Austin Business Proposal

Group 15

5/6/2020

Data Mining and Predictive Analysis

Austin's AirBNB Strategy

Market: Austin

Team 15: Huan Deng, Adam Gard, Mohit Jaju, Sydney King, Yao Xiao, Yuanhao Zhang

"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 and infringement/plagiarism issue upon any existing copyrighted materials." Contact Member: Adam Gard Team Member 2: Huan Deng Team Member 3: Mohit Jaju Team Member 4: Sydney King Team Member 5: Yao Xiao Team Member 6: Yuanhao Zhang

Executive Summary

The proposition outlined attempts to position a business strategically by discerning key marketing elements that affect the AirBNB offering and its price. Some of these elements are hidden, such as the return on investment (ROI), while others are blatant. In this case, location is a blatant variable of high priority while purchasing a property in the Austin market, making it critical for investors to set an appropriate pricing strategy so as to enhance the booking rate. This proposal developed a series of maps to assist in the visualization of the demand of houses in different neighborhoods and their associated fair price in the current market. An analysis of the difference between high instances of noun usage listings with high booking rates were more likely to describe the entertainment capabilities of the house. Additional analysis was conducted of the Property Type variable and it showed that hotel rooms or apartments were more likely to be booked than houses. Further text analysis led to the creation of a return on investment (ROI) variable as it relates to the words used in the amenities write up, a variable that had no missing values in the entirety of the dataset, has many future uses for airBNB investors. Besides the very broad knowledge of what words bring the highest ROI, the words were found to have three distinct clusterings which were interpreted as "easy fixes", "atmosphere" and "entertainment," which can be used by the airBNB investor to further guide their business decisions.

1. Research Question

1.a. The Big Question

At the onset of the project, the team realized that while the designator of a "high booking rate" should be strived for by every airBNB investor, not every listing will be able to acquire this distinction. With this in mind, the team

decided that we should slightly deviate from using high booking rate as our standard and instead attempt to find a way that will bring the most value to all forms of investors within the Austin region. Which led to our primary question of, “How can we present a full view of what is already bringing the most value owners, and how can we present that value in a way that is easy to interpret and achieve?”

1.b. Question Engagement

1.b.1. The City of Austin

When looking at the city and culture of Austin, an investor would want to know details such as what are the city’s best attractions and safest neighborhoods to help one experience what Austin is known for during their stay. The relationship between a city’s popularity is directly tied to the entertainment it offers visitors, as such our group looked into the following questions: What are the most popular attractions in Austin? Are Airbnb’s with high booking rates closer to these attractions? Additionally, what are the most popular neighborhoods with high booking rates and what are the living conditions for these neighborhoods like?

1.b.2. Property Values

In terms of the housing purchase, we will additionally answer: whether the locations with high demand are in the Austin Airbnb Market, what is the current fair market price in the market and if it changes for listings downtown, as well as what should the investor charge.

(MOHIT)

1.b.3. Amenity Returns on Investment

AirBNB hosts predominantly use the income as supplementary with other occupations as their primary income. With this reality in mind, the most productive way to improve the value of a rental is to conduct renovations on previously purchased properties. Yet, knowing which renovation provides the highest return on investment (ROI) is different depending on the expected use of the property, with renovations for future sales being different than rental renovations. While everyone knows that redoing the bathroom or kitchen will increase the overall value of the property, sometimes these major renovations are too much for a short-term vacation rental. AirBNBs tend to have a high turnover of guests, making it less valuable to have the most meticulous carpet inlay upon deep inspection as opposed to that same inlay installed a bit cheaper. Nearly everyone knows these basics about home improvement, and it is from the most basic variable in the AirBNB data, amenities, that the highest “real value” can be extracted. We explain it as “real value” due to the tangible aspect of the words within the amenities variable and the fact that it has no missing values throughout the entirety of both training and test data sets. The tangible aspect of the word “pool” is inherent and easily understood, giving the return on investment a concrete relationship to the idea of having a pool somehow related to the listing will increase the overall value of that listing. Obviously, some words such as “wide” can have multiple meanings depending upon its complementary words or phrasing. Direct connections with the words used and their overall value is expected to provide a deeper understanding and create an internal map for the investor as to what they should and should not do to gain the most value out of their AirBNB listings.

2. Methodology

2.a. Data Preparation and Analysis

2.a.1. Broad Cleaning

Some variables were recognized to not be of the appropriate type and they were the first targets during the data cleaning. The change of the values to their proper type looked like such, delete the '$' in cleaning_fee and convert it to numeric and change the high_booking rate to factor. All these changes were used to eliminate errors that would occur from classification problems and dependent value sets. Additionally, there are a plethora of NA values in some variables that were filled with imputed values. The NA values in numeric variables were chosen to fill it with the mean of the variable. Factor variable NAs were filled with the most common level in the column, making it the most common level in that variable. Due to an outlier's effect on variance and the standard deviation of a data distribution, the distribution becomes skewed in the direction of the outliers which makes it difficult to analyze the data. As a result of that, we needed to remove the outliers from the dataset. Determination of the outliers was based on quartiles and after detect_outliers, they were transformed to the mean value of this column. High bias variables were used to determine every factor variable and remove them from our model. Luckily, we can simply detect bias on function skim(dataset). Additionally, we removed the variables host_has_profile_pic and requires_license. Next, due to bijection causing multicollinearity in a model, which would influence the result of our prediction result. The whichAreBijection function to check bijection in all columns and determined that there is no bijection in the cleaned dataset. Finally, standardizing the features around the center and 0 with a standard deviation of 1 is important when we compare measurements that have different units. Variables that are measured at different scales do not contribute equally to the analysis and might end up creating a bias. Some variables in our data set like Price and Maximum_Nights have different units. Different scales can cause different influences on dependent variables. As a result of that, we scaled the variables with function build_scales and fastScale which resulted in a clean data set..

2.a.2. Amenity Text Cleaning and ROI

In an effort to include all of the potential words in the Amenities variable, the training and test sets from the Kaggle model were combined. Due to the fact that by attempting to acquire the ROI of words that might not be normally distributed across all listings, the variable High_Booking_Rate was removed, which is the Kaggle competition's measured variable. It is worth noting that while the data frame was constrained to Austin, the following procedures were tested with the entire AirBNB data set. The variables chosen from which ROI was calculated are Amenities, ID, Availability_365 and Price. The variable Availability_365 was transformed into the variable Unavail by simply subtracting 365 days from the available days of the listing. Visibility of the amount of time a listing was in use would help to justify the financial returns that are being returned to the owner over the course of the year. Also, a few general assumptions about the normality of use were extrapolated for all of the properties in the datatset. These assumptions of unavailability include: renovations, owner's use of the listing, administrative holds, property maintenance, guest changeover. While there is a potential for outliers, such as some hosts actively living in their listing unless they temporarily leave, these unavailable listings are assumed to be balanced by listings that are rarely booked. Income is a transformation variable derived from multiplying the Price of the listing (ID) by the amount of time the listing (ID) is Unavail (unavailable) and represents the annual income of the listing. Along with the creation of those modified variables, the Amenities variable is text mined using the tidytext library in order to return all of the words in each of the listings (ID). The individual words are aggregated, counted and transformed into the variable Instances, which is the overall amount of times that word appears across all listings in the dataframe. Other transformed variables are Val (short for value) and Tot (short for total). Val creates partial returns on investment for each individual instance of a word in each listing (i.e. For the first instance of "Pool" out of 3 instancesin ID 101 it returns 1/3rd of the ID's overall ROI for "Pool"). Tot is the ROI (and renamed as such) and is the sum of all of the partial ROIs for every instance of the Word.

$$ROI_{Word} = \frac{(\sum(Unavail * Price)_{ID})_{Word}}{n_{Word}}$$ ### 2.a.2. Austin Related Text Cleaning The cleaned data set was filtered to only include the Description variable of the AirBNBs in Austin. The Description variable is used by a listing's host to provide insight into what the city has to offer. # Compiled Data Cleansing

```
setwd("C:\\Users\\Adam\\OneDrive\\UMD\\Semester 2\\BUDT758T - Data Mining and Analytic
s\\Kaggle\\predict")

library("tidyverse")
```

```
## Warning: package 'tidyverse' was built under R version 3.6.3
```

```
## -- Attaching packages
---------------------------------------------------------------------------------------
------------------------------------------------ tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  3.0.0     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
## Warning: package 'tibble' was built under R version 3.6.3
```

```
## Warning: package 'tidyr' was built under R version 3.6.3
```

```
## Warning: package 'readr' was built under R version 3.6.3
```

```
## Warning: package 'purrr' was built under R version 3.6.3
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
## Warning: package 'stringr' was built under R version 3.6.3
```

```
## Warning: package 'forcats' was built under R version 3.6.3
```

```
## -- Conflicts
---------------------------------------------------------------------------------------
------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library("tidymodels")
```

```
## Warning: package 'tidymodels' was built under R version 3.6.3
```

```
## -- Attaching packages
--------------------------------------------------------------------------------
------------------------------------------------ tidymodels 0.1.0 --
```

```
## v broom     0.5.5      v rsample   0.0.6
## v dials     0.0.6      v tune      0.1.0
## v infer     0.5.1      v workflows 0.1.1
## v parsnip   0.1.0      v yardstick 0.0.6
## v recipes   0.1.10
```

```
## Warning: package 'broom' was built under R version 3.6.3
```

```
## Warning: package 'dials' was built under R version 3.6.3
```

```
## Warning: package 'scales' was built under R version 3.6.3
```

```
## Warning: package 'infer' was built under R version 3.6.3
```

```
## Warning: package 'parsnip' was built under R version 3.6.3
```

```
## Warning: package 'recipes' was built under R version 3.6.3
```

```
## Warning: package 'rsample' was built under R version 3.6.3
```

```
## Warning: package 'tune' was built under R version 3.6.3
```

```
## Warning: package 'workflows' was built under R version 3.6.3
```

```
## Warning: package 'yardstick' was built under R version 3.6.3
```

```
## -- Conflicts
-------------------------------------------------------------------------------------
------------------------------------------------- tidymodels_conflicts() --
## 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()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
```

```
library("plotly")
```

```
## Warning: package 'plotly' was built under R version 3.6.3
```

```
##
## 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("skimr")
library("caret")
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

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

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

```
library("ggrepel")
```

```
## Warning: package 'ggrepel' was built under R version 3.6.3
```

```
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("rockchalk")
```

```
## Warning: package 'rockchalk' was built under R version 3.6.3
```

```
##
## Attaching package: 'rockchalk'
```

```
## The following object is masked from 'package:dplyr':
##
##     summarize
```

```
library("Hmisc")
```

```
## Warning: package 'Hmisc' was built under R version 3.6.3
```

```
## Loading required package: survival
```

```
## Warning: package 'survival' was built under R version 3.6.3
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: Formula
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following object is masked from 'package:rockchalk':
##
##     summarize
```

```
## The following object is masked from 'package:plotly':
##
##     subplot
```

```
## The following object is masked from 'package:parsnip':
##
##     translate
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
library("stringr")
library("tidytext")
```

```
## Warning: package 'tidytext' was built under R version 3.6.3
```

```
library("gsubfn")
```

```
## Warning: package 'gsubfn' was built under R version 3.6.3
```

```
## Loading required package: proto
```

```
## Warning: package 'proto' 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("stopwords")
```

```
## Warning: package 'stopwords' was built under R version 3.6.3
```

```
##
## Attaching package: 'stopwords'
```

```
## The following object is masked from 'package:tm':
##
##     stopwords
```

```
library("gridExtra")
```

```
## Warning: package 'gridExtra' was built under R version 3.6.3
```

```
##
## Attaching package: 'gridExtra'
```

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

```
library("lattice")
library("factoextra")
```

```
## Warning: package 'factoextra' was built under R version 3.6.3
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3
WBa
```

```
library("cowplot")
```

```
## Warning: package 'cowplot' was built under R version 3.6.3
```

```
##
## ************************************************************
```

```
## Note: As of version 1.0.0, cowplot does not change the
```

```
##   default ggplot2 theme anymore. To recover the previous
```

```
##   behavior, execute:
##   theme_set(theme_cowplot())
```

```
## ************************************************************
```

```
library("topicmodels")
```

```
## Warning: package 'topicmodels' was built under R version 3.6.3
```

```
library("glmnet")
```

```
## Warning: package 'glmnet' was built under R version 3.6.3
```

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

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 3.0-2
```

```
library("leaflet")
```

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

```
library("leaflet.extras")
```

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

```
library("wordcloud")
```

```
## Warning: package 'wordcloud' was built under R version 3.6.3
```

```
## Loading required package: RColorBrewer
```

```
library("wordcloud2")
```

```
## Warning: package 'wordcloud2' was built under R version 3.6.3
```

```
library("tm")
library("RColorBrewer")
library("dataPreparation")
```

```
## Warning: package 'dataPreparation' was built under R version 3.6.3
```

```
## Loading required package: lubridate
```

```
## Warning: package 'lubridate' was built under R version 3.6.3
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:cowplot':
##
##     stamp
```

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

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

```
## Loading required package: progress
```

```
## Warning: package 'progress' was built under R version 3.6.3
```

```
## dataPreparation 0.4.3
```

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

```
library("udpipe")
```

```
## Warning: package 'udpipe' was built under R version 3.6.3
```

```
library("lubridate")

# 1 Data Preparation
dfTrain <- read.csv("airbnbTrain.csv")
dfTest <- read.csv("airbnbTest.csv")

## Data Cleaning
dfTrain$security_deposit <- as.numeric(gsub('\\$|,', '', dfTrain$security_deposit))
dfTest$security_deposit <- as.numeric(gsub('\\$|,', '', dfTest$security_deposit))
dfTrain$cleaning_fee <- as.numeric(gsub('\\$|,', '', dfTrain$cleaning_fee))
dfTest$cleaning_fee <- as.numeric(gsub('\\$|,', '', dfTest$cleaning_fee))
dfTrain$extra_people <- as.numeric(gsub('\\$|,', '', dfTrain$extra_people))
dfTest$extra_people <- as.numeric(gsub('\\$|,', '', dfTest$extra_people))
dfTrain$price <- as.numeric(gsub('\\$|,', '', dfTrain$price))
dfTest$price <- as.numeric(gsub('\\$|,', '', dfTest$price))
dfTrain$cleaning_fee <- ifelse(is.na(dfTrain$cleaning_fee),0,dfTrain$cleaning_fee)
dfTest$cleaning_fee <- ifelse(is.na(dfTest$cleaning_fee),0,dfTest$cleaning_fee)
dfTrain$extra_people <- ifelse(is.na(dfTrain$extra_people),0,dfTrain$extra_people)
dfTest$extra_people <- ifelse(is.na(dfTest$extra_people),0,dfTest$extra_people)
dfTrain$price <- ifelse(is.na(dfTrain$price),0,dfTrain$price)
dfTest$price <- ifelse(is.na(dfTest$price),0,dfTest$price)
dfTrain$security_deposit <- ifelse(is.na(dfTrain$security_deposit),0,dfTrain$security_
deposit)
dfTest$security_deposit <- ifelse(is.na(dfTest$security_deposit),0,dfTest$security_dep
osit)
dfTrain$bedrooms = ifelse(is.na(dfTrain$bedrooms), ave(dfTrain$bedrooms, FUN = functio
n(x) median(x, na.rm = TRUE)), dfTrain$bedrooms)
dfTrain$beds = ifelse(is.na(dfTrain$beds), ave(dfTrain$beds, FUN = function(x) median
(x, na.rm = TRUE)), dfTrain$beds)
dfTrain$bathrooms = ifelse(is.na(dfTrain$bathrooms), ave(dfTrain$bathrooms, FUN = func
tion(x) median(x, na.rm = TRUE)), dfTrain$bathrooms)
dfTrain$host_identity_verified = ifelse(is.na(dfTrain$host_identity_verified),FALSE, d
fTrain$host_identity_verified)
dfTrain$host_is_superhost = ifelse(is.na(dfTrain$host_is_superhost),FALSE, dfTrain$hos
t_is_superhost)
dfTest$bedrooms = ifelse(is.na(dfTest$bedrooms), ave(dfTest$bedrooms, FUN = function
(x) median(x, na.rm = TRUE)), dfTest$bedrooms)
dfTest$beds = ifelse(is.na(dfTest$beds), ave(dfTest$beds, FUN = function(x) median(x,
na.rm = TRUE)), dfTest$beds)
dfTest$bathrooms = ifelse(is.na(dfTest$bathrooms), ave(dfTest$bathrooms, FUN = functio
n(x) median(x, na.rm = TRUE)), dfTest$bathrooms)
dfTest$host_identity_verified = ifelse(is.na(dfTest$host_identity_verified),FALSE, dfT
est$host_identity_verified)
dfTest$host_is_superhost = ifelse(is.na(dfTest$host_is_superhost),FALSE, dfTest$host_i
s_superhost)
dfTrain$high_booking_rate <- as.factor(dfTrain$high_booking_rate)
dfTrain$property_type <- as.numeric(dfTrain$property_type)
dfTest$property_type <- as.numeric(dfTest$property_type)
dfTrain$room_type <- as.numeric(dfTrain$room_type)
dfTest$room_type <- as.numeric(dfTest$room_type)
```

```
dfTrain$bed_type <- as.numeric(dfTrain$bed_type)
dfTest$bed_type <- as.numeric(dfTest$bed_type)
dfTrain$cancellation_policy <- as.numeric(dfTrain$cancellation_policy)
dfTest$cancellation_policy <- as.numeric(dfTest$cancellation_policy)

## Additional Cleaning
#Function to check and remove outliers
outlierCheck <- function(dt, var) {
     var_name <- eval(substitute(var),eval(dt))
     na1 <- sum(is.na(var_name))
     m1 <- mean(var_name, na.rm = T)
     boxplot(var_name, main="With outliers")
     hist(var_name, main="With outliers", xlab=NA, ylab=NA)
     outlier <- boxplot.stats(var_name)$out
     mo <- mean(outlier)
     var_name <- ifelse(var_name %in% outlier, NA, var_name)
     boxplot(var_name, main="Without outliers")
     hist(var_name, main="Without outliers", xlab=NA, ylab=NA)
     title("Outlier Check", outer=TRUE)
     na2 <- sum(is.na(var_name))
     cat("Outliers identified:", na2 - na1, "n")
     cat("Propotion (%) of outliers:", round((na2 - na1) / sum(!is.na(var_name))*100,
1), "n")
     cat("Mean of the outliers:", round(mo, 2), "n")
     m2 <- mean(var_name, na.rm = T)
     cat("Mean without removing outliers:", round(m1, 2), "n")
     cat("Mean if we remove outliers:", round(m2, 2), "n")
}
outlierRemove <- function(dt, var) {
     var_name <- eval(substitute(var),eval(dt))
     outlier <- boxplot.stats(var_name)$out
     var_name <- ifelse(var_name %in% outlier, NA, var_name)
     dt[as.character(substitute(var))] <- invisible(var_name)
     assign(as.character(as.list(match.call())$dt), dt, envir = .GlobalEnv)
     cat("Outliers successfully removed", "n")
     return(invisible(dt))
}
#Calculate scales
df <-
  read_csv("airbnbTrain.csv")
```

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

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

```
scales <-
  build_scales(dataSet = df, cols = colnames(Filter(is.numeric, subset(df, select=-c(i
d,high_booking_rate)))), verbose = TRUE)
```

```
## [1] "build_scales: I will compute scale on  23 numeric columns."
## [1] "build_scales: it took me: 0.08s to compute scale for 23 numeric columns."
```

```
df <- fastScale(dataSet = df, scales = scales, verbose = TRUE)
```

```
## [1] "fastScale: I will scale 23 numeric columns."
## [1] "fastScale: it took me: 0.2s to scale 23 numeric columns."
```

```
#Finding bijection
bijections <- whichAreBijection(dataSet = df, verbose = TRUE)
```

```
## [1] "whichAreBijection: it took me 3.2s to identify 0 column(s) to drop."
```

```
## Create Austin DF for Wordcloud
df <-
  read_csv("airbnbTrain.csv")
```

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

```
#filter Austin data
df_Austin<- subset(df, market == "Austin" | market == "Texas - Austin")
#filter to only those with high booking rate
#filter Austin data
df_Austin<- subset(df, market == "Austin" | market == "Texas - Austin" | high_booking_
rate == "1")
#drop numeric rows with missing value:bathrooms, bedrooms, beds
df_Austin<- df_Austin[is.na(df_Austin$bathrooms) != TRUE & is.na(df_Austin$bedrooms) !
= TRUE & is.na(df_Austin$beds) != TRUE,]
#change these rows to numeric:cleaning fee, extra_people, host_response_rate, price, s
ecurity_deposit
df_Austin<-
  df_Austin %>%
  mutate(cleaning_fee =  as.numeric(str_remove_all(cleaning_fee, "[$]")),
         extra_people =as.numeric(str_remove_all(extra_people, "[$]")),
         host_response_rate = as.numeric(str_remove_all(host_response_rate, "[%]")),
          price = as.numeric(str_remove_all(price, "[$]")),
          security_deposit = as.numeric(str_remove_all(security_deposit, "[$]")))
```

```
## Warning: NAs introduced by coercion
```

```
## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

```
#fill these columns having na with their mean:cleaning fee, host_response_rate, host_l
istings_ count, price, review_scores_accuracy, review_scores_checkin, review_scores_cl
eanliness, review_scores_communication, review_scores_location, review_scores_rating,
review_scores_value, security_deposit
df_Austin<-
  df_Austin %>%
  mutate(cleaning_fee = ifelse(is.na(cleaning_fee), mean(cleaning_fee,na.rm = TRUE),cl
eaning_fee),
         host_response_rate = ifelse(is.na(host_response_rate), mean(host_response_rat
e,na.rm = TRUE),host_response_rate),
         host_listings_count = ifelse(is.na(host_listings_count), mean(host_listings_c
ount,na.rm = TRUE),host_listings_count),
         host_response_rate = ifelse(is.na(host_response_rate), mean(host_response_rat
e,na.rm = TRUE),host_response_rate),
         price  = ifelse(is.na(price), mean(price,na.rm = TRUE),price),
         review_scores_accuracy  = ifelse(is.na(review_scores_accuracy), mean(review_s
cores_accuracy,na.rm = TRUE),review_scores_accuracy),
         review_scores_checkin  = ifelse(is.na(review_scores_checkin), mean(review_sco
res_checkin,na.rm = TRUE),review_scores_checkin),
         review_scores_cleanliness  = ifelse(is.na(review_scores_cleanliness), mean(re
view_scores_cleanliness,na.rm = TRUE),review_scores_cleanliness),
         review_scores_communication  = ifelse(is.na(review_scores_communication), mea
n(review_scores_communication,na.rm = TRUE),review_scores_communication),
         review_scores_location  = ifelse(is.na(review_scores_location), mean(review_s
cores_location,na.rm = TRUE),review_scores_location),
         review_scores_rating  = ifelse(is.na(review_scores_rating), mean(review_score
s_rating,na.rm = TRUE),review_scores_rating),
         review_scores_value  = ifelse(is.na(review_scores_value), mean(review_scores_
value,na.rm = TRUE),review_scores_value),
         security_deposit  = ifelse(is.na(security_deposit), mean(security_deposit,na.
rm = TRUE),security_deposit)
         )
#fill host_response_time missing values with mode: within an hour and change it to fac
tor
df_Austin<-
  df_Austin %>%
  mutate(host_response_time = as.factor(ifelse(host_response_time == "N/A" | is.na(hos
t_response_time) == TRUE, "within an hour", host_response_time)))
#classify all NA in neighborhood to others
df_Austin<-
  df_Austin %>%
  mutate(neighbourhood = as.factor(ifelse(is.na(neighbourhood) == TRUE, "Others", neig
hbourhood)))
#change these variables to factor:bed_type, cancellation_policy, property_type, room_t
ype,
df_Austin<-
  df_Austin %>%
  mutate(bed_type = as.factor(bed_type),
         cancellation_policy = as.factor(cancellation_policy),
         property_type = as.factor(property_type),
```

```
         bed_type = as.factor(bed_type),
         room_type = as.factor(room_type))
# fill NA in host_has_profile_pic, host_identity_verified, host_is_superhost with mode
of each column
df_Austin<-
  df_Austin %>%
  mutate(host_has_profile_pic = ifelse(is.na(host_has_profile_pic) == TRUE, TRUE, host
_has_profile_pic),
         host_identity_verified = ifelse(is.na(host_identity_verified) == TRUE, FALSE,
host_identity_verified),
         host_is_superhost = ifelse(is.na(host_is_superhost) == TRUE, FALSE, host_is_s
uperhost))
#Selecting all bnb's in Austin , TX by filtering just but ID , city and state
df_Austin %>%
  select( id, state, city) %>%
  filter(state == "TX", city == "Austin")
```

id <dbl>	state <chr>	city <chr>
1104266	TX	Austin
1073887	TX	Austin
1157991	TX	Austin
1173607	TX	Austin
1029791	TX	Austin
1011718	TX	Austin
1135433	TX	Austin
1158218	TX	Austin
1064797	TX	Austin
1123484	TX	Austin

1-10 of 6,646 rows Previous 1 2 3 4 5 6 … 665 Next

```
#Selecting description of bnb's in Austin
df_Austindesc <- df_Austin %>%
  select(id,city, description) %>%
  filter(city == "Austin")
#tokenize words
dfTidyAustin <- df_Austindesc %>%
  unnest_tokens(word, description)
#Remove stop words
dfTidyAustin <-
  dfTidyAustin %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```
#counting the frequency of words
dfTidyAustin <- df_Austindesc %>%
  unnest_tokens(word, description) %>%
  anti_join(stop_words)%>%
  dplyr::count(word,  sort=TRUE)
```

```
## Joining, by = "word"
```

```
## Create the Train for Amenities DF
dftrain <-
read_csv("airbnbTrain.csv") %>%
rename_all(tolower)
```

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

```
## Create the Test DF
dftest <-
read_csv("airbnbTest.csv") %>%
rename_all(tolower)
```

```
## 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 14 more columns
## )
## See spec(...) for full column specifications.
```

```
###Drops non-matching column between Train and Test for the Complete DF
dft1 <- dftrain
drop <- c("high_booking_rate")
dft1 <- dft1[,!(names(dft1) %in% drop)]
###Creates Combined DF
full <- rbind(dft1,dftest)
##Changes Price to numeric and Selects Austin location
aus <- full %>%
  mutate(price=as.numeric(str_remove_all(price,"[$]"))) %>%
  filter(str_detect(`{randomcontrol}`,"102"))
```

```
## Warning: NAs introduced by coercion
```

```
##Filter to only variables for Amenities ROI
amen <- aus[c("amenities","id","availability_365","price")]
##Create DF for Text Mine
text_amen <- tibble(id = amen$id, text =amen$amenities, avail= amen$availability_365,
price=amen$price)
##Dissociate Words
word_amen <- text_amen %>%
  unnest_tokens(word, text)
##Remove Stop Words
data(stop_words)
word_amen <- word_amen %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```
##Count instance of word
count <- word_amen %>% count(word, sort=TRUE)
##Append count to mined DF
word_amen <- word_amen %>% left_join(count, by = "word") %>%
  mutate(unavail = 365-avail)
##Reduce size of DF and append Instances of the Word
word_n <- word_amen %>%
  select(word,n,id) %>%
  rename(instances=n)
## Income by word for ID
inc_amen <- word_amen %>%
    group_by(id) %>%
  mutate(income = unavail*price) %>%
  left_join(word_n, by=c("word","id"))     #Binds Overall Instances of each Word to ea
ch Word/ID
##Determining the Income of Each Word/ID
winc_amen <- inc_amen%>%
  select(word,income,instances,n) %>%
  group_by(word,income,instances) %>%
  select(income) %>%
  tally() %>%
  mutate(val=as.integer(income*n), na.rm=TRUE) %>%
  arrange(desc(val))
```

```
## Adding missing grouping variables: `id`
```

```
## Adding missing grouping variables: `word`, `instances`
```

```
## Reduce DF for easier summarise
filt_amen <- winc_amen %>%
  select(word,val, instances)
```

```
## Adding missing grouping variables: `income`
```

```
##Creating total Value per EACH instance of Word/ID
tot_amen <- filt_amen %>%
  group_by(word) %>%
 mutate(tot=sum(val, na.rm=TRUE)/instances)
###Sums all of the PARTIAL ROIs and shows their Instances
roi_amen <- tot_amen %>%
  group_by(word,tot) %>%
  select(tot) %>%
  tally() %>%
  rename(ROI=tot) %>%
  mutate(ROI=as.integer(ROI)) %>%
  rename(Instances=n) %>%
  mutate(Instances=as.integer(Instances))
```

```
## Adding missing grouping variables: `word`
```

```
## Outlier Detection and Filter Other Variables
df <-
  read_csv("cleaned_airbnb.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   access = col_character(),
##   amenities = col_character(),
##   bed_type = col_character(),
##   cancellation_policy = col_character(),
##   city = col_character(),
##   description = col_character(),
##   host_about = col_character(),
##   host_acceptance_rate = col_character(),
##   host_has_profile_pic = col_logical(),
##   host_identity_verified = col_logical(),
##   host_is_superhost = col_logical(),
##   host_location = col_character(),
##   host_neighbourhood = col_character(),
##   host_response_time = col_character(),
##   host_since = col_date(format = ""),
##   host_verifications = col_character(),
##   house_rules = col_character(),
##   instant_bookable = col_logical(),
##   interaction = col_character(),
##   is_business_travel_ready = col_logical()
##   # ... with 15 more columns
## )
```

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

```
## Text analysis of Description Variable
# text analysis for description
df <-
  read_csv("cleaned_airbnb.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   access = col_character(),
##   amenities = col_character(),
##   bed_type = col_character(),
##   cancellation_policy = col_character(),
##   city = col_character(),
##   description = col_character(),
##   host_about = col_character(),
##   host_acceptance_rate = col_character(),
##   host_has_profile_pic = col_logical(),
##   host_identity_verified = col_logical(),
##   host_is_superhost = col_logical(),
##   host_location = col_character(),
##   host_neighbourhood = col_character(),
##   host_response_time = col_character(),
##   host_since = col_date(format = ""),
##   host_verifications = col_character(),
##   house_rules = col_character(),
##   instant_bookable = col_logical(),
##   interaction = col_character(),
##   is_business_travel_ready = col_logical()
##   # ... with 15 more columns
## )
## See spec(...) for full column specifications.
```

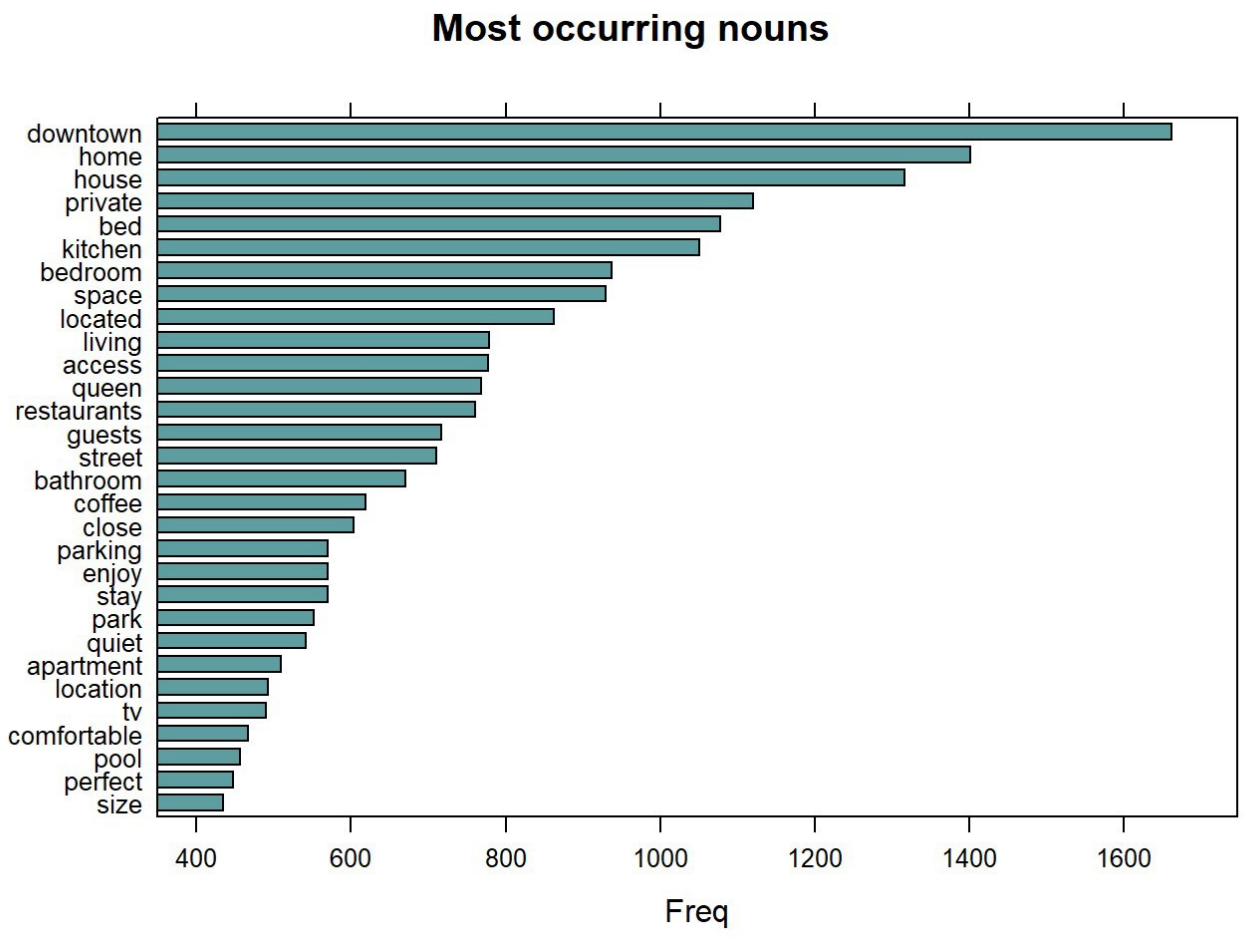
```
# split the dataset based on high booking rate
dfdescription1 <-
  df %>%
  filter(high_booking_rate == 1) %>%
  select(description) %>%
  unnest_tokens(word, description)
dfdescription2 <-
  df %>%
  filter(high_booking_rate == 0) %>%
  select(description) %>%
  unnest_tokens(word, description)
#remove stop words
dfdescription1 <-
 dfdescription1 %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```
dfdescription2 <-
 dfdescription2 %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```
#remove some meaningless words manully
dfdescription1 <-
  dfdescription1 %>%
  filter(!(word %in% c("austin","neighborhood","walk","walking","miles","10","minute
s","1","2","3","4","5","6","7","8","9","10","north","south","east","west","texas")))
dfdescription2 <-
  dfdescription2 %>%
  filter(!(word %in% c("austin","neighborhood","walk","walking","miles","10","minute
s","1","2","3","4","5","6","7","8","9","10","north","south","east","west","texas")))

#Calculate and show the freq of top freq words
#Visualizatoin
stats <- subset(dfdescription1, word %in% c("NOUN"))
stats <- txt_freq(dfdescription1$word)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 30), col = "cadetblue", main = "Most occurring
nouns", xlab = "Freq")
```




```
wordcloud(words = stats$key, freq = stats$freq, min.freq = 1,max.words=200, random.ord
er=FALSE, rot.per=0.35,            colors=brewer.pal(8, "Dark2"))
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## experience could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## fridge could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## shopping could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## microwave could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## shared could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## upstairs could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## rental could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## friendly could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : couch
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## travelers could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## equipped could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## sleeps could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : block
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## convention could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## netflix could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## unique could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## family could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## internet could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## provided could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## rainey could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## natural could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## stocked could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : soco
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## separate could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : style
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## laundry could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : story
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## appliances could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : trees
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## remodeled could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## provide could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : brand
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : sleep
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : trail
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## furnished could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : super
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## comfortably could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## university could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : grill
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## cottage could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## historic could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## couples could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## travis could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : creek
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## community could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : night
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## nearby could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## friends could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## sleeping could not be fit on page. It will not be plotted.
```

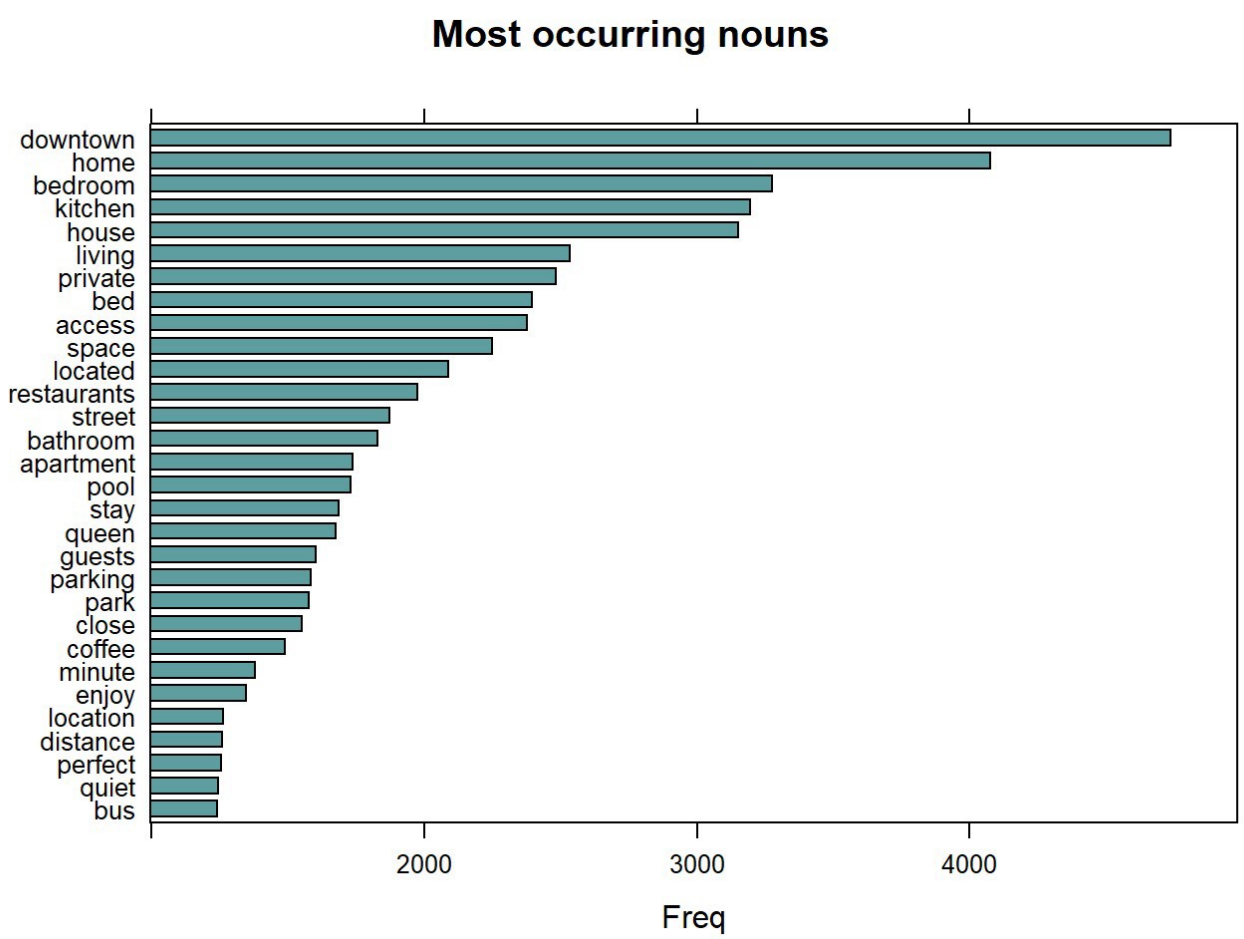
```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## mattresses could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## windows could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## questions could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## grocery could not be fit on page. It will not be plotted.
```


```
stats <- subset(dfdescription2, word %in% c("NOUN"))
stats <- txt_freq(dfdescription2$word)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 30), col = "cadetblue", main = "Most occurring
nouns", xlab = "Freq")
```




```
wordcloud(words = stats$key, freq = stats$freq, min.freq = 1,max.words=200, random.ord
er=FALSE, rot.per=0.35,            colors=brewer.pal(8, "Dark2"))
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## community could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## shower could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## includes could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## garage could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## experience could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## bathrooms could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## furnished could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : spot
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## questions could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## towels could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## mattresses could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## downstairs could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## appliances could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : views
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## domain could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## amazing could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## you'll could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## campus could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## sleeps could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : brand
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : porch
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : creek
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : cable
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## rainey could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : block
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## venues could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## business could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : gym
## could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## friends could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## privacy could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## trails could not be fit on page. It will not be plotted.
```

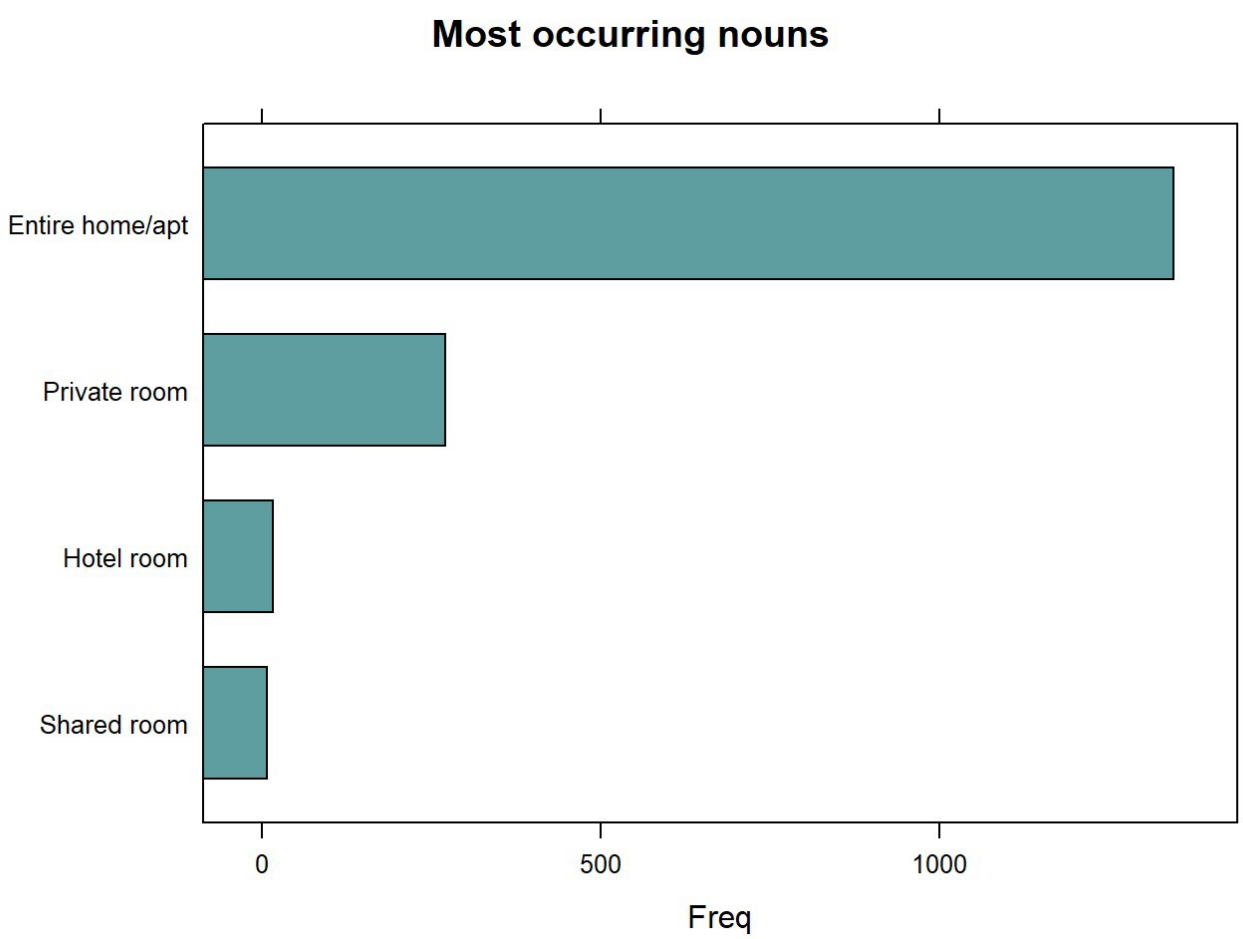
```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## netflix could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : huge
## could not be fit on page. It will not be plotted.
```

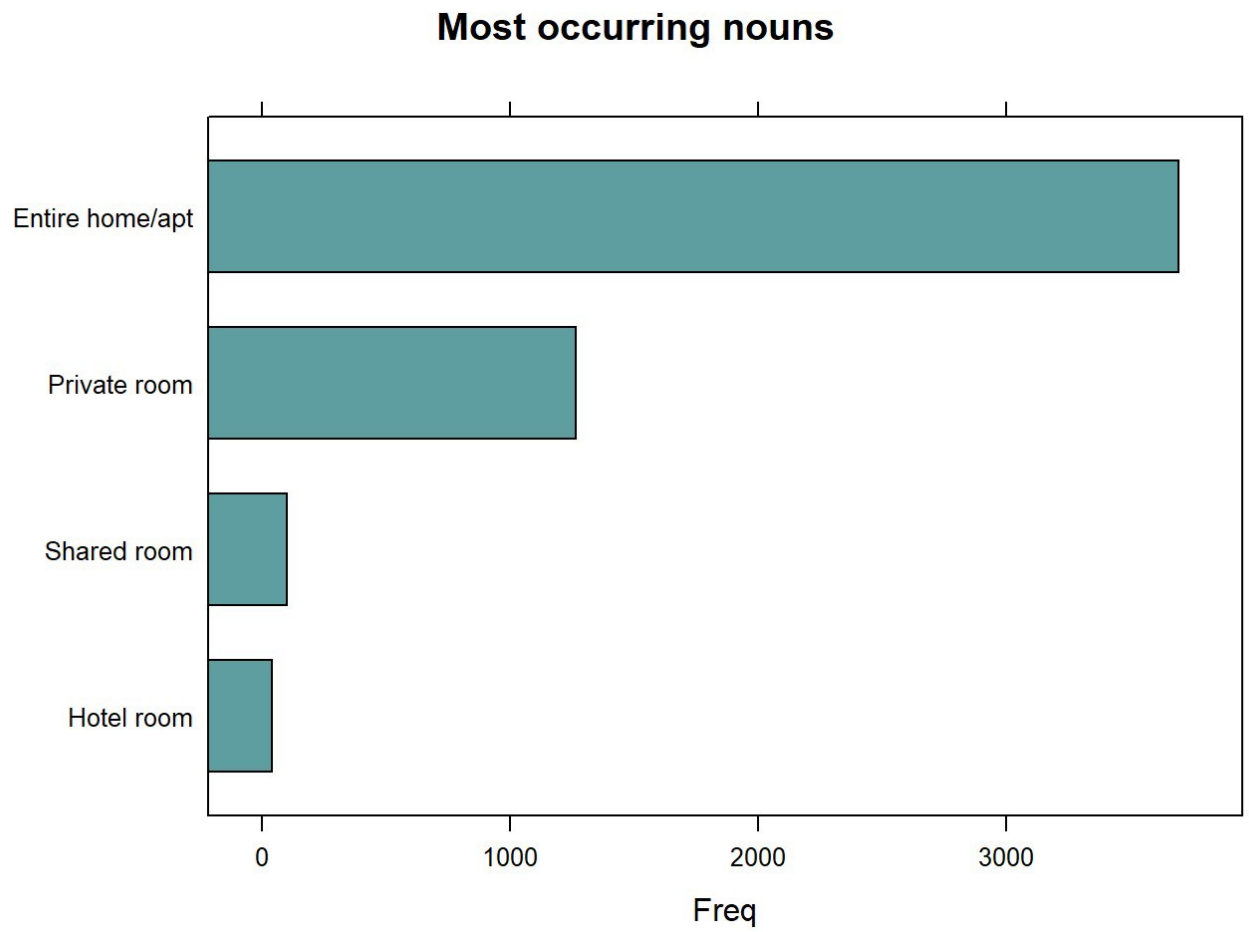
```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, :
## rental could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = stats$key, freq = stats$freq, min.freq = 1, : comfy
## could not be fit on page. It will not be plotted.
```

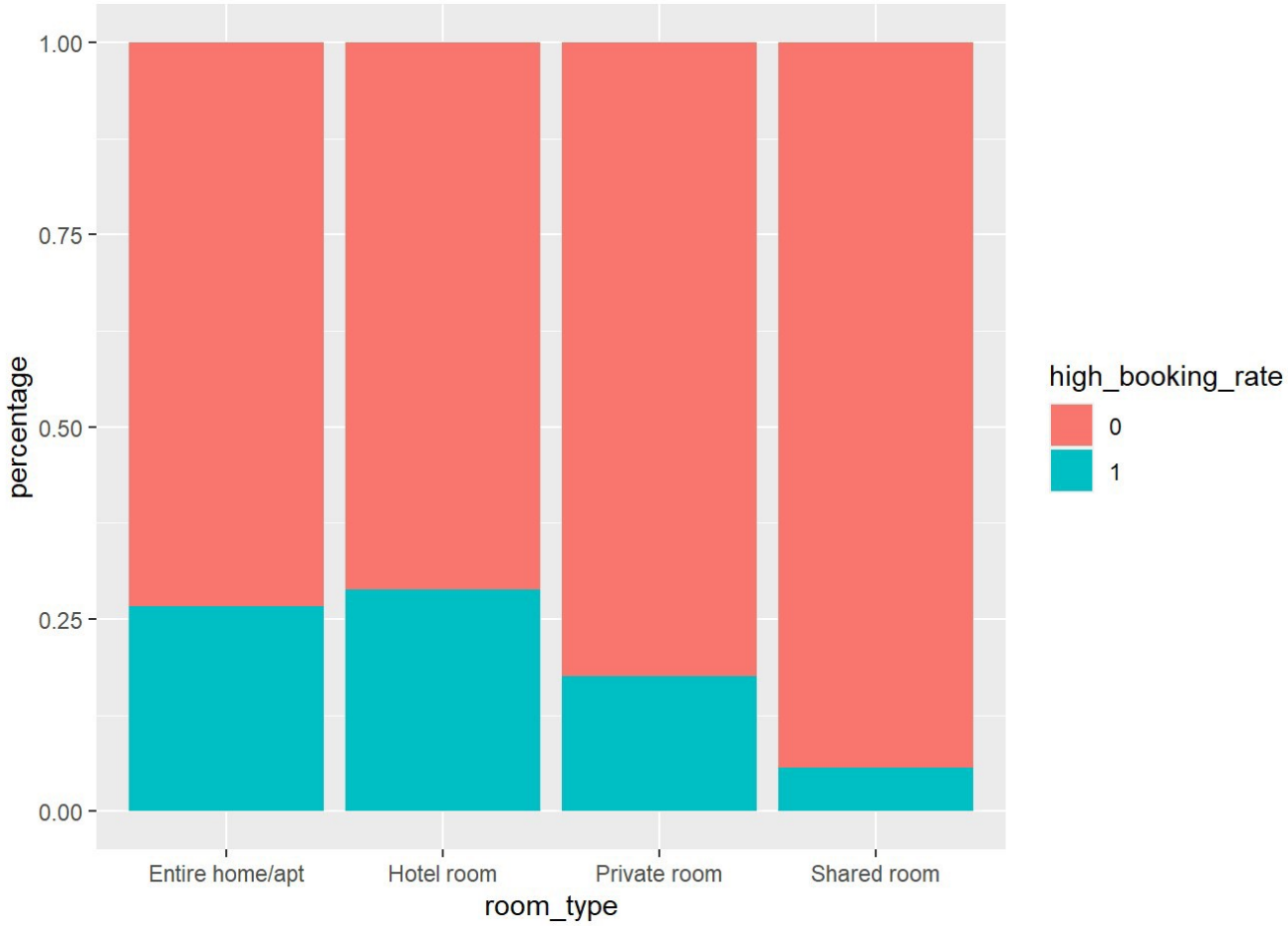

```
# Most popular property type
#split the data based on high booking rate
room_type1 <-
  df %>%
  filter(high_booking_rate == 1) %>%
  select(room_type)
room_type2 <-
  df %>%
  filter(high_booking_rate == 0) %>%
  select(room_type)

#Calculate the freq of each room type
stats <- subset(room_type1, room_type %in% c("NOUN"))
stats <- txt_freq(room_type1$room_type)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 30), col = "cadetblue", main = "Most occurring
nouns", xlab = "Freq")
```




```
stats <- subset(room_type2, room_type %in% c("NOUN"))
stats <- txt_freq(room_type2$room_type)
stats$key <- factor(stats$key, levels = rev(stats$key))
barchart(key ~ freq, data = head(stats, 30), col = "cadetblue", main = "Most occurring
nouns", xlab = "Freq")
```




```
#Count total number of each room type
roomtype <-
  df %>%
  select(room_type,high_booking_rate) %>%
  group_by(room_type,high_booking_rate) %>%
  summarise(count=n())
names(roomtype)[names(roomtype) == "count"] <- "category"
roomtypeall <-
  df %>%
  select(room_type) %>%
  group_by(room_type) %>%
  summarise(count=n()) %>%
  as_tibble()
names(roomtypeall)[names(roomtypeall) == "count"] <- "all"
# Calculate percentage of every room type
roomtypepercentage <-
  merge(roomtype, roomtypeall, by="room_type")
roomtypepercentage$percentage <- 0
roomtypepercentage$percentage <- roomtypepercentage$category/roomtypepercentage$all
roomtypepercentage$high_booking_rate <- as.factor(roomtypepercentage$high_booking_rat
e)
#show the result
ggplot(roomtypepercentage, aes(x = room_type, y = percentage, fill = high_booking_rate
)) + geom_bar(position = "fill",stat = "identity")
```



2.b. Data Mining and Predictive Analytics

2.b.1. Model Discrimination

Different models in the predictive analytics like KNN, decision tree, logistic and lasso model. In the variable decision process, the first branch of a preliminary variable filter was by the number of missing values since a variable with too many missing values is useless. Then, factor variable was filtered based on the bias of the data. In this process, we remove Host_Has_Profile_Pic and Requires_License because almost 99.9 percent of data in these two columns are same. After that, we chose text data that has usage values to create a dummy variable and added them into our prediction model to increase the prediction accuracy.

2.b.2. Model Determination

As seen from the above analysis, multiple models were tried on the training data set and XG Boost came out to give the best results in terms of Estimations and Accuracy. On dividing the training data to validate the models performance, an estimation of the 98.84% and an accuracy of 94.17% was sought. This can help the investors to prredict how well the booking rates for a new property can be, given the details of its property type, cleaning fee and other parameters used in the model. ### Dividing data into training and Validation

```
dfT <- dfTrain %>% sample_frac(.80)
dfVal <- setdiff(dfTrain, dfT)
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 3.6.3
```

```
##
## Attaching package: 'xgboost'
```

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

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

Defining the XGBoost matrix for Training

```
set.seed(123)

X_train = xgb.DMatrix(as.matrix(dfT %>% select(-high_booking_rate)))
y_train = dfT$high_booking_rate
```

Defining the XGBoost matrix for Testing

```
set.seed(123)

X_test = xgb.DMatrix(as.matrix(dfVal %>% select(-high_booking_rate)))
y_test = dfVal$high_booking_rate
xgb_trcontrol = trainControl(
  method = "cv",
  number = 10,
  allowParallel = TRUE,
  verboseIter = FALSE,
  returnData = FALSE
)
xgbGrid <- expand.grid(nrounds = c(100,200),  # this is n_estimators in the python cod
e above
                       max_depth = c(10, 15, 20, 25),
                       colsample_bytree = seq(0.5, 0.9, length.out = 5),
                       ## The values below are default values in the sklearn-api.
                       eta = 0.1,
                       gamma=0,
                       min_child_weight = 1,
                       subsample = 1
                      )
```

Training the model on Training data

```
set.seed(123)
xgb_model = train(
  X_train, y_train,
  trControl = xgb_trcontrol,
  tuneGrid = xgbGrid,
  metric = "Accuracy",
  method = "xgbTree"
)
```

Getting probabilities on predicted data fileds

```
resultsXG <-
  xgb_model %>%
    predict(X_test, type= 'prob') %>%
  bind_cols(dfVal,predictedClass=.)
resultsXG
```

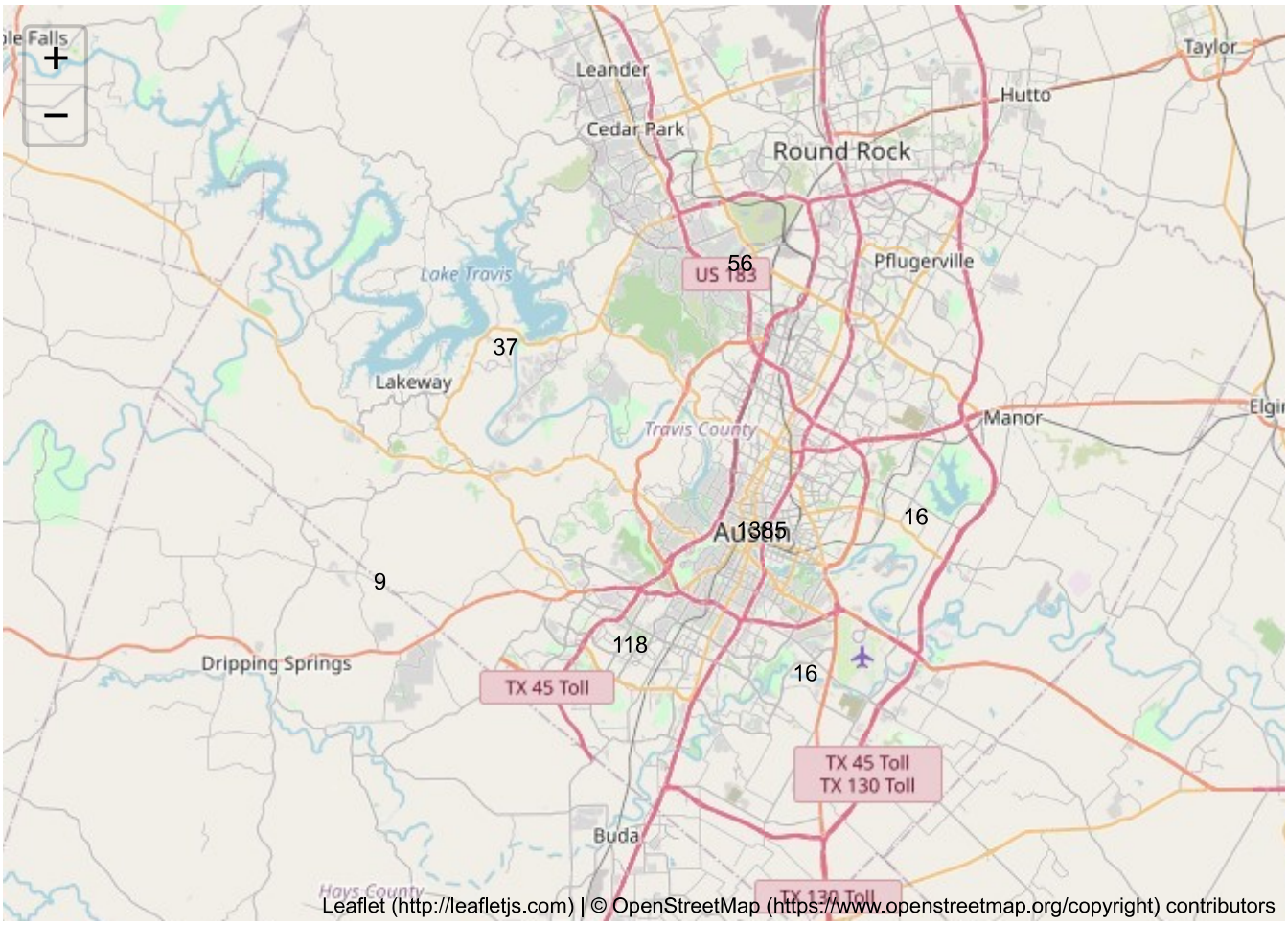
Learning estimates and accuracy

```
resultsXG %>%
  roc_auc(truth = high_booking_rate, `1`) %>%
  arrange(desc(.estimate))
resultsXG %>%
  roc_curve(truth = high_booking_rate, `1`) %>% # get values to plot an ROC curve
  ggplot(aes(x = 1 - specificity, y = sensitivity)) + # plota ROC curve for each model
  geom_line(size = 1.1) +
  geom_abline(slope = 1, intercept = 0, size = 0.4) +
  scale_color_manual(values = c("#183CF2","#66FFFF","#7F00FF","#B2FF66","#000000","#99
0000","#FF9933")) +
  coord_fixed()
resultsXGRaw <-
  xgb_model %>%
    predict(X_test, type= 'raw') %>%
  bind_cols(dfVal,predictedClass=.)
resultsXGRaw %>%
  xtabs(~predictedClass+high_booking_rate, .) %>%
  confusionMatrix(positive = '1')
```

2.b.3. Location Valuation

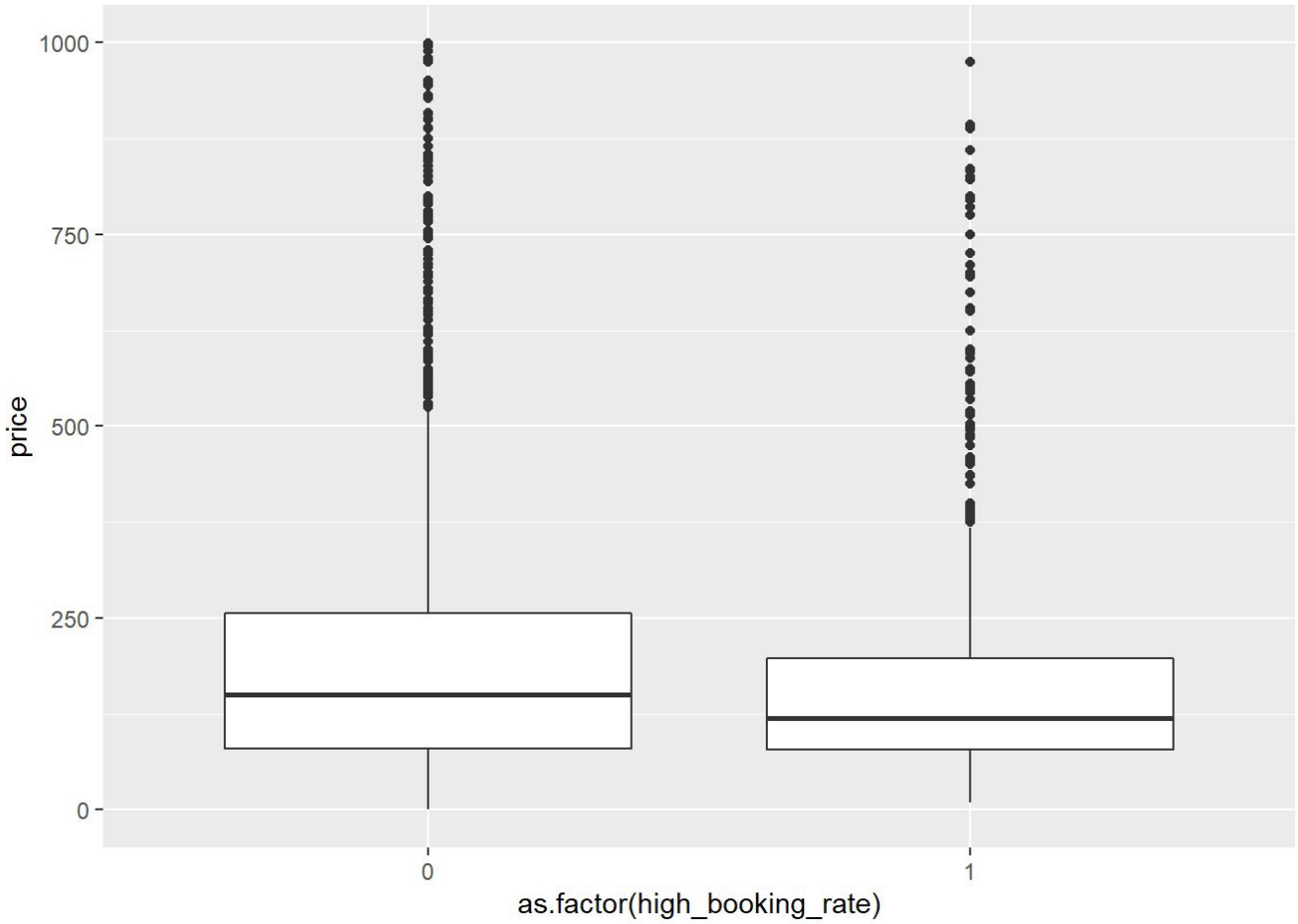
To get an overview of the house demand in Austin area, we have filtered the houses with high booking rates, and use the library leaflet to create a map to visualize the house numbers. Especially, we gave used the function addSearchOSM() to add a search button to enhance the interaction of the map. In this sense, we can employ the city background and tourist information to find the attractions and shopping mall near the top three demands of AirBNB housing regions.

```
# Filter the high booking rate
dfhigh <-
  df %>%
  filter(high_booking_rate == 1) %>%
  select(latitude, longitude)
# Using latitude and longtitude to create a plot
Plothigh <-
  leaflet() %>%
  addProviderTiles("OpenStreetMap") %>%
  addCircleMarkers(data = dfhigh, radius = 3, clusterOptions = markerClusterOptions
())%>%
  addSearchOSM() # Add a search function
Plothigh
```



In terms of formulating a price strategy for an investor to enhance booking rate, we have analyzed the price through boxplot, and its components of price including cleaning fee, the fee for extra person and the number of guest included. Furthermore, in order to obtain a whole picture of how the price in different places varies, we also create a map to visualize it. Lastly, for the map provider tiles, we have tried a lot of maps and find the “Open Street Map” is more useful in our case since it has the street names, the name of building and all the components that we need for analyzing the house price.

```
# Create box plot for price, cleaning fee, guest include
# the following is just one example
plotPrice <- df %>%
  ggplot(mapping = aes(x = as.factor(high_booking_rate), y = price)) +geom_boxplot()
plotPrice
```

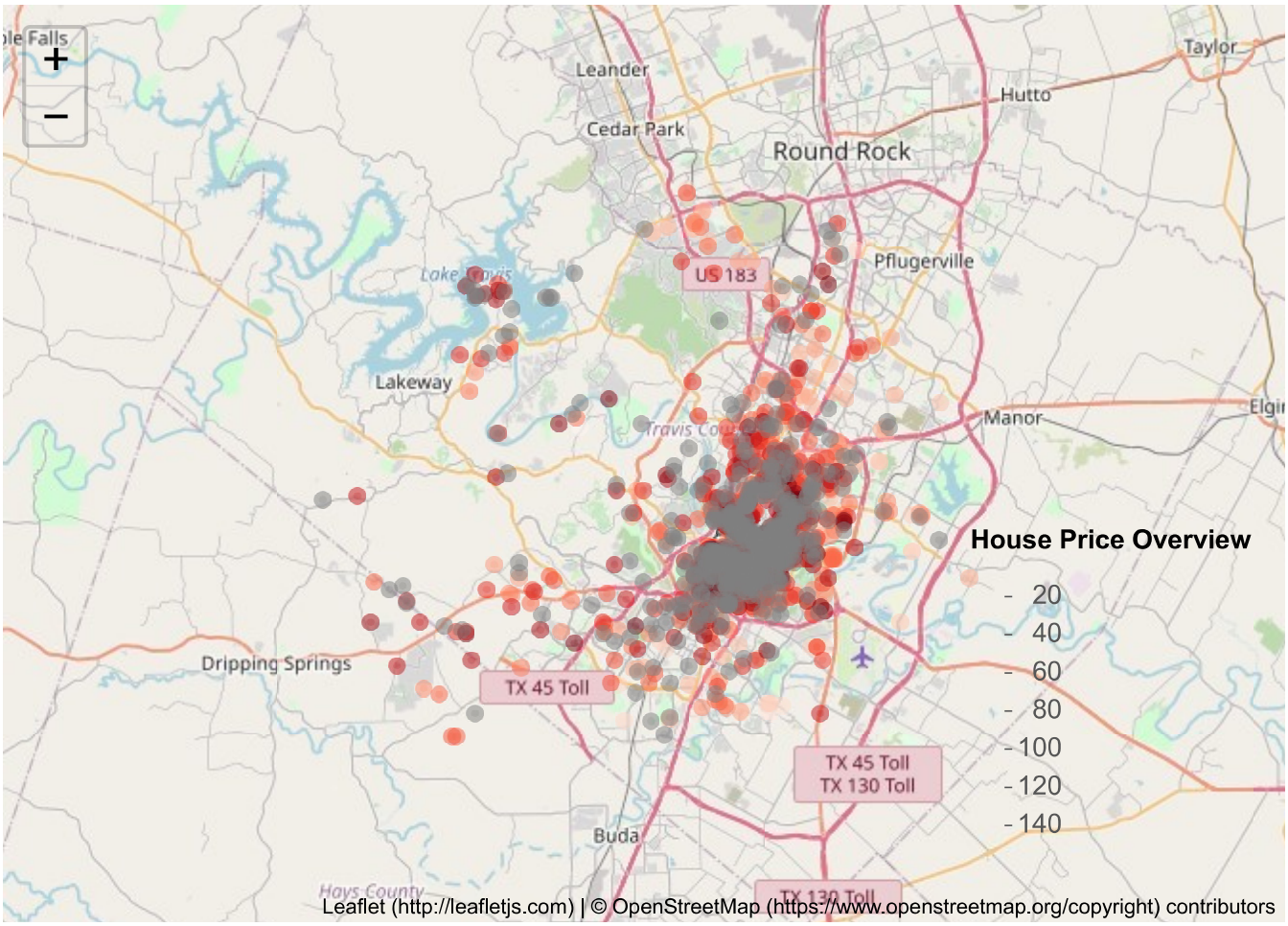



```
# Overview of Price
dfprice <-
  df %>%
  filter(high_booking_rate == 1) %>%
  select(latitude, longitude, price) %>%
  arrange(price)

# Create a map for the overiew of the house in different area
pal <- colorNumeric(palette = "Reds", domain = c(10:145), reverse = FALSE)
Plotprice <-
  leaflet() %>%
  addProviderTiles("OpenStreetMap") %>%
  addCircleMarkers(data= dfprice, radius = 2, color = ~pal(price))%>%
  addLegend(title = "House Price Overview", pal = pal, values = c(10:145), position =
"bottomright")
```

```
## Assuming "longitude" and "latitude" are longitude and latitude, respectively
```

```
## Warning in pal(price): Some values were outside the color scale and will be
## treated as NA

## Warning in pal(price): Some values were outside the color scale and will be
## treated as NA
```

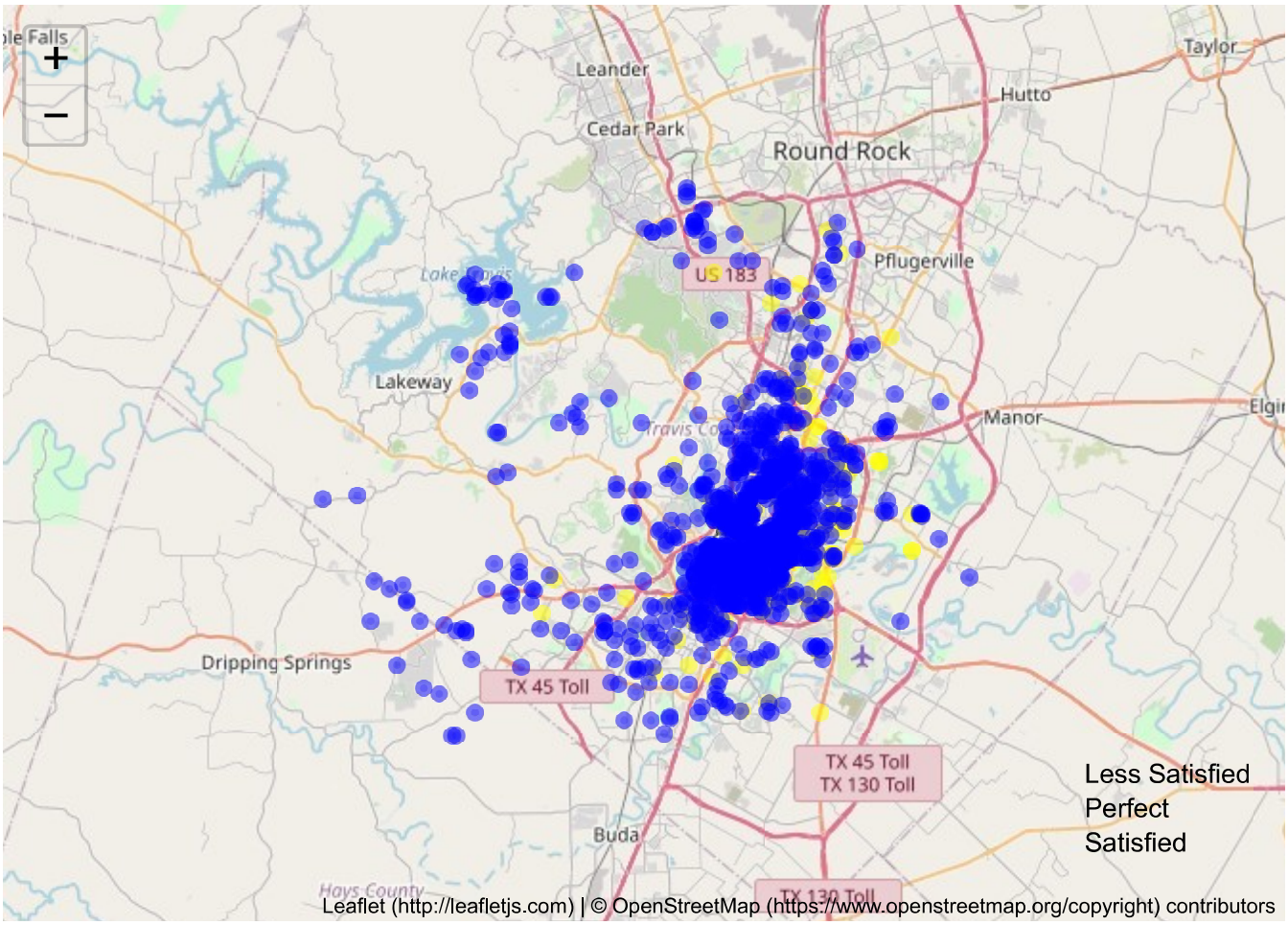
```
Plotprice
```



```
# Overview of Review Scores Location
dfRS <-
  df %>%
  filter(high_booking_rate == 1) %>%
  select(latitude, longitude, review_scores_location)
dfRS <-
dfRS %>%
   mutate(review =ifelse(review_scores_location == 8, "Less Satisfied", ifelse(review_
scores_location == 9, "Satisfied", "Perfect"))) %>%
  arrange(review_scores_location)
pal <- colorFactor(palette = c("red", "yellow", "blue"), levels = c("Less Satisfied",
"Satisfied", "Perfect"))
plotreview <-
  leaflet() %>%
  addProviderTiles("OpenStreetMap") %>%   # CartoDB is another provider
  addCircleMarkers(data = dfRS, radius = 2,
                         color = ~pal(review), label = ~review) %>%
  addLegend(position = "bottomright",pal = pal, values = c("Less Satisfied", "Satisfie
d", "Perfect"))
```

```
## Assuming "longitude" and "latitude" are longitude and latitude, respectively
```

```
plotreview
```



2.b.4. PCA

A new dataframe for use in principle component analysis (PCA) testing was created that contained eight variables. The PCA dataframe included the variables: Review_Scores_Value, Review_Scores_Rating, Beds, ROI, Instances, Availability_365, Amenities and ID. These variables were chosen due to their correlations with ROI and their comparatively stable multicollinearity with the broader dataset and their ability to be scaled. Matrix scaling of the data with the previously defined variables, sans ID, was conducted. The scaled dataframe was then used to create a correlation matrix from which the related eigenvalues could be discerned and removed from the model. Unfortunately, the proportion of variance amidst all the variables was slim and no discernable elbow could be detected from the cumulative Scree Plot. The first two variables, ROI and Instances, carried the greatest descriptive weight, 24% and 21% respectively, and were chosen to conduct further descriptive tests. A new scaled dataframe that contained only the ROI and Instances variables was scaled and had its eigenvalues determined. A scaled plot of the calculated PCA scores was created to discern word locations along the scaled factors. This plot would also be used as a comparative tool for the K Means clustering as seen in 3.a. Synthesis.

```
### PCA with Comparative Variables

## Expand/Clean DF
aus <- full %>%
  mutate(price=as.numeric(str_remove_all(price,"[$]"))) %>%
    filter(str_detect(`{randomcontrol}`,"102"))
```

```
## Warning: NAs introduced by coercion
```

```
##Create DF
austin <- tibble(id = aus$id, text =aus$amenities, avail= aus$availability_365, price=
aus$price, review_value = aus$review_scores_value, review_rating=aus$review_scores_rat
ing, beds = aus$beds)
aus2 <- austin %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```
count2 <- aus2 %>% count(word, sort=TRUE)
aus2 <- aus2 %>% left_join(count2, by = "word") %>%
  mutate(unavail = 365-avail)
aus_n <- aus2 %>%
  select(word,n,id) %>%
  rename(instances=n)
inc_aus <- aus2 %>%
    group_by(id) %>%
  mutate(income = unavail*price) %>%
  left_join(aus_n, by=c("word","id"))
aus_roi_amen <- inc_aus %>%
  left_join (roi_amen[c("ROI","word")], by="word") %>%
  drop_na()
##PCA - Scale the data
scale_amen1 <- scale(aus_roi_amen[c("ROI","instances","income","beds","unavail","revie
w_rating","review_value","price")])
##PCA - Create correlation mattrix and eigen values
amen2.cov <- cov(scale_amen1)
amen2.eigen <- eigen(amen2.cov)
str(amen2.eigen)
```

```
## List of 2
##  $ values : num [1:8] 1.93 1.66 1.47 1.19 0.8 ...
##  $ vectors: num [1:8, 1:8] -0.0443 0.0113 -0.4741 -0.5279 0.058 ...
##  - attr(*, "class")= chr "eigen"
```

```
##PCA - Extract the loadings
phi2 <- amen2.eigen$vectors[,]
phi2
```

```
##             [,1]        [,2]        [,3]        [,4]          [,5]        [,6]
## [1,] -0.04426145  0.01285779 -0.01849837  0.71572360  0.6850721255  0.12602713
## [2,]  0.01129467  0.06502381  0.11803296  0.69158645 -0.7048457777 -0.08100875
## [3,] -0.47413242 -0.19781252  0.52925021 -0.04611276 -0.0125921121  0.25113270
## [4,] -0.52790282 -0.09291687 -0.33670437  0.04091068  0.0505026329 -0.74320675
## [5,]  0.05802436 -0.07965807  0.75533566 -0.03091953  0.1425846034 -0.45420595
## [6,]  0.14024859 -0.68988587 -0.07511819  0.04172879 -0.0536630978  0.12212215
## [7,]  0.21742644 -0.66383937 -0.09096414  0.05289668  0.0007522853 -0.13218181
## [8,] -0.65124471 -0.15771051 -0.08832464 -0.01234801 -0.0891564479  0.35134305
##              [,7]         [,8]
## [1,]  0.007622330 -0.002896703
## [2,] -0.006803014  0.002112332
## [3,] -0.062002175 -0.621905154
## [4,]  0.117092124 -0.170364913
## [5,]  0.067065908  0.433207273
## [6,]  0.691726152  0.026928841
## [7,] -0.695051192 -0.020043484
## [8,] -0.127521441  0.628809572
```

```
##PCA - Find the correlations
phi2 <- -phi2
row.names(phi2) <- c("ROI","instances","income","beds","unavail","review_rating","revi
ew_value","price")
colnames(phi2) <- c("PC1","PC2","PC3","PC4","PC5","PC6","PC7","PC8")
phi2
```

```
##                       PC1         PC2         PC3         PC4           PC5
## ROI            0.04426145 -0.01285779  0.01849837 -0.71572360 -0.6850721255
## instances     -0.01129467 -0.06502381 -0.11803296 -0.69158645  0.7048457777
## income         0.47413242  0.19781252 -0.52925021  0.04611276  0.0125921121
## beds           0.52790282  0.09291687  0.33670437 -0.04091068 -0.0505026329
## unavail       -0.05802436  0.07965807 -0.75533566  0.03091953 -0.1425846034
## review_rating -0.14024859  0.68988587  0.07511819 -0.04172879  0.0536630978
## review_value  -0.21742644  0.66383937  0.09096414 -0.05289668 -0.0007522853
## price          0.65124471  0.15771051  0.08832464  0.01234801  0.0891564479
##                       PC6          PC7          PC8
## ROI           -0.12602713 -0.007622330  0.002896703
## instances      0.08100875  0.006803014 -0.002112332
## income        -0.25113270  0.062002175  0.621905154
## beds           0.74320675 -0.117092124  0.170364913
## unavail        0.45420595 -0.067065908 -0.433207273
## review_rating -0.12212215 -0.691726152 -0.026928841
## review_value   0.13218181  0.695051192  0.020043484
## price         -0.35134305  0.127521441 -0.628809572
```

```
##PCA - Calculate Principal Components Scores
PC1a <- as.matrix(scale_amen1)%*%phi2[,1]
PC2a <- as.matrix(scale_amen1)%*%phi2[,2]
PC3a <- as.matrix(scale_amen1)%*%phi2[,3]
PC4a <- as.matrix(scale_amen1)%*%phi2[,4]
PC5a <- as.matrix(scale_amen1)%*%phi2[,5]
PC6a <- as.matrix(scale_amen1)%*%phi2[,6]
PC7a <- as.matrix(scale_amen1)%*%phi2[,7]
PC8a <- as.matrix(scale_amen1)%*%phi2[,8]
##PCA - Create DF with PC Scores
PCz <- data.frame(word=row.names(aus_roi_amen),PC1a,PC2a,PC3a,PC4a,PC5a,PC6a,PC7a,PC8
a)
PCz
```

w... <fctr>	PC1a <dbl>	PC2a <dbl>	PC3a <dbl>	PC4a <dbl>	PC5a <dbl>	
1	0.0933069632	-1.440073e+00	-1.4447628756	0.0359432105	0.5885361428	4.697
2	0.0941160365	-1.440340e+00	-1.4444783181	0.0224468114	0.5762640612	4.674
3	0.0941289640	-1.440327e+00	-1.4444461243	0.0224444648	0.5759386538	4.674
4	0.1246385397	-1.344057e+00	-1.2572861370	0.8746883350	-0.7120962795	3.053

w...	PC1a	PC2a	PC3a	PC4a	PC5a	
<fctr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
5	0.1011151393	-1.440241e+00	-1.4380167507	-0.0634472506	0.4513911801	4.460
6	0.1196200363	-1.474864e+00	-1.4788013083	-0.7370055813	0.3919238580	4.142
7	0.1307826134	-1.492964e+00	-1.4987837478	-1.1076693135	0.3344426057	3.931
8	0.1045897966	-1.423922e+00	-1.4078178983	0.1021517120	0.2631460406	4.237
9	0.1247183216	-1.342319e+00	-1.2543325814	0.8959281591	-0.7269906618	3.038
10	0.1104028580	-1.306976e+00	-1.2085822043	1.5265454348	-0.7474047761	3.223

1-10 of 10,000 rows | 1-7 of 9 columns Previous 1 2 3 4 5 6 ... 1000 Next

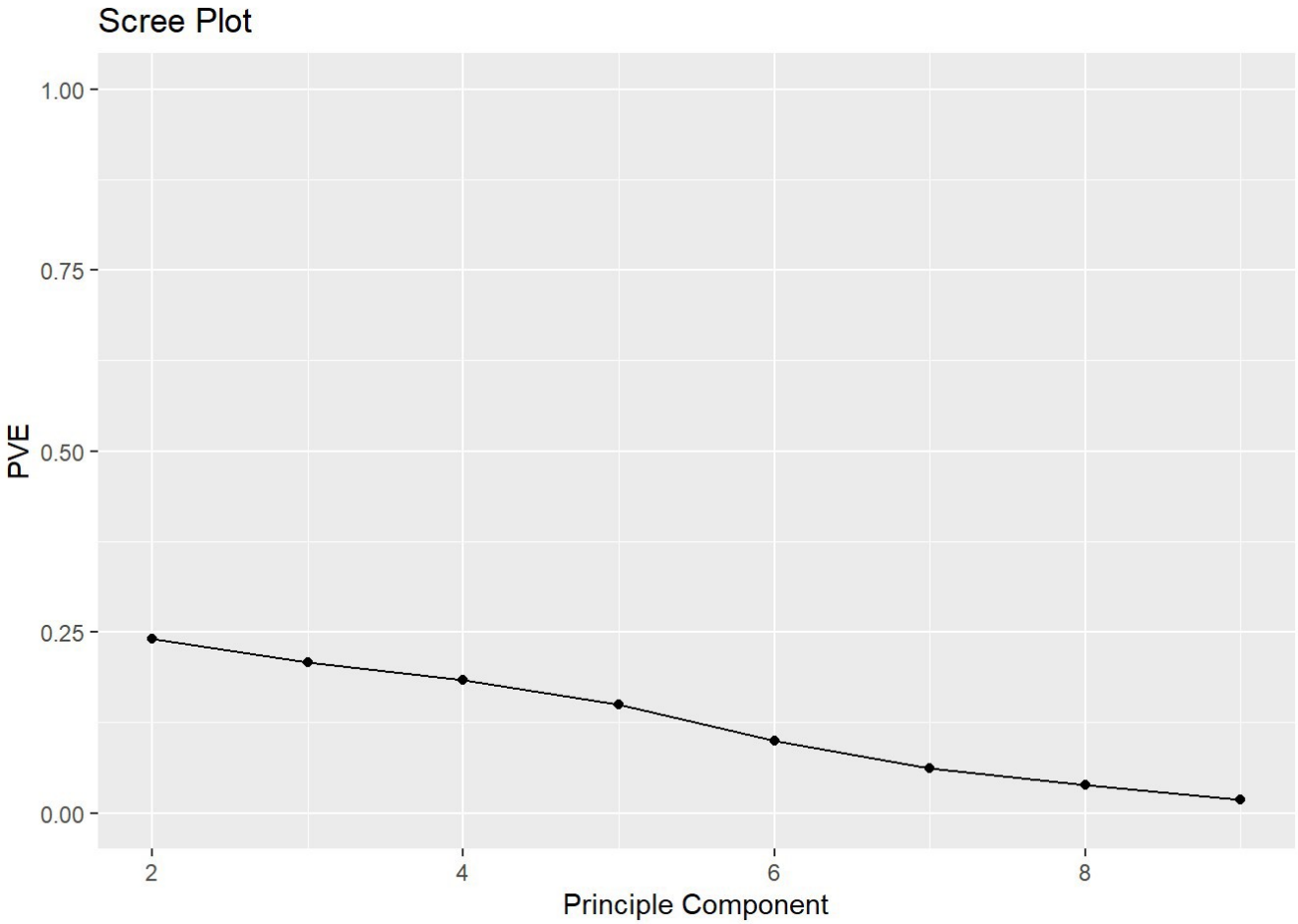
```
##PCA - Proportion of Variance
PVE2 <- amen2.eigen$values/sum(amen2.eigen$values)
round(PVE2,2)  #ROI = 24%, #Instances = 21%
```

```
## [1] 0.24 0.21 0.18 0.15 0.10 0.06 0.04 0.02
```

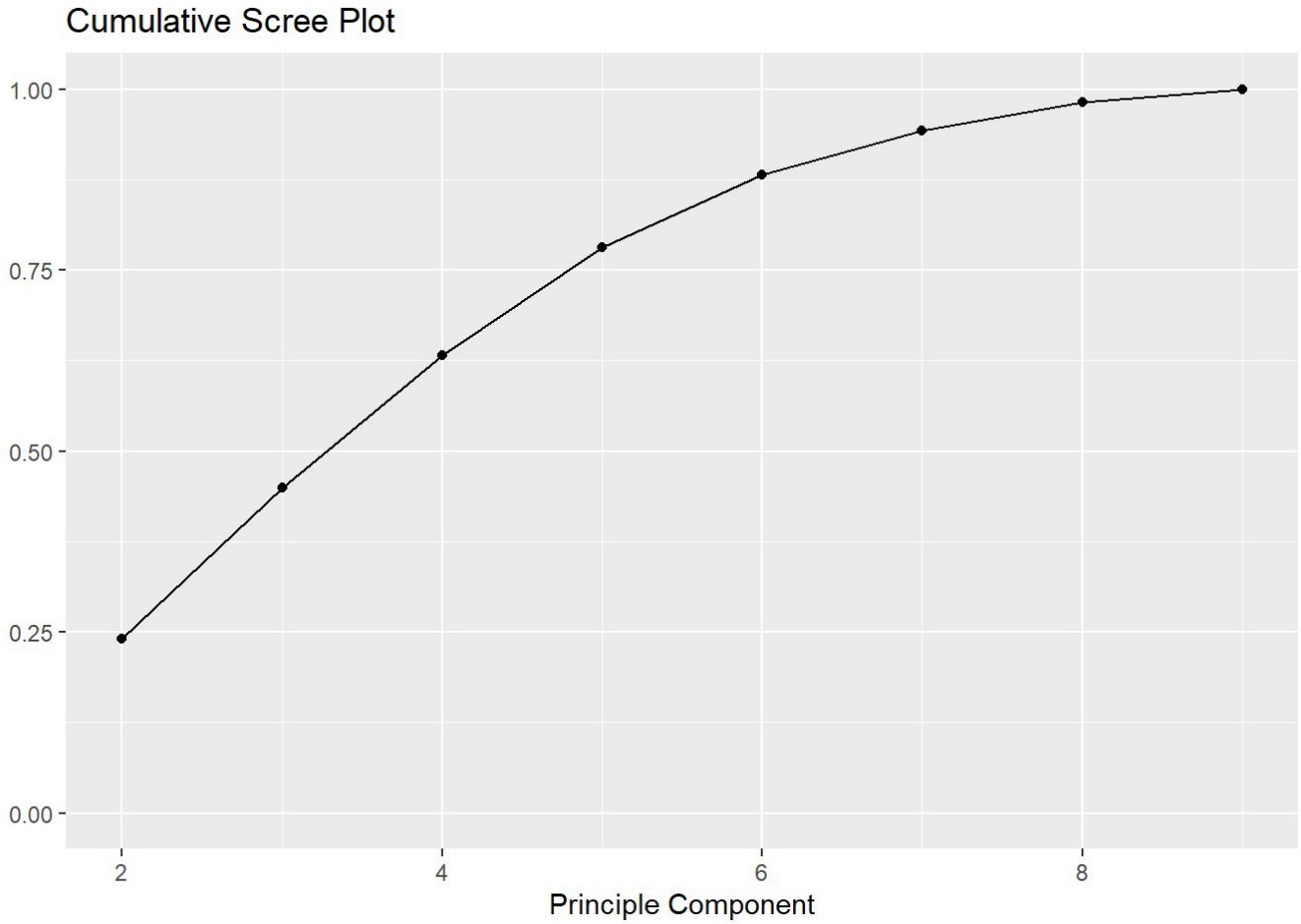
```
PVE2
```

```
## [1] 0.24089356 0.20796265 0.18312967 0.14933324 0.10006104 0.06107708 0.03924415
## [8] 0.01829861
```

```
##PCA - Scree Plot
PVEplot2 <- qplot(c(2:9),PVE2)+
  geom_line()+
  xlab("Principle Component")+
  ylab("PVE")+
  ggtitle("Scree Plot")+
  ylim(0,1)
PVEplot2
```




```
##PCA - Cumulative Scree Plot
cumPVE2 <- qplot(c(2:9),cumsum(PVE2))+
  geom_line()+
  xlab("Principle Component")+
  ylab(NULL)+
  ggtitle("Cumulative Scree Plot")+
  ylim(0,1)
cumPVE2
```




```
### PCA with First 2 Variables (ROI, Instances)

##PCA -Set Word as DF Index
rownames(roi_amen) <- roi_amen$word
```

```
## Warning: Setting row names on a tibble is deprecated.
```

```
##PCA - Scale the data
scale_amen <- scale(roi_amen[c("ROI","Instances")])
##PCA - Create correlation mattrix and eigen values
amen.cov <- cov(scale_amen)
amen.eigen <- eigen(amen.cov)
str(amen.eigen)
```

```
## List of 2
##  $ values : num [1:2] 1.204 0.796
##  $ vectors: num [1:2, 1:2] -0.707 -0.707 0.707 -0.707
##  - attr(*, "class")= chr "eigen"
```

```
##PCA - Extract the loadings
phi <- amen.eigen$vectors[,]
phi
```

```
##            [,1]       [,2]
## [1,] -0.7071068  0.7071068
## [2,] -0.7071068 -0.7071068
```

```
##PCA - Find the correlations
phi <- -phi
row.names(phi) <- c("ROI","Instances")
colnames(phi) <- c("PC1","PC2")
phi
```

```
##                  PC1        PC2
## ROI        0.7071068 -0.7071068
## Instances 0.7071068  0.7071068
```

```
##PCA - Calculate Principal Components Scores
PC1 <- as.matrix(scale_amen)%*%phi[,1]
PC2 <- as.matrix(scale_amen)%*%phi[,2]
##PCA - Create DF with PC Scores
PC <- data.frame(word=row.names(roi_amen),PC1,PC2)
```

2.b.5. K Means Clustering

After conducting a PCA and determining the viability of the variables ROI and Instances, a K Means clustering model was used in order to further extrapolate usable direction for the investor. The K Means model was chosen as a dimensionality reduction technique due to its potential direct applicability of the groupings the model determines. With ease of understanding at the root of the final proposal, a minimal number of clusters was determined to be the best course of action. In accordance with the PCA, a k means of two was initially used. Further experimentation resulted in a slight increase to a k equal to three which was used as the basis for the clustering analysis. All the k means clustering models were conducted on four different data sets. The first data set included all of the words in the Austin data frame (314 Words), the second data set is the top 10% (31) words ordered by ROI, the third data set is the bottom 20% (62) words ordered by ROI, and the fourth data set is a combination of the top 10% and bottom 20%. Expansion of the data set with the Words that have the least ROI from 10% to 20% was determined primarily due to the correlation matrix. No correlation can be seen between any of the bottom 31 Words, creating issues within the k means algorithm and forcing an expansion of the bottom ROI data set.

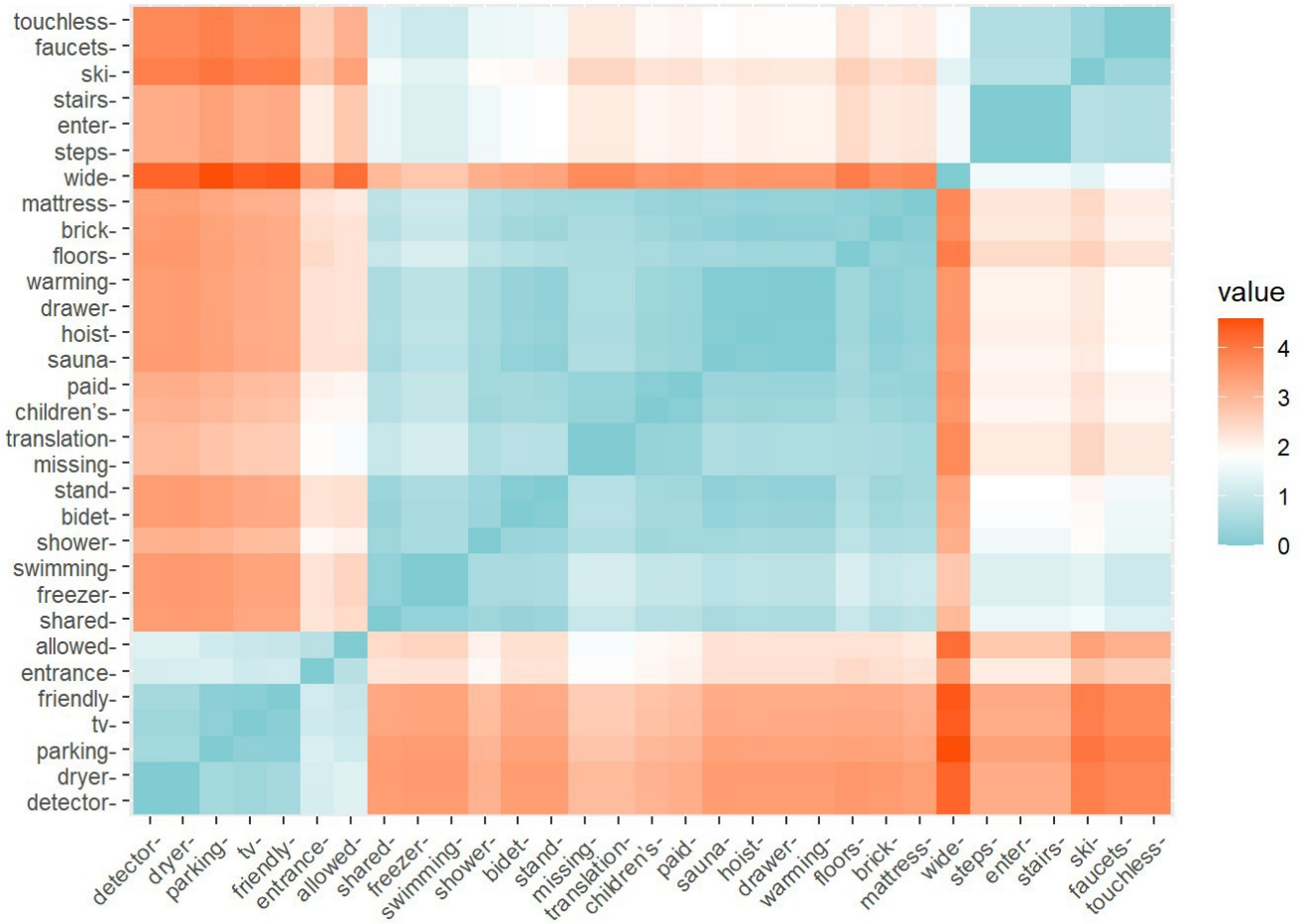
```
### KMeans Clustering

## Data Prep
set.seed(101)

## DFs of ROI
## Top of ROI and Overall Data Set
rev_amen <- roi_amen %>% arrange(desc(ROI))  ##sets words from highest ROI to least RO
I
rownames(rev_amen) <- rev_amen$word                  ##creates DF of just correctly or
dered words
```

```
## Warning: Setting row names on a tibble is deprecated.
```

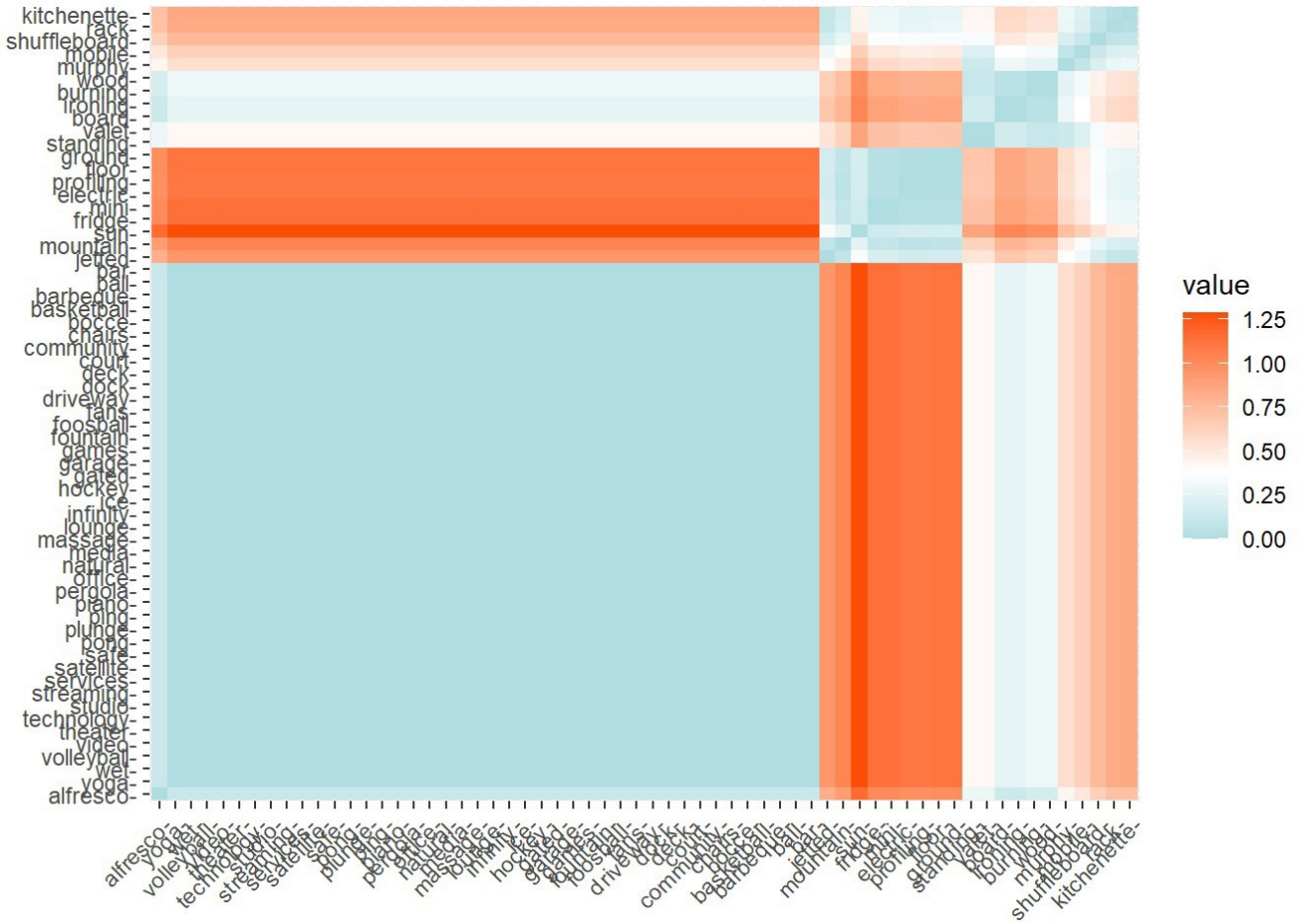
```
rownames(scale_amen) <- roi_amen$word         ##appends word to index (might not need)
pscale_amen <- data.frame(scale_amen) %>%
  arrange(desc(ROI))                          ##Arranges DF by ROI but word is dropped
rownames(pscale_amen) <- rev_amen$word     ##Appends word by new ROI order for Top Wor
ds
#pscale_amen
## Bottom arrangement of ROI
revn_amen <- roi_amen %>% arrange(-desc(ROI))  ##sets words from least ROI to highest
ROI
rownames(revn_amen) <- revn_amen$word                  ##creates DF of just correctly
ordered words
```

```
## Warning: Setting row names on a tibble is deprecated.
```

```
scale_amenn <- scale_amen                      ## New scale DF
revn_word <- revn_amen$word                       ## DF of just the rearranged words
rownames(scale_amenn) <- roi_amen$word         ##appends word to index (might not nee
d)
pscalen_amen <- data.frame(scale_amenn) %>%
  arrange(-desc(ROI))                          ##Arranges DF by ROI but word is dropped
rownames(pscalen_amen) <- revn_amen$word     ##Appends word by new ROI order for Top W
ords

## Cluster Matrix Visualizations
## Make a visualization matrix of Top 31 (10%) Words
pscale_amen31 <- pscale_amen %>%
 head(n=31)                                 ## Reduces size of visualization matrix
dist_amen31 <- get_dist(pscale_amen31)
top31viz <- fviz_dist(dist_amen31,gradient = list(low='#00AFBB',mid="white",high='#FC4
E07'))
top31viz
```




```
## Make a visualization matrix of Bottom 62 (20%) Words
pscale_amen31b <- pscalen_amen %>%
 head(n=62)                                  ## Reduces size of visualization matrix
dist_amen31b <- get_dist(pscale_amen31b)
bot31viz <- fviz_dist(dist_amen31b,gradient = list(low='#00AFBB',mid="white",high='#FC
4E07'))
bot31viz
```

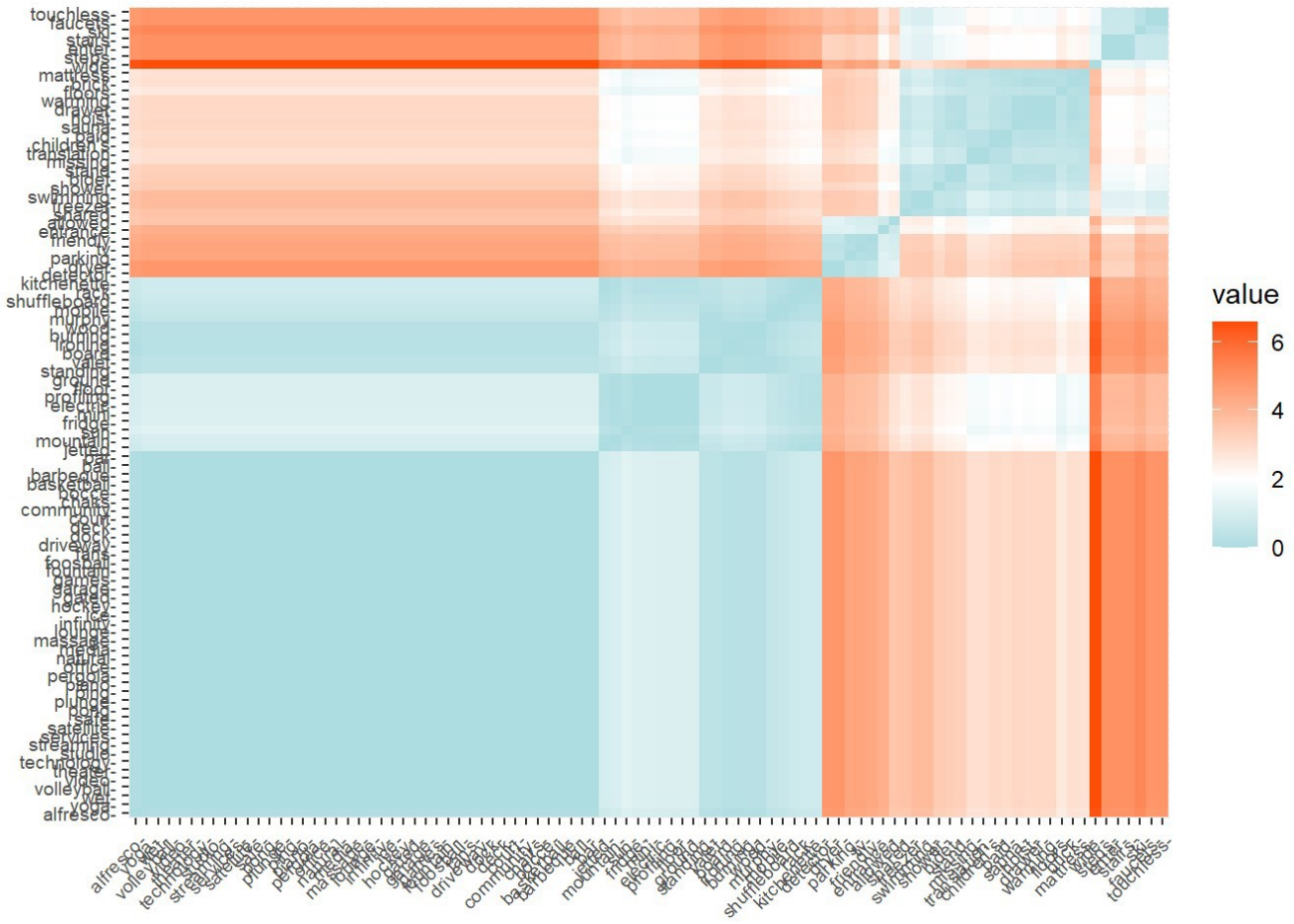



```
## Combine top 10% and bottom 20%
comb_scale_amen <- pscale_amen31 %>% bind_rows(pscale_amen31b)       ## Combined Values
top31word <- data.frame(word=rev_amen$word %>% head(n=31))           ## Top 10% words
bot31word <- data.frame(word=revn_amen$word %>% head(n=62))          ## Bottom 20% word
s
words_com <-
  bind_rows(top31word,bot31word)                                     ## Top and Bottom
Words
```

```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character
```

```
## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector
```

```
rownames(comb_scale_amen) <- words_com$word                    ## Indx Words
## Make a visualization matrix of Bottom 62 (20%) and Top 31 (10%) Words
pscale_amen31c <- comb_scale_amen
dist_amen31c <- get_dist(pscale_amen31c)
combviz <- fviz_dist(dist_amen31c,lab_size= 7,gradient = list(low='#00AFBB',mid="whit
e",high='#FC4E07'))
combviz
```



2.b.6. Evaluation of Austin's Allure

Starting with the dataset, our team wanted to analyze how many total AirBNBs there are in Austin with a high booking rate. Given that Texas is a large state, the results show that there are over 40,000 AirBNB's in Austin with a high booking rate. By adding the description to this analysis, we were able to incorporate text mining to help derive the most popular neighborhoods, living conditions, and common attractions that are near the AirBNBs. By counting the frequency of words that occur most in the descriptions, we were able to see which words are the most common when searching for an AirBNBs in Austin. We selected the top 10 words and plotted on a bar graph so an investor can visually see what are the most common words that occur within this selective search. Additionally, stop words were removed from the data to ensure that we were gathering the most accurate data possible to make good predictions. Word Clouds were then made to help enhance the visualization to show the top words that are used to describe AirBNBs in Austin, the most popular attractions, living conditions, and lastly the most popular neighborhoods that have AirBNBs with high booking rates.

3. Results and Findings

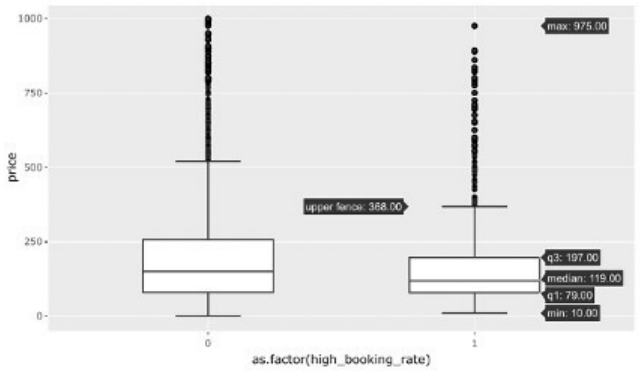
3.a. Synthesis

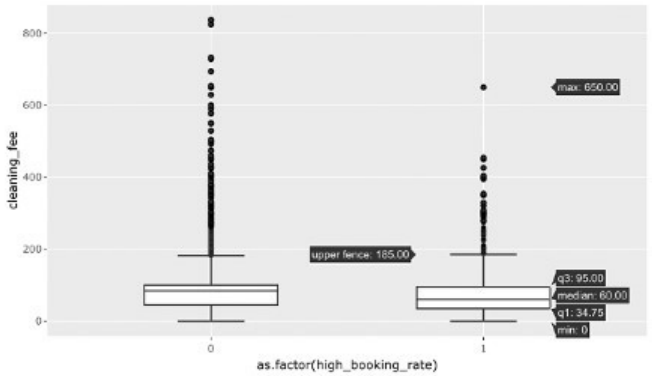
Based on all studies, models and plots, it can be concluded that for new investments, it is advised to study the host verifications and category if the host is a superhost. Amenitites listing should be done accurately and in an appealing way as the count of amenities had a very distinctive effect on the price as well as booking rates. Availibility is another factor that an investor should look at are as it affects the booking rates. It is also seen that there are too many correlated features and these are eliminated in the model. They cannot be left out to be studied before investing as they might affect the price and booking strategy. These concepts will be explained in depth by the following data synthesis.

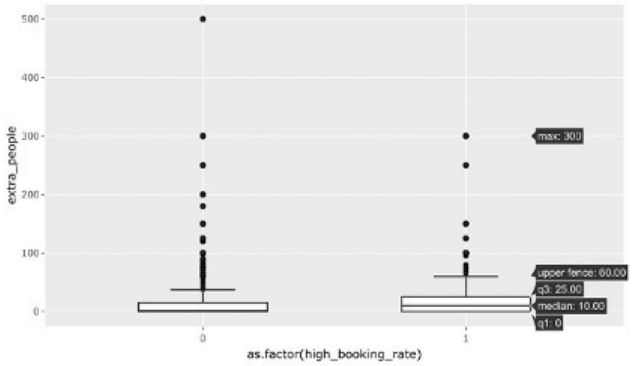
3.a.1. Housing Value

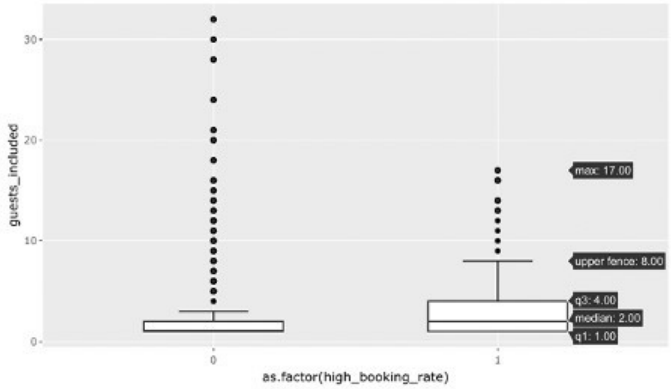
The demand in the current houses with high booking rate, we can see from the map below.

Houses located in the downtown are in extremely high demand while small portion of the houses are in the rural area. Specially, there is a small peak of demand in the rural area, which is around Canyon Lake, Comal County, Texas. When we look at closely, we can see that the TOP 3 houses are unsurprisingly located near places of interests and shipping center. The last thing we should pay attention to is that it might not be a wise decision to buy/rent a houses near Austin International airport. In terms of the price for the Austin Airbnb house, we can see from the map below.









Equation

The price for the most of the house is somewhere between $79 to $197, the average price is $119 although there is a tiny portion of houses worth between $368 and $975. At first glance, we can see that the houses located in the downtown is the most expensive. However, when we take a close look, there is not an apparent pattern of where the house tends to more pricey in the downtown. Airbnb renters tend to be fine with paying a cleaning fee of around $60 after the whole stay and approximately $10 for the extra per person. Most of the houses with high booking rate tend to be allowed to live with 2 people. In conclusion, the housing purchase strategies are presented as follows: 1. Investing in a house in the downtown seems to be a safe decision, especially those near place of interest and shopping center.

2. Creating a luxury hotel is not a great idea due to the current fair market value. 3. The Austin market seems to be good option for a new investor since you do not need to dig into too much in terms of the accurate location once it is in the downtown.

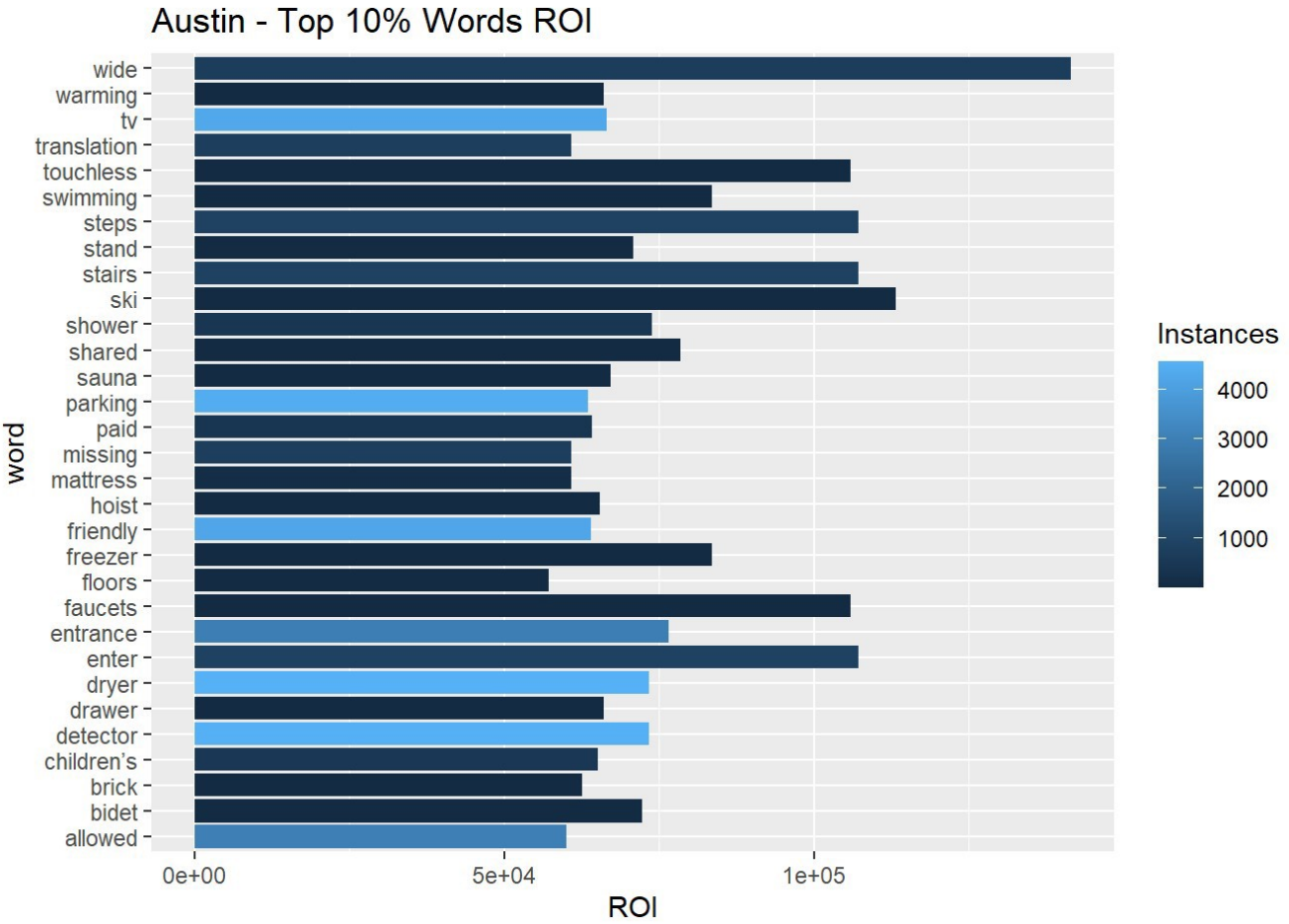
4. It is fine to ask for a reasonable cleaning fee and charge for additional people. Also, a listing that can

accommodate two people will always be popular.

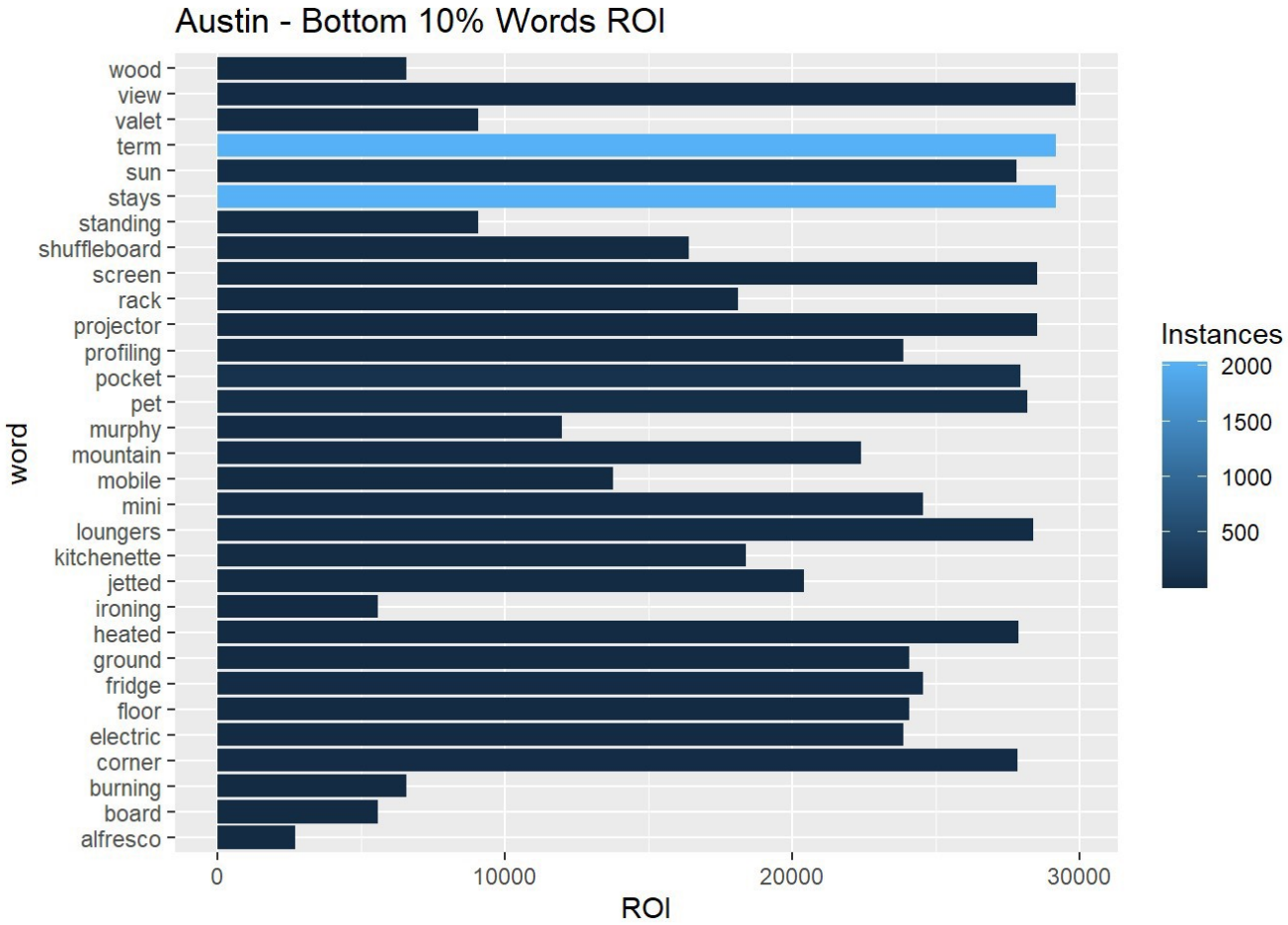
3.a.2. Return on Investment (ROI)

Reduction of the Amenities variable into its individual Words and using these as a way to provide insight into the most valuable updates and renovations to a property is directly in line with our primary research question. The provided visualization of the top 10% most valuable words in relation to the Amenities variable gives investors a generally clear answer as to what they should do to provide the largest returns from their investment property. While some words like “wide” are open to interpretation, they can be further explored by text-mining n-grams, or they can be used as vague distinguishers in the investor’s Amenities description. Investors should keep in mind that while words in the top 10% might have high yearly returns, the most valuable words are those that appear the most. As for those words that appear in the bottom 10%, the words with higher use are less valuable and should be avoided. While the Words provided are shown to have a positive or negative return on the investment, they should not be used as the sole discriminator when evaluating current or future properties.

```
###Visuals of top and bottom 10% of Words

##Top 20
top20_amen <- roi_amen %>%
  arrange(desc(ROI))
top20_amen <-  head(top20_amen, n=31)

##Bottom 20
bot20_amen <- roi_amen %>%
  filter(ROI>"0") %>%
  arrange(-desc(ROI))
bot20_amen <-  head(bot20_amen, n=31)

#Visuals
t20_am <- top20_amen %>%
  ggplot(mapping=aes(word,ROI,fill=Instances))+
  geom_col()+
  coord_flip()+
  ggtitle("Austin - Top 10% Words ROI")
t20_am
```

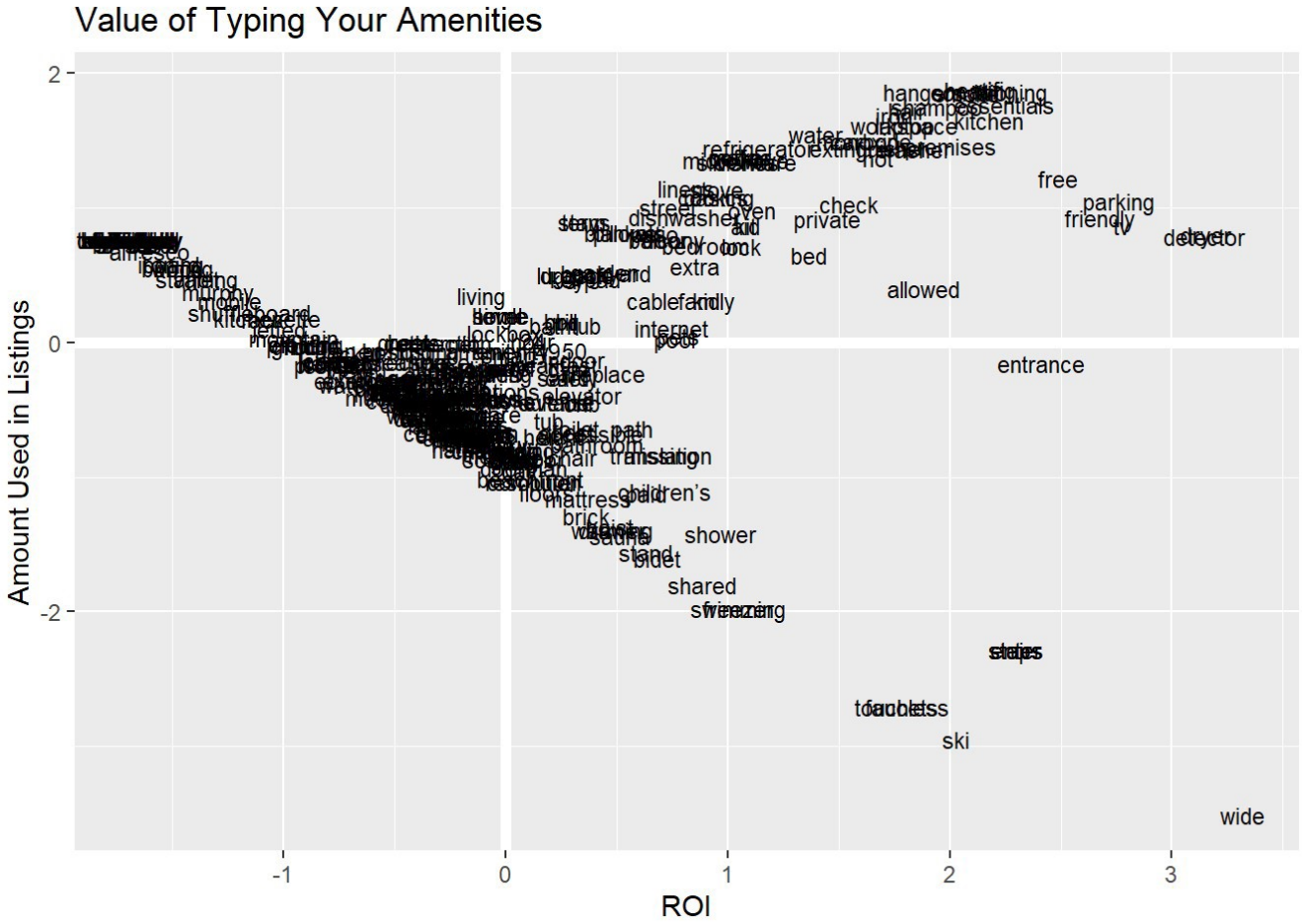



```
bot20_am <- bot20_amen %>%
  ggplot(mapping=aes(word,ROI,label=word,fill=Instances))+
  geom_col()+
  coord_flip()+
  ggtitle("Austin - Bottom 10% Words ROI")
bot20_am
```



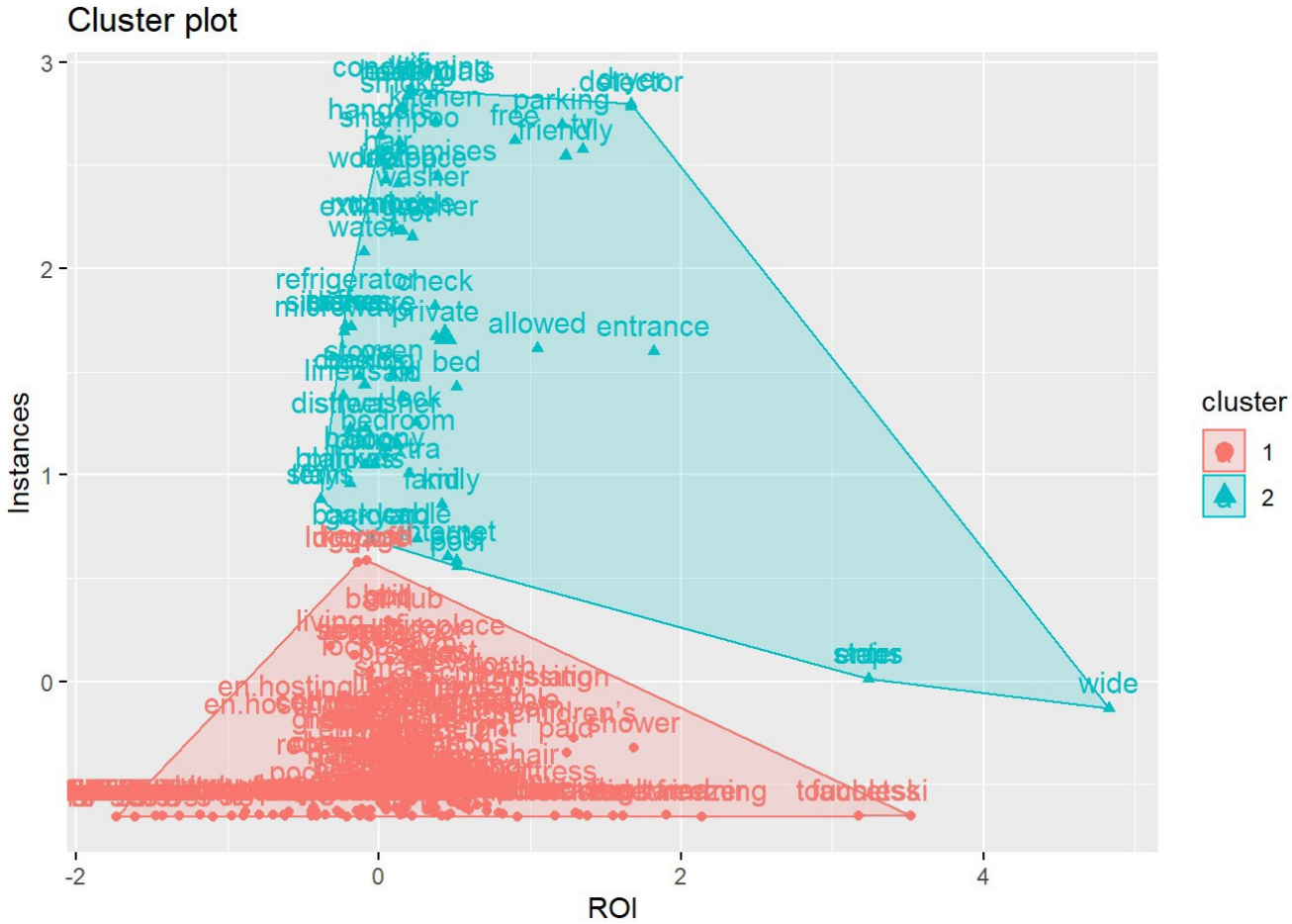
3.a.3. Principle Component Analysis (PCA) Principle component analysis (PCA) was used as a confirmatory tool as well as an exploratory tool. The expansion of the model to include eight closely related variables was an attempt to discern the viability of utilizing the variables ROI and Instances as a means of explaining the differentiation between the Words in the Amenities variable. As the Cumulative Scree plot shows, ROI and Instances are the broadest explanatory variables, but only slightly. The close relationship between both ROI and Instances with the other variables caused there to be no “elbow,” forcing the experimenter to make their own decision as to how many component variables to use in future models. In an attempt to follow the most easily understood variables and those that were determined to be the most explanatory, the experimenter chose the ROI and Instances variables. After reducing the data set to the ROI and Instances variables, a plot of the Words along those axis was used to provide an initial glimpse as to each Word’s affinity. The PCA plot of the Words was also used for comparative analysis against the plots within the K means clusters. While the extended axis of the PCA plot better outlines a Word’s affinity towards one of the four corners, most of the Words are seen to cluster along a negative line underneath the point of origin. This line does have a positive branch of words that have an increasing ROI and are used more frequently, which would be the best for potential investors. Outliers to this logic, such as “wide,” would also be viable investments due to their lack of use and positive living-space connotation.

```
##PCA -Plot of the Words
ggplot(PC, aes(PC1, PC2)) +
  modelr::geom_ref_line(h = 0) +
  modelr::geom_ref_line(v = 0) +
  geom_text(aes(label = word), size = 3) +
  xlab("ROI") +
  ylab("Amount Used in Listings") +
  ggtitle("Value of Typing Your Amenities")
```

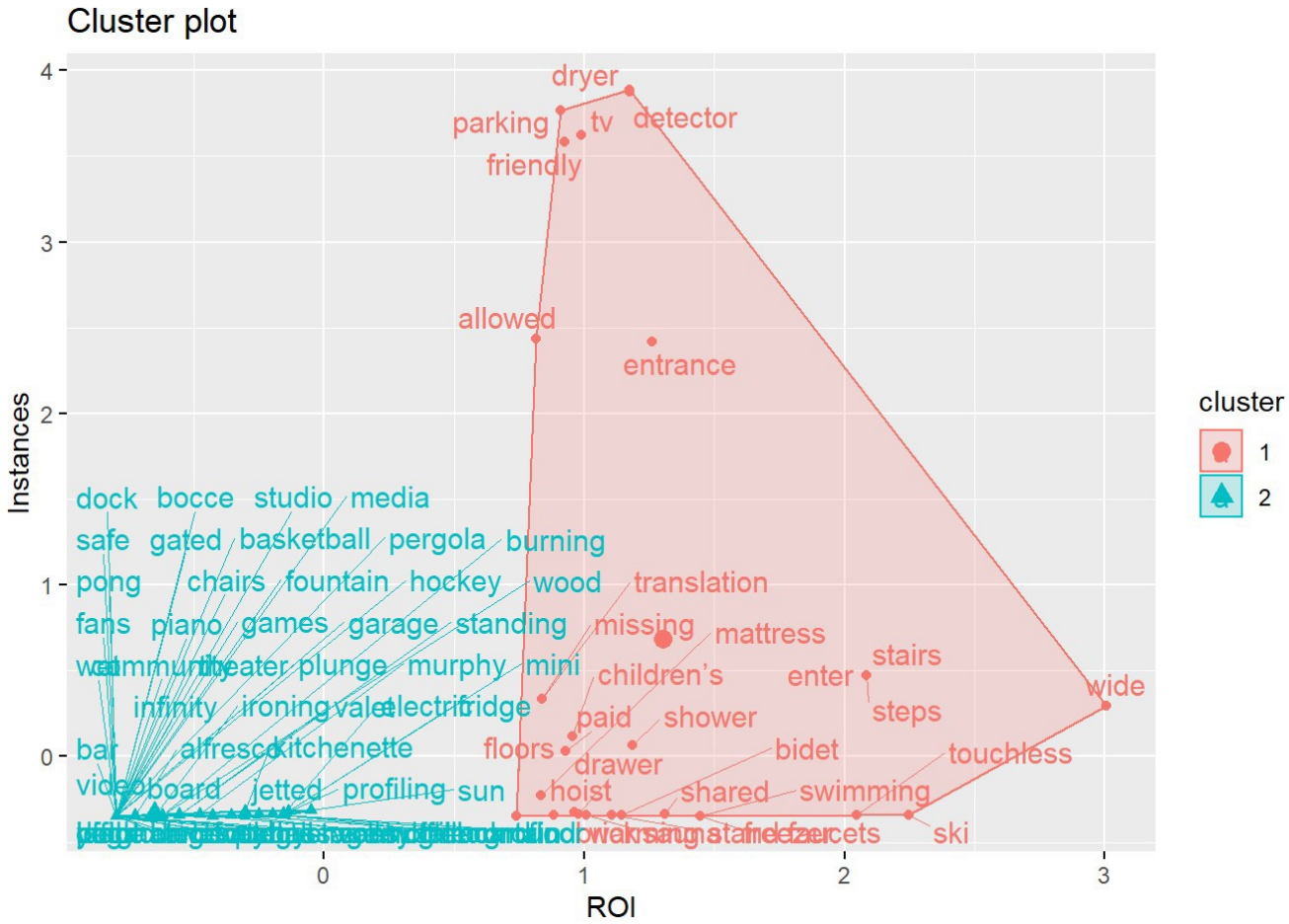


3.a.4. K Means Clustering K Means clustering was used to further expand the knowledge of each word's value and how they could be aggregated into distinct groups for the investor to engage. Provision of the broadest spectrum for the cluster segmentation occurred by using four slightly different data frames. After comparing the clusters, there was very little differentiation between any of the boundaries. As such, the data frame with the top 10% and bottom 20% was the primary candidate used for exploratory analysis. In accordance with our interpretation of the PCA, an initial k equal to two was used. Unfortunately, the resulting two clusters too broad, and no discernable differentiation between the Words was able to be extracted. Since the Scree Plot showed no distinct number for k, a k equal to three was used next. This number was decided upon not only from the Scree Plot but also as an attempt to engage with investors more easily. The resulting clusters provided discernable distinctions that were later characterized as such: Cluster 1 are the "Easy Fixes" (faucets, mattress, bidet, touchless), Cluster 2 are the "Atmosphere" of the listing (friendly, allowed, parking), and Cluster 3 is the "Entertainment" (basketball, mountain, yoga, media). Investors can discern what they specifically want in the terms of renovating their listing by following the cluster designator. These clusters provide an investor with not only the most valuable and least valuable amenities for their listing, but it also breaks them into segments so that the investor can easily assess their individual needs and act appropriately.

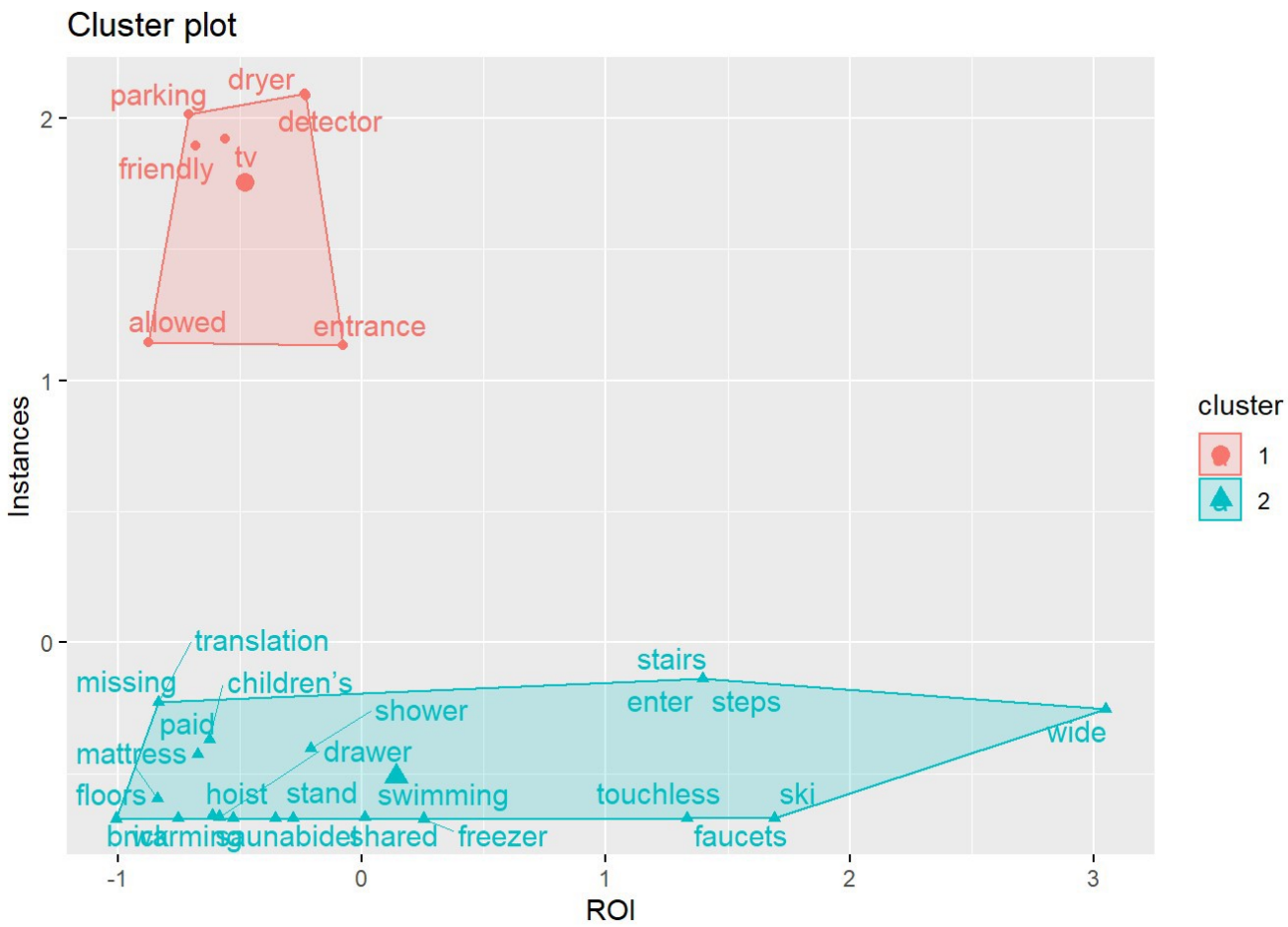
```
## Clustering Visualizations
## K=2
# First K Means clustering of 2 by 20 with Full Data Set
km_amen <- kmeans(pscale_amen,centers=2,nstart=20)
#km_amen                                        ## Results of the overall Data
fviz_cluster(km_amen, data=pscale_amen)       ## Visualization of the overall Data
```



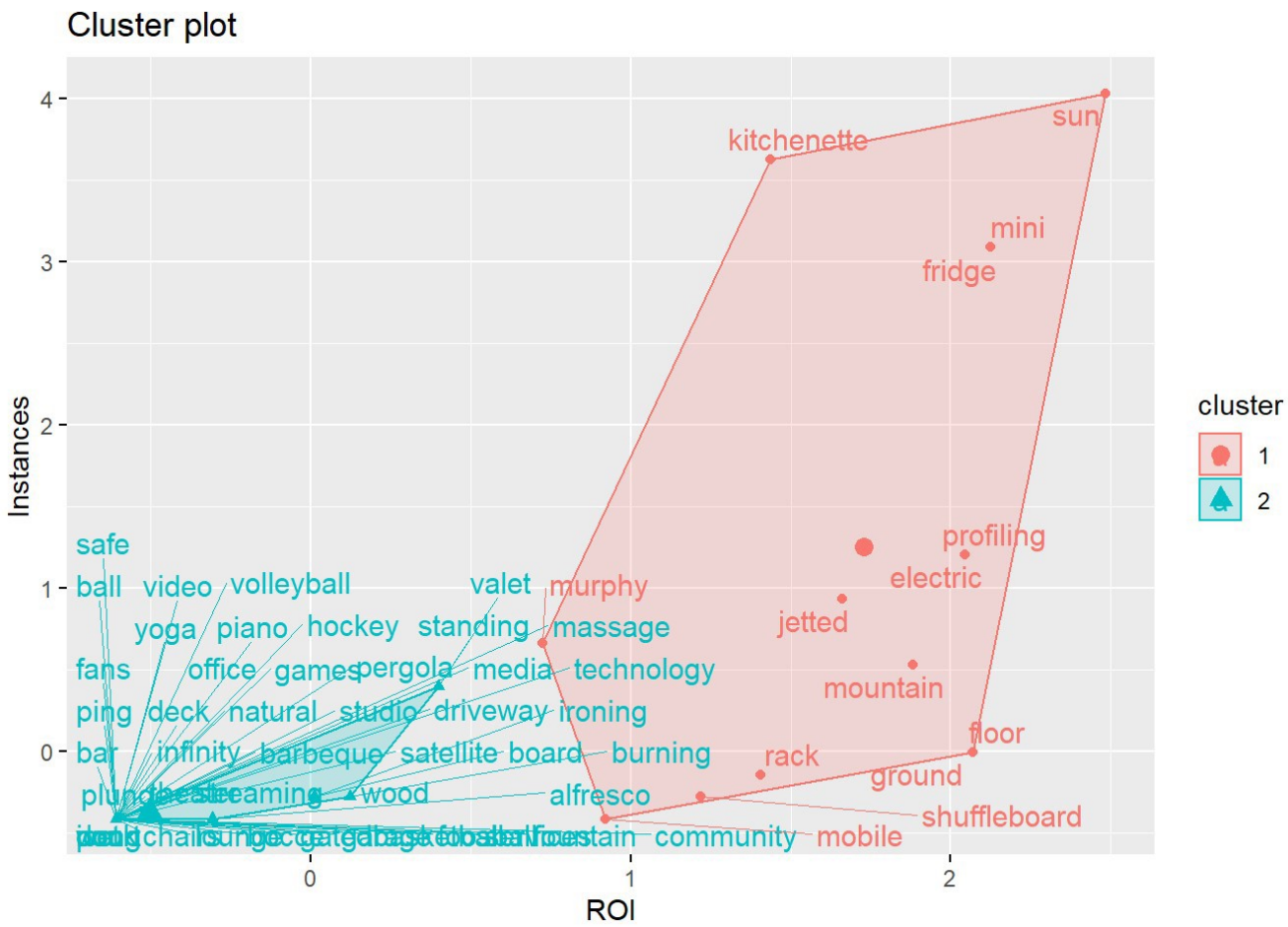

```
# First K Means clustering of 2 by 20 with Top10/Bot20 Pieces
kmc_amen <- kmeans(comb_scale_amen,centers=2,nstart=20)
#kmc_amen                                        ## Results of the overall Data
fviz_cluster(kmc_amen, data=comb_scale_amen,repel=TRUE)      ## Visualization of the T
op10/Bot20 Data
```



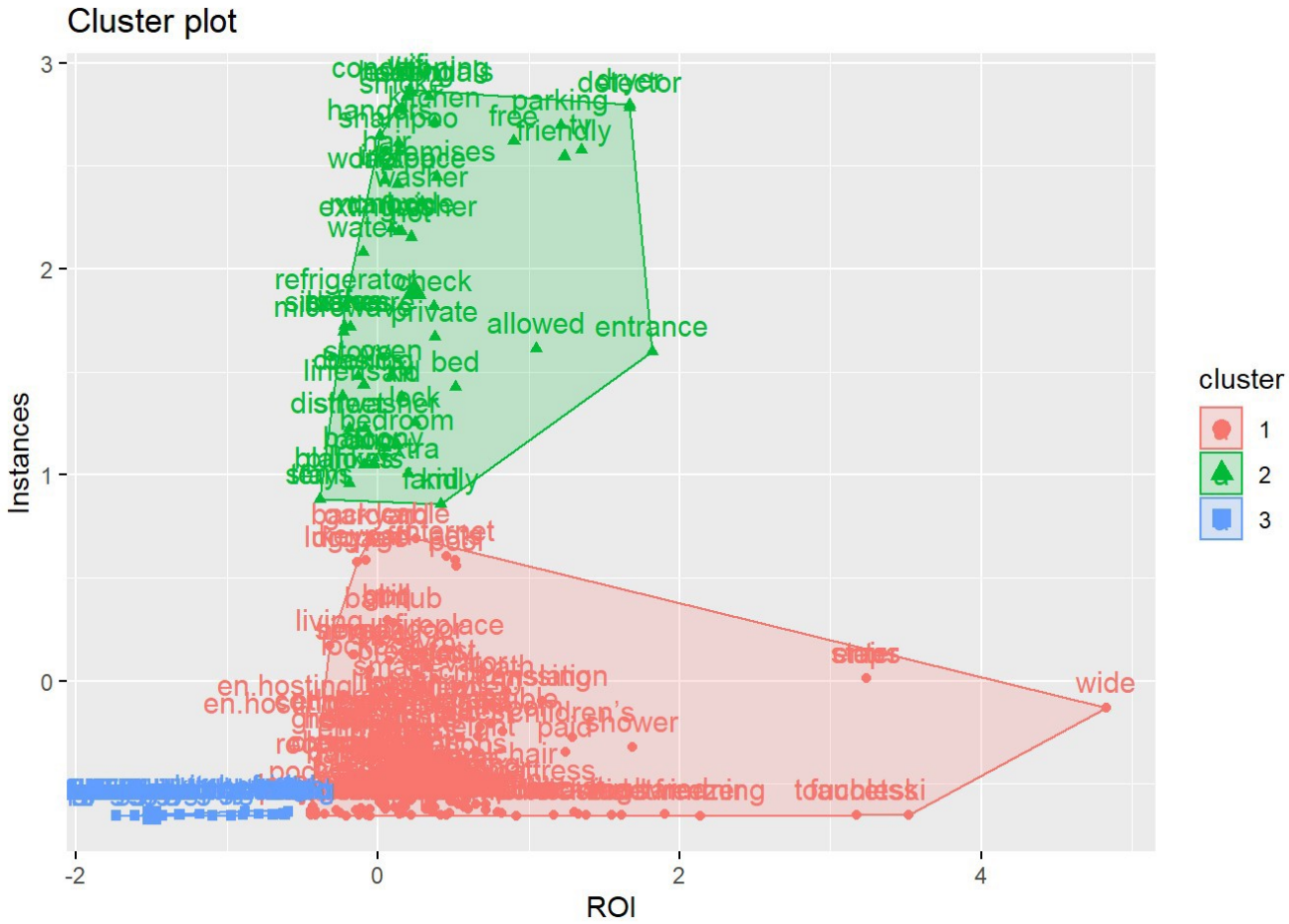

```
# Top 10% K Means clustering of 2 by 20
km_amen31 <- kmeans(pscale_amen31,centers=2,nstart=20)
#km_amen31                                    ## Results of the overall Data
fviz_cluster(km_amen31, data=pscale_amen31,repel=TRUE)      ## Visualization of the To
p10 Data
```



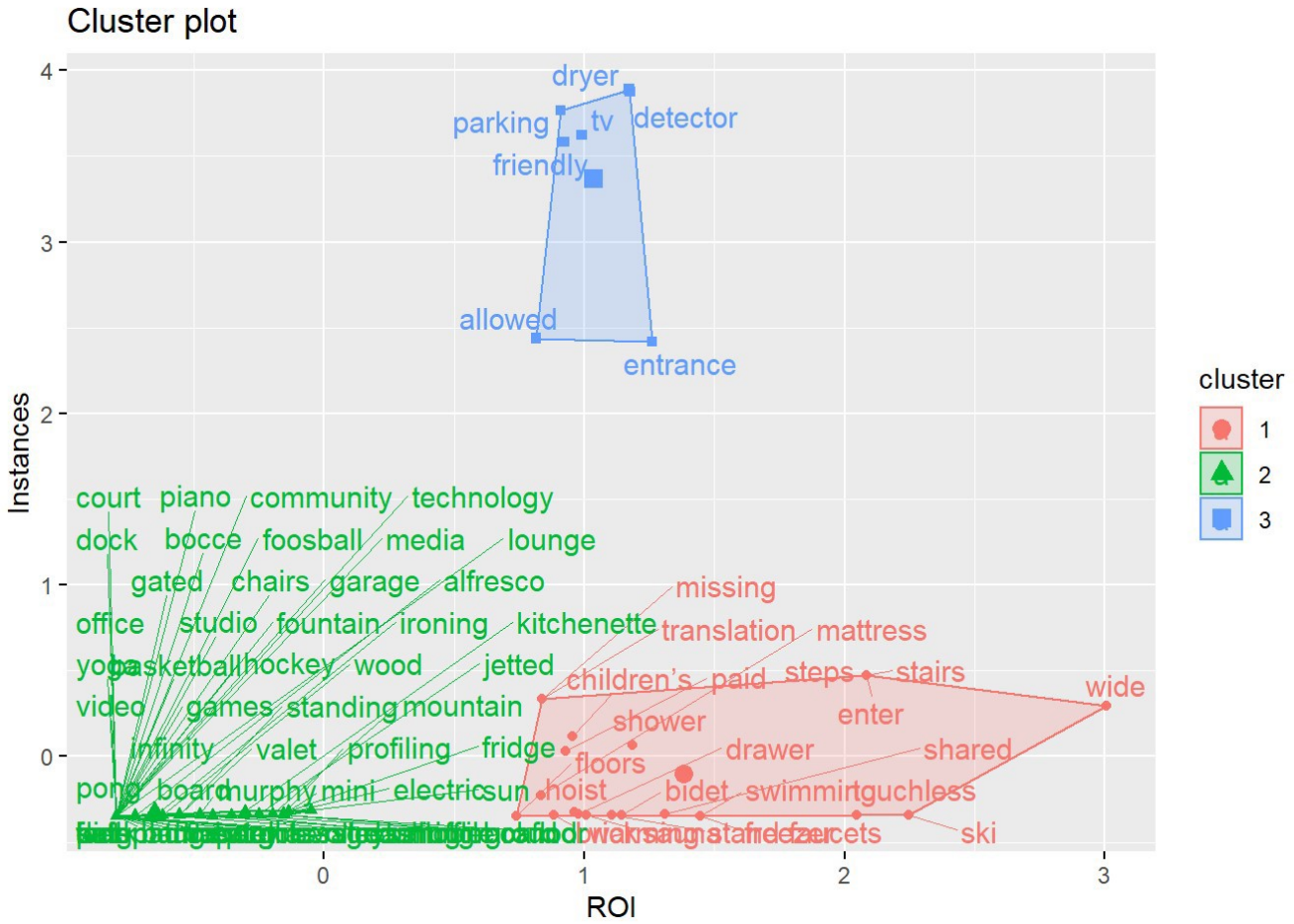

```
# Bottom 20% K Means clustering of 2 by 20
km_amen31b <- kmeans(pscale_amen31b,centers=2,nstart=20)
#km_amen31b                                    ## Results of the overall Data
fviz_cluster(km_amen31b, data=pscale_amen31b,repel=TRUE)      ## Visualization of the
Bot20 Data
```




```
## K=3
## K Means Clustering of 3 by 20

## First K Means with Full Data Set
km2_amen <- kmeans(pscale_amen,centers=3,nstart=20)
#km2_amen                                    ## Results of the overall Data
fviz_cluster(km2_amen, data=pscale_amen)      ## Visualization of the overall Data
```

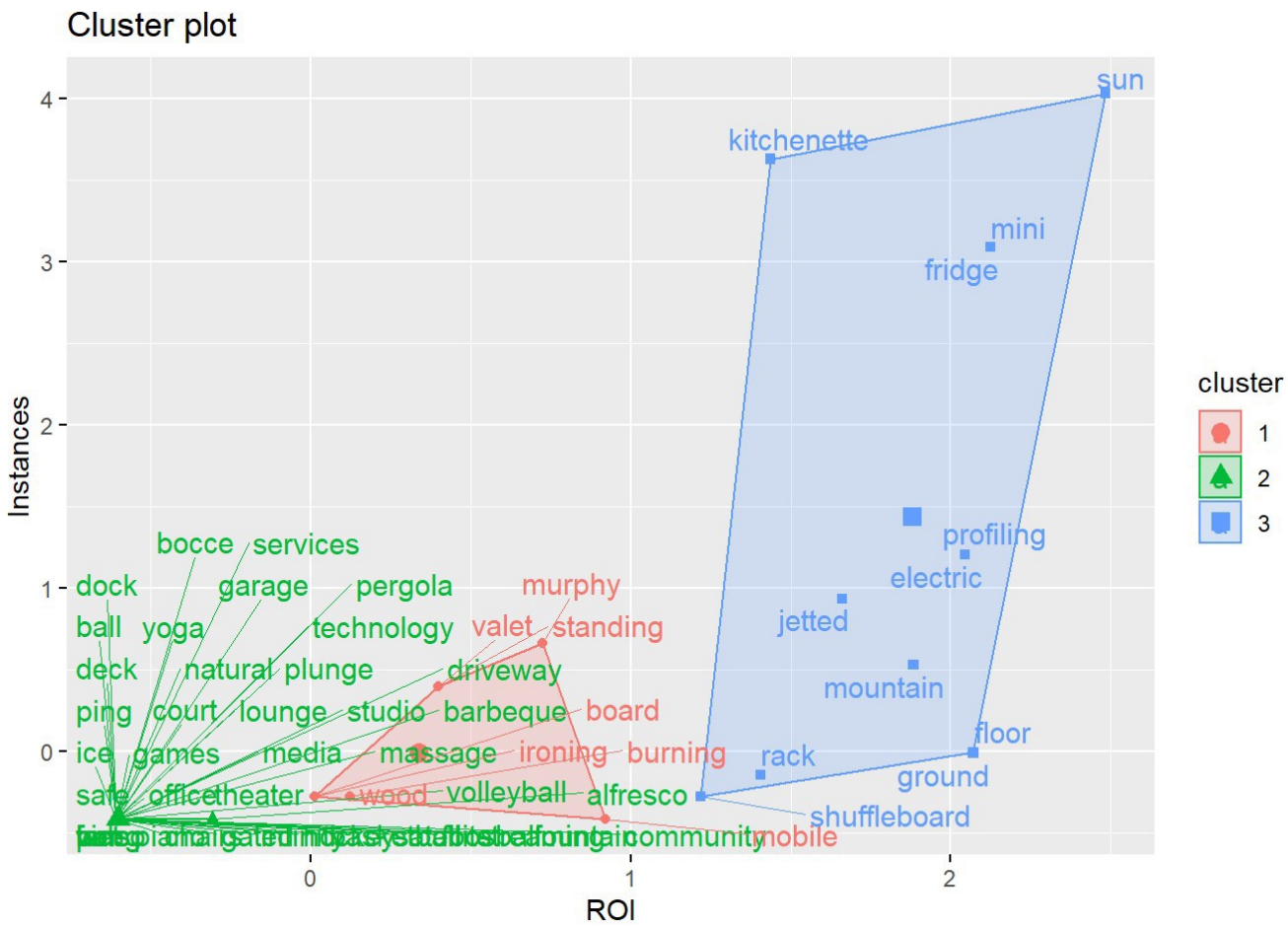



```
## First K Means clustering with Top10/Bot20 Pieces
kmc2c_amen <- kmeans(comb_scale_amen,centers=3,nstart=20)
#kmc2c_amen                                    ## Results of the overall Data
fviz_cluster(kmc2c_amen, data=comb_scale_amen,repel=TRUE)      ## Visualization of the
Top10/Bot20 Data
```




```
## Top 10% K Means
km2t_amen31 <- kmeans(pscale_amen31,centers=3,nstart=20)
#km2t_amen31                                    ## Results of the overall Data
fviz_cluster(km2t_amen31, data=pscale_amen31,repel=TRUE)      ## Visualization of the
Top10 Data
```


```
## Bottom 20% K Means
km2b_amen31b <- kmeans(pscale_amen31b,centers=3,nstart=20)
#km2b_amen31b                                    ## Results of the overall Data
fviz_cluster(km2b_amen31b, data=pscale_amen31b,repel=TRUE)     ## Visualization of the
Bot20 Data
```



3.a.5. Culture of Austin

Insertion of the different csv's that are specifically tailored to the living conditions, popular attractions, and top neighborhoods of Austin this helped develop a better understanding of what Austin holds. These attractions help emphasize the strength of Austin's culture and city background given they are known for music festivals such as South by Southwest (SXSW), great nature parks like Barton Springs and Zilker Park, and takes pride in the sports team from the University of Texas at Austin. While looking at the neighborhoods and living conditions, an emphasis should be placed upon the top neighborhoods with high booking rates. These neighborhoods should be deeply considered by any potential investors looking to place their listing where guests are looking to stay. As an investor, it is important to consider these details especially when most guests are visiting Austin for their first time. When a customer searches for an AirBNB they would want to see words such as perfect getaway, safe environment or close to the airport to ensure that their travel plans will flow accordingly. These factors are very important to consider, as a safe guest can explore and engage in different activities during their stay and is more likely to return. The wordclouds show a deep connection between the city history and the cultural background of Austin. Filters of the data to show the number of AirBNBs within Austin provide investors visibility of the most popular words that are shown to provide high booking rates within the city of Austin.

```
set.seed(1234)
#Most popular words when looking for an airbnb in Austin developed into a wordcloud fo
r better visualization
wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n, random.order=FALSE, rot.pe
r=0.35,colors=brewer.pal(8, "Dark2"))
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : microwave could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : stocked could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : travelers could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : provide could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : convenient could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : covered could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : email could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : west could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : university could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : happy could not be fit on page. It will not be plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : remodeled could not be fit on page. It will not be
## plotted.
```

```
## Warning in wordcloud(words = dfTidyAustin$word, freq = dfTidyAustin$n,
## random.order = FALSE, : quick could not be fit on page. It will not be plotted.
```

```
words <- c("Highland", "Cherrywood", "Windsor Park", "Mueller", "Travis Heights","Clar
ksville", "Bouldin Creek", "Old Enfield" )
freqs <- c(100, 70, 80,70,60,200,40,30)
set.seed(3)
#Wordcloud for the top communities in Austin where there are AirBnb Listings
wordcloud(words = words, freq = freqs, colors=brewer.pal(8, "Dark2"))
```


```
#Load csv with list of popular attractions
austinAttractions <- read_csv("attractions.csv")
```

```
## Parsed with column specification:
## cols(
##   `popular attractions` = col_character()
## )
```

```
words2 <- c("SXSW", "University of Texas (UT)", "downtown", "Barton Springs", "Hill Co
untry", "Downtown Airport", "restaurants","food trucks", "Boggy Creek", "Austin City L
imits Festival", "Town Lake","Zilker Park", "Hancock Golf Course", "Alamo Drafthouse",
"Saxton Pub", "Franklin Barbeque","Austin Convention Center", "Chi'Lantron", "Hope Out
door Gallery", "The Tavern", "Graffiti Park", "Perdernales Falls","Live Oak Breqing Co
mpany")
freqs2 <- c(800, 30, 70,60,40,100,40,20,200,10,20, 30, 200, 15, 15, 25, 5,5,15, 25, 3
5,20, 15)
set.seed(3)
#Wordcloud for the most popular attractions in Austin where there are AirBnb Listings
with high booking rates
wordcloud(words = words2, freq = freqs2, colors=brewer.pal(8, "Dark2"))
```


```
#Top words to describe living conditions of airbnbs with high booking rate  within Aus
tin.
livingConditions <- read_csv("conditions.csv")
```

```
## Parsed with column specification:
## cols(
##   `living conditions` = col_character()
## )
```

```
words3 <- c("perfect","comfortable","spatious", "love","peaceful", "serene","roomy" ,"
custom interior", "safe", "quiet", "luxury","cozy","remodeled","characterful","little
historic duplex","awesome spot","lovely","perfect getaway")
freqs3 <- c(800, 30, 70,60,40,100,40,20,200,10,20, 30, 200, 15, 15, 25, 5,5)
#Wordcloud for different communities in Austin where there are AirBnb Listings
wordcloud(words = words3, freq = freqs2, colors=brewer.pal(8, "Dark2"))
```

3.b. Investor Recommendations After analysis of the Austin market, its most notable aspect is that it is extremely investor friendly and it can provide significant returns for investors of all experience levels. The sporadic values of listings throughout all the neighborhoods makes it easier to make competitive price points regardless of the nearest listing values. While all neighborhoods are shown to have sporadic densities of high booking statuses, the highest overall probability seems to be aggregated within downtown Austin. Even the size of the property doesn't seem to matter, as a small "hotel" type listing still returned an average of $119 per night. Yet, investors can increase their profitability by improving the appeal of their listing through changes that are aligned within three clusters. "Easy fixes" can provide near immediate increases to the value's listing and can be as simple as installing new faucets in the bathrooms. The "atmosphere" cluster revolves around how available the property is to accommodate the guest, which can be as simple as being an accessible host to increasing the size of the property's parking area. Lastly, the "entertainment" grouping can be as narrow as having a home entertainment system to as broad as stating that there is a pool, which was a very common and profitable word to have in the Amenities section of the listing. With all of this in mind, it becomes clear that an investment in Texas' capital city will provide a significant return on investment for experienced and new hosts alike.

4. Conclusion and Discussion

4.a. Findings

When analyzing Airbnb data for Austin from an investor standpoint it is important to consider pricing strategy, the city/culture and background as well as location overall. From our analysis we can conclude that houses located in the downtown area are in extremely high demand while small portions of the houses are in the rural area. Additionally, there is a small peak of demand in the rural area, which is around Canyon Lake, Comal County, Texas. When we look at it closely, we can see that the top 3% of houses are unsurprisingly located near places of interest and shipping centers. Last thing we should pay attention to is that it might not be a wise decision to buy/rent a house near Austin International airport since the popular attractions and airbnbs are not close to the airport.

In regards to price, our findings discover that typical prices for the most of the airbnbs lies somewhere between $79 to $197, the average price is $119 although there is a tiny portion of houses worth between $368 and $975. At first glance, we can see that the houses located in the downtown area are the most expensive. However, when we take a closer look, there is not an apparent pattern of where the house tends to be more pricey in the downtown.

For the housing pricing strategy, some of our conclusions are : Investing a house in the downtown seems to be a safe decision, especially those near place of interest and shopping center, Creating a luxury house is not a great idea due to the current fair market value.Austin market seems to be a good option for a green hand investor since you do not need to dig into too much in terms of the accurate location once it is in the downtown. It is fine to ask for a reasonable cleaning fee and extra people fee and a room that can accommodate two persons will always be popular.

Therefore in terms of the project relevancy,we have found that in regards to purchasing - Airbnb residents tend to be fine with paying a cleaning fee of around $60 after the whole stay and approximately $10 for the extra per person. Most of the houses with a high booking rate tend to be allowed to live with 2 people.

Lastly, in terms of amenities , it is important to note which amenities listed within the description have a higher impact on the high booking rate. We have found from our analysis that having a pool and a tv at Airbnb and mentioning that in the description can also help achieve a higher booking rate. Subsequently, hosts tend to describe the facilities in the house and the surrounding facilities of these houses in the descriptions of their house. It is common to note that hosts always mention bedrooms, bathrooms and kitchen in description generally. Customers tend to pay more attention to the housing capacity, position and surrounding facility of the house. There seems no significant difference between most occurring nouns of two types of house. Yet, from gathering that most houses which mention pool or tv and other entertainment facilities might have more success in achieving a higher booking rate.

Overall, Austin is a popular city for tourist attractions, business trips, and pleasure. From popular attractions such as Barton Springs, Zilker Park, and University of Texas the city has a lot to offer not only to its inhabitants but its plethora of visitors as well. From our findings , we have discovered that most AirBnbs are purchased in the downtown area, even though it costs more to book due to closeness in popular attractions. Lastly, our findings conclude that a high booking rate is most affected by the number and type of amenities in addition to verified hosts within the designated area.

4.b. Limitations

Limitations within the research cover issues with imputation, regionality, text mining and extrapolation. Imputations issues such as the way missing values were handled could negatively change the reality of the data. This occurred from the efforts to set every numerical missing value to the mean value of its variable and set every factor missing value to the mode of its variable. This method of imputation can improve the

performance of our analysis but also has the possibility of having a very limited effect. Dealing with missing values in this singular way can make some changes in the original data. For example, if the missing price of a house is much higher than other values and it is set to the mean value, this would be a dramatic change in reality. As a result of that, we should find some better ways to handle the missing value instead of simply setting them to mean value. Additional limitations came in the form of the text analysis. In some attempts with the tidytext platform, some of the variables returned incomplete characters, random punctuations, or foreign characters. This occurs from the way R recognizes text, which is very different from human beings, as it is encoded to automatically recognize Arabic numerals as nouns. Text analysis was not incorporated into the prediction model for the competition, which could have greatly improvement in the accuracy. The valuable information contained in the textual data was constrained by the group's domain knowledge, resulting in individual information silos that were note compiled until the end. Finally, the general analysis of the AirBNB data is singularly focused and is not an exact representation of vacation rental usage within the city of Austin. Extrapolation of the law of things recognizes that the assumption of isolated locations with less bookings on AirBNB might not be reality, as there might be evidence of high booking rates on other sites such as VRBO. Fallacies of over analysis can cause misinterpretations of the data and cause researchers to ignore the regionality that would help their predictions.

4.c. Future Research

The Austin data has one peculiar property of much emphasis on the host verified status and host category (whether a host is suerhost). This is contrasting to the overall trend but isnt contradictary to it. The Austin model can be extended to similar areas of the US but should be looked for the correlated varuables that are left in this model as they might act differently in other areas with respect to varying weather conditions, geographhic factors and political attributes as well. The neighborhood will also be a feature that should be excluded for running this model on other areas.

References

Robinson, J. S. and D. (n.d.). Text Mining with R. Retrieved May 6, 2020, from https://www.tidytextmining.com/ (https://www.tidytextmining.com/) Text Mining: Word Relationships · UC Business Analytics R Programming Guide. (n.d.). Retrieved May 6, 2020, from http://uc-r.github.io/word_relationships (http://uc-r.github.io/word_relationships) Textclean.pdf. (n.d.). Retrieved May 6, 2020, from https://cran.r-project.org/web/packages/textclean/textclean.pdf (https://cran.r-project.org/web/packages/textclean/textclean.pdf)