Exploratory Data Analysis of Airbnb Listing

Art Tay

Introduction

Airbnb, Inc. is a software company that operates a marketplace for people to rent their properties on an ad hoc basis (Wikipedia). The specific dataset addressed in this report comes from Inside Airbnb, "a mission driven project that provides data ... about Airbnb's impact on residential communities." (Website). The data set contains 12 predictive variables: 4 categorical and 8 numeric. All of the roughly 40,000 listings in this particular dataset were located in New York City. The ultimate goal of this analysis was to categorize the different types of Airbnb listings and their hosts, with a particular emphasis on determining which listings had the most business traffic.

Summary of Results

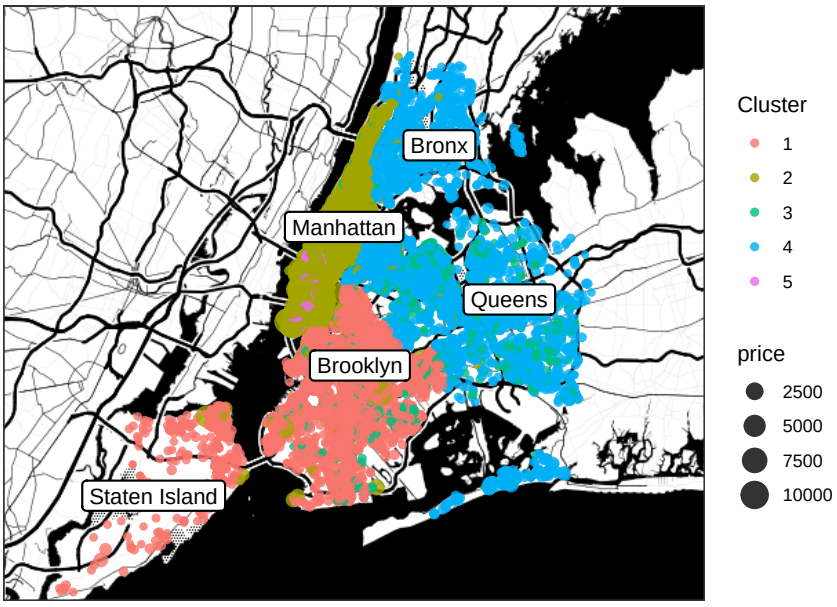

Figure 1: Geographic plot of cluster and price.

Ultimately 5 different groups or clusters of Airbnbs were identified. Clusters 1, 2, and 4 were mainly geographically tied. Cluster 1 contained most of the listings from Staten Island and Brooklyn, Cluster 2 was tied to Manhattan and Cluster 4 encompassed the Bronx and Queens. Cluster 5 in made up of the most expensive listings. They are all located in Manhattan, but are rarely booked. Finally, Cluster 3 was determined to be the busiest Airbnbs. Cluster 3 was geographically disperse, medianly priced, and had the most number and recency of reviews. If we assume that every review represents at least one booking, then Cluster 3 is by far the listings that attract the most business.

Methodology

Data Cleaning

Before the modeling process could begin 3 important issues had to be address in the raw data: missing values, feature engineering, and skewness.

Missing Values

Upon first inspection there are only `NA` values in the `reviews_per_month` variable; however, there are also blanks in `last_review`, and zero values that appear erroneous. Any zeros in `price`, `lattitude`, `longitude`, and `minimum_night` were replaced with `NA`. Furthermore, any blank spaces in a factor variable was also considered to be `NA`. Below is a plot of the pattern and quantity of missing values after recoding.

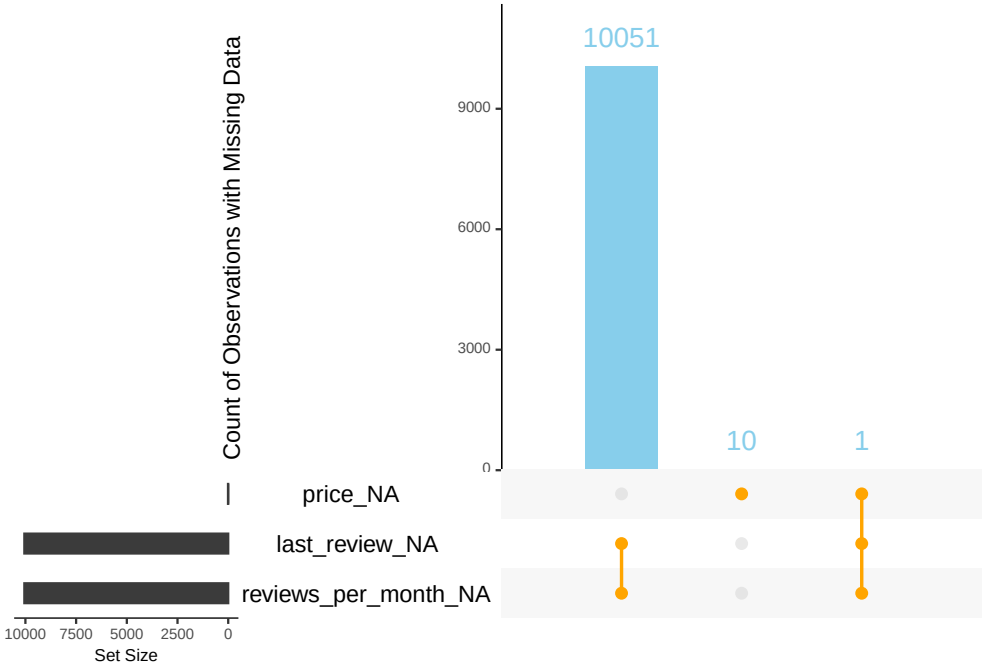

Figure 2: Quantity and pattern of missing data. The blue bars represent the quantity of the pattern shone on the matrix below. The black bar show the number of missing values of the variables to the right across all patterns.

Although it appears that there are roughly 10,000 missing values for `review_per_month` and `last_review`, the pattern is not random, in fact, **Figure 2** indications that these two variables are always missing together. Upon further inspection, the missingness of these two variable perfectly correlates with `number_of_review`

being equal to 0. Ultimately these are not actually missing values, they simple indicate that a particular listing has never been reviewed.

There were 11 listing where the `price` was `0`. Because price is such a key factor in the desirability of a particular listing, these `11` values were imputed using the k-nearest neighbors algorithm, where $k = 10$, because of the large number of observations. Since `price` is numeric, the median of the neighbor was used for the imputed value.

Feature Engineering

First, `NAs` for `review_per_month` were replaced with `0`. Next, `last_review` was replaced by `last_review_year` to reduce the number of factor levels. In hindsight, the day and month could have also been extracted as new features. Then, the `neighbourhood` factor was dropped because of the copious number of levels and the majority of the information being captured in `neighbourhood_group`. Perhaps a hierarchical feature could have been engineered. Finally, all factor variables were converted into `n` dummy variables and all numeric variables were normalized. This step was a requirement of the imputation algorithm used. Note that `impStatus` was added as a binary factor `1` if price was imputed, and `0` otherwise.

Skewness

Skewness was a key issue among almost every single numeric variable.

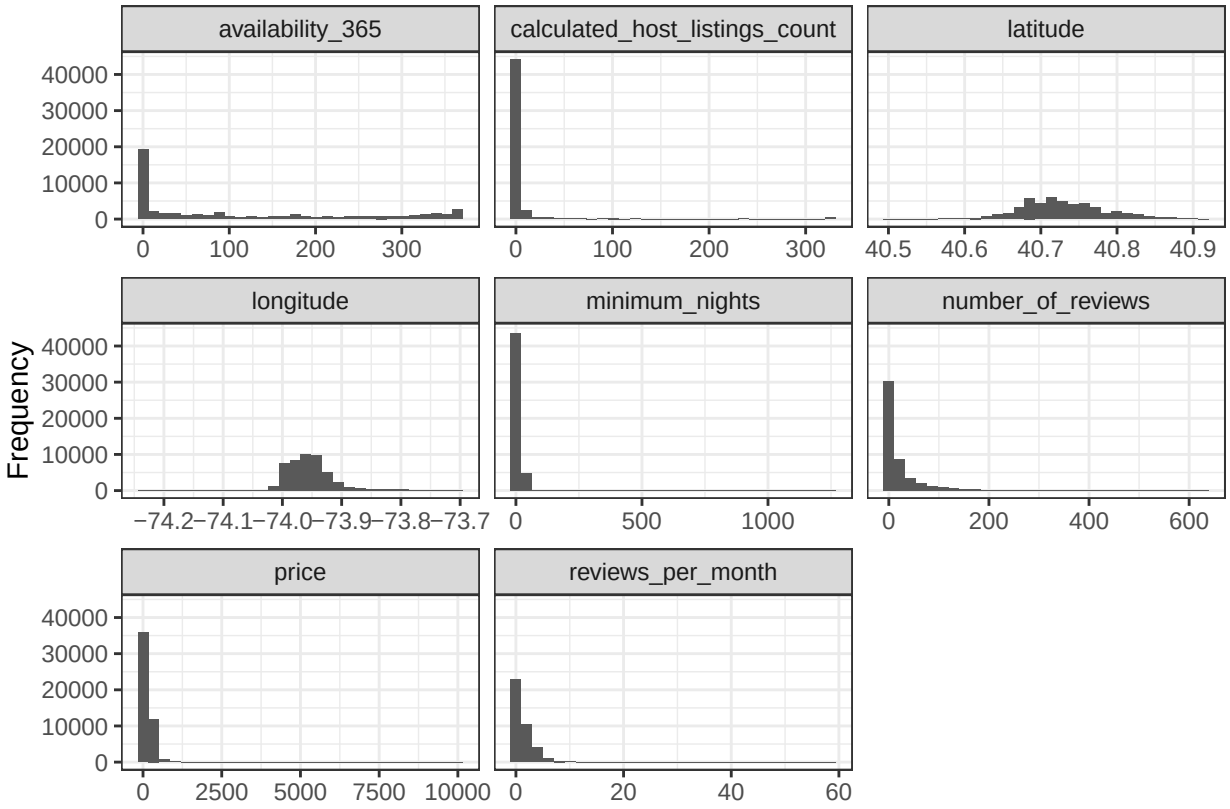

Figure 3: Histogram of raw numeric features before any transformations. Most distribution are clearly right skewed

The `price` was transformed using a log function, since all zero values were treat as `NA`. This was done to help the imputation process. The decision was made to not transform any of the other numeric predictors before clustering was completed. There are competing views in the literature about whether or not clustering algorithms are effected by skewness either negatively of positively. Because the clusters were ultimately fit using Euclidean distances and inter-cluster variability, applying transformations that would artificially reduce variance seem incorrect. Post clustering a Yeo-Johnson transformation was applied to the numeric variables only to increase visibility of plot features. Yeo-Johnson was chosen over Box-Cox because it can work with normalized data (negative values).

Cluster Modeling

A clustering algorithm was applied to the entire cleaned and imputed dataset as outlined above. The clustering algorithm used was implemented by the `NBClust` package. Clusters were determined based on a minimization of total within-cluster variance where dissimilarity was defined by Euclidean distance. This is know as WARD clustering (Ward, 1963). The number of cluster was determined based on maximizing the "silhouette" index (Rousseeuw, 1987). 2-15 clusters were tried.

Result

The cluster model determined that 5 was the optimal number of clusters.

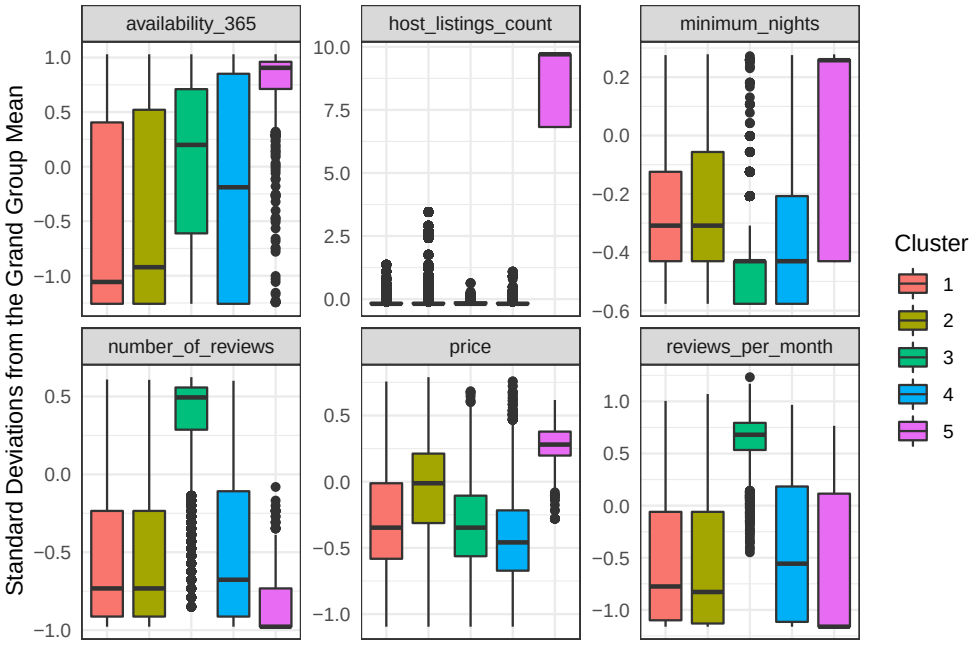

Figure 4: Side by side cluster grouped boxplots faceted across numeric predictors.The data was normalized to have a mean of zero and standard deviation of one as well as transformed via Yeos-Johnson. Grand Group Mean refers to the sample average of a given numeric variable across all clusters.

Cluster 3 and cluster 5 have the most variation among the five groupings. As mentioned in the introduction, cluster 3 appears to be the busiest type of Airbnbs. This conclusion was mostly draw from the `number_of_reviews` and the `review_per_month` since all review should correspond to a booking. **Figure 4** clearly illustrates that cluster 3 had the listings with the highest total number and frequency of reviews. Cluster 3 had a median almost an entire standard deviation above any other cluster.

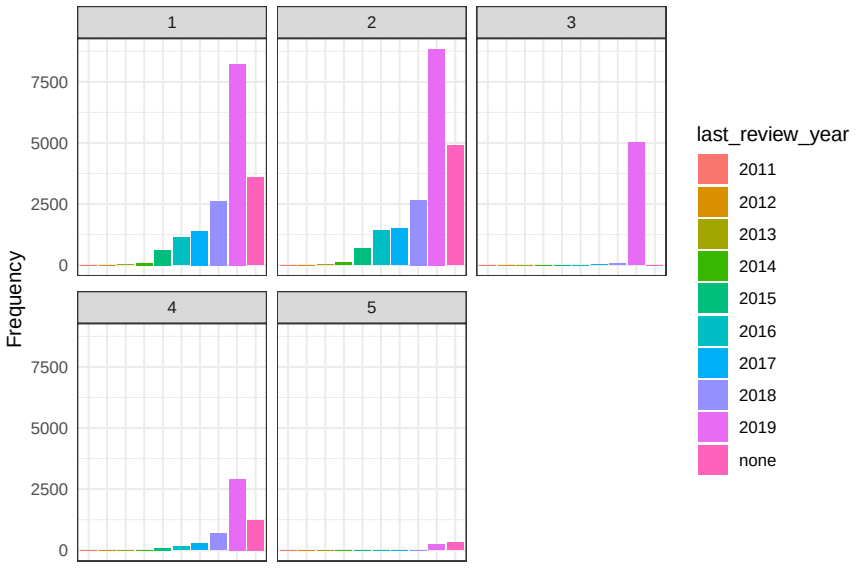

Figure 5: The frequency of which year the listing was last reviewed.

Figure 5 furthers that the majority of these reviews happened in the most recent year (in the data). Although cluster 3 had a middle of the road availability, this variable is quite inconclusive. One might think a lower availability means that a particular listing is being booked more often; however, according to the data dictionary provided by Inside Airbnb, this field could be low because the host is only renting it out for a small window, when they are on vacation for example. This means that a low availability could mean the listing is booked often or it could mean the host does not rent it often. A medium availability might actually be the best indicator of high volume business as the Airbnb could be offered and booked frequently on a rolling basis. Cluster 3 has one of the lowest median price points, which might make it attractive to more travelers.

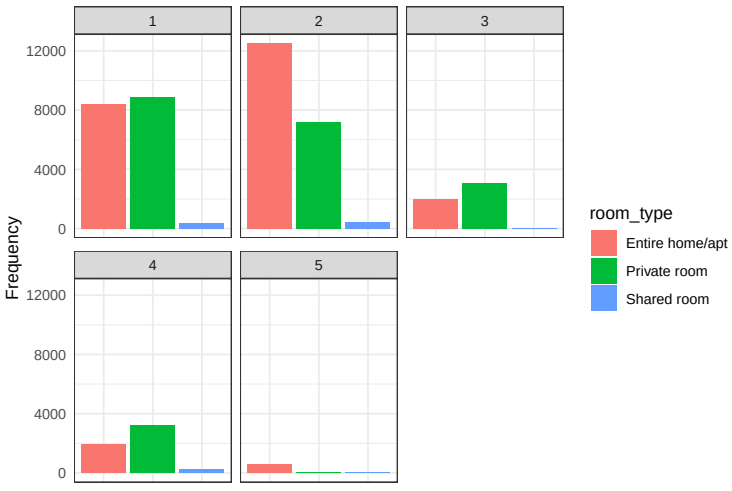

Figure 6: Frequency of the listing's room type across different clusters.

Figure 6 gives more color on the pricing in cluster 3 as it is mostly comprised of private rooms instead of entire homes or apartments.

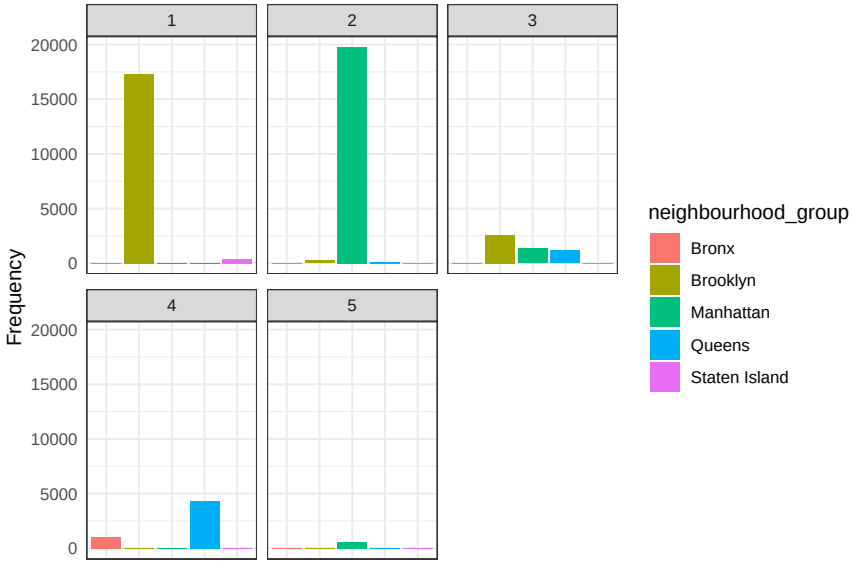

Figure 7: Borough frequency faceted across different clusters.

Unlike all of the other clusters, cluster 3 was geographically disperse across the 5 boroughs. This can be seen in **Figure 1** and **Figure 7**.

Although, cluster 3 probably generates the most business in terms of number of bookings, it might not be the best in term of revenue. **Figure 4** shows that cluster 5 has the highest median price and well as the highest duration requirement. This means that every booking would generate the hosts a large cash flow. Furthermore, cluster 5 has the most listings per host of any of the clusters. This implies that cluster 5 might represent corporate or investor owned properties. This might also explain why **Figure 6** shows that all the listings in cluster 5 are for the entire unit. On the other hand, Figure 1 also demonstrates that these properties are rarely ever booked. Cluster 5 has the lowest number of reviews as well as the highest amount of open availability.

Conclusion

The data was able to be grouped into 5 clusters. 3 of the cluster were based on geography a gave little insight into the business operations of these listings. Cluster 3 represents what most people think of when they think of a successful Airbnb. Single property owners that rent out a room to make some extra cash. Geographically disperse, medianly priced, and booked as well as availability often. Cluster 5 might also represent successful listings, all be it with a different business strategy. All the listings in cluster 5 are expensive whole unit rental properties located in Manhattan. Furthermore multiple listings are own by each host, and they require longer rental periods. Although not booked often, they appear to make up for their lack of volume with a premium price point. If these listings are profitable, it might indicate that Airbnb is going the way of corporate landlords instead of the individuals it was originally marketed to.

Appendix - Code

```
# Libraries
library(VIM)
library(naniar)
library(tidyverse)
library(fastDummies)
library(NbClust)
library(recipes)
library(corrplot)
```

```
# Load in Data
data_full <- read.csv("AB_NYC_2019.csv", stringsAsFactors = T, header = T)
```

Data Cleaning

```
# Removing uninformative variables (names).
data_quant <- data_full %>% select(-c(id, host_id, name, host_name))
```

Missing Data

```
# Code value that might mean missing.
# price == 0 -> NA
# lattitude == 0 -> NA
# longitude == 0 -> NA
# min_night == 0 -> NA

data_quant_mis <- data_quant %>%
    mutate(price = ifelse(price == 0, NA, price)) %>%
    mutate(latitude = ifelse(latitude == 0, NA, latitude)) %>%
    mutate(longitude = ifelse(longitude == 0, NA, longitude)) %>%
    mutate(minimum_nights = ifelse(minimum_nights == 0, NA, minimum_nights)) %>%
    # It was done using replace to maintain factor coding
    mutate(room_type = replace(room_type,
        room_type == "" | room_type == " ", NA)) %>%
    mutate(neighbourhood_group = replace(neighbourhood_group,
        neighbourhood_group == "" | neighbourhood_group == " ",  NA)) %>%
    mutate(neighbourhood = replace(neighbourhood,
        neighbourhood == "" | neighbourhood == " ", NA)) %>%
    mutate(last_review = replace(last_review,
        last_review == "" | room_type == " ", NA))

save(data_quant_mis, file = "Report Figures/data_quant_mis.rds")
```

```
# Plot the pattern of missing values
miss_pattern_plot <- gg_miss_upset(data_quant_mis, nintersects = NA, text.scale = 2,
    mainbar.y.label = "Count of Observations with Missing Data",
    point.size = 5, line.size = 2, matrix.color = "orange",
    main.bar.color = "skyblue")
```

Feature Engineering

```
# Handles all missing values that are not price
data_quant_fe <- data_quant_mis %>%
                 # Change NA reviews per month to be 0.
                 mutate(reviews_per_month =
                    ifelse(is.na(reviews_per_month), 0, reviews_per_month))%>%
                 # Create a new variable last_review_year to reduce dimensionality.
                 mutate(last_review_year = substring(last_review, 1, 4)) %>%
                 # Modify NA last_review to be a new level "none".
                 mutate(last_review_year = replace(last_review_year,
                    is.na(last_review_year), "none")) %>%
                 # Cast to be factor.
                 mutate(last_review_year = as.factor(last_review_year)) %>%
                 # Remove old last review variable
                 select(-last_review) %>%
                 # natural log transform price.
                 mutate(price = log(price))
```

```
# Centering a scale numerics.
data_quant_mis_numeric <- data_quant_fe %>% select(where(is.numeric))

# Retain vector to enable back transformation.
means <- apply(data_quant_mis_numeric, MARGIN = 2, FUN = mean, na.rm = T)
sds <- apply(data_quant_mis_numeric, MARGIN = 2, FUN = sd, na.rm = T)

# Apply transformation
for(i in 1:length(means)){
    data_quant_mis_numeric[, i] <-
        (data_quant_mis_numeric[, i] - means[i]) / sds[i]
}
```

```
# Dummy variables (one-hot-encoding)
data_quant_mis_factor <- data_quant_fe %>% select(where(is.factor)) %>%
                                           # too many levels.
                                           select(-neighbourhood)

data_quant_mis_factor <- dummy_cols(data_quant_mis_factor,
                                    remove_selected_columns = T)

# Combine cleaned data set for imputation.
data_quant_clean <- cbind(data_quant_mis_numeric, data_quant_mis_factor)
```

```
# kNN imputeation with k = 10
data_quant_clean_imputed <- VIM::kNN(as.matrix(data_quant_clean), k = 10)

# Create a factor based on whether or not price as imputed
impStatus <- as.numeric(data_quant_clean_imputed$price_imp)
```

```
# get the original columns
data_quant_clean_imputed <- data_quant_clean_imputed[, 1:26]
data_quant_clean_imputed$impStatus <- impStatus
```

Imputation

Skewness

```
# Histogram of all numerics
plot_skew <- data_quant %>%
           select(where(is.numeric)) %>%
           pivot_longer(cols = everything()) %>%
           ggplot(aes(value)) +
           geom_histogram() +
           facet_wrap(~name, scales = "free_x") +
           theme_bw() +
           labs(x = "", y = "Frequency")

save(plot_skew, file = "Report Figures/plot_skew.rds")
```

Clustering

```
#data_clusters <- NbClust(data = data_quant_clean_imputed,
                         #distance = "euclidean",
                         #method = "ward.D",
                         #index = "silhouette")
```

```
#save(data_clusters, file = "data_clusters.rds")
load(file = "data_clusters.rds")
```

Cluster 1 cleaning

```
# Back Transform numeric data.
# Extract numeric columns.
data_imputed_numeric <- data_quant_clean_imputed %>%
                        select(1:8)

# Undo centering and scaling.
for(i in 1:length(means)){
    data_imputed_numeric[, i] <-
        data_imputed_numeric[, i] * sds[i] + means[i]
}

# Undo log transformation on price.
data_imputed_numeric <- data_imputed_numeric %>%
                        mutate(price = exp(price)) %>%
                        # fixes rounding error
                        mutate(availability_365 = round(availability_365))
```

```
# Extract original factor formats.
data_imputed_factor <- data_quant_fe %>% select(where(is.factor))

# Confirm no missing
#sum(is.na(data_imputed_factor))
```

```
data_clean_cluster_1 <- cbind(data_imputed_numeric,
                              data_imputed_factor,
                              "impStatus" = as.factor(
                                    data_quant_clean_imputed$impStatus),
                              "Cluster" = as.factor(
                                    data_clusters$Best.partition))
```

```
cluster_summary_numeric <- data_clean_cluster_1 %>% group_by(Cluster) %>%
                           summarise(across(where(is.numeric),
                           list(median = median, sd = sd)))

cluster_summary_factor <- data_clean_cluster_1 %>%
                           select(where(is.factor)) %>%
                           select(-impStatus) %>%
                           select(Cluster, neighbourhood_group) %>%
                           pivot_longer(!Cluster) %>%
                           count(Cluster, name, value, .drop = F, sort = T)
```

Visualizations

Factor Plots

```
plot_neighborhoods <- cluster_summary_factor %>%
           ggplot(aes(x = value, y = n, fill = value)) +
           geom_bar(stat = 'identity') +
           facet_wrap(~Cluster, scales = "free_x") +
           labs(x = "", y = "Frequency",
               fill = "neighbourhood_group") +
           theme_bw() +
           theme(axis.text.x = element_blank(),
                 axis.ticks = element_blank())

save(plot_neighborhoods, file = "Report Figures/plot_neighborhoods.rds")
```

```
# Plot of room type frequency by cluster.
cluster_summary_type<- data_clean_cluster_1 %>%
                           select(where(is.factor)) %>%
                           select(-impStatus) %>%
                           select(Cluster, room_type) %>%
                           pivot_longer(!Cluster) %>%
                           count(Cluster, name, value, .drop = F, sort = T)

plot_type <- cluster_summary_type %>%
           ggplot(aes(x = value, y = n, fill = value)) +
```

```
            geom_bar(stat = 'identity') +
            facet_wrap(~Cluster, scales = "free_x") +
            labs(x = "", y = "Frequency",
                fill = "room_type") +
            theme_bw() +
            theme(axis.text.x = element_blank(),
                  axis.ticks = element_blank())

save(plot_type, file = "Report Figures/plot_type.rds")
```

```
# Plot last_review_year frequency by cluster.
cluster_summary_year <- data_clean_cluster_1 %>%
                          select(where(is.factor)) %>%
                          select(-impStatus) %>%
                          select(Cluster, last_review_year) %>%
                          pivot_longer(!Cluster) %>%
                          count(Cluster, name, value, .drop = F, sort = T)

plot_year <- cluster_summary_year %>%
            ggplot(aes(x = value, y = n, fill = value)) +
            geom_bar(stat = 'identity') +
            facet_wrap(~Cluster, scales = "free_x") +
            labs(x = "", y = "Frequency",
                fill = "last_review_year") +
            theme_bw() +
            theme(axis.text.x = element_blank(),
                  axis.ticks = element_blank())

save(plot_year, file = "Report Figures/plot_year.rds")
```

Numeric Plots

```
remove_skew <- recipe(~., x = data_imputed_numeric) %>%
               step_normalize(all_predictors()) %>%
               step_YeoJohnson(all_predictors()) %>%
               prep(retain = T)

data_noskew_cluster_1 <- bake(remove_skew, new_data = NULL)

data_noskew_cluster_1 <- cbind(data_noskew_cluster_1,
                               data_imputed_factor,
                               "impStatus" = as.factor(
                                     data_quant_clean_imputed$impStatus),
                               "Cluster" = as.factor(
                                     data_clusters$Best.partition))

numeric_plot_noskew_1 <- data_noskew_cluster_1 %>%
                select(price, number_of_reviews,
                         reviews_per_month,
                         calculated_host_listings_count,
                         availability_365, Cluster, minimum_nights) %>%
```

```
                rename(host_listings_count = calculated_host_listings_count) %>%
                pivot_longer(!Cluster) %>%
                ggplot(aes(y = value, x = Cluster, fill = Cluster)) +
                geom_boxplot() +
                facet_wrap(~name, scale = "free_y") +
                labs(x = "", y = "Standard Deviations from the Grand Group Mean") +
                theme_bw() +
                theme(axis.text.x = element_blank(),
                    axis.ticks = element_blank())

save(numeric_plot_noskew_1, file = "Report Figures/numeric_plot_noskew_1.rds")
```

Map Plot

```
#NYC_Map <- get_stamenmap(
    #bbox = c(left = -74.3, bottom = 40.5, right = -73.6, top = 40.95),
    #maptype = "toner-background", color = "bw")
```

```
#save(NYC_Map, file = "Report Figures/NYC_Map.rds")
load("Report Figures/NYC_Map.rds")
```

```
df_neighbourhood_labels <- data.frame(
    longitude = c(-73.8648, -73.9442, -73.96, -73.7949, -74.1502),
    latitude = c(40.8448, 40.6782, 40.7831, 40.7282, 40.5795),
    borough = c("Bronx", "Brooklyn", "Manhattan", "Queens", "Staten Island")
)
```

```
map_plot <- ggmap(NYC_Map) + geom_point(data = data_clean_cluster_1,
                             mapping = aes(x = longitude, y = latitude,
                             col = Cluster, size = price,),
                             alpha = .8) +
                  geom_label(data = df_neighbourhood_labels,
                             aes(x = longitude, y = latitude,
                             label = borough), label.size = 0.5) +
                  labs(x = "", y = "") +
                  theme_bw() +
                  theme(axis.text.x = element_blank(),
                        axis.text.y = element_blank(),
                        axis.ticks = element_blank())

save(map_plot, file = "Report Figures/map_plot.rds")
```