PSTAT131 Final Project

Inwoong Bae (6433692\805-206-6644)

3 10, 2020

Introduction

Thie research question is which factor has positive or negative impacts to housing price and which prediction model will show future the housing prices based on the historical dataset appropriately? This is important for regional economic analysis to predict the future values of housings and real estates. We know the price is based on the size of the house, number of bathrooms and bedrooms, how much the house is closed to transportations, location and etc, but we need to be more specific during the analysis.

Data

This dataset contains records of apartments for sale and for rent in the city of São Paulo, Brazil. The dataset consists of 16 different variables (5 numeric variables, 6 logistic variables, 5 categorial variables) and about 13,000 records of apartment sales and rents advertised in April, 2019 from multiple websites.

Data Structure:

Price: Final price advertised (R$ Brazilian Real)

Condo: Condominium expenses (unknown values are marked as zero)

Size : The property size in Square Meters m^2 (private areas only)

Rooms: Number of bedrooms

Toilets: Number of toilets (all toilets)

Suites: Number of bedrooms with a private bathroom (en suite)

Parking: Number of parking spots

Elevator: Binary value: 1 if there is elevator in the building, 0 otherwise

Furnished: Binary value: 1 if the property is funished, 0 otherwise

Swimming Pool: Binary value: 1 if the property has swimming pool, 0 otherwise

New: Binary value: 1 if the property is very recent, 0 otherwise

District: The neighborhood and city where the property is located.

Negotiation Type: Sale or Rent

Property Type: The property type

Latitude: Geographic location

Longitude: Geographic location

References

https://www.kaggle.com/argonalyst/sao-paulo-real-estate-sale-rent-april-2019

Methods

Analysis Plan

- Data processing, filtering, clustering if necessary.
- Summary and plots(Scatter plot, boxplot, …)

1. Logistic Regression
2. Decision Tree
3. Random Forest
4. Model Decision and Conclusion

```
## Warning: package 'knitr' was built under R version 3.6.2
```

Data processing and filtering

```r
#read data from Excel file.
saodata <- read.csv("C:/Users/Inwoong/Downloads/sao-paulo-properties-april-2019.csv")
#overview
dim(saodata)
```

```
## [1] 13640    16
```

```r
summary(saodata)
```

```
##      Price              Condo           Size           Rooms
##  Min.   :     480   Min.   :   0   Min.   : 30.0   Min.   : 1.00
##  1st Qu.:    1859   1st Qu.: 290   1st Qu.: 50.0   1st Qu.: 2.00
##  Median :    8100   Median : 500   Median : 65.0   Median : 2.00
##  Mean   :  287738   Mean   : 690   Mean   : 84.4   Mean   : 2.31
##  3rd Qu.:  360000   3rd Qu.: 835   3rd Qu.: 94.0   3rd Qu.: 3.00
##  Max.   :10000000   Max.   :9500   Max.   :880.0   Max.   :10.00
##
##     Toilets         Suites         Parking        Elevator       Furnished
##  Min.   :1.00   Min.   :0.000   Min.   :0.00   Min.   :0.000   Min.   :0.000
##  1st Qu.:2.00   1st Qu.:1.000   1st Qu.:1.00   1st Qu.:0.000   1st Qu.:0.000
##  Median :2.00   Median :1.000   Median :1.00   Median :0.000   Median :0.000
##  Mean   :2.07   Mean   :0.981   Mean   :1.39   Mean   :0.354   Mean   :0.147
##  3rd Qu.:2.00   3rd Qu.:1.000   3rd Qu.:2.00   3rd Qu.:1.000   3rd Qu.:0.000
##  Max.   :8.00   Max.   :6.000   Max.   :9.00   Max.   :1.000   Max.   :1.000
##
##  Swimming.Pool        New                                District
##  Min.   :0.000   Min.   :0.0000   Moema/S o Paulo     :  293
##  1st Qu.:0.000   1st Qu.:0.0000   Mooca/S o Paulo     :  288
##  Median :1.000   Median :0.0000   Br s/S o Paulo     :  255
##  Mean   :0.512   Mean   :0.0156   Bela Vista/S o Paulo:  250
##  3rd Qu.:1.000   3rd Qu.:0.0000   Brooklin/S o Paulo  :  250
##  Max.   :1.000   Max.   :1.0000   Pinheiros/S o Paulo :  249
##                                   (Other)             :12055
##  Negotiation.Type   Property.Type      Latitude       Longitude
##  rent:7228        apartment:13640   Min.   :-46.7   Min.   :-58.4
##  sale:6412                          1st Qu.:-23.6   1st Qu.:-46.7
##                                     Median :-23.6   Median :-46.6
##                                     Mean   :-22.1   Mean   :-43.6
##                                     3rd Qu.:-23.5   3rd Qu.:-46.6
##                                     Max.   :  0.0   Max.   :  0.0
##
```

```
str(saodata)
```

```
## 'data.frame':    13640 obs. of  16 variables:
##  $ Price           : int  930 1000 1000 1000 1300 1170 1000 900 1000 1000 ...
##  $ Condo           : int  220 148 100 200 410 0 180 150 0 0 ...
##  $ Size            : int  47 45 48 48 55 50 52 40 65 100 ...
##  $ Rooms           : int  2 2 2 2 2 2 1 2 2 2 ...
##  $ Toilets         : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Suites          : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Parking         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Elevator        : int  0 0 0 0 1 0 1 0 0 0 ...
##  $ Furnished       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Swimming.Pool   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ New             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ District        : Factor w/ 96 levels "Alto de Pinheiros/S o Paulo",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Negotiation.Type: Factor w/ 2 levels "rent","sale": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Property.Type   : Factor w/ 1 level "apartment": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Latitude        : num  -23.5 -23.6 -23.5 -23.5 -23.5 ...
##  $ Longitude       : num  -46.5 -46.5 -46.5 -46.5 -46.5 ...
```

```
#check missingness in dataset
apply(is.na(saodata), 2, sum)
```

```
##            Price            Condo             Size            Rooms
##                0                0                0                0
##          Toilets           Suites          Parking         Elevator
##                0                0                0                0
##        Furnished    Swimming.Pool              New         District
##                0                0                0                0
## Negotiation.Type    Property.Type         Latitude        Longitude
##                0                0                0                0
```

```
# Select parameters in the dataset.
## New, Property.Type is out of the model because they have only one type.
## District is related to the location(Latitude and Longitude). I therefore remove this variable from m
library(dplyr)
```

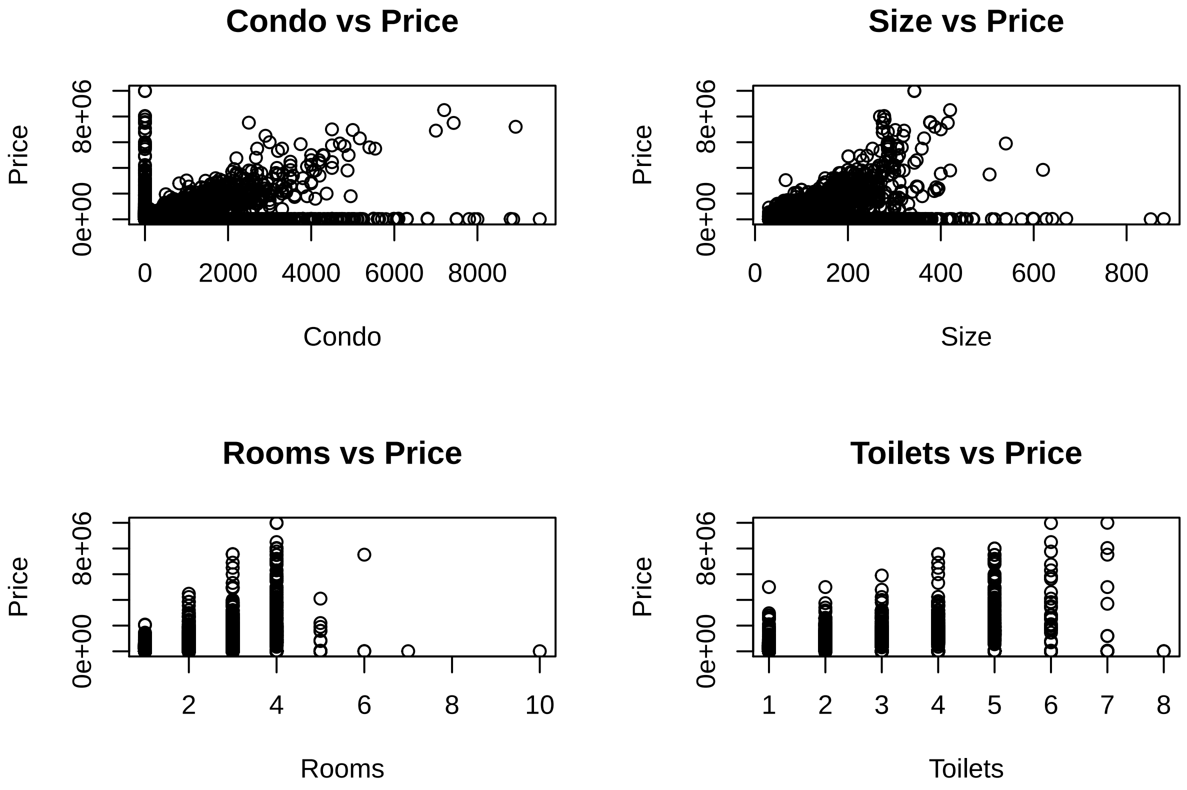
```
##
## Attaching package: 'dplyr'
```

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

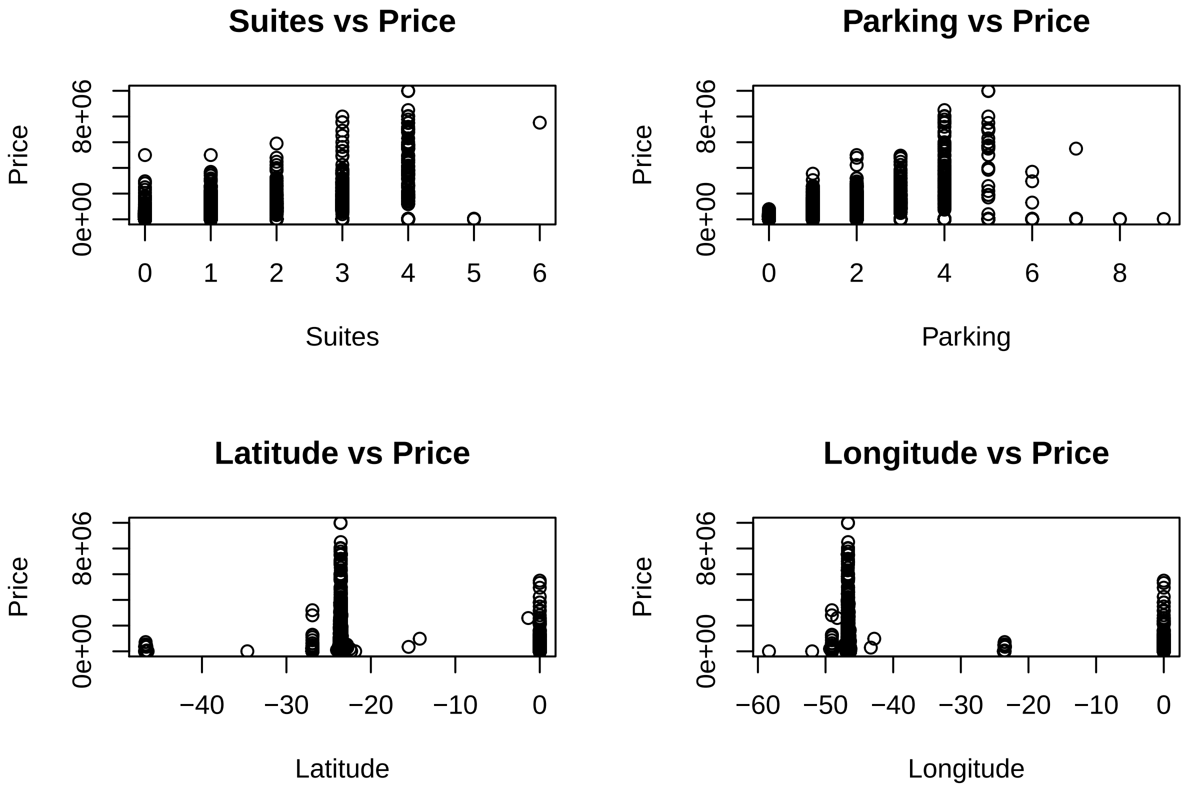
```
saodata <- dplyr::select(saodata, -c(New, Property.Type, District))
head(saodata)
```

```
##   Price Condo Size Rooms Toilets Suites Parking Elevator Furnished
## 1   930   220   47     2       2      1       1        0         0
## 2  1000   148   45     2       2      1       1        0         0
## 3  1000   100   48     2       2      1       1        0         0
## 4  1000   200   48     2       2      1       1        0         0
## 5  1300   410   55     2       2      1       1        1         0
## 6  1170     0   50     2       2      1       1        0         0
##   Swimming.Pool Negotiation.Type Latitude Longitude
## 1             0             rent -23.5431  -46.4795
## 2             0             rent -23.5502  -46.4807
## 3             0             rent -23.5428  -46.4857
## 4             0             rent -23.5472  -46.4830
## 5             0             rent -23.5250  -46.4824
## 6             0             rent -23.5488  -46.4772
```

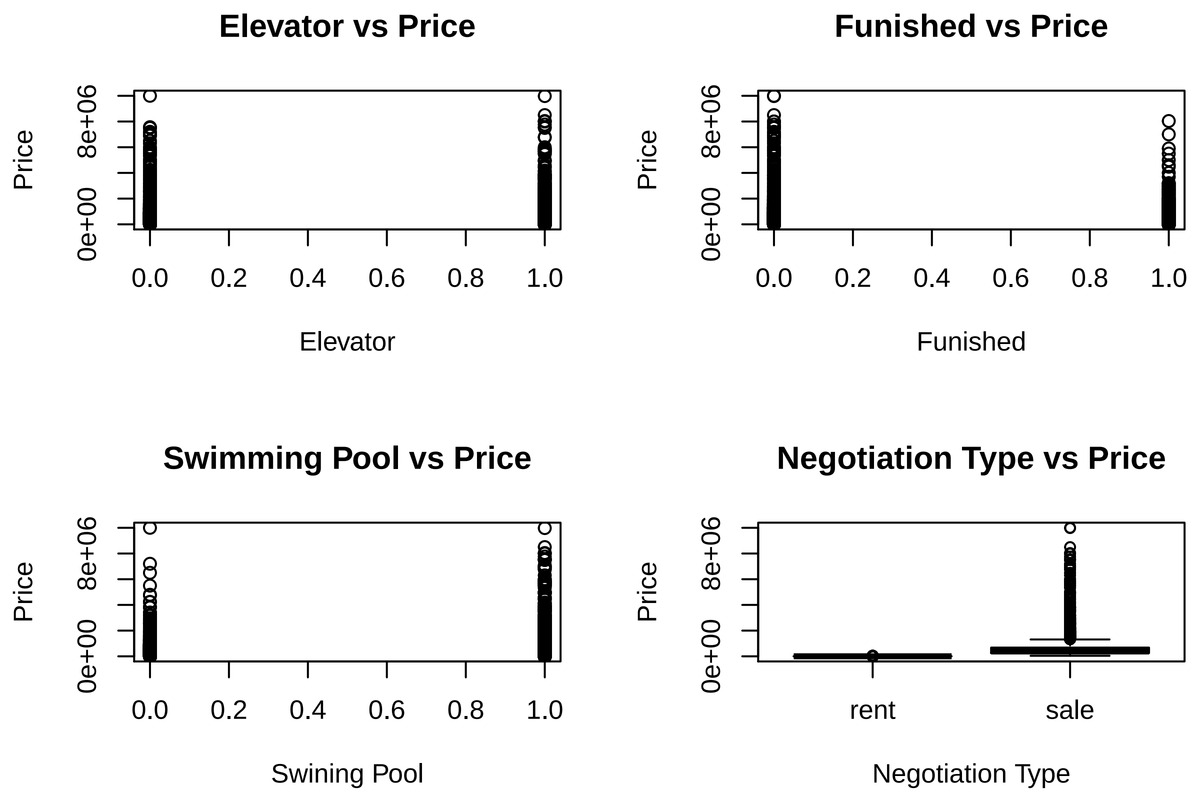
```
#plots(Price vs other parameters)
library(scatterplot3d)
par(mfrow=c(2,2))
plot(saodata$Condo, saodata$Price, main = "Condo vs Price", xlab = "Condo", ylab = "Price")
plot(saodata$Size, saodata$Price, main = "Size vs Price", xlab = "Size", ylab = "Price")
plot(saodata$Rooms, saodata$Price, main = "Rooms vs Price", xlab = "Rooms", ylab = "Price")
plot(saodata$Toilets, saodata$Price, main = "Toilets vs Price", xlab = "Toilets", ylab = "Price")
```



```r
par(mfrow=c(2,2))
plot(saodata$Suites, saodata$Price, main = "Suites vs Price", xlab = "Suites", ylab = "Price")
plot(saodata$Parking, saodata$Price, main = "Parking vs Price", xlab = "Parking", ylab = "Price")
plot(saodata$Latitude, saodata$Price, main = "Latitude vs Price", xlab = "Latitude", ylab = "Price")
plot(saodata$Longitude, saodata$Price, main = "Longitude vs Price", xlab = "Longitude", ylab = "Price")
```



```
par(mfrow=c(2,2))
plot(saodata$Elevator, saodata$Price, main = "Elevator vs Price", xlab = "Elevator", ylab = "Price")
plot(saodata$Furnished, saodata$Price, main = "Funished vs Price", xlab = "Funished", ylab = "Price")
plot(saodata$Swimming.Pool, saodata$Price, main = "Swimming Pool vs Price", xlab = "Swining Pool", ylab
plot(saodata$Negotiation.Type, saodata$Price, main = "Negotiation Type vs Price", xlab = "Negotiation T
```



```r
# Split the data
set.seed(20200305)
train <- sample(1:dim(saodata)[1], dim(saodata)[1]*0.75, rep=FALSE)
test <- -train
training_data<- saodata[train, ]
testing_data= saodata[test, ]
dim(training_data)
```

```
## [1] 10230    13
```

```r
dim(testing_data)
```

```
## [1] 3410   13
```

Logistic Regression

```r
logistic_train <- training_data
logistic_test <- testing_data
#get the number of average price of all housing.
mean_log <- mean(logistic_train$Price)
#If the price is over the average, Price is 1. If not, the price is 0.
logistic_train$Price <- ifelse(logistic_train$Price <= mean_log , 0, 1)
logistic_test$Price <- ifelse(logistic_test$Price <= mean_log , 0, 1)
#Build logistic model
```

```r
#glm.fit <-  glm(Price~., data=logistic_train, family=binomial)
glm.fit <-  glm(Price~Condo+Size+Rooms+Toilets+Suites+Parking+Elevator+Furnished+Swimming.Pool+Negotiat
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Price ~ Condo + Size + Rooms + Toilets + Suites +
##     Parking + Elevator + Furnished + Swimming.Pool + Negotiation.Type +
##     Latitude + Longitude, family = binomial, data = logistic_train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -3.608   0.000   0.000   0.005   2.983
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -1.61e+04   9.11e+05   -0.02   0.9859
## Condo                 2.57e-03   2.24e-04   11.47   < 2e-16 ***
## Size                  1.49e-01   6.51e-03   22.86   < 2e-16 ***
## Rooms                -1.20e+00   1.14e-01  -10.46   < 2e-16 ***
## Toilets               7.60e-01   3.94e-01    1.93   0.0537 .
## Suites               -9.51e-01   3.89e-01   -2.45   0.0144 *
## Parking               9.02e-01   1.78e-01    5.08   3.7e-07 ***
## Elevator              1.95e-01   1.03e-01    1.90   0.0572 .
## Furnished             4.50e-01   1.65e-01    2.72   0.0065 **
## Swimming.Pool         1.50e+00   9.75e-02   15.40   < 2e-16 ***
## Negotiation.Typesale  1.61e+04   9.11e+05    0.02   0.9859
## Latitude             -3.08e-02   4.36e-02   -0.71   0.4798
## Longitude             5.62e-03   2.22e-02    0.25   0.7997
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 12724.9  on 10229  degrees of freedom
## Residual deviance:  2929.8  on 10217  degrees of freedom
## AIC: 2956
##
## Number of Fisher Scoring iterations: 14
```

Interpretation of the logistic regression model

The variable Condo has a coefficient 2.569e-03. For every one unit change in Condo, the log odds of getting higher price than average increases by 2.569e-03, holding other variables fixed.

The variable Size has a coefficient 1.487e-01. For a one unit increase in Size, the log odds of getting higher price than average increases by 1.487e-01, holding other variables fixed.

The variable Rooms has a coefficient -1.195e+00. For a one unit increase in Rooms, the log odds of getting higher price than average decreases by 1.195e+00, holding other variables fixed.

The variable Toilets has a coefficient 7.601e-01. For a one unit increase in Toilets, the log odds of getting higher price than average increases by 7.601e-01, holding other variables fixed.

The variable Suites has a coefficient -9.512e-01. For a one unit increase in Suites, the log odds of getting higher price than average decreases by 9.512e-01, holding other variables fixed.

The variable Parking has a coefficient 9.024e-01. For a one unit increase in Parking, the log odds of getting higher price than average increases by 9.024e-01, holding other variables fixed.

The variable Furnished has a coefficient 4.504e-01, meaning that the indicator function of 1 has a regression coefficient 4.504e-01. That being said, furnished room changes the log odds of getting higher price than average to increase by 4.504e-01.

The variable Swimming.Pool has a coefficient 1.502e+00, meaning that the indicator function of 1 has a regression coefficient 1.502e+00. That being said, existence of a swimming pool changes the log odds of getting higher price than average to increase by 1.502e+00.

The variable Negotiation.Type has a coefficient 1.610e+04, meaning that the indicator function of Sale has a regression coefficient 1.610e+04. That being said, Sale type verses Rent changes the log odds of getting higher price than average to increase by 1.610e+04.

The variable Latitude has a coefficient -3.079e-02. For a one unit increase in Latitude, the log odds of getting higher price than average decreases by 3.079e-02, holding other variables fixed.

The variable Longitude has a coefficient 5.625e-03. For a one unit increase in Longitude, the log odds of getting higher price than average increases by 5.625e-03, holding other variables fixed.

```r
# Model selection
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
full.model <-  glm(Price~Condo+Size+Rooms+Toilets+Suites+Parking+Elevator+Furnished, data=logistic_trai
summary(full.model)
```

```
##
## Call:
## glm(formula = Price ~ Condo + Size + Rooms + Toilets + Suites +
##     Parking + Elevator + Furnished, family = binomial, data = logistic_train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -3.055  -0.874  -0.735   1.261   2.584
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.37e+00   8.99e-02  -26.37  < 2e-16 ***
## Condo       -5.10e-04   4.86e-05  -10.50  < 2e-16 ***
## Size         3.83e-03   7.64e-04    5.01  5.3e-07 ***
## Rooms        4.21e-01   4.03e-02   10.43  < 2e-16 ***
## Toilets      2.83e-01   5.51e-02    5.14  2.8e-07 ***
## Suites      -3.67e-01   6.40e-02   -5.73  1.0e-08 ***
## Parking      1.26e-01   4.45e-02    2.83   0.0047 **
## Elevator     5.62e-01   4.63e-02   12.16  < 2e-16 ***
## Furnished   -1.78e-02   6.40e-02   -0.28   0.7808
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 12725  on 10229  degrees of freedom
## Residual deviance: 12127  on 10221  degrees of freedom
## AIC: 12145
##
## Number of Fisher Scoring iterations: 4
```

When the logistic regression model has all preditors, the model has the smallest AIC, which means the full model is the most significant model.

```r
phat <- predict(glm.fit, type = 'response')
yhat <- phat > 0.5
#Confusion matrix table
ct <- table(obs=logistic_train$Price,pred=yhat)
ct
```

```
##    pred
## obs FALSE TRUE
##   0  6695  327
##   1   329 2879
```

```r
#TPR
tpr <- ct[2,2]/(ct[2,1]+ct[2,2])
tpr
```

```
## [1] 0.897444
```

```r
#FPR
fpr <- ct[1,2]/(ct[1,1]+ct[1,2])
fpr
```

```
## [1] 0.0465679
```

```r
#Misclassification rate
mean(yhat != logistic_train$Price)
```

```
## [1] 0.0641251
```

```r
# install.packages("ROCR")
library(ROCR)
```

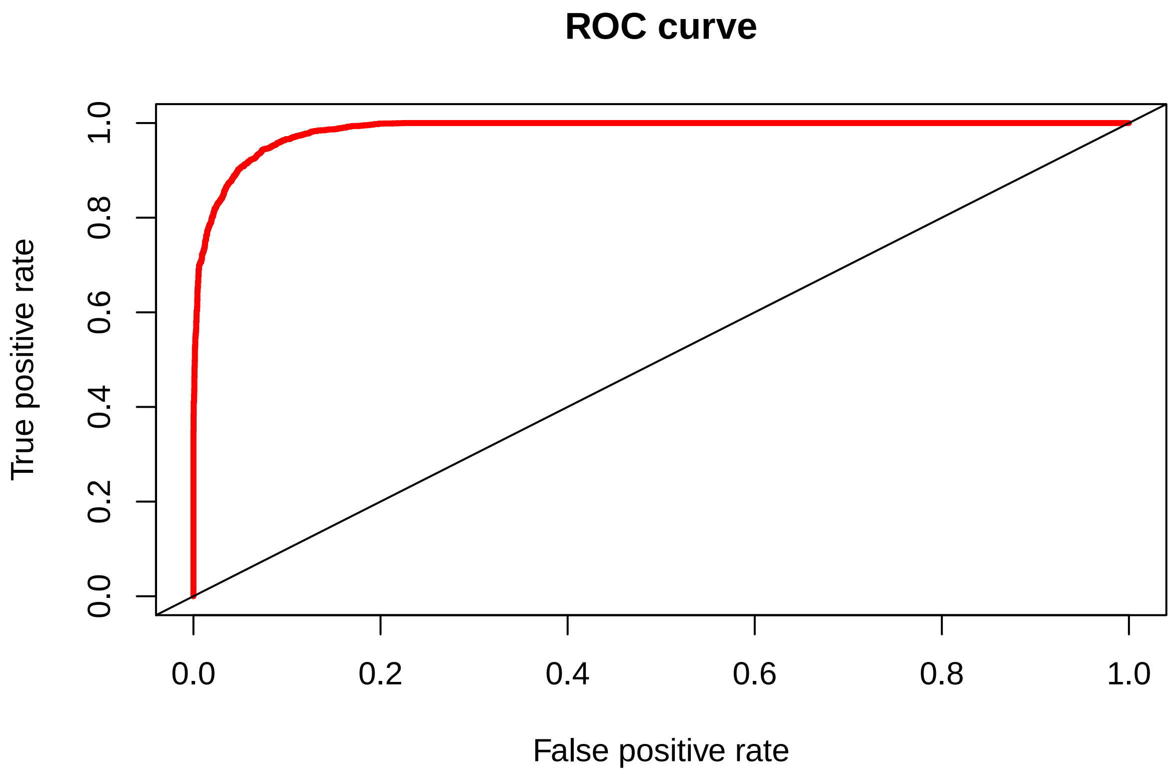
```
## Warning: package 'ROCR' was built under R version 3.6.3
## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.6.3

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

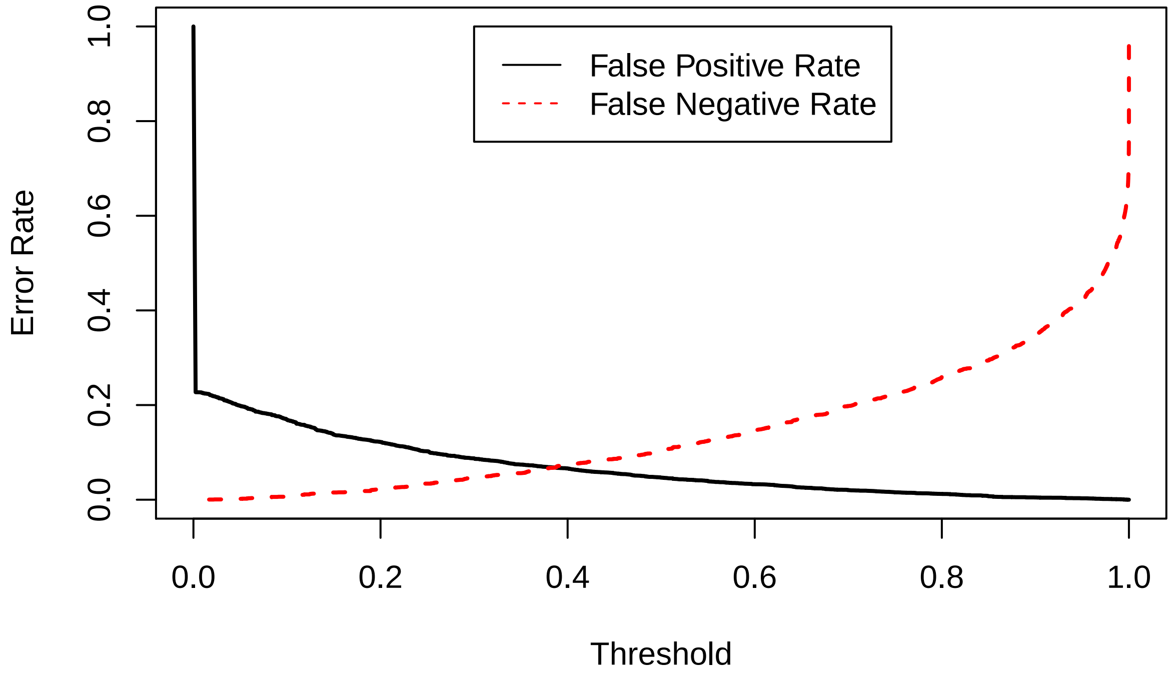
```
# First arument is the logistic_train, second is true labels
pred = prediction(phat, logistic_train$Price)
perf = performance(pred, measure = "tpr", x.measure = "fpr")
# plot ROC curve
plot(perf, col=2, lwd=3, main="ROC curve")
abline(0,1)
```



```
# Calculate AUC
auc = performance(pred, "auc")@y.values
auc
```

```
## [[1]]
## [1] 0.985185
```

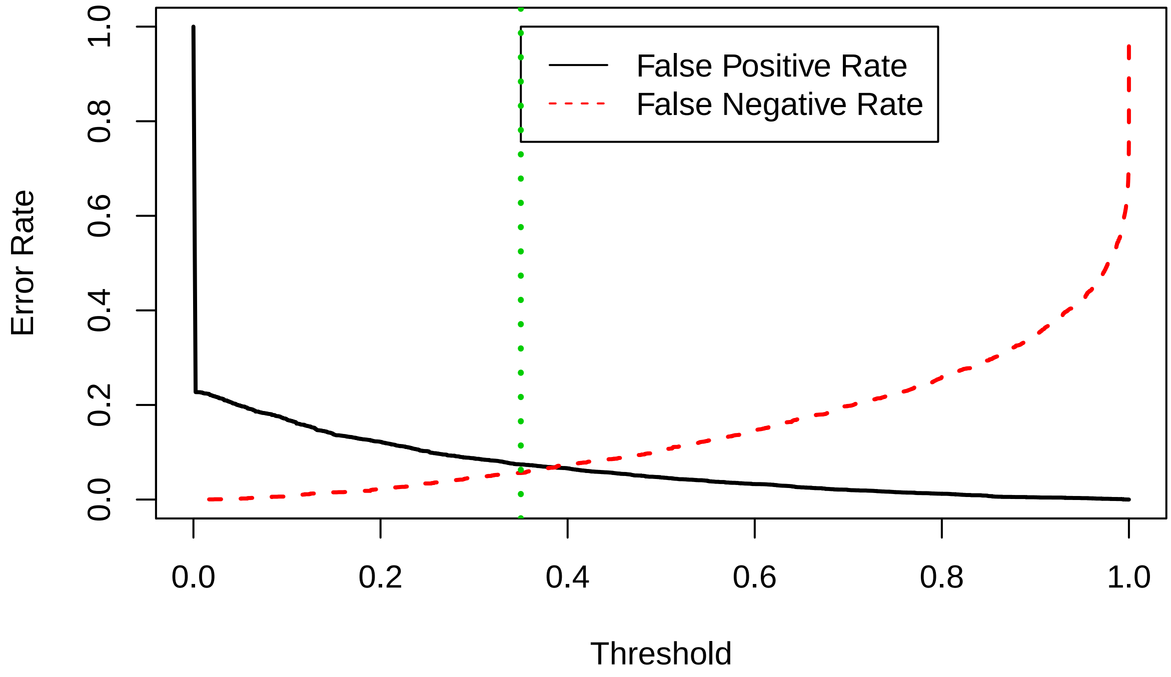
```
#Obtain FPR and FNR from performance() output:
# FPR
fpr = performance(pred, "fpr")@y.values[[1]]
cutoff = performance(pred, "fpr")@x.values[[1]]
# FNR
fnr = performance(pred,"fnr")@y.values[[1]]
# Plot FPR and FNR versus threshold values using matplot():
# Plot
matplot(cutoff, cbind(fpr,fnr), type="l",lwd=2, xlab="Threshold",ylab="Error Rate")
# Add legend to the plot
legend(0.3, 1, legend=c("False Positive Rate","False Negative Rate"),
col=c(1,2), lty=c(1,2))
```



```
# Calculate the euclidean distance between (FPR,FNR) and (0,0)
rate = as.data.frame(cbind(Cutoff=cutoff, FPR=fpr, FNR=fnr))
rate$distance = sqrt((rate[,2])^2+(rate[,3])^2)
# Select the probability threshold with the smallest euclidean distance
index = which.min(rate$distance)
best = rate$Cutoff[index]
best
```

```
## [1] 0.349976
```

```
# Plot
matplot(cutoff, cbind(fpr,fnr), type="l",lwd=2, xlab="Threshold",ylab="Error Rate")
# Add legend to the plot
legend(0.35, 1, legend=c("False Positive Rate","False Negative Rate"),
col=c(1,2), lty=c(1,2))
# Add the best value
abline(v=best, col=3, lty=3, lwd=3)
```



```
#Therefore, our best cutoff value is 0.34998. That means, probabilities for higher price is less than 0
```

Decision Tree

```
library(tree)
```

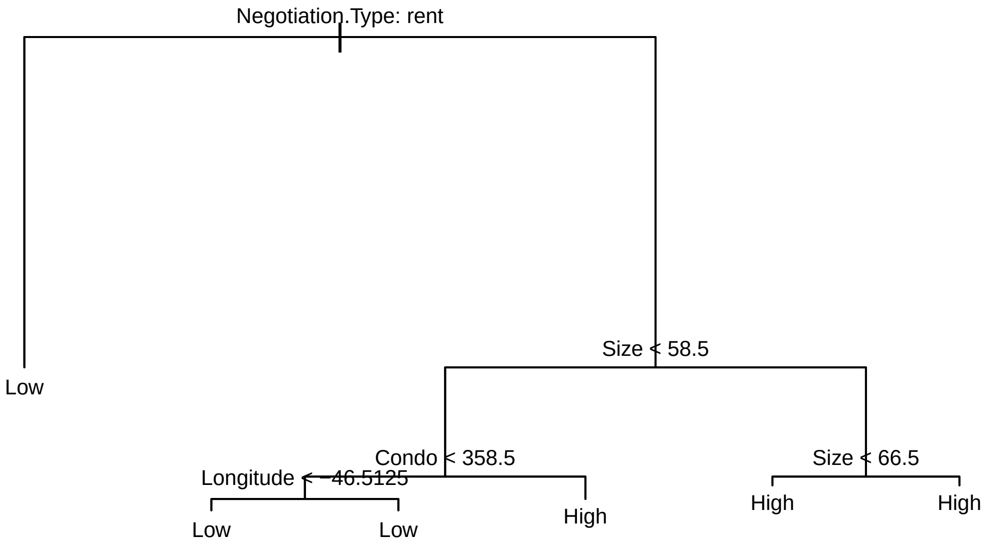
```
## Warning: package 'tree' was built under R version 3.6.2
```

```
logistic_train_class <- training_data
logistic_test_class <- testing_data
logistic_train_class$Price <- as.factor(ifelse(logistic_train_class$Price <= mean_log , "Low", "High"))
logistic_test_class$Price <- as.factor(ifelse(logistic_test_class$Price <= mean_log , "Low", "High"))

#classification tree with class
fit <- tree(Price ~ Condo+Size+Rooms+Toilets+Suites+Parking+Elevator+Furnished+Swimming.Pool+Negotiatio
summary(fit)
```

```
##
## Classification tree:
## tree(formula = Price ~ Condo + Size + Rooms + Toilets + Suites +
##     Parking + Elevator + Furnished + Swimming.Pool + Negotiation.Type +
##     Latitude + Longitude, data = logistic_train_class)
## Variables actually used in tree construction:
## [1] "Negotiation.Type" "Size"             "Condo"            "Longitude"
## Number of terminal nodes:  6
## Residual mean deviance:  0.3 = 3070 / 10200
## Misclassification error rate: 0.0656 = 671 / 10230
```

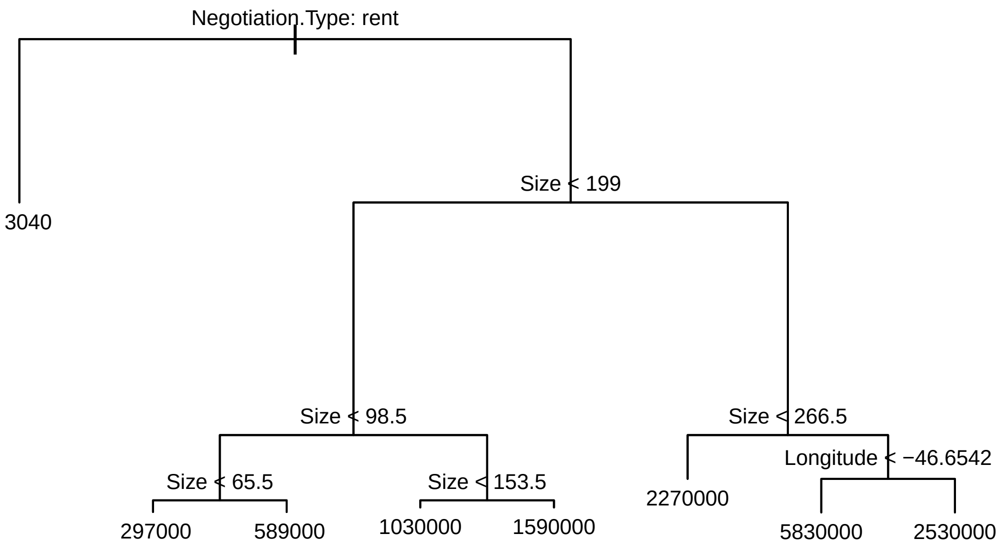
```
plot(fit)
text(fit, pretty = 0, cex = .7)
```



```
#classification tree without class
fit2<- tree(Price ~ Condo+Size+Rooms+Toilets+Suites+Parking+Elevator+Furnished+Swimming.Pool+Negotiatio
summary(fit2)
```

```
##
## Regression tree:
## tree(formula = Price ~ Condo + Size + Rooms + Toilets + Suites +
##     Parking + Elevator + Furnished + Swimming.Pool + Negotiation.Type +
##     Latitude + Longitude, data = training_data)
## Variables actually used in tree construction:
## [1] "Negotiation.Type" "Size"             "Longitude"
## Number of terminal nodes:  8
## Residual mean deviance:  5.79e+10 = 5.92e+14 / 10200
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -3720000   -18000    -1240        0     2960  4170000
```

```
plot(fit2)
text(fit2, pretty = 0, cex = .7)
```



```
#Confusion matrix table
yhat.testset <- predict(fit, logistic_test_class, type = "class")
y.testset <- logistic_test_class$Price
error <- table(yhat.testset, y.testset)
error
```

```
##             y.testset
## yhat.testset High  Low
##         High 1010  154
##         Low    79 2167
```

```
# Test accuracy rate
sum(diag(error))/sum(error)
```

```
## [1] 0.931672
```

```r
# Test error rate (Classification Error)
1-sum(diag(error))/sum(error)
```

```
## [1] 0.0683284
```

```r
yhat.testset2 <- predict(fit2, testing_data)
y.testset2 <- testing_data$Price
error <- table(yhat.testset2, y.testset2)
# Test accuracy rate
sum(diag(error))/sum(error)
```

```
## [1] 0.000293255
```

```r
# Test error rate (Classification Error)
1-sum(diag(error))/sum(error)
```

```
## [1] 0.999707
```

Since the test error rate of the classification tree model with class is much lower than classification tree model without class, the classification model with class is significant.

Pruned Tree

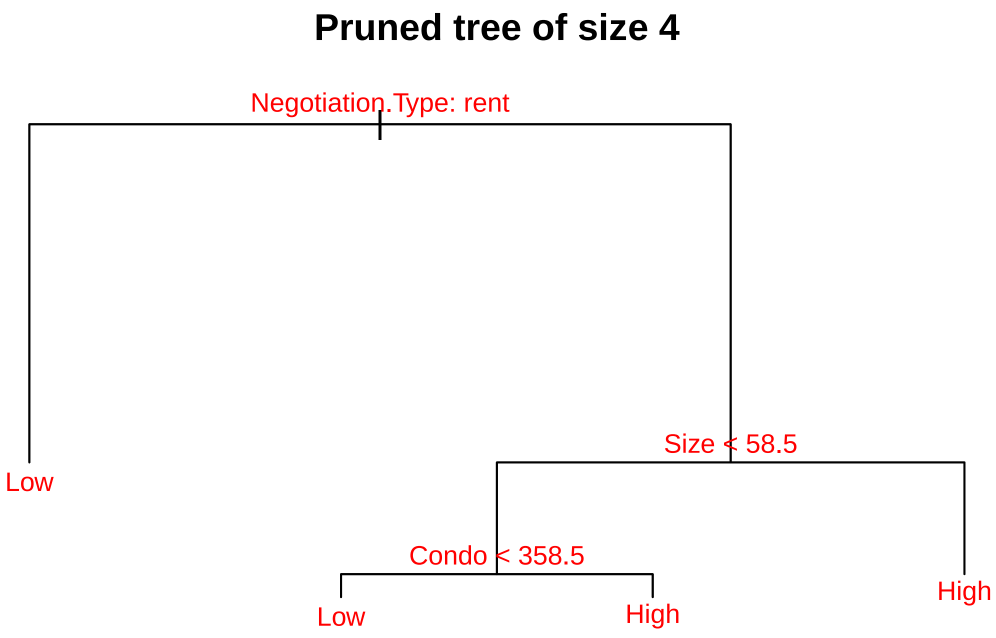
```r
prune <- prune.tree(fit, k = 0:50, method = "misclass")
# Best size
best.prune <- prune$size[which.min(prune$dev)]
best.prune
```

```
## [1] 4
```

```r
#From the output, the ‘best’ size is 4 since this number of terminal nodes corresponds to the smallest 
set.seed(3)
cv <- cv.tree(fit, FUN=prune.misclass, K=50)
# Print out cv
cv
```

```
## $size
## [1] 6 4 3 2 1
##
## $dev
## [1]  725  725  879 1596 3208
##
## $k
## [1] -Inf    0  204  720 1613
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```r
# Prune tree
pt.prune <- prune.misclass (fit, best=best.prune)
# Plot pruned tree
plot(pt.prune)
text(pt.prune, pretty=0, col = "red", cex = .8)
title("Pruned tree of size 4")
```



```r
# Predict on test set
pred.pt.prune <- predict(pt.prune, logistic_test_class, type="class")
# Obtain confusion matrix
err.pt.prune <- table(pred.pt.prune, y.testset)
err.pt.prune
```

```
##              y.testset
## pred.pt.prune High  Low
##          High 1010  154
##          Low    79 2167
```

```r
# Test accuracy rate
sum(diag(err.pt.prune))/sum(err.pt.prune)
```

```
## [1] 0.931672
```

```
# Test error rate (Classification Error)
1-sum(diag(err.pt.prune))/sum(err.pt.prune)
```

```
## [1] 0.0683284
```

Even if I applied pruned tree to get better model, the test error rate remains same.

Bagging Tree and Random Forest model

```
library(dplyr)
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.6.2
```

```
## randomForest 4.6-14
```

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

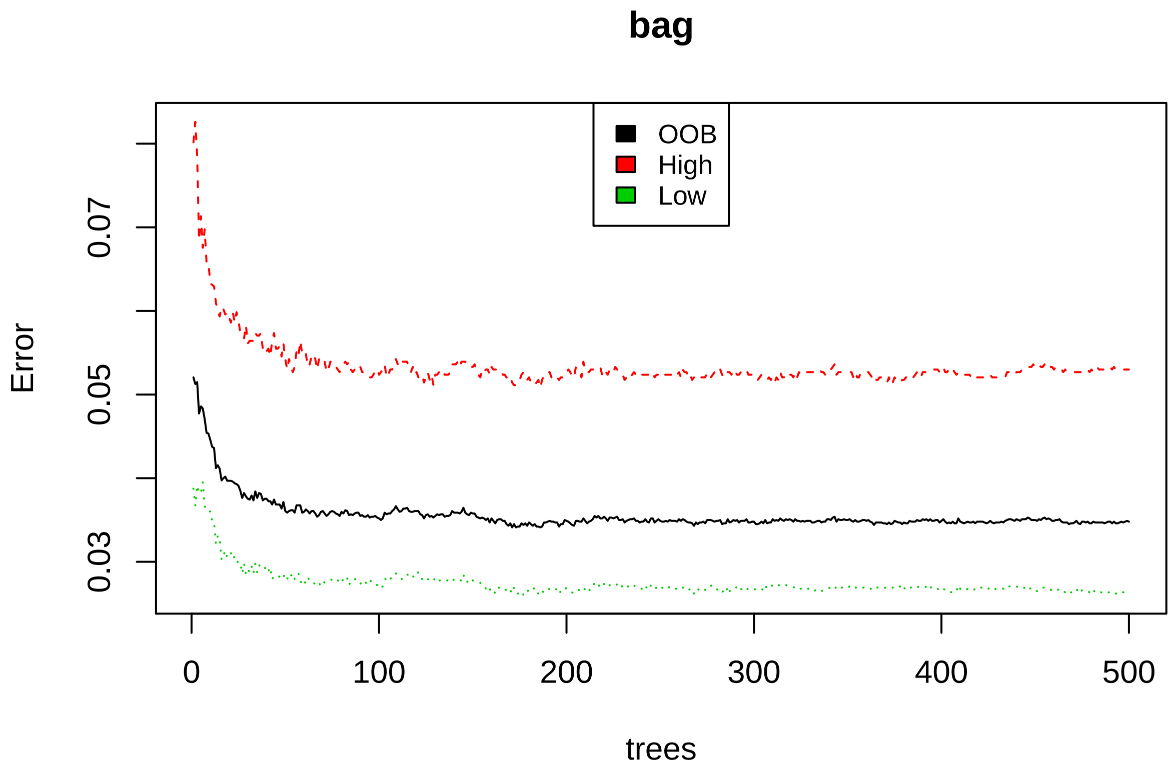
```
##
## Attaching package: 'randomForest'
```

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

```
library(tree)
#Bagging Tree
bag <- randomForest(Price~Condo+Size+Rooms+Toilets+Suites+Parking+Elevator+Furnished+Swimming.Pool+Nego
bag
```

```
##
## Call:
##  randomForest(formula = Price ~ Condo + Size + Rooms + Toilets +      Suites + Parking + Elevator +
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 12
##
##         OOB estimate of  error rate: 3.48%
## Confusion matrix:
##      High  Low class.error
## High 3038  170   0.0529925
## Low   186 6836   0.0264882
```

```
plot(bag)
legend("top", colnames(bag$err.rate),col=1:4,cex=0.8,fill=1:4)
```



```
#confusion matrix
yhat.bag = predict(bag, newdata = logistic_test_class)
bag.err = table(pred = yhat.bag, truth = logistic_test_class$Price)
test.bag.err = 1 - sum(diag(bag.err))/sum(bag.err)
test.bag.err
```

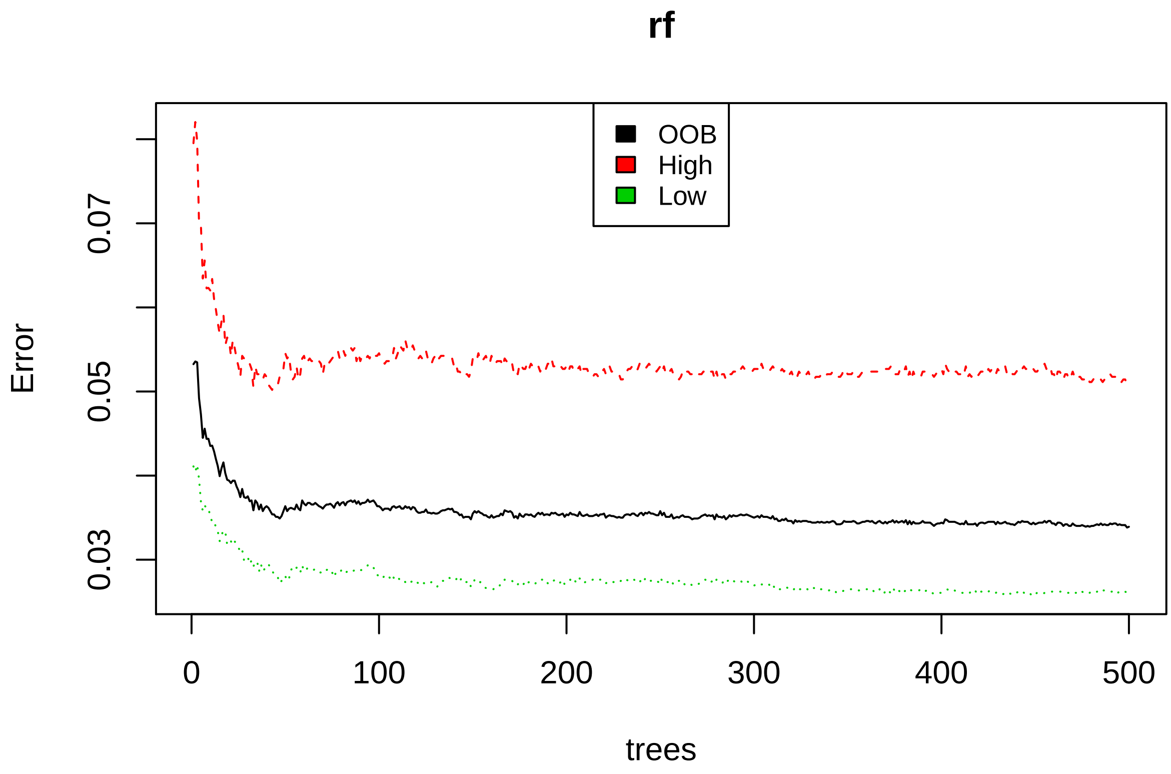
```
## [1] 0.0316716
```

The test set error rate associated with the bagged classification tree is 0.0316

```
#Random Forest
rf <- randomForest(Price~Condo+Size+Rooms+Toilets+Suites+Parking+Elevator+Furnished+Swimming.Pool+Negot
rf
```

```
##
## Call:
##  randomForest(formula = Price ~ Condo + Size + Rooms + Toilets +      Suites + Parking + Elevator + 
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 12
##
##         OOB estimate of  error rate: 3.39%
## Confusion matrix:
##      High  Low class.error
## High 3044  164   0.0511222
## Low   183 6839   0.0260610
```

```r
plot(rf)
legend("top", colnames(rf$err.rate),col=1:4,cex=0.8,fill=1:4)
```



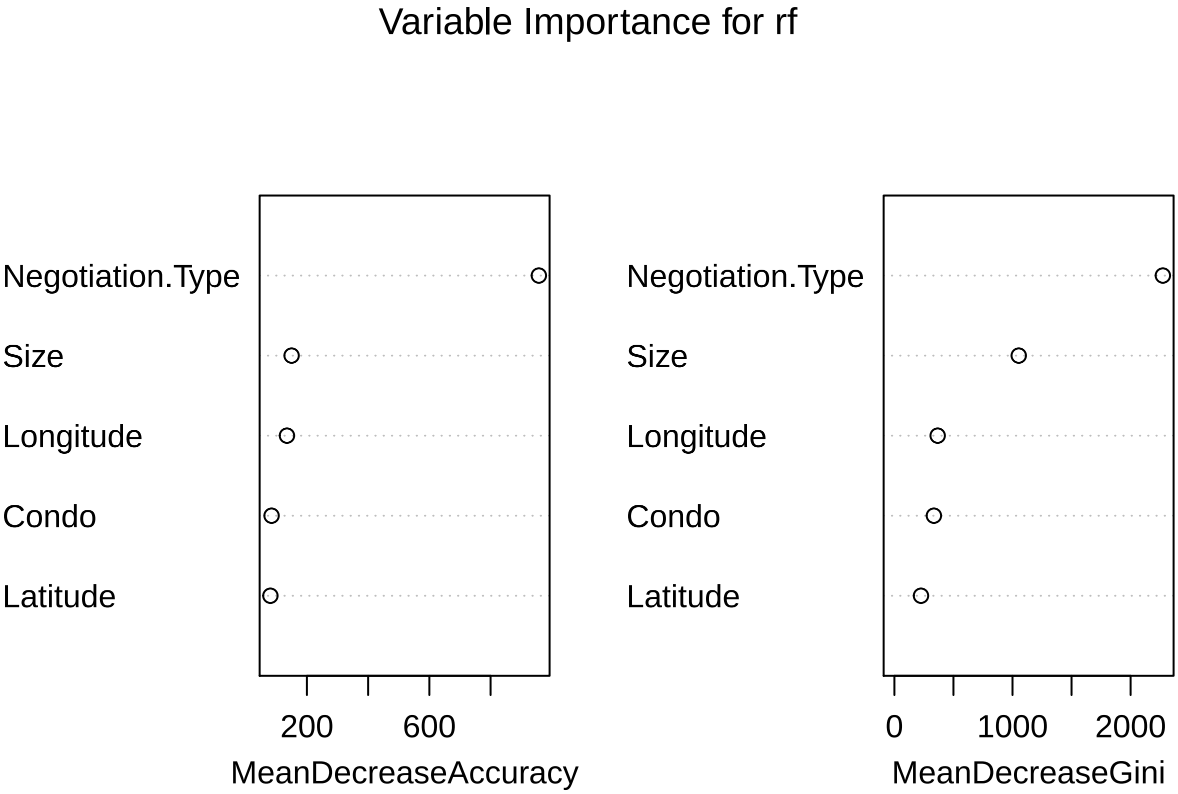
```r
yhat.rf = predict (rf, newdata = logistic_test_class)
# Confusion matrix
rf.err = table(pred = yhat.rf, truth = logistic_test_class$Price)
test.rf.err = 1 - sum(diag(rf.err))/sum(rf.err)
# Test error rate
test.rf.err
```

```
## [1] 0.0322581
```

```r
importance(rf)
```

```
##                     High       Low MeanDecreaseAccuracy MeanDecreaseGini
## Condo            5.98763  81.77486              84.5799         334.4486
## Size            73.40752 119.29500             150.0157        1052.8905
## Rooms           26.78820  22.38165              35.5415          29.0150
## Toilets          5.99065   9.61840              11.2781          11.7216
## Suites          15.28685   7.60921              16.8849          10.8644
## Parking         11.90394  13.10237              18.3228          15.8911
## Elevator         7.30762  18.94925              20.4841          16.0372
## Furnished        7.21419  21.77851              22.0330          14.9544
## Swimming.Pool   34.62240  40.71451              50.8982          53.7812
## Negotiation.Type 812.10487 764.64823             957.7058        2273.0749
## Latitude           59.90652  52.10808              80.6961         225.3113
## Longitude          73.54912 115.19641             134.8107         366.5115
```

```
varImpPlot(rf, sort=T, main="Variable Importance for rf", n.var=5)
```



The results indicate that across all of the trees considered in the random forest, the Negotiation.Type is by far the most important variable in terms of Model Accuracy and Size is the secondary most important variable.

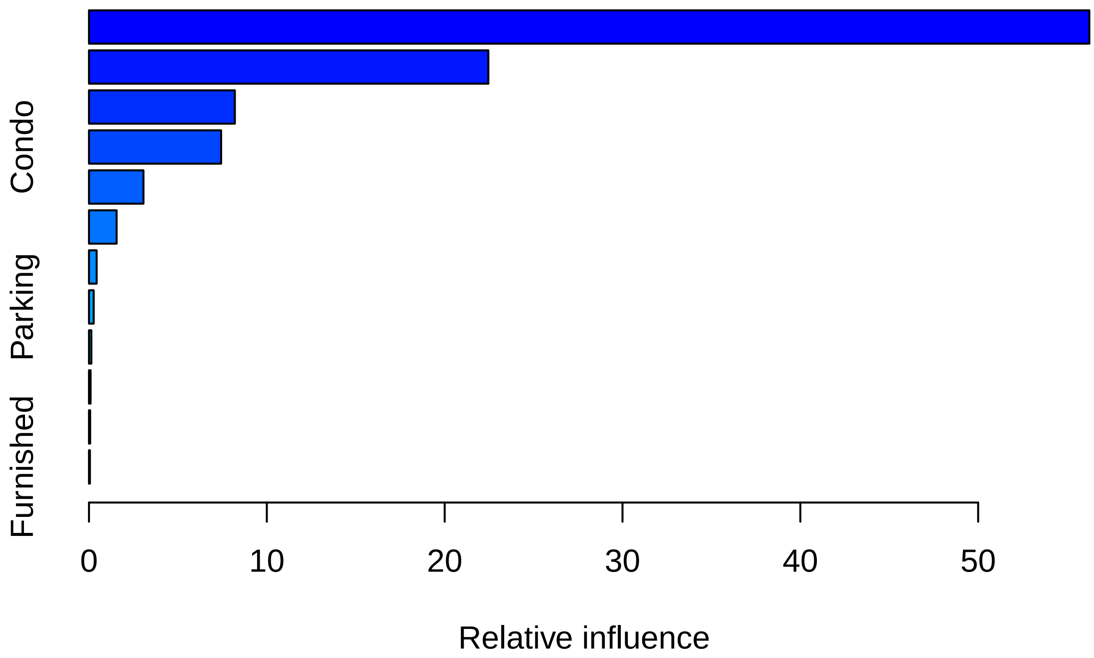
Boosting

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 3.6.2
```

```
## Loaded gbm 2.1.5
```

```
set.seed(1)
boost <- gbm(ifelse(Price=="High",1,0)~Condo+Size+Rooms+Toilets+Suites+Parking+Elevator+Furnished+Swimmi
summary(boost)
```



```
##                               var    rel.inf
## Negotiation.Type Negotiation.Type 56.2521617
## Size                         Size 22.4572285
## Longitude               Longitude  8.2025809
## Condo                       Condo  7.4298397
## Latitude                 Latitude  3.0638884
## Swimming.Pool       Swimming.Pool  1.5546862
## Rooms                       Rooms  0.4318611
## Parking                   Parking  0.2658957
## Toilets                   Toilets  0.1332400
## Elevator                 Elevator  0.0900062
## Suites                     Suites  0.0666510
## Furnished               Furnished  0.0519606
```

```
yhat.boost <- predict(boost, newdata = logistic_test_class, n.trees=500)
# Confusion matrix
boost.err <- table(pred = yhat.boost, truth = logistic_test_class$Price)
test.boost.err <-1-sum(diag(boost.err))/sum(boost.err)
test.boost.err
```

```
## [1] 0.999707
```

Testing error seems not okay with boosting model because it is over 0.9.

Conclusion

According to all of tests, Randomforest model has the lowest test error rate, which means the most significant model. All predictors excluding New, District, Property.Type are included because the model has the lowest AIC when it has all of the predictors. This study has some limitation because the dataset has information of very narrow area. It might have different results if the dataset includes information from other areas in Brazil. Also, this study is not effectively explained due to lack of other predictors such as existence of transfortation, number of criminal cases near the area, etc. This study will result in having different conclusion if the dataset is expanded.