Exploratory Data Analysis of Iowa Housing Data for Simple Regression Modelling

Serra Uzun, MSDS_410 FALL 2020
09/20/2020

Introduction

The Ames dataset that we will be using for our analysis consists of detailed housing information in Iowa. The dataset includes the data regarding the house characteristics, such as style, condition, room counts, areas and other multiple interiors and exterior design elements, and information on properties of the lot it is built on, mechanical systems. The exploratory data analysis (EDA) we will conduct will determine the most key predictor variables within the batch of predictor variables we aim to look into that are most likely to have a strong positive or negative correlation with our response variable.

The Dataset

As the initial step, we study the dataset's properties to better understand the data we currently hold. Our Ames dataset consists of 1,135 observations and 56 variables. From these 56 variables, there is only one response variable and 55 predictor variables that we are looking to downsize for a better and more efficient regression model. For our EDA as well as our regression model, we will be focusing on the variables in Table 1 below:

Variable	Definition	Response or Predictor	Data Type
SalePrice	Sale price of the house	Response	Integer
TotalSqftCalc	Total area of the house in square feet	Predictor	Integer
TotalBathCalc	Total number of baths in the house	Predictor	Numeric
TotRmsAbvGrd	Total number of rooms above ground	Predictor	Integer
QualityIndex	Quality index	Predictor	Integer
OverallQual	Score for overall quality	Predictor	Integer
OverallCond	Score for overall condition	Predictor	Integer

Table 1: Variable Types and Definitions

We observe no missing values within any of the variables, yet we see a possible outlier in our response variable, SalePrice, due to the broad range and high variance. This indicates that we may need to conduct a logarithmic transformation on our response variable to normalize and prevent our analysis from producing skewed, inaccurate results.

Variable	Obs.	Miss.	Min	1st Qtr.	Median	Mean	3rd Qtr.	Max	Range	Variance	Std.	Coef. Var
SalePrice	1,135	0	62,383	142,188	177,500	197,212	229,900	755,000	692,617	5,979,050,377	77,324	0
TotalSqftCalc	1,135	0	825	1,630	1,955	2,122	2,486	5,771	4,946	502,987	709	0
TotalBathCalc	1,135	0	1	2	3	2	3	5	4	1	1	0
TotRmsAbvGrd	1,135	0	4	30	6	7	40	12	8	2	1	0
QualityIndex	1,135	0	12	6	35	34	7	72	60	52	7	0
OverallQual	1,135	0	3	5	6	6	7	10	7	2	1	0
OverallCond	1,135	0	3	5	5	6	6	9	6	1	1	0

Table 2: Descriptive Summary Statistics

Data Analysis & Modeling

The correlation analysis between the predictor variables and response variable is done by having our response variable in a standard form and in a logarithmic transformed form to compare and observe any change in results. Any correlation between 0 and 1 is considered positive, indicating that the variables are positive linear related, whereas the correlation between 0 and -1 is the indicator of a negative linear relationship. The higher negative or positive correlation score a predictor variable receives when analyzed against the response variable indicates a strong linear relationship that needs to be further studied. Below is the SalePrice's, our response variable's correlation with the six predictor variables we have chosen to utilize for our study.

TotalSqftCalc	TotalBathCalc	QualityIndex	TotRmsAbvGrd	OverallQual	OverallCond
0.786	0.671	0.611	0.647	0.825	-0.238

Table 3: Predictor Variable Correlation to Response Variable SalePrice

When the correlation between the selected predictor variables, TotalSqftCalc (Total Square Footage), TotalBathCalc (Total Number of Bathrooms), QualityIndex (Quality Index), TotRmsAbvGrd (Total Number of Rooms Above Grade), Overall Qual (Overall Quality), and finally OverallCond (Overall Condition), and response variable SalePrice modeled we see that five out of six predictor variables have a noticeable positive correlation with the response variable. The predictor variables with the highest positive correlation with SalePrice are OverallQual with 0.825, followed by TotalSqftCalc, TotalBathCalc, TorRmsAbvGrd, and QualityIndex with 0.786, 0.671, 0.647, and 0.611, respectively. OverallCond is the only predictor variable that is negatively correlated with the SalePrice, with a correlation of -0.238, presented in Table 3.

The scatterplots of selected predictor variables plotted against the response variable SalePrice in Figure 1 presents the clear and distinct correlation also was shown in Table 3. The LOESS line in each scatterplot is straight, which means the correlations we have obtained through the correlation analysis and scatter plots of predictor variables against the response variable are distinct and viable.

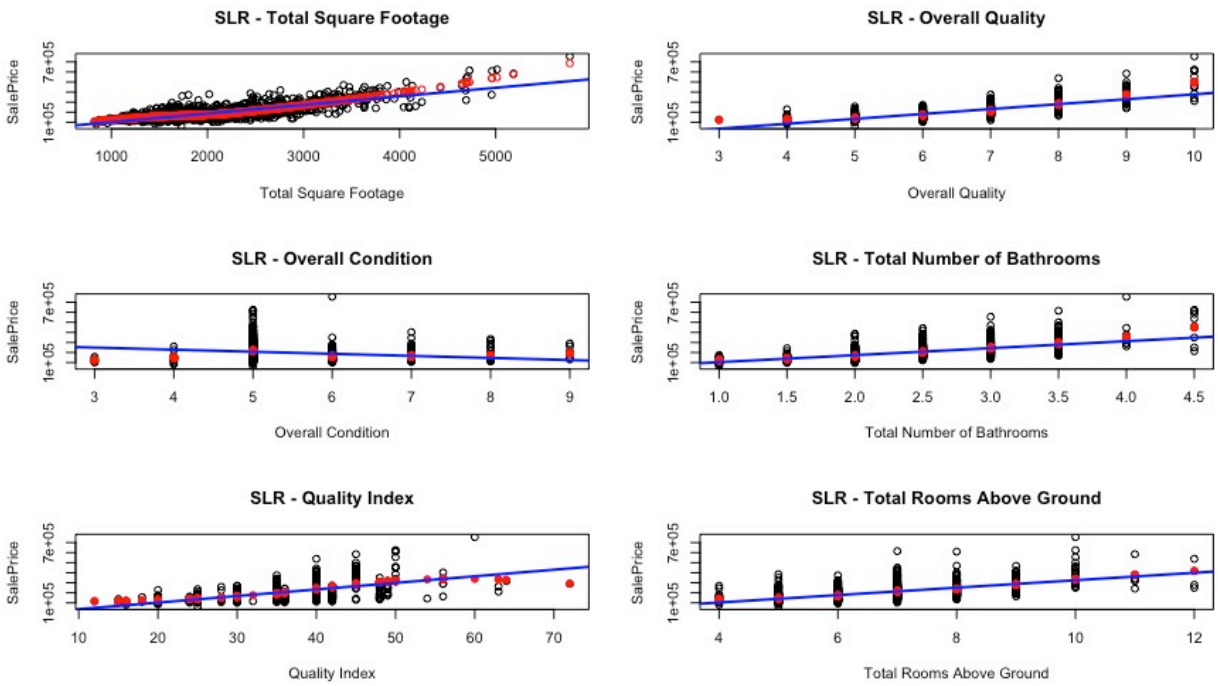


Figure 1: Scatterplots of Selected Variables against Response Variable SalePrice

A secondary step was taken to transform the response variable SalePrice into a logarithmic form to observe improvements in correlation values between predictor variables and log(SalePrice). Table 4 presented the outcome of the correlation matrix ran for predictor variables vs. logarithm of SalePrice. This approach's outcome resulted in a general improvement in correlations between the response variable and all predictors except for TotalSqftCalc. As the decrease in correlation between TotalSqftCal and SalePrice when SalePrice is transformed is not significant, we can conclude that converting SalePrice to logarithmic form has improved our results.

TotalSqftCalc	TotalBathCalc	QualityIndex	TotRmsAbvGrd	OverallQual	OverallCond
0.767	0.728	0.634	0.679	0.860	-0.256

Table 4: Predictor Variable Correlation to Response Variable log(SalePrice)

Figure 2 is showing the scatterplots for logarithmic form SalePrice, and predictor variables present a very similar outcome with the SalePrice scatterplots before response variable transformation yet with a slight improvement in relationships. The LOESS line remains straight, as the correlation between variables is in a feasible condition.

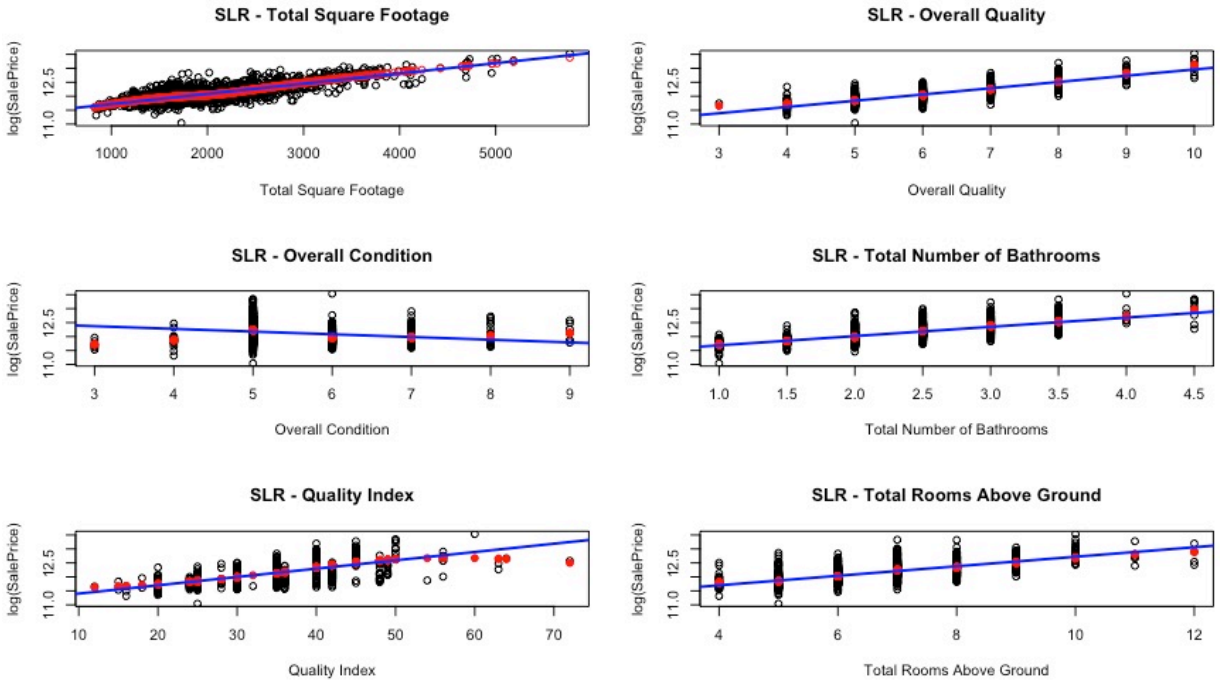


Figure 2: Scatterplots of Selected Variables against Response Variable log(SalePrice)

Upon reviewing our correlation matrices and scatterplots for both SalePrice and log(SalePrice), we can identify that the highest positive correlations the response variables have is with OverallQual, TotalSqftCalc and TotalBathCalc. These three variables are the most critical indicators of the SalePrice of a house in Iowa state. On the other hand, in both matrices and plots, OverallCond was the only predictor variable that was negatively correlated with the response variable. While the negative correlation was slight, we have gained the insight that the increase in OverallCond of the house does not positively impact SalePrice and vice versa.

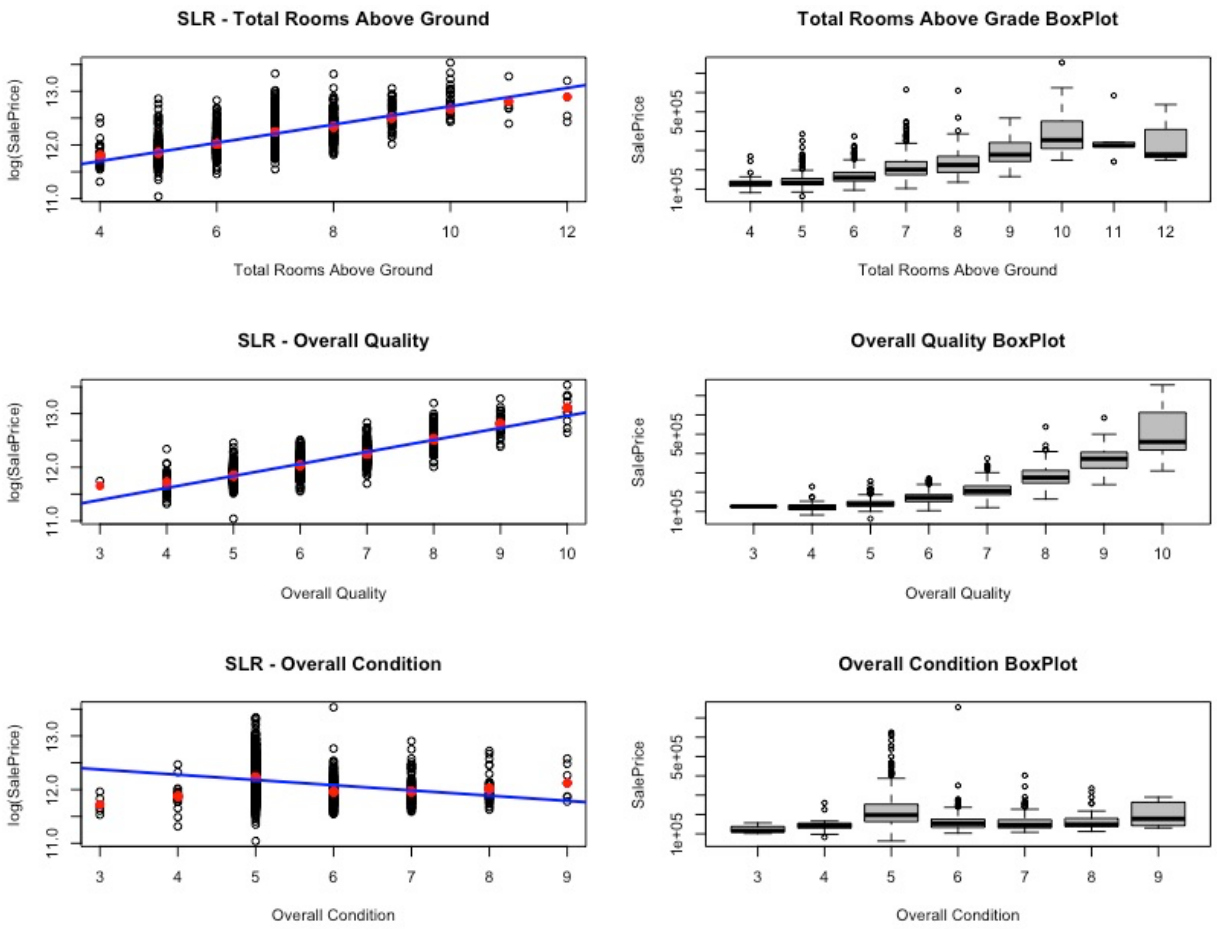


Figure 2: Scatterplots and Boxplots of Three Selected Variables against Response Variable log(SalePrice)

Continuous data is more desired for the type and size of analysis conducted with the Ames dataset. As an example, we can look at the TotalSqftCalc variable, which is the only continuous predictor variable where we will see the LOESS line is a much better fit over the continuous data, therefore producing more accurate and viable results. The scatter and box plots in Figure 2 shows us that OverallQual (Overall Quality) and TotRmsAbvGrd (Total Rooms Above Grade) perform well within the analysis even though they are discrete variables, yet OverallCond (Overall Condition) variable could benefit from being transformed from being a dscrete variable to continuous variable. Therefore, in order to improve our results and better understand the relationship between the SalePrice and Overall Condition, it would be crucial to change the predictor variable from discrete to continuous.

Conclusion

In conclusion, our exploratory data and correlation analysis for the response variable, SalePrice, and selected predictor variables from Ames dataset in general present a significant positive correlation. In addition to analyzing SalePrice, we also investigated the impact of response variable logarithmic transportation on the established correlations. Our analysis showed a slight improvement in correlations as we transformed the response variable, while our top indicators remained as OverallQual, TotalSqftCalc, and TotalBathCalc. Finally, we have determined that the OverallCond predictor variable, which is also the only variable that is negatively correlated to SalePrice, is the predictor variable that could benefit from a transformation from being discrete to continuous. This transformation is likely to improve the results and help us better understand the negative relationship between OverallCond and SalePrice.

The R code for Assignment #1

```
# Serra Uzun
# MSDS_410 Supervised Learning Methods_FALL 2020
# 09.20.2020
# Assignment_01

#Import dataset
ames.df <- readRDS('/Users/serrauzun/Desktop/MSDS_410_Supervised/ames_sample.
Rdata');

# columns and the types using the structure
str(ames.df)

## 'data.frame':    1135 obs. of  56 variables:
##  $ SID            : int  1 2 3 4 5 6 10 11 13 14 ...
##  $ PID            : int  526301100 526350040 526351010 526353030 527105010
527105030 527162130 527163010 527166040 527180040 ...
##  $ LotFrontage    : int  141 80 81 93 74 78 60 75 63 85 ...
##  $ LotArea        : int  31770 11622 14267 11160 13830 9978 7500 10000 840
2 10176 ...
##  $ LotConfig      : chr  "Corner" "Inside" "Corner" "Corner" ...
##  $ Neighborhood   : chr  "NAmes" "NAmes" "NAmes" "NAmes" ...
##  $ HouseStyle     : chr  "1Story" "1Story" "1Story" "1Story" ...
##  $ OverallQual    : int  6 5 6 7 5 6 7 6 6 7 ...
##  $ OverallCond    : int  5 6 6 5 5 6 5 5 5 5 ...
##  $ YearBuilt      : int  1960 1961 1958 1968 1997 1998 1999 1993 1998 1990
...
##  $ YearRemodel    : int  1960 1961 1958 1968 1998 1998 1999 1994 1998 1990
...
##  $ Exterior1      : chr  "BrkFace" "VinylSd" "Wd Sdng" "BrkFace" ...
##  $ BsmtFinSF1     : int  639 468 923 1065 791 602 0 0 0 637 ...
##  $ BsmtFinSF2     : int  0 144 0 0 0 0 0 0 0 0 ...
##  $ CentralAir     : chr  "Y" "Y" "Y" "Y" ...
##  $ GrLivArea      : int  1656 896 1329 2110 1629 1604 1804 1655 1465 1341
...
##  $ BsmtFullBath   : int  1 0 0 1 0 0 0 0 0 1 ...
##  $ BsmtHalfBath   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ FullBath       : int  1 1 1 2 2 2 2 2 2 1 ...
##  $ HalfBath       : int  0 0 1 1 1 1 1 1 1 1 ...
##  $ BedroomAbvGr   : int  3 2 3 3 3 3 3 3 3 2 ...
##  $ TotRmsAbvGrd   : int  7 5 6 8 6 7 7 7 7 5 ...
##  $ Fireplaces     : int  2 0 0 2 1 1 1 1 1 1 ...
##  $ GarageCars     : int  2 1 1 2 2 2 2 2 2 2 ...
##  $ GarageArea     : int  528 730 312 522 482 470 442 440 393 506 ...
##  $ WoodDeckSF     : int  210 140 393 0 212 360 140 157 0 192 ...
##  $ OpenPorchSF    : int  62 0 36 0 34 36 60 84 75 0 ...
##  $ EnclosedPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ThreeSsnPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ScreenPorch    : int  0 120 0 0 0 0 0 0 0 0 ...
```

```
##  $ PoolArea       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ MoSold         : int  5 6 6 4 3 6 6 4 5 2 ...
##  $ YrSold         : int  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010
...
##  $ SaleCondition  : chr  "Normal" "Normal" "Normal" "Normal" ...
##  $ SalePrice      : int  215000 105000 172000 244000 189900 195500 189000
175900 180400 171500 ...
##  $ TotalSqftCalc  : int  2295 1508 2252 3175 2420 2206 1804 1655 1465 1978
...
##  $ TotalBathCalc  : num  2 1 1.5 3.5 2.5 2.5 2.5 2.5 2.5 2.5 ...
##  $ CornerLotInd   : num  1 0 1 1 0 0 0 1 0 0 ...
##  $ FireplaceInd1  : num  0 0 0 0 1 1 1 1 1 1 ...
##  $ FireplaceInd2  : num  1 0 0 1 0 0 0 0 0 0 ...
##  $ FireplaceAdder1: num  1 0 0 1 1 1 1 1 1 1 ...
##  $ FireplaceAdder2: num  1 0 0 1 0 0 0 0 0 0 ...
##  $ CentralAirInd  : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ BrickInd       : num  1 0 0 1 0 0 0 0 0 0 ...
##  $ VinylSidingInd : num  0 1 0 0 1 1 1 0 1 0 ...
##  $ PoolInd        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ WoodDeckInd    : num  1 1 1 0 1 1 1 1 0 1 ...
##  $ PorchInd       : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ QualityIndex   : int  30 30 36 35 25 36 35 30 30 35 ...
##  $ I2006          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ I2007          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ I2008          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ I2009          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ I2010          : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ u              : num  0.74166 0.88833 0.00677 0.23496 0.50085 ...
##  $ train          : num  0 0 1 1 1 1 1 1 1 1 ...
##  - attr(*, "na.action")= 'omit' Named int  9 17 18 19 31 33 34 49 51 56 ..
.
##   ..- attr(*, "names")= chr  "12" "23" "24" "25" ...
```

```
colnames(ames.df)
```

```
##  [1] "SID"             "PID"             "LotFrontage"     "LotArea"
##  [5] "LotConfig"       "Neighborhood"    "HouseStyle"      "OverallQual"
##  [9] "OverallCond"     "YearBuilt"       "YearRemodel"     "Exterior1"
## [13] "BsmtFinSF1"      "BsmtFinSF2"      "CentralAir"      "GrLivArea"
## [17] "BsmtFullBath"    "BsmtHalfBath"    "FullBath"        "HalfBath"
## [21] "BedroomAbvGr"    "TotRmsAbvGrd"    "Fireplaces"      "GarageCars"
## [25] "GarageArea"      "WoodDeckSF"      "OpenPorchSF"     "EnclosedPorch"
## [29] "ThreeSsnPorch"   "ScreenPorch"     "PoolArea"        "MoSold"
## [33] "YrSold"          "SaleCondition"   "SalePrice"       "TotalSqftCalc"
## [37] "TotalBathCalc"   "CornerLotInd"    "FireplaceInd1"   "FireplaceInd2"
## [41] "FireplaceAdder1" "FireplaceAdder2" "CentralAirInd"   "BrickInd"
## [45] "VinylSidingInd"  "PoolInd"         "WoodDeckInd"     "PorchInd"
## [49] "QualityIndex"    "I2006"           "I2007"           "I2008"
## [53] "I2009"           "I2010"           "u"               "train"
```

```r
# subset of predictor variables;
small.df <- ames.df[,c('SalePrice','TotalSqftCalc','TotalBathCalc','QualityIndex',
    'TotRmsAbvGrd','OverallQual','OverallCond')];
str(small.df)
```

```
## 'data.frame':    1135 obs. of  7 variables:
##  $ SalePrice    : int  215000 105000 172000 244000 189900 195500 189000 175900 180400 171500 ...
##  $ TotalSqftCalc: int  2295 1508 2252 3175 2420 2206 1804 1655 1465 1978 ...
##  $ TotalBathCalc: num  2 1 1.5 3.5 2.5 2.5 2.5 2.5 2.5 2.5 ...
##  $ QualityIndex : int  30 30 36 35 25 36 35 30 30 35 ...
##  $ TotRmsAbvGrd : int  7 5 6 8 6 7 7 7 7 5 ...
##  $ OverallQual  : int  6 5 6 7 5 6 7 6 6 7 ...
##  $ OverallCond  : int  5 6 6 5 5 6 5 5 5 5 ...
```

```r
# count of missing values for each variable
sapply(small.df, function(x) sum(is.na(x)))
```

```
##     SalePrice TotalSqftCalc TotalBathCalc  QualityIndex  TotRmsAbvGrd
##             0             0             0             0             0
##   OverallQual   OverallCond
##             0             0
```

```r
#summary statistics of small.df
summary(small.df)
```

```
##    SalePrice      TotalSqftCalc  TotalBathCalc    QualityIndex   TotRmsAbvGrd
##  Min.   : 62383   Min.   : 825   Min.   :1.000   Min.   :12.0   Min.   : 4.00
##  1st Qu.:142188   1st Qu.:1630   1st Qu.:2.000   1st Qu.:30.0   1st Qu.: 6.00
##  Median :177500   Median :1955   Median :2.500   Median :35.0   Median : 6.00
##  Mean   :197212   Mean   :2122   Mean   :2.338   Mean   :34.4   Mean   : 6.53
##  3rd Qu.:229900   3rd Qu.:2486   3rd Qu.:3.000   3rd Qu.:40.0   3rd Qu.: 7.00
##  Max.   :755000   Max.   :5771   Max.   :4.500   Max.   :72.0   Max.   :12.00
##   OverallQual      OverallCond
##  Min.   : 3.000   Min.   :3.000
##  1st Qu.: 5.000   1st Qu.:5.000
##  Median : 6.000   Median :5.000
##  Mean   : 6.302   Mean   :5.515
##  3rd Qu.: 7.000   3rd Qu.:6.000
##  Max.   :10.000   Max.   :9.000
```

```r
#a more detailed descriptive summary with stat.desc
install.packages("pastecs",repos = "http://cran.us.r-project.org")
```

```
## Installing package into '/Users/serrauzun/Library/R/3.5/library'
## (as 'lib' is unspecified)

##
## The downloaded binary packages are in
##  /var/folders/7x/6nv1vk957xl_frh93z69k4cm0000gn/T//RtmpKIHNxN/downloaded_p
ackages
```

```r
library(pastecs)
small.df.summary <- as.data.frame(t(round(stat.desc(small.df), 1)))
small.df.summary
```

```
##               nbr.val nbr.null nbr.na   min      max    range         sum
## SalePrice        1135        0      0 62383 755000.0 692617.0 223835036.0
## TotalSqftCalc    1135        0      0   825   5771.0   4946.0   2408234.0
## TotalBathCalc    1135        0      0     1      4.5      3.5      2653.5
## QualityIndex     1135        0      0    12     72.0     60.0     39044.0
## TotRmsAbvGrd     1135        0      0     4     12.0      8.0      7411.0
## OverallQual      1135        0      0     3     10.0      7.0      7153.0
## OverallCond      1135        0      0     3      9.0      6.0      6260.0
##                 median     mean SE.mean CI.mean.0.95          var std.dev
## SalePrice     177500.0 197211.5  2295.2       4503.3 5979050376.5 77324.3
## TotalSqftCalc   1955.0   2121.8    21.1         41.3     502986.5   709.2
## TotalBathCalc      2.5      2.3     0.0          0.0          0.6     0.7
## QualityIndex      35.0     34.4     0.2          0.4         52.0     7.2
## TotRmsAbvGrd       6.0      6.5     0.0          0.1          1.8     1.4
## OverallQual        6.0      6.3     0.0          0.1          1.7     1.3
## OverallCond        5.0      5.5     0.0          0.1          0.8     0.9
##               coef.var
## SalePrice          0.4
## TotalSqftCalc      0.3
## TotalBathCalc      0.3
## QualityIndex       0.2
## TotRmsAbvGrd       0.2
## OverallQual        0.2
## OverallCond        0.2
```

```r
##############################################################################
########
# Correlation Plots
##############################################################################
########

# Install and load the corrplot package
#install.packages('corrplot', dependencies=TRUE)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
# Correlations with SalePrice;
cor(small.df)
```

```
##                SalePrice TotalSqftCalc TotalBathCalc QualityIndex TotRmsAb
vGrd
## SalePrice      1.0000000     0.7856833     0.6714113    0.6107868    0.647
2545
## TotalSqftCalc  0.7856833     1.0000000     0.7032977    0.4222650    0.590
0370
## TotalBathCalc  0.6714113     0.7032977     1.0000000    0.4028544    0.564
9834
## QualityIndex   0.6107868     0.4222650     0.4028544    1.0000000    0.375
0965
## TotRmsAbvGrd   0.6472545     0.5900370     0.5649834    0.3750965    1.000
0000
## OverallQual    0.8245066     0.5465695     0.5954692    0.7325319    0.584
3620
## OverallCond   -0.2384085    -0.1485253    -0.2368500    0.4102262   -0.260
9945
##               OverallQual OverallCond
## SalePrice       0.8245066  -0.2384085
## TotalSqftCalc   0.5465695  -0.1485253
## TotalBathCalc   0.5954692  -0.2368500
## QualityIndex    0.7325319   0.4102262
## TotRmsAbvGrd    0.5843620  -0.2609945
## OverallQual     1.0000000  -0.3095300
## OverallCond    -0.3095300   1.0000000
```

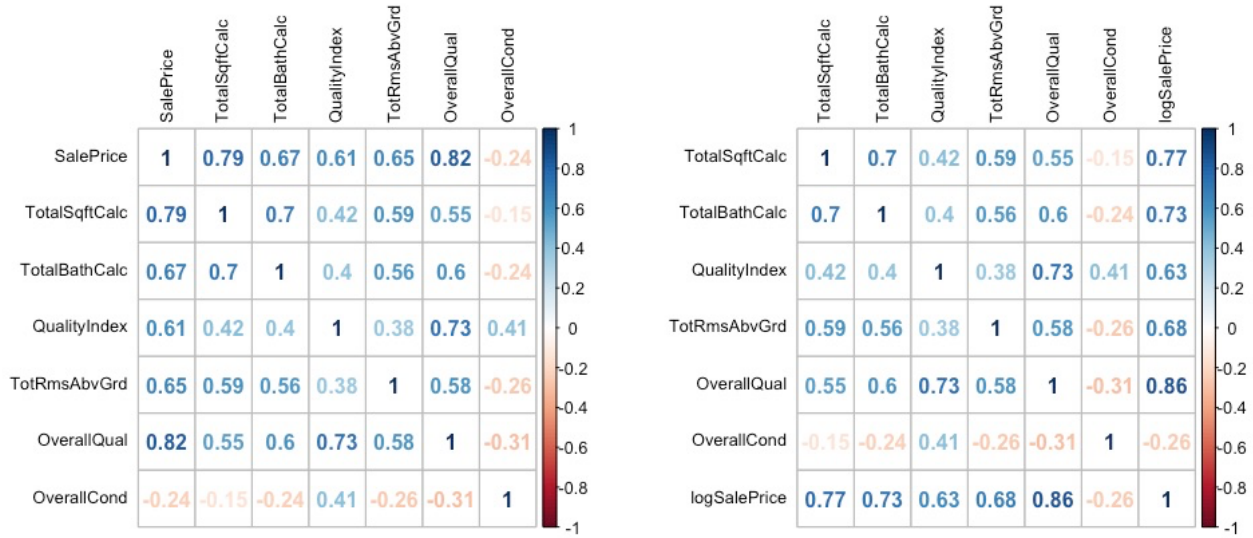
```r
# Correlations with log(SalePrice);
# Need to drop SalePrice and add logSalePrice;
log.df <- subset(small.df, select=-c(SalePrice));
head(log.df)
```

```
##   TotalSqftCalc TotalBathCalc QualityIndex TotRmsAbvGrd OverallQual Overal
lCond
## 1          2295           2.0           30            7           6
5
## 2          1508           1.0           30            5           5
6
## 3          2252           1.5           36            6           6
6
## 4          3175           3.5           35            8           7
5
## 5          2420           2.5           25            6           5
5
## 6          2206           2.5           36            7           6
6
```

```r
log.df$logSalePrice <- log(small.df$SalePrice);
head(log.df)
```

```
##   TotalSqftCalc TotalBathCalc QualityIndex TotRmsAbvGrd OverallQual Overal
lCond
## 1          2295           2.0           30            7           6
5
## 2          1508           1.0           30            5           5
6
## 3          2252           1.5           36            6           6
6
## 4          3175           3.5           35            8           7
5
## 5          2420           2.5           25            6           5
5
## 6          2206           2.5           36            7           6
6
##   logSalePrice
## 1     12.27839
## 2     11.56172
## 3     12.05525
## 4     12.40492
## 5     12.15425
## 6     12.18332
```

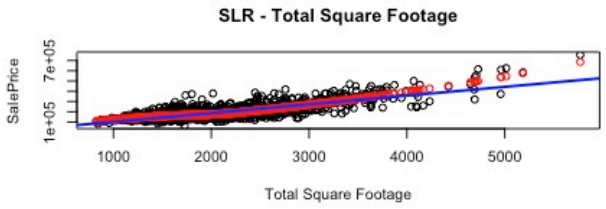
```r
sales_cor <- as.data.frame(cor(small.df))
sales_log_cor <- as.data.frame(cor(log.df))

par(mfrow = c(1,2))
corrplot(cor(small.df),method='number',tl.cex = 0.8, tl.col = "black")
corrplot(cor(log.df),method='number', tl.cex = 0.8, tl.col = "black")
```

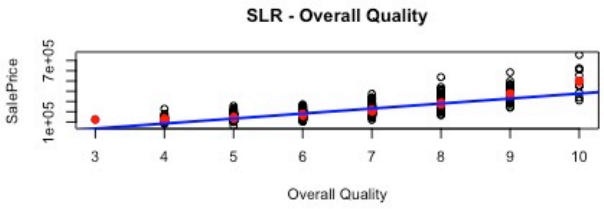


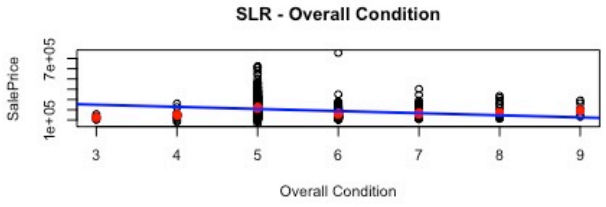

```r
##############################################################################
########
# Scatterplots, Scatterplot Smoothers, and Simple Linear Regression
# Response Variable: SalePrice
##############################################################################
```

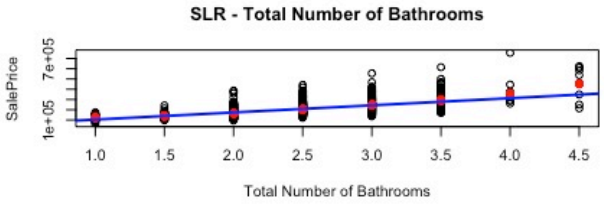
```
########
par(mfrow = c(3,2))

### TotalSfqtCalc
loess.1 <- loess(SalePrice ~ TotalSqftCalc,data=small.df);
lm.1 <- lm(SalePrice ~ TotalSqftCalc,data=small.df);

plot(small.df$TotalSqftCalc, small.df$SalePrice,xlab='Total Square Footage',y
lab='SalePrice')
points(loess.1$x,loess.1$fitted,type='p',col='red')
abline(coef=lm.1$coef,col='blue',lwd=2)
title('SLR - Total Square Footage')

### OverallQual
loess.2 <- loess(SalePrice ~ OverallQual,data=small.df);

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 6

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

lm.2 <- lm(SalePrice ~ OverallQual,data=small.df);

plot(small.df$SalePrice ~ small.df$OverallQual,xlab='Overall Quality',ylab='S
alePrice')
points(loess.2$x,loess.2$fitted,type='p',col='red',pch=19)
abline(coef=lm.2$coef,col='blue',lwd=2)
title('SLR - Overall Quality')

# Note that loess() outputs warning messages.  Can we guess why?

### OverallCond
loess.3 <- loess(SalePrice ~ OverallCond,data=small.df);

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 5

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

lm.3 <- lm(SalePrice ~ OverallCond,data=small.df);
```

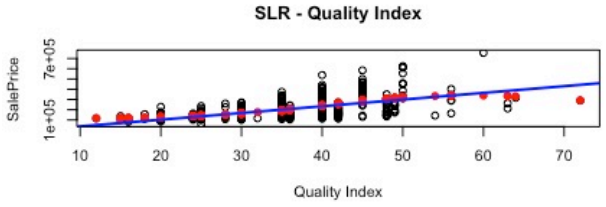
```r
plot(small.df$SalePrice ~ small.df$OverallCond,xlab='Overall Condition',ylab=
'SalePrice')
points(loess.3$x,loess.3$fitted,type='p',col='red',pch=19)
abline(coef=lm.3$coef,col='blue',lwd=2)
title('SLR - Overall Condition')

### TotalBathCalc
loess.4 <- loess(SalePrice ~ TotalBathCalc ,data=small.df);
lm.4 <- lm(SalePrice ~ TotalBathCalc,data=small.df);

plot(small.df$SalePrice ~ small.df$TotalBathCalc,xlab='Total Number of Bathro
oms',ylab='SalePrice')
points(loess.4$x,loess.4$fitted,type='p',col='red',pch=19)
abline(coef=lm.4$coef,col='blue',lwd=2)
title('SLR - Total Number of Bathrooms')

### QualityIndex
loess.5 <- loess(SalePrice ~ QualityIndex ,data=small.df);
lm.5 <- lm(SalePrice ~ QualityIndex,data=small.df);

plot(small.df$SalePrice ~ small.df$QualityIndex,xlab='Quality Index',ylab='Sa
lePrice')
points(loess.5$x,loess.5$fitted,type='p',col='red',pch=19)
abline(coef=lm.5$coef,col='blue',lwd=2)
title('SLR - Quality Index')

### TotRmsAbvGrd
loess.6 <- loess(SalePrice ~ TotRmsAbvGrd ,data=small.df);

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 6

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

lm.6 <- lm(SalePrice ~ TotRmsAbvGrd,data=small.df);

plot(small.df$SalePrice ~ small.df$TotRmsAbvGrd,xlab='Total Rooms Above Groun
d',ylab='SalePrice')
points(loess.6$x,loess.6$fitted,type='p',col='red',pch=19)
abline(coef=lm.6$coef,col='blue',lwd=2)
title('SLR - Total Rooms Above Ground')
```

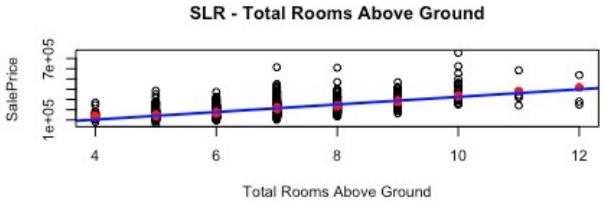








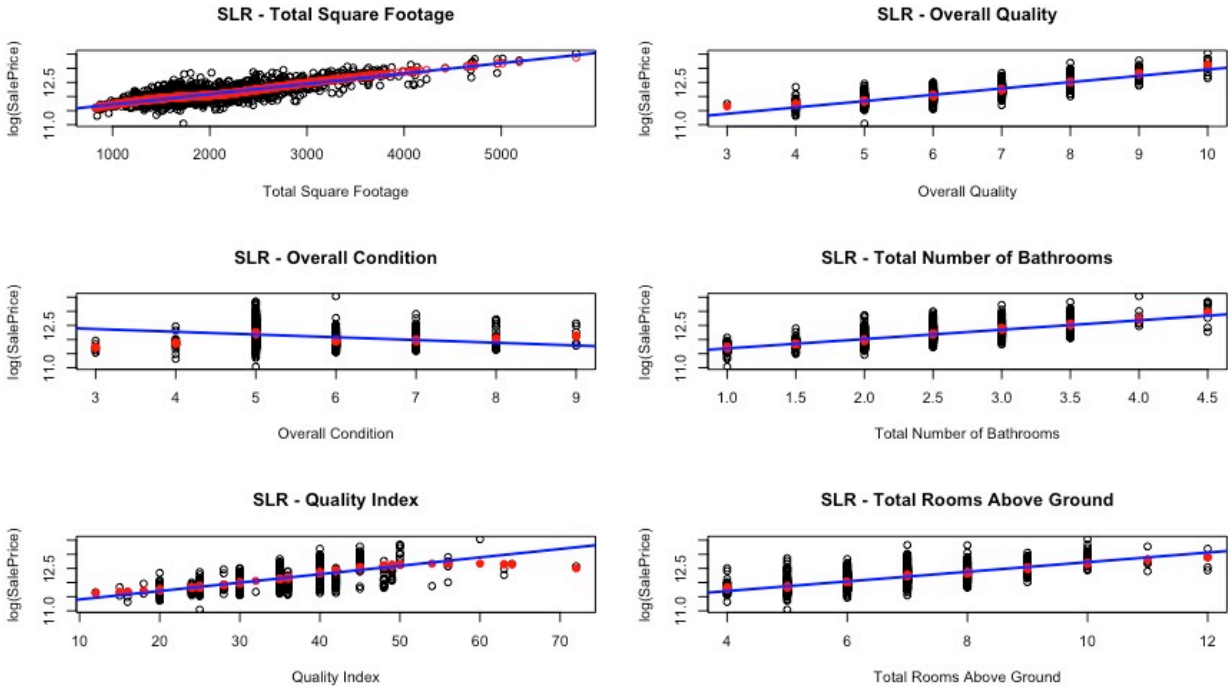





```
################################################################################
########
# Scatterplots, Scatterplot Smoothers, and Simple Linear Regression
# Response Variable: log(SalePrice)
################################################################################
########

par(mfrow = c(3,2))

### TotalSfqtCalc
loess.1 <- loess(log(SalePrice) ~ TotalSqftCalc,data=small.df);
lm.1 <- lm(log(SalePrice) ~ TotalSqftCalc,data=small.df);

plot(small.df$TotalSqftCalc, log(small.df$SalePrice),xlab='Total Square Foota
ge',ylab='log(SalePrice)')
points(loess.1$x,loess.1$fitted,type='p',col='red')
abline(coef=lm.1$coef,col='blue',lwd=2)
title('SLR - Total Square Footage')

### OverallQual
loess.2 <- loess(log(SalePrice) ~ OverallQual,data=small.df);

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 6

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

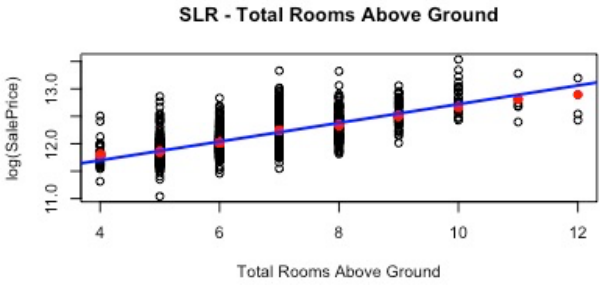
```r
lm.2 <- lm(log(SalePrice) ~ OverallQual,data=small.df);

plot(log(small.df$SalePrice) ~ small.df$OverallQual,xlab='Overall Quality',yl
ab='log(SalePrice)')
points(loess.2$x,loess.2$fitted,type='p',col='red',pch=19)
abline(coef=lm.2$coef,col='blue',lwd=2)
title('SLR - Overall Quality')

# Note that loess() outputs warning messages.  Can we guess why?

### OverallCond
loess.3 <- loess(log(SalePrice) ~ OverallCond,data=small.df);

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 5

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 1

lm.3 <- lm(log(SalePrice) ~ OverallCond,data=small.df);

plot(log(small.df$SalePrice) ~ small.df$OverallCond,xlab='Overall Condition',
ylab='log(SalePrice)')
points(loess.3$x,loess.3$fitted,type='p',col='red',pch=19)
abline(coef=lm.3$coef,col='blue',lwd=2)
title('SLR - Overall Condition')

### TotalBathCalc
loess.4 <- loess(log(SalePrice) ~ TotalBathCalc,data=small.df);
lm.4 <- lm(log(SalePrice) ~ TotalBathCalc,data=small.df);

plot(log(small.df$SalePrice) ~ small.df$TotalBathCalc,xlab='Total Number of B
athrooms',ylab='log(SalePrice)')
points(loess.4$x,loess.4$fitted,type='p',col='red',pch=19)
abline(coef=lm.4$coef,col='blue',lwd=2)
title('SLR - Total Number of Bathrooms')

### QualityIndex
loess.5 <- loess(log(SalePrice) ~ QualityIndex,data=small.df);
lm.5 <- lm(log(SalePrice) ~ QualityIndex,data=small.df);

plot(log(small.df$SalePrice) ~ small.df$QualityIndex,xlab='Quality Index',yla
b='log(SalePrice)')
points(loess.5$x,loess.5$fitted,type='p',col='red',pch=19)
abline(coef=lm.5$coef,col='blue',lwd=2)
```

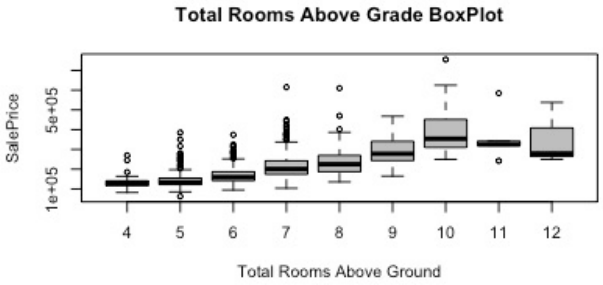
```
title('SLR - Quality Index')

### TotRmsAbvGrd
loess.6 <- loess(log(SalePrice) ~ TotRmsAbvGrd,data=small.df);

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 6

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

lm.6 <- lm(log(SalePrice) ~ TotRmsAbvGrd,data=small.df);

plot(log(small.df$SalePrice) ~ small.df$TotRmsAbvGrd,xlab='Total Rooms Above
Ground',ylab='log(SalePrice)')
points(loess.6$x,loess.6$fitted,type='p',col='red',pch=19)
abline(coef=lm.6$coef,col='blue',lwd=2)
title('SLR - Total Rooms Above Ground')
```

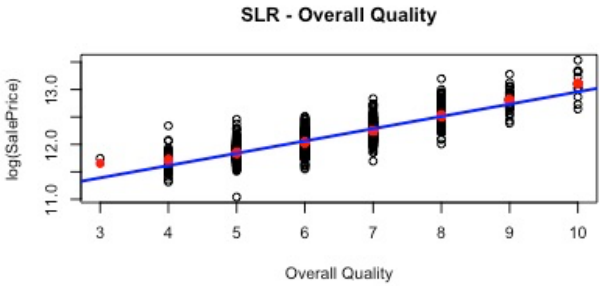


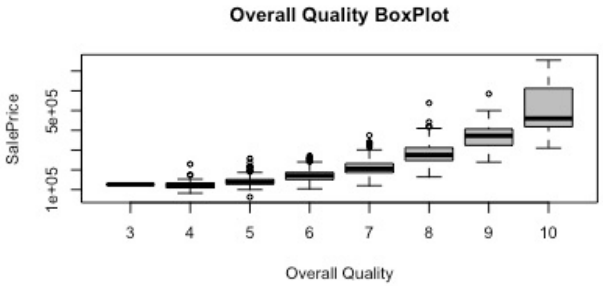

```
##############################################################################
########
# Discrete or Continuous?
##############################################################################
########

par(mfrow = c(3,2))
```

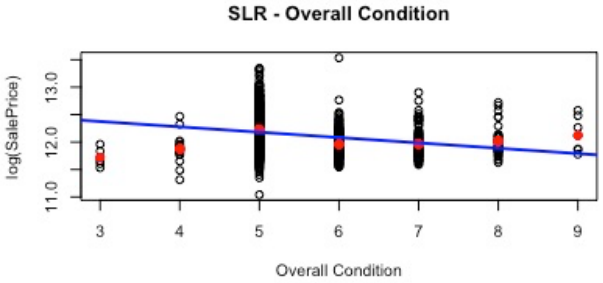
```r
#Total Rooms Above Grade
plot(log(small.df$SalePrice) ~ small.df$TotRmsAbvGrd,xlab='Total Rooms Above
Ground',ylab='log(SalePrice)')
points(loess.6$x,loess.6$fitted,type='p',col='red',pch=19)
abline(coef=lm.6$coef,col='blue',lwd=2)
title('SLR - Total Rooms Above Ground')
boxplot(small.df$SalePrice ~ small.df$TotRmsAbvGrd,col = "gray",xlab='Total R
ooms Above Ground',ylab='SalePrice', main = "Total Rooms Above Grade BoxPlot"
)

#Overall Quality
plot(log(small.df$SalePrice) ~ small.df$OverallQual,xlab='Overall Quality',yl
ab='log(SalePrice)')
points(loess.2$x,loess.2$fitted,type='p',col='red',pch=19)
abline(coef=lm.2$coef,col='blue',lwd=2)
title('SLR - Overall Quality')
boxplot(small.df$SalePrice ~ small.df$OverallQual,col = "gray", xlab='Overall
Quality',ylab='SalePrice', main = "Overall Quality BoxPlot")

#Overall Condition
plot(log(small.df$SalePrice) ~ small.df$OverallCond,xlab='Overall Condition',
ylab='log(SalePrice)')
points(loess.3$x,loess.3$fitted,type='p',col='red',pch=19)
abline(coef=lm.3$coef,col='blue',lwd=2)
title('SLR - Overall Condition')
boxplot(small.df$SalePrice ~ small.df$OverallCond, col = "gray",xlab='Overall
Condition',ylab='SalePrice', main = "Overall Condition BoxPlot")
```

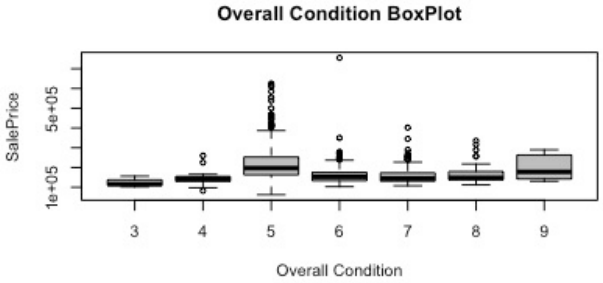













```
###Knitr
install.packages("knitr",repos = "http://cran.us.r-project.org")

## Installing package into '/Users/serrauzun/Library/R/3.5/library'
## (as 'lib' is unspecified)

##
##   There is a binary version available but the source version is later:
##       binary source needs_compilation
## knitr   1.28   1.29             FALSE

## installing the source package 'knitr'

library(knitr)

install.packages("rmarkdown",repos = "http://cran.us.r-project.org")

## Installing package into '/Users/serrauzun/Library/R/3.5/library'
## (as 'lib' is unspecified)

##
##   There is a binary version available but the source version is later:
```

```
##           binary source needs_compilation
## rmarkdown    2.1    2.3             FALSE

## installing the source package 'rmarkdown'
```

```
library(rmarkdown)
```