Regression: Predicting House Prices

Jason Liu

Oct 2, 2016

StatementThis project serves the purpose of demonstrating machine learning technique, regression, in R. The case is provided by the Coursera course: Machine Learning Foundamentals-- Case Studies, taught by Washington Unversity. This project is corresponding to the one professors show in IPython notebook.

Load Data and Packages

```
library(MASS)
library(ggplot2)
library(GGally)
library(caret)
library(randomForest)
library(readr)
library(magrittr)
library(dplyr)
library(sqldf)
library(Metrics)
Data<- read.csv('home_data.csv')
```

Data Preprocessing and Exploration

In general, the machine learning project get started with data preprocessing. In my opinion, the data preprocessing should include two important steps: Data cleansing and feature engineering. Having a look at the dataset, one can conclude that the integrity of dataset is perfect since it does not contain any 'NA' values. Therefore, we will focus on feature selection and engineering.

Data Cleansing

Although the dimension of dataset is not quite high, we still need to reduce the dimension in order to simplify the regression problem. Empiricalll judged, we can assume that the first feature, date, can be viewed as a redundant variable due to the fact that the dataset has already got time of build and rennovation. Therefore, together with 'id', the column 'date' should be delected from the dataset directly. Meanwhile, the categorical variables should be transformed to the factor.

```
data<-Data[,-c(1,2)]
```

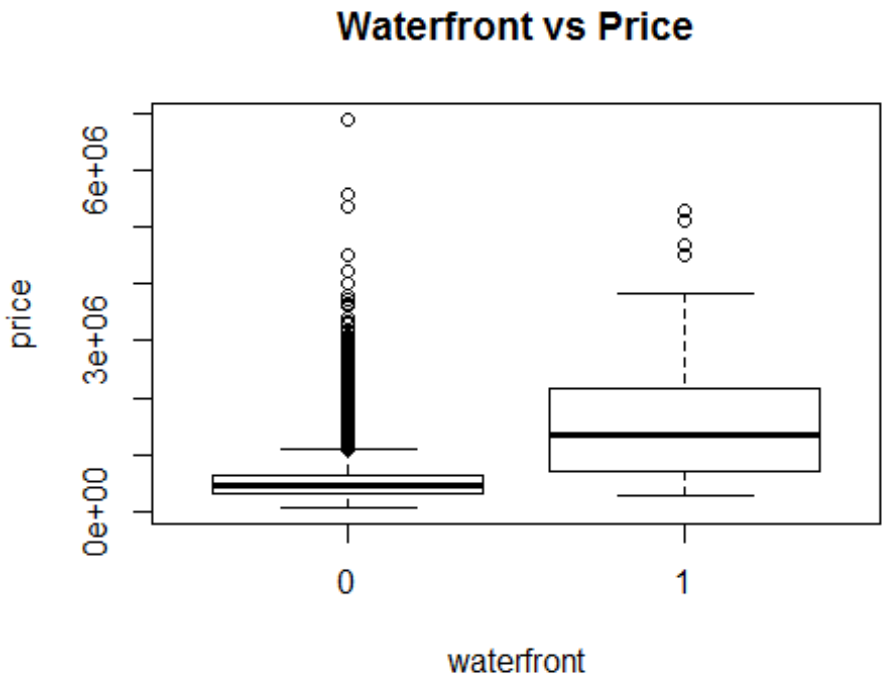
Spliting Dataset

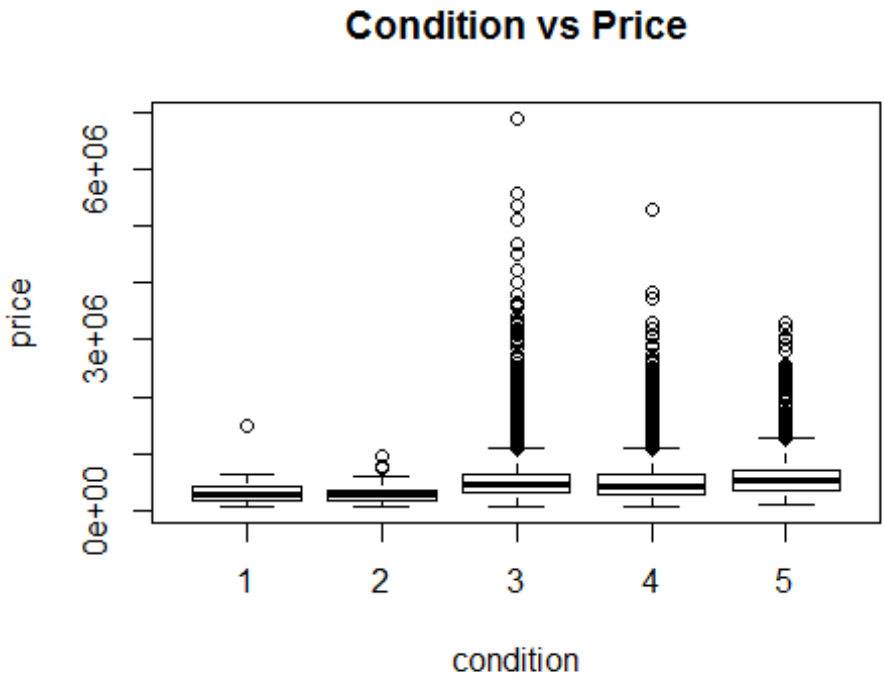
Before data exploration, one should split the dataset into two part: a training set for model fitting and a test set for the evaluation of the model. The data exploration should be done on the training set.

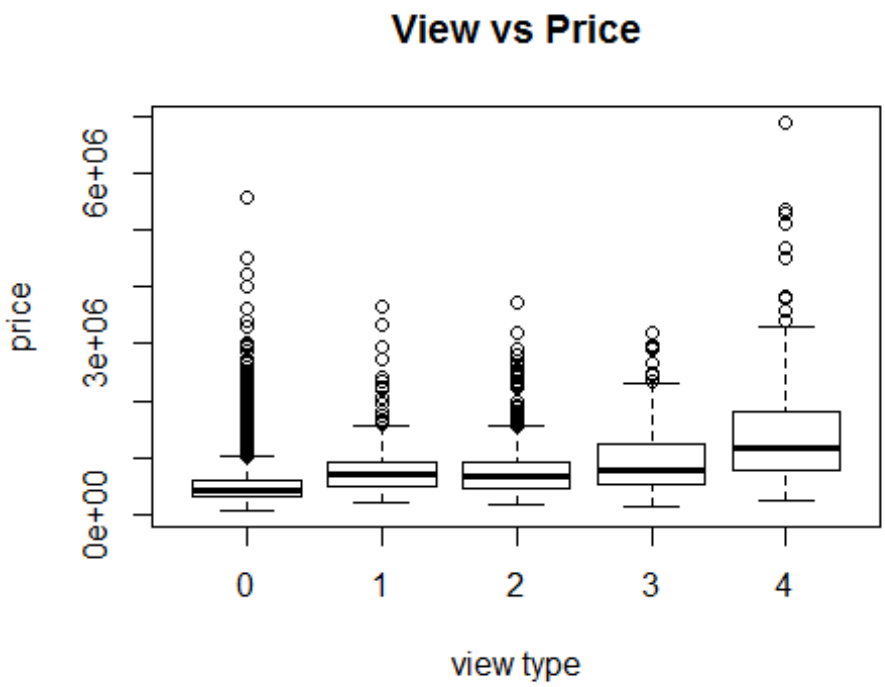
```
set.seed(2222)
intrain<-createDataPartition(p=0.75,y = data$price,list = FALSE)
train<- data[intrain,]
test<- data[-intrain,]
```

Explore Categorical Variables

The next step is to transform the categorical variables into factors. Based on my experience, the variables of 'waterfront' ,'condition' and 'view' should be the categorical variables. After the transformation, I create 3 box plots that can reflect the relationship between the features and the response variables.


Waterfront vs Price
price
0e+00
3e+06
6e+06
0
1
waterfront


Condition vs Price
price
0e+00
3e+06
6e+06
1
2
3
4
5
condition

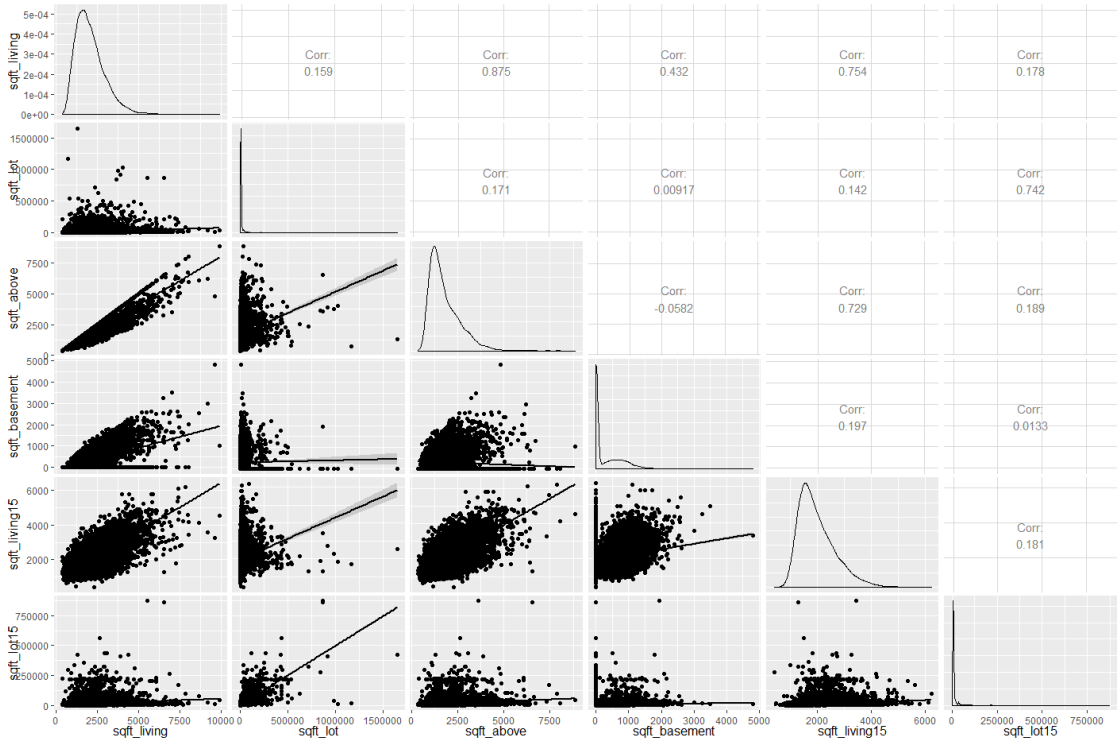


We can see that whether the house has a waterfront has really big impact on the prices of the house. Meanwhile, the condition of house is a column of data that is not evenly distributed. The medians of prices for each condition are close to each other but the variance in '3rd condition' is obviously larger that the others. In terms of view type, we can find that the view types pf 3 and 4 have slight price advantages in comparison with the other types.

Explore Size-Related Factors

It should be also noted that the size of house has great impact on the price. After exploring the dataset, one can find out that the data set has several variables related to the size of house. If we just incorporate those variables in regression model simutaneously, the correlation among factors may result in multicolinearity, leading to the problem of overfitting. Therefore, we need explore them before implementing feature engineering.

```
Size<-train %>% select(contains("sqft"))
ggpairs(Size,lower = list(continuous = "smooth",method='lm'))
```



From the above pairs plot one can summarize that some variables are strongly correlated to other variables. **The square of living room is always correlated with the square of above area due to the structure of buildings.** Meanwhile, we find that the statistics of house size are strongly correlated with those of 15 nearest neighbours. Therefore, one can assume a fact that **the neighbourhood can be a poential factor that may influence the price of house**.

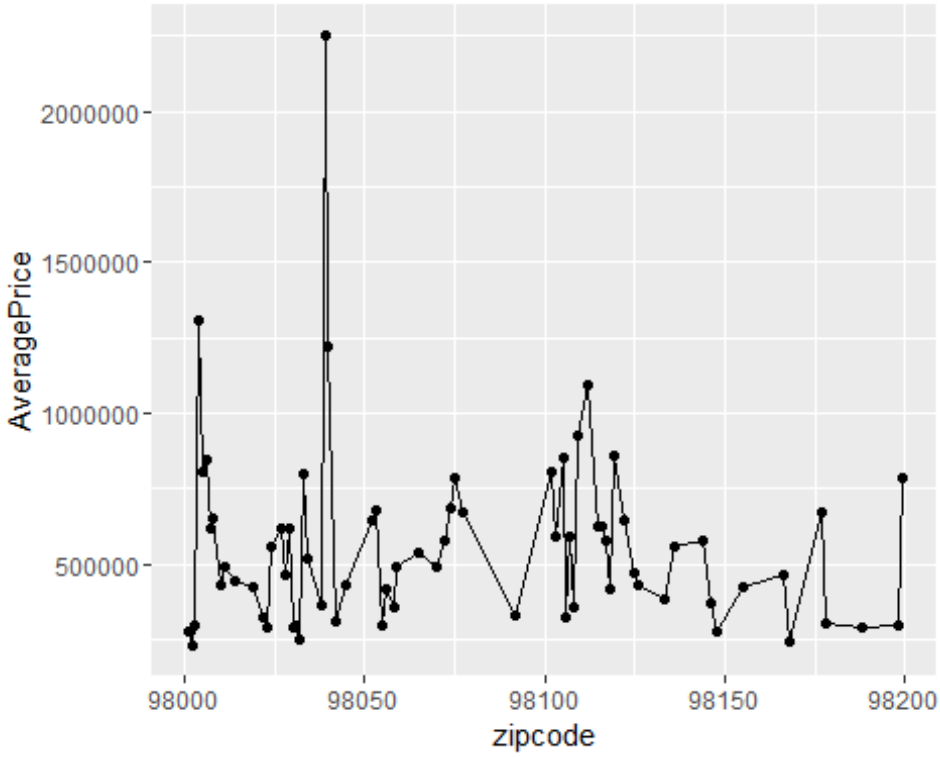
Explore the Impact of Neighbourhood

Through the above analysis, one can summarize that houses in the similar area may have similar size. Considering that the size of house may directly influence the house price and zip code can be viewed as a convenient indicator of the location of the house, we explore the relationship between the zip code and the price.

```
zipcode<-sqldf('SELECT zipcode, avg(price) as AveragePrice from train
group by zipcode')

## Loading required package: tcltk

g<-
ggplot(zipcode,aes(x=zipcode,y=AveragePrice))+geom_point()+geom_line(st
at="identity")
g
```



Through the above plot, we can find that in some area, the price of house is indeed high. The potential reason is probably that in some region, the houses are relatively large, therefore impacting the prices.

Correlations among Variables

The final step of the data exploration is to check the correlations among factors, including the response variable. We present a correlation matrix below in order to clarify the correlations.

```
CorrelationMatrix<-cor(train[,c(1:6,10:19)])
print(CorrelationMatrix)
```

```
##                    price     bedrooms   bathrooms sqft_living
## price          1.00000000  0.301976442  0.52087492  0.69919043
## bedrooms       0.30197644  1.000000000  0.50948765  0.57139762
## bathrooms      0.52087492  0.509487646  1.00000000  0.75379192
## sqft_living    0.69919043  0.571397618  0.75379192  1.00000000
## sqft_lot       0.08190279  0.028049232  0.07778229  0.15884076
## floors         0.26155193  0.181535058  0.50203090  0.35840017
## grade          0.67194492  0.353305660  0.65975392  0.76283088
## sqft_above     0.60125479  0.475746282  0.68457643  0.87509684
## sqft_basement  0.32192499  0.292184354  0.27929589  0.43216314
## yr_built       0.06342919  0.156224825  0.50847635  0.32381394
## yr_renovated   0.11179313  0.014432427  0.05146432  0.04903643
## zipcode       -0.05310629 -0.154291741 -0.20235777 -0.19896970
## lat            0.31211350 -0.006777609  0.02181690  0.05534792
```

```
## long           0.01739695  0.126449294  0.21990330  0.23354761
## sqft_living15  0.58635323  0.386849187  0.56371130  0.75438822
## sqft_lot15     0.08303690  0.029656556  0.08676231  0.17755066
##                   sqft_lot       floors        grade   sqft_above
## price          0.081902786  0.261551929  0.671944920  0.601254792
## bedrooms       0.028049232  0.181535058  0.353305660  0.475746282
## bathrooms      0.077782293  0.502030901  0.659753920  0.684576433
## sqft_living    0.158840758  0.358400172  0.762830878  0.875096835
## sqft_lot       1.000000000 -0.009215084  0.100773373  0.170920215
## floors        -0.009215084  1.000000000  0.459777238  0.528238227
## grade          0.100773373  0.459777238  1.000000000  0.753934382
## sqft_above     0.170920215  0.528238227  0.753934382  1.000000000
## sqft_basement  0.009166477 -0.245007686  0.168699129 -0.058237380
## yr_built       0.051516149  0.483802012  0.451512218  0.429699596
## yr_renovated   0.010768687  0.009393182  0.009050975  0.019614665
## zipcode       -0.129555186 -0.055486763 -0.184204492 -0.259478844
## lat           -0.087870430  0.047218468  0.117703134 -0.001801678
## long           0.224253898  0.122910479  0.196738879  0.337665549
## sqft_living15  0.141704571  0.282476169  0.714607458  0.729219680
## sqft_lot15     0.742193104 -0.010255900  0.117981002  0.189408813
##                sqft_basement    yr_built yr_renovated     zipcode
## price           0.321924989  0.06342919  0.111793132 -0.05310629
## bedrooms        0.292184354  0.15622483  0.014432427 -0.15429174
## bathrooms       0.279295891  0.50847635  0.051464316 -0.20235777
## sqft_living     0.432163137  0.32381394  0.049036433 -0.19896970
## sqft_lot        0.009166477  0.05151615  0.010768687 -0.12955519
## floors         -0.245007686  0.48380201  0.009393182 -0.05548676
## grade           0.168699129  0.45151222  0.009050975 -0.18420449
## sqft_above     -0.058237380  0.42969960  0.019614665 -0.25947884
## sqft_basement   1.000000000 -0.13273494  0.064603635  0.07307571
## yr_built       -0.132734939  1.00000000 -0.225615683 -0.34464663
## yr_renovated    0.064603635 -0.22561568  1.000000000  0.06809965
## zipcode         0.073075711 -0.34464663  0.068099652  1.00000000
## lat             0.117530693 -0.14935457  0.032886795  0.26497937
## long           -0.147441558  0.41182840 -0.068444820 -0.56582517
## sqft_living15   0.197337082  0.33009946 -0.007638116 -0.28020348
## sqft_lot15      0.013309919  0.07256627  0.006446733 -0.14731011
##                        lat        long sqft_living15   sqft_lot15
## price          0.312113500  0.01739695   0.586353234  0.083036905
## bedrooms      -0.006777609  0.12644929   0.386849187  0.029656556
## bathrooms      0.021816898  0.21990330   0.563711296  0.086762306
## sqft_living    0.055347921  0.23354761   0.754388219  0.177550657
## sqft_lot      -0.087870430  0.22425390   0.141704571  0.742193104
## floors         0.047218468  0.12291048   0.282476169 -0.010255900
## grade          0.117703134  0.19673888   0.714607458  0.117981002
## sqft_above    -0.001801678  0.33766555   0.729219680  0.189408813
## sqft_basement  0.117530693 -0.14744156   0.197337082  0.013309919
## yr_built      -0.149354571  0.41182840   0.330099465  0.072566274
## yr_renovated   0.032886795 -0.06844482  -0.007638116  0.006446733
## zipcode        0.264979370 -0.56582517  -0.280203476 -0.147310111
```

```
## lat            1.000000000 -0.13747772   0.052364925 -0.083261728
## long          -0.137477720  1.00000000   0.332218373  0.250023760
## sqft_living15  0.052364925  0.33221837   1.000000000  0.180716895
## sqft_lot15    -0.083261728  0.25002376   0.180716895  1.000000000
```

Through the correlation matrix, one can find out that some features are highly correlated with each other. For example, the features of square feet of the house and that of the neighours will be the pair of variables that may result in multicolinearity. In the next part, I will build multiple regression models and evaluate them one by one.

Fitting Models

Regression Model

The first model we choose to fit is the classic regression model. Regression model should always be the first choice when one wants to perform predictive analytics on the continous variable. We first

```
features_reg1<-
c('price','bedrooms','bathrooms','sqft_living','sqft_lot','floors','zip
code')
reg1<- lm(price~.,data = train[,features_reg1])
summary(reg1)

##
## Call:
## lm(formula = price ~ ., data = train[, features_reg1])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1316816  -141601   -23278    99736  4012530
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.412e+07  3.775e+06 -14.335  < 2e-16 ***
## bedrooms    -5.627e+04  2.648e+03 -21.250  < 2e-16 ***
## bathrooms    9.069e+03  4.352e+03   2.084   0.0372 *
## sqft_living  3.144e+02  3.586e+00  87.673  < 2e-16 ***
## sqft_lot    -2.681e-01  4.993e-02  -5.370 7.99e-08 ***
## floors      -1.135e+02  4.312e+03  -0.026   0.9790
## zipcode      5.524e+02  3.848e+01  14.356  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 254100 on 16203 degrees of freedom
## Multiple R-squared:  0.5108, Adjusted R-squared:  0.5106
## F-statistic:  2819 on 6 and 16203 DF,  p-value: < 2.2e-16
```

Above presents the sumary of the first regression model. One can find out that the significances of coefficient are mostly able to pass the hythesis tests. The R-square is not good enough.

Then we incorporate more features in the model. The categorical variables as well as the grade and year built are included.

```
features_reg2<-
append(features_reg1,c('waterfront','view','condition','grade','yr_buil
t'))
reg2<-lm(price~.,data = train[,features_reg2])
summary(reg2)
```

```
##
## Call:
## lm(formula = price ~ ., data = train[, features_reg2])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1266527  -109927    -9623    89108  4159205
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.638e+06  3.490e+06   2.189   0.0286 *
## bedrooms    -3.665e+04  2.302e+03 -15.920  < 2e-16 ***
## bathrooms    4.393e+04  3.904e+03  11.254  < 2e-16 ***
## sqft_living  1.649e+02  3.796e+00  43.443  < 2e-16 ***
## sqft_lot    -2.267e-01  4.236e-02  -5.352 8.84e-08 ***
## floors       2.286e+04  3.959e+03   5.774 7.86e-09 ***
## zipcode     -1.555e+01  3.501e+01  -0.444   0.6569
## waterfront   5.325e+05  2.089e+04  25.487  < 2e-16 ***
## view         4.530e+04  2.553e+03  17.741  < 2e-16 ***
## condition    1.835e+04  2.838e+03   6.466 1.03e-10 ***
## grade        1.283e+05  2.468e+03  51.979  < 2e-16 ***
## yr_built    -3.539e+03  8.105e+01 -43.669  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 215300 on 16198 degrees of freedom
## Multiple R-squared:  0.6489, Adjusted R-squared:  0.6487
## F-statistic:  2721 on 11 and 16198 DF,  p-value: < 2.2e-16
```

The adjusted R square has obvious improvement. However, the coefficient of zipcode is no longer statistically significant, which is not in accordance with the assumption made in data exploration.

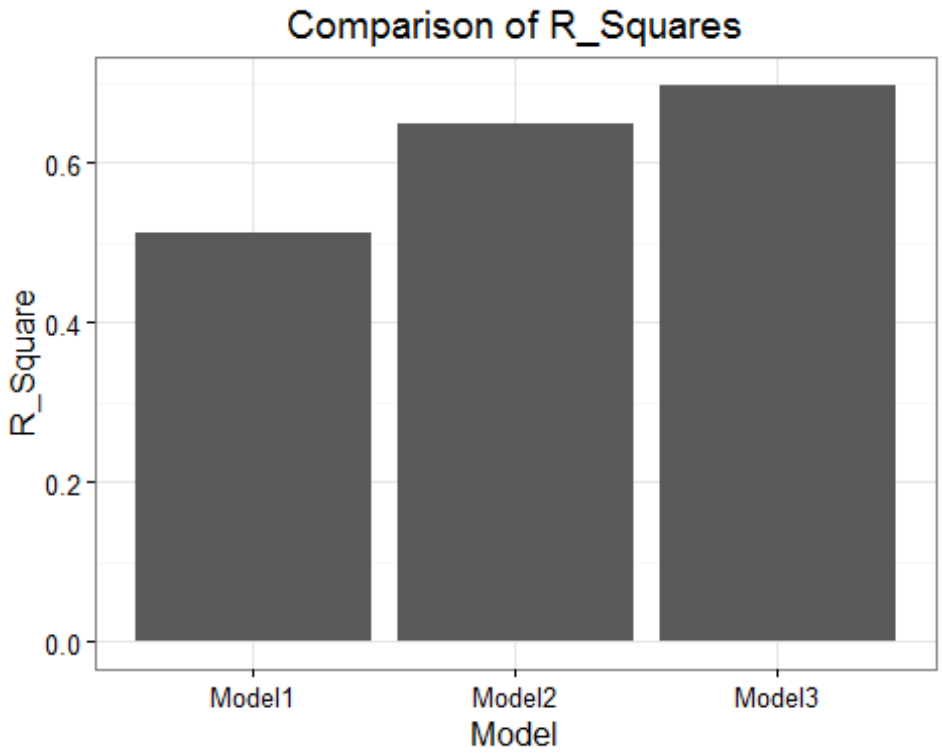
Finally, we build a regression model with all variables included. The summary of model is also presented.

```
reg3<- lm(price~.,data=train)
summary(reg3)
```

```
##
## Call:
## lm(formula = price ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1224671   -99318   -10249    76096  4111117
##
## Coefficients: (1 not defined because of singularities)
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    5.352e+06  3.357e+06   1.594  0.11091
## bedrooms      -3.388e+04  2.142e+03 -15.816  < 2e-16 ***
## bathrooms      3.995e+04  3.721e+03  10.737  < 2e-16 ***
## sqft_living    1.458e+02  5.036e+00  28.956  < 2e-16 ***
## sqft_lot       1.711e-01  5.751e-02   2.975  0.00293 **
## floors         7.012e+03  4.116e+03   1.703  0.08851 .
## waterfront     5.437e+05  1.945e+04  27.956  < 2e-16 ***
## view           5.272e+04  2.431e+03  21.685  < 2e-16 ***
## condition      2.584e+04  2.688e+03   9.613  < 2e-16 ***
## grade          9.933e+04  2.482e+03  40.018  < 2e-16 ***
## sqft_above     2.820e+01  4.986e+00   5.657 1.57e-08 ***
## sqft_basement         NA         NA      NA       NA
## yr_built      -2.561e+03  8.330e+01 -30.749  < 2e-16 ***
## yr_renovated   1.092e+01  4.237e+00   2.577  0.00998 **
## zipcode       -5.656e+02  3.785e+01 -14.944  < 2e-16 ***
## lat            6.040e+05  1.230e+04  49.089  < 2e-16 ***
## long          -2.106e+05  1.505e+04 -13.998  < 2e-16 ***
## sqft_living15  2.256e+01  3.972e+00   5.680 1.37e-08 ***
## sqft_lot15    -3.928e-01  8.748e-02  -4.490 7.16e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 199900 on 16192 degrees of freedom
## Multiple R-squared:  0.6976, Adjusted R-squared:  0.6973
## F-statistic:  2197 on 17 and 16192 DF,  p-value: < 2.2e-16
```

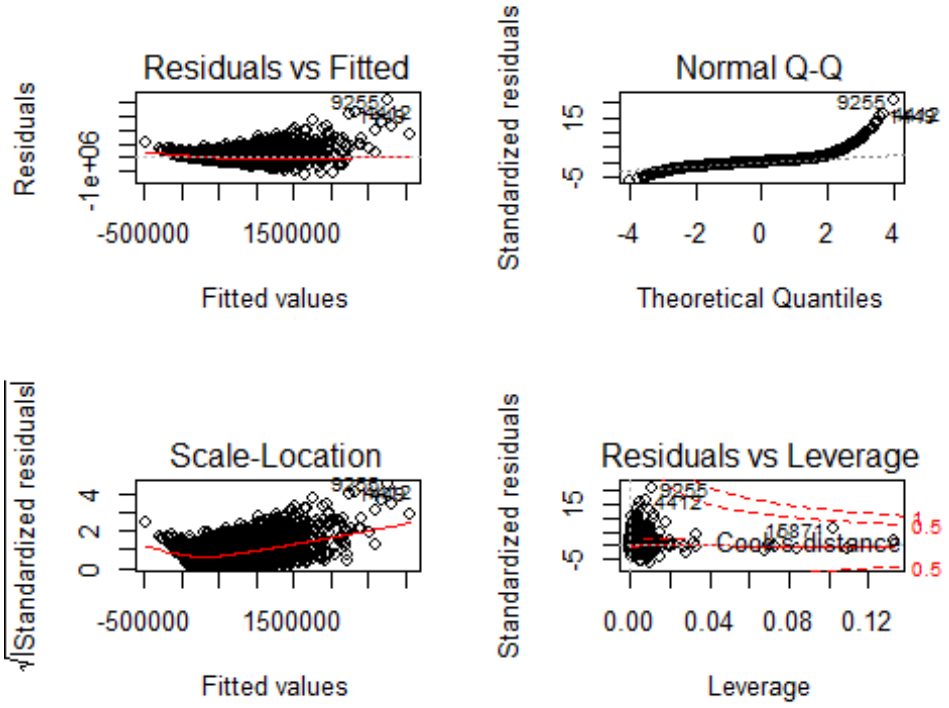
The adjusted R squared value is higher than the previous models while almost all variables have a coefficient that is statistically significant. I put the R squares of three model in one plot order to make the comparison more convenient.

```
RSquare<-
data.frame(Model=c('Model1','Model2','Model3'),R_Square=c(summary(reg1)
$r.squared,summary(reg2)$r.squared,summary(reg3)$r.squared))
ggplot(RSquare,aes(x=Model,y=R_Square))+geom_bar(stat =
'identity')+ggtitle('Comparison of R_Squares')+theme_bw()
```



For the last model, We plot the diagonsis of the final regression model below.

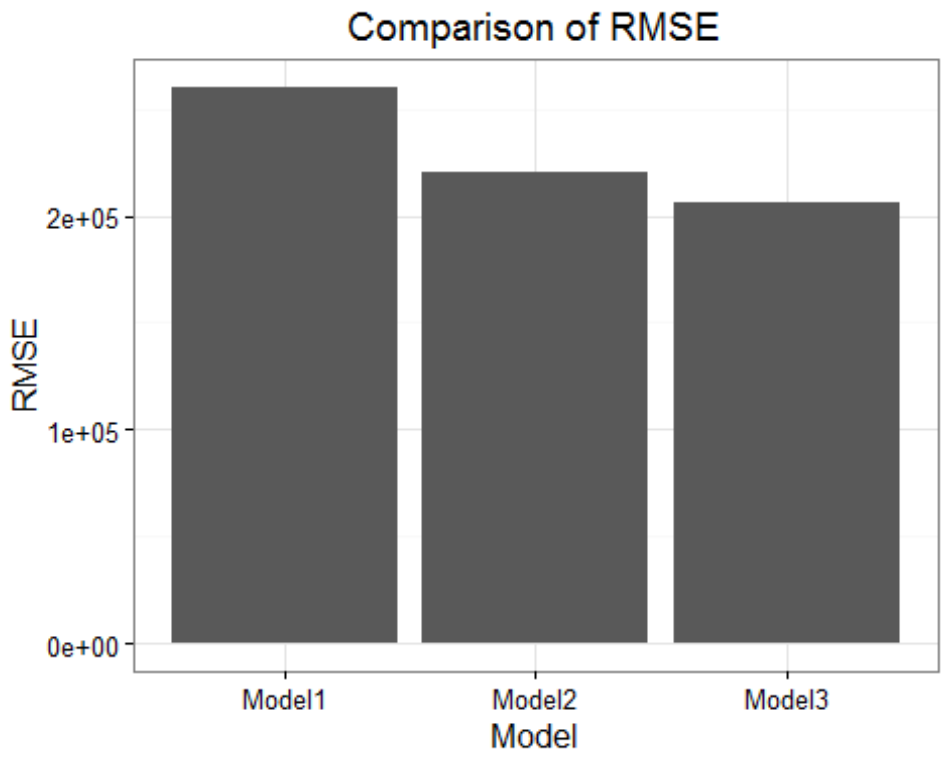
```
par(mfrow=c(2,2))
plot(reg3)
```



The diagnosis of the residual seems to be reasonable. The residuals are patternless and almost normally distributed. Meanwhile, the outliers with high leverage do not have huge impact on the model.

Therefore, we use the regression model to make predictions based on the test set. In order to further compare the three regression model, we put the RMSE of three models together for a direct comparison.

```
prediction_reg1<- predict(reg1,newdata=test[,features_reg1[-1]])
Accu1<-rmse(test$price,prediction_reg1)
prediction_reg2<- predict(reg2,newdata=test[,features_reg2[-1]])
Accu2<-rmse(test$price,prediction_reg2)
prediction_reg3<- predict(reg3,newdata=test[,-1])
Accu3<-rmse(test$price,prediction_reg3)
Accuracy<-
data.frame(Model=c('Model1','Model2','Model3'),RMSE=c(Accu1,Accu2,Accu3
))
ggplot(Accuracy,aes(x=Model,y=RMSE))+geom_bar(stat =
'identity')+ggtitle('Comparison of RMSE')+theme_bw()
```



It is obvious that the regression model with more features can provide a better prediction performance in this case. The regression model always has great intepretability, but the accuracy of prediction can not be guranteed.

Tree-Based Model

Additionally, I try two advanced machine learning models for the comparison.

- Random Forest*

The first model I choose is the random forest, which is a popular model that always can provide accurate result. In order to prevent overfitting, I use 3-fold cross validation to tune the parameter and test the model.

```
###Accu_rf= 137233.2481
rf<- randomForest(price~.,data=train)
rf_prediction<- predict(rf,newdata=test[,-1])
Accu_rf<- rmse(test$price,rf_prediction)
```

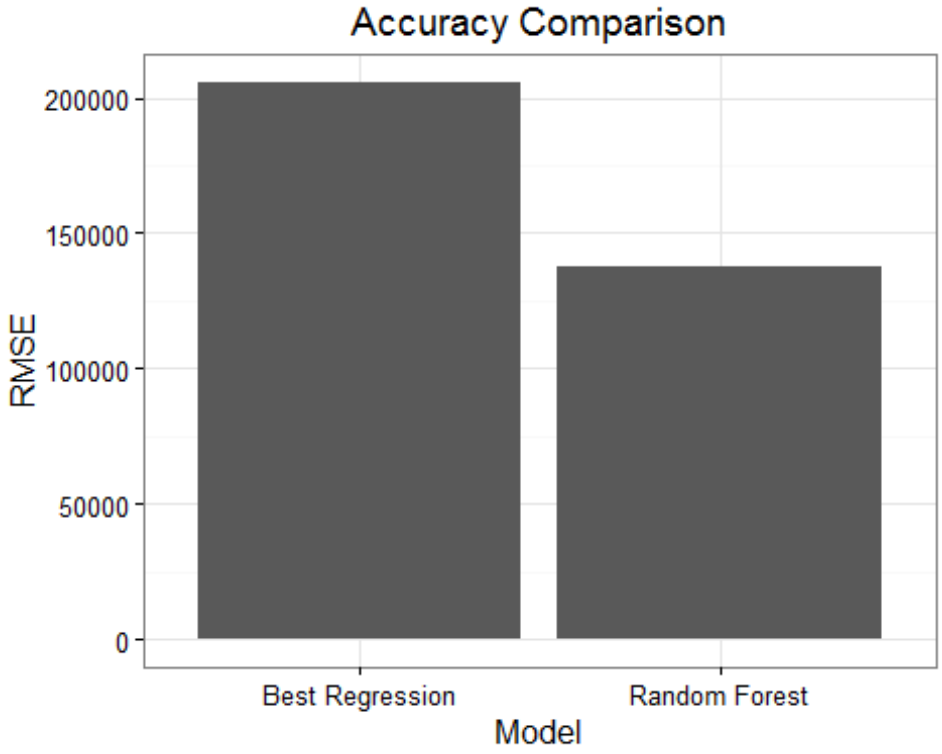
Due to the intolerable speed of the model running, I choose to present the result directly in R-markdown file.

```
Accu_rf<-137233.2481
```

We can see that the random forest model may provide a prediction that is way better than the basic regression model.

Conclusion

Through the above analysis, one can find out that the regression models have great intepretability.



However, the regression model cannot provide the same accuracy as the random forest model, although it is a blackbox to some extent.