Data Mining & Predictive Anlaysis Project Report

Exploring factors affect high booking rate in Airbnb in Miami market

Team 20

Market Assigned to team: **Miami**

"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 apporved the report as submitted; the text, images, illustrations, and other items included in the manuscript do not carry any infringe/plagiarism issue upon any existing copyrighted materials"

Member	**Names of the signed team members**
Contact Member	Weibo Chen
Team member 2	Jiading Chen
Team member 3	Wei-Cheng Huang
Team member 4	Changnan Jing
Team member 5	Wenzhe Wu

Executive Summary

As one of the most popular vacation cities, Miami attracts more than 20 millions visitors from all-over the world and therefore has a huge market for Airbnb. Understanding what visitors care about and looking for becomes significant for Airbnb investors. Supported by multiple research papers on hotel Industries, we infer that location, amenities and transportations can be contributing factors for Airbnb properties being frequently booked in Miami.

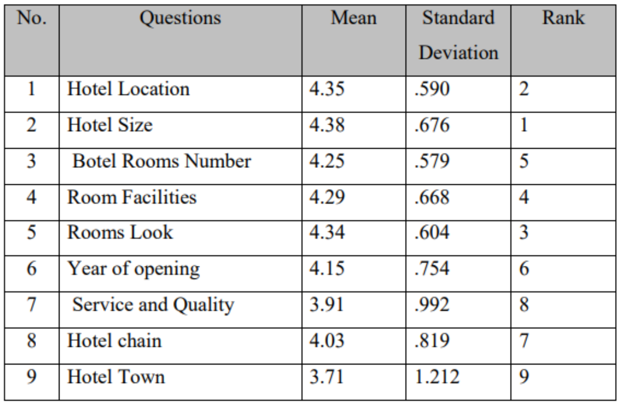
We did a deep analysis on the Airbnb Miami market date using spatial mapping, classification, text mining, and modeling techniques. After deeply and carefully analyzing these factors, we concluded that three factors we mention have substantial impacts on properties being frequently booked and can potentially create huge profit for investors if they follow our recommendations.

Research Question

Before we step into analysis and build explanatory models, it is necessary to research and make sure we are on the right direction. We did a literature review and formed up our research questions for this Airbnb data.

1. Is location a key factor of success for Airbnb in a vacation city?

   According to “A Study of Large Hotel Occupancy Rates on the Island of St. Lucia”,

No.	Questions	Mean	Standard Deviation	Rank
1	Hotel Location	4.35	.590	2
2	Hotel Size	4.38	.676	1
3	Botel Rooms Number	4.25	.579	5
4	Room Facilities	4.29	.668	4
5	Rooms Look	4.34	.604	3
6	Year of opening	4.15	.754	6
7	Service and Quality	3.91	.992	8
8	Hotel chain	4.03	.819	7
9	Hotel Town	3.71	1.212	9

   the location of hotels relative to tour sites and attractions, such as beaches and historic areas, are critical for success. Researchers investigated the importance of location or proximity of hotels, motels, guesthouses, and similar establishments to specific sites and believed that location is the top factor for tourists to make a decision. We are curious if this is also true for Airbnb business.

2. Which neighbourhoods are mostly worthy for investment in Miami?

   Inspired by article *“The 9 Best Neighborhoods to Live in South Miami”* by The Storage Queens, we would like to find out if there are some neighbourhoods outstandingly popular and/or possessing higher booking rates when comparing with other neighbourhoods. If they exist, we would like to know what and where are they.

3. What factors are affecting the booking rate of Airbnb?

   According to *“Factors Affecting Hotel Occupancy Rate”*, researchers found that location, amenities and transportation are in the top ten factors of affecting the performance of a hotel. Again, we want to prove that this also works for Airbnb.

Methodology

The Methodology contains three parts.

1. Geographical Clustering
2. Amenity information extraction
3. Transportation information extraction

The works will be explained in each division.

```
library("tidyverse")
library("ggplot2")
library("plotly")
library("skimr")
library('rgdal')
library('dbscan')
library('caret')
library("skimr")
library("tidytext")
library('stringr')
```

```
dfTrain <- read_csv('F:/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.
```

```
dfTest <- read_csv('F:/airbnbTest.csv')
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   id = col_double(),
##   accommodates = col_double(),
##   availability_30 = col_double(),
##   availability_365 = col_double(),
##   availability_60 = col_double(),
##   availability_90 = col_double(),
##   bathrooms = col_double(),
##   bedrooms = col_double(),
##   beds = col_double(),
##   guests_included = col_double(),
##   host_has_profile_pic = col_logical(),
##   host_identity_verified = col_logical(),
##   host_is_superhost = col_logical(),
##   host_listings_count = col_double(),
##   host_since = col_date(format = ""),
##   instant_bookable = col_logical(),
##   is_business_travel_ready = col_logical(),
##   is_location_exact = col_logical(),
##   latitude = col_double(),
##   longitude = col_double()
##   # ... with 14 more columns
## )
## See spec(...) for full column specifications.
```

```
### Filter specific market
dfTrain_Miami <- dfTrain %>% filter(substr(`{randomControl}`,1,3) == 113)
```

Geographical Clustering

The location of property, as the research shown, is at the top of the most influencing feature of users preferance. In this part, we will try to find most valuable locations for airbnb hosts and investors.

Let's see the Miami neighbourhood information first.

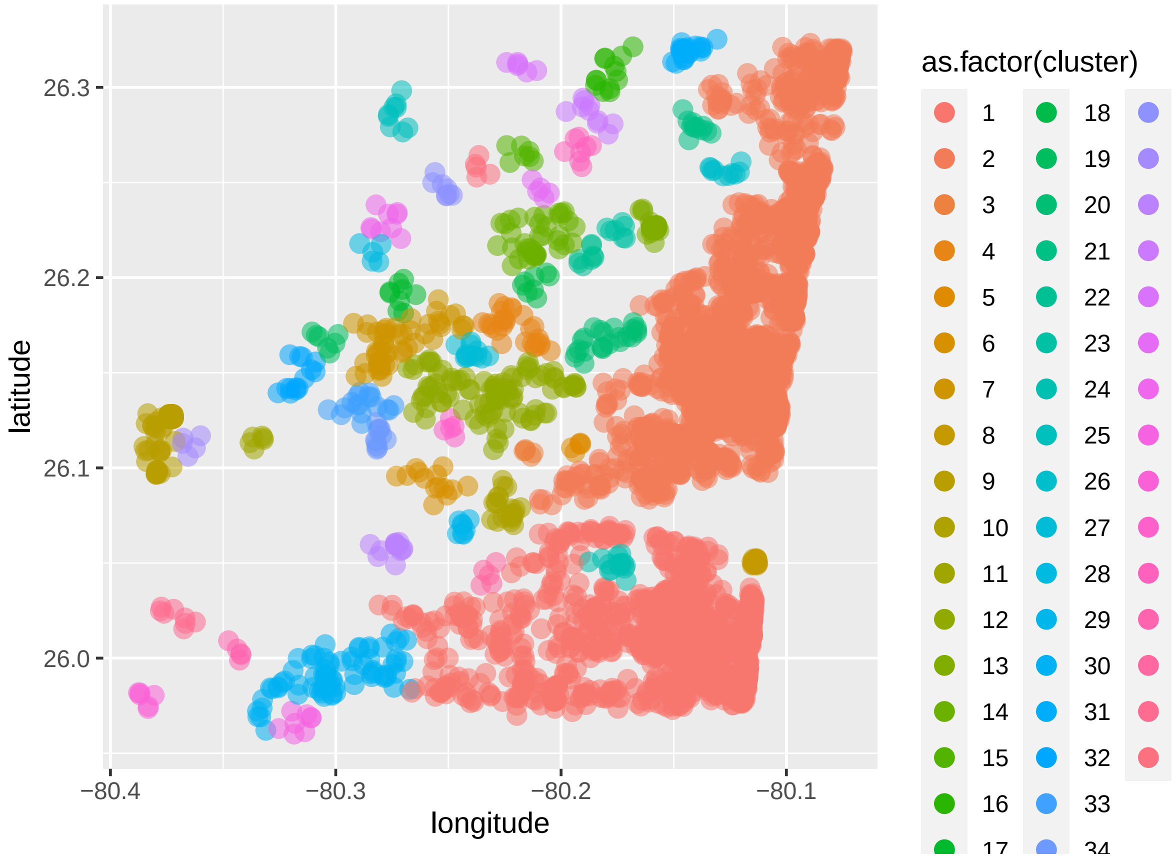
```
dfTrain_Miami %>%
  select(neighbourhood) %>%
  mutate(valid=ifelse(is.na(neighbourhood),0,1)) %>%
  group_by(valid) %>%
  tally() %>%
  ungroup() %>%
  mutate(pct=n/sum(n))
```

```
## # A tibble: 2 x 3
##   valid     n    pct
##   <dbl> <int>  <dbl>
## 1     0  6025 0.973
## 2     1   170 0.0274
```

Most Miami hosts didn't input their neighbourhood information, we need a detour.
The geographic data at hand is now only longitude and latitude,based on Longitude and Latitude, we can select KMeans, HC and DBSCAN as our canditating algorithms. They all perform good in clustering, but we finally chose DBSCAN to be our selection.

Comparing the other two algorithms, the selcetion of k value of KMeans can be too subjective to fit our model properly. What's more, KMeans and HC cannot handle the scenarios of clusters nested with others. However, in our model, the occurance of 'urban villages' are reasonable speculation. So, DBSCAN is our final choice.

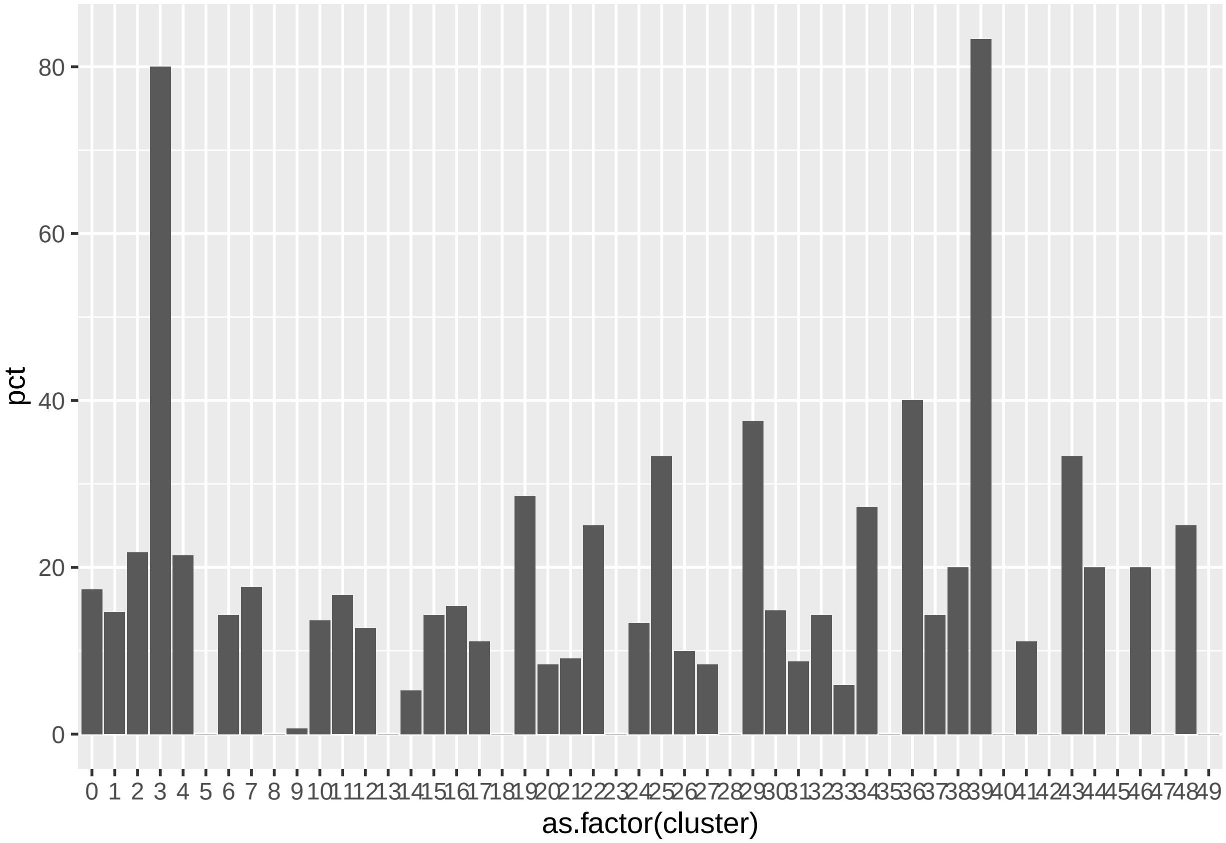
```r
### Visualize the clusters
dfgeo_Miami <- dfTrain_Miami %>%
  select(id,high_booking_rate,latitude,longitude)
clusters <- dbscan(select(dfgeo_Miami,longitude,latitude), eps = 0.0079)
dfgeo_Miami$cluster <- clusters$cluster

groups <- dfgeo_Miami %>% filter(cluster != 0)
noise <- dfgeo_Miami %>% filter(cluster == 0)

clusterplot <- ggplot(dfgeo_Miami, aes(x = longitude, y = latitude, alpha = 0.5)) +
  geom_point(aes(colour = as.factor(cluster)), groups,size = 3)
#ggplotly(clusterplot)
clusterplot
```



Eps value was tuned based on several iterations. With the increase of eps, the number of cluster decreases. As the host of houses, they always would like to know where is the best location specifically. However, the more cluster doesn't mean a better clustering.

So, how many clusters are reasonable for this problem? According to Wikipedia, Miami-Dade County has nineteen cities, six towns, and nine villages. If the division is too detailed, it means that there may be 1 or 2 observations in one cluster, which is not convincing. If the division is too broad, variance among one division may also be high. Considering the division of clusters doesn't have to follow Administrative division, so, the number of clusters can range from 30 to 50, and the least number of observation in one division should not be less than 5.

```
### Visualize the percent of high_booking_rate within each cluster
dfgeo_Miami %>%
  group_by(cluster) %>%
  mutate(pct = 100*sum(high_booking_rate)/length(high_booking_rate)) %>%
  group_by(cluster,pct) %>%
  tally() %>%
  ggplot(aes(x = as.factor(cluster), y = pct)) +
  geom_bar(stat="identity")
```



Now we have the range of the number of clusters, while determining the best division, the percentage is also an important attribute. We can focus on the percentage of houses having a high booking rate out of all houses in that geographical community. On the contrary, the locations that have a lower percentage will be the least considered. As a result, the quality of clustering can be reflected as the ratio of 'extreme' clusterings. If the percentage is mediocre, which is around 50%, we cannot recommend to new owners that this is a good place that can bring you a great fortune or not. Here, we calculate the result that the number of so called 'valid rows' divided by all number of rows to be our clustering quality attribute. we consider that it is a good clustering result when the number of clusters whose high booking rate percentage are more than 70% or less than 30% occupies the majority of all clusters.

```
### Verryfing if the eps parameter satisfies our requirements
### (with iterations but only shows the best parameter(eps=0.0079) outputs)
dfgeo_Miami %>%
  group_by(cluster) %>%
  mutate(pct = 100*sum(high_booking_rate)/length(high_booking_rate)) %>%
  group_by(cluster,pct) %>%
  tally() %>%
  arrange(desc(pct))
```

```
## # A tibble: 50 x 3
## # Groups:   cluster [50]
##    cluster   pct     n
##      <int> <dbl> <int>
##  1      39  83.3     6
##  2       3  80       5
##  3      36  40       5
##  4      29  37.5     8
##  5      25  33.3     9
##  6      43  33.3     6
##  7      19  28.6     7
##  8      34  27.3    11
##  9      22  25       8
## 10      48  25       8
## # ... with 40 more rows
```

```
dfgeo_Miami %>%
  group_by(cluster) %>%
  mutate(pct = 100*sum(high_booking_rate)/length(high_booking_rate)) %>%
  group_by(cluster,pct) %>%
  tally() %>%
  ungroup() %>%
  summarise(pct_median=median(pct))
```

```
## # A tibble: 1 x 1
##   pct_median
##        <dbl>
## 1       14.0
```

```
rowcount_valid <-
  dfgeo_Miami %>%
  group_by(cluster) %>%
  mutate(pct = 100*sum(high_booking_rate)/length(high_booking_rate)) %>%
  group_by(cluster,pct) %>%
  tally() %>%
  filter(pct > 70 | pct < 30) %>%
  nrow()

rowcount <-
  dfgeo_Miami %>%
  group_by(cluster) %>%
  mutate(pct = 100*sum(high_booking_rate)/length(high_booking_rate)) %>%
  group_by(cluster,pct) %>%
  tally() %>%
  nrow()

rowcount_valid/rowcount
```

```
## [1] 0.92
```

Finally, we found eps equals to 0.0079 could serve our purpose well. We will use this value as our clustering parameter.

Amenity Exploration

Next, We would like to find out which kind of amenities that will catch airbnb users' eye.

Amenities that provide from airbnb hosts could definitly make a difference on users making their choices.

The way we doing this is to first split text data in column "amenities" into information carrying atomic pieces. Then, by setting them as dummy variables, we can build a model to check which information has significant effect on explananing high booking rate.

```
### Slice the whole dataset into a smaller one
dfamenity_Miami <- dfTrain_Miami %>%
  select(id, high_booking_rate, amenities)

### Text data cleaning
dfamenity_Miami$amenities <- gsub('[0-9]+', '', dfamenity_Miami$amenities)
dfamenity_Miami$amenities <- tolower(dfamenity_Miami$amenities)
dfamenity_Miami$amenities <- gsub("[^[:alnum:][:space:],]",'',dfamenity_Miami$amenities)
dfamenity_Miami$amenities <- gsub(' ','_',dfamenity_Miami$amenities)

### Split text string into atomic word
dfamenity_Miami_disjoint <-
  dfamenity_Miami %>%
  unnest_tokens(word, amenities) %>%
  anti_join(stop_words)
```

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

```
### Store low frequence words as a vector
lowfreqword <- dfamenity_Miami_disjoint %>%
  filter(!is.na(word)) %>%
  group_by(word) %>%
  summarise(count=n()) %>%
  filter(count<=40) %>%
  select(word) %>%
  unlist() %>%
  unname()

### Split amenities into a list
t <- strsplit(dfamenity_Miami$amenities, split = ",")
```

```
### Eliminate duplicate words and store them as a vector
tags <- unique(str_trim(unlist(t)))
tags <- c(tags)

### Form up an id vs. dummy variables dataframe and assign dummy names
df2 <- as.data.frame(
  Reduce(
    cbind, lapply(
      tags, function(i) sapply(
        t, function(j) +
          (any(grepl(i, j, fixed=TRUE),
               na.rm = TRUE))))))
names(df2) <- tags

### Excluding low frequence words
df2 <- df2[ , !(names(df2) %in% lowfreqword)]

### Assign id to each row
df2 <- df2 %>%
  mutate(id = dfamenity_Miami$id)

### Merge dummy dataset back to original
dfamenity_Miami_Dummy <-
  merge(x = dfamenity_Miami, y = df2, by = "id") %>%
  select(-amenities,-id) %>%
  mutate(high_booking_rate = as_factor(high_booking_rate))
```

We would like know which amenity contribute the most, therefore a lasso regression could do the job and we hope to check the varimp plot to draw our conclusions.

```
lambdav <- 10^seq(-5,2,length=100)
set.seed(123)
fitlasso <- train(formula = high_booking_rate ~., family = "binomial",
                  data = dfamenity_Miami_Dummy,method='glmnet',
                  trControl=trainControl(method='cv',number=10),
                  tuneGrid=expand.grid(alpha=1,lambda=lambdav))
```

```
## Error in names(res$trainingData) %in% as.character(form[[2]]): argument "form" is missing, with no de
```

```
varImp(fitlasso)$importance %>%
  rownames_to_column(var="Variables") %>%
  mutate(Importance=scales::percent(Overall/100)) %>%
  arrange(desc(Overall)) %>%
  as_tibble()
```

```
## Error in varImp(fitlasso): object 'fitlasso' not found
```

It seems lasso regression doesn't work, perhaps it is because the matrix being too sparse. Anyway, another detour to go.

```r
### Using logistic regression to check variables significant
fitLRM<-
  glm(formula = high_booking_rate ~., family = "binomial", data = dfamenity_Miami_Dummy)
summary(fitLRM)
```

```
## 
## Call:
## glm(formula = high_booking_rate ~ ., family = "binomial", data = dfamenity_Miami_Dummy)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -2.3550  -0.5444  -0.2981  -0.1386   3.3471  
## 
## Coefficients:
##                                       Estimate Std. Error z value Pr(>|z|)    
## (Intercept)                          -4.484643   0.701904  -6.389 1.67e-10 ***
## tv                                    0.103262   0.230026   0.449 0.653493    
## cable_tv                              0.113598   0.096208   1.181 0.237700    
## wifi                                  0.875369   0.404035   2.167 0.030268 *  
## air_conditioning                      0.797871   0.633883   1.259 0.208137    
## pool                                 -0.289487   0.095329  -3.037 0.002392 ** 
## kitchen                              -0.487466   0.153324  -3.179 0.001476 ** 
## gym                                  -0.577365   0.170518  -3.386 0.000709 ***
## elevator                             -0.550333   0.182089  -3.022 0.002508 ** 
## heating                              -0.092543   0.096985  -0.954 0.339982    
## washer                               -0.256041   0.127940  -2.001 0.045364 *  
## dryer                                -0.614229   0.251623  -2.441 0.014644 *  
## smoke_detector                       -0.157480   0.171482  -0.918 0.358437    
## essentials                            0.284413   0.301023   0.945 0.344751    
## hangers                              -0.226401   0.190845  -1.186 0.235501    
## iron                                 -0.073396   0.157207  -0.467 0.640587    
## self_checkin                          1.747985   1.190021   1.469 0.141868    
## building_staff                       -1.139230   1.192719  -0.955 0.339500    
## private_living_room                  -0.077889   0.113604  -0.686 0.492956    
## bed_linens                           -0.209177   0.134422  -1.556 0.119680    
## refrigerator                          0.224282   0.231464   0.969 0.332558    
## dishes_and_silverware                 0.241045   0.171225   1.408 0.159199    
## patio_or_balcony                     -0.094016   0.108697  -0.865 0.387076    
## long_term_stays_allowed              -0.112614   0.101148  -1.113 0.265554    
## cleaning_before_checkout             -0.363464   0.198146  -1.834 0.066606 .  
## waterfront                           -0.274637   0.147140  -1.867 0.061970 .  
## paid_parking_on_premises             -0.102179   0.214788  -0.476 0.634273    
## free_parking_on_premises             -0.299029   0.162427  -1.841 0.065622 .  
## hot_tub                              -0.140233   0.140710  -0.997 0.318953    
## familykid_friendly                    1.145346   0.097362  11.764  < 2e-16 ***
## carbon_monoxide_detector              0.332826   0.098036   3.395 0.000686 ***
## fire_extinguisher                    -0.096957   0.103286  -0.939 0.347872    
## shampoo                               0.546777   0.121345   4.506 6.61e-06 ***
## laptop_friendly_workspace             0.245678   0.109679   2.240 0.025093 *  
## bathtub                               0.065894   0.120612   0.546 0.584838    
## microwave                             0.389339   0.222878   1.747 0.080660 .  
## coffee_maker                          0.293089   0.173510   1.689 0.091186 .  
## dishwasher                           -0.209279   0.121749  -1.719 0.085625 .  
## cooking_basics                        0.068992   0.162586   0.424 0.671318    
## oven                                 -0.130067   0.161978  -0.803 0.421980    
## stove                                 0.123946   0.177777   0.697 0.485677    
## pets_allowed                         -0.315138   0.105937  -2.975 0.002932 ** 
## first_aid_kit                        -0.121288   0.095951  -1.264 0.206206    
## hot_water                             0.172375   0.124767   1.382 0.167104    
## host_greets_you                       0.619397   0.163350   3.792 0.000150 ***
## hair_dryer                            0.490747   0.190037   2.582 0.009812 ** 
## lock_on_bedroom_door                  0.078122   0.089921   0.869 0.384964    
## internet                             -0.096984   0.125570  -0.772 0.439910    
## smoking_allowed                      -0.311309   0.212348  -1.466 0.142639    
## safety_card                           0.371508   0.119802   3.101 0.001929 ** 
## hour_checkin                          0.574708   0.147847   3.887 0.000101 ***
## private_entrance                     -0.152690   0.091648  -1.666 0.095704 .  
## pack_n_playtravel_crib                0.112810   0.229711   0.491 0.623357    
## single_level_home                     0.230690   0.107567   2.145 0.031984 *  
## bbq_grill                            -0.423421   0.116403  -3.638 0.000275 ***
## garden_or_backyard                   -0.016370   0.113343  -0.144 0.885163    
## wide_entrance_for_guests             -0.214922   0.364729  -0.589 0.555683    
## flat_path_to_guest_entrance           0.340560   0.190060   1.792 0.073156 .  
## welllit_path_to_entrance             -0.051380   0.156177  -0.329 0.742165    
## no_stairs_or_steps_to_enter          -0.084302   0.154022  -0.547 0.584149    
## wide_entrance                         0.089891   0.363191   0.248 0.804518    
## accessibleheight_toilet               0.293720   0.440425   0.667 0.504836    
## wide_entryway                        -0.192279   0.234425  -0.820 0.412094    
## wheelchair_accessible                 0.296283   0.206721   1.433 0.151785    
## beach_essentials                      0.283094   0.114843   2.465 0.013699 *  
## other                                 1.321634   0.170636   7.745 9.53e-15 ***
## beachfront                           -0.317283   0.171558  -1.849 0.064398 .  
## keypad                               -0.377950   1.189142  -0.318 0.750611    
## indoor_fireplace                     -0.088863   0.212221  -0.419 0.675415    
## extra_pillows_and_blankets            0.320853   0.126155   2.543 0.010980 *  
## breakfast                             0.008855   0.193312   0.046 0.963466    
## free_street_parking                  -0.031258   0.110652  -0.282 0.777572    
## ethernet_connection                  -0.236069   0.149773  -1.576 0.114984    
## luggage_dropoff_allowed               0.226398   0.102009   2.219 0.026460 *  
## extra_space_around_bed               -0.085565   0.203313  -0.421 0.673863    
## smart_lock                           -1.055533   1.188419  -0.888 0.374443    
## crib                                 -0.126004   0.220341  -0.572 0.567418    
## paid_parking_off_premises            -0.639487   0.232815  -2.747 0.006019 ** 
## doorman                               0.425997   0.294175   1.448 0.147587    
## wide_hallways                         0.140460   0.194333   0.723 0.469816    
## disabled_parking_spot                -0.163582   0.209390  -0.781 0.434665    
## suitable_for_events                  -0.422730   0.229327  -1.843 0.065278 .  
## high_chair                           -0.259941   0.163309  -1.592 0.111449    
## childrens_books_and_toys              0.115499   0.178421   0.647 0.517412    
## lockbox                              -0.486144   1.192022  -0.408 0.683397    
## pocket_wifi                           0.526442   0.221382   2.378 0.017408 *  
## lake_access                          -0.159674   0.284039  -0.562 0.574010    
## translation_missing_enhostingamenity  1.463165   0.150068   9.750  < 2e-16 ***
## roomdarkening_shades                  0.189499   0.137859   1.375 0.169259    
## fixed_grab_bars_for_shower            0.439916   0.322829   1.363 0.172979    
## babysitter_recommendations           -0.281233   0.242712  -1.159 0.246573    
## childrens_dinnerware                  0.035061   0.257072   0.136 0.891516    
## accessibleheight_bed                 0.193033   0.210523   0.917 0.359184
## wide_doorway_to_guest_bathroom       0.667176   0.295867   2.255 0.024134 *
## handheld_shower_head                 0.099797   0.209650   0.476 0.634062
## pool_with_pool_hoist                 0.420918   0.528670   0.796 0.425926
## baby_bath                            0.500544   0.295915   1.692 0.090739 .
## wide_clearance_to_shower            -0.772138   0.363699  -2.123 0.033753 *
## `_toilet`                           -0.280559   0.476871  -0.588 0.556308
## ev_charger                           0.533239   0.433655   1.230 0.218833
## pets_live_on_this_property           0.192493   0.343220   0.561 0.574904
## dogs                                 0.555966   0.386810   1.437 0.150630
## window_guards                       -0.256055   0.316594  -0.809 0.418642
## buzzerwireless_intercom             -0.283438   0.358113  -0.791 0.428667
## game_console                         0.608391   0.314453   1.935 0.053020 .
## cats                                 0.143971   0.428690   0.336 0.736992
## outlet_covers                       -0.354868   0.359534  -0.987 0.323632
## stair_gates                         -0.694716   0.478300  -1.452 0.146371
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5682.7  on 6194  degrees of freedom
## Residual deviance: 4148.2  on 6087  degrees of freedom
## AIC: 4364.2
##
## Number of Fisher Scoring iterations: 6
```

The logistic regression provide the results. However, 107 dummy variables are too much, we would prefer decrease the number. Stepwise selection is applied here to eliminate unnecessary variables.

```
#Running this chunk could take considerable time, DO WITH CAUTION
#library(MASS)
#fitLRM2 <- fitLRM %>% stepAIC(trace = FALSE)
#detach("package:MASS", unload = TRUE)
fitLRM2 <-
  glm(formula = high_booking_rate ~ wifi + air_conditioning + pool +
    kitchen + gym + elevator + washer + dryer + self_checkin +
    building_staff + bed_linens + dishes_and_silverware + cleaning_before_checkout +
    waterfront + free_parking_on_premises + familykid_friendly +
    carbon_monoxide_detector + shampoo + laptop_friendly_workspace +
    microwave + coffee_maker + dishwasher + pets_allowed + first_aid_kit +
    host_greets_you + hair_dryer + safety_card + hour_checkin +
    private_entrance + single_level_home + bbq_grill + beach_essentials +
    other + beachfront + extra_pillows_and_blankets + ethernet_connection +
    luggage_dropoff_allowed + smart_lock + paid_parking_off_premises +
    doorman + suitable_for_events + high_chair + pocket_wifi +
    translation_missing_enhostingamenity + roomdarkening_shades +
    wide_doorway_to_guest_bathroom + wide_clearance_to_shower +
    dogs + game_console, family = "binomial", data = dfamenity_Miami_Dummy)
summary(fitLRM2)
```

```
##
## Call:
```

```
## glm(formula = high_booking_rate ~ wifi + air_conditioning + pool +
##     kitchen + gym + elevator + washer + dryer + self_checkin +
##     building_staff + bed_linens + dishes_and_silverware + cleaning_before_checkout +
##     waterfront + free_parking_on_premises + familykid_friendly +
##     carbon_monoxide_detector + shampoo + laptop_friendly_workspace +
##     microwave + coffee_maker + dishwasher + pets_allowed + first_aid_kit +
##     host_greets_you + hair_dryer + safety_card + hour_checkin +
##     private_entrance + single_level_home + bbq_grill + beach_essentials +
##     other + beachfront + extra_pillows_and_blankets + ethernet_connection +
##     luggage_dropoff_allowed + smart_lock + paid_parking_off_premises +
##     doorman + suitable_for_events + high_chair + pocket_wifi +
##     translation_missing_enhostingamenity + roomdarkening_shades +
##     wide_doorway_to_guest_bathroom + wide_clearance_to_shower +
##     dogs + game_console, family = "binomial", data = dfamenity_Miami_Dummy)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5330  -0.5515  -0.3035  -0.1440   3.3224
##
## Coefficients:
##                                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)                           -4.33927    0.66947  -6.482 9.07e-11 ***
## wifi                                   0.76418    0.39503   1.934 0.053057 .
## air_conditioning                       0.86061    0.62592   1.375 0.169148
## pool                                  -0.32445    0.09235  -3.513 0.000442 ***
## kitchen                               -0.50249    0.13907  -3.613 0.000302 ***
## gym                                   -0.63323    0.16422  -3.856 0.000115 ***
## elevator                              -0.52963    0.17224  -3.075 0.002105 **
## washer                                -0.27684    0.12541  -2.208 0.027276 *
## dryer                                 -0.58530    0.24937  -2.347 0.018917 *
## self_checkin                           1.33142    0.11676  11.404  < 2e-16 ***
## building_staff                        -0.73820    0.17999  -4.101 4.11e-05 ***
## bed_linens                            -0.18881    0.12837  -1.471 0.141335
## dishes_and_silverware                  0.27630    0.15633   1.767 0.077166 .
## cleaning_before_checkout              -0.39547    0.19280  -2.051 0.040246 *
## waterfront                            -0.30093    0.13937  -2.159 0.030831 *
## free_parking_on_premises              -0.28330    0.14052  -2.016 0.043788 *
## familykid_friendly                     1.12596    0.09127  12.336  < 2e-16 ***
## carbon_monoxide_detector               0.29229    0.09118   3.206 0.001347 **
## shampoo                                0.51847    0.11664   4.445 8.79e-06 ***
## laptop_friendly_workspace              0.19943    0.10512   1.897 0.057797 .
## microwave                              0.53157    0.18090   2.939 0.003298 **
## coffee_maker                           0.33498    0.16808   1.993 0.046269 *
## dishwasher                            -0.23017    0.11001  -2.092 0.036423 *
## pets_allowed                          -0.35628    0.10292  -3.462 0.000537 ***
## first_aid_kit                         -0.13695    0.09037  -1.515 0.129664
## host_greets_you                        0.65141    0.15336   4.248 2.16e-05 ***
## hair_dryer                             0.42960    0.17850   2.407 0.016098 *
## safety_card                            0.36036    0.11626   3.099 0.001939 **
## hour_checkin                           0.54981    0.12970   4.239 2.25e-05 ***
## private_entrance                      -0.15340    0.08802  -1.743 0.081355 .
## single_level_home                      0.25699    0.10011   2.567 0.010259 *
## bbq_grill                             -0.43314    0.10599  -4.086 4.38e-05 ***
## beach_essentials                       0.26411    0.10965   2.409 0.016012 *
## other                                 1.31323    0.16186   8.113 4.93e-16 ***
## beachfront                           -0.29857    0.16273  -1.835 0.066551 .
## extra_pillows_and_blankets            0.35216    0.12134   2.902 0.003705 **
## ethernet_connection                  -0.26056    0.14506  -1.796 0.072469 .
## luggage_dropoff_allowed               0.21651    0.09360   2.313 0.020708 *
## smart_lock                           -0.63723    0.14703  -4.334 1.46e-05 ***
## paid_parking_off_premises            -0.61950    0.22925  -2.702 0.006886 **
## doorman                               0.56043    0.26702   2.099 0.035833 *
## suitable_for_events                  -0.44381    0.22353  -1.986 0.047088 *
## high_chair                           -0.25898    0.13590  -1.906 0.056696 .
## pocket_wifi                           0.48862    0.21471   2.276 0.022865 *
## translation_missing_enhostingamenity  1.44473    0.14380  10.047  < 2e-16 ***
## roomdarkening_shades                  0.18349    0.12717   1.443 0.149060
## wide_doorway_to_guest_bathroom        0.70912    0.24764   2.863 0.004190 **
## wide_clearance_to_shower             -0.83169    0.30106  -2.763 0.005736 **
## dogs                                  0.80493    0.24512   3.284 0.001024 **
## game_console                          0.52119    0.29785   1.750 0.080147 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5682.7  on 6194  degrees of freedom
## Residual deviance: 4186.5  on 6145  degrees of freedom
## AIC: 4286.5
##
## Number of Fisher Scoring iterations: 6
```

The reduced model Looks good. But with a careful observation, we found there are many variables share common dimensions, such as "pet allowed, dog", "wifi, pocket wifi". We prefer to manually decrease the information dimensions by our domain knowledge.

```
### Divide 7 dimensions to reflect generalize 107 features
entertaining <-
  c("wifi","pocket_wifi","air_conditioning",
    "game_console","ethernet_connection","beach_essentials")

key_features <-
  c("pool","kitchen","gym","bbq_grill")

essentials <-
  c("washer","dryer","shampoo","coffee_maker",
    "dishwasher","microwave",
    "dishes_and_silverware","hair_dryer","high_chair",
    "building_staff","bed_linens","extra_pillows_and_blankets")

property_design <-
  c("elevator","wide_doorway_to_guest_bathroom",
    "wide_clearance_to_shower","roomdarkening_shades",
    "private_entrance","beachfront","waterfront",
    "single_level_home","suitable_for_events",
    "laptop_friendly_workspace","familykid_friendly")

specialty <-
```

```r
  c("self_checkin","hour_checkin","host_greets_you",
    "luggage_dropoff_allowed","cleaning_before_checkout",
    "paid_parking_off_premises","free_parking_on_premises")

pet_rules <-
  c("dogs","dog","pets_allowed","cats","cat")

safety <-
  c("first_aid_kit","smart_lock",
    "safety_card","carbon_monoxide_detector","doorman")

### Reforge the dataset with dummies
dfamenity_Miami_Dummy_2 <- dfamenity_Miami_disjoint %>%
  select(-high_booking_rate) %>%
  mutate(entertaining = ifelse(word %in% entertaining, 1, 0),
         key_features = ifelse(word %in% key_features, 1, 0),
         essentials = ifelse(word %in% essentials, 1, 0),
         property_design = ifelse(word %in% property_design, 1, 0),
         specialty = ifelse(word %in% specialty, 1, 0),
         pet_rules = ifelse(word %in% pet_rules, 1, 0),
         safety = ifelse(word %in% safety, 1, 0)) %>%
  select(-word) %>%
  group_by(id) %>%
  mutate(entertaining=sum(entertaining),
         key_features=sum(key_features),
         essentials=sum(essentials),
         property_design=sum(property_design),
         specialty=sum(specialty),
         pet_rules=sum(pet_rules),
         safety=sum(safety)) %>%
  ungroup() %>%
  distinct(id, .keep_all = TRUE) %>%
  right_join(dfamenity_Miami, by="id") %>%
  mutate(entertaining = ifelse((entertaining==0|is.na(entertaining)), 0, 1),
         key_features = ifelse((key_features==0|is.na(key_features)), 0, 1),
         essentials = ifelse((essentials==0|is.na(essentials)), 0, 1),
         property_design = ifelse((property_design==0|is.na(property_design)), 0, 1),
         specialty = ifelse((specialty==0|is.na(specialty)), 0, 1),
         pet_rules = ifelse((pet_rules==0|is.na(pet_rules)), 0, 1),
         safety = ifelse((safety==0|is.na(safety)), 0, 1)) %>%
  select(-amenities)
```

Now we decreased the number of our dummy variables by repalcing them with dimension indicators, we hope to check how well our classification performs.

```r
### Chi-Square tests for each dummy variables
chisq.test(dfamenity_Miami_Dummy_2$high_booking_rate,dfamenity_Miami_Dummy_2$entertaining)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dfamenity_Miami_Dummy_2$high_booking_rate and dfamenity_Miami_Dummy_2$entertaining
## X-squared = 3.3781, df = 1, p-value = 0.06607
```

```r
chisq.test(dfamenity_Miami_Dummy_2$high_booking_rate,dfamenity_Miami_Dummy_2$key_features)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dfamenity_Miami_Dummy_2$high_booking_rate and dfamenity_Miami_Dummy_2$key_features
## X-squared = 17.572, df = 1, p-value = 2.766e-05
```

```r
chisq.test(dfamenity_Miami_Dummy_2$high_booking_rate,dfamenity_Miami_Dummy_2$essentials)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dfamenity_Miami_Dummy_2$high_booking_rate and dfamenity_Miami_Dummy_2$essentials
## X-squared = 20.813, df = 1, p-value = 5.063e-06
```

```r
chisq.test(dfamenity_Miami_Dummy_2$high_booking_rate,dfamenity_Miami_Dummy_2$property_design)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dfamenity_Miami_Dummy_2$high_booking_rate and dfamenity_Miami_Dummy_2$property_design
## X-squared = 49.994, df = 1, p-value = 1.542e-12
```

```r
chisq.test(dfamenity_Miami_Dummy_2$high_booking_rate,dfamenity_Miami_Dummy_2$specialty)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dfamenity_Miami_Dummy_2$high_booking_rate and dfamenity_Miami_Dummy_2$specialty
## X-squared = 62.444, df = 1, p-value = 2.741e-15
```

```r
chisq.test(dfamenity_Miami_Dummy_2$high_booking_rate,dfamenity_Miami_Dummy_2$pet_rules)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dfamenity_Miami_Dummy_2$high_booking_rate and dfamenity_Miami_Dummy_2$pet_rules
## X-squared = 5.0084, df = 1, p-value = 0.02522
```

```r
chisq.test(dfamenity_Miami_Dummy_2$high_booking_rate,dfamenity_Miami_Dummy_2$safety)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dfamenity_Miami_Dummy_2$high_booking_rate and dfamenity_Miami_Dummy_2$safety
## X-squared = 95.087, df = 1, p-value < 2.2e-16
```

The “entertaining” doesn’t work well; also, “pet allow” looks like on the verge of being knock out. But overall, we are satisfied with the results.

We also would like to see how well they perform when we put them in a model simultaneously.

```
### Rerun the logistic regression
fitLRM3<-
  glm(formula = high_booking_rate ~.-id, family = "binomial", data = dfamenity_Miami_Dummy_2)
summary(fitLRM3)
```

```
##
## Call:
## glm(formula = high_booking_rate ~ . - id, family = "binomial",
##     data = dfamenity_Miami_Dummy_2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9896  -0.6842  -0.5310  -0.3119   3.0393
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -4.60860    0.80308  -5.739 9.54e-09 ***
## entertaining    -0.11620    0.78001  -0.149  0.88158
## key_features    -0.80482    0.13109  -6.140 8.28e-10 ***
## essentials       1.20088    0.38750   3.099  0.00194 **
## property_design  1.04212    0.18123   5.750 8.91e-09 ***
## specialty        1.33001    0.23988   5.545 2.95e-08 ***
## pet_rules        0.06869    0.08081   0.850  0.39529
## safety           0.62375    0.08388   7.436 1.03e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5682.7  on 6194  degrees of freedom
## Residual deviance: 5439.2  on 6187  degrees of freedom
## AIC: 5455.2
##
## Number of Fisher Scoring iterations: 5
```

The results shown that **entertaining** are **pet__rules** are not siginificant, providing limited contributions in explaining the high booking rate.

To dig further information, we want to check the adjusted odds ratios (and confidence intervals) for all 7 dimensions

```
suppressMessages(exp(cbind(coef(fitLRM3), confint(fitLRM3))))
```

```
##                                    2.5 %     97.5 %
## (Intercept)     0.009965723 0.001459956 0.03917096
## entertaining    0.890301530 0.233990291 5.85432107
## key_features    0.447170265 0.346838993 0.58016007
## essentials      3.323026102 1.657565990 7.71487166
## property_design 2.835222825 2.017275641 4.11367243
## specialty       3.781075448 2.429862024 6.25827720
## pet_rules       1.071103050 0.913011793 1.25339264
## safety          1.865903618 1.585826721 2.20349904
```

To our surprise, “key feature”, representing the dimension that the property includes facilities like “kitchen”, “gym” are related to a low booking rate.

Transportation

Another important feature we are interested with is whether transportation information is another focus of airbnb users?

Different to “amenities” that are filled in fixed format, “transit” are filled in free format, meaning it could be much harder to find common expressions and extract useful information from “transit”.

The transportation information exploration will be quite similar to amenity.

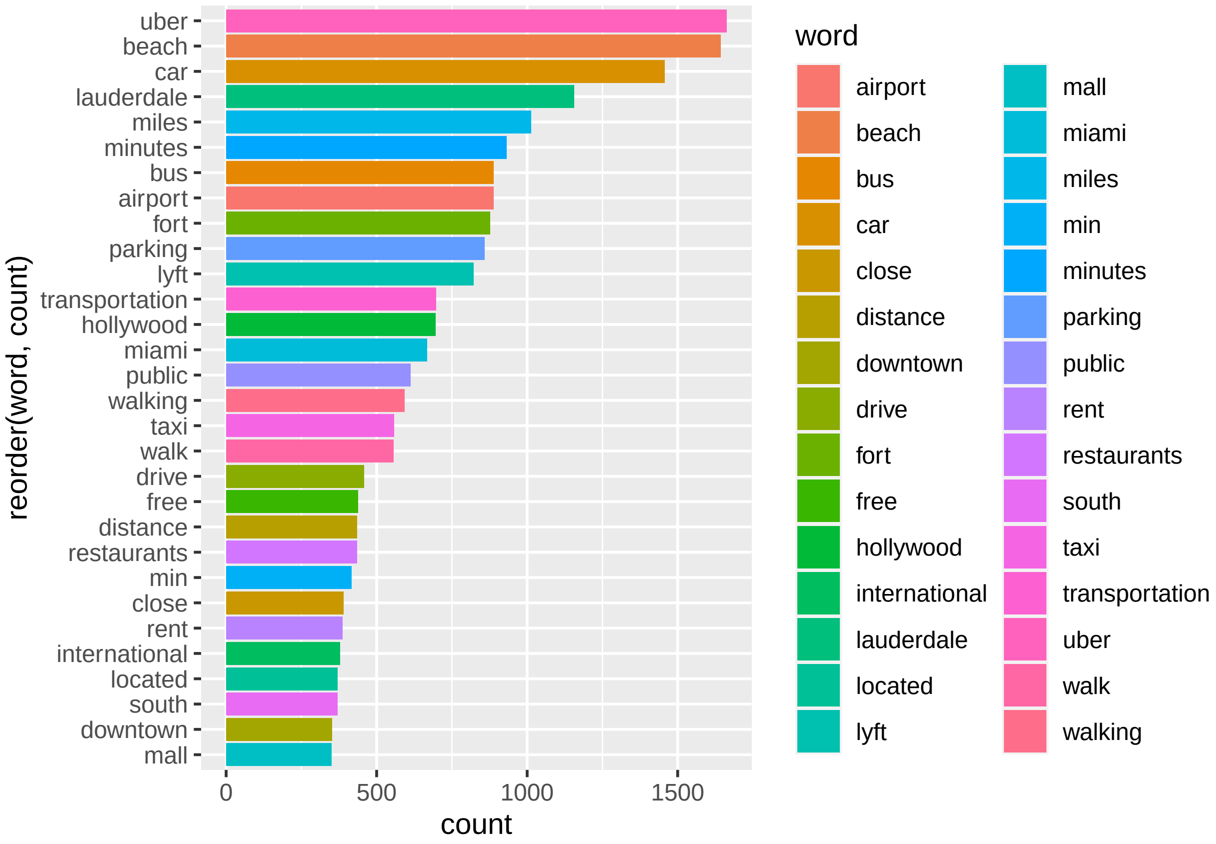
```
df_trainsit <- dfTrain_Miami %>%
  rename(reviewText = transit)
df_trainsit$reviewText <- gsub('[0-9]+', '', df_trainsit$reviewText)
df_trainsit$reviewText <- tolower(df_trainsit$reviewText)
df_trainsit$reviewText <- gsub('[[:punct:] ]+|/|@|\\|',' ',df_trainsit$reviewText)

dft_tidy <-
  df_trainsit %>%
  unnest_tokens(word, reviewText) %>%
  anti_join(stop_words)
```

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

```
### Show words those appeared more than 100 times
dfwordcount <- dft_tidy %>%
  filter(!is.na(word)) %>%
  group_by(word) %>%
  summarise(count=n()) %>%
  filter(count>=100)
dfwordcount %>%
  arrange(desc(count))
```

```
## # A tibble: 113 x 2
##    word       count
##    <chr>      <int>
##  1 uber        1662
##  2 beach       1643
##  3 car         1457
##  4 lauderdale  1156
##  5 miles       1012
##  6 minutes      932
##  7 airport      888
##  8 bus          888
##  9 fort          877
## 10 parking       859
## # ... with 103 more rows
```

```
### Visualize the word frequence
dfwordcount %>%
  arrange(desc(count)) %>%
  head(30) %>%
  ggplot(aes(reorder(word,count), count, fill = word)) +
  geom_col(show.legend = TRUE) +
  coord_flip() +
  scale_x_reordered()
```



From the word vector, we used naked eye research and extract words related to transporation. Then we assigned them into two main categories: *public* or *rental+ride*.

```
public <-
  c('bus', 'buses', 'busses', 'train', 'trains',
    'walk', 'walking', 'walkable', 'foot', 'bike',
    'bikes', 'citibike', 'biking', 'bicycle', 'bicycles',
    'bikeshare', 'rideshare', 'cycle', 'biketown', 'biki',
    'citibikes','pick', 'pickup', 'picks', 'scooters', 'scooter',
    'scoot','metro', 'subway', 'subways', 'rail', 'amtrak',
    'express', 'metrolink', 'railroad','shuttle', 'shuttles',
    'trolley', 'trolleys','commute')
```

```r
ride_rental <-
  c('uber', 'ubers', 'über', 'uberx',
    'lyft', 'lyfts', 'cab', 'cabs',
    'taxi', 'taxis', 'car', 'cars',
    'driving', 'ride', 'rides', 'riding',
    'rental', 'rent', 'rentals', 'renting',
    'rented', 'streetcar', 'streetcars', 'zipcar')

### Reforge the dataset with dummies
dft_dummy <- dft_tidy %>%
  mutate(public = ifelse(word %in% public, 1, 0)) %>%
  mutate(ride_rental = ifelse(word %in% ride_rental, 1, 0)) %>%
  select(id,public,ride_rental) %>%
  group_by(id) %>%
  mutate(public=sum(public),
         ride_rental=sum(ride_rental)) %>%
  ungroup() %>%
  distinct(id, .keep_all = TRUE) %>%
  right_join(select(dfTrain_Miami,id,high_booking_rate), by="id") %>%
  mutate(public = ifelse((public==0|is.na(public)), 0, 1),
         ride_rental = ifelse((ride_rental==0|is.na(ride_rental)), 0, 1))
```

We would like to test how well our dummy variables work using the chi-square test.

```r
chisq.test(dft_dummy$high_booking_rate,dft_dummy$public)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dft_dummy$high_booking_rate and dft_dummy$public
## X-squared = 200.15, df = 1, p-value < 2.2e-16
```

```r
chisq.test(dft_dummy$high_booking_rate,dft_dummy$ride_rental)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dft_dummy$high_booking_rate and dft_dummy$ride_rental
## X-squared = 271.15, df = 1, p-value < 2.2e-16
```

The outputs are excellent, the dummy variables we made really do the trick to tell the difference between booking rate.

Again, we also test whether they will affect each other while putting them together.

```r
fitLRM4<-
  glm(formula = high_booking_rate ~.-id, family = "binomial", data = dft_dummy)
summary(fitLRM4)
```

```
##
## Call:
```

```
## glm(formula = high_booking_rate ~ . - id, family = "binomial",
##     data = dft_dummy)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8580  -0.6941  -0.4691  -0.4691   2.1268
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.15154    0.05290 -40.675  < 2e-16 ***
## public       0.49070    0.08172   6.004 1.92e-09 ***
## ride_rental  0.85095    0.08171  10.415  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5682.7  on 6194  degrees of freedom
## Residual deviance: 5383.8  on 6192  degrees of freedom
## AIC: 5389.8
##
## Number of Fisher Scoring iterations: 4
```

These two dummies still work very well.

We also curious about the adjusted odds ratios for these two dummies

```r
suppressMessages(exp(cbind(coef(fitLRM4), confint(fitLRM4))))
```

```
##                             2.5 %    97.5 %
## (Intercept) 0.1163051 0.1047101 0.1288437
## public      1.6334551 1.3916156 1.9172161
## ride_rental 2.3418686 1.9955364 2.7490792
```

It seems that people do care about transporation convenience. However, public transportation in miami is not that attractive. Airbnb users in Miami are more care about private transportation feature.

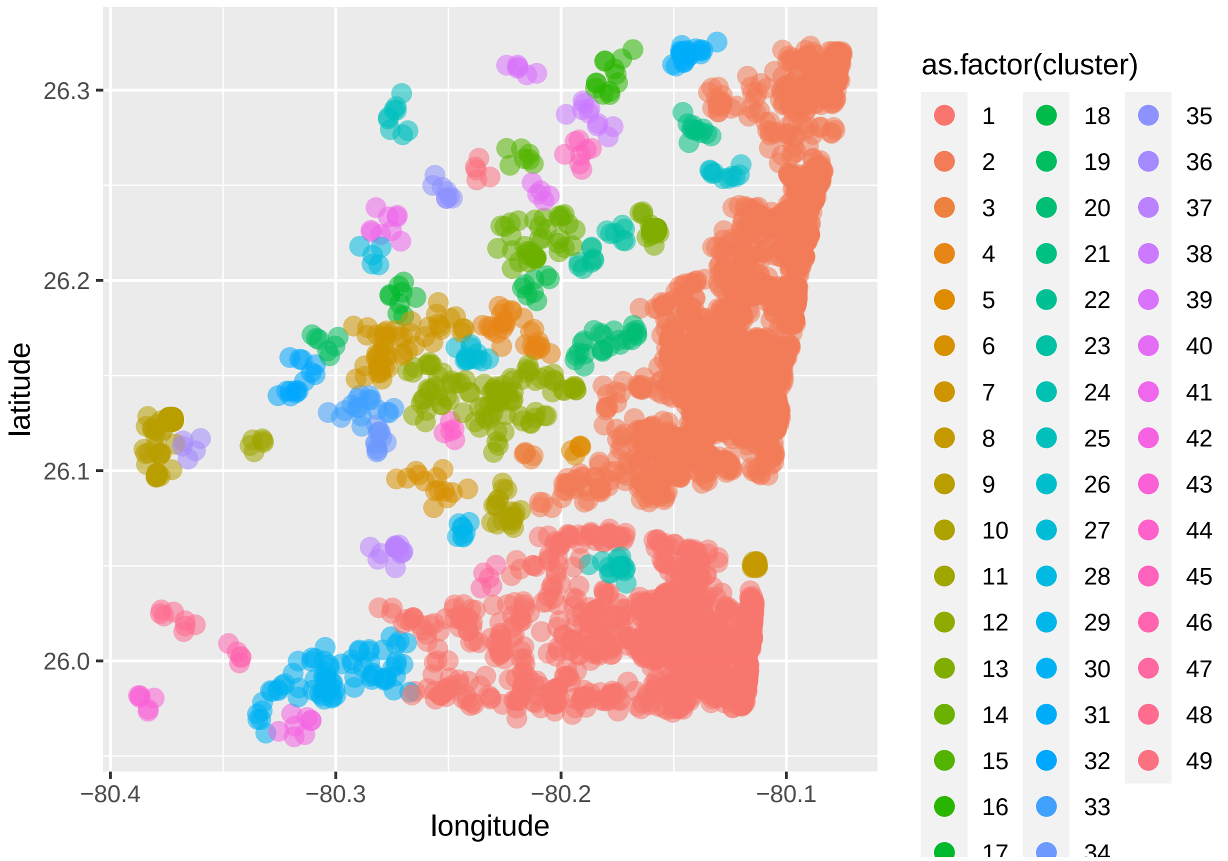
Results & Findings

The results and Findings will also be divided in to 3 sections.

Geographical Clustering

(obtaining the results)

```
clusterplot
```



```
dfgeo_Miami %>%
  group_by(cluster) %>%
  mutate(pct = 100*sum(high_booking_rate)/length(high_booking_rate)) %>%
  group_by(cluster,pct) %>%
  tally() %>%
  arrange(desc(pct))
```

```
## # A tibble: 50 x 3
## # Groups:   cluster [50]
##    cluster   pct     n
##      <int> <dbl> <int>
##  1      39  83.3     6
##  2       3  80       5
##  3      36  40       5
##  4       29  37.5     8
##  5       25  33.3     9
##  6       43  33.3     6
##  7       19  28.6     7
##  8       34  27.3    11
##  9       22  25       8
## 10       48  25       8
## # ... with 40 more rows
```

```
dfgeo_Miami %>%
  group_by(cluster) %>%
  mutate(pct = 100*sum(high_booking_rate)/length(high_booking_rate)) %>%
  group_by(cluster,pct) %>%
  tally() %>%
  ungroup() %>%
  summarise(pct_median=median(pct))
```

```
## # A tibble: 1 x 1
##   pct_median
##        <dbl>
## 1       14.0
```

From the table we can see that clusters are not divded evenly. There are two clusters contain the majority of properties and the rests are divided by the remaining 47 clusters.

The median of high booking rate percent is 13.96, which is a good news for new investors that both of the two major clusters are above the 50th percentile. However, cluster 2 has a much higher ranking than cluster 1. Based on the geo-location, we found the properties in cluster 2 are clustered more close to the beachfront than cluster 1, which gives us a sign that properties at beachfront will enjoy a relatively high booking rate.

The highest booking rate clusters are cluster 39 and 3, cluster 39 is at . We suppose these two clusters are at the some famous tourist spot or wealthy suburb. However, the data within clusters are too few and thus hard to make a persuasive recommendation: these two places might be good for sophisticated investors.

One more insight from these two major clusters is that we considered the low percent rate in absolute values come from the homogenization of the properties within these two clusters. Since many of the properties share very common geographical features, airbnb users will probably randomly pickup one property and hence causing the low absolute percent rate values. Therefore, exploring other features that can make one's property more outstanding will be our future tasks.

Amenity Exploration

(Obtaining the results)

```
summary(fitLRM3)
```

```
##
## Call:
## glm(formula = high_booking_rate ~ . - id, family = "binomial",
##     data = dfamenity_Miami_Dummy_2)
##
```

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9896  -0.6842  -0.5310  -0.3119   3.0393
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -4.60860    0.80308  -5.739 9.54e-09 ***
## entertaining    -0.11620    0.78001  -0.149  0.88158
## key_features    -0.80482    0.13109  -6.140 8.28e-10 ***
## essentials       1.20088    0.38750   3.099  0.00194 **
## property_design  1.04212    0.18123   5.750 8.91e-09 ***
## specialty        1.33001    0.23988   5.545 2.95e-08 ***
## pet_rules        0.06869    0.08081   0.850  0.39529
## safety           0.62375    0.08388   7.436 1.03e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5682.7  on 6194  degrees of freedom
## Residual deviance: 5439.2  on 6187  degrees of freedom
## AIC: 5455.2
##
## Number of Fisher Scoring iterations: 5
```

The results from the logistic regression model shown that **entertaining** are **pet_rules** are not siginificant. The reason we could think of is that some “dimensions” we hypothesized are not completely independent to others (not perpendicular in dimensions) and thus causes serious collinearity.

```
suppressMessages(exp(cbind(coef(fitLRM3), confint(fitLRM3))))
```

```
##                                    2.5 %     97.5 %
## (Intercept)     0.009965723 0.001459956 0.03917096
## entertaining    0.890301530 0.233990291 5.85432107
## key_features    0.447170265 0.346838993 0.58016007
## essentials      3.323026102 1.657565990 7.71487166
## property_design 2.835222825 2.017275641 4.11367243
## specialty       3.781075448 2.429862024 6.25827720
## pet_rules       1.071103050 0.913011793 1.25339264
## safety          1.865903618 1.585826721 2.20349904
```

Most of the results are as our expected except for “key features”. We have no idea why hosts enlist facilities liks “gym”, “pool”, “grill”, “kitchen” has an negative effect on high booking rate, and works so significantly.

Overall, we are satisfied with our outputs. We hypothesized 7 dimensions to capture the features shown in amenities and most of them proved to be useful. We would recommend investors and hosts to fill in the amenities that cover up these 7 dimensions as much as possible.

Transportation

(Obtaining the results)

```
summary(fitLRM4)
```

```
##
## Call:
## glm(formula = high_booking_rate ~ . - id, family = "binomial",
##     data = dft_dummy)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8580  -0.6941  -0.4691  -0.4691   2.1268
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.15154    0.05290 -40.675  < 2e-16 ***
## public       0.49070    0.08172   6.004 1.92e-09 ***
## ride_rental  0.85095    0.08171  10.415  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5682.7  on 6194  degrees of freedom
## Residual deviance: 5383.8  on 6192  degrees of freedom
## AIC: 5389.8
##
## Number of Fisher Scoring iterations: 4
```

The two dummies we proposed works well in reflecting the difference in booking rate.

```
suppressMessages(exp(cbind(coef(fitLRM4), confint(fitLRM4))))
```

```
##                           2.5 %    97.5 %
## (Intercept) 0.1163051 0.1047101 0.1288437
## public      1.6334551 1.3916156 1.9172161
## ride_rental 2.3418686 1.9955364 2.7490792
```

These results show that airbnb users do care about transportation convenience, and private transportation options like uber/lyft services or rental car services seems to be more decisive when compared to public transportation services to the factor of high booking rate. This makes sense because most airbnb users in Miami are for tourism and private transportation options will enable them enjoy the vacations with more satisfaction. Therefore we would recommend hosts and investors to fill in the transportation availability around their properties. Public options are good, but private options are better.

Conclusion & Discussion

Conclusion

As a nutshell, we conclude our findings as follows:

To achieve a high booking rate,

1. Select a nice location. For a safe and long term investment, beachfront is always the best choice! Besides that, somewhere around the center also looks promising. However, because of lacking sufficient data, this recommedation should be taken with caution and only for risk seekers: The low observations and comparably high booking rate within those locations indicates they could be very promising new lands which are blind spots to other common players. Chances are that investors could earn a great amount from those places.
2. Fill in the amenities and cover up all 7 dimensions we generalized as much as possible:
   - Entertaining
   - Key features
   - Pet allowance
   - Safety
   - Essentials
   - Specialty
   - Property Design
3. Fill in the available transportation options around your properties. Mention about how convenient to reach transportation from your properties will benefit the booking rate. Mentioning public transportation is good, but mentioning private transportation options such as pickup services and rental car services availability are far better!

Limitation

Our research has several flaws.

1. Since we lack neighbourhood information, we use geographical information and **Dbscan** package to "reconstruct" the 'neighbourhood' information. However, this method is data hungry and since our data is not very balanced, it segemented too many clusters with few observations, which makes us hard to draw some conclusions with confidence.
2. The dimension deduction in amenity infomation extraction is naive and no data support. Because of that, the dimensions we hypothesized are not completely successful: some of them are not completely independent to others. Conducting a principal component analysis on these amenity may produce more insights and better describing the latent dimensions that these amenities are representing. Also, we only verify those dimensions by qualitative analysis, a quantitative analysis may also prodive to which extent should a host describe about his/her property.
3. Transportation text mining is much harder than we thought since the "transit" column in airbnb is in free format. Currently we use bag of words –> filter stop word –> filter out least common words and then do naked eye research to find patterns. However, these processes could be misleading since:
   1. Elminate some of the inputs carrying useful information.

      For example, input like "Next to 95" is saying this property being next to a bus station, but it will be eliminated after our cleaning processes.

2. Misclassifications due to ambiguity

   Words like “rental”, “renting” are ambiguous. We do not actually know if it is refering to hosts’ rental property or rental car services. We consider them uniformly refering rental car services but can introduce potential misclassifications.

Future Research Plan

In our future research plan, we plan to

1. Narrow our scope and focus on cluster 1 and 2 to find out what features made beachfront properties stand out.
2. Try to dig deeper and, on the premise that if it is not caused by too few observations within clusters 3 and 39, explore which features make these two clusters have outstandingly high booking rate.
3. Projecting the miami segementation map onto our data to divide it in a more reasonable and close to reality way.
4. Find out the reason why “key features” in our hypothesis works on the opposite direction to our common understanding.
5. Do a Principal Component Analysis on amenities to extract latent dimensions.
6. Focus on explaining other features, such as:

   - What qualify a host being a super host?
   - High booking rate doesn’t necessarily relates to high revenue because it will require hosts spending more efforts on maintaining his/her properties, how to justify investors’ investments within miami market?

Reference

Al Saleem, A. S. M. R., & Al-Juboori, N. F. M. (2013, October). Factors Affecting Hotels Occupancy Rate (An Empirical ... Retrieved May 8, 2020, from https://journal-archieves36.webs.com/142-159.pdf

Kassambara, A. (2018, October 25). DBSCAN: Density-Based Clustering Essentials. Retrieved May 8, 2020, from https://www.datanovia.com/en/lessons/dbscan-density-based-clustering-essentials/

Bhattacharyya, S. (2019, December 11). DBSCAN Algorithm: Complete Guide and Application with Python Scikit-Learn. Retrieved May 8, 2020, from https://towardsdatascience.com/dbscan-algorithm-complete-guide-and-application-with-python-scikit-learn-d690cbae4c5d

The Storage Queens. (2018, March 15). The 9 Best Neighborhoods to Live in South Miami: CubeSmart. Retrieved from https://www.cubesmart.com/blog/city-guides/miami/the-9-best-neighborhoods-to-live-in-south-miami/

Appendix

We mentioned in our presentation that we reached an AUC=1 in our sub-divided train-test pair in Miami Training set. It turned out the coding has some mistakes. We reconstructed and tuned the codes to cover all of our information. But because of the existence of extremely low observations in certain categories, the sub-train set and sub-test set can not be in the same shape. Therefore, we couldn't test the performance of our hypothesized dummy variables and this attempt is ceased.