BA810: Supervised Machine Learning

Team Project: Predicting Prices of NYC Airbnbs

Team 8

- Eunjin Jeong (U83102024)
- Lujain Alqassar (U85675324)
- Maria Stella Vardanega (U06003669)
- Yipeng Guo (U17061528)
- Yongxian Lun (U22529000)

Objective: Help Consumer Accurately Select Satisfactory Airbnb

According to the data set, we have their location, districts, number of reviews and other related factors. From consumers' perspective, we are thinking if we can reasonably predict the unknown prices of NYC Airbnbs, that will be a great help for people travelling to NYC. This will help them to accurately select the satisfactory Airbnb according to the price and region.

Dataset: New York City Airbnb Open Data (Data source from http://insideairbnb.com/ (http://insideairbnb.com/))

This data file includes all needed information to find out more about hosts, geographical availability, necessary metrics to make predictions and draw conclusions. And this public data set is part of Airbnb.

Summary Of the Dataset

Originally, after data cleansing, we have 38,821 rows and 16 columns including price in our data set.

To explore the factors that powerfully influence the price of Airbnb, we chose to narrow the features of our dataset down to the following variables:

- latitude
- longitude
- minimum_nights: minimum nights consumers should reserve
- number of reviews: total number of reviews this Airbnb received
- reviews per month: average number of reviews this Airbnb received monthly
- calculated_host_listing_count: amount of Airbnb per host
- availability_365: number of days when Airbnb is available for booking
- neighborhood generally: neighborhood this Airbnb lies (Manhattan, Brooklyn, Bronx, Queens – we make them into dummy variables)
- room_type: Airbnb space type (Entire home/apt or private room – we make them into dummy variables)

1. Basic Descriptive Analyses

1.1 Data Load and Cleansing

```
#Library Loading
library(data.table)
library(ggplot2)
library(ggthemes)
library(scales)
theme_set(theme_bw())
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-2
```

```
library(rpart)
library(randomForest)
```

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

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

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

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

```
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
library(dplyr)
```

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

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

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

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

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

```
library(scales)
```

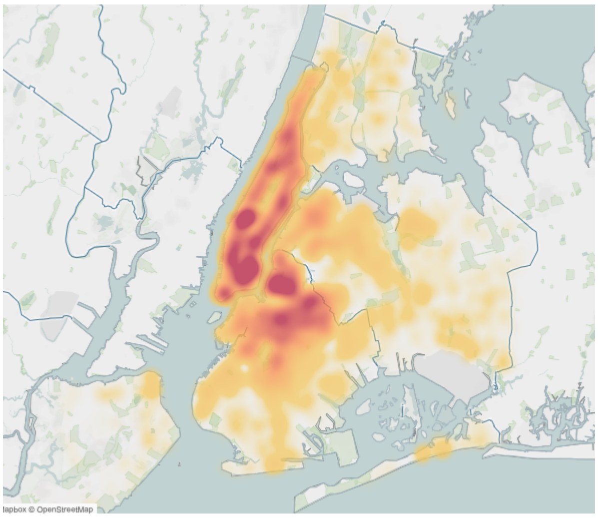
```
#Data Loading
nyc <- fread('/Users/carosnote/Desktop/nyc.csv', stringsAsFactors = T)

#Deleting NA values and price equals 0
nyc [nyc ==""] <-NA
nyc <- nyc[complete.cases(nyc),]
str(nyc)
```

```
## Classes 'data.table' and 'data.frame':   38821 obs. of  16 variables:
##  $ id                            : int  2539 2595 3831 5022 5099 5121 5178 5203 52
38 5295 ...
##  $ name                          : Factor w/ 47895 levels ""," Private 1 bdrm Leff
erts Gr, BK apt",..: 12322 37447 14777 18686 24439 7941 24471 14700 17040 5503 ...
##  $ host_id                       : int  2787 2845 4869 7192 7322 7356 8967 7490 75
49 7702 ...
##  $ host_name                     : Factor w/ 11453 levels "","'Cil","(Ari) HENRY L
EE",..: 4991 4787 6205 5925 1933 3544 9640 6869 1231 6026 ...
##  $ neighbourhood_group           : Factor w/ 5 levels "Bronx","Brooklyn",..: 2 3 2
3 3 2 3 3 3 3 ...
##  $ neighbourhood                 : Factor w/ 221 levels "Allerton","Arden Height
s",..: 109 128 42 62 138 14 96 203 36 203 ...
##  $ latitude                      : num  40.6 40.8 40.7 40.8 40.7 ...
##  $ longitude                     : num  -74 -74 -74 -73.9 -74 ...
##  $ room_type                     : Factor w/ 3 levels "Entire home/apt",..: 2 1 1
1 1 2 2 2 1 1 ...
##  $ price                         : int  149 225 89 80 200 60 79 79 150 135 ...
##  $ minimum_nights                : int  1 1 1 10 3 45 2 2 1 5 ...
##  $ number_of_reviews             : int  9 45 270 9 74 49 430 118 160 53 ...
##  $ last_review                   : IDate, format: "2018-10-19" "2019-05-21" ...
##  $ reviews_per_month             : num  0.21 0.38 4.64 0.1 0.59 0.4 3.47 0.99 1.33
0.43 ...
##  $ calculated_host_listings_count: int  6 2 1 1 1 1 1 1 4 1 ...
##  $ availability_365              : int  365 355 194 0 129 0 220 0 188 6 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
nyc <- nyc[nyc$price != 0]

#Creating dummy variables
#neighborhood_group
nyc$brooklyn <- ifelse(nyc$neighbourhood_group == "Brooklyn", 1, 0)
nyc$manhattan <- ifelse(nyc$neighbourhood_group == "Manhattan", 1, 0)
nyc$queens <- ifelse(nyc$neighbourhood_group == "Queens", 1, 0)
nyc$bronx <- ifelse(nyc$neighbourhood_group == "Bronx", 1, 0)
#room_Type
nyc$private_room <- ifelse(nyc$room_type == "Private room", 1, 0)
nyc$shared_room <- ifelse(nyc$room_type == "Shared room", 1, 0)
```

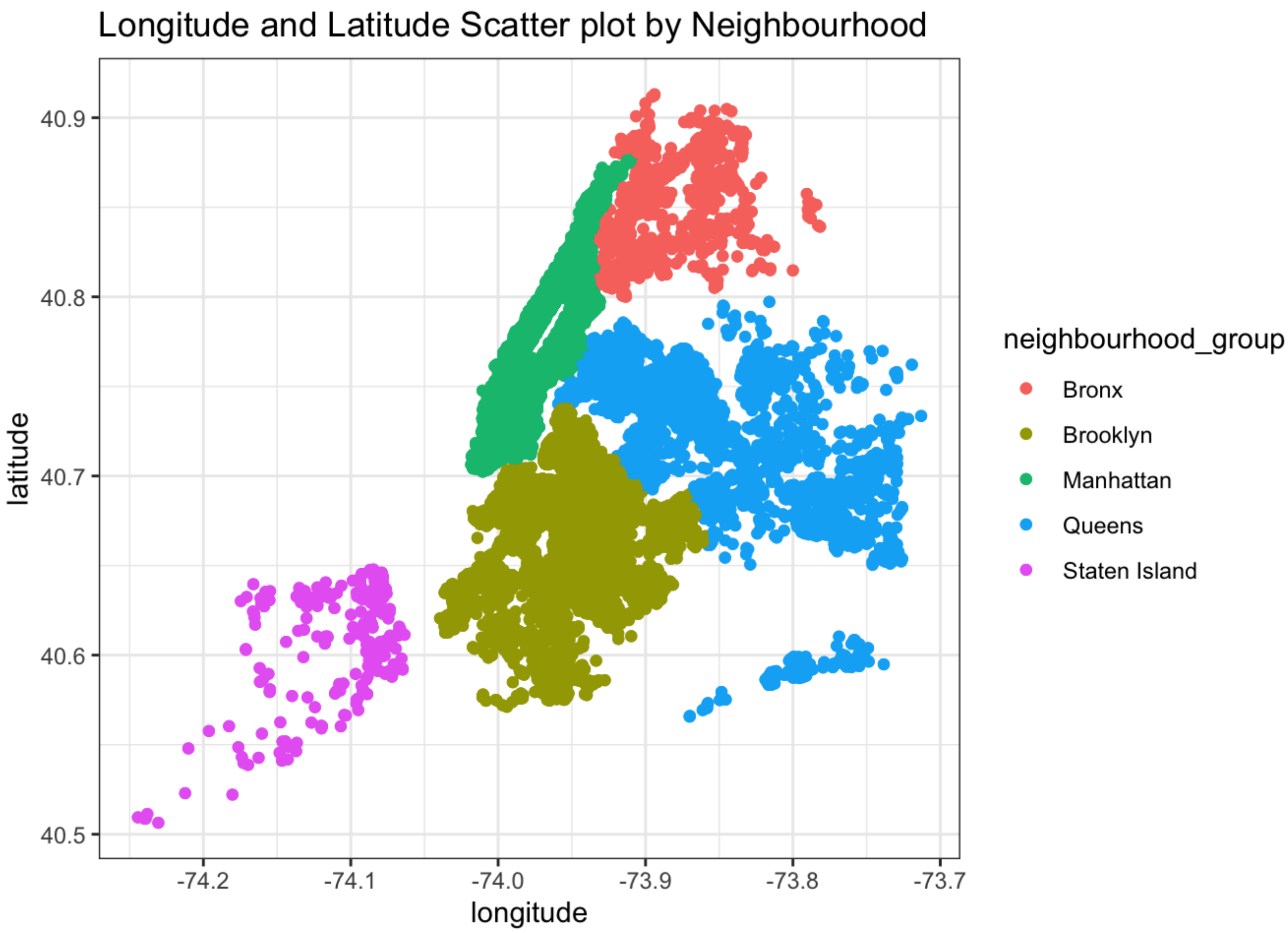
1.2 New York Airbnb Price Heat Map

```
knitr::include_graphics("/Users/carosnote/desktop/price heat-map.png")
```



1.3 Longitude and Latitude Scatterplot by Neighbourhoods

```
ggplot(nyc, aes(x=longitude, y=latitude, col=neighbourhood_group)) +
  geom_point() +
  geom_jitter() +
  labs(title="Longitude and Latitude Scatter plot by Neighbourhood")
```

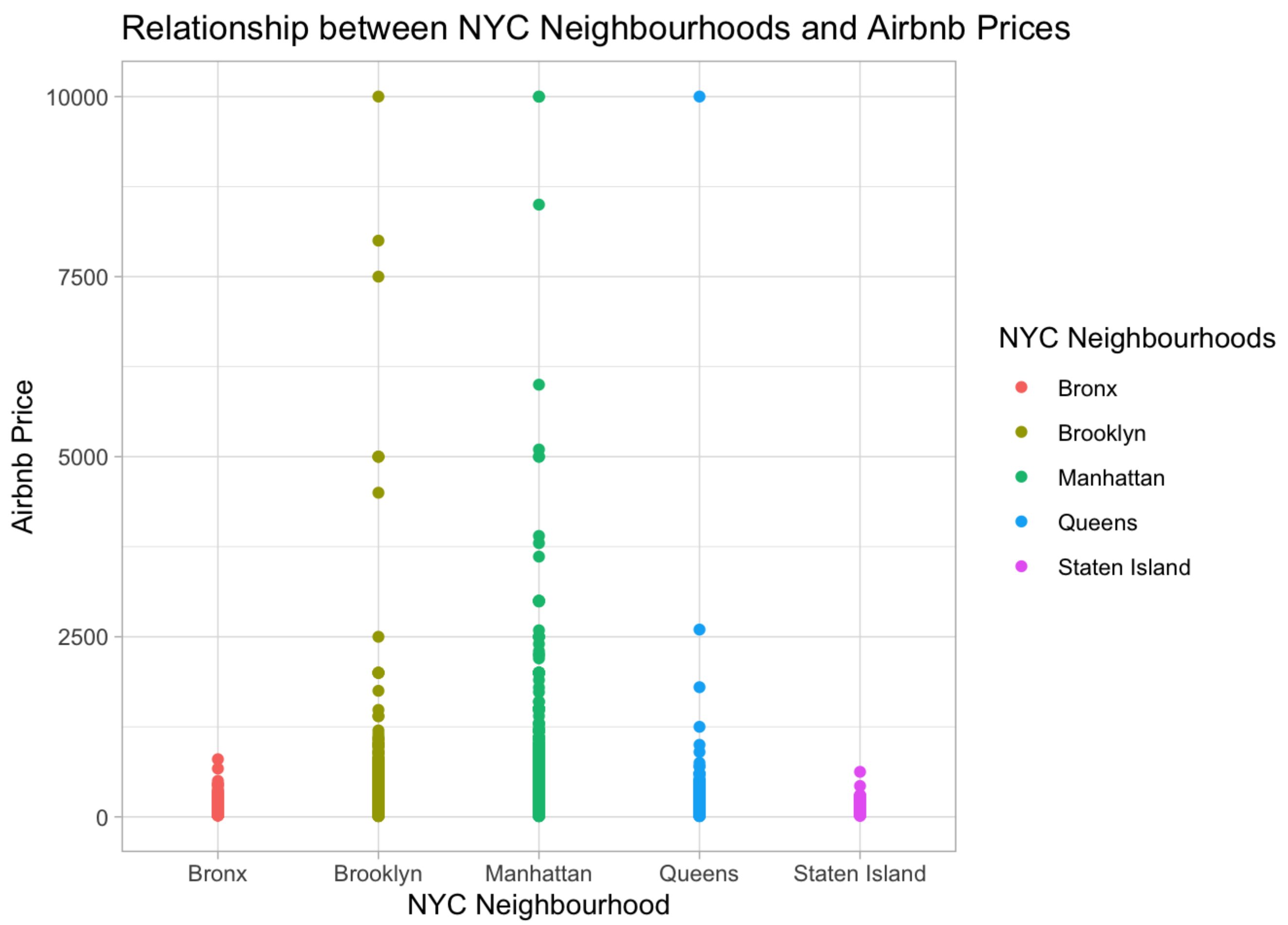


1.4 Relationship Between NYC Neighbourhoods and Airbnb Price

```
#Calculating average price
nyc %>%
  group_by(neighbourhood_group) %>%
  summarise(average_price=mean(price))
```

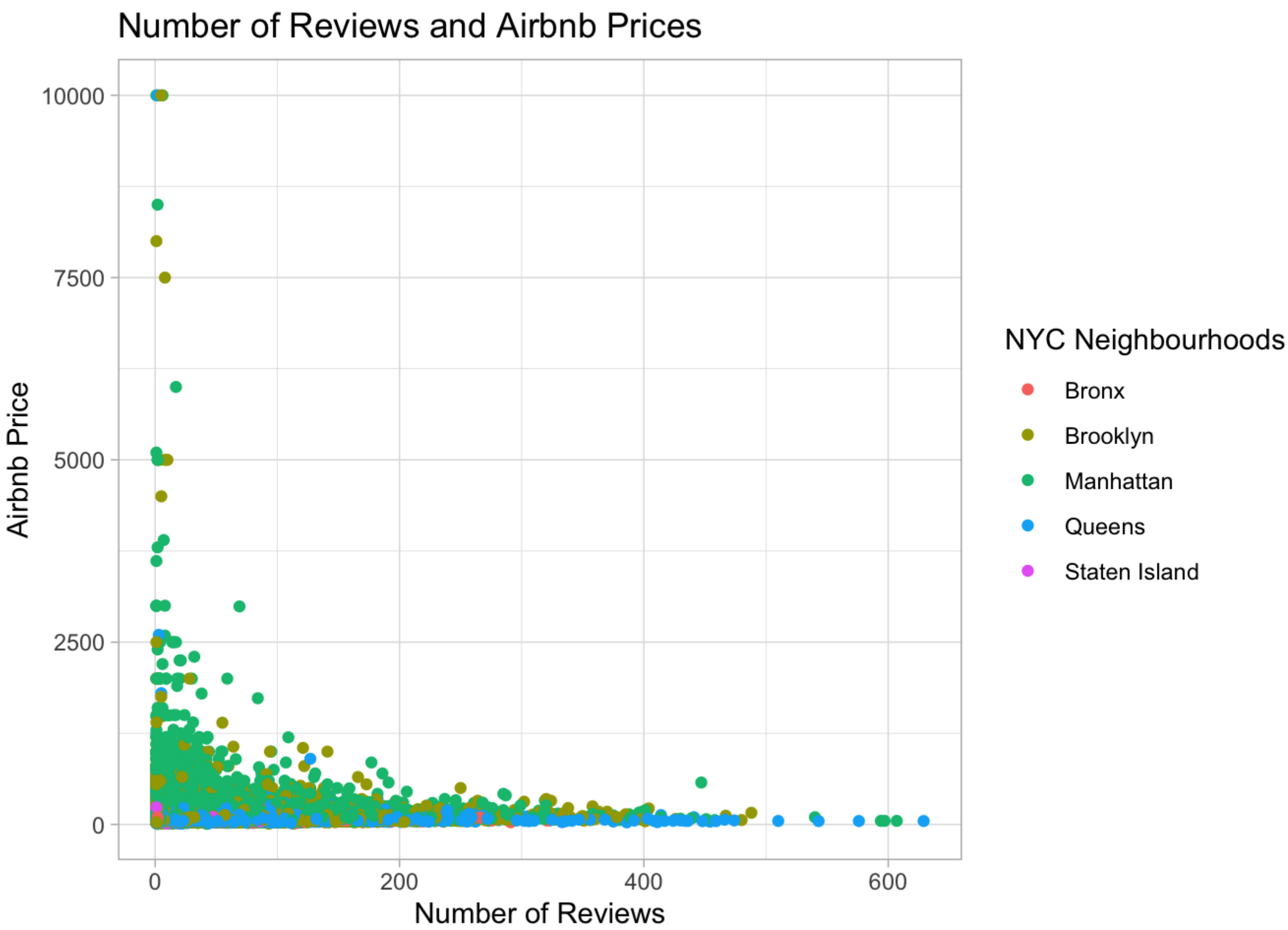
```
## # A tibble: 5 × 2
##   neighbourhood_group average_price
##   <fct>                       <dbl>
## 1 Bronx                        79.6
## 2 Brooklyn                    122.
## 3 Manhattan                   180.
## 4 Queens                       95.8
## 5 Staten Island                90.0
```

```
#Making plot
ggplot(nyc, aes(x=neighbourhood_group, y=price, color=neighbourhood_group)) +
  geom_point() +
  theme_light() +
  scale_colour_discrete("NYC Neighbourhoods") +
  labs(title="Relationship between NYC Neighbourhoods and Airbnb Prices", y="Airbnb P
rice", x="NYC Neighbourhood")
```



1.5 Relationship Between Number of Reviews and Airbnb Price

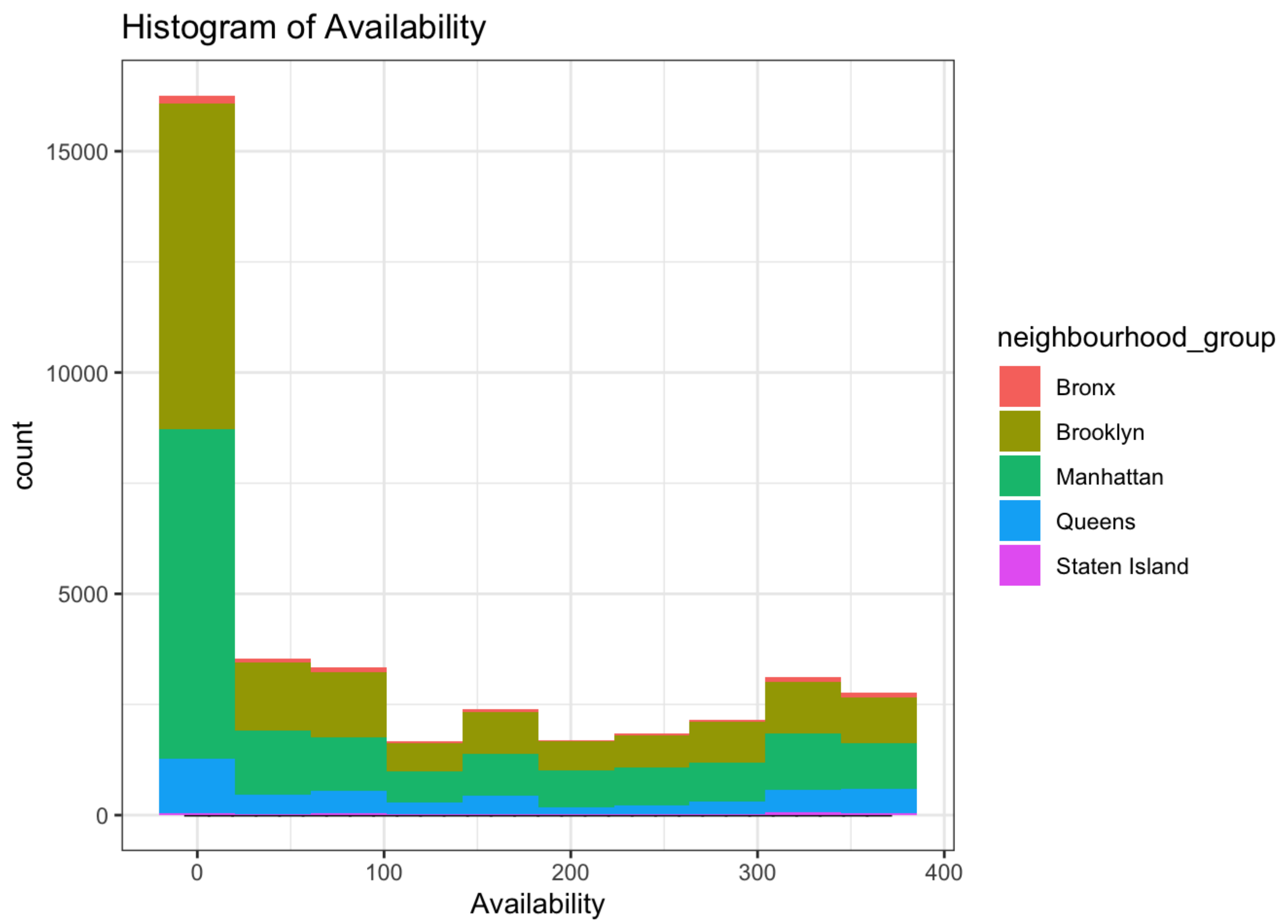
```
ggplot(nyc, aes(x=number_of_reviews, y=price, color=neighbourhood_group)) +
  geom_point() +
  theme_light() +
  scale_colour_discrete("NYC Neighbourhoods") +
  labs(title="Number of Reviews and Airbnb Prices", y="Airbnb Price", x="Number of Re
views")
```



1.6 Histogram of the Number of Availability

```
ggplot(nyc, aes(x=availability_365, fill = neighbourhood_group)) +
  geom_histogram(color="black", fill="blue") +
  stat_bin(bins=10) +
  labs(title="Histogram of Availability", y="count", x="Availability")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



2. Apply Machine Learning Models

2.1 Create Matrix and Target & Split Train and Test Set

```
set.seed(1220)
dt = sort(sample(nrow(nyc), nrow(nyc)*.8))
train<- as.data.frame(nyc[dt,])
test<- as.data.frame(nyc[-dt,])
f1 <- as.formula(price ~ latitude + longitude +
                   minimum_nights + number_of_reviews + reviews_per_month +
                   calculated_host_listings_count + availability_365 + brooklyn +
                   manhattan + queens + bronx + private_room + shared_room)

x1_train <- model.matrix(f1, train)[, -1]
y_train <- train$price
x1_test <- model.matrix(f1, test)[, -1]
y_test <- test$price
```

2.2 Linear Regression

```
set.seed(1220)
#f1--Important--contain all the features
fit.lm1 <- lm(f1, train)

#MSE on the training data
yhat.train.lm1 <- predict(fit.lm1)
mse.train.lm1 <- mean((y_train - yhat.train.lm1)^2)

#MSE on the test data
yhat.test.lm1 <- predict(fit.lm1, test)
mse.test.lm1 <- mean((y_test - yhat.test.lm1)^2)

summary(fit.lm1)
```

```
##
## Call:
## lm(formula = f1, data = train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -202.7  -52.7  -17.8   17.1 9974.4
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -2.744e+04  3.038e+03  -9.035  < 2e-16 ***
## latitude                       -1.117e+02  3.002e+01  -3.721 0.000198 ***
## longitude                      -4.332e+02  3.405e+01 -12.721  < 2e-16 ***
## minimum_nights                 -3.056e-01  6.155e-02  -4.965 6.90e-07 ***
## number_of_reviews              -1.886e-01  2.489e-02  -7.577 3.63e-14 ***
## reviews_per_month              -3.091e-01  7.170e-01  -0.431 0.666391
## calculated_host_listings_count -8.010e-02  3.965e-02  -2.020 0.043379 *
## availability_365                1.697e-01  8.147e-03  20.834  < 2e-16 ***
## brooklyn                        1.196e+02  1.208e+01   9.896  < 2e-16 ***
## manhattan                       1.667e+02  1.230e+01  13.555  < 2e-16 ***
## queens                          1.393e+02  1.367e+01  10.193  < 2e-16 ***
## bronx                           1.309e+02  1.553e+01   8.429  < 2e-16 ***
## private_room                   -1.022e+02  2.049e+00 -49.873  < 2e-16 ***
## shared_room                    -1.354e+02  6.855e+00 -19.752  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 173.4 on 31034 degrees of freedom
## Multiple R-squared:  0.1318, Adjusted R-squared:  0.1315
## F-statistic: 362.5 on 13 and 31034 DF,  p-value: < 2.2e-16
```

```
mse.test.lm1
```

```
## [1] 50951.58
```

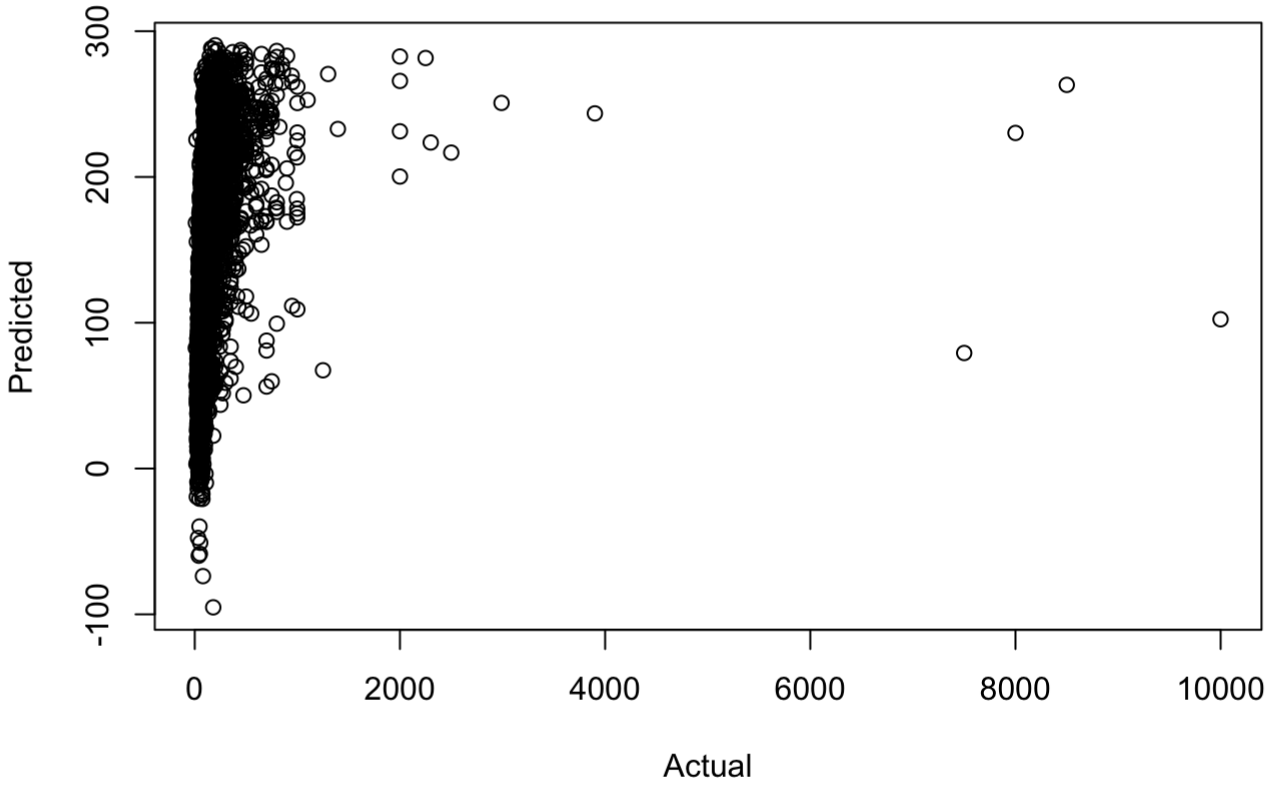
```
mse.train.lm1
```

```
## [1] 30059.35
```

Rationale: We fit the linear regression model to the data set and found out MSE train is about 30,059 and MSE test is 50,059. Both of the MSEs are really large. To determine whether a linear model is better, we will compare it to other models.

Showing Results:

```
plot(y_test, yhat.test.lm1, xlab='Actual', ylab='Predicted')
```

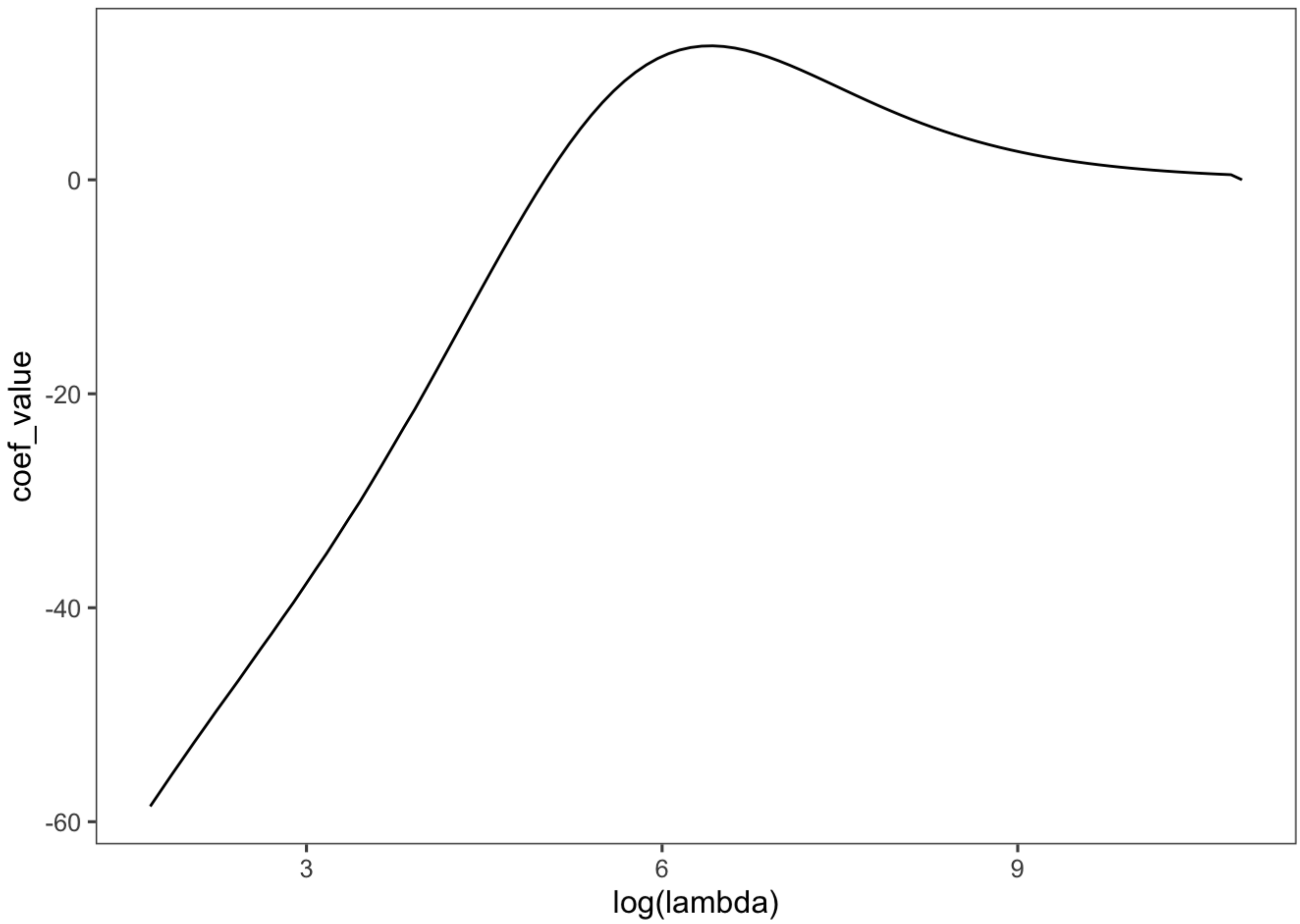


2.3 Ridge Regression

```
set.seed(1220)
#Invoking ridge regression:
fit.ridge <- glmnet(x1_train, y_train, alpha = 0)

#Looking at the smallest lamdba:
min(fit.ridge$lambda)
```

```
## [1] 5.375646
```

```
ridge.coef <- predict(fit.ridge,
                      type = "coefficients",
                      s = fit.ridge$lambda)

#Plotting the coefficients as a function of lambda:
to_plot <- data.table(
  lambda = fit.ridge$lambda,
  coef_value = ridge.coef[2, ]
)
ggplot(to_plot, aes(log(lambda), coef_value)) +
  geom_line() +
  theme_few()
```




```r
#Selecting the best value lambda for the hyper-parameter to use for our model:
fit.ridge <- cv.glmnet(x1_train, y_train, alpha = 0, nfolds = 10)

#Computing our train MSEs:
yhat.train.ridge <- predict(fit.ridge, x1_train, s = fit.ridge$lambda.min)
mse.train.ridge <- mean((y_train - yhat.train.ridge)^2)

#Computing our test MSEs:
yhat.test.ridge <- predict(fit.ridge, x1_test, s = fit.ridge$lambda.min)
mse.test.ridge <- mean((y_test - yhat.test.ridge)^2)
```

```r
#Inspecting the beta coefficients of our model:
coef(fit.ridge)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                                            s1
## (Intercept)                      1.422186e+02
## latitude                         1.286825e-34
## longitude                       -6.453201e-34
## minimum_nights                   2.525860e-37
## number_of_reviews               -1.491021e-37
## reviews_per_month               -3.819269e-36
## calculated_host_listings_count   4.066603e-37
## availability_365                 1.149204e-37
## brooklyn                        -3.649806e-35
## manhattan                        6.645040e-35
## queens                          -5.248613e-35
## bronx                           -6.316957e-35
## private_room                    -1.090672e-34
## shared_room                     -8.087889e-35
```

```
mse.test.ridge
```

```
## [1] 51067.98
```

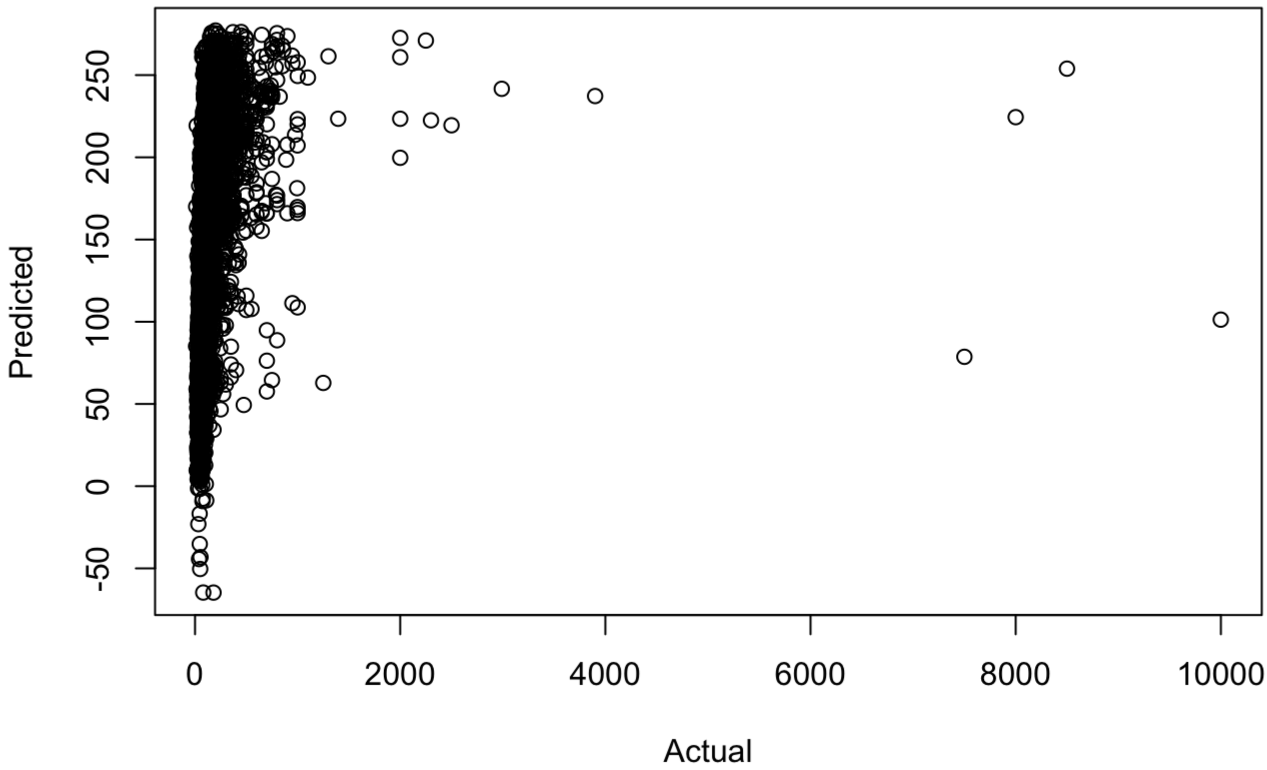
```
mse.train.ridge
```

```
## [1] 30135.35
```

Rationale: After fitting the model to the ridge regression model, we calculated both the MSE train and MSE test to see how it performs from the data we have to data we haven't seen yet. The MSE train was 30,135 and the MSE test was 51,068. Compared to other models, it would most likely not be the best option.

Showing Results:

```
plot(y_test, yhat.test.ridge, xlab='Actual', ylab='Predicted')
```

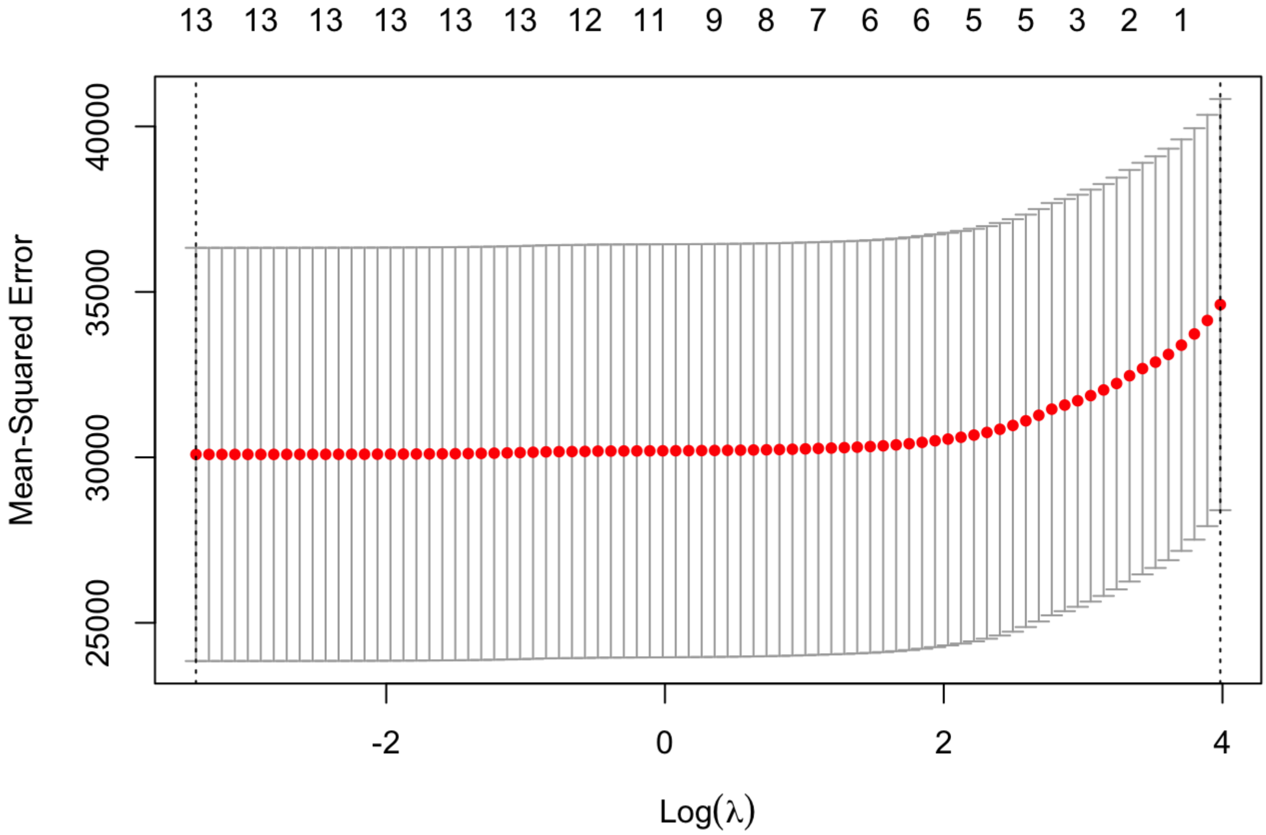


2.3 Lasso Regression

```
set.seed(1220)
fit.lasso <- cv.glmnet(x1_train, y_train, alpha=1, nfolds = 10)

best_lambda <- fit.lasso$lambda.min
best_lambda
```

```
## [1] 0.03455505
```

```
plot(fit.lasso)
```




```
# Rebuilding the model with the best lambda value identified:
lasso_best <- glmnet(x1_train, y_train, alpha=1, lambda=best_lambda)

# Inspecting Beta Coefficients:
coef(lasso_best)
```

```
## 14 x 1 sparse Matrix of class "dgCMatrix"
##                                          s0
## (Intercept)                   -2.647830e+04
## latitude                      -1.047882e+02
## longitude                     -4.164778e+02
## minimum_nights                -3.012754e-01
## number_of_reviews             -1.874846e-01
## reviews_per_month             -3.237068e-01
## calculated_host_listings_count -7.507583e-02
## availability_365               1.685378e-01
## brooklyn                       1.059670e+02
## manhattan                      1.529005e+02
## queens                         1.241639e+02
## bronx                          1.149781e+02
## private_room                  -1.022651e+02
## shared_room                   -1.350724e+02
```

```
#Computing our tain MSEs:
yhat.train.lasso <- predict(lasso_best, s=best_lambda, newx = x1_train)
mse.train.lasso <- mean((y_train - yhat.train.lasso)^2)

#Computing our test MSEs:
yhat.test.lasso <- predict(lasso_best, s=best_lambda, newx = x1_test)
mse.test.lasso <- mean((y_test - yhat.test.lasso)^2)


mse.test.lasso
```

```
## [1] 50959.44
```

```
mse.train.lasso
```

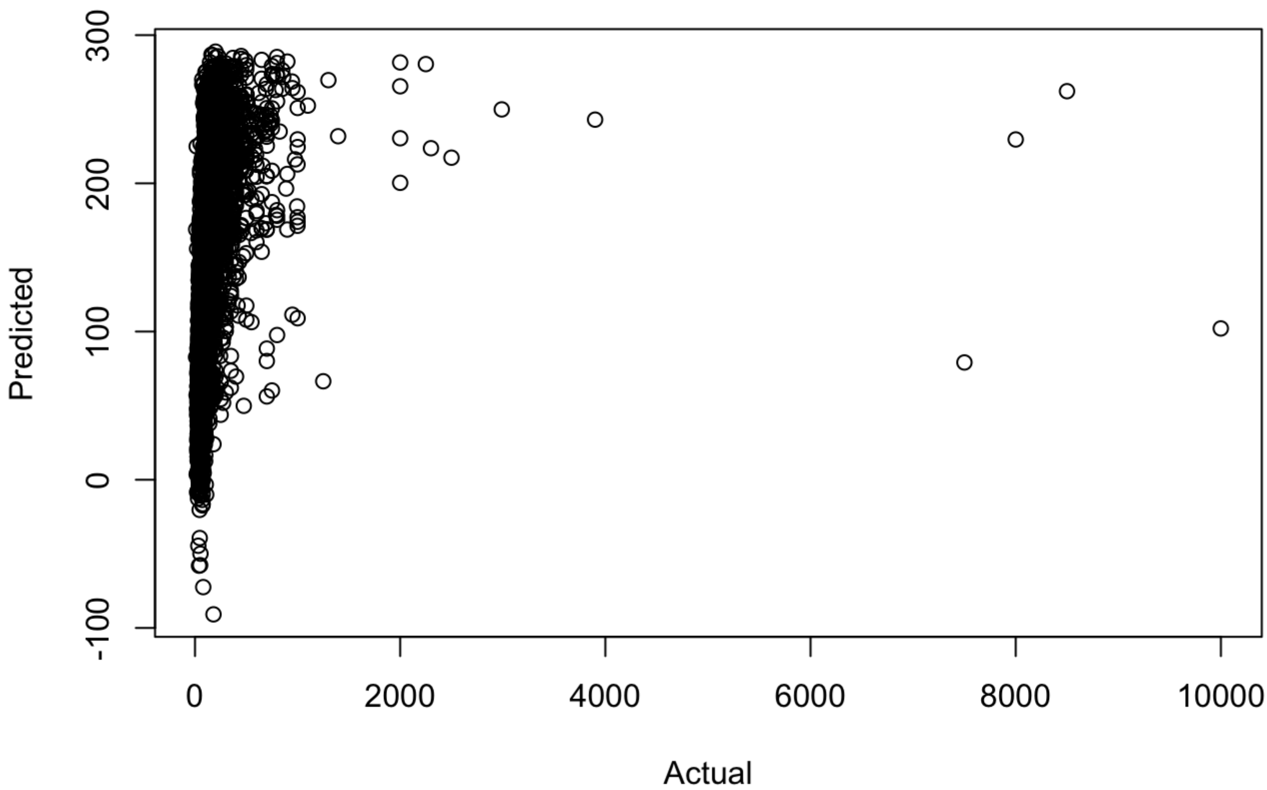
```
## [1] 30060.62
```

Rationale: After fitting the model to the lasso regression, we calculated the MSE train and test to see how they compared to other models. The MSE test was about 50,959 and the MSE train was about 30,061.

These will both be interpreted in the context of the other MSEs found for other models.

Showing Results:

```
plot(y_test, yhat.test.lasso, xlab='Actual', ylab='Predicted')
```



2.4 Decision Tree

```
set.seed(1220)
fit.tree <- rpart(f1,
                  train,
                  control = rpart.control(cp = 0.001))
```

```
# Computing our train MSEs:
yhat.train.tree <- predict(fit.tree, train)
mse.train.tree <- mean((yhat.train.tree - y_train) ^ 2)

# Computing our trst MSEs:
yhat.test.tree <- predict(fit.tree, test)
mse.test.tree <- mean((yhat.test.tree - y_test) ^ 2)
```

```
mse.train.tree
```

```
## [1] 25243.88
```

```
mse.test.tree
```

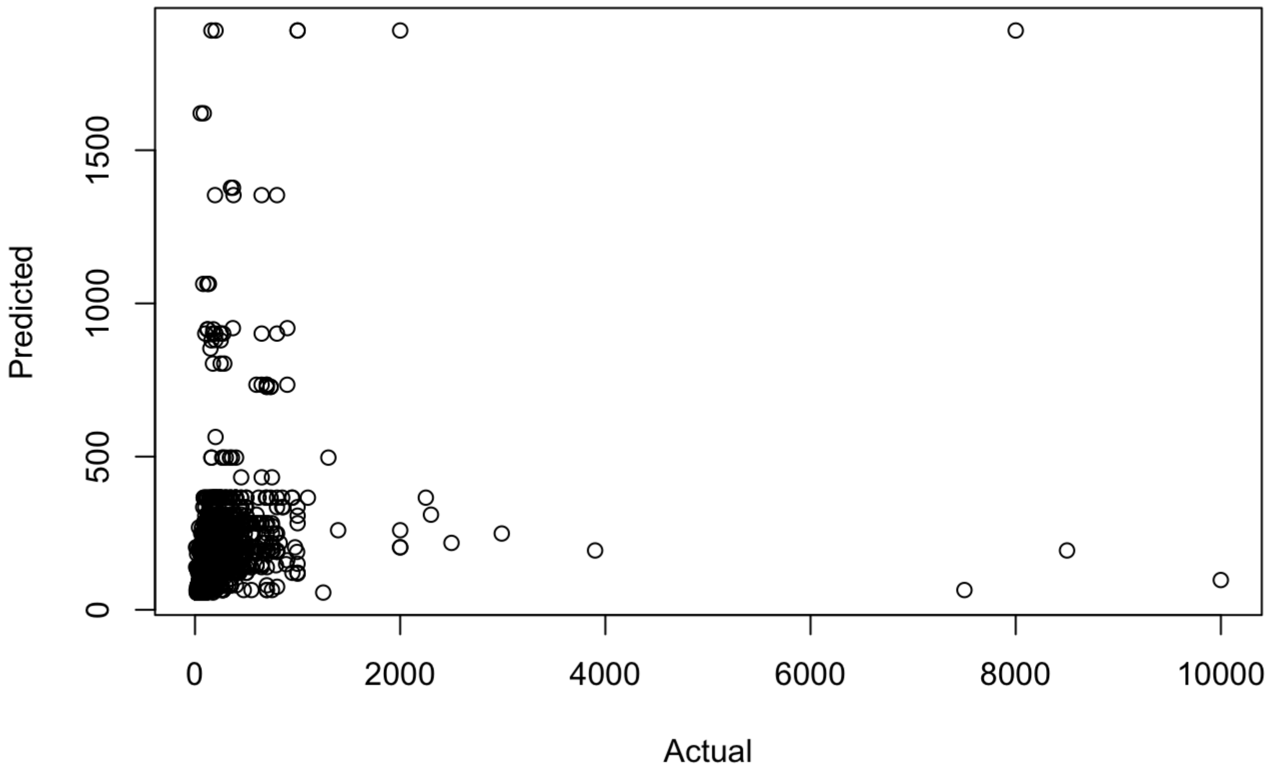
```
## [1] 50274.08
```

Rationale: After fitting the model to the decision tree we calculated the MSE train and test to see how they compared to other models. The MSE train was 25,244 whilst the MSE test was larger, as expected, amounting to: 50,274.

These will both be interpreted in the context of the other MSEs found for other models.

Showing Results:

```
plot(y_test, yhat.test.tree, xlab='Actual', ylab='Predicted')
```



2.5 Random Forest

```
set.seed(1220)
fit.rf <- randomForest(f1, train, do.trace=T, mtry=7, ntree=500)
```

```
##      |      Out-of-bag   |
## Tree |      MSE  %Var(y) |
##    1 | 3.626e+04   104.73 |
##    2 | 4.054e+04   117.09 |
##    3 | 3.56e+04   102.81 |
##    4 | 3.945e+04   113.93 |
##    5 | 4.14e+04   119.58 |
##    6 | 3.826e+04   110.51 |
##    7 | 4.319e+04   124.75 |
##    8 | 4.187e+04   120.94 |
##    9 | 3.941e+04   113.82 |
##   10 | 3.593e+04   103.77 |
##   11 | 3.467e+04   100.14 |
##   12 | 3.681e+04   106.31 |
##   13 | 3.466e+04   100.09 |
##   14 | 3.389e+04    97.87 |
##   15 | 3.351e+04    96.78 |
##   16 | 3.28e+04    94.74 |
##   17 | 3.243e+04    93.65 |
##   18 | 3.229e+04    93.25 |
##   19 | 3.231e+04    93.31 |
##   20 | 3.21e+04    92.70 |
##   21 | 3.199e+04    92.41 |
##   22 |  3.2e+04    92.42 |
##   23 | 3.177e+04    91.77 |
##   24 | 3.162e+04    91.31 |
##   25 | 3.155e+04    91.12 |
##   26 | 3.147e+04    90.91 |
##   27 | 3.129e+04    90.38 |
##   28 | 3.106e+04    89.71 |
##   29 | 3.095e+04    89.39 |
##   30 | 3.072e+04    88.73 |
##   31 | 3.067e+04    88.58 |
##   32 | 3.055e+04    88.23 |
##   33 | 3.045e+04    87.94 |
##   34 | 3.036e+04    87.70 |
##   35 | 3.023e+04    87.30 |
##   36 | 3.019e+04    87.20 |
##   37 | 3.012e+04    86.99 |
##   38 | 2.997e+04    86.55 |
##   39 | 2.997e+04    86.56 |
##   40 | 2.993e+04    86.45 |
##   41 | 2.984e+04    86.18 |
##   42 | 2.987e+04    86.28 |
##   43 | 2.979e+04    86.04 |
##   44 | 2.981e+04    86.11 |
##   45 | 2.979e+04    86.05 |
##   46 | 2.981e+04    86.10 |
##   47 | 2.976e+04    85.95 |
##   48 | 2.973e+04    85.87 |
##   49 | 2.967e+04    85.68 |
##   50 | 2.959e+04    85.45 |
##   51 | 2.95e+04    85.21 |
##   52 | 2.946e+04    85.08 |
##   53 | 2.95e+04    85.21 |
##   54 | 2.951e+04    85.23 |
##   55 | 2.951e+04    85.23 |
```

```
##    56 | 2.945e+04    85.05 |
##    57 | 2.947e+04    85.11 |
##    58 | 2.943e+04    85.01 |
##    59 | 2.937e+04    84.82 |
##    60 | 2.936e+04    84.79 |
##    61 | 2.932e+04    84.67 |
##    62 | 2.928e+04    84.56 |
##    63 | 2.928e+04    84.56 |
##    64 | 2.927e+04    84.54 |
##    65 | 2.924e+04    84.45 |
##    66 | 2.924e+04    84.44 |
##    67 | 2.918e+04    84.28 |
##    68 | 2.916e+04    84.22 |
##    69 | 2.915e+04    84.20 |
##    70 | 2.921e+04    84.37 |
##    71 | 2.917e+04    84.26 |
##    72 | 2.916e+04    84.22 |
##    73 | 2.911e+04    84.06 |
##    74 | 2.906e+04    83.94 |
##    75 | 2.903e+04    83.84 |
##    76 | 2.902e+04    83.81 |
##    77 | 2.902e+04    83.81 |
##    78 | 2.895e+04    83.61 |
##    79 | 2.894e+04    83.59 |
##    80 | 2.898e+04    83.69 |
##    81 | 2.898e+04    83.70 |
##    82 |   2.9e+04    83.76 |
##    83 | 2.897e+04    83.66 |
##    84 | 2.895e+04    83.62 |
##    85 | 2.894e+04    83.58 |
##    86 | 2.891e+04    83.49 |
##    87 | 2.89e+04    83.47 |
##    88 | 2.889e+04    83.45 |
##    89 | 2.887e+04    83.39 |
##    90 | 2.887e+04    83.39 |
##    91 | 2.883e+04    83.28 |
##    92 | 2.881e+04    83.20 |
##    93 | 2.877e+04    83.10 |
##    94 | 2.877e+04    83.08 |
##    95 | 2.875e+04    83.04 |
##    96 | 2.875e+04    83.02 |
##    97 | 2.877e+04    83.08 |
##    98 | 2.874e+04    83.01 |
##    99 | 2.873e+04    82.99 |
##   100 | 2.872e+04    82.95 |
##   101 | 2.873e+04    82.97 |
##   102 | 2.871e+04    82.93 |
##   103 | 2.869e+04    82.86 |
##   104 | 2.869e+04    82.87 |
##   105 | 2.869e+04    82.87 |
##   106 | 2.869e+04    82.86 |
##   107 | 2.867e+04    82.81 |
##   108 | 2.866e+04    82.78 |
##   109 | 2.865e+04    82.75 |
##   110 | 2.864e+04    82.71 |
##   111 | 2.863e+04    82.69 |
##   112 | 2.862e+04    82.66 |
##   113 | 2.86e+04    82.60 |
```

```
##   114 | 2.861e+04    82.64 |
##   115 | 2.862e+04    82.66 |
##   116 | 2.862e+04    82.66 |
##   117 | 2.86e+04    82.59 |
##   118 | 2.862e+04    82.66 |
##   119 | 2.861e+04    82.62 |
##   120 | 2.867e+04    82.80 |
##   121 | 2.865e+04    82.76 |
##   122 | 2.865e+04    82.74 |
##   123 | 2.864e+04    82.72 |
##   124 | 2.862e+04    82.65 |
##   125 | 2.861e+04    82.64 |
##   126 | 2.864e+04    82.71 |
##   127 | 2.863e+04    82.69 |
##   128 | 2.862e+04    82.67 |
##   129 | 2.861e+04    82.62 |
##   130 | 2.863e+04    82.68 |
##   131 | 2.862e+04    82.67 |
##   132 | 2.863e+04    82.70 |
##   133 | 2.862e+04    82.66 |
##   134 | 2.861e+04    82.64 |
##   135 | 2.861e+04    82.64 |
##   136 | 2.86e+04    82.61 |
##   137 | 2.86e+04    82.59 |
##   138 | 2.86e+04    82.59 |
##   139 | 2.859e+04    82.57 |
##   140 | 2.858e+04    82.55 |
##   141 | 2.859e+04    82.56 |
##   142 | 2.86e+04    82.61 |
##   143 | 2.859e+04    82.59 |
##   144 | 2.859e+04    82.57 |
##   145 | 2.859e+04    82.57 |
##   146 | 2.859e+04    82.58 |
##   147 | 2.858e+04    82.55 |
##   148 | 2.857e+04    82.51 |
##   149 | 2.856e+04    82.48 |
##   150 | 2.856e+04    82.48 |
##   151 | 2.856e+04    82.50 |
##   152 | 2.855e+04    82.47 |
##   153 | 2.856e+04    82.48 |
##   154 | 2.854e+04    82.43 |
##   155 | 2.854e+04    82.42 |
##   156 | 2.855e+04    82.44 |
##   157 | 2.855e+04    82.46 |
##   158 | 2.856e+04    82.50 |
##   159 | 2.854e+04    82.43 |
##   160 | 2.854e+04    82.43 |
##   161 | 2.855e+04    82.46 |
##   162 | 2.855e+04    82.46 |
##   163 | 2.853e+04    82.41 |
##   164 | 2.852e+04    82.38 |
##   165 | 2.851e+04    82.35 |
##   166 | 2.852e+04    82.37 |
##   167 | 2.852e+04    82.37 |
##   168 | 2.852e+04    82.38 |
##   169 | 2.853e+04    82.39 |
##   170 | 2.852e+04    82.38 |
##   171 | 2.851e+04    82.34 |
```

```
##   172 | 2.85e+04    82.32 |
##   173 | 2.85e+04    82.31 |
##   174 | 2.851e+04    82.33 |
##   175 | 2.85e+04    82.32 |
##   176 | 2.85e+04    82.32 |
##   177 | 2.849e+04    82.29 |
##   178 | 2.848e+04    82.27 |
##   179 | 2.849e+04    82.28 |
##   180 | 2.848e+04    82.25 |
##   181 | 2.846e+04    82.20 |
##   182 | 2.846e+04    82.19 |
##   183 | 2.845e+04    82.18 |
##   184 | 2.844e+04    82.13 |
##   185 | 2.842e+04    82.09 |
##   186 | 2.841e+04    82.07 |
##   187 | 2.841e+04    82.04 |
##   188 | 2.839e+04    82.00 |
##   189 | 2.839e+04    82.00 |
##   190 | 2.838e+04    81.98 |
##   191 | 2.839e+04    81.99 |
##   192 | 2.84e+04    82.02 |
##   193 | 2.839e+04    82.00 |
##   194 | 2.84e+04    82.02 |
##   195 | 2.839e+04    82.01 |
##   196 | 2.839e+04    82.01 |
##   197 | 2.839e+04    81.99 |
##   198 | 2.839e+04    81.99 |
##   199 | 2.838e+04    81.97 |
##   200 | 2.839e+04    81.99 |
##   201 | 2.838e+04    81.96 |
##   202 | 2.838e+04    81.96 |
##   203 | 2.838e+04    81.96 |
##   204 | 2.836e+04    81.92 |
##   205 | 2.836e+04    81.91 |
##   206 | 2.836e+04    81.91 |
##   207 | 2.836e+04    81.91 |
##   208 | 2.835e+04    81.88 |
##   209 | 2.836e+04    81.90 |
##   210 | 2.835e+04    81.89 |
##   211 | 2.835e+04    81.89 |
##   212 | 2.834e+04    81.86 |
##   213 | 2.833e+04    81.84 |
##   214 | 2.835e+04    81.87 |
##   215 | 2.835e+04    81.89 |
##   216 | 2.835e+04    81.88 |
##   217 | 2.835e+04    81.89 |
##   218 | 2.835e+04    81.89 |
##   219 | 2.835e+04    81.87 |
##   220 | 2.835e+04    81.88 |
##   221 | 2.835e+04    81.87 |
##   222 | 2.836e+04    81.91 |
##   223 | 2.837e+04    81.94 |
##   224 | 2.837e+04    81.93 |
##   225 | 2.837e+04    81.93 |
##   226 | 2.838e+04    81.96 |
##   227 | 2.838e+04    81.96 |
##   228 | 2.837e+04    81.93 |
##   229 | 2.836e+04    81.92 |
```

```
##   230 | 2.836e+04    81.92 |
##   231 | 2.836e+04    81.91 |
##   232 | 2.836e+04    81.91 |
##   233 | 2.836e+04    81.91 |
##   234 | 2.836e+04    81.90 |
##   235 | 2.836e+04    81.92 |
##   236 | 2.838e+04    81.96 |
##   237 | 2.836e+04    81.92 |
##   238 | 2.836e+04    81.90 |
##   239 | 2.835e+04    81.87 |
##   240 | 2.834e+04    81.86 |
##   241 | 2.834e+04    81.85 |
##   242 | 2.834e+04    81.85 |
##   243 | 2.834e+04    81.84 |
##   244 | 2.834e+04    81.86 |
##   245 | 2.834e+04    81.85 |
##   246 | 2.835e+04    81.89 |
##   247 | 2.835e+04    81.87 |
##   248 | 2.834e+04    81.86 |
##   249 | 2.834e+04    81.86 |
##   250 | 2.835e+04    81.90 |
##   251 | 2.836e+04    81.90 |
##   252 | 2.836e+04    81.90 |
##   253 | 2.835e+04    81.89 |
##   254 | 2.834e+04    81.86 |
##   255 | 2.833e+04    81.84 |
##   256 | 2.832e+04    81.81 |
##   257 | 2.833e+04    81.82 |
##   258 | 2.833e+04    81.83 |
##   259 | 2.834e+04    81.85 |
##   260 | 2.833e+04    81.83 |
##   261 | 2.833e+04    81.83 |
##   262 | 2.833e+04    81.82 |
##   263 | 2.833e+04    81.82 |
##   264 | 2.834e+04    81.86 |
##   265 | 2.834e+04    81.85 |
##   266 | 2.834e+04    81.84 |
##   267 | 2.833e+04    81.82 |
##   268 | 2.832e+04    81.79 |
##   269 | 2.832e+04    81.80 |
##   270 | 2.833e+04    81.82 |
##   271 | 2.833e+04    81.83 |
##   272 | 2.834e+04    81.84 |
##   273 | 2.834e+04    81.84 |
##   274 | 2.834e+04    81.84 |
##   275 | 2.833e+04    81.82 |
##   276 | 2.832e+04    81.80 |
##   277 | 2.832e+04    81.79 |
##   278 | 2.831e+04    81.77 |
##   279 | 2.831e+04    81.76 |
##   280 | 2.832e+04    81.79 |
##   281 | 2.832e+04    81.78 |
##   282 | 2.832e+04    81.79 |
##   283 | 2.831e+04    81.77 |
##   284 | 2.831e+04    81.76 |
##   285 | 2.83e+04    81.75 |
##   286 | 2.83e+04    81.75 |
##   287 | 2.83e+04    81.75 |
```

```
##   288 | 2.83e+04    81.75 |
##   289 | 2.83e+04    81.73 |
##   290 | 2.83e+04    81.73 |
##   291 | 2.829e+04    81.72 |
##   292 | 2.829e+04    81.72 |
##   293 | 2.829e+04    81.71 |
##   294 | 2.829e+04    81.70 |
##   295 | 2.828e+04    81.68 |
##   296 | 2.828e+04    81.69 |
##   297 | 2.828e+04    81.68 |
##   298 | 2.828e+04    81.68 |
##   299 | 2.828e+04    81.69 |
##   300 | 2.828e+04    81.68 |
##   301 | 2.827e+04    81.66 |
##   302 | 2.827e+04    81.65 |
##   303 | 2.827e+04    81.66 |
##   304 | 2.827e+04    81.65 |
##   305 | 2.827e+04    81.64 |
##   306 | 2.827e+04    81.65 |
##   307 | 2.828e+04    81.67 |
##   308 | 2.827e+04    81.64 |
##   309 | 2.828e+04    81.66 |
##   310 | 2.828e+04    81.69 |
##   311 | 2.83e+04    81.73 |
##   312 | 2.829e+04    81.72 |
##   313 | 2.83e+04    81.73 |
##   314 | 2.83e+04    81.74 |
##   315 | 2.83e+04    81.74 |
##   316 | 2.83e+04    81.74 |
##   317 | 2.83e+04    81.73 |
##   318 | 2.83e+04    81.72 |
##   319 | 2.83e+04    81.74 |
##   320 | 2.83e+04    81.74 |
##   321 | 2.83e+04    81.73 |
##   322 | 2.83e+04    81.72 |
##   323 | 2.83e+04    81.72 |
##   324 | 2.83e+04    81.74 |
##   325 | 2.83e+04    81.74 |
##   326 | 2.83e+04    81.73 |
##   327 | 2.83e+04    81.72 |
##   328 | 2.83e+04    81.73 |
##   329 | 2.83e+04    81.73 |
##   330 | 2.83e+04    81.73 |
##   331 | 2.83e+04    81.72 |
##   332 | 2.829e+04    81.72 |
##   333 | 2.831e+04    81.77 |
##   334 | 2.831e+04    81.75 |
##   335 | 2.831e+04    81.77 |
##   336 | 2.831e+04    81.78 |
##   337 | 2.831e+04    81.77 |
##   338 | 2.832e+04    81.78 |
##   339 | 2.832e+04    81.81 |
##   340 | 2.833e+04    81.82 |
##   341 | 2.833e+04    81.83 |
##   342 | 2.833e+04    81.84 |
##   343 | 2.834e+04    81.85 |
##   344 | 2.834e+04    81.86 |
##   345 | 2.834e+04    81.85 |
```

```
##   346 | 2.834e+04    81.86 |
##   347 | 2.834e+04    81.85 |
##   348 | 2.834e+04    81.84 |
##   349 | 2.833e+04    81.83 |
##   350 | 2.833e+04    81.83 |
##   351 | 2.834e+04    81.84 |
##   352 | 2.834e+04    81.84 |
##   353 | 2.834e+04    81.84 |
##   354 | 2.833e+04    81.83 |
##   355 | 2.833e+04    81.83 |
##   356 | 2.834e+04    81.84 |
##   357 | 2.834e+04    81.84 |
##   358 | 2.834e+04    81.85 |
##   359 | 2.834e+04    81.84 |
##   360 | 2.833e+04    81.83 |
##   361 | 2.833e+04    81.84 |
##   362 | 2.833e+04    81.84 |
##   363 | 2.833e+04    81.83 |
##   364 | 2.833e+04    81.82 |
##   365 | 2.833e+04    81.83 |
##   366 | 2.833e+04    81.82 |
##   367 | 2.833e+04    81.81 |
##   368 | 2.832e+04    81.80 |
##   369 | 2.832e+04    81.80 |
##   370 | 2.832e+04    81.81 |
##   371 | 2.832e+04    81.79 |
##   372 | 2.833e+04    81.81 |
##   373 | 2.832e+04    81.80 |
##   374 | 2.833e+04    81.82 |
##   375 | 2.833e+04    81.81 |
##   376 | 2.832e+04    81.80 |
##   377 | 2.832e+04    81.81 |
##   378 | 2.832e+04    81.79 |
##   379 | 2.831e+04    81.77 |
##   380 | 2.831e+04    81.76 |
##   381 | 2.83e+04    81.74 |
##   382 | 2.829e+04    81.72 |
##   383 | 2.83e+04    81.72 |
##   384 | 2.829e+04    81.72 |
##   385 | 2.829e+04    81.69 |
##   386 | 2.828e+04    81.69 |
##   387 | 2.829e+04    81.72 |
##   388 | 2.83e+04    81.73 |
##   389 | 2.83e+04    81.73 |
##   390 | 2.831e+04    81.75 |
##   391 | 2.831e+04    81.77 |
##   392 | 2.831e+04    81.77 |
##   393 | 2.831e+04    81.76 |
##   394 | 2.831e+04    81.75 |
##   395 | 2.83e+04    81.75 |
##   396 | 2.83e+04    81.74 |
##   397 | 2.83e+04    81.73 |
##   398 | 2.83e+04    81.74 |
##   399 | 2.829e+04    81.72 |
##   400 | 2.829e+04    81.70 |
##   401 | 2.829e+04    81.71 |
##   402 | 2.829e+04    81.70 |
##   403 | 2.829e+04    81.71 |
```

```
## 404 | 2.829e+04   81.71 |
## 405 | 2.829e+04   81.72 |
## 406 | 2.829e+04   81.69 |
## 407 | 2.829e+04   81.71 |
## 408 | 2.829e+04   81.72 |
## 409 | 2.829e+04   81.71 |
## 410 | 2.829e+04   81.70 |
## 411 | 2.829e+04   81.72 |
## 412 | 2.829e+04   81.72 |
## 413 | 2.829e+04   81.71 |
## 414 | 2.83e+04   81.73 |
## 415 | 2.83e+04   81.73 |
## 416 | 2.83e+04   81.74 |
## 417 | 2.83e+04   81.72 |
## 418 | 2.83e+04   81.73 |
## 419 | 2.83e+04   81.73 |
## 420 | 2.829e+04   81.72 |
## 421 | 2.83e+04   81.74 |
## 422 | 2.83e+04   81.74 |
## 423 | 2.83e+04   81.73 |
## 424 | 2.83e+04   81.73 |
## 425 | 2.83e+04   81.74 |
## 426 | 2.831e+04   81.75 |
## 427 | 2.83e+04   81.75 |
## 428 | 2.83e+04   81.73 |
## 429 | 2.83e+04   81.72 |
## 430 | 2.829e+04   81.71 |
## 431 | 2.829e+04   81.71 |
## 432 | 2.829e+04   81.71 |
## 433 | 2.83e+04   81.73 |
## 434 | 2.83e+04   81.73 |
## 435 | 2.832e+04   81.78 |
## 436 | 2.831e+04   81.78 |
## 437 | 2.831e+04   81.77 |
## 438 | 2.831e+04   81.77 |
## 439 | 2.831e+04   81.77 |
## 440 | 2.83e+04   81.75 |
## 441 | 2.83e+04   81.73 |
## 442 | 2.829e+04   81.72 |
## 443 | 2.829e+04   81.72 |
## 444 | 2.829e+04   81.71 |
## 445 | 2.829e+04   81.71 |
## 446 | 2.829e+04   81.72 |
## 447 | 2.829e+04   81.72 |
## 448 | 2.829e+04   81.70 |
## 449 | 2.829e+04   81.69 |
## 450 | 2.828e+04   81.68 |
## 451 | 2.828e+04   81.68 |
## 452 | 2.828e+04   81.68 |
## 453 | 2.828e+04   81.67 |
## 454 | 2.829e+04   81.70 |
## 455 | 2.828e+04   81.68 |
## 456 | 2.828e+04   81.69 |
## 457 | 2.828e+04   81.69 |
## 458 | 2.829e+04   81.69 |
## 459 | 2.828e+04   81.69 |
## 460 | 2.829e+04   81.72 |
## 461 | 2.83e+04   81.73 |
```

```
##  462 | 2.83e+04    81.72 |
##  463 | 2.829e+04    81.72 |
##  464 | 2.829e+04    81.71 |
##  465 | 2.829e+04    81.70 |
##  466 | 2.829e+04    81.70 |
##  467 | 2.828e+04    81.68 |
##  468 | 2.828e+04    81.67 |
##  469 | 2.828e+04    81.67 |
##  470 | 2.827e+04    81.66 |
##  471 | 2.827e+04    81.66 |
##  472 | 2.828e+04    81.67 |
##  473 | 2.828e+04    81.67 |
##  474 | 2.828e+04    81.67 |
##  475 | 2.828e+04    81.67 |
##  476 | 2.827e+04    81.66 |
##  477 | 2.827e+04    81.64 |
##  478 | 2.827e+04    81.64 |
##  479 | 2.826e+04    81.63 |
##  480 | 2.826e+04    81.62 |
##  481 | 2.826e+04    81.62 |
##  482 | 2.825e+04    81.61 |
##  483 | 2.825e+04    81.60 |
##  484 | 2.825e+04    81.60 |
##  485 | 2.825e+04    81.59 |
##  486 | 2.825e+04    81.59 |
##  487 | 2.825e+04    81.60 |
##  488 | 2.826e+04    81.62 |
##  489 | 2.826e+04    81.62 |
##  490 | 2.826e+04    81.61 |
##  491 | 2.825e+04    81.60 |
##  492 | 2.825e+04    81.60 |
##  493 | 2.825e+04    81.60 |
##  494 | 2.825e+04    81.59 |
##  495 | 2.825e+04    81.58 |
##  496 | 2.824e+04    81.56 |
##  497 | 2.824e+04    81.56 |
##  498 | 2.824e+04    81.58 |
##  499 | 2.825e+04    81.59 |
##  500 | 2.825e+04    81.58 |
```

```
#Computing our train MSEs:
yhat.train.rf <- predict(fit.rf, train, ntree=500)
mse.train.rf  <- mean((yhat.train.rf - y_train)^2)

#Computing our test MSEs:
yhat.test.rf <- predict(fit.rf, test, ntree=500)
mse.test.rf <- mean((yhat.test.rf - y_test)^2)

mse.train.rf
```

```
## [1] 6745.319
```

```
mse.test.rf
```

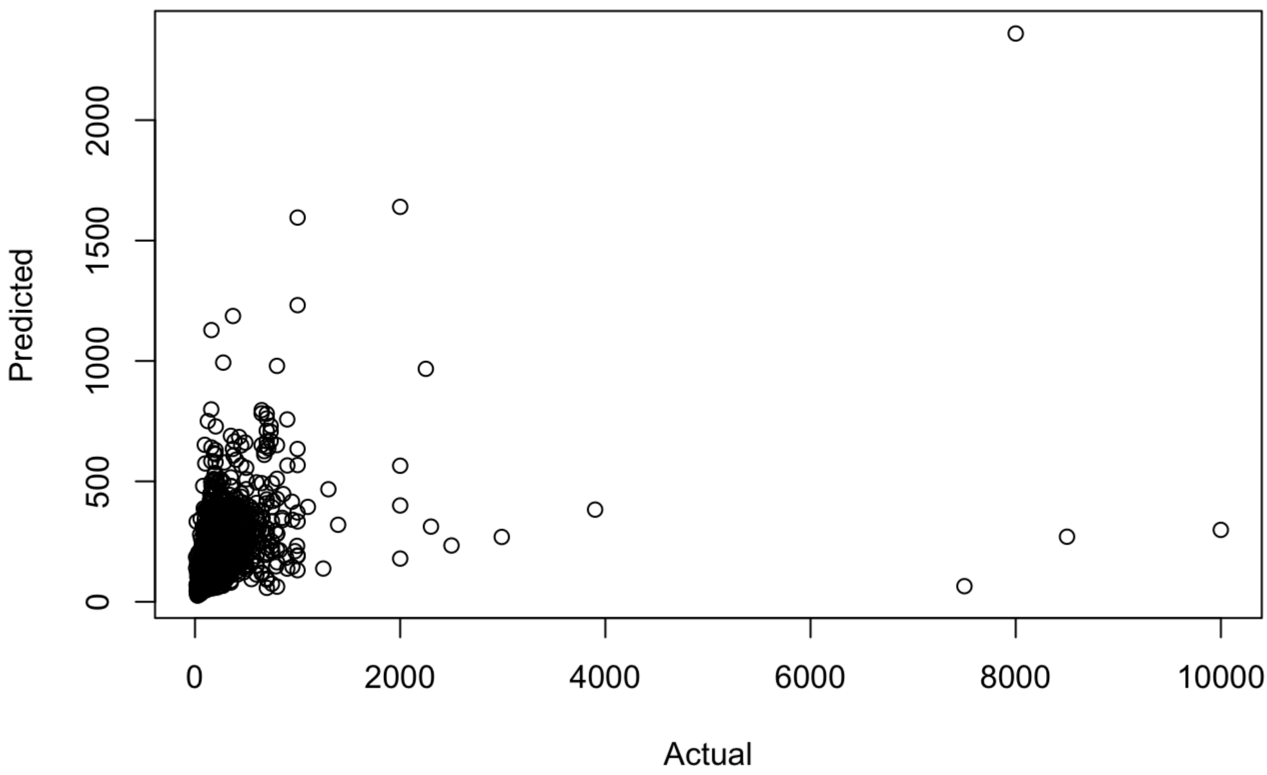
```
## [1] 44695.97
```

Rationale: To fit the Random Forest model, we initially set the number of trees as 500 and hyper parameter m as 7. After fitting the model to the Random Forest we calculated the MSE train and MSE test.

The MSE train is 6,745 and the MSE test is 44,696, which seems our random forest model over fits to the train data set.

Showing Results:

```
plot(y_test, yhat.test.rf, xlab='Actual', ylab='Predicted')
```



2.6 Boosting

```
set.seed(1220)
fit.btree <- gbm(f1,
                 data = train,
                 distribution = "gaussian",
                 n.trees = 500,
                 interaction.depth = 2,
                 shrinkage = 0.001)

relative.influence(fit.btree)
```

```
## n.trees not given. Using 500 trees.
```

```
##                         latitude                         longitude 
##                                0                        2214708102 
##                   minimum_nights                 number_of_reviews 
##                       1209408189                                 0 
##                reviews_per_month calculated_host_listings_count 
##                                0                                 0 
##                 availability_365                          brooklyn 
##                         41798815                                 0 
##                        manhattan                            queens 
##                        816192928                                 0 
##                            bronx                      private_room 
##                                0                       14194601906 
##                      shared_room 
##                                0
```

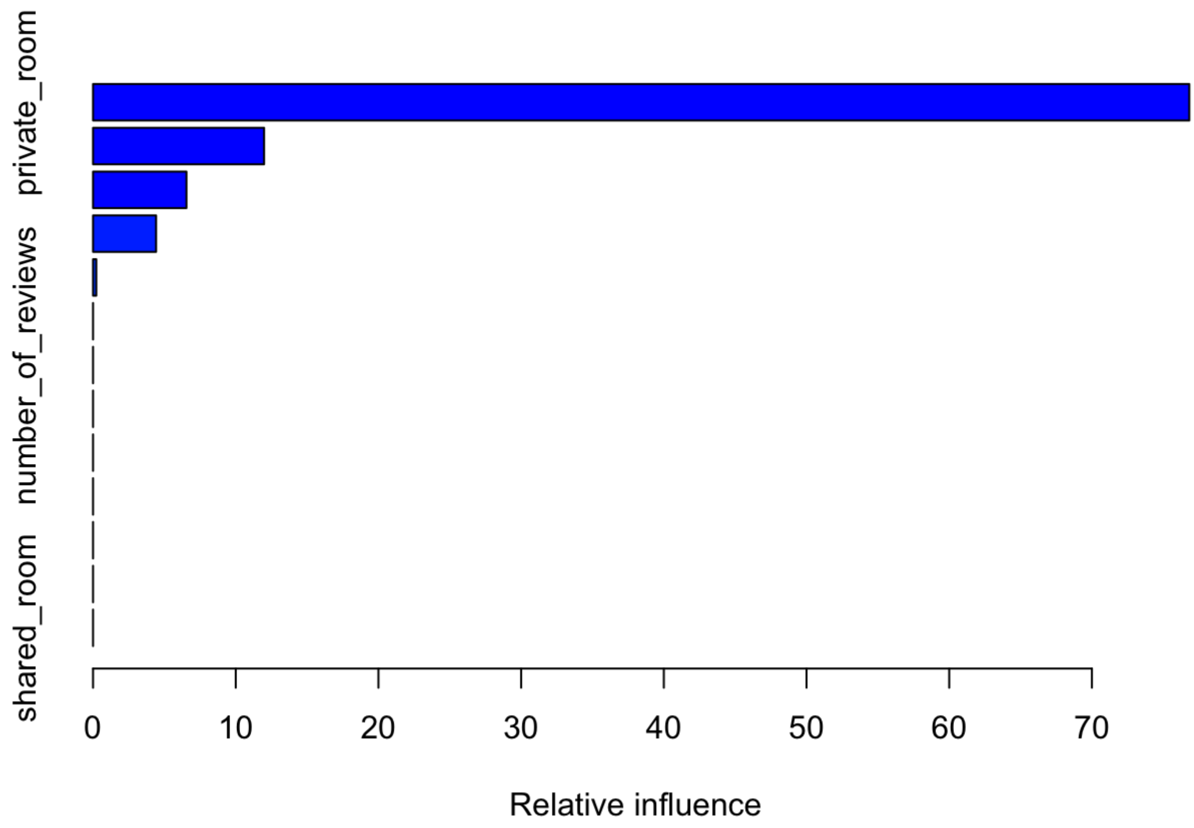
```r
#Computing our train MSEs:
yhat.train.btree <- predict(fit.btree, train, n.trees = 100)
mse.train.btree <- mean((yhat.train.btree - y_train) ^ 2)

#Computing our test MSEs:
yhat.test.btree <- predict(fit.btree, test, n.trees = 100)
mse.test.btree <- mean((yhat.test.btree - y_test) ^ 2)

mse.train.btree
```

```
## [1] 33981.86
```

```r
mse.test.btree
```

```
## [1] 54893.66
```

```r
summary(fit.btree)
```

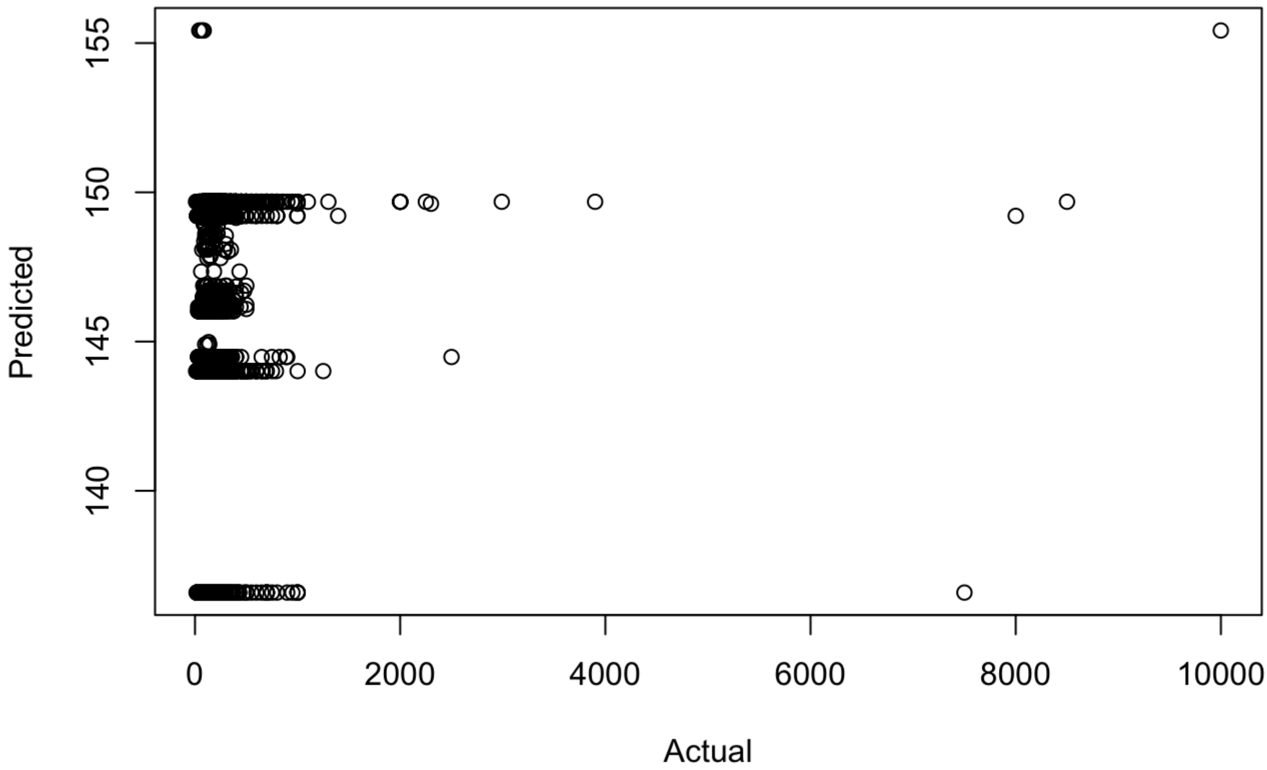



```
##                                                               var    rel.inf
## private_room                                        private_room 76.8242937
## longitude                                              longitude 11.9864852
## minimum_nights                                    minimum_nights  6.5455819
## manhattan                                              manhattan  4.4174148
## availability_365                                availability_365  0.2262243
## latitude                                                latitude  0.0000000
## number_of_reviews                              number_of_reviews  0.0000000
## reviews_per_month                              reviews_per_month  0.0000000
## calculated_host_listings_count calculated_host_listings_count  0.0000000
## brooklyn                                                brooklyn  0.0000000
## queens                                                    queens  0.0000000
## bronx                                                      bronx  0.0000000
## shared_room                                          shared_room  0.0000000
```

Rationale: By fitting the boosting model to the data set, we observed that MSE train is 33,982 and MSE test is 54,894. The MSE test here is slightly higher than the one in the linear model. We'll compare models all together and discuss the results later.

Showing Results:

```
plot(y_test, yhat.test.btree, xlab='Actual', ylab='Predicted')
```



Predicted
155
150
145
140
0
2000
4000
6000
8000
10000
Actual

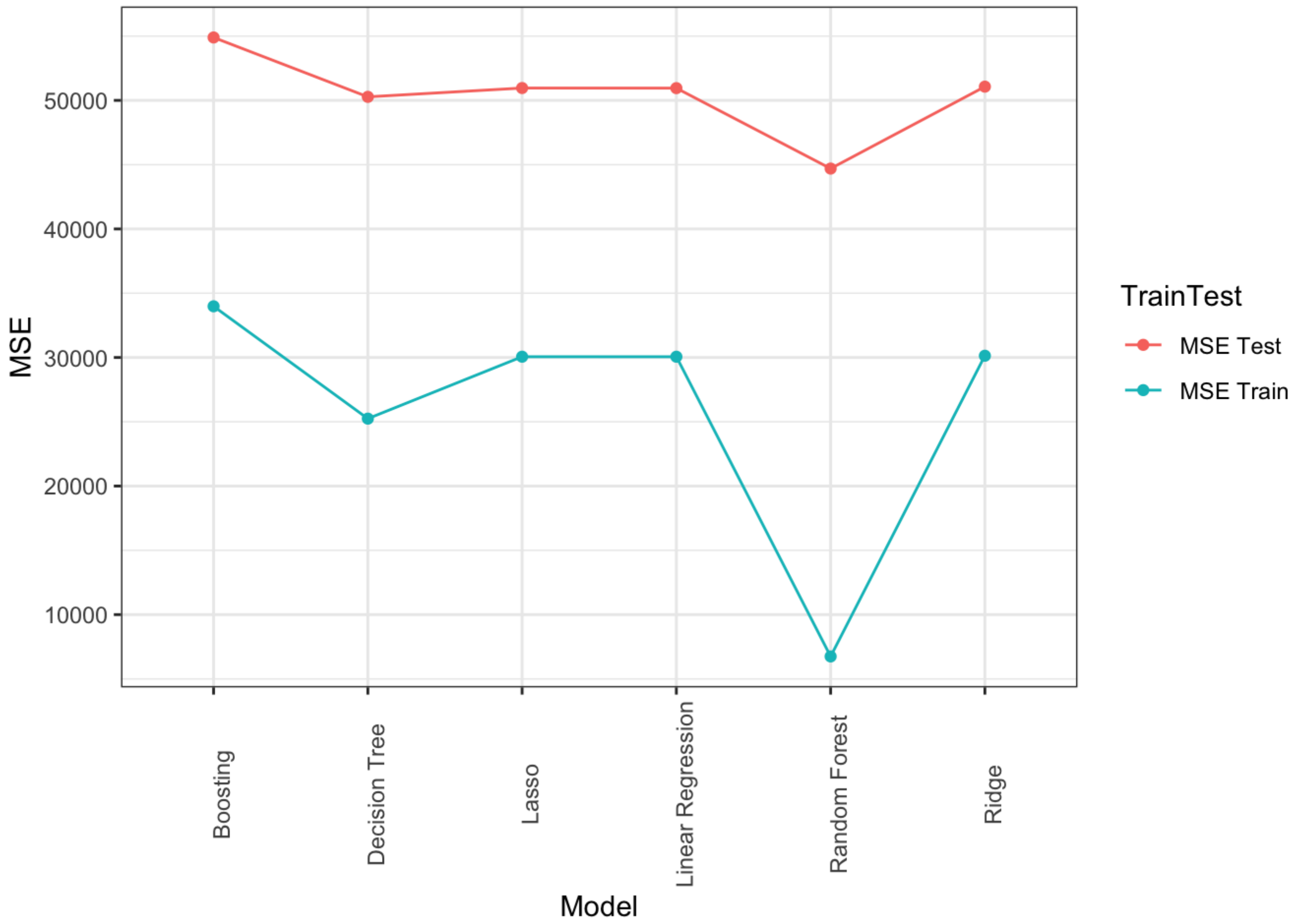
3. ML Model Performance

3.1 MSE Table

```
mse.performance <- data.table(
  TrainTest = c('MSE Train', 'MSE Train', 'MSE Train', 'MSE Train', 'MSE Train', 'MSE
Train', 'MSE Test', 'MSE Test', 'MSE Test', 'MSE Test', 'MSE Test', 'MSE Test'),
  MSE = c(mse.train.btree, mse.train.lasso, mse.train.lm1, mse.train.rf, mse.train.ri
dge, mse.train.tree, mse.test.btree, mse.test.lasso, mse.test.lm1, mse.test.rf, mse.t
est.ridge, mse.test.tree),
  Model = c('Boosting', 'Lasso', 'Linear Regression', 'Random Forest', 'Ridge', 'Deci
sion Tree', 'Boosting', 'Lasso', 'Linear Regression', 'Random Forest', 'Ridge', 'Deci
sion Tree')
)

mse.performance
```

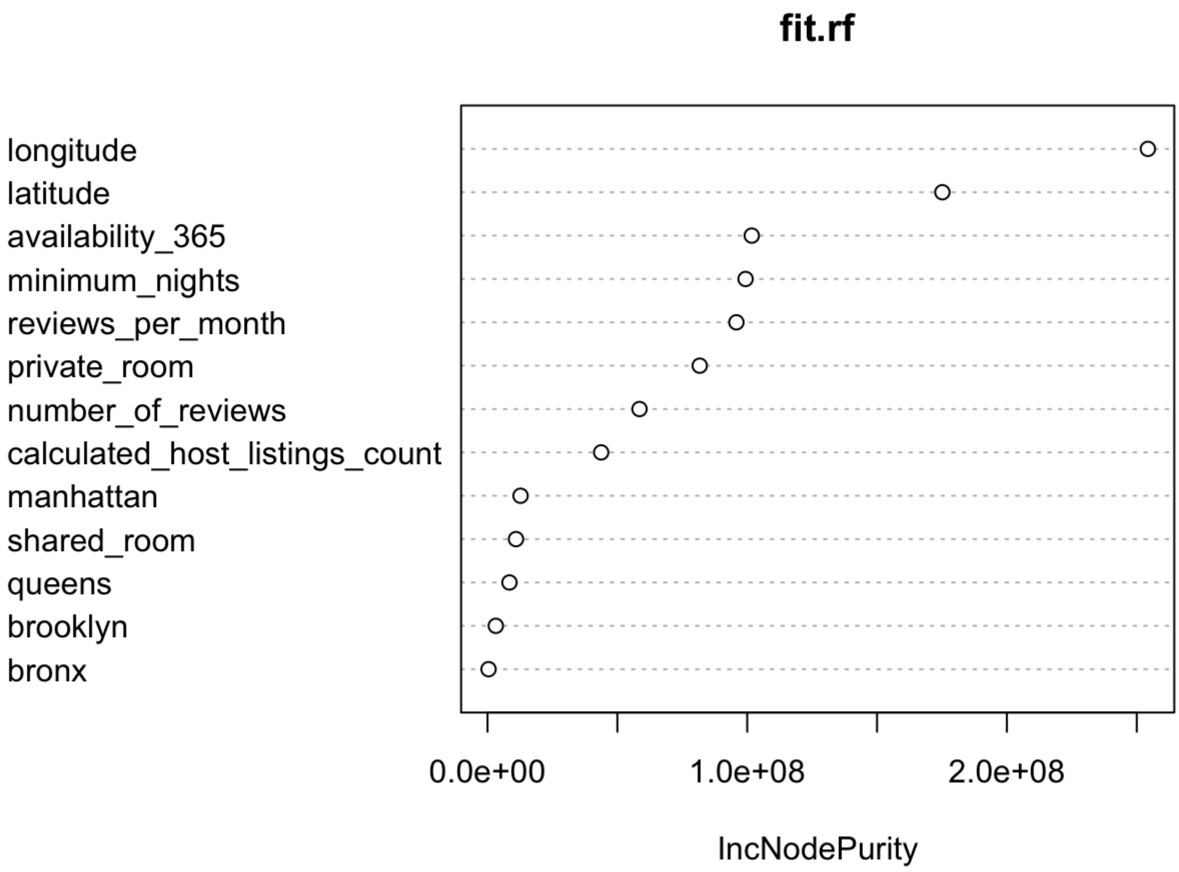
```
##     TrainTest       MSE             Model
##  1: MSE Train 33981.859          Boosting
##  2: MSE Train 30060.624             Lasso
##  3: MSE Train 30059.354 Linear Regression
##  4: MSE Train  6745.319     Random Forest
##  5: MSE Train 30135.350             Ridge
##  6: MSE Train 25243.883     Decision Tree
##  7:  MSE Test 54893.658          Boosting
##  8:  MSE Test 50959.439             Lasso
##  9:  MSE Test 50951.575 Linear Regression
## 10:  MSE Test 44695.965     Random Forest
## 11:  MSE Test 51067.982             Ridge
## 12:  MSE Test 50274.077     Decision Tree
```

```
ggplot(mse.performance, aes(Model, MSE, color=TrainTest, group=TrainTest)) +
  geom_point() +
  geom_line() +
  theme(axis.text.x = element_text(angle = 90))
```

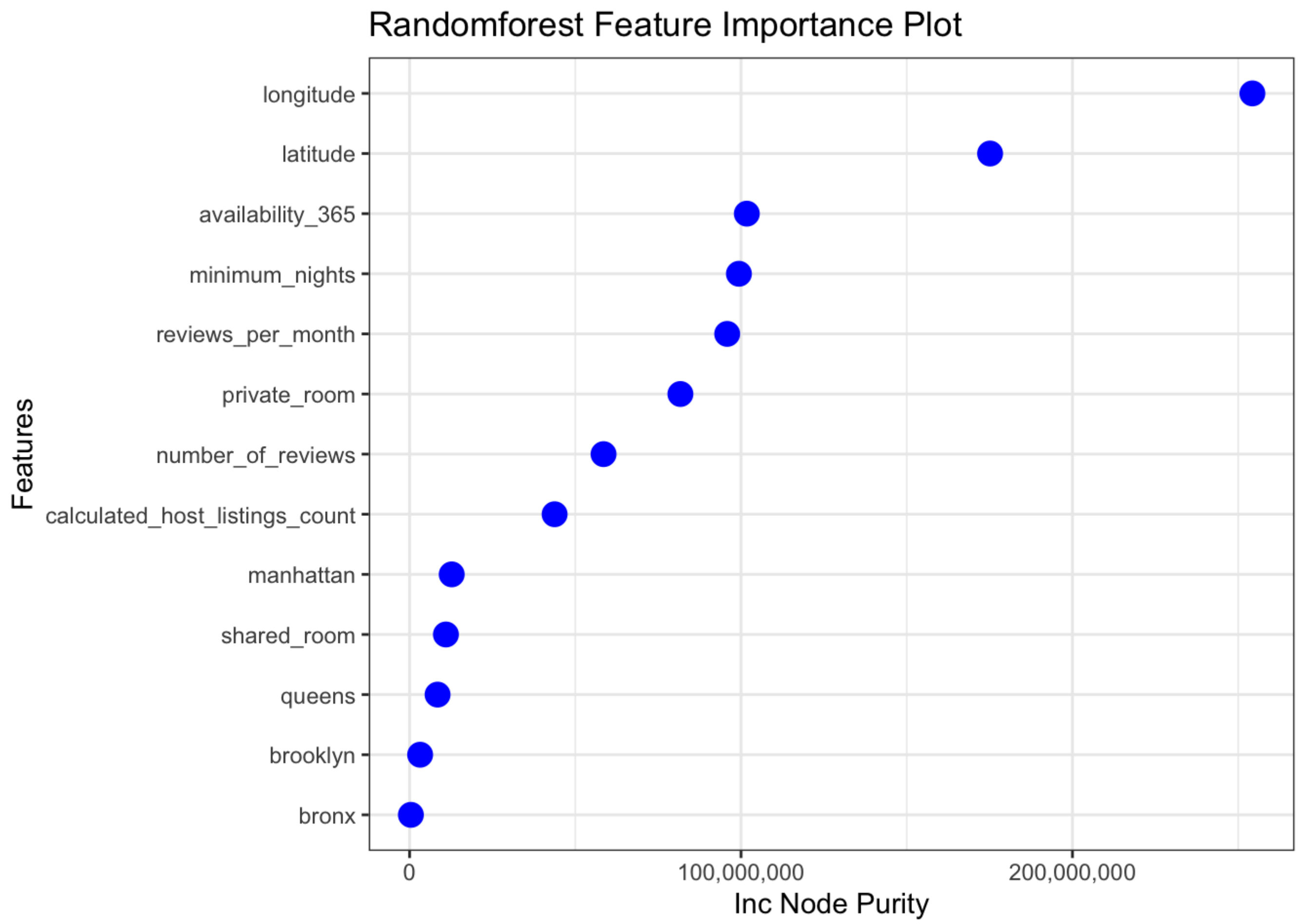


3.2 RandomForest Feature Importance Plot

```
#saving varImp object
imp <- varImpPlot(fit.rf)
```




```
#creating dataframe for plot
imp <- as.data.frame(imp)
imp$varnames <- rownames(imp)
rownames(imp) <- NULL
#visualizing plot
ggplot(imp, aes(x=reorder(varnames, IncNodePurity), y=IncNodePurity)) +
  geom_point(color = "blue", size=4) +
  labs(title="Randomforest Feature Importance Plot", y="Inc Node Purity", x="Feature
s") +
  coord_flip() +
  scale_y_continuous(labels = comma)
```



4. Conclusion

4.1 Main Results of ML Models

Model Application: Among the Models considered here, Random Forest with less MSE test seems to be the best one to predict prices.

Important Features: The location, including latitude and longitude plays the most vital role in the prices. The result corresponds to our expectations that Airbnb prices in various areas vary differently.

In addition to the location, “minimum nights”, “availability 365”, “number of reviews” and “room type” actually show significance in predicting the price as well.

4.2 Next Step

Advantages:

- Help customers find satisfactory Airbnb within a reasonable price range.
- Hosts are expected to set their own prices for their properties. Using Machine Learning techniques can help hosts predict prices for listing or new properties.

Directions for the Future:

- Improve the model with natural language processing (NLP) of customer reviews (for sentiment analysis or look for keyword).
- Include a wider variety of data types: actual average prices paid per night, review ratings, cancellation policy, etc..