House Prediction : Advance Regression Techniques

Vaishnavi Bihare

San Diego State University

MIS 749_01: Business Analytics

Dr. Aaron Elkins

May 9, 2020

Executive Summary

The “Ames Housing dataset” complied by Dean de cock is being used in this project. The main objective of this project is to analyze, clean the data, choose strongest predictors and build the model to predict the Sale Price. We will be predicting the Sale Price of individual residential property as described in the Kaggle dataset.

The data has been split into 50% train and 50% test sets. The testing dataset consists of 1459 observations with 80 variables and the training dataset consists of 1460 observations each with 81 variables including the Sale Price. The dataset consists of 2919 observations in all, consisting of:

1. 14 discrete
2. 23 nominals
3. 20 continuous
4. 23 ordinal variables

The training dataset is used for training the model and the testing dataset is used for predicting the Sales Price and evaluating the model performance.

There were various steps involved in predicting the Sale Price for the test data, from understanding the data to modelling it so that the outcome could be the best. Understanding the data is a major part of Data analysis, because the more you understand the better your predictions could get.

Then extensive EDA was performed on each predictor and its values and the potential implications that might have on the final model. Quite a few visualizations were included to provide insight to the interpretation that was being made on the structure of the data and to give us a better understanding of variables.

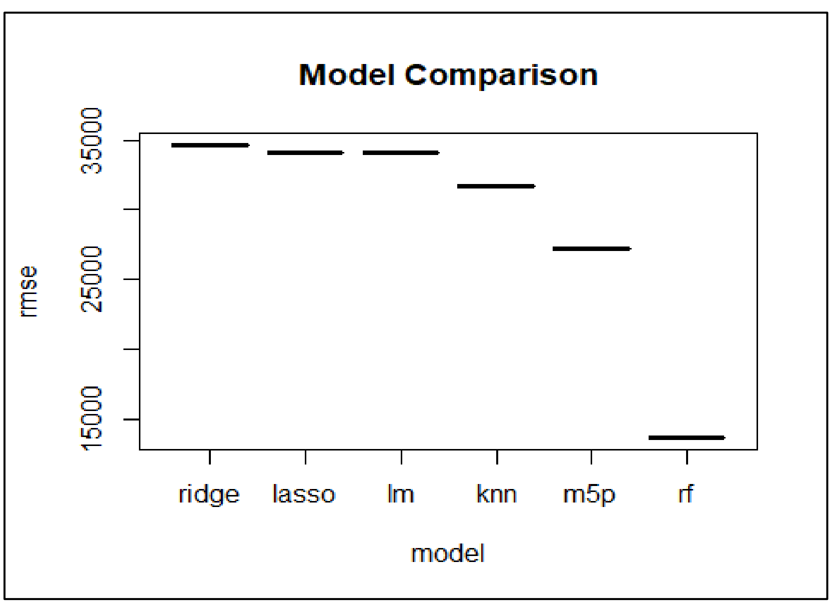
After understanding the variables, the dataset contained a large amount of NULL values, which were then curated and replaced with appropriate values to make the dataset compatible for analysis. It was performed, where I replaced the NA's and factor the character variables.

Next step was creating the models to understand which variables played an important role in predicting the Sale Price and then using that result to create our final models.

Regression techniques such as : Linear Regression, The Lasso, Ridge Regression, Random Forest, K-Nearest Neighbor and M5P were used, and based on RMSE and R square value, model selection to predict the Sale Price for the test dataset was done .

The Random Forest Model outperformed the other models and was chosen as the model of choice for this dataset, despite the m5p model being a close second. This model was then used to make SalePrice predictions on the test dataset and performed well.

Plots were then made to display which model had the best performance metrics among all the ones selected.



Discovery and Data Preparation

It was a kind of a struggle to find a perfect dataset, I knew I wanted to do Regression, but finding a dataset which qualified the requirements of our project and was also engaging was a little bit difficult. But then after looking around , I found this “House price Prediction” on Kaggle. It had both categorical and numerical variables and I thought it will be a good opportunity to perform regression with such a huge number of variables.

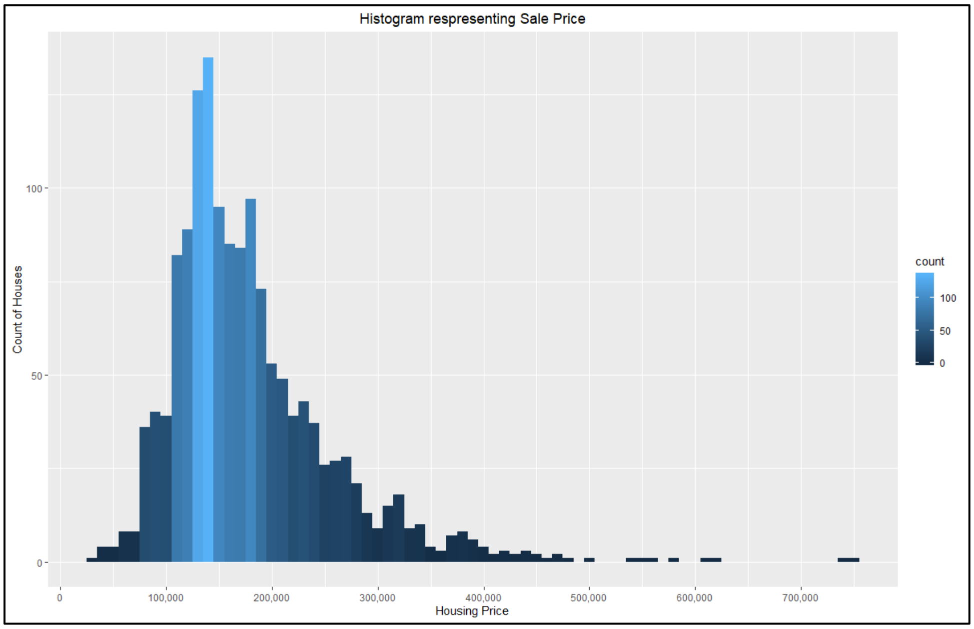
The dataset consisted 2919 rows and 79 predictors, and 1 response variable i.e. “SalePrice”. The train dataset consists of 1460 rows and 81 columns, where as the test dataset consists of 1459 rows and 80 columns. The structure of data is such that it contains 14 discrete, 23 nominal, 20 continuous and 23 ordinal variables (not including Sales Price). Following is a brief summary of all the predictors:

1. MSSubClass: Identifies the type of dwelling involved in the sale.(numerical)
2. MSZoning: Identifies the general zoning classification of the sale.(categorical)
3. LotFrontage: Linear feet of street connected to property(numerical)
4. LotArea: Lot size in square feet(numerical)
5. Street: Type of road access to property(categorical)
6. Alley: Type of alley access to property(categorical)
7. LotShape: General shape of property(categorical)
8. LandContour: Flatness of the property(categorical)
9. Utilities: Type of utilities available(categorical)
10. LotConfig: Lot configuration(categorical)
11. LandSlope: Slope of property(categorical)
12. Neighborhood: Physical locations within Ames city limits(categorical)
13. Condition1: Proximity to various conditions(categorical)
14. Condition2: Proximity to various conditions (if more than one is present) (categorical)
15. BldgType: Type of dwelling(categorical)
16. HouseStyle: Style of dwelling(categorical)
17. OverallQual: Rates the overall material and finish of the house(numerical)

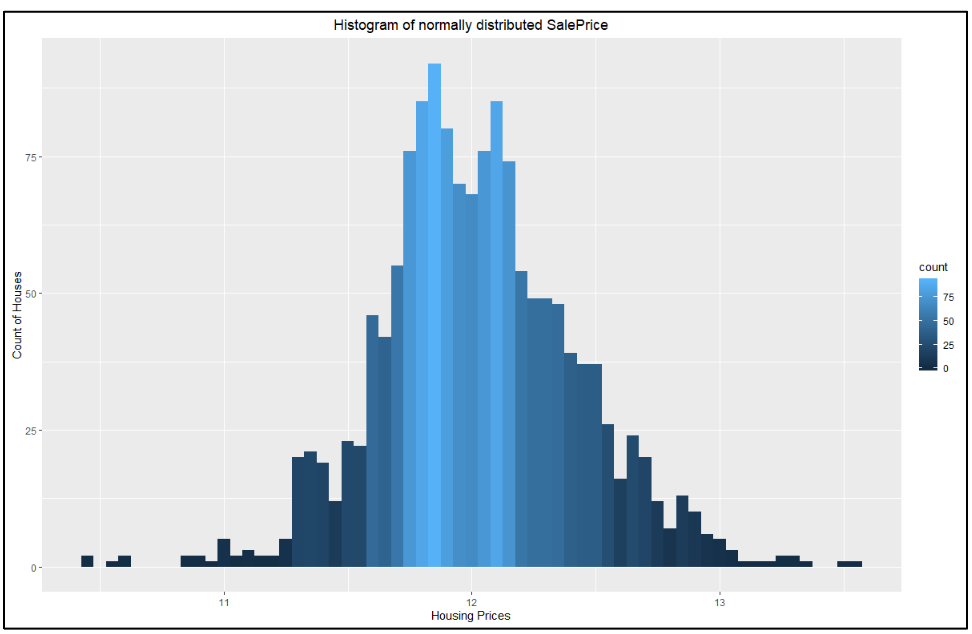
18. OverallCond: Rates the overall condition of the house(numerical)
19. YearBuilt: Original construction date(numerical)
20. RoofMatl: Roof material(categorical)
21. YearRemodAdd: Remodel date (same as construction date if no remodeling or additions) (numerical)
22. Exterior1st: Exterior covering on house(categorical)
23. RoofStyle: Type of roof(categorical)
24. Exterior2nd: Exterior covering on house (if more than one material) (categorical)
25. MasVnrType: Masonry veneer type(categorical)
26. MasVnrArea: Masonry veneer area in square feet(numerical)
27. ExterQual: Evaluates the quality of the material on the exterior(categorical)
28. ExterCond: Evaluates the present condition of the material on the exterior(categorical)
29. Foundation: Type of foundation(categorical)
30. BsmtQual: Evaluates the height of the basement(categorical)
31. BsmtCond: Evaluates the general condition of the basement. (categorical)
32. BsmtExposure: Refers to walkout or garden level walls(categorical)
33. BsmtFinType1: Rating of basement finished area(categorical)
34. BsmtFinSF1: Type 1 finished square feet(numerical)
35. BsmtFinSF2: Type 2 finished square feet(numerical)
36. HeatingQC: Heating quality and condition(categorical)
37. BsmtUnfSF: Unfinished square feet of basement area(numerical)
38. CentralAir: Central air conditioning(categorical)
39. TotalBsmtSF: Total square feet of basement area(numerical)
40. Electrical: Electrical system(categorical)
41. Heating: Type of heating(categorical)
42. BsmtFinType2: Rating of basement finished area (if multiple types) (categorical)
43. 1stFlrSF: First Floor square feet(numerical)
44. TotRmsAbvGrd: Total rooms above grade (does not include bathrooms) (numerical)
45. 2ndFlrSF: Second floor square feet(numerical)
46. Functional: Home functionality (Assume typical unless deductions are warranted) (categorical)
47. LowQualFinSF: Low quality finished square feet (all floors) (numerical)
48. Fireplaces: Number of fireplaces(numerical)

49. GrLivArea: Above grade (ground) living area square feet(numerical)
50. FireplaceQu: Fireplace quality(categorical)
51. BsmtFullBath: Basement full bathrooms(numerical)
52. GarageType: Garage location(categorical)
53. BsmtHalfBath: Basement half bathrooms(numerical)
54. GarageYrBlt: Year garage was built(numerical)
55. FullBath: Full bathrooms above grade(numerical)
56. GarageFinish: Interior finish of the garage(categorical)
57. HalfBath: Half baths above grade(numerical)
58. GarageCars: Size of garage in car capacity(numerical)
59. Bedroom: Bedrooms above grade (does NOT include basement bedrooms) (numerical)
60. GarageArea: Size of garage in square feet(numerical)
61. Kitchen: Kitchens above grade(numerical)
62. GarageQual: Garage quality(categorical)
63. KitchenQual: Kitchen quality(categorical)
64. WoodDeckSF: Wood deck area in square feet(numerical)
65. GarageCond: Garage condition(categorical)
66. OpenPorchSF: Open porch area in square feet(numerical)
67. PavedDrive: Paved driveway(categorical)
68. EnclosedPorch: Enclosed porch area in square feet(numerical)
69. Fence: Fence quality(categorical)
70. 3SsnPorch: Three season porch area in square feet(numerical)
71. MiscFeature: Miscellaneous feature not covered in other categories
72. ScreenPorch: Screen porch area in square feet(numerical)
73. MiscVal: $Value of miscellaneous feature(numerical)
74. PoolArea: Pool area in square feet(numerical)
75. MoSold: Month Sold (MM) (numerical)
76. PoolQC: Pool quality(categorical)
77. YrSold: Year Sold (YYYY) (numerical)
78. SaleCondition: Condition of sale(categorical)
79. SaleType: Type of sale(categorical)

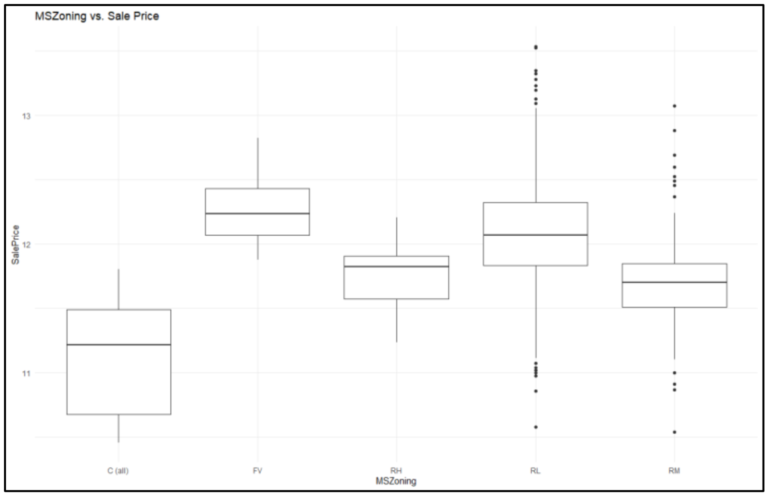
SalePrice is our target variable and also the dependent variable for prediction. According to the assumptions of Linear regression, data should be normally distributed. By checking the distribution of SalePrice we can decide if we need non-linear transformation, like log term to make better predictions.



It is right skewed, hence taking the log transformation was necessary. After the log transformation it looked like:

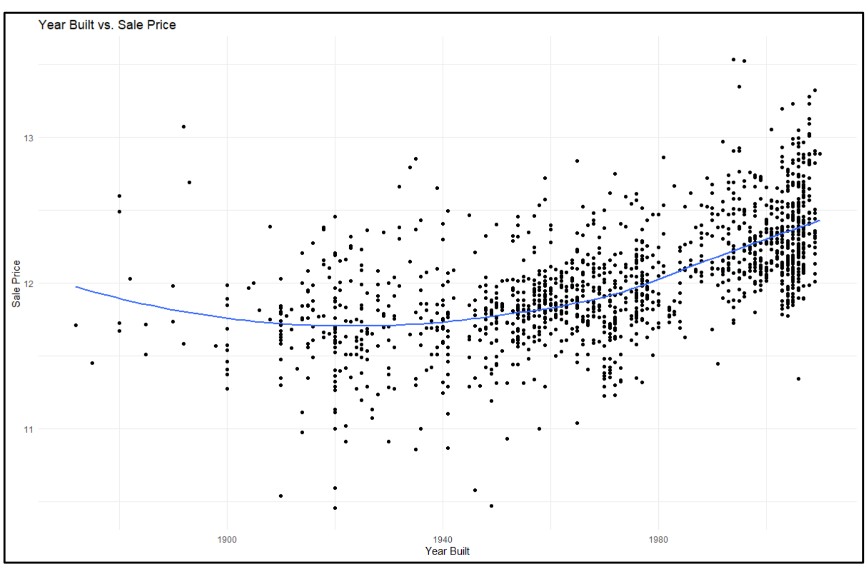


After this I tried to get understanding of different variables with respect to the target variable. First one it MSZONING: Identifies the general zoning classification of the sale. It is a categorical variable.



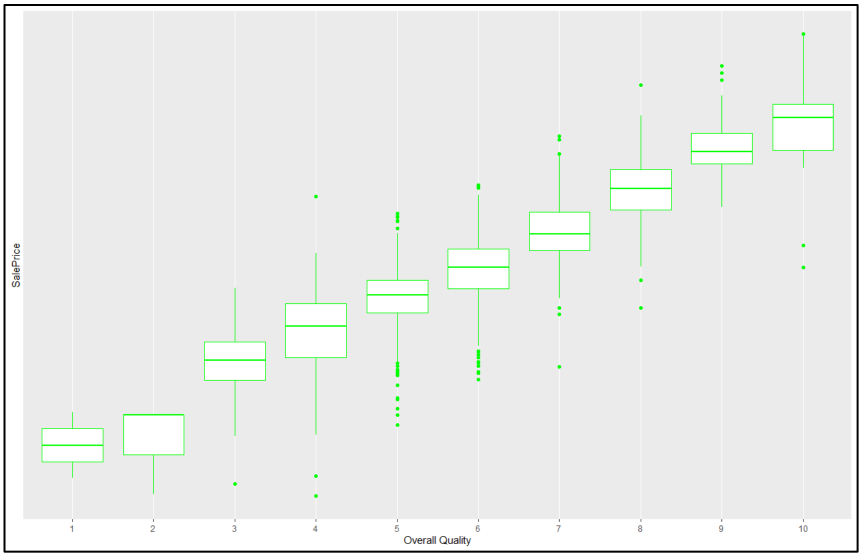
This shows the distribution of SalePrice by MSZoning. The sales in the "Floating village Residential" area have the highest average SalePrice, whereas the "Commercial" area have the lowest average SalePrice, which is kind of strange. As Commercial areas are generally more in demand.

Next is "SalePrice vs YearBuilt" - Sale price seems to have weird correlation to the year built.



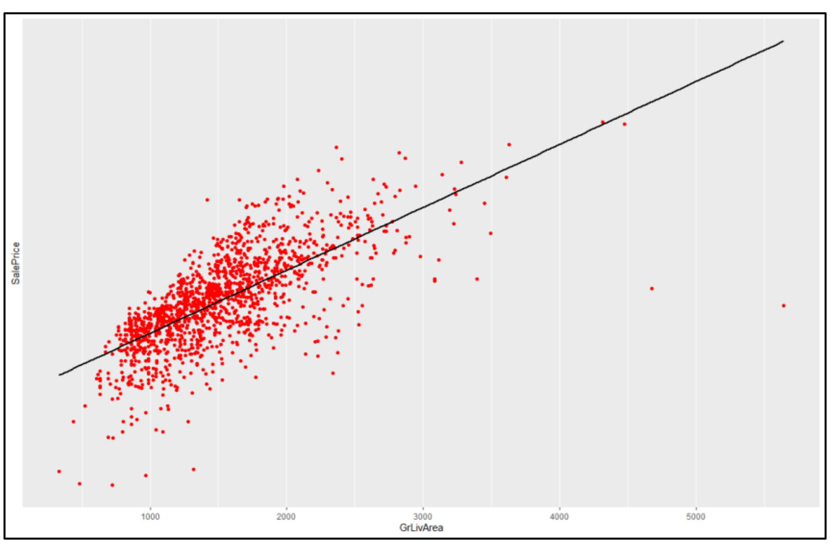
The newer the house, the lower the price based on this scatterplot, but it seems to increase after 1960's. Whereas, I expected the prices to be higher if the house was newer, so this was interesting.

Next is Overall Quality,



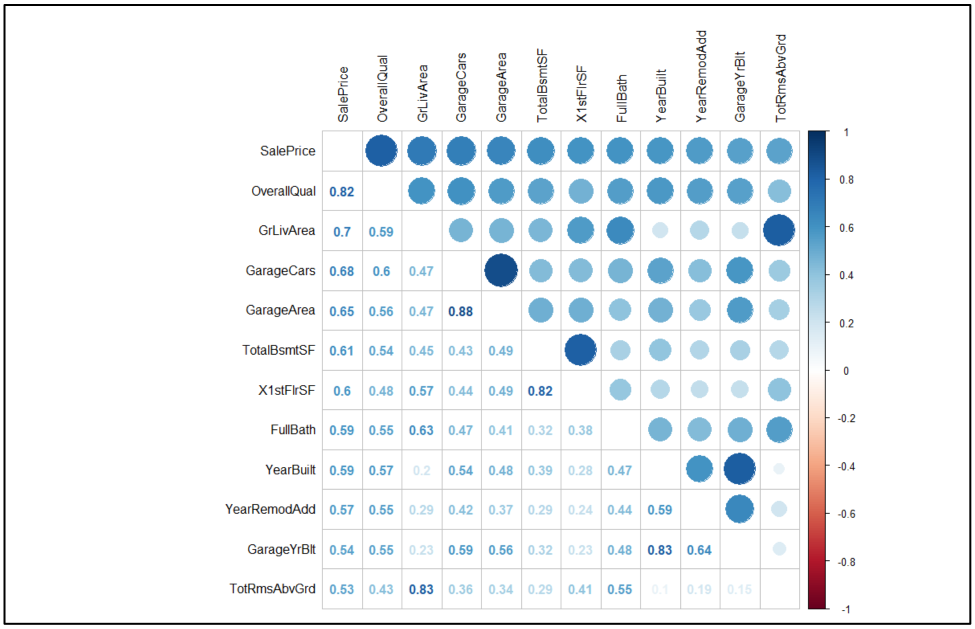
We can see that there is positive correlation between Sale Price of the house and the Overall Quality of the house. That means the Sale Price does depend on the Quality score of the house.

Let us also see GrLiveArea : Above grade (ground) living area square feet vs Sale Price. Below graph displays the relationship between them:



There is a linear relationship between them. And seems like they have high correlation. Let's verify the assumption about the correlation in the next step.

To verify it, I created a plot that shows correlation of numeric values (Correlation >0.5).



Hence, the assumption was true, Overall Quality and ground living area to have a strong correlation with the response variable.

After performing exploratory data analysis, it was noted that NA represented a category for variables - Alley, PoolQC, Fence, MiscFeature, BsmtQual, BsmtCond, BsmtExposure, BsmtFinType1, FireplaceQu, GarageType, GarageFinish, GarageQual and GarageCond, therefore it was replaced with "Unknown" so that the system would not consider it as garbage value. There were some variables with missing values (2 category variables – MasVnrType and Electrical and 3 continuous variables – LotFrontage, MAsVnrArea and GarageYrBlt). For the categorical variables, NA was replaced with "Unknown" and assigned a level. For the Continuous variables, the NA value was replaced by their mean values.

Model Planning and Building

I have used four modelling methods on the train dataset to find the strongest predictors. These methods are Linear regression, Ridge Regression, Lasso model, and Ridge/Lasso mixed model. Then according to it the strongest predictors were discovered. Below table displays the strongest predictors of each model.

Linear Model	Ridge Model	Lasso Model	Ridge-Lasso mixed model
PoolQC	OverallQual	GrLivArea	OverallQual
Utilities	GrLivArea	OverallQual	GrLivArea
Street	X1stFlrSF	PoolQC	X1stFlrSF
GarageCars	BsmtQual	PoolArea	BsmtQual
KitchenAbvGr	KitchenQual	GarageCars	KitchenQual
OverallQual	PoolQC	BsmtQual	PoolQC
ExterQual	X2ndFlrSF	KitchenQual	X2ndFlrSF
Condition2	GarageCars	ExterQual	GarageCars
BsmtQual	ExterQual	YearBuilt	ExterQual
KitchenQual	PoolArea	MasVnrArea	PoolArea
BsmtFullBath	TotRmsAbvGrd	OverallCond	TotRmsAbvGrd
RoofMatl	MasVnrArea	TotRmsAbvGrd	MasVnrArea
OverallCond	OverallCond	MSSubClass	MSSubClass
LandSlope	MSSubClass	LotArea	LotArea
MasVnrType	LotArea	BsmtExposure	YearBuilt

Comparing the output of all the above models and techniques we reached to a conclusion that the strongest predictors which can be used to develop a model are as below:

PoolQC, GrLivArea, X2ndFlrSF ,GarageCars,OverallQual,KitchenQual,BsmtQual,OverallCond,X1stFlrSF,

MSSubClass,PoolArea,ExterQual,MasVnrArea,TotRmsAbvGrd,GarageArea,LotArea.

After selecting the strongest predictors, the next step will be using caret package to fit the models.

The following techniques were used to train the "train dataset" : Linear Model, Ridge Regression, The Lasso.

1. **Linear Model Performance:** RMSE: 36852.74 and R squared: 0.7925

   (Validated- 10-fold Cross Validation)

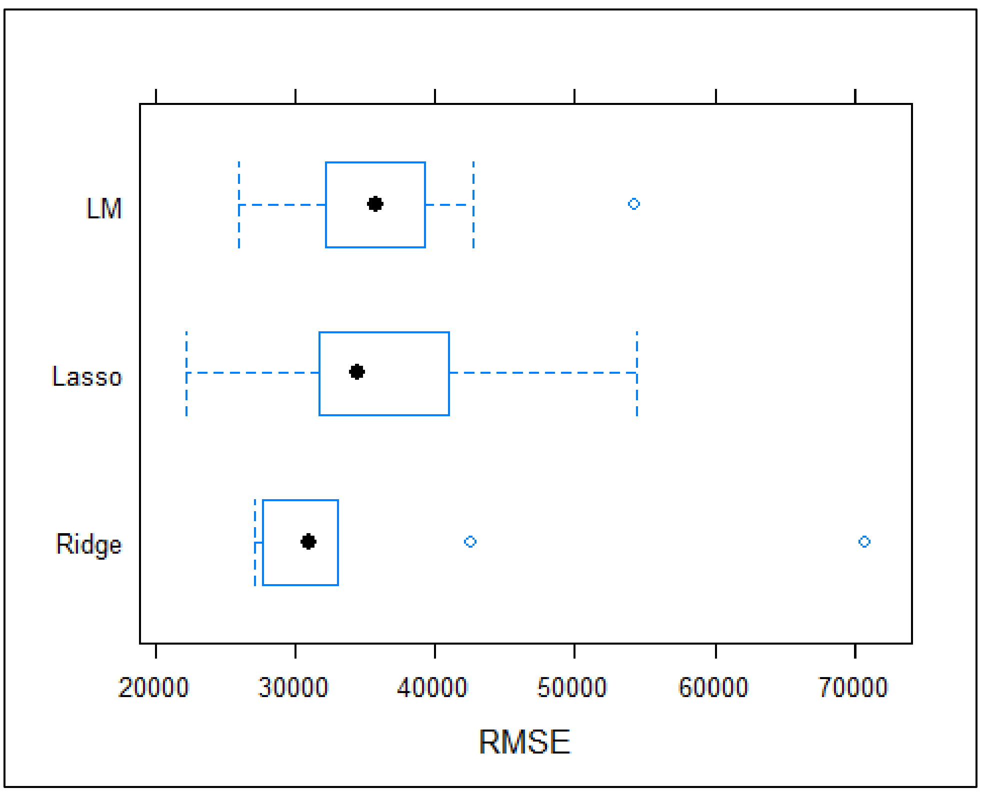
2. **Ridge Regression Performance:** RMSE: 35153.58 & R squared: 0.8155

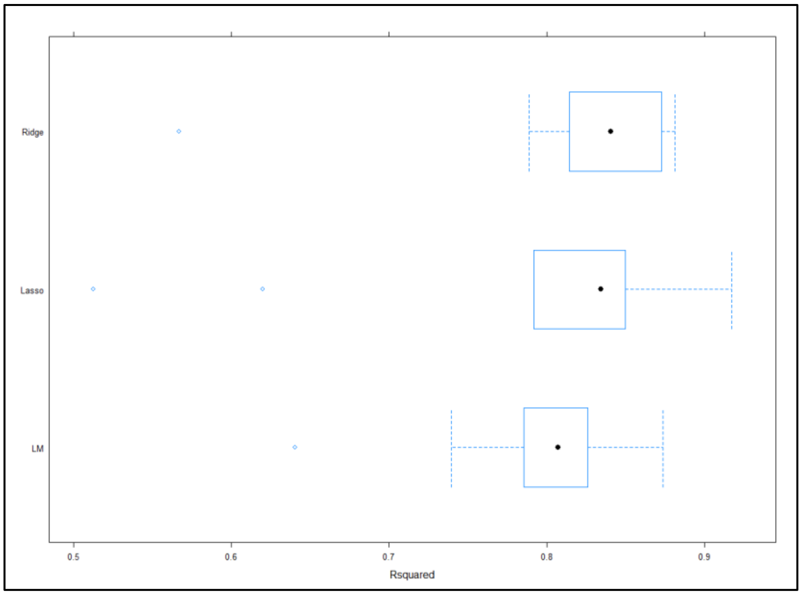
   (Validated- 10-fold Cross Validation)

3. **Lasso Model Performance:** RMSE: 36439.88 and R squared: 0.7863

   (Validated- 10- fold Cross Validated)

Below plots shows the RMSE and R-squared values(performance) of the above model:



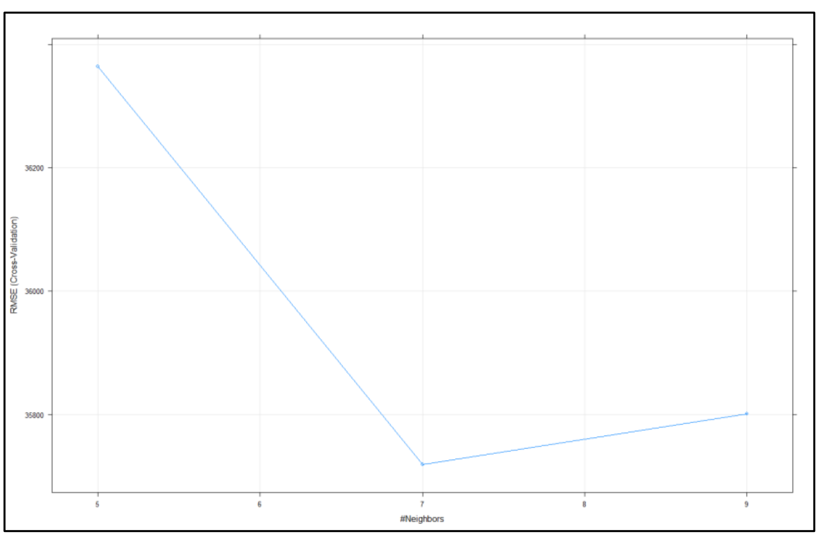


If we see in the above figure, ridge gives us the best performance, with around maximum of 0.8155 R squared value.

Hence to get better performance, additional modelling techniques were used.

K-nearest neighbor, Random forest and M5P were used to train the data, and the results were better.

4. **KNN Model Performance:** RMSE: 35718.86 & R- squared :0.8026 (Validated by 10-fold Cross Validation)



In the above figure we can see that when K=7, the model performs the best, with a least RMSE value.

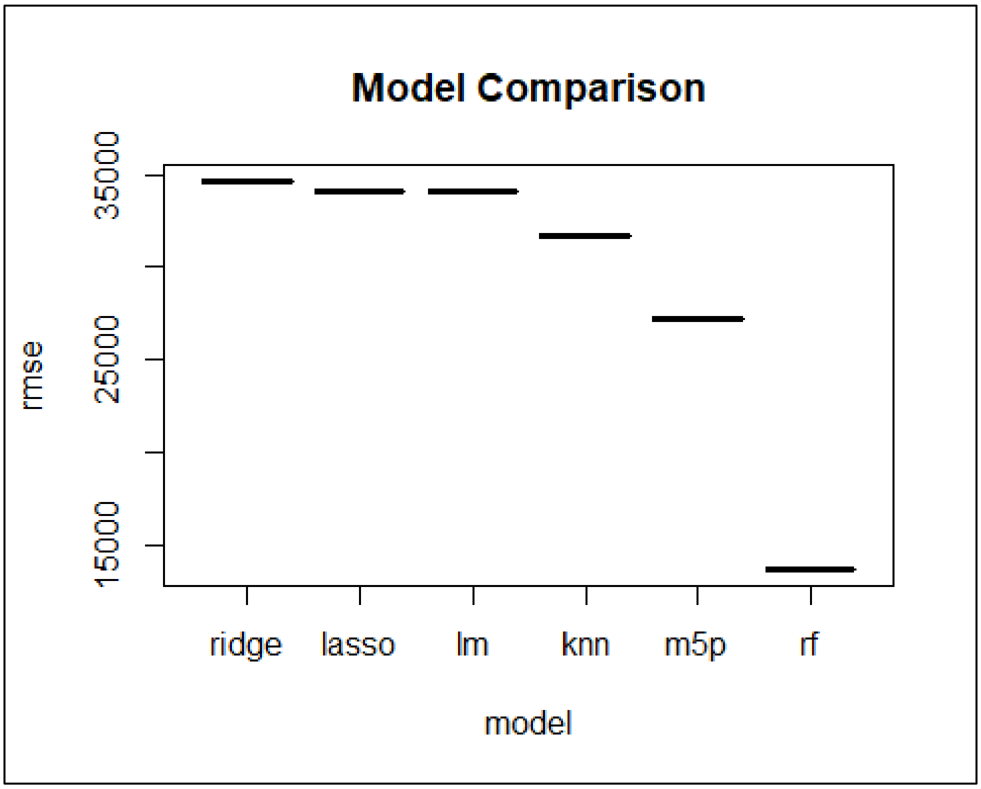
Random Forest: RMSE : 13728.89 (Validated by 10-fold Cross Validation)

M5P Model: M5P is based on Quinlan's M5 algorithm for inducing trees of regression models. M5P combines a conventional decision tree with the possibility of linear regression functions at the nodes.

Performance: RMSE : 27274.42

Results and Performance

We have applied different modeling methods on the strongest predictors to find the most accurate model. These methods included Linear, Ridge, Lasso, KNN, Random forest and M5P. The below plot compares the RMSEs for all the models and shows the minimum and maximum RMSE of the models.



Hence, according to the results, the best model would be the Random Forest, with the lowest RMSE value. So, we will use this model to predict the "SalePrice" for the test data set.

Discussion and Recommendations

The following would be my recommendation:

1. I think the dataset is huge, and the analyst needs to spend more time in understanding various variables so that a better model can be created, I did not have enough time, but in future I would like to do that.
2. Techniques like GBM, XGBoost, Boosting should also be performed on the train data set, they might give a better result.
3. We can also use splines, step function, gam, etc. to understand the dataset and for the better performance and prediction.

Appendix

Project_MIS749

Vaishnavi Bihare

5/5/2020

Load the Packages

Importing the train and test data

```
train <- read.csv("train.csv", stringsAsFactors = FALSE)
test <- read.csv("test.csv",stringsAsFactors = FALSE)
```

1. Data Inspection, Data Cleaning and Data Modelling

Structure of the data

```
dim(train)

## [1] 1460   81

str(train)

## 'data.frame':    1460 obs. of  81 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : chr  "RL" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7
420 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "Reg" "IR1" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "FR2" "Inside" "Corner" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "CollgCr" "Veenker" "CollgCr" "Crawfor" ...
##  $ Condition1   : chr  "Norm" "Feedr" "Norm" "Norm" ...
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "2Story" "1Story" "2Story" "2Story" ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 .
..
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 .
```

```
..
## $ RoofStyle    : chr  "Gable" "Gable" "Gable" "Gable" ...
## $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
## $ Exterior1st  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Sdng" ...
## $ Exterior2nd  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Shng" ...
## $ MasVnrType   : chr  "BrkFace" "None" "BrkFace" "None" ...
## $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
## $ ExterQual    : chr  "Gd" "TA" "Gd" "TA" ...
## $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
## $ Foundation   : chr  "PConc" "CBlock" "PConc" "BrkTil" ...
## $ BsmtQual     : chr  "Gd" "Gd" "Gd" "TA" ...
## $ BsmtCond     : chr  "TA" "TA" "TA" "Gd" ...
## $ BsmtExposure : chr  "No" "Gd" "Mn" "No" ...
## $ BsmtFinType1 : chr  "GLQ" "ALQ" "GLQ" "ALQ" ...
## $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
## $ BsmtFinType2 : chr  "Unf" "Unf" "Unf" "Unf" ...
## $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
## $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
## $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
## $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
## $ HeatingQC    : chr  "Ex" "Ex" "Ex" "Gd" ...
## $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
## $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
## $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
## $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
## $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
## $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 .
..
## $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
## $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
## $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
## $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
## $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
## $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
## $ KitchenQual  : chr  "Gd" "TA" "Gd" "Gd" ...
## $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
## $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
## $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
## $ FireplaceQu  : chr  NA "TA" "TA" "Gd" ...
## $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Detchd" ...
## $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 .
..
## $ GarageFinish : chr  "RFn" "RFn" "RFn" "Unf" ...
## $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
## $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
## $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
## $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
## $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
## $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
## $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
```

```
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : chr  NA NA NA NA ...
##  $ Fence        : chr  NA NA NA NA ...
##  $ MiscFeature  : chr  NA NA NA NA ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 .
..
##  $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
##  $ SaleCondition: chr  "Normal" "Normal" "Normal" "Abnorml" ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 20
0000 129900 118000 ...
```

```
#Percentage of Missing data in train dataset
sum(is.na(train)) / (nrow(train) *ncol(train))
```

```
## [1] 0.05889565
```

```
#Checking for duplicate row
cat("The number of duplicated rows are", nrow(train) - nrow(unique(train)))
```

```
## The number of duplicated rows are 0
```

```
#The housing train data set has 1460 rows and 81 features with the target fea
ture Sale Price.

dim(test)
```

```
## [1] 1459   80
```

```
str(test)
```

```
## 'data.frame':    1459 obs. of  80 variables:
##  $ Id           : int  1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 .
..
##  $ MSSubClass   : int  20 20 60 60 120 60 20 60 20 20 ...
##  $ MSZoning     : chr  "RH" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  80 81 74 78 43 75 NA 63 85 70 ...
##  $ LotArea      : int  11622 14267 13830 9978 5005 10000 7980 8402 10176 8
400 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "IR1" "IR1" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "Corner" "Inside" "Inside" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "NAmes" "NAmes" "Gilbert" "Gilbert" ...
##  $ Condition1   : chr  "Feedr" "Norm" "Norm" "Norm" ...
```

```
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "1Story" "1Story" "2Story" "2Story" ...
##  $ OverallQual  : int  5 6 5 6 8 6 6 6 7 4 ...
##  $ OverallCond  : int  6 6 5 6 5 5 7 5 5 5 ...
##  $ YearBuilt    : int  1961 1958 1997 1998 1992 1993 1992 1998 1990 1970 .
..
##  $ YearRemodAdd : int  1961 1958 1998 1998 1992 1994 2007 1998 1990 1970 .
..
##  $ RoofStyle    : chr  "Gable" "Hip" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "VinylSd" "Wd Sdng" "VinylSd" "VinylSd" ...
##  $ Exterior2nd  : chr  "VinylSd" "Wd Sdng" "VinylSd" "VinylSd" ...
##  $ MasVnrType   : chr  "None" "BrkFace" "None" "BrkFace" ...
##  $ MasVnrArea   : int  0 108 0 20 0 0 0 0 0 0 ...
##  $ ExterQual    : chr  "TA" "TA" "TA" "TA" ...
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "CBlock" "CBlock" "PConc" "PConc" ...
##  $ BsmtQual     : chr  "TA" "TA" "Gd" "TA" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "TA" ...
##  $ BsmtExposure : chr  "No" "No" "No" "No" ...
##  $ BsmtFinType1 : chr  "Rec" "ALQ" "GLQ" "GLQ" ...
##  $ BsmtFinSF1   : int  468 923 791 602 263 0 935 0 637 804 ...
##  $ BsmtFinType2 : chr  "LwQ" "Unf" "Unf" "Unf" ...
##  $ BsmtFinSF2   : int  144 0 0 0 0 0 0 0 0 78 ...
##  $ BsmtUnfSF    : int  270 406 137 324 1017 763 233 789 663 0 ...
##  $ TotalBsmtSF  : int  882 1329 928 926 1280 763 1168 789 1300 882 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "TA" "TA" "Gd" "Ex" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
##  $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
##  $ X1stFlrSF    : int  896 1329 928 926 1280 763 1187 789 1341 882 ...
##  $ X2ndFlrSF    : int  0 0 701 678 0 892 0 676 0 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  896 1329 1629 1604 1280 1655 1187 1465 1341 882 ...
##  $ BsmtFullBath : int  0 0 0 0 0 0 1 0 1 1 ...
##  $ BsmtHalfBath : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  1 1 2 2 2 2 2 2 1 1 ...
##  $ HalfBath     : int  0 1 1 1 0 1 0 1 1 0 ...
##  $ BedroomAbvGr : int  2 3 3 3 2 3 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ KitchenQual  : chr  "TA" "Gd" "TA" "Gd" ...
##  $ TotRmsAbvGrd : int  5 6 6 7 5 7 6 7 5 4 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  0 0 1 1 0 1 0 1 1 0 ...
##  $ FireplaceQu  : chr  NA NA "TA" "Gd" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Attchd" ...
##  $ GarageYrBlt  : int  1961 1958 1997 1998 1992 1993 1992 1998 1990 1970 .
..
##  $ GarageFinish : chr  "Unf" "Unf" "Fin" "Fin" ...
```

```
## $ GarageCars   : int  1 1 2 2 2 2 2 2 2 2 ...
## $ GarageArea   : int  730 312 482 470 506 440 420 393 506 525 ...
## $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
## $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
## $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
## $ WoodDeckSF   : int  140 393 212 360 0 157 483 0 192 240 ...
## $ OpenPorchSF  : int  0 36 34 36 82 84 21 75 0 0 ...
## $ EnclosedPorch: int  0 0 0 0 0 0 0 0 0 0 ...
## $ X3SsnPorch   : int  0 0 0 0 0 0 0 0 0 0 ...
## $ ScreenPorch  : int  120 0 0 0 144 0 0 0 0 0 ...
## $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
## $ PoolQC       : chr  NA NA NA NA ...
## $ Fence        : chr  "MnPrv" NA "MnPrv" NA ...
## $ MiscFeature  : chr  NA "Gar2" NA NA ...
## $ MiscVal      : int  0 12500 0 0 0 0 500 0 0 0 ...
## $ MoSold       : int  6 6 3 6 1 4 3 5 2 4 ...
## $ YrSold       : int  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010 .
..
## $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
## $ SaleCondition: chr  "Normal" "Normal" "Normal" "Normal" ...

#Percentage of missing data in test dataset
sum(is.na(test)) / (nrow(test) * ncol(test))

## [1] 0.05997258
```

Combine them together (Later use)

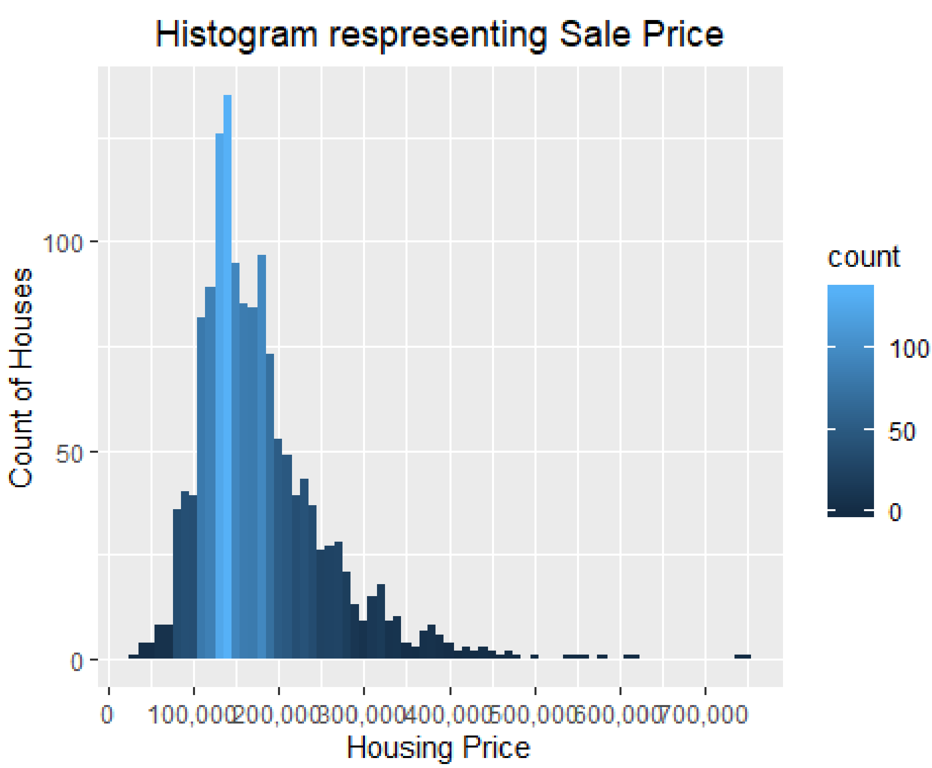
```
test$SalePrice<-rep(NA,1459)

train$isTrain <- 1

test$isTrain <- 0

house_data <- rbind(train,test)
```

Hence we have 2919 rows and 82 colums in total

Understanding the data

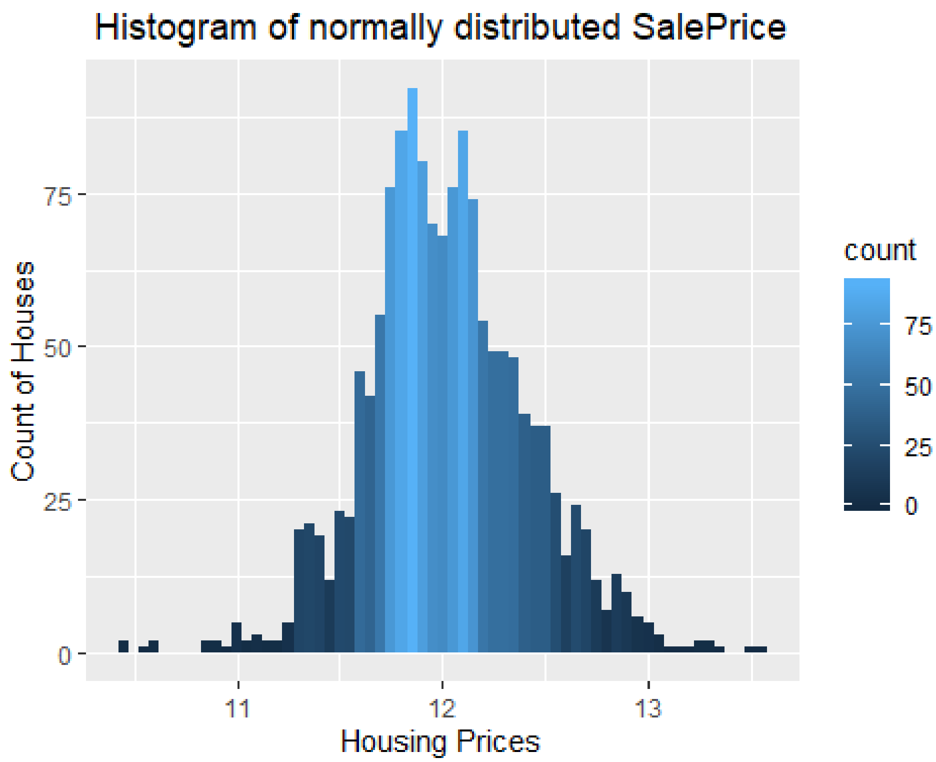
```
numeric_variables <- which(sapply(train, is.numeric))
numeric_headers <- names(numeric_variables)
cat('Here we have', length(numeric_variables),'numeric variables')

## Here we have 39 numeric variables

character_variables <- which(sapply(train, is.character))
character_header <- names(character_variables)
cat('There are', length(character_variables),'character variables')

## There are 43 character variables
```

```
summary(train$SalePrice)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   34900  129975  163000  180921  214000  755000

#Understanding the Variables

#Response Variable SalePrice

options(scipen=10000)
ggplot(data = train[!is.na(train$SalePrice),], aes(x=SalePrice, fill= ..count
..))+
  geom_histogram(binwidth = 10000)+
  scale_x_continuous(breaks = seq(0,800000, by=100000), labels= comma)+
  ggtitle("Histogram respresenting Sale Price")+
  xlab("Housing Price")+
  ylab("Count of Houses")+
  theme(plot.title = element_text(hjust =0.5))
```

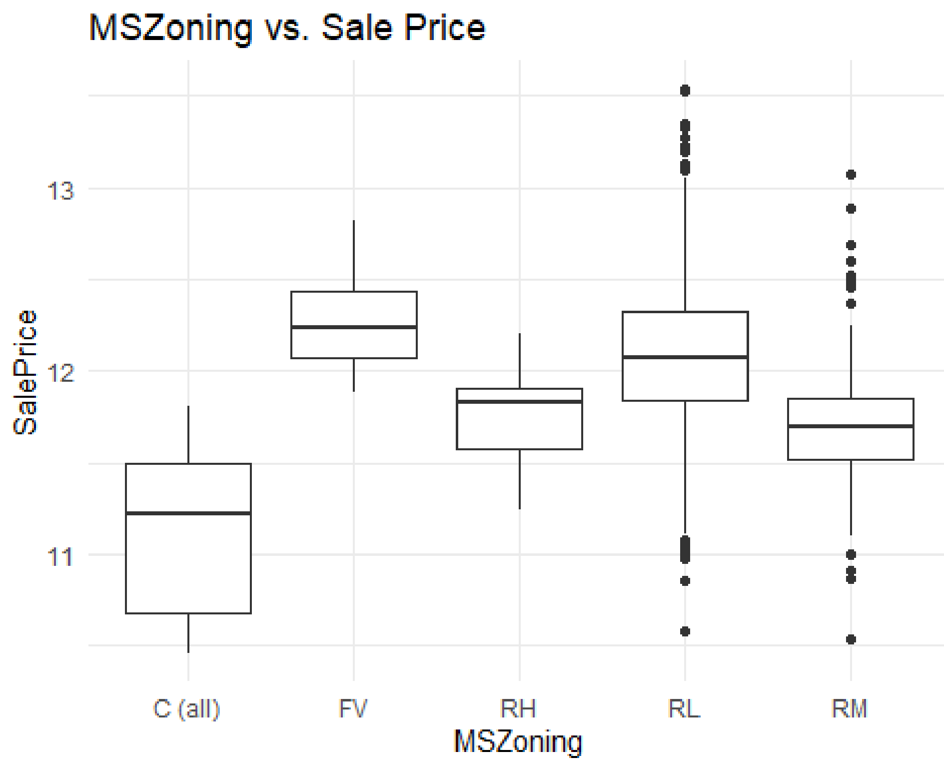



```
#It is right skewed, lets take the log to make it normally distributed

train$SalePrice <- log(train$SalePrice )

ggplot(train, aes(x=SalePrice, fill=..count..))+geom_histogram(binwidth = 0.0
5)+
  ggtitle("Histogram of normally distributed SalePrice")+
  ylab("Count of Houses")+
```

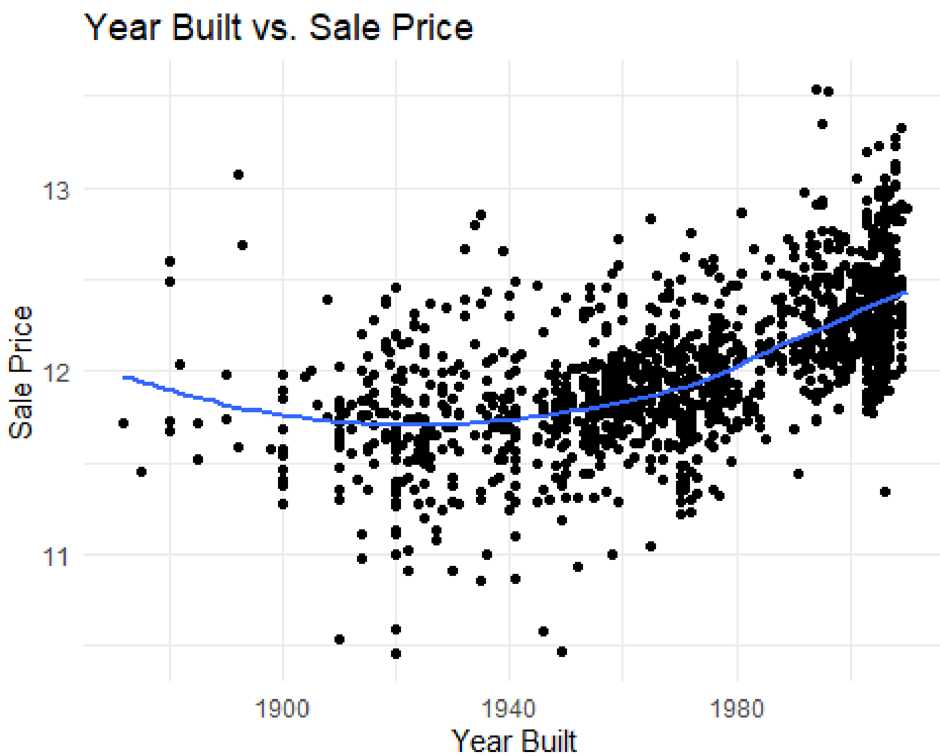
```
  xlab("Housing Prices")+
  theme(plot.title = element_text(hjust = 0.5))
```



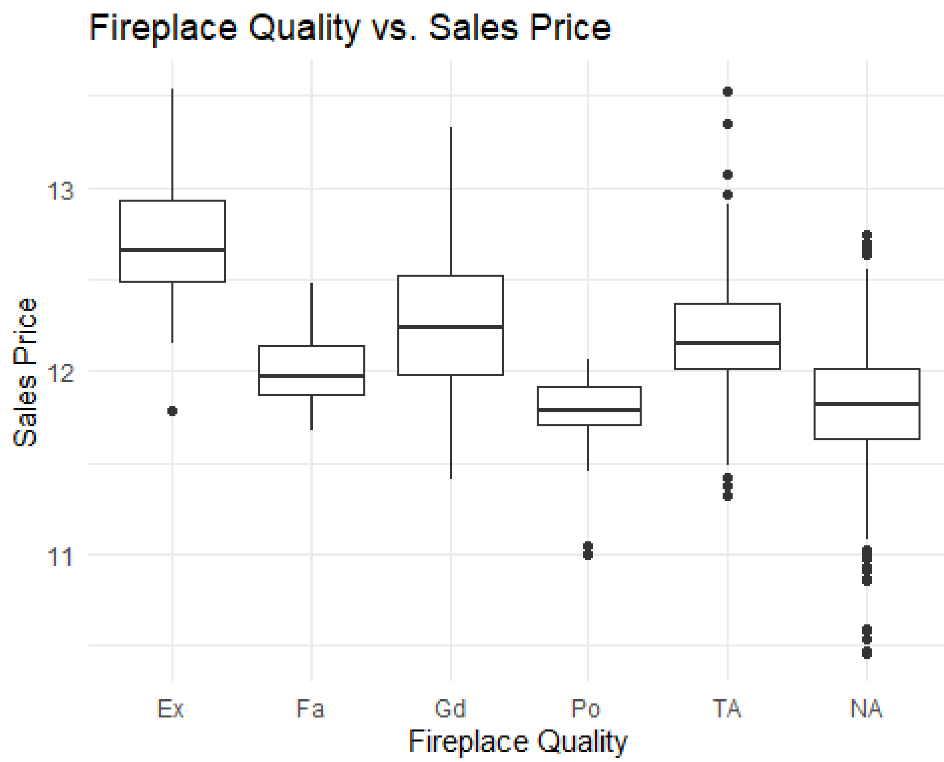

```
#MSZoning vs SalePrice
ggplot(data = train[!is.na(train$MSZoning) & train$SalePrice > 0,], aes(x = M
SZoning, y = SalePrice)) +
  geom_boxplot(na.rm = T) +
  theme_minimal() +
  labs(x = "MSZoning",
       y = "SalePrice",
       title = "MSZoning vs. Sale Price")
```



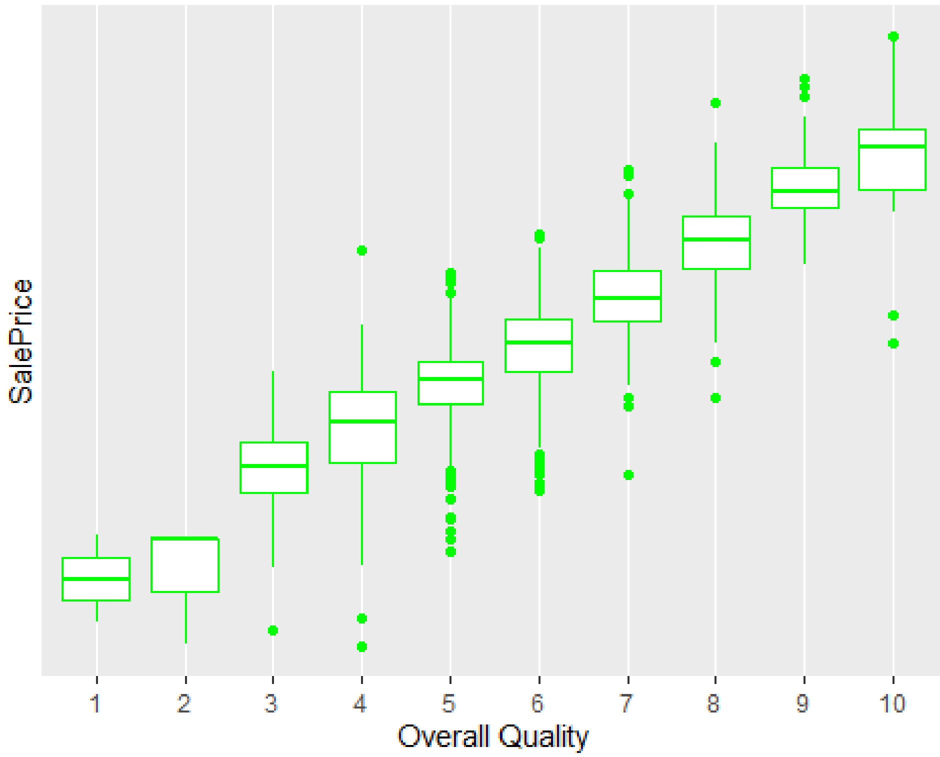

```r
#Year Built and SalePrice
ggplot(data = train[train$SalePrice>0,], aes(x = YearBuilt, y = SalePrice)) +
  geom_point(na.rm = TRUE) +
  geom_smooth(method = "loess", se = FALSE, na.rm = TRUE) +
  theme_minimal() +
  labs(x = "Year Built",
       y = "Sale Price",
       title = "Year Built vs. Sale Price")
```



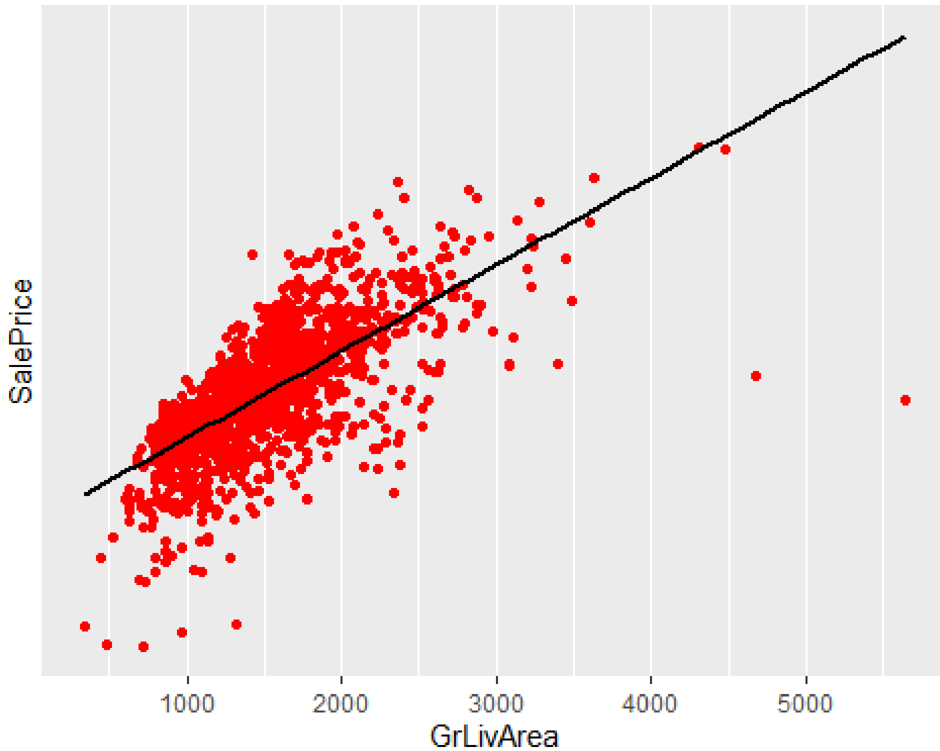

```
#Fireplace vs SalePrice
ggplot(data = train[train$SalePrice>0,], aes(x = FireplaceQu, y = SalePrice))
+
  geom_boxplot(na.rm = T) +
  theme_minimal() +
  labs(x = "Fireplace Quality",
       y = "Sales Price",
       title = "Fireplace Quality vs. Sales Price")
```



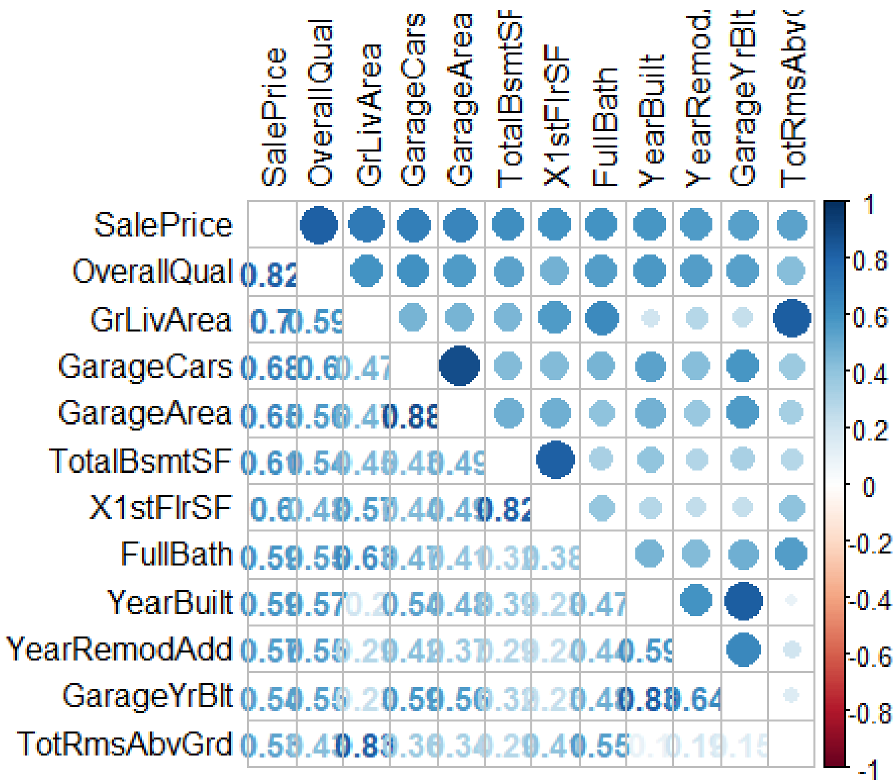

```
#We can see NA's, we'll remove them later

#Let's do overallQual next, as it is impotant to understand the how sale pric
e is affected by the Overall Quality

ggplot(data=train[!is.na(train$SalePrice),], aes(x=factor(OverallQual),y=Sale
Price))+
  geom_boxplot(color='green')+
  scale_y_continuous(breaks = seq(0, 800000, by=100000), labels = comma)+
  labs(x="Overall Quality")
```




```
#we can see positive correlation here

#Next is  Above grade living area

ggplot(data=train[!is.na(train$SalePrice),], aes(x=GrLivArea,y=SalePrice))+
  geom_point(col='red')+ geom_smooth(method = "lm", se= FALSE, color='black',
aes(group=1))+
  scale_y_continuous(breaks = seq(0, 800000, by=100000), labels = comma)
```




```
#we can see positive correlation here also, lets verify it by a correlation p
lot

#Correlation of the numeric values

data_nv <- train[,numeric_variables]
cor_va <- cor(data_nv, use="pairwise.complete.obs")

## Warning in cor(data_nv, use = "pairwise.complete.obs"): the standard devia
tion
## is zero

#Sorting the values with the decreasing correlation with the SalePrice
sorted_cor <- as.matrix(sort(cor_va[,'SalePrice'], decreasing = TRUE))
high_cor <- names(which(apply(sorted_cor, 1, function(x) abs(x)>0.5)))
cor_va <- cor_va[high_cor,high_cor]

corrplot.mixed(cor_va, tl.col='Black', tl.pos="lt")
```



Data Cleaning

```r
#Dropping ID as it is not necessary for prediction
house_data= house_data[-1]
#view(house_data)

# To find number of missing value for all variable in combined dataset (Train
+Test)

sapply(house_data[,1:80], function(x) sum(is.na(x)))
```

```
##     MSSubClass      MSZoning   LotFrontage       LotArea        Street
##              0             4           486             0             0
##          Alley      LotShape   LandContour     Utilities     LotConfig
##           2721             0             0             2             0
##      LandSlope  Neighborhood    Condition1    Condition2      BldgType
##              0             0             0             0             0
##     HouseStyle   OverallQual   OverallCond     YearBuilt  YearRemodAdd
##              0             0             0             0             0
##      RoofStyle      RoofMatl   Exterior1st   Exterior2nd    MasVnrType
##              0             0             1             1            24
##     MasVnrArea     ExterQual     ExterCond    Foundation      BsmtQual
##             23             0             0             0            81
##       BsmtCond  BsmtExposure  BsmtFinType1    BsmtFinSF1  BsmtFinType2
##             82            82            79             1            80
##     BsmtFinSF2     BsmtUnfSF   TotalBsmtSF       Heating     HeatingQC
##              1             1             1             0             0
```

```
##    CentralAir    Electrical      X1stFlrSF      X2ndFlrSF   LowQualFinSF
##             0             1              0              0              0
##     GrLivArea  BsmtFullBath   BsmtHalfBath       FullBath       HalfBath
##             0             2              2              0              0
##  BedroomAbvGr  KitchenAbvGr    KitchenQual   TotRmsAbvGrd     Functional
##             0             0              1              0              2
##    Fireplaces   FireplaceQu     GarageType    GarageYrBlt   GarageFinish
##             0          1420            157            159            159
##    GarageCars    GarageArea     GarageQual     GarageCond     PavedDrive
##             1             1            159            159              0
##    WoodDeckSF   OpenPorchSF  EnclosedPorch     X3SsnPorch    ScreenPorch
##             0             0              0              0              0
##      PoolArea        PoolQC          Fence    MiscFeature        MiscVal
##             0          2909           2348           2814              0
##        MoSold        YrSold       SaleType  SaleCondition      SalePrice
##             0             0              1              0           1459
```

```
house_data$Alley [is.na(house_data$Alley)] <- "Unknown"
house_data$MasVnrType[is.na(house_data$MasVnrType)] <- "Unknown"
house_data$BsmtQual[is.na(house_data$BsmtQual)] <- "Unknown"
house_data$BsmtCond[is.na(house_data$BsmtCond)] <- "Unknown"
house_data$BsmtExposure[is.na(house_data$BsmtExposure)] <- "Unknown"
house_data$BsmtFinType1[is.na(house_data$BsmtFinType1)] <- "Unknown"
house_data$BsmtFinType2[is.na(house_data$BsmtFinType2)] <- "Unknown"
house_data$Electrical[is.na(house_data$Electrical)] <- "Unknown"
house_data$FireplaceQu[is.na(house_data$FireplaceQu)] <- "Unknown"
house_data$GarageType[is.na(house_data$GarageType)] <- "Unknown"
house_data$GarageFinish[is.na(house_data$GarageFinish)] <- "Unknown"
house_data$GarageQual[is.na(house_data$GarageQual)] <- "Unknown"
house_data$GarageCond[is.na(house_data$GarageCond)] <- "Unknown"
house_data$PoolQC[is.na(house_data$PoolQC)] <- "Unknown"
house_data$Fence[is.na(house_data$Fence)] <- "Unknown"
house_data$MiscFeature[is.na(house_data$MiscFeature)] <- "Unknown"

house_data$LotFrontage[is.na(house_data$LotFrontage)] <- round(mean(house_dat
a$LotFrontage,na.rm=TRUE))
house_data$MasVnrArea[is.na(house_data$MasVnrArea)] <- round(mean(house_data$
MasVnrArea,na.rm=TRUE))
house_data$GarageYrBlt[is.na(house_data$GarageYrBlt)] <- round(mean(house_dat
a$GarageYrBlt,na.rm=TRUE))


elements <- names(house_data)
elements <- elements[elements != "SalePrice"]
for(i in elements)
{
  if(is.character(house_data[[i]]))
  {
    levels <- sort(unique(c(house_data[[i]])))
    house_data[[i]] <- factor(house_data[[i]],levels=levels)
```

```r
  }
}

for (i in elements)
{
  if(class(levels(house_data[[i]])) == "character")
    house_data[[i]] <- seq_along(levels(house_data[[i]]))[house_data[[i]]]
}

str(house_data)
```

```
## 'data.frame':    2919 obs. of  81 variables:
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : int  4 4 4 4 4 4 4 4 5 4 ...
##  $ LotFrontage  : num  65 80 68 60 84 85 75 69 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7
420 ...
##  $ Street       : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Alley        : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ LotShape     : int  4 4 1 1 1 1 4 1 4 4 ...
##  $ LandContour  : int  4 4 4 4 4 4 4 4 4 4 ...
##  $ Utilities    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ LotConfig    : int  5 3 5 1 3 5 5 1 5 1 ...
##  $ LandSlope    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Neighborhood : int  6 25 6 7 14 12 21 17 18 4 ...
##  $ Condition1   : int  3 2 3 3 3 3 3 5 1 1 ...
##  $ Condition2   : int  3 3 3 3 3 3 3 3 3 1 ...
##  $ BldgType     : int  1 1 1 1 1 1 1 1 1 2 ...
##  $ HouseStyle   : int  6 3 6 6 6 1 3 6 1 2 ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 .
..
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 .
..
##  $ RoofStyle    : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ RoofMatl     : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Exterior1st  : int  13 9 13 14 13 13 13 7 4 9 ...
##  $ Exterior2nd  : int  14 9 14 16 14 14 14 7 16 9 ...
##  $ MasVnrType   : int  2 3 2 3 2 3 4 4 3 3 ...
##  $ MasVnrArea   : num  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : int  3 4 3 4 3 4 3 4 4 4 ...
##  $ ExterCond    : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ Foundation   : int  3 2 3 1 3 6 3 2 1 1 ...
##  $ BsmtQual     : int  3 3 3 4 3 3 1 3 4 4 ...
##  $ BsmtCond     : int  4 4 4 2 4 4 4 4 4 4 ...
##  $ BsmtExposure : int  4 2 3 4 1 4 1 3 4 4 ...
##  $ BsmtFinType1 : int  3 1 3 1 3 3 3 1 6 3 ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : int  6 6 6 6 6 6 6 2 6 6 ...
```

```
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ HeatingQC    : int  1 1 1 3 1 1 1 1 3 1 ...
##  $ CentralAir   : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Electrical   : int  5 5 5 5 5 5 5 5 2 5 ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 .
..
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : int  3 4 3 3 3 4 3 4 4 4 ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : int  7 7 7 7 7 7 7 7 3 7 ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : int  6 5 5 3 5 6 3 5 5 5 ...
##  $ GarageType   : int  2 2 2 6 2 2 2 2 6 2 ...
##  $ GarageYrBlt  : num  2003 1976 2001 1998 2000 ...
##  $ GarageFinish : int  2 2 2 3 2 3 2 2 3 2 ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : int  5 5 5 5 5 5 5 5 2 3 ...
##  $ GarageCond   : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ PavedDrive   : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : int  4 4 4 4 4 4 4 4 4 4 ...
##  $ Fence        : int  5 5 5 5 5 3 5 5 5 5 ...
##  $ MiscFeature  : int  5 5 5 5 5 3 5 3 5 5 ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 .
..
##  $ SaleType     : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ SaleCondition: int  5 5 5 1 5 5 5 5 1 5 ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 20
0000 129900 118000 ...
##  $ isTrain      : num  1 1 1 1 1 1 1 1 1 1 ...
```

```r
#rmse
rmse <- function(actualVal, predictedVal)
{
  sqrt(mean((actualVal - predictedVal)^2))
}
```

Model Creation

```r
#Seting the seed to make the output reproduceable
set.seed(100)

#Creating test and train data
train <- house_data[house_data$isTrain==1,]

test <- house_data[house_data$isTrain==0,]

#Model Development

linearmodel <- lm(SalePrice ~ . , data=train)

varImp(linearmodel)

##                  Overall
## MSSubClass    2.58542574
## MSZoning      1.05977696
## LotFrontage   3.34358050
## LotArea       4.12280195
## Street        2.15138849
## Alley         1.81728926
## LotShape      1.42338481
## LandContour   2.49276242
## Utilities     1.63711567
## LotConfig     0.04029703
## LandSlope     1.30243965
## Neighborhood  1.70091898
## Condition1    1.12359443
## Condition2    2.87162918
## BldgType      1.78083850
## HouseStyle    1.42538480
## OverallQual   9.23222256
## OverallCond   4.91008682
## YearBuilt     2.77318196
## YearRemodAdd  0.28611233
## RoofStyle     1.70009434
## RoofMatl      3.73436320
## Exterior1st   2.32365699
## Exterior2nd   1.15190680
## MasVnrType    2.93595006
## MasVnrArea    5.54935598
## ExterQual     5.02218566
```

```
## ExterCond     0.62735606
## Foundation    0.82999512
## BsmtQual      6.15126857
## BsmtCond      1.91468463
## BsmtExposure  4.09270230
## BsmtFinType1  0.96585189
## BsmtFinSF1    1.51683470
## BsmtFinType2  0.76809038
## BsmtFinSF2    1.15157481
## BsmtUnfSF     0.32220968
## Heating       0.59963687
## HeatingQC     1.35536386
## CentralAir    0.31073250
## Electrical    0.48390165
## X1stFlrSF     8.69718318
## X2ndFlrSF     9.11748090
## LowQualFinSF  0.33204132
## BsmtFullBath  2.65177373
## BsmtHalfBath  0.28937694
## FullBath      1.33769865
## HalfBath      0.05895027
## BedroomAbvGr  2.10870490
## KitchenAbvGr  2.33714424
## KitchenQual   6.06817548
## TotRmsAbvGrd  3.06073118
## Functional    4.38494148
## Fireplaces    1.29507549
## FireplaceQu   1.38782324
## GarageType    1.51200488
## GarageYrBlt   0.70335323
## GarageFinish  0.14641664
## GarageCars    4.41656537
## GarageArea    0.36756382
## GarageQual    0.22409443
## GarageCond    1.44417954
## PavedDrive    0.47444662
## WoodDeckSF    2.96572675
## OpenPorchSF   0.17778525
## EnclosedPorch 0.08124308
## X3SsnPorch    1.07936769
## ScreenPorch   2.86800960
## PoolArea      5.98560249
## PoolQC        6.37784754
## Fence         0.27059305
## MiscFeature   0.78581593
## MiscVal       0.33455145
## MoSold        0.65179414
## YrSold        1.50531850
## SaleType      1.10105784
## SaleCondition 3.74727086
```

```r
head(sort(abs(linearmodel$coefficients),decreasing = TRUE),n=16)
```

```
##  (Intercept)       PoolQC    Utilities       Street   GarageCars KitchenAb
vGr
##  1923197.358    84653.789    53419.807    29769.487    11913.315    11627.
733
##  OverallQual    ExterQual   Condition2     BsmtQual  KitchenQual BsmtFullB
ath
##    10781.284     9752.369     9461.616     8584.389     8556.774     6406.
135
##     RoofMatl  OverallCond    LandSlope   MasVnrType
##     5532.875     5072.054     4910.317     4399.851
```

```r
#Hence the strongest predictors from linear model are: Pool Quality, Types of
Utilities available,
#Street, Garage Cars, Overall Quality, External Quality, Condition2, BsmtQual
, Kitched Qual, BsmtFullBath,
#RoofMatl, LandSlope, Heating, Overall Condition

#now lets do Ridge Regression

ridge_model <- train(SalePrice ~ . , data=train, preProcess= c("center", "sca
le"),method = "glmnet",
                     tuneGrid= expand.grid(alpha=0,lambda = seq(0,10, .1)))

varImp(ridge_model)
```

```
## glmnet variable importance
##
##   only 20 most important variables shown (out of 80)
##
##              Overall
## OverallQual   100.00
## GrLivArea      81.62
## X1stFlrSF      56.16
## BsmtQual       54.08
## KitchenQual    52.24
## PoolQC         51.69
## X2ndFlrSF      51.04
## GarageCars     50.07
## ExterQual      48.59
## PoolArea       45.29
## TotRmsAbvGrd   43.19
## MasVnrArea     42.45
## OverallCond    34.48
## MSSubClass     30.55
## LotArea        28.77
## YearBuilt      28.63
## BsmtExposure   28.60
## Functional     26.55
```

```
## RoofMatl       24.21
## FullBath       24.05

#laaso

lassoModel <- train(SalePrice ~ ., data = train, preProcess = c("center", "sc
ale"),method = "glmnet",
                    tuneGrid= expand.grid(alpha=1,lambda = seq(0,10, .1)))

varImp(lassoModel)

## glmnet variable importance
##
##   only 20 most important variables shown (out of 80)
##
##                Overall
## GrLivArea      100.00
## OverallQual     65.88
## PoolQC          49.84
## PoolArea        46.05
## GarageCars      36.82
## BsmtQual        33.81
## KitchenQual     31.23
## ExterQual       29.48
## YearBuilt       27.09
## MasVnrArea      26.05
## OverallCond     24.23
## TotRmsAbvGrd    24.08
## MSSubClass      22.13
## LotArea         18.78
## BsmtExposure    18.03
## Functional      17.28
## LotFrontage     15.31
## Exterior1st     15.14
## BsmtFinSF1      14.89
## BsmtFullBath    14.58

#lasso and ridge model(Mix model)
mixModel <- train(SalePrice ~ ., data = train, preProcess = c("center", "scal
e"),method = "glmnet",
                  tuneGrid= expand.grid(alpha=0:1,lambda = seq(0,10, .1)))

varImp(mixModel)

## glmnet variable importance
##
##   only 20 most important variables shown (out of 80)
##
##                Overall
## OverallQual    100.00
## GrLivArea       81.62
```

```
## X1stFlrSF      56.16
## BsmtQual       54.08
## KitchenQual    52.24
## PoolQC         51.69
## X2ndFlrSF      51.04
## GarageCars     50.07
## ExterQual      48.59
## PoolArea       45.29
## TotRmsAbvGrd   43.19
## MasVnrArea     42.45
## OverallCond    34.48
## MSSubClass     30.55
## LotArea        28.77
## YearBuilt      28.63
## BsmtExposure   28.60
## Functional     26.55
## RoofMatl       24.21
## FullBath       24.05
```

Comparing the output of all the above models and techniques we reached to a conclusion that the strongest predictors which can be used to develop a model are: PoolQC,GrLivArea, X2stFlrSF,GarageCars,OverallQual,KitchenQual,BsmtQual,OverallCond,X1stFlrSF, MSSubClass,PoolArea,ExterQual,MasVnrArea,TotRmsAbvGrd,GarageArea,LotArea

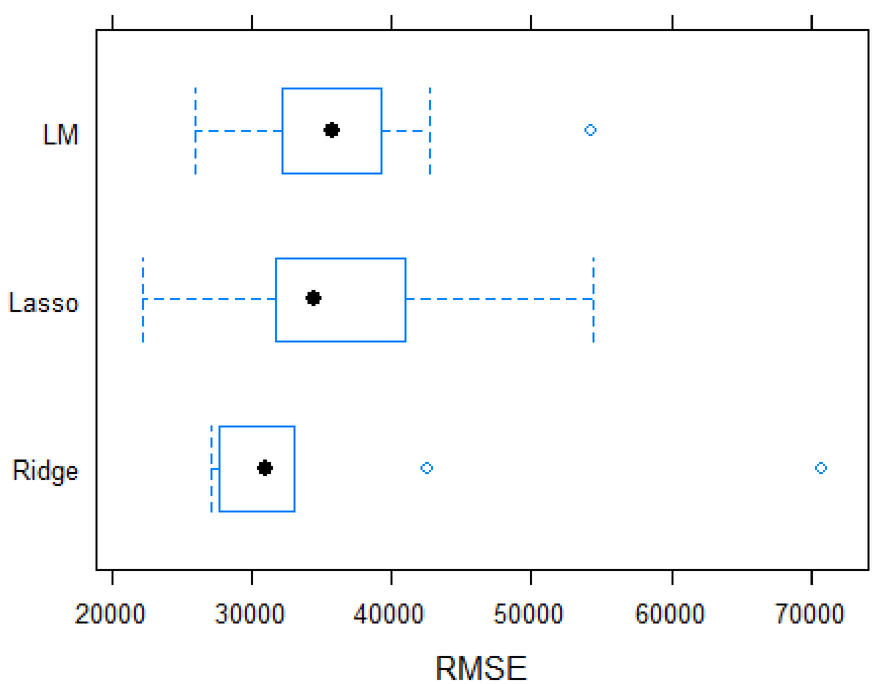
Different modeling methods were then applied on the above predictors to find the most accurate model.

```
ctrl <- trainControl(method = "cv", number=10)
set.seed(100)

#Linear Model
linear.model <- train(SalePrice ~ PoolQC + GrLivArea  + GarageCars + GarageAr
ea + TotRmsAbvGrd +OverallQual
                      + KitchenQual + BsmtQual + OverallCond + X1stFlrSF  + P
oolArea + ExterQual +
                        MasVnrArea + MSSubClass + X2ndFlrSF + LotArea ,
                      data = train,
                      method = "lm", trControl=ctrl)

linear.model

## Linear Regression
##
## 1460 samples
##   16 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1314, 1314, 1313, 1312, 1315, 1315, ...
## Resampling results:
```

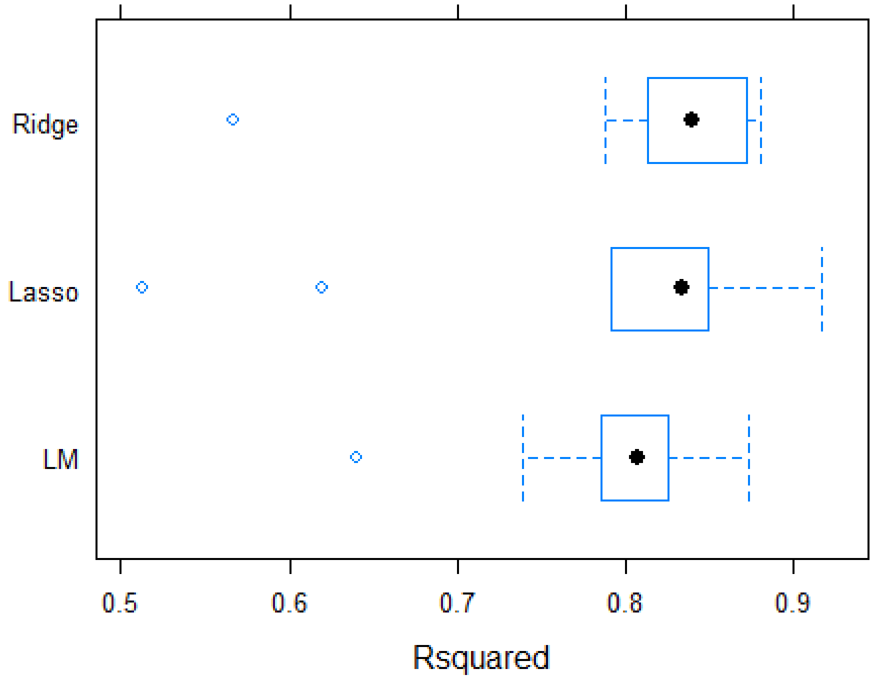
```
##
##   RMSE      Rsquared   MAE
##   36852.74  0.7925789  22475.26
##
## Tuning parameter 'intercept' was held constant at a value of TRUE

#Out Sample RMSE and Rsquare value

RMSE_linear <- rmse(train$SalePrice, predict(linear.model,newdata = train))

#Insample RMSE value
RMSE_linear

## [1] 34045.76

#Ridge Regression
model.ridge <- train(SalePrice ~ PoolQC + GrLivArea  + GarageCars + GarageAre
a + TotRmsAbvGrd +OverallQual
                     + KitchenQual + BsmtQual + OverallCond + X1stFlrSF  + Po
olArea + ExterQual +
                       MasVnrArea + MSSubClass + X2ndFlrSF + LotArea ,
                     data = train, preProcess = c("center", "scale"),
                     method = "ridge",trControl=ctrl)

model.ridge

## Ridge Regression
##
## 1460 samples
##   16 predictor
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1312, 1315, 1313, 1316, 1314, 1313, ...
## Resampling results across tuning parameters:
##
##   lambda  RMSE      Rsquared   MAE
##   0.0000  35739.78  0.8040563  22403.23
##   0.0001  35736.19  0.8041020  22401.78
##   0.1000  35153.58  0.8155487  23098.21
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was lambda = 0.1.

#Out Sample RMSE and Rsquare value
RMSE_ridge <- rmse(train$SalePrice, predict(model.ridge,newdata = train))

#Insample RMSE value
RMSE_ridge

## [1] 34635.65
```

```
#Lasso model
model.lasso <- train(SalePrice ~ PoolQC + GrLivArea  + GarageCars + GarageAre
a + TotRmsAbvGrd +OverallQual
                     + KitchenQual + BsmtQual + OverallCond + X1stFlrSF   + Po
olArea + ExterQual +
                       MasVnrArea + MSSubClass + X2ndFlrSF + LotArea ,
                     data = train, preProcess = c("center", "scale"),
                     method = "lasso", trControl=ctrl)

model.lasso

## The lasso
##
## 1460 samples
##   16 predictor
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1314, 1316, 1314, 1314, 1314, 1314, ...
## Resampling results across tuning parameters:
##
##   fraction  RMSE      Rsquared   MAE
##   0.1       67232.77  0.6341332  47530.30
##   0.5       37853.13  0.7873644  23627.15
##   0.9       36439.88  0.7863152  22469.90
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was fraction = 0.9.

#Out Sample RMSE and Rsquare value

RMSE_lasso <- rmse(train$SalePrice, predict(model.lasso,newdata = train))

#Insample RMSE value
RMSE_lasso

## [1] 34050.39

#Comparing the above model

c.models<- list("LM"=linear.model, "Ridge" = model.ridge,
                "Lasso" = model.lasso)

house_price.resamples<- resamples(c.models)
summary(house_price.resamples)

##
## Call:
## summary.resamples(object = house_price.resamples)
##
## Models: LM, Ridge, Lasso
```

```
## Number of resamples: 10
##
## MAE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## LM    18600.05 21552.41 22449.97 22475.26 23692.89 25449.43    0
## Ridge 19667.47 20769.55 22187.51 23098.21 23815.91 31335.45    0
## Lasso 17255.20 20834.99 22768.88 22469.90 23585.99 28380.99    0
##
## RMSE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## LM    25968.25 32299.61 35750.37 36852.74 38928.64 54295.18    0
## Ridge 27074.83 28039.59 30939.04 35153.58 32755.49 70756.79    0
## Lasso 22155.85 31819.17 34508.61 36439.88 39803.61 54441.83    0
##
## Rsquared
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LM    0.6407153 0.7872002 0.8070497 0.7925789 0.8257684 0.8735021    0
## Ridge 0.5670659 0.8142333 0.8404306 0.8155487 0.8700927 0.8811027    0
## Lasso 0.5128600 0.7939297 0.8343243 0.7863152 0.8485938 0.9168959    0

#plot performances
bwplot(house_price.resamples, metric="RMSE")
```

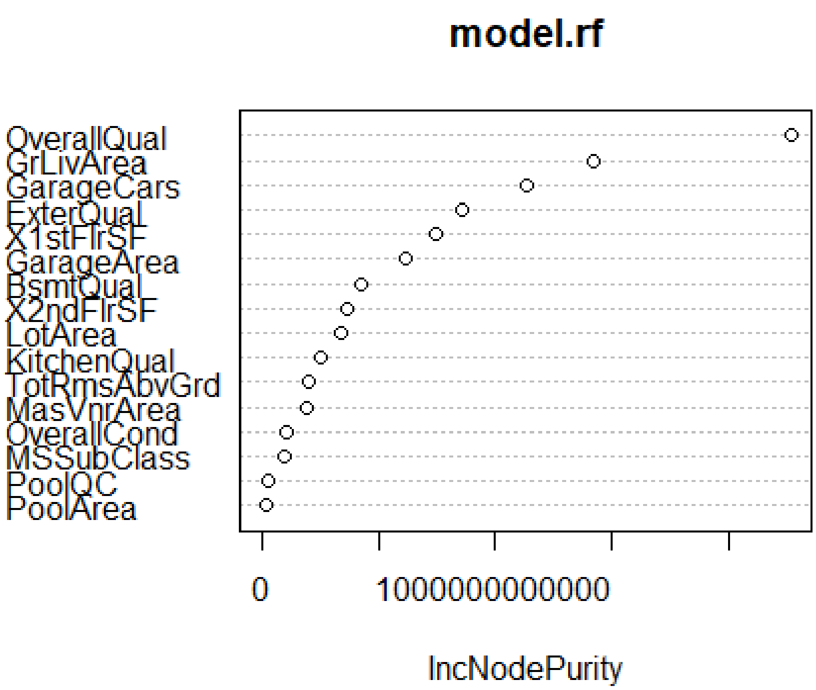



```
bwplot(house_price.resamples, metric="Rsquared")
```




```
#Lets apply other techniques to get the best result
#kNN
model.Knn <- train(SalePrice ~ PoolQC + GrLivArea  + GarageCars + GarageArea
+ TotRmsAbvGrd +OverallQual
                   + KitchenQual + BsmtQual + OverallCond + X1stFlrSF  + Pool
Area + ExterQual +
                     MasVnrArea + MSSubClass + X2ndFlrSF + LotArea ,
                   data = train, preProcess = c("center", "scale"),
                   method = "knn", trControl=ctrl)

model.Knn

## k-Nearest Neighbors
##
## 1460 samples
##   16 predictor
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1314, 1313, 1314, 1313, 1314, 1315, ...
## Resampling results across tuning parameters:
##
##   k  RMSE      Rsquared   MAE
##   5  36364.04  0.7939360  21678.53
##   7  35718.86  0.8026448  21476.93
##   9  35801.01  0.8033714  21442.72
##
```

```
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 7.

#Out Sample RMSE and Rsquare value

RMSE_knn <- rmse(train$SalePrice, predict(model.Knn,newdata = train))

#Insample RMSE value
RMSE_knn

## [1] 31733.93

#Random Forest
library(randomForest)

model.rf<- randomForest(SalePrice~PoolQC + GrLivArea  + GarageCars + GarageAr
ea + TotRmsAbvGrd +OverallQual
                                + KitchenQual + BsmtQual + OverallCond + X1stFlrS
F  + PoolArea + ExterQual +
                                  MasVnrArea + MSSubClass + X2ndFlrSF + LotArea ,
                                data = train, trControl=ctrl)

importance<- importance(model.rf)

varImpPlot(model.rf)
```




```
model.rf
```

```
##
## Call:
##  randomForest(formula = SalePrice ~ PoolQC + GrLivArea + GarageCars +
GarageArea + TotRmsAbvGrd + OverallQual + KitchenQual + BsmtQual +      Overa
llCond + X1stFlrSF + PoolArea + ExterQual + MasVnrArea +      MSSubClass + X2
ndFlrSF + LotArea, data = train, trControl = ctrl)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 954506792
##                     % Var explained: 84.87
```

```
RMSE_rf <- rmse(train$SalePrice, predict(model.rf,newdata = train))
#Insample RMSE value
RMSE_rf
```

```
## [1] 13728.89
```

```
#M5P Model
library(RWeka)
model.m5p <- M5P(SalePrice ~ PoolQC + GrLivArea  + GarageCars + GarageArea +
TotRmsAbvGrd +OverallQual
                 + KitchenQual + BsmtQual + OverallCond + X1stFlrSF  + PoolAr
ea + ExterQual +
                  MasVnrArea + MSSubClass + X2ndFlrSF + LotArea ,
                 data = train)

model.m5p
```

```
## M5 pruned model tree:
## (using smoothed linear models)
##
## OverallQual <= 6.5 :
## |   GrLivArea <= 1378.5 : LM1 (564/21.39%)
## |   GrLivArea >  1378.5 :
## |   |   GarageCars <= 1.5 : LM2 (101/27.117%)
## |   |   GarageCars >  1.5 : LM3 (247/27.413%)
## OverallQual >  6.5 :
## |   OverallQual <= 7.5 : LM4 (319/31.119%)
## |   OverallQual >  7.5 :
## |   |   OverallQual <= 8.5 : LM5 (168/47.225%)
## |   |   OverallQual >  8.5 :
## |   |   |   GrLivArea <= 2229 : LM6 (35/43.976%)
## |   |   |   GrLivArea >  2229 :
## |   |   |   |   GrLivArea <= 3374 : LM7 (20/39.779%)
## |   |   |   |   GrLivArea >  3374 :
## |   |   |   |   |   GrLivArea <= 4576 : LM8 (4/68.461%)
## |   |   |   |   |   GrLivArea >  4576 : LM9 (2/15.583%)
##
## LM num: 1
```

```
## SalePrice =
##  21.089 * GrLivArea
##  + 6257.1089 * GarageCars
##  + 13.2408 * GarageArea
##  - 2314.1835 * TotRmsAbvGrd
##  + 10548.5951 * OverallQual
##  - 2163.6664 * KitchenQual
##  - 4164.0973 * BsmtQual
##  + 5220.5964 * OverallCond
##  + 49.07 * X1stFlrSF
##  - 303.4032 * ExterQual
##  + 14.1699 * MasVnrArea
##  + 28.7601 * MSSubClass
##  + 1.715 * X2ndFlrSF
##  + 1.4625 * LotArea
##  - 18709.7037
##
## LM num: 2
## SalePrice =
##  28923.7253 * GarageCars
##  - 63.7126 * GarageArea
##  - 354.7381 * TotRmsAbvGrd
##  + 2344.897 * OverallQual
##  - 863.3543 * KitchenQual
##  - 1749.9333 * BsmtQual
##  + 8204.9744 * OverallCond
##  + 48.7777 * X1stFlrSF
##  - 1361.5974 * ExterQual
##  + 45.8548 * MasVnrArea
##  - 27.7631 * MSSubClass
##  + 30.967 * X2ndFlrSF
##  + 1.1578 * LotArea
##  + 8885.9181
##
## LM num: 3
## SalePrice =
##  -26.2054 * GrLivArea
##  + 1415.2361 * GarageCars
##  + 18.6206 * GarageArea
##  - 3307.3275 * TotRmsAbvGrd
##  + 15761.9258 * OverallQual
##  - 11043.3855 * KitchenQual
##  - 10665.8092 * BsmtQual
##  + 5881.8769 * OverallCond
##  + 73.0225 * X1stFlrSF
##  - 816.249 * ExterQual
##  + 1.0361 * MasVnrArea
##  - 218.5027 * MSSubClass
##  + 71.4855 * X2ndFlrSF
##  + 0.6184 * LotArea
```

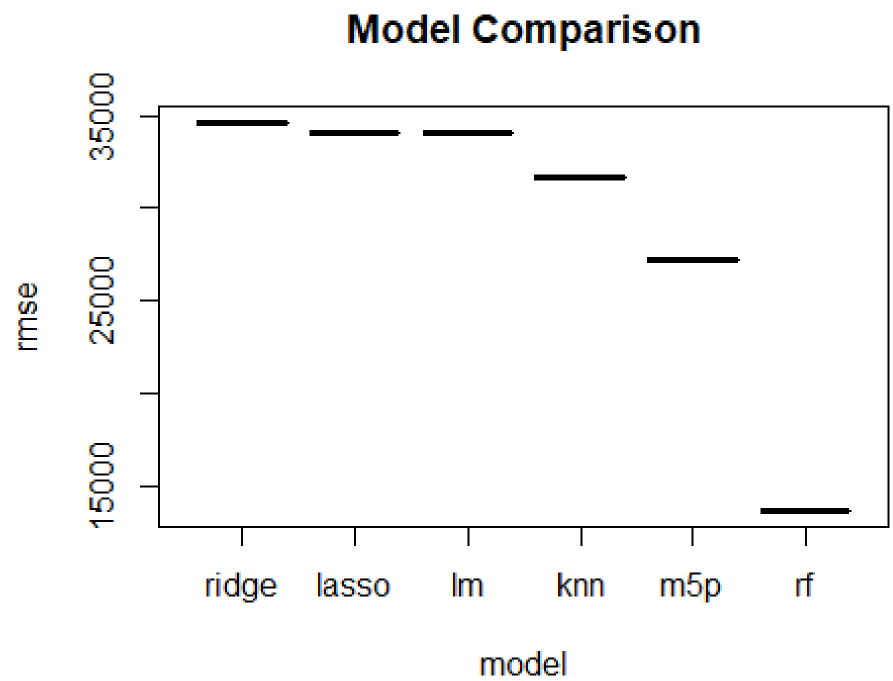
```
##  + 70697.2159
##
## LM num: 4
## SalePrice =
##  -69.7891 * GrLivArea
##  + 1838.8178 * GarageCars
##  + 39.3316 * GarageArea
##  - 3332.3965 * TotRmsAbvGrd
##  + 1548.8296 * OverallQual
##  - 9899.0273 * KitchenQual
##  - 12743.7818 * BsmtQual
##  + 3134.7302 * OverallCond
##  + 149.3833 * X1stFlrSF
##  - 12438.1819 * ExterQual
##  + 18.821 * MasVnrArea
##  - 132.6008 * MSSubClass
##  + 134.3199 * X2ndFlrSF
##  + 0.8174 * LotArea
##  + 158592.9139
##
## LM num: 5
## SalePrice =
##  62.052 * GrLivArea
##  + 25610.6261 * GarageCars
##  - 7.7815 * GarageArea
##  - 4354.4511 * TotRmsAbvGrd
##  + 4032.3156 * OverallQual
##  - 9891.3392 * KitchenQual
##  - 13646.8536 * BsmtQual
##  + 424.7599 * OverallCond
##  + 39.0812 * X1stFlrSF
##  - 626.6508 * ExterQual
##  + 23.6526 * MasVnrArea
##  - 14.9452 * MSSubClass
##  + 1.7271 * X2ndFlrSF
##  + 1.8842 * LotArea
##  + 74113.8107
##
## LM num: 6
## SalePrice =
##  102.9409 * GrLivArea
##  + 13649.2613 * GarageCars
##  - 15.4645 * GarageArea
##  + 6944.7476 * OverallQual
##  - 2607.9454 * KitchenQual
##  - 11691.8254 * BsmtQual
##  + 424.7599 * OverallCond
##  + 37.9352 * X1stFlrSF
##  - 626.6508 * ExterQual
##  + 7.7891 * MasVnrArea
```

```
##  - 241.5029 * MSSubClass
##  - 0.2532 * X2ndFlrSF
##  - 1.4225 * LotArea
##  + 41886.1015
##
## LM num: 7
## SalePrice =
##  73.0149 * GrLivArea
##  + 13649.2613 * GarageCars
##  - 15.4645 * GarageArea
##  + 13539.1812 * TotRmsAbvGrd
##  + 6944.7476 * OverallQual
##  - 2607.9454 * KitchenQual
##  - 29605.5903 * BsmtQual
##  + 424.7599 * OverallCond
##  + 29.2452 * X1stFlrSF
##  - 626.6508 * ExterQual
##  + 7.7891 * MasVnrArea
##  - 291.2351 * MSSubClass
##  - 0.2532 * X2ndFlrSF
##  - 4.1759 * LotArea
##  + 74182.0664
##
## LM num: 8
## SalePrice =
##  74.9013 * GrLivArea
##  + 13649.2613 * GarageCars
##  - 15.4645 * GarageArea
##  + 6944.7476 * OverallQual
##  - 2607.9454 * KitchenQual
##  - 29676.9436 * BsmtQual
##  + 424.7599 * OverallCond
##  - 11.7055 * X1stFlrSF
##  - 626.6508 * ExterQual
##  + 7.7891 * MasVnrArea
##  - 291.2351 * MSSubClass
##  - 0.2532 * X2ndFlrSF
##  - 5.8296 * LotArea
##  + 343561.7385
##
## LM num: 9
## SalePrice =
##  65.292 * GrLivArea
##  + 13649.2613 * GarageCars
##  - 15.4645 * GarageArea
##  + 6944.7476 * OverallQual
##  - 2607.9454 * KitchenQual
##  - 29676.9436 * BsmtQual
##  + 424.7599 * OverallCond
##  - 11.7055 * X1stFlrSF
```

```
##   - 626.6508 * ExterQual
##   + 7.7891 * MasVnrArea
##   - 291.2351 * MSSubClass
##   - 0.2532 * X2ndFlrSF
##   - 5.8296 * LotArea
##   + 373612.5419
##
## Number of Rules : 9

#Out Sample RMSE and Rsquare value

RMSE_m5p <- rmse(train$SalePrice, predict(model.m5p,newdata = train))

#Insample RMSE value
RMSE_m5p

## [1] 27274.42
```

Model Comparison

```
#This helped me finalize the best model.

test.rmse <- data.frame( model = c("lm", "ridge", "lasso", "knn", "rf", "m5p"
),
                         rmse = c(RMSE_linear,RMSE_ridge, RMSE_lasso, RMSE_kn
n, RMSE_rf, RMSE_m5p))


test.rmse <- test.rmse[order(test.rmse$rmse, decreasing = TRUE),]

test.rmse$model <- factor(test.rmse$model, levels = test.rmse$model)

plot(test.rmse, main = "Model Comparison")
```



Predicting the SalePrice for the test dataset

```
prediction <- predict(model.rf, test)

prediction <- data.frame( prediction= prediction)

write.table(prediction, file="Prediction1.csv", row.names=FALSE, col.names=TR
UE)
```