BA with R Project

Airbnb Rental – New York City Dataset

Sharma, Pulkit - pxs180032

Ambike, Ajinkya Shrikant - axa180049

Sowmya, Suresh - sxs180027

Jadhav, Chinmay Narendra - cnj180001

Han, Yanhong - yxh164130

Summary:

In this project, we are analyzing the Airbnb Rental – New York dataset for the month of October, 2017. Our original dataset contains 31 variables and 44,300 records. Our goal is to analyze the data set which will help us to provide recommendations to the host, that would help increase their profit from the rental accommodation.

Key Question:

The key question we are addressing in this project is "predicting the variables which creates an impact to the rooms or rental house, so as to increase the profit gained from the rental accommodations" and "predicting the price of a room or accommodation".

About the Company:

The dataset we are analyzing is a rental dataset owned by the most popular rental company, Airbnb. It is a privately held global company that operates an online marketplace and hospitality service which is accessible via its websites and mobile applications. Members can use the service to arrange or offer lodging, primarily homestays, or tourism experiences. The company does not own any of the real estate listings, nor does it host events. As a broker, it receives commissions from every booking.

Guests can search for lodging using filters such as lodging type, dates, location, and price. Before booking, users must provide personal and payment information.

Hosts provide prices and other details for their rental or event listings. Pricing is determined by the host, with recommendations from Airbnb.

By October 2013, Airbnb had served nine million guests since its founding in August 2008 and in December 2013, the company reported it had over six million new guests and nearly 250,000 properties were added.

With homes and bedrooms available in 34,000 cities and 190 countries, Airbnb operates in more than double the number of territories.

The NewYork.csv file which we downloaded from Kaggle is a file about Airbnb rentals information in the New York city. The information is taken in October 2017 about various Airbnb rentals in the city of New York. The csv file contains information about its amenities, number of bedrooms, the location of the apartment including its longitude and latitude, it's id, whether it's an apartment or a boat, the number of rooms in an apartment and many more attributes. Airbnb is operated through out United states in all the major cities.

Data collection:

Each one of us from the group came up with a dataset. Including the Airbnb dataset, we had five datasets to choose from namely, IMDB dataset, A blood bank dataset, IBM HR dataset and Chicago Transportation permits dataset.

The IMDB dataset and the Chicago Transportation Departments permits dataset were too huge for us to run locally as there were millions of records in the dataset.

The blood bank dataset and the IBM HR dataset had many empty records which reduced our dataset to only a few hundred rows after cleaning it.

Thus, we chose to analyze the Airbnb rental dataset as it had enough number of records and variables to develop regression models and we also found the dataset challenging as it was unclean and the variables were all not in the same simple format.

We collected the dataset from Kaggle and the original csv file had 31 columns and 44300 rows. We wanted to choose a dataset which was pretty good in size with 31 columns which is neither too big nor too small. Airbnb has a good brand value in the market and we felt the dataset very interesting to perform analytics operations and data visualization.

The following is the description of the columns within our csv file

Column Name	**Type**	**Description**
id	Continuous	Id of the host.
host_response_time	Categorical	The time within which the host respond to the customer request.
host_response_rate	Continuous	Indicates the rate at which the hosts respond.
host_is super_host	Categorical	Indicates whether the host is super host or not.
host_has_profile_pic	Categorical	Indicates whether the host has a profile picture on portal.
neighbourhood_cleansed	Categorical	Gives information about cleanliness of the neighborhood.
latitude	Continuous	Gives the latitude of the location.
longitude	Continuous	Gives the longitude of the location.
is_location_exact	Categorical	Indicates whether the Airbnb property location is exact.
property_type	Categorical	Describes the type of the property whether its home or private room.
room_type	Categorical	Describes
accommodates	Continuous	Gives the number of persons it can accommodate
bathrooms	Continuous	Gives the number of bathrooms in each property.
Bedrooms	Continuous	Gives the number of bedrooms in each property.
Beds	Continuous	Gives the number of beds in each room.
Bed type	Categorical	Type of bed included
Amenities	Categorical	The facilities which are included in the property.
Price	Continuous	Gives the price of each property.
Square Feet	Continuous	Size of the rental in terms of square feet.
Guests_included	Continuous	Gives the number of guests allowed per apartment.
Minimum_nights	Continuous	Gives the number of minimum nights a customer should book to avail the rental property.
Maximum nights	Continuous	Gives the number of maximum nights a customer can stay in the property.
Calender_updated	Categorical	Indicates when the calendar (i.e) the availability status of a rental accommodation for a period of time, updated last.

availability_30	Continuous	Indicates the availability of the property through a month.
number_of_reviews	Continuous	Gives the total number of reviews for each property type.
Review_score_ratings	Continuous	Gives the number of reviews on each score rating.
Instant bookable	Categorical	Describes whether the property is available immediately.
Is_business_travel_ready	Categorical	Describes if the property has all the amenities required for business travels.
Cancellation_policy	Categorical	Indicates if we can cancel the booking.
require_guest_profile_picture	Categorical	Indicates whether the profile picture of guest is required or not.
reviews_per_month	Continuous	Describes number of reviews per month.

Dataset Cleaning:

Cleaning our dataset was the most challenging part. As mentioned earlier, all our variables were not in the same format for us to start analysis right away. We analyzed each and every variable for the number of null values it had. We found out that, the variable “square feet” had 43,768 null values in it. We decided to remove that variable from our dataset, as it had no chances of contributing to our analysis with so many blank values in it.

R code for loading the dataset and checking for null values:

```
#Reading the data
newyork.df<-read.csv('New York.csv')
head(newyork.df)
str(newyork.df)
summary(newyork.df)

#To check the number of missing values
sapply(newyork.df, function(x) sum(is.na(x)))
##square_feet variable has 43768 records having NA out of 44317. This shows that the data was not collected/recorded properly for this variable
##Thus, we need to eliminate the square_feet variable.
```

Total number of null values in the data set

```
> sapply(newyork.df, function(x) sum(is.na(x)))
                            id             host_response_time             host_response_rate
                             0                              0                              0
            host_is_superhost          host_has_profile_pic        neighbourhood_cleansed
                             0                              0                              0
                      latitude                      longitude              is_location_exact
                             0                              0                              0
                 property_type                      room_type                   accommodates
                             0                              0                              0
                     bathrooms                       bedrooms                           beds
                           144                             73                             91
                      bed_type                      amenities                    square_feet
                             0                              0                          43768
                         price               guests_included                minimum_nights
                             0                              0                              0
                maximum_nights              calendar_updated                availability_30
                             0                              0                              0
            number_of_reviews          review_scores_rating                instant_bookable
                             0                          10099                              0
      is_business_travel_ready           cancellation_policy require_guest_profile_picture
                             0                              0                              0
              reviews_per_month
                          9474
> |
```

After removing the “square feet” variable, we again checked for null and NA values in the dataset. We found that, there were “N/A” values that were not considered NA by R. The “N/A” values were available in two features, host_response_time and host_response_rate. We converted the “N/A” to NA and tried removing them from our dataset.

R code for removing square feet variable and checking null values again.

```
#Eliminating the square_feet variable
names(newyork.df)
newyork.df<-data.frame(newyork.df[,-18])

#To check the number of missing values
sapply(newyork.df, function(x) sum(is.na(x)))
#Removing the NA values
newyork.df<-na.omit(newyork.df)
sapply(newyork.df, function(x) sum(is.na(x)))
View(newyork.df)
#Even after removing the NA values the data still contains "N/A" values
##which needs to replaced with NA
newyork.df[newyork.df=="N/A"] <- NA
sapply(newyork.df, function(x) sum(is.na(x)))
```

Total Number of Null values after removing square feet column and cleaning the data.

```
> sapply(newyork.df, function(x) sum(is.na(x)))
                            id            host_response_time            host_response_rate
                             0                             0                             0
            host_is_superhost          host_has_profile_pic        neighbourhood_cleansed
                             0                             0                             0
                      latitude                     longitude             is_location_exact
                             0                             0                             0
                 property_type                     room_type                  accommodates
                             0                             0                             0
                     bathrooms                      bedrooms                          beds
                             0                             0                             0
                      bed_type                     amenities                         price
                             0                             0                             0
               guests_included                minimum_nights                maximum_nights
                             0                             0                             0
              calendar_updated               availability_30             number_of_reviews
                             0                             0                             0
         review_scores_rating              instant_bookable      is_business_travel_ready
                             0                             0                             0
          cancellation_policy require_guest_profile_picture             reviews_per_month
                             0                             0                             0
>
```

But then, we were left with only around 26000 records for our analysis. In order to retain the number of records for our analysis. We decided to impute the values for the two variables, host_response_time and host_response_rate, with “Not Specified” and “0” respectively, since the values were missing in only these variables. Ending up with 34000(approx) records after all the cleaning.

R code for imputing NA values and checking null values again.

```
#Imputing the NA values
newyork.df$host_response_time<-as.character(newyork.df$host_response_time)
newyork.df$host_response_time[is.na(newyork.df$host_response_time)]<- "Not specified"
newyork.df$host_response_rate<-as.character(newyork.df$host_response_rate)
newyork.df$host_response_rate[is.na(newyork.df$host_response_rate)]<-"0%"
sapply(newyork.df, function(x) sum(is.na(x)))

##Removing the blank values
newyork.df <- newyork.df[-which(newyork.df$host_response_time == ""), ]
View(newyork.df)
```

A variable named “Amenities” had data in some random format with many data in one cell which hindered us from proceeding with the analysis. Initially, we removed the special characters in the data cell by replacing them with “ ” . Then, we separated those values into individual variables and converted them to dummy variables using the “recast” function.

Amenities column given in the dataset

beds	bed_type	amenities	price	guests_included
1	Real Bed	{"Wireless Internet","Air conditioning",Kitchen,Breakfast,Heati...	50	1
1	Real Bed	{TV,Internet,"Wireless Internet","Air conditioning",Kitchen,"Fre...	125	1
1	Real Bed	{Internet,"Wireless Internet","Air conditioning",Kitchen,"Free ...	69	2
5	Real Bed	{TV,"Air conditioning",Kitchen,"Free parking on premises",He...	150	2
1	Real Bed	{"Cable TV","Wireless Internet","Air conditioning","Free parki...	101	1
1	Real Bed	{TV,"Wireless Internet","Air conditioning",Kitchen,"Free parkin...	125	1
2	Real Bed	{TV,"Cable TV",Internet,"Wireless Internet","Air conditioning",...	115	2
3	Real Bed	{"Wireless Internet","Air conditioning",Kitchen,Breakfast,Heati...	62	4
2	Real Bed	{TV,"Cable TV",Internet,"Wireless Internet","Air conditioning",...	85	2
1	Real Bed	{Internet,"Wireless Internet","Air conditioning",Kitchen,"Free ...	44	1
1	Real Bed	{TV,"Cable TV",Internet,"Wireless Internet","Air conditioning",...	47	1
2	Real Bed	{TV,Internet,"Wireless Internet","Air conditioning",Kitchen,"Fa...	39	1
1	Real Bed	{TV,"Wireless Internet","Air conditioning",Kitchen,"Free parkin...	49	1
1	Real Bed	{TV,"Cable TV",Internet,"Wireless Internet","Air conditioning",...	85	1
1	Real Bed	{TV,"Wireless Internet","Air conditioning",Kitchen,Heating,"Fa...	50	1
1	Real Bed	{TV,"Cable TV",Internet,"Wireless Internet","Air conditioning",...	105	2

```
#Splitting the amenities column into readable information
Amen<-as.character(newyork.df$amenities)
Amen<-gsub("[][!#$%()*.:;\"<=>@^_`|~.{}]", "", Amen)
library(stringr)
StrA<-(str_split_fixed(Amen, ",", 87))
Id<-as.data.frame(newyork.df[,1])
Final<-cbind(Id,StrA)
library(reshape2)
colnames(Final)[1] <- "id"
Amenities<-recast(Final, id ~ value, id.var = "id", length)
```

After this we ended up having around 266 new variables with a lot of redundancy in it. After removing all the redundant data by combining similar variables using the logical OR operator like variables Baby bath, Baby monitor and Babysitter recommendations are all combined to one variable Baby_friendly. Similarly we combined all the other amenities variables and ended up having 26 new variables added to our original dataset. Thus, our dataset now contains 55 variables in total.

```
#Combining the Amenities required for further analysis
newyork.df$'24_hour_checkin'<-as.factor(Amenities$`24-hour check-in`)
newyork.df$'Air_Conditioning'<-as.factor(Amenities$`Air conditioning`)
newyork.df$'Heating'<-as.factor(Amenities$Heating)
newyork.df$'Television'<-as.factor(as.numeric(Amenities$TV|Amenities$`Cable TV`))
newyork.df$'Baby_Friendly'<-as.factor(as.numeric(Amenities$`Baby bath`|Amenities$`Baby monitor`|Amenities$`Childrenâ???Ts dinnerware`
                                        |Amenities$`Childrenâ???Ts books and toys`|Amenities$`Babysitter recommendations`))
newyork.df$'Family/Kid_Friendly'<-as.factor(Amenities$`Family/kid friendly`)
newyork.df$'Coffee_Maker'<-as.factor(Amenities$`Coffee maker`)
newyork.df$'Fire_Extinguisher'<-as.factor(Amenities$`Fire extinguisher`)
newyork.df$'First_Aid_Kit'<-as.factor(Amenities$`First aid kit`)
newyork.df$'Free_Parking'<-as.factor(as.numeric(Amenities$`Free parking on premises`|Amenities$`Free parking on street`))
newyork.df$'Hot_water'<-as.factor(as.numeric(Amenities$`Hot water`|Amenities$`Hot water kettle`))
newyork.df$'Hair_dryer'<-as.factor(Amenities$`Hair dryer`)
newyork.df$'Indoor_Fireplace'<-as.factor(Amenities$`Indoor fireplace`)
newyork.df$'Patio/balcony'<-as.factor(Amenities$`Patio or balcony`)
newyork.df$'private_bathroom'<-as.factor(Amenities$`Private bathroom`)
newyork.df$'Self_checkin'<-as.factor(Amenities$`Self Check-In`)
newyork.df$'Suitable_for_events'<-as.factor(Amenities$`Suitable for events`)
newyork.df$'Breakfast'<-as.factor(Amenities$Breakfast)
newyork.df$'Elevator'<-as.factor(Amenities$Elevator)
newyork.df$'Kitchen'<-as.factor(as.numeric(Amenities$Kitchen|Amenities$Microwave|Amenities$Oven|Amenities$Stove))
newyork.df$'Pillow&Mattress'<-as.factor(as.numeric(Amenities$`Extra pillows and blankets`|Amenities$`Firm mattress`|Amenities$`Bed linens`))
newyork.df$'Laptop_Friendly_Workspace'<-as.factor(as.numeric(Amenities$`Laptop friendly workspace`|Amenities$`Wireless Internet`
                                                |Amenities$`Ethernet connection`|Amenities$Internet|Amenities$`Pocket wifi`))
newyork.df$'Pets_Allowed'<-as.factor(as.numeric(Amenities$`Pets allowed`|Amenities$`Pets live on this property`|Amenities$Cats|Amenities$Dogs|Amenities$`Other pets`))
newyork.df$'Washer/Dryer'<-as.factor(as.numeric(Amenities$Washer|Amenities$`Washer / Dryer`|as.numeric(!Amenities$Dryer)))
newyork.df$'Wheelchair_Accessible'<-as.factor(as.numeric(Amenities$`Accessible-height bed`|Amenities$`Accessible-height toilet`
                                                ||Amenities$`Wheelchair accessible`|Amenities$`Step-free access`))
newyork.df$'Essentials'<-as.factor(as.numeric(Amenities$Essentials|Amenities$Hangers|Amenities$Iron))
```

After splitting the data in Amenities variable into individual variables, we deleted the Amenities variable form our dataset.

```
#removing the amenities column
names(newyork.df)
newyork.df<-data.frame(newyork.df[,-c(17)])
```

Next, we found that the variable neighbourhood_cleansed to be a factor variable, with 217 different levels in it. For us to carry out our analysis, we required proper location names in this column. Thus, we mapped and combined the factors with the location variable in the dataset using the logical OR operator and created a new variable “Location” in the dataset which has 8 levels in it namely, Bronx; Brooklyn; Downtown, Manhattan; Midtown, Manhattan; Queens; Roosevelt Island; Staten Island; Uptown, Manhattan.

After all these cleaning, we are left with 33844 records and 54 variables for our analysis.

```
#The Variable neighbourhood_cleansed has 217 different levels
##Mapping neighbourhoods to 8 different locations
str(newyork.df$neighbourhood_cleansed)
newyork.df$Location[newyork.df$neighbourhood_cleansed == 'Marble Hill'| newyork.df$neighbourhood_cleansed =='Allerton'
                    | newyork.df$neighbourhood_cleansed == 'Baychester'| newyork.df$neighbourhood_cleansed == 'Belmont'| newyork.df$neighbourhood_cleansed =='City Island'
                    | newyork.df$neighbourhood_cleansed =='Claremont Village'| newyork.df$neighbourhood_cleansed == 'Clason Point'
                    |newyork.df$neighbourhood_cleansed =='Co-op City'| newyork.df$neighbourhood_cleansed == 'East Morrisania'| newyork.df$neighbourhood_cleansed == 'Eastchester'
                    |newyork.df$neighbourhood_cleansed == 'Edenwald'
                    | newyork.df$neighbourhood_cleansed =='Fieldston'| newyork.df$neighbourhood_cleansed == 'Fordham'| newyork.df$neighbourhood_cleansed == 'Highbridge'
                    | newyork.df$neighbourhood_cleansed == 'Hunts Point'
                    | newyork.df$neighbourhood_cleansed =='Inwood'| newyork.df$neighbourhood_cleansed == 'Kingsbridge'
                    | newyork.df$neighbourhood_cleansed =='Morris Heights'| newyork.df$neighbourhood_cleansed == 'Morris Park'| newyork.df$neighbourhood_cleansed == 'Morrisania'
                    | newyork.df$neighbourhood_cleansed == 'Mott Haven'| newyork.df$neighbourhood_cleansed == 'Mount Eden'
                    | newyork.df$neighbourhood_cleansed =='Mount Hope'| newyork.df$neighbourhood_cleansed == 'North Riverdale'
                    | newyork.df$neighbourhood_cleansed == 'Norwood, Parkchester'| newyork.df$neighbourhood_cleansed == 'Pelham Bay'
                    | newyork.df$neighbourhood_cleansed =='Pelham Gardens'| newyork.df$neighbourhood_cleansed == 'Port Morris'| newyork.df$neighbourhood_cleansed == 'Riverdale'
                    | newyork.df$neighbourhood_cleansed == 'Schuylerville'| newyork.df$neighbourhood_cleansed == 'Soundview'
                    | newyork.df$neighbourhood_cleansed =='Throgs Neck'| newyork.df$neighbourhood_cleansed == 'Tremont'| newyork.df$neighbourhood_cleansed == 'Unionport'
                    | newyork.df$neighbourhood_cleansed == 'University Heights'| newyork.df$neighbourhood_cleansed == 'Van Nest'
                    | newyork.df$neighbourhood_cleansed =='Wakefield'| newyork.df$neighbourhood_cleansed == 'West Farms'
                    | newyork.df$neighbourhood_cleansed == 'Westchester Square'| newyork.df$neighbourhood_cleansed == 'Williamsbridge'
                    | newyork.df$neighbourhood_cleansed =='Woodlawn'
                    | newyork.df$neighbourhood_cleansed =='Bronxdale'| newyork.df$neighbourhood_cleansed =='Longwood'
                    |newyork.df$neighbourhood_cleansed =='Norwood'|newyork.df$neighbourhood_cleansed == 'Parkchester'] <- 'Bronx, NY'
newyork.df$Location[newyork.df$neighbourhood_cleansed =='Bath Beach'| newyork.df$neighbourhood_cleansed == 'Bay Ridge'
                    | newyork.df$neighbourhood_cleansed =='Bedford-Stuyvesant'| newyork.df$neighbourhood_cleansed =='Bensonhurst'
                    | newyork.df$neighbourhood_cleansed == 'Bergen Beach'
                    | newyork.df$neighbourhood_cleansed =='Boerum Hill'| newyork.df$neighbourhood_cleansed =='Borough Park'
                    | newyork.df$neighbourhood_cleansed =='Brooklyn Heights'| newyork.df$neighbourhood_cleansed =='Brownsville'
                    | newyork.df$neighbourhood_cleansed =='Bushwick'| newyork.df$neighbourhood_cleansed =='Canarsie'| newyork.df$neighbourhood_cleansed == 'Carroll Gardens'
                    | newyork.df$neighbourhood_cleansed =='Clinton Hill'| newyork.df$neighbourhood_cleansed == 'Cobble Hill'| newyork.df$neighbourhood_cleansed =='Columbia St'
                    | newyork.df$neighbourhood_cleansed =='Concord'
                    | newyork.df$neighbourhood_cleansed =='Coney Island'| newyork.df$neighbourhood_cleansed == 'Crown Heights'| newyork.df$neighbourhood_cleansed =='Cypress Hills'
                    | newyork.df$neighbourhood_cleansed =='Downtown Brooklyn'| newyork.df$neighbourhood_cleansed =='Dyker Heights'
                    | newyork.df$neighbourhood_cleansed =='East Flatbush'
                    | newyork.df$neighbourhood_cleansed =='East New York'| newyork.df$neighbourhood_cleansed == 'Flatbush'| newyork.df$neighbourhood_cleansed == 'Flatlands'
                    | newyork.df$neighbourhood_cleansed =='Fort Greene'| newyork.df$neighbourhood_cleansed =='Fort Hamilton'| newyork.df$neighbourhood_cleansed =='Gowanus'
                    | newyork.df$neighbourhood_cleansed =='Gravesend'
                    | newyork.df$neighbourhood_cleansed =='Greenpoint'
                    | newyork.df$neighbourhood_cleansed =='Kensington'| newyork.df$neighbourhood_cleansed =='Manhattan Beach'| newyork.df$neighbourhood_cleansed =='Mill Basin'
                    | newyork.df$neighbourhood_cleansed =='Navy Yard'| newyork.df$neighbourhood_cleansed =='Park Slope'| newyork.df$neighbourhood_cleansed =='Prospect Heights'
                    | newyork.df$neighbourhood_cleansed =='Prospect-Lefferts Gardens'| newyork.df$neighbourhood_cleansed =='Red Hook'
                    | newyork.df$neighbourhood_cleansed =='Sea Gate'
                    | newyork.df$neighbourhood_cleansed =='Sheepshead Bay'| newyork.df$neighbourhood_cleansed =='South Slope'
                    | newyork.df$neighbourhood_cleansed == 'Spuyten Duyvil'
                    | newyork.df$neighbourhood_cleansed =='Sunset Park'| newyork.df$neighbourhood_cleansed == 'Vinegar Hill'| newyork.df$neighbourhood_cleansed == 'Williamsburg'
                    | newyork.df$neighbourhood_cleansed == 'Windsor Terrace'
                    | newyork.df$neighbourhood_cleansed == 'DUMBO'|newyork.df$neighbourhood_cleansed=='Midwood'] <- 'Brooklyn, NY'
newyork.df$Location[newyork.df$neighbourhood_cleansed =='Civic Center'| newyork.df$neighbourhood_cleansed =='NoHo'| newyork.df$neighbourhood_cleansed =='Nolita'
                    | newyork.df$neighbourhood_cleansed =='Lower East Side'| newyork.df$neighbourhood_cleansed =='Tribeca'| newyork.df$neighbourhood_cleansed =='Two Bridges'
                    | newyork.df$neighbourhood_cleansed =='Chinatown'
                    | newyork.df$neighbourhood_cleansed =='East Village'| newyork.df$neighbourhood_cleansed =='Financial District'
                    | newyork.df$neighbourhood_cleansed =='Greenwich Village'
                    | newyork.df$neighbourhood_cleansed =='Little Italy'| newyork.df$neighbourhood_cleansed == 'SoHo'| newyork.df$neighbourhood_cleansed =='Stuyvesant Town'
                    | newyork.df$neighbourhood_cleansed == 'West Village'
                    | newyork.df$neighbourhood_cleansed =='Battery Park City'] <- 'Downtown, Manhattan, NY'
newyork.df$Location[newyork.df$neighbourhood_cleansed =='East Harlem'| newyork.df$neighbourhood_cleansed =='Harlem'
                    | newyork.df$neighbourhood_cleansed =='Washington Heights'] <- 'Uptown, Manhattan, NY'
newyork.df$Location[newyork.df$neighbourhood_cleansed =="Upper East Side"| newyork.df$neighbourhood_cleansed == "Midtown"
                    | newyork.df$neighbourhood_cleansed == "Oakwood"| newyork.df$neighbourhood_cleansed == "Chelsea"
                    | newyork.df$neighbourhood_cleansed == "Concourse"| newyork.df$neighbourhood_cleansed == "Concourse Village"
                    | newyork.df$neighbourhood_cleansed == "Flatiron District"| newyork.df$neighbourhood_cleansed == "Hell's Kitchen"
                    | newyork.df$neighbourhood_cleansed == "Kips Bay"| newyork.df$neighbourhood_cleansed == "Morningside Heights"
                    | newyork.df$neighbourhood_cleansed == "Murray Hill"| newyork.df$neighbourhood_cleansed == "Theater District"
                    | newyork.df$neighbourhood_cleansed == "Gramercy"|newyork.df$neighbourhood_cleansed=='Melrose'
                    |newyork.df$neighbourhood_cleansed=='Upper West Side'] <- 'Midtown, Manhattan, NY'
newyork.df$Location[newyork.df$neighbourhood_cleansed =='Arverne'| newyork.df$neighbourhood_cleansed == 'Astoria'| newyork.df$neighbourhood_cleansed =='Bay Terrace'
                    | newyork.df$neighbourhood_cleansed =='Bayside'| newyork.df$neighbourhood_cleansed =='Bayswater'| newyork.df$neighbourhood_cleansed =='Briarwood'
                    | newyork.df$neighbourhood_cleansed =='Brighton Beach'| newyork.df$neighbourhood_cleansed == 'Cambria Heights'
                    | newyork.df$neighbourhood_cleansed =='College Point'
                    | newyork.df$neighbourhood_cleansed =='Corona'| newyork.df$neighbourhood_cleansed == 'Ditmars Steinway'| newyork.df$neighbourhood_cleansed =='Douglaston'
                    | newyork.df$neighbourhood_cleansed =='East Elmhurst'| newyork.df$neighbourhood_cleansed =='Edgemere'| newyork.df$neighbourhood_cleansed == "Elmhurst"
                    | newyork.df$neighbourhood_cleansed =='Far Rockaway'| newyork.df$neighbourhood_cleansed == 'Flushing'| newyork.df$neighbourhood_cleansed == 'Forest Hills'
                    | newyork.df$neighbourhood_cleansed =='Fresh Meadows'| newyork.df$neighbourhood_cleansed == 'Glen Oaks'| newyork.df$neighbourhood_cleansed =='Glendale'
                    | newyork.df$neighbourhood_cleansed =='Hollis'| newyork.df$neighbourhood_cleansed == 'Hollis Hills'| newyork.df$neighbourhood_cleansed =='Holliswood'
                    | newyork.df$neighbourhood_cleansed =='Howard Beach'| newyork.df$neighbourhood_cleansed == 'Jackson Heights'| newyork.df$neighbourhood_cleansed == 'Jamaica'
```

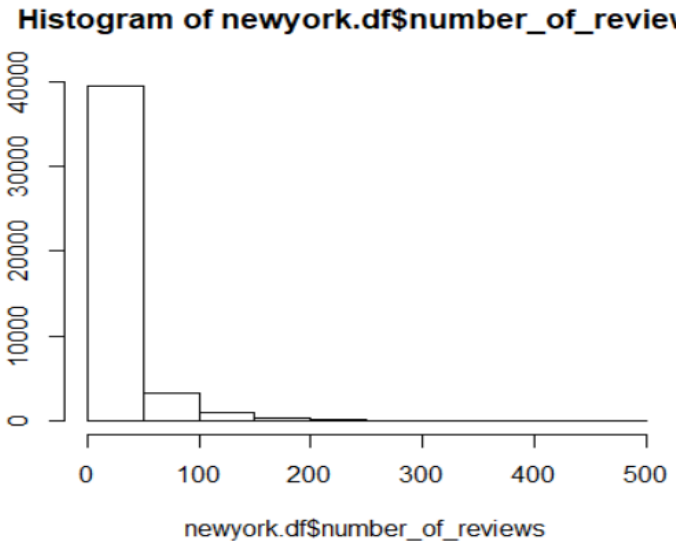
```
                    | newyork.df$neighbourhood_cleansed =='Howard Beach'| newyork.df$neighbourhood_cleansed == 'Jackson Heights'| newyork.df$neighbourhood_cleansed == 'Jamaica'
                    | newyork.df$neighbourhood_cleansed =='Jamaica Estates'| newyork.df$neighbourhood_cleansed == 'Jamaica Hills'
                    | newyork.df$neighbourhood_cleansed =='Kew Gardens'
                    | newyork.df$neighbourhood_cleansed =='Kew Gardens Hills'| newyork.df$neighbourhood_cleansed =='Laurelton'| newyork.df$neighbourhood_cleansed =='Little Neck'
                    | newyork.df$neighbourhood_cleansed =='Long Island City'
                    | newyork.df$neighbourhood_cleansed =='Maspeth'| newyork.df$neighbourhood_cleansed == 'Middle Village'| newyork.df$neighbourhood_cleansed =='Neponsit'
                    | newyork.df$neighbourhood_cleansed =='Ozone Park'| newyork.df$neighbourhood_cleansed == 'Queens Village'| newyork.df$neighbourhood_cleansed == 'Rego Park'
                    | newyork.df$neighbourhood_cleansed =='Richmond Hill'
                    | newyork.df$neighbourhood_cleansed =='Ridgewood'| newyork.df$neighbourhood_cleansed == 'Rockaway Beach'| newyork.df$neighbourhood_cleansed =='Rosedale'
                    | newyork.df$neighbourhood_cleansed =='South Ozone Park'| newyork.df$neighbourhood_cleansed == 'Springfield Gardens'
                    | newyork.df$neighbourhood_cleansed == 'St. Albans'
                    | newyork.df$neighbourhood_cleansed =='Sunnyside'| newyork.df$neighbourhood_cleansed =='Whitestone'| newyork.df$neighbourhood_cleansed =='Woodhaven'
                    | newyork.df$neighbourhood_cleansed == 'Woodside'
                    | newyork.df$neighbourhood_cleansed =='Belle Harbor'| newyork.df$neighbourhood_cleansed == 'Bellerose'] <- 'Queens, NY'
newyork.df$Location[newyork.df$neighbourhood_cleansed =='Clifton'| newyork.df$neighbourhood_cleansed =='Dongan Hills'| newyork.df$neighbourhood_cleansed =='Lighthouse Hill'
                    | newyork.df$neighbourhood_cleansed =='Arden Heights'| newyork.df$neighbourhood_cleansed =='Arrochar'| newyork.df$neighbourhood_cleansed =='Castleton Corners'
                    | newyork.df$neighbourhood_cleansed =='Eltingville'| newyork.df$neighbourhood_cleansed =='Emerson Hill'| newyork.df$neighbourhood_cleansed =='Graniteville'
                    | newyork.df$neighbourhood_cleansed =='Grant City'
                    | newyork.df$neighbourhood_cleansed =='Great Kills'| newyork.df$neighbourhood_cleansed =='Grymes Hill'| newyork.df$neighbourhood_cleansed =='Mariners Harbor'
                    | newyork.df$neighbourhood_cleansed =='Midland Beach'| newyork.df$neighbourhood_cleansed == 'New Brighton'
                    | newyork.df$neighbourhood_cleansed == 'New Springville'
                    | newyork.df$neighbourhood_cleansed =='Port Richmond'| newyork.df$neighbourhood_cleansed == 'Randall Manor'
                    | newyork.df$neighbourhood_cleansed == 'Richmondtown'
                    | newyork.df$neighbourhood_cleansed =='Shore Acres'| newyork.df$neighbourhood_cleansed =='Silver Lake'| newyork.df$neighbourhood_cleansed =='St. George'
                    | newyork.df$neighbourhood_cleansed =='Stapleton'| newyork.df$neighbourhood_cleansed == 'Todt Hill'| newyork.df$neighbourhood_cleansed =='Tompkinsville'
                    | newyork.df$neighbourhood_cleansed =='Tottenville'| newyork.df$neighbourhood_cleansed == 'West Brighton'| newyork.df$neighbourhood_cleansed == 'Castle Hill'
                    | newyork.df$neighbourhood_cleansed == 'Howland Hook'
                    | newyork.df$neighbourhood_cleansed =='Huguenot'| newyork.df$neighbourhood_cleansed == 'Rosebank'| newyork.df$neighbourhood_cleansed == 'Woodrow'
                    | newyork.df$neighbourhood_cleansed == 'South Beach'] <- 'Staten Island'
newyork.df$Location[newyork.df$neighbourhood_cleansed =='Roosevelt Island'] <- 'Roosevelt Island, NY'
newyork.df$Location <- as.factor(newyork.df$Location)
str(newyork.df)
#Now, the datset has one more column named "Location" with 8 levels
levels(newyork.df$Location)
```

For our analysis, we converted the variable, “review_score_rating” to a range of 1 to 5 from a range of 0 to 100 and created a new variable “rating”.

```
#Converting the review_scores_rating on a scale of 1 to 5
newyork.df$rating[newyork.df$review_scores_rating >= 0 & newyork.df$review_scores_rating <= 10] <- '1'
newyork.df$rating[newyork.df$review_scores_rating > 10 & newyork.df$review_scores_rating <= 35] <- '2'
newyork.df$rating[newyork.df$review_scores_rating > 35 & newyork.df$review_scores_rating <= 60] <- '3'
newyork.df$rating[newyork.df$review_scores_rating > 60 & newyork.df$review_scores_rating <= 90] <- '4'
newyork.df$rating[newyork.df$review_scores_rating > 90 & newyork.df$review_scores_rating <= 100] <- '5'

newyork.df$rating <- as.factor(newyork.df$rating)
str(newyork.df)
#removing the review_scores_rating column
names(newyork.df)
newyork.df<-data.frame(newyork.df[,-c(6,24)])
```

Exploratory Data Analysis:

Our goal is to determine the success variables in the dataset and provide suggestions and recommendations to the host so that he can improve his profit. We do not have a direct profit variable in the dataset so there was a need for a variable which could judge the success of our model. We decided not to go with success based on ratings model because the number of reviews were not evenly distributed, ie many had very few number of views which effects our analysis. So we decided to go with the success based on the revenue model.



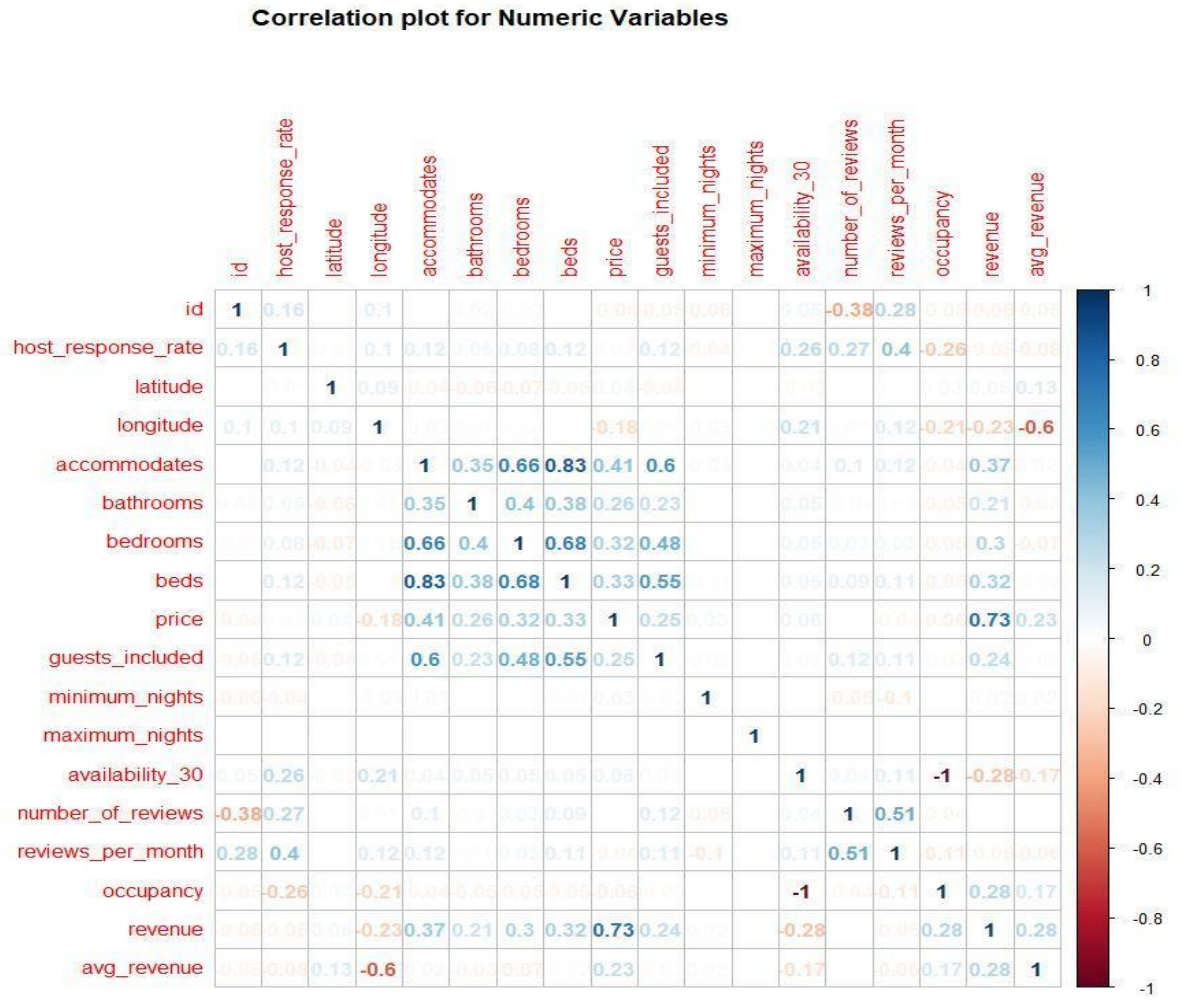

```
#Calculating the occupancy and the revenue generated
oc <- function(a){30-a}
newyork.df$occupancy <- oc(newyork.df$availability_30)
mul <- function(a,b){a*b}
newyork.df$revenue <- mul(newyork.df$price, newyork.df$occupancy)
success<-as.integer(ifelse(newyork.df$success_based_on_revenue=="0",0,1))
```

We then calculated the average value of the revenue variable with respect to its location. Depending on this average value of that particular location, we classify our accommodation as successful or not as "success_based_on_revenue" in our dataset.

```
#Calculating the Success variable based on the revenue generated
agg<-data.frame(aggregate(newyork.df$revenue, by = list(newyork.df$Location), FUN = mean))

newyork.df$avg_revenue[newyork.df$Location == 'Bronx, NY'] <- agg[1,2]
newyork.df$avg_revenue[newyork.df$Location == 'Brooklyn, NY'] <- agg[2,2]
newyork.df$avg_revenue[newyork.df$Location == 'Downtown, Manhattan, NY'] <- agg[3,2]
newyork.df$avg_revenue[newyork.df$Location == 'Midtown, Manhattan, NY'] <- agg[4,2]
newyork.df$avg_revenue[newyork.df$Location == 'Queens, NY'] <- agg[5,2]
newyork.df$avg_revenue[newyork.df$Location == 'Roosevelt Island, NY'] <- agg[6,2]
newyork.df$avg_revenue[newyork.df$Location == 'Staten Island'] <- agg[7,2]
newyork.df$avg_revenue[newyork.df$Location == 'Uptown, Manhattan, NY'] <- agg[8,2]

newyork.df$success_based_on_revenue <- ifelse(newyork.df$revenue >= newyork.df$avg_revenue, 1,0)
table(newyork.df$success_based_on_revenue)
newyork.df$success_based_on_revenue <- as.factor(newyork.df$success_based_on_revenue)

str(newyork.df)
```

Code for correlation matrix for all the variables:

```
numeric.var <- sapply(newyork.df, is.numeric)
corr.matrix <- cor(newyork.df[,numeric.var])
corrplot(corr.matrix, main = '\n\n Correlation plot for Numeric Variables',
         method = "number")
```



From the correlation matrix we find that we find the maximum correlation between the following variables:

- Beds and accommodates.
- Bedroom and accommodates
- Guests_included and accommodates
- Beds and bedroom.
- Guests_included and beds
- Revenue and price
- Revenues_per_month and number_of_reviews.

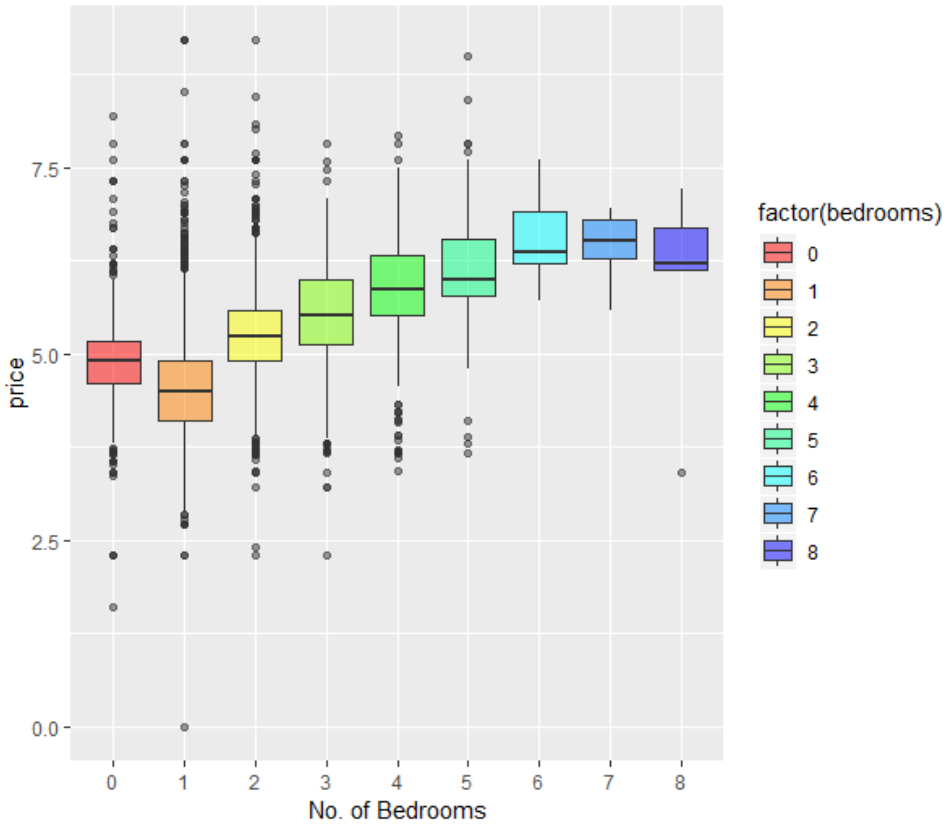
- Avg_revenue and longitude (negatively corelated)

As the number of accommodates increases the number of beds increases and accordingly it is interpreted for other variables.

Since we can see that the average and longitude are negatively corelated, as the longitude increases the average revenue decreases.

Box plot for No of Bedrooms and Price:

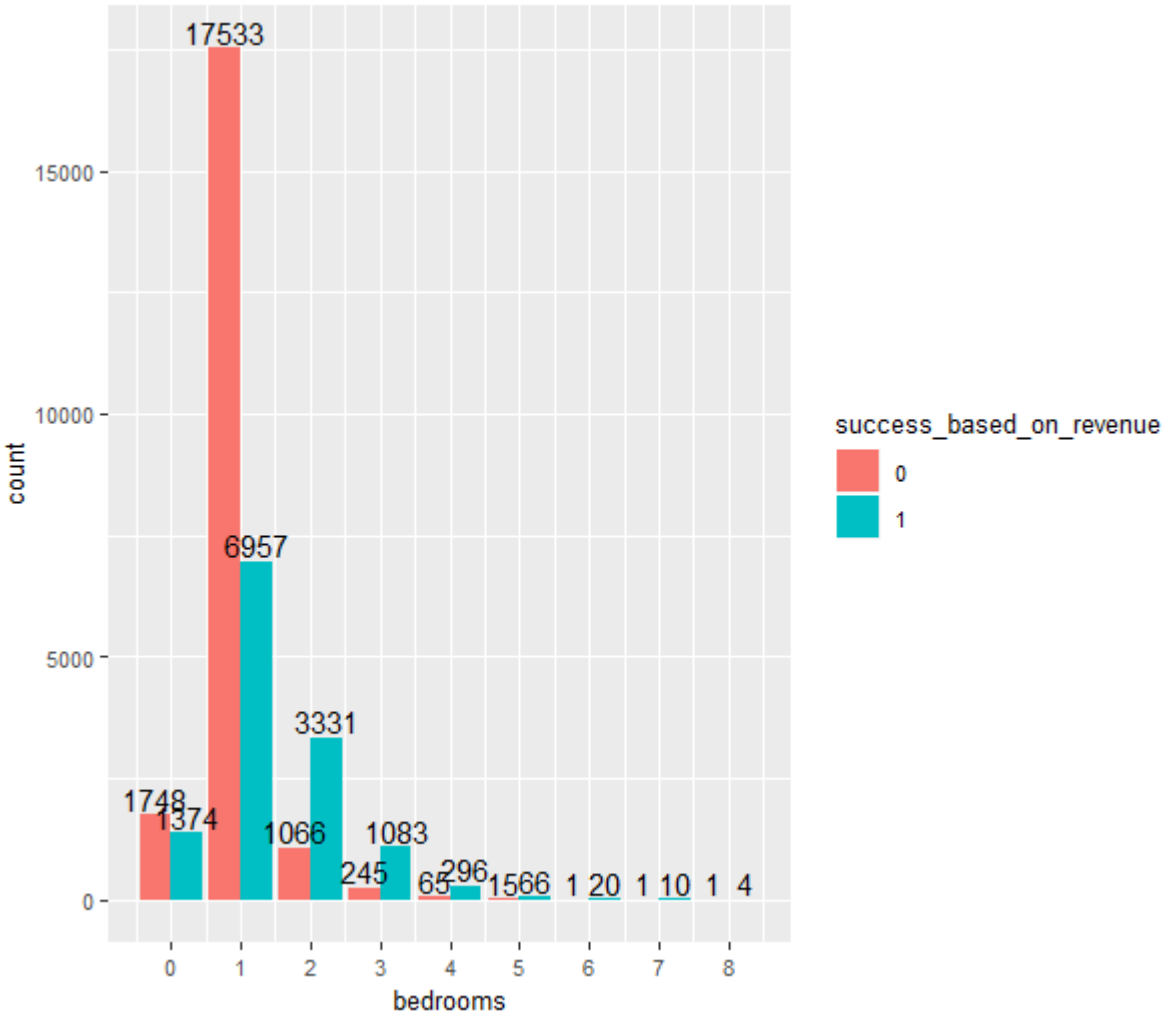
```
ggplot(newyork.df, aes(x = factor(bedrooms) , y = log(price), fill = factor(bedrooms))) +
  geom_boxplot(alpha=0.5) +
  labs(x = "No. of Bedrooms", y = "price")+
  scale_fill_manual(values= rainbow(12))
```



From the above box plot we can interpret that as we increase the number of bedrooms the price also increases. We can see that the median of the box plots keeps increasing till the number of bedrooms become 7 and then the median decreases drastically. So, we can infer that till 7 bedrooms the price goes on increasing and then it drops.

Plot for relationship between No of Bedrooms and Success:

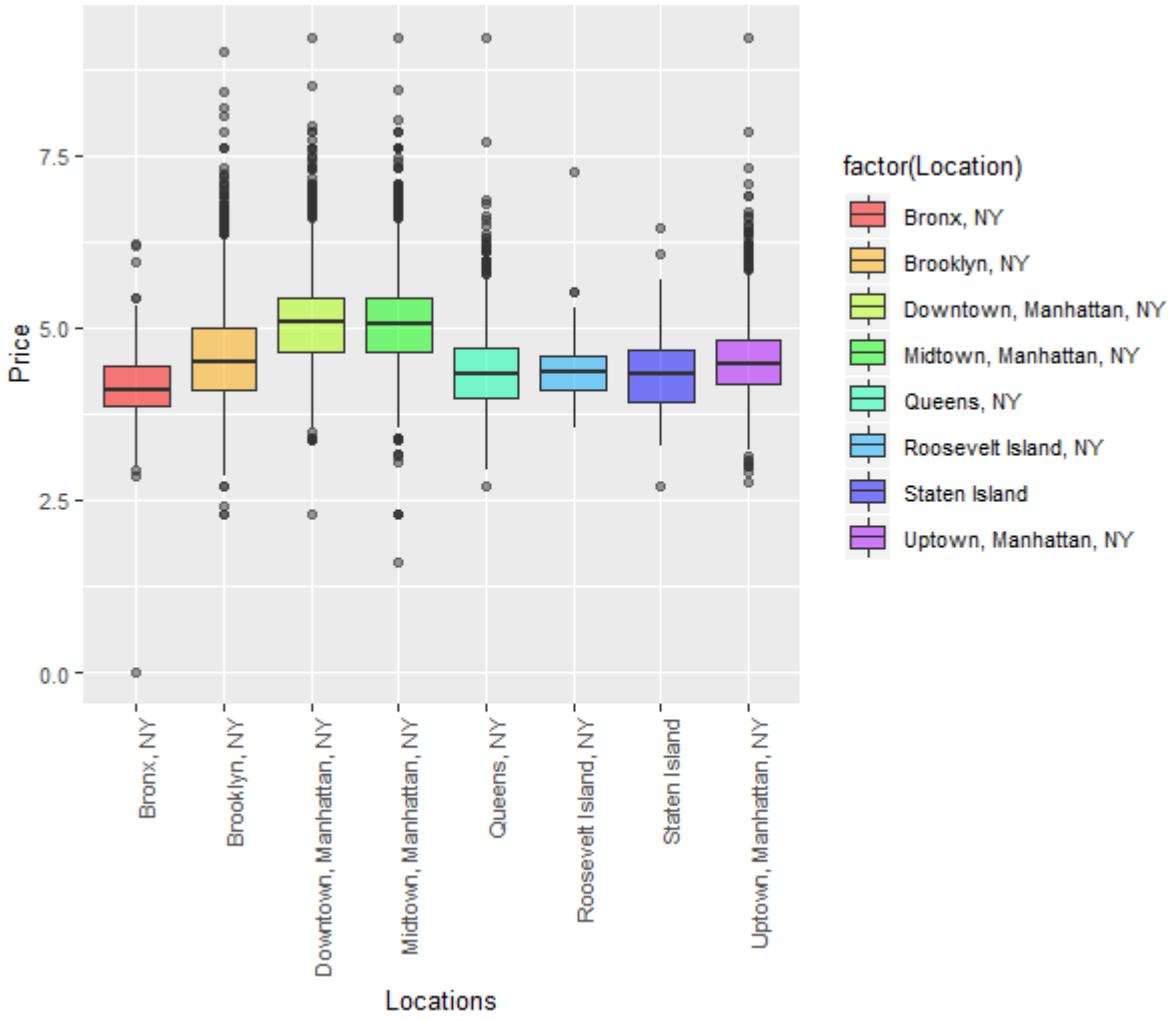
```
ggplot(newyork.df, aes(x=bedrooms,fill=success_based_on_revenue)) +
  geom_bar(position = "dodge") + scale_x_continuous(breaks=0:14) +
  geom_text(aes(label=..count..),stat="count",position=position_dodge(0.9),vjust=-0.2)+
  theme(text = element_text(size=10))
```



From this bar plot we can infer that for an accommodation with one bedroom, the frequency is highest. But the number of success is less than the failure. So, for 2 bedrooms and 3 bedrooms, though people prefer it lesser number of times, the success rate is higher in both these cases as compared to the success rate of single bedroom accommodation.

Plot for Locations and Price:

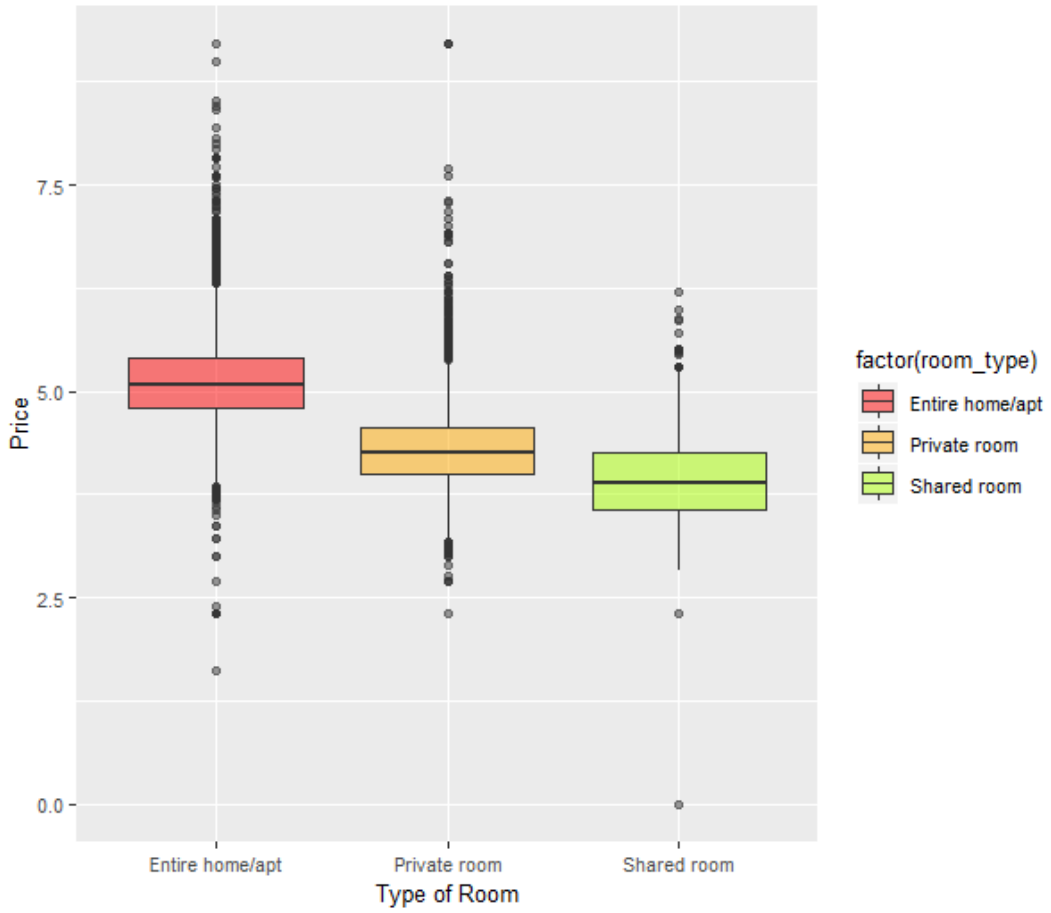
```
ggplot(newyork.df, aes(x = factor(Location) , y = log(price), fill = factor(Location))) +
  geom_boxplot(alpha=0.5) +
  labs(x = "Locations", y = "Price")+
  scale_fill_manual(values= rainbow(9))+
  theme(text = element_text(size=10),axis.text.x = element_text(angle=90, hjust=1))
```



From the plot above, we can infer that, the locations Downtown Manhattan and Midtown Manhattan has the highest price, as their median is higher when compared with other locations.

Boxplot for Type of Room and Price:

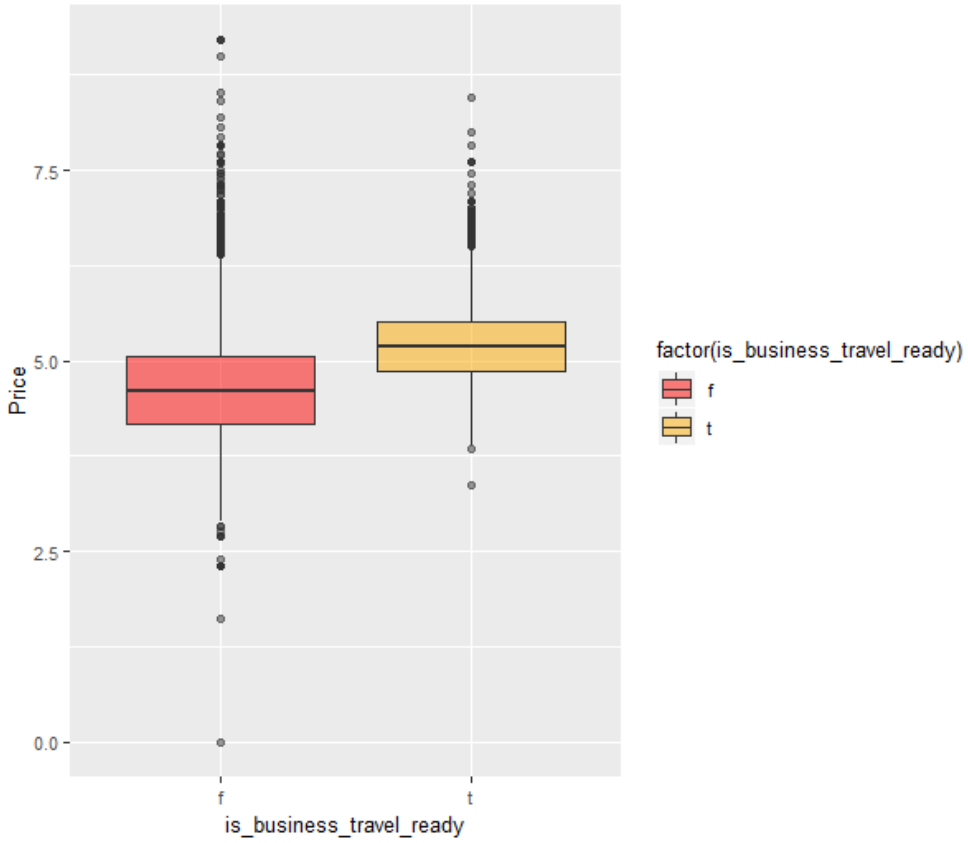
```
ggplot(newyork.df, aes(x = factor(room_type) , y = log(price), fill = factor(room_type))) +
  geom_boxplot(alpha=0.5) +
  labs(x = "Type of Room", y = "Price")+
  scale_fill_manual(values= rainbow(9))+
  theme(text = element_text(size=10))
```



From the Boxplot we can infer that the price of the Entire home/apartment is more than the price of a private room or shared room.

Boxplot for is_business_travel_ready and Price:

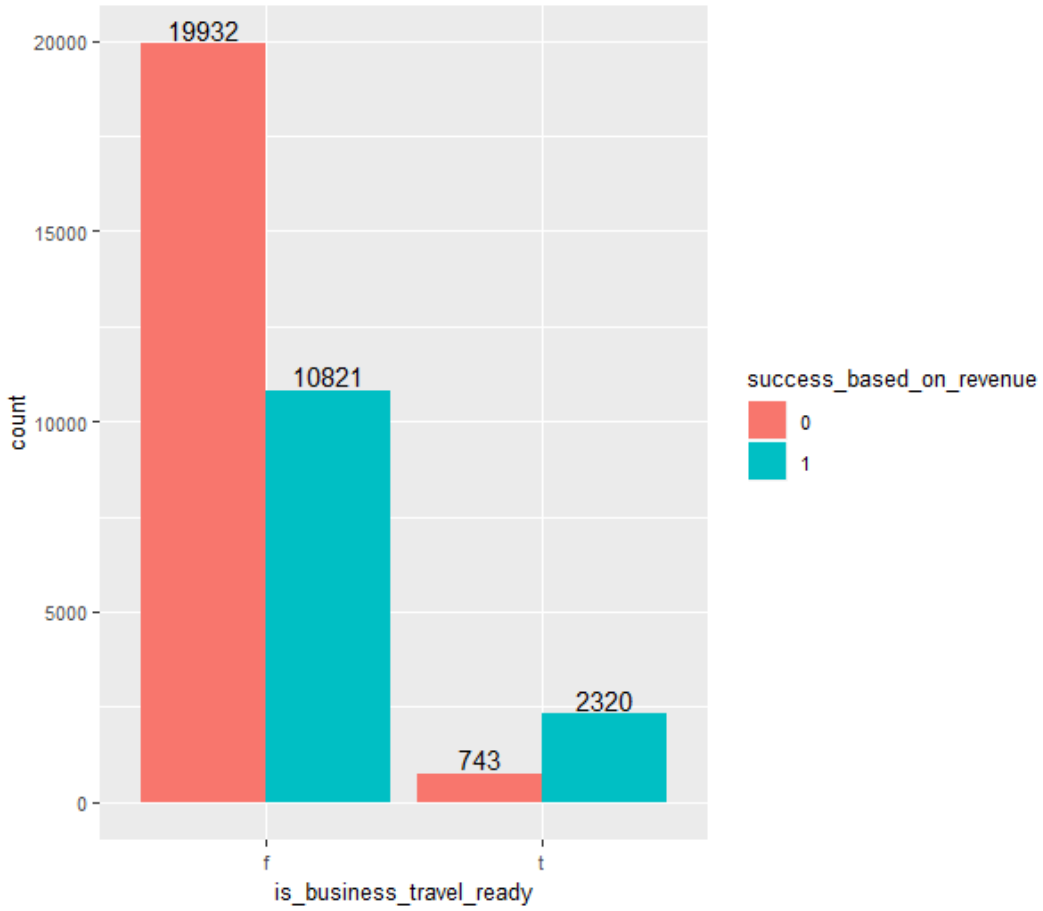
```
ggplot(newyork.df, aes(x = factor(is_business_travel_ready) , y = log(price),
                       fill = factor(is_business_travel_ready))) +
  geom_boxplot(alpha=0.5) +
  labs(x = "is_business_travel_ready", y = "Price")+
  scale_fill_manual(values= rainbow(9))+
  theme(text = element_text(size=10))
```



The accommodations which are business_travel_ready have higher price than the properties which are not.

Plot for is_business_travel_ready and count:

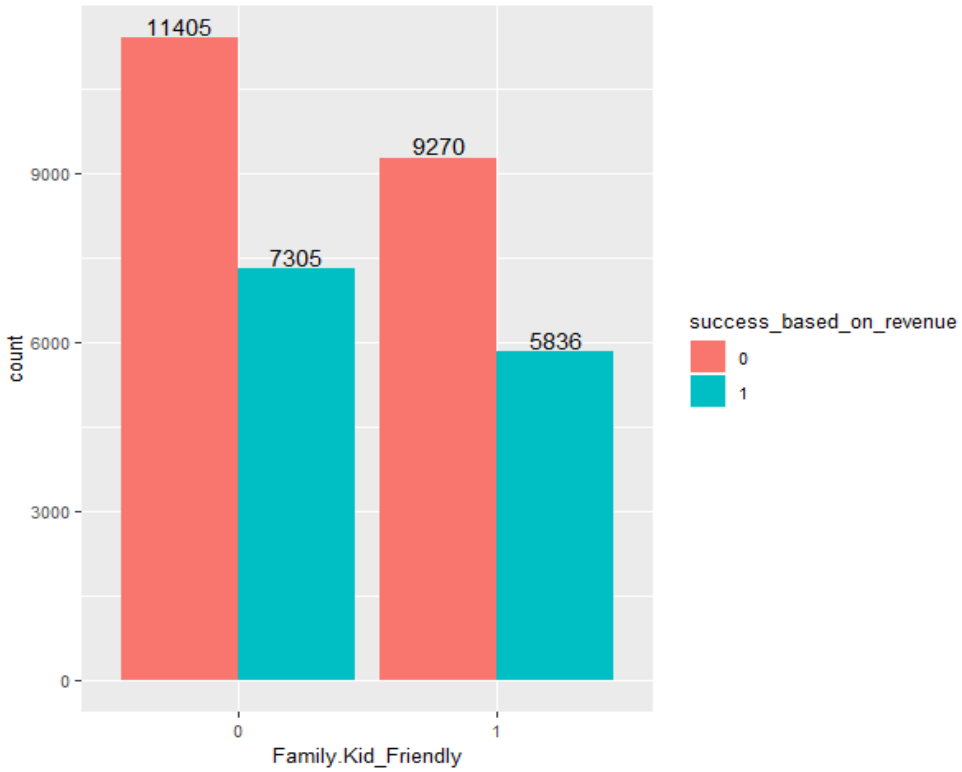
```
ggplot(newyork.df, aes(x=is_business_travel_ready,fill=success_based_on_revenue)) +
  geom_bar(position = "dodge") +
  geom_text(aes(label=..count..),stat="count",position=position_dodge(0.9),vjust=-0.2)+
  theme(text = element_text(size=10))
```



From the bar plot, we find that, properties which are not business ready are occupied frequently, but their success rate is lesser than the failure rate. Those properties which are business ready have higher success rate despite less occupancy comparatively.

Plot for Family_Kid_friendly and Count:

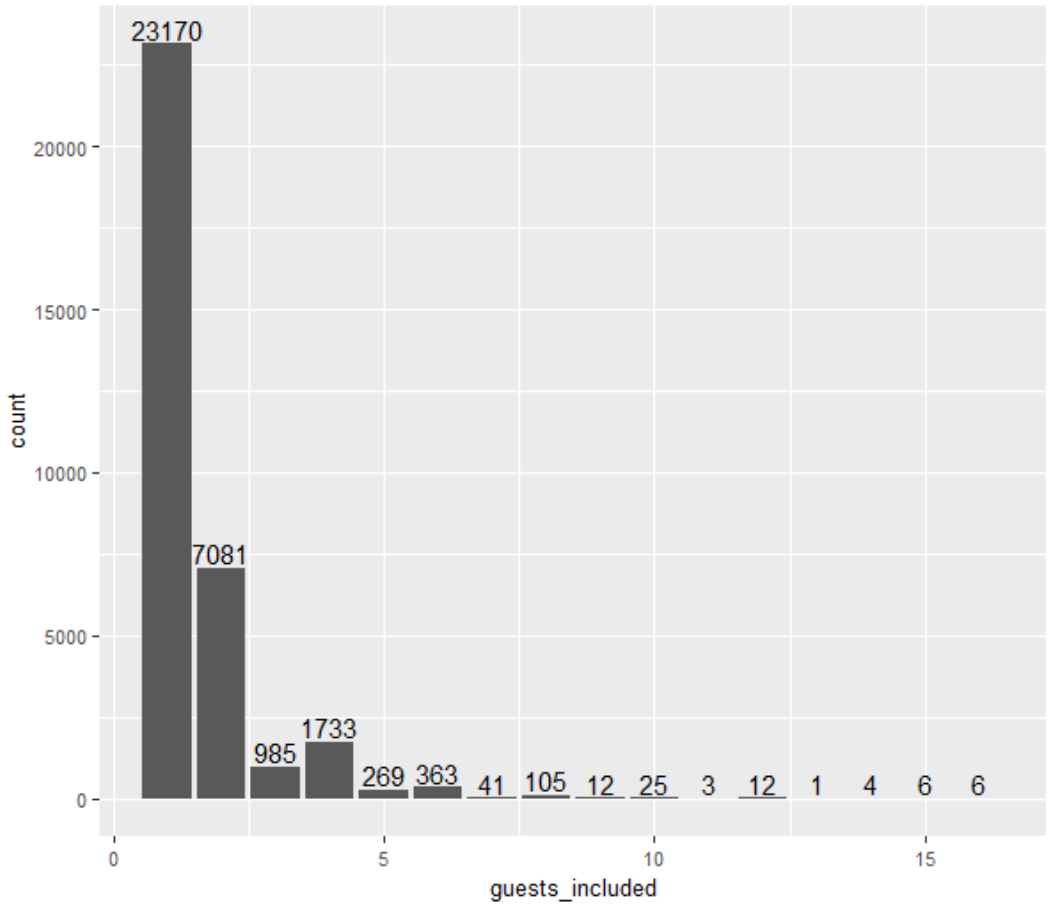
```
ggplot(newyork.df, aes(x=Family.Kid_Friendly,fill=success_based_on_revenue)) +
  geom_bar(position = "dodge") +
  geom_text(aes(label=..count..),stat="count",position=position_dodge(0.9),vjust=-0.2)+
  theme(text = element_text(size=10))
```



We find from this plot that the properties which are not kid friendly are occupied at a higher rate but the revenue generated from these properties are lesser in comparison with properties that are not kids friendly despite lower occupancy rate.

Plot for guests_included and count

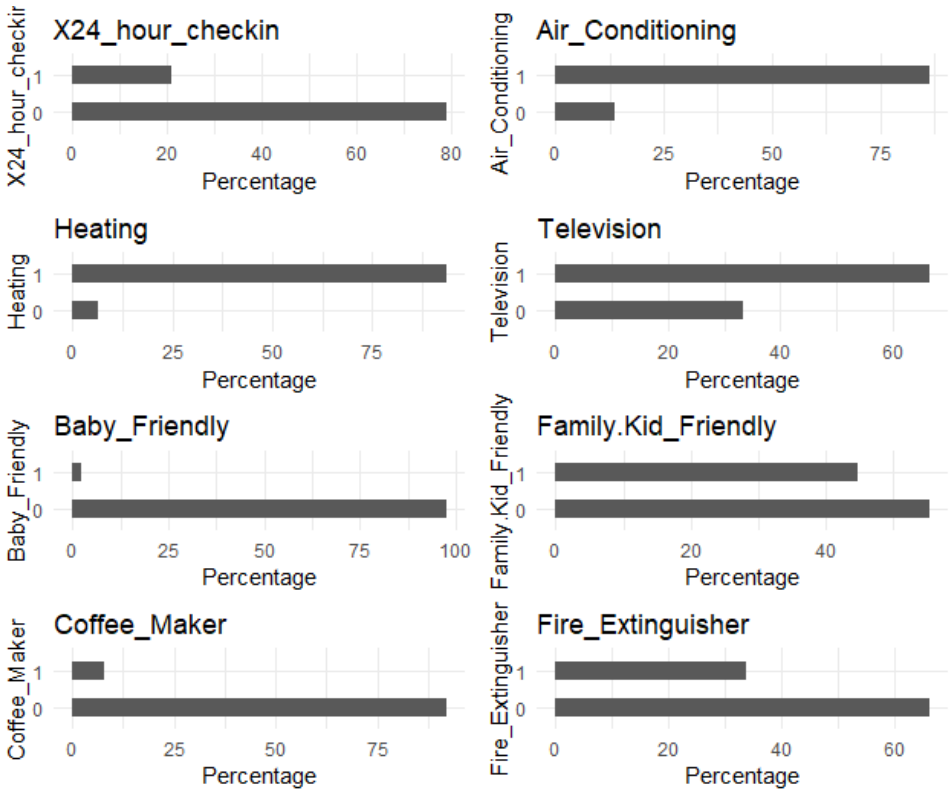
```
ggplot(newyork.df, aes(x=guests_included)) +
     geom_bar(position = "dodge") +
     geom_text(aes(label=..count..),stat="count",position=position_dodge(0.9),vjust=-0.2)+
     theme(text = element_text(size=10))
```



In this plot we find that the bar plot is right skewed. We can infer that the maximum number of guests often ranges between 1 and 5 in New York.

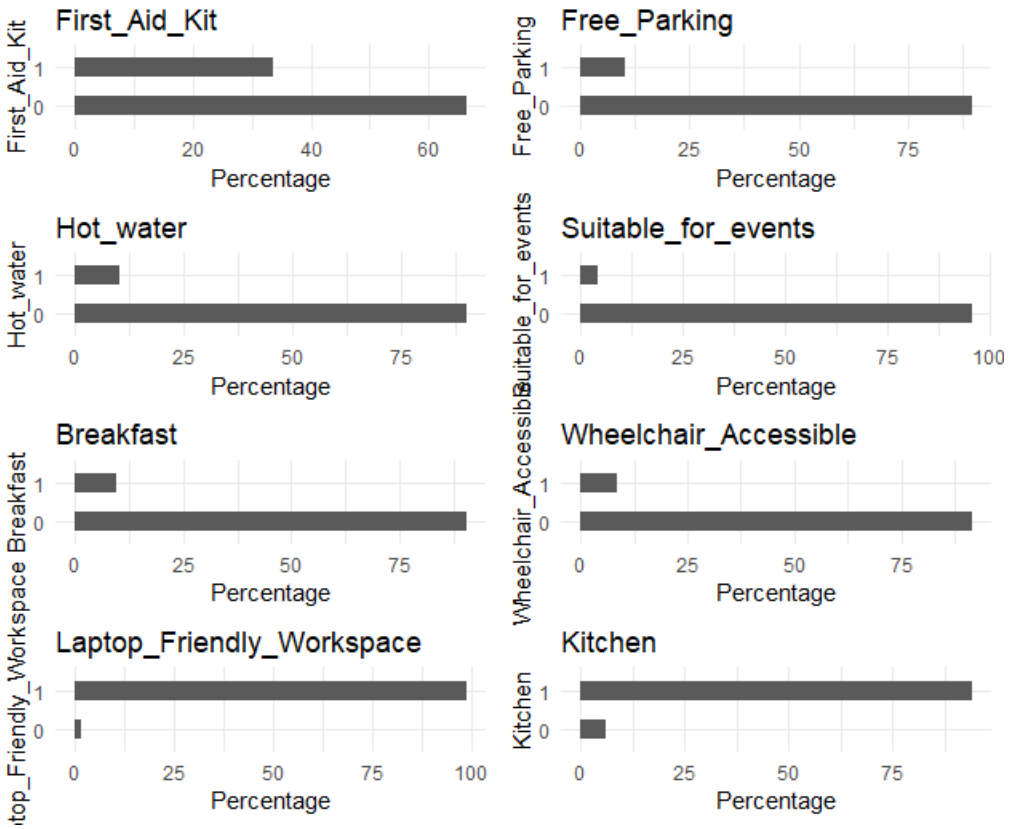
Analysis on the availability of amenities in the accommodation:

```
p1 <- ggplot(newyork.df, aes(x=X24_hour_checkin)) + ggtitle("X24_hour_checkin") + xlab("X24_hour_checkin") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) +
  ylab("Percentage") + coord_flip() + theme_minimal()
p2 <- ggplot(newyork.df, aes(x=Air_Conditioning)) + ggtitle("Air_Conditioning") + xlab("Air_Conditioning") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p3 <- ggplot(newyork.df, aes(x=Heating)) + ggtitle("Heating") + xlab("Heating") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p4 <- ggplot(newyork.df, aes(x=Television)) + ggtitle("Television") + xlab("Television") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p5 <- ggplot(newyork.df, aes(x=Baby_Friendly)) + ggtitle("Baby_Friendly") + xlab("Baby_Friendly") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p6 <- ggplot(newyork.df, aes(x=Family.Kid_Friendly)) + ggtitle("Family.Kid_Friendly") + xlab("Family.Kid_Friendly") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p7 <- ggplot(newyork.df, aes(x=Coffee_Maker)) + ggtitle("Coffee_Maker") + xlab("Coffee_Maker") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p8 <- ggplot(newyork.df, aes(x=Fire_Extinguisher)) + ggtitle("Fire_Extinguisher") + xlab("Fire_Extinguisher") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()

grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8,ncol=2)
```



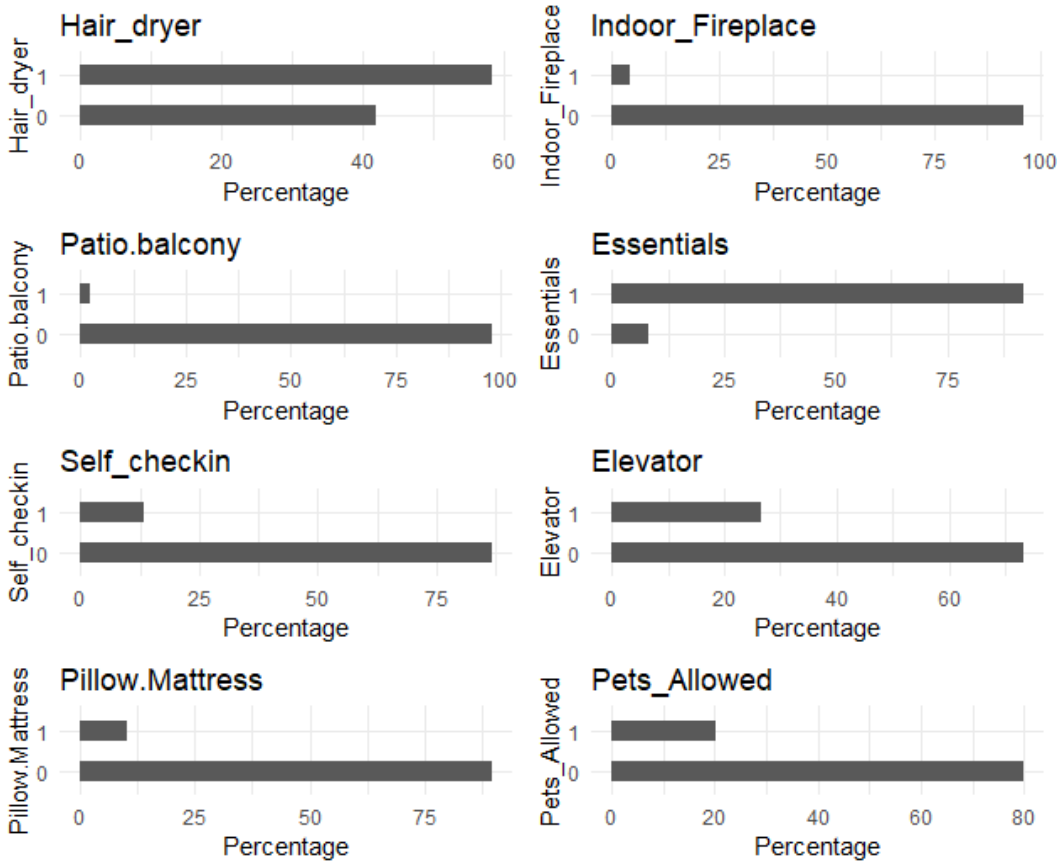
We have plotted grid of the amenities and 1 represents the availability of that amenity in that property and 0 represents the unavailability. For example for X24_hour_checkin we find from the grid that 20% of the property has 24 hour checking and almost 80% of the property does not have 24 hour checking. Similar interpretation can be given to all the remaining amenities.

```
p9 <- ggplot(newyork.df, aes(x=First_Aid_Kit)) + ggtitle("First_Aid_Kit") + xlab("First_Aid_Kit") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p10 <- ggplot(newyork.df, aes(x=Free_Parking)) + ggtitle("Free_Parking") + xlab("Free_Parking") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p11 <- ggplot(newyork.df, aes(x=Hot_water)) + ggtitle("Hot_water") + xlab("Hot_water") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p12 <- ggplot(newyork.df, aes(x=Suitable_for_events)) + ggtitle("Suitable_for_events") + xlab("Suitable_for_events") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p13 <- ggplot(newyork.df, aes(x=Breakfast)) + ggtitle("Breakfast") + xlab("Breakfast") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p14 <- ggplot(newyork.df, aes(x=Wheelchair_Accessible)) + ggtitle("Wheelchair_Accessible") + xlab("Wheelchair_Accessible") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p15 <- ggplot(newyork.df, aes(x=Laptop_Friendly_Workspace)) + ggtitle("Laptop_Friendly_Workspace") + xlab("Laptop_Friendly_Workspace") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p16 <- ggplot(newyork.df, aes(x=Kitchen)) + ggtitle("Kitchen") + xlab("Kitchen") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
grid.arrange(p9, p10, p11, p12, p13, p14, p15, p16, ncol=2)
```



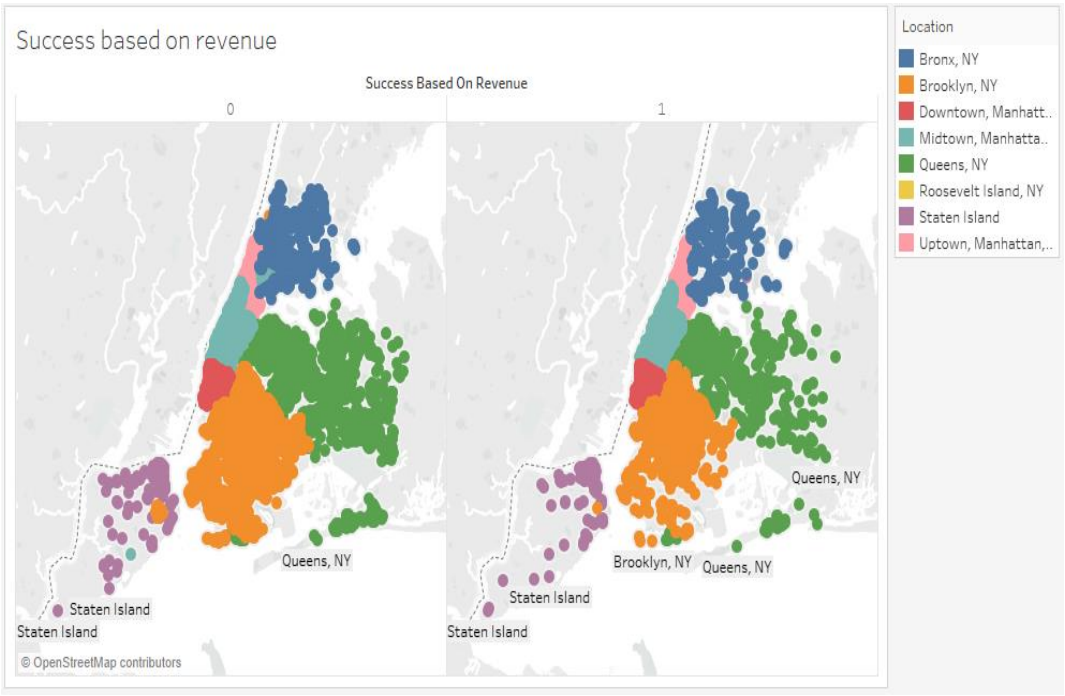
We have plotted grid of the amenities and 1 represents the availability of that amenity in that property and 0 represents the unavailability.For example for First_Aid_Kit we find from the grid that almost 25% of the property has first aid kit and almost 75% of the property does not have it..Similar interpretation can be given to all the remaining amenities.

```
p17 <- ggplot(newyork.df, aes(x=Hair_dryer)) + ggtitle("Hair_dryer") + xlab("Hair_dryer") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p18 <- ggplot(newyork.df, aes(x=Indoor_Fireplace)) + ggtitle("Indoor_Fireplace") + xlab("Indoor_Fireplace") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p19 <- ggplot(newyork.df, aes(x=Patio.balcony)) + ggtitle("Patio.balcony") + xlab("Patio.balcony") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p20 <- ggplot(newyork.df, aes(x=Essentials)) + ggtitle("Essentials") + xlab("Essentials") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p21 <- ggplot(newyork.df, aes(x=Self_checkin)) + ggtitle("Self_checkin") + xlab("Self_checkin") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p22 <- ggplot(newyork.df, aes(x=Elevator)) + ggtitle("Elevator") + xlab("Elevator") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p23 <- ggplot(newyork.df, aes(x=Pillow.Mattress)) + ggtitle("Pillow.Mattress") + xlab("Pillow.Mattress") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
p24 <- ggplot(newyork.df, aes(x=Pets_Allowed)) + ggtitle("Pets_Allowed") + xlab("Pets_Allowed") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentage") + coord_flip() + theme_minimal()
grid.arrange(p17, p18, p19, p20, p21, p22, p23, p24, ncol=2)
```



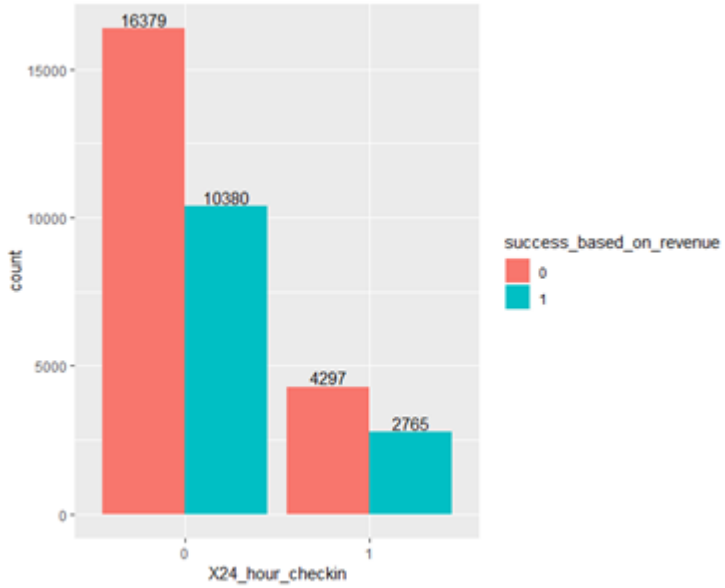
We have plotted grid of the amenities and 1 represents the availability of that amenity in that property and 0 represents the unavailability. For example for Hair_dryer we find from the grid that almost 57% of the property has Hair_dryer and almost 43% of the property does not have it. Similar interpretation can be given to all the remaining amenities.

Map plot for success_based_on_revenue depending on location:



The following are the graph plots depicting the impacts of amenities on Success Rate and Average Revenue.

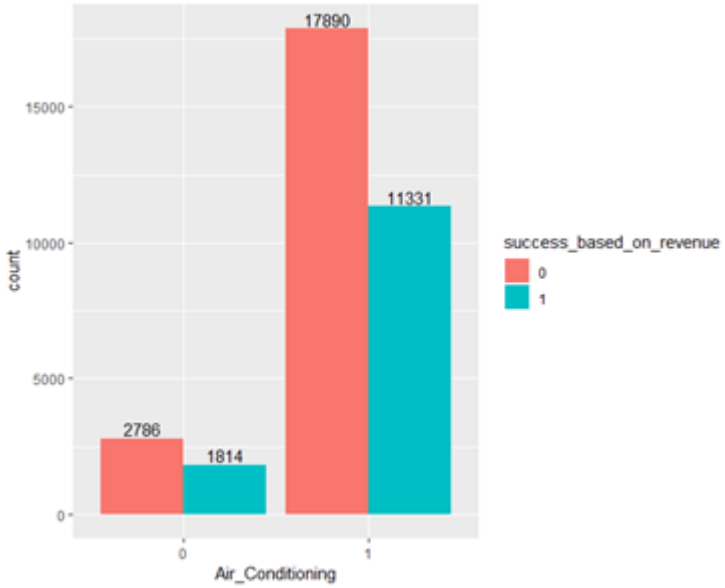
Frequency distribution of X24_hour_checkin considering the success based on revenue.




```
[1] "X24_hour_checkin"
  yes.No SuccessRate Avg.Revenue
1      0   0.3879069    3357.409
2      1   0.3915321    3270.957
```

In those properties where we have 24 hour checking, the success rate is 39.15% generating the average revenue of 3270$ per month.

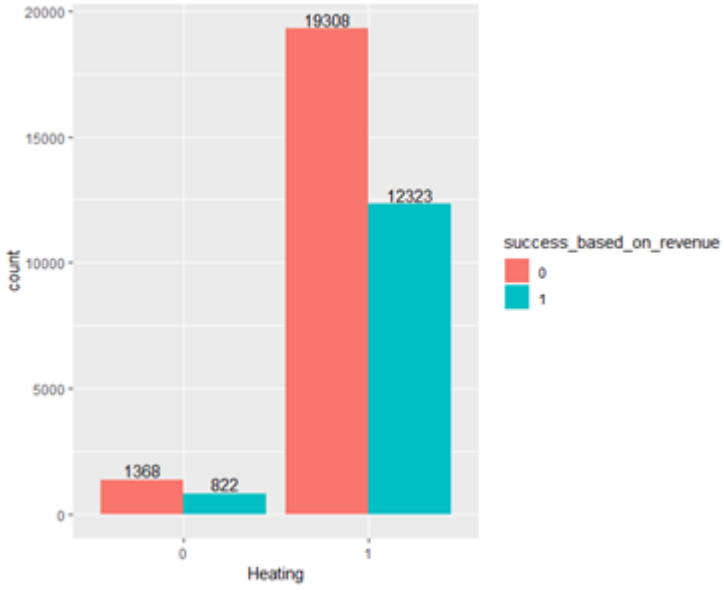
Frequency distribution of Air_Conditioning considering the success based on revenue.




```
[1] "Air_Conditioning"
  yes.No SuccessRate Avg.Revenue
1      0   0.3943478    3383.503
2      1   0.3877691    3332.408
```

In those properties where we have Air Conditioning, the success rate is 38.77% generating the average revenue of 3332$ per month.

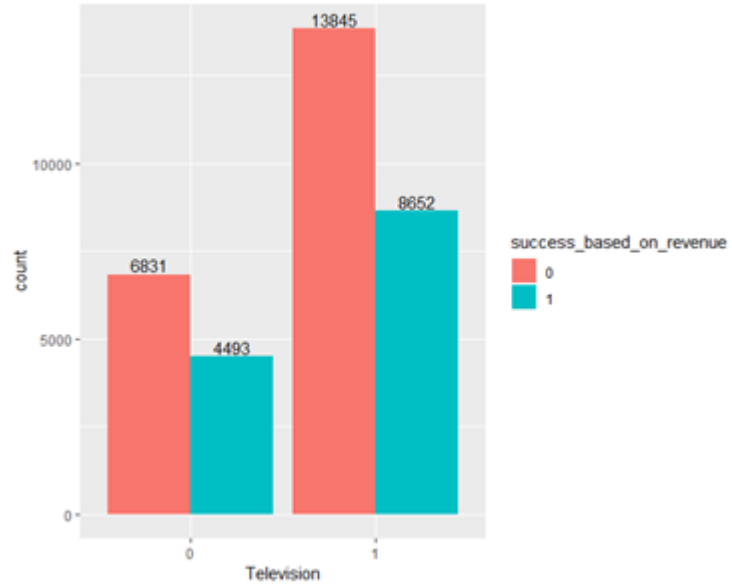
Frequency distribution of Heating considering the success based on revenue.




```
[1] "Heating"
  yes.No SuccessRate Avg.Revenue
1      0   0.3753425    3398.816
2      1   0.3895862    3335.241
```

In those properties where we have Heating, the success rate is 38.95% generating the average revenue of 3335$ per month.

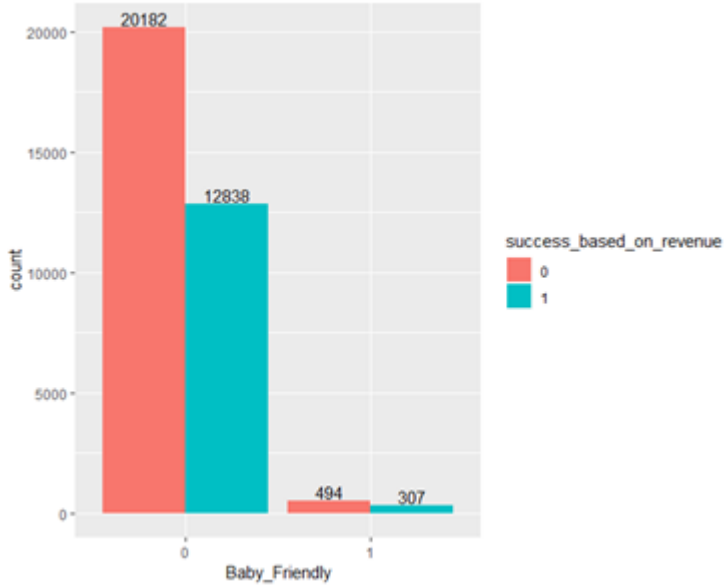
Frequency distribution of Television considering the success based on revenue.




```
[1] "Television"
  yes.No SuccessRate Avg.Revenue
1      0   0.3967679    3381.775
2      1   0.3845846    3318.007
```

In those properties where we have Television, the success rate is 38.45% generating the average revenue of 3318$ per month.

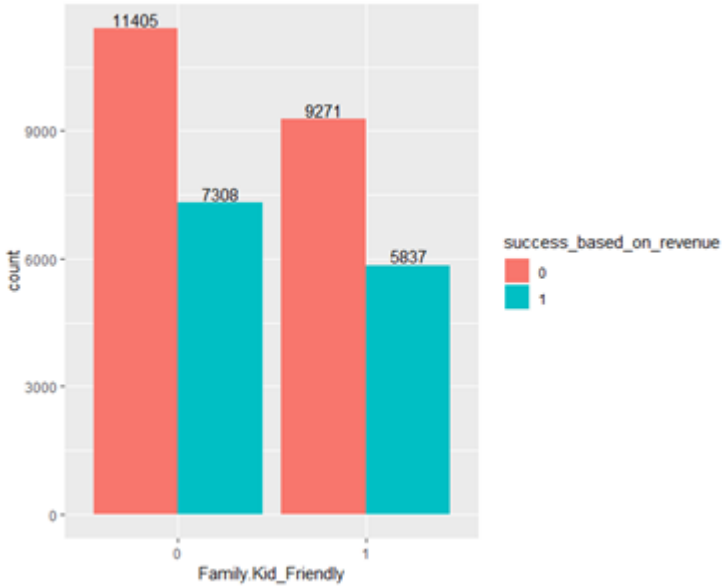
Frequency distribution of Baby_Friendly considering the success based on revenue.




```
[1] "Baby_Friendly"
  yes.No SuccessRate Avg.Revenue
1      0   0.3887947    3342.043
2      1   0.3832709    3228.665
```

In those properties where we have Baby friendly amenities, the success rate is 38.32% generating the average revenue of 3228$ per month.

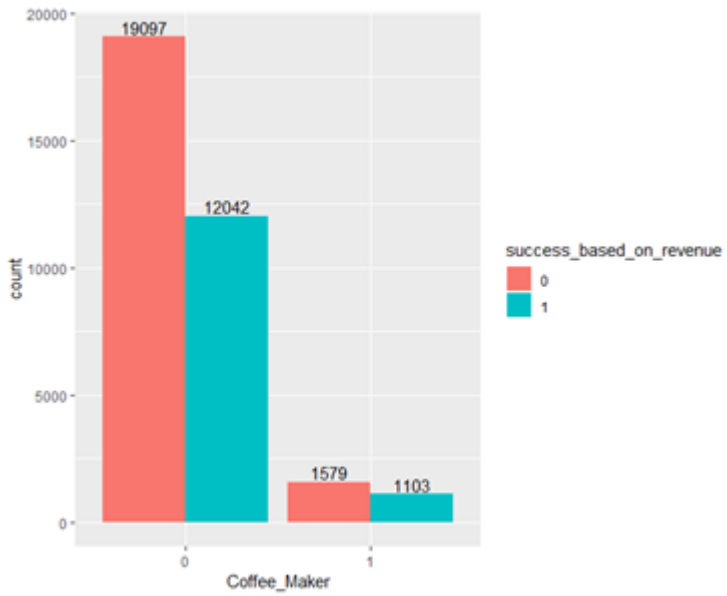
Frequency distribution of Family.Kid_Friendly considering the success based on revenue.




```
[1] "Family.Kid_Friendly"
  yes.No SuccessRate Avg.Revenue
1      0   0.3905306    3289.917
2      1   0.3863516    3400.596
```

In those properties which are Family and Kid Friendly, the success rate is 38.63% generating the average revenue of 3400$ per month.

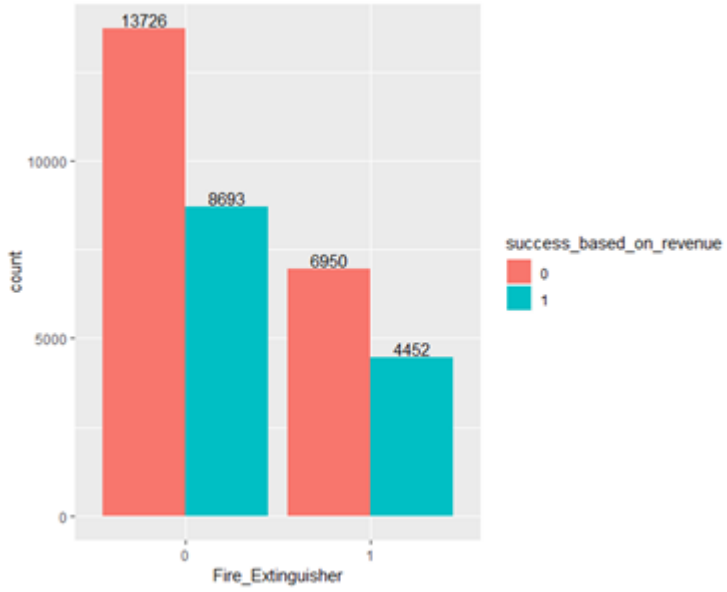
Frequency distribution of Coffee_Maker considering the success based on revenue.




```
[1] "Coffee_Maker"
  yes.No SuccessRate Avg.Revenue
1      0   0.3867176    3328.477
2      1   0.4112603    3465.691
```

In those properties where we have Coffee_Maker, the success rate is 41.12% generating the average revenue of 3465$ per month.

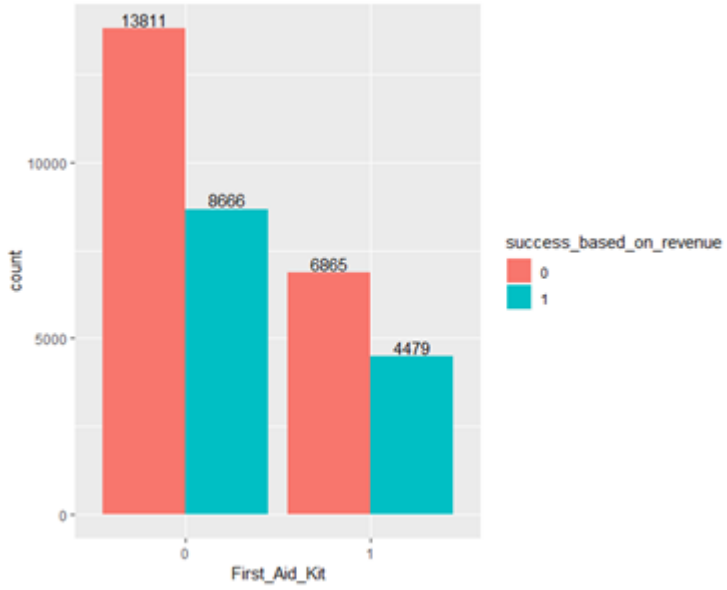
Frequency distribution of Fire_Extinguisher considering the success based on revenue.




```
[1] "Fire_Extinguisher"
  yes.No SuccessRate Avg.Revenue
1      0   0.3877515    3323.382
2      1   0.3904578    3370.770
```

In those properties where we have Fire Extinguisher, the success rate is 39.04% generating the average revenue of 3370$ per month.

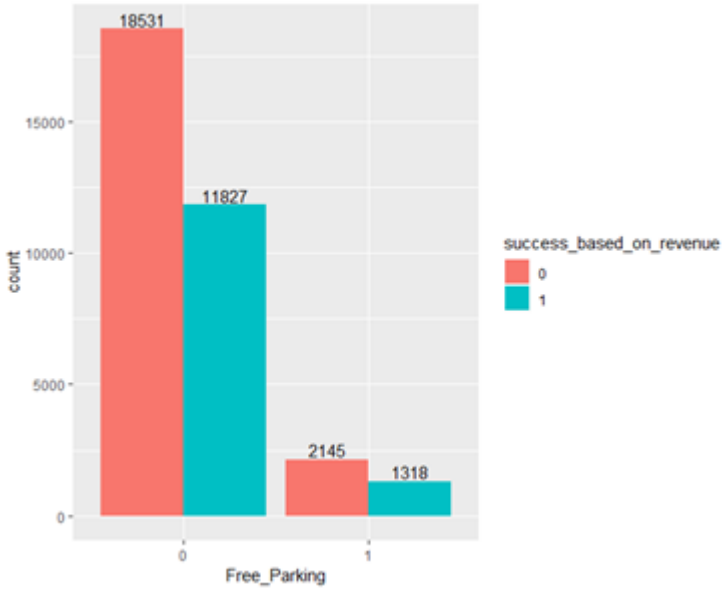
Frequency distribution of First_Aid_Kit considering the success based on revenue.




```
[1] "First_Aid_Kit"
  yes.No SuccessRate Avg.Revenue
1      0   0.3855497    3309.177
2      1   0.3948343    3399.158
```

In those properties where we have First Aid kit, the success rate is 39.48% generating the average revenue of 3399$ per month.

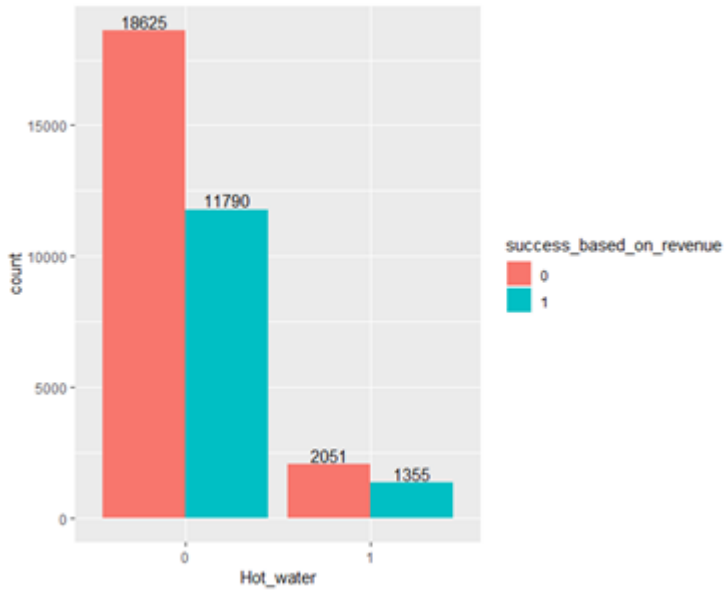
Frequency distribution of Free_Parking considering the success based on revenue.




```
[1] "Free_Parking"
  yes.No SuccessRate Avg.Revenue
1      0   0.3895843    3336.225
2      1   0.3805949    3366.821
```

In those properties where we have Free Parking, the success rate is 38.05% generating the average revenue of 3366$ per month.

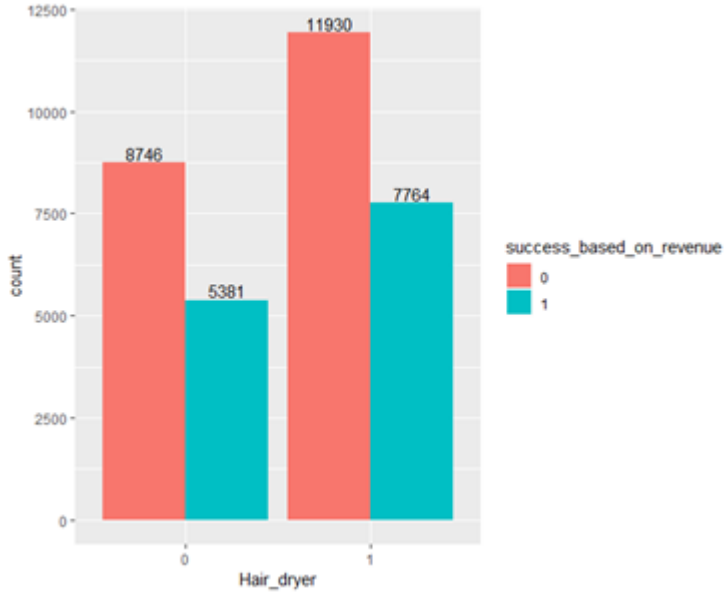
Frequency distribution of Hot_water considering the success based on revenue.




```
[1] "Hot_water"
  yes.No SuccessRate Avg.Revenue
1      0   0.3876377    3330.541
2      1   0.3978274    3418.085
```

In those properties where we have Hot_Water, the success rate is 39.78% generating the average revenue of 3418$ per month.

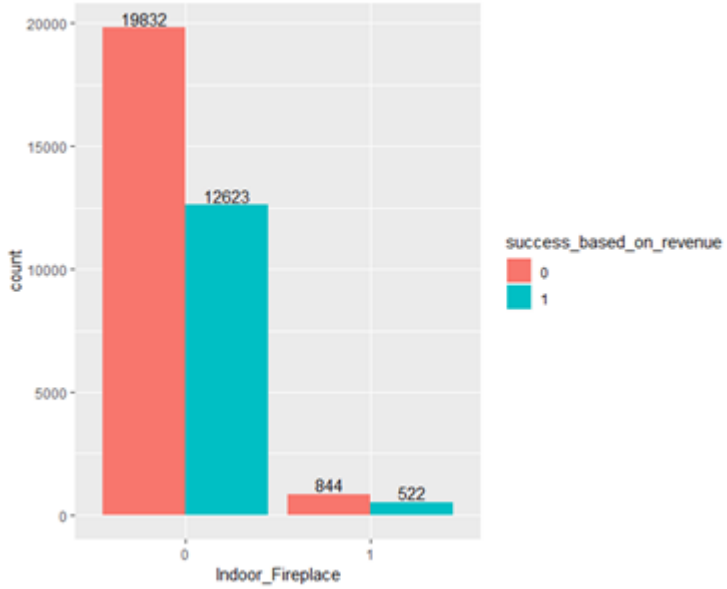
Frequency distribution of Hair_dryer considering the success based on revenue.




```
[1] "Hair_dryer"
  yes.No SuccessRate Avg.Revenue
1      0   0.3809018    3259.842
2      1   0.3942317    3396.396
```

In those properties where we have Hair dryer, the success rate is 39.42% generating the average revenue of 3396$ per month.

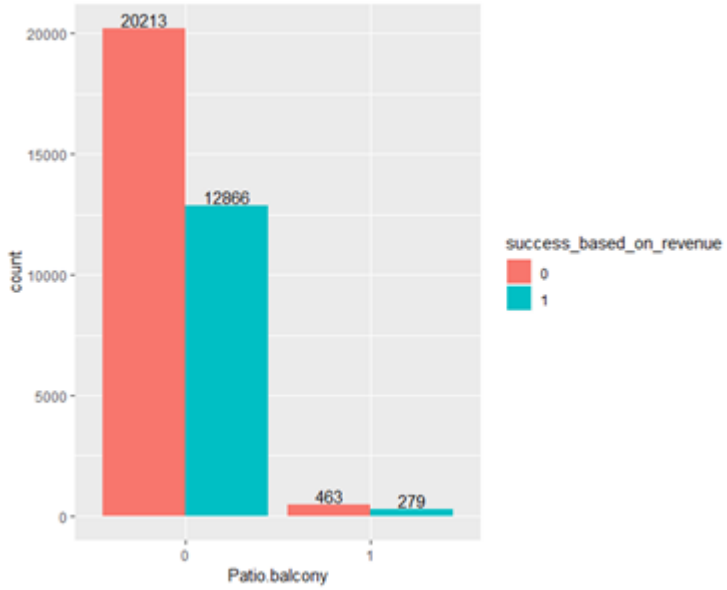
Frequency distribution of Indoor_Fireplace considering the success based on revenue.




```
[1] "Indoor_Fireplace"
  yes.No SuccessRate Avg.Revenue
1      0   0.3889385    3332.245
2      1   0.3821376    3508.347
```

In those properties where we have Indoor Fireplace, the success rate is 38.12% generating the average revenue of 3508$ per month.

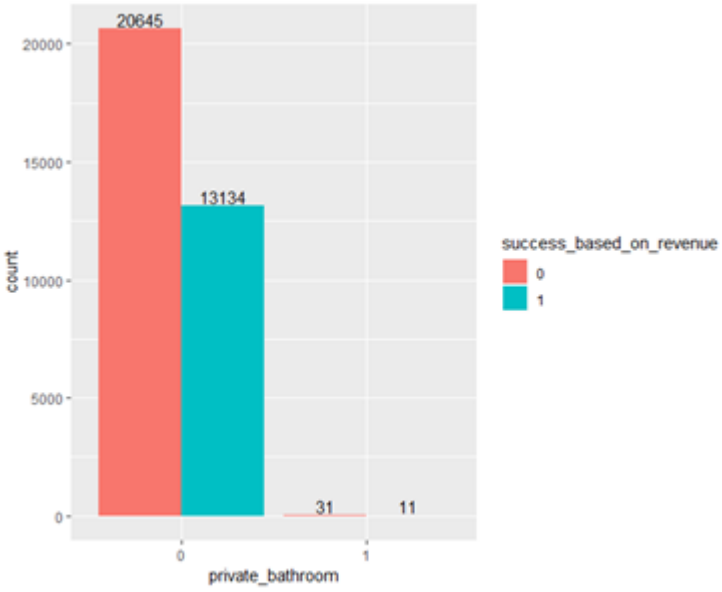
Frequency distribution of Patio.balcony considering the success based on revenue.




```
[1] "Patio.balcony"
  yes.No SuccessRate Avg.Revenue
1      0   0.3889477    3342.443
2      1   0.3760108    3201.814
```

In those properties where we have Patio balcony, the success rate is 37.60% generating the average revenue of 3201$ per month.

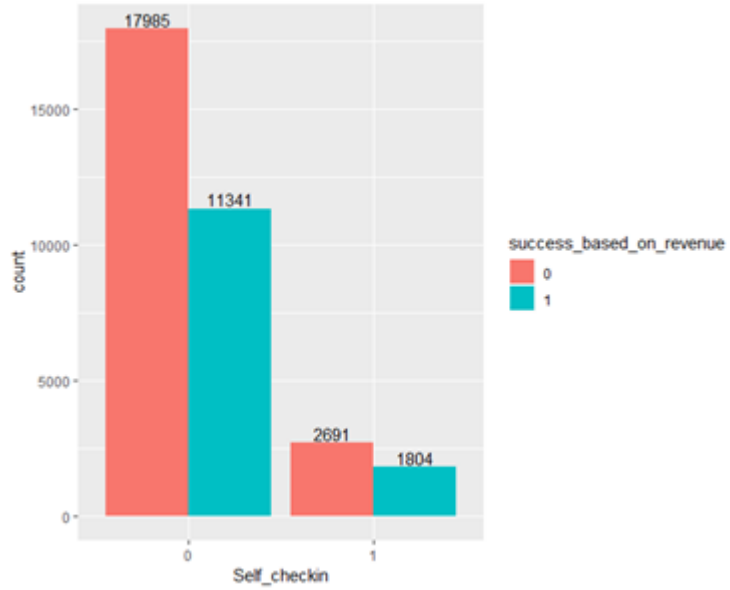
Frequency distribution of private_bathroom considering the success based on revenue.




```
[1] "private_bathroom"
  yes.No SuccessRate Avg.Revenue
1      0   0.3888215    3340.221
2      1   0.2619048    2645.048
```

In those properties where we have private_bathroom, the success rate is 26.19% generating the average revenue of 2645$ per month.

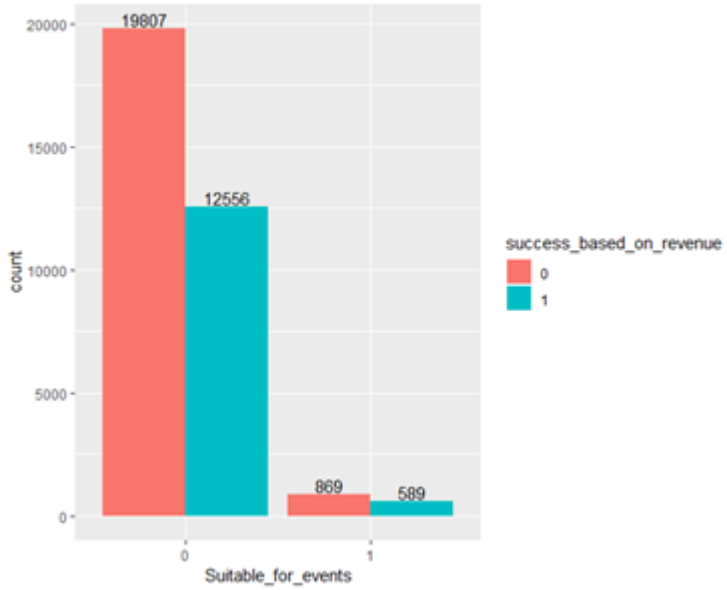
Frequency distribution of Self_checkin considering the success based on revenue.




```
[1] "Self_checkin"
  yes.No SuccessRate Avg.Revenue
1      0   0.3867217    3317.695
2      1   0.4013348    3480.689
```

In those properties where we have self_checkin, the success rate is 40.13% generating the average revenue of 3480$ per month.

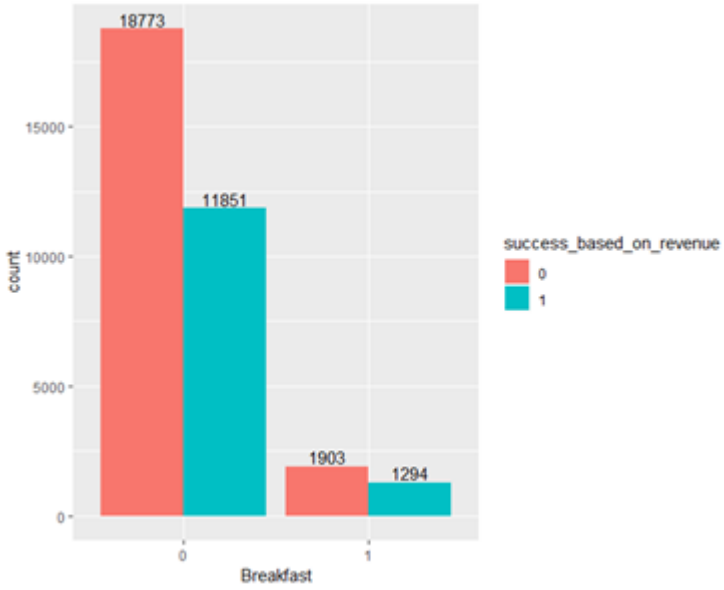
Frequency distribution of Suitable_for_events considering the success based on revenue.




```
[1] "Suitable_for_events"
  yes.No SuccessRate Avg.Revenue
1      0   0.3879739    3334.406
2      1   0.4039781    3449.269
```

In those properties which are suitable for events, the success rate is 40.39% generating the average revenue of 3449$ per month.

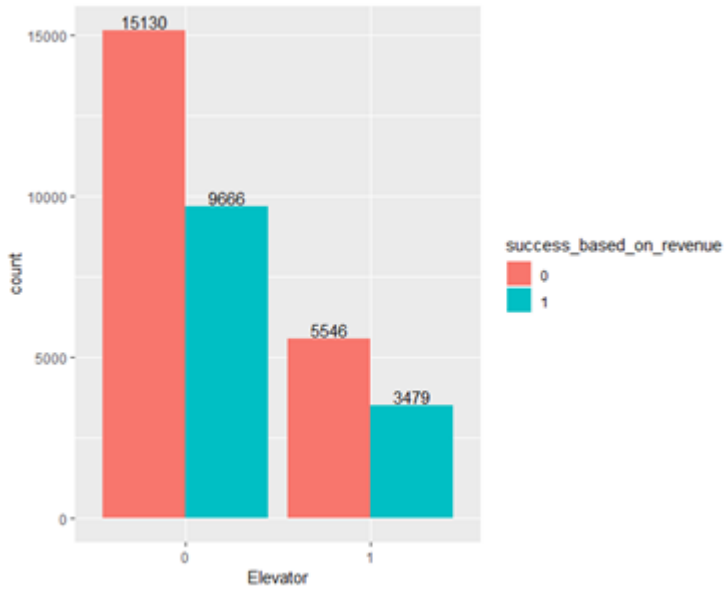
Frequency distribution of Breakfast considering the success based on revenue.




```
[1] "Breakfast"
  yes.No SuccessRate Avg.Revenue
1      0   0.3869841    3323.668
2      1   0.4047545    3489.649
```

In those properties where we have Breakfast, the success rate is 40.47% generating the average revenue of 3489$ per month.

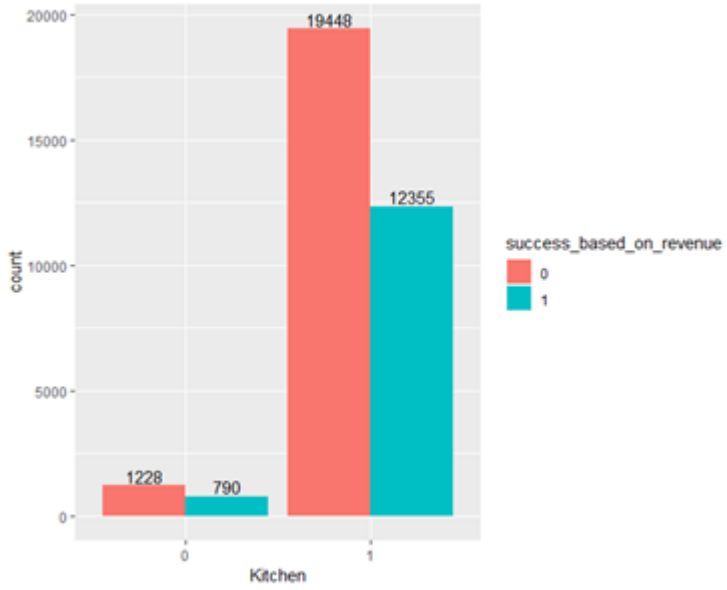
Frequency distribution of elevator considering the success based on revenue.




```
[1] "Elevator"
  yes.No SuccessRate Avg.Revenue
1      0   0.3898209    3349.325
2      1   0.3854848    3311.973
```

In those properties where we have elevator, the success rate is 38.54% generating the average revenue of 3311$ per month.

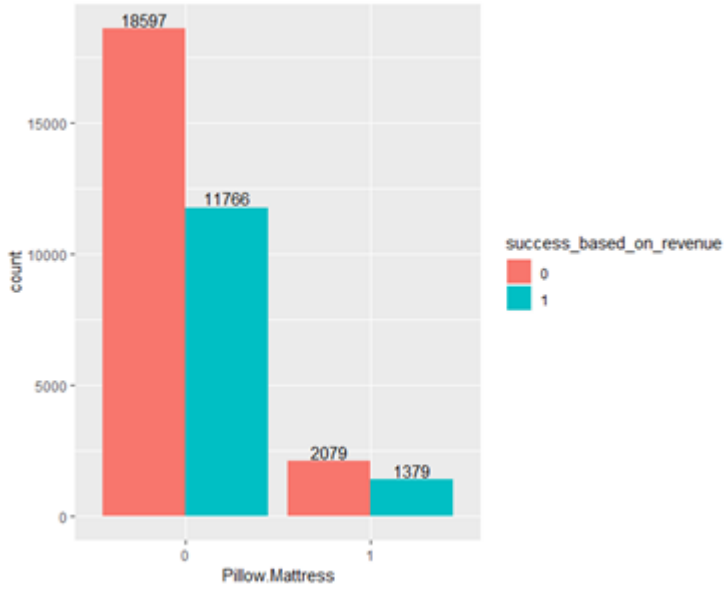
Frequency distribution of Kitchen considering the success based on revenue.




```
[1] "Kitchen"
  yes.No SuccessRate Avg.Revenue
1      0   0.3914767    3364.498
2      1   0.3884854    3337.763
```

In those properties where we have kitchen, the success rate is 38.84% generating the average revenue of 3337$ per month.

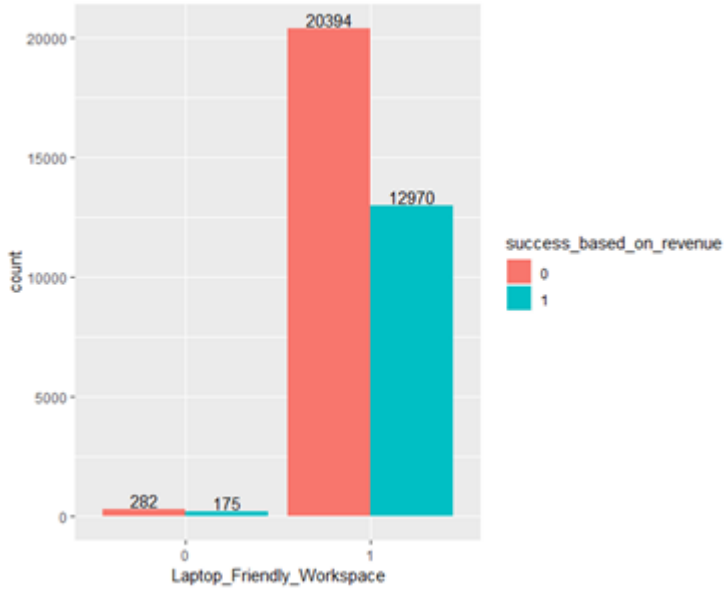
Frequency distribution of Pillow.Mattress considering the success based on revenue.




```
[1] "Pillow.Mattress"
  yes.No SuccessRate Avg.Revenue
1      0   0.3875111    3329.456
2      1   0.3987854    3426.296
```

In those properties where we have Pillow and Mattress, the success rate is 39.87% generating the average revenue of 3426$ per month.

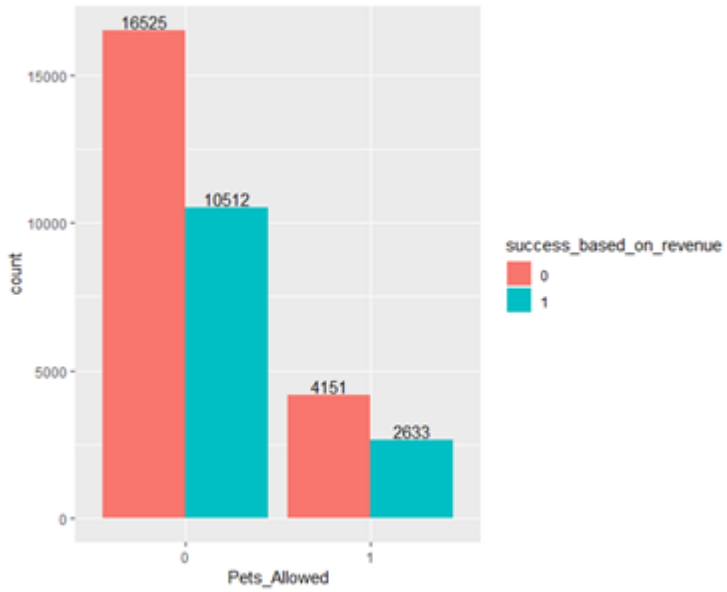
Frequency distribution of Laptop_Friendly_Workspace considering the success based on revenue.




```
[1] "Laptop_Friendly_Workspace"
  yes.No SuccessRate Avg.Revenue
1      0   0.3829322    3296.013
2      1   0.3887424    3339.951
```

In those properties where we have Laptop Friendly Workspace, the success rate is 38.87% generating the average revenue of 3339$ per month.

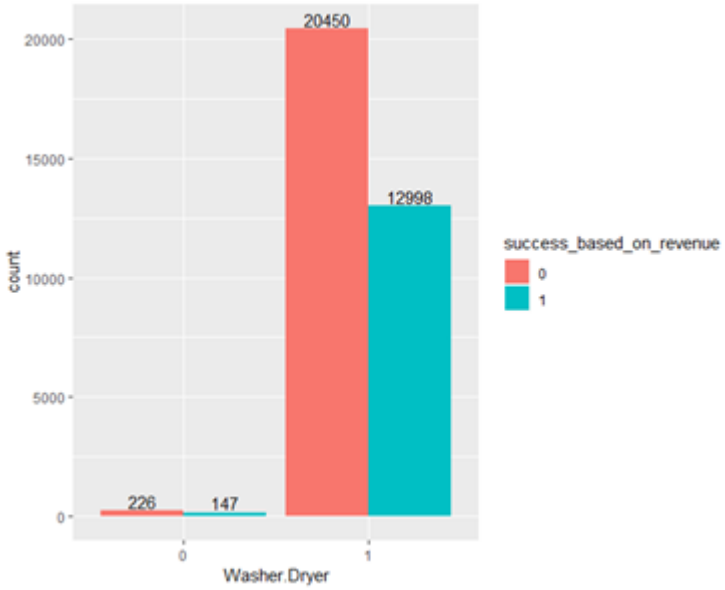
Frequency distribution of Pets_Allowed considering the success based on revenue.




```
[1] "Pets_Allowed"
  yes.No SuccessRate Avg.Revenue
1      0   0.3888005    3354.739
2      1   0.3881191    3278.058
```

In those properties where we allow Pets, the success rate is 38.81% generating the average revenue of 3278$ per month.

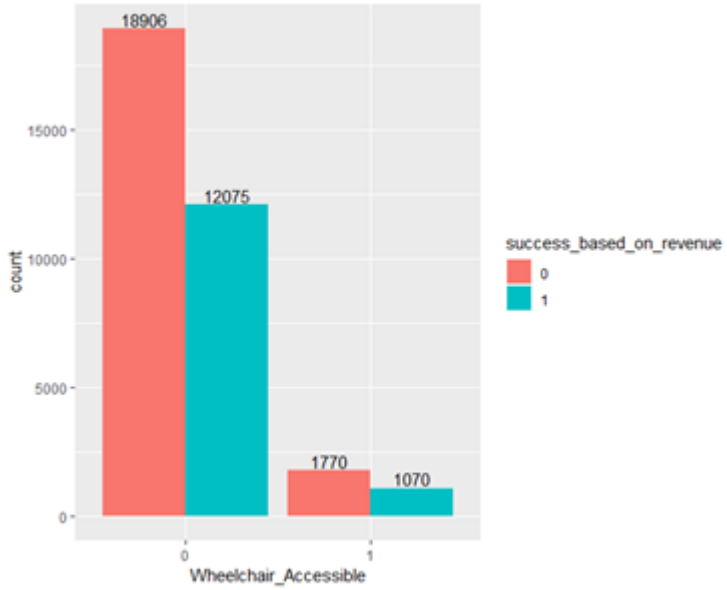
Frequency distribution of Washer.Dryer considering the success based on revenue.




```
[1] "Washer.Dryer"
  yes.No SuccessRate Avg.Revenue
1      0   0.3941019    3158.359
2      1   0.3886032    3341.376
```

In those properties where we have washer and dryer, the success rate is 38.86% generating the average revenue of 3341$ per month.

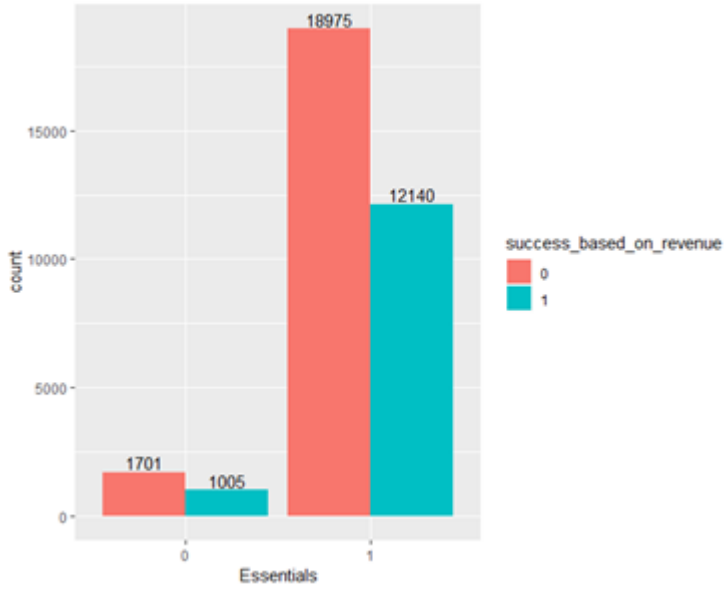
Frequency distribution of Wheelchair_Accessible considering the success based on revenue.




```
[1] "Wheelchair_Accessible"
  yes.No SuccessRate Avg.Revenue
1      0   0.3897550    3353.393
2      1   0.3767606    3186.249
```

In those properties where we have Wheelchair access , the success rate is 37.67% generating the average revenue of 3186$ per month.

Frequency distribution of Essentials considering the success based on revenue.




```
[1] "Essentials"
  yes.No SuccessRate Avg.Revenue
1      0   0.3713969    3020.236
2      1   0.3901655    3367.111
```

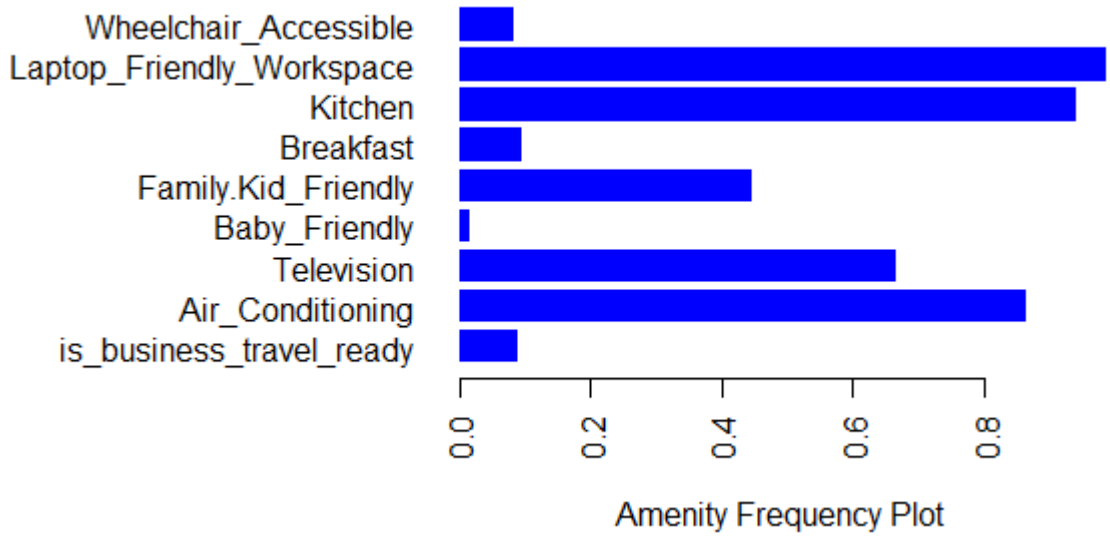
In those properties where we have essentials, the success rate is 39.01% generating the average revenue of 3367$ per month.

Thus, from the above exploratory analysis it is evident that the inclusion of any specific amenity in a property is not significantly contributing in determining the success of that property.

Association Rules:

Amenity-Frequency Plot

```
##Associative rules [AMENITIES]
### Create a Binary Incidence Matrix
amenities.df <- newyork.df[, c(24,29,31,32,33,45,47,49,52)]
amenities.df$is_business_travel_ready <- ifelse(amenities.df$is_business_travel_ready == "t", 1, 0)
amenities.df <- ifelse(amenities.df == 1, 1, 0)
amenities.mat <- as.matrix(amenities.df)


###  Convert Binary Incidence Matrix -> Transactions Database
amenities.trans <- as(amenities.mat, "transactions")
inspect(head(amenities.trans, 10))

### Amenity-Frequency Plot
itemFrequencyPlot(amenities.trans, horiz = TRUE, col = "blue", border = NA,xlab = "Amenity Frequency Plot")
```



The graph above explains the frequency of the amenities included in the properties. It can be inferred that, the amenities Laptop_Friendly _Workspace, Kitchen and Air_Conditioning are available with most most properties in New York city.

```
### Apriori function
rules <- apriori(amenities.trans,
                 parameter = list(supp= 200/4000, conf = 0.5, target = "rules"))

### Inspecting top 30 rules
inspect(head(sort(rules, by = "lift"),10))
```

```
     lhs                                                                                                  rhs                        support    confidence lift     count
[1]  {Laptop_Friendly_Workspace,Wheelchair_Accessible}                                                 => {Family.Kid_Friendly} 0.05031223 0.6011315  1.345409  1700
[2]  {Wheelchair_Accessible}                                                                           => {Family.Kid_Friendly} 0.05046021 0.6007752  1.344611  1705
[3]  {Air_Conditioning,Laptop_Friendly_Workspace,Wheelchair_Accessible}                                => {Television}          0.06158809 0.7970126  1.198019  2081
[4]  {Air_Conditioning,Wheelchair_Accessible}                                                          => {Television}          0.06176566 0.7968690  1.197803  2087
[5]  {Air_Conditioning,Kitchen,Laptop_Friendly_Workspace,Wheelchair_Accessible}                        => {Television}          0.05984196 0.7963765  1.197062  2022
[6]  {Air_Conditioning,Kitchen,Wheelchair_Accessible}                                                  => {Television}          0.05998994 0.7961508  1.196723  2027
[7]  {Laptop_Friendly_Workspace,Wheelchair_Accessible}                                                 => {Television}          0.06505076 0.7772277  1.168279  2198
[8]  {Wheelchair_Accessible}                                                                           => {Television}          0.06522833 0.7766032  1.167340  2204
[9]  {Kitchen,Laptop_Friendly_Workspace,Wheelchair_Accessible}                                         => {Television}          0.06312705 0.7759185  1.166311  2133
[10] {Air_Conditioning,Family.Kid_Friendly,Kitchen,Laptop_Friendly_Workspace}                          => {Television}          0.29983131 0.7756086  1.165845 10131
```

From the above results, we can interpret that when a property is a Laptop_Friendly_Workspace and accessible with wheelchair, then it is also Family and kid friendly 5% of the times. The lift for this association group is 1.34 which is low which means that the two associations just picked by randomly, they do not have interrelationship when they happened to include in the same Airbnb. The inference can be made in the same way for the other groups too.

Dimension Reduction:

The two Dimension Reduction techniques that are known to us is the Principal Component Analysis (PCA) and the Linear Discriminant Analysis (LDA). Our dataset predominantly contains categorical variables. Thus, we cannot perform PCA on our dataset. LDA is the technique we implemented for dimension reduction.

Linear Discriminant Analysis (LDA):

A new dataset is created in which the values of variable price is less than 250. This dataset is a subset of the original dataset. As LDA is sensitive to outliers, and 90% of the price in the dataset is lesser than 250. Thus, to avoid outliers, the new dataset was created.

```
##LDA
#90% of the price are lower than 250, so get rid of the outliers
newyork.df1 <- subset(newyork.df,newyork.df$price <= 250)
```

Analysis of price variable is to be carried out using LDA. Thus, the price variable is converted into a categorical variable with three levels, high, medium and low since LDA can be implemented only on categorical variables.

```
#Creating a Price Categorical variable
newyork.df1$priceCat<- NA
#separate the price into 3 price categories by 33%, 66%, 99% of the dataset
newyork.df1$priceCat <- as.factor(ifelse(newyork.df1$price >= 1 & newyork.df1$price <= 75, 'Low',
                                         ifelse(newyork.df1$price > 127 & newyork.df1$price <= 250,'Medium','High')))
```

This new dataset is split into training and validation dataset, with 70% data in the training dataset and the remaining 30% in the validation dataset and chose the significant variables.

```
#Splitting the dataset into training and validation data[Train=70%  Validation=30%]
set.seed(96)
training.index1 <- createDataPartition(newyork.df1$price, p = 0.7, list = FALSE)
#select significant variables
final.train1 <- newyork.df1[training.index1, c(2,3,9,10,11,12,13,21,22,54,55,57,60)]
final.valid1 <- newyork.df1[-training.index1, c(2,3,9,10,11,12,13,21,22,54,55,57,60)]
```

The training and the validation dataset is then normalized as LDA is sensitive to scales.

```
#Normalizing the data
norm.values1  <- preProcess(final.train1, method = c("center", "scale"))
# Transform the data using the estimated parameters
train.norm1 <- predict(norm.values1, final.train1)
valid.norm1 <- predict(norm.values1, final.valid1)
```

Now, LDA is developed on this normalized training dataset, with price Category as the target variable and all the significant variables chosen as the predictors.

```
LDA <-lda(priceCat ~ ., data= train.norm1)
LDA
summary(LDA)
```

Summary Result:

```
Coefficients of linear discriminants:
                                        LD1         LD2
host_response_timeNot specified      -0.06834776  0.331532903
host_response_timewithin a day       -0.05076036  0.015596621
host_response_timewithin a few hours -0.07201738 -0.132083251
host_response_timewithin an hour     -0.10557652  0.009270453
host_response_rate                    0.01931989  0.073207952
property_typeBed & Breakfast          0.58934946  0.959111623
property_typeBoutique hotel           1.63601764 -2.355867985
property_typeBungalow                 1.51178257 -0.843266422
property_typeCondominium              0.15721991  1.088304993
property_typeDorm                    -0.79751486 -1.174278526
property_typeGuest suite             -0.10841993  1.060229518
property_typeGuesthouse              -0.34754322 -1.304029623
property_typeHostel                  -0.07021692  0.210805011
property_typeHouse                   -0.16286829 -0.671565140
property_typeLoft                     0.21410531 -0.102492807
property_typeOther                   -0.24198708 -0.114805890
property_typeTimeshare               -0.11760030 -2.044624275
property_typeTownhouse                0.05877577  0.196664671
property_typeVacation home            1.89436634 -2.969644030
property_typeVilla                   -0.56456806 -1.459324541
room_typePrivate room                -0.92573226 -1.534579129
room_typeShared room                 -1.37598650 -4.077266611
accommodates                          0.12146108  0.168448423
bathrooms                            -0.03435349 -0.156816486
bedrooms                              0.02008446 -0.222342161
availability_30                       1.01494897 -0.374023858
number_of_reviews                     0.01085940  0.137962673
LocationBrooklyn, NY                  0.51593397  1.203600803
LocationDowntown, Manhattan, NY       0.86885815  2.259830102
LocationMidtown, Manhattan, NY        0.92375720  2.257355731
LocationQueens, NY                    0.28754063  0.917424755
LocationRoosevelt Island, NY          0.48435545  3.067254613
LocationStaten Island                -0.01931538  0.607236837
LocationUptown, Manhattan, NY         0.54460271  1.808867835
rating3                              -0.37833617 -0.405550130
rating4                              -0.42031923  0.210673527
rating5                              -0.31835797  0.535348763
revenue                               1.96112259 -1.186334197
```

```
Proportion of trace:
   LD1    LD2
0.9848 0.0152
```

From the above result of the LDA, it can be interpreted that the original 3 categories of price has been reduced to 2 LDA components. LD1 explains a variance of 98.48% in the dataset.

The significant variables in LD1 and LD2 can be chosen from the coefficients.

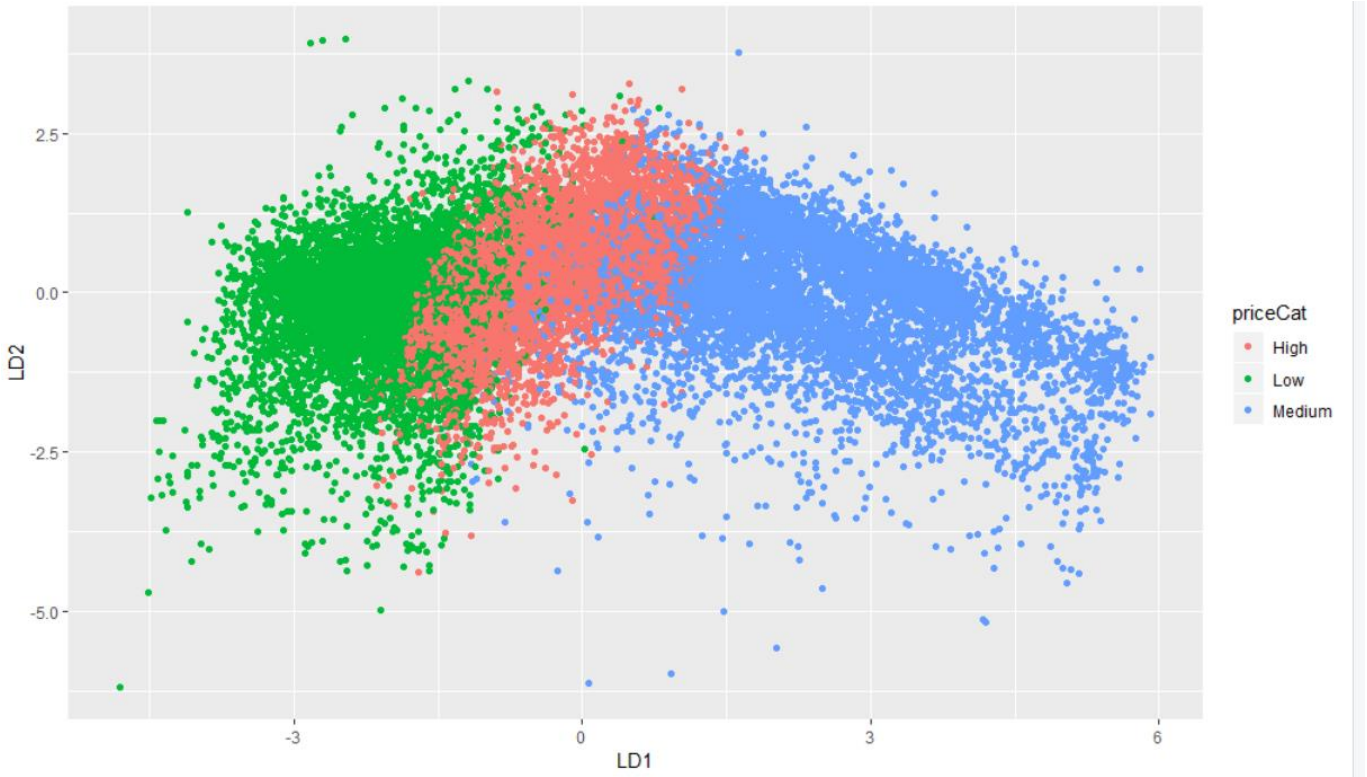
The most significant variable in LD1 is, revenue with a value of 1.96. The most significant variable in LD2 is room_typeShared Room with a value of -4.077.

Accuracy of LDA model:

```
> LDA.reg.pred <- predict(LDA, valid.norm1)
> confusionMatrix(LDA.reg.pred$class,valid.norm1$priceCat)
Confusion Matrix and Statistics

          Reference
Prediction High  Low Medium
    High   2372  381    416
    Low     599 2711     13
    Medium   34    0   2689

Overall Statistics

               Accuracy : 0.8434
                 95% CI : (0.8358, 0.8508)
    No Information Rate : 0.3384
    P-Value [Acc > NIR] : < 0.00000000000000022

                  Kappa : 0.7652
 Mcnemar's Test P-Value : < 0.00000000000000022

Statistics by Class:

                     Class: High Class: Low Class: Medium
Sensitivity               0.7894     0.8768        0.8624
Specificity               0.8717     0.9000        0.9944
Pos Pred Value            0.7485     0.8158        0.9875
Neg Pred Value            0.8953     0.9353        0.9339
Prevalence                0.3261     0.3355        0.3384
Detection Rate            0.2574     0.2942        0.2918
Detection Prevalence      0.3439     0.3606        0.2955
Balanced Accuracy         0.8305     0.8884        0.9284
```

LDA is run on the validation dataset and rom the confusion matrix, the accuracy of LDA model is found to be 84.34%.

Graphical representation of the accuracy of LDA model is shown below,



From the graph, it can be inferred that, our model has a good performance in classifying price categories. The graph shows three different classes represented in green, orange and blue with very few data misclassified in each class.

Regression:

Looking at the dataset, it contains many categorical variables than numeric ones. Thus, we had to consider regression models that worked the best with categorical variables in it. The best models for analyzing our dataset are Logistic Regression, Linear Regression and Decision Tree Model.

We split the dataset into training and validation datasets. We had a training dataset with 80% data and a validation dataset with the remaining 20%.

```
###Linear Regression
set.seed(96)
training.index <- createDataPartition(newyork.df$price, p = 0.8, list = FALSE)
final.train <- newyork.df[training.index, ]
final.valid <- newyork.df[-training.index, ]
```

Logistic Regression:

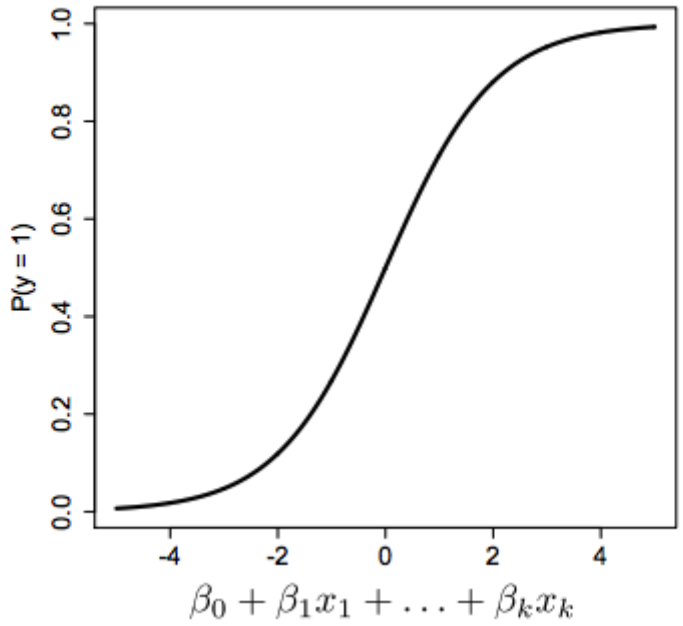
It's an extension of linear regression where the dependent variable is categorical and not continuous. It predicts the probability of the outcome variable.

Logistic regression can be binomial or multinomial. In the binomial or binary logistic regression, the outcome can have only two possible types of values (e.g. "Yes" or "No", "Success" or "Failure"). Multinomial logistic refers to cases where the outcome can have three or more possible types of values (e.g., "good" vs. "very good" vs. "best"). Generally, the outcome is coded as "0" and "1" in binary logistic regression.

Representation of Logistic regression:

Logistic Response Function:

$$P(y=1) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\ldots+\beta_k x_k)}}$$



Positive values are predictive of class 1

Negative values are predictive of class 0

The coefficients, or β values, are selected to maximize the likelihood of predicting a high probability for observations actually belonging to class 1 and predicting a low probability for observations actually belonging to class 0. The output of this function is always between 0 and 1

Odds Ratio:

Odds: P(y=1)/P(y=0)

The odds ratio for a variable in logistic regression represents how the odds change with a 1 unit increase in that variable holding all other variables constant.

Odds > 1 if y = 1 is more likely

Odds < 1 if y = 0 is more likely

The Logit

$$\text{Odds} = e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\ldots+\beta_k x_k}$$

$$log(\text{Odds}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k$$

This is called the “Logit” and looks like linear regression

The bigger the Logit is, the bigger is P (y = 1).

Baseline Model:

The baseline model in case of Logistic Regression is to predict the most frequent outcome as the outcome for all data points. (In the case of Linear regression, the baseline model predicts the average of all data points as the outcome)

Logistic Regression tries to –

- Model the probability of an event occurring depending on the values of the independent variables
- Estimate the probability that an event occurs vs the probability that the event does not occur
- Predict the effect of a series of variables on a binary response variable
- Classify observations by estimating the probability that an observation is in a particular category or not.

The output of a Logistic regression model is a probability. We can select a threshold value. If the probability is greater than this threshold value, the event is predicted to happen otherwise it is predicted not to happen.

A confusion or classification matrix compares the actual outcomes to the predicted outcomes. The rows are labelled with actual outcomes while the columns are labelled with predicted outcomes.

Linear Regression:

Simple Linear Regression Model

Model:

Simple linear regression model allows us to study the relationships between two continuous numeric variables.

- An input variable (x) also called the predictor/explanatory/independent variable.

- An output variable(y) also called the response/outcome/dependent variable.

Technique:

The technique used to determine the line of best fit by minimizing the sum of squares for simple linear regression is called Least Squares method. It helps us to determine values of b0 and b1 that minimizes the sum of squared differences around the prediction line.

The equation for our simple linear regression model can be written as

Predicted(Y) = b0 + b1(x)

where,

B0 = Y intercept for the population

B1 = Slope of the population

Assumptions:

We must follow certain rules before inputting the data to the regression model.

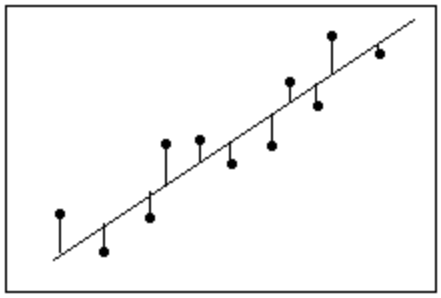
- There is a linear relationship between input and output variables

- We must make sure to remove the outliers

- Remove the over fitting data, i.e. remove the ones that are most correlated

-The data must be normalized for more reliable predictions

What Is Goodness-of-Fit for a Linear Model?



Definition: Residual = Observed value - Fitted value

Linear regression calculates an equation that minimizes the distance between the fitted line and all of the data points. Technically, ordinary least squares (OLS) regression minimizes the sum of the squared residuals.

In general, a model fits the data well if the differences between the observed values and the model's predicted values are small and unbiased.

Before you look at the statistical measures for goodness-of-fit, you should check the residual plots. Residual plots can reveal unwanted residual patterns that indicate biased results more effectively than numbers. When your residual plots pass muster, you can trust your numerical results and check the goodness-of-fit statistics.

What Is R-squared?

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

The definition of R-squared is fairly straight-forward; it is the percentage of the response variable variation that is explained by a linear model. Or:

R-squared = Explained variation / Total variation

R-squared is always between 0 and 100%:

0% indicates that the model explains none of the variability of the response data around its mean.

100% indicates that the model explains all the variability of the response data around its mean.

In general, the higher the R-squared, the better the model fits your data.

Linear Discriminant Analysis:

There are many possible techniques for classification of data. Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are two commonly used techniques for data classification and dimensionality reduction. Linear Discriminant Analysis is a dimensionality reduction technique used as a preprocessing step in Machine Learning and pattern classification applications. The main goal of dimensionality reduction techniques is to reduce the dimensions by removing the redundant and dependent features by transforming the features from higher dimensional space to a space with lower dimensions. Linear Discriminant Analysis is a supervised classification technique which takes labels into consideration.
The prime difference between LDA and PCA is that PCA does more of feature classification and LDA does data classification. In PCA, the shape and location of the original data sets changes when transformed to a different space whereas LDA doesn't change the location but only tries to provide more class separability and draw a decision region between the given classes. This method also helps to better understand the distribution of the feature data.

How does Linear Discriminant Analysis work?

The goal of Linear Discriminant Analysis is to project the features in higher dimension space onto a lower dimensional space.

This can be achieved in three steps:

The first step is to calculate the separability between different classes (i.e. the distance between the mean of different classes) also called as between-class variance

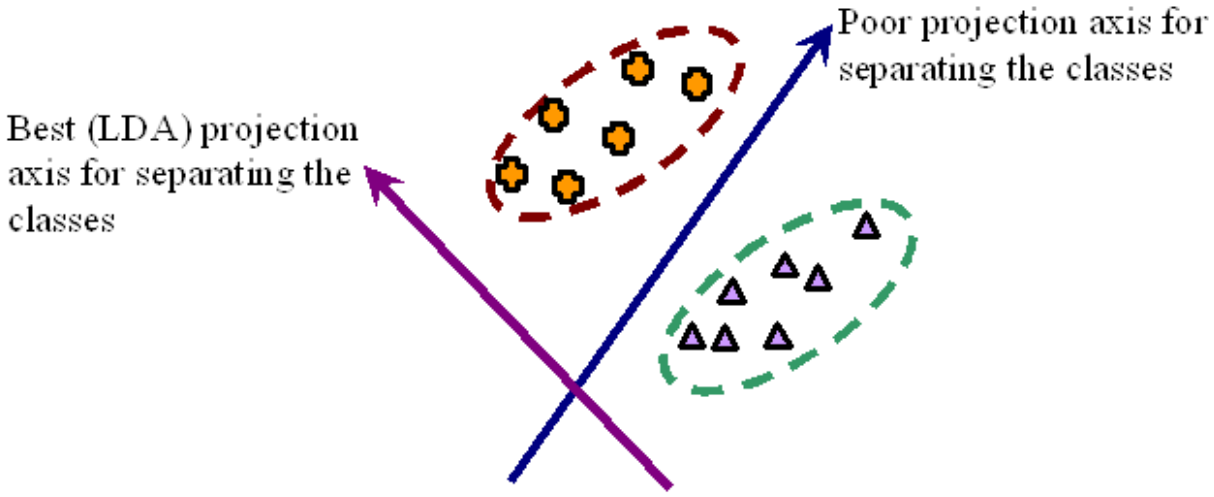
$$S_b = \sum_{i=1}^{g} N_i(\bar{x}_i - \bar{x})(\bar{x}_i - \bar{x})^T$$

Second Step is to calculate the distance between the mean and sample of each class, which is called the within class variance

$$S_w = \sum_{i=1}^{g} (N_i - 1)S_i = \sum_{i=1}^{g} \sum_{j=1}^{N_i} (x_{i,j} - \bar{x}_i)(x_{i,j} - \bar{x}_i)^T$$

The third step is to construct the lower dimensional space which maximizes the between class variance and minimizes the within class variance. Let P be the lower dimensional space projection, which is called Fisher's criterion.

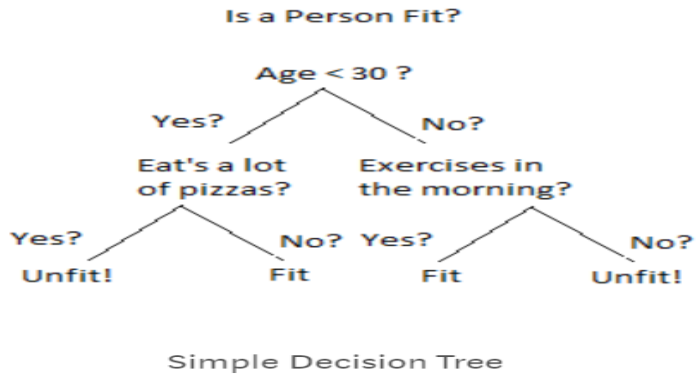
$$P_{lda} = \arg\max_{P} \frac{\left| P^T S_b P \right|}{\left| P^T S_w P \right|}$$



Decision Tree:

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. Decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node has two or more branches. Leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.



Simple Decision Tree

Our Analysis:

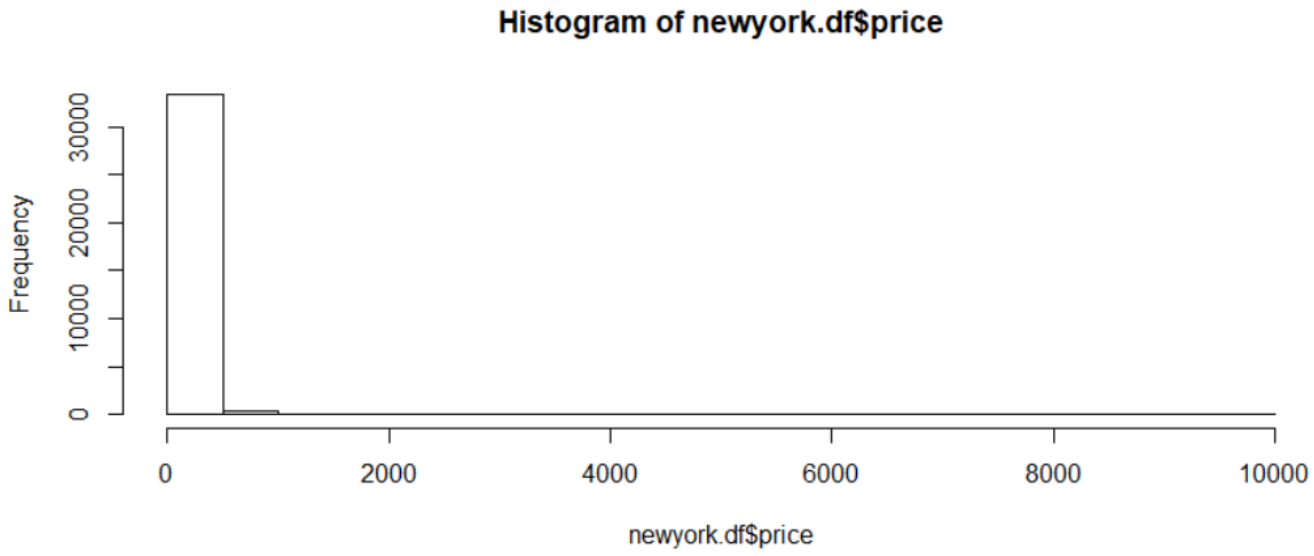
Linear Regression:

Firstly, we performed linear regression to predict the price of the rental properties. After running the linear regression, with price as our target variable and all the other variables in the dataset as our predictor variables, we found that the Adjusted R squared value was only **0.675**. This was because the price variable was not distributed properly. We plotted a histogram to check how the distribution of price variable is and found it to be highly skewed to the right. In order to make the distribution even, we decided to go with log(price) for better analysis.

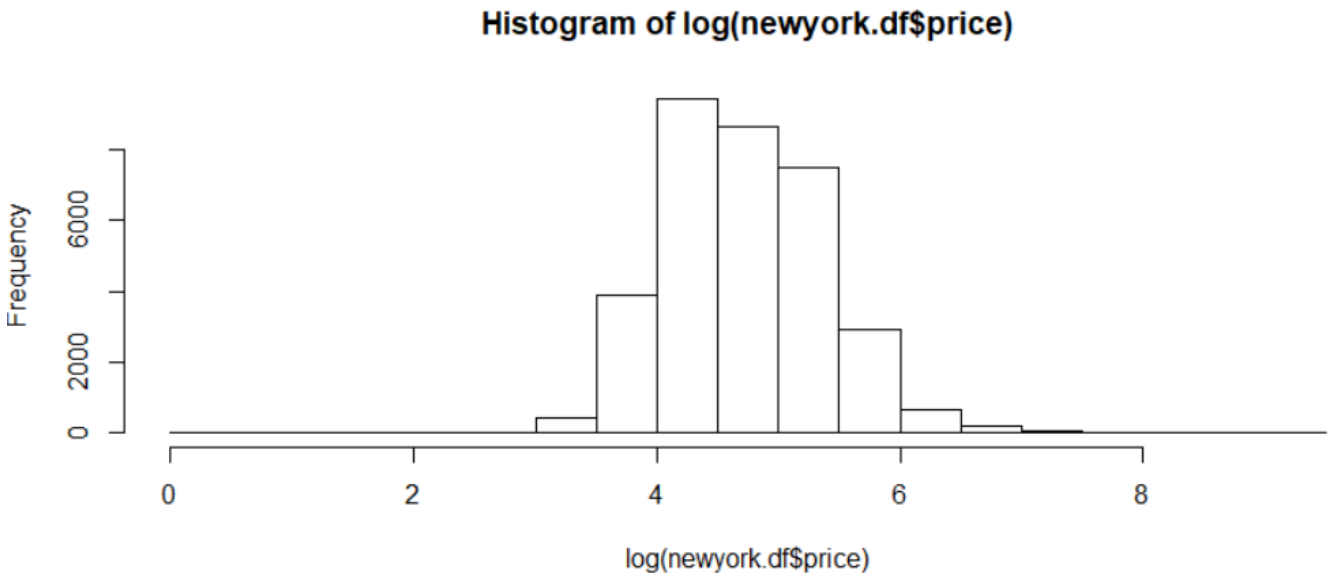
After running the linear regression with log(price), the adjusted R square value turned up to be **0.81**

```
#Filtering the price variable
hist(newyork.df$price)
summary(newyork.df$price)
newyork.df<-subset(newyork.df,price>0)

LM<-lm(price ~ ., data= final.train)
options(scipen=999)
summary(LM)

LM1<-lm(log(price) ~ ., data= final.train)
options(scipen=999)
summary(LM1)
```

Highly skewed distribution of price:



Distribution of price after taking log(price):



From the summary of the linear regression, we chose the significant variables depending upon the p-value having 0.05 as our threshold to be host_is_superhost, latitude, longitude, property_type, room_type, accommodates, bathrooms, minimum_nights, maximum_nights, availability_30, number_of_reviews, is_business_travel_ready, cancellation_policy, reviews_per_month, Heating, Family.Kid_Friendly, Suitable_for_Events, Essentials, Location, rating, revenue, success_based_on_revenue and run a linear regression with price as our target variable and these significant variables as our predictors.

```
##Selecting the significant predictors

names(final.train)
Sign_Pred<-final.train[,c(4,6,7,9,10,11,12,18,19,21,22,24,25,27,30,33,44,53,54,55,57,59)]
vif_names<-names(Sign_Pred)
vif_names
def_form <- as.formula(paste("log(price) ~" ,paste(vif_names, collapse = " +"),sep = ""))
Final_LM <- lm(def_form, data = final.train)
summary(Final_LM)
```

We tested our model on the validation dataset and found the accuracy to be around **90%**. The results of our regression model are available in the Empirical Results section below.

```
linear.reg.pred <- predict(Final_LM, final.valid)
predictnew<-data.frame(cbind(Actuals=log(final.valid$price),Pred=linear.reg.pred))
CorAcc<-cor(predictnew)

#Plotting the actual and predicted
ggplot(predictnew,aes(x=Actuals,y=Pred))+geom_point()+geom_abline(color="blue")
```

```
> RMSE(linear.reg.pred,log(final.valid$price))
[1] 0.2718482
```

The root mean square error of any model must be as low as possible. So in our case its 0.2718 .So it can be concluded that our model is a good model.

Logistic Regression:

We ran Logistic Regression on our training dataset with the success_based_on_revenue as our target variable and all the other variables in the dataset as the predictors except for id, price and calender_updated_months variables.

```
#Logistic regression with all variables
Sign_Pred1<-final.train[,c(2:15,17:19,21:55)]
vif_names1<-names(Sign_Pred1)
def_form1 <- as.formula(paste("success_based_on_revenue ~" ,paste(vif_names1, collapse = " +"),sep = ""))
Lreg<-glm(def_form1, data= final.train, family= "binomial")
options(scipen=999)
summary(Lreg)
varImp(Lreg)
```

From the above logistic regression result, the significant variables were chosen, and another logistic regression was run with the same success_based_on_revenue as our target variable and the significant variables as our predictors. The results from Model 1 indicate that there are only 17 significant variables considering the P value to be 0.05 which are, host_is_superhost, latitude, longitude, property_type, room_type, accommodates, bathrooms, bedrooms, guests_included, minimum_nights,availability_30, number_of_reviews, is_business_travel_ready, require_guest_profile_picture, reviews_per_month, family_kid_friendly, location.

```
#Logistic regression with significant variables
Sign_Pred2<-final.train[,c(4,6,7,9:13,17,18,21,22,24,26,27,33,54)]
vif_names2<-names(Sign_Pred2)
def_form2 <- as.formula(paste("success_based_on_revenue ~" ,paste(vif_names2, collapse = " +"),sep = ""))
LogModel<-glm(def_form2, data= final.train, family= "binomial")
options(scipen=999)
summary(LogModel)
```

The result generated from the above code is the final logistic regression model.

The model was tested on the validation dataset and the accuracy turned out to be **85.19%**. The results of our regression model are available in the Empirical Results section below.

Decision Tree:

Created a decision tree model, with success_based_on_revenue as our target variable and all other variables in the dataset as the predictors. The results of the decision tree are explained in the Empirical results and findings section of the report.

```
# Create a Decision tree model with default parameters
Tree <- rpart(success_based_on_revenue~ ., data = final.train,cp=0.006)
options(scipen = 999)
prp(Tree)
```

Empirical Results and Finding:

Linear Regression:

Summary output of linear regression

```
    Min      1Q  Median      3Q     Max
-7.1131 -0.1592 -0.0003  0.1582  3.5096

Coefficients:
                                        Estimate         Std. Error t value             Pr(>|t|)
(Intercept)                         -184.9596605499303    6.4306641281131 -28.762 < 0.0000000000000002 ***
host_is_superhostt                     0.0545942463746    0.0056966548320   9.584 < 0.0000000000000002 ***
latitude                               1.1250099180100    0.0757537941882  14.851 < 0.0000000000000002 ***
longitude                             -1.9322601907666    0.0740920181792 -26.079 < 0.0000000000000002 ***
property_typeBed & Breakfast           0.1605398180282    0.0322290058757   4.981    0.000000636301127 ***
property_typeBoutique hotel            0.2836975204989    0.0893629407459   3.175             0.001502 **
property_typeBungalow                  0.3259347709792    0.0898221951512   3.629             0.000285 ***
property_typeCondominium               0.1481652653418    0.0165259863812   8.966 < 0.0000000000000002 ***
property_typeDorm                     -0.0900424803552    0.0755839180941  -1.191             0.233551
property_typeGuest suite               0.1081804794861    0.0500853498279   2.160             0.030789 *
property_typeGuesthouse               -0.0223739418641    0.0567930281685  -0.394             0.693617
property_typeHostel                   -0.1310156101507    0.0633723017776  -2.067             0.038708 *
property_typeHouse                     0.0021931591730    0.0072116130256   0.304             0.761043
property_typeLoft                      0.1087484420621    0.0131772317812   8.253 < 0.0000000000000002 ***
property_typeOther                    -0.0324474956443    0.0235302310666  -1.379             0.167917
property_typeTimeshare                 0.3625949070737    0.0501398651234   7.232    0.000000000000492 ***
property_typeTownhouse                 0.0280486347023    0.0140506471017   1.996             0.045918 *
property_typeVacation home             0.3395292067965    0.1264585398810   2.685             0.007260 **
property_typeVilla                     0.0599610844386    0.0708614601863   0.846             0.397465
room_typePrivate room                 -0.3040431702331    0.0051664379360 -58.850 < 0.0000000000000002 ***
room_typeShared room                  -0.7025863816095    0.0129254522662 -54.357 < 0.0000000000000002 ***
accommodates                           0.0697856951610    0.0013080444615  53.351 < 0.0000000000000002 ***
bathrooms                              0.0832407003785    0.0051212893346  16.254 < 0.0000000000000002 ***
minimum_nights                        -0.0008308774584    0.0001424353145  -5.833    0.000000005502867 ***
maximum_nights                         0.0000000001901    0.0000000005468   0.348             0.728017
availability_30                        0.0213156639172    0.0002478609807  85.998 < 0.0000000000000002 ***
number_of_reviews                      0.0000857111720    0.0000621654280   1.379             0.167982
cancellation_policymoderate            0.0038128411056    0.0053099235404   0.718             0.472728
cancellation_policystrict             -0.0014503606341    0.0048488767269  -0.299             0.764857
cancellation_policysuper_strict_30     0.2860497521997    0.1631319235872   1.753             0.079531 .
cancellation_policysuper_strict_60     1.2162634360732    0.2829115688647   4.299    0.000017218505730 ***
reviews_per_month                     -0.0179875948020    0.0013555383043 -13.270 < 0.0000000000000002 ***
Heating1                              -0.0199191274167    0.0076486454006  -2.604             0.009213 **
Family.Kid_Friendly1                  -0.0120200917316    0.0037575075168  -3.199             0.001381 **
Suitable_for_events1                   0.0294282967066    0.0091464998782   3.217             0.001295 **
Essentials1                            0.0252585020195    0.0069184021539   3.651             0.000262 ***
LocationBrooklyn, NY                   0.4624881638861    0.0186726345421  24.768 < 0.0000000000000002 ***
LocationDowntown, Manhattan, NY        0.7734263155997    0.0183947075135  42.046 < 0.0000000000000002 ***
LocationMidtown, Manhattan, NY         0.7310433269929    0.0161730416452  45.201 < 0.0000000000000002 ***
LocationQueens, NY                     0.3515507059541    0.0166393302269  21.128 < 0.0000000000000002 ***
LocationRoosevelt Island, NY           0.4627724571473    0.0471275717906   9.820 < 0.0000000000000002 ***
LocationStaten Island                 -0.0982547601722    0.0352865743345  -2.784             0.005366 **
LocationUptown, Manhattan, NY          0.3447373390802    0.0146247056458  23.572 < 0.0000000000000002 ***
rating3                               -0.1639244199386    0.0469793701105  -3.489             0.000485 ***
rating4                               -0.1673970048479    0.0443565654326  -3.774             0.000161 ***
rating5                               -0.1006992770516    0.0442681382308  -2.275             0.022929 *
revenue                                0.0000384581131    0.0000005333186  72.111 < 0.0000000000000002 ***
success_based_on_revenue1              0.4168517665724    0.0053588115324  77.788 < 0.0000000000000002 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2823 on 23605 degrees of freedom
Multiple R-squared:  0.8115,    Adjusted R-squared:  0.8112
F-statistic:  2163 on 47 and 23605 DF,  p-value: < 0.00000000000000022
```

Interpretations:

From the estimated value of host_is_superhostt, we can infer that, the price of property whose owner is a super host is 5.5% higher when compared to properties whose owner is not a super host.

From the estimated value of latitude, we can infer that, for every unit increase in latitude, there will be 208% increase in price.

From the estimated value of longitude, we can infer that, for every unit increase in longitude, there will be 85% decrease in price.

From the estimated value of propert_typeBed & Breakfast, we can infer that, the price of property whose owner is a super host is 17.42% higher when compared to properties whose owner is not a super host.

From the estimated value of accommodates, we can infer that, for every unit increase in accommodates, there will be a 7.21% increase in price.

From the estimated value of bathrooms, we can infer that, for every unit increase in bathrooms, there will be 8.68% increase in price.

From the estimated value of minimum_nights, we can infer that, for every unit increase in minimum_nights, there will be 0.08% decrease in price.

From the estimated value of availability_30, we can infer that, for every unit increase in availability_30, there will be 2.15% increase in price.

From the estimated value of cancellation_policysuper_strict_60, we can infer that, the price of property whose cancellation policies are strict is 235% higher when compared to properties whose cancellation policies are flexible.

From the estimated value of reviews_per_month, we can infer that, for every unit increase in reviews_per_month, there will be 1.79% decrease in price.

From the estimated value of Heating1, we can infer that, the price of property which has a heater in it is 1.97% lower when compared to properties with no heater in it.

From the estimated value of Family.Kid_Friendly1, we can infer that, the price of property which is family and kids friendly is 1.19% lower when compared to properties which are not family and kids friendly.

From the estimated value of Suitable_for_events1, we can infer that, the price of property which are suitable for events and parties is 2.98% higher when compared to properties which are not suitable for events and parties.

From the estimated value of Essentials1, we can infer that, the price of property with bed, bath and kitchen essentials in them are 2.56% higher when compared to properties which do not have bed, bath and kitchen essentials.

From the estimated value of revenue, we can infer that, for every unit increase in revenue, there will be 0.003% increase in price.

From the estimated value of success_based_on_revenue1, we can infer that, the price of property which successfully generates revenue for the owners are 51.67% higher when compared to properties which fails to generate revenue for the owners.

From the estimated value of property_typebed & Breakfast, we can infer that, the price of property which provides bed and breakfast are 17.42% higher when compared to properties that are Apartments.

In a similar way as above, we can interpret the change in price for all property types in comparison with property type Apartment.

From the estimated value of Location Brooklyn, NY, we can infer that, the price of property which are located in Brooklyn are 58.78% higher when compared to properties that are located in Bronx.

In a similar way as above, we can interpret the change in price for all properties at different locations in comparison with properties in location Bronx.

From the estimated value of rating3, we can infer that, the price of property which has a rating 3 are 15.10% lesser when compared to properties that have rating 2.

In a similar way as above, we can interpret the change in price for all properties having rating 4 and 5 in comparison with the properties with rating 2.

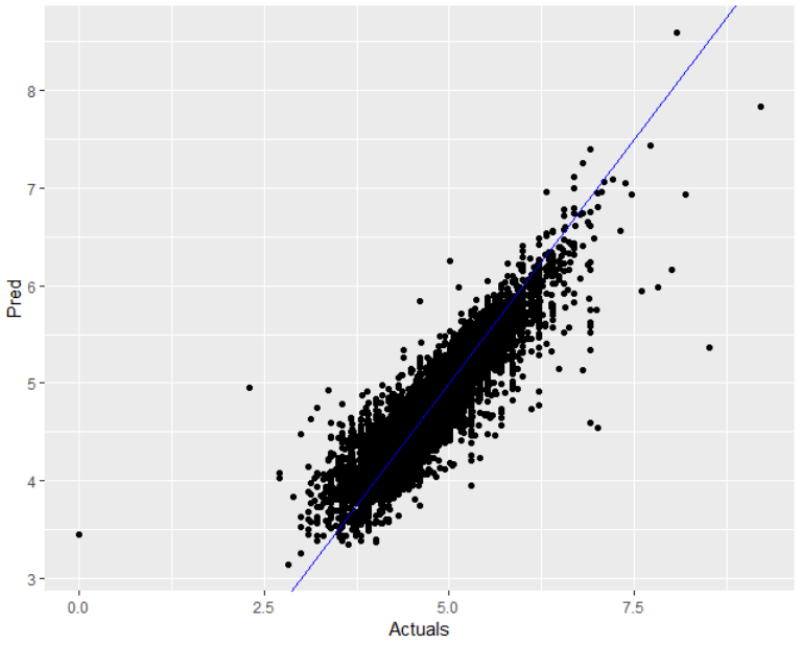
Accuracy of Linear Regression model:

```
> CorAcc
           Actuals      Pred
Actuals 1.0000000 0.9062844
Pred    0.9062844 1.0000000
```

From the correlation table above, we can infer that the accuracy of our predicted model is **90.6%**

Below is the scatter plot for the actual and predicted values of our regression model. From the graph, we can infer that, since all values are positioned very close to the regression line (blue diagonal line), our actual and predicted values are quite similar. Thus, our linear regression model is a good fit for predicting Price.

The plot of actual vs predicted values



Logistic Regression:

```
Call:
glm(formula = def_form2, family = "binomial", data = final.train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.9237  -0.4536  -0.1026   0.4951   4.3197

Coefficients:
                                 Estimate    Std. Error z value             Pr(>|z|)
(Intercept)                 -1968.4784315    85.4705664 -23.031 < 0.0000000000000002 ***
host_is_superhostt              0.5098948     0.0637962   7.993  0.00000000000000132 ***
latitude                       10.7661445     0.9190204  11.715 < 0.0000000000000002 ***
longitude                     -20.7114013     0.9518738 -21.759 < 0.0000000000000002 ***
property_typeBed & Breakfast    1.3579601     0.4024145   3.375             0.000739 ***
property_typeBoutique hotel     0.9570614     1.0395049   0.921             0.357212
property_typeBungalow           1.5219734     1.8487328   0.823             0.410365
property_typeCondominium        1.0280620     0.1946167   5.282  0.00000012743498180 ***
property_typeDorm              -0.9758069     1.3675913  -0.714             0.475523
property_typeGuest suite        0.5009073     0.5422278   0.924             0.355593
property_typeGuesthouse        -1.3986983     1.1570461  -1.209             0.226719
property_typeHostel            -9.9378464   111.5057744  -0.089             0.928983
property_typeHouse             -0.0635606     0.0934909  -0.680             0.496594
property_typeLoft               0.8168571     0.1462209   5.586  0.00000002317436010 ***
property_typeOther             -0.0190341     0.2823365  -0.067             0.946250
property_typeTimeshare          3.9810414     0.5459479   7.292  0.00000000000030543 ***
property_typeTownhouse          0.4697270     0.1757278   2.673             0.007517 **
property_typeVacation home      1.2506767     1.5017972   0.833             0.404965
property_typeVilla              1.3628922     0.9353062   1.457             0.145072
room_typePrivate room          -3.0473809     0.0558832 -54.531 < 0.0000000000000002 ***
room_typeShared room           -4.0072459     0.2702591 -14.827 < 0.0000000000000002 ***
accommodates                    0.3529408     0.0189685  18.607 < 0.0000000000000002 ***
bathrooms                       0.2392445     0.0674894   3.545             0.000393 ***
bedrooms                        0.5337259     0.0374543  14.250 < 0.0000000000000002 ***
guests_included                 0.2180739     0.0260439   8.373 < 0.0000000000000002 ***
minimum_nights                 -0.0129620     0.0030715  -4.220  0.00002442877121699 ***
availability_30                -0.2238519     0.0049092 -45.599 < 0.0000000000000002 ***
number_of_reviews              -0.0023055     0.0007055  -3.268             0.001084 **
is_business_travel_readyt       0.6139816     0.0734097   8.364 < 0.0000000000000002 ***
require_guest_profile_picturet  0.1604567     0.1085547   1.478             0.139376
reviews_per_month              -0.0451865     0.0166417  -2.715             0.006622 **
Family.Kid_Friendly1           -0.0775389     0.0402391  -1.927             0.053985 .

LocationBrooklyn, NY                -1.8663787     0.2131875  -8.755 < 0.0000000000000002 ***
LocationDowntown, Manhattan, NY     -3.7452549     0.2093105 -17.893 < 0.0000000000000002 ***
LocationMidtown, Manhattan, NY      -3.8282373     0.1860422 -20.577 < 0.0000000000000002 ***
LocationQueens, NY                   0.1112775     0.1877826   0.593             0.553458
LocationRoosevelt Island, NY        -1.1665740     0.4848957  -2.406             0.016136 *
LocationStaten Island               -3.0891015     0.4168714  -7.410  0.00000000000012611 ***
LocationUptown, Manhattan, NY       -2.5801584     0.1670909 -15.442 < 0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 31578  on 23652  degrees of freedom
Residual deviance: 15951  on 23614  degrees of freedom
AIC: 16029

Number of Fisher Scoring iterations: 12
```

Interpretation from summary of logistic regression:

Odds of the owner being successful when he is a super host is 1.6651 times higher than when he is not a super host.

Odds of the owner being successful with property type providing bed and breakfast is 3.8 times higher than the Apartment.

Odds of the owner being successful with the property type Condominium is 2.79 times higher than the type Apartment.

Odds of the owner being successful with the property type Loft is 2.26 times higher than the type Apartment.

Odds of the owner being successful with the property type Timeshare is 53.57 times higher than the type Apartment.

Odds of the owner being successful with the property type Townhouse is 1.59 times higher than the type Apartment.

Odds of the owner being successful with the room_typePrivate room is 0.047 times lower than the type Apartment.

Odds of the owner being successful with the room_typeShared room is 0.018 times lower than the type Apartment.

When “accommodates” go up by one unit odds of owner being successful goes up by 1.42 times.

When “bathrooms” go up by one unit odds of owner being successful goes up by 1.27 times.

When “bedrooms” go up by one unit odds of owner being successful goes up by 1.71 times.

When “guests_included” go up by one unit odds of owner being successful goes up by 1.24 times.

When “minimum_nights” go up by one unit odds of owner being successful goes up by 0.99 times.

When “availability_30” go up by one-unit odds of owner being successful goes up by 0.80 times.

When “number of reviews” go up by one-unit odds of owner being successful goes up by 1 times.

Odds of the owner being successful with a property that is business travel ready is 1.85 times higher than properties that are not business travel ready.

Odds of the owner being successful with property in the location, Brooklyn is 0.15 times lower than property in location, Bronx.

Odds of the owner being successful in Downtown, Manhattan is 0.02 times lower than Bronx.

Odds of the owner being successful in Midtown, Manhattan is 0.02 times lower than Bronx.

Odds of the owner being successful in Roosevelt Island is 0.31 times lower than Bronx.

Odds of the owner being successful in Staten Island is 0.05 times lower than Bronx.

Odds of the owner being successful in Uptown, Manhattan is 0.08 times lower than Bronx.

```
> LogModel.pred <- predict(LogModel, final.valid[,-59],type = "response")
> predict<-ifelse(LogModel.pred > 0.5,1,0)
> confusionMatrix(table(Predicted = predict, Actual = final.valid$success_based_on_revenue),positive = '1')
Confusion Matrix and Statistics

         Actual
Predicted    0    1
        0 5340  676
        1  825 3295

               Accuracy : 0.8519
                 95% CI : (0.8448, 0.8588)
    No Information Rate : 0.6082
    P-Value [Acc > NIR] : < 0.00000000000000022

                  Kappa : 0.6913
 Mcnemar's Test P-Value : 0.0001334

            Sensitivity : 0.8298
            Specificity : 0.8662
         Pos Pred Value : 0.7998
         Neg Pred Value : 0.8876
             Prevalence : 0.3918
         Detection Rate : 0.3251
   Detection Prevalence : 0.4065
      Balanced Accuracy : 0.8480

       'Positive' Class : 1
```

Accuracy of Logistic Regression model:

The accuracy of our logistic regression model is calculated using a confusion matrix. From the confusion matrix, we can interpret that, 5340 0's are classified correctly by our model and 676 0's are misclassified as 1. 3295 1's are classified correctly and 825 1's are misclassified as 0.

Accuracy of the model is 85.19% which is calculated by taking the ratio of the number of correctly classified classes over the total number of classes from the confusion matrix.

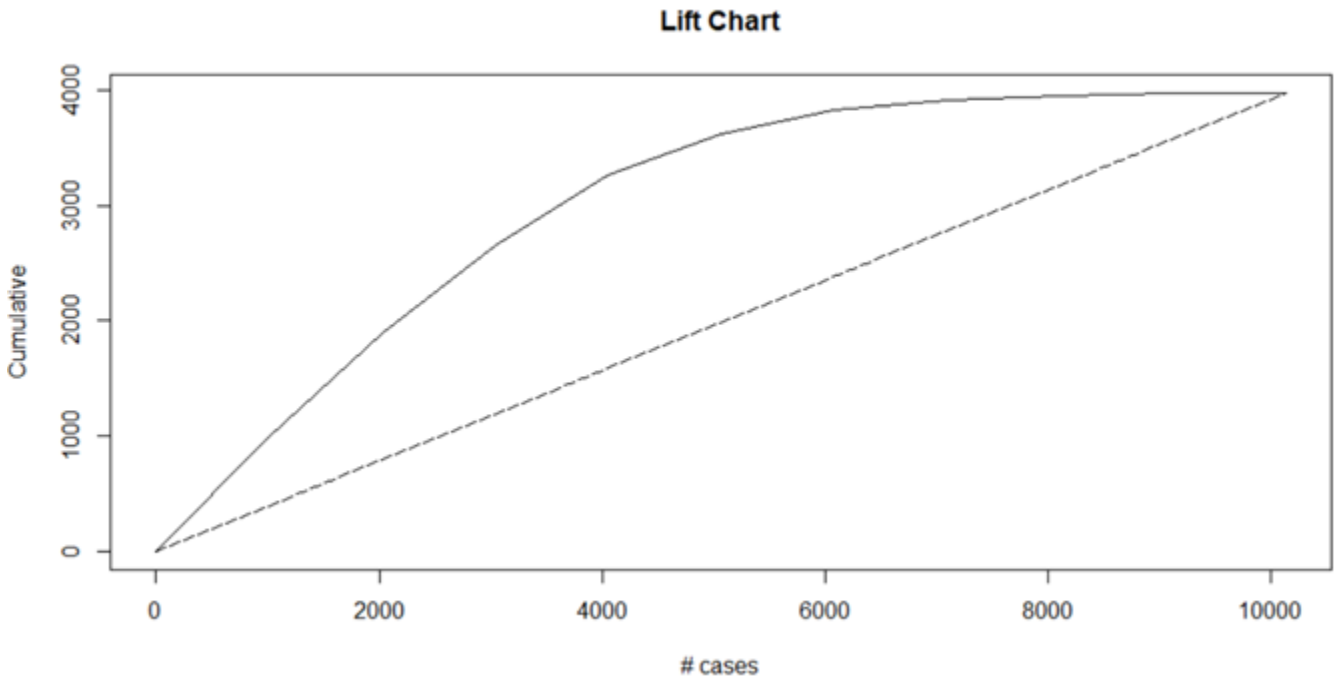
Specificity of the model is 86.62%. It is the proportion of the negatives (0's) that were correctly classified as 0 by our model.

Sensitivity of the model is 82.98%. It is the proportion of the positives (1's) that were correctly classified as 1 by our model.

Graphical representation of the accuracy of logistic regression model is represented with a lift chart and a decile chart.

Lift Chart:

```
### Plot Lift Chart
plot(c(0,gain$cume.pct.of.total*sum(success_num))~c(0,gain$cume.obs),
     xlab = "# cases", ylab = "Cumulative", main = "Lift Chart", type = "l")
lines(c(0,sum(success_num))~c(0, dim(final.valid)[1]), lty = 5)
```



From the lift chart above, it can be inferred that, the cumulative success based on revenue is around 800 for 2000 cases for the baseline model.

The curve above the baseline model shows that, for 2000 cases, the model's cumulative success based on revenue is around 1800.

To calculate how well our model performs in comparison with the baseline model, we take the ratio,

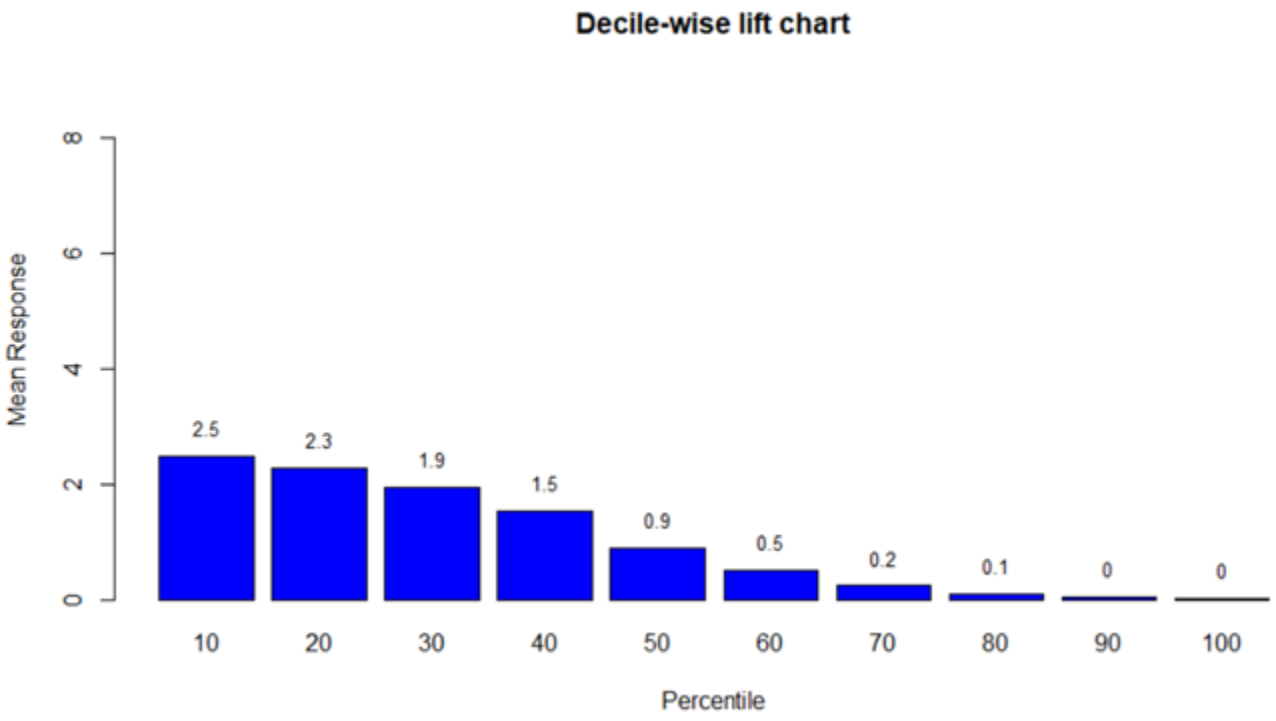
Cumulative success based on revenue of predicted model / Cumulative success based on baseline model.

=> 1800/800 = 2.25

Thus, the developed logistic regression model predicts 2.25 times more accurately than the baseline model.

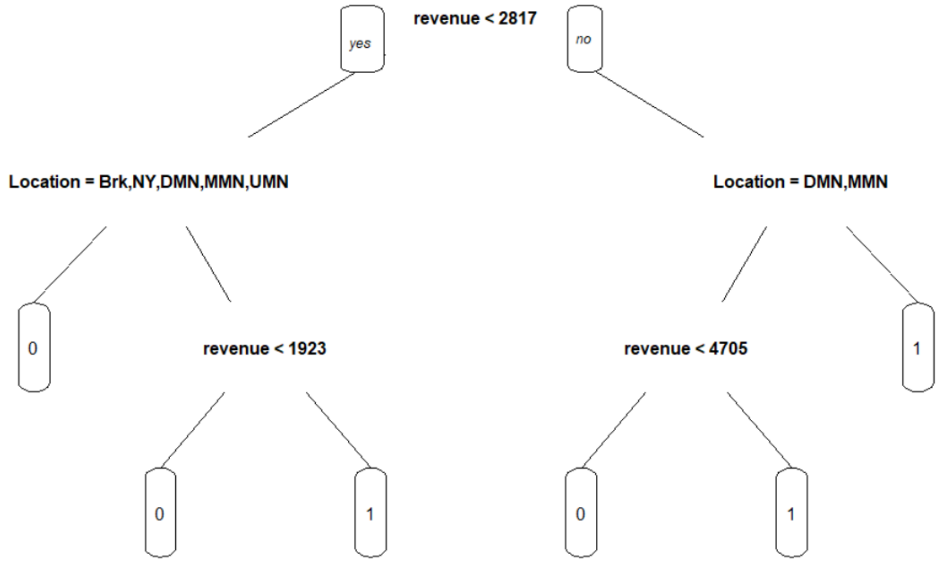
Decile Chart:

```
### Plot decile-wise chart
heights <- gain$mean.resp/mean(success_num)
midpoints <- barplot(heights, names.arg = gain$depth,  ylim = c(0,9), col = "blue",
                     xlab = "Percentile", ylab = "Mean Response",
                     main = "Decile-wise lift chart")
### add labels to columns
text(midpoints, heights+0.5, labels=round(heights, 1), cex = 0.8)
```



For plotting the decile chart, the data is first sorted in descending order. The sorted data is then plotted by taking the data in percentiles of 10. By the definition of decile chart, the mean response of the baseline model is 1. It can be interpreted from the decile chart above that, for the first 10 percentile, the developed logistic regression model predicts 2.5 times better than the baseline model.

Decision Tree



The model has chosen revenue as the root node, as it would have the highest information gain.

Case 1: Owner should be successful with revenue less than 2817.

For the case above, from the decision tree, it can be inferred that, revenue should be less than 2817 which is the left part of the tree. Then, if the location is either Brooklyn, Downtown, Manhattan, Midtown Manhattan or uptown Manhattan, then the tree is again split on the condition if revenue is less than 1923.If location is not one of the above then the host is not successful. If the property is in one of the above locations and revenue is greater than 1923, then the host is found to be successful. Else if the revenue is lesser than 1923 then the host is not successful.

In a similar way, we can interpret if the host is successful when the revenue is greater than 2817 (right side branches of the tree)

Accuracy of Decision Tree Model:

```
Confusion Matrix and Statistics


      0    1
0 6141   24
1   83 3888

               Accuracy : 0.9894
                 95% CI : (0.9873, 0.9913)
    No Information Rate : 0.614
    P-Value [Acc > NIR] : < 0.00000000000000022

                  Kappa : 0.9778
 Mcnemar's Test P-Value : 0.00000002058

            Sensitivity : 0.9939
            Specificity : 0.9867
         Pos Pred Value : 0.9791
         Neg Pred Value : 0.9961
             Prevalence : 0.3860
         Detection Rate : 0.3836
   Detection Prevalence : 0.3918
      Balanced Accuracy : 0.9903

       'Positive' Class : 1
```

The accuracy of the decision tree model is calculated from the confusion matrix. From the confusion matrix, we can interpret that, 6141 0's are classified correctly by our model and 24 0's are misclassified as 1. 3888 1's are classified correctly and 83 1's are misclassified as 0.

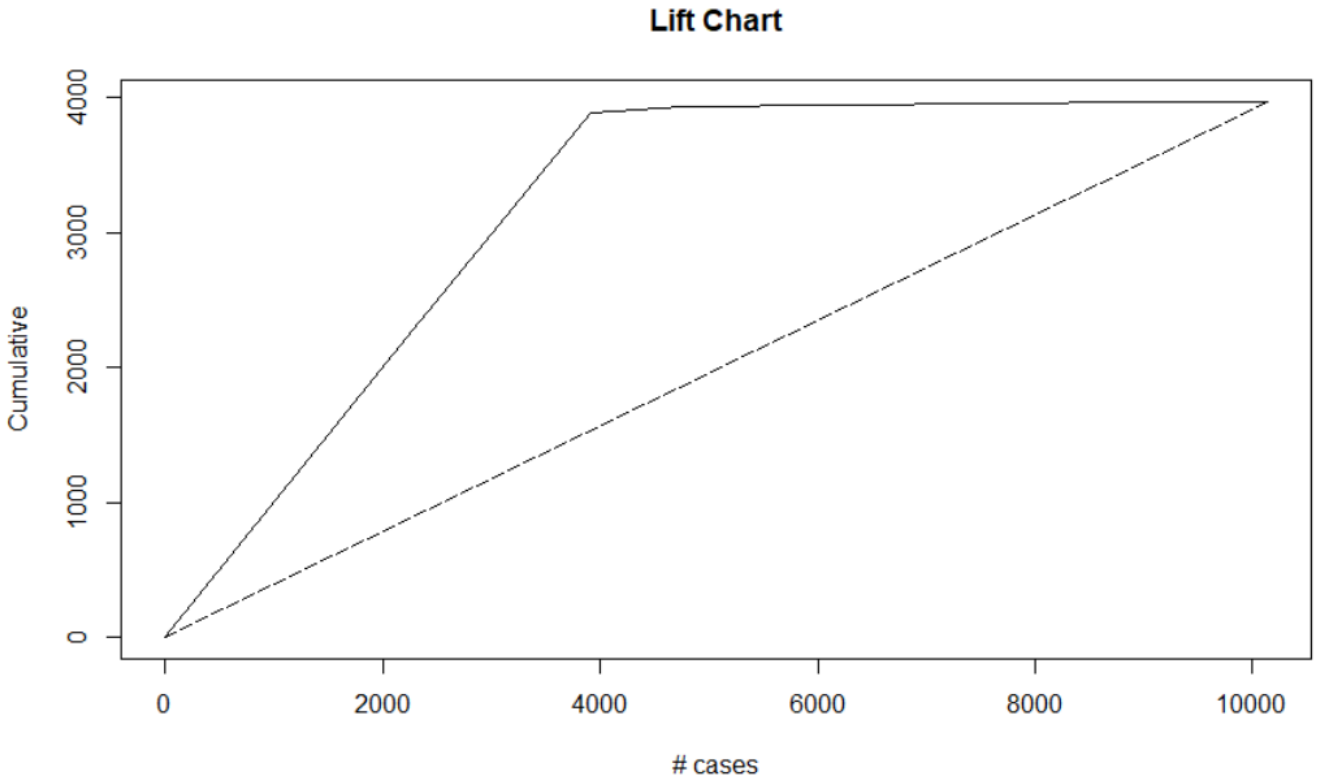
Accuracy of the model is 98.94% which is calculated by taking the ratio of the number of correctly classified classes over the total number of classes from the confusion matrix.

Specificity of the model is 98.67%. It is the proportion of the negatives (0's) that were correctly classified as 0 by our model.

Sensitivity of the model is 99.39%. It is the proportion of the positives (1's) that were correctly classified as 1 by our model.

Graphical representation of the accuracy of Decision Tree model is represented with a lift chart.

Lift Chart:



From the lift chart above, it can be inferred that, the cumulative success based on revenue is around 800 for 2000 cases for the baseline model.

The curve above the baseline model shows that, for 2000 cases, the model's cumulative success based on revenue is around 2000.

To calculate how well our model performs in comparison with the baseline model, we take the ratio,

Cumulative success based on revenue of predicted model / Cumulative success based on baseline model.

=> 2000/800 = 2.5

Thus, the developed logistic regression model predicts 2.5 times more accurately than the baseline model.

Recommendations/Suggestions:

Linear Regression:

Taking into consideration the future Airbnb host were willing to sublet their property with Airbnb, by using our linear predictive model we can suggest them the prices they can quote for their property, depending on their property type, room type, amenities and other attributes of the rentals.

Logistic Regressions:

We can suggest the Airbnb host that for a particular property to be successful he must focus on a property type which is providing bed & breakfast, loft, timeshare, townhouse, private room and business travel amenities. And also if the host is considering the location then the host should prefer Bronx in comparison to Roosevelt island because it has more chances of being successful. Thus, using our logistic predictive model we can relevantly suggest the Airbnb host what different parameters they should consider for increasing the probability for the success of their property.

Decision Tree:

From the model, we can recommend the owner to sublet a property in the following locations for him/her to be successful: Brooklyn, New York; Downtown, Manhattan; Uptown, Manhattan; Midtown, Manhattan.

The link for our code:

https://drive.google.com/open?id=1_94fu02vUGPNuCoZGcFV5hK3TmBFD2P1

Sources and Citations:

Parul Pandey. August 2018. "A Guide to Machine Learning in R for Beginners: Logistic Regression". Analytics Vidhya..

https://medium.com/analytics-vidhya/a-guide-to-machine-learning-in-r-for-beginners-part-5-4c00f2366b90

Sanchit Minocha. March 2017. "Linear Regression. Back to Basics". Data Science Group, IITR.

https://medium.com/data-science-group-iitr/linear-regression-back-to-basics-e4819829d78b

Chirag Sehra. January 2018. "Decision Trees Explained Easily". Medium.

https://medium.com/@chiragsehra42/decision-trees-explained-easily-28f23241248

Shireen Elhabian, Aly A. Farag. September 2009. "A Tutorial on Data Reduction". Elhabian_LDA09.

http://www.sci.utah.edu/~shireen/pdfs/tutorials/Elhabian_LDA09.pdf

Personal Report

It was indeed an amazing learning experience by working on such an extensive project on Airbnb rental dataset and it was indeed a great exposure to program with R. All the major concepts learnt in the course throughout the semester such as linear regression, logistic regression, Data Visualization,Linear discriminant analysis were applied in this project. The data we have taken was from website Kaggle and it was about Airbnb rental information. We are trying to find out how the host will be successful based on various parameters. There were few challenges that we faced with the data. The data which we downloaded from Kaggle was not a cleaned data. So our first challenge was to clean the data. The data had many null values and blank or empty cells. So our first aim was to removing or substituting the null values with some data. For example, we had a column named amenities which had multiple values in a single cell. So, our first task was to clean the amenities column and then we performed the data cleaning on all the other columns. The next challenge was to apply which algorithm on our data. So, with discussion with the group members we decided to go with the algorithms such as Linear regression, Logistic regression, Linear discriminant analysis and decision tree as very elaborately explained as we discussed above in the main report.

Once we started coding for the algorithms we faced few challenges such as unexpected errors in the code which we eventually resolved. We learnt many things from this project. Along with the technical coding skills we also learnt the soft skills such as working with each other in a team, adjusting with each other with their strengths and weaknesses. As far as technical skills are concerned our concepts of R programming and business analytics were developed and now I feel much more confident with doing Business analytics using R.

Chinmay was very much familiar with R programming so it was very easy and comfortable for our entire group to work with him.I worked on the data visualization interpretation. I, Chinmay and Pulkit worked on linear regression and logistic regression. Yanhong did the Linear discriminant analysis part. I and Sowmya also did majority of the group project report along with the small inputs of other 3 group members. Data cleaning was done collectively by all five of us.

Our findings and results were as expected and finally we were able to reach the completion of the report with recommendations or suggestions to the host of Airbnb.

Again, it was a great working experience with all the members of my team.