Problem Set 4

Due February 21, 2023

Instructions

- Read all of these instructions closely.
- This problem set is due Tuesday, February 21, 2023 at 4pm.
- Submit files via Github:
  1. the .Rmd (R Markdown) file
  2. the knitted .pdf file
  3. anything else the particular problem set might require
- Use a copy of this file, perhaps with your name or initials appended to the file name, to write your answers to the questions. You'll see there is a designated space where your answers should begin.
- Knitting the .Rmd file to a .pdf file *as you work* will ensure your code runs without errors and is working how you expect. Knit early and often. You've already read the instruction that a knitted .pdf is required when you submit.
- Per the syllabus, I will not accept any late work. Keep in mind the two lowest problem set scores are dropped. Turn in what you have.
- Clarification on the expectations for problem set submissions (posted in Slack, copied here):
  - Always print the output of the code I'm requesting.
    - Ex: If I want you to create a vector x with elements 1 through 10, print x after creating it so I can see it worked.
  - Write any written answers in the space outside the code chunk, not inside with an R comment.
    - R comments are great to clarify code, but not for answering the question.
  - Make sure any code or written content is not cut off in the pdf.
    - This really should only apply to code, because if you follow item 2 in this list, the pdf will compile your written answers nicely.

Question 1–Webscraping

This problem set has only one question. Our goal is to scrape certain contact info for all Notre Dame political science core faculty. This activity will ask you to navigate across webpages, find the information of interest within each page, store it in a dataframe in R, and then write the dataframe to a csv.

The information you need to scrape for each of the 50 core faculty members is:

- link to their ND webpage (done during inclass activity)
- name
- title (e.g., Assistant Professor, Associate Professor, etc.)
- fields of study
- office hours (done during inclass activity)
- office location (optional! this one is tricky)
- phone
- email

Each of these pieces of information should be stored in a separate column in an R dataframe.

Additional problem set details:

- Include short, informative R comments describing your code as needed.
- If you prefer, you can complete this problem set in an R script (.R file). Since this problem set doesn't require any any written answers and is one large task, an .R file might be easier than a .Rmd file. It's your choice. *Therefore, a knitted pdf is not required for this assignment*
- In addition to the R code, you must submit a .csv file containing the scraped information. I recommend the `write.csv` function.

```
library(rvest)

# get html of "Core Faculty" page
url <- "https://politicalscience.nd.edu/people/core-faculty/"
html <- read_html(url)

# get 50 faculty links on that page
faculty_links <- html %>%
  html_nodes('h2.faculty-item-name') %>%
  html_nodes('a') %>%
  html_attr('href')


# set up dataframe to store results
faculty_df <- data.frame("link" = faculty_links,
                         "name" = NA,
                         "title" = NA,
                         "fields" = NA,
                         "office_hours" = NA,
                         "office_location" = NA,
                         "phone" = NA,
                         "email" = NA)

# URL to paste faculty link to inside loop
base_url <- "https://politicalscience.nd.edu"

# iterate by **index** to fill
# in each row of the dataframe
for(i in 1:nrow(faculty_df)){
```

```r
# paste the faculty extension to the base URL
url <- paste0(base_url, faculty_df$link[i])
# get html
html <- read_html(url)

# Name ---
# from page title
faculty_df$name[i] <- html %>%
  html_nodes('h1.page-title') %>%
  html_text2()

# The rest of the info from
# "faculty details" div
details_div <- html %>%
  html_nodes('div.faculty-details')

# Fields ---
# has a specific class
faculty_df$fields[i] <-details_div %>%
  html_nodes('p.faculty-fields') %>%
  html_text2()

# Title ---
# is only thing italicized
faculty_df$title[i] <- details_div %>%
  html_nodes("p") %>%
  html_nodes("em") %>%
  html_text2()

# Email ---
# email and CV both might have href,
# so need to investigate links a little bit
# to grab the right one
links <- details_div %>%
  html_nodes("a") %>%
  html_text2()

email_idx <- grepl(pattern = "@nd.edu", x = links)
faculty_df$email[i] <- ifelse(length(email_idx) != 0, links[email_idx], NA)


# The result of the elements don't
# have distinguishing element features
# Need to be more creative
details_text <- details_div %>%
  html_nodes('p') %>%
  html_text2()

# Office hours ---
# Find element with our text skills
oh_idx <- which(grepl("Office Hours", details_text))
oh_idx <- ifelse(length(oh_idx) == 0, NA, oh_idx) #if empty, assign NA
if(!is.na(oh_idx)){
```

```
    faculty_df$office_hours[i] <- details_text[oh_idx]
  }

  # Phone ---
  # Find element with our text skills
  phone_idx <- which(grepl(pattern = "574", x = details_text))
  phone_idx <- ifelse(length(phone_idx) == 0, NA, phone_idx) #if empty, assign NA
  if(!is.na(phone_idx)){
    faculty_df$phone[i] <- details_text[phone_idx]
  }

  # Office location ---
  # Element before the phone number
  if(!is.na(phone_idx)){
      faculty_df$office_location[i] <- details_text[phone_idx-1]
  }

  # What if no office location listed?
  # If phone and office hours listed,
  # but next to each other, then
  # the office location is omitted.
  # Admittedly convoluted solution :)
  if(all(!is.na(c(phone_idx, oh_idx)))){
      if(phone_idx-1 == oh_idx){
        faculty_df$office_location[i] <- NA
    }
  }
}
write.csv(faculty_df, file = "faculty_df.csv")
```