**USE CASE STUDY REPORT**

**Group No**.: Group 13

**Student Names**: Yaodong Chen and Yuxiao Ma

**Summary**

When people travel, they pay more and more attention to the prices of their upcoming apartments, especially in densely populated cities like New York. When customers have more choices, they prefer to get the equal experience at a low price. On the other hand, the host also needs to determine a reasonable unit price based on the rental situation of its house to attract more customers to book. The study was conducted on April 18, 2020. The goal of the study is to examine the apartment price floating in New York City and make meaningful predictions and comparisons through different models. In this study, Naïve Bayes, KNN, linear regression models, regression tree, and neural net are implemented. The data is originated in Airbnb official website. Data cleaning, visualization, exploration, and multiple regression models are also implemented in the study. Throughout the study, the price of the apartment listings varies, and the neural network model shows the best prediction result among all models.

# I. Background and Introduction

Founded in August of 2008 in San Francisco, California. Airbnb is a trusted marketplace for people to discover and book unique accommodations around the world. People can book on mobile phone or on their computer and Airbnb provides every choice that people might have to give them the best experience ever. Furthermore, Airbnb connects travelers to diverse experience and different culture in over 34,000 cities and 190 countries (About Airbnb). However, there are voices saying that the way that people using Airbnb is disrupting the hotel industry and affecting the neighborhoods because more and more guests rent the entire home all year round. This dataset contains the listing activity metrics in New York City for 2019 and we want to discover more information about hosts and geographical availability. To simplify the problem, the prediction of the actual price will be conducted. At the same time, we will calculate the average price of all listings and convert it to two binary variables to examine which host set the price above the average or below for another approach. The purpose of the method will allow hosts to adjust their price based on the prediction.

**Data Origin**

The data set is from Kaggle website. The entire data set can be found here: https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data

**Key Attributes**

There are 16 columns total and here is an explanation of each column:

|  |  |
| --- | --- |
| **Variable Name** | **Explanation** |
| id | listing id |
| name | name of the listing |
| host\_id | host id |
| host\_name | name of the host |
| neighborhood\_group | location |
| neighborhood | area |
| latitude | latitude coordinates |
| longitude | longitude coordinates |
| room\_type | listing space type |
| price | price in dollars |
| minimum\_nights | # of nights minimum |
| number\_of\_reviews | # of reviews |
| last\_review | latest review |
| reviews\_per\_month | # of reviews per month |
| calculated\_host\_listings\_count | # of listing per host |
| availability\_365 | # of days when listing is available for booking |

# The Problem

* What can we learn between different hosts and areas?
* Determine the most popular home and predict the correlations between the popularity and the area of each host.
* Predict the correlations between different attributes.
* Predict the correlation between the price and the area of each host based on the location.

# The goal of the study

* Predict the price of an apartment and check whether existing price of apartment listings from provided information is above or below of the average price of current listings in NYC. Hosts can adjust their price based on the prediction model.

# Possible Solution

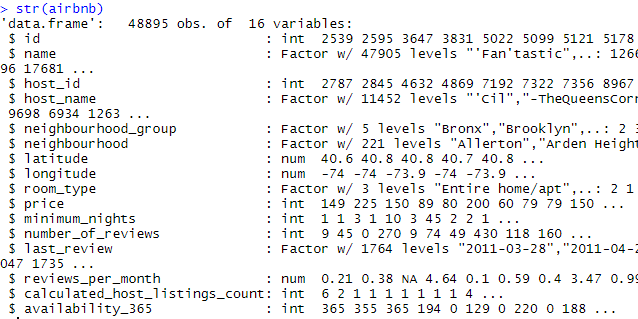
* Implement data cleaning, data exploration, data visualization to process the data.
* Generate dummy values for data attributes for transformation to get better interpretation.
* Split the data set into training set and testing set for prediction purposes. In regression tree, we split the data into training, validation, and test set.
* Implement Naïve Bayes, KNN, linear regression, Regression tree, and neural network model to predict price for training set.

**II. Data Exploration and Visualization**

In this part we will focus on data exploration and visualization to get a general view of the data set. From the data set, we can know the price and location of different hosts, since the data is hard to read, we can use different basic charts to reflect some key points about the data.

**Data set**

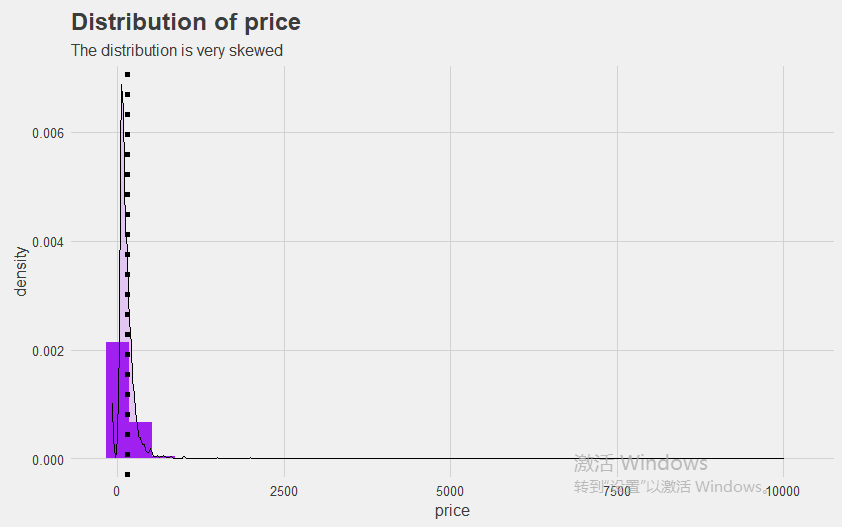
We use str() to take a look at the dataset as a whole:



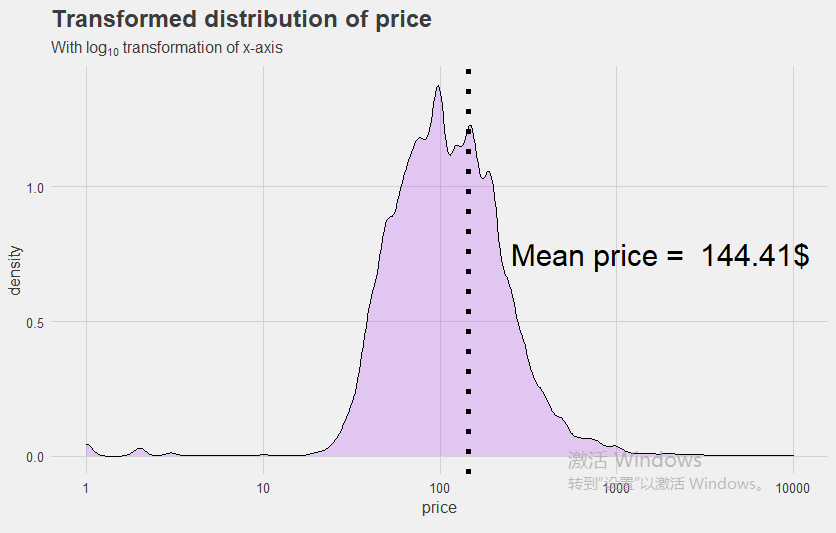
There are 48895 variables in total, and there are 16 categories. We are looking at the raw data right now and we will perform data cleaning to remove unnecessary attribute in next step.

**Density Chart**

We use density chart to reflect the density of price in different regions of New York City. After this, we are able to find the mean of all prices.



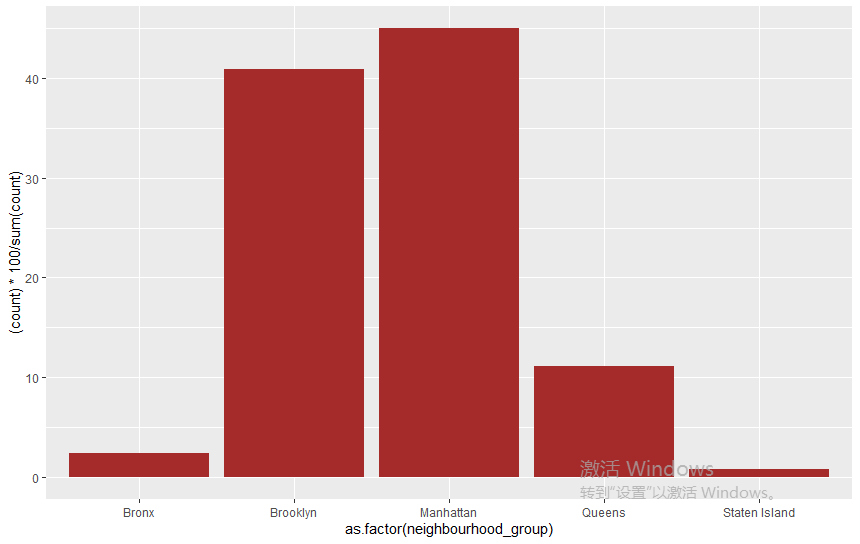
From the chart above, we are unable to find the range of the price. To get a better interpretation result, we use log10 transformation to make the visualization more clearly.



After this step, we can clearly see the distribution of price, and the plot shows that average price is $144.41 for one night. Meanwhile, we can see that the distribution of price satisfies normal distribution approximately. We can say that most of the hosts put a decent price on their listings and they do not overcharge their customers.

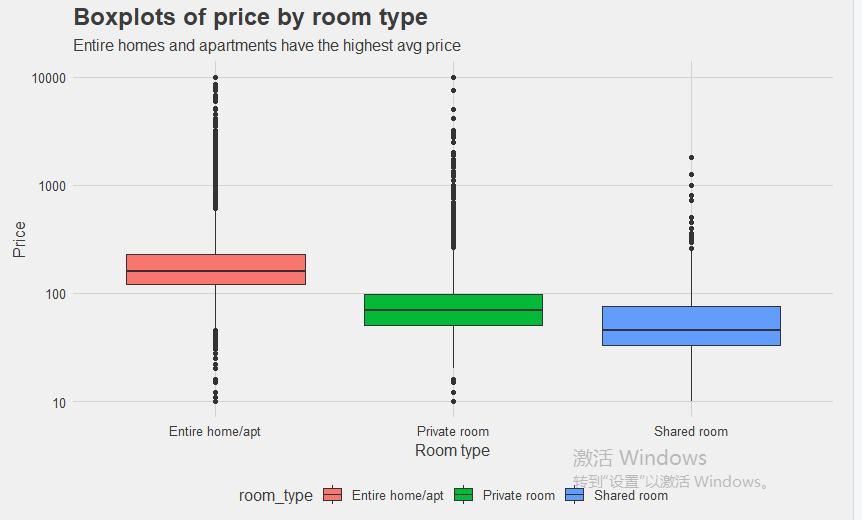
**Histogram for Distribution of Hosts**

We use a histogram to analyze the distribution of hosts in five different areas in NYC.



**Boxplot for Types of Rooms**

We analyze the price of different type of rooms by using Boxplots.

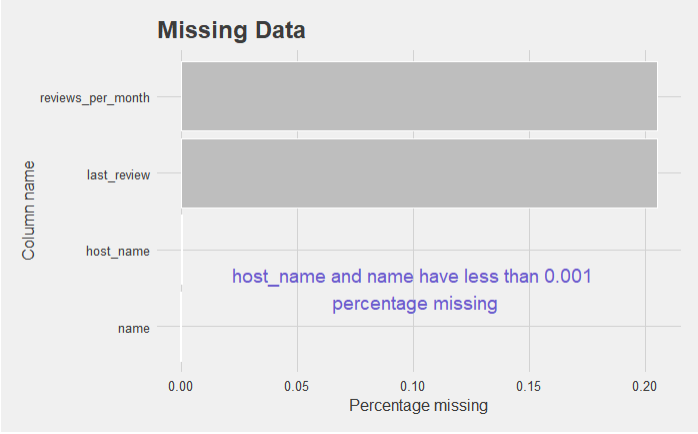


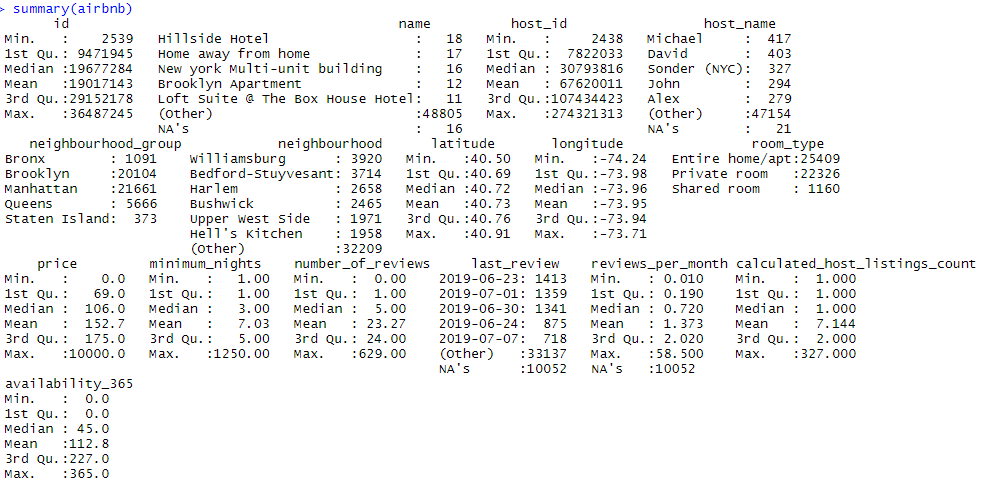
From the plot we can see Manhattan has the highest percentage of listings among the other areas. Brooklyn also possess a number of listings that is almost similar to Manhattan. The other areas have considerably less percentages compared to those two. The reason is obvious because Manhattan and Brooklyn are highly commercialized cities, and Manhattan is known to have the highest population in New York City. To accommodate a larger number of people, including workers or travelers, the demand for the number of houses should be high. Meanwhile, the hosts are also willing to rent house if the house is vacant, which will economically reduce their expense on the rent by themselves. This explains the reason for Manhattan having the highest percentage of listings. Furthermore, the box plot tells us the price of different room type. When we compare the price and different room type, we can tell that entire home/apt has the highest price because people will have more space and more privacy. The price is reasonable at the moment for different type of rooms.

**III. Data Preparation and Preprocessing**

**Data Quality**

The data set is a reputable source from Airbnb website. There are some missing values in review section. We need to handle the missing values before we handle the data set.





The box plot indicates the percentage of missing values inside each attributes. We can tell that *reviews\_per\_month* and *last\_review* have the most missing values. In this case, it is quite normal to see it happens because some people forget to leave a review for the apartment that they stayed. Additionally, the review attribute does not influence the price category prediction that much, so we can just ignore it. The number of missing value of *name* and *host\_name* is pretty small, so we can just remove them. Overall, the data does not have too much missing values except the review section, when analyzing, we will remove them in the columns.

**Variable Selection**

Variable selection is an important part of the process because unnecessary attribute could be confusing and affect the performance of the model. Not all attributes are useful information. If our goal is to focus on the price of the listings, we will choose related. In this case study, we will choose *neighborhood\_group*, *room\_type*, *minimum\_nights*, *number\_of\_reviews*, *reviews\_per\_month*, *calculated\_host\_listing\_count*, *availability\_365* as our predictors to predict *Price*, the *price* outcome would be either 0 or 1.

**Variable Converting**

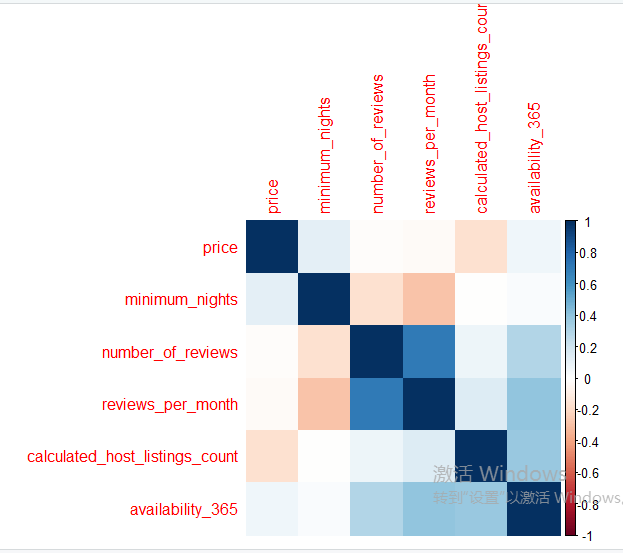
Some of the attributes are not numerical values. For example, in our chosen predictors above, *neighbourhood\_group* and *room\_type* are not numerical variable. In this case, we will convert them into numerical variable for better prediction in our model. Since there are five areas in total, we will label them from 1 to 5. There are three different types of room in total, and we will label them from 1 to 3. For the *Price* attribute, we will calculate the average of all and divide them into two groups 0 and 1. We will show the detailed techniques in the following content.

**IV. Data Mining Techniques and Implementation**

In this part we focus on how to deal with data, we will find a model to predict the price of hosts precisely.

**Correlation Matrix**

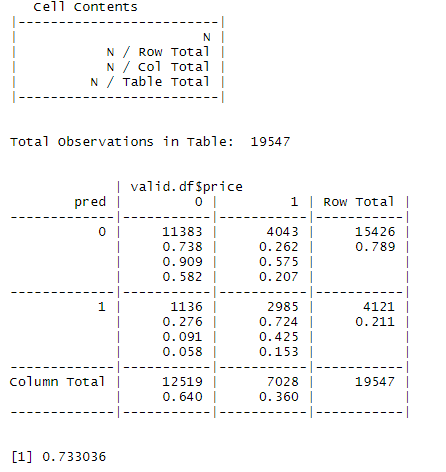
At first, we make a correlation matrix for different aspects of hosts. From the matrix, we can find out the relationship between different variables, then select some useless variables and get rid of them. After that we will use some algorithms to make prediction to the prices of hosts.



The correlation matrix shows that there is a strong positive correlation between *number\_of\_reviews* and *review\_per\_month*. There is a weak positive correlation between *availability\_365* and *calculated\_host\_listings\_count*. There is a weak negative correlation between *minimum\_nights* and *reviews\_per\_month*. From our original data it can be seen that the prices of a host have too many values, it will take longer to make prediction for the accurate price of a host, and maybe computer performance is not enough for making a model. To overcome the difficulties, we simplify the price data at first, and then divide them into two levels.

**Naïve Bayes Model**

The first step is using Naïve Bayes method to make a model to predict the level of price. From the chart above, we can see that the mean of price in New York is $144.41. To simplify the process and make it easier, if the price is greater than or equal to 145, we set the price as 1. If the price is smaller than 145, then we set it to 0. Then we can predict a host whether his/her offered price is above or below the average price. The output is as following:



As we can see, the accuracy rate of validation set is 0.64. The overall accuracy is about 0.733. It is reasonable that the prediction accuracy is kind of low, because the correlation we mentioned above does not shows a strong correlation between predicted value *price* and other attributes. But still, regardless of the quality of the data set, the comparison between different models is the key idea, and our objective is to find the best performance between different models.

**KNN Model**

Then we try to implement KNN method and use cross table to verify the accuracy.

The variables we used for prediction:

neighborhood\_group

room\_type

minimum\_nights

number\_of\_reviews

reviews\_per\_month

calculated\_host\_listing\_count

availability\_365

Since neighborhood\_group and room type are not numeric variables, we change them into numeric variables for better prediction:

For neighborhood groups:

Brooklyn = 1

Manhattan = 2

Queens = 3

Staten Island = 4

Bronx = 5

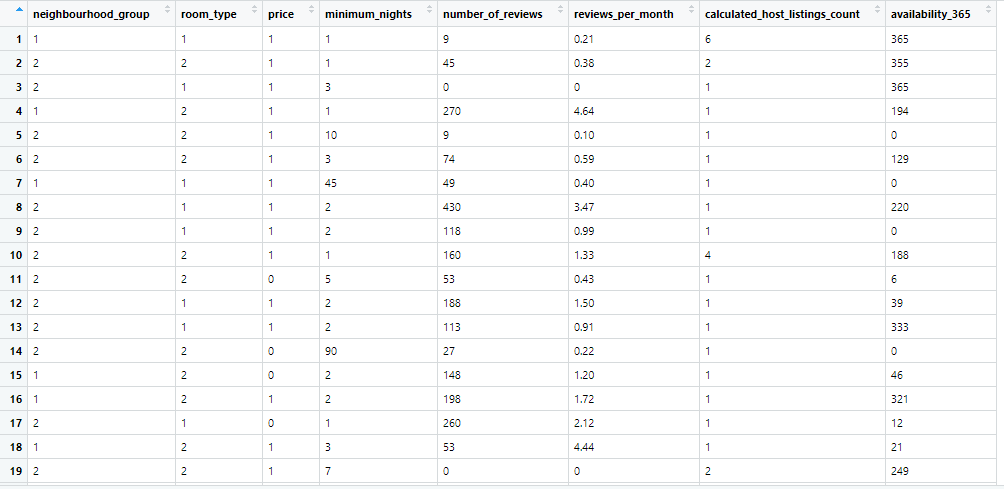
For room type:

Private room = 1

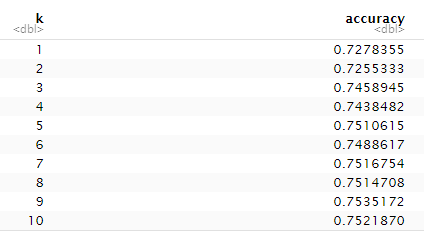
Entire home/apt =2

Shared room =3

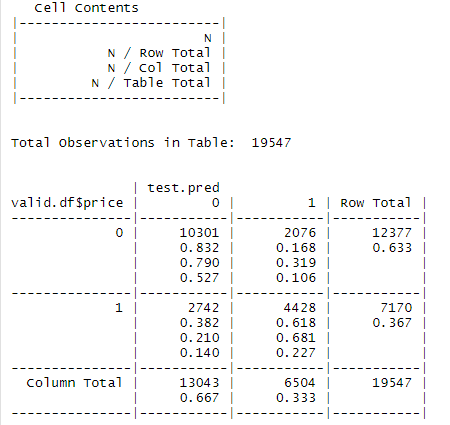
The data frame is as following:



For KNN, we set k values from 1 to 10 to see its performance and test its accuracy as the output. The output is shown in the following:



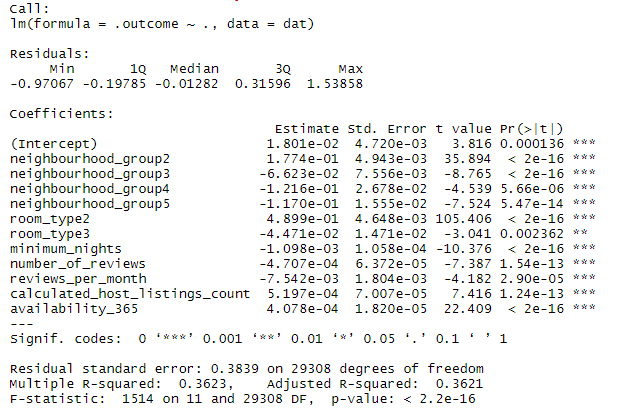
From the result, we can tell that when k=9, the model performs with a best accuracy of 0.7535. After running KNN algorithm, we show the classification matrix for the validation data that results from the best k.



There are 19547 total observations in the table. When using the best k (k=9), the accuracy rate of the validation set is 0.667. The overall accuracy is about 0.7535. We cannot say that the model is not good for not getting high accuracy, but KNN model does perform above the average.

**Linear Regression Model**

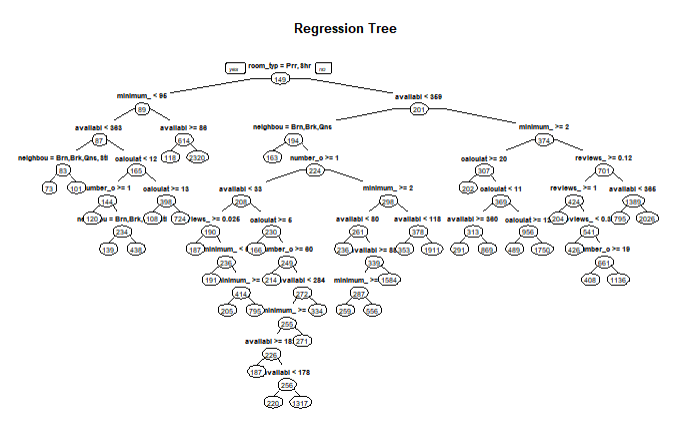
Then we tried to use liner regression model, the result is as following:



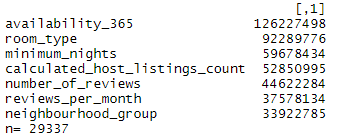
From the R-squared value, we can tell that linear regression explains 36% of the variation in the depend variable. And for the residual’s analysis of the linear regression model, the maximum error is 1.53858 indicates that the model could predict with at least 1.53858 error for one observation.

**Regression Tree**

We implemented regression tree to check the output value of the price. First, we output a regression tree and we want to know the level of importance of each attribute. We want to check which attribute will influence the price most.

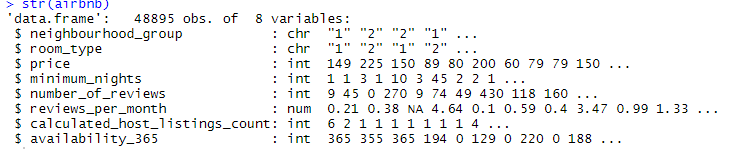


The following table shows the level of importance:

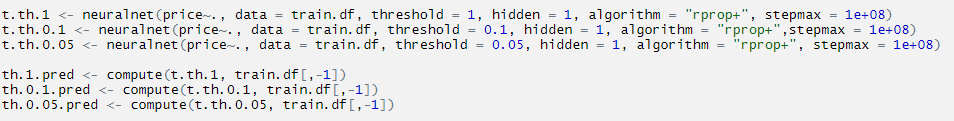


We can say that the *availability\_365* would affect the price most, and *neightborhood\_group* would cause minor effect on the price.

**Neural Network**

****

We also implemented neural network to see the prediction performance. We set different threshold value to see the performance. The threshold values are 1, 0.5, 0.1. To see the performance evaluation, we just need to compute the average RMSE error.



**V. Performance Evaluation**

We calculated RMSE value of three prediction models and compare their RMSE value.

* Naïve Bayes model shows a RMSE value of 0.4623257



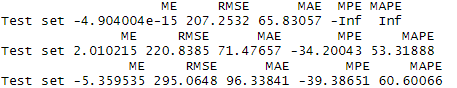
* KNN model shows a RMSE value of

****

* Linear Regression model shows a RMSE value of 0.4627161

****

* Regression tree shows a RMSE value of 220.8385 on the validation set.

****

* The RMSE value of Neural network is about 0.155.

[1] 0.15733873597

[1] 0.148202552042

[1] 0.154720708324

When we compare the value of RMSE from different prediction models, we can conclude that neural network shows the best performance among all.

**VI. Discussion and Recommendation**

We utilized five different models to predict the price of the listings. We analyzed the performance and calculated the error to each of the model. When we interpret the result, we found out if we want to obtain higher accuracy from the model, we can make following recommendations for further model prediction so that we could possibly get a better performance result on the data set.

* More related and useful information is needed for the data set. If we want to predict numerical price of the listings, we might need more information like the price for the past few years. In other words, we could get more accurate result with more positive correlation between different attributes.
* More prediction models can be implemented to compare performance on the data set. In this case, we only use five models. In further learning, we could use more models to make prediction.
* More performance evaluation can be used for better and precise model comparison. The number of performance evaluation we used is limited, we should provide more evaluation such as life chart, ROC curve.
* The data mining techniques can be improved and enhanced. We could standardize the data set before we use the model to do the prediction.

**VII. Summary**

From the prediction process above, we find out that neural network has the best performance. The purpose is not only about getting high accuracy, but also about finding the right model to fit in the data set. In this case, the quality of the relevance of different attributes related to *price* is not ideal. For the entire data set, we selected eight useful predictors to make prediction. Varies method was implemented to clean the data set. We implemented data mining techniques to generate and evaluate different model performance. Finally, we discussed the advantages and disadvantages of different models and make further suggestions for studying. The skills of data processing and model processing still need to be improved, looking forward to more learning in the future.

**Appendix: R Code for use case study**

library(tidyverse)

library(e1071)  
library(class)  
library(sampling)  
library(class)  
library(ggthemes)  
library(GGally)

library(ggExtra)  
library(caret)

library(glmnet)

library(corrplot)

library(leaflet)  
library(kableExtra)

library(RColorBrewer)  
library(plotly)

library(gmodels)  
library(kknn)

# read data  
airbnb<-read.csv("D:/assignment/7275project/AB\_NYC\_2019.csv",stringsAsFactors=FALSE,quote = "",header=T,na.strings=c("","NA"))  
  
th <- theme\_fivethirtyeight() + theme(axis.title = element\_text(), axis.title.x = element\_text()) # global theme for ggplot2 objects  
set.seed(252)  
  
#Data selection  
airbnb$id <- as.factor(airbnb$id)  
airbnb$host\_id <- as.factor(airbnb$host\_id)  
head(airbnb) %>% kable() %>% kable\_styling()

names\_to\_delete <- c("id", "host\_id")  
airbnb[names\_to\_delete] <- NULL  
apply(airbnb,2,function(x) sum(is.na(x)))

apply(airbnb,2,function(x) sum(x==''))

sapply(airbnb, function(x) sum(is.na(x)))

head(airbnb) %>% kable() %>% kable\_styling()

names\_to\_delete <- c("id", "host\_id")  
airbnb[names\_to\_delete] <- NULL

#missing data boxplot  
missing\_airbnb <- airbnb %>% summarise\_all(~(sum(is.na(.))/n()))  
missing\_airbnb <- gather(missing\_airbnb, key = "variables", value = "percent\_missing")  
missing\_airbnb <- missing\_airbnb[missing\_airbnb$percent\_missing > 0.0, ]   
ggplot(missing\_airbnb, aes(x = reorder(variables, percent\_missing), y = percent\_missing)) +  
 geom\_bar(stat = "identity", fill = "grey", aes(color = I('white')), size = 0.3)+  
 xlab('variables')+  
 coord\_flip() +   
 th +  
 ggtitle("Missing Data") +  
 xlab("Column name") +  
 ylab("Percentage missing") +  
 annotate("text", x = 1.5, y = 0.1,label = "host\_name and name have less than 0.001\n percentage missing", color = "slateblue", size = 5)

#price distribution  
airbnb$price<- as.numeric(as.character(airbnb$price))

airbnb$minimum\_nights<- as.numeric(as.character(airbnb$minimum\_nights))

airbnb$number\_of\_reviews<- as.numeric(as.character(airbnb$number\_of\_reviews))

airbnb$reviews\_per\_month<- as.numeric(as.character(airbnb$reviews\_per\_month))

airbnb$calculated\_host\_listings\_count<- as.numeric(as.character(airbnb$calculated\_host\_listings\_count))

airbnb$availability\_365<- as.numeric(as.character(airbnb$availability\_365))

ggplot(airbnb, aes(price)) +  
 geom\_histogram(bins = 30, aes(y = ..density..), fill = "purple") +   
 geom\_density(alpha = 0.2, fill = "purple") +  
 th +  
 ggtitle("Distribution of price",  
 subtitle = "The distribution is very skewed") +  
 theme(axis.title = element\_text(), axis.title.x = element\_text()) +  
 geom\_vline(xintercept = round(mean(airbnb$price,na.rm=TRUE), 2), size = 2, linetype = 3)

ggplot(airbnb, aes(price)) +  
 geom\_density(alpha = 0.2, fill = "purple") +  
 th +  
 ggtitle("Transformed distribution of price",  
 subtitle = expression("With" ~'log'[10] ~ "transformation of x-axis")) +  
 geom\_vline(xintercept = round(mean(airbnb$price,na.rm=TRUE), 2), size = 2, linetype = 3) +  
 scale\_x\_log10() +  
 annotate("text", x = 1800, y = 0.75,label = paste("Mean price = ", paste0(round(mean(airbnb$price,na.rm=TRUE), 2), "$")),  
 size = 8)

## Warning in self$trans$transform(x): 产生了NaNs

## Warning: Transformation introduced infinite values in continuous x-axis

## Warning: Removed 13974 rows containing non-finite values (stat\_density).

#boxplot  
ty\_airbnb<-filter(airbnb, room\_type == "Entire home/apt"|room\_type == "Private room"|room\_type == "Shared room")  
ggplot(ty\_airbnb, aes(x=as.factor(neighbourhood\_group), y=(..count..)\*100/sum(..count..) )) +geom\_bar( fill="brown" )

airbnb$room\_type<-as.factor(airbnb$room\_type)  
ggplot(ty\_airbnb, aes(x = room\_type, y = price)) +  
 geom\_boxplot(aes(fill = room\_type)) + scale\_y\_log10() +  
 th +   
 xlab("Room type") +   
 ylab("Price") +  
 ggtitle("Boxplots of price by room type",  
 subtitle = "Entire homes and apartments have the highest avg price") +  
 geom\_hline(yintercept = mean(airbnb$price), color = "purple", linetype = 2)

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning: Removed 7 rows containing non-finite values (stat\_boxplot).

## Warning: Removed 1 rows containing missing values (geom\_hline).

#correlation matrix between attributes  
airbnb\_cor <- ty\_airbnb[, sapply(ty\_airbnb, is.numeric)]  
airbnb\_cor <- airbnb\_cor[complete.cases(airbnb\_cor), ]  
correlation\_matrix <- cor(airbnb\_cor, method = "spearman")  
corrplot(correlation\_matrix, method = "color")

#data processing  
new.airbnb<-filter(ty\_airbnb,price>10)  
new.airbnb[is.na(new.airbnb)]<-0  
new.airbnb$`price` <- ifelse(new.airbnb$`price` >=145, 1, 0)  
  
  
new.airbnb$room\_type<-str\_replace(new.airbnb$room\_type,'Private room','1')  
new.airbnb$room\_type<-str\_replace(new.airbnb$room\_type,'Entire home/apt','2')  
new.airbnb$room\_type<-str\_replace(new.airbnb$room\_type,'Shared room','3')  
  
  
new.airbnb$neighbourhood\_group<-str\_replace(new.airbnb$neighbourhood\_group,'Brooklyn','1')  
new.airbnb$neighbourhood\_group<-str\_replace(new.airbnb$neighbourhood\_group,'Manhattan','2')  
new.airbnb$neighbourhood\_group<-str\_replace(new.airbnb$neighbourhood\_group,'Queens','3')  
new.airbnb$neighbourhood\_group<-str\_replace(new.airbnb$neighbourhood\_group,'Staten Island','4')  
new.airbnb$neighbourhood\_group<-str\_replace(new.airbnb$neighbourhood\_group,'Bronx','5')  
  
new.airbnb$price<-as.numeric(new.airbnb$price)  
new.airbnb<-new.airbnb[,-c(1,2,4,5,6,11)]

# Naive Bayes model prediction  
train.index<-sample(c(1:dim(new.airbnb)[1]), dim(new.airbnb)[1]\*0.6)  
train.df<-new.airbnb[train.index,]   
valid.df<-new.airbnb[-train.index,]   
  
  
nb\_model<-naiveBayes(as.factor(price)~., data = train.df)   
pred<-predict(nb\_model,valid.df)  
CrossTable(x = pred, y = valid.df$price, prop.chisq = FALSE)

Accuracy=(9504+2660)/16594  
Accuracy

## [1] 0.733036

RMSE <- function(error) { sqrt(mean(error^2)) }  
RMSE(pred)

# KNN model prediction, we split the dataset into 60% training, and 40% validation set.   
  
library(caret)  
library(gmodels)  
set.seed(2)  
  
train.index <- sample(c(1:dim(new.airbnb)[1]), dim(new.airbnb)[1]\*0.6)   
train.df <- new.airbnb[train.index, ]  
valid.df <- new.airbnb[-train.index, ]  
  
train.response <- train.df$price  
valid.response <- valid.df$price  
  
train <- model.matrix(~neighbourhood\_group+room\_type+minimum\_nights+number\_of\_reviews+reviews\_per\_month+calculated\_host\_listings\_count,  
 data = train.df)  
valid <- model.matrix(~neighbourhood\_group+room\_type+minimum\_nights+number\_of\_reviews+reviews\_per\_month+calculated\_host\_listings\_count,  
 data = valid.df)  
  
train.air <- as.data.frame(train)  
valid.air <- as.data.frame(valid)

# Choice of k

knn.1 <- knn(train, valid, train.response, k=1)

knn.2 <- knn(train, valid, train.response, k=2)

knn.3 <- knn(train, valid, train.response, k=3)

knn.4 <- knn(train, valid, train.response, k=4)

knn.5 <- knn(train, valid, train.response, k=5)

knn.6 <- knn(train, valid, train.response, k=6)

knn.7 <- knn(train, valid, train.response, k=7)

knn.8 <- knn(train, valid, train.response, k=8)

knn.9 <- knn(train, valid, train.response, k=9)

knn.10 <- knn(train, valid, train.response, k=10)

confusionMatrix(table(knn.1, valid.response))

confusionMatrix(table(knn.2, valid.response))

confusionMatrix(table(knn.3, valid.response))

confusionMatrix(table(knn.4, valid.response))

confusionMatrix(table(knn.5, valid.response))

confusionMatrix(table(knn.6, valid.response))

confusionMatrix(table(knn.7, valid.response))

confusionMatrix(table(knn.8, valid.response))

confusionMatrix(table(knn.9, valid.response))

confusionMatrix(table(knn.10, valid.response))

# Alternate approach  
  
accuracy.df <- data.frame(k=seq(1,10,1), accuracy = rep(0,10))  
for (i in 1:10){  
 knn.pred <- knn(train = train, test = valid, cl=train.df[, 3] ,k=i)  
 accuracy.df[i,2] <- confusionMatrix(knn.pred, as.factor(valid.df$price))$overall[1]  
}  
accuracy.df

# classification matrix  
test.pred = knn(train = train, test = valid, cl=train.df[, 3] ,k=9)  
CrossTable(x=valid.df$price, y=test.pred, prop.chisq = FALSE)

# RMSE  
rmse <- function(error){sqrt(mean(error^2))}  
rmse(as.numeric(test.pred))

# Linear Regression model prediction  
train.index<-sample(c(1:dim(new.airbnb)[1]), dim(new.airbnb)[1]\*0.6)  
train.df<-new.airbnb[train.index,]   
valid.df<-new.airbnb[-train.index,]  
lm\_model <- train(price ~. , data = train.df, method = "lm")

summary(lm\_model)

prediction<-predict(lm\_model,valid.df)  
RMSE <- function(error) { sqrt(mean(error^2)) }  
RMSE(prediction)

```{r}

# regression tree

library(readxl)

library(class)

library(rpart)

library(rpart.plot)

library(forecast)

# read data

airbnb <- read.csv("C:/Users/Yaodong/Desktop/NYC.csv", header = TRUE, na.strings=c("","NA"))

# Split the data into train 60%, valid 20%, and test dataset 20%

train <- airbnb[1:29337,]

valid <- airbnb[29338:39117,]

test <- airbnb[39118:48895,]

# Construct the regression tree

airbnb.plot <- rpart(price~neighbourhood\_group+room\_type+minimum\_nights+number\_of\_reviews+reviews\_per\_month+calculated\_host\_listings\_count+availability\_365, data = train, method = "anova",cp=0.001)

prp(airbnb.plot, type=1, tweak = 2.2, main="Regression Tree")

# Check importance

t(t(airbnb.plot$variable.importance))

train.data <- train[,c("neighbourhood\_group","room\_type","minimum\_nights","number\_of\_reviews","reviews\_per\_month","calculated\_host\_listings\_count","availability\_365", "price")]

valid.data <- valid[,c("neighbourhood\_group","room\_type","minimum\_nights","number\_of\_reviews","reviews\_per\_month","calculated\_host\_listings\_count","availability\_365","price")]

test.data <- test[,c("neighbourhood\_group","room\_type","minimum\_nights","number\_of\_reviews","reviews\_per\_month","calculated\_host\_listings\_count","availability\_365","price")]

# Return the optimal cp value associated with the minimum error

best.pred <- airbnb$cptable[which.min(airbnb$cptable[,"xerror"]),"CP"]

tree.prune <- prune(airbnb.plot, cp=best.pred)

tree.prune

# Prediction

train.pred <- predict(tree.prune, train.data)

valid.pred <- predict(tree.prune, valid.data)

test.pred <- predict(tree.prune, test.data)

# Compute the accuracy

accuracy(train.pred, train$price)

accuracy(valid.pred, valid$price)

accuracy(test.pred, test$price)

# Compute the Root Mean Square Error

rmse <- function(error){sqrt(mean(error^2))}

train.error <- (train.pred-train$Price)

valid.error <- (valid.pred-valid$Price)

test.error <- (test.pred-test$Price)

rmse(train.error)

rmse(valid.error)

rmse(test.error)

```

```{r}

# Neural network

library(dummies)

library(neuralnet)

library(tidyverse)

airbnb <- read.csv("C:/Users/Yaodong/Desktop/NYC.csv", header = TRUE, na.strings=c("","NA"))

airbnb <- airbnb[,c(5,9,10,11,12,14,15,16)]

airbnb$room\_type<-str\_replace(airbnb$room\_type,'Private room','1')

airbnb$room\_type<-str\_replace(airbnb$room\_type,'Entire home/apt','2')

airbnb$room\_type<-str\_replace(airbnb$room\_type,'Shared room','3')

airbnb$neighbourhood\_group<-str\_replace(airbnb$neighbourhood\_group,'Brooklyn','1')

airbnb$neighbourhood\_group<-str\_replace(airbnb$neighbourhood\_group,'Manhattan','2')

airbnb$neighbourhood\_group<-str\_replace(airbnb$neighbourhood\_group,'Queens','3')

airbnb$neighbourhood\_group<-str\_replace(airbnb$neighbourhood\_group,'Staten Island','4')

airbnb$neighbourhood\_group<-str\_replace(airbnb$neighbourhood\_group,'Bronx','5')

airbnb$neighbourhood\_group<-as.integer(factor(airbnb$neighbourhood\_group))

airbnb$room\_type<-as.integer(factor(airbnb$room\_type))

airbnb$reviews\_per\_month<-as.integer(factor(airbnb$reviews\_per\_month))

airbnb$price<-as.integer(factor(airbnb$price))

airbnb$minimum\_nights<-as.integer(factor(airbnb$minimum\_nights))

airbnb$number\_of\_reviews<-as.integer(factor(airbnb$number\_of\_reviews))

airbnb$calculated\_host\_listings\_count<-as.integer(factor(airbnb$calculated\_host\_listings\_count))

airbnb$availability\_365<-as.integer(factor(airbnb$availability\_365))

airbnb.final <- na.omit(airbnb)

airbnb.final <- as.data.frame(lapply(airbnb.final, normalized.data))

set.seed(20)

normalized.data <- function(a){return((a-mean(a))/(max(a)-min(a)))

}

rmse <- function(b){return(sqrt(mean(b^2)))}

train.data <- sample(row.names(airbnb.final), 0.5\*dim(airbnb.final)[1])

valid.data <- setdiff(row.names(airbnb.final), train.data)

train.df <- airbnb.final[train.data,]

valid.df <- airbnb.final[valid.data,]

t.name <- names(airbnb.final)

t.formula <- as.formula(paste("price~",

paste(t.name[!t.name%in%"price"],

collapse="+")))

t.th.1 <- neuralnet(t.formula, data = train.df, threshold = 1, hidden = 1, algorithm = "rprop+", stepmax = 1e+08)

t.th.0.5 <- neuralnet(t.formula, data = train.df, threshold = 0.5, hidden = 1, algorithm = "rprop+", stepmax = 1e+08)

t.th.0.1 <- neuralnet(t.formula, data = train.df, threshold = 0.1, hidden = 1, algorithm = "rprop+",stepmax = 1e+08)

th.1.pred <- neuralnet::compute(t.th.1, valid.df[,-1])

th.0.5.pred <- neuralnet::compute(t.th.0.5, valid.df[,-1])

th.0.1.pred <- neuralnet::compute(t.th.0.1, valid.df[,-1])

error.th.1 <- th.1.pred$net.result-train.df$price

error.th.0.5 <- th.0.5.pred$net.result-train.df$price

error.th.0.1 <- th.0.1.pred$net.result-train.df$price

options(digits = 12)

rmse(error.th.1)

rmse(error.th.0.5)

rmse(error.th.0.1)