Real Estate: Optimizing Pricing and Quality

Noah Foilb

DSE1002, Section A, Foitios Kokkotos, Department of Engineering,

Merrimack College, North Andover, Massachusetts

(May 6th, 2021)

Abstract

This report is for people who are realtors/real estate agents or for general use for people who are looking into buying a house. There will be talk of two key components when buying a house. The first being what are the key factors that contribute to the price of the house. By creating a predictive model that can accurately predict the cost of a house based off an overly detailed list of variables we can theoretically calculate the price of a house given a list of details of a house. For a real estate agent this would mean being able to easily find an optimal price to sell a house within a range of prices calculated through a predetermined algorithm and to help to make sure you are not underselling a house. For people planning on buying a property, this model can help predict how much money they should expect to spend to buy a house that fits withing their preferred requirements. The second key component when buying a house, it not only the sales price but also the overall quality of the house. By creating a predictive model, we can predict the overall quality of a house based off a range of variables we can theoretically predict the overall quality of a house. This model will create two classes, one class will consist of the houses that are predicted to have an overall quality of less than 7 and another class of overall quality greater than and equal to 7. For a real estate agent, this would help them scout houses they would want to sell, as going house to house would be a hassle and there is a known correlation between selling houses and the quality of the house. For people planning on buying a property, this model can them find a house that is not only good quality but also cheap as the sales price will be a response variable of the model. The overall accuracy of these models will be discussed below but overall, both target variables will have an accurate model created for both, hence this will be a report will mark our successes.

# Background

Throughout American history, capitalism has been widely known and has accepted as the mindset of this nation. In modern day, people try to find the cheapest price for the newest products and companies try to sell these products for the highest price. Yet when it comes to buying a new house there is not standard price. Most prices can be talked down and two similar houses may be priced completely differently. The significance of an overly detailed backyard or garage may not matter to the buyer of a house, but the price and

overall quality of a house will always matter not only to the buyer, but the seller as well. I have created a few models that can help both audiences. This report will not only be for the company I am writing this report for but also for the public.

# Executive Summary

One of the first things that is needed to be clarified is the target variables and the target audiences as these models can have multiple purposes.

The first model I have created predicts the price of a house when given a list of many detailed response variables. This list ranges from the type of neighborhood they live in, to the square footing of the second floors bathroom. If modeled accurately there are several applications that such a model can be used for. The first is for real estate agents or realtor. Being able to put certain details about a house into a program to accurately find out how much a house is worth is a great way to easily find an accurate sales price for such a house. Not only that but also the agents may try to sell the house at a higher price to decrease it when negotiating a price with a customer.

This model can also be used by a customer who is trying to accurately find the price of a theoretical house when giving their preferred details of a house. For example, imagine there is a website that will ask for “How big do you want your driveway to be”, “Do you want a pool”, “How many bedrooms” and will take what you said and create a range of prices that your ideal house will be worth. Practically, this can help people figure out how much money they need to save to purchase their dream house. Another use buyer could use this website for would be to make sure the house they are currently looking into buying is within reason as no one wants to overpay for a house.

In terms of the model itself, the best fit model that will accurately predict the sales price of a house is a random forest model, as when compared to the other models, this model has the lowest RMSE value. I used RMSE to compare the models instead of R-Squared since R-Squared is a relive measure of fit, while RMSE is an absolute measure of fit. Since the random forest model not only explained 91.9% of the variance it also had the lowest RMSE value making it the best model to predict the sales price of the house.

The second model predicts if a house will have an overall quality of 7/10 or greater. As there is a known correlation between the quality of a house and how fast such a house gets sold there are many practical uses for such a model in the real estate business. Not only can this be used to potentially predict how fast a house can be sold based off the overall quality, but it can also be used to scout houses that people may be interested in selling. Since companies will obviously want to sell houses fast and for more, selling houses of higher overall quality makes sense So being able to optimize the search in finding houses that a company wants to buy seems like a very helpful tool to have for a company.

This model can yet again also be used by the public for different uses. This model can theoretically be on a website that is publicly accessible. The first use can be for rehabbers (People who buy and flip houses). Rehabbers can use this algorithm to be able to act like a real estate agent and be able to buy a house that has a potentially higher overall quality then it currently has and can sell it fast and for more money. The second use can yet again be for people who are trying to find out what quality their dream house will be. This is like the other model for people looking for their dream house except this model can help them estimate how hard it is to find such a house as the higher quality that is predicted the harder it will be to find since higher quality houses get sold more.

In terms of the model itself the best fitting model there was pretty similar model between the logistic regression, decision Tree, and the random forest but I chose the random forest for multiple reasons. The first reason being since all these model’s accuracy are within 3% of each other choosing was mostly a matter of choice and deduction. Lastly, as the model with the highest accuracy, tit does not get affected by human error when compared to the logistic regression as I must choose the predictors myself.

# Data & Approach

Before starting on the code, I needed to look over my data (Ames) and make sure there every one of the 50 attributes were usable and would not create an error later. My first impressions while looking at the data was anxiety as I was overwhelmed by each new variable. While trying to make sense of the data I found a .txt file attached to the folder I downloaded from Kaggle that explained each variable and what they do. For the first few weeks all I did was get myself comfortable with the variable names and meanings.

To create a realistic dataset, I had to first investigate the Sale.Condition, which specified if anything special had happened around the sale or if the house was unique for any reason. I had filtered the dataset by Sale.Condition = “Normal” to make sure the dataset was for normal circumstances.

The next big realization I made was realizing I needed to change over half of the attributes as they use NA as a descriptor rather than to signify there was a missing value. So, I had to change all these attributes to get them to run properly in my models: LotFrontage, Alley, MasVnrArea, Garage. Yr.Blt, Fireplace.Qu, Garage.Type, Garage.Finish, Garage.Qual, Garage.Cond, Bsmt.Qual, Bsmt.Cond, Bsmt.Exposure, BsmtFin.Type.1, BsmtFin.Type.2, Pool.QC, Fence and Misc.Feature.

There were also five attributes I ended up taking out of the dataset. Each one I got rid of for one reason out of two. The attribute did not make sense, or they were formatted weird and would create errors in my models. The first two attributes I got rid of were PID and Order. These attributes did not have to do with the data but was rather a number used to make each row unique. The other three variables were Utilities, Roof.Matl and Bsmt.Cond. These attributes would create errors that were not logical and would require a lot of cleaning. I also found these not useful when looking at their correlations and using logic.

To give a brief description of what I will be talking about the detailed findings section I will be talking about my overall approach. First, I will look at my clean data and make logical conclusions about which columns are more important than the others.

Then I will create my null model for SalePrice as this will be my first target variable that I will addressing. Next, I will sort through the correlations and create a multiple regression model. After analyzing the accuracy of this model, I will edit this model and adjust what predictors I want to use. After this I will analyze the accuracy and create a completely new model, the regression tree. After creating this model over the entire dataset, I will analysis it and move on to my fourth and final model for SalePrice, the random forest. As mentioned before this model will perform the best in both target variables. After creating the model, I will test and find its accuracy. After the creation of these models, I will compare each against each other to see which is the best model. After the completion and analysis of these models I will be comparing each and determining which model is the best to predict SalePrice.

This leads to the final half of my code, where I start to create new models to predict if t a house if going to have an overall quality 7 or greater. I will first create a new binary column that will explain if the overall quality is 7 or greater. Then after that I will create my null model, which will be a regulator to help me understand if my future models are better then worse then guessing. After analyzing I will create my new model, the logistic regression model. After analyzing I will adjust and create a new logistic regression model. After a final analysis I will move on to my next model, the decision tree. After analyzing again, I will create my fourth and final model, the random forest. After analyzing this I compare them and decide which model is the best to predict the overall quality which is greater than or equal to 7.

# Detailed Findings

Assuming by now that our libraries are loaded in and our data has been cleaned and re-engineered, we can start the process of creating our models but with extreme detail.

# Dataset Analysis

The first thing I did when looking at the dataset is I used the summary command to find out details about each attribute and how many NAs are in each. After getting comfortable with the data, I realized one key concept between them all. There were a lot of connections between certain attributes. A lot of categorical “Garage” attributes could be replaced with numerical “Garage” attributes. For example, Garage.Qual could be used instead of Garage.Type as they have high correlation. After making these connections and changing the data I set out to look at the data by looking at the correlations. I found that a lot of numerical predictors had a high correlation with SalePrice. Most had some sort of significant correlation (greater than .5). After analyzing the correlations matrix, I created my test and train datasets using a .75/.25 split. After the train test split, I talked about how SalePrice is the target variable as this is equivalent to the price of the house.

# Null Model for SalePrice

After the initial data analysis, I created my null model on my training data by using the lm command and displaying my model which can be found in Table 1. As most of this information is not useful, I will not talk about the values of such model since this is supposed to be an inaccurate model which the only purpose of it is to base my other models off it.

Table 1: This is the displayed intercept, standard error, T value and P value.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Estimate | Std. Error | T value | Pr(>|t|) |
| Intercept | 174702 | 1670 | 104.6 | 2e-16 |

After looking at the null model I calculated the RMSE, which I will be using to analyze my data throughout the models as it is an absolute measurement of fit. I got a RMSE of 1,002,400$ which is a horrible value for a RMSE in terms of money. As expected, this model is not great yet to visualize how bad this model is I created Graph 1 to not only show the data but also show the null model.

Graph 1: This puts the SalePrice per 1k against the Overall.Qual of the house. Red line represents null model. Chart, scatter chart

Description automatically generated

As we can see null model will predict every value as the “Estimate” as seen in Table 1, which is extremely inaccurate.

# Multiple Regression (SP)

As I began to continue my model making, one of the most manually draining models is the multiple regression model. The first task at hand for creating the multiple regression model is selecting the predictors I want to use in my model. In order to do this I first selected the top correlations that made sense to put into my model. The first few being Overall.Qual and Overall.Cond as they made the most sense to put in. The next few, I put in were also high correlations, such as Gr.Liv.Area and Full.bath which makes sense as the price of a house should go up as there are bathrooms with bathtubs in them or the garage area is expanded. After that I messed around with each attribute and if they made sense to add and they added value to my model I would add them. A detailed list of all my variables can be found in Table 2, as we can see how they preformed in my model.

Table 2: As the predictors are put into the lm command, we can see how well the model does in this table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Estimate | Std. Error | T value | Pr(>|t|) |
| Intercept | -1.26e+6 | 8.3e+4 | -15 | 2e-16 |
| Overall.Qual | 1.49e+4 | 7.7e+2 | 19 | 2e-16 |
| Overall.Cond | 5.45e+3 | 6.8e+2 | 8 | 2e-15 |
| Year.Built | 3.78e+2 | 3.5e+1 | 11 | 2e-16 |
| Gr.Liv.Area | 6.29e+1 | 2.1e+0 | 30 | 2e-16 |
| Total.Bsmt.SF | 4.07e+1 | 1.9e+0 | 21 | 2e-16 |
| Full.Bath | -1.25e+4 | 1.8e+3 | -7 | 2e-12 |
| Year.Remod.Add | 2.22e+2 | 4.4e+1 | 5 | 5e-7 |
| Mas.Vnr.Area | 3.84e+1 | 4.2e+0 | 8 | 2e-15 |
| Garage.Cars | 9.52e+3 | 1.2e+3 | 8 | 2e-15 |

Going through Table 2, we do not find anything not statistically significant. The Std. Error, T value and Pr all look great! In order to assure that these predictors are good as well I used the confint command to see the 95% intervals to make sure they are statistically significant, and they are! This means this is a model, but we need to make sure it is an accurate model.

Using the same form of measurement as the null model I calculated the RMSE and found the RMSE value to be 415,000$ which is a 587,400 $ difference which is much better than the null model. Not only that but this model has a Adj R-Squared of 85.22%. This is a good model. Let us look at another graph, Graph 2, using a different predictor on the x axis to show this model.

Graph 2: Having the Gr.Liv.Area on the x axis and SalePrice on y axis shows their high correlation.

Chart, scatter chart

Description automatically generated

As we can see in Graph 2, there is a high correlation between SalePrice and Gr.Liv.Area and with a model with a RMSE of 415,000$ this is currently our best model yet.

# Regression Tree (SP)

The next model I used was a regression tree model. For this model I let the algorithm calculate which predictors it wanted to use for its model as this gets rid of the human factor. I used my target variable in the command rpart on my training dataset. This gave me a sophisticated looking decision tree as we can see in Graph 3. This decision tree uses the attributes: Overall.Qual, Gr.Liv.Area, Neighborhood, Total.Bsmt and BsmtFin.SF.1.

Graph 3: This is the visualized decision tree we found by using the command rpart.plot on the model.

Diagram

Description automatically generated

One interesting observation I noticed is that for lower priced houses the difference between 156K and 249K greatly depend on the garage living area and the most expensive houses depend on the Basement statistics.

After analyzing Graph 3, I calculated the RMSE to see how well it did, compared to the other graphs. This model produced an RMSE of 531,400$ which is worse than the Multiple regression model by 116,400$

# Random Forest (SP)

My final model that I created is a random forest model. This model was the best model I had created. The first thing I did was the same as the regression tree and I let the computer decide what attributes it wants to select. I then ran my randomForest command using a ntree of 400 and a mtry of 20. This took a while, but it produced a great model. This model explained 91.9% of the variance and when I calculated the RMSE it came up to a 322,200$ which is 92,800$ better than my current best model, the multiple regression model.

# SalePrice Models

When comparing the models to each other I found that the best model I created was the random forest model which was better then all the rest based off the RMSE. Hence my final model will be the random forest model.

# Null Model for Overall.Qual

To create a model that will predict if a house has an overall quality over 6, I needed to create a new coluum that will transform the Overall.Qual into a binary categorical column that can used. I did this and I called it Overall.Binary. One thing to note is that I had to make sure to not involve Overall.Qual in any of my models or there would be a 100% accuracy.

The first step I took to creating my null model was to create the new column called Overall.Binary. My next step was to use the glm command and create my null model. As expected, there was a null Deviance of 2300 (Max) and my next step was to analyze the confusion matrix. null model would give an accuracy of 66.8% if it guesses that the overall quality was lower then 7. This is a standard null as we can see in Graph 4.

Graph 4: Having Overall.Qual on the x axis and SalePrice on y axis shows their high correlation and everything to the left of the red line is the null model.

Chart, scatter chart

Description automatically generated

As we can see the null model acts exactly as we would have expected it to and gives us an insignificant model with an accuracy of 66.8%.

# Logistic Regression (OB)

My next model I created was a logistic regression model. My first few steps were very similar to that of my multiple regression model, and I started to compare and test different predictors that would decrease the residual deviance in the model. I used a glm command to create my model and I used much less predictors this time as there were not many variables that effected Overall.Binary. You can see the list of my response variables in Table 3, where it describes how each variable did in the model.

Table 3: As the predictors are put into the glm command, we can see how well the model does in this table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Estimate | Std. Error | Z value | Pr(>|t|) |
| Intercept | -8.05e+0 | 4.3e-1 | -19 | 2e-16 |
| SalePrice | 5.12e-5 | 3.2e-6 | 16 | 2e-16 |
| Gr.Liv.Area | -7.89e-4 | 2.8e+2 | -3 | 4e-3 |
| Garage.Cars | 3.74e-1 | 1.7e+1 | 2 | 2e-2 |
| Lot.Area | -1.36e-4 | 2.0e+0 | -7 | 2e-11 |

This model shows that it is statistically significant but given that there were not many variables I could add and the garage attributes I used were barely significant due to their Pr value this model has some questionable predictive powers.

My next step was to test that by finding the confusion matrix and calculating the accuracy. The accuracy calculated from this when choosing the optimal threshold of .41, it is 87.12%, which is not bad! This model still worked when used on the test set, so it has proven itself to be a valid model. We can see some visual validity in Graph 5.

Graph 5: Having Overall.Binary on the y axis and SalePrice on x axis shows their high correlation and shows that there is a logistic connection between them.

Chart, scatter chart

Description automatically generated

As we can see from the graph there is a logistic connection between them. Not only yet but this model isn’t bad, but I feel as if it has not reached it potential at this point as the computer can see things I visually cannot.

# Decision Tree (OB)

My next model I created was decision tree that was calculated automatically using its own algorithms. One big error I can into was when I created Overall.Binary, I never switched it to type class, which gave me an error later. So, I had to make sure to make the column to type class before putting it into the decision tree. The decision tree model selected its own response variables of Neighborhood, Exter.Qual and SalePrice which we can see in Graph 6.

Graph 6: The stems in order from highest to lowest are Exter.Qual, SalePrice and them Neighborhood.

Diagram, timeline

Description automatically generated

After analyzing the tree, I found the confusion matrix and calculated the accuracy, which was 86.4% While this model has a lower accuracy then the logistic regression model. I would select this as my model since this is more statistically significant.

# Random Forest (OB)

My final model I created was a random forest, which as I mentioned before was my best model. I let the randomForest command select what predictors it wanted, and I used 400 number of trees and 20 variables tried at each split. This not only gave me a OOB estimate of error rate of 10.17% but also once I calculated the confusion matrix, gave me an accuracy of around 90%. This is not only the most accurate, but the most statistically significant model I have when trying to predict the Overall.Binary.

# Overall.Binary Models

When comparing the accuracies and the overall statistically significant of each of my models. I would select the random forest model as not only was it the most accurate but the most statistically significant.

# Validity & Reliability

When discussing the validity and reliability of my models and the potentially future assessments of my model I need to consider what I already know. When looking at my SalePrice models I see that I have currently looked at not only what makes it statistically significant (t value, Pr, St. Error, 95% confidence interval and R-Squared) but also, I compared the RMSE which is an absolute measure of fit. In future assessments, maybe I could try cross validating or try other forms of accuracy, but I do not know how to do that yet on RStudio.

For my Overall.Binary Models, one big change I would do if dig deeper into the response variable selection for my Logisitic regression model to get a more statistically accurate model. I would also investigate potentially customizing the target variable to get a more accurate model. For example, instead of greater then or equal to 7, maybe choose 8 which may also depend on what the client wants. Lastly, I could also use the confusion matrix more wisely to not only get a more optimal threshold but also use f-score, recall and precision.

Final Project

# Load in libraries

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.0.3

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##

Final Project

# Load in libraries

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.0.3

##   
## Attaching package: 'dplyr'

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

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

library(mdsr)

## Warning: package 'mdsr' was built under R version 4.0.3

library(NHANES)

## Warning: package 'NHANES' was built under R version 4.0.4

library(broom)

## Warning: package 'broom' was built under R version 4.0.3

library(mosaicData)  
library(mosaic)

## Warning: package 'mosaic' was built under R version 4.0.3

## Registered S3 method overwritten by 'mosaic':  
## method from   
## fortify.SpatialPolygonsDataFrame ggplot2

##   
## The 'mosaic' package masks several functions from core packages in order to add   
## additional features. The original behavior of these functions should not be affected by this.

##   
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':  
##   
## mean

## The following object is masked from 'package:ggplot2':  
##   
## stat

## The following objects are masked from 'package:dplyr':  
##   
## count, do, tally

## The following objects are masked from 'package:stats':  
##   
## binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,  
## quantile, sd, t.test, var

## The following objects are masked from 'package:base':  
##   
## max, mean, min, prod, range, sample, sum

library(mdsr)  
library(ggplot2)  
library(randomForest)

## Warning: package 'randomForest' was built under R version 4.0.4

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

## The following object is masked from 'package:dplyr':  
##   
## combine

library(mosaicCore)

## Warning: package 'mosaicCore' was built under R version 4.0.3

##   
## Attaching package: 'mosaicCore'

## The following objects are masked from 'package:dplyr':  
##   
## count, tally

library(readxl)  
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.0.4

## Loading required package: rpart

library(rpart)  
library(nnet)  
library(NeuralNetTools)

## Warning: package 'NeuralNetTools' was built under R version 4.0.5

library(class)  
library(e1071)

## Warning: package 'e1071' was built under R version 4.0.5

library(neuralnet)

## Warning: package 'neuralnet' was built under R version 4.0.5

##   
## Attaching package: 'neuralnet'

## The following object is masked from 'package:dplyr':  
##   
## compute

# Load in dataset

Ames <- read.csv("Ames.csv",quote = "")

# Lets look at the Na values

summary(Ames)

## Order PID MS.SubClass MS.Zoning   
## Min. : 1.0 Min. :5.263e+08 Min. : 20.00 Length:2930   
## 1st Qu.: 733.2 1st Qu.:5.285e+08 1st Qu.: 20.00 Class :character   
## Median :1465.5 Median :5.355e+08 Median : 50.00 Mode :character   
## Mean :1465.5 Mean :7.145e+08 Mean : 57.39   
## 3rd Qu.:2197.8 3rd Qu.:9.072e+08 3rd Qu.: 70.00   
## Max. :2930.0 Max. :1.007e+09 Max. :190.00   
##   
## Lot.Frontage Lot.Area Street Alley   
## Min. : 21.00 Min. : 1300 Length:2930 Length:2930   
## 1st Qu.: 58.00 1st Qu.: 7440 Class :character Class :character   
## Median : 68.00 Median : 9436 Mode :character Mode :character   
## Mean : 69.22 Mean : 10148   
## 3rd Qu.: 80.00 3rd Qu.: 11555   
## Max. :313.00 Max. :215245   
## NA's :490   
## Lot.Shape Land.Contour Utilities Lot.Config   
## Length:2930 Length:2930 Length:2930 Length:2930   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## Land.Slope Neighborhood Condition.1 Condition.2   
## Length:2930 Length:2930 Length:2930 Length:2930   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## Bldg.Type House.Style Overall.Qual Overall.Cond   
## Length:2930 Length:2930 Min. : 1.000 Min. :1.000   
## Class :character Class :character 1st Qu.: 5.000 1st Qu.:5.000   
## Mode :character Mode :character Median : 6.000 Median :5.000   
## Mean : 6.095 Mean :5.563   
## 3rd Qu.: 7.000 3rd Qu.:6.000   
## Max. :10.000 Max. :9.000   
##   
## Year.Built Year.Remod.Add Roof.Style Roof.Matl   
## Min. :1872 Min. :1950 Length:2930 Length:2930   
## 1st Qu.:1954 1st Qu.:1965 Class :character Class :character   
## Median :1973 Median :1993 Mode :character Mode :character   
## Mean :1971 Mean :1984   
## 3rd Qu.:2001 3rd Qu.:2004   
## Max. :2010 Max. :2010   
##   
## Exterior.1st Exterior.2nd Mas.Vnr.Type Mas.Vnr.Area   
## Length:2930 Length:2930 Length:2930 Min. : 0.0   
## Class :character Class :character Class :character 1st Qu.: 0.0   
## Mode :character Mode :character Mode :character Median : 0.0   
## Mean : 101.9   
## 3rd Qu.: 164.0   
## Max. :1600.0   
## NA's :23   
## Exter.Qual Exter.Cond Foundation Bsmt.Qual   
## Length:2930 Length:2930 Length:2930 Length:2930   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## Bsmt.Cond Bsmt.Exposure BsmtFin.Type.1 BsmtFin.SF.1   
## Length:2930 Length:2930 Length:2930 Min. : 0.0   
## Class :character Class :character Class :character 1st Qu.: 0.0   
## Mode :character Mode :character Mode :character Median : 370.0   
## Mean : 442.6   
## 3rd Qu.: 734.0   
## Max. :5644.0   
## NA's :1   
## BsmtFin.Type.2 BsmtFin.SF.2 Bsmt.Unf.SF Total.Bsmt.SF   
## Length:2930 Min. : 0.00 Min. : 0.0 Min. : 0   
## Class :character 1st Qu.: 0.00 1st Qu.: 219.0 1st Qu.: 793   
## Mode :character Median : 0.00 Median : 466.0 Median : 990   
## Mean : 49.72 Mean : 559.3 Mean :1052   
## 3rd Qu.: 0.00 3rd Qu.: 802.0 3rd Qu.:1302   
## Max. :1526.00 Max. :2336.0 Max. :6110   
## NA's :1 NA's :1 NA's :1   
## Heating Heating.QC Central.Air Electrical   
## Length:2930 Length:2930 Length:2930 Length:2930   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## X1st.Flr.SF X2nd.Flr.SF Low.Qual.Fin.SF Gr.Liv.Area   
## Min. : 334.0 Min. : 0.0 Min. : 0.000 Min. : 334   
## 1st Qu.: 876.2 1st Qu.: 0.0 1st Qu.: 0.000 1st Qu.:1126   
## Median :1084.0 Median : 0.0 Median : 0.000 Median :1442   
## Mean :1159.6 Mean : 335.5 Mean : 4.677 Mean :1500   
## 3rd Qu.:1384.0 3rd Qu.: 703.8 3rd Qu.: 0.000 3rd Qu.:1743   
## Max. :5095.0 Max. :2065.0 Max. :1064.000 Max. :5642   
##   
## Bsmt.Full.Bath Bsmt.Half.Bath Full.Bath Half.Bath   
## Min. :0.0000 Min. :0.00000 Min. :0.000 Min. :0.0000   
## 1st Qu.:0.0000 1st Qu.:0.00000 1st Qu.:1.000 1st Qu.:0.0000   
## Median :0.0000 Median :0.00000 Median :2.000 Median :0.0000   
## Mean :0.4314 Mean :0.06113 Mean :1.567 Mean :0.3795   
## 3rd Qu.:1.0000 3rd Qu.:0.00000 3rd Qu.:2.000 3rd Qu.:1.0000   
## Max. :3.0000 Max. :2.00000 Max. :4.000 Max. :2.0000   
## NA's :2 NA's :2   
## Bedroom.AbvGr Kitchen.AbvGr Kitchen.Qual TotRms.AbvGrd   
## Min. :0.000 Min. :0.000 Length:2930 Min. : 2.000   
## 1st Qu.:2.000 1st Qu.:1.000 Class :character 1st Qu.: 5.000   
## Median :3.000 Median :1.000 Mode :character Median : 6.000   
## Mean :2.854 Mean :1.044 Mean : 6.443   
## 3rd Qu.:3.000 3rd Qu.:1.000 3rd Qu.: 7.000   
## Max. :8.000 Max. :3.000 Max. :15.000   
##   
## Functional Fireplaces Fireplace.Qu Garage.Type   
## Length:2930 Min. :0.0000 Length:2930 Length:2930   
## Class :character 1st Qu.:0.0000 Class :character Class :character   
## Mode :character Median :1.0000 Mode :character Mode :character   
## Mean :0.5993   
## 3rd Qu.:1.0000   
## Max. :4.0000   
##   
## Garage.Yr.Blt Garage.Finish Garage.Cars Garage.Area   
## Min. :1895 Length:2930 Min. :0.000 Min. : 0.0   
## 1st Qu.:1960 Class :character 1st Qu.:1.000 1st Qu.: 320.0   
## Median :1979 Mode :character Median :2.000 Median : 480.0   
## Mean :1978 Mean :1.767 Mean : 472.8   
## 3rd Qu.:2002 3rd Qu.:2.000 3rd Qu.: 576.0   
## Max. :2207 Max. :5.000 Max. :1488.0   
## NA's :159 NA's :1 NA's :1   
## Garage.Qual Garage.Cond Paved.Drive Wood.Deck.SF   
## Length:2930 Length:2930 Length:2930 Min. : 0.00   
## Class :character Class :character Class :character 1st Qu.: 0.00   
## Mode :character Mode :character Mode :character Median : 0.00   
## Mean : 93.75   
## 3rd Qu.: 168.00   
## Max. :1424.00   
##   
## Open.Porch.SF Enclosed.Porch X3Ssn.Porch Screen.Porch  
## Min. : 0.00 Min. : 0.00 Min. : 0.000 Min. : 0   
## 1st Qu.: 0.00 1st Qu.: 0.00 1st Qu.: 0.000 1st Qu.: 0   
## Median : 27.00 Median : 0.00 Median : 0.000 Median : 0   
## Mean : 47.53 Mean : 23.01 Mean : 2.592 Mean : 16   
## 3rd Qu.: 70.00 3rd Qu.: 0.00 3rd Qu.: 0.000 3rd Qu.: 0   
## Max. :742.00 Max. :1012.00 Max. :508.000 Max. :576   
##   
## Pool.Area Pool.QC Fence Misc.Feature   
## Min. : 0.000 Length:2930 Length:2930 Length:2930   
## 1st Qu.: 0.000 Class :character Class :character Class :character   
## Median : 0.000 Mode :character Mode :character Mode :character   
## Mean : 2.243   
## 3rd Qu.: 0.000   
## Max. :800.000   
##   
## Misc.Val Mo.Sold Yr.Sold Sale.Type   
## Min. : 0.00 Min. : 1.000 Min. :2006 Length:2930   
## 1st Qu.: 0.00 1st Qu.: 4.000 1st Qu.:2007 Class :character   
## Median : 0.00 Median : 6.000 Median :2008 Mode :character   
## Mean : 50.63 Mean : 6.216 Mean :2008   
## 3rd Qu.: 0.00 3rd Qu.: 8.000 3rd Qu.:2009   
## Max. :17000.00 Max. :12.000 Max. :2010   
##   
## Sale.Condition SalePrice   
## Length:2930 Min. : 12789   
## Class :character 1st Qu.:129500   
## Mode :character Median :160000   
## Mean :180796   
## 3rd Qu.:213500   
## Max. :755000   
##

# Lets look at the dataset column by column

Ames <- read.csv("Ames.csv",quote = "")  
  
# PID & Order   
Ames = subset(Ames, select = -c(PID,Order,Utilities, Roof.Matl,Bsmt.Cond)) # Get rid of these as these will mess up the models  
  
# LotFrontage   
Ames[c("Lot.Frontage")][is.na(Ames[c("Lot.Frontage")])] <- 0  
  
# Alley  
Ames[c("Alley")][is.na(Ames[c("Alley")])] <- "No Alley"  
  
# MasVnrArea  
Ames[c("Mas.Vnr.Area")][is.na(Ames[c("Mas.Vnr.Area")])] <- 0  
  
# Garage.Yr.Blt  
Ames[c("Garage.Yr.Blt")][is.na(Ames[c("Garage.Yr.Blt")])] <- 1978  
  
# Fireplace.Qu  
Ames[c("Fireplace.Qu")][is.na(Ames[c("Fireplace.Qu")])] <- "No Fireplace"  
  
# Garage.Type  
Ames[c("Garage.Type")][is.na(Ames[c("Garage.Type")])] <- "No Garage"  
  
# Garage.Finish  
Ames[c("Garage.Finish")][is.na(Ames[c("Garage.Finish")])] <- "No Garage"  
  
# Garage.Quality  
Ames[c("Garage.Qual")][is.na(Ames[c("Garage.Qual")])] <- "No Garage"  
  
# Garage.Cond  
Ames[c("Garage.Cond")][is.na(Ames[c("Garage.Cond")])] <- "No Garage"  
  
# Bsmt.Qual  
Ames[c("Bsmt.Qual")][is.na(Ames[c("Bsmt.Qual")])] <- "No Basement"  
  
# Bsmt.Cond  
#Ames[c("Bsmt.Cond")][is.na(Ames[c("Bsmt.Cond")])] <- "No Basement"  
  
# Bsmt.Exposure  
Ames[c("Bsmt.Exposure")][is.na(Ames[c("Bsmt.Exposure")])] <- "No Basement"  
  
# BsmtFin.Type.1  
Ames[c("BsmtFin.Type.1")][is.na(Ames[c("BsmtFin.Type.1")])] <- "No Basement"  
  
# BsmtFin.Type.2  
Ames[c("BsmtFin.Type.2")][is.na(Ames[c("BsmtFin.Type.2")])] <- "No Basement"  
  
# Pool.QC  
Ames[c("Pool.QC")][is.na(Ames[c("Pool.QC")])] <- "No Pool"  
  
# Fence  
Ames[c("Fence")][is.na(Ames[c("Fence")])] <- "No Fence"  
  
# Misc.Feature  
Ames[c("Misc.Feature")][is.na(Ames[c("Misc.Feature")])] <- "None"  
  
  
# We also only want where the sales conidtion is normal so  
Ames <- Ames %>%  
 filter(Sale.Condition == "Normal") %>%  
 na.omit()  
  
# Fix MS.Zoning (This Became an issue later on)  
Ames[Ames$MS.Zoning == "C (all)","MS.Zoning"] <- "Other"  
Ames[Ames$MS.Zoning == "A (agr)","MS.Zoning"] <- "Other"  
Ames[Ames$MS.Zoning == "I (all)","MS.Zoning"] <- "Other"

# Correlation

Ames\_Quan <- Ames %>%  
 select(SalePrice,MS.SubClass,Lot.Area,Lot.Frontage,Overall.Qual,Overall.Cond, Year.Built, Year.Remod.Add,Mas.Vnr.Area,BsmtFin.SF.1,BsmtFin.SF.2, Bsmt.Unf.SF, Total.Bsmt.SF, X1st.Flr.SF, X2nd.Flr.SF,Low.Qual.Fin.SF,Gr.Liv.Area, Bsmt.Full.Bath,Bsmt.Half.Bath,Full.Bath,Half.Bath,Bedroom.AbvGr,Kitchen.AbvGr,TotRms.AbvGrd,Fireplaces,Garage.Yr.Blt,Garage.Cars,Garage.Area,Wood.Deck.SF,Open.Porch.SF,Enclosed.Porch,X3Ssn.Porch,Screen.Porch,Pool.Area,Misc.Val,Mo.Sold,Yr.Sold)  
# Can only select numerical attributes  
  
cor(Ames\_Quan)

## SalePrice MS.SubClass Lot.Area Lot.Frontage  
## SalePrice 1.000000000 -0.075193072 0.273481329 0.159325799  
## MS.SubClass -0.075193072 1.000000000 -0.190739915 -0.229194750  
## Lot.Area 0.273481329 -0.190739915 1.000000000 0.108546243  
## Lot.Frontage 0.159325799 -0.229194750 0.108546243 1.000000000  
## Overall.Qual 0.788352219 0.067745183 0.069236930 0.071523019  
## Overall.Cond -0.092137386 -0.080243471 -0.029526362 -0.006199587  
## Year.Built 0.533658797 0.062017355 0.006812163 -0.032309029  
## Year.Remod.Add 0.501693385 0.065884489 0.006884182 0.011970154  
## Mas.Vnr.Area 0.487244127 0.017190459 0.088844075 0.086880057  
## BsmtFin.SF.1 0.455086938 -0.077447320 0.163026112 0.015038191  
## BsmtFin.SF.2 0.038563700 -0.076408020 0.082713478 0.019282656  
## Bsmt.Unf.SF 0.160663128 -0.107771250 0.012352745 0.115521387  
## Total.Bsmt.SF 0.645102431 -0.222272902 0.214818515 0.141735132  
## X1st.Flr.SF 0.637450315 -0.263961808 0.296307656 0.192465704  
## X2nd.Flr.SF 0.294032181 0.310380332 0.016612054 -0.019400105  
## Low.Qual.Fin.SF -0.025741914 0.014419970 0.002820987 0.008942490  
## Gr.Liv.Area 0.736905671 0.075600122 0.238234264 0.128834365  
## Bsmt.Full.Bath 0.287816662 -0.007428545 0.121155914 0.006294133  
## Bsmt.Half.Bath -0.028877366 -0.018439993 0.025743928 -0.027729936  
## Full.Bath 0.548011061 0.146292219 0.114791873 0.044755524  
## Half.Bath 0.291755002 0.188592646 0.008279069 -0.045451132  
## Bedroom.AbvGr 0.182304422 -0.035440534 0.134008640 0.105382432  
## Kitchen.AbvGr -0.114322853 0.253611143 -0.013171006 0.020707396  
## TotRms.AbvGrd 0.495767243 0.032388727 0.190566771 0.161691107  
## Fireplaces 0.492723739 -0.043748639 0.235933607 0.027491216  
## Garage.Yr.Blt 0.489510079 0.115114813 -0.021330618 -0.024470363  
## Garage.Cars 0.632522561 -0.022537621 0.167336175 0.101149002  
## Garage.Area 0.621774058 -0.079025419 0.185897056 0.133348873  
## Wood.Deck.SF 0.350808179 -0.015432265 0.157493379 -0.002685556  
## Open.Porch.SF 0.325160475 -0.004452985 0.068591855 0.048942761  
## Enclosed.Porch -0.117817356 -0.027954934 0.014641504 0.030442830  
## X3Ssn.Porch 0.016129935 -0.044666145 0.019737514 -0.002368941  
## Screen.Porch 0.116879613 -0.053171712 0.041869617 0.059727938  
## Pool.Area 0.035761745 -0.014258419 0.054546719 0.077033482  
## Misc.Val -0.016624770 -0.029768463 0.042165310 -0.019623629  
## Mo.Sold 0.001283849 0.010976013 0.005762339 -0.003060657  
## Yr.Sold 0.022578942 -0.029088128 -0.011226021 0.018729452  
## Overall.Qual Overall.Cond Year.Built Year.Remod.Add  
## SalePrice 0.788352219 -0.092137386 0.533658797 0.501693385  
## MS.SubClass 0.067745183 -0.080243471 0.062017355 0.065884489  
## Lot.Area 0.069236930 -0.029526362 0.006812163 0.006884182  
## Lot.Frontage 0.071523019 -0.006199587 -0.032309029 0.011970154  
## Overall.Qual 1.000000000 -0.088112677 0.558435511 0.529169932  
## Overall.Cond -0.088112677 1.000000000 -0.401395455 0.077346551  
## Year.Built 0.558435511 -0.401395455 1.000000000 0.553887600  
## Year.Remod.Add 0.529169932 0.077346551 0.553887600 1.000000000  
## Mas.Vnr.Area 0.379377224 -0.135439069 0.297005289 0.158448152  
## BsmtFin.SF.1 0.263950808 -0.061300935 0.306064960 0.146021889  
## BsmtFin.SF.2 -0.017205234 0.024654278 0.001989174 -0.036164817  
## Bsmt.Unf.SF 0.242691621 -0.119397500 0.069803505 0.112035529  
## Total.Bsmt.SF 0.509732178 -0.173815977 0.384386244 0.247719254  
## X1st.Flr.SF 0.436880787 -0.154232101 0.278008090 0.193082649  
## X2nd.Flr.SF 0.258312556 0.015614966 0.019094009 0.177087419  
## Low.Qual.Fin.SF -0.043486779 0.020345943 -0.128629147 -0.057067553  
## Gr.Liv.Area 0.552619095 -0.100605447 0.214320902 0.296081147  
## Bsmt.Full.Bath 0.168643883 -0.063770129 0.232123260 0.145086263  
## Bsmt.Half.Bath -0.045104005 0.093256435 -0.026483551 -0.040948720  
## Full.Bath 0.503658232 -0.213571687 0.442483649 0.423697506  
## Half.Bath 0.270562395 -0.093795062 0.274667875 0.212188788  
## Bedroom.AbvGr 0.081799331 -0.004509344 -0.042122633 -0.005398359  
## Kitchen.AbvGr -0.152859765 -0.084198317 -0.138032549 -0.147691792  
## TotRms.AbvGrd 0.355726368 -0.071061096 0.084647400 0.179021515  
## Fireplaces 0.388839953 -0.042978107 0.160137439 0.116390835  
## Garage.Yr.Blt 0.513300491 -0.316763364 0.781832306 0.586352681  
## Garage.Cars 0.566769047 -0.191470982 0.507354096 0.376293818  
## Garage.Area 0.519950782 -0.158458551 0.449250508 0.325463008  
## Wood.Deck.SF 0.254998275 0.013213588 0.230611484 0.221776189  
## Open.Porch.SF 0.292679518 -0.064886176 0.181044904 0.230302751  
## Enclosed.Porch -0.128964017 0.075765951 -0.350686435 -0.199042238  
## X3Ssn.Porch 0.007364426 0.046171615 0.003610568 0.024230828  
## Screen.Porch 0.049492446 0.056999746 -0.047430830 -0.045113898  
## Pool.Area -0.001059056 -0.023332377 0.006811695 -0.018137041  
## Misc.Val -0.023652056 0.044354497 -0.023647199 -0.013008178  
## Mo.Sold 0.005240751 0.032592541 -0.017797739 0.001978713  
## Yr.Sold 0.024626923 0.007681017 0.038575486 0.086782624  
## Mas.Vnr.Area BsmtFin.SF.1 BsmtFin.SF.2 Bsmt.Unf.SF  
## SalePrice 0.4872441266 0.455086938 0.038563700 0.160663128  
## MS.SubClass 0.0171904587 -0.077447320 -0.076408020 -0.107771250  
## Lot.Area 0.0888440751 0.163026112 0.082713478 0.012352745  
## Lot.Frontage 0.0868800568 0.015038191 0.019282656 0.115521387  
## Overall.Qual 0.3793772241 0.263950808 -0.017205234 0.242691621  
## Overall.Cond -0.1354390687 -0.061300935 0.024654278 -0.119397500  
## Year.Built 0.2970052889 0.306064960 0.001989174 0.069803505  
## Year.Remod.Add 0.1584481521 0.146021889 -0.036164817 0.112035529  
## Mas.Vnr.Area 1.0000000000 0.256030704 -0.001086365 0.074920780  
## BsmtFin.SF.1 0.2560307042 1.000000000 -0.056692060 -0.481774758  
## BsmtFin.SF.2 -0.0010863654 -0.056692060 1.000000000 -0.246204941  
## Bsmt.Unf.SF 0.0749207798 -0.481774758 -0.246204941 1.000000000  
## Total.Bsmt.SF 0.3372413769 0.503238995 0.124665829 0.423375499  
## X1st.Flr.SF 0.3411885212 0.417073283 0.107723084 0.305147505  
## X2nd.Flr.SF 0.1333663922 -0.182448316 -0.111355655 0.017255485  
## Low.Qual.Fin.SF -0.0532420603 -0.070382938 -0.004108984 0.049509886  
## Gr.Liv.Area 0.3695907043 0.147185752 -0.017208663 0.249834204  
## Bsmt.Full.Bath 0.1157459712 0.626171722 0.178966087 -0.387333353  
## Bsmt.Half.Bath -0.0003215569 0.063989821 0.101338118 -0.098536093  
## Full.Bath 0.2477938998 0.080184518 -0.068408890 0.262700060  
## Half.Bath 0.1910007040 -0.027549459 -0.034510821 -0.042736449  
## Bedroom.AbvGr 0.1105924760 -0.102688217 -0.041384850 0.211790872  
## Kitchen.AbvGr -0.0344850059 -0.121457545 -0.043642840 0.111777712  
## TotRms.AbvGrd 0.2498295091 0.001105134 -0.050376450 0.266399639  
## Fireplaces 0.2331703299 0.279186821 0.056244635 0.006933431  
## Garage.Yr.Blt 0.2298466469 0.200834328 -0.034225444 0.110568360  
## Garage.Cars 0.3318757722 0.256153924 0.002775868 0.155589562  
## Garage.Area 0.3275380210 0.288351280 0.026920355 0.135490857  
## Wood.Deck.SF 0.1469707469 0.224631238 0.100627962 -0.039479962  
## Open.Porch.SF 0.1308748558 0.106795286 -0.009945302 0.102602574  
## Enclosed.Porch -0.0991506688 -0.107165159 0.004307315 0.026546982  
## X3Ssn.Porch 0.0040798071 0.033223691 -0.023169574 0.002228535  
## Screen.Porch 0.0598229997 0.098024335 0.053642859 -0.048482274  
## Pool.Area -0.0028180326 0.019628007 0.058084120 -0.032735540  
## Misc.Val -0.0203512581 0.016968410 -0.004768288 -0.020840759  
## Mo.Sold -0.0190621133 -0.013893507 -0.004773267 0.004849702  
## Yr.Sold -0.0185889614 0.024424503 -0.004196422 0.008976161  
## Total.Bsmt.SF X1st.Flr.SF X2nd.Flr.SF Low.Qual.Fin.SF  
## SalePrice 0.645102431 0.637450315 0.2940321808 -0.025741914  
## MS.SubClass -0.222272902 -0.263961808 0.3103803322 0.014419970  
## Lot.Area 0.214818515 0.296307656 0.0166120538 0.002820987  
## Lot.Frontage 0.141735132 0.192465704 -0.0194001051 0.008942490  
## Overall.Qual 0.509732178 0.436880787 0.2583125556 -0.043486779  
## Overall.Cond -0.173815977 -0.154232101 0.0156149660 0.020345943  
## Year.Built 0.384386244 0.278008090 0.0190940091 -0.128629147  
## Year.Remod.Add 0.247719254 0.193082649 0.1770874193 -0.057067553  
## Mas.Vnr.Area 0.337241377 0.341188521 0.1333663922 -0.053242060  
## BsmtFin.SF.1 0.503238995 0.417073283 -0.1824483163 -0.070382938  
## BsmtFin.SF.2 0.124665829 0.107723084 -0.1113556547 -0.004108984  
## Bsmt.Unf.SF 0.423375499 0.305147505 0.0172554850 0.049509886  
## Total.Bsmt.SF 1.000000000 0.783960517 -0.2168120977 -0.022995387  
## X1st.Flr.SF 0.783960517 1.000000000 -0.2656260648 -0.004925290  
## X2nd.Flr.SF -0.216812098 -0.265626065 1.0000000000 0.008478562  
## Low.Qual.Fin.SF -0.022995387 -0.004925290 0.0084785616 1.000000000  
## Gr.Liv.Area 0.397920200 0.519486191 0.6808361139 0.097356733  
## Bsmt.Full.Bath 0.320683326 0.246212093 -0.1695464525 -0.041695895  
## Bsmt.Half.Bath 0.008586064 0.002589908 -0.0591744837 -0.024504991  
## Full.Bath 0.320496543 0.363297041 0.4224853089 0.014662691  
## Half.Bath -0.086744178 -0.141831726 0.6267659595 -0.027502928  
## Bedroom.AbvGr 0.093671110 0.130843482 0.5122748630 0.049097651  
## Kitchen.AbvGr -0.028636241 0.077843349 0.0835237862 -0.014248981  
## TotRms.AbvGrd 0.251436996 0.363213310 0.5937119191 0.107509055  
## Fireplaces 0.316284680 0.398693059 0.1760811362 0.007026635  
## Garage.Yr.Blt 0.302974276 0.220554799 0.0933312434 -0.033677398  
## Garage.Cars 0.421461043 0.427795480 0.1904990803 -0.039243417  
## Garage.Area 0.444249624 0.460315984 0.1400568228 -0.027278762  
## Wood.Deck.SF 0.232478897 0.237008699 0.0812563088 -0.004264646  
## Open.Porch.SF 0.209448386 0.190639265 0.1854272995 -0.001165802  
## Enclosed.Porch -0.080323364 -0.064692641 0.0561834246 0.056751827  
## X3Ssn.Porch 0.026110692 0.039490233 -0.0268081785 -0.002121040  
## Screen.Porch 0.073740869 0.102785113 0.0182586755 0.015010027  
## Pool.Area 0.011787326 0.083649356 -0.0005772319 -0.005431249  
## Misc.Val -0.006053311 -0.004536627 -0.0011747371 -0.005335646  
## Mo.Sold -0.011289215 0.009088845 0.0241758410 0.008890273  
## Yr.Sold 0.032263926 0.009033117 -0.0003916401 -0.008088556  
## Gr.Liv.Area Bsmt.Full.Bath Bsmt.Half.Bath Full.Bath  
## SalePrice 0.736905671 0.287816662 -0.0288773662 0.54801106  
## MS.SubClass 0.075600122 -0.007428545 -0.0184399930 0.14629222  
## Lot.Area 0.238234264 0.121155914 0.0257439281 0.11479187  
## Lot.Frontage 0.128834365 0.006294133 -0.0277299361 0.04475552  
## Overall.Qual 0.552619095 0.168643883 -0.0451040046 0.50365823  
## Overall.Cond -0.100605447 -0.063770129 0.0932564350 -0.21357169  
## Year.Built 0.214320902 0.232123260 -0.0264835514 0.44248365  
## Year.Remod.Add 0.296081147 0.145086263 -0.0409487202 0.42369751  
## Mas.Vnr.Area 0.369590704 0.115745971 -0.0003215569 0.24779390  
## BsmtFin.SF.1 0.147185752 0.626171722 0.0639898214 0.08018452  
## BsmtFin.SF.2 -0.017208663 0.178966087 0.1013381178 -0.06840889  
## Bsmt.Unf.SF 0.249834204 -0.387333353 -0.0985360929 0.26270006  
## Total.Bsmt.SF 0.397920200 0.320683326 0.0085860641 0.32049654  
## X1st.Flr.SF 0.519486191 0.246212093 0.0025899083 0.36329704  
## X2nd.Flr.SF 0.680836114 -0.169546452 -0.0591744837 0.42248531  
## Low.Qual.Fin.SF 0.097356733 -0.041695895 -0.0245049906 0.01466269  
## Gr.Liv.Area 1.000000000 0.032438363 -0.0524307240 0.64711234  
## Bsmt.Full.Bath 0.032438363 1.000000000 -0.1678366233 -0.03197262  
## Bsmt.Half.Bath -0.052430724 -0.167836623 1.0000000000 -0.04006940  
## Full.Bath 0.647112339 -0.031972615 -0.0400694025 1.00000000  
## Half.Bath 0.442236167 -0.047622555 -0.0706570910 0.17497398  
## Bedroom.AbvGr 0.554158014 -0.156817354 0.0145916945 0.38114687  
## Kitchen.AbvGr 0.130864927 -0.071483042 -0.0342594635 0.19665972  
## TotRms.AbvGrd 0.806464625 -0.070249194 -0.0435401146 0.54102974  
## Fireplaces 0.456177494 0.145675980 0.0411237356 0.24279708  
## Garage.Yr.Blt 0.245250777 0.167331970 -0.0483143969 0.46265985  
## Garage.Cars 0.486474399 0.161339312 -0.0205119312 0.47252439  
## Garage.Area 0.467704829 0.183582255 -0.0077519295 0.40350687  
## Wood.Deck.SF 0.249777290 0.187538846 0.0512309344 0.17888317  
## Open.Porch.SF 0.306813423 0.082443064 -0.0257762287 0.25178376  
## Enclosed.Porch 0.006005248 -0.083045261 -0.0207219599 -0.10641186  
## X3Ssn.Porch 0.005969687 0.018282013 -0.0188168936 0.01567559  
## Screen.Porch 0.094953591 0.052317386 0.0291815312 -0.01368191  
## Pool.Area 0.062035695 0.034756420 0.0724573940 0.00487660  
## Misc.Val -0.004953090 -0.027707536 -0.0020363607 -0.02040612  
## Mo.Sold 0.028964158 -0.001986077 0.0281782766 0.03018969  
## Yr.Sold 0.005706991 0.046212845 -0.0263461229 0.03976711  
## Half.Bath Bedroom.AbvGr Kitchen.AbvGr TotRms.AbvGrd  
## SalePrice 0.291755002 0.1823044219 -0.11432285 0.495767243  
## MS.SubClass 0.188592646 -0.0354405336 0.25361114 0.032388727  
## Lot.Area 0.008279069 0.1340086401 -0.01317101 0.190566771  
## Lot.Frontage -0.045451132 0.1053824320 0.02070740 0.161691107  
## Overall.Qual 0.270562395 0.0817993306 -0.15285977 0.355726368  
## Overall.Cond -0.093795062 -0.0045093438 -0.08419832 -0.071061096  
## Year.Built 0.274667875 -0.0421226325 -0.13803255 0.084647400  
## Year.Remod.Add 0.212188788 -0.0053983585 -0.14769179 0.179021515  
## Mas.Vnr.Area 0.191000704 0.1105924760 -0.03448501 0.249829509  
## BsmtFin.SF.1 -0.027549459 -0.1026882167 -0.12145754 0.001105134  
## BsmtFin.SF.2 -0.034510821 -0.0413848496 -0.04364284 -0.050376450  
## Bsmt.Unf.SF -0.042736449 0.2117908719 0.11177771 0.266399639  
## Total.Bsmt.SF -0.086744178 0.0936711097 -0.02863624 0.251436996  
## X1st.Flr.SF -0.141831726 0.1308434816 0.07784335 0.363213310  
## X2nd.Flr.SF 0.626765960 0.5122748630 0.08352379 0.593711919  
## Low.Qual.Fin.SF -0.027502928 0.0490976507 -0.01424898 0.107509055  
## Gr.Liv.Area 0.442236167 0.5541580137 0.13086493 0.806464625  
## Bsmt.Full.Bath -0.047622555 -0.1568173545 -0.07148304 -0.070249194  
## Bsmt.Half.Bath -0.070657091 0.0145916945 -0.03425946 -0.043540115  
## Full.Bath 0.174973982 0.3811468702 0.19665972 0.541029745  
## Half.Bath 1.000000000 0.2697296000 -0.04410651 0.343856617  
## Bedroom.AbvGr 0.269729600 1.0000000000 0.24217560 0.699137957  
## Kitchen.AbvGr -0.044106514 0.2421756042 1.00000000 0.296567437  
## TotRms.AbvGrd 0.343856617 0.6991379569 0.29656744 1.000000000  
## Fireplaces 0.186360651 0.0949917580 -0.10975250 0.298577894  
## Garage.Yr.Blt 0.229915527 -0.0307298467 -0.08434499 0.131463531  
## Garage.Cars 0.225466691 0.1291688337 -0.01980472 0.348428146  
## Garage.Area 0.162590340 0.1224210542 -0.03532338 0.317658332  
## Wood.Deck.SF 0.116019283 0.0291861844 -0.09262468 0.145693403  
## Open.Porch.SF 0.171988985 0.0808734705 -0.07180584 0.214363801  
## Enclosed.Porch -0.069955089 0.0535774269 0.02745518 0.025733946  
## X3Ssn.Porch -0.027155357 -0.0416074656 -0.01983423 -0.020289157  
## Screen.Porch 0.037872026 0.0118106995 -0.05363293 0.041026026  
## Pool.Area -0.002136603 0.0168013829 -0.01135937 0.048575674  
## Misc.Val 0.018767840 0.0110520404 0.03615248 0.008737649  
## Mo.Sold -0.003642132 0.0547963930 0.05160060 0.032666771  
## Yr.Sold 0.018506896 -0.0006006264 0.01387813 0.001048981  
## Fireplaces Garage.Yr.Blt Garage.Cars Garage.Area  
## SalePrice 0.492723739 0.489510079 0.632522561 0.62177406  
## MS.SubClass -0.043748639 0.115114813 -0.022537621 -0.07902542  
## Lot.Area 0.235933607 -0.021330618 0.167336175 0.18589706  
## Lot.Frontage 0.027491216 -0.024470363 0.101149002 0.13334887  
## Overall.Qual 0.388839953 0.513300491 0.566769047 0.51995078  
## Overall.Cond -0.042978107 -0.316763364 -0.191470982 -0.15845855  
## Year.Built 0.160137439 0.781832306 0.507354096 0.44925051  
## Year.Remod.Add 0.116390835 0.586352681 0.376293818 0.32546301  
## Mas.Vnr.Area 0.233170330 0.229846647 0.331875772 0.32753802  
## BsmtFin.SF.1 0.279186821 0.200834328 0.256153924 0.28835128  
## BsmtFin.SF.2 0.056244635 -0.034225444 0.002775868 0.02692036  
## Bsmt.Unf.SF 0.006933431 0.110568360 0.155589562 0.13549086  
## Total.Bsmt.SF 0.316284680 0.302974276 0.421461043 0.44424962  
## X1st.Flr.SF 0.398693059 0.220554799 0.427795480 0.46031598  
## X2nd.Flr.SF 0.176081136 0.093331243 0.190499080 0.14005682  
## Low.Qual.Fin.SF 0.007026635 -0.033677398 -0.039243417 -0.02727876  
## Gr.Liv.Area 0.456177494 0.245250777 0.486474399 0.46770483  
## Bsmt.Full.Bath 0.145675980 0.167331970 0.161339312 0.18358225  
## Bsmt.Half.Bath 0.041123736 -0.048314397 -0.020511931 -0.00775193  
## Full.Bath 0.242797083 0.462659854 0.472524390 0.40350687  
## Half.Bath 0.186360651 0.229915527 0.225466691 0.16259034  
## Bedroom.AbvGr 0.094991758 -0.030729847 0.129168834 0.12242105  
## Kitchen.AbvGr -0.109752501 -0.084344988 -0.019804716 -0.03532338  
## TotRms.AbvGrd 0.298577894 0.131463531 0.348428146 0.31765833  
## Fireplaces 1.000000000 0.077218585 0.307990725 0.27039912  
## Garage.Yr.Blt 0.077218585 1.000000000 0.472006737 0.46069313  
## Garage.Cars 0.307990725 0.472006737 1.000000000 0.88679909  
## Garage.Area 0.270399116 0.460693129 0.886799095 1.00000000  
## Wood.Deck.SF 0.225068006 0.230028224 0.234608022 0.24010975  
## Open.Porch.SF 0.166898005 0.201237983 0.207988468 0.22335791  
## Enclosed.Porch -0.005467580 -0.266968719 -0.120798577 -0.09786605  
## X3Ssn.Porch 0.015850153 0.007725845 0.014064078 0.02214665  
## Screen.Porch 0.183170274 -0.069149093 0.035520834 0.05284494  
## Pool.Area 0.085766527 -0.007983258 0.026716217 0.03027143  
## Misc.Val -0.015504525 -0.021201824 -0.037660838 -0.02422874  
## Mo.Sold 0.039918773 -0.013871265 0.032830803 0.02145565  
## Yr.Sold -0.007054911 0.050563465 0.015094783 0.01567345  
## Wood.Deck.SF Open.Porch.SF Enclosed.Porch X3Ssn.Porch  
## SalePrice 0.350808179 0.325160475 -0.117817356 0.016129935  
## MS.SubClass -0.015432265 -0.004452985 -0.027954934 -0.044666145  
## Lot.Area 0.157493379 0.068591855 0.014641504 0.019737514  
## Lot.Frontage -0.002685556 0.048942761 0.030442830 -0.002368941  
## Overall.Qual 0.254998275 0.292679518 -0.128964017 0.007364426  
## Overall.Cond 0.013213588 -0.064886176 0.075765951 0.046171615  
## Year.Built 0.230611484 0.181044904 -0.350686435 0.003610568  
## Year.Remod.Add 0.221776189 0.230302751 -0.199042238 0.024230828  
## Mas.Vnr.Area 0.146970747 0.130874856 -0.099150669 0.004079807  
## BsmtFin.SF.1 0.224631238 0.106795286 -0.107165159 0.033223691  
## BsmtFin.SF.2 0.100627962 -0.009945302 0.004307315 -0.023169574  
## Bsmt.Unf.SF -0.039479962 0.102602574 0.026546982 0.002228535  
## Total.Bsmt.SF 0.232478897 0.209448386 -0.080323364 0.026110692  
## X1st.Flr.SF 0.237008699 0.190639265 -0.064692641 0.039490233  
## X2nd.Flr.SF 0.081256309 0.185427299 0.056183425 -0.026808178  
## Low.Qual.Fin.SF -0.004264646 -0.001165802 0.056751827 -0.002121040  
## Gr.Liv.Area 0.249777290 0.306813423 0.006005248 0.005969687  
## Bsmt.Full.Bath 0.187538846 0.082443064 -0.083045261 0.018282013  
## Bsmt.Half.Bath 0.051230934 -0.025776229 -0.020721960 -0.018816894  
## Full.Bath 0.178883166 0.251783763 -0.106411860 0.015675589  
## Half.Bath 0.116019283 0.171988985 -0.069955089 -0.027155357  
## Bedroom.AbvGr 0.029186184 0.080873471 0.053577427 -0.041607466  
## Kitchen.AbvGr -0.092624682 -0.071805843 0.027455177 -0.019834227  
## TotRms.AbvGrd 0.145693403 0.214363801 0.025733946 -0.020289157  
## Fireplaces 0.225068006 0.166898005 -0.005467580 0.015850153  
## Garage.Yr.Blt 0.230028224 0.201237983 -0.266968719 0.007725845  
## Garage.Cars 0.234608022 0.207988468 -0.120798577 0.014064078  
## Garage.Area 0.240109747 0.223357914 -0.097866053 0.022146646  
## Wood.Deck.SF 1.000000000 0.042823399 -0.116613561 -0.008441427  
## Open.Porch.SF 0.042823399 1.000000000 -0.068379244 -0.020487453  
## Enclosed.Porch -0.116613561 -0.068379244 1.000000000 -0.030699070  
## X3Ssn.Porch -0.008441427 -0.020487453 -0.030699070 1.000000000  
## Screen.Porch -0.059903319 0.046002116 -0.068509018 -0.028247612  
## Pool.Area 0.082747314 0.043714528 0.079458303 -0.005608138  
## Misc.Val 0.021203883 0.007125746 0.011008653 0.001375270  
## Mo.Sold 0.020046935 0.020996826 0.003329210 0.022529515  
## Yr.Sold 0.020656625 -0.029001785 -0.002200507 0.019350667  
## Screen.Porch Pool.Area Misc.Val Mo.Sold  
## SalePrice 0.116879613 0.0357617451 -0.016624770 0.001283849  
## MS.SubClass -0.053171712 -0.0142584195 -0.029768463 0.010976013  
## Lot.Area 0.041869617 0.0545467191 0.042165310 0.005762339  
## Lot.Frontage 0.059727938 0.0770334821 -0.019623629 -0.003060657  
## Overall.Qual 0.049492446 -0.0010590557 -0.023652056 0.005240751  
## Overall.Cond 0.056999746 -0.0233323770 0.044354497 0.032592541  
## Year.Built -0.047430830 0.0068116947 -0.023647199 -0.017797739  
## Year.Remod.Add -0.045113898 -0.0181370411 -0.013008178 0.001978713  
## Mas.Vnr.Area 0.059823000 -0.0028180326 -0.020351258 -0.019062113  
## BsmtFin.SF.1 0.098024335 0.0196280073 0.016968410 -0.013893507  
## BsmtFin.SF.2 0.053642859 0.0580841198 -0.004768288 -0.004773267  
## Bsmt.Unf.SF -0.048482274 -0.0327355398 -0.020840759 0.004849702  
## Total.Bsmt.SF 0.073740869 0.0117873255 -0.006053311 -0.011289215  
## X1st.Flr.SF 0.102785113 0.0836493557 -0.004536627 0.009088845  
## X2nd.Flr.SF 0.018258676 -0.0005772319 -0.001174737 0.024175841  
## Low.Qual.Fin.SF 0.015010027 -0.0054312487 -0.005335646 0.008890273  
## Gr.Liv.Area 0.094953591 0.0620356946 -0.004953090 0.028964158  
## Bsmt.Full.Bath 0.052317386 0.0347564196 -0.027707536 -0.001986077  
## Bsmt.Half.Bath 0.029181531 0.0724573940 -0.002036361 0.028178277  
## Full.Bath -0.013681909 0.0048766004 -0.020406115 0.030189687  
## Half.Bath 0.037872026 -0.0021366033 0.018767840 -0.003642132  
## Bedroom.AbvGr 0.011810699 0.0168013829 0.011052040 0.054796393  
## Kitchen.AbvGr -0.053632927 -0.0113593750 0.036152475 0.051600597  
## TotRms.AbvGrd 0.041026026 0.0485756742 0.008737649 0.032666771  
## Fireplaces 0.183170274 0.0857665271 -0.015504525 0.039918773  
## Garage.Yr.Blt -0.069149093 -0.0079832584 -0.021201824 -0.013871265  
## Garage.Cars 0.035520834 0.0267162170 -0.037660838 0.032830803  
## Garage.Area 0.052844941 0.0302714309 -0.024228745 0.021455649  
## Wood.Deck.SF -0.059903319 0.0827473137 0.021203883 0.020046935  
## Open.Porch.SF 0.046002116 0.0437145276 0.007125746 0.020996826  
## Enclosed.Porch -0.068509018 0.0794583026 0.011008653 0.003329210  
## X3Ssn.Porch -0.028247612 -0.0056081382 0.001375270 0.022529515  
## Screen.Porch 1.000000000 0.0438865929 0.003804365 0.020451376  
## Pool.Area 0.043886593 1.0000000000 0.020984913 -0.045874973  
## Misc.Val 0.003804365 0.0209849132 1.000000000 -0.006054692  
## Mo.Sold 0.020451376 -0.0458749728 -0.006054692 1.000000000  
## Yr.Sold -0.012371564 -0.0501950757 0.018607610 -0.129275124  
## Yr.Sold  
## SalePrice 0.0225789421  
## MS.SubClass -0.0290881282  
## Lot.Area -0.0112260210  
## Lot.Frontage 0.0187294519  
## Overall.Qual 0.0246269230  
## Overall.Cond 0.0076810171  
## Year.Built 0.0385754864  
## Year.Remod.Add 0.0867826237  
## Mas.Vnr.Area -0.0185889614  
## BsmtFin.SF.1 0.0244245030  
## BsmtFin.SF.2 -0.0041964215  
## Bsmt.Unf.SF 0.0089761606  
## Total.Bsmt.SF 0.0322639255  
## X1st.Flr.SF 0.0090331166  
## X2nd.Flr.SF -0.0003916401  
## Low.Qual.Fin.SF -0.0080885564  
## Gr.Liv.Area 0.0057069914  
## Bsmt.Full.Bath 0.0462128446  
## Bsmt.Half.Bath -0.0263461229  
## Full.Bath 0.0397671130  
## Half.Bath 0.0185068956  
## Bedroom.AbvGr -0.0006006264  
## Kitchen.AbvGr 0.0138781335  
## TotRms.AbvGrd 0.0010489814  
## Fireplaces -0.0070549111  
## Garage.Yr.Blt 0.0505634649  
## Garage.Cars 0.0150947826  
## Garage.Area 0.0156734509  
## Wood.Deck.SF 0.0206566251  
## Open.Porch.SF -0.0290017852  
## Enclosed.Porch -0.0022005066  
## X3Ssn.Porch 0.0193506673  
## Screen.Porch -0.0123715636  
## Pool.Area -0.0501950757  
## Misc.Val 0.0186076103  
## Mo.Sold -0.1292751244  
## Yr.Sold 1.0000000000

# Train and Test Datasets

set.seed(100)  
train <- Ames %>% sample\_frac(size = 0.75)  
test <- Ames %>% setdiff(train)

# SalePrice

# The target variable for my project will SalePrice as I will be trying to predict the amount a house is worth when given specific information about the house.

# Null Model for SalePrice

Null.mod <- lm(SalePrice~1,data=train)  
msummary(Null.mod)

## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 174702 1670 104.6 <2e-16 \*\*\*  
##   
## Residual standard error: 71050 on 1808 degrees of freedom

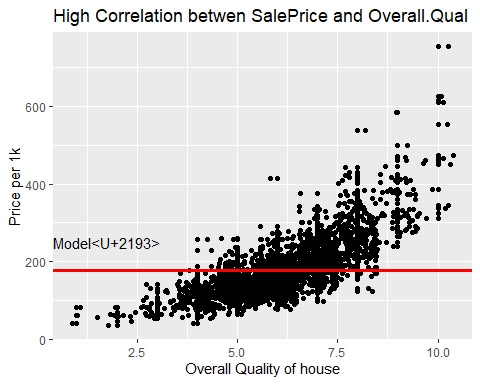
# Null Model Test

pred1 <- predict(Null.mod, newdata = test)  
results <- data.frame(pred = pred1, original = test$SalePrice)  
results$resid<-round(results$pred-results$original,0)  
MSE = (sum(results$resid^2)/length(results))  
RMSE <- as.integer(round(sqrt(MSE)/100)\*100)  
RMSE

## [1] 1002400

# Null Model Visualization

ggplot(data = Ames, aes(x = Overall.Qual, y = SalePrice/1000)) +geom\_point() + geom\_jitter(width = .49)+geom\_hline(yintercept = 175.714, color = "red",size = 1.3) + ggtitle("High Correlation betwen SalePrice and Overall.Qual") + xlab("Overall Quality of house") + ylab("Price per 1k") + annotate(geom = "text",x=1.4,y = 250, label = "Null Model↓")



# Null Model Analysis

# The Null model is what is expected of it. It estimates every price to be 175,714$. This model is not good but will be used for reference for the other models.

# Multiple Regression Model for SalePrice

lin.mod <- lm(SalePrice ~Overall.Qual+ Overall.Cond + Year.Built + Gr.Liv.Area + Total.Bsmt.SF + Full.Bath + Year.Built + Year.Remod.Add + Mas.Vnr.Area + Garage.Cars, data = train)  
msummary(lin.mod)

## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.267e+06 8.303e+04 -15.259 < 2e-16 \*\*\*  
## Overall.Qual 1.493e+04 7.656e+02 19.496 < 2e-16 \*\*\*  
## Overall.Cond 5.459e+03 6.847e+02 7.974 2.71e-15 \*\*\*  
## Year.Built 3.782e+02 3.561e+01 10.620 < 2e-16 \*\*\*  
## Gr.Liv.Area 6.293e+01 2.106e+00 29.885 < 2e-16 \*\*\*  
## Total.Bsmt.SF 4.067e+01 1.930e+00 21.073 < 2e-16 \*\*\*  
## Full.Bath -1.246e+04 1.765e+03 -7.058 2.40e-12 \*\*\*  
## Year.Remod.Add 2.226e+02 4.410e+01 5.048 4.91e-07 \*\*\*  
## Mas.Vnr.Area 3.840e+01 4.205e+00 9.131 < 2e-16 \*\*\*  
## Garage.Cars 9.515e+03 1.184e+03 8.036 1.66e-15 \*\*\*  
##   
## Residual standard error: 27310 on 1799 degrees of freedom  
## Multiple R-squared: 0.853, Adjusted R-squared: 0.8522   
## F-statistic: 1160 on 9 and 1799 DF, p-value: < 2.2e-16

confint(lin.mod)

## 2.5 % 97.5 %  
## (Intercept) -1.429773e+06 -1.104086e+06  
## Overall.Qual 1.342479e+04 1.642790e+04  
## Overall.Cond 4.116526e+03 6.802146e+03  
## Year.Built 3.083443e+02 4.480230e+02  
## Gr.Liv.Area 5.880382e+01 6.706421e+01  
## Total.Bsmt.SF 3.688481e+01 4.445501e+01  
## Full.Bath -1.591831e+04 -8.995325e+03  
## Year.Remod.Add 1.361256e+02 3.090973e+02  
## Mas.Vnr.Area 3.014976e+01 4.664585e+01  
## Garage.Cars 7.192671e+03 1.183683e+04

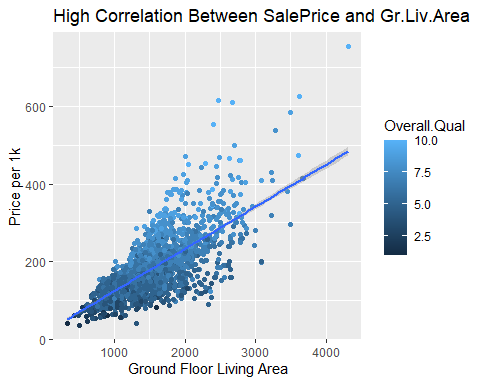
# Multiple Regression Model Test

pred1 <- predict(lin.mod, newdata = test)  
results <- data.frame(pred = pred1, original = test$SalePrice)  
results$resid<-round(results$pred-results$original,0)  
MSE = (sum(results$resid^2)/length(results))  
RMSE <- as.integer(round(sqrt(MSE)/100)\*100)  
RMSE

## [1] 420900

# Mulitple Regression Model Visualization

ggplot(data = Ames, aes(x = Gr.Liv.Area, y = SalePrice/1000, color = Overall.Qual)) + geom\_point() + geom\_jitter(width = .49) + geom\_smooth(stat = "lm") + ggtitle("High Correlation Between SalePrice and Gr.Liv.Area") + xlab("Ground Floor Living Area") + ylab("Price per 1k")



# Multiple Regression Model Analysis

# This is a great model but it does not work well on the dataset according to the residuals. The model itself shows no type of error as it has great P values, T values, 95% confidence interval makes sense and the Adj R^2 is great! I also believe the residual error gets punished to much by ^2 it. As the higher the residuals the greater it gets punished so when using data with naturally higher numbers it gets to become a higher error. So instead I compared it with the RMSE which would be 415,000$ difference.  
  
# This isnt my first model. I toyed around with different predictors, both categorical and numerical. Most categorical variables were not statistically significant or didnt effect the model and I believe in KISS (Keep it simple stupid).  
  
# Compared to the Null Model this model is great! The RMSE is much less then the Null models RMSE, by over half!

# Regression Tree Model for SalePrice

form<-as.formula("SalePrice ~ .")  
Reg\_tree <- rpart(form, data=train)  
Reg\_tree

## n= 1809   
##   
## node), split, n, deviance, yval  
## \* denotes terminal node  
##   
## 1) root 1809 9.127448e+12 174701.50   
## 2) Overall.Qual< 7.5 1567 3.450968e+12 156147.80   
## 4) Neighborhood=Blueste,BrDale,BrkSide,Edwards,IDOTRR,Landmrk,MeadowV,Mitchel,NAmes,NPkVill,OldTown,Sawyer,SWISU 967 1.203048e+12 133764.00   
## 8) Gr.Liv.Area< 1376.5 635 4.623898e+11 122074.70   
## 16) Overall.Qual< 4.5 147 8.804783e+10 94992.36 \*  
## 17) Overall.Qual>=4.5 488 2.340462e+11 130232.70 \*  
## 9) Gr.Liv.Area>=1376.5 332 4.879418e+11 156121.40 \*  
## 5) Neighborhood=Blmngtn,ClearCr,CollgCr,Crawfor,Gilbert,GrnHill,NoRidge,NridgHt,NWAmes,SawyerW,Somerst,Timber,Veenker 600 9.825585e+11 192223.20   
## 10) Gr.Liv.Area< 1482 240 1.878665e+11 163623.40 \*  
## 11) Gr.Liv.Area>=1482 360 4.675145e+11 211289.60   
## 22) Gr.Liv.Area< 2036 286 2.476093e+11 201481.60 \*  
## 23) Gr.Liv.Area>=2036 74 8.606072e+10 249196.40 \*  
## 3) Overall.Qual>=7.5 242 1.644211e+12 294840.00   
## 6) Gr.Liv.Area< 2293 180 6.275599e+11 266293.90   
## 12) Total.Bsmt.SF< 1709.5 134 2.333129e+11 243726.80 \*  
## 13) Total.Bsmt.SF>=1709.5 46 1.272113e+11 332032.70 \*  
## 7) Gr.Liv.Area>=2293 62 4.441310e+11 377715.80   
## 14) BsmtFin.SF.1< 1265.5 48 1.125295e+11 344717.30 \*  
## 15) BsmtFin.SF.1>=1265.5 14 1.001321e+11 490853.60 \*

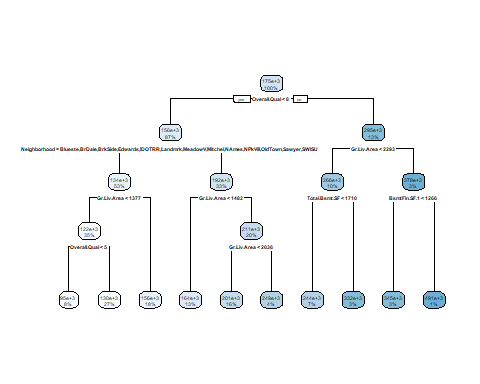
# Regression Tree Model Test

pred1 <- predict(Reg\_tree, newdata = test)  
results <- data.frame(pred = pred1, original = test$SalePrice)  
results$resid<-round(results$pred-results$original,0)  
MSE = (sum(results$resid^2)/length(results))  
RMSE <- as.integer(round(sqrt(MSE)/100)\*100)  
RMSE

## [1] 531400

# Regression Tree Model Visualization

rpart.plot(Reg\_tree)



# Regression Tree Model Analysis

# While the Regression Tree does look more sophisticated it preforms worse then the Multiple Regression Model as the RMSE is now 531400$. As this is worse, I will be using the Multiple Regression Model.

# Random Forest Model for SalePrice

Ran\_tree <-randomForest(form, data=train, ntree=400, mtry=20)  
Ran\_tree

##   
## Call:  
## randomForest(formula = form, data = train, ntree = 400, mtry = 20)   
## Type of random forest: regression  
## Number of trees: 400  
## No. of variables tried at each split: 20  
##   
## Mean of squared residuals: 408676747  
## % Var explained: 91.9

# Random Forest Model Test

pred1 <- predict(Ran\_tree, newdata = test)  
results <- data.frame(pred = pred1, original = test$SalePrice)  
results$resid<-round(results$pred-results$original,0)  
MSE = (sum(results$resid^2)/length(results))  
RMSE <- as.integer(round(sqrt(MSE)/100)\*100)  
RMSE

## [1] 322200

# Random Forest Model Analysis

# The Random Forest Model is by far the best Model created. The first ting I noticed was the high Variance. This model explains 92% of the variance which is higher then the rest. But also the RMSE is ~ 320,000$ which is the best so far!

# Overall.Qual (> 6)

Ames <- Ames %>%  
 mutate(Overall.Binary = if\_else(Overall.Qual < 7, 0,1))  
  
# I created my own categorical binary column to predict if the overall quality is 7 or above. This is for people interested in buying a good home.

# Train and Test Datasets

set.seed(100)  
train <- Ames %>% sample\_frac(size = 0.75)  
test <- Ames %>% setdiff(train)

# Null Model for Overall.Binary

Null.mod <- glm( Overall.Binary ~ 1,data=train,family = binomial)  
Null.mod

##   
## Call: glm(formula = Overall.Binary ~ 1, family = binomial, data = train)  
##   
## Coefficients:  
## (Intercept)   
## -0.6981   
##   
## Degrees of Freedom: 1808 Total (i.e. Null); 1808 Residual  
## Null Deviance: 2300   
## Residual Deviance: 2300 AIC: 2302

# Null Model Test

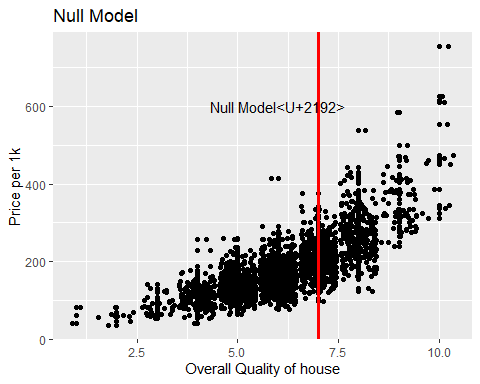
table(train$Overall.Binary)/sum(table(train$Overall.Binary))\*100

##   
## 0 1   
## 66.77722 33.22278

# 66.7% Accurate

# Null Model Visualization

ggplot(data = Ames, aes(x = Overall.Qual, y = SalePrice/1000)) +geom\_point() + geom\_jitter(width = .45,height = .01) + ggtitle("Null Model") + xlab("Overall Quality of house") + ylab("Price per 1k") + geom\_vline(xintercept = 7,color = "red", size = 1.3)+ annotate(geom = "text",x=6,y = 600, label = "Null Model→")



# Null Model Analysis

# This null model Assumes that the overall Quality of a house is below 7/10 and is 66.77% accurate in doing so.

# Logisitic Regression Classifier for Overall.Binary

Log.Cla <- glm(Overall.Binary ~ SalePrice + Gr.Liv.Area + Garage.Cars +Lot.Area, data = train, family = "binomial")  
msummary(Log.Cla)

## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -8.052e+00 4.302e-01 -18.717 < 2e-16 \*\*\*  
## SalePrice 5.122e-05 3.266e-06 15.680 < 2e-16 \*\*\*  
## Gr.Liv.Area -7.887e-04 2.751e-04 -2.867 0.00414 \*\*   
## Garage.Cars 3.741e-01 1.701e-01 2.200 0.02783 \*   
## Lot.Area -1.361e-04 2.028e-05 -6.712 1.92e-11 \*\*\*  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2300.1 on 1808 degrees of freedom  
## Residual deviance: 1146.2 on 1804 degrees of freedom  
## AIC: 1156.2  
##   
## Number of Fisher Scoring iterations: 6

confint(Log.Cla)

## Waiting for profiling to be done...

## 2.5 % 97.5 %  
## (Intercept) -8.925671e+00 -7.237361e+00  
## SalePrice 4.503465e-05 5.785098e-05  
## Gr.Liv.Area -1.336197e-03 -2.571762e-04  
## Garage.Cars 4.251441e-02 7.097155e-01  
## Lot.Area -1.771672e-04 -9.802782e-05

# Logisitic Regression Classifier Test

logreg.probs = predict(Log.Cla,train,type="response")  
logreg.pred = rep("0",1809)  
logreg.pred[logreg.probs>.41]="1"  
confusion <- table(logreg.pred, train$Overall.Binary)  
confusion

##   
## logreg.pred 0 1  
## 0 1093 118  
## 1 115 483

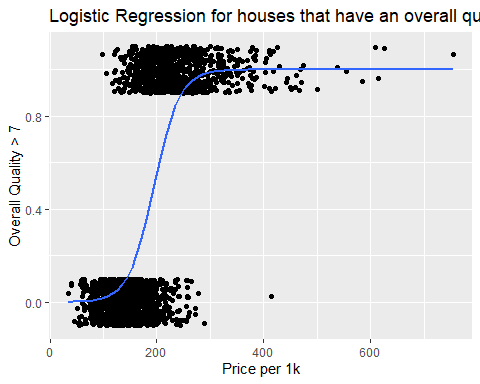
sum(diag(confusion))/nrow(train)

## [1] 0.8711996

# Logisitic Regression Classifier Visualization

ggplot(data = Ames, aes(x = SalePrice/1000, y = Overall.Binary)) + geom\_jitter(width = .1, height = .1) + geom\_smooth(method = "glm", method.args = list(family = "binomial"),se = FALSE) + ggtitle("Logistic Regression for houses that have an overall quality > 6") + xlab("Price per 1k") + ylab("Overall Quality > 7") #+ geom\_hline(yintercept = .41, color = "red", size = 1.3) #+ annotate(geom = "text",x=350,y = .5, label = "Threshold Value (.41) ↓")

## `geom\_smooth()` using formula 'y ~ x'



# Logisitic Regression Classifier Analysis

# By selecting a threshold valye of .41 we can achieve an optimal accuracy of 87%. This is the best model I could come up with and by messing with all the variables, non seemed to change the Null Deviance or residual deviance at all. This was an okay model but still did better then the Null model.

# Decision Tree Classifier for Overall.Binary

test$Overall.Binary <- factor(test$Overall.Binary)  
train$Overall.Binary <- factor(train$Overall.Binary)  
  
form <- as.formula("Overall.Binary ~ . -Overall.Qual")  
Dec\_tree <- rpart(form,data = train)  
Dec\_tree

## n= 1809   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 1809 601 0 (0.66777225 0.33222775)   
## 2) Exter.Qual=Fa,TA 1191 116 0 (0.90260285 0.09739715) \*  
## 3) Exter.Qual=Ex,Gd 618 133 1 (0.21521036 0.78478964)   
## 6) SalePrice< 180250 146 58 0 (0.60273973 0.39726027)   
## 12) Neighborhood=BrkSide,CollgCr,Crawfor,IDOTRR,Mitchel,NAmes,NPkVill,OldTown,Sawyer,SawyerW,Timber,Veenker 84 13 0 (0.84523810 0.15476190) \*  
## 13) Neighborhood=Blmngtn,Blueste,ClearCr,Edwards,Gilbert,NridgHt,NWAmes,Somerst,StoneBr,SWISU 62 17 1 (0.27419355 0.72580645) \*  
## 7) SalePrice>=180250 472 45 1 (0.09533898 0.90466102) \*

# Decision Tree Classifer

T\_pred = predict(Dec\_tree,test,type ="class")  
confusion <- table(test$Overall.Binary,T\_pred)  
confusion

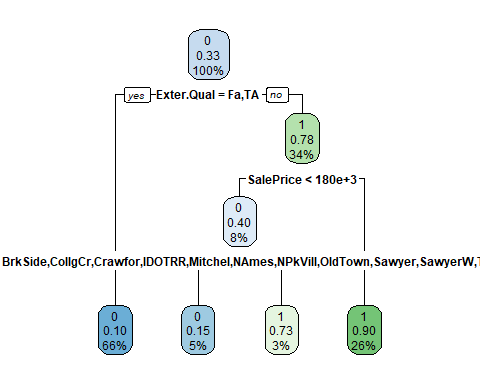
## T\_pred  
## 0 1  
## 0 359 17  
## 1 65 162

Accuracy <- sum(diag(confusion))/sum(confusion)  
Accuracy

## [1] 0.8640133

# Decision Tree Classifer Visualization

rpart.plot(Dec\_tree)



# Decision Tree Classifer Analysis

# This decision tree is about the same accuracy as the logistic regression classifier. The big upside in using this model is it goes through every variable and detects the most important variables and uses them for the decision tree. So while the Logistic regression model is better accuracy (by 1%) I would choose the decision tree as it gets rid of most of the human error.

# Random Forest Classifier for Overall.Binary

Ran\_tree <-randomForest(form, data=train, ntree=400, mtry=20)  
Ran\_tree

##   
## Call:  
## randomForest(formula = form, data = train, ntree = 400, mtry = 20)   
## Type of random forest: classification  
## Number of trees: 400  
## No. of variables tried at each split: 20  
##   
## OOB estimate of error rate: 10.17%  
## Confusion matrix:  
## 0 1 class.error  
## 0 1142 66 0.05463576  
## 1 118 483 0.19633943

# Random Forest Classifier Test

sum(diag(Ran\_tree$confusion))/nrow(train)

## [1] 0.8982863

# Random Forest Classifier Analysis

# The Random Forest Model is by far the best classifier created. This model is 90% accurate! and with a low OOb estimate of error rate of 10.17% this is by far the best Classifier.