Paris EDA*

Sakura Noskor

March 4, 2024

1 Data

Used Rstudio and R (R Core Team 2022) to create this with the help of arrow(Richardson et al. 2024), lubricate (Grolemund and Wickham 2011), ggplot (Wickham 2016), tidyverse (Wickham et al. 2019), mice (van Buuren and Groothuis-Oudshoorn 2011), modelsummary (Arel-Bundock 2022), and naniar (Tierney and Cook 2023).

2 EDA

First we load the data from the site and save it

```
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
  )

write_csv(airbnb_data, "airbnb_data.csv")
```

*Code and some data from this paper are available at: github repo.

```
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
  )

write_parquet(
  x = airbnb_data_selected,
  sink =
    "2023-12-12-paris-airbnblistings-select_variables.parquet"
  )

rm(airbnb_data)
```

Then we play with the data to see the values. Here we check the values for '$' and clean that up.

```
airbnb_data_selected$price |>
  str_split("") |>
  unlist() |>
  unique()
```

```
 [1] "$" "1" "5" "0" "." "4" "6" "8" "7" "3" "2" "9" NA  ","
```

```
airbnb_data_selected |>
  select(price) |>
  filter(str_detect(price, ","))
```

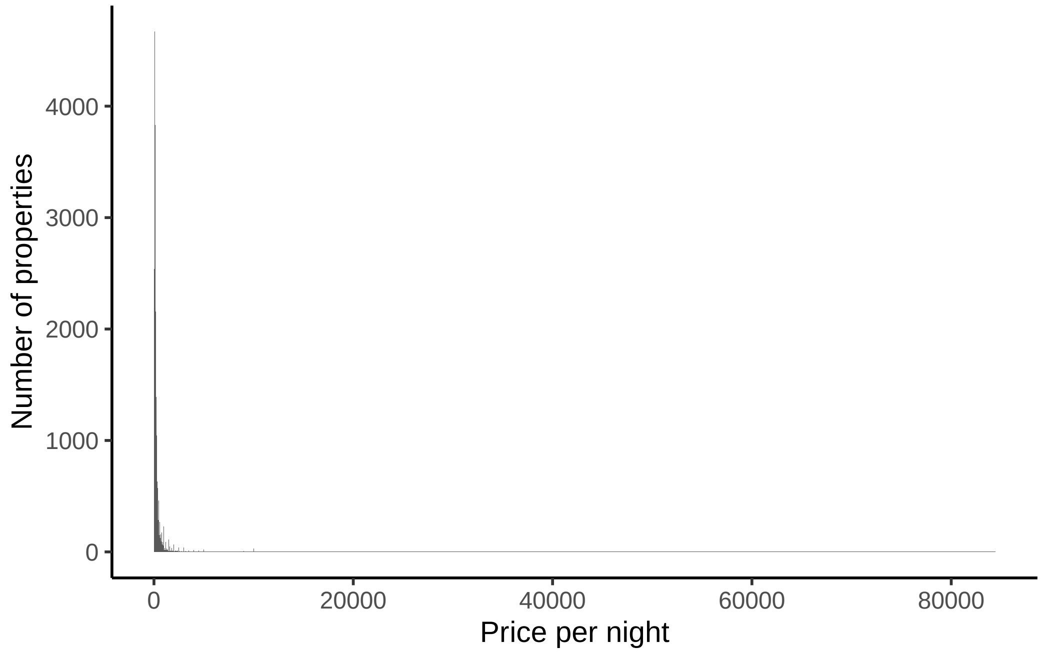
```
# A tibble: 1,550 x 1
```

```
   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
```

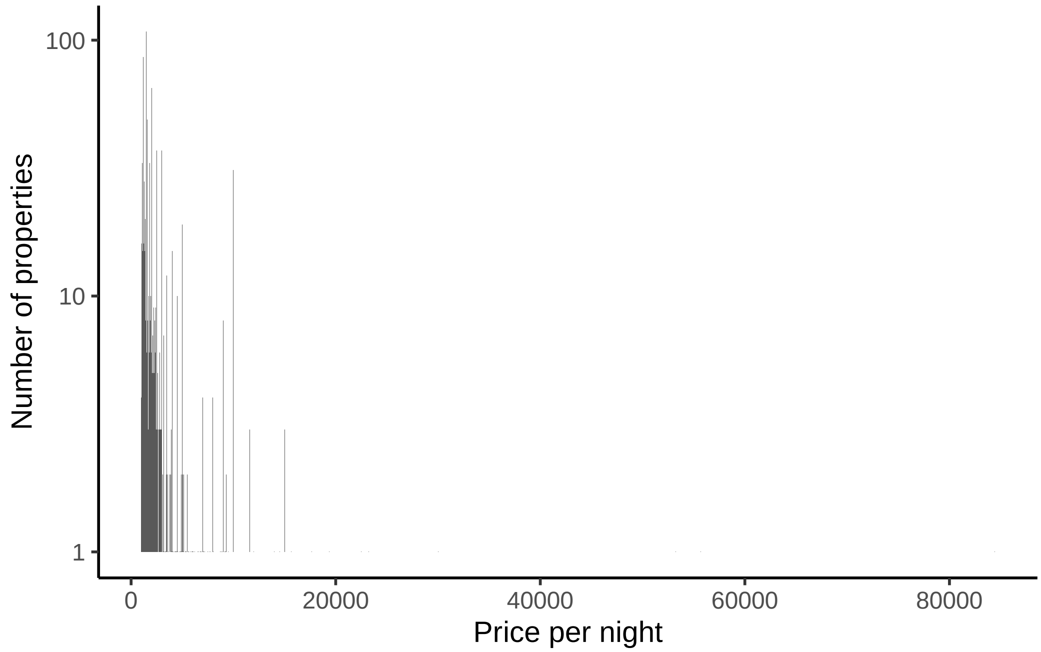
```
airbnb_data_selected <-
  airbnb_data_selected |>
  mutate(
    price = str_remove_all(price, "[\\$,]"),
    price = as.integer(price)
  )
```

Then we graph the distribution

```
airbnb_data_selected |>
  ggplot(aes(x = price)) +
  geom_histogram(binwidth = 10) +
  theme_classic() +
  labs(
    x = "Price per night",
    y = "Number of properties"
  )
```

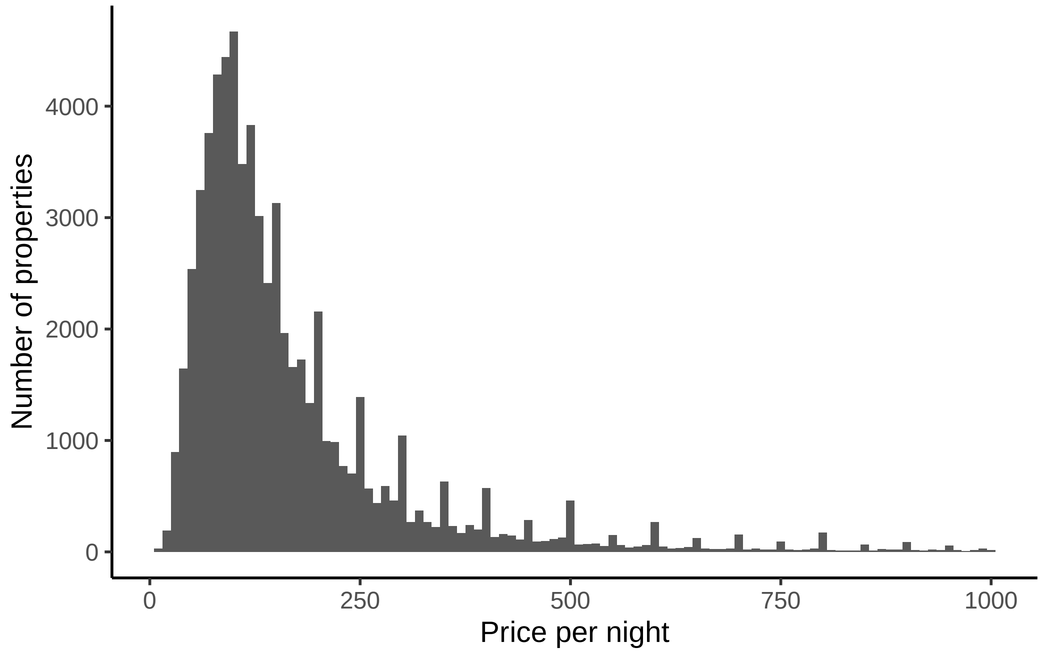



```
airbnb_data_selected |>
  filter(price > 1000) |>
  ggplot(aes(x = price)) +
  geom_histogram(binwidth = 10) +
  theme_classic() +
  labs(
    x = "Price per night",
    y = "Number of properties"
  ) +
  scale_y_log10()
```

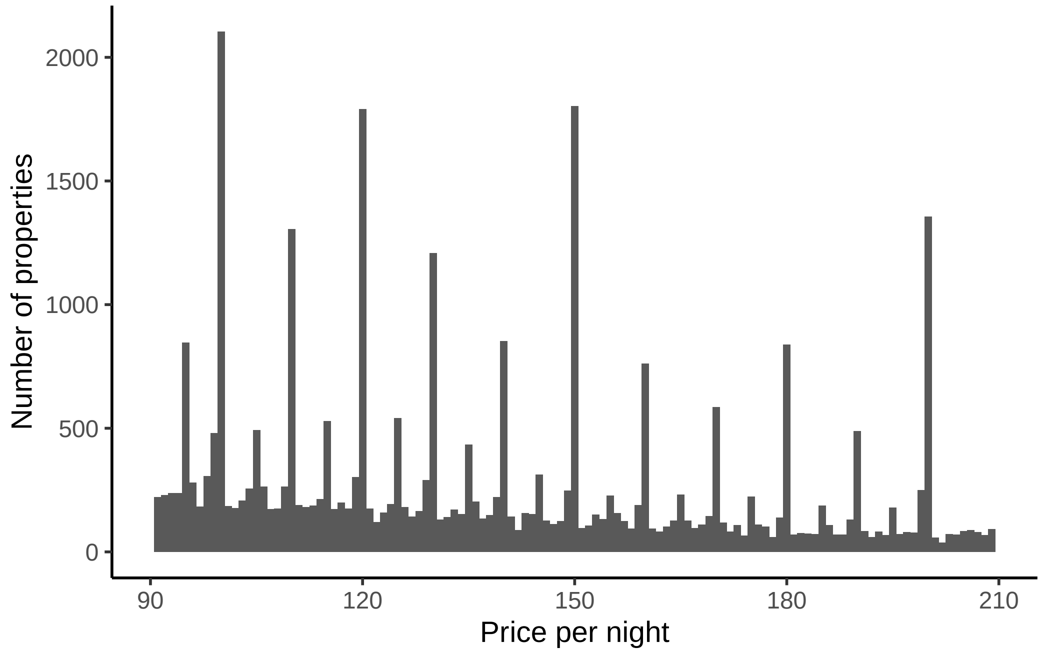


Then we filter the data to see how many properties have a nightly cost of less than $1000 and also between $90 and $20

```
airbnb_data_selected |>
  filter(price < 1000) |>
  ggplot(aes(x = price)) +
  geom_histogram(binwidth = 10) +
  theme_classic() +
  labs(
    x = "Price per night",
    y = "Number of properties"
  )
```




```r
airbnb_data_selected |>
  filter(price > 90) |>
  filter(price < 210) |>
  ggplot(aes(x = price)) +
  geom_histogram(binwidth = 1) +
  theme_classic() +
  labs(
    x = "Price per night",
    y = "Number of properties"
  )
```



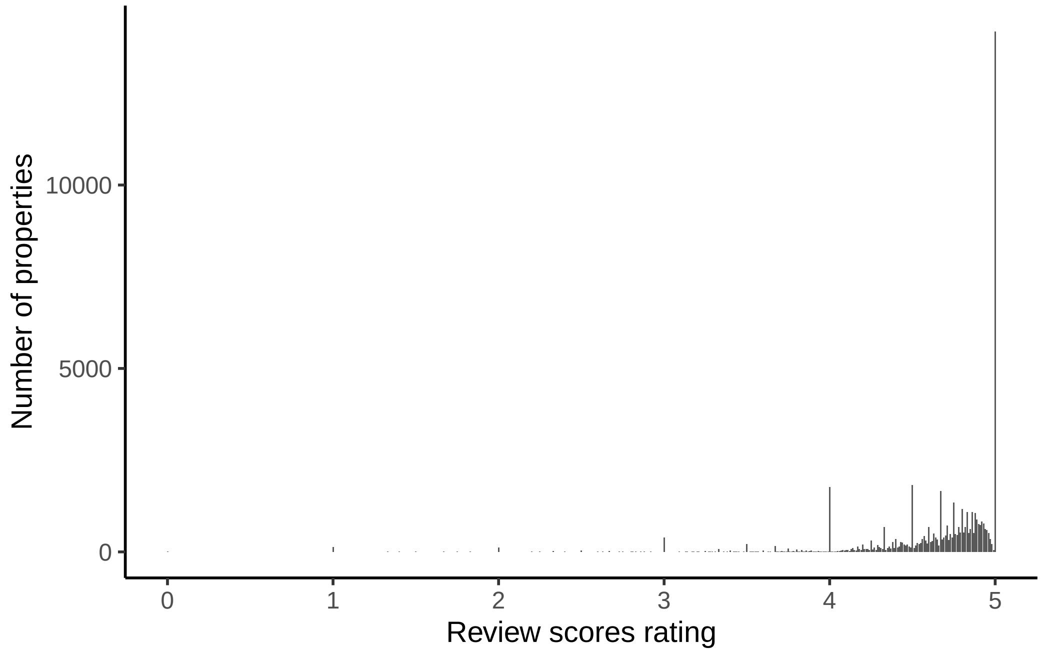
Then we filter for the houses that are less than $1000 and also filter them so there are no NA values in superhosts column and turn those values to binary (0 for false and 1 for true). Then we graph for visual representation.

```
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>
```

```r
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)
  )

airbnb_data_no_superhost_nas |>
  ggplot(aes(x = review_scores_rating)) +
  geom_bar() +
  theme_classic() +
  labs(
    x = "Review scores rating",
    y = "Number of properties"
  )
```



Then we filter to see the how many places has a specific average review score.

```
airbnb_data_no_superhost_nas |>
  filter(is.na(review_scores_rating)) |>
  nrow()
```

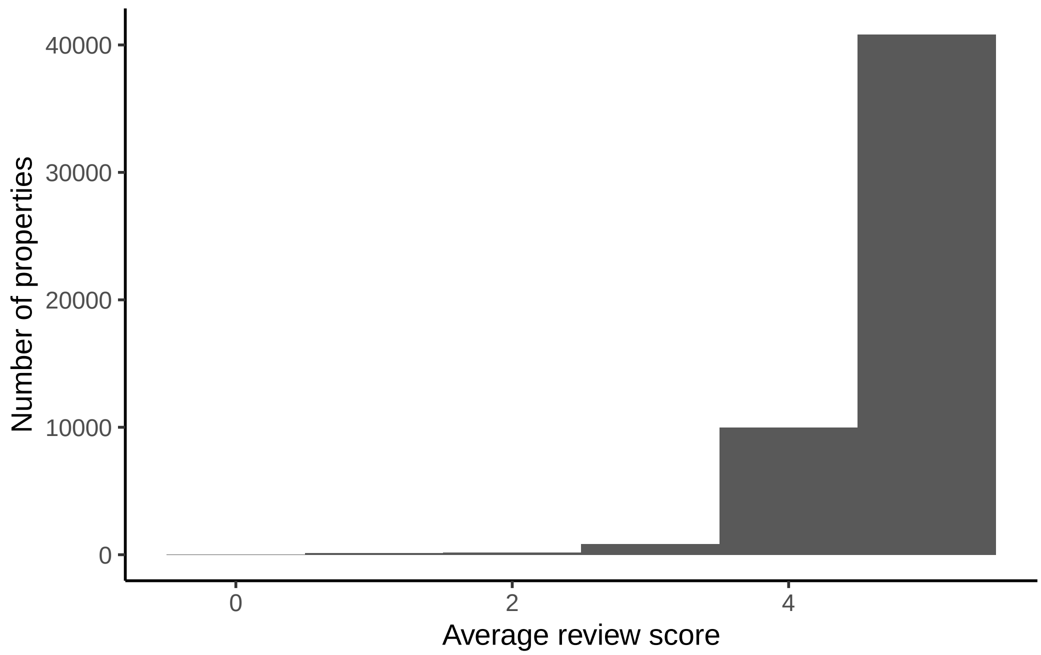
```
[1] 13497
```

```
airbnb_data_no_superhost_nas |>
  filter(is.na(review_scores_rating)) |>
  select(number_of_reviews) |>
  table()
```

```
number_of_reviews
    0
13497
```

```
airbnb_data_no_superhost_nas |>
  filter(!is.na(review_scores_rating)) |>
  ggplot(aes(x = review_scores_rating)) +
```

```
  geom_histogram(binwidth = 1) +
  theme_classic() +
  labs(
    x = "Average review score",
    y = "Number of properties"
  )
```



We see that most number are high in this graph.

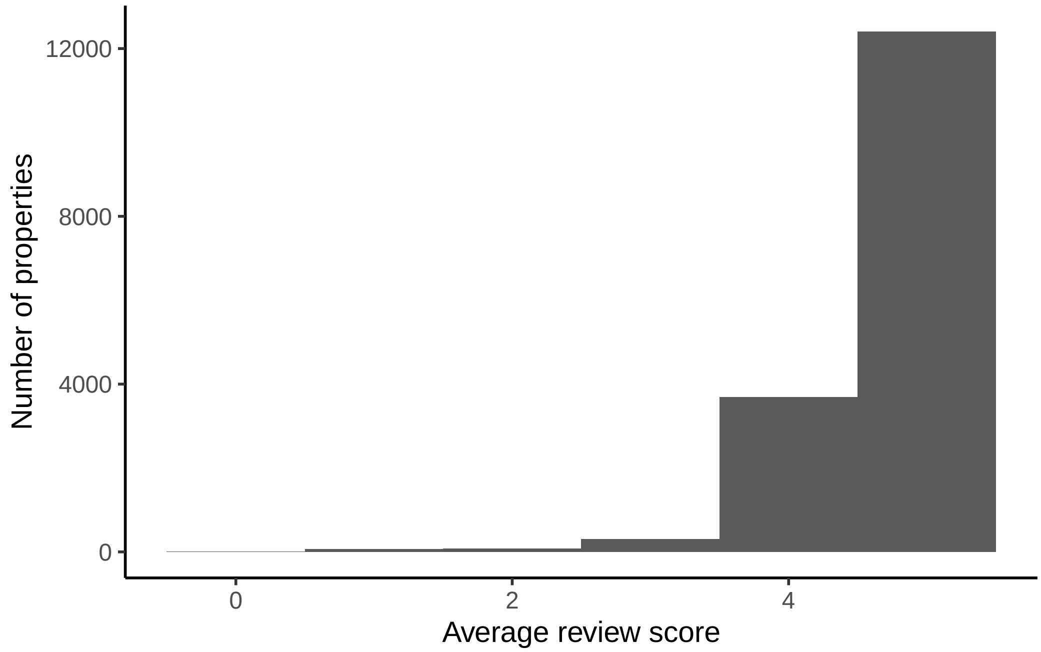
Then we look at response time for the houses that has reviews.

```
airbnb_data_has_reviews <-
  airbnb_data_no_superhost_nas |>
  filter(!is.na(review_scores_rating))


airbnb_data_has_reviews |>
  count(host_response_time)
```

```
# A tibble: 6 x 2
  host_response_time     n
  <chr>              <int>
```

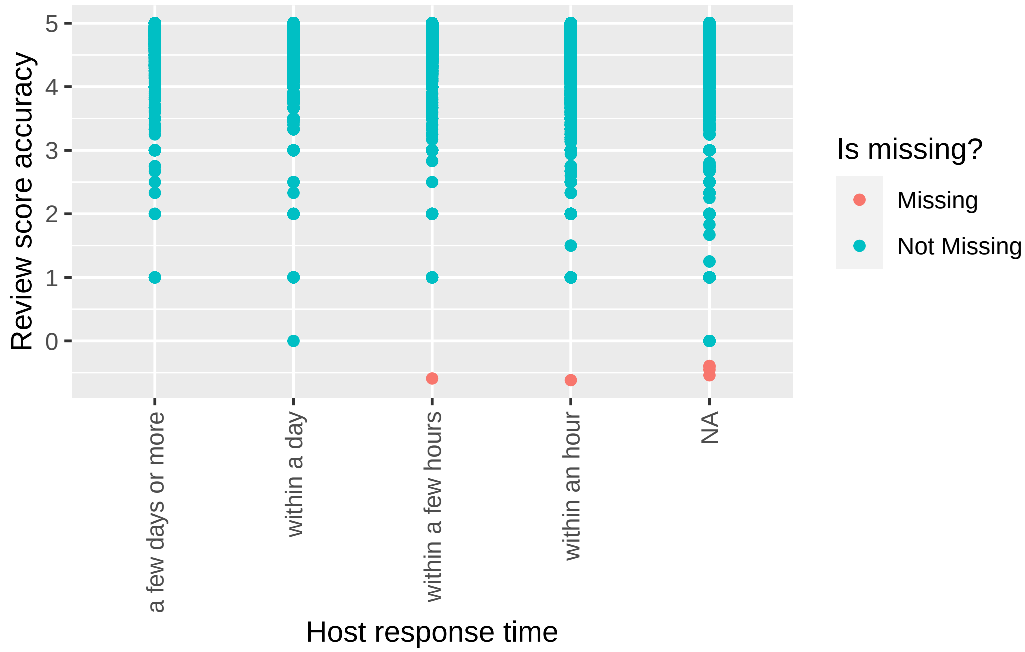
```
1 N/A                16531
2 a few days or more  1243
3 within a day        5297
4 within a few hours  6811
5 within an hour     22094
6 <NA>                   2
```

```r
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)
  )

airbnb_data_has_reviews |>
  filter(is.na(host_response_time)) |>
  ggplot(aes(x = review_scores_rating)) +
  geom_histogram(binwidth = 1) +
  theme_classic() +
  labs(
    x = "Average review score",
    y = "Number of properties"
  )
```



Ggplot drops missing values but we want to include them so..

```
airbnb_data_has_reviews |>
  ggplot(aes(
    x = host_response_time,
    y = review_scores_accuracy
  )) +
  geom_miss_point() +
  labs(
    x = "Host response time",
    y = "Review score accuracy",
    color = "Is missing?"
  ) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```

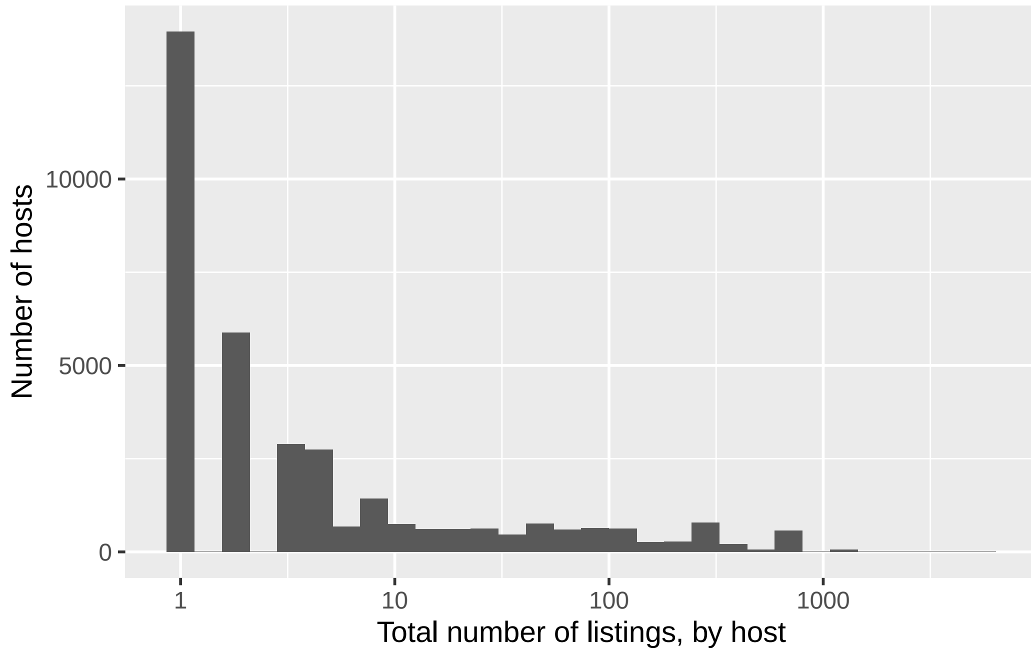


Remove people with NA as their response time

```
airbnb_data_selected <-
  airbnb_data_has_reviews |>
  filter(!is.na(host_response_time))
```

Now number of people that hosted an airbnb in paris

```
airbnb_data_selected |>
  ggplot(aes(x = host_total_listings_count)) +
  geom_histogram() +
  scale_x_log10() +
  labs(
    x = "Total number of listings, by host",
    y = "Number of hosts"
  )
```



The long tail is unusual. Now we clean the NA values

```
airbnb_data_selected |>
  filter(host_total_listings_count >= 500) |>
  head()
```

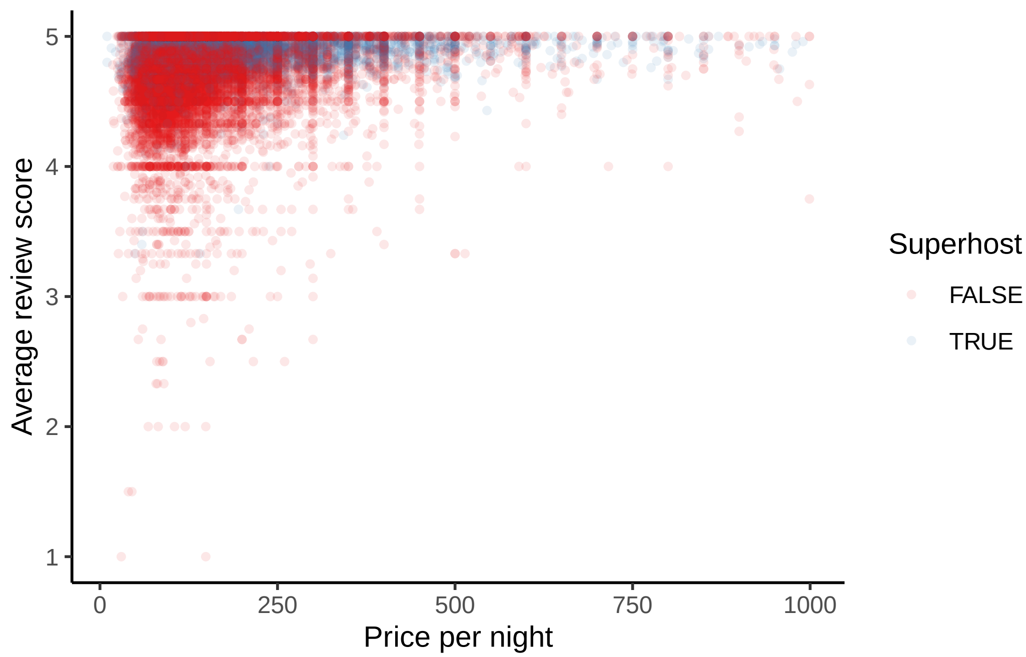
```
# A tibble: 6 x 13
   host_id host_response_time host_is_superhost host_total_listings_count
     <dbl> <fct>              <lgl>                                 <dbl>
1 50502817 within an hour     FALSE                                   778
2 50502817 within an hour     FALSE                    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

Nothing is weird so lets take a look at people with one property:

```r
airbnb_data_selected <-
  airbnb_data_selected |>
  add_count(host_id) |>
  filter(n == 1) |>
  select(-n)
```

We want to look at the correlation between price and review so...

```r
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(
    x = "Price per night",
    y = "Average review score",
    color = "Superhost"
  ) +
  scale_color_brewer(palette = "Set1")
```



Mostly shows that lower the price the more the average review is but lower ratings more probable in cheaper houses as compared to the high priced houses on airbnb.

Now we look at the superhost's response time:

```
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
```

but this table has NA values so lets look at hosts response time by whether they are a superhost or not.

```
airbnb_data_selected |>
  tabyl(host_response_time, host_is_superhost) |>
  adorn_percentages("col") |>
  adorn_pct_formatting(digits = 0) |>
  adorn_ns() |>
  adorn_title()
```

```
                    host_is_superhost
 host_response_time             FALSE        TRUE
 a few days or more         6%   (953)  0%    (24)
       within a day        22% (3,511) 12%   (770)
 within a few hours        24% (3,802) 26% (1,614)
     within an hour        48% (7,554) 61% (3,819)
```

Then we look at the neighbourhood.

```
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%
```

Then we estimate the model using `glm` and we use `modelsummary()` to see the values.

```
logistic_reg_superhost_response_review <-
  glm(
    host_is_superhost ~
      host_response_time +
      review_scores_rating,
    data = airbnb_data_selected,
    family = binomial
  )

modelsummary(logistic_reg_superhost_response_review)
```

Each row directly correlates to the likelihood of person being a superhost

```
#we save the data
write_parquet(
  x = airbnb_data_selected,
  sink = "2023-05-05-paris-airbnblistings-analysis_dataset.parquet"
  )
```

Arel-Bundock, Vincent. 2022. "modelsummary: Data and Model Summaries in R." *Journal of Statistical Software* 103 (1): 1–23. https://doi.org/10.18637/jss.v103.i01.

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/.

R Core Team. 2022. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Richardson, Neal, Ian Cook, Nic Crane, Dewey Dunnington, Romain François, Jonathan Keane, Dragoș Moldovan-Grünfeld, Jeroen Ooms, Jacob Wujciak-Jens, and Apache Arrow. 2024. *Arrow: Integration to 'Apache' 'Arrow'*. https://CRAN.R-project.org/package=arrow.

	(1)
(Intercept)	−16.262
	(0.481)
host_response_timewithin a day	2.019
	(0.211)
host_response_timewithin a few hours	2.695
	(0.210)
host_response_timewithin an hour	2.972
	(0.209)
review_scores_rating	2.624
	(0.089)
Num.Obs.	22 047
AIC	24 165.0
BIC	24 205.0
Log.Lik.	−12 077.507
RMSE	0.43

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.

van Buuren, Stef, and Karin Groothuis-Oudshoorn. 2011. "mice: Multivariate Imputation by Chained Equations in r." *Journal of Statistical Software* 45 (3): 1–67. https://doi.org/10.18637/jss.v045.i03.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

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.