Data Mining and Predictive Analysis

BUDT758T

**A close up of a logo

Description automatically generated**

Project Title:

Exploration, Explanation and Prediction in the world of Investing in Airbnb

Market Assigned:

Las Vegas

“We, the undersigned, certify that the report submitted is our own original work; all authors participated in the work in a substantive way; all authors have seen and approved the report as submitted; the text, images, illustrations and other items included in the manuscript do not carry any infringement/plagiarism issue upon any existing copyrighted materials”

|  |  |
| --- | --- |
|  | **Names of signed team members** |
| Contact member | Shashank Puthanveedu |
| Team member 2 | Tianlin Xiao |
| Team member 3 | Xinyu Zhao |
| Team member 4 | Tanmaya Pisal |
| Team member 5 | Wanzhen Li |
| Team member 6 | Tongyue Hu |

## 

Table of Contents

[I. Executive Summary: 3](#_Toc39819750)

[II. Kaggle Competition 3](#_Toc39819751)

[III. Getting the rows for the “Las Vegas” Market 13](#_Toc39819752)

[IV. Data Cleaning and Preprocessing 14](#_Toc39819753)

[V. Understanding the Market 18](#_Toc39819754)

[VI. Data Exploration 20](#_Toc39819755)

[VII. Prediction of the High Booking Rate 26](#_Toc39819756)

[VIII. Variable Importance and Model Selection 36](#_Toc39819757)

[IX. Explanatory modelling: The age old Logistic Regression to the Rescue 37](#_Toc39819758)

[X. Conclusion and Discussion: 39](#_Toc39819759)

[XI. References: 39](#_Toc39819760)

## I. Executive Summary:

The primary goal of the project is to understand and analyze the trends in the Airbnb booking rates in the Las Vegas market. Booking rate in Airbnb is the ratio of how many people looked at your listing vs how many people actually booked. Thus, booking rate could be a measure of how well you are doing as an Airbnb host. But it is not a complete and concise metric. Though having a high booking rate is clear indication of a flourishing host a low booking rate cannot be used for inferences. It may be misleading at times. For example, if a host has his property completely booked for the next 3 months, his booking rate would be low as his listing is not available to book for people who view the listing. Thus, we need to take other factors into consideration before making an inference from a low booking rate.

In our study we particularly study the probability of having a high booking rate for the Airbnb listings in Las Vegas. The report includes an extensive summary of the market research conducted by the team and the results and inferences from the data analysis. Based on the market study we can arrive at a decision that Airbnb in Las Vegas is a good investment for small investors, and it is possible for them to make a profit. Further, we looked at the factors that need to be considered by a new investor while buying a property. We built predictive models catering to the varying risk profile of the investor to make predictions on the future high booking rate of their prospective properties. We also developed explanatory models to explain the high booking rate in present listings and to identify the factors that influence it the most. These factors can be useful for the existing investors who wish to upgrade their properties to have a better booking rate. The report also covers the exploratory analysis performed to find the correlations between the high booking rate and individual market factors. The report tries to make a few inferences and conclusion by blending the market analysis with the findings from the data modelling and predictive analytics to achieve perfect symbiosis. Thus, we will be able to predict and explain the high booking rate among Airbnb listings in the Vegas market.

The report also covers details of the Kaggle Competition where our team analyzed the patterns in high booking rate for Airbnb listing across the United States to create a predictive model to predict the same. We used multiple data pre-processing, modelling and predictive analytics techniques to develop a model that could predict the high booking rate in the properties with an AUC of 0.94545. The final model used by the team for the Kaggle competition was an XGBoost model with the parameters as nrounds = 100, max\_depth = 7, eta = 0.3, gamma = 0, colsample\_bytree = 0.4, min\_child\_weight = 2.25 and subsample = 1. These parameters were decided based on the performance of the model while tuning using a grid search. AUC over the 10-fold cross validations was used as a metric of performance while tuning. The main variables in the predictive model were review\_scores\_checkin, review\_scores\_rating, host\_is\_superhostTRUE, host\_listings\_count, review\_scores\_value, host\_since, self\_check\_in, family\_kids\_friendly, minimum\_nights, reviews, cleaning\_fee, hot\_waterTRUE, host\_response\_timewithin an hour, price, internet. The report covers the R code used for the Kaggle competition for quick reference.

## II. Kaggle Competition

**Load and pre process the data:**

We read the Train and Test data sets into data frames so that we can do pre processing steps to improve data quality.

tsLCOrg <-  
 read\_csv("airbnbTrain.csv") %>%   
 rename\_all(tolower)

head(tsLCOrg)

Remove all the text and highly biased variables. The text variables can be used in future for extracting new dummy variables to better explain the high booking rate. For the current scope of the project we are removing all the text variables.

dfAirbnbTrainSet <-  
 tsLCOrg %>%   
 select(-access, -description,-host\_about,-host\_location,-host\_neighbourhood,-house\_rules,-interaction,-latitude,-longitude,-neighbourhood,-neighborhood\_overview,-notes,-space,-square\_feet,-transit,-zipcode,-monthly\_price,-weekly\_price,-city, -state,-'{randomcontrol}',-host\_acceptance\_rate,-host\_response\_rate,-host\_has\_profile\_pic,-is\_business\_travel\_ready,-require\_guest\_phone\_verification,-require\_guest\_profile\_picture)

Extract features from the list variables “amenities” and store them into dummy variables for further use.

dfAirbnbTrainSet$amenities <- lapply(strsplit(as.character(dfAirbnbTrainSet$amenities),split=','),trimws)  
  
dfAirbnbTrainSet <-  
 dfAirbnbTrainSet %>%   
 mutate(no\_of\_amenities=lengths(amenities))  
  
dfAirbnbTrainSet <- dfAirbnbTrainSet %>%   
 mutate(pool = str\_detect(dfAirbnbTrainSet$amenities,"Pool"),  
 wifi = str\_detect(dfAirbnbTrainSet$amenities,"Wifi"),  
 essentials = str\_detect(dfAirbnbTrainSet$amenities,"Essentials"),  
 free\_parking\_on\_premises= str\_detect(dfAirbnbTrainSet$amenities,"Free parking on premises"),  
 long\_term\_stays\_allowed = str\_detect(dfAirbnbTrainSet$amenities,"Long term stays allowed"),  
 family\_kids\_friendly = str\_detect(dfAirbnbTrainSet$amenities,"Family/kid friendly"),  
 laptop\_friendly\_workspace = str\_detect(dfAirbnbTrainSet$amenities, "Laptop friendly workspace"),  
 refridgerator = str\_detect(dfAirbnbTrainSet$amenities,"Refrigerator"),  
 coffee\_maker = str\_detect(dfAirbnbTrainSet$amenities, "Coffee maker"),  
 gym = str\_detect(dfAirbnbTrainSet$amenities, "Gym"),  
 self\_check\_in = str\_detect(dfAirbnbTrainSet$amenities, "Self check-in"),  
 cooking\_basics = str\_detect(dfAirbnbTrainSet$amenities, "Cooking basics"),  
 microwave = str\_detect(dfAirbnbTrainSet$amenities, "Microwave"),  
 smoke\_detector = str\_detect(dfAirbnbTrainSet$amenities, "Smoke detector"),  
 host\_greets = str\_detect(dfAirbnbTrainSet$amenities, "Host greets you"),  
 first\_aid = str\_detect(dfAirbnbTrainSet$amenities, "First aid kit"),  
 washer = str\_detect(dfAirbnbTrainSet$amenities, "Washer"),  
 dryer = str\_detect(dfAirbnbTrainSet$amenities, "Dryer"),  
 bedroom\_lock = str\_detect(dfAirbnbTrainSet$amenities, "Lock on bedroom door"),  
 hot\_water = str\_detect(dfAirbnbTrainSet$amenities, "Hot water"),  
 heating = str\_detect(dfAirbnbTrainSet$amenities, "Heating"),  
 fire\_extinguisher = str\_detect(dfAirbnbTrainSet$amenities, "Fire extinguisher"),  
 safety\_card = str\_detect(dfAirbnbTrainSet$amenities, "Safety card"),  
 private\_entrance = str\_detect(dfAirbnbTrainSet$amenities, "Private entrance"),  
 dishes = str\_detect(dfAirbnbTrainSet$amenities, "Dishes and silverware"),  
 luggage = str\_detect(dfAirbnbTrainSet$amenities, "Luggage dropoff allowed"),  
 cable = str\_detect(dfAirbnbTrainSet$amenities, "Cable TV"),  
 breakfast = str\_detect(dfAirbnbTrainSet$amenities, "Breakfast"),  
 dishwasher = str\_detect(dfAirbnbTrainSet$amenities, "Dishwasher"),  
 events = str\_detect(dfAirbnbTrainSet$amenities, "Suitable for events"),  
 linens = str\_detect(dfAirbnbTrainSet$amenities, "Bed linens"),  
 balcony = str\_detect(dfAirbnbTrainSet$amenities, "Patio or balcony"),  
 garden = str\_detect(dfAirbnbTrainSet$amenities, "Garden or backyard"),  
 living\_room = str\_detect(dfAirbnbTrainSet$amenities, "Private living room"),  
 lockbox = str\_detect(dfAirbnbTrainSet$amenities, "Lockbox"),  
 fireplace = str\_detect(dfAirbnbTrainSet$amenities, "Indoor fireplace"),  
 hr\_checkin = str\_detect(dfAirbnbTrainSet$amenities, "24-hour check-in"),  
 bbq = str\_detect(dfAirbnbTrainSet$amenities, "BBQ grill"),  
 internet = str\_detect(dfAirbnbTrainSet$amenities, "Internet"),  
 hairdryer = str\_detect(dfAirbnbTrainSet$amenities, "Hair dryer"),  
 air\_con = str\_detect(dfAirbnbTrainSet$amenities, "Air conditioning"),  
 cleaning = str\_detect(dfAirbnbTrainSet$amenities, "Cleaning before checkout"),  
 entryway = str\_detect(dfAirbnbTrainSet$amenities, "Wide entrance"),  
 pets = str\_detect(dfAirbnbTrainSet$amenities, "Pets allowed"),  
 smoking = str\_detect(dfAirbnbTrainSet$amenities, "Smoking allowed"),  
 shower = str\_detect(dfAirbnbTrainSet$amenities, "Step-free shower"),  
 hot\_tub = str\_detect(dfAirbnbTrainSet$amenities, "Hot tub"),  
 no\_stairs = str\_detect(dfAirbnbTrainSet$amenities, "No stairs or steps to enter"),  
 paid\_parking = str\_detect(dfAirbnbTrainSet$amenities, "Paid parking off premises"))  
  
  
head(dfAirbnbTrainSet$no\_of\_amenities)

Extract features from the list variables “host\_verifications” and store them into dummy variables for further use.

dfAirbnbTrainSet$host\_verifications <- lapply(strsplit(as.character(dfAirbnbTrainSet$host\_verifications),split=','),trimws)  
  
dfAirbnbTrainSet <-  
 dfAirbnbTrainSet %>%   
 mutate(no\_of\_host\_verifications=lengths(host\_verifications))  
  
dfAirbnbTrainSet <- dfAirbnbTrainSet %>%   
 mutate(government\_id = str\_detect(dfAirbnbTrainSet$host\_verifications,"government\_id"),  
 reviews = str\_detect(dfAirbnbTrainSet$host\_verifications,"reviews"),  
 jumio = str\_detect(dfAirbnbTrainSet$host\_verifications,"jumio"),  
 kba = str\_detect(dfAirbnbTrainSet$host\_verifications, "kba"))  
  
head(dfAirbnbTrainSet$no\_of\_host\_verifications)

Drop the list based columns which are no longer needed as we have extracted the required information from them

dfFinalTrain <-  
 dfFinalTrain %>%   
 select(-amenities,-host\_verifications)

Reduce the levels in the factor variables “market” and “property\_type”. This helps us remove the levels which have very few observations. These levels may cause a bias in the model and also lead to rank-deficiency errors in the model.

dfFinalTrain <-  
 dfAirbnbTrainSet %>%   
 group\_by(market) %>%   
 mutate(Flag = ifelse (n() < 100, 1, 0)) %>%   
 ungroup %>%   
 mutate(market = ifelse(Flag == 1, "Other (Domestic)", market)) %>%   
 select(-Flag)

dfFinalTrain$market[dfFinalTrain$market=="South Bay CA"]<-"South Bay, CA"  
dfFinalTrain$market[dfFinalTrain$market=="East Bay CA"]<-"East Bay, CA"

dfFinalTrain <-  
 dfFinalTrain %>%   
 group\_by(property\_type) %>%   
 mutate(Flag = ifelse (n() < 20, 1, 0)) %>%   
 ungroup %>%   
 mutate(property\_type = ifelse(Flag == 1, "Other", property\_type)) %>%   
 select(-Flag)

Drop the rows from the test data where we have NAs and we can see that it could be a mis representation of the data. Like a house must have a bathroom, beds and market information. But if the listing has these columns as NA we cannot impute them as these are data discrepancies.

dfFinalTrain <-  
 dfFinalTrain %>%   
 drop\_na(host\_is\_superhost,bathrooms,beds,market)

Perform similar operations on the Test data to make it synchronous with the Train data. This is essential as the discrepancies between the two will stop us from being able to generate the predicted probabilities as the new levels or variables in test data cannot be understood by the model that was built on the train data.

tsLCOrgTest <-  
 read\_csv("airbnbTest.csv") %>%   
 rename\_all(tolower)

dfAirbnbTestSet <-  
 tsLCOrgTest %>%   
 select(-access, -description,-host\_about,-host\_location,-host\_neighbourhood,-house\_rules,-interaction,-latitude,-longitude,-neighbourhood,-neighborhood\_overview,-notes,-space,-square\_feet,-transit,-zipcode,-monthly\_price,-weekly\_price,-city, -state,-'{randomcontrol}',-host\_acceptance\_rate,host\_response\_rate,-host\_has\_profile\_pic,-is\_business\_travel\_ready,-require\_guest\_phone\_verification,-require\_guest\_profile\_picture)

dfAirbnbTestSet$amenities <- lapply(strsplit(as.character(dfAirbnbTestSet$amenities),split=','),trimws)  
  
dfAirbnbTestSet <-  
 dfAirbnbTestSet %>%   
 mutate(no\_of\_amenities=lengths(amenities))  
  
dfAirbnbTestSet <- dfAirbnbTestSet %>%   
 mutate(pool = str\_detect(dfAirbnbTestSet$amenities,"Pool"),  
 wifi = str\_detect(dfAirbnbTestSet$amenities,"Wifi"),  
 essentials = str\_detect(dfAirbnbTestSet$amenities,"Essentials"),  
 free\_parking\_on\_premises= str\_detect(dfAirbnbTestSet$amenities,"Free parking on premises"),  
 long\_term\_stays\_allowed = str\_detect(dfAirbnbTestSet$amenities,"Long term stays allowed"),  
 family\_kids\_friendly = str\_detect(dfAirbnbTestSet$amenities,"Family/kid friendly"),  
 laptop\_friendly\_workspace = str\_detect(dfAirbnbTestSet$amenities, "Laptop friendly workspace"),  
 refridgerator = str\_detect(dfAirbnbTestSet$amenities,"Refrigerator"),  
 coffee\_maker = str\_detect(dfAirbnbTestSet$amenities, "Coffee maker"),  
 gym = str\_detect(dfAirbnbTestSet$amenities, "Gym"),  
 self\_check\_in = str\_detect(dfAirbnbTestSet$amenities, "Self check-in"),  
 cooking\_basics = str\_detect(dfAirbnbTestSet$amenities, "Cooking basics"),  
 microwave = str\_detect(dfAirbnbTestSet$amenities, "Microwave"),  
 smoke\_detector = str\_detect(dfAirbnbTestSet$amenities, "Smoke detector"),  
 host\_greets = str\_detect(dfAirbnbTestSet$amenities, "Host greets you"),  
 first\_aid = str\_detect(dfAirbnbTestSet$amenities, "First aid kit"),  
 washer = str\_detect(dfAirbnbTestSet$amenities, "Washer"),  
 dryer = str\_detect(dfAirbnbTestSet$amenities, "Dryer"),  
 bedroom\_lock = str\_detect(dfAirbnbTestSet$amenities, "Lock on bedroom door"),  
 hot\_water = str\_detect(dfAirbnbTestSet$amenities, "Hot water"),  
 heating = str\_detect(dfAirbnbTestSet$amenities, "Heating"),  
 fire\_extinguisher = str\_detect(dfAirbnbTestSet$amenities, "Fire extinguisher"),  
 safety\_card = str\_detect(dfAirbnbTestSet$amenities, "Safety card"),  
 private\_entrance = str\_detect(dfAirbnbTestSet$amenities, "Private entrance"),  
 dishes = str\_detect(dfAirbnbTestSet$amenities, "Dishes and silverware"),  
 luggage = str\_detect(dfAirbnbTestSet$amenities, "Luggage dropoff allowed"),  
 cable = str\_detect(dfAirbnbTestSet$amenities, "Cable TV"),  
 breakfast = str\_detect(dfAirbnbTestSet$amenities, "Breakfast"),  
 dishwasher = str\_detect(dfAirbnbTestSet$amenities, "Dishwasher"),  
 events = str\_detect(dfAirbnbTestSet$amenities, "Suitable for events"),  
 linens = str\_detect(dfAirbnbTestSet$amenities, "Bed linens"),  
 balcony = str\_detect(dfAirbnbTestSet$amenities, "Patio or balcony"),  
 garden = str\_detect(dfAirbnbTestSet$amenities, "Garden or backyard"),  
 living\_room = str\_detect(dfAirbnbTestSet$amenities, "Private living room"),  
 lockbox = str\_detect(dfAirbnbTestSet$amenities, "Lockbox"),  
 fireplace = str\_detect(dfAirbnbTestSet$amenities, "Indoor fireplace"),  
 hr\_checkin = str\_detect(dfAirbnbTestSet$amenities, "24 -hour check-in"),  
 bbq = str\_detect(dfAirbnbTestSet$amenities, "BBQ grill"),  
 internet = str\_detect(dfAirbnbTestSet$amenities, "Internet"),  
 hairdryer = str\_detect(dfAirbnbTestSet$amenities, "Hair dryer"),  
 air\_con = str\_detect(dfAirbnbTestSet$amenities, "Air conditioning"),  
 cleaning = str\_detect(dfAirbnbTestSet$amenities, "Cleaning before checkout"),  
 entryway = str\_detect(dfAirbnbTestSet$amenities, "Wide entrance"),  
 pets = str\_detect(dfAirbnbTestSet$amenities, "Pets allowed"),  
 smoking = str\_detect(dfAirbnbTestSet$amenities, "Smoking allowed"),  
 shower = str\_detect(dfAirbnbTestSet$amenities, "Step-free shower"),  
 hot\_tub = str\_detect(dfAirbnbTestSet$amenities, "Hot tub"),  
 no\_stairs = str\_detect(dfAirbnbTestSet$amenities, "No stairs or steps to enter"),  
 paid\_parking = str\_detect(dfAirbnbTestSet$amenities, "Paid parking off premises"))  
  
head(dfAirbnbTestSet$no\_of\_amenities)

dfAirbnbTestSet$host\_verifications <- lapply(strsplit(as.character(dfAirbnbTestSet$host\_verifications),split=','),trimws)  
  
dfAirbnbTestSet <-  
 dfAirbnbTestSet %>%   
 mutate(no\_of\_host\_verifications=lengths(host\_verifications))  
  
dfAirbnbTestSet <- dfAirbnbTestSet %>%   
 mutate(government\_id = str\_detect(dfAirbnbTestSet$host\_verifications,"government\_id"),  
 reviews = str\_detect(dfAirbnbTestSet$host\_verifications,"reviews"),  
 jumio = str\_detect(dfAirbnbTestSet$host\_verifications,"jumio"),  
 kba = str\_detect(dfAirbnbTestSet$host\_verifications, "kba"))  
  
head(dfAirbnbTestSet$no\_of\_host\_verifications)

dfFinalTest <-  
 dfAirbnbTestSet %>%   
 group\_by(market) %>%   
 mutate(Flag = ifelse (n() < 100, 1, 0)) %>%   
 ungroup %>%   
 mutate(market = ifelse(Flag == 1, "Other (Domestic)", market)) %>%   
 select(-Flag)

dfFinalTest <-  
 dfFinalTest %>%   
 group\_by(property\_type) %>%   
 mutate(Flag = ifelse (n() < 50, 1, 0)) %>%   
 ungroup %>%   
 mutate(property\_type = ifelse(Flag == 1, "Other", property\_type)) %>%   
 select(-Flag)

dfFinalTest$market[dfFinalTest$market=="South Bay CA"]<-"South Bay, CA"  
dfFinalTest$market[dfFinalTest$market=="East Bay CA"]<-"East Bay, CA"

dfFinalTest$host\_response\_time[is.na(dfFinalTest$host\_response\_time)]<-'N/A'  
dfFinalTest$host\_identity\_verified[is.na(dfFinalTest$host\_identity\_verified)]<-FALSE  
dfFinalTest$host\_is\_superhost[is.na(dfFinalTest$host\_is\_superhost)]<-FALSE  
  
dfFinalTest$bathrooms[is.na(dfFinalTest$bathrooms)]<-1  
dfFinalTest$beds[is.na(dfFinalTest$beds)]<-2  
  
dfFinalTest$host\_listings\_count[is.na(dfFinalTest$host\_listings\_count)]<-1  
dfFinalTest$host\_since[is.na(dfFinalTest$host\_since)]<-as.Date('2015-11-09')

dfFinalTest <-  
 dfFinalTest %>%   
 select(-amenities,-host\_verifications)

Next we re-code the variable type in the system. We convert the variables to numeric, factor and date so that we can easily build models using them. We also had to perform some basic string operations to convert the currency data into numeric format. We also performed some imputations in the data wherever applicable.

dfFinalTrain$cleaning\_fee <- as.numeric(gsub('\\$|,','', dfFinalTrain$cleaning\_fee))  
dfFinalTest$cleaning\_fee <- as.numeric(gsub('\\$|,','', dfFinalTest$cleaning\_fee))

dfFinalTrain$cleaning\_fee[is.na(dfFinalTrain$cleaning\_fee)]<-88.6  
dfFinalTest$cleaning\_fee[is.na(dfFinalTest$cleaning\_fee)]<-94.9

dfFinalTrain$security\_deposit <- as.numeric(gsub('\\$|,','', dfFinalTrain$security\_deposit))  
dfFinalTest$security\_deposit <- as.numeric(gsub('\\$|,','', dfFinalTest$security\_deposit))

dfFinalTrain$security\_deposit[is.na(dfFinalTrain$security\_deposit)]<-231.0  
dfFinalTest$security\_deposit[is.na(dfFinalTest$security\_deposit)]<-228.0

dfFinalTest$host\_listings\_count[is.na(dfFinalTest$host\_listings\_count)]<-1

dfFinalTrain$review\_scores\_rating[is.na(dfFinalTrain$review\_scores\_rating)]<- 0  
dfFinalTest$review\_scores\_rating[is.na(dfFinalTest$review\_scores\_rating)]<-0  
  
dfFinalTrain$review\_scores\_accuracy[is.na(dfFinalTrain$review\_scores\_accuracy)]<-0  
dfFinalTest$review\_scores\_accuracy[is.na(dfFinalTest$review\_scores\_accuracy)]<-0  
   
dfFinalTrain$review\_scores\_checkin[is.na(dfFinalTrain$review\_scores\_checkin)]<-0  
dfFinalTest$review\_scores\_checkin[is.na(dfFinalTest$review\_scores\_checkin)]<-0  
   
dfFinalTrain$review\_scores\_cleanliness[is.na(dfFinalTrain$review\_scores\_cleanliness)]<-0  
dfFinalTest$review\_scores\_cleanliness[is.na(dfFinalTest$review\_scores\_cleanliness)]<-0  
  
dfFinalTrain$review\_scores\_communication[is.na(dfFinalTrain$review\_scores\_communication)]<-0  
dfFinalTest$review\_scores\_communication[is.na(dfFinalTest$review\_scores\_communication)]<-0  
   
dfFinalTrain$review\_scores\_location[is.na(dfFinalTrain$review\_scores\_location)]<-0  
dfFinalTest$review\_scores\_location[is.na(dfFinalTest$review\_scores\_location)]<-0  
  
dfFinalTrain$review\_scores\_value[is.na(dfFinalTrain$review\_scores\_value)]<-0  
dfFinalTest$review\_scores\_value[is.na(dfFinalTest$review\_scores\_value)]<-0

dfFinalTest$host\_since[is.na(dfFinalTest$host\_since)]<-as.Date('2015-11-09')

dfFinalTrain$bedrooms[is.na(dfFinalTrain$bedrooms)]<-1  
dfFinalTest$bedrooms[is.na(dfFinalTest$bedrooms)]<-1

dfFinalTrain$price <- as.numeric(gsub('\\$|,','', dfFinalTrain$price))  
dfFinalTrain$extra\_people <- as.numeric(gsub('\\$|,','', dfFinalTrain$extra\_people))  
dfFinalTest$price <- as.numeric(gsub('\\$|,','', dfFinalTest$price))  
dfFinalTest$extra\_people<- as.numeric(gsub('\\$|,','', dfFinalTest$extra\_people))  
  
dfFinalTrain$no\_of\_amenities <- as.numeric(dfFinalTrain$no\_of\_amenities)  
dfFinalTest$no\_of\_amenities <- as.numeric(dfFinalTest$no\_of\_amenities)

colsToFactorTrain <-  
 c('high\_booking\_rate', 'cancellation\_policy', 'instant\_bookable','host\_identity\_verified','host\_is\_superhost',"host\_response\_time","is\_location\_exact","market","requires\_license","room\_type","property\_type","bed\_type","government\_id","jumio","reviews","free\_parking\_on\_premises","pool","wifi","essentials","long\_term\_stays\_allowed","family\_kids\_friendly","laptop\_friendly\_workspace","refridgerator", "coffee\_maker","gym","self\_check\_in","cooking\_basics","microwave","smoke\_detector","host\_greets","washer","dryer","first\_aid","bedroom\_lock","hot\_water","kba","heating", "fire\_extinguisher", "safety\_card","private\_entrance","dishes","luggage","cable","breakfast","dishwasher","events","linens","balcony","garden","living\_room","lockbox","fireplace","hr\_checkin","bbq","internet","hairdryer","air\_con","cleaning","entryway","pets","smoking","shower","hot\_tub","no\_stairs","wheelchair","paid\_parking")  
  
colsToFactorTest <-  
 c('cancellation\_policy', 'instant\_bookable','host\_identity\_verified','host\_is\_superhost',"host\_response\_time","is\_location\_exact","market","requires\_license","room\_type","property\_type","bed\_type","government\_id","jumio","reviews","free\_parking\_on\_premises","pool","wifi","essentials","long\_term\_stays\_allowed","family\_kids\_friendly","laptop\_friendly\_workspace","refridgerator", "coffee\_maker","gym","self\_check\_in","cooking\_basics","microwave","smoke\_detector","host\_greets","washer","dryer","first\_aid","bedroom\_lock","hot\_water","kba","heating", "fire\_extinguisher", "safety\_card","private\_entrance","dishes","luggage","cable","breakfast","dishwasher","events","linens","balcony","garden","living\_room","lockbox","fireplace","hr\_checkin","bbq","internet","hairdryer","air\_con","cleaning","entryway","pets","smoking","shower","hot\_tub","no\_stairs","wheelchair","paid\_parking")

dfFinalTrain <-  
 dfFinalTrain %>%   
 mutate\_at(colsToFactorTrain, ~factor(.))  
  
dfFinalTest <-  
 dfFinalTest%>%   
 mutate\_at(colsToFactorTest, ~factor(.))

**Building the Model**

Once we have the train and test datasets ready with all the required variables and the correct formatting we can go ahead and build the model.

We tried a lot of models ranging from basic linear probability model, to logistic model to ensemble models like bagged trees, random forests and XGBoost. Finally we choose the XGBoost model as it gave a better performance in terms of AUC which was the evaluating criteria for the Kaggle Competition. Once we finalized the model we performed a grid tuning for finding the optimal parameters for the model. This enabled us to fine tune the model to get the best result out of it.

The tune grid used for the purpose is as below,

tune.gridxgb <- expand.grid(eta = c(0.05,0.3, 0.075), # 3   
 nrounds = c(50, 75, 100), # 3  
 max\_depth = 4:7, # 4  
 min\_child\_weight = c(2.0, 2.25), #2   
 colsample\_bytree = c(0.3, 0.4, 0.5), # 3  
 gamma = 0, #1  
 subsample = 1) # 1  
# 3\*3\*4\*2\*3\*1\*1 = 216  
dim(tune.gridxgb)

We used 10-fold cross validation to validate and estimate the best model tuning parameters.

train.control <- trainControl(  
 method = "repeatedcv",  
 number = 10, ## 10-fold CV  
 repeats = 3,## repeated three times  
 summaryFunction = twoClassSummary,   
 classProbs = TRUE  
 )

Finally, we built the XGBoost model using the “xgbTree” method in Caret library. The performance metric to be used for deciding the best model was set as “ROC” which is the AUC value for each cross validation.

gridxgbFit <- train(make.names(high\_booking\_rate)~.-id,   
 data=dfFinalTrain,   
 method = "xgbTree",   
 tuneGrid =tune.gridxgb,  
 trControl = train.control,  
 metric="ROC"  
 )

Once the results were obtained, we validated the maximum AUC achieved by the model and the corresponding tuning parameters. This, helped us judge the model before actually uploading it to Kaggle.

gridxgbFit$results %>%   
 arrange(desc(ROC))

The Variable importance of the model helped us understand which are the important variables towards the prediction. The Important variables from this output are as below, review\_scores\_checkin, review\_scores\_rating, host\_is\_superhostTRUE, host\_listings\_count, review\_scores\_value, host\_since, self\_check\_in, family\_kids\_friendly, minimum\_nights, reviews, cleaning\_fee, hot\_waterTRUE, host\_response\_timewithin an hour, price, internet.

varImp(gridxgbFit)$importance %>% # Add scale=FALSE inside VarImp if you don't want to scale  
 rownames\_to\_column(var = "Variable") %>%  
 mutate(Importance = scales::percent(Overall/100)) %>%   
 arrange(desc(Overall)) %>%   
 as\_tibble()

**Making predictions**

Generate the predicted probabilities for each entry in the test data set and store the value as a new column called “predictedProb”.

resultsXGBoost <-  
 gridxgbbFit %>%   
 predict(dfFinalTest, type='prob') %>%   
 bind\_cols(dfFinalTest, predictedProb=.$X1)

Rename the “predictedProb” column to “high\_booking\_rate” to match with the Kaggle upload format and guidelines.

uploadDf <-  
 resultsXGBoost %>%   
 select(predictedProb, id) %>%   
 rename(high\_booking\_rate = predictedProb )

Generate the file to upload into the Kaggle submissions and store it in the machine.

uploadDf %>%  
 write\_csv("data/upload/KaggleComp\_Upload.csv")

## III. Getting the rows for the “Las Vegas” Market

Load the entire train set into the R environment

dfAirBnBTrain <- read\_csv("data/airbnbTrain.csv")

## Parsed with column specification:  
## cols(  
## .default = col\_character(),  
## id = col\_double(),  
## high\_booking\_rate = col\_double(),  
## accommodates = col\_double(),  
## availability\_30 = col\_double(),  
## availability\_365 = col\_double(),  
## availability\_60 = col\_double(),  
## availability\_90 = col\_double(),  
## bathrooms = col\_double(),  
## bedrooms = col\_double(),  
## beds = col\_double(),  
## guests\_included = col\_double(),  
## host\_has\_profile\_pic = col\_logical(),  
## host\_identity\_verified = col\_logical(),  
## host\_is\_superhost = col\_logical(),  
## host\_listings\_count = col\_double(),  
## host\_since = col\_date(format = ""),  
## instant\_bookable = col\_logical(),  
## is\_business\_travel\_ready = col\_logical(),  
## is\_location\_exact = col\_logical(),  
## latitude = col\_double()  
## # ... with 15 more columns  
## )

## See spec(...) for full column specifications.

We extract the data for Las Vegas from the entire dataset using the “randomControl” which is used for housekeeping purposes in the dataset. We take all the rows which have randomControl between 111000 and 112000 and then we also filter by market to remove any anomalous rows that have entered the list. We validated the row count for Las Vegas and found that the extracted dataframe contains all the Las Vegas listings.

dfLasVegas <-  
 dfAirBnBTrain %>%   
 filter(`{randomControl}` >= 111000 & `{randomControl}` <112000) %>%   
 filter(market == "Las Vegas")

Ensure that there are no listings outside of Las Vegas in the data.

unique(dfLasVegas$market)

## [1] "Las Vegas"

Store the data for future Reference.

dfLasVegas %>%  
 write\_csv("data/LasVegasData.csv")

## IV. Data Cleaning and Preprocessing

Data cleaning and preprocessing is again the most important step in any analysis. This helps us to build the best possible model from the available data.

dfLasVegasFull <-  
 read\_csv("LasVegasData.csv") %>%   
 rename\_all(tolower)

## Parsed with column specification:  
## cols(  
## .default = col\_character(),  
## id = col\_double(),  
## high\_booking\_rate = col\_double(),  
## accommodates = col\_double(),  
## availability\_30 = col\_double(),  
## availability\_365 = col\_double(),  
## availability\_60 = col\_double(),  
## availability\_90 = col\_double(),  
## bathrooms = col\_double(),  
## bedrooms = col\_double(),  
## beds = col\_double(),  
## guests\_included = col\_double(),  
## host\_has\_profile\_pic = col\_logical(),  
## host\_identity\_verified = col\_logical(),  
## host\_is\_superhost = col\_logical(),  
## host\_listings\_count = col\_double(),  
## host\_since = col\_date(format = ""),  
## instant\_bookable = col\_logical(),  
## is\_business\_travel\_ready = col\_logical(),  
## is\_location\_exact = col\_logical(),  
## latitude = col\_double()  
## # ... with 17 more columns  
## )

## See spec(...) for full column specifications.

Remove all the variables that will not be used for analysis. These include text variables, variables with highly biased observations and the variables which have no significance to the business problem being studied.

dfLasVegas <-  
 dfLasVegasFull %>%   
 select(-access, -description,-host\_about,-host\_location,-host\_neighbourhood,-house\_rules,-interaction,-latitude,-longitude,-neighbourhood,-neighborhood\_overview,-notes,-space,-square\_feet,-transit,-zipcode,-monthly\_price,-weekly\_price,-city, -state,-'{randomcontrol}',-host\_acceptance\_rate,-host\_response\_rate,-host\_has\_profile\_pic,-is\_business\_travel\_ready,-require\_guest\_phone\_verification,-require\_guest\_profile\_picture)

Extract more information from the variables and create dummy variables to hold the new information.

dfLasVegas$amenities <- lapply(strsplit(as.character(dfLasVegas$amenities),split=','),trimws)  
  
dfLasVegas <-  
 dfLasVegas %>%   
 mutate(no\_of\_amenities=lengths(amenities))  
  
dfLasVegas <- dfLasVegas %>%   
 mutate(pool = str\_detect(dfLasVegas$amenities,"Pool"),  
 wifi = str\_detect(dfLasVegas$amenities,"Wifi"),  
 essentials = str\_detect(dfLasVegas$amenities,"Essentials"),  
 free\_parking\_on\_premises= str\_detect(dfLasVegas$amenities,"Free parking on premises"),  
 long\_term\_stays\_allowed = str\_detect(dfLasVegas$amenities,"Long term stays allowed"),  
 family\_kids\_friendly = str\_detect(dfLasVegas$amenities,"Family/kid friendly"),  
 laptop\_friendly\_workspace = str\_detect(dfLasVegas$amenities, "Laptop friendly workspace"),  
 refridgerator = str\_detect(dfLasVegas$amenities,"Refrigerator"),  
 coffee\_maker = str\_detect(dfLasVegas$amenities, "Coffee maker"),  
 gym = str\_detect(dfLasVegas$amenities, "Gym"))

dfLasVegas$host\_verifications <- lapply(strsplit(as.character(dfLasVegas$host\_verifications),split=','),trimws)  
  
dfLasVegas <-  
 dfLasVegas %>%   
 mutate(no\_of\_host\_verifications=lengths(host\_verifications))  
  
dfLasVegas <- dfLasVegas %>%   
 mutate(government\_id = str\_detect(dfLasVegas$host\_verifications,"government\_id"),  
 reviews = str\_detect(dfLasVegas$host\_verifications,"reviews"),  
 jumio = str\_detect(dfLasVegas$host\_verifications,"jumio"))

Drop the list based columns which are no longer needed as we have extracted the required information from them

dfLasVegas <-  
 dfLasVegas %>%   
 select(-amenities,-host\_verifications)

Reduce the levels in the factor variables “market” and “property\_type”. This helps us remove the levels which have very few observations. These levels may cause a bias in the model and also lead to rank-deficiency errors in the model.

dfLasVegas <-  
 dfLasVegas %>%   
 group\_by(property\_type) %>%   
 mutate(Flag = ifelse (n() < 20, 1, 0)) %>%   
 ungroup %>%   
 mutate(property\_type = ifelse(Flag == 1, "OTHER", property\_type)) %>%   
 select(-Flag)

Next we re-code the variable type in the system. We convert the variables to numeric, factor and date so that we can easily build models using them. We also had to perform some basic string operations to convert the currency data into numeric format. We also performed some imputations in the data wherever applicable.

dfLasVegas$host\_response\_time[is.na(dfLasVegas$host\_response\_time)]<-'N/A'  
dfLasVegas$host\_identity\_verified[is.na(dfLasVegas$host\_identity\_verified)]<-FALSE  
dfLasVegas$host\_is\_superhost[is.na(dfLasVegas$host\_is\_superhost)]<-FALSE  
  
dfLasVegas$bathrooms[is.na(dfLasVegas$bathrooms)]<-1  
dfLasVegas$beds[is.na(dfLasVegas$beds)]<-2  
dfLasVegas$bedrooms[is.na(dfLasVegas$bedrooms)]<-2  
  
dfLasVegas$host\_listings\_count[is.na(dfLasVegas$host\_listings\_count)]<-1  
dfLasVegas$host\_since[is.na(dfLasVegas$host\_since)]<-as.Date('2015-11-09')

dfLasVegas$price <- as.numeric(gsub('\\$|,','', dfLasVegas$price))  
dfLasVegas$extra\_people <- as.numeric(gsub('\\$|,','', dfLasVegas$extra\_people))  
dfLasVegas$cleaning\_fee <- as.numeric(gsub('\\$|,','', dfLasVegas$cleaning\_fee))  
dfLasVegas$security\_deposit <- as.numeric(gsub('\\$|,','', dfLasVegas$security\_deposit))

dfLasVegas$cleaning\_fee[is.na(dfLasVegas$cleaning\_fee)]<-99.2  
dfLasVegas$security\_deposit[is.na(dfLasVegas$security\_deposit)]<-0

dfLasVegas$review\_scores\_rating[is.na(dfLasVegas$review\_scores\_rating)]<-0  
  
dfLasVegas$review\_scores\_accuracy[is.na(dfLasVegas$review\_scores\_accuracy)]<-0  
  
dfLasVegas$review\_scores\_checkin[is.na(dfLasVegas$review\_scores\_checkin)]<-0  
  
dfLasVegas$review\_scores\_cleanliness[is.na(dfLasVegas$review\_scores\_cleanliness)]<-0  
  
dfLasVegas$review\_scores\_communication[is.na(dfLasVegas$review\_scores\_communication)]<-0  
   
dfLasVegas$review\_scores\_location[is.na(dfLasVegas$review\_scores\_location)]<-0  
  
dfLasVegas$review\_scores\_value[is.na(dfLasVegas$review\_scores\_value)]<-0

colsToFactor <-  
 c('high\_booking\_rate','bed\_type', 'cancellation\_policy', 'host\_response\_time', 'market','property\_type','room\_type', 'host\_identity\_verified','host\_is\_superhost','instant\_bookable',"is\_location\_exact","requires\_license","pool","wifi","essentials","free\_parking\_on\_premises","long\_term\_stays\_allowed","family\_kids\_friendly","laptop\_friendly\_workspace","refridgerator","coffee\_maker","gym","government\_id","reviews","jumio")

dfLasVegas <-  
 dfLasVegas %>%   
 mutate\_at(colsToFactor, ~factor(.))

Drop “market” and “requires\_license” variables as these variables have the same value for all the listings in the dataset. This is as expected, and we do not need these values to explain any features. Market is common for all listings as we are studying the same market Las Vegas. Also, requires license is not significant as the regulations on short term rentals in Las Vegas require all owners to have a license before they can list their properties on Airbnb.

dfLasVegas <-  
 dfLasVegas %>%   
 select(-market,-requires\_license)

## V. Understanding the Market

Now that we have the data ready let’s try and understand the market we are studying and the investment we are making.

**Features of the Market**

* Airbnb is primarily a combination of Hospitality and Real estate Industry.
* Las Vegas is famous for its casino industry together with the hotel industry because most casinos are owned by the hotel.
* No Seasonality in the market as tourism flourishes in Las Vegas all year round.
* Growing tourism industry with around 42.5 Million people visiting Las Vegas in 2019 which was a million more than 2018.
* Las Vegas has about 113 Hotels with over 100, 000 beds.
* Las Vegas has about 6000 Airbnb listings with about 15000 beds.
* Out of these 4782 listings have a low booking rate which is about 77% of the total listings.

**So what is the market like in Vegas**

To understand the investment options lets study the market from 4 perspectives,

1. **Competition**

Clearly, in Vegas the main competition to Airbnb is Hotels,

*Hotels*

* Hotels have the business center and located in the business district, and people can save time on the transportation. This makes them an attractive choice for business conventions and meetings.
* Most of the hotels in Las Vegas are big brands and chain hotels like MGM, they have the reputation for comfortable room, fancy environment and professional service.
* Price plays little role for small groups. The Airbnb and the Hotel listings for 2 people cost the same.
* The hotel has a comparable advantage in the Strip area since they offer complimentary services like, pool and casino access and entertainment shows.
* All these things make them a great choice for the avid traveller.

*Airbnb*

* 10% of Visitors came to Vegas for a Business Convention. Airbnb misses out this chunk of visitors.
* Sometimes, the condition of Airbnb is not as good as it shown on the web page, which is under-expected. This leads to lack of trust and customer retention is difficult.
* Price is the advantage of Airbnb which materializes if you are travelling in large groups.
* The Airbnb could have the advantage on the outdoor area like the Red Rock Canyon National Conservation Area.
* Airbnbs are generally just bed and breakfast which provide a place to stay with not many other facilities which makes them functional but less fancy and attractive for the tourists.

1. **Supply Side i.e. the real estate market of Las Vegas**

* Real estate market in Las Vegas is currently stable with a median house price of $220K and 1- year appreciation rate of 9%.
* There is a wide range of properties available from $19K to $21000K.

1. **Demand Side i.e. the tourists who visit Las Vegas**

* Among visitors who stayed overnight in Las Vegas, 86% stayed in a hotel.
* Mean age of tourists visiting Las Vegas was 45.1 Years with 20% of them in the age group between 21-29.
* Average size of a group was 2.2 visitors.
* They stayed an average 4.4 days and 3.4 nights during the trip.
* 15% of the visitors bought a tour package for their trip.

1. **Market Regulations**

Las Vegas has a lot of regulations concerning short term rentals.

* A non-refundable $500 permit application fee on any would-be short-term rental operator.
* Limiting overnight guests to 12 or fewer per property.
* Denying such permits to any new Airbnbs within 660 feet of any existing listing.
* If a home has five or more bedrooms, owners must also keep a licensed security company on call to tackle complaints.
* Most parts have declared short term rentals illegal.

All the regulations seem fair when we consider that there are many jobs offered to residents by these hotels or casinos. If the Airbnb take away the market from hotels, many people will lose their jobs, so to protect people not lose their jobs, it’s reasonable that local government want to limit the development of the Airbnb.

But there is always a silver lining,

AirBnB has been approved and is legal in the Las Vegas/Henderson market as of 10/14/2019 and since then there has been a steady boom in Airbnbs and many people are interested in investing in the same.

**So is Airbnb a good investment option?**

*Advantages of Airbnb*

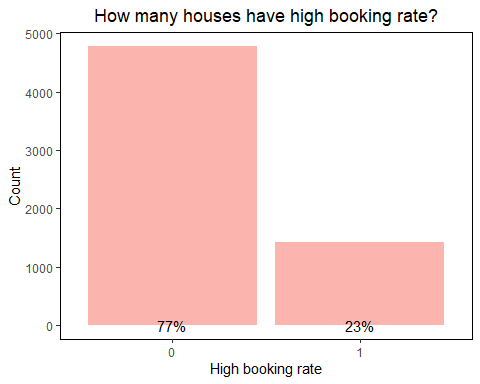
* Low cost of investment as compared to investing in the hotel industry.
* Operation and maintenance costs are also low.
* Competition is less fierce than the hotel industry and is not monopolized by few major players.
* Power of the Airbnb platform is a great advantage with network effects.
* Ease of disinvestment as selling a housing property at the right price is possible.

## VI. Data Exploration

1. Check the proportion of listings with low booking vs high booking rates in Las Vegas. This enables us to understand and set a baseline for our analysis.

*Answer:* We find that the number of houses with a low booking rate is way much larger. Almost 77% of the houses have low booking rates

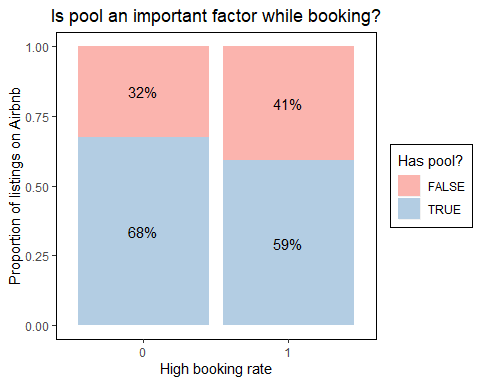
plot\_high\_booking <-   
 ggplot(dfLasVegas,aes(x=high\_booking\_rate,fill="blue"))+  
 geom\_bar() +ggtitle("How many houses have high booking rate?") +  
 geom\_text(aes(label=scales::percent(..count../sum(..count..))),  
 stat='count',position=position\_fill(vjust=0.5)) +  
 theme(  
 legend.position = "none",   
 plot.title = element\_text(hjust = 0.5),  
 panel.background = element\_rect(fill = "transparent"),   
 plot.background = element\_rect(fill = "transparent", color = NA),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),   
 legend.background = element\_rect(fill = "transparent"),   
 legend.box.background = element\_rect(fill = "transparent"),  
 panel.border = element\_rect(colour = "black", fill=NA)  
 )+  
 labs(x="High booking rate", y = "Count")+  
 scale\_fill\_brewer(palette="Pastel1")  
   
plot\_high\_booking



1. Usually hotels provide a lot of complimentary services a pool is among them. Maybe listings in Airbnb don’t have a pool, so people choose not to book, is this the reason?

*Answer:* The plot shows that among houses with a low booking rate, about 2/3 have pools. Besides, among houses with a high booking rate, only a little bit more than 1/2 have pools. Thus, pool seems not a main factor that affect booking rate.

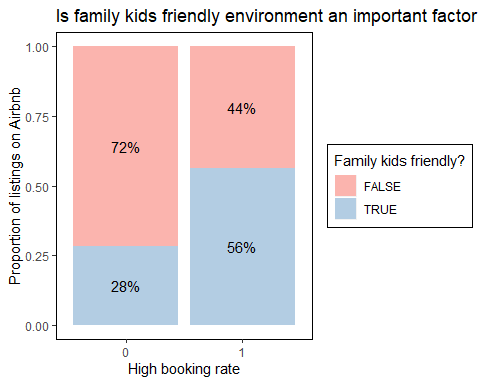
percentData <- dfLasVegas %>% group\_by(high\_booking\_rate) %>% count(pool) %>%  
 mutate(ratio=scales::percent(n/sum(n)))  
  
plot\_pool <-  
 ggplot(data = dfLasVegas, aes(x=high\_booking\_rate,fill = pool))+  
 geom\_bar(position="fill") +ggtitle("Is pool an important factor while booking?")+  
 geom\_text(data=percentData, aes(y=n,label=ratio),position=position\_fill(vjust=0.5)) +  
 theme(  
 plot.title = element\_text(hjust = 0.5),  
 panel.background = element\_rect(fill = "transparent"),   
 plot.background = element\_rect(fill = "transparent", color = NA),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),   
 legend.background = element\_rect(fill = "transparent"),   
 legend.box.background = element\_rect(fill = "transparent"),  
 panel.border = element\_rect(colour = "black", fill=NA)  
 )+  
 labs(x="High booking rate", y = "Proportion of listings on Airbnb")+  
 scale\_fill\_brewer(palette="Pastel1", name = "Has pool?")  
  
plot\_pool



1. If pool is not the reason, how about listings that are family friendly family\_kids\_friendly?

*Answer:* The plot shows that among listings with a low booking rate, about 72% aren’t family\_kids\_friendly. Thus, family\_kids\_friendly seems to affect booking rate positively.

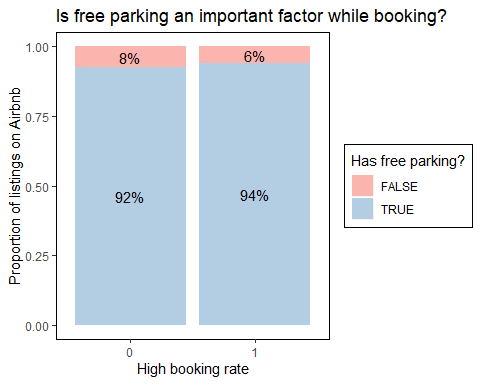
percentData <- dfLasVegas %>% group\_by(high\_booking\_rate) %>% count(family\_kids\_friendly) %>%  
 mutate(ratio=scales::percent(n/sum(n)))  
  
plot\_family\_kid <-  
 ggplot(data = dfLasVegas, aes(x=high\_booking\_rate,fill = family\_kids\_friendly))+  
 geom\_bar(position="fill") +ggtitle("Is family kids friendly environment an important factor while booking?")+  
 geom\_text(data=percentData, aes(y=n,label=ratio),position=position\_fill(vjust=0.5)) +  
 theme(  
 panel.background = element\_rect(fill = "transparent"),   
 plot.background = element\_rect(fill = "transparent", color = NA),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),   
 legend.background = element\_rect(fill = "transparent"),   
 legend.box.background = element\_rect(fill = "transparent"),  
 panel.border = element\_rect(colour = "black", fill=NA)  
 )+  
 labs(x="High booking rate", y = "Proportion of listings on Airbnb")+  
 scale\_fill\_brewer(palette="Pastel1", name = "Family kids friendly?")  
  
plot\_family\_kid



1. How about parking issue? Will people refuse to rent a property becuase of its lack of free parking?

*Answer:* The plot shows that among properties with a low booking rate, almost all of them provide free parking. Also, almost all of the properties have free parking. So parking is not a big factor for booking rate.

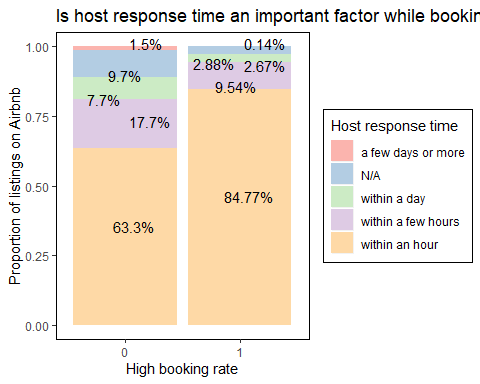
percentData <- dfLasVegas %>% group\_by(high\_booking\_rate) %>% count(free\_parking\_on\_premises) %>%  
 mutate(ratio=scales::percent(n/sum(n)))  
  
plot\_parking<-  
 ggplot(data = dfLasVegas, aes(x=high\_booking\_rate,fill = free\_parking\_on\_premises)) +  
 geom\_bar(position = "fill") + ggtitle("Is free parking an important factor while booking?")+  
 geom\_text(data=percentData, aes(y=n,label=ratio),position=position\_fill(vjust=0.5)) +  
 theme(  
 panel.background = element\_rect(fill = "transparent"),   
 plot.background = element\_rect(fill = "transparent", color = NA),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),   
 legend.background = element\_rect(fill = "transparent"),   
 legend.box.background = element\_rect(fill = "transparent"),  
 panel.border = element\_rect(colour = "black", fill=NA)  
 )+  
 labs(x="High booking rate", y = "Proportion of listings on Airbnb")+  
 scale\_fill\_brewer(palette="Pastel1", name = "Has free parking?")  
  
plot\_parking



1. Many customers will send hosts questions while reviewing airbnb website, some people will book other houses if the host cannot reply quickly. Seems host\_response\_time is related to high booking rate in this way, is this true?

*Answer:* About 84% of the hosts with high booking rate responded within the hour, this number is only close to 63% for the listings with low booking rate. Thus, a quick response does have an impact on the booking rates.

library(ggrepel)  
  
percentData <- dfLasVegas %>% group\_by(high\_booking\_rate) %>% count(host\_response\_time) %>%  
 mutate(ratio=scales::percent(n/sum(n)))  
  
plot\_time<-  
 ggplot(data = dfLasVegas, aes(x=high\_booking\_rate,fill = host\_response\_time )) +  
 geom\_bar(position = "fill") +ggtitle("Is host response time an important factor while booking?") +  
 geom\_text\_repel(data=percentData, aes(y=n,label=ratio),position=position\_fill(vjust=0.5)) +  
 theme(  
 panel.background = element\_rect(fill = "transparent"),   
 plot.background = element\_rect(fill = "transparent", color = NA),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),   
 legend.background = element\_rect(fill = "transparent"),   
 legend.box.background = element\_rect(fill = "transparent"),  
 panel.border = element\_rect(colour = "black", fill=NA)  
 )+  
 labs(x="High booking rate", y = "Proportion of listings on Airbnb")+  
 scale\_fill\_brewer(palette="Pastel1", name = "Host response time")  
  
plot\_time

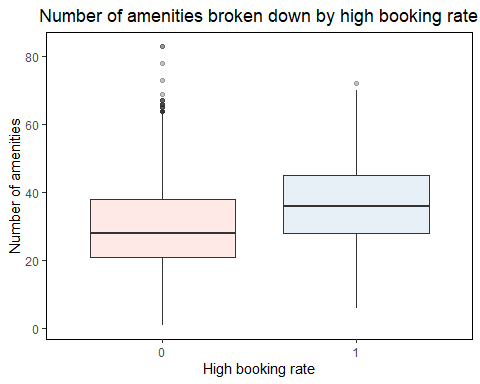


1. Explore whether amenities is associated with high booking rate

*Answer:* The boxplot shows the mean of number of amenities is higher for the houses with a high\_booking\_rate, so it seems to be associated with high booking rate.

plot\_no\_amentities<-  
 ggplot(data=dfLasVegas, aes(x=high\_booking\_rate, y= no\_of\_amenities, fill = high\_booking\_rate)) +  
 geom\_boxplot(alpha = 0.3) + ggtitle("Number of amenities broken down by high booking rate")+  
 theme(  
 legend.position="none",  
 plot.title = element\_text(hjust = 0.5),  
 panel.background = element\_rect(fill = "transparent"),   
 plot.background = element\_rect(fill = "transparent", color = NA),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),  
 panel.border = element\_rect(colour = "black", fill=NA)  
 )+  
 labs(x="High booking rate", y = "Number of amenities")+  
 scale\_fill\_brewer(palette="Pastel1")  
   
   
  
ggplotly(plot\_no\_amentities)

plot\_no\_amentities



## VII. Prediction of the High Booking Rate

**For the Investor who wants to buy a new property**

It is essential for us to be able to predict if a new property will have a high booking rate or not accurately. This will enable new investors to make informed investment choices and decisions.

For this purpose, we tried multiple models viz., Linear Probability model, Logistic Regression model, Latent discriminant analysis, Quadratic discriminant analysis, KNN model, Decision tree and bagged trees.

Based on the comparison of performance metrics of the above models on the test data, we decided to go ahead with bagged trees model for our business case. Below is the predictive analysis,

Split the data set into test and train sets

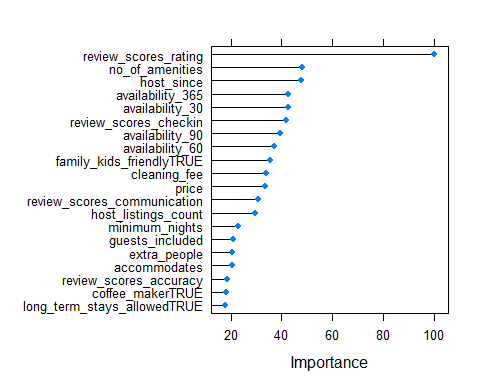
#Setting the seed  
set.seed(123)  
  
#Partitioning the dataset into a 70% - 30% split  
dfLasVegasTrain <- dfLasVegas %>% sample\_frac(.7)  
dfLasVegasTest <- setdiff(dfLasVegas, dfLasVegasTrain)

Run the model on the train data using “treebag” method in the caret library.

set.seed(123)  
fitBaggedTree <- train(high\_booking\_rate ~ .-id, data=dfLasVegasTrain, method='treebag', trControl=trainControl(method='cv', number=10))

See the variables plotted by importance (according to the bagged tree)

plot(varImp(fitBaggedTree), top=20)



Make predictions and generate the predicted Probability for each observation in the test data set

resultsBaggedTree <-  
 fitBaggedTree %>%   
 predict(dfLasVegasTest, type='prob') %>%   
 bind\_cols(dfLasVegasTest, predictedProb=.$'1')

Before we start the analysis based on sensitivity, specificity and accuracy lets define some terms with respect to our business case,

* **False Positives**: When the model predicts the listing to have high booking rate but in reality, it does not have a high booking rate.
* **False Negatives**: When the model predicts the listing to not have high booking rate but in reality, it does have a high booking rate.

We need a model that will reduce the False Positives as they can cause real monetary losses to the investor. False Negatives lead to the investor losing out on a potentially good investment, but the loss is virtual as there is no real monetary losses involved.

Thus, generally we would like to focus on specificity and accuracy as our metrics and then sensitivity.

Get the sensitivity, specificity and accuracy using the AUC library for all the possible cut off values.

library(AUC)

## AUC 0.3.0

## Type AUCNews() to see the change log and ?AUC to get an overview.

##   
## Attaching package: 'AUC'

## The following objects are masked from 'package:caret':  
##   
## sensitivity, specificity

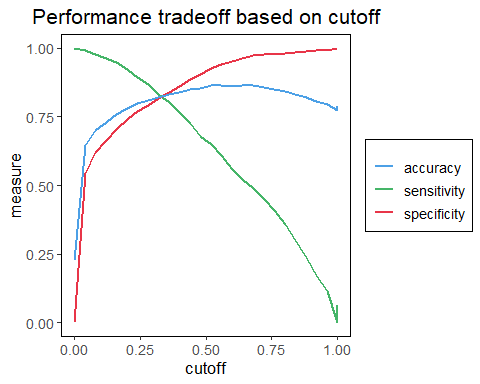
## The following object is masked from 'package:yardstick':  
##   
## accuracy

sens<- AUC::sensitivity(resultsBaggedTree$predictedProb,resultsBaggedTree$high\_booking\_rate, perc.rank = FALSE)  
spec <- AUC::specificity(resultsBaggedTree$predictedProb,resultsBaggedTree$high\_booking\_rate, perc.rank = FALSE)  
acc <- AUC::accuracy(resultsBaggedTree$predictedProb,resultsBaggedTree$high\_booking\_rate, perc.rank = FALSE)

Plot the sensitivity, specificity and accuracy in a single plot.

plot <-  
 ggplot()+  
 geom\_line(aes(x=sens$cutoffs,y=sens$measure,colour="sensitivity"),size=1)+  
 geom\_line(aes(x=spec$cutoffs,y=spec$measure, colour="specificity"),size=1)+  
 geom\_line(aes(x=acc$cutoffs,y=acc$measure,colour="accuracy"),size=1)+  
 xlab("cutoff") + ylab("measure") +  
 ggtitle("Performance tradeoff based on cutoff") +  
 guides(colour=guide\_legend(title=NULL)) +  
 theme(  
 plot.title = element\_text(hjust = 0.5),  
 text = element\_text(size = 13),  
 panel.background = element\_rect(fill = "transparent"),   
 plot.background = element\_rect(fill = "transparent", color = NA),   
 panel.grid.major = element\_blank(),   
 panel.grid.minor = element\_blank(),   
 legend.background = element\_rect(fill = "transparent"),   
 legend.box.background = element\_rect(fill = "transparent"),  
 legend.key = element\_rect(fill = "transparent"),  
 panel.border = element\_rect(colour = "black", fill=NA)  
 )+  
 scale\_color\_manual(name = "", values = c("accuracy"='#4ca0e6',"sensitivity"='#45b568',"specificity"='#e83548'))  
  
ggplotly(plot)

plot



**For the Risk Neutral Investor:**

The Risk neutral investor does not take risk into account while investing. So all he would care about is the accuracy of the prediction. As per the graph plotted earlier the highest accuracy is achieved at a cut off of 0.5

resultsBaggedTree <- resultsBaggedTree %>%  
 mutate(predictedClass = as.factor(ifelse(predictedProb > 0.5, 1, 0)),)

Below is the performance of the model at a cut off of 0.5

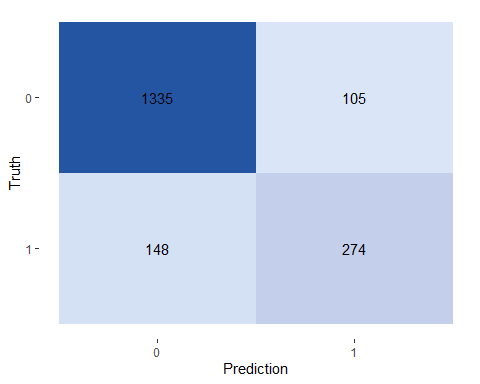
resultsBaggedTree %>%   
 xtabs(~predictedClass+high\_booking\_rate, .) %>%   
 confusionMatrix(positive = '1')

## Confusion Matrix and Statistics  
##   
## high\_booking\_rate  
## predictedClass 0 1  
## 0 1335 148  
## 1 105 274  
##   
## Accuracy : 0.8641   
## 95% CI : (0.8477, 0.8794)  
## No Information Rate : 0.7734   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5979   
##   
## Mcnemar's Test P-Value : 0.008278   
##   
## Sensitivity : 0.6493   
## Specificity : 0.9271   
## Pos Pred Value : 0.7230   
## Neg Pred Value : 0.9002   
## Prevalence : 0.2266   
## Detection Rate : 0.1472   
## Detection Prevalence : 0.2035   
## Balanced Accuracy : 0.7882   
##   
## 'Positive' Class : 1   
##

Below is the confusion matrix of the model prediction. This looks quite good with high specificity which is desirable.

resultsBaggedTree %>%   
 conf\_mat(truth=high\_booking\_rate, estimate=predictedClass) %>%   
 autoplot(type="heatmap") +  
 scale\_fill\_gradient(low="#dae5f7", high ="#2455a3")

## Scale for 'fill' is already present. Adding another scale for 'fill', which  
## will replace the existing scale.



**For the Risk Seeking Investor:**

The Risk seeking investor cares about making a profit while investing. They may not mind a few losses So False negatives should be as low as possible as the false negatives means investment opportunity missed. So, all they would care about is the sensitivity of the prediction. As per the graph plotted earlier the highest sensitivity is achieved at a cut-off of 0.3 without largely compromising on the accuracy and specificity.

resultsBaggedTree <- resultsBaggedTree %>%  
 mutate(predictedClass = as.factor(ifelse(predictedProb > 0.3, 1, 0)),)

Below is the performance of the model at a cut off of 0.3

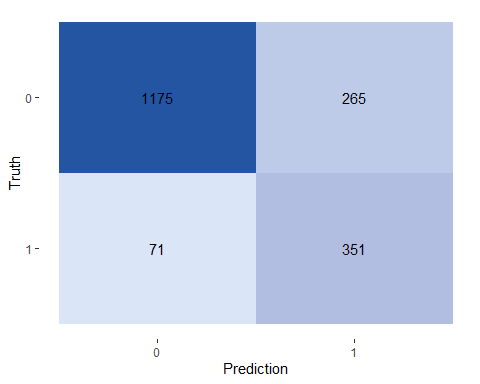
resultsBaggedTree %>%   
 xtabs(~predictedClass+high\_booking\_rate, .) %>%   
 confusionMatrix(positive = '1')

## Confusion Matrix and Statistics  
##   
## high\_booking\_rate  
## predictedClass 0 1  
## 0 1175 71  
## 1 265 351  
##   
## Accuracy : 0.8195   
## 95% CI : (0.8013, 0.8368)  
## No Information Rate : 0.7734   
## P-Value [Acc > NIR] : 6.071e-07   
##   
## Kappa : 0.5572   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8318   
## Specificity : 0.8160   
## Pos Pred Value : 0.5698   
## Neg Pred Value : 0.9430   
## Prevalence : 0.2266   
## Detection Rate : 0.1885   
## Detection Prevalence : 0.3308   
## Balanced Accuracy : 0.8239   
##   
## 'Positive' Class : 1   
##

Below is the confusion matrix of the model prediction. This looks quite good with high sensitivity which is desirable for a risk seeking investor as it can make sure they do not miss out on any property that has a chance of high booking rate.

resultsBaggedTree %>%   
 conf\_mat(truth=high\_booking\_rate, estimate=predictedClass) %>%   
 autoplot(type="heatmap") +  
 scale\_fill\_gradient(low="#dae5f7", high ="#2455a3")

## Scale for 'fill' is already present. Adding another scale for 'fill', which  
## will replace the existing scale.



**For the Risk Averse Investor:**

The Risk Averse investor cares about making a profit while investing but without making any losses. They want to minimize the risk as much as possible. The False positives should be as low as possible. So, all they would care about is the specificity of the prediction. As per the graph plotted earlier the highest specificity is achieved at a cut-off of 0.7 without largely compromising on accuracy and sensitivity.

resultsBaggedTree <- resultsBaggedTree %>%  
 mutate(predictedClass = as.factor(ifelse(predictedProb > 0.7, 1, 0)),)

Below is the performance of the model at a cut off of 0.7

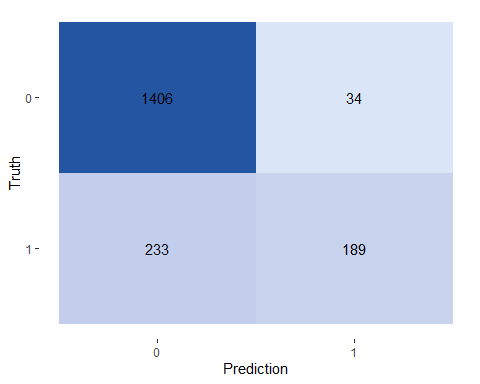
resultsBaggedTree %>%   
 xtabs(~predictedClass+high\_booking\_rate, .) %>%   
 confusionMatrix(positive = '1')

## Confusion Matrix and Statistics  
##   
## high\_booking\_rate  
## predictedClass 0 1  
## 0 1406 233  
## 1 34 189  
##   
## Accuracy : 0.8566   
## 95% CI : (0.8399, 0.8722)  
## No Information Rate : 0.7734   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5091   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.4479   
## Specificity : 0.9764   
## Pos Pred Value : 0.8475   
## Neg Pred Value : 0.8578   
## Prevalence : 0.2266   
## Detection Rate : 0.1015   
## Detection Prevalence : 0.1198   
## Balanced Accuracy : 0.7121   
##   
## 'Positive' Class : 1   
##

Below is the confusion matrix of the model prediction. This looks quite good with high specificity which is desirable for a risk averse investor as it can make sure they invest only on property that has a chance of high booking rate.

resultsBaggedTree %>%   
 conf\_mat(truth=high\_booking\_rate, estimate=predictedClass) %>%   
 autoplot(type="heatmap") +  
 scale\_fill\_gradient(low="#dae5f7", high ="#2455a3")

## Scale for 'fill' is already present. Adding another scale for 'fill', which  
## will replace the existing scale.



## VIII. Variable Importance and Model Selection

**For the Investor who is invested and wants to improve the booking rate**

For someone who is invested and wants to improve the booking rate they need to look at the factors that influence the booking rate the most. The factors that can increase the chances of the listing having a high booking rate the to the maximum.

Model selection and variable importance can be done by a variety of methods viz., stepwise selection, backward or forward selection, Lasso, Ridge, Group Lasso or Elastic net.

We tried all the approaches and chose elastic net as it gave us the best of both lasso and ridge while giving us the highest accuracy in the test data.

We used a grid search in elastic net to find the best tune.

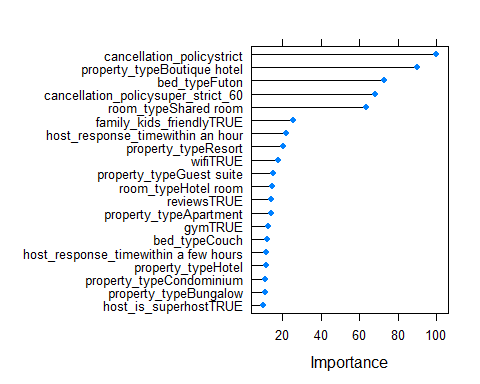
Below is the elastic net output,

set.seed(123)  
dflTrain <- dfLasVegas %>% sample\_frac(0.7)  
dflTest <- dplyr::setdiff(dfLasVegas,dflTrain)

set.seed(123)  
fitElasticNetGrid <- train(high\_booking\_rate ~ .-id, family='binomial', data=dflTrain, method='glmnet', trControl=trainControl(method='cv', number=10), tuneLength=10)

This graph shows us the important factors that an existing investor can use to upgrade himself and his listing.

plot(varImp(fitElasticNetGrid), top = 20)



## IX. Explanatory modelling: The age old Logistic Regression to the Rescue

Though we got the variable importance using elastic net model, it is really the logistic regression that will help us quantify the effect of the variables.

Thus, we built a logistic model on the data as below.

The variables were picked based on the statistical significance (derived from p-values in previous iterations) and business significance (derived from readings and research).

fitLogistic <- glm(formula = high\_booking\_rate ~ review\_scores\_accuracy+review\_scores\_checkin+review\_scores\_rating+pool+wifi+instant\_bookable+long\_term\_stays\_allowed+refridgerator+jumio+family\_kids\_friendly+coffee\_maker+laptop\_friendly\_workspace+availability\_365 , data = dfLasVegas, family = binomial)  
summary(fitLogistic)

##   
## Call:  
## glm(formula = high\_booking\_rate ~ review\_scores\_accuracy + review\_scores\_checkin +   
## review\_scores\_rating + pool + wifi + instant\_bookable + long\_term\_stays\_allowed +   
## refridgerator + jumio + family\_kids\_friendly + coffee\_maker +   
## laptop\_friendly\_workspace + availability\_365, family = binomial,   
## data = dfLasVegas)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.8965 -0.7116 -0.4571 -0.2190 2.6978   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -8.7420992 1.0033706 -8.713 < 2e-16 \*\*\*  
## review\_scores\_accuracy 0.4123586 0.0931785 4.425 9.62e-06 \*\*\*  
## review\_scores\_checkin 0.6571913 0.0987714 6.654 2.86e-11 \*\*\*  
## review\_scores\_rating -0.0539633 0.0088792 -6.077 1.22e-09 \*\*\*  
## poolTRUE -0.7411933 0.0730092 -10.152 < 2e-16 \*\*\*  
## wifiTRUE 0.7315982 0.2854204 2.563 0.010370 \*   
## instant\_bookableTRUE 0.5058959 0.0693369 7.296 2.96e-13 \*\*\*  
## long\_term\_stays\_allowedTRUE 0.3839917 0.0742223 5.174 2.30e-07 \*\*\*  
## refridgeratorTRUE 0.5948503 0.1268408 4.690 2.74e-06 \*\*\*  
## jumioTRUE 0.2329486 0.0678713 3.432 0.000599 \*\*\*  
## family\_kids\_friendlyTRUE 1.3419588 0.0719024 18.664 < 2e-16 \*\*\*  
## coffee\_makerTRUE 0.3624106 0.1216370 2.979 0.002888 \*\*   
## laptop\_friendly\_workspaceTRUE 0.4596207 0.0870141 5.282 1.28e-07 \*\*\*  
## availability\_365 -0.0005898 0.0002644 -2.231 0.025713 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 6688.3 on 6206 degrees of freedom  
## Residual deviance: 5600.0 on 6193 degrees of freedom  
## AIC: 5628  
##   
## Number of Fisher Scoring iterations: 5

We can see the below coefficients which will help us get the odds ratio and comment on the change in odds for high booking rate based on each variable.

This can be extremely essential when determining the strategy for upgrade of existing properties by the owners.

exp(coef(fitLogistic))

## (Intercept) review\_scores\_accuracy   
## 0.0001597183 1.5103759189   
## review\_scores\_checkin review\_scores\_rating   
## 1.9293656158 0.9474668957   
## poolTRUE wifiTRUE   
## 0.4765449233 2.0783996045   
## instant\_bookableTRUE long\_term\_stays\_allowedTRUE   
## 1.6584706260 1.4681332195   
## refridgeratorTRUE jumioTRUE   
## 1.8127595461 1.2623165487   
## family\_kids\_friendlyTRUE coffee\_makerTRUE   
## 3.8265316213 1.4367887708   
## laptop\_friendly\_workspaceTRUE availability\_365   
## 1.5834733120 0.9994103674

## X. Conclusion and Discussion:

* Though the Hotel industry is big and strong Airbnb could be the way to go for small investments
* Good Reviews and long list of amenities go a long way in making it successful
* Focus on customer service and be family friendly in your offerings
* Lower price for large groups is the best bet

## XI. References:

* Real estate trends in Las Vegas,
* <https://www.brownellteamrealtors.com/market-statistics/> <https://www.fortunebuilders.com/las-vegas-real-estate-market-trends/>
* Tourism Demographics,
* <https://assets.simpleviewcms.com/simpleview/image/upload/v1/clients/lasvegas/2018_Las_Vegas_Visitors_Profile_Study_94443c1d-334f-4d0b-b997-5c8800f990b0.pdf>
* Market Research,
* <https://www.investopedia.com/articles/personal-finance/031815/hotels-vs-airbnb-vegas-visitors.asp> <https://www.investopedia.com/articles/personal-finance/031815/hotels-vs-airbnb-vegas-visitors.asp> <https://www.cntraveler.com/story/why-airbnb-cant-crack-las-vegas> <https://www.fox5vegas.com/news/local/short-term-vacation-rental-market-booming-in-las-vegas-valley-pushing-out-buyers/article_afff7ee4-b8d5-11e9-a333-6f54592625b0.html>