Project 3- Webscraping and data

Yenus Ibrahim Ayalew

January 15, 2026 | 23:46:00 | CET

b) The file `friends_info.xml` contains information about all episodes from the TV series Friends (accessible at https://raw.githubusercontent.com/intro-to-data-science-24/labs/main/data/friends_info.xml). Import the file using the `xml2::read_xml()` function. (Hint: this is an XML document, so you will need to use the xml analogues of `rvest::read_html()`, `rvest::html_nodes()`, and `rvest::html_text()`).

- Use XPath expressions to extract the season, episode number, and title of all episodes. Store this information in a data frame (`episode_df`) and render the first 5 rows as an HTML table.
- Use XPath expressions to return the vector `directed_by_ross` providing the titles of all episodes directed by Ross himself (**David Schwimmer**).
- Create a chart that reports the frequency of episodes mentioning one of the six main characters (Ross, Rachel, Monica, Chandler, Joey, Phoebe) in the title.

```r
# load required packages
library(xml2)
library(dplyr)
library(ggplot2)
library(knitr)

# import XML file
friends_info <- read_xml(
  "https://raw.githubusercontent.com/intro-to-data-science-24/labs/main/data/friends_info.xml"
)

# extract all episode nodes
episodes <- xml_find_all(friends_info, ".//episode")

# extract season, episode number, and title using XPath
episode_df <- data.frame(
  season = sapply(episodes, function(x)
    xml_integer(xml_find_first(x, "./season"))),
  episode_number = sapply(episodes, function(x)
    xml_integer(xml_find_first(x, "./episode_number"))),
  title = sapply(episodes, function(x)
    xml_text(xml_find_first(x, "./title"))),
  stringsAsFactors = FALSE
)

# render first 5 rows as HTML table
kable(head(episode_df, 5))
```

season	episode_number	title
1	1	The Pilot
1	2	The One with the Sonogram at the End
1	3	The One with the Thumb
1	4	The One with George Stephanopoulos
1	5	The One with the East German Laundry Detergent

```r
# titles of episodes directed by David Schwimmer (Ross)
directed_by_ross <- xml_text(
  xml_find_all(
    friends_info,
    ".//episode[directed_by='David Schwimmer']/title"
  )
)

directed_by_ross
```

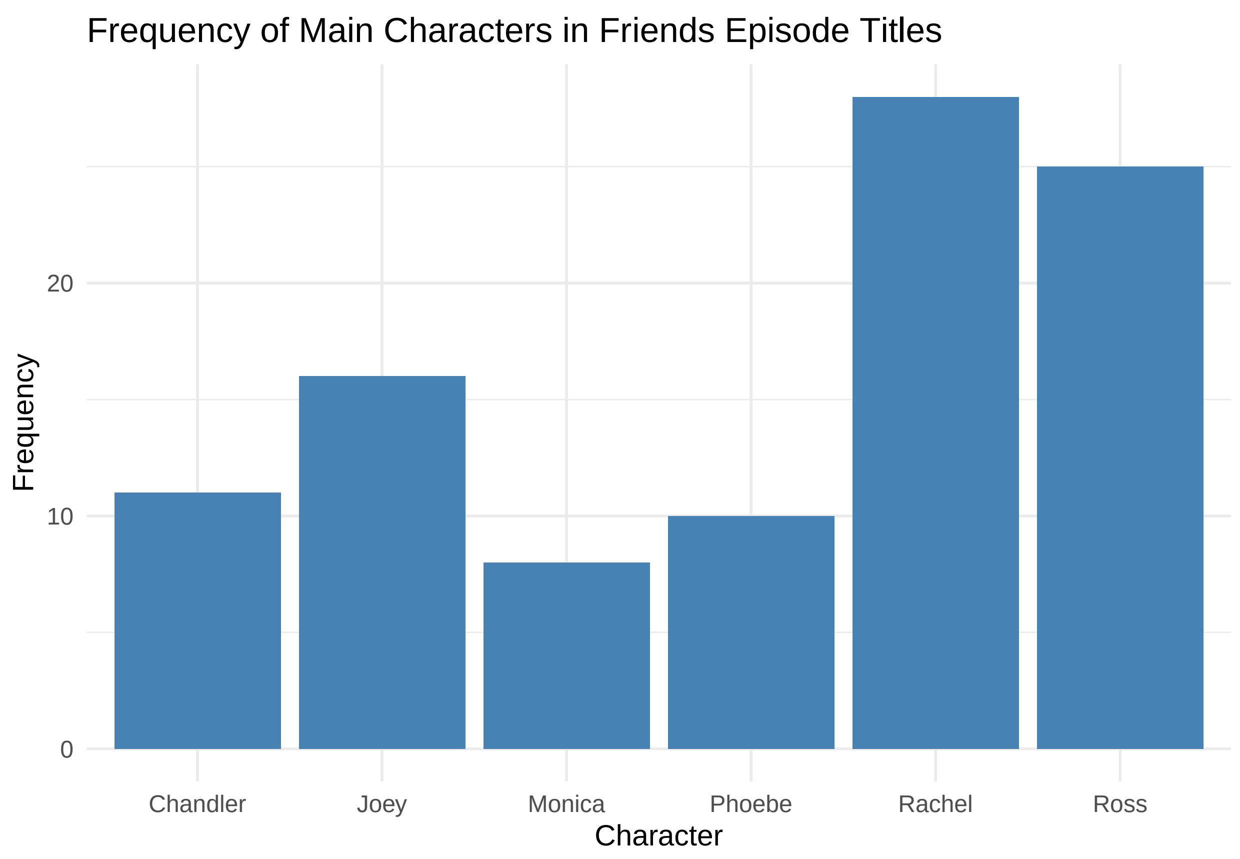
```
 [1] "The One on the Last Night"
 [2] "The One with Rachel's Assistant"
 [3] "The One with Ross's Library Book"
 [4] "The One with All the Candy"
 [5] "The One with the Truth About London"
 [6] "The One with the Red Sweater"
 [7] "The One with the Stripper"
 [8] "The One Where Joey Dates Rachel"
 [9] "The One with Phoebe's Birthday Dinner"
[10] "The One with the Birth Mother"
```

```r
# frequency of main characters mentioned in episode titles
characters <- c("Ross", "Rachel", "Monica", "Chandler", "Joey", "Phoebe")

char_freq <- sapply(characters, function(name) {
  sum(grepl(name, episode_df$title, ignore.case = TRUE))
})

char_df <- data.frame(
  Character = characters,
  Frequency = char_freq
)

# plot the frequency chart
ggplot(char_df, aes(x = Character, y = Frequency)) +
  geom_col(fill = "steelblue") +
  labs(
    title = "Frequency of Main Characters in Friends Episode Titles",
    x = "Character",
    y = "Frequency"
  ) +
  theme_minimal()
```



Task 2 - Scraping members of the European Parliament

The European Parliament's website maintains a full list of MEPs. For this exercise, you will focus on scraping data from the HTML code, so please don't make use of the linked XML files.

a) Extract a data frame (`meps_df`) with the variables listed below, print the first 3 observations, and check the number of rows.

Variable	Data type	Description
`name`	`<chr>`	Full name of MEP
`ep_party_group`	`<chr>`	EP party group
`country`	`<chr>`	Country of MEP
`nat_party`	`<chr>`	National party of MEP
`profile_url`	`<chr>`	URL linking to MEP profile
`mep_id`	`<chr>`	MEP numeric identifier included as a path in `profile_url`

```r
url <- "https://www.europarl.europa.eu/meps/en/full-list/all"
url_parsed <- rvest::read_html(url)
```

```r
# Construct data frame

## using xpath selectors (from selector gadget)
mep_df <-
  data.frame(
    name = url_parsed %>% rvest::html_elements(xpath = '//*[(@id = "docMembersList")]//*[contains(conca
    party_group = url_parsed %>% rvest::html_elements(xpath = '//*[contains(concat( " ", @class, " " ),
    nat_party = url_parsed %>% rvest::html_elements(xpath = '//*[contains(concat( " ", @class, " " ), c
    country = url_parsed %>% rvest::html_elements(xpath = '//*[contains(concat( " ", @class, " " ), conc
    profile_url = url_parsed %>% rvest::html_elements(xpath = '//*[(@id = "docMembersList")]//*[contains
    stringsAsFactors = FALSE
    ) %>%
  dplyr::mutate(mep_id = stringr::str_extract(profile_url, "\\d+"))

# print table
mep_df %>% slice_head(n = 3) %>% knitr::kable()
```

name	party_group	nat_party	country	profile_url	mep_id
Mika AALTOLA	Group of the European People's Party (Christian Democrats)	Kansallinen Kokoomus	Finland	https://www.europarl.europa.eu/meps/en/256810	256810
Maravillas ABADÍA JOVER	Group of the European People's Party (Christian Democrats)	Partido Popular	Spain	https://www.europarl.europa.eu/meps/en/257043	257043
Magdalena ADAMOWICZ	Group of the European People's Party (Christian Democrats)	Independent	Poland	https://www.europarl.europa.eu/meps/en/197490	197490

b) Now, using the `profile_url` data in the table. Provide polite code that downloads the first 10 of the linked HTMLs to a local folder retaining the MEP IDs as file names.

- Explain why your code follows best practice of polite scraping by implementing at least three practices (bullet points are sufficient).
- Provide proof that the download was performed successfully by listing the file names and reporting the total number of files contained by the folder.
- Make sure that the folder itself is not synced to GitHub using `.gitignore`.

```r
folder <- "mep_profiles/"
dir.create(folder)

for (i in 1:10) {
  if (!file.exists(paste0(folder, basename(mep_df$mep_id[i]), ".html"))) {
    try(download.file(mep_df$profile_url[i],
                      destfile = paste0(folder, mep_df$mep_id[i], ".html"),
                      headers = c("From" = "munzert@hertie-school.org")))
    Sys.sleep(runif(1, 1, 2))
  }
}

list.files(folder)
```

```
 [1] "113523.html" "197400.html" "197403.html" "197490.html" "256810.html"
 [6] "256820.html" "256869.html" "256956.html" "256987.html" "257043.html"
```

```r
length(list.files(folder))
```

```
[1] 10
```