



# An Analysis of Customer Reviews on Airbnb

Marcell P. Granát & Zója M. Szabó

March 9, 2021

## Contents

|                           |    |
|---------------------------|----|
| Introduction              | 3  |
| Theoretical consideration | 9  |
| Data                      | 9  |
| Explore data              | 9  |
| Modell building           | 9  |
| Consequences              | 9  |
| Summary                   | 9  |
| Appendix: R codes         | 11 |

### Abstract

Public perception of shared goods and/or services has changed significantly in the last few years. Shared accommodations have gained so great popularity, that house and flat sharing platforms like Airbnb now rival some of the world's largest businesses in hospitality. Sharing of personal properties provides an opportunity for owners to lower the transaction costs of operating short-term rentals and online rental marketplaces connect people who want to rent out their dwellings with the ones who are looking for accommodations. This study is aimed at determining the perceived behavior of individuals choosing Airbnb and introducing what factors influence user ratings and consumer adoption of Airbnb while assuming that customer feedbacks significantly influence consumer choice. We also analyze the market trends of the Hungarian Airbnb accommodations as primary examples of sharing or collaborative economy. Weekly data was collected for the Hungarian accommodation establishments all over the country. We aimed to build a complete dataset of the active suppliers by using automated "web scraping" techniques during a certain window of time. Our database contained customer ratings, reviews and pieces of public information concerning the rooms. We performed an advanced text analysis of the variables mentioned previously.

## List of Tables

## List of Figures

|   |  |   |
|---|--|---|
| 1 | Proportion of internet bookings of the main means of accommodation by countries and the partner type . . . . . | 3 |
| 2 | Starting points to our scrapping algorithm . . . . .   | 4 |
| 3 | Most common languages found in the comments by counties . . . . .  | 5 |
| 4 | Scatter plot of overall ratings and prices . . . . .   | 6 |
| 5 | Empirical cumulative distribution functions of overall rating scores and the number of reviews                 | 7 |

## Introduction

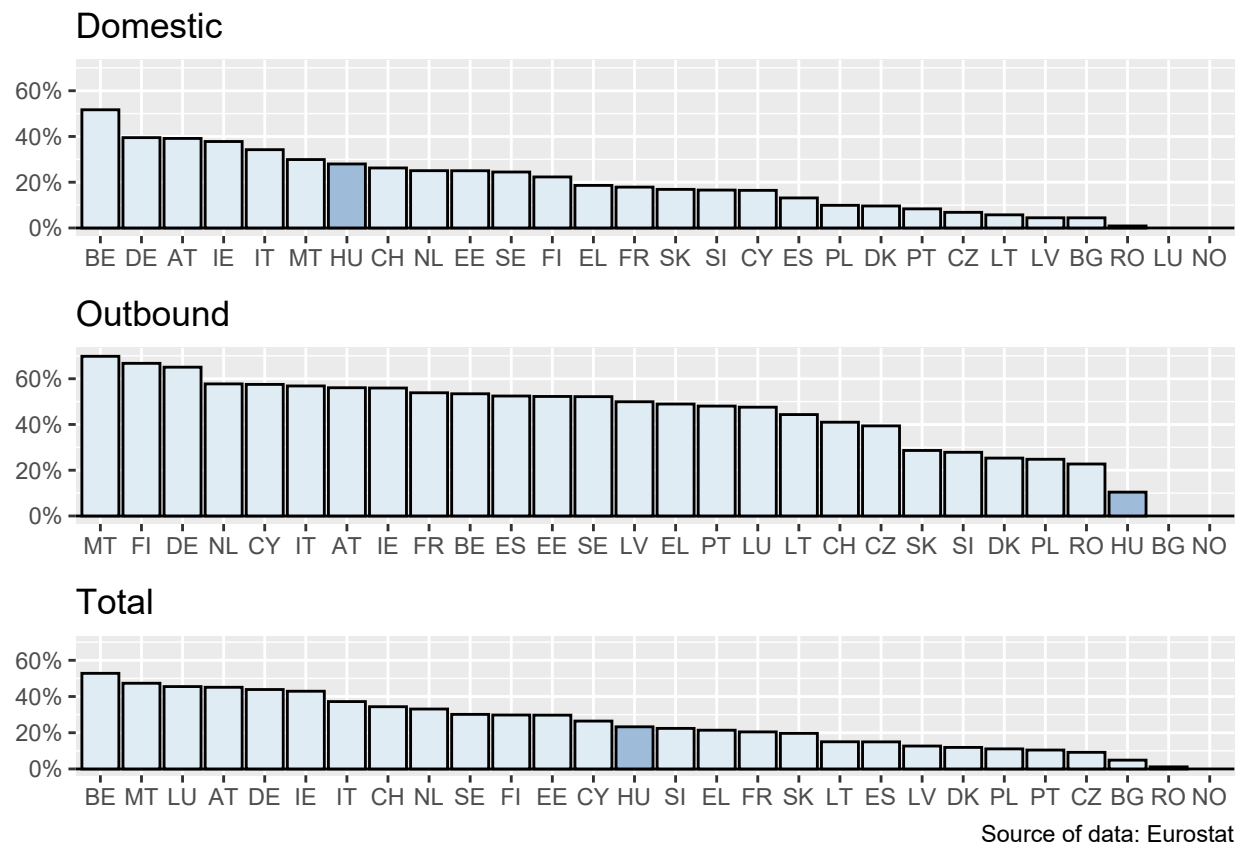


Figure 1: Proportion of internet bookings of the main means of accommodation by countries and the partner type

[leíró statok: eu bizottság felhívása]

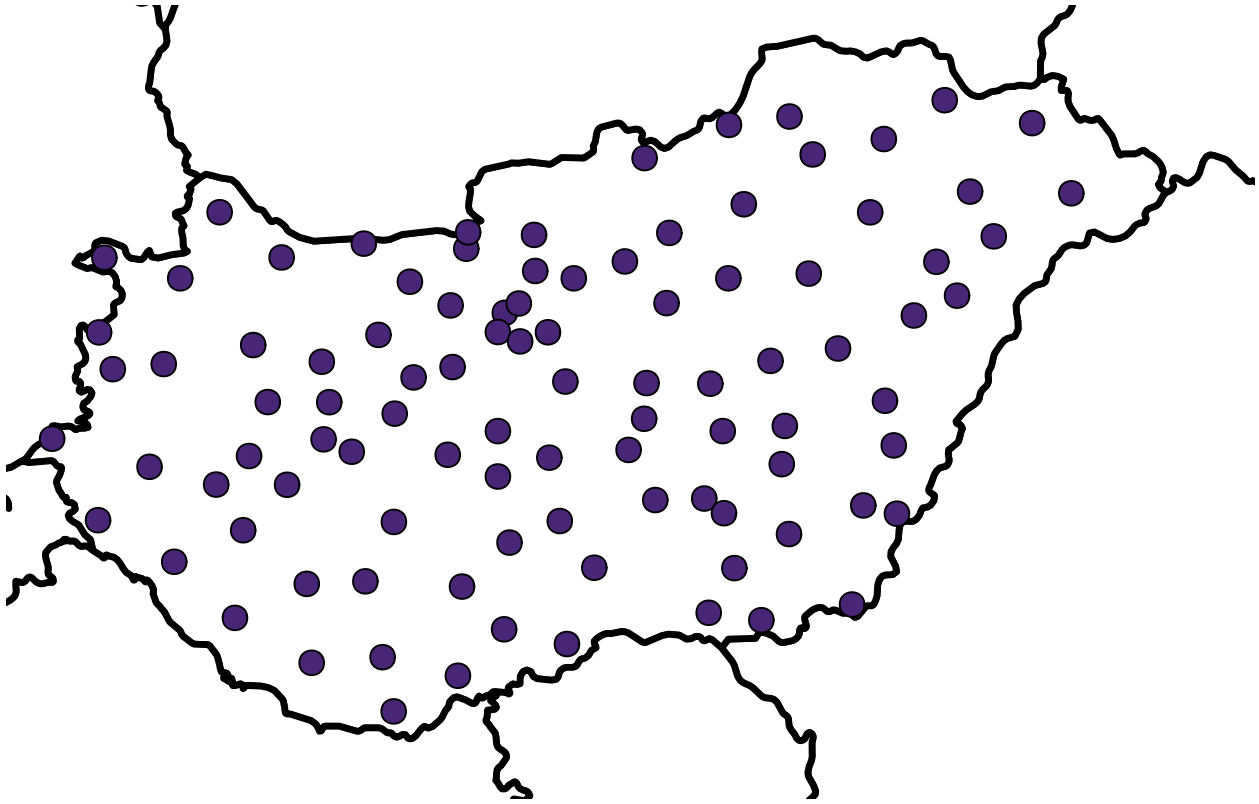


Figure 2: Starting points to our scrapping algorithm

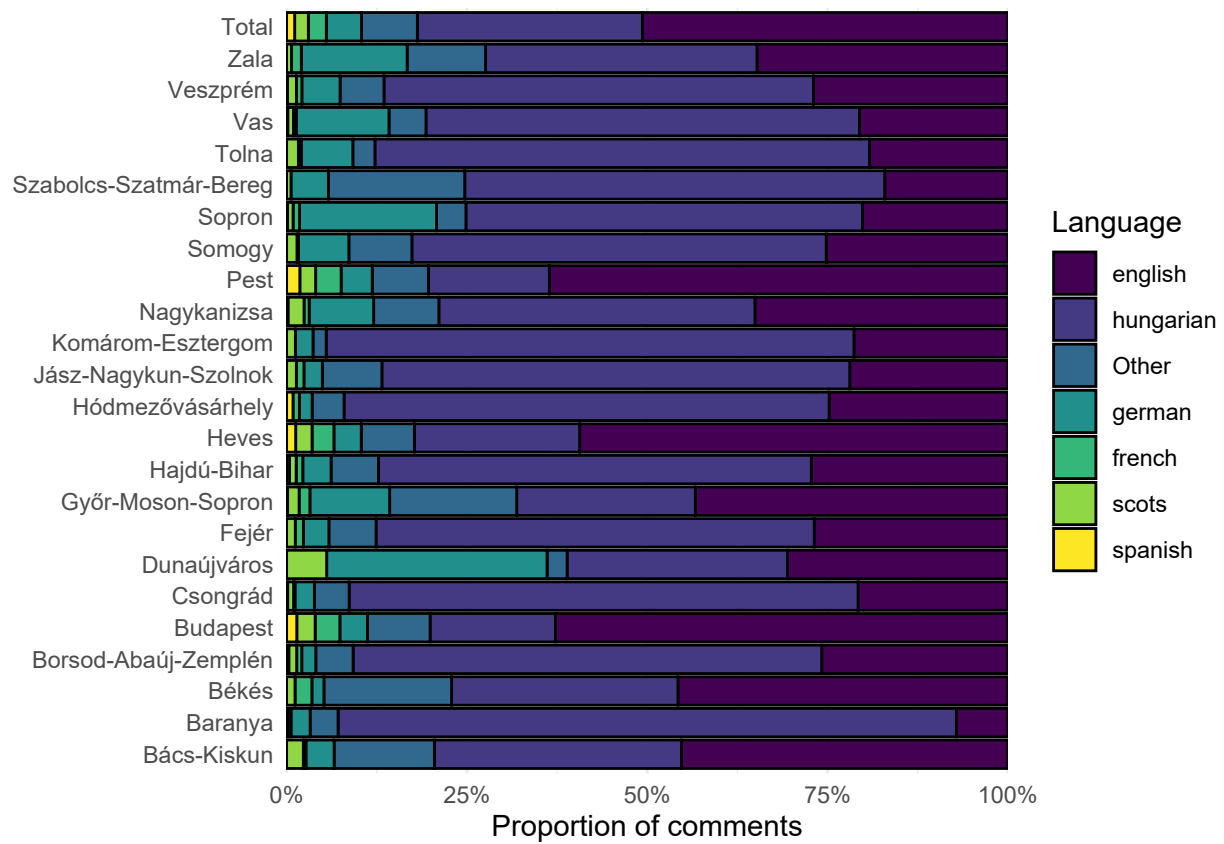


Figure 3: Most common languages found in the comments by counties

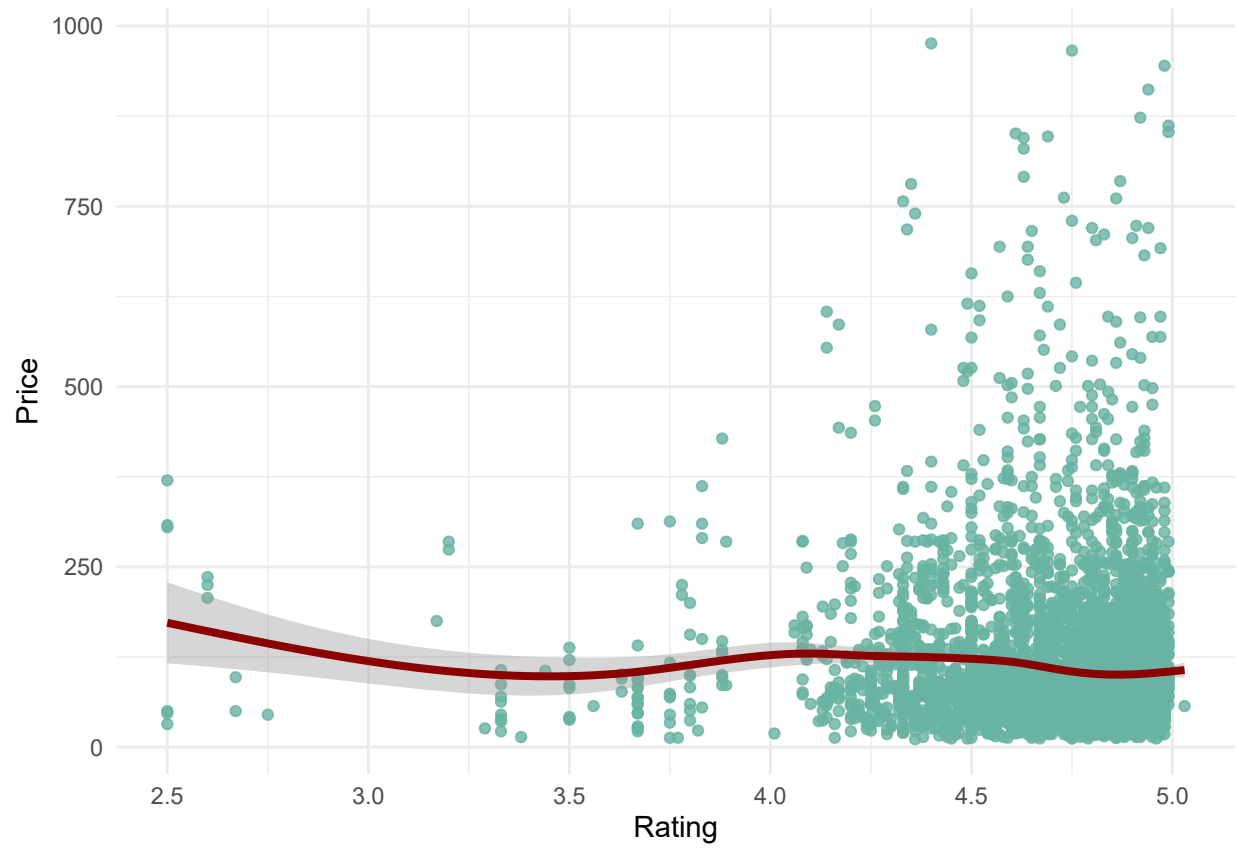


Figure 4: Scatter plot of overall ratings and prices

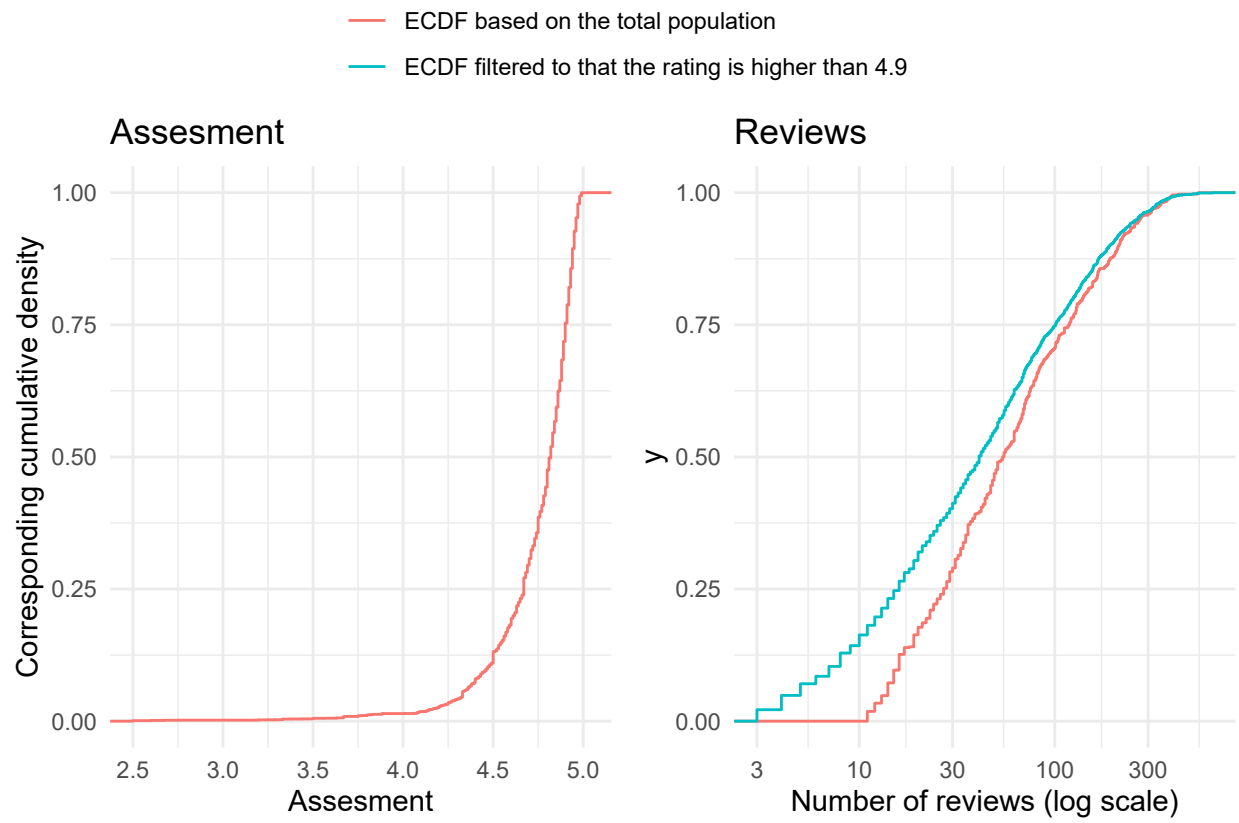
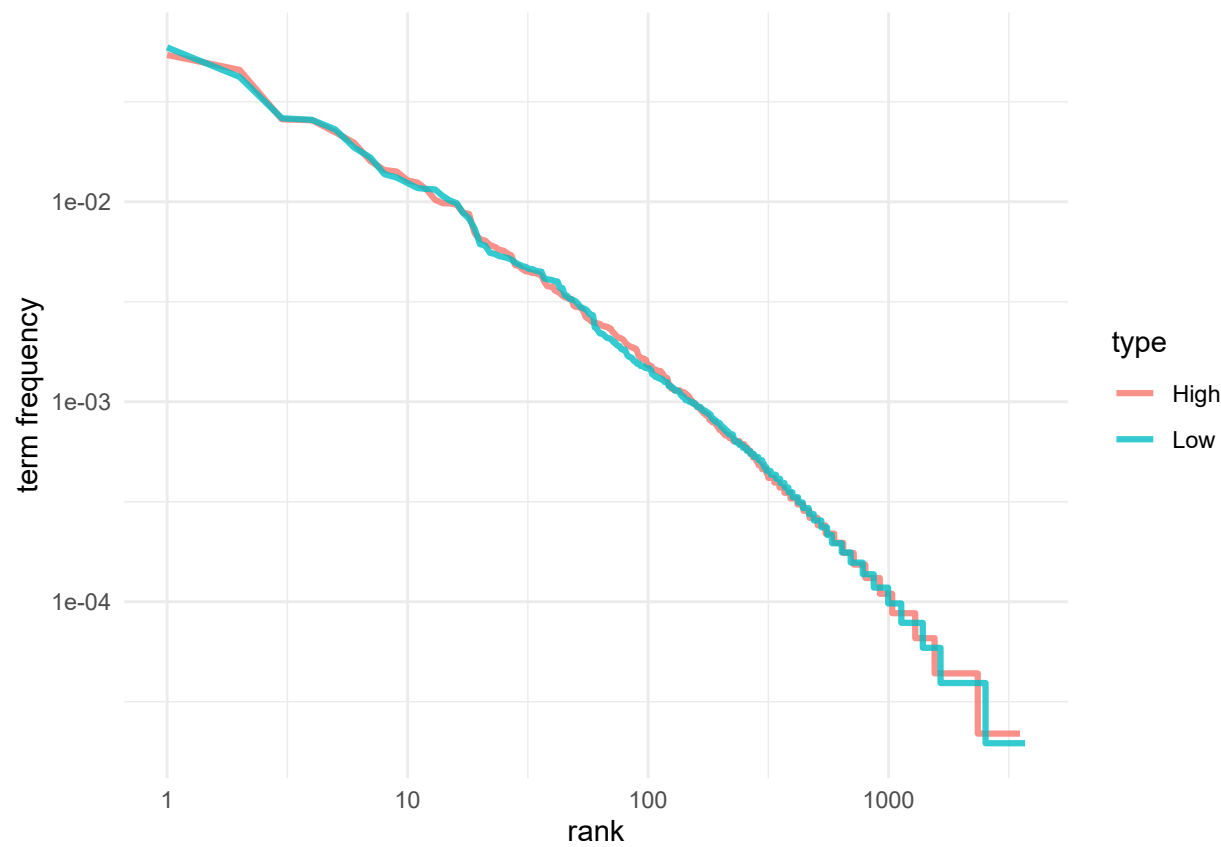


Figure 5: Empirical cumulative distribution functions of overall rating scores and the number of reviews







Theoretical consideration

Data

Explore data

Modell building

Consequences

Summary

## References

Silge, J. & Robinson, D. (2017), *Text mining with R: A tidy approach*, O'Reilly Media, Sebastopol, CA.

Zervas, G., Proserpio, D. & Byers, J. W. (2017), ‘The rise of the sharing economy: Estimating the impact of airbnb on the hotel industry’, *Journal of Marketing Research* **54**(5), 687–705.

## Appendix: R codes

```
1  # Packages -----
2
3  library(tidyverse)
4  library(knitr)
5  library(tidytext)
6  library(rvest)
7  library(parallel)
8  library(RSelenium)
9  library(rnaturalearth)
10 library(rnaturalearthdata)
11
12 # Introduction -----
13
14 f.plot_eurostat <- function(x){
15   eurostat::get_eurostat('tour_dem_torg', time_format = 'num') %>%
16     filter(trip_arr %in% c('ACC_WEB', 'TOTAL') & duration == 'N1-3' & time == 2017 &
17            purpose == 'TOTAL') %>%
18     pivot_wider(names_from = trip_arr, values_from = values) %>%
19     mutate(
20       value = ACC_WEB/TOTAL
21     ) %>%
22     filter(partner == x) %>%
23     mutate(geo = fct_reorder(geo, -value)) %>%
24     ggplot() +
25     aes(geo, value, fill = geo == 'HU') +
26     geom_hline(yintercept = 0) +
27     geom_col(color = 'black') +
28     scale_fill_brewer(palette = 3, guide = F) +
29     scale_y_continuous(labels = scales::percent, limits = c(0, .7)) +
30     labs(
31       x = NULL, y = NULL, title = case_when(
32         x == 'DOM' ~ 'Domestic',
33         x == 'OUT' ~ 'Outbound',
34         T ~ 'Total'
35       )
36     )
37 }
38
39 ggpubr::ggarrange(
40   f.plot_eurostat('DOM'),
41   f.plot_eurostat('OUT'),
42   f.plot_eurostat('WORLD') +
43     labs(caption = 'Source of data: Eurostat'),
44   ncol = 1
45 )
46
47 # Data -----
48
49 # Webscraping =====
50
51 cities <- readxl::read_excel("cities.xlsx") %>%
52   # Please find this attached file at the corresponding GitHub repository
```

```

53   # This contains the cities we searched on Airbnb.com and the URL to the searching
54   mutate(
55     URL = str_replace_all(URL, 'airbnb.hu', 'airbnb.com')
56   )
57
58   cities <- cities %>%
59     apply(1, function(x) {
60       n_rooms <- read_html(x[2]) %>%
61         html_nodes("._1snxcqc") %>%
62         html_text() %>%
63         {gsub(' .*', '', .)}
64       data.frame(city = x[1], url = x[2], n_rooms = n_rooms)
65     }
66   ) %>%
67   reduce(rbind)
68
69   ### Recursive price filter #####
70
71   RecPri_df <- cities %>%
72     filter(n_rooms == 'Több' | n_rooms == '300+' | n_rooms == '300' )
73
74   for (i in 1:nrow(RecPri_df)) {
75     run <- T
76     df <- seq(from = 10, to = 1500, length.out = 5) %>%
77       floor() %>%
78       {na.omit(data.frame(p1 = ., p2 = lead(.)))} %>%
79       mutate(
80         url = paste0(RecPri_df[i, 2], '&price_max=', p2, '&price_min=', p1)
81       )
82
83     df$n_rooms <- sapply(df$url, function(url) {
84       read_html(url) %>%
85         html_nodes("._1snxcqc") %>%
86         html_text()
87     })
88
89
90     while (run) {
91       df_still <- df %>%
92         filter(str_detect(n_rooms, "300"))
93
94       df <- df %>%
95         filter(!str_detect(n_rooms, "300"))
96
97       if (nrow(df_still) > 0) {
98
99
100        df_still <- apply(df_still, 1, function(x) {
101          seq(from = x[1], to = x[2], length.out = 3) %>%
102            floor() %>%
103            {na.omit(data.frame(p1 = ., p2 = lead(.)))} %>%
104            mutate(
105              url = paste0(RecPri_df[i, 2], '&price_max=', p2, '&price_min=', p1)

```

```

106   )
107   }
108   ) %>%
109   reduce(rbind)
110
111   df_still$n_rooms <- sapply(df_still$url, function(url) {
112     tryCatch(
113       read_html(url) %>%
114         html_nodes("._1snxcqc") %>%
115         html_text() %>%
116         as.character(),
117       error = function(e) NA)
118   }
119   )
120
121   df <- rbind(df, na.omit(df_still))
122   print(df)
123 } else {
124   run <- F
125   cities <- df %>%
126     mutate(city = RecPri_df[i, 1]) %>%
127     select(city, url, n_rooms) %>%
128     rbind(cities)
129 }
130 }
131 }
132
133 ### Repeat the search to different dates and merge the result #####
134
135 cities <- cities %>%
136   filter(!str_detect(n_rooms, 'Több') & !str_detect(n_rooms, '300'))
137
138 start_dates <- c(seq(from = 1, to = 89, by = 7), seq(from = 5, to = 92, by = 7)) %>%
139   sort()
140
141 room_list_total <- data.frame()
142
143 for (i in start_dates) {
144   cities_current <- cities %>%
145     mutate(
146       url = str_replace_all(url, '2021-07-01',
147                             as.character(as.Date(i, origin = '2021-05-30'))),
148       url = str_replace_all(url, '2021-07-04',
149                             as.character(as.Date(ifelse(i %% 7 == 1, i + 3, i + 2),
150                                                     origin = '2021-05-30'))))
151     )
152
153   cl <- makeCluster(7)
154   clusterExport(cl, list("cities"), envir = environment())
155   clusterEvalQ(cl, library(rvest))
156   clusterEvalQ(cl, library(tidyverse))
157
158   all_source <- parApply(cl = cl, cities_current, 1, function(x) {

```

```
159 page <- read_html(x[2])
160 n_rooms <- page %>%
161   html_nodes("._1snxcqc") %>%
162   html_text() %>%
163   {gsub(" s.*", "", .)} %>%
164   as.numeric()
165
166 df <- data.frame(city = x[1], url = x[2])
167
168 if (!is.na(n_rooms) && n_rooms > 20) {
169
170   df <- data.frame(
171     city = x[1],
172     url = html_nodes(page, xpath =
173       paste0('/html/body/div[4]/div/div/div/div[1]/main/div',
174         '/div/div[1]/div[2]/div/div/div[1]/nav/div/a[1]')) %>%
175     html_attr('href') %>%
176     {paste0('https://www.airbnb.com', .)},
177     v = seq(from = 20, to = (n_rooms %/% 20)*20, by = 20)
178   ) %>%
179   mutate(
180     url = str_replace(url, 'items_offset=20', paste0('items_offset=', v))
181   ) %>%
182   select(-v) %>%
183   rbind(df)
184 }
185 df
186 })
187
188 all_source <- reduce(all_source, rbind)
189
190 room_list <- parApply(cl = cl, all_source, 1, function(x) {
191   tryCatch({
192     page <- read_html(x[2])
193     URL <- page %>%
194       html_nodes('._gjfol0') %>%
195       html_attr('href') %>%
196       {paste0('https://www.airbnb.com', .)}
197     price <- page %>%
198       html_nodes('span._olc9rf0') %>%
199       html_text() %>%
200       .[1:length(URL)]
201     place <- page %>%
202       html_nodes('._b14dlit') %>%
203       html_text() %>%
204       .[1:length(URL)]
205     assesment <- page %>%
206       html_nodes('._18khxk1') %>%
207       html_text() %>%
208       .[1:length(URL)]
209     data.frame(
210       city = x[1],
211       title = page %>%
```

```

212     html_nodes('._gjfol0') %>%
213     html_attr('aria-label'),
214     URL, price, place, assesment
215   )
216 }, error = function(e) NULL)
217 })
218
219 stopCluster(cl)
220
221 room_list <- reduce(Filter(f = Negate(is.null), room_list), rbind)
222 room_list_total <- rbind(room_list_total, room_list)
223
224 }
225
226 ### Scrape all the rooms #####
227
228 room_list_total <- tibble(room_list_total) %>%
229   filter(!duplicated(id)) %>%
230   filter(!is.na(assessment)) %>%
231   filter(str_detect(assessment, '\\.')) %>%
232   mutate(
233     price = as.numeric(str_remove_all(price, '\\$')),
234     n_reviews = gsub(pattern = '.*[\\]', replacement = '', x = assessment) %>%
235       gsub(pattern = '.*', replacement = '') %>%
236       as.numeric(),
237     assesment = gsub('\\s.*', '', assessment) %>%
238       as.numeric()
239   )
240
241 room_interval <- 1:nrow(room_list_total)
242
243 raw_dat <- list()
244
245 rD <- rsDriver(verbose = TRUE,
246               port=48458L,
247               chromeversion = '88.0.4324.27',
248               check = TRUE)
249
250 remDr <- rD$client
251
252 for (i in room_interval) {
253   remDr$navigate(pull(room_list_total[i, ], URL))
254
255   url_descript <- character()
256   url_amenities <- character()
257   url_reviews <- character()
258   host <- character()
259   rules <- character()
260   assesment <- character()
261   bed <- character()
262   comments <- character()
263   amenities <- character()
264   stars <- character()

```

```

265
266 Sys.sleep(4)
267 page_room <- remDr$getPageSource()[[1]] %>%
268   read_html()
269
270 page_room %>%
271   html_nodes("._13e0raay") %>%
272   html_attr("href") %>%
273   {paste0("https://www.airbnb.com", .)} %>%
274   {
275     url_amenities <- str_subset(., "amenities")
276     url_reviews <- str_subset(., "reviews")
277   }
278
279 host <- page_room %>%
280   html_nodes("._14i3z6h") %>% # host
281   html_text()
282
283 rules <- page_room %>%
284   html_nodes("._u827kd") %>% # rules
285   html_text()
286
287 bed <- page_room %>%
288   html_nodes("._1a5glfg") %>% # bed
289   html_text()
290
291 stars <- page_room %>%
292   html_nodes("._1s11ltsf") %>%
293   html_text()
294
295 if (length(url_reviews) != 0 && url_reviews != 'https://www.airbnb.com') {
296
297   remDr$navigate(url_reviews)
298   Sys.sleep(6)
299   comments <- remDr$getPageSource()[[1]] %>%
300     read_html() %>%
301     html_nodes('._1xib9m0') %>%
302     html_text()
303
304   remDr$navigate(url_amenities)
305   Sys.sleep(3)
306   amenities <- remDr$getPageSource()[[1]] %>%
307     read_html() %>%
308     html_nodes("._vzrbjl") %>%
309     html_text()
310 }
311
312 raw_dat[[length(raw_dat) + 1]] <- list(
313   source = room_list_total[i, ],
314   url_amenities = url_amenities,
315   url_reviews = url_reviews,
316   host = host,
317   rules = rules,
318   bed = bed,

```



```

319   comments = comments,
320   amenities = amenities,
321   stars = stars
322 )
323 }
324
325 hun_cities <- read_csv("worldcities.csv") %>%
326   filter(country == "Hungary") %>%
327   select(city, lat, lng, admin_name, population)
328
329 cities <- readxl::read_excel("cities.xlsx")
330
331 world <- ne_countries(scale = "large", returnclass = "sf")
332
333 merge(cities, hun_cities, by = 'city') %>%
334   tibble() %>%
335   ggplot() +
336   geom_sf(data = world, size = 1.2, fill = 'white', color = 'black') +
337   coord_sf(xlim = c(16, 23.4), ylim = c(45.5, 48.7), expand = FALSE) +
338   geom_point(aes(x = lng, y = lat), size = 4, alpha = 1, color = 'black',
339             shape = 21, fill = viridis::viridis(1, begin = .1)) +
340   theme_void()
341
342 lapply(dat, function(x) {
343   tibble(city = x[["source"]][["city"]], comments = x$comments) %>%
344     mutate(language = textcat::textcat(comments))
345 }) %>%
346   reduce(rbind) %>%
347   mutate(
348     language = fct_lump(language, n = 6) %>%
349     fct_infreq()
350   ) %>%
351   merge(hun_cities) %>%
352   {rbind(., mutate(., admin_name = 'Total'))} %>%
353   mutate(admin_name = fct_reorder(admin_name, admin_name == 'Total')) %>%
354   select(admin_name, language) %>%
355   na.omit() %>%
356   ggplot() +
357   aes(y = admin_name, fill = language) +
358   scale_x_continuous(labels = scales::percent, limits = c(0,1), expand = c(0,0)) +
359   geom_bar(color = "black", position = position_fill()) +
360   scale_fill_viridis_d() +
361   labs(x = 'Proportion of comments', y = NULL, fill = "Language")
362
363 room_list <- room_list_total %>%
364   filter(!duplicated(id)) %>%
365   filter(!is.na(assessment)) %>%
366   filter(str_detect(assessment, '\\.'))
367
368 room_list %>%
369   ggplot() +
370   aes(assessment, price) +
371   geom_point(color="#69b3a2", alpha=0.8) +

```

```

372   geom_smooth() +
373   labs(x = 'Rating', y = 'Price')
374
375   ggpubr::ggarrange(
376   room_list %>%
377     ggplot() +
378     stat_ecdf(aes(x = assesment, color = 'ECDF based on the total population')) +
379     geom_blank(aes(color = 'ECDF filtered to that the rating is higher than 4.9')) +
380     labs(x = 'Assesment', y = 'Corresponding cumulative density', title = 'Assesment',
381          color = NULL),
382   room_list %>%
383     ggplot() +
384     stat_ecdf(data = filter(room_list, assesment > 4.9), mapping = aes(n_reviews,
385                               color = 'ECDF filtered to that the rating is higher than 4.9')) +
386     stat_ecdf(aes(x = n_reviews, color = 'ECDF in the total population')) +
387     scale_x_log10() +
388     labs(x = 'Number of reviews (log scale)', title = 'Reviews') +
389     theme(
390       legend.position = 'bottom'
391     ), common.legend = T
392 )
393
394 dat_words_eng <- lapply(dat, function(x) {
395   tryCatch({
396     tibble(comments = x$comments) %>%
397       mutate(language = textcat::textcat(comments)) %>%
398       filter(language == "english") %>%
399       tail(-2) %>%
400       tidytext::unnest_tokens(words, comments, to_lower = T) %>%
401       mutate(assesment = x[["source"]][["assesment"]],
402              n_reviews = x[["source"]][["n_reviews"]])
403   }, error = function(e) NULL)
404 }
405 ) %>%
406 {reduce(Filter(f = Negate(is.null), .), rbind)}
407
408 freq_by_rank_eng <- dat_words_eng %>%
409   filter(n_reviews >= 15) %>%
410   mutate(
411     type = as.numeric(Hmisc::cut2(assesment, g = 10, levels.mean = T)),
412     type = case_when(
413       type == min(type) ~ 'Low',
414       type == max(type) ~ 'High',
415       T ~ 'Middle'
416     ),
417   ) %>%
418 {
419   df <- .
420   df <- df %>%
421     group_by(type, words) %>%
422     summarise(n = n()) %>%
423     ungroup()
424   merge(

```

```
425     df,
426     df %>%
427       group_by(type) %>%
428       summarise(total = sum(n))
429   )
430 } %>%
431 arrange(desc(n)) %>%
432 group_by(type) %>%
433 mutate(rank = row_number(),
434         `term frequency` = n/total) %>%
435 ungroup()
436
437 freq_by_rank_eng %>%
438   filter(type == 'High' | type == 'Low') %>%
439   ggplot(aes(rank, `term frequency`, color = type)) +
440   geom_line(size = 1.1, alpha = 0.8) +
441   scale_x_log10() +
442   scale_y_log10()
443
444 freq_by_rank_eng %>%
445   select(words, type, n) %>%
446   bind_tf_idf(term = words, document = type, n = n) %>%
447   anti_join(data.frame(words = c(stopwords::stopwords(), "also", "can"))) %>%
448   filter(type != 'Middle' & n > 20) %>%
449   arrange(desc(tf_idf)) %>%
450   group_by(type) %>%
451   group_modify(~ head(mutate(.x, rank = row_number()), 50)) %>%
452   reshape2::acast(words ~ type, value.var = "rank", fill = 0) %>%
453   wordcloud::comparison.cloud(colors = viridis::viridis(2, direction = -1, end = .7),
454                               max.words = 50)
```