Exploratory Data Analysis

Art Tay

Report

Appendix - Code

```
# Libraries
library(VIM)
library(mice)
library(naniar)
library(tidyverse)
library(fastDummies)
library(NbClust)
library(kernlab)
```

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

Data Cleaning

```
# Data cleaning

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

Missing Data

```
# 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 %>%
```

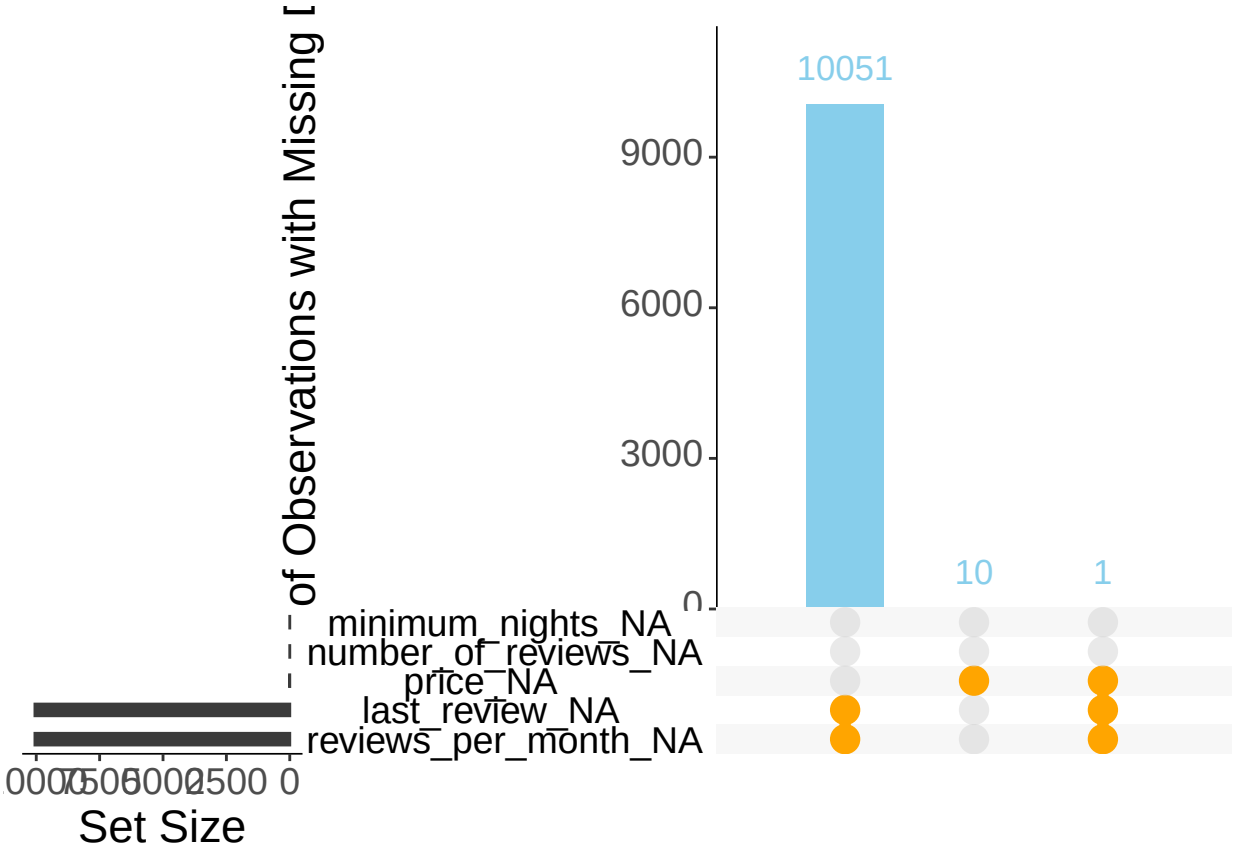
```
    mutate(price = ifelse(price == 0, NA, price)) %>%
    mutate(latitude = ifelse(latitude == 0, NA, latitude)) %>%
    mutate(longitude = ifelse(longitude == 0, NA, longitude)) %>%
    mutate(minimum_nights = ifelse(minimum_nights == 0, NA, minimum_nights)) %>%
    # It was done using replace to 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))

# data_quant_mis <- as.data.frame(data_quant_mis)

#colnames(data_quant_mis) <- colnames(data_quant)
str(data_quant_mis)
```

```
## 'data.frame':    48895 obs. of  12 variables:
##  $ neighbourhood_group           : Factor w/ 5 levels "Bronx","Brooklyn",..: 2 3 3 2 3 3 2 3 3 3 ...
##  $ neighbourhood                 : Factor w/ 221 levels "Allerton","Arden Heights",..: 109 128 95 42
##  $ latitude                      : num  40.6 40.8 40.8 40.7 40.8 ...
##  $ longitude                     : num  -74 -74 -73.9 -74 -73.9 ...
##  $ room_type                     : Factor w/ 3 levels "Entire home/apt",..: 2 1 2 1 1 1 2 2 2 1 ...
##  $ price                         : int  149 225 150 89 80 200 60 79 79 150 ...
##  $ minimum_nights                : int  1 1 3 1 10 3 45 2 2 1 ...
##  $ number_of_reviews             : int  9 45 0 270 9 74 49 430 118 160 ...
##  $ last_review                   : Factor w/ 1765 levels "","2011-03-28",..: 1503 1717 NA 1762 1534 1
##  $ reviews_per_month             : num  0.21 0.38 NA 4.64 0.1 0.59 0.4 3.47 0.99 1.33 ...
##  $ calculated_host_listings_count: int  6 2 1 1 1 1 1 1 1 4 ...
##  $ availability_365              : int  365 355 365 194 0 129 0 220 0 188 ...
```

```
# Plot the pattern of missing values
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")
```




```
# Check the subset of points with missing prices.
mis_price <- data_full %>% filter(price == 0)

# Only 3 are missing availability.
# Most in Brooklyn, but no clear relationship.
# Seem to be MAR.
# Imputation seems valid.

# Check if all last review and number review missing together.
sum(is.na(data_quant_mis$last_review) &
    !is.na(data_quant_mis$reviews_per_month))
```

```
## [1] 0
```

```
mis_reviews <- data_full %>% filter(is.na(data_quant_mis$last_review)) %>%
                select(reviews_per_month, last_review, number_of_reviews)

# There two values are always missing together.
# Check if total review is zero also.
sum(mis_reviews$number_of_reviews > 0)
```

```
## [1] 0
```

```r
# Number of Review, Reviews per month, and last review date are all missing
# together.
```

Feature Engineering

```r
# 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))
# str(data_quant_fe)
```

```r
# 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]
}

# Tests
#means_2 <- apply(data_quant_mis_numeric, MARGIN = 2, FUN = mean, na.rm = T)
#sds_2 <- apply(data_quant_mis_numeric, MARGIN = 2, FUN = sd, na.rm = T)

# Back transformation code.
#for(i in 1:length(means)){
    #data_quant_mis_numeric[, i] <-
        #data_quant_mis_numeric[, i] * sds[i] + means[i]
#}
```

```r
# 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

# Check 11 imputed values
#sum(data_quant_clean_imputed$impStatus)
```

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")
```

```
# Dummy variables (one-hot-encoding)
factor_all <- data_quant_fe %>% select(where(is.factor))

factor_all <- dummy_cols(factor_all,
                         remove_selected_columns = T)
```

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

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

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

# get the original columns
data_clean_imputed_all <- data_clean_imputed_all[, 1:247]
data_clean_imputed_all$impStatus <- impStatus

# Check 11 imputed values
sum(data_clean_imputed_all$impStatus)
```

```
## [1] 11
```

```
#data_clusters_2 <- NbClust(data = data_clean_imputed_all,
                           #distance = "euclidean",
                           #method = "ward.D",
                           #index = "silhouette")
```

```
#save(data_clusters_2, file = "data_clusters_2.rds")
load(file = "data_clusters_2.rds")
```

Kernel Based Clustering

```
#rbf <- rbfdot(sigma = 0.05)
#kernel_distances <- kernelMatrix(
    #kernel = rbf,
    #x = as.matrix(data_quant_clean_imputed))
```

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

```
#kernel_clusters <- NbClust(data = data_quant_clean_imputed,
                           #diss = kernel_distances,
                           #distance = NULL,
                           #method = "ward.D",
                           #min.nc = 2,
                           #max.nc = 20,
                           #index = "silhouette")
```

Investigation of Clusters

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))
```

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

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

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

Cluster 2 cleaning

```r
# Back Transform numeric data.
# Extract numeric columns.
data_imputed_numeric_all <- data_clean_imputed_all %>%
                        select(1:8)

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

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

```r
data_clean_cluster_2 <- cbind(data_imputed_numeric_all,
                              data_imputed_factor,
                              "impStatus" = data_clean_imputed_all$impStatus,
                              "Cluster" = data_clusters_2$Best.partition)
```

Visualizations

```r
# boxplot by neighborhood
plot_x <- data_quant %>%
    ggplot(aes(x = neighbourhood_group, y = log(price))) +
    geom_boxplot()
```

Cluster Statistics

Clustering 1

```r
plot_prices_c1 <- data_clean_cluster_1 %>%
                  select(price, Cluster) %>%
                  ggplot(aes(price)) +
                  geom_histogram() +
                  facet_wrap(~Cluster)
```

Check Skewness

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