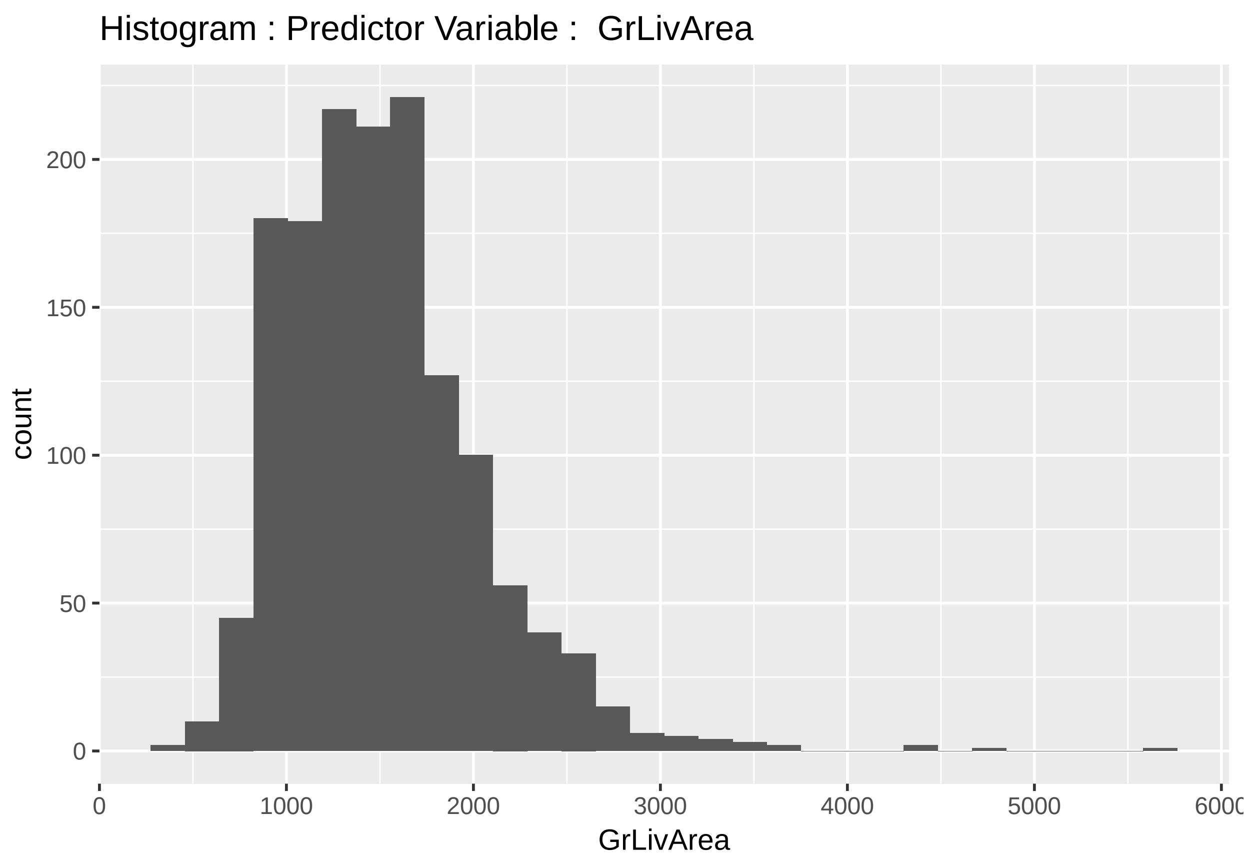


Plot Histogram: SalePrice and Histogram : Living Area

```
hist.SalePrice <- ggplot(traindata, aes(x = SalePrice)) + geom_histogram() + ggtitle("Histogram : Respo

hist.GrLivArea <- ggplot(traindata, aes(x = GrLivArea)) + geom_histogram() + ggtitle("Histogram : Predi

par(mfrow = c(1, 2))
hist.SalePrice
```




```
hist.GrLivArea
```



We find that for both , the response variable **SalePrice** as well as the predictor variable **GrLivArea** , the distributions are strongly right skewed. Both the distributions are crudely normal.

1.6.1 BOXCOX Transformation

We will perform BoxCox Transformation and explore the correlation between the Response variable and Predictor variable.

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.3.3
```

```
lambda.SalePrice <- BoxCox.lambda(traindata$SalePrice)  # Indicates Using As Is , Y^1

lambda.GrLivArea <- BoxCox.lambda(traindata$GrLivArea)  # Indicates Using as Is , X^1
```

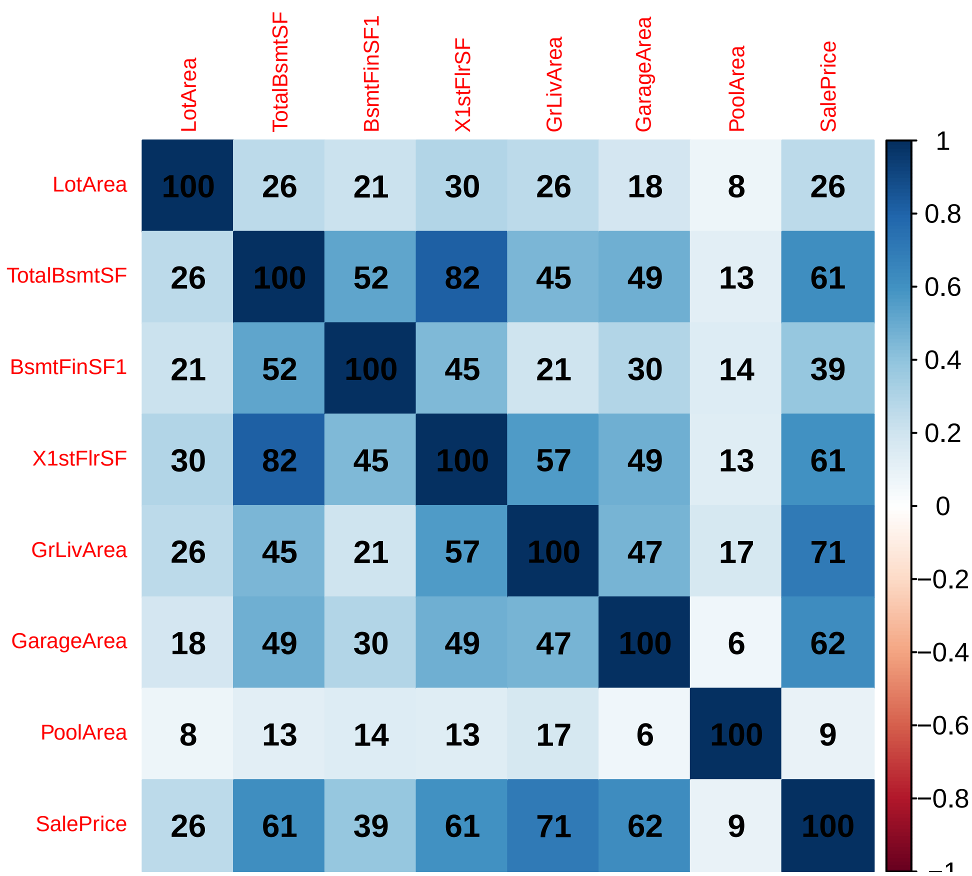
The lambda value for SalePrice and GrLivArea both equal 1 , suggesting transformation of Y^1 for SalePrice and X^1 for GrLivArea, which essentially means no transformation

1.6.2 Correlation Matrix

To check out the correlation among the predictor variables and the response variables and also for any multicollinearity, we plot the pairs plot.

Numerical Variables

```r
cortrain <- dplyr::select(traindata, LotArea, TotalBsmtSF, BsmtFinSF1, X1stFlrSF, GrLivArea, GarageArea

cormat <- as.matrix(cor(cortrain, use = "pairwise.complete.obs"))
corrplot(cormat, method = "color", tl.cex = 0.7, addCoef.col = "black", addCoefasPercent = TRUE)
```



We find that the predictor variable **GrLivArea** is highly correlated to the **SalePrice** at a correlation value of 71%. We also observe other predictors which seem to be significantly correlated viz. TotalBsmtSF, X1stFlrSF, GarageArea. We may further analyze thier significane during modeling.

Verifying with Confidence Interval Test

```r
t.test(traindata$GrLivArea, traindata$SalePrice)
```

```
##
##  Welch Two Sample t-test
##
## data:  traindata$GrLivArea and traindata$SalePrice
## t = -86.288, df = 1459.1, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -183484.2 -175327.3
## sample estimates:
##  mean of x  mean of y
##   1515.464 180921.196
```

```r
t.test(traindata$GrLivArea, traindata$SalePrice)$conf.int
```

```
## [1] -183484.2 -175327.3
```

```
## attr(,"conf.level")
## [1] 0.95
```

A **95% confidence** interval for the difference in the means of X and Y is given by [**-183484.2, -175327.3**] The p-value associated with this hypothesis test is near-zero, so the null hypothesis that there is no correlation between the variables is rejected.

Let H_0: Variable X and Y are not correlated Let H_A: Variable X and Y are correlated

To check programmatically for correlation :

```r
cortest <- cor.test(X, Y, method = "pearson", conf.level = 0.99)
```

A 99% confidence interval for the difference in the means of X and Y is given by [0.6733974, 0.7406408]

The p-value associated with this hypothesis test is almost zero, so the null hypothesis that H_0: Variable X and Y are not correlated is rejected. Therefore, it can be said with **99% Confidence** that there exists a correlation between the GrLivArea and SalePrice and the correlation value lies in between [**0.6733974, 0.7406408**] Also from the plot we see that the correlation of 0.71 exists between GrLivArea and SalePrice

1.7 Linear Algebra And Correlation

Lets invert the correlation matrix, to get the Precision Matrix, which contains the Variance Inflation Factors across the diagonal.

```
corrtraindata <- dplyr::select(traindata, GrLivArea, SalePrice)

# Deriving Correlation MAtrix

cormatrix <- cor(corrtraindata)
pander(cormatrix)
```

	GrLivArea	SalePrice
GrLivArea	1	0.7086
SalePrice	0.7086	1

```
# Inverting Correlation Matrix to get Precision Matrix.
precimatrix <- solve(cormatrix)
pander(precimatrix)
```

	GrLivArea	SalePrice
GrLivArea	2.009	-1.423
SalePrice	-1.423	2.009

Multiplying Correlation Matrix by Precision Matrix, we expect an Identity Matrix

```
multi_cor_preci <- cormatrix %*% precimatrix
pander(multi_cor_preci)  # We get Identity matrix
```

	GrLivArea	SalePrice
GrLivArea	1	0
SalePrice	0	1

Multiplying Precision Matrix by Correlation Matrix, we expect an Identity Matrix

```
multi_preci_cor <- precimatrix %*% cormatrix
pander(multi_preci_cor)  # We get Identity matrix
```

	GrLivArea	SalePrice
GrLivArea	1	0
SalePrice	0	1

1.8 Calculus Based Probability And Statistics

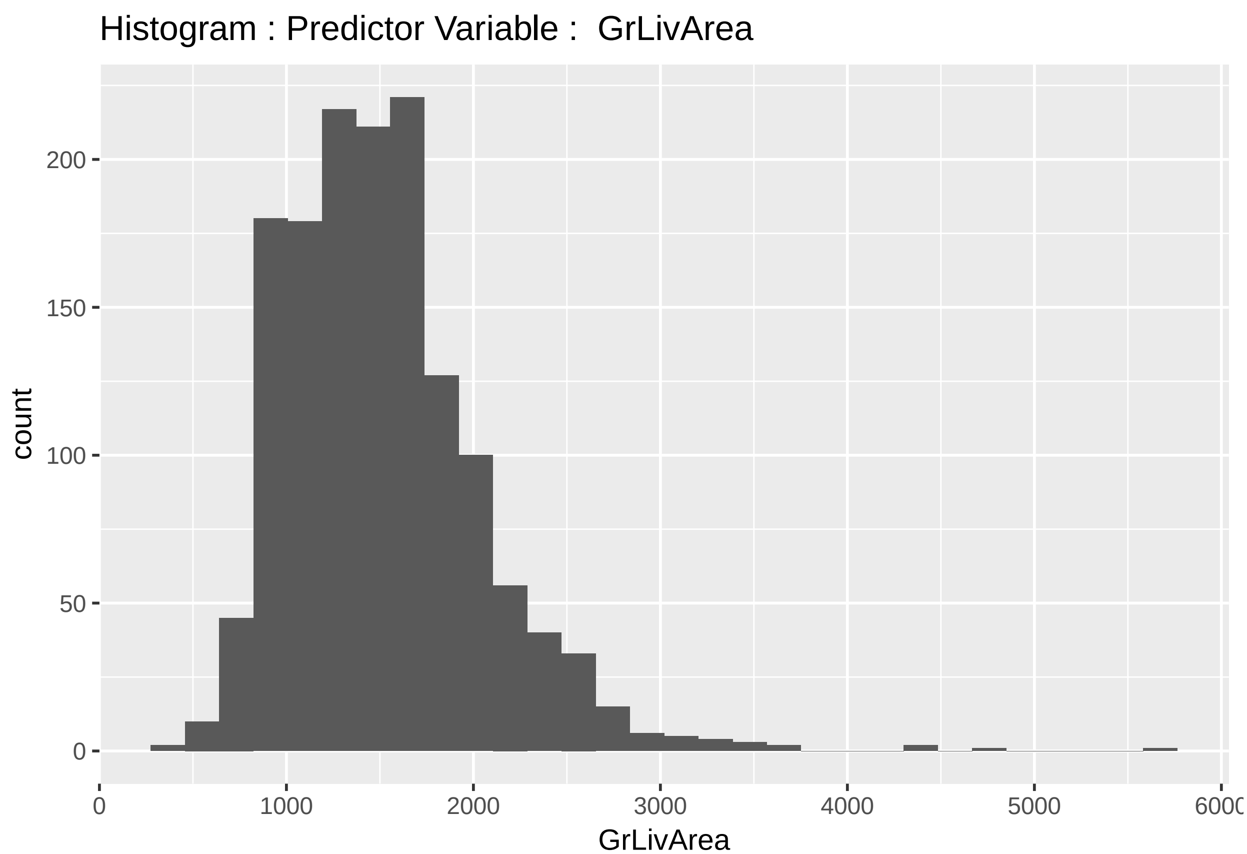
For the independant variable, GrLivArea, we have the minimum value at 334.

The minimum value is above zero. Hence we do not need to add/shift the values above zero. We take an exponential density function to fit.

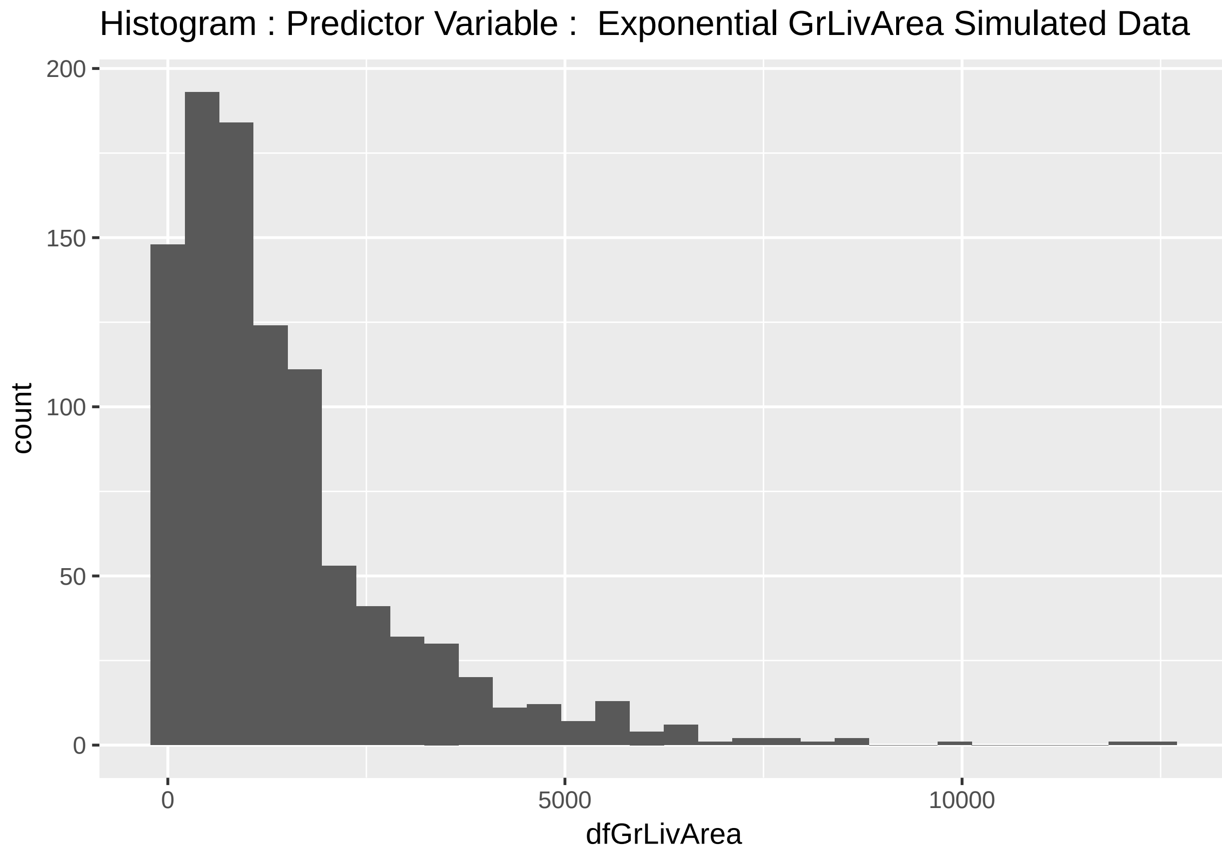
```
# Finding lambda value y fitting GrLivArea to exponential distribution

expoGrLivArea <- fitdistr(traindata$GrLivArea, "exponential")
lambda <- expoGrLivArea$estimate[[1]]
```

The $\lambda = 6.599 \times 10^{-4}$ is the value obtained , which is the optimal value of parameter for this distribution. We take a thousand samples from this distribution. We will compare these values with the original non-transformed values.

```
# Taking 1000 samples of the exponential distribution for GrLivArea

set.seed(100)
GrLivArea.samples.1000 <- rexp(1000, lambda)

dfGrLivArea <- data.frame(GrLivArea.samples.1000)


hist.GrLivArea.exp1000 <- ggplot(data.frame(dfGrLivArea), aes(dfGrLivArea)) + geom_histogram() + ggtitl

par(mfrow = c(1, 2))

hist.GrLivArea
```




```
hist.GrLivArea.exp1000
```




```
mean(GrLivArea.samples.1000)
```

```
## [1] 1472.244
```

Comparing the two histograms , for the original GrLivArea and the Sampled Exponential GrLivArea, we find : 1. The original distribution was appeared crudely normal with very strong skewnees to the right. It was centred with highest frequency and mean around 1515. 2. The simulation of transformed GrLivArea is centred more closer toward 0. Also it is more skewed to the right as compared to the original predictor data.

1.9 Modeling

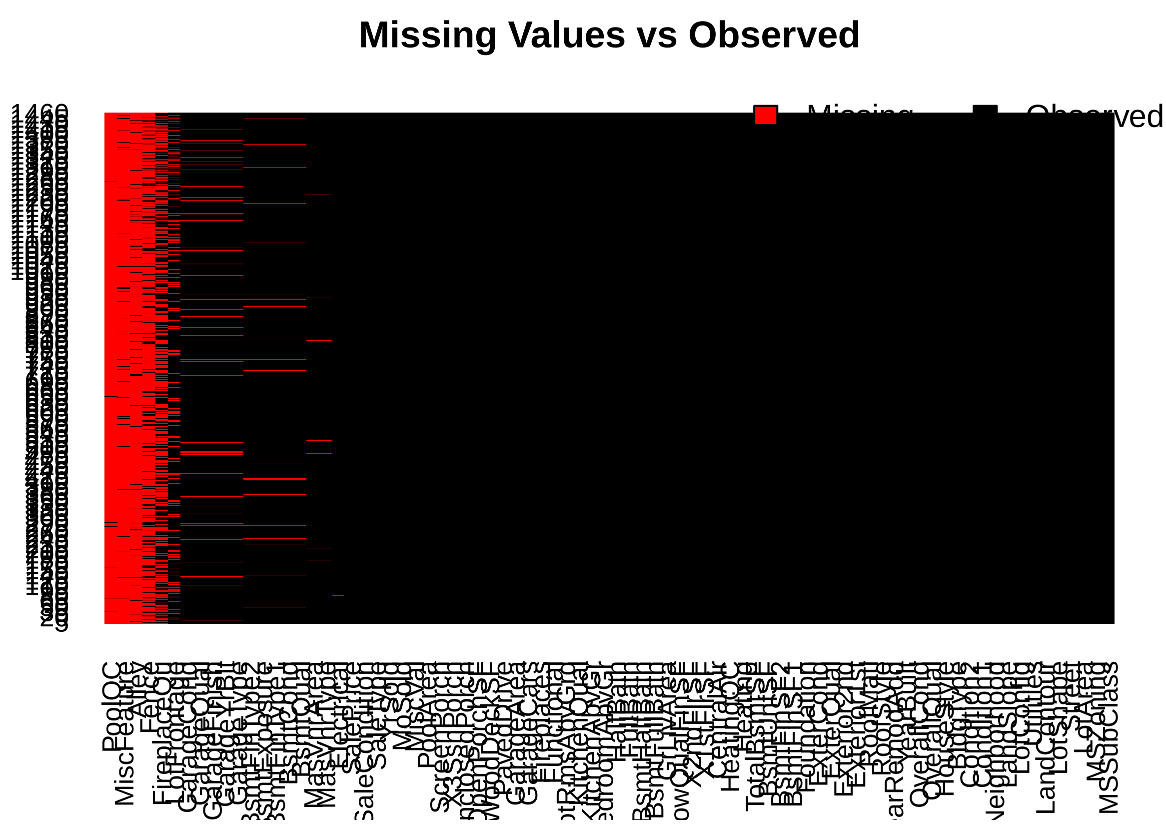
Before we go ahead and build a model, we would need to cleanse the data. From the following missing values map, we find that MasVnrArea, BsmtFinType1, BsmtFinType2 have less number of NAs while GarageType(81 NAs),GarageYrBlt (81 NAs), GarageQual (81 NAs), GarageCond (81 NAs)

Functional has high number of missing values , 1360.Similarly, PoolQC Fence, Alley,FireplaceQu MiscFeature also have high values and are categorical. Since it categorical and also does not appear to be very important one , due to large number of NAs, lets compute our model without considering this.

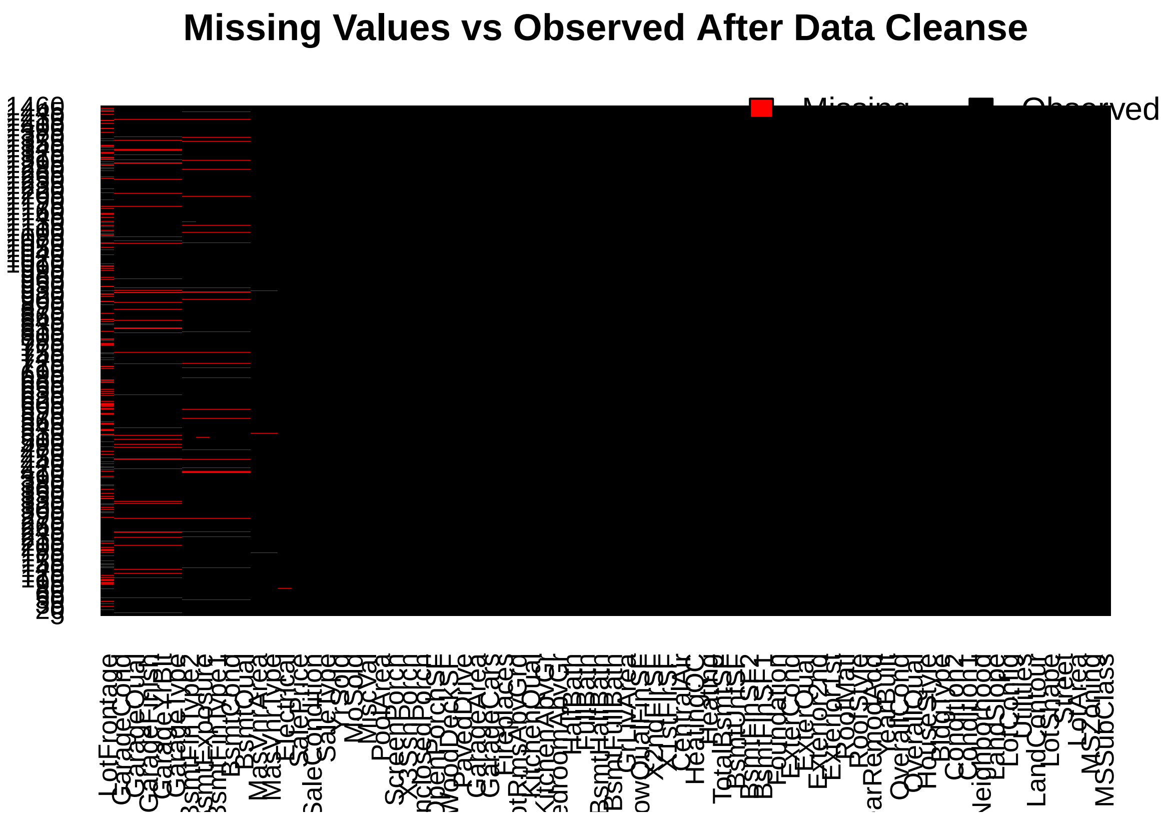
For other numerical variables , we can safely remove these observations for modeling sake, as the amount of traindata is large enough or the study.

So after removing the categorical predictos with high NA values viz. Functional,PoolQC,Fence,MiscFeature, Alley,FireplaceQu, we will proceed with complete cases for building our model.

```
missmap(traindata, legend = TRUE, main = "Missing Values vs Observed", col = c("red", "black"))
```




```
traindatamodel <- dplyr::select(traindata, -Functional, -PoolQC, -Fence, -MiscFeature, -Alley, -Firepla

# Verifying that no NAs are present
missmap(traindatamodel, legend = TRUE, main = "Missing Values vs Observed After Data Cleanse", col = c(
```




```r
traindatamodel <- data.frame(traindatamodel[complete.cases(traindatamodel), ])
nrow(traindatamodel)
```

```
## [1] 1094
```

```r
ncol(traindatamodel)
```

```
## [1] 74
```

We now have 1094 observations in our dataset. Since our evaluation data does not have the SalePrice, we go ahead for splitting our training data so as to crossvalidate the models constructed. We verify model with splitting the train data into 80:20 ratio by randomly selecting the observation data for further analysis of models (since evaluation data lacks the target response variable)

```r
set.seed(100)
randomobs <- sample(seq_len(nrow(traindatamodel)), size = floor(0.8 * nrow(traindatamodel)))

trainnew <- traindatamodel[randomobs, ]
testnew <- traindatamodel[-randomobs, ]

# View(traindatamodel)
```

1.9.1 Model 1 : Random Forest Model

We start with considering the numeric variables since those are found to be more significant. We will apply this to the training dataset and then crossvalidate with the test.

The "caret" package train function is used with method of "random forest"" . A 5 fold cross validation is used. From the summary we see that for 19 mtry (number of variables used), we get the least Root Mean Squared Error value of 31583.31

From the plot of the fit, we observe how the RMSE decreases as the predictors increase , however RMSE is lowest at 19 predictors and then increases back.

The top 19 predictor variables that give the best performance (least RMSE value) are plotted in the Variable Importance Plot below.

We also see this by crossvalidating against the testdata which was partitionaed from the main training dataset , RMSE as 32369.93

```
numetrain <- names(trainnew)[which(sapply(trainnew, is.numeric))]
traindatamodel.n <- trainnew[numetrain]



numetest <- names(testnew)[which(sapply(testnew, is.numeric))]
testdatamodel.n <- testnew[numetest]

# Imputing numeric missing values with 0
testdatamodel.n[is.na(testdatamodel.n)] <- 0


# Random Forest model

modelrf <- train(SalePrice ~ ., data = traindatamodel.n, method = "rf", trControl = trainControl(method
    allowParallel = TRUE)
```

```
## Warning: package 'randomForest' was built under R version 3.3.2
```

```
# show the model summary
summary(modelrf)
```

```
##                 Length Class      Mode
## call                 7 -none-     call
## type                 1 -none-     character
## predicted          875 -none-     numeric
## mse                500 -none-     numeric
## rsq                500 -none-     numeric
## oob.times          875 -none-     numeric
## importance          72 -none-     numeric
## importanceSD        36 -none-     numeric
## localImportance      0 -none-     NULL
## proximity       765625 -none-     numeric
## ntree                1 -none-     numeric
## mtry                 1 -none-     numeric
## forest              11 -none-     list
## coefs                0 -none-     NULL
## y                  875 -none-     numeric
## test                 0 -none-     NULL
## inbag                0 -none-     NULL
## xNames              36 -none-     character
## problemType          1 -none-     character
## tuneValue            1 data.frame list
## obsLevels            1 -none-     logical
```

```r
print(modelrf)
```

```
## Random Forest
##
## 875 samples
##  36 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 700, 702, 700, 700, 698
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared
##    2    34324.36  0.8417271
##   19    31583.31  0.8517676
##   36    32777.99  0.8380481
##
## RMSE was used to select the optimal model using  the smallest value.
## The final value used for the model was mtry = 19.
```

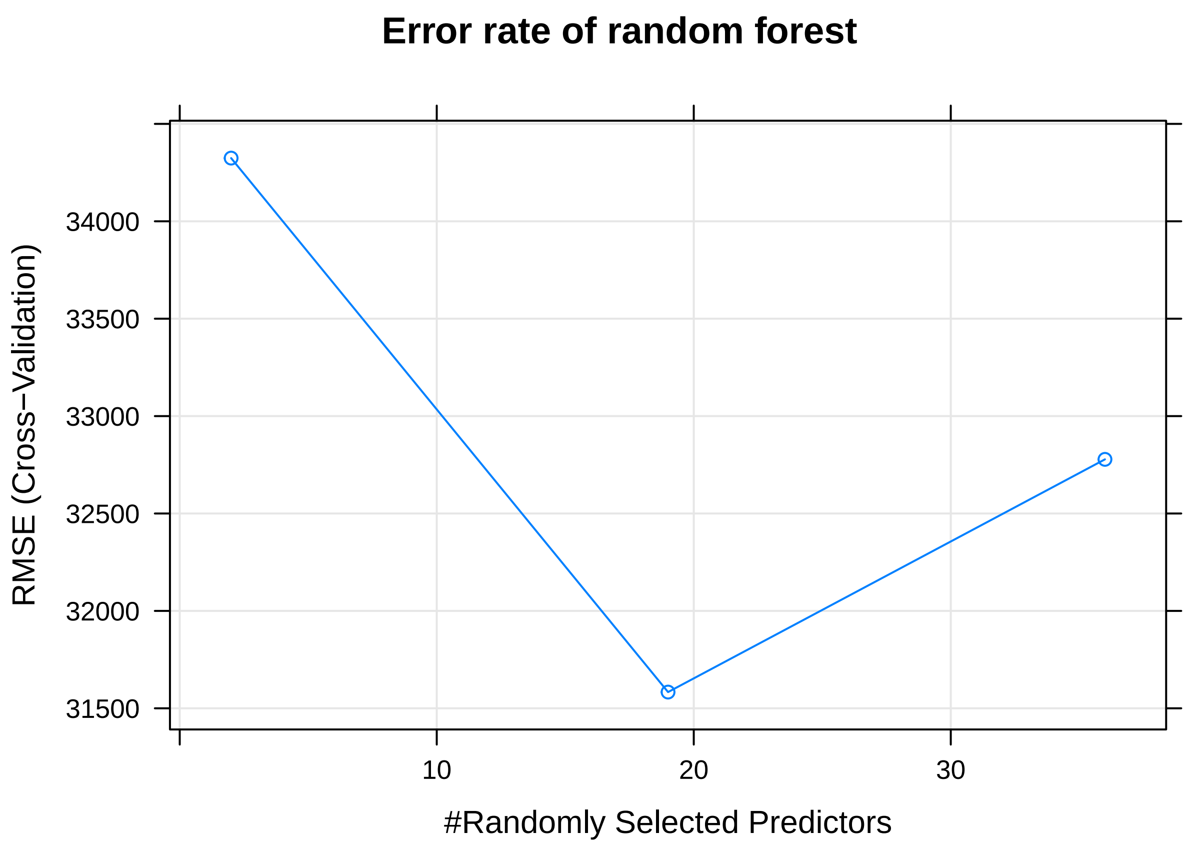
```r
# Crossvalidating for test model
predtest <- predict(modelrf, testdatamodel.n)

# Finding Error MSE
model1.mse <- mean((testdatamodel.n$SalePrice - predtest)^2, na.rm = TRUE)
model1.mse
```

```
## [1] 1047812326
```

```r
model1.rmse <- sqrt(model1.mse)
model1.rmse
```

```
## [1] 32369.93
```

```r
RMSE <- sqrt(sum((predtest - testdatamodel.n$SalePrice)^2)/length(predtest))


plot(modelrf, main = "Error rate of random forest")
```

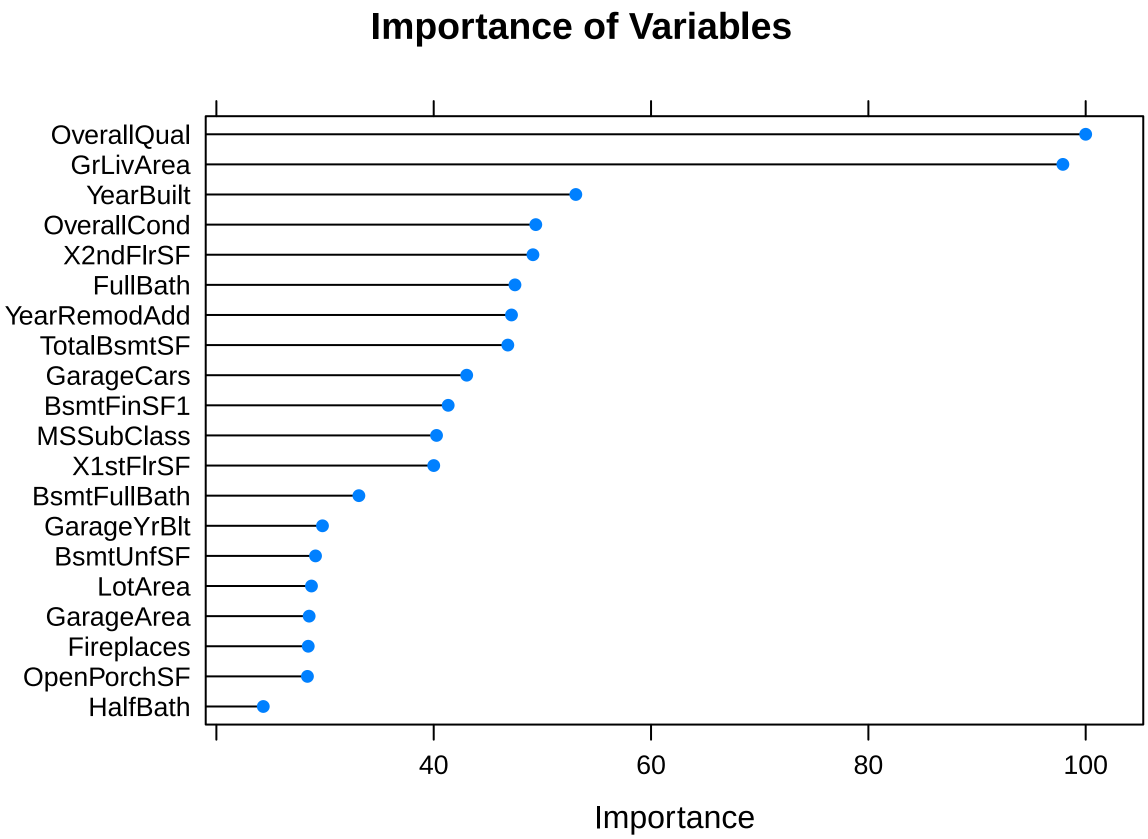



```r
## variable importance

rfImp <- varImp(modelrf)
rfImp
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 36)
##
##              Overall
## OverallQual   100.00
## GrLivArea      97.90
## YearBuilt      53.08
## OverallCond    49.39
## X2ndFlrSF      49.13
## FullBath       47.48
## YearRemodAdd   47.16
## TotalBsmtSF    46.82
## GarageCars     43.04
## BsmtFinSF1     41.33
## MSSubClass     40.26
## X1stFlrSF      40.00
## BsmtFullBath   33.11
## GarageYrBlt    29.76
## BsmtUnfSF      29.12
## LotArea        28.75
```

```
## GarageArea     28.54
## Fireplaces     28.45
## OpenPorchSF    28.38
## HalfBath       24.31
```

```r
plot(rfImp, top = 20, main = "Importance of Variables")
```



The variable importance plot is a critical output of the random forest algorith. For each variable in your matrix it tells you how important that variable is in classifying the data. The plot shows each variable on the y-axis, and their importance on the x-axis. They are ordered top-to-bottom as most- to least-important. We see that **OverallQual**, **GrLivArea**, have the highest importance and impacts the SalePrice of the house. Followed by **YearBuilt**, **OverallCond,**, **X2ndFlrSF**, **FullBath** and others.

1.9.2 Predictions on Evaluation Data

We now use this Model for predicting the housing sale price evaluation dataset.

For submitting to Kaggle , we will apply the model to entire original evaluation dataset.

```r
# Obtaining the numeric variables form the evaluation dataset
numeric_var <- names(evaldataorig)[which(sapply(evaldataorig, is.numeric))]
evaldataorig.n <- evaldataorig[numeric_var]

# Imputing missing values with 0
evaldataorig.n[is.na(evaldataorig.n)] <- 0
```

```r
# Applying the model for predicting SalePrice
predeval <- predict(modelrf, evaldataorig.n)

evalDF <- as.data.frame(cbind(evaldataorig$Id, predeval))
colnames(evalDF) <- c("Id", "SalePrice")
dim(evalDF)  #  1459 rows for Kaggle submission
```

```
## [1] 1459    2
```

```r
# Submitted to Kaggle
write.csv(evalDF, file = "predictedSP_RF.csv", quote = FALSE, row.names = FALSE)


# adding to evaluation data set
evaldataorig$SalePrice <- predeval

# Updated EvaluationCSV with predictions from model
write.csv(evaldataorig, "predicted_SalePrice.csv", row.names = FALSE)
```

Kaggle UserName : kbhave

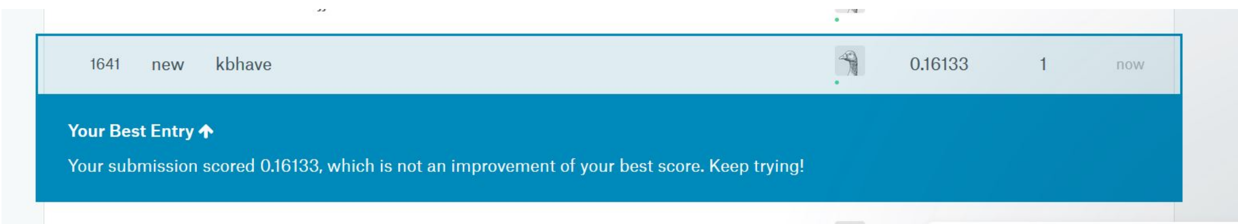


Figure 1: Kaggle Housing Score
