Web Scrapping Project - STAT260

Luan Nguyen - 301571025

2023-03-19

Requirement

Because every website have a different build structure, in this particular report, we will scrap data from this website: `https://ca.trustpilot.com/review/www.apple.com`

Quick knowledge about HTML, each HTML element will be associated with a class, id, element tag,... So we will use those to distinct and extract the element that we want.

To be more efficient, we will use **element selector gadget**, you can read more about the usage instruction in the website.

We will also use these R packages: `tidyverse`, `rvest`, `stringr`, `lubridate`. Detail of the packages is in the below

Outline steps

1. Find the number of the last page
2. Generate a vector that store all the pages
3. Scrap data from each of the sub-webpage from the vector
4. Combine the data into a single comprehensive data frame

Scrapping

Install the package if needed

```
# install.packages("tidyverse")
# install.packages("rvest")
# install.packages("stringr")
# install.packages("lubridate")
library(tidyverse)
library(dplyr)
library(rvest)
library(stringr)
library(lubridate)
```

- `rvest`: read and process HTML data (select element, class, or text)
- `stringr`: string manipulation (ex. turn string to date)
- `lubridate`: manipulate date time format (ex. turn string to date time format)

Find the number of the last page

```
url <- "https://ca.trustpilot.com/review/www.apple.com"
first_page <- read_html(url)
```

`read_html()` will read from the URL, and return the source html code

Generate a vector that store all the pages

Because this website reviews other company's website, each company's website will have different number of pages depends on how popular the site. Because of that, we make a function that will return the last page number of a particular company it review (Apple.com in this case) so we can reuse in the future for other company.

```
get_last_page <- function(html) {
  pages_data <- html %>%
    html_nodes(".pagination-link_item__mkuN3") %>%
    html_text()
  pages_data[length(pages_data)] %>%
    as.numeric()
}
(last_page <- get_last_page(first_page))
```

```
## [1] 293
```

- `html_nodes()`: find the section of the html source code that contain the input node.
- `html_text()`: return the content of that section.
- `as.numeric()`: convert it to numeric type

Scrap data from each of the sub-webpage from the vector

Now we will scrapping each element and store it into a vector, we will start with the comment

```
get_review_comment <- function(html) {
  html %>%
    html_nodes('.link_notUnderlined__szqki .typography_appearance-default__AAY17') %>%
    html_text() %>%
    str_trim()
}

get_review_name <- function(html) {
  html %>%
    html_nodes('.styles_consumerDetails__ZFieb .typography_appearance-default__AAY17') %>%
    html_text() %>%
    str_trim()
}

get_review_date <- function(html) {
  html %>%
    html_nodes('.typography_body-m__xgxZ_.typography_color-black__5LYEn') %>%
```

```
    html_text() %>%
    str_match("(?<=experience: ).*$") %>%
    mdy()
}
```

- Same as above, we use element selector gadget to select element that we want to scrap, copy that into `rvest::html_nodes()` then extract the text using `rvest::html_text()`. `stringr::str_trim()` to trim the white space at the start and end of the content. The result is a vector with all the comment.
- The date is a bit tricky, it's a string contains unnecessary letter so we use RegEx and `stringr::str_match()` to extract the date part only, then convert to data format using `lubridate::mdy()`

Last part is the rating, because it's a picture type so instead of reading the section content, we will read the attribute content instead

```
get_review_rate <- function(html) {
  html %>%
    html_nodes('.styles_reviewHeader__iU9Px img') %>%
    html_attrs() %>%
    map(1) %>%
    str_trim() %>%
    str_extract("(?<=Rated ).(?= out)") %>%
    as.numeric()
}
```

- `rvest::html_attrs()`: read the attributes in that node
- `purrr::map()`: select the first attribute in this case, because it contain the number of stars reviewing.
- Then we use `stringr::str_extract()` and RegEx again to extract the useful content which is the star number

Combine the data into a single comprehensive data frame

Finally, we will write a function to combine all the function above and make data frame from a single URL then another function to combine all the data frame into a tibble

```
get_data_table <- function(url, companyName) {
  html <- read_html(url)
  comment <- get_review_comment(html)
  reviewer <- get_review_name(html)
  date <- get_review_date(html)
  rate <- get_review_rate(html)
  reviewTibble <- tibble(comment = comment,
                         reviewer = reviewer,
                         date = date,
                         rate = rate)
  reviewTibble <- reviewTibble %>%
    mutate(company = companyName) %>%
    select(company, reviewer, date, rate, comment)
}
```

After have all the vector such as reviewer name, comment, date, rate, we combine them as a tibble, `dplyr::mutate()` new column that contains company name, then `dplyr::select()` to arrange columns keep the consistent of all tibbles

Now we will write a function with loop control that will read all the page of that particular company

```
scrape_write_table <- function(url, companyName) {
  first_page = read_html(url)
  last_page_number = get_last_page(first_page)
  list_of_pages <- str_c(url, '?page=', 1:last_page_number)
  dataTibble <- NULL
  for(i in 1:length(list_of_pages)) {
    data <- get_data_table(list_of_pages[i], companyName)
    dataTibble <- bind_rows(dataTibble, data)
  }
  return(dataTibble)
}
```

- `dplyr::bind_rows()`: combine first tibble input and second tibble input
- We loop until the end of the `list_of_pages`, then export the tibble data frame.

We will have the output as follow

```
(data <- scrape_write_table(url, "APPLE"))
```

```
## # A tibble: 5,855 x 5
##    company reviewer          date        rate comment
##    <chr>   <chr>             <date>     <dbl> <chr>
##  1 APPLE   Abimbola Bamikole 2023-01-04     1 If you like beautiful pictures do~
##  2 APPLE   Janet Wright      2023-03-16     1 I am extremely dissapointed by Ap~
##  3 APPLE   Rob Leroy         2023-03-18     1 Just can never resolve their issu~
##  4 APPLE   Ice House Design  2023-03-15     1 Still look great but wow, they're~
##  5 APPLE   noreen darragh    2023-03-19     1 This Company is so annoying
##  6 APPLE   Paul Morgan       2023-03-14     1 Ipad keyboard a disaster
##  7 APPLE   wizard            2023-03-14     1 Awful call centre IT system!
##  8 APPLE   Nicole Sireci     2023-03-17     1 Customer service is horrible
##  9 APPLE   nets              2023-03-16     1 Horrible service
## 10 APPLE   Charlotte         2023-03-11     1 Taking awful service to whole new~
## # ... with 5,845 more rows
```

Problem note

1. Website update

One obvious problem is that the website owner decide to change the webpage like upgrade, update, changing structure of the site which will require us to review the code

2. Rate limited

I was trying to scrap a company with 57,599 reviewer in 2537 pages which mean require 2537 request to the server, after ran the code, my IP address was being block from further access to the website to prevent server overload or even DDoS attack. In such case, I would either set up a proxy server or limited the rate to wait a couple minutes or even hours between each request

3. Inconsistent data

During scrapping, I noticed some inconsistencies of the website, such as some review will have no comment but only title and rate, which will lead to inconsistent length between vector which can leads to error or incorrect tibble. In such case, I would extract data by each comment which will be much longer

4. UI Interaction

Some data will only show up after an interaction such as click a button for data to show up. In such case, there is API or click bot that can help you with this.

Citation list

https://ca.trustpilot.com/review/www.canadianappliance.ca

https://ca.trustpilot.com/review/www.apple.com

https://github.com/abhimotgi/dataslice/blob/master/R/Web%20Scraping%20with%20RVest%20Part%201.R

https://www.youtube.com/watch?v=v8Yh_4oE-Fs&t=209s

https://www.datacamp.com/tutorial/r-web-scraping-rvest

https://axiom.ai/blog/5-problems-webscrapers.html