EDA of Airbnb Paris

Junwei Chen

Table of contents

Loading the necessary library

Libraries like Janitor (Firke 2023), Knitr (Xie 2014), Lubridate (Grolemund and Wickham 2011), Naniar (Tierney and Cook 2023) and Tidyverse (Wickham et al. 2019) have been used to conduct the EDA on the Airbnb data for Paris.

```
library(janitor)
library(knitr)
library(lubridate)
library(naniar)
library(tidyverse)
```

Loading the data

Using the link of the data-set and storing it in data-frame named "airbnb_data"

```
url <-
  paste0(
    "http://data.insideairbnb.com/france/ile-de-france/paris/2023-12-12/data/listings.csv.gz
  )

airbnb_data <-
  read_csv(
    file = url,
    guess_max = 20000
  )

# getting the dataset in csv format
write_csv(airbnb_data, "airbnb_data.csv")


airbnb_data
```

```
# A tibble: 74,329 x 75
      id listing_url             scrape_id last_scraped source name  description
   <dbl> <chr>                       <dbl> <date>       <chr>  <chr> <lgl>
 1  3109 https://www.airbnb.com~   2.02e13 2023-12-12   city ~ Rent~ NA
 2  5396 https://www.airbnb.com~   2.02e13 2023-12-14   city ~ Rent~ NA
 3 81106 https://www.airbnb.com~   2.02e13 2023-12-13   city ~ Rent~ NA
 4  7397 https://www.airbnb.com~   2.02e13 2023-12-13   city ~ Rent~ NA
 5  7964 https://www.airbnb.com~   2.02e13 2023-12-12   city ~ Rent~ NA
 6 81615 https://www.airbnb.com~   2.02e13 2023-12-13   city ~ Rent~ NA
 7  9359 https://www.airbnb.com~   2.02e13 2023-12-13   city ~ Rent~ NA
 8 81870 https://www.airbnb.com~   2.02e13 2023-12-13   city ~ Rent~ NA
 9  9952 https://www.airbnb.com~   2.02e13 2023-12-14   city ~ Rent~ NA
10 86053 https://www.airbnb.com~   2.02e13 2023-12-13   city ~ Rent~ NA
# i 74,319 more rows
# i 68 more variables: neighborhood_overview <chr>, picture_url <chr>,
#   host_id <dbl>, host_url <chr>, host_name <chr>, host_since <date>,
#   host_location <chr>, host_about <chr>, host_response_time <chr>,
#   host_response_rate <chr>, host_acceptance_rate <chr>,
#   host_is_superhost <lgl>, host_thumbnail_url <chr>, host_picture_url <chr>,
#   host_neighbourhood <chr>, host_listings_count <dbl>, ...
```

Sub-setting the data

```
# Choosing relevant fields from the dataset
airbnb_data_selected <-
  airbnb_data |>
  select(
    host_id,
    host_response_time,
    host_is_superhost,
    host_total_listings_count,
    neighbourhood_cleansed,
    bathrooms,
    bedrooms,
    price,
    number_of_reviews,
    review_scores_rating,
    review_scores_accuracy,
    review_scores_value
  )
```

```
# Producing frist few contents from the price column
airbnb_data_selected$price |>
  head()
```

```
[1] "$150.00" "$146.00" "$110.00" "$140.00" "$180.00" "$71.00"
```

```
# Selecting only those prices which has ',' (comma) i.e. more than 3 digits
airbnb_data_selected |>
  select(price) |>
  filter(str_detect(price, ","))
```

```
# A tibble: 1,550 x 1
   price
   <chr>
 1 $1,200.00
 2 $8,000.00
 3 $7,000.00
 4 $1,997.00
 5 $1,000.00
 6 $1,286.00
```

```
 7 $2,300.00
 8 $1,500.00
 9 $1,200.00
10 $1,357.00
# i 1,540 more rows
```

```
# Converting all data points of price column into integer
airbnb_data_selected <-
  airbnb_data_selected |>
  mutate(
    price = str_remove_all(price, "[\\$,]"),
    price = as.integer(price)
  )
```

Visual EDA

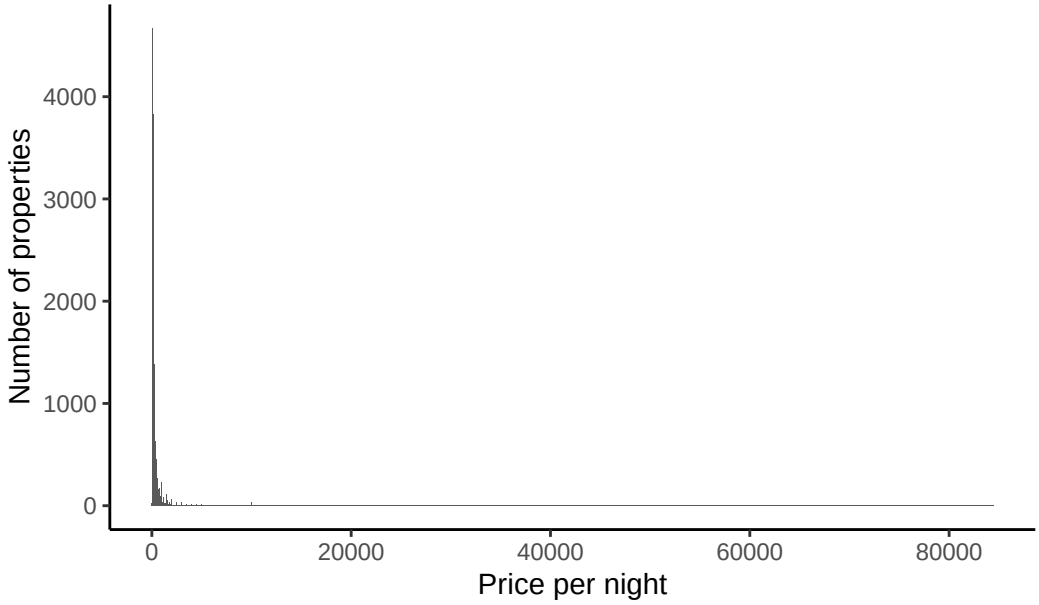
Based on Price

Figure 1 shows the number of properties across differerent ranges. It shows a large concentration of properties at the lower end of the price scale. It shows a sharp decline as the price increases, which is indicative that most of the properties are priced at the lower end of the range. However, the exact distribution is difficult to decipher because of the scaling issues.

Figure 2 shows the histogram similar to the Figure 1 but with an added scaling on logarithmic scale. It filters out properties priced at 1000 units or less. Logarithmic transformation or scaling is a general technique used when data spans have variable orders of magnitude. This allows for a visual representation where the distribution of higher priced properties are more clearly visible. The scaling helped in dealing with the skewness of the data towards lower prices and provided a better visualization of the distribution.

```
# Plotting the histogram for price
airbnb_data_selected |>
  ggplot(aes(x = price)) +
  geom_histogram(binwidth = 10) +
  theme_classic() +
  labs(title = "Figure 1: Price per night",
    x = "Price per night",
    y = "Number of properties"
  )
```

Figure 1: Price per night




```
# Plotting histogram by scaling the data points on log scale
airbnb_data_selected |>
  filter(price > 1000) |>
  ggplot(aes(x = price)) +
  geom_histogram(binwidth = 10) +
  theme_classic() +
  labs(title = "Figure 2: Price per night after scaling on log",
    x = "Price per night",
    y = "Number of properties"
  ) +
  scale_y_log10()
```

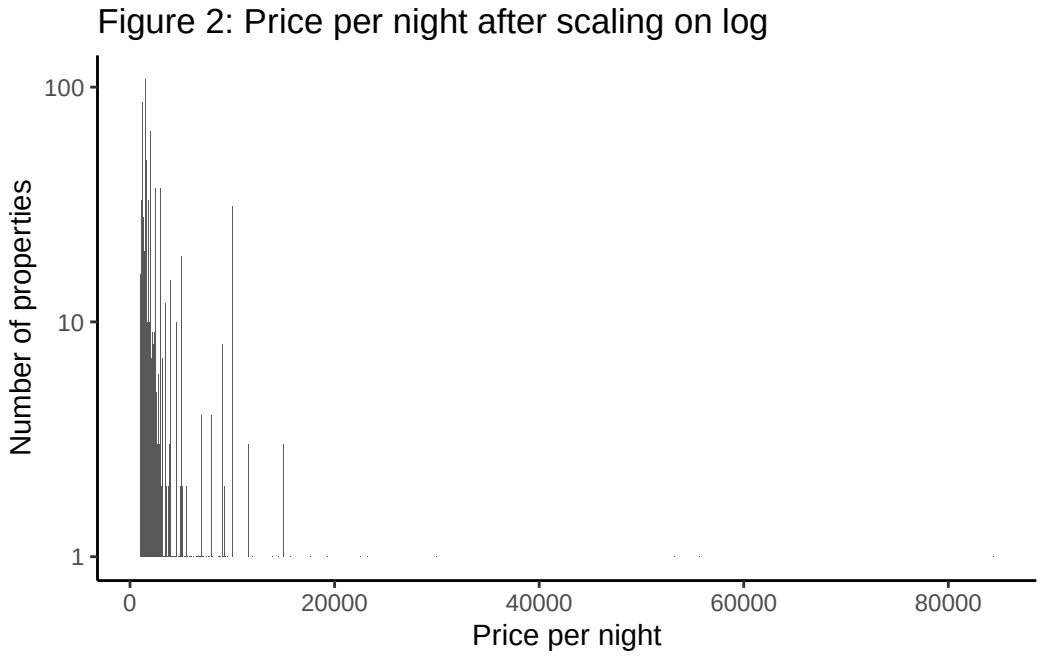
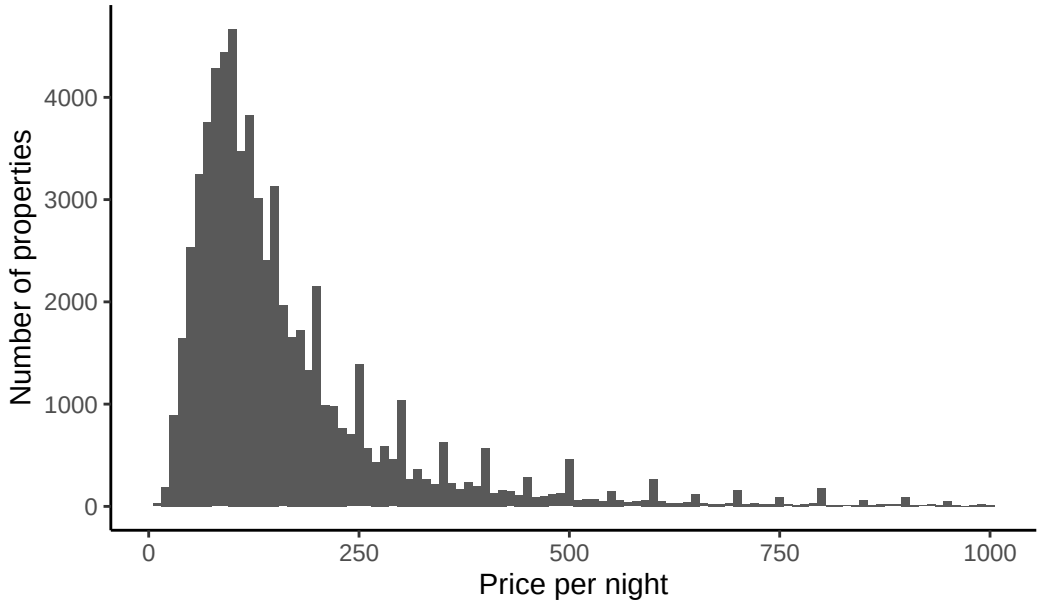


Figure 3 shows a histogram histogram for the property price distribution less than $1000. It can be seen than the number of properties decreases with the increase in price suggesting that the properties in lower range of price are in more abundance. Figure 4 shows histogram of the filtered data for fetching results between $90 and $210 per night. This filter helps narrow down the search results of the lower spectrum as seen in Figure 1. It zooms in on a specific range to provide a detailed view of pricing patterns.

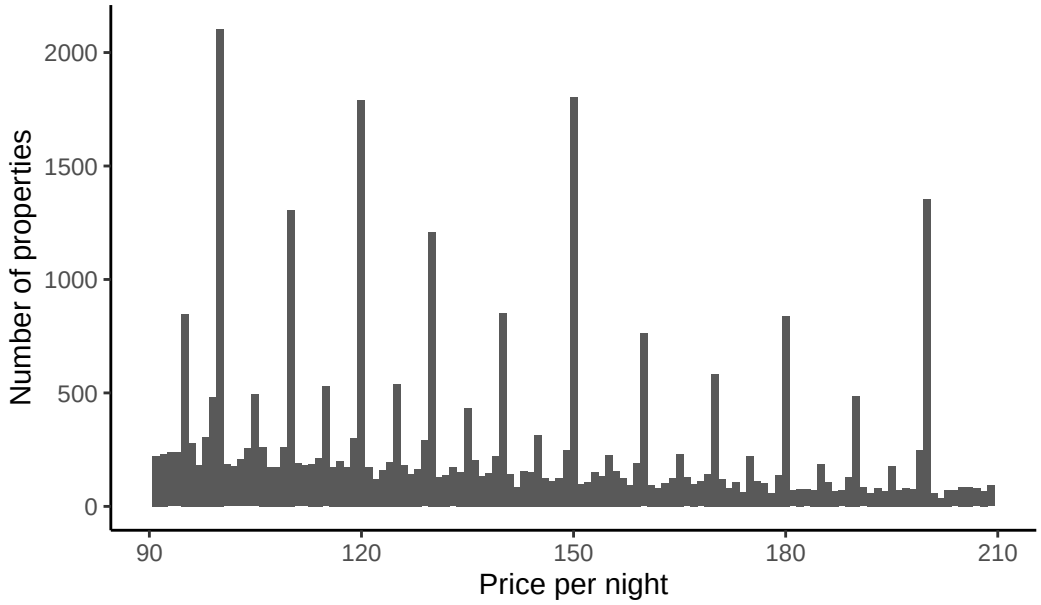
```
airbnb_data_selected |>
  filter(price < 1000) |>
  ggplot(aes(x = price)) +
  geom_histogram(binwidth = 10) +
  theme_classic() +
  labs(title = "Figure 3: Histogram for prices less than $1000",
    x = "Price per night",
    y = "Number of properties"
  )
```

Figure 3: Histogram for prices less than $1000




```
airbnb_data_selected |>
  filter(price > 90) |>
  filter(price < 210) |>
  ggplot(aes(x = price)) +
  geom_histogram(binwidth = 1) +
  theme_classic() +
  labs(title = "Figure 4: Histogram for prices between $90 and $210",
    x = "Price per night",
    y = "Number of properties"
  )
```

Figure 4: Histogram for prices between $90 and $210



Based on Scores

```
# Filtering for less than 1000
airbnb_data_less_1000 <-
  airbnb_data_selected |>
  filter(price < 1000)
```

```
airbnb_data_less_1000 |>
  filter(is.na(host_is_superhost))
```

```
# A tibble: 83 x 12
    host_id host_response_time host_is_superhost host_total_listings_count
      <dbl> <chr>              <lgl>                                  <dbl>
 1 29138344 within an hour     NA                                         3
 2  5869840 within a few hours NA                                         7
 3 35125972 within an hour     NA                                         3
 4 13827149 within a few hours NA                                         3
 5 62919059 within a few hours NA                                         3
 6 22167607 N/A                NA                                         2
 7 10259782 N/A                NA                                         2
 8 62919059 within a few hours NA                                         3
```

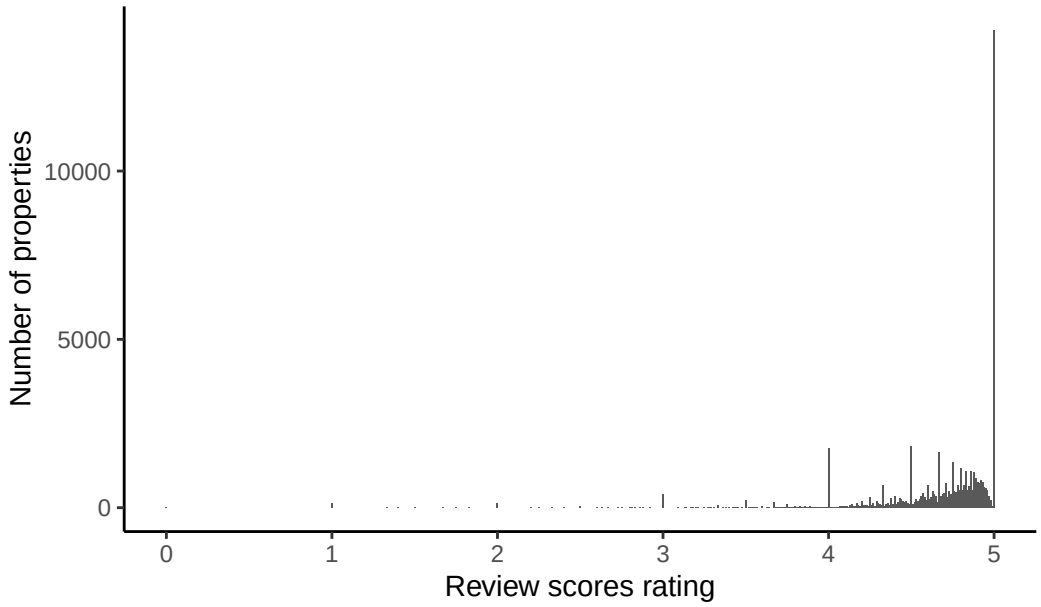
```
 9 20056470 N/A                NA                                          4
10 20056470 N/A                NA                                          4
# i 73 more rows
# i 8 more variables: neighbourhood_cleansed <chr>, bathrooms <lgl>,
#   bedrooms <dbl>, price <int>, number_of_reviews <dbl>,
#   review_scores_rating <dbl>, review_scores_accuracy <dbl>,
#   review_scores_value <dbl>
```

```
# Adding additional column to add information for hosts
airbnb_data_no_superhost_nas <-
  airbnb_data_less_1000 |>
  filter(!is.na(host_is_superhost)) |>
  mutate(
    host_is_superhost_binary =
      as.numeric(host_is_superhost)
  )
```

Figure 5 suggests that there are high frequency of properties with a score of around 5, indicating that the customers are generally satisfied. This shows that properties in Paris are generally built well and leaves a satisfactory impression on the customers.

```
# Creating bar plot to show the distibution of review scores
airbnb_data_no_superhost_nas |>
  ggplot(aes(x = review_scores_rating)) +
  geom_bar() +
  theme_classic() +
  labs(title = "Figure 5: Distribution of scores",
    x = "Review scores rating",
    y = "Number of properties"
  )
```

Figure 5: Distribution of scores



It can be seen that there are 13,497 properties in the data-set that do not have a super-host status and indicates that there are missing review score ratings. This can be common for new listings or for those listings which have been dormant or inactive for a long time. It is important to note that these missing reviews are not the case of missing data, but a case where certain properties have not been reviewed by any customer. These large number of listings without an review can be a matter of concern for Airbnb Paris as it indicates that large number of properties have either been iactive or not famous enough to get a review.

```
# Calculating number of rows which has missing score ratings
airbnb_data_no_superhost_nas |>
  filter(is.na(review_scores_rating)) |>
  nrow()
```

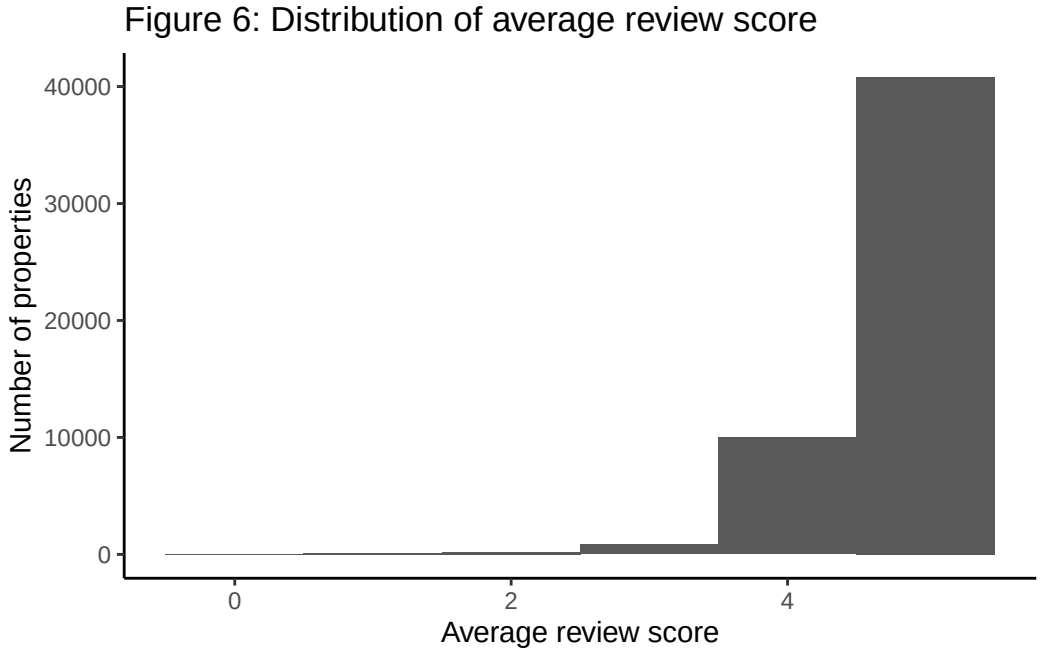
```
[1] 13497
```

```
# Creating a table to show frequency of each unique value
airbnb_data_no_superhost_nas |>
  filter(is.na(review_scores_rating)) |>
  select(number_of_reviews) |>
  table()
```

```
number_of_reviews
    0
13497
```

Figure 6 shows the histogram for the average review scores with a bin width of 1. The plot can be seen to have the majority of the scores at 4 or higher. This suggests that the properties generally have a satisfactory and favorable reviews.

```
# Plotting histogram plot showing the distribution of review scores
airbnb_data_no_superhost_nas |>
  filter(!is.na(review_scores_rating)) |>
  ggplot(aes(x = review_scores_rating)) +
  geom_histogram(binwidth = 1) +
  theme_classic() +
  labs(title = "Figure 6: Distribution of average review score ",
    x = "Average review score",
    y = "Number of properties"
  )
```




```
# Filtering where ratings are available
airbnb_data_has_reviews <-
  airbnb_data_no_superhost_nas |>
  filter(!is.na(review_scores_rating))
```

The majority of hosts respond within an hour, amounting to 22,094, while there are notable number of listings where the response time is not available, amounting to 16,531.

```r
# Creating table for different response time counts
airbnb_data_has_reviews |>
  count(host_response_time)
```

```
# A tibble: 6 x 2
  host_response_time     n
  <chr>              <int>
1 N/A                16531
2 a few days or more  1243
3 within a day        5297
4 within a few hours  6811
5 within an hour     22094
6 <NA>                   2
```

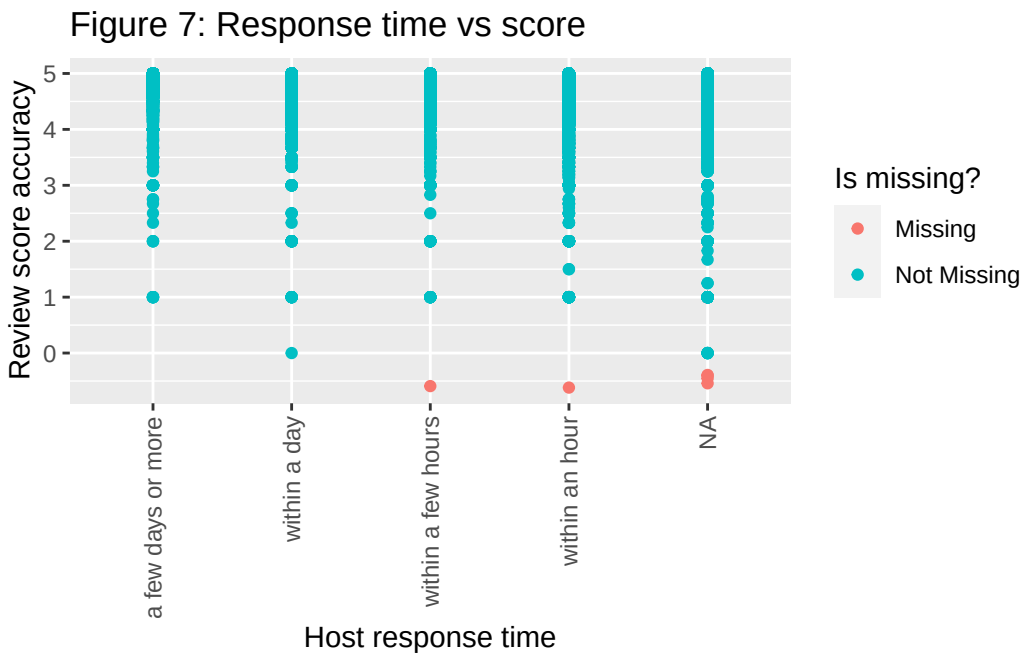
```r
# Updating the host response time in main data frame
airbnb_data_has_reviews <-
  airbnb_data_has_reviews |>
  mutate(
    host_response_time = if_else(
      host_response_time == "N/A",
      NA_character_,
      host_response_time
    ),
    host_response_time = factor(host_response_time)
  )
```

Based on Review Scores and Response Time

Figure 7 illustrates the relationship between the response of a host's time and the properties review score. Most of the data points in the scatter plot consists of an available review score indicated by blue color. The plot shows that most of the hosts respond in a few hours to max within a day. It also shows that most properties are having high review score.

```r
# Plot to show the relationship between response time of hosts and accuracy scores
airbnb_data_has_reviews |>
  ggplot(aes(
    x = host_response_time,
    y = review_scores_accuracy
  )) +
  geom_miss_point() +
```

```
  labs(title = "Figure 7: Response time vs score",
    x = "Host response time",
    y = "Review score accuracy",
    color = "Is missing?"
  ) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```




```
# Filtering data and showing only thsoe data points where host repsponse time is present
airbnb_data_selected <-
  airbnb_data_has_reviews |>
  filter(!is.na(host_response_time))
```

```
# Filtering the data based on listong counts more than 500
airbnb_data_selected |>
  filter(host_total_listings_count >= 500) |>
  head()
```

```
# A tibble: 6 x 13
   host_id host_response_time host_is_superhost host_total_listings_count
     <dbl> <fct>              <lgl>                                 <dbl>
1 50502817 within an hour     FALSE                                   778
2 50502817 within an hour     FALSE                                   778
```

```
3 50502817 within an hour     FALSE                                   778
4 50502817 within an hour     FALSE                                   778
5 50502817 within an hour     FALSE                                   778
6 50502817 within an hour     FALSE                                   778
# i 9 more variables: neighbourhood_cleansed <chr>, bathrooms <lgl>,
#   bedrooms <dbl>, price <int>, number_of_reviews <dbl>,
#   review_scores_rating <dbl>, review_scores_accuracy <dbl>,
#   review_scores_value <dbl>, host_is_superhost_binary <dbl>
```

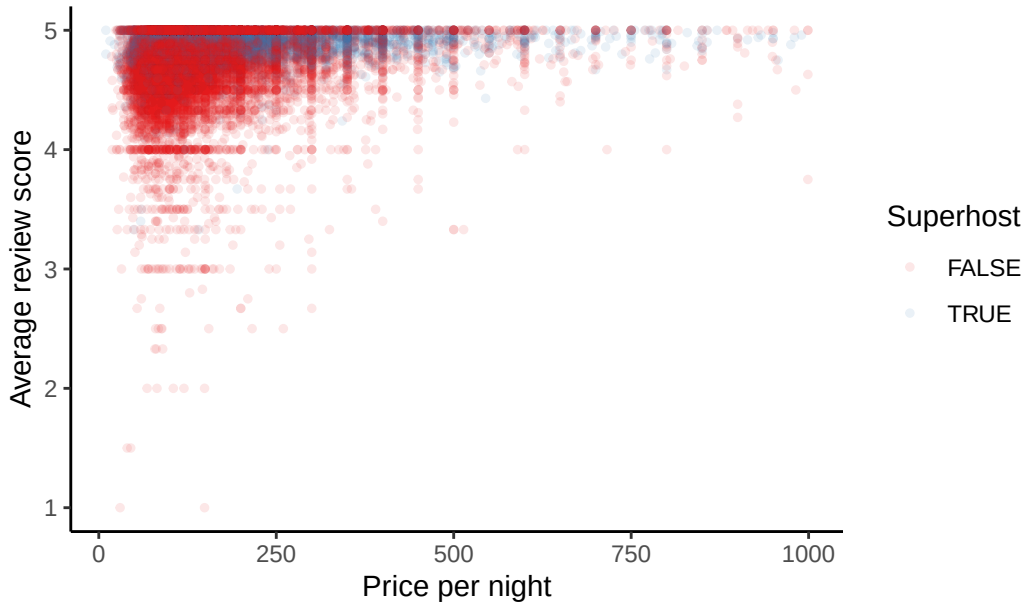
Based on Price and Review Score

```
# Filtering data where hist id occurs exactly once
airbnb_data_selected <-
  airbnb_data_selected |>
  add_count(host_id) |>
  filter(n == 1) |>
  select(-n)
#
```

Figure 8 shows a scatter plot encompassing the relationship between average score and price per night. The super host listings are seen to be clustered generally at the higher end of the review score range. This indicates that the super hosts are generally receiving higher scores. The concentration of points at lower prices are very dense, indicating more availability of affordable listings with a wide range of review scores.

```
# Plotting the superhosts and the review counts to show relationship between price and averag
airbnb_data_selected |>
  filter(number_of_reviews > 1) |>
  ggplot(aes(x = price, y = review_scores_rating,
             color = host_is_superhost)) +
  geom_point(size = 1, alpha = 0.1) +
  theme_classic() +
  labs(title = "Figure 8: Relationship between price and average score",
    x = "Price per night",
    y = "Average review score",
    color = "Superhost"
  ) +
  scale_color_brewer(palette = "Set1")
```

Figure 8: Relationship between price and average score



About 28% of the listings are comprised of super hosts which is a substantial proportion and can indicate an impression of quality services.

```
# Showing proportions of unique values of superhosts
airbnb_data_selected |>
  count(host_is_superhost) |>
  mutate(
    proportion = n / sum(n),
    proportion = round(proportion, digits = 2)
  )
```

```
# A tibble: 2 x 3
  host_is_superhost     n proportion
  <lgl>             <int>      <dbl>
1 FALSE             15820       0.72
2 TRUE               6227       0.28
```

The neighborhood distribution shows that the listings are maximum in Buttes-Montmartre area with a 12.9% density. It is followed by 10% proportion of Popincourt, and so on.

```
# Showing the percentage of unique occurence of values for neighborhood
airbnb_data_selected |>
  tabyl(neighbourhood_cleansed) |>
```

```
  adorn_pct_formatting() |>
  arrange(-n) |>
  filter(n > 100) |>
  adorn_totals("row") |>
  head()
```

```
neighbourhood_cleansed    n percent
     Buttes-Montmartre 2842   12.9%
            Popincourt 2202   10.0%
              Entrepôt 1713    7.8%
             Vaugirard 1681    7.6%
          Ménilmontant 1438    6.5%
        Buttes-Chaumont 1430    6.5%
```

References

Firke, Sam. 2023. *Janitor: Simple Tools for Examining and Cleaning Dirty Data*. https://CRAN.R-project.org/package=janitor.

Grolemund, Garrett, and Hadley Wickham. 2011. "Dates and Times Made Easy with lubridate." *Journal of Statistical Software* 40 (3): 1–25. https://www.jstatsoft.org/v40/i03/.

Tierney, Nicholas, and Dianne Cook. 2023. "Expanding Tidy Data Principles to Facilitate Missing Data Exploration, Visualization and Assessment of Imputations." *Journal of Statistical Software* 105 (7): 1–31. https://doi.org/10.18637/jss.v105.i07.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.

Xie, Yihui. 2014. "Knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC.