

Airbnb Seattle Market, An Exploratory & Predictive Analysis

Market Assigned to the team : SEATTLE

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

	Names of the signed team members
Contact Member	Shweta Malla
Team Member 1	Ankita Mittal
Team Member 2	Umang Patel
Team Member 3	Avinav Dixit
Team Member 4	Kumar Saket

```
library(stringr)
library(tidyr)
library(mice)

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

##
## Attaching package: 'mice'

## The following objects are masked from 'package:base':
##
##      cbind, rbind

library(dplyr)

##
## Attaching package: 'dplyr'

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

```

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

library(mice)
library(randomForest)

## Warning: package 'randomForest' was built under R version 3.6.3
## randomForest 4.6-14

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

##
## Attaching package: 'randomForest'

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

library(tidyverse)

## -- Attaching packages ----- tidyverse 1.
3.0 --

## v ggplot2 3.2.1      v purrr 0.3.3
## v tibble 2.1.3      v forcats 0.4.0
## v readr 1.3.1

## -- Conflicts ----- tidyverse_conflict
s() --
## x randomForest::combine() masks dplyr::combine()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag() masks stats::lag()
## x ggplot2::margin() masks randomForest::margin()

library(tidymodels)

## Registered S3 method overwritten by 'xts':
## method from
## as.zoo.xts zoo

## -- Attaching packages ----- tidymodels 0.
0.3 --

## v broom 0.5.3      v recipes 0.1.9
## v dials 0.0.4      v rsample 0.0.5
## v infer 0.5.1      v yardstick 0.0.4
## v parsnip 0.0.5

## -- Conflicts ----- tidymodels_conflict
s() --
## x randomForest::combine() masks dplyr::combine()

```

```
## x scales::discard()      masks purrr::discard()
## x dplyr::filter()        masks stats::filter()
## x recipes::fixed()       masks stringr::fixed()
## x dplyr::lag()            masks stats::lag()
## x dials::margin()         masks ggplot2::margin(), randomForest::margin()
## x yardstick::spec()       masks readr::spec()
## x recipes::step()         masks stats::step()
## x recipes::yj_trans()     masks scales::yj_trans()

library(caret)

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following objects are masked from 'package:yardstick':
##
##   precision, recall

## The following object is masked from 'package:purrr':
##
##   lift

library(skimr)
library(plotly)

##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##   last_plot

## The following object is masked from 'package:stats':
##
##   filter

## The following object is masked from 'package:graphics':
##
##   layout

library(leaflet)

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

library(widgetframe)

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

## Loading required package: htmlwidgets

library(bs4Dash)
```

```
## Warning: package 'bs4Dash' was built under R version 3.6.3
##
## Attaching package: 'bs4Dash'
## The following object is masked from 'package:graphics':
##
##      box
library(Imap)
##
## Attaching package: 'Imap'
## The following object is masked from 'package:purrr':
##
##      imap
library(tidytext)
library(wordcloud)
## Warning: package 'wordcloud' was built under R version 3.6.3
## Loading required package: RColorBrewer
library(lubridate)
##
## Attaching package: 'lubridate'
## The following object is masked from 'package:base':
##
##      date
library(tsibble)
## Warning: package 'tsibble' was built under R version 3.6.3
##
## Attaching package: 'tsibble'
## The following objects are masked from 'package:lubridate':
##
##      interval, new_interval
## The following object is masked from 'package:dplyr':
##
##      id
library(tmap)
## Warning: package 'tmap' was built under R version 3.6.3
library(tm)
```

```
## Warning: package 'tm' was built under R version 3.6.3
## Loading required package: NLP
##
## Attaching package: 'NLP'
## The following object is masked from 'package:ggplot2':
##
##      annotate
library(RColorBrewer)
```

Executive Summary

All of the research and analysis done for this project is with the aim of building a case to present to investors to make them understand most profitable locations to buy properties for Airbnb's in Seattle. We have done this by running various exploratory and predictive analysis models. For data exploration we have various visualizations like box plots, bar graphs, maps and tables to understand relations between different variables and high booking rate. For predictive analytics we have run kNN, Random Forest, XG Boost and Logistic regression with XG Boost giving us the best accuracy. Through our analysis, we figured out what factors attract the most number of customers to present to the investors.

Research Questions

What are the popular times to join Airbnb among the hosts? According to our research on the city and the dataset, there has been a steady increase in apartment construction due to rising rental demands since 2010. This has led to a large number of hosts joining the lucrative business since then. A trend can be seen from the dataset wherein maximum number of hosts (about 19%) have joined Airbnb in the months of August to December. Our analysis suggests that this time is ideal for an establishment to be set up and gain enough traction to cater to a huge crowd which prefer the months of June to August to visit the city. Events like the Seattle International Film Festival (May-June), Seattle Pride (June), Seattle International Beer Fest (July) and Seattle Art Fair (August) attract a large audience from all over the country. Chilly winds and a lot of rainy days make this city almost unvisitable. Despite this, the city has a lot to offer, which led us to believe that there has to be a popular time when the weather is pleasant, which will see a surge in the tourist numbers. Accordingly, the hosts should have a favourite time to enlist their establishment with the company.

What effects price of the establishment in the city? In order to find factors affecting the price of an establishment in the city we built a number of models and compared the Root Mean Square Error to choose the best one. Using our domain knowledge gained by the research we did on the city and its establishments, variables like bathrooms, bed type, bedrooms, host verification, superhost, host listings count, maximum nights, amenities count, host response time, instant bookable, number of beds were taken as important variables. Using these factors we built a linear regression model, a KNN model and a Random Forest model with the last one giving the best analysis for the chosen variables.

The model concluded that these variables were statistically significant in deciding the price of an establishment. It is all about the money! The price of an establishment will ultimately decide how much profit the establishment will make. Therefore, we took this to be an important factor in deciding whether to invest in an establishment. Given all other factors about an establishment are known, the predicted price can help an investor calculate the worth of the property in the future.

What are the best neighbourhoods in Seattle to invest in? Neighbourhood plays an important role in the value of any establishment. Though our analysis we found this to be true for this city as well. Certain neighbourhoods had a noticeably larger number of establishment with high booking rates. Further their availability for 30, 60, 90 and 365 days were very low. Combining both analysis led us to believe that these neighbourhoods including Capitol Hill, First Hill, Belltown, Queen Anne and Madison Valley are particularly popular among the tourists. Further Capitol Hill also have some of the most expensive rentals (about 13%) which have a high booking rate. Expanding on the first question, as the festivals are held in certain locations, we considered location to be a deciding factor for an establishment to perform well. Building on this notion we considered some of the most popular establishments with high rents and plotted a map of the popular areas. We reached to the conclusion that for Seattle, location of the property was an important factor to decide if it will be profitable.

Can COVID-19 be a factor in deciding how to keep a property profitable? Data about the highly infected areas in the city was fetched. This allowed us to compare the data against the establishments in the vicinity. These establishments can be used by authorities to be rented for providing quality accommodation to front line workers while keeping their businesses running. This would not only promote the property but also keep the property profitable in the times when businesses are taking huge hits. Economies of all the countries have been effected largely due to COVID-19. Many businesses have shut down and others are suffering gravely. In such harsh times, can we somehow take advantage of the situation? This motivated us to look for a viable solution to deal with the declining profits.

METHODOLOGY

The dataset provided to us was not clean and we had to skim through the data. We had to go through each column and replace the NA values. In some cases we had to replace the missing values with the mean and in some cases with the median whereas in others we had to replace them with a 0 or FALSE, example `host_is_superhost`, `cleaning_fee`. (We performed EDA on various variables to see the relation with higher booking rate. We did a lot of research on the airbnb website as well as through external sources. We used the data dictionary to understand the variables and built our research on that. We used the `skim` and `str` function to understand our variables. We ran regression to see what variables that were significant for us. We then used those in our final model, `xgboost`, which gave us the highest accuracy. We also included some variables which we thought were important after our online research. (For determining whether a property will have a higher booking rate, we put ourselves in the shoes of the customer booking an airbnb property. We confirmed our thinking through online research. After running our model on the entire data on kaggle, we achieved an accuracy of 89% using random forest. We then decided to test the model

only on our Seattle data. We ran Knn, logistic, random forest and xgBoost for our Seattle dataset. The xgboost which used mostly numeric and logical variables, performed the best with an accuracy of 95%. We have used various text mining techniques and predictive models in our data. XGBoost offers several advanced features for model tuning, computing environments and algorithm enhancement. The algorithm efficiently reduces computing time and allocates an optimal usage of memory resources on the system. This model creates a strong model based on weak classifiers. This helps improve the accuracy. We optimized the parameters of this model to our dataset to achieve this.

```
dfAirbnbTrainSeattle<- read_csv("AirbnbSeattleTrain.csv")

## Parsed with column specification:
## cols(
##   .default = col_character(),
##   id = col_double(),
##   high_booking_rate = col_double(),
##   accommodates = col_double(),
##   availability_30 = col_double(),
##   availability_365 = col_double(),
##   availability_60 = col_double(),
##   availability_90 = col_double(),
##   bathrooms = col_double(),
##   bedrooms = col_double(),
##   beds = col_double(),
##   guests_included = col_double(),
##   host_has_profile_pic = col_logical(),
##   host_identity_verified = col_logical(),
##   host_is_superhost = col_logical(),
##   host_listings_count = col_double(),
##   host_since = col_date(format = ""),
##   instant_bookable = col_logical(),
##   is_business_travel_ready = col_logical(),
##   is_location_exact = col_logical(),
##   latitude = col_double()
##   # ... with 16 more columns
## )

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

## Warning: 1 parsing failure.
##   row    col                                expected actual                                file
## 4740 zipcode no trailing characters -4417 'AirbnbSeattleTrain.csv'

dfAirbnbTestSeattle <- read_csv("AirbnbSeattleTest.csv")

## Parsed with column specification:
## cols(
##   .default = col_character(),
##   id = col_double(),
##   accommodates = col_double(),
```

```

## availability_30 = col_double(),
## availability_365 = col_double(),
## availability_60 = col_double(),
## availability_90 = col_double(),
## bathrooms = col_double(),
## bedrooms = col_double(),
## beds = col_double(),
## guests_included = col_double(),
## host_has_profile_pic = col_logical(),
## host_identity_verified = col_logical(),
## host_is_superhost = col_logical(),
## host_listings_count = col_double(),
## host_since = col_date(format = ""),
## instant_bookable = col_logical(),
## is_business_travel_ready = col_logical(),
## is_location_exact = col_logical(),
## latitude = col_double(),
## longitude = col_double()
## # ... with 15 more columns
## )
## See spec(...) for full column specifications.

## Warning: 1 parsing failure.
## row      col      expected actual      file
## 1016 zipcode no trailing characters -4417 'AirbnbSeattleTest.csv'

table(dfAirbnbTrainSeattle$market)

##
##           Los Angeles Other (International)           Seattle
##                1                1                5399

skim(dfAirbnbTrainSeattle)

```

Data summary

Name	dfAirbnbTrainSeattle
Number of rows	5414
Number of columns	66

Column type frequency:

character	30
Date	1
logical	9
numeric	26

Group variables None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
access	1927	0.64	1	1000	0	3059	0
amenities	0	1.00	2	1574	0	4701	0
bed_type	0	1.00	5	13	0	5	0
cancellation_policy	0	1.00	6	27	0	6	0
city	2	1.00	7	16	0	4	0
cleaning_fee	465	0.91	5	7	0	174	0
description	45	0.99	3	1000	0	4766	3
extra_people	0	1.00	5	7	0	45	0
host_about	1462	0.73	1	4134	0	2348	10
host_acceptance_rate	3	1.00	3	3	0	1	0
host_location	9	1.00	2	47	0	172	0
host_neighbourhood	523	0.90	4	25	0	123	0
host_response_rate	3	1.00	2	4	0	35	0
host_response_time	3	1.00	3	18	0	5	0
host_verifications	0	1.00	4	168	0	231	0
house_rules	1247	0.77	4	1000	0	3133	0
interaction	1407	0.74	1	1000	0	3131	0
market	13	1.00	7	21	0	3	0
monthly_price	4978	0.08	7	10	0	164	0
neighborhood_overview	1583	0.71	7	1000	0	3223	0
neighbourhood	1	1.00	4	25	0	79	0
notes	2312	0.57	1	1000	0	2484	0
price	0	1.00	5	9	0	349	0
property_type	0	1.00	4	18	0	27	0
room_type	0	1.00	10	15	0	4	0

security_deposit	926	0.83	5	9	0	62	0
space	1069	0.80	2	100	0	3756	0
				0			
state	1	1.00	2	2	0	2	0
transit	1670	0.69	7	100	0	3279	0
				0			
weekly_price	4828	0.11	7	9	0	212	0


Variable type: Date















skim_variable	n_missing	complete_rate	min	max	median	n_unique
host_since	3	1	2008-03-03	2019-11-20	2015-04-17	1992

Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
host_has_profile_pic	3	1	1.00	TRU: 5406, FAL: 5
host_identity_verified	3	1	0.56	TRU: 3010, FAL: 2401
host_is_superhost	3	1	0.42	FAL: 3135, TRU: 2276
instant_bookable	0	1	0.52	TRU: 2802, FAL: 2612
is_business_travel_ready	0	1	0.00	FAL: 5414
is_location_exact	0	1	0.83	TRU: 4479, FAL: 935
require_guest_phone_verification	0	1	0.07	FAL: 5058, TRU: 356
require_guest_profile_picture	0	1	0.06	FAL: 5081, TRU: 333
requires_license	0	1	1.00	TRU: 5401, FAL: 13

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
id	0	1.00	1101785.41	58340.81	1000048.00	1051123.75	1102283.50	1152303.25	1202090.00	

high_booking_rate	0	1.00	0.39	0.49	0.00	0.00	0.00	1.00	1.00	
accommodates	0	1.00	3.65	2.28	1.00	2.00	3.00	4.00	28.00	
availability_30	0	1.00	12.47	11.21	0.00	0.00	12.00	23.00	30.00	
availability_365	0	1.00	141.19	133.89	0.00	9.00	90.00	278.00	365.00	
availability_60	0	1.00	26.94	21.99	0.00	0.00	28.00	47.00	60.00	
availability_90	0	1.00	43.72	33.93	0.00	0.00	49.00	76.00	90.00	
bathrooms	2	1.00	1.32	0.64	0.00	1.00	1.00	1.50	8.00	
bedrooms	4	1.00	1.38	1.01	0.00	1.00	1.00	2.00	8.00	
beds	3	1.00	1.89	1.47	0.00	1.00	1.00	2.00	17.00	
guests_included	0	1.00	2.03	1.67	1.00	1.00	1.00	2.00	16.00	
host_listings_count	3	1.00	151.10	436.83	0.00	1.00	2.00	10.00	1825.00	
latitude	0	1.00	47.63	0.05	47.50	47.61	47.62	47.66	47.73	
longitude	0	1.00	-122.33	0.03	-122.42	-122.35	-122.33	-122.31	-122.24	
maximum_nights	0	1.00	603.83	532.20	1.00	29.00	1124.00	1125.00	1125.00	

minimum_nights	0	1.00	4.93	13.30	1.00	1.00	2.00	3.00	400.00	
review_scores_accuracy	786	0.85	9.72	0.74	2.00	10.00	10.00	10.00	10.00	
review_scores_checkin	787	0.85	9.80	0.68	2.00	10.00	10.00	10.00	10.00	
review_scores_cleanliness	786	0.85	9.64	0.76	2.00	9.00	10.00	10.00	10.00	
review_scores_communication	786	0.85	9.80	0.69	2.00	10.00	10.00	10.00	10.00	
review_scores_location	788	0.85	9.80	0.58	2.00	10.00	10.00	10.00	10.00	
review_scores_rating	784	0.86	95.08	7.57	20.00	94.00	97.00	99.00	100.00	
review_scores_value	787	0.85	9.51	0.82	2.00	9.00	10.00	10.00	10.00	
square_feet	5152	0.05	698.21	326.59	0.00	580.00	600.00	1000.00	2100.00	
zipcode	99	0.98	98116.34	17.48	98101.00	98104.00	98115.00	98122.00	98199.00	
{randomControl}	0	1.00	124497.35	289.13	124000.00	124244.25	124497.00	124750.00	124999.00	

#Availability_365

#it's clean

```
dfAirbnbTrainSeattle$availability_365 <- as.numeric(dfAirbnbTrainSeattle$availability_365)
```

```
dfAirbnbTestSeattle$availability_365 <- as.numeric(dfAirbnbTestSeattle$availability_365)
```

#amenities_count

```
dfAirbnbTrainSeattle$amenities_count<-
```

```
  sapply(dfAirbnbTrainSeattle$amenities, function(x) length(unlist(strsplit(as.character(x), ','))))
```

```
dfAirbnbTestSeattle$amenities_count<-
  sapply(dfAirbnbTestSeattle$amenities, function(x) length(unlist(strsplit(as
.character(x), ','))))
```

#host_verifications_count

```
dfAirbnbTrainSeattle$host_verification_count <-
  sapply(dfAirbnbTrainSeattle$host_verifications, function(x) length(unlist(s
trsplit(as.character(x), ','))))
```

```
dfAirbnbTestSeattle$host_verification_count <-
  sapply(dfAirbnbTestSeattle$host_verifications, function(x) length(unlist(st
rsplit(as.character(x), ','))))
```

#host_age

```
dfAirbnbTrainSeattle$host_since <- as.Date(dfAirbnbTrainSeattle$host_since)
dfAirbnbTrainSeattle$host_age <- as.integer(difftime(Sys.Date(), dfAirbnbTrai
nSeattle$host_since, units = 'weeks'))
summary(dfAirbnbTrainSeattle$host_age)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      24.0   196.0   264.0   266.7   349.0   635.0         3
```

```
dfAirbnbTrainSeattle$host_age[is.na(dfAirbnbTrainSeattle$host_age)] <- 301.6
```

```
dfAirbnbTestSeattle$host_since <- as.Date(dfAirbnbTestSeattle$host_since)
dfAirbnbTestSeattle$host_age <- as.integer(difftime(Sys.Date(), dfAirbnbTestS
eattle$host_since, units = 'weeks'))
summary(dfAirbnbTestSeattle$host_age)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      25.0   194.0   258.0   264.8   349.0   599.0
```

```
dfAirbnbTestSeattle$host_age[is.na(dfAirbnbTestSeattle$host_age)] <- 301.6
```

#Bathrooms

```
dfAirbnbTrainSeattle$bathrooms <- as.numeric(dfAirbnbTrainSeattle$bathrooms)
dfAirbnbTestSeattle$bathrooms <- as.numeric(dfAirbnbTestSeattle$bathrooms)
summary(dfAirbnbTrainSeattle$bathrooms)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      0.000   1.000   1.000   1.315   1.500   8.000         2
```

```
table(dfAirbnbTrainSeattle$bathrooms)
```

```
##
##      0  0.5    1  1.5    2  2.5    3  3.5    4  4.5    5    8
##      9   19 3988  290  617  218  131  118   12    7    1    2
```

#Bed_Type

```
table(dfAirbnbTrainSeattle$bed_type)
```

```
##
##      Airbed      Couch      Futon Pull-out Sofa      Real Bed
##      10         4         27         19         5354
```

```
dfAirbnbTrainSeattle$bed_type <- as.character(dfAirbnbTrainSeattle$bed_type)
dfAirbnbTrainSeattle$bed_type[dfAirbnbTrainSeattle$bed_type == ''] <- 'Airbed'
```

```
dfAirbnbTrainSeattle$bed_type[dfAirbnbTrainSeattle$bed_type == '100%'] <- 'Airbed'
```

```
dfAirbnbTrainSeattle$bed_type[dfAirbnbTrainSeattle$bed_type == '81%'] <- 'Airbed'
```

```
dfAirbnbTrainSeattle$bed_type <- as.factor(dfAirbnbTrainSeattle$bed_type)
```

```
dfAirbnbTestSeattle$bed_type <- as.character(dfAirbnbTestSeattle$bed_type)
```

```
dfAirbnbTestSeattle$bed_type[dfAirbnbTestSeattle$bed_type == ''] <- 'Airbed'
```

```
dfAirbnbTestSeattle$bed_type[dfAirbnbTestSeattle$bed_type == '100%'] <- 'Airbed'
```

```
dfAirbnbTestSeattle$bed_type[dfAirbnbTestSeattle$bed_type == '81%'] <- 'Airbed'
```

```
dfAirbnbTestSeattle$bed_type <- as.factor(dfAirbnbTestSeattle$bed_type)
```

#bedrooms

#Mean = 1.36

```
dfAirbnbTrainSeattle$bedrooms <- as.numeric(dfAirbnbTrainSeattle$bedrooms)
```

```
dfAirbnbTrainSeattle$bedrooms[is.na(dfAirbnbTrainSeattle$bedrooms)] <- 3.5
```

```
summary(dfAirbnbTrainSeattle$bedrooms)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.000  1.000  1.000  1.383  2.000  8.000
```

```
table(dfAirbnbTrainSeattle$bedrooms)
```

```
##
##      0      1      2      3  3.5      4      5      6      7      8
## 670 3000 1058  468    4  145   51    9    8    1
```

```
dfAirbnbTestSeattle$bedrooms <- as.numeric(dfAirbnbTestSeattle$bedrooms)
```

```
dfAirbnbTestSeattle$bedrooms[is.na(dfAirbnbTestSeattle$bedrooms)] <- 3.5
```

#beds

```
summary(dfAirbnbTrainSeattle$beds)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
##      0.000  1.000  1.000  1.893  2.000  17.000     3
```

```
dfAirbnbTrainSeattle$beds <- as.numeric(dfAirbnbTrainSeattle$beds)
```

```
table(dfAirbnbTrainSeattle$beds)
```

```
##
##      0      1      2      3      4      5      6      7      8      9     10     11     12     13     14
15
##    97 2757 1375   659   263   117    65    27    13     2    10     3    18     1     2
1
##    17
##     1

dfAirbnbTestSeattle$beds <- as.numeric(dfAirbnbTestSeattle$beds)

#host_identity_Verified

table(dfAirbnbTrainSeattle$host_identity_verified)

##
## FALSE  TRUE
##  2401   3010

dfAirbnbTrainSeattle$host_identity_verified <- as.character(dfAirbnbTrainSeattle$host_identity_verified)
dfAirbnbTrainSeattle$host_identity_verified[dfAirbnbTrainSeattle$host_identity_verified == ''] <- 'TRUE'
dfAirbnbTrainSeattle$host_identity_verified[dfAirbnbTrainSeattle$host_identity_verified == '$2,960.00'] <- 'TRUE'
dfAirbnbTrainSeattle$host_identity_verified[dfAirbnbTrainSeattle$host_identity_verified == '$3,200.00'] <- 'TRUE'
dfAirbnbTrainSeattle$host_identity_verified[dfAirbnbTrainSeattle$host_identity_verified == '$8,500.00'] <- 'TRUE'

summary(dfAirbnbTrainSeattle$host_identity_verified)

##      Length      Class      Mode
##      5414 character character

dfAirbnbTrainSeattle$host_identity_verified <- as.factor(dfAirbnbTrainSeattle$host_identity_verified)

levels(dfAirbnbTrainSeattle$host_identity_verified)

## [1] "FALSE" "TRUE"

dfAirbnbTestSeattle$host_identity_verified <- as.character(dfAirbnbTestSeattle$host_identity_verified)
dfAirbnbTestSeattle$host_identity_verified[dfAirbnbTestSeattle$host_identity_verified == ''] <- 'TRUE'
dfAirbnbTestSeattle$host_identity_verified[dfAirbnbTestSeattle$host_identity_verified == '$2,960.00'] <- 'TRUE'
dfAirbnbTestSeattle$host_identity_verified[dfAirbnbTestSeattle$host_identity_verified == '$3,200.00'] <- 'TRUE'
dfAirbnbTestSeattle$host_identity_verified[dfAirbnbTestSeattle$host_identity_verified == '$8,500.00'] <- 'TRUE'
```

```
dfAirbnbTestSeattle$host_identity_verified <- as.factor(dfAirbnbTestSeattle$host_identity_verified)
```

```
#host_is_superhost
```

```
summary(dfAirbnbTrainSeattle$host_is_superhost)
```

```
##      Mode    FALSE      TRUE    NA's  
## logical   3135     2276         3
```

```
dfAirbnbTrainSeattle$host_is_superhost <- as.character(dfAirbnbTrainSeattle$host_is_superhost)  
table(dfAirbnbTrainSeattle$host_is_superhost)
```

```
##  
## FALSE  TRUE  
##  3135  2276
```

```
dfAirbnbTrainSeattle$host_is_superhost <- ifelse(dfAirbnbTrainSeattle$host_is_superhost == 'FALSE', 'FALSE', 'TRUE')  
dfAirbnbTrainSeattle$host_is_superhost <- as.factor(dfAirbnbTrainSeattle$host_is_superhost)
```

```
dfAirbnbTestSeattle$host_is_superhost <- as.character(dfAirbnbTestSeattle$host_is_superhost)  
table(dfAirbnbTestSeattle$host_is_superhost)
```

```
##  
## FALSE  TRUE  
##  1049   755
```

```
dfAirbnbTestSeattle$host_is_superhost <- ifelse(dfAirbnbTestSeattle$host_is_superhost == 'FALSE', 'FALSE', 'TRUE')  
dfAirbnbTestSeattle$host_is_superhost <- as.factor(dfAirbnbTestSeattle$host_is_superhost)
```

```
#host_listings_count
```

```
summary(dfAirbnbTrainSeattle$host_listings_count)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's  
##       0.0      1.0      2.0  151.1   10.0  1825.0         3
```

```
dfAirbnbTrainSeattle$host_listings_count <- as.numeric(dfAirbnbTrainSeattle$host_listings_count)  
dfAirbnbTestSeattle$host_listings_count <- as.numeric(dfAirbnbTestSeattle$host_listings_count)
```



```
#maximum_nights
```

```
summary(dfAirbnbTrainSeattle$maximum_nights)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1.0    29.0   1124.0   603.8  1125.0   1125.0
```

```
dfAirbnbTrainSeattle$maximum_nights <- as.numeric(dfAirbnbTrainSeattle$maximum_nights)
```

```
table(dfAirbnbTrainSeattle$maximum_nights)
```

```
##
##      1      2      3      4      5      6      7      8      9     10     11     12     13     14     15
16      6     17     35     37     56     29    221     39     14     85      5     47      7    228     46
9      17     18     20     21     23     24     25     26     27     28     29     30     31     32     33
35      3      4     42     76      1      2     22      5     17    225     90    347     78      9      3
15     36     37     40     42     44     45     48     50     55     59     60     61     62     63     65
66      1      1     13      3      1     24      1      2      5      3    126      1      5      1      2
1     70     72     78     79     80     85     88     89     90     91     92     93     95     96     99
100     3      1      1      1      2      1      2      7    171      2      2      1      9      1      1
26    120    125    140    150    160    170    180    181    185    186    190    200    210    215    220
240     63      1      1      8      1      3     81      1      1      1      2      8      4      1      1
3     250    270    300    325    330    360    364    365    366    500    555    600    720    730    740
800      1      1     11      1      1      5      4    218      3      2      1      1      1      3      1
1    1000  1095  1100  1118  1120  1123  1124  1125
##      9      1      1      1      1      8     60  2657
```

```
dfAirbnbTrainSeattle$maximum_nights[dfAirbnbTrainSeattle$maximum_nights > 1125] <- 1125
```

```
dfAirbnbTrainSeattle$maximum_nights[is.na(dfAirbnbTrainSeattle$maximum_nights)] <- 668
```

```
dfAirbnbTestSeattle$maximum_nights <- as.numeric(dfAirbnbTestSeattle$maximum_nights)
```

```
table(dfAirbnbTestSeattle$maximum_nights)
```

```
##
##      1      2      3      4      5      6      7      8      9     10     12     13
```

```

14
##      8      6      9     11     13      9     81     10      3     28     14      2
69
##     15     16     17     18     19     20     21     22     24     25     26     27
28
##     21      5      4      1      1     13     22      3      4      4      1      3
83
##     29     30     31     32     33     35     40     45     47     50     55     58
59
##     30    120     33      1      3      7      2      6      1      1      1      1
1
##     60     61     62     65     70     71     75     80     82     89     90     95
100
##     41      1      1      1      1      1      1      1      1      1     56      2
11
##    120    125    150    153    155    160    175    180    198    200    240    246
260
##     20      2      5      1      1      1      1     26      1      4      2      1
1
##    300    330    356    360    365    369   1000   1095   1100   1120   1121   1123   1
124
##      2      1      1      4     82      1      5      1      1      2      2      2
15
##   1125 10000 1e+05
##    863      1      1

dfAirbnbTestSeattle$maximum_nights[dfAirbnbTestSeattle$maximum_nights > 1125]
<- 1125
dfAirbnbTestSeattle$maximum_nights[is.na(dfAirbnbTestSeattle$maximum_nights)]
<- 668

#minimum_nights
dfAirbnbTrainSeattle$minimum_nights<- as.numeric(dfAirbnbTrainSeattle$minimum
_nights)
summary(dfAirbnbTrainSeattle$minimum_nights)

##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.000  1.000   2.000   4.931  3.000 400.000

table(dfAirbnbTrainSeattle$minimum_nights)

##
##      1      2      3      4      5      6      7      8     10     13     14     15     17     19     20
21
## 1841 2081  625  137   99   30   89   4   14    2   18    2    1    1   10
9
##    23    25    28    29    30    31    32   40   45   55   60   80   81   90  120
150
##      1      3     13      2  397      7      1      1      1      1      5      1      1      6      5
1

```

```

## 180 210 345 365 400
## 1 1 1 1 1

dfAirbnbTrainSeattle$minimum_nights[is.na(dfAirbnbTrainSeattle$minimum_nights)] <- 1
dfAirbnbTrainSeattle$minimum_nights[dfAirbnbTrainSeattle$minimum_nights == 0] <- 1
dfAirbnbTrainSeattle$minimum_nights[dfAirbnbTrainSeattle$minimum_nights == 10000000] <- 1

dfAirbnbTestSeattle$minimum_nights<- as.numeric(dfAirbnbTestSeattle$minimum_nights)
summary(dfAirbnbTestSeattle$minimum_nights)

## Min. 1st Qu. Median Mean 3rd Qu. Max.
## 1.000 1.000 2.000 5.216 3.000 365.000

table(dfAirbnbTestSeattle$minimum_nights)

##
## 1 2 3 4 5 6 7 8 9 10 13 14 15 18 20 21 28 30 3
1 55
## 617 710 206 39 41 7 14 1 1 7 1 7 2 1 1 4 2 128
4 2
## 70 108 180 330 365
## 1 1 5 1 1

dfAirbnbTestSeattle$minimum_nights[is.na(dfAirbnbTestSeattle$minimum_nights)] <- 1
dfAirbnbTestSeattle$minimum_nights[dfAirbnbTestSeattle$minimum_nights == 0] <- 1
dfAirbnbTestSeattle$minimum_nights[dfAirbnbTestSeattle$minimum_nights == 10000000] <- 1

#price

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(price = as.numeric(str_replace_all(dfAirbnbTrainSeattle$price, "[$,]", '')))

meanPrice <- mean(dfAirbnbTrainSeattle$price, na.rm = TRUE)

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(price = replace_na(price, meanPrice))

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(price = as.numeric(str_replace_all(dfAirbnbTestS

```

```

eattle$price,"[$,]",''))))

meanPrice <- mean(dfAirbnbTestSeattle$price, na.rm = TRUE)

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(price = replace_na(price, meanPrice))

#dfAirbnbTrainSeattle$high_booking_rate <- dfAirbnbTrainSeattle_train_y$high_
booking_rate
dfAirbnbTrainSeattle$high_booking_rate <- as.factor(dfAirbnbTrainSeattle$high_
_booking_rate)

# Final code for transformation of cleaning fee

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(cleaning_fee = as.numeric(str_replace_all(dfAirb
nbTrainSeattle$cleaning_fee,"[$,]",'')))

meanCleaningFee <- mean(dfAirbnbTrainSeattle$cleaning_fee, na.rm = TRUE)

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(cleaning_fee = replace_na(cleaning_fee, meanCleani
ngFee))

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(cleaning_fee = as.numeric(str_replace_all(dfAirb
nbTestSeattle$cleaning_fee,"[$,]",'')))

meanCleaningFee <- mean(dfAirbnbTestSeattle$cleaning_fee, na.rm = TRUE)

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(cleaning_fee = replace_na(cleaning_fee, meanCleani
ngFee))

# Final code for transformation of extra people
dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(extra_people = as.numeric(str_replace_all(dfAirb
nbTrainSeattle$extra_people,"[$,]",'')))

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(extra_people = as.numeric(str_replace_all(dfAirb
nbTestSeattle$extra_people,"[$,]",'')))

#guests_included
dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(guests_included=as.numeric(str_replace_all(dfAirbnbTrainSeattle$gues
ts_included,"[,%]",'')))

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%

```

```

    mutate(guests_included = ifelse(guests_included%in% c("17", "18", "19", "20",
, "21", "22", "24", "25", "28", "30", "32", "34"), "Other", guests_included))

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(guests_included=as.numeric(str_replace_all(dfAirbnbTestSeattle$guest
s_included,"[,%]", '')))

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(guests_included = ifelse(guests_included%in% c("17", "18", "19", "20",
, "21", "22", "24", "25", "28", "30", "32", "34"), "Other", guests_included))

#host_response_rate
dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(host_response_rate = as.numeric(str_replace_all(
dfAirbnbTrainSeattle$host_response_rate,"[,%]", '')))

## Warning: NAs introduced by coercion

meanRespRate <- mean(dfAirbnbTrainSeattle$host_response_rate, na.rm = TRUE)

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(host_response_rate = replace_na(host_response_rate
, meanRespRate))

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(host_response_rate = as.numeric(str_replace_all(
dfAirbnbTestSeattle$host_response_rate,"[,%]", '')))

## Warning: NAs introduced by coercion

meanRespRate <- mean(dfAirbnbTestSeattle$host_response_rate, na.rm = TRUE)

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(host_response_rate = replace_na(host_response_rate
, meanRespRate))

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(property_type = ifelse(property_type %in% c("Castle", "Igloo", "Tree
house", "Cave", "Dome house", "In-law", "Nature lodge", "Tipi", "Vacation hom
e", "Barn", "Campsite", "Chalet", "Dorm", "Houseboat", "Island", "Bus", "Casa
particular (Cuba)", "Earth house", "Hut", "Lighthouse", "Pension (South Korea
)", "Timeshare", "Train", "Yurt"), "Other", property_type))

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(property_type = ifelse(property_type %in% c("Castle", "Igloo", "Mins
u (Taiwan)", "Treehouse", "Cave", "Dome house", "In-law", "Nature lodge", "Ti
pi", "Tent", "Vacation home", "Barn", "Campsite", "Chalet", "Dorm", "Houseboa
t", "Island", "Bus", "Casa particular (Cuba)", "Earth house", "Hut", "Lightho
use", "Pension (South Korea)", "Timeshare", "Train", "Yurt"), "Other", proper
ty_type))

```

```

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(cancellation_policy = ifelse(cancellation_policy %in% c("luxury_super_strict_125", "luxury_super_strict_95", "luxury_no_refund"), "Other", cancellation_policy))

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(cancellation_policy = ifelse(cancellation_policy %in% c("luxury_super_strict_125", "luxury_moderate", "luxury_super_strict_95", "luxury_no_refund"), "Other", cancellation_policy))

dfAirbnbTestSeattle$property_type <- as.factor(dfAirbnbTestSeattle$property_type)

# Final code for transformation of security deposit
dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(security_deposit = as.numeric(str_replace_all(dfAirbnbTrainSeattle$security_deposit, "[$,]", '')))

meanSecurityDeposit <- mean(dfAirbnbTrainSeattle$security_deposit, na.rm = TRUE)

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(security_deposit = replace_na(security_deposit, meanSecurityDeposit))

dfAirbnbTrainSeattle$bed_type<-as.factor(dfAirbnbTrainSeattle$bed_type)
dfAirbnbTrainSeattle$host_identity_verified<-as.factor(dfAirbnbTrainSeattle$host_identity_verified)
dfAirbnbTrainSeattle$host_is_superhost<-as.factor(dfAirbnbTrainSeattle$host_is_superhost)
dfAirbnbTrainSeattle$market<-as.factor(dfAirbnbTrainSeattle$market)
dfAirbnbTrainSeattle$instant_bookable<-as.factor(dfAirbnbTrainSeattle$instant_bookable)
dfAirbnbTrainSeattle$is_location_exact<-as.factor(dfAirbnbTrainSeattle$is_location_exact)
dfAirbnbTrainSeattle$property_type<-as.factor(dfAirbnbTrainSeattle$property_type)
dfAirbnbTrainSeattle$requires_license<-as.factor(dfAirbnbTrainSeattle$requires_license)

dfAirbnbTrainSeattle$availability_365<- as.factor(dfAirbnbTrainSeattle$availability_365)
dfAirbnbTrainSeattle$bathrooms<-as.factor(dfAirbnbTrainSeattle$bathrooms)
dfAirbnbTrainSeattle$bed_type<-as.factor(dfAirbnbTrainSeattle$bed_type)
dfAirbnbTrainSeattle$host_identity_verified<-as.factor(dfAirbnbTrainSeattle$host_identity_verified)
dfAirbnbTrainSeattle$host_is_superhost<-as.factor(dfAirbnbTrainSeattle$host_is_superhost)
dfAirbnbTrainSeattle$host_listings_count<-as.factor(dfAirbnbTrainSeattle$host_listings_count)

```

```

dfAirbnbTrainSeattle$price<-as.factor(dfAirbnbTrainSeattle$price)
dfAirbnbTrainSeattle$cleaning_fee<-as.factor(dfAirbnbTrainSeattle$cleaning_fee)
dfAirbnbTrainSeattle$market<-as.factor(dfAirbnbTrainSeattle$market)
dfAirbnbTrainSeattle$guests_included<-as.factor(dfAirbnbTrainSeattle$guests_included)
dfAirbnbTrainSeattle$minimum_nights<-as.factor(dfAirbnbTrainSeattle$minimum_nights)
dfAirbnbTrainSeattle$amenities_count<-as.factor(dfAirbnbTrainSeattle$amenities_count)
# dfAirbnbTrainSeattle$host_verification_count<-as.factor(head(dfAirbnbTrainSeattle)$host_verification_count)
dfAirbnbTrainSeattle$host_age<-as.factor(dfAirbnbTrainSeattle$host_age)
dfAirbnbTrainSeattle$host_response_rate<-as.factor(dfAirbnbTrainSeattle$host_response_rate)
dfAirbnbTrainSeattle$instant_bookable<-as.factor(dfAirbnbTrainSeattle$instant_bookable)
dfAirbnbTrainSeattle$is_location_exact<-as.factor(dfAirbnbTrainSeattle$is_location_exact)
dfAirbnbTrainSeattle$market<-as.factor(dfAirbnbTrainSeattle$market)
dfAirbnbTrainSeattle$property_type<-as.factor(dfAirbnbTrainSeattle$property_type)
dfAirbnbTrainSeattle$requires_license<-as.factor(dfAirbnbTrainSeattle$requires_license)

# Final code for transformation of monthly price

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(monthly_price = as.numeric(str_replace_all(dfAirbnbTrainSeattle$monthly_price,"[$,]",'')))

meanMonthly_price <- mean(dfAirbnbTrainSeattle$monthly_price, na.rm = TRUE)

dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(monthly_price = replace_na(monthly_price, meanMonthly_price))

# Final code for transformation of monthly price

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(monthly_price = as.numeric(str_replace_all(dfAirbnbTestSeattle$monthly_price,"[$,]",'')))

meanMonthly_price <- mean(dfAirbnbTestSeattle$cleaning_fee, na.rm = TRUE)

dfAirbnbTestSeattle <- dfAirbnbTestSeattle %>%
  mutate(monthly_price = replace_na(monthly_price, meanMonthly_price))

```

```
dfAirbnbTrainSeattle %>%
  leaflet() %>%
  addProviderTiles(provider = "CartoDB.Positron") %>%
  addCircleMarkers(color = "cyan", radius = 0.5, weight =
0.1) %>%
  setView(zoom = 12, lat = 47.615258, lng = -122.335892)

## Assuming "longitude" and "latitude" are longitude and latitude, respective
ly
```

###The map shows the location of listings in Seattle.As can be seen most of the listings are from central seattle.

```
dfAirbnbTrainSeattle %>%
  group_by(neighbourhood) %>%
  leaflet() %>%
  addTiles() %>%
  addCircleMarkers(clusterOptions = markerClusterOptions(
), label = ~price) %>%
  setView(zoom = 12, lat = 47.615258, lng = -122.335892)

## Assuming "longitude" and "latitude" are longitude and latitude, respective
ly
```

###The map here shows number of listings in the respective neighborhood, and we found the top neighborhoods in terms of proerties listed.

```
dfBestNeighbourhood <- dfAirbnbTrainSeattle %>%
  filter(neighbourhood %in% c("Wallingford", "South Lake Union", "Queen Anne"
, "Minor", "Pike Place Market", "First Hill", "Central Business District", "Capi
tol Hill", "Belltown", "Ballard"))

dfBestNeighbourhood <-
  dfBestNeighbourhood %>%
  mutate(minimum_nights = as.double(minimum_nights),
         weekly_price = as.double(weekly_price),
         availability_365 = as.double(availability_365))

## Warning: NAs introduced by coercion

dfBestNSummary <- dfBestNeighbourhood %>%
  group_by(neighbourhood) %>%
  mutate(meanMonthlyPrice = mean(monthly_price), meanAvailability_30 = mean(a
vailability_30), meanAvailability_60 = mean(availability_60), meanAvailabilit
y_90 = mean(availability_90), meanAvailability_365 = mean(availability_365),
  meanMinimum_nights = mean(minimum_nights),
  meanBeds = mean(beds),
  meanSquareFeet = mean(square_feet),
  meanExtraPeople = mean(extra_people))
```



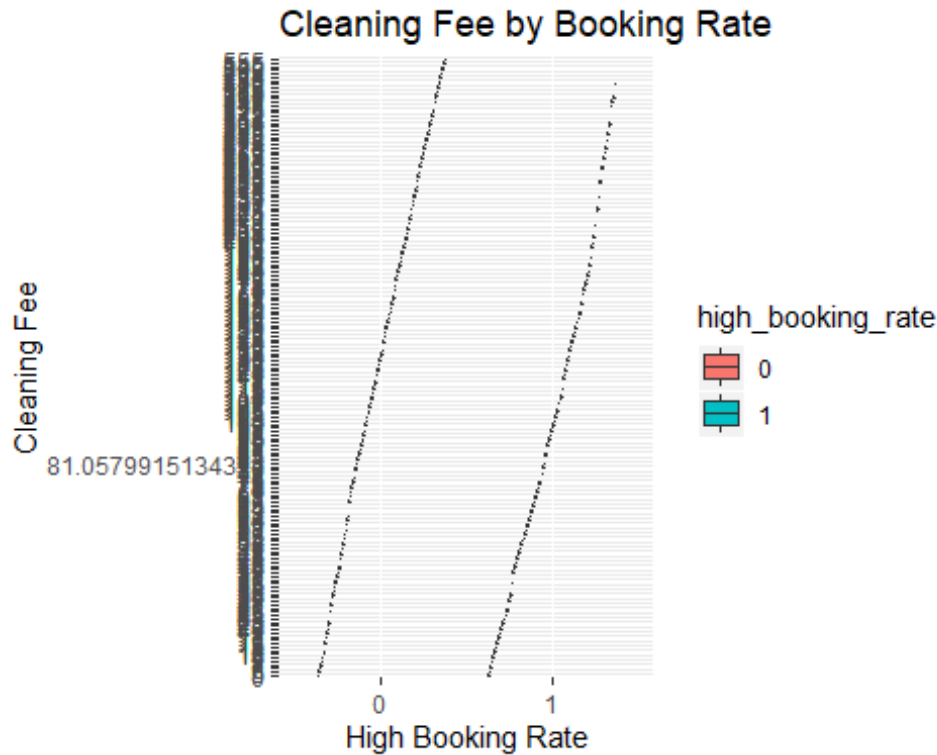
```

dfBestNSummary %>%
  group_by(neighbourhood, meanAvailability_30) %>%
  select(neighbourhood, meanAvailability_30)

## # A tibble: 2,539 x 2
## # Groups:   neighbourhood, meanAvailability_30 [10]
##   neighbourhood      meanAvailability_30
##   <chr>                <dbl>
## 1 First Hill           16.8
## 2 Wallingford          9.86
## 3 Wallingford          9.86
## 4 Capitol Hill        10.3
## 5 Belltown            13.4
## 6 Central Business District 17.6
## 7 Belltown            13.4
## 8 First Hill           16.8
## 9 Belltown            13.4
## 10 Belltown           13.4
## # ... with 2,529 more rows

# Cleaning Fee vs Booking Rate
plot <- dfAirbnbTrainSeattle %>%
  ggplot(aes(high_booking_rate, cleaning_fee, fill = high_booking_rate)) + geom_boxplot() + labs(x = "High Booking Rate", y = "Cleaning Fee", title = "Cleaning Fee by Booking Rate")
plot

```



###It can be seen that as the cleaning fee charges increase which contribute to increase in the rent of the property, the booking rate decreases.

```
#Amenities Frequency
dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(amenities = gsub("[{}\\"]", "", amenities))

tokens <- str_split(as.character(dfAirbnbTrainSeattle$amenities), ',')

amenitiesCountdf <- as.data.frame(table(unlist(tokens))) %>% arrange(desc(Freq))
amenitiesCountdf
```

	Var1	Freq
## 1	Wifi	5296
## 2	Essentials	5294
## 3	Heating	5233
## 4	Smoke detector	5214
## 5	Shampoo	4940
## 6	Kitchen	4783
## 7	Hangers	4763
## 8	Hair dryer	4670
## 9	Carbon monoxide detector	4666
## 10	Laptop friendly workspace	4383
## 11	Iron	4348
## 12	TV	4283
## 13	Washer	4219

## 14	Dryer	4207
## 15	Fire extinguisher	3588
## 16	Hot water	3488
## 17	Refrigerator	3115
## 18	Dishes and silverware	3029
## 19	Self check-in	3017
## 20	Microwave	2950
## 21	Coffee maker	2871
## 22	First aid kit	2700
## 23	Stove	2538
## 24	Cooking basics	2486
## 25	Oven	2459
## 26	Bed linens	2451
## 27	Private entrance	2416
## 28	Free street parking	2352
## 29	Free parking on premises	2212
## 30	Family/kid friendly	2154
## 31	Extra pillows and blankets	2034
## 32	Dishwasher	1903
## 33	Internet	1848
## 34	Long term stays allowed	1630
## 35	Cable TV	1562
## 36	Lock on bedroom door	1556
## 37	No stairs or steps to enter	1530
## 38	Keypad	1497
## 39	Patio or balcony	1483
## 40	Luggage dropoff allowed	1414
## 41	Air conditioning	1360
## 42	Elevator	1227
## 43	Garden or backyard	1208
## 44	Indoor fireplace	1081
## 45	Lockbox	1015
## 46	Safety card	982
## 47	Gym	898
## 48	Bathtub	890
## 49	24-hour check-in	864
## 50	Pets allowed	854
## 51	BBQ grill	830
## 52	Private living room	793
## 53	Pack 'n Play/travel crib	681
## 54	translation missing: en.hosting_amenity_50	676
## 55	Smart lock	605
## 56	Well-lit path to entrance	559
## 57	translation missing: en.hosting_amenity_49	544
## 58	Pets live on this property	532
## 59	Single level home	451
## 60	Breakfast	447
## 61	Children's books and toys	445
## 62	Paid parking off premises	393
## 63	Room-darkening shades	382

## 64	Other	335
## 65	Wide entrance for guests	321
## 66	Host greets you	310
## 67	Hot tub	307
## 68	High chair	302
## 69	Dog(s)	300
## 70	Ethernet connection	298
## 71	Wide hallways	282
## 72	Lake access	275
## 73	Paid parking on premises	272
## 74	Extra space around bed	247
## 75	Flat path to guest entrance	247
## 76	Pool	229
## 77	Wide entrance	223
## 78	Cat(s)	217
## 79	Children's dinnerware	215
## 80	Wide entryway	215
## 81	Wireless intercom	209
## 82	Full kitchen	207
## 83	Bathroom essentials	186
## 84	Bedroom comforts	186
## 85	Accessible-height bed	179
## 86	Bath towel	178
## 87	Body soap	178
## 88	Toilet paper	178
## 89	Buzzer/wireless intercom	176
## 90	Handheld shower head	172
## 91	Suitable for events	165
## 92	Building staff	163
## 93	Babysitter recommendations	159
## 94	Accessible-height toilet	150
## 95	Cleaning before checkout	147
## 96	Wheelchair accessible	130
## 97	Outlet covers	126
## 98	Crib	124
## 99	Central air conditioning	123
## 100	Wide doorway to guest bathroom	121
## 101	Game console	116
## 102	Doorman	115
## 103	Stair gates	95
## 104	Smoking allowed	86
## 105	Beach essentials	76
## 106	Fireplace guards	73
## 107	Outdoor seating	72
## 108	toilet	70
## 109	Wide clearance to shower	70
## 110	Netflix	69
## 111	Smart TV	66
## 112	Hot water kettle	64
## 113	Baby bath	63

## 114	Disabled parking spot	62
## 115	Waterfront	62
## 116	Breakfast table	54
## 117	Pocket wifi	54
## 118	Fixed grab bars for shower	53
## 119	EV charger	52
## 120	Changing table	44
## 121	Gas oven	43
## 122	Walk-in shower	41
## 123	Baby monitor	39
## 124	Memory foam mattress	39
## 125	En suite bathroom	35
## 126	Beachfront	33
## 127	Balcony	32
## 128	Other pet(s)	32
## 129	Step-free shower	30
## 130	Formal dining area	29
## 131	Window guards	29
## 132	Kitchenette	28
## 133	Heated floors	27
## 134	HBO GO	25
## 135	Ceiling fan	24
## 136	Espresso machine	24
## 137	Sound system	24
## 138	Terrace	24
## 139	Pillow-top mattress	23
## 140	Rain shower	22
## 141	Outdoor parking	21
## 142	DVD player	20
## 143	Mountain view	20
## 144	Convection oven	19
## 145	Fire pit	19
## 146	Soaking tub	18
## 147	Mini fridge	17
## 148	Amazon Echo	16
## 149	Exercise equipment	14
## 150	Fixed grab bars for toilet	14
## 151	Day bed	13
## 152	Shower chair	11
## 153	Sun loungers	11
## 154	Table corner guards	11
## 155	Electric profiling bed	10
## 156	Bathtub with bath chair	8
## 157	Shared gym	8
## 158		7
## 159	Firm mattress	7
## 160	Heat lamps	7
## 161	Printer	7
## 162	Private bathroom	7
## 163	Beach view	6

## 164	Shared hot tub	6
## 165	Shared pool	6
## 166	Warming drawer	6
## 167	Ground floor access	5
## 168	Jetted tub	4
## 169	Mudroom	4
## 170	Private hot tub	4
## 171	Wine cooler	4
## 172	Hammock	3
## 173	Private gym	3
## 174	Standing valet	3
## 175	High-resolution computer monitor	2
## 176	Murphy bed	2
## 177	Pool with pool hoist	2
## 178	Projector and screen	2
## 179	Sauna	2
## 180	Stand alone steam shower	2
## 181	Steam oven	2
## 182	Air purifier	1
## 183	Double oven	1
## 184	Heated towel rack	1
## 185	Ski-in/Ski-out	1
## 186	Tennis court	1

###The top 10 amenities provided at the rentals listed with Airbnb, Wifi is provided by the almost all the hosts and luxury amenities like ski-in/ski-out and Tennis court only provided by one host.

#Amenities Word Cloud

```
dfAirbnbTrainSeattle <- dfAirbnbTrainSeattle %>%
  mutate(amenities = gsub("[{}\\"]", "", amenities))
tokens <- str_split(as.character(dfAirbnbTrainSeattle$amenities), ',')
amenitiesCountdf <- as.data.frame(table(unlist(tokens))) %>% arrange(desc(Freq))
```

```
set.seed(123)
```

```
wordcloud(words = amenitiesCountdf$Var1, freq = amenitiesCountdf$Freq, min.freq = 1, max.words=75, random.order=FALSE, rot.per=0.35, colors=brewer.pal(8, "Dark2"))
```

```
## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Smoke detector could not be fit on page. It will
not be
## plotted.
```

```
## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Hair dryer could not be fit on page. It will not
be
## plotted.
```

```
## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Carbon monoxide detector could not be fit on page
. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Laptop friendly workspace could not be fit on pag
e. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Washer could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Dryer could not be fit on page. It will not be pl
otted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Fire extinguisher could not be fit on page. It wi
ll not
## be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Hot water could not be fit on page. It will not b
e
## plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Refrigerator could not be fit on page. It will no
t be
## plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Dishes and silverware could not be fit on page. I
t will
## not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Self check-in could not be fit on page. It will n
ot be
## plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Coffee maker could not be fit on page. It will no
t be
## plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : First aid kit could not be fit on page. It will n
ot be
## plotted.
```

```
## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Cooking basics could not be fit on page. It will
not be
## plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Bed linens could not be fit on page. It will not
be
## plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Private entrance could not be fit on page. It wil
l not
## be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Free street parking could not be fit on page. It
will
## not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Free parking on premises could not be fit on page
. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Family/kid friendly could not be fit on page. It
will
## not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Extra pillows and blankets could not be fit on pa
ge. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Long term stays allowed could not be fit on page.
It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Lock on bedroom door could not be fit on page. It
will
## not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : No stairs or steps to enter could not be fit on p
age.
## It will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Patio or balcony could not be fit on page. It wil
```



```
l not
## be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Luggage dropoff allowed could not be fit on page. It
It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Air conditioning could not be fit on page. It wil
l not
## be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Garden or backyard could not be fit on page. It w
ill
## not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Indoor fireplace could not be fit on page. It wil
l not
## be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Safety card could not be fit on page. It will not
be
## plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : 24-hour check-in could not be fit on page. It wil
l not
## be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Private living room could not be fit on page. It
will
## not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Pack 'n Play/travel crib could not be fit on page
. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : translation missing: en.hosting_amenity_50 could
not be
## fit on page. It will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Well-lit path to entrance could not be fit on pag
e. It
## will not be plotted.
```

```
## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : translation missing: en.hosting_amenity_49 could
not be
## fit on page. It will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Pets live on this property could not be fit on pa
ge. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Single level home could not be fit on page. It wi
ll not
## be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Children's books and toys could not be fit on pag
e. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Paid parking off premises could not be fit on pag
e. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Room-darkening shades could not be fit on page. I
t will
## not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Wide entrance for guests could not be fit on page
. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Paid parking on premises could not be fit on page
. It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Extra space around bed could not be fit on page.
It
## will not be plotted.

## Warning in wordcloud(words = amenitiesCountdf$Var1, freq =
## amenitiesCountdf$Freq, : Flat path to guest entrance could not be fit on p
age.
## It will not be plotted.
```



```
# install.packages("corpus")

library(corpus)

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

# Neighborhood Overview Word Cloud
corpus <- Corpus(VectorSource(dfAirbnbTrainSeattle$neighborhood_overview))
corpus = tm_map(corpus, PlainTextDocument)

## Warning in tm_map.SimpleCorpus(corpus, PlainTextDocument): transformation
drops
## documents

corpus = tm_map(corpus, tolower)

## Warning in tm_map.SimpleCorpus(corpus, tolower): transformation drops docu
ments

corpus = tm_map(corpus, removePunctuation)

## Warning in tm_map.SimpleCorpus(corpus, removePunctuation): transformation
drops
## documents

corpus = tm_map(corpus, removeWords, c("seattle", "please", "will", "use", "'
re one", "guests", "house", "allowed", "keep", "can", "take", "like", "per",
"make", "neighborhood", stopwords("english")))
```

```

## Warning in tm_map.SimpleCorpus(corpus, removeWords, c("seattle", "please",
:
## transformation drops documents

corpus = tm_map(corpus, stripWhitespace)

## Warning in tm_map.SimpleCorpus(corpus, stripWhitespace): transformation dr
ops
## documents

dtm <- TermDocumentMatrix(corpus)
matrix <- as.matrix(dtm)
words <- sort(rowSums(matrix), decreasing=TRUE)
wordCountdf <- data.frame(word = names(words), freq = words)
set.seed(123)
wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, min.freq = 1, ma
x.words=49, rot.per=0.35, random.order=FALSE, colors=brewer.pal(8, "Dark2"))

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
## washington could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
many
## could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
## grocery could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
## capitol could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
## university could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
home
## could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
short
## could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
nearby
## could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
## district could not be fit on page. It will not be plotted.

## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :
easy
## could not be fit on page. It will not be plotted.

```

```
## Warning in wordcloud(words = wordCountdf$word, freq = wordCountdf$freq, :  
parks  
## could not be fit on page. It will not be plotted.
```



```
# Transit Word Cloud  
corpus <- Corpus(VectorSource(dfAirbnbTrainSeattle$transit))  
corpus = tm_map(corpus, PlainTextDocument)  
  
## Warning in tm_map.SimpleCorpus(corpus, PlainTextDocument): transformation  
drops  
## documents  
  
corpus = tm_map(corpus, tolower)  
  
## Warning in tm_map.SimpleCorpus(corpus, tolower): transformation drops docu  
ments  
  
corpus = tm_map(corpus, removePunctuation)  
  
## Warning in tm_map.SimpleCorpus(corpus, removePunctuation): transformation  
drops  
## documents  
  
corpus = tm_map(corpus, removeWords, c("seattle", "please", "will", "use", "re  
one", "guests", "house", "allowed", "keep", "can", "take", "like", "per",  
"make", "neighborhood", stopwords("english")))
```



```

unnest_tokens(word, text)

data(stop_words)
word_cancellation_policy <- word_cancellation_policy %>%
  anti_join(stop_words)

## Joining, by = "word"

count_cancellation_policy <- word_cancellation_policy %>% count(word, sort=TRUE)
count_cancellation_policy

## # A tibble: 6 x 2
##   word                                n
##   <chr>                             <int>
## 1 strict_14_with_grace_period    2104
## 2 moderate                      1905
## 3 flexible                      1192
## 4 strict                        149
## 5 super_strict_30                42
## 6 super_strict_60                22

#to split host_since into year, month and day

dfAirbnbSeattleTrain1 <- dfAirbnbTrainSeattle
any(is.na(dfAirbnbSeattleTrain1$host_since))

## [1] TRUE

sum(is.na(dfAirbnbSeattleTrain1$host_since))

## [1] 3

nrow(dfAirbnbSeattleTrain1)

## [1] 5414

na<-which(!complete.cases( dfAirbnbSeattleTrain1 $host_since))
dfAirbnbSeattleTrain1 <- dfAirbnbSeattleTrain1 [-na,]
any(is.na( dfAirbnbSeattleTrain1 $host_since))

## [1] FALSE

str( dfAirbnbSeattleTrain1 $host_since)

## Date[1:5411], format: "2014-09-06" "2012-08-15" "2018-12-04" "2014-04-16"
## "2014-11-26" ...

library(data.table)

##
## Attaching package: 'data.table'

```

```

## The following object is masked from 'package:tsibble':
##
##      key

## The following objects are masked from 'package:lubridate':
##
##      hour, isoweek, mday, minute, month, quarter, second, wday, week,
##      yday, year

## The following object is masked from 'package:purrr':
##
##      transpose

## The following objects are masked from 'package:dplyr':
##
##      between, first, last

setDT(dfAirbnbSeattleTrain1)[, Month_Year := format(as.Date(host_since), "%Y-%m")]
dfAirbnbSeattleTrain1<-dfAirbnbSeattleTrain1 %>%
  dplyr::mutate(year = lubridate::year(host_since),
               month = lubridate::month(host_since),
               day = lubridate::day(host_since))
library(tsibble)
dfAirbnbSeattleTrain1$Month_Year2 <- yearmonth(dfAirbnbSeattleTrain1$host_since)
#str(dfAirbnbSeattleTrain1)

#In which year maximum number of people joined Airbnb
dfAirbnbSeattleTrain1 %>%
  group_by(year) %>%
  tally() %>%
  mutate(pct = 100*n/sum(n)) %>%
  arrange(desc(pct))

## # A tibble: 12 x 3
##   year      n    pct
##   <dbl> <int> <dbl>
## 1  2015   1042  19.3
## 2  2013    817  15.1
## 3  2014    792  14.6
## 4  2016    737  13.6
## 5  2017    479   8.85
## 6  2012    429   7.93
## 7  2018    368   6.80
## 8  2019    330   6.10
## 9  2011    278   5.14
## 10 2010     97   1.79
## 11 2009     35   0.647
## 12 2008      7   0.129

```


Most of the hosts joined Airbnb in the year 2015 which decreased to 6.09% by the year 2019. This could be attributed to strong in-migration and low inventory of reasonably priced housing available for sale contributed to rising rental demand since 2010 (~1.79%). As lenders became more and more confident in the economic recovery builders responded by significantly increasing apartment construction.

In 2015, permitting increased by 33 percent, followed by 14,700 multifamily units permitted in 2016. Of the estimated 20,800 unit under construction, ~ 40% were in the Downtown/ Capitol Hill/Queen Anne market area, which includes the neighborhoods encompassing Amazon.com campuses. ### These then prevailing conditions must have motivated to invest in available business options.

#during which particular months of the year did hosts preferred joining

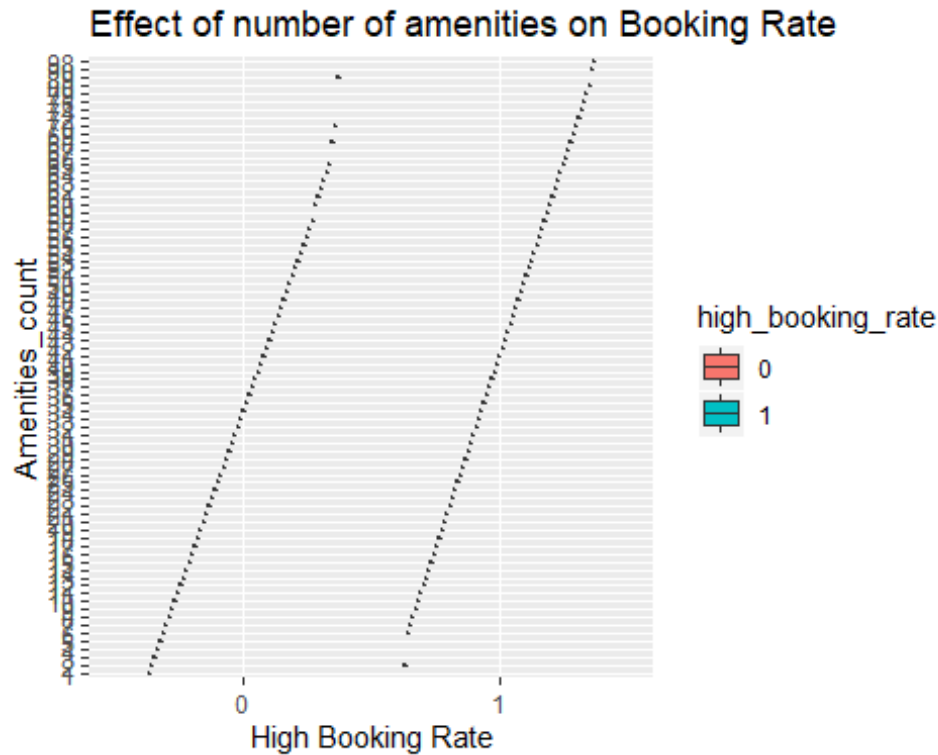
```
dfAirbnbSeattleTrain1 %>%  
  group_by(Month_Year2) %>%  
  tally() %>%  
  mutate(pct = 100*n/sum(n)) %>%  
  arrange(desc(pct))
```

```
## # A tibble: 129 x 3  
##   Month_Year2      n    pct  
##   <mtm> <int> <dbl>  
## 1 2013 Aug   279  5.16  
## 2 2015 Nov   198  3.66  
## 3 2015 Dec   149  2.75  
## 4 2016 Jul   147  2.72  
## 5 2016 Jan   111  2.05  
## 6 2014 Aug   100  1.85  
## 7 2017 Feb    98  1.81  
## 8 2014 Jul    97  1.79  
## 9 2017 Jul    92  1.70  
## 10 2015 Jul    84  1.55  
## # ... with 119 more rows
```

A large number of people joined Airbnb as host in August 2013. In 2013 Forbes, ranked Seattle No. 9 on its list of the Best Places for Business and Careers. Seattle was economically the fastest growing city in 2013. These then prevailing economically sound conditions must have encouraged the property owners to leverage them.

#amenities_count on booking rate

```
plot5<- dfAirbnbTrainSeattle%>%  
  ggplot(aes(y= amenities_count, x= as.factor(high_booking_rate), fill = high_Booking_rate)) +  
  geom_boxplot() + labs(x = "High Booking Rate", y = "Amenities_count", title = "Effect of number of amenities on Booking Rate")  
plot5
```



For properties listed on Airbnb as the number of amenities offered increases the booking rate also increase.

###The effect of Security deposit on high booking rate

```
#security deposit on booking rate
dfAirbnbTrainSeattle %>%
  group_by(security_deposit) %>%
  filter(high_booking_rate==1)%>%
  tally()%>%
  mutate(pct = 100*n/sum(n)) %>%
  arrange(desc(pct))

## # A tibble: 47 x 3
##   security_deposit      n    pct
##   <dbl> <int> <dbl>
## 1         0      623 29.2
## 2      252.    272 12.8
## 3       100    217 10.2
## 4       200    204  9.58
## 5       250    179  8.40
## 6       500    177  8.31
## 7       300    161  7.56
## 8       150     98  4.60
## 9      1000     50  2.35
## 10      400     28  1.31
## # ... with 37 more rows
```

```
plot4<- dfAirbnbTrainSeattle%>%
  ggplot(aes(y= security_deposit, x= as.factor(high_booking_rate),fill = high_booking_rate) )+ geom_boxplot()+ labs(x = "High Booking Rate", y = "Security Deposit", title = "Security Deposit by Booking Rate")
plot4
```



As the security deposit for a property increases, the booking rate decreases. But the impact is not significantly distinguishable on high booking rate as it is almost comparable.

determine range of monthly price distribution across neighborhoods

```
dfAirbnbTrainSeattle %>%
  group_by(neighbourhood) %>%
  filter(monthly_price > 5000) %>%
  tally() %>%
  mutate(pct = 100*n/sum(n)) %>%
  arrange(desc(pct))
```

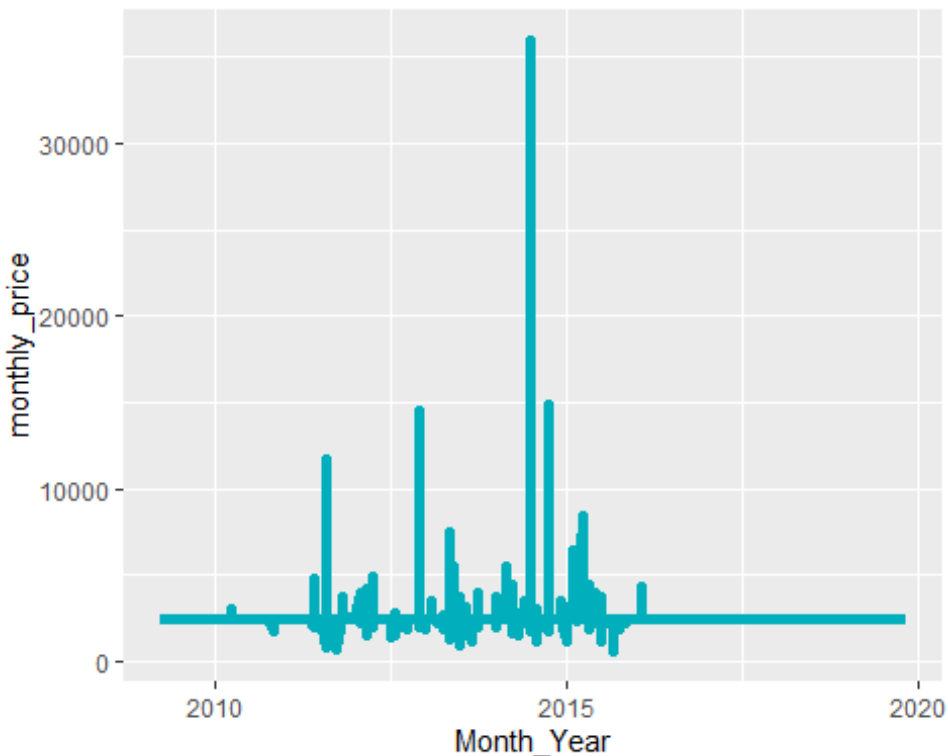
```
## # A tibble: 18 x 3
##   neighbourhood      n  pct
##   <chr>          <int> <dbl>
## 1 Capitol Hill      3 13.0
## 2 Bryant            2  8.70
## 3 Magnolia          2  8.70
## 4 Queen Anne        2  8.70
## 5 Alki              1  4.35
## 6 Ballard           1  4.35
```

```
## 7 Belltown          1 4.35
## 8 Dunlap            1 4.35
## 9 Eastlake          1 4.35
## 10 Genesee          1 4.35
## 11 Green Lake       1 4.35
## 12 Leschi           1 4.35
## 13 Licton Springs   1 4.35
## 14 Maple Leaf       1 4.35
## 15 Mathews Beach    1 4.35
## 16 Minor            1 4.35
## 17 South Lake Union 1 4.35
## 18 Stevens         1 4.35
```

According to the given dataset 13% of the most Expensive rentals(monthly_price>5000) are located in Capitol hill as well. The maximum rent for a property listed is 36000 dollars in Ballard.

```
# monthly_price of top neighborhood listings across years(after joining)
dfBestNeighbourhood1 <- dfBestNeighbourhood
dfBestNeighbourhood1$Month_Year <- yearmonth(dfBestNeighbourhood1$host_since)
plot3<- dfBestNeighbourhood1%>%

  ggplot(aes(x= Month_Year, y= monthly_price))+ geom_line(color= "#00AFBB",
size = 2)
plot3
```



no significant observations.

```
dfAirbnbTrainSeattle %>%
  group_by(neighbourhood) %>%
  filter(high_booking_rate==1)%>%
  tally()%>%
  mutate(pct = 100*n/sum(n)) %>%
  arrange(desc(pct))

## # A tibble: 77 x 3
##   neighbourhood      n    pct
##   <chr>          <int> <dbl>
## 1 Capitol Hill      187  8.78
## 2 Queen Anne        150  7.04
## 3 Minor             141  6.62
## 4 Ballard           128  6.01
## 5 Belltown          124  5.82
## 6 North Beacon Hill   95  4.46
## 7 Wallingford         87  4.08
## 8 Fremont            78  3.66
## 9 First Hill          62  2.91
## 10 Columbia City      56  2.63
## # ... with 67 more rows
```

Listings in Capitol Hill contributes maximum to the properties with high booking rate.

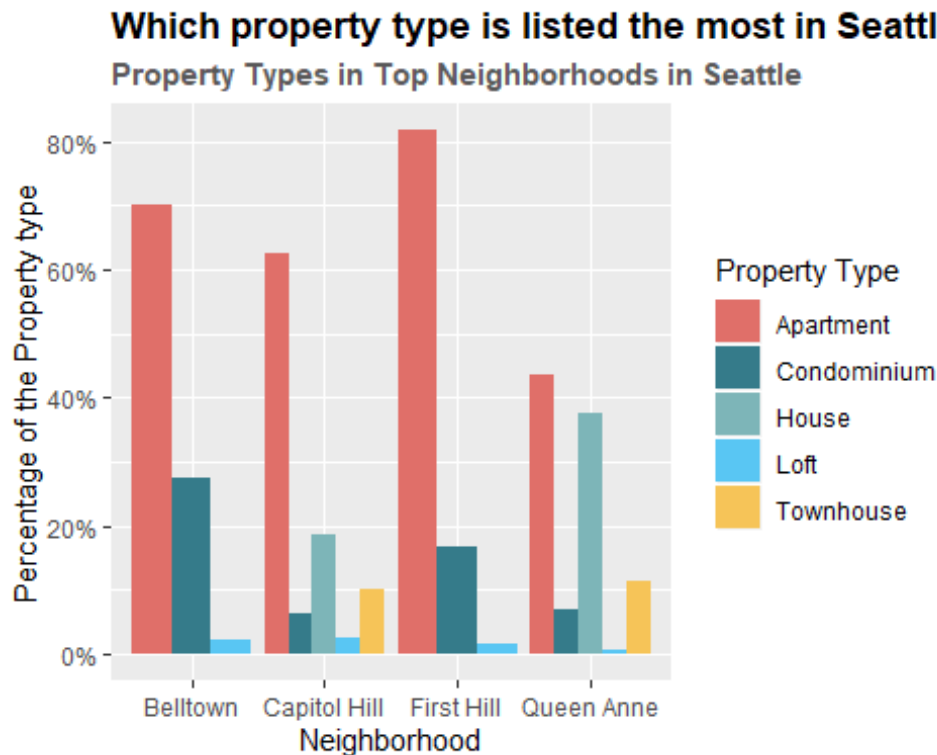
```
##Property Types in Top Neighborhoods in Seattle
dfAirbnbSeattlePrpTyp <- dfAirbnbTrainSeattle %>% group_by(neighbourhood,prop
erty_type) %>% summarize(Freq = n())
dfAirbnbSeattlePrpTyp <- dfAirbnbSeattlePrpTyp %>% filter(property_type %in%
c("Apartment","House","Condominium","Townhouse", "Loft"))
dfAirbnbSeattlePrpTyp <- dfAirbnbSeattlePrpTyp %>% filter (neighbourhood %in%
c("Capitol Hill","Queen Anne","Belltown","First Hill"))
dfAirbnbSeattleSpcPrp<- dfAirbnbTrainSeattle %>% filter(property_type %in% c
("Apartment","House","Condominium","Townhouse", "Loft"))%>% group_by(neighbou
rhood) %>% summarize(sum = n())
propertyratio <- merge(dfAirbnbSeattlePrpTyp, dfAirbnbSeattleSpcPrp, by="neig
hbourhood")
propertyratio <- propertyratio %>% mutate(ratio = Freq/sum)
ggplot(propertyratio, aes(x=neighbourhood, y=ratio, fill = property_type)) +
  geom_bar(position = "dodge",stat="identity") +
  scale_fill_discrete(name = "Property Type") +
  scale_y_continuous(labels = scales::percent) +
  ggtitle("Which property type is listed the most in Seattle?",
    subtitle = "Property Types in Top Neighborhoods in Seattle ") +
  theme(plot.title = element_text(face = "bold")) +
  theme(plot.subtitle = element_text(face = "bold", color = "grey35"))
) +
  theme(plot.caption = element_text(color = "grey68"))+scale_color_g
radient(low="#d3cbcb", high="#852eaa")+
```

```

scale_fill_manual("Property Type", values=c("#e06f69", "#357b8a", "#7db5b8", "#59c6f3", "#f6c458")) +
  xlab("Neighborhood") + ylab("Percentage of the Property type")

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

```



In top neighborhoods the property type which are listed the most are Apartments and Houses. In Capitol Hill which has a variety of property types to be rented ~65 % of the listed properties are Apartments.

what is the contribution of property_type on high booking rate

```

dfAirbnbTrainSeattle %>%
  group_by(property_type) %>%
  filter(high_booking_rate==1) %>%
  tally() %>%
  mutate(pct = 100*n/sum(n)) %>%
  arrange(desc(pct))

```

A tibble: 20 x 3

##	property_type	n	pct
##	<fct>	<int>	<dbl>
##	1 House	619	29.1
##	2 Apartment	615	28.9
##	3 Guest suite	361	16.9
##	4 Townhouse	181	8.50

##	5	Guesthouse	106	4.98
##	6	Condominium	104	4.88
##	7	Loft	40	1.88
##	8	Bungalow	33	1.55
##	9	Cottage	17	0.798
##	10	Serviced apartment	13	0.610
##	11	Other	10	0.469
##	12	Tiny house	8	0.376
##	13	Cabin	5	0.235
##	14	Bed and breakfast	4	0.188
##	15	Camper/RV	4	0.188
##	16	Boat	3	0.141
##	17	Boutique hotel	2	0.0939
##	18	Hostel	2	0.0939
##	19	Villa	2	0.0939
##	20	Tent	1	0.0469

Apartments and Houses contribute ~60 % to high booking rate, whereas condos only contribute ~4%. It confirms with our research for initial acquisition of homes For Property type if one wants to run an Airbnb “hotel” it is advisable to buy a house or apartments since almost all condo buildings in Seattle prohibit this type of rental. The advantage of single-family homes is that they generally charge much more in rent and the tenants may stay for multiple years.

RESULTS AND FINDINGS:

It's a great time to invest in the Seattle housing market based on our research as real estate properties are selling at the market price which is a good sign for Seattle real estate investors. Home prices in Seattle will fall. The city is shifting toward the “neutral” territory.

Seattle is also a bustling city for business, the surrounding suburbs house eight Fortune 500 companies including Amazon, Nordstrom, Microsoft, and Starbucks. Other large companies include MSNBC, Costco, Capital One Investing, and T-Mobile. Amazon HQ2 has effected the Seattle Housing Market, the median home price in the Seattle housing market shot up. Home prices in the metro area went up 47% which is nearly twice as high as the national increase of 24%. After Washington State outlawed rent control, the Market is friendly for Landlords as the rents can be raised to keep up with inflation and demand. We would *recommend* according to our findings to invest in or near Capitol Hill and *University District* for steady income. To invest in Apartment or House. And Finally, to list the property with airbnb during Nov or December [as also obtained in analysis] to leverage the popular times of Jan_apr and most

importantly Jun-September when most of the tourists visit Seattle for various festivals and events like Seattle International Film Festival (May-June), Capitol Hill Block Party (July), Bumbershoot (September). Most of the tourists return year round and therefore Airbnb platform can be used during these periods for network effects and to create a loyal customer base.

PREDICTIVE ANALYSIS:

Running xGBoost model. Making a copy of the dataset since we need to change data types of some variables.

```
##
airbnbSeattleTrain<- dfAirbnbTrainSeattle

## Parsed with column specification:
## cols(
##   .default = col_character(),
##   id = col_double(),
##   high_booking_rate = col_double(),
##   accommodates = col_double(),
##   availability_30 = col_double(),
##   availability_365 = col_double(),
##   availability_60 = col_double(),
##   availability_90 = col_double(),
##   bathrooms = col_double(),
##   bedrooms = col_double(),
##   beds = col_double(),
##   guests_included = col_double(),
##   host_has_profile_pic = col_logical(),
##   host_identity_verified = col_logical(),
##   host_is_superhost = col_logical(),
##   host_listings_count = col_double(),
##   host_since = col_date(format = ""),
##   instant_bookable = col_logical(),
##   is_business_travel_ready = col_logical(),
##   is_location_exact = col_logical(),
##   latitude = col_double()
##   # ... with 16 more columns
## )

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

## Warning: 1 parsing failure.
##   row      col      expected actual      file
## 4740 zipcode no trailing characters -4417 'AirbnbSeattleTrain.csv'

airbnbSeattleTest <- dfAirbnbTestSeattle
```



```

## Parsed with column specification:
## cols(
##   .default = col_character(),
##   id = col_double(),
##   accommodates = col_double(),
##   availability_30 = col_double(),
##   availability_365 = col_double(),
##   availability_60 = col_double(),
##   availability_90 = col_double(),
##   bathrooms = col_double(),
##   bedrooms = col_double(),
##   beds = col_double(),
##   guests_included = col_double(),
##   host_has_profile_pic = col_logical(),
##   host_identity_verified = col_logical(),
##   host_is_superhost = col_logical(),
##   host_listings_count = col_double(),
##   host_since = col_date(format = ""),
##   instant_bookable = col_logical(),
##   is_business_travel_ready = col_logical(),
##   is_location_exact = col_logical(),
##   latitude = col_double(),
##   longitude = col_double()
##   # ... with 15 more columns
## )
## See spec(...) for full column specifications.

## Warning: 1 parsing failure.
##   row    col                                expected actual                                file
## 1016 zipcode no trailing characters -4417 'AirbnbSeattleTest.csv'

dropCols <- c("weekly_price","id","zipcode","host_acceptance_rate","host_has_
profile_pic","square_feet","latitude","longitude","require_guest_phone_verifi
cation","require_guest_profile_picture","space","state","transit","monthly_pr
ice","access","city","description","interaction","neighborhood_overview","hos
t_about","host_location","host_neighbourhood","host_response_rate","host_resp
onse_time","host_since","host_verifications","house_rules","notes","is_locati
on_exact","market","requires_license","review_scores_accuracy","review_score
s_checkin","review_scores_cleanliness","review_scores_communication","review_
scores_location","review_scores_rating","review_scores_value","host_listings_
count","is_business_travel_ready","availability_30","availability_60","avail
ability_90","amenities","neighbourhood")

airbnbSeattleTrain <- airbnbSeattleTrain %>% select(- (dropCols))
airbnbSeattleTest <- airbnbSeattleTest %>% select(-(dropCols))

#Changing datatype to numeric to run xgBoost for Train

airbnbSeattleTrain$extra_people <- as.numeric(gsub('\\$|', ' ', airbnbSeattle
Train$extra_people))

```

```

airbnbSeattleTrain$price <- as.numeric(gsub('\\$|', '', airbnbSeattleTrain$price))
airbnbSeattleTrain$security_deposit <- as.numeric(gsub('\\$|', '', airbnbSeattleTrain$security_deposit))
airbnbSeattleTrain$cleaning_fee <- as.numeric(gsub('\\$|', '', airbnbSeattleTrain$cleaning_fee))

airbnbSeattleTest$extra_people <- as.numeric(gsub('\\$|', '', airbnbSeattleTest$extra_people))
airbnbSeattleTest$price <- as.numeric(gsub('\\$|', '', airbnbSeattleTest$price))
airbnbSeattleTest$security_deposit <- as.numeric(gsub('\\$|', '', airbnbSeattleTest$security_deposit))
airbnbSeattleTest$cleaning_fee <- as.numeric(gsub('\\$|', '', airbnbSeattleTest$cleaning_fee))

airbnbSeattleTrain <- airbnbSeattleTrain %>%
  mutate(cleaning_fee = as.numeric(str_replace_all(airbnbSeattleTrain$cleaning_fee, "[$,]", '')))

meanCleaningFee <- mean(airbnbSeattleTrain$cleaning_fee, na.rm = TRUE)

airbnbSeattleTrain <- airbnbSeattleTrain %>%
  mutate(cleaning_fee = replace_na(cleaning_fee, meanCleaningFee))

airbnbSeattleTest <- airbnbSeattleTest %>%
  mutate(cleaning_fee = as.numeric(str_replace_all(airbnbSeattleTest$cleaning_fee, "[$,]", '')))

meanCleaningFee <- mean(airbnbSeattleTest$cleaning_fee, na.rm = TRUE)

airbnbSeattleTest <- airbnbSeattleTest %>%
  mutate(cleaning_fee = replace_na(cleaning_fee, meanCleaningFee))

airbnbSeattleTrain <- airbnbSeattleTrain %>%
  mutate(extra_people = as.numeric(str_replace_all(airbnbSeattleTrain$extra_people, "[$,]", '')))

airbnbSeattleTest <- airbnbSeattleTest %>%
  mutate(extra_people = as.numeric(str_replace_all(airbnbSeattleTest$extra_people, "[$,]", '')))

airbnbSeattleTrain <- airbnbSeattleTrain %>%
  mutate(price = as.numeric(str_replace_all(airbnbSeattleTrain$price, "[$,]", '')))

meanPrice <- mean(airbnbSeattleTrain$price, na.rm = TRUE)

```

```

airbnbSeattleTrain <- airbnbSeattleTrain %>%
  mutate(price = replace_na(price, meanPrice))

airbnbSeattleTest <- airbnbSeattleTest %>%
  mutate(price = as.numeric(str_replace_all(airbnbSeattle
Test$price,"[$,]",'')))

meanPrice <- mean(airbnbSeattleTest$price, na.rm = TRUE)

airbnbSeattleTest <- airbnbSeattleTest %>%
  mutate(price = replace_na(price, meanPrice))

airbnbSeattleTrain$security_deposit <- ifelse(is.na(airbnbSeattleTrain$securi
ty_deposit),0,airbnbSeattleTrain$security_deposit)
airbnbSeattleTest$security_deposit <- ifelse(is.na(airbnbSeattleTest$security
_deposit),0,airbnbSeattleTest$security_deposit)

#Mean = 1.36
airbnbSeattleTrain$bedrooms <- as.numeric(airbnbSeattleTrain$bedrooms)
airbnbSeattleTrain$bedrooms[is.na(airbnbSeattleTrain$bedrooms)] <- 3.5
summary(airbnbSeattleTrain$bedrooms)

##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.000   1.000   1.000   1.383   2.000   8.000

table(airbnbSeattleTrain$bedrooms)

##
##      0      1      2      3  3.5      4      5      6      7      8
##  670 3000 1058  468     4  145   51     9     8     1

airbnbSeattleTest$bedrooms <- as.numeric(airbnbSeattleTest$bedrooms)
airbnbSeattleTest$bedrooms[is.na(airbnbSeattleTest$bedrooms)] <- 3.5

airbnbSeattleTrain$beds = ifelse(is.na(airbnbSeattleTrain$beds), ave(airbnbSe
attleTrain$beds, FUN = function(x) median(x, na.rm = TRUE)), airbnbSeattleTra
in$beds)

airbnbSeattleTest$beds = ifelse(is.na(airbnbSeattleTest$beds), ave(airbnbSeat
tleTest$beds, FUN = function(x) median(x, na.rm = TRUE)), airbnbSeattleTest$b
eds)

airbnbSeattleTrain$bathrooms = ifelse(is.na(airbnbSeattleTrain$bathrooms), av
e(airbnbSeattleTrain$bathrooms, FUN = function(x) median(x, na.rm = TRUE)), a
irbnbSeattleTrain$bathrooms)

airbnbSeattleTest$bathrooms = ifelse(is.na(airbnbSeattleTest$bathrooms), ave(
airbnbSeattleTest$bathrooms, FUN = function(x) median(x, na.rm = TRUE)), airb
nbSeattleTest$bathrooms)

```

```

airbnbSeattleTrain$host_identity_verified = ifelse(is.na(airbnbSeattleTrain$host_identity_verified), FALSE, airbnbSeattleTrain$host_identity_verified)

airbnbSeattleTest$host_identity_verified = ifelse(is.na(airbnbSeattleTest$host_identity_verified), FALSE, airbnbSeattleTest$host_identity_verified)

airbnbSeattleTrain$host_is_superhost = ifelse(is.na(airbnbSeattleTrain$host_is_superhost), FALSE, airbnbSeattleTrain$host_is_superhost)

airbnbSeattleTest$host_is_superhost = ifelse(is.na(airbnbSeattleTest$host_is_superhost), FALSE, airbnbSeattleTest$host_is_superhost)

airbnbSeattleTrain$high_booking_rate <- as.factor(airbnbSeattleTrain$high_booking_rate)
airbnbSeattleTrain$cancellation_policy <- as.numeric(airbnbSeattleTrain$cancellation_policy)

## Warning: NAs introduced by coercion

airbnbSeattleTest$cancellation_policy <- as.numeric(airbnbSeattleTest$cancellation_policy)

## Warning: NAs introduced by coercion

airbnbSeattleTrain$room_type <- as.numeric(airbnbSeattleTrain$room_type)

## Warning: NAs introduced by coercion

airbnbSeattleTest$room_type <- as.numeric(airbnbSeattleTest$room_type)

## Warning: NAs introduced by coercion

airbnbSeattleTrain$bed_type <- as.numeric(airbnbSeattleTrain$bed_type)

## Warning: NAs introduced by coercion

airbnbSeattleTest$bed_type <- as.numeric(airbnbSeattleTest$bed_type)

## Warning: NAs introduced by coercion

airbnbSeattleTrain$property_type <- as.numeric(airbnbSeattleTrain$property_type)

## Warning: NAs introduced by coercion

airbnbSeattleTest$property_type <- as.numeric(airbnbSeattleTest$property_type)

## Warning: NAs introduced by coercion

airbnbSeattleTrain$instant_bookable <- as.numeric(airbnbSeattleTrain$instant_bookable)
airbnbSeattleTest$instant_bookable <- as.numeric(airbnbSeattleTest$instant_bookable)

```

```

airbnbSeattleTrain$host_identity_verified <- as.numeric(airbnbSeattleTrain$host_identity_verified)
airbnbSeattleTest$host_identity_verified <- as.numeric(airbnbSeattleTest$host_identity_verified)
airbnbSeattleTrain$host_is_superhost <- as.numeric(airbnbSeattleTrain$host_is_superhost)
airbnbSeattleTest$host_is_superhost <- as.numeric(airbnbSeattleTest$host_is_superhost)

```

#running the xgBoost model

```

dfTrain <- airbnbSeattleTrain %>% sample_frac(.80)
dfValid <- setdiff(airbnbSeattleTrain, dfTrain)

library("xgboost")

##
## Attaching package: 'xgboost'

## The following object is masked from 'package:plotly':
##
##     slice

## The following object is masked from 'package:dplyr':
##
##     slice

set.seed(123)
X_train = xgb.DMatrix(as.matrix(dfTrain %>% select(-(high_booking_rate))))
Y_train = dfTrain$high_booking_rate

set.seed(123)

X_test = xgb.DMatrix(as.matrix(dfValid %>% select(-high_booking_rate)))
Y_test = dfValid$high_booking_rate

xgb_trcontrol = trainControl(
  method = "cv",
  number = 10,
  allowParallel = TRUE,
  verboseIter = FALSE,
  returnData = FALSE
)

xgbGrid <- expand.grid(nrounds = c(100,200),
                      max_depth = c(10, 15, 20, 25),
                      colsample_bytree = seq(0.5, 0.9, length.out = 5),
                      eta = 0.1,
                      gamma=0,
                      min_child_weight = 1,
                      subsample = 1
)

```

```

xgb_model = train(
  X_train, Y_train,
  trControl = xgb_trcontrol,
  tuneGrid = xgbGrid,
  method = "xgbTree"
)

resultsXg <-
  xgb_model %>%
    predict(X_test, type= 'prob') %>%
    bind_cols(dfValid,predictedClass=.)
resultsXg

## # A tibble: 5,414 x 23
##   high_booking_ra... accommodates availability_365 bathrooms bed_type bedro
oms
##   <fct>                <dbl>                <dbl>      <dbl>      <dbl>      <d
bl>
## 1 1                3                16          1        NA
0
## 2 1                4                 0          1        NA
1
## 3 0                2               90          1        NA
1
## 4 0                1                 0          1        NA
1
## 5 0                6              160          1        NA
2
## 6 0                4              358          2        NA
2
## 7 1                2              137          1        NA
1
## 8 0                4                 0          1        NA
0
## 9 0                2               66          1        NA
1
## 10 0               3              273          1        NA
1
## # ... with 5,404 more rows, and 17 more variables: beds <dbl>,
## #   cancellation_policy <dbl>, cleaning_fee <dbl>, extra_people <dbl>,
## #   guests_included <dbl>, host_identity_verified <dbl>,
## #   host_is_superhost <dbl>, instant_bookable <dbl>, maximum_nights <dbl>,
## #   minimum_nights <dbl>, price <dbl>, property_type <dbl>, room_type <dbl>
>,
## #   security_deposit <dbl>, `{randomControl}` <dbl>, `0` <dbl>, `1` <dbl>

names(resultsXg)[23] <- "predictedClass"

#resultsXg$predictedClass<- as.numeric(resultsXg$predictedClass)

```

```
resultsXg = resultsXg %>% mutate(predictedHbr = as.factor(ifelse(predictedClass > 0.6, 1, 0)))
```

```
resultsXg
```

```
## # A tibble: 5,414 x 24
```

```
##   high_booking_rate accommodates availability_365 bathrooms bed_type bedrooms
```

```
##   <dbl>           <dbl>           <dbl>           <dbl>           <dbl>           <dbl>
```

```
## 1 1 3 16 1 NA
```

```
## 2 1 4 0 1 NA
```

```
## 3 0 2 90 1 NA
```

```
## 4 0 1 0 1 NA
```

```
## 5 0 6 160 1 NA
```

```
## 6 0 4 358 2 NA
```

```
## 7 1 2 137 1 NA
```

```
## 8 0 4 0 1 NA
```

```
## 9 0 2 66 1 NA
```

```
## 10 0 3 273 1 NA
```

```
## ... with 5,404 more rows, and 18 more variables: beds <dbl>,
```

```
## cancellation_policy <dbl>, cleaning_fee <dbl>, extra_people <dbl>,
```

```
## guests_included <dbl>, host_identity_verified <dbl>,
```

```
## host_is_superhost <dbl>, instant_bookable <dbl>, maximum_nights <dbl>,
```

```
## minimum_nights <dbl>, price <dbl>, property_type <dbl>, room_type <dbl>,
```

```
## security_deposit <dbl>, {randomControl} <dbl>, {0} <dbl>,
```

```
## predictedClass <dbl>, predictedHbr <dbl>
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
##
```

```
## Accuracy : 0.9559
## 95% CI : (0.95, 0.9612)
## No Information Rate : 0.6066
## P-Value [Acc > NIR] : < 2.2e-16
##
## Kappa : 0.9069
##
## McNemar's Test P-Value : 3.204e-06
##
## Sensitivity : 0.9268
## Specificity : 0.9747
## Pos Pred Value : 0.9596
## Neg Pred Value : 0.9535
## Prevalence : 0.3934
## Detection Rate : 0.3646
## Detection Prevalence : 0.3799
## Balanced Accuracy : 0.9507
##
## 'Positive' Class : 1
##
```

We see the specificity and sensitivity are high. We have a good model here to predict high booking rate. Investor can use this model and see if their listings will have a good booking rate or not.

CONCLUSION AND DISCUSSION

Findings

1. Belltown was the neighbourhood with the highest listings.
2. Capitol Hill was the neighbourhood which had the most expensive as well as the least expensive properties.
3. Apartments were the most common property type in Seattle.
4. The spread of properties was almost consistent across the whole state. There is a little higher concentration in central Seattle compared to the rest of the state.
5. Restaurants were the most common attraction in a neighbourhood.
6. Bus was the most popular type of transit.
7. Wifi, Shampoo, Essentials and Kitchens were the most common amenities.

Future Research

1. If the data that is gathered and has less NA values, we can give more accurate recommendations with the help of the models we run.
2. If seasonality can be added to the data, it will help us greatly in future predictions.
3. If we have a column for the periods of time the properties are occupied, it can help do better EDA to present to the investors.
4. If we have some additional parameters we can calculate the time required to break-even on the investment. This will help build a strong case for investors.

Limitations

1. The data scraped for this research was not clean and there were a lot of missing values due to which we had to remove all the NA values which led to the analysis being skewed.
2. We had difficulties in merging with other data sets to do a more in-depth analysis.
3. We were unable to get reviews of the users. The reviews provided by the owners would be biased in their favor and could not give us the correct picture.

REFERENCES

- <http://insideairbnb.com/seattle/>
- <https://www.mashvisor.com/blog/airbnb-profit-calculator-guide/>
- <https://www.mashvisor.com/blog/invest-airbnb-seattle-2018/>
- <https://medium.com/datadriveninvestor/a-closer-look-into-the-data-of-seattles-airbnb-market-d9812ea863f5>
<https://medium.com/@sarah.alsalman96/seattle-airbnb-data-analysis-982add28d731>
- <https://santorinidave.com/best-time-to-visit-seattle>(<https://jinglescode.github.io/datascience/2019/07/13/airbnb-in-seattle-data-analysis/>)