Assignment #1
Lauren Camero

Introduction:

In this report, we have completed a data survey, a data quality check, and an initial exploratory data analysis for the Ames, Iowa housing data set. We have begun the process of defining the sample population and modelling the data to provide estimates for typical homes values in Ames, Iowa.

Data:

We used a data set from the Ames Assessor's Office for individual resident properties sold in Ames, Iowa from 2006 to 2010. In the data survey and data quality check, we observed several housing qualities that could be excluded from our analysis to better target the population and created a waterfall drop condition subset of our data. Our eligible population required that we have one family homes in normal sales condition on a paved road. Similarly, houses in the refined data set had to be built after 1920 with a general living area of less than 800 square feet and have typical home functionality. Narrowing our population decreased our number of observations from 2,930 to 1,688.

Data Survey:

In our data survey, we used our data dictionary to observe the different categorical variables to eliminate some units and better target the type of home value we would like to predict. We chose a normal sales condition since other conditions might not provide the real value of the house. This eliminated 423 observations. We selected single family homes built after 1920 on paved roads since we are targeting the typical home. Targeting single family homes decreased our population by 505 records.

```
01: Not SFR                   505
02: Non-Normal Sale           423
03: Street Not Paved            6
04: Built Pre-1920            121
05: Not typical functionality 137
06: LT 800 SqFt                50
99: Eligible Sample          1688
```

Initial Exploratory Data Analysis:

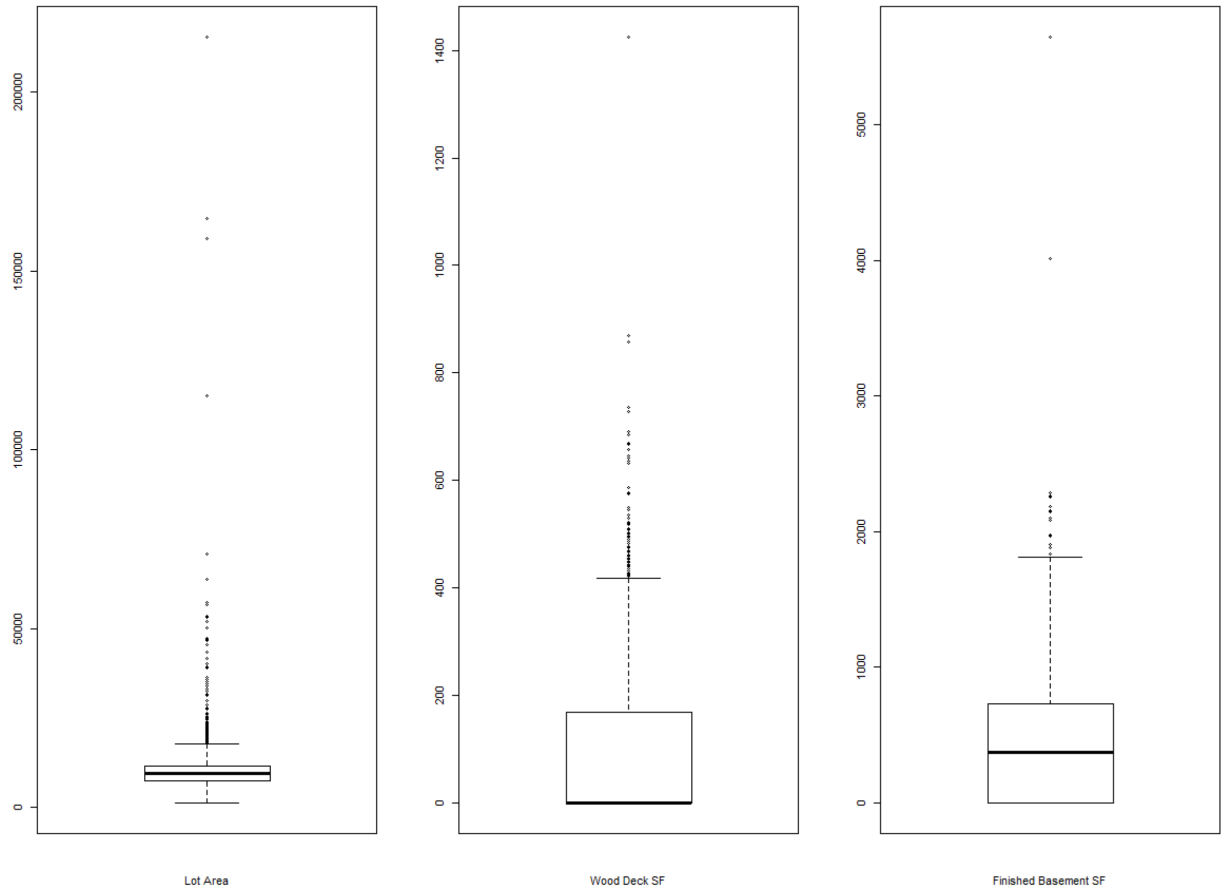
In our data quality check, we observed the following 20 variables:

Sales Price
Year Built
Lot Area
Overall Quality
Overall Condition
Remodel Year
Number of full baths
Number of half baths
Number of bedrooms above ground
Number of rooms above ground

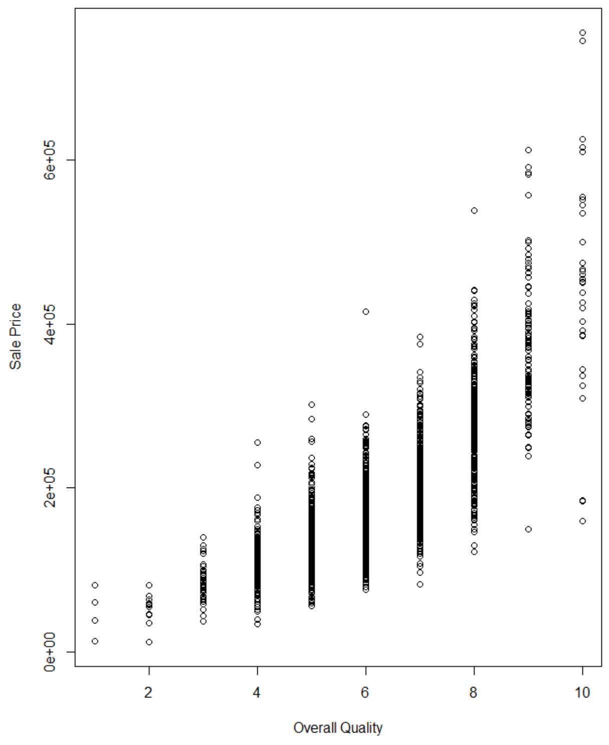
Number of fireplaces
Pool area
Month Sold
Year Sold
Lot frontage
Square feet of finished basement
Square feet of unfinished basement
First floor square feet
Second floor square feet
Wood deck square feet

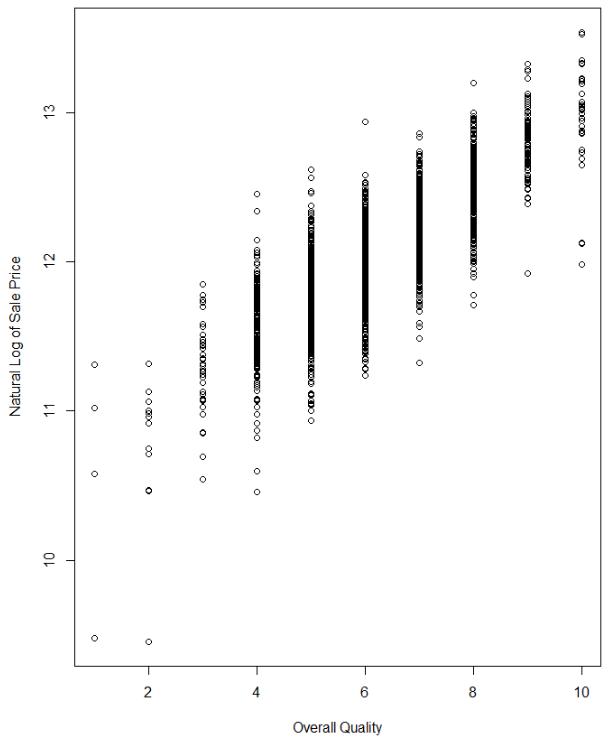
We considered eliminating records based on sale price since there is a large jump between the 75th percentile and the 100th percentile, but after observing records on homes sole for greater than $600,000, we observed that those homes had extra amenities and concluded that these were outliers.

We detected questionable data with the variables wood deck square feet, finished basement square feet, and total rooms above ground with extreme one-off outliers illustrated in the image below.



We have compared the outputs of changing our response variable and using the natural log of sales prices. We compared how three potential predictor variables changed with this transformation and found the results with the natural log of sales price to be more manageable. Below is an illustration of the comparison between sale price and the natural log of sale price in relation to overall quality.





Conclusions:

In conclusion, we have analyzed our data set, began structuring our population, and cleaned out unnecessary incorrect data. We now have a better understanding and are prepared to begin modeling to provide estimates for typical homes values in Ames, Iowa.

Code:

```
# Lauren Camero
# 1.14.2018
# read_ames.R

# Read in csv file for Ames housing data;
setwd('C:/Users/lcamero/Desktop/Predict 410')
file.name <- paste('ames_housing_data.csv',sep='');

# Read in the csv file into an R data frame;
ames.df <- read.csv(file.name,header=TRUE,stringsAsFactors=FALSE);

# Show the header of the data frame;
head(ames.df)

# List out the contents of the data frame;
# Use the structure function str();
str(ames.df)

# Show the distribution (and levels) of a discrete/factor type variable;
table(ames.df$LotShape)

# Note that table() will suppress the count of missing values;
# Here is how we force table() to show the missing values as a level;
table(ames.df$Fence,useNA=c('always'))
table(ames.df$SalePrice)


#################################################################################################
# Notes:
#################################################################################################

# Predictor variables first, not missing values first.Now that we have our data read into R, we need to define our
sample population.
#	Let's think about our problem, and let's consult our data dictionary to find fields that would identify
observations
#	that do not belong in our sample population.

#################################################################################################
# Add waterfall
#################################################################################################

# Single ifelse() statement
# ifelse(condition, value if condition is TRUE, value if the condition is FALSE)

# Nested ifelse() statement
# ifelse(condition1, value if condition1 is TRUE,
#	ifelse(condition2, value if condition2 is TRUE,
#	value if neither condition1 nor condition2 is TRUE
#	)
# )
# Create a waterfall of drop conditions;
# Work the data frame as a 'table' like you would in SAS or SQL;
ames.df$dropCondition <- ifelse(ames.df$BldgType!='1Fam','01: Not SFR',
                    ifelse(ames.df$SaleCondition!='Normal','02: Non-Normal Sale',
                        ifelse(ames.df$Street!='Pave','03: Street Not Paved',
                            ifelse(ames.df$YearBuilt <1920,'04: Built Pre-1920',
                                ifelse(ames.df$Functional != 'Typ','05: Not typical functionality',
                                    ifelse(ames.df$GrLivArea <800,'06: LT 800 SqFt',
                                        '99: Eligible Sample')
                                )))));


table(ames.df$dropCondition)

# Save the table
waterfall <- table(ames.df$dropCondition);

# Format the table as a column matrix for presentation;
as.matrix(waterfall,7,1)


# Eliminate all observations that are not part of the eligible sample population;
eligible.population <- subset(ames.df,dropCondition=='99: Eligible Sample');

# Check that all remaining observations are eligible;
#variable 1
table(ames.df$SalePrice)
summary(ames.df$SalePrice)
quantile(ames.df$SalePrice)
mean(ames.df$SalePrice)
sd(ames.df$SalePrice)

ames.df$dropCondition1 <- ifelse(ames.df$SalePrice > 600000,'Recheck',
                        '99: Eligible Sample')

noneligible.population <- subset(ames.df,dropCondition1 !='99: Eligible Sample');

#variable 2
table(ames.df$YearBuilt)
summary(ames.df$YearBuilt)
quantile(ames.df$YearBuilt)
mean(ames.df$YearBuilt)
sd(ames.df$YearBuilt)
plot(ames.df$YearBuilt,ames.df$SalePrice)
boxplot(ames.df$YearBuilt)

#variable 3
table(ames.df$LotArea)
summary(ames.df$LotArea)
quantile(ames.df$LotArea)
mean(ames.df$LotArea)
sd(ames.df$LotArea)
```

```
plot(ames.df$LotArea,ames.df$SalePrice)
boxplot(ames.df$LotArea)

#variable 4
table(ames.df$OverallQual)
summary(ames.df$OverallQual)
quantile(ames.df$OverallQual)
mean(ames.df$OverallQual)
sd(ames.df$OverallQual)
plot(ames.df$OverallQual,ames.df$SalePrice)
boxplot(ames.df$OverallQual)

#variable 5
table(ames.df$OverallCond)
summary(ames.df$OverallCond)
quantile(ames.df$OverallCond)
mean(ames.df$OverallCond)
sd(ames.df$OverallCond)
plot(ames.df$OverallCond,ames.df$SalePrice)
boxplot(ames.df$OverallCond)

#variable 6
table(ames.df$YearRemodel)
summary(ames.df$YearRemodel)
quantile(ames.df$YearRemodel)
mean(ames.df$YearRemodel)
sd(ames.df$YearRemodel)
plot(ames.df$YearRemodel,ames.df$SalePrice)
boxplot(ames.df$YearRemodel)

#variable 7
table(ames.df$FullBath)
summary(ames.df$FullBath)
quantile(ames.df$FullBath)
mean(ames.df$FullBath)
sd(ames.df$FullBath)
plot(ames.df$FullBath,ames.df$SalePrice)
boxplot(ames.df$FullBath)

#variable 8
table(ames.df$HalfBath)
summary(ames.df$HalfBath)
quantile(ames.df$HalfBath)
mean(ames.df$HalfBath)
sd(ames.df$HalfBath)
plot(ames.df$HalfBath,ames.df$SalePrice)
boxplot(ames.df$HalfBath)

#variable 9
table(ames.df$BedroomAbvGr)
summary(ames.df$BedroomAbvGr)
quantile(ames.df$BedroomAbvGr)
mean(ames.df$BedroomAbvGr)
```

```
sd(ames.df$BedroomAbvGr)
plot(ames.df$BedroomAbvGr,ames.df$SalePrice)
boxplot(ames.df$BedroomAbvGr)

#variable 10
table(ames.df$TotRmsAbvGrd)
summary(ames.df$TotRmsAbvGrd)
quantile(ames.df$TotRmsAbvGrd)
mean(ames.df$TotRmsAbvGrd)
sd(ames.df$TotRmsAbvGrd)
plot(ames.df$TotRmsAbvGrd,ames.df$SalePrice)
boxplot(ames.df$TotRmsAbvGrd)

#variable 11
table(ames.df$Fireplaces)
summary(ames.df$Fireplaces)
quantile(ames.df$Fireplaces)
mean(ames.df$Fireplaces)
sd(ames.df$Fireplaces)
plot(ames.df$Fireplaces,ames.df$SalePrice)
boxplot(ames.df$Fireplaces)

#variable 12
table(ames.df$PoolArea)
summary(ames.df$PoolArea)
quantile(ames.df$PoolArea)
mean(ames.df$PoolArea)
sd(ames.df$PoolArea)
plot(ames.df$PoolArea,ames.df$SalePrice)
boxplot(ames.df$PoolArea)

#variable 13
table(ames.df$MoSold)
summary(ames.df$MoSold)
quantile(ames.df$MoSold)
mean(ames.df$MoSold)
sd(ames.df$MoSold)
plot(ames.df$MoSold,ames.df$SalePrice)
boxplot(ames.df$MoSold)

#variable 14
table(ames.df$YrSold)
summary(ames.df$YrSold)
quantile(ames.df$YrSold)
mean(ames.df$YrSold)
sd(ames.df$YrSold)
plot(ames.df$YrSold,ames.df$SalePrice)
boxplot(ames.df$YrSold)

#variable 15
table(ames.df$LotFrontage)
summary(ames.df$LotFrontage)
mean(ames.df$LotFrontage)
```

```
sd(ames.df$LotFrontage)
plot(ames.df$LotFrontage,ames.df$SalePrice)
boxplot(ames.df$LotFrontage)

#variable 16
table(ames.df$BsmtFinSF1)
summary(ames.df$BsmtFinSF1)
mean(ames.df$BsmtFinSF1)
sd(ames.df$BsmtFinSF1v)
plot(ames.df$BsmtFinSF1,ames.df$SalePrice)
boxplot(ames.df$BsmtFinSF1)

#variable 17
table(ames.df$BsmtUnfSF)
summary(ames.df$BsmtUnfSF)
mean(ames.df$BsmtUnfSF)
sd(ames.df$BsmtUnfSF)
plot(ames.df$BsmtUnfSF,ames.df$SalePrice)
boxplot(ames.df$BsmtUnfSF)

#variable 18
table(ames.df$FirstFlrSF)
summary(ames.df$FirstFlrSF)
mean(ames.df$FirstFlrSF)
sd(ames.df$FirstFlrSF)
plot(ames.df$FirstFlrSF,ames.df$SalePrice)
boxplot(ames.df$FirstFlrSF)

#variable 19
table(ames.df$SecondFlrSF)
summary(ames.df$SecondFlrSF)
mean(ames.df$SecondFlrSF)
sd(ames.df$SecondFlrSF)
plot(ames.df$SecondFlrSF,ames.df$SalePrice)
boxplot(ames.df$SecondFlrSF)

#variable 20
table(ames.df$WoodDeckSF)
summary(ames.df$WoodDeckSF)
mean(ames.df$WoodDeckSF)
sd(ames.df$WoodDeckSF)
plot(ames.df$WoodDeckSF,ames.df$SalePrice)
boxplot(ames.df$WoodDeckSF)

## boxplot of high outliers
par(mfrow=c(1,3))
boxplot(ames.df$LotArea, xlab = 'Lot Area')
boxplot(ames.df$WoodDeckSF, xlab = 'Wood Deck SF')
boxplot(ames.df$BsmtFinSF1, xlab = 'Finished Basement SF')
##Check if log(saleprice) is better

#variable 2
par(mfrow=c(1,2))
```

```
plot(ames.df$YearBuilt,ames.df$SalePrice)
plot(ames.df$YearBuilt,log(ames.df$SalePrice))

#variable 3
par(mfrow=c(1,2))
plot(ames.df$LotArea,ames.df$SalePrice)
plot(ames.df$LotArea,log(ames.df$SalePrice))

#variable 4
par(mfrow=c(1,2))
plot(ames.df$OverallQual,ames.df$SalePrice)
plot(ames.df$OverallQual,log(ames.df$SalePrice))
```

Assignment #2
Lauren Camero

Introduction:

In this report, our objective is to provide approximations of values for the typical home in Ames, Iowa. We used a data set from the Ames Assessor's Office for individual resident properties sold in Ames, Iowa from 2006 to 2010. We are beginning to build regression models to estimate the home sale price.

First, we defined the sample population by using drop conditions to limit our sample population to the observations that best serve our purpose. We then created a waterfall to exclude these drop conditions and defined our sample population.

Next, we completed simple linear regression models after observing our variables in an exploratory data analysis (EDA) and selected two variables that we believe will be good predictors of sale price using descriptive statistics. We fit two simple linear regression models and assessed their goodness of fit.

After analyzing and selecting our two predictor variables, we combined our two simple linear regression models into a multiple regression model and assessed the goodness of fit of this output.

Finally, we transformed the response variable from sale price to the natural logarithm of sale price and compared the new model outputs to the untransformed version of our response variable to understand which form of sales price should be utilized in our model. The EDA and models were built in R programming language.

Data:

The data set is comprised of 82 variables measured from 2930 individual residential properties sold in Ames, IA from 2006 to 2010 collected by Ames Assessor's Office for individual resident properties. Description of the variables can be found in the referenced data documentation.

Sample Definition:

In the data survey and data quality check, we observed several housing qualities that could be excluded from our analysis to better target the population and created a waterfall drop condition subset of our data. Our eligible population required that we have one-family homes in normal sales condition on a paved road. Similarly, houses in the refined data set had to be built after 1950 with a general living area greater than 800 square feet and have a basement. Narrowing our population decreased our number of observations from 2,930 to 1,470 as shown in Figure 1.

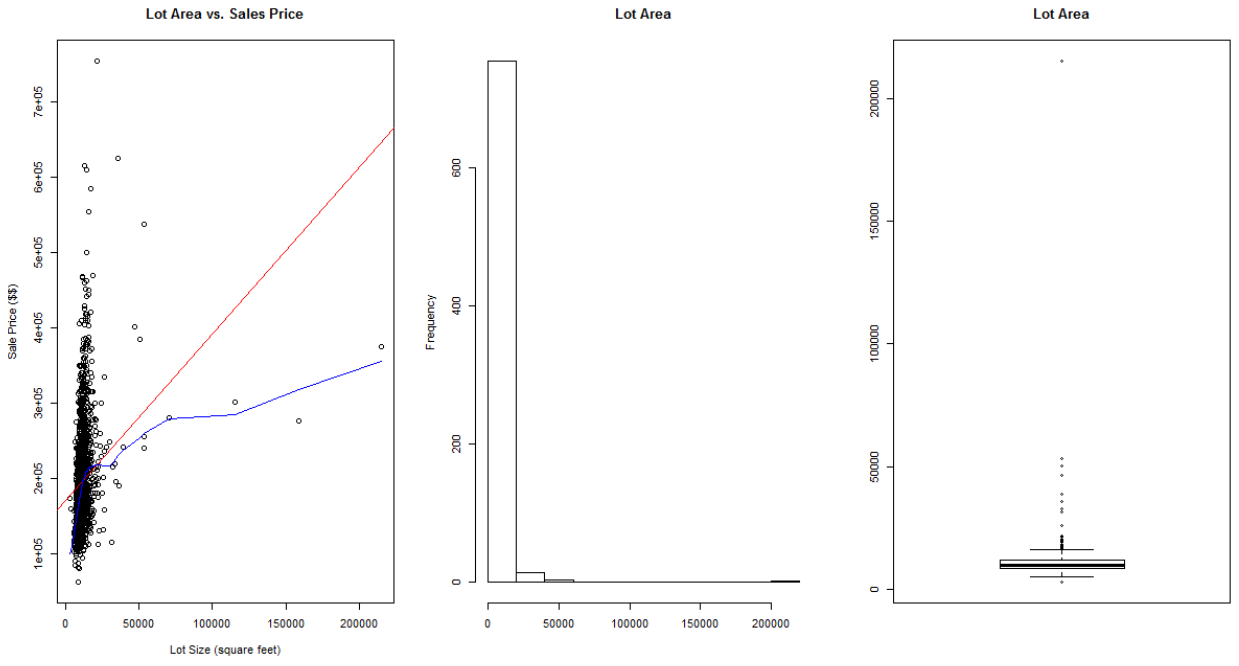
Figure 1: Waterfall Conditions

Variable	Drop Condition	Number of Properties Dropped
Building Type	Not equal to single family detached	505
Sale Condition	Not equal to normal	423
Street	Not Paved	6
Year Built	Built before 1950	489
Square Feet of Total Basement	Less than 1 square feet	28
Square Feet of Above Ground Living Area	Less than 800 square feet	9

Exploratory Data Analysis:

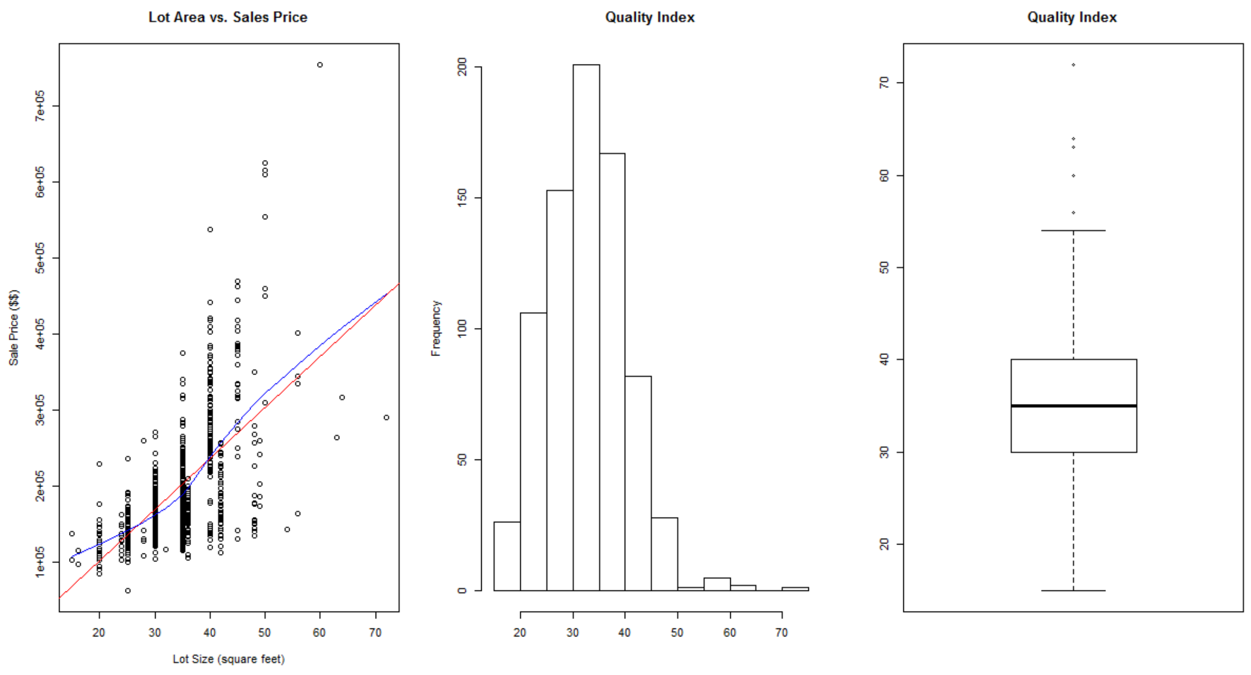
We have explored ten variables to assess which two would be the best possible predictors for the response variable. Initially, we thought Lot Area would be a good predictor since it is a continuous variable and would vary and affect the value of the home. Figure 2 illustrates that extreme outliers and positive skewness complicate this assumption. Clearly, lot area does not have a strong correlation with sale price.

Figure 2: EDA of Lot Area with blue line as lowess smoother and red line depicting ordinary least squares



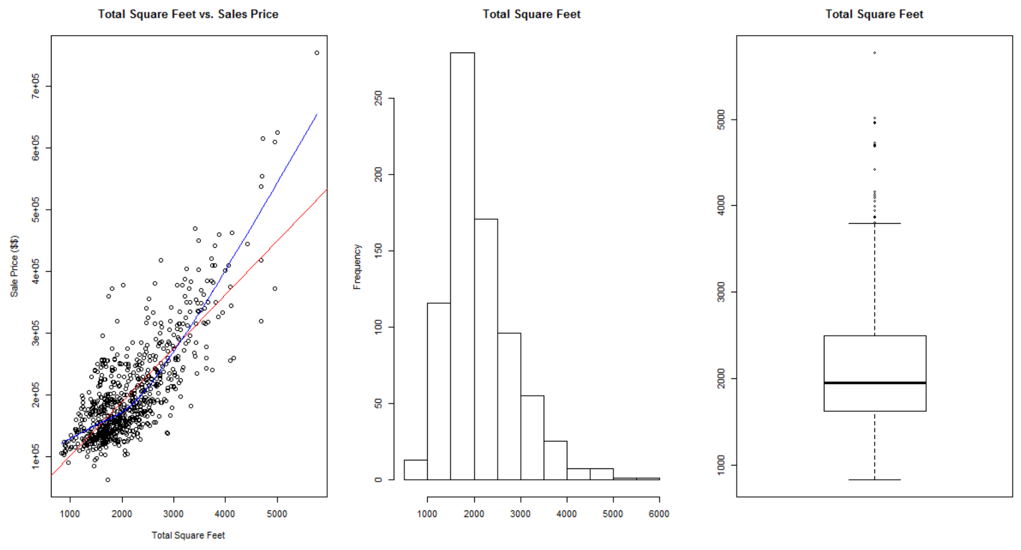
We looked at the quality index we had created next. The quality index was a metric that we had calculated by taking the two ordinal variables of overall condition and overall quality and multiplying them. As you can see in Figure 3, this variable illustrates almost a normal distribution with a slight positive skew and a higher correlation to sales price.

Figure 3: EDA of Quality Index with blue line as lowess smoother and red line depicting ordinary least squares



Finally, we observed the total square feet variable that we had created by adding the basement finished and unfinished square feet with the above ground living area and assesed our new variable and its relationship with sales price to find that this also visually appears to be a good predictor variable with homoscedastic variance. The total square feet is positively skewed with the median square feet being around 2,000.

Figure 4: EDA of Total Square Feet with blue line as lowess smoother and red line depicting ordinary least squares



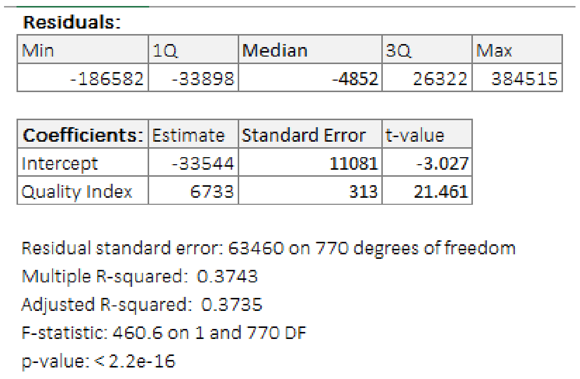
Simple Linear Regression Models:

Using our chosen predictor variables of total square feet and quality index we fit two simple linear regression models and assessed their goodness of fit. We assess the goodness of fit of our models using the Q-Q plot to check if the residuals follow a normal distribution and the predictor versus residuals scatterplot to see if there is a relationship.

Model #1: Quality Index:

The scatterplot for the Predictor and the residuals illustrates a funnel. This means that the span of the residuals is increasing as the predictor value increases and suggests that we may want to transform a variable. The Q-Q plot in figure 6 illustrates normality between -1 and 1, but the residuals do not follow a normal distribution above and below this range. The R-squared for this model is 0.3743, much higher than the Lot Area variable which was another criteria used to pick these two predictor variables.

Figure 5: Descriptive Statistics of simple linear regression on Quality Index



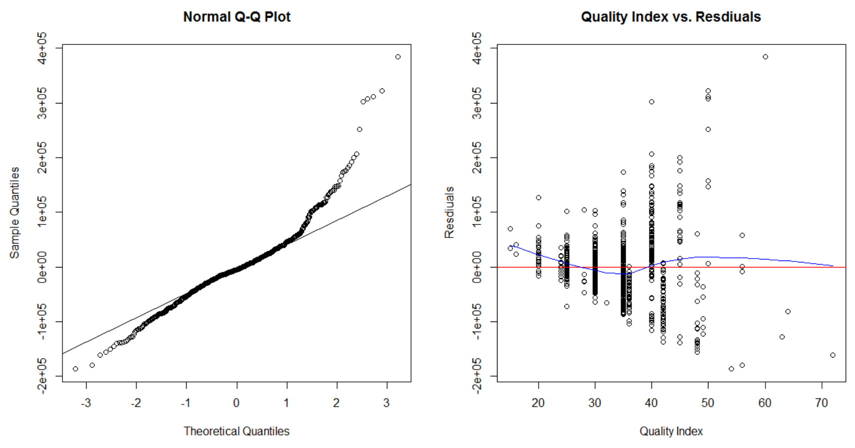
Residuals:

Min	1Q	Median	3Q	Max
-186582	-33898	-4852	26322	384515

Coefficients:	Estimate	Standard Error	t-value
Intercept	-33544	11081	-3.027
Quality Index	6733	313	21.461

Residual standard error: 63460 on 770 degrees of freedom
Multiple R-squared: 0.3743
Adjusted R-squared: 0.3735
F-statistic: 460.6 on 1 and 770 DF
p-value: < 2.2e-16

Figure 6: Q-Q plot and Residual vs. Predictor plot for Quality Index simple linear regression with blue line as lowess smoother and red line depicting ordinary least squares



Model #2: Total Square Feet

The scatterplot for the Predictor and the residuals is visually correlated with the lowess line and the OLS line very close in the range of 1000 to 4000. The Q-Q plot in figure 6 illustrates normality between -2 and

2, but the residuals do not follow a normal distribution above and below this range. The R-squared for this model was 0.6266. We may need to transform a variable.

Figure 7: Descriptive Statistics of simple linear regression on Total Square Feet

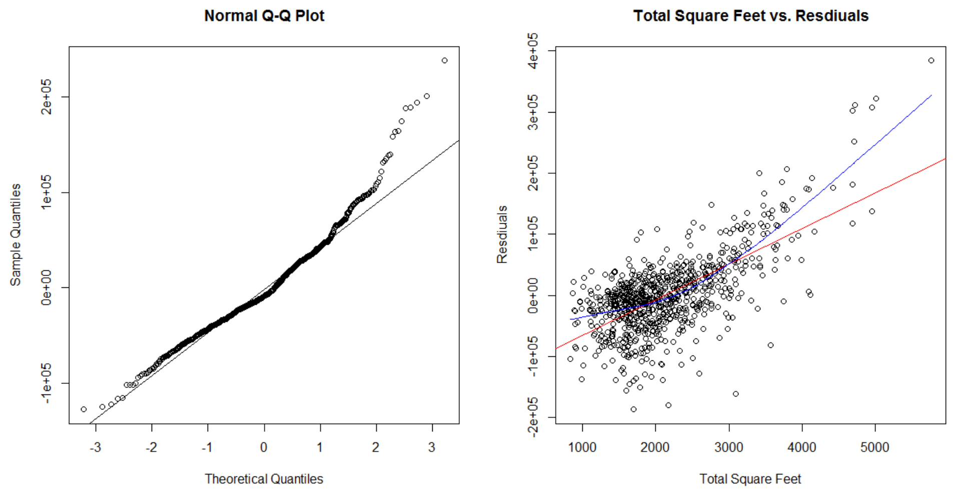
Residuals:

Min	1Q	Median	3Q	Max
-127918	-32327	-8465	28628	238843

Coefficients:	Estimate	Standard Error	t-value
Intercept	14348	5435	2.64
Total Square Feet	86	2	35.94

Residual standard error: 49030 on 770 degrees of freedom
Multiple R-squared: 0.6266
Adjusted R-squared: 0.6261
F-statistic: 1292 on 1 and 770 DF
p-value: < 2.2e-16

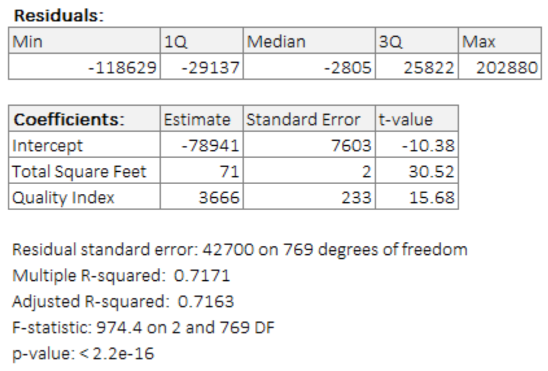
Figure 8: Q-Q plot and Residual vs. Predictor plot for Quality Index simple linear regression with blue line as lowess smoother and red line depicting ordinary least squares



Model #3: Multiple Linear Regression Model

Looking at the Q-Q plot in Figure 10, we see that the multiple regression works better than either of the simple linear regressions because the residuals follow a normal distribution between -3 and 2. Similarly, the adjusted R-squared in Figure 9 is 0.7163 which is higher than either of the R-squared outputs in Model 1 and Model 2. Although the model has improved by adding a variable in this case, we will not always get the same outcome and must test the adjusted R-squared when creating a multiple regression model.

Figure 9: Descriptive Statistics of simple linear regression on Total Square Feet and Quality Index

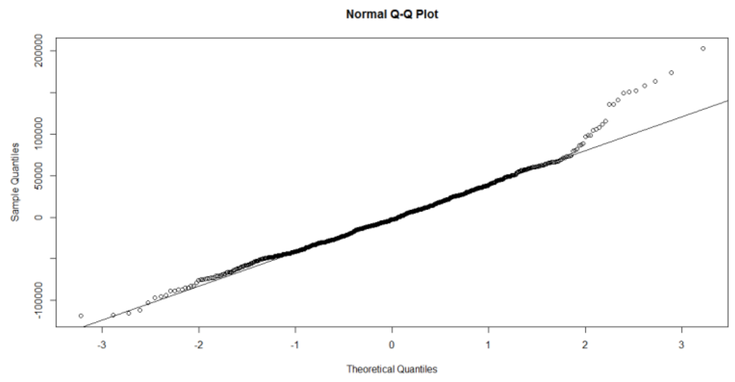

Residuals:

Min	1Q	Median	3Q	Max
-118629	-29137	-2805	25822	202880

Coefficients:	Estimate	Standard Error	t-value
Intercept	-78941	7603	-10.38
Total Square Feet	71	2	30.52
Quality Index	3666	233	15.68

Residual standard error: 42700 on 769 degrees of freedom
Multiple R-squared: 0.7171
Adjusted R-squared: 0.7163
F-statistic: 974.4 on 2 and 769 DF
p-value: < 2.2e-16

Figure 10: Q-Q plot for Total Square Feet and Quality Index



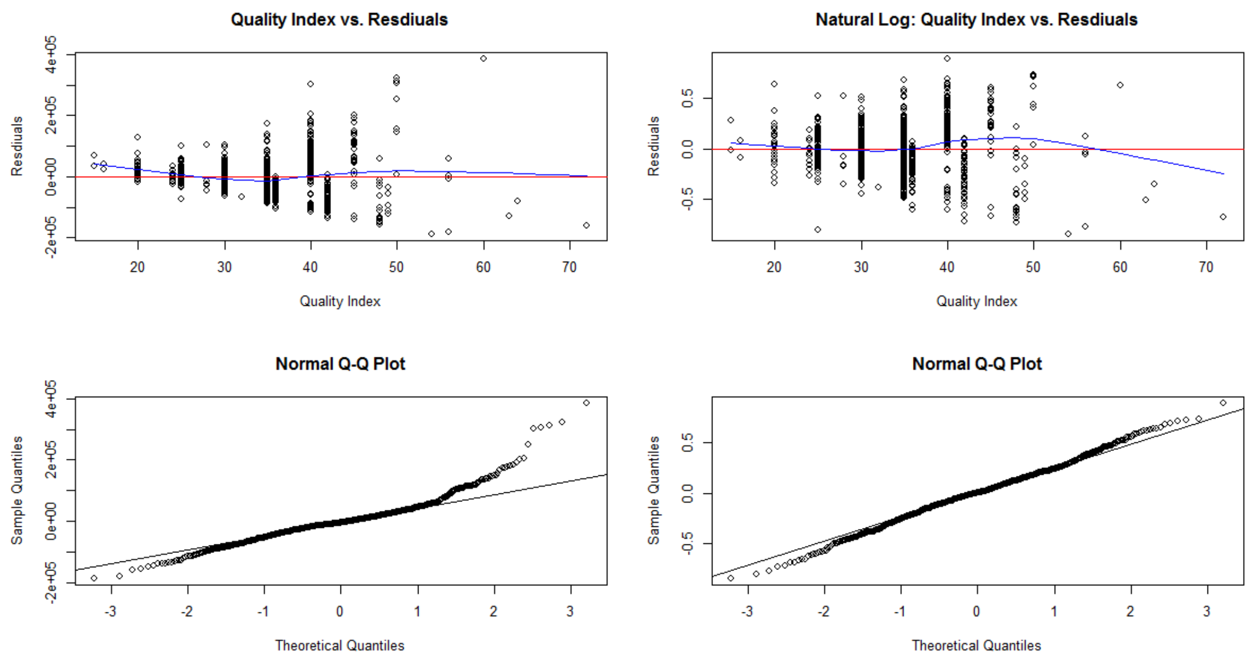
Log SalePrice Response Models:

We will now transform our response variable and refit our models to test if the natural log of sales price produces a better fit for our predictors than the untransformed.

Model #4: Quality Index and the natural log of Sales Price:

As shown in Figure 11, transforming the sales price using the natural log improves the fit of the model for the Quality Index. The Q-Q plot shows that the residuals are very close to forming a normal distribution. In addition, the R-Squared is 0.4033 which is higher than the untransformed model. Quality index has the most improved fit after the transformation in comparison to the other two models.

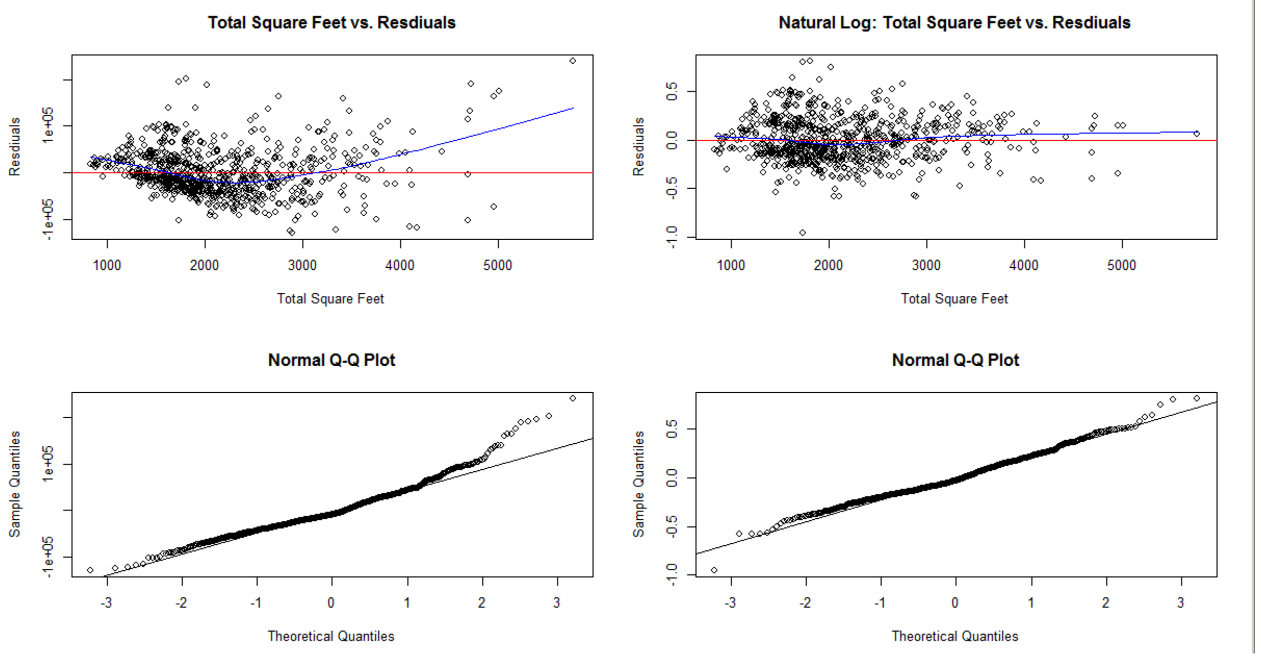
Figure 11: Goodness of Fit test comparing Sales Price and the Natural Log of Sales Price



Model #5: Total Square Feet and the natural log of Sales Price:

Figure 12 illustrates an improvement in the fit of the model after transforming our response variable as well. We see an improved linearity when comparing the OLS line and the lowess line in the predictor versus residuals scatterplot. We also can see an in increase in normality of the residuals shown in the Q-Q plot. Our R-squared statistic has decreased from 0.6266 to 0.5958.

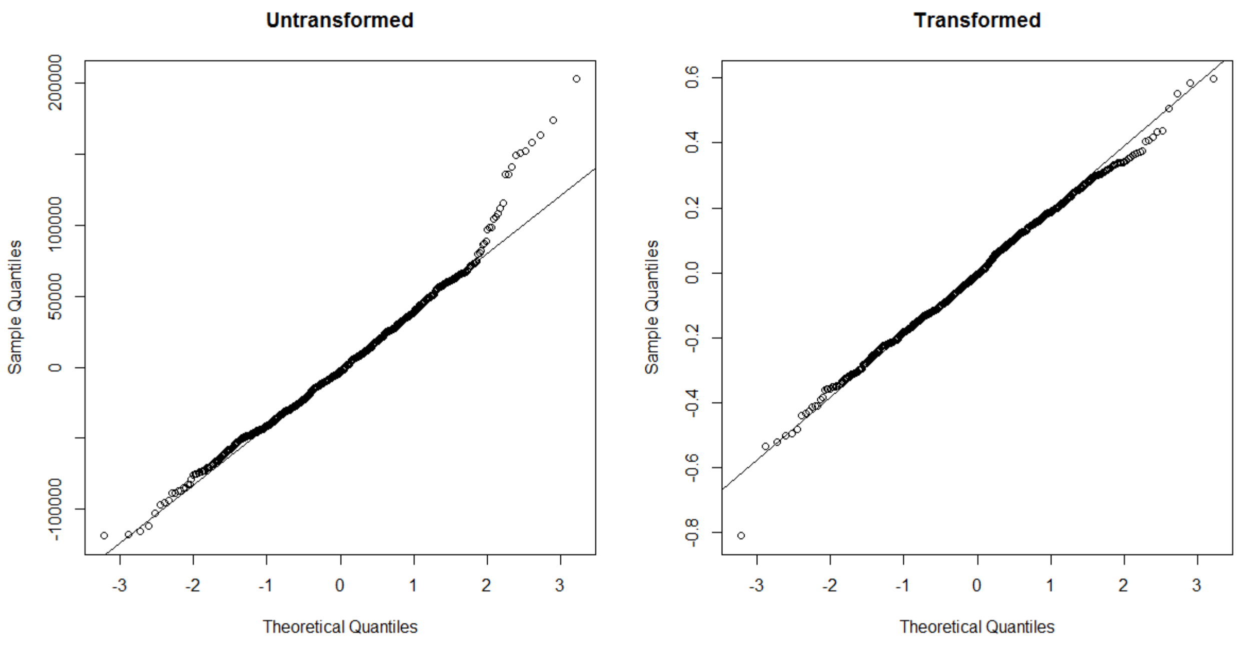
Figure 12: Goodness of Fit test comparing Sales Price and the Natural Log of Sales Price



Model #6: Multiple Regression and the natural log of Sales Price:

Finally, we observe how our multiple regression performs using a natural log of sales price. We see that our adjusted R-squared has decreased from 0.7163 to 0.708. Our Q-Q plot shows us that our normality for our residuals has improved since our residuals are linear.

Figure 13: Q-Q Plot Sales Price and the Natural Log of Sales Price



Conclusions:

We have discovered that the quality index and the total square feet are good predictor variables in the objective to estimate the value of homes in Ames, Iowa. We have observed that a multiple regression using both variables will provide us a better estimation since the model is better fit. Additionally, we have discovered that our models improve if we transform the response variable, sales price, by taking its natural log. Since we have seen an improvement in our model by adding predictor variables, we may want to include another variable and test the fitness of that model.

References:

Dean De Cock. (2011). Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project, Journal of Statistics Education, 19:3. Retrieved from http://amstat.tandfonline.com/doi/abs/10.1080/10691898.2011.11889627#.WciC-WinFTE
Dean De Cock. Ames Housing Data Documentation. Retrieved from https://ww2.amstat.org/publications/jse/v19n3/decock/DataDocumentation.txt

Code:

```
# Lauren Camero
# 1.20.2018
# assignment#2.R

# Read in csv file for Ames housing data;

# Note that back slash is an escape character in R so we use \\ when we want \;
setwd('C:/Users/lcamero/Desktop/Predict 410')
file.name <- paste('ames_housing_data.csv',sep='');

# Read in the csv file into an R data frame;
ames.df <- read.csv(file.name,header=TRUE,stringsAsFactors=FALSE);


# Create a waterfall of drop conditions;
ames.df$dropCondition <- ifelse(ames.df$BldgType!='1Fam','01: Not SFR',
        ifelse(ames.df$SaleCondition!='Normal','02: Non-Normal Sale',
        ifelse(ames.df$Street!='Pave','03: Street Not Paved',
        ifelse(ames.df$YearBuilt <1950,'04: Built Pre-1950',
        ifelse(ames.df$TotalBsmtSF <1,'05: No Basement',
        ifelse(ames.df$GrLivArea <800,'06: LT 800 SqFt',
        '99: Eligible Sample')
        )))));

# Save the table
waterfall <- table(ames.df$dropCondition);

# Format the table as a column matrix for presentation;
as.matrix(waterfall,7,1)


# Eliminate all observations that are not part of the eligible sample population;
eligible.population <- subset(ames.df,dropCondition=='99: Eligible Sample');

# Check that all remaining observations are eligible;
table(eligible.population$dropCondition);


##########################################################################################
# Create a list of interesting predictor variables
##########################################################################################

str(ames.df)

# Predictor variables that I like:
LotFrontage
LotArea
LotConfig
Neighborhood
HouseStyle
OverallQual
OverallCond
YearBuilt
YearRemodel
Exterior1
BsmtFinSF1
BsmtFinSF2
CentralAir
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
```

```
TotRmsAbvGrd
Fireplaces
GarageCars
GarageArea
WoodDeckSF
OpenPorchSF
EnclosedPorch
ThreeSsnPorch
ScreenPorch
PoolArea
MoSold
YrSold
SaleCondition
SalePrice

# Make a vector of the names;

keep.vars <- c('SID','PID','LotFrontage','LotArea','LotConfig','Neighborhood',
'HouseStyle','OverallQual','OverallCond','YearBuilt','YearRemodel','Exterior1',
'BsmtFinSF1','BsmtFinSF2','CentralAir','GrLivArea','BsmtFullBath','BsmtHalfBath',
'FullBath','HalfBath','BedroomAbvGr','TotRmsAbvGrd','Fireplaces','GarageCars',
'GarageArea','WoodDeckSF','OpenPorchSF','EnclosedPorch','ThreeSsnPorch',
'ScreenPorch','PoolArea','MoSold','YrSold','SaleCondition','SalePrice'
);


# Note that the R data frame is a (rectangular) list object which means that it can be
# accessed in two ways - as a matrix or as a list;
# Note that the keep.vars are the COLUMNS that we want to keep, not the rows;

skinny.df <- eligible.population[,keep.vars];

# Use the structure command to view the contents of the data frame;
str(skinny.df)



##########################################################################################
# Delete observations with missing values
##########################################################################################
sample.df <- na.omit(skinny.df);

# Check the change in dimension;
dim(skinny.df)
dim(sample.df)

dim(skinny.df)-dim(sample.df)



##########################################################################################
# Define some discrete variables and indicator variables
##########################################################################################

# Define total square footage
sample.df$TotalSqftCalc <- sample.df$BsmtFinSF1+sample.df$BsmtFinSF2+sample.df$GrLivArea;

# Define total bathrooms
sample.df$TotalBathCalc <- sample.df$BsmtFullBath + 0.5*sample.df$BsmtHalfBath ++
                + sample.df$FullBath + 0.5*sample.df$HalfBath;


# Corner lot indicator
# ifelse(condition,valueTrue,valueFalse)
sample.df$CornerLotInd <- ifelse(sample.df$LotConfig=='Corner',1,0);
```

```
# Check how the indicator is assigned
table(sample.df$CornerLotInd,sample.df$LotConfig)

# Define two indicators for fire places
table(sample.df$Fireplaces)

# Intercept Adjustment for a single fireplace
sample.df$FireplaceInd1 <- ifelse((sample.df$Fireplaces>0)&(sample.df$Fireplaces<2),1,0);
table(sample.df$FireplaceInd1,sample.df$Fireplaces)

# Intercept Adjustment for 2 or more fireplaces
sample.df$FireplaceInd2 <- ifelse((sample.df$Fireplaces>1),1,0);
table(sample.df$FireplaceInd2,sample.df$Fireplaces)

# Additive Intercept Adjustment for a single fireplace
sample.df$FireplaceAdder1 <- ifelse((sample.df$Fireplaces>0),1,0);

# Additive Intercept Adjustment for 2 or more fireplaces
sample.df$FireplaceAdder2 <- ifelse((sample.df$Fireplaces>1),1,0);

table(sample.df$FireplaceAdder1,sample.df$Fireplaces)
table(sample.df$FireplaceAdder2,sample.df$Fireplaces)



# Central Air Indicator
sample.df$CentralAirInd <- ifelse(sample.df$CentralAir=='Y',1,0);
table(sample.df$CentralAirInd)
# Looks like this is not useful since almost all homes have central air


# Exterior Siding Type
sample.df$BrickInd <- ifelse(sample.df$Exterior1=='BrkFace',1,0);
sample.df$VinylSidingInd <- ifelse(sample.df$Exterior1=='VinylSd',1,0);

# Pool Indicator
sample.df$PoolInd <- ifelse(sample.df$PoolArea>0,1,0);

# Wood Deck Indicator
sample.df$WoodDeckInd <- ifelse(sample.df$WoodDeckSF>0,1,0);

# Porch Indicator - Open Porch OR Screen Porch
sample.df$PorchInd <- ifelse((sample.df$OpenPorchSF>0)||(sample.df$ScreenPorch>0),1,0);

# Quality Index
sample.df$QualityIndex <- sample.df$OverallQual*sample.df$OverallCond;

table(sample.df$QualityIndex)


# Year Sold Indicators
sample.df$I2006 <- ifelse(sample.df$YrSold==2006,1,0);
sample.df$I2007 <- ifelse(sample.df$YrSold==2007,1,0);
sample.df$I2008 <- ifelse(sample.df$YrSold==2008,1,0);
sample.df$I2009 <- ifelse(sample.df$YrSold==2009,1,0);
sample.df$I2010 <- ifelse(sample.df$YrSold==2010,1,0);

table(sample.df$YrSold)
table(sample.df$I2006)
table(sample.df$I2007)
table(sample.df$I2008)
table(sample.df$I2009)
table(sample.df$I2010)
```

```
# List out sample.df
str(sample.df)



##########################################################################################
# Add a train/test flag to split the sample
##########################################################################################
sample.df$u <- runif(n=dim(sample.df)[1],min=0,max=1);
sample.df$train <- ifelse(sample.df$u<0.70,1,0);

# Check the counts on the train/test split
table(sample.df$train)

# Check the train/test split as a percentage of whole
table(sample.df$train)/dim(sample.df)[1]



##########################################################################################
# Save data frame as an .RData data object
##########################################################################################

# Save the R data frame as an .RData object
saveRDS(sample.df,file='ames_housing_data.csv');

# Read (or reload) the .RData object as an R data frame
a <- readRDS('ames_housing_data.csv');

# Check it
str(a)

########################################################################################
#Begin the EDA with Variable 1: total square feet
########################################################################################


# Check it
str(sample.df)

# Technically we should perform our EDA on the training data set
train.df <- subset(sample.df,train==1);



##################################################################################
# BASE R Scatterplot
##################################################################################

# Let's control the R plot to make it pretty
plot(train.df$TotalSqftCalc,train.df$SalePrice/1000,xlab='Total SQFT',ylab='Sale Price (000)',
     main='SQFT and Sale Price')

# In low dimensions scatterplots can be useful for visualizing the relationship
# between the response and a predictor variable


##################################################################################
# BASE R Box Plot
##################################################################################

boxplot(train.df$SalePrice/1000 ~ train.df$Neighborhood, las=2)
title('Sale Price By Neighborhood')

# Boxplots are the tool of choice for visualizing factor variables
```

```
####################################################################################
# Treating a discrete variable as continuous
####################################################################################

plot(train.df$TotalSqftCalc,train.df$SalePrice)

##########################################################################################
#Begin the EDA with Variable 2: Lot Area
##########################################################################################

par(mfrow = c(1, 3))
plot(eligible.population$LotArea, eligible.population$SalePrice,
     ylab = "Sale Price ($$)",
     xlab = "Lot Size (square feet)",
     main = 'Lot Area vs. Sales Price')
abline(lm(train.df$SalePrice ~ train.df$LotArea), col = "red")
lines(lowess(train.df$LotArea, train.df$SalePrice),
      col = "blue")
hist(train.df$LotArea, main='Lot Area', xlab = '')
boxplot(train.df$LotArea, main='Lot Area')

par(mfrow = c(1, 3))
plot(train.df$QualityIndex, train.df$SalePrice,
     ylab = "Sale Price ($$)",
     xlab = "Lot Size (square feet)",
     main = 'Lot Area vs. Sales Price')
abline(lm(train.df$SalePrice ~ train.df$QualityIndex), col = "red")
lines(lowess(train.df$QualityIndex, train.df$SalePrice),
      col = "blue")
hist(train.df$QualityIndex, main='Quality Index', xlab = '')
boxplot(train.df$QualityIndex, main='Quality Index')

par(mfrow = c(1, 3))
plot(train.df$TotalSqftCalc, train.df$SalePrice,
     ylab = "Sale Price ($$)",
     xlab = "Total Square Feet",
     main = 'Total Square Feet vs. Sales Price')
abline(lm(train.df$SalePrice ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc, train.df$SalePrice),
      col = "blue")
hist(train.df$TotalSqftCalc, main='Total Square Feet', xlab = '')
boxplot(train.df$TotalSqftCalc, main='Total Square Feet')


#title(main = "Residential Properties

# Sometime we will choose to treat a discrete variable as a continuous variable.
# In these cases the discrete variable needs to have 'enough' values, and a 'nice'
# relationship with the response variable.

#####################################################################################
#fit a regular model on Lot Area - model 2
#####################################################################################

# Fit a linear regression model with R
model.2 <- lm(SalePrice ~ LotArea, data=train.df)

# Display model summary
summary(model.2)

# List out components of lm object
names(model.2)

# Access a component of lm object
model.2$coefficients
```

```
# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.2 <- mean(model.2$residuals^2)
mae.2 <- mean(abs(model.2$residuals))

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.2)

# Hint:  We need to check the assumptions of normality and homoscedasticity;
# (1) QQ Plot
# (2) Scatterplot of residuals versus predictor


# Use the Base R functon qqplot() to assess the normality of the residuals
qqnorm(model.2$residuals)
qqline(model.2$residuals)

# Make a scatterplot
plot(train.df$LotArea,model.2$residuals)
title('Residual vs Predictor')

# this ended up with a very low r squared, so I am looking at another variable
####################################################################################
#fit a regular model on Quality Index model - 3
####################################################################################

# Fit a linear regression model with R
model.3 <- lm(SalePrice ~ QualityIndex, data=train.df)

# Display model summary
summary(model.3)

# List out components of lm object
names(model.3)

# Access a component of lm object
model.3$coefficients

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.3 <- mean(model.3$residuals^2)
mae.3 <- mean(abs(model.3$residuals))

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.2)

# Hint:  We need to check the assumptions of normality and homoscedasticity;
# (1) QQ Plot
# (2) Scatterplot of residuals versus predictor

par(mfrow = c(1, 2), oma = c(0, 0, 2, 0))
# Use the Base R functon qqplot() to assess the normality of the residuals
qqnorm(model.3$residuals)
qqline(model.3$residuals)

# Make a scatterplot
#plot(train.df$QualityIndex,model.3$residuals, xlab = 'Quality Index', ylab = 'Resdiuals')
#title('Quality Index: Residual vs Predictor')
plot(train.df$QualityIndex, model.3$residuals,
    ylab = "Resdiuals",
    xlab = "Quality Index",
    main = 'Quality Index vs. Resdiuals')
abline(lm(model.3$residuals ~ train.df$QualityIndex), col = "red")
lines(lowess(train.df$QualityIndex,model.3$residuals),
     col = "blue")
```

```
# Note that these plots are sufficient to assess the GOF of a regression model in Predict 410.

####################################################################################
#fit a regular model on total Sqaure Feet - model 1
####################################################################################

# Fit a linear regression model with R
model.1 <- lm(SalePrice ~ TotalSqftCalc, data=train.df)

# Display model summary
summary(model.1)

# List out components of lm object
names(model.1)

# Access a component of lm object
model.1$coefficients

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.1 <- mean(model.1$residuals^2)
mae.1 <- mean(abs(model.1$residuals))


# BASE R diagnostic plot for lm object
plot(model.1)

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.1)

# Hint:  We need to check the assumptions of normality and homoscedasticity;
# (1) QQ Plot
# (2) Scatterplot of residuals versus predictor


# Use the Base R functon qqplot() to assess the normality of the residuals
par(mfrow = c(1, 2), oma = c(0, 0, 2, 0))
qqnorm(model.1$residuals)
qqline(model.1$residuals)

# Make a scatterplot
#plot(train.df$TotalSqftCalc,model.1$residuals)
#title('Residual vs Predictor')
plot(train.df$TotalSqftCalc, model.3$residuals,
    ylab = "Resdiuals",
    xlab = "Total Square Feet",
    main = 'Total Square Feet vs. Resdiuals')
abline(lm(model.3$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.3$residuals),
     col = "blue")

# Note that these plots are sufficient to assess the GOF of a regression model in Predict 410.

####################################################################################
#fit a regular model on total Sqaure Feet & Quality Index - model 4
####################################################################################

# Fit a linear regression model with R
model.4 <- lm(SalePrice ~ TotalSqftCalc + QualityIndex, data=train.df)

# Display model summary
summary(model.4)

# List out components of lm object
names(model.4)
```

```
# Access a component of lm object
model.4$coefficients

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.4 <- mean(model.1$residuals^2)
mae.4 <- mean(abs(model.1$residuals))


# BASE R diagnostic plot for lm object
plot(model.4)
# Not toouseful for writing a report

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.4)

# Hint:  We need to check the assumptions of normality and homoscedasticity;
# (1) QQ Plot
# (2) Scatterplot of residuals versus predictor


# Use the Base R functon qqplot() to assess the normality of the residuals
par(mfrow = c(1, ), oma = c(0, 0, 2, 0))
qqnorm(model.4$residuals)
qqline(model.4$residuals)

# Make a scatterplot
plot(train.df$TotalSqftCalc+train.df$QualityIndex,model.4$residuals)
title('Residual vs Predictor 1')

# Make a scatterplot
plot(train.df$QualityIndex,model.4$residuals)
title('Residual vs Predictor 2')
# Note that these plots are sufficient to assess the GOF of a regression model in Predict 410.



##########################################################################################
# Diagnostic plots using Weisberg's car package
##########################################################################################
# First we need to install the car package
# See Section 1.2.4 p.31 of CAR (Companion to Applied Regression)

# Install package and all other needed packages
#install.packages('car', dependencies=TRUE)

# Load library into your active R session
#library(car)

# Use the function qqPlot() in the car package to assess the Studentized residuals
qqPlot(model.1)

# Note that this is not the typical QQ plot.  Also note that the Studentized residuals
# have a different distribution than the standard residuals.


# Cook's Distance Plot
influenceIndexPlot(model.1,vars=c('Cook','hat'))

# Note that the x index is not correct.  They are the row labels from the original
# data frame.

rownames(train.df) <- seq(1,length(model.1$residuals),1)
model.1 <- lm(SalePrice ~ TotalSqftCalc, data=train.df)
influenceIndexPlot(model.1,vars=c('Cook','hat'))
```

```
# If we want the labels correct, then we have to go back to the original data frame
# and fix the row names, then refit the model, and then call the plot.


####################################################################################
#add loess smoother to the scatterplots
####################################################################################

loessMod50 <- loess(model.1$residuals~ train.df$TotalSqftCalc,  span=0.50) # 50% smoothing span

smoothed50 <- predict(loessMod50)

par(mfrow = c(1, 1))
plot(y = model.1$residuals, x = train.df$TotalSqftCalc, type="l"
     , main="Loess Smoothing and Prediction", xlab="Total Square Feet", ylab="Residuals")
lines(smoothed50, train.df$TotalSqftCalc, col="blue")

#definitely need to fix the loess model


####################################################################################
#Switch Quality Index model - 5 to log(SalesPrice)
####################################################################################

# Fit a linear regression model with R
model.5 <- lm(log(SalePrice) ~ QualityIndex, data=train.df)

# Display model summary
summary(model.5)

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.5 <- mean(model.5$residuals^2)
mae.5 <- mean(abs(model.5$residuals))

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.5)

# Use the Base R functon qqplot() to assess the normality of the residuals
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
# Make a scatterplot
plot(train.df$QualityIndex, model.3$residuals,
    ylab = "Resdiuals",
    xlab = "Quality Index",
    main = 'Quality Index vs. Resdiuals')
abline(lm(model.3$residuals ~ train.df$QualityIndex), col = "red")
lines(lowess(train.df$QualityIndex,model.3$residuals),
     col = "blue")

plot(train.df$QualityIndex, model.5$residuals,
    ylab = "Resdiuals",
    xlab = "Quality Index",
    main = 'Natural Log: Quality Index vs. Resdiuals')
abline(lm(model.5$residuals ~ train.df$QualityIndex), col = "red")
lines(lowess(train.df$QualityIndex,model.5$residuals),
     col = "blue")

qqnorm(model.3$residuals)
qqline(model.3$residuals)

qqnorm(model.5$residuals)
qqline(model.5$residuals)
```

```
######################################################################################
#Switch TotalSqftCalc model - 6 to log(SalesPrice)
######################################################################################

# Fit a linear regression model with R
model.6 <- lm(log(SalePrice) ~ TotalSqftCalc, data=train.df)

# Display model summary
summary(model.6)

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.6 <- mean(model.6$residuals^2)
mae.6 <- mean(abs(model.6$residuals))

# BASE R diagnostic plot for lm object
plot(model.6)

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))

plot(train.df$TotalSqftCalc, model.1$residuals,
     ylab = "Resdiuals",
     xlab = "Total Square Feet",
     main = 'Total Square Feet vs. Resdiuals')
abline(lm(model.1$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.1$residuals),
      col = "blue")

plot(train.df$TotalSqftCalc, model.6$residuals,
     ylab = "Resdiuals",
     xlab = "Total Square Feet",
     main = 'Natural Log: Total Square Feet vs. Resdiuals')
abline(lm(model.6$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.6$residuals),
      col = "blue")

qqnorm(model.1$residuals)
qqline(model.1$residuals)

qqnorm(model.6$residuals)
qqline(model.6$residuals)



######################################################################################
#Switch multiple regression model - 7 to log(SalesPrice)
######################################################################################

# Fit a linear regression model with R
model.7 <- lm(log(SalePrice) ~ TotalSqftCalc + QualityIndex, data=train.df)

# Display model summary
summary(model.7)

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.7 <- mean(model.7$residuals^2)
mae.7 <- mean(abs(model.7$residuals))

# BASE R diagnostic plot for lm object
plot(model.7)

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.7)
```

```
# Make a scatterplot
plot(train.df$TotalSqftCalc,model.7$residuals)
title('Residual vs Predictor 1')

# Make a scatterplot
plot(train.df$QualityIndex,model.7$residuals)
title('Residual vs Predictor 2')

par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))

plot(train.df$TotalSqftCalc, model.1$residuals,
    ylab = "Resdiuals",
    xlab = "Total Square Feet",
    main = 'Total Square Feet vs. Resdiuals')
abline(lm(model.1$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.1$residuals),
     col = "blue")

plot(train.df$TotalSqftCalc, model.6$residuals,
    ylab = "Resdiuals",
    xlab = "Total Square Feet",
    main = 'Natural Log: Total Square Feet vs. Resdiuals')
abline(lm(model.6$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.6$residuals),
     col = "blue")
par(mfrow = c(1, 2), oma = c(0, 0, 2, 0))
qqnorm(model.4$residuals, main = 'Untransformed')
qqline(model.4$residuals)

qqnorm(model.7$residuals, main = 'Transformed')
qqline(model.7$residuals)
```

Assignment #3
Lauren Camero

Introduction:

In this report, our objective is to provide approximations of values for the typical home in Ames, Iowa. We used a data set from the Ames Assessor's Office for individual resident properties sold in Ames, Iowa from 2006 to 2010. We are beginning to build regression models to estimate the home sale price.

First, we defined the sample population by using drop conditions to limit our sample population to the observations that best serve our purpose. We then created a waterfall to exclude these drop conditions and defined our sample population.

Next, we completed simple linear regression models after observing our variables in an exploratory data analysis (EDA) and selected two variables that we believe will be good predictors of sale price using descriptive statistics. We fit two simple linear regression models and assessed their goodness of fit.

After analyzing and selecting our two predictor variables, we combined our two simple linear regression models into a multiple regression model and assessed the goodness of fit of this output. We, furthermore, examined the goodness of fit of our model by neighborhood to evaluate which neighborhood the model would perform best in.

Finally, we transformed the response variable from sale price to the natural logarithm of sale price and compared the new model outputs to the untransformed version of our response variable to understand which form of sales price should be utilized in our model. The EDA and models were built in R programming language.

Data:

The data set is comprised of 82 variables measured from 2930 individual residential properties sold in Ames, IA from 2006 to 2010 collected by Ames Assessor's Office for individual resident properties. Description of the variables can be found in the referenced data documentation.

Sample Definition:

In the data survey and data quality check, we observed several housing qualities that could be excluded from our analysis to better target the population and created a waterfall drop condition subset of our data. Our eligible population required that we have one-family homes in normal sales condition on a paved road. Similarly, houses in the refined data set had to be built after 1950 with a general living area greater than 800 square feet with a basement. After observing outliers in our Lot Area variables, we added a condition that the Lot Area had to be less than 40,000 square feet. Narrowing our population decreased our number of observations from 2,930 to 1,461 as shown in Figure 1.

Figure 1: Waterfall Conditions

Variable	Drop Condition	Number of Properties Dropped
Building Type	Not equal to single famile detached	505
Sale Condition	Not equal to normal	423
Street	Not Paved	6
Year Built	Built before 1950	489
Square Feet of Total Basement	Less than 1 sqaure feet	28
Square Feet of Above Ground Living Area	Less than 800 square feet	9
Lot Area	Lot Area less than 40,000 square feet	9

Exploratory Data Analysis:

Below is a quick comparison or correlations between six variables we examined to pick our best predictor variables. As you can see, the Total Square Feet and Quality Index variables had the highest correlation with Sales Price.

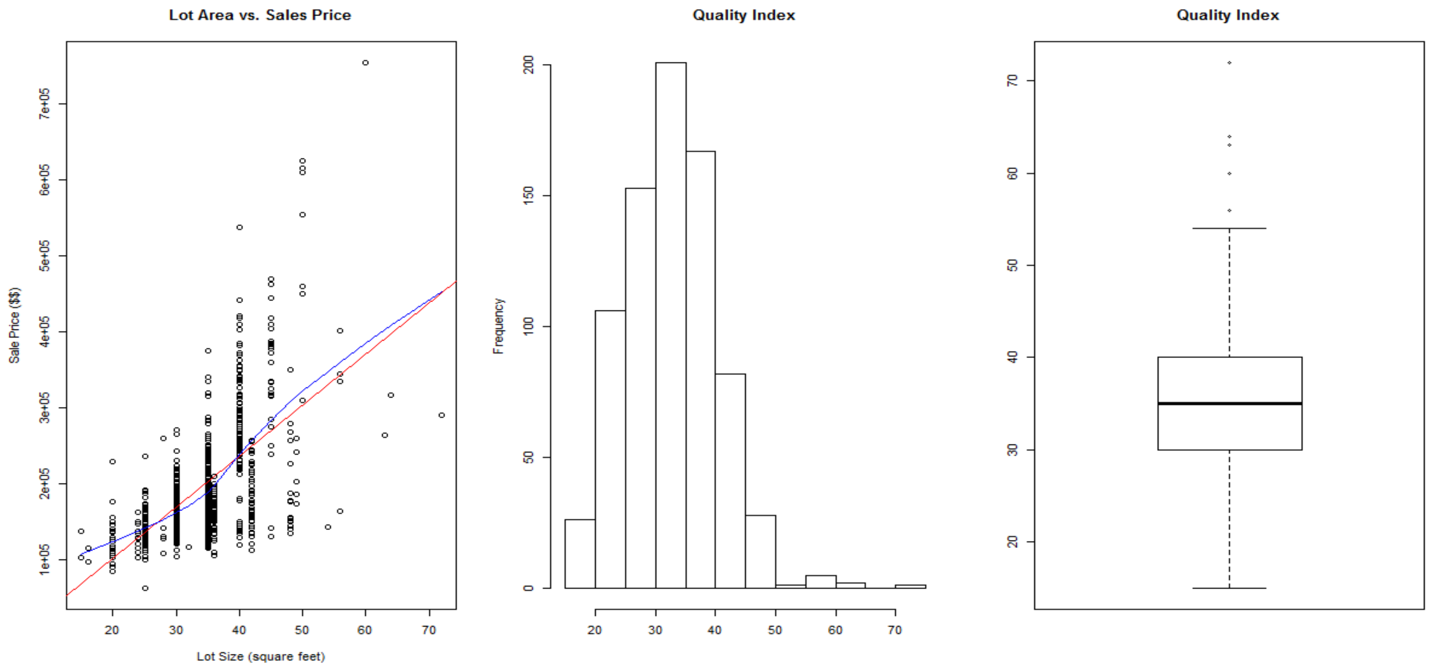
Figure 2: Comparison of correlations between possible predictor variables and Sales Price

Correlations to Sales Price

Total Square Feet	0.7781
Lot Area	0.3667
Quality Index	0.6198
Basement Full Baths	0.2861
Bedrooms Above Ground	0.1605
Fireplaces	0.4942

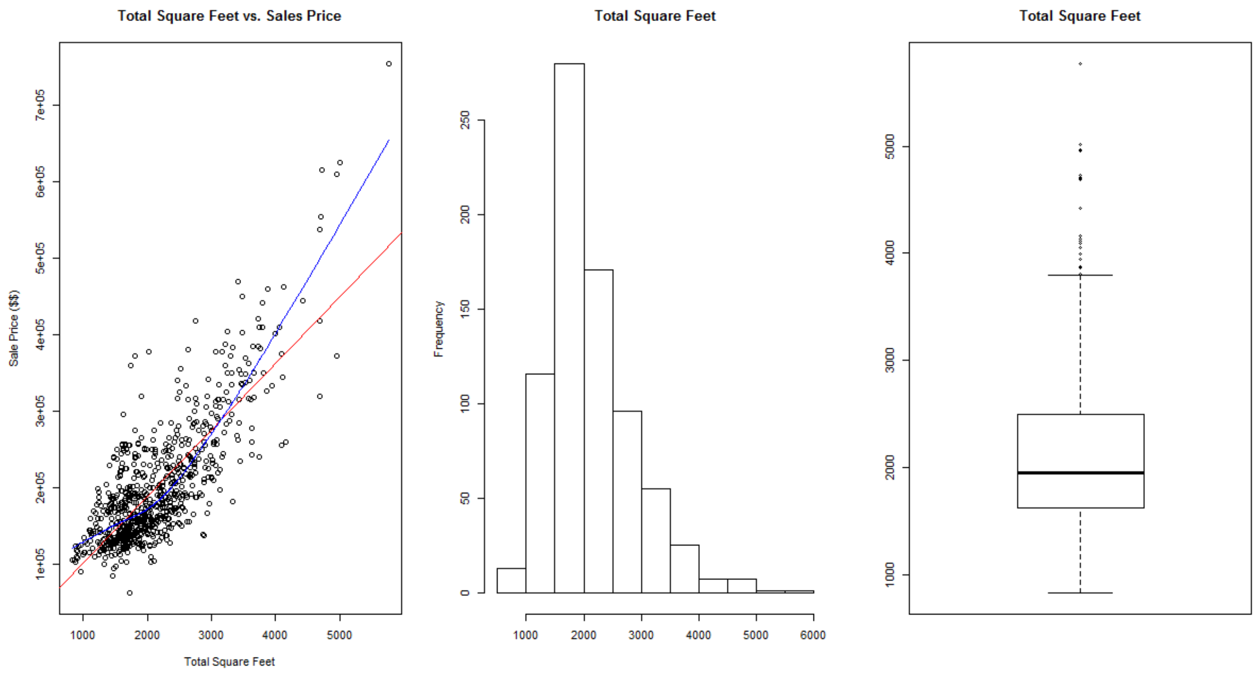
We looked at the Quality Index we had created first. The Quality Index was a metric that we had calculated by taking the two ordinal variables of overall condition and overall quality and multiplying them. As you can see in Figure 3, this variable illustrates almost a normal distribution with a slight positive skew and a higher correlation to sales price.

Figure 3: EDA of Quality Index with blue line as lowess smoother and red line depicting ordinary least squares



We next observed the total square feet variable that we had created by adding the basement finished and unfinished square feet with the above ground living area and assesed our new variable and its relationship with sales price to find that this also visually appears to be a good predictor variable with homoscedastic variance. The total square feet is positively skewed with the median square feet being around 2,000.

Figure 4: EDA of Total Square Feet with blue line as lowess smoother and red line depicting ordinary least squares



Simple Linear Regression Models:

Using our chosen predictor variables of total square feet and quality index, we fit two simple linear regression models and assessed their goodness of fit. We set up the simple linear regression using the formula below and one predictor variable at a time.

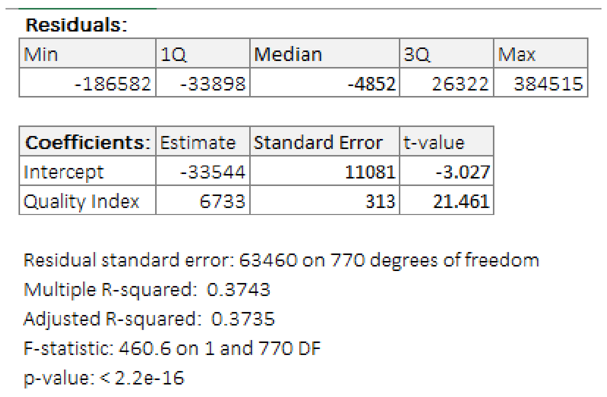
$$Y = \beta_0 + \beta_1 X + \epsilon$$

We assess the goodness of fit of our models using the Q-Q plot to check if the residuals follow a normal distribution and the predictor versus residuals scatterplot to see if there is a relationship.

Model #1: Quality Index:

The scatterplot for the Predictor and the residuals illustrates a funnel. This means that the span of the residuals is increasing as the predictor value increases and suggests that we may want to transform a variable. The Q-Q plot in figure 6 illustrates normality between -1 and 1, but the residuals do not follow a normal distribution above and below this range. The R-squared for this model is 0.3743, much higher than the Lot Area variable and was another criterion used to pick these two predictor variables. Additionally, we saw high significance in our p-value letting us know there is a likely relationship between our response and predictor variables.

Figure 5: Descriptive Statistics of simple linear regression on Quality Index



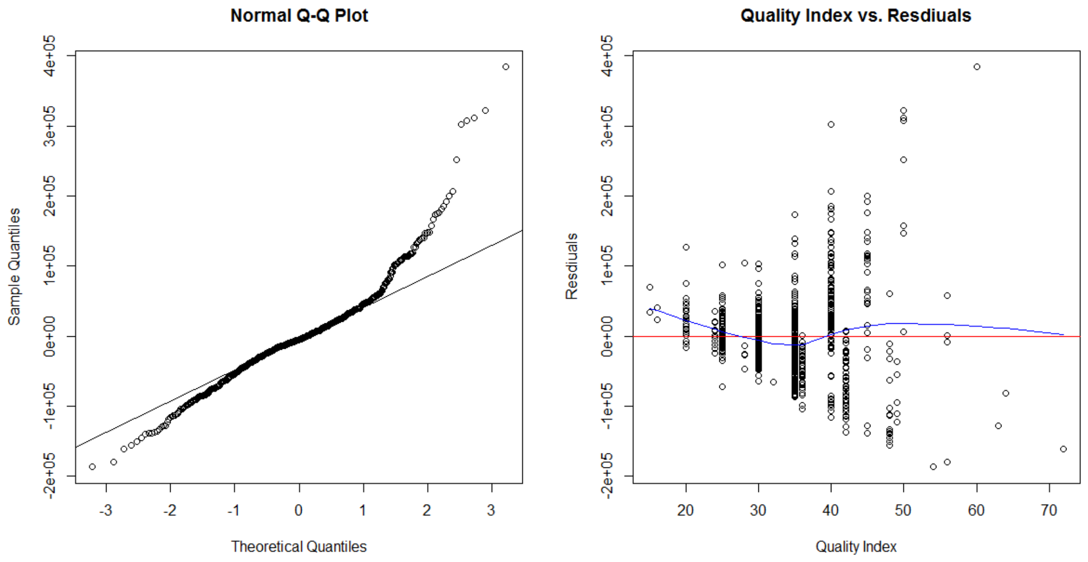
Residuals:

Min	1Q	Median	3Q	Max
-186582	-33898	-4852	26322	384515

Coefficients:	Estimate	Standard Error	t-value
Intercept	-33544	11081	-3.027
Quality Index	6733	313	21.461

Residual standard error: 63460 on 770 degrees of freedom
Multiple R-squared: 0.3743
Adjusted R-squared: 0.3735
F-statistic: 460.6 on 1 and 770 DF
p-value: < 2.2e-16

Figure 6: Q-Q plot and Residual vs. Predictor plot for Quality Index simple linear regression with blue line as lowess smoother and red line depicting ordinary least squares



Model #2: Total Square Feet

The scatterplot for the Predictor and the Residuals is visually correlated with the lowess line and the OLS line very close in the range from 1000 to 4000. The Q-Q plot in figure 6 illustrates normality between -2 and 2, but the residuals do not follow a normal distribution above and below this range. The R-squared for this model was 0.6266. We may need to transform a variable. We see a high significance in our p-value letting us know there is a likely relationship between our response and predictor variables. Our t-value is high in comparison to the standard error which also indicates that a relationship exists.

Figure 7: Descriptive Statistics of simple linear regression on Total Square Feet

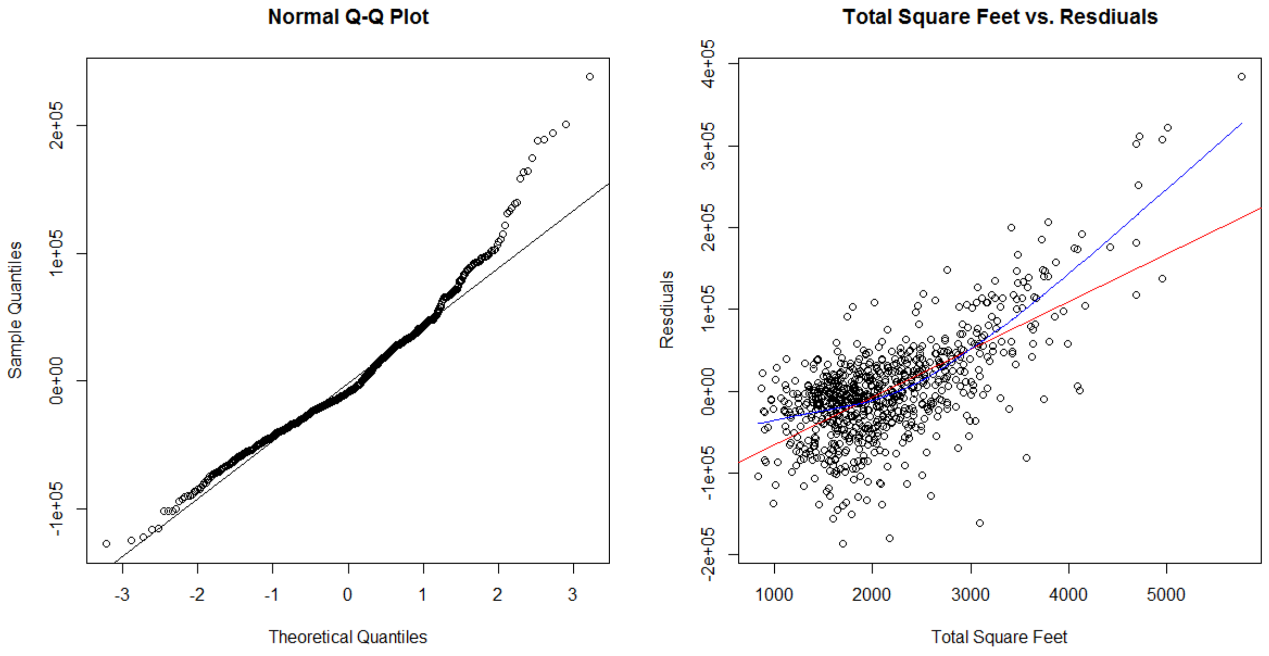
Residuals:

Min	1Q	Median	3Q	Max
-127918	-32327	-8465	28628	238843

Coefficients:	Estimate	Standard Error	t-value
Intercept	14348	5435	2.64
Total Square Feet	86	2	35.94

Residual standard error: 49030 on 770 degrees of freedom
Multiple R-squared: 0.6266
Adjusted R-squared: 0.6261
F-statistic: 1292 on 1 and 770 DF
p-value: < 2.2e-16

Figure 8: Q-Q plot and Residual vs. Predictor plot for Quality Index simple linear regression with blue line as lowess smoother and red line depicting ordinary least squares



Model #3: Multiple Linear Regression Model

Combining our two predictor variables, we form our first Multiple Linear Regression Model and assess it. Looking at the Q-Q plot in Figure 10, we see that the MLR works better than either of the simple linear regressions because the residuals follow a normal distribution between -3 and 2. Similarly, the adjusted R-squared in Figure 9 is 0.7163 which is higher than either of the R-squared outputs in Model 1 or Model 2. Although the model has improved by adding a variable in this case, we will not always get the same outcome and must test the adjusted R-squared when creating a multiple regression model. Additionally, we saw a high significance for our p-value letting us know there is a likely relationship between our response and predictor variables. One item to note is that the standard error is quite high in relation to our estimated intercept meaning the estimate may vary substantially when running the model again. Although our t-value for Total Square Feet is far from zero, our t-value for Quality Index is not as significant when compared to the standard error of the coefficient. This indicates that the response variable's relationship with Quality Index is weaker than its relationship with Total Square Feet.

Figure 9: Descriptive Statistics of simple linear regression on Total Square Feet and Quality Index & the Q-Q plot of residuals

Residuals:

Min	1Q	Median	3Q	Max
-118629	-29137	-2805	25822	202880

Coefficients:	Estimate	Standard Error	t-value
Intercept	-78941	7603	-10.38
Total Square Feet	71	2	30.52
Quality Index	3666	233	15.68

Residual standard error: 42700 on 769 degrees of freedom
Multiple R-squared: 0.7171
Adjusted R-squared: 0.7163
F-statistic: 974.4 on 2 and 769 DF
p-value: < 2.2e-16

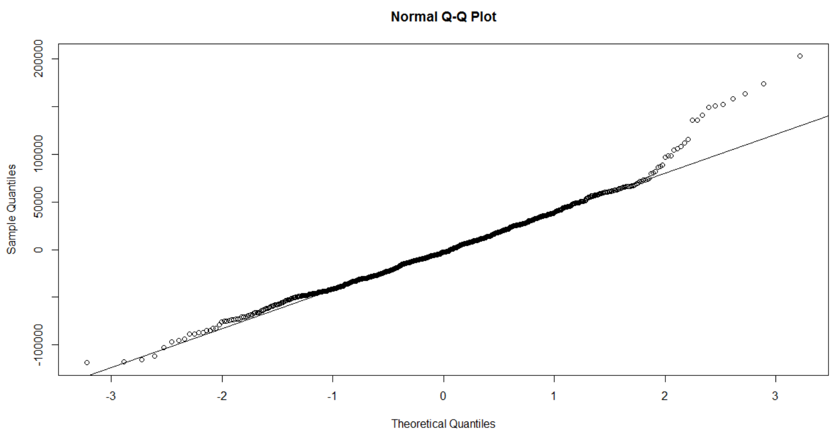


Figure 11 exemplifies the comparison between the two simple linear regressions and the multiple linear regression and illustrates the increase in statistical significance by combining the variables into one model.

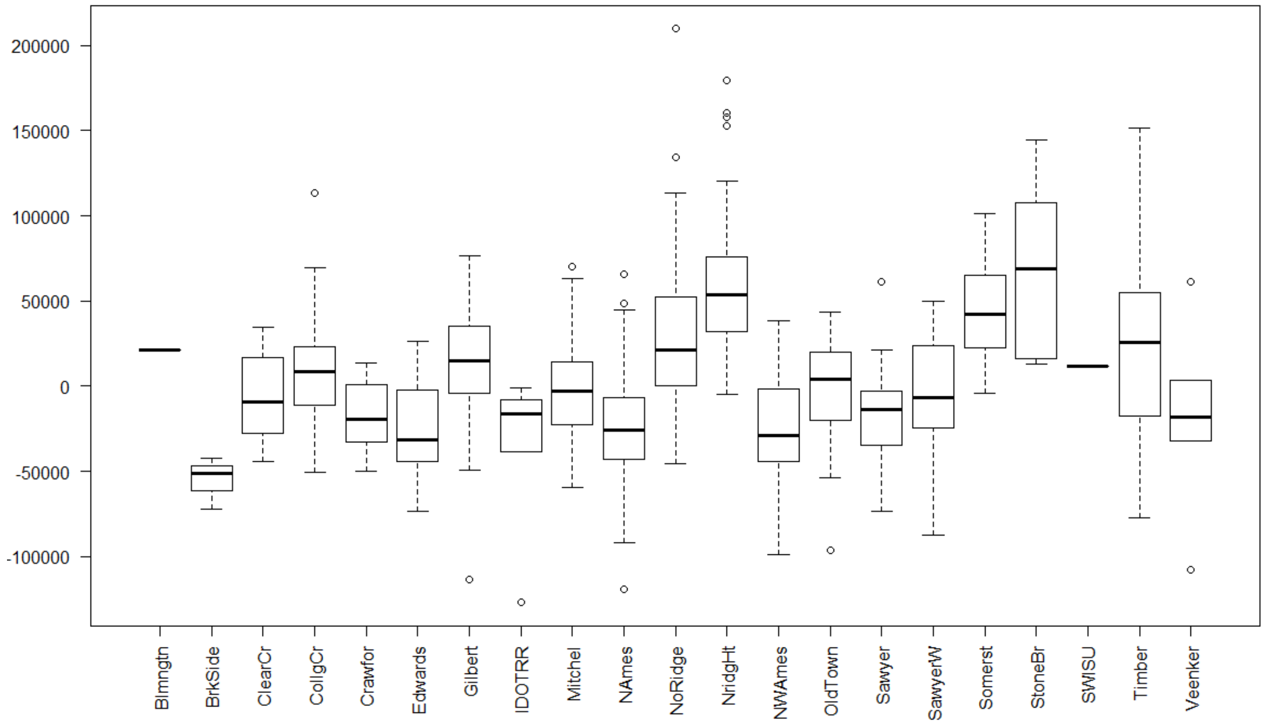
Figure 10: Comparison of statistical tests on models 1-3

SLR		MLR
Quality Index	Total Square Feet	Combined
Residual standard error: 63460 on 770 degrees of freedom	Residual standard error: 49030 on 770 degrees of freedom	Residual standard error: 42700 on 769 degrees of freedom
Multiple R-squared: 0.3743	Multiple R-squared: 0.6266	Multiple R-squared: 0.7171
Adjusted R-squared: 0.3735	Adjusted R-squared: 0.6261	Adjusted R-squared: 0.7163
F-statistic: 460.6 on 1 and 770 DF	F-statistic: 1292 on 1 and 770 DF	F-statistic: 974.4 on 2 and 769 DF
p-value: < 2.2e-16	p-value: < 2.2e-16	p-value: < 2.2e-16

Neighborhood Accuracy:

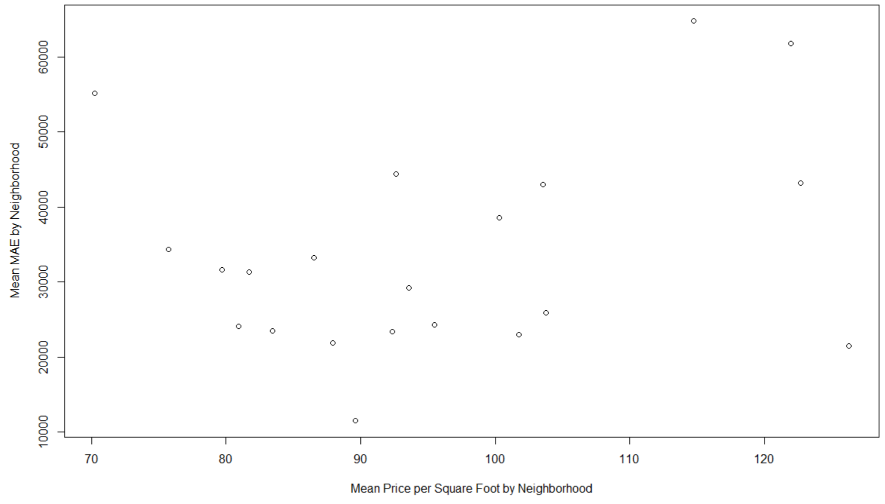
We now want to see if our model will work for all categories of our discreet variable, Neighborhoods. Figure 12 illustrates a boxplot of the residuals of the multiple linear regression by neighborhood. We can see from the graph that Crawford Heights and the South and West side of Iowa University fit our model well. In contrast, we see residual outliers and large variances in Timberland and North Ridge. We can also perceive that we are consistently underpredicting in Brookside and consistently overpredicting in Bloomington Heights.

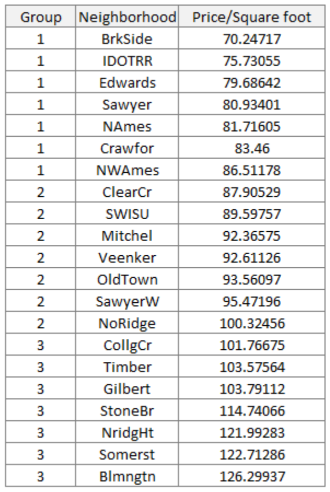
Figure 11: boxplot of residuals by neighborhood



We then plot the Mean Absolute Error (MAE) by neighborhood and Sales Price per Square feet in Figure 13 and see no strong relationship between the two. Figure 12 displays the grouping of the 21 different neighborhoods into three groups by Price per Square Foot.

Figure 12: MAE vs. Mean Sale Price per Square Feet by Neighborhood & table grouping neighborhoods

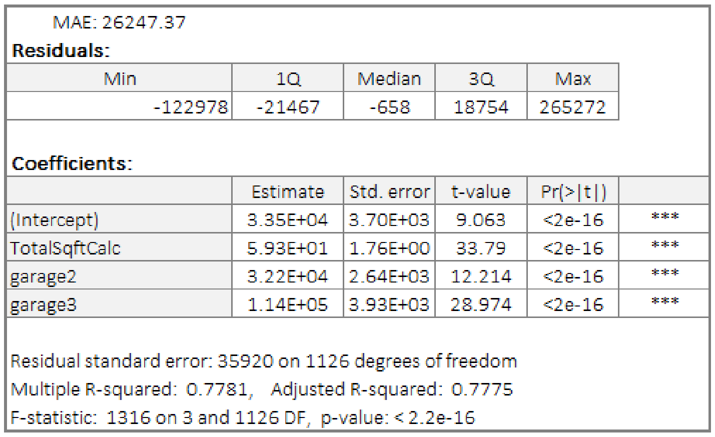




Group	Neighborhood	Price/Square foot
1	BrkSide	70.24717
1	IDOTRR	75.73055
1	Edwards	79.68642
1	Sawyer	80.93401
1	NAmes	81.71605
1	Crawfor	83.46
1	NWAmes	86.51178
2	ClearCr	87.90529
2	SWISU	89.59757
2	Mitchel	92.36575
2	Veenker	92.61126
2	OldTown	93.56097
2	SawyerW	95.47196
2	NoRidge	100.32456
3	CollgCr	101.76675
3	Timber	103.57564
3	Gilbert	103.79112
3	StoneBr	114.74066
3	NridgHt	121.99283
3	Somerst	122.71286
3	Blmngtn	126.29937

We chose a dummy variable, Garage Cars, to set up and observe a multiple linear regression (MLR) on a family of indicators and assess the goodness of fit of our new model. Our baseline category for the new MLR is a zero-capacity car sized garage. The MAE of the multiple regression using size of the garage in car capacity is 26,247, whereas the MAE of the original MLR is 31,836. The decrease in error indicates that the dummy variable improves the model. Figure 13 shows us that all predictor variables were significant in this model.

Figure 13: Summary statistics of MLR with dummy variable



MAE: 26247.37

Residuals:

Min	1Q	Median	3Q	Max
-122978	-21467	-658	18754	265272

Coefficients:

	Estimate	Std. error	t-value	Pr(>|t|)	
(Intercept)	3.35E+04	3.70E+03	9.063	<2e-16	***
TotalSqftCalc	5.93E+01	1.76E+00	33.79	<2e-16	***
garage2	3.22E+04	2.64E+03	12.214	<2e-16	***
garage3	1.14E+05	3.93E+03	28.974	<2e-16	***

Residual standard error: 35920 on 1126 degrees of freedom
Multiple R-squared: 0.7781, Adjusted R-squared: 0.7775
F-statistic: 1316 on 3 and 1126 DF, p-value: < 2.2e-16

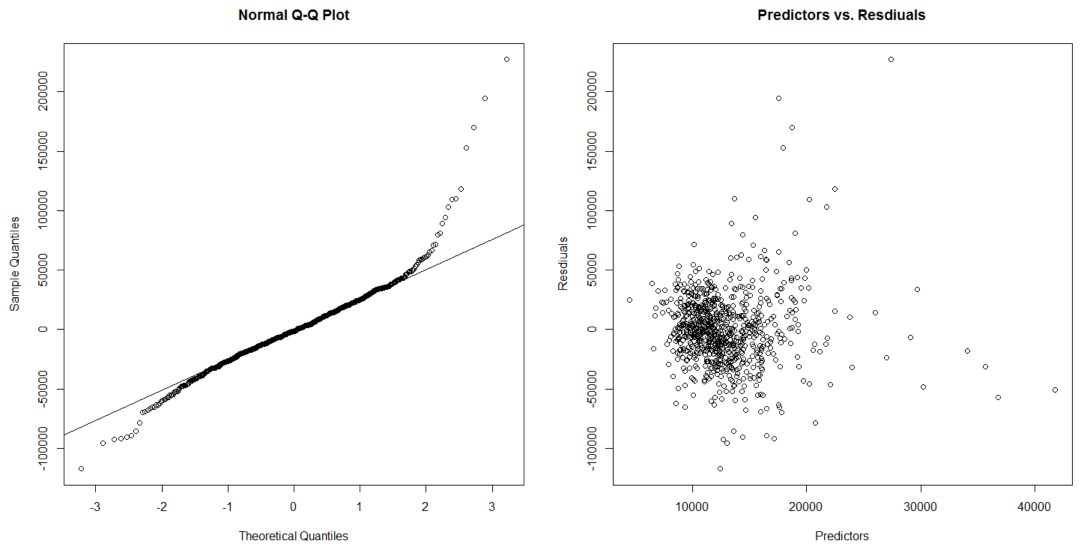
Sale Price versus Log Sale Price as the Response:

We will now transform our response variable and refit our models to test if the natural log of sales price produces a better fit for our predictors than the untransformed. We take four continuous variables and our dummy variable and create a Multiple Linear Regression comparing the transformed and untransformed response variable

Sales Price Model:

Our model consists of five variables: total square feet, quality index, lot area, total rooms above ground, and garage capacity. Figure 14 shows us that our residuals are normal from -2 to 2 but have many outliers beyond that range. The residuals to predictor plot does not oscillate around zero but are clumped together in a certain range. This is a sign that a transformation would be beneficial.

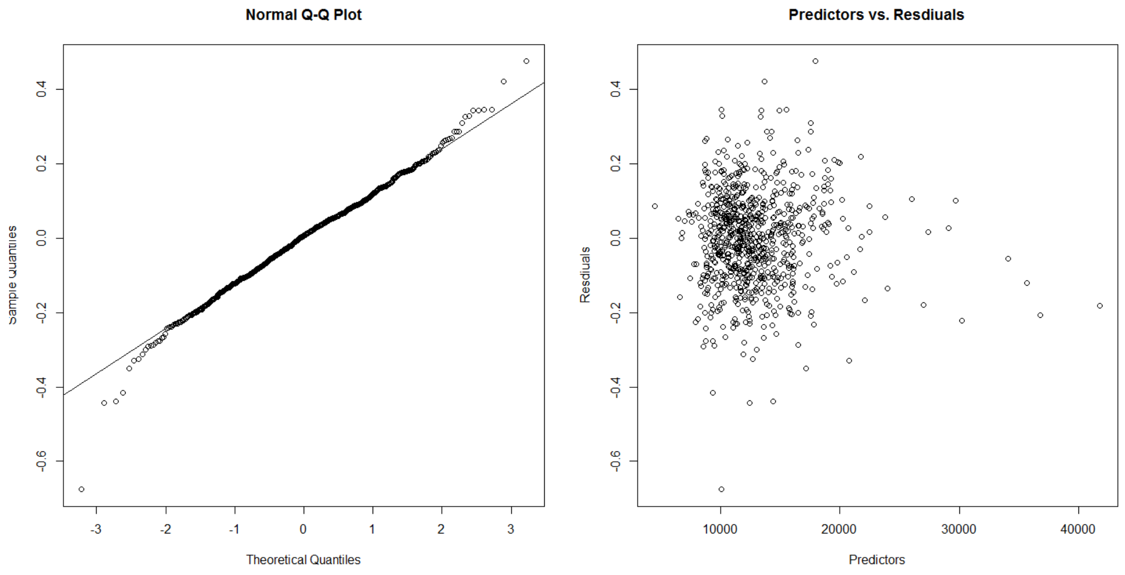
Figure 14: Q-Q plot and Plot of Predictors vs. Residuals for untransformed response variable



Log Sale Price Response Model:

In comparison to the untransformed response variable, Figure 14 shows us that our residuals are normal from -2 to 2 and only slightly off the linear path beyond that range. The residuals to predictor plot oscillates around zero and are clumped together but less densely.

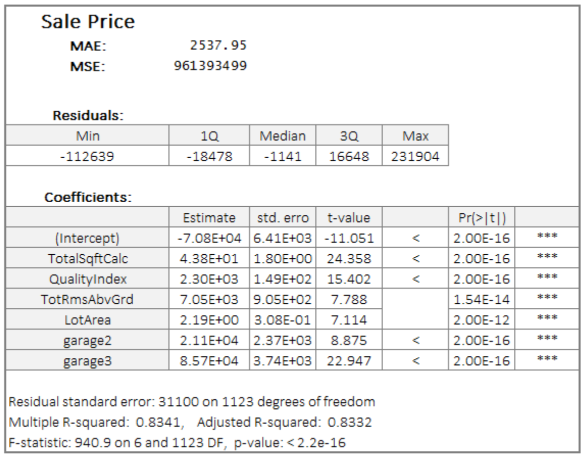
Figure 15: Q-Q plot and Plot of Predictors vs. Residuals for transformed response variable



Comparison and Discussion of Model Fits:

Interpreting the two summary statistics of the multiple linear regression, we can see that the model performed better when we used the natural log of our response variable. The Adjusted R-Squared was higher at 0.8526 rather than 0.8332. Similarly, our standard error decreased. To compare the Mean Absolute Error (MAE) and the Mean Squared Error (MSE) normalized, we must convert the Sale Price back to the untransformed version. Once converted, we can see that the natural log of Sale Price is a better overall response variable for the model since both the MAE and MSE are lower than the untransformed model. Generally, if the values of a variable range over one order of magnitude, using a log transformation will be beneficial. None of our other variables vary as greatly as the Sales Price, so we will only manipulate the response variable for this model.

Figure 16: Summaries of MLR with Sales Price and the log of Sales Price as response variables



Sale Price

MAE: 2537.95
MSE: 961393499

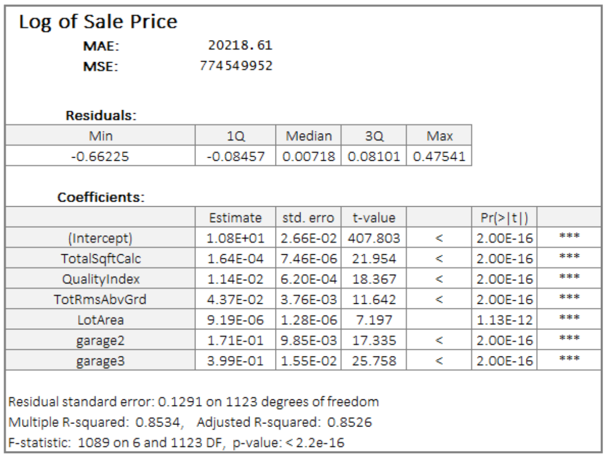
Residuals:

Min	1Q	Median	3Q	Max
-112639	-18478	-1141	16648	231904

Coefficients:

	Estimate	std. erro	t-value		Pr(>|t|)	
(Intercept)	-7.08E+04	6.41E+03	-11.051	<	2.00E-16	***
TotalSqftCalc	4.38E+01	1.80E+00	24.358	<	2.00E-16	***
QualityIndex	2.30E+03	1.49E+02	15.402	<	2.00E-16	***
TotRmsAbvGrd	7.05E+03	9.05E+02	7.788		1.54E-14	***
LotArea	2.19E+00	3.08E-01	7.114		2.00E-12	***
garage2	2.11E+04	2.37E+03	8.875	<	2.00E-16	***
garage3	8.57E+04	3.74E+03	22.947	<	2.00E-16	***

Residual standard error: 31100 on 1123 degrees of freedom
Multiple R-squared: 0.8341,   Adjusted R-squared: 0.8332
F-statistic: 940.9 on 6 and 1123 DF, p-value: < 2.2e-16



Log of Sale Price

MAE: 20218.61
MSE: 774549952

Residuals:

Min	1Q	Median	3Q	Max
-0.66225	-0.08457	0.00718	0.08101	0.47541

Coefficients:

	Estimate	std. erro	t-value		Pr(>|t|)	
(Intercept)	1.08E+01	2.66E-02	407.803	<	2.00E-16	***
TotalSqftCalc	1.64E-04	7.46E-06	21.954	<	2.00E-16	***
QualityIndex	1.14E-02	6.20E-04	18.367	<	2.00E-16	***
TotRmsAbvGrd	4.37E-02	3.76E-03	11.642	<	2.00E-16	***
LotArea	9.19E-06	1.28E-06	7.197		1.13E-12	***
garage2	1.71E-01	9.85E-03	17.335	<	2.00E-16	***
garage3	3.99E-01	1.55E-02	25.758	<	2.00E-16	***

Residual standard error: 0.1291 on 1123 degrees of freedom
Multiple R-squared: 0.8534,   Adjusted R-squared: 0.8526
F-statistic: 1089 on 6 and 1123 DF, p-value: < 2.2e-16

Conclusions:

We have discovered that the quality index and the total square feet are good predictor variables in the objective to estimate the value of homes in Ames, Iowa. We have observed that a multiple regression using both variables will provide us a better estimation since the model is better fit. Additionally, we have discovered that our models improve if we transform the response variable, sales price, by taking its natural log. Since we have seen an improvement in our model by adding predictor variables, we may want to include another variable and test the fitness of that model.

References:

Dean De Cock. (2011). Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project, Journal of Statistics Education, 19:3. Retrieved from http://amstat.tandfonline.com/doi/abs/10.1080/10691898.2011.11889627#.WciC-WinFTE
Dean De Cock. Ames Housing Data Documentation. Retrieved from https://ww2.amstat.org/publications/jse/v19n3/decock/DataDocumentation.txt

Code:

```
# Lauren Camero
# 1.20.2018
# assignment#2.R

# Read in csv file for Ames housing data;

# Note that back slash is an escape character in R so we use \\ when we want \;
setwd('C:/Users/lcamero/Desktop/Predict 410')
file.name <- paste('ames_housing_data.csv',sep='');

# Read in the csv file into an R data frame;
ames.df <- read.csv(file.name,header=TRUE,stringsAsFactors=FALSE);


# Create a waterfall of drop conditions;
ames.df$dropCondition <- ifelse(ames.df$BldgType!='1Fam','01: Not SFR',
        ifelse(ames.df$SaleCondition!='Normal','02: Non-Normal Sale',
        ifelse(ames.df$Street!='Pave','03: Street Not Paved',
        ifelse(ames.df$YearBuilt <1950,'04: Built Pre-1950',
        ifelse(ames.df$TotalBsmtSF <1,'05: No Basement',
        ifelse(ames.df$GrLivArea <800,'06: LT 800 SqFt',
        ifelse(ames.df$LotArea >40000,'07: LT under 40000 SqFt',
        '99: Eligible Sample')
        ))))));

# Save the table
waterfall <- table(ames.df$dropCondition);

# Format the table as a column matrix for presentation;
as.matrix(waterfall,7,1)


# Eliminate all observations that are not part of the eligible sample population;
eligible.population <- subset(ames.df,dropCondition=='99: Eligible Sample');

# Check that all remaining observations are eligible;
table(eligible.population$dropCondition);


##############################################################################################
# Create a list of interesting predictor variables
##############################################################################################

str(ames.df)

# Predictor variables that I like:
LotFrontage
LotArea
LotConfig
Neighborhood
HouseStyle
OverallQual
OverallCond
YearBuilt
YearRemodel
Exterior1
BsmtFinSF1
BsmtFinSF2
CentralAir
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
```

BedroomAbvGr
TotRmsAbvGrd
Fireplaces
GarageCars
GarageArea
WoodDeckSF
OpenPorchSF
EnclosedPorch
ThreeSsnPorch
ScreenPorch
PoolArea
MoSold
YrSold
SaleCondition
SalePrice

```
# Make a vector of the names;

keep.vars <- c('SID','PID','LotFrontage','LotArea','LotConfig','Neighborhood',
'HouseStyle','OverallQual','OverallCond','YearBuilt','YearRemodel','Exterior1',
'BsmtFinSF1','BsmtFinSF2','CentralAir','GrLivArea','BsmtFullBath','BsmtHalfBath',
'FullBath','HalfBath','BedroomAbvGr','TotRmsAbvGrd','Fireplaces','GarageCars',
'GarageArea','WoodDeckSF','OpenPorchSF','EnclosedPorch','ThreeSsnPorch',
'ScreenPorch','PoolArea','MoSold','YrSold','SaleCondition','SalePrice'
);


# Note that the R data frame is a (rectangular) list object which means that it can be
# accessed in two ways - as a matrix or as a list;
# Note that the keep.vars are the COLUMNS that we want to keep, not the rows;

skinny.df <- eligible.population[,keep.vars];

# Use the structure command to view the contents of the data frame;
str(skinny.df)




##########################################################################################
# Delete observations with missing values
##########################################################################################
sample.df <- na.omit(skinny.df);

# Check the change in dimension;
dim(skinny.df)
dim(sample.df)

dim(skinny.df)-dim(sample.df)



##########################################################################################
# Define some discrete variables and indicator variables
##########################################################################################

# Define total square footage
sample.df$TotalSqftCalc <- sample.df$BsmtFinSF1+sample.df$BsmtFinSF2+sample.df$GrLivArea;

# Define total bathrooms
sample.df$TotalBathCalc <- sample.df$BsmtFullBath + 0.5*sample.df$BsmtHalfBath ++
                  + sample.df$FullBath + 0.5*sample.df$HalfBath;


# Corner lot indicator
# ifelse(condition,valueTrue,valueFalse)
```

```
sample.df$CornerLotInd <- ifelse(sample.df$LotConfig=='Corner',1,0);

# Check how the indicator is assigned
table(sample.df$CornerLotInd,sample.df$LotConfig)

# Define two indicators for fire places
table(sample.df$Fireplaces)

# Intercept Adjustment for a single fireplace
sample.df$FireplaceInd1 <- ifelse((sample.df$Fireplaces>0)&(sample.df$Fireplaces<2),1,0);
table(sample.df$FireplaceInd1,sample.df$Fireplaces)

# Intercept Adjustment for 2 or more fireplaces
sample.df$FireplaceInd2 <- ifelse((sample.df$Fireplaces>1),1,0);
table(sample.df$FireplaceInd2,sample.df$Fireplaces)

# Additive Intercept Adjustment for a single fireplace
sample.df$FireplaceAdder1 <- ifelse((sample.df$Fireplaces>0),1,0);

# Additive Intercept Adjustment for 2 or more fireplaces
sample.df$FireplaceAdder2 <- ifelse((sample.df$Fireplaces>1),1,0);

table(sample.df$FireplaceAdder1,sample.df$Fireplaces)
table(sample.df$FireplaceAdder2,sample.df$Fireplaces)



# Central Air Indicator
sample.df$CentralAirInd <- ifelse(sample.df$CentralAir=='Y',1,0);
table(sample.df$CentralAirInd)
# Looks like this is not useful since almost all homes have central air


# Exterior Siding Type
sample.df$BrickInd <- ifelse(sample.df$Exterior1=='BrkFace',1,0);
sample.df$VinylSidingInd <- ifelse(sample.df$Exterior1=='VinylSd',1,0);

# Pool Indicator
sample.df$PoolInd <- ifelse(sample.df$PoolArea>0,1,0);

# Wood Deck Indicator
sample.df$WoodDeckInd <- ifelse(sample.df$WoodDeckSF>0,1,0);

# Porch Indicator - Open Porch OR Screen Porch
sample.df$PorchInd <- ifelse((sample.df$OpenPorchSF>0)||(sample.df$ScreenPorch>0),1,0);

# Quality Index
sample.df$QualityIndex <- sample.df$OverallQual*sample.df$OverallCond;

table(sample.df$QualityIndex)


# Year Sold Indicators
sample.df$I2006 <- ifelse(sample.df$YrSold==2006,1,0);
sample.df$I2007 <- ifelse(sample.df$YrSold==2007,1,0);
sample.df$I2008 <- ifelse(sample.df$YrSold==2008,1,0);
sample.df$I2009 <- ifelse(sample.df$YrSold==2009,1,0);
sample.df$I2010 <- ifelse(sample.df$YrSold==2010,1,0);

table(sample.df$YrSold)
table(sample.df$I2006)
table(sample.df$I2007)
table(sample.df$I2008)
table(sample.df$I2009)
table(sample.df$I2010)
```

```
# List out sample.df
str(sample.df)



##########################################################################################
# Add a train/test flag to split the sample
##########################################################################################
sample.df$u <- runif(n=dim(sample.df)[1],min=0,max=1);
sample.df$train <- ifelse(sample.df$u<0.70,1,0);

# Check the counts on the train/test split
table(sample.df$train)

# Check the train/test split as a percentage of whole
table(sample.df$train)/dim(sample.df)[1]



##########################################################################################
# Save data frame as an .RData data object
##########################################################################################

# Save the R data frame as an .RData object
saveRDS(sample.df,file='ames_housing_data.csv');

# Read (or reload) the .RData object as an R data frame
a <- readRDS('ames_housing_data.csv');

# Check it
str(a)

########################################################################################
#Begin the EDA with Variable 1: total square feet
########################################################################################


# Check it
str(sample.df)

# Technically we should perform our EDA on the training data set
train.df <- subset(sample.df,train==1);



##################################################################################
# BASE R Scatterplot
##################################################################################

# Let's control the R plot to make it pretty
plot(train.df$TotalSqftCalc,train.df$SalePrice/1000,xlab='Total SQFT',ylab='Sale Price (000)',
    main='SQFT and Sale Price')

# In low dimensions scatterplots can be useful for visualizing the relationship
# between the response and a predictor variable


##################################################################################
# BASE R Box Plot
##################################################################################
par(mfrow = c(1, 1))
boxplot(train.df$SalePrice/1000 ~ train.df$Neighborhood, las=2)
title('Sale Price By Neighborhood')

# Boxplots are the tool of choice for visualizing factor variables
```

```
###################################################################################
# Treating a discrete variable as continuous
###################################################################################

plot(train.df$TotalSqftCalc,train.df$SalePrice)

#########################################################################################
#Begin the EDA with Variable 2: Lot Area
#########################################################################################

par(mfrow = c(1, 3))
plot(train.df$LotArea, train.df$SalePrice,
    ylab = "Sale Price ($$)",
    xlab = "Lot Size (square feet)",
    main = 'Lot Area vs. Sales Price')
abline(lm(train.df$SalePrice ~ train.df$LotArea), col = "red")
lines(lowess(train.df$LotArea, train.df$SalePrice),
     col = "blue")
hist(train.df$LotArea, main='Lot Area', xlab = '')
boxplot(train.df$LotArea, main='Lot Area')

par(mfrow = c(1, 3))
plot(train.df$QualityIndex, train.df$SalePrice,
    ylab = "Sale Price ($$)",
    xlab = "Quality Index",
    main = 'Quality Index vs. Sales Price')
abline(lm(train.df$SalePrice ~ train.df$QualityIndex), col = "red")
lines(lowess(train.df$QualityIndex, train.df$SalePrice),
     col = "blue")
hist(train.df$QualityIndex, main='Quality Index', xlab = '')
boxplot(train.df$QualityIndex, main='Quality Index')

par(mfrow = c(1, 3))
plot(train.df$TotalSqftCalc, train.df$SalePrice,
    ylab = "Sale Price ($$)",
    xlab = "Total Square Feet",
    main = 'Total Square Feet vs. Sales Price')
abline(lm(train.df$SalePrice ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc, train.df$SalePrice),
     col = "blue")
hist(train.df$TotalSqftCalc, main='Total Square Feet', xlab = '')
boxplot(train.df$TotalSqftCalc, main='Total Square Feet')


cor(train.df$TotalSqftCalc,train.df$SalePrice)
cor(train.df$LotArea,train.df$SalePrice)
cor(train.df$QualityIndex,train.df$SalePrice)
cor(train.df$BsmtFullBath,train.df$SalePrice)
cor(train.df$BedroomAbvGr,train.df$SalePrice)
cor(train.df$Fireplaces,train.df$SalePrice)

table(sample.df$BldgType)


#title(main = "Residential Properties

# Sometime we will choose to treat a discrete variable as a continuous variable.
# In these cases the discrete variable needs to have 'enough' values, and a 'nice'
# relationship with the response variable.

####################################################################################
#fit a regular model on Lot Area - model 2
####################################################################################

# Fit a linear regression model with R
```

```
model.2 <- lm(SalePrice ~ LotArea, data=train.df)

# Display model summary
summary(model.2)

# List out components of lm object
names(model.2)

# Access a component of lm object
model.2$coefficients

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.2 <- mean(model.2$residuals^2)
mae.2 <- mean(abs(model.2$residuals))

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.2)

# Hint:  We need to check the assumptions of normality and homoscedasticity;
# (1) QQ Plot
# (2) Scatterplot of residuals versus predictor


# Use the Base R functon qqplot() to assess the normality of the residuals
qqnorm(model.2$residuals)
qqline(model.2$residuals)

# Make a scatterplot
plot(train.df$LotArea,model.2$residuals)
title('Residual vs Predictor')

# this ended up with a very low r squared, so I am looking at another variable
############################################################################################
#fit a regular model on Quality Index model - 3
############################################################################################

# Fit a linear regression model with R
model.3 <- lm(SalePrice ~ QualityIndex, data=train.df)

# Display model summary
summary(model.3)

# List out components of lm object
names(model.3)

# Access a component of lm object
model.3$coefficients

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.3 <- mean(model.3$residuals^2)
mae.3 <- mean(abs(model.3$residuals))

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.2)

# Hint:  We need to check the assumptions of normality and homoscedasticity;
# (1) QQ Plot
# (2) Scatterplot of residuals versus predictor

par(mfrow = c(1, 2), oma = c(0, 0, 2, 0))
# Use the Base R functon qqplot() to assess the normality of the residuals
qqnorm(model.3$residuals)
qqline(model.3$residuals)
```

```
# Make a scatterplot
#plot(train.df$QualityIndex,model.3$residuals, xlab = 'Quality Index', ylab = 'Resdiuals')
#title('Quality Index: Residual vs Predictor')
plot(train.df$QualityIndex, model.3$residuals,
     ylab = "Resdiuals",
     xlab = "Quality Index",
     main = 'Quality Index vs. Resdiuals')
abline(lm(model.3$residuals ~ train.df$QualityIndex), col = "red")
lines(lowess(train.df$QualityIndex,model.3$residuals),
      col = "blue")


# Note that these plots are sufficient to assess the GOF of a regression model in Predict 410.

#######################################################################################
#fit a regular model on total Sqaure Feet - model 1
#######################################################################################

# Fit a linear regression model with R
model.1 <- lm(SalePrice ~ TotalSqftCalc, data=train.df)

# Display model summary
summary(model.1)

# List out components of lm object
names(model.1)

# Access a component of lm object
model.1$coefficients

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.1 <- mean(model.1$residuals^2)
mae.1 <- mean(abs(model.1$residuals))


# BASE R diagnostic plot for lm object
plot(model.1)

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.1)

# Hint:  We need to check the assumptions of normality and homoscedasticity;
# (1) QQ Plot
# (2) Scatterplot of residuals versus predictor


# Use the Base R functon qqplot() to assess the normality of the residuals
par(mfrow = c(1, 2), oma = c(0, 0, 2, 0))
qqnorm(model.1$residuals)
qqline(model.1$residuals)

# Make a scatterplot
#plot(train.df$TotalSqftCalc,model.1$residuals)
#title('Residual vs Predictor')
plot(train.df$TotalSqftCalc, model.3$residuals,
     ylab = "Resdiuals",
     xlab = "Total Square Feet",
     main = 'Total Square Feet vs. Resdiuals')
abline(lm(model.3$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.3$residuals),
      col = "blue")

# Note that these plots are sufficient to assess the GOF of a regression model in Predict 410.

#######################################################################################
```

```
#fit a regular model on total Sqaure Feet & Quality Index - model 4
#####################################################################################

# Fit a linear regression model with R
model.4 <- lm(SalePrice ~ TotalSqftCalc + QualityIndex, data=train.df)

# Display model summary
summary(model.4)

# List out components of lm object
names(model.4)

# Access a component of lm object
model.4$coefficients

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.4 <- mean(model.4$residuals^2)
mae.4 <- mean(abs(model.4$residuals))


# BASE R diagnostic plot for lm object
plot(model.4)
# Not toouseful for writing a report

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.4)

# Hint:  We need to check the assumptions of normality and homoscedasticity;
# (1) QQ Plot
# (2) Scatterplot of residuals versus predictor


# Use the Base R functon qqplot() to assess the normality of the residuals
par(mfrow = c(1, ), oma = c(0, 0, 2, 0))
qqnorm(model.4$residuals)
qqline(model.4$residuals)

# Make a scatterplot
plot(train.df$TotalSqftCalc+train.df$QualityIndex,model.4$residuals)
title('Residual vs Predictor 1')

# Make a scatterplot
plot(train.df$QualityIndex,model.4$residuals)
title('Residual vs Predictor 2')
# Note that these plots are sufficient to assess the GOF of a regression model in Predict 410.

par(mfrow = c(1, 1))
boxplot(model.4$residuals ~ train.df$Neighborhood, las=2)

meanSalePrice <- aggregate(train.df$SalePrice, by=list(Neighborhood=train.df$Neighborhood), FUN=mean)
colnames(meanSalePrice) <- c('Neighborhood','meanSalePrice')

mae.neighborhood <- aggregate(abs(model.4$residuals), by=list(Neighborhood=train.df$Neighborhood), FUN=mean)
mae.4 <- mean(abs(model.4$residuals))
train.df$price.per.sqft <- train.df$SalePrice/train.df$TotalSqftCalc
mean.price.per.neighborhood <- aggregate(train.df$price.per.sqft, by=list(Neighborhood=train.df$Neighborhood), FUN=mean)

plot(mean.price.per.neighborhood$x, mae.neighborhood$x,
    xlab = 'Mean Price per Square Foot by Neighborhood',
    ylab = 'Mean MAE by Neighborhood')

#########################################################################################
#Code a family of indicator variables to include in your multiple regression model
# Table GarageCars
table(sample.df$GarageCars)
```

```
# Let's create a family of indicator variables;
# We will take the baseline category to be 0;
# What does this mean?  Why am I creating three indicator variables?
# Should I be creating an indicator variable for 0?
sample.df$garage1 <- ifelse(sample.df$GarageCars==1,1,0);
sample.df$garage2 <- ifelse(sample.df$GarageCars==2,1,0);
sample.df$garage3 <- ifelse(sample.df$GarageCars>=3,1,0);


# Check the indicator assignment against the original table results;
table(sample.df$garage1)
table(sample.df$garage2)
table(sample.df$garage3)



# Fit a linear regression model with R
model.10 <- lm(SalePrice ~ TotalSqftCalc + garage1 + garage2 + garage3, data=sample.df)

# Display model summary
summary(model.10)


# Fit a second model;
model.20 <- lm(SalePrice ~ TotalSqftCalc + garage2 + garage3, data=sample.df)

# Display model summary
summary(model.20)
#our base category is 0
mae.20 <- mean(abs(model.20$residuals))

mae.20
mae.4
#the MAE of the multiple regression using size of the garage in car capacity is much less than the original MLR

###########################################################################################
#Difference in sales price vs log of sales price
###########################################################################################
model.30 <- lm(SalePrice ~ TotalSqftCalc + QualityIndex + TotRmsAbvGrd  + LotArea + garage2 + garage3, data=train.df)
model.40 <- lm(log(SalePrice) ~ TotalSqftCalc + QualityIndex + TotRmsAbvGrd  + LotArea + garage2 + garage3, data=train.df)

summary(model.30)
summary(model.40)

mse.30 <- mean(model.30$residuals^2);
mae.30 <- mean(abs(model.30$residuals));

mse.40 <- mean((train.df$SalePrice-exp(model.40$fitted.values))^2);
mae.40 <- mean(abs(train.df$SalePrice-exp(model.40$fitted.values)));

mse.30
mae.30
mse.40
mae.40


# For the orig model
par(mfrow = c(1, 2), oma = c(0, 0, 2, 0))
qqnorm(model.30$residuals)
qqline(model.30$residuals)

# Make a scatterplot
#plot(train.df$TotalSqftCalc,model.1$residuals)
#title('Residual vs Predictor')
plot(train.df$TotalSqftCalc + train.df$QualityIndex + train.df$TotRmsAbvGrd  + train.df$LotArea
    + train.df$garage2 + train.df$garage3, model.30$residuals,
    ylab = "Resdiuals",
    xlab = "Predictors",
    main = 'Predictors vs. Resdiuals')

#for the log model
par(mfrow = c(1, 2), oma = c(0, 0, 2, 0))
qqnorm(model.40$residuals)
qqline(model.40$residuals)

# Make a scatterplot
#plot(train.df$TotalSqftCalc,model.1$residuals)
#title('Residual vs Predictor')
plot(train.df$TotalSqftCalc + train.df$QualityIndex + train.df$TotRmsAbvGrd  + train.df$LotArea
    + train.df$garage2 + train.df$garage3, model.40$residuals,
    ylab = "Resdiuals",
    xlab = "Predictors",
    main = 'Predictors vs. Resdiuals')


##########################################################################################
# Diagnostic plots using Weisberg's car package
##########################################################################################
# First we need to install the car package
# See Section 1.2.4 p.31 of CAR (Companion to Applied Regression)

# Install package and all other needed packages
#install.packages('car', dependencies=TRUE)

# Load library into your active R session
#library(car)

# Use the function qqPlot() in the car package to assess the Studentized residuals
qqPlot(model.1)

# Note that this is not the typical QQ plot.  Also note that the Studentized residuals
# have a different distribution than the standard residuals.


# Cook's Distance Plot
influenceIndexPlot(model.1,vars=c('Cook','hat'))

# Note that the x index is not correct.  They are the row labels from the original
# data frame.

rownames(train.df) <- seq(1,length(model.1$residuals),1)
model.1 <- lm(SalePrice ~ TotalSqftCalc, data=train.df)
influenceIndexPlot(model.1,vars=c('Cook','hat'))

# If we want the labels correct, then we have to go back to the original data frame
# and fix the row names, then refit the model, and then call the plot.


####################################################################################
#add loess smoother to the scatterplots
####################################################################################

loessMod50 <- loess(model.1$residuals~ train.df$TotalSqftCalc,  span=0.50) # 50% smoothing span

smoothed50 <- predict(loessMod50)

par(mfrow = c(1, 1))
plot(y = model.1$residuals, x = train.df$TotalSqftCalc, type="l"
    , main="Loess Smoothing and Prediction", xlab="Total Square Feet", ylab="Residuals")
lines(smoothed50, train.df$TotalSqftCalc, col="blue")
```

```
#definitely need to fix the loess model


#####################################################################################
#Switch Quality Index model - 5 to log(SalesPrice)
#####################################################################################

# Fit a linear regression model with R
model.5 <- lm(log(SalePrice) ~ QualityIndex, data=train.df)

# Display model summary
summary(model.5)

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.5 <- mean(model.5$residuals^2)
mae.5 <- mean(abs(model.5$residuals))

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.5)

# Use the Base R functon qqplot() to assess the normality of the residuals
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
# Make a scatterplot
plot(train.df$QualityIndex, model.3$residuals,
     ylab = "Resdiuals",
     xlab = "Quality Index",
     main = 'Quality Index vs. Resdiuals')
abline(lm(model.3$residuals ~ train.df$QualityIndex), col = "red")
lines(lowess(train.df$QualityIndex,model.3$residuals),
      col = "blue")

plot(train.df$QualityIndex, model.5$residuals,
     ylab = "Resdiuals",
     xlab = "Quality Index",
     main = 'Natural Log: Quality Index vs. Resdiuals')
abline(lm(model.5$residuals ~ train.df$QualityIndex), col = "red")
lines(lowess(train.df$QualityIndex,model.5$residuals),
      col = "blue")

qqnorm(model.3$residuals)
qqline(model.3$residuals)

qqnorm(model.5$residuals)
qqline(model.5$residuals)




#####################################################################################
#Switch TotalSqftCalc model - 6 to log(SalesPrice)
#####################################################################################

# Fit a linear regression model with R
model.6 <- lm(log(SalePrice) ~ TotalSqftCalc, data=train.df)

# Display model summary
summary(model.6)

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.6 <- mean(model.6$residuals^2)
mae.6 <- mean(abs(model.6$residuals))

# BASE R diagnostic plot for lm object
plot(model.6)
```

```r
# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))

plot(train.df$TotalSqftCalc, model.1$residuals,
    ylab = "Resdiuals",
    xlab = "Total Square Feet",
    main = 'Total Square Feet vs. Resdiuals')
abline(lm(model.1$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.1$residuals),
     col = "blue")

plot(train.df$TotalSqftCalc, model.6$residuals,
    ylab = "Resdiuals",
    xlab = "Total Square Feet",
    main = 'Natural Log: Total Square Feet vs. Resdiuals')
abline(lm(model.6$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.6$residuals),
     col = "blue")

qqnorm(model.1$residuals)
qqline(model.1$residuals)

qqnorm(model.6$residuals)
qqline(model.6$residuals)



##########################################################################################
#Switch multiple regression model - 7 to log(SalesPrice)
##########################################################################################

# Fit a linear regression model with R
model.7 <- lm(log(SalePrice) ~ TotalSqftCalc + QualityIndex, data=train.df)

# Display model summary
summary(model.7)

# Access residuals to compute Mean Square Error (MSE) and Mean Absolute Error (MAE)
mse.7 <- mean(model.7$residuals^2)
mae.7 <- mean(abs(model.7$residuals))

# BASE R diagnostic plot for lm object
plot(model.7)

# Panel the plots
par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))
plot(model.7)


# Make a scatterplot
plot(train.df$TotalSqftCalc,model.7$residuals)
title('Residual vs Predictor 1')

# Make a scatterplot
plot(train.df$QualityIndex,model.7$residuals)
title('Residual vs Predictor 2')

par(mfrow = c(2, 2), oma = c(0, 0, 2, 0))

plot(train.df$TotalSqftCalc, model.1$residuals,
    ylab = "Resdiuals",
    xlab = "Total Square Feet",
    main = 'Total Square Feet vs. Resdiuals')
abline(lm(model.1$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.1$residuals),
     col = "blue")
```

```
plot(train.df$TotalSqftCalc, model.6$residuals,
    ylab = "Resdiuals",
    xlab = "Total Square Feet",
    main = 'Natural Log: Total Square Feet vs. Resdiuals')
abline(lm(model.6$residuals ~ train.df$TotalSqftCalc), col = "red")
lines(lowess(train.df$TotalSqftCalc,model.6$residuals),
     col = "blue")
par(mfrow = c(1, 2), oma = c(0, 0, 2, 0))
qqnorm(model.4$residuals, main = 'Untransformed')
qqline(model.4$residuals)

qqnorm(model.7$residuals, main = 'Transformed')
qqline(model.7$residuals)
```