Homework - Week 13

Write your name here

2025-11-20

Preface

The goal of this assignment is to help you gain familiarity with web scraping. As always, please come to office hours and reach out to your teaching staff if you have any questions.

Data

We will get data for this assignment from the web.

1. Let's warm up by building on our work to scrape stock prices from this week's example. We want to scrape Goldman Sach's historical daily share price from https://stooq.com/q/d/?s=GS.us. However, the site will temporarily block us from accessing the content if it receives too many requests (from multiple students). So instead we'll use a copy of site's html code saved in `stooq/GS.html`. Use `read_html()` to read it in. Extract the table using the selector `"table#fth1"` and convert it to a data frame. Clean the data, converting Date to a date and all other variables to doubles. Print the first 10 rows of the resulting data frame. What is the earliest date for which prices are available?

Note: We're providing code for this problem to get you started, but you can also review example-13-solutions on posit cloud!

```
gs_prices <- read_html("stooq/GS.html") |>
  html_element("table#fth1") |>
  html_table(header = TRUE) |>
  select(-`No.`, -Change) |>
  type_convert() |>
  mutate(Date = dmy(Date))

gs_prices
```

```
# A tibble: 40 x 6
   Date        Open  High   Low Close  Volume
   <date>     <dbl> <dbl> <dbl> <dbl>   <dbl>
 1 2025-11-20  795.  816   772.  774. 2597161
 2 2025-11-19  776.  788.  775   786. 1521113
 3 2025-11-18  769.  788.  767.  776. 1673456
 4 2025-11-17  789.  797.  770.  776. 1850550
 5 2025-11-14  796.  801.  778   791. 2552397
 6 2025-11-13  833.  841.  805.  806. 2557357
 7 2025-11-12  814.  840   814.  839. 2954130
 8 2025-11-11  798.  812   796.  810. 1521454
 9 2025-11-10  796.  806.  789.  797. 1698196
10 2025-11-07  783   787.  763.  786. 2098378
# i 30 more rows
```

The earliest date for which prices are available is...

2. Now generalize your code from question 1 into a function that takes a ticker as an input and returns historical daily prices. As before, clean the data and convert Date to a date and all other variables to doubles. Have the function return the data frame. Test the function with a ticker of your choice, so that it prints the first 10 rows of the data frame below.

Note: Normally for web scraping we would write a flexible url that depends on the argument provided to the function. In this case, we'll use the argument to read in the appropriate html file.

3. Adapt your function from question 2 to create a variable `symbol` in the data frame that contains the value of the argument `ticker`. Use your revised function in conjunction with `map()` to scrape prices for several of the top employers for Dyson graduates: Bank of America, Barclays, Capital One, Citigroup, Goldman Sachs, J.P. Morgan, Lazard, and Morgan Stanley. Assign the resulting list to `prices_list`. Use the function `bind_rows()` to combine the list of data frames into a single data frame (i.e., write `bind_rows(prices_list)`). How many rows are in the data frame?

There are … rows in the data frame.

4. Tickers can be pretty hard to decipher. Modify your function to scrape company names from using the selector `"#f18"`, and store it in a variable `company` in the data frame your function returns. Remove the leading text `"Historical data: "`, the trailing ticker in parentheses, and any whitespace to get just the company's name. Use the revised function to create a data frame for the companies from question 3. How many columns are in the data frame?

Note: It's best practice to minimize requests to websites when scraping, so please only `read_html()` once within the function.

There are … columns in the data frame.

5. Use the variable `Open` in the data frame from question 4 to compute cumulative returns for each company over the period of data you scraped. Plot the amounts in a graph with the best performing stock first and worst performing stock last, labeled using company names (not tickers).

6. Use the variable `Open` in the data frame from question 4 to plot share prices over time for each company in facets using the option `scales = "free_y"`. Make sure the facets are labeled with company names (not tickers).

Just for fun

Note: These two questions are not required as part of this homework, and they may be difficult to complete if the site is sensitive to the number of queries it receives from the Posit Cloud server. They will not factor into your grade either way (positive or negative). We're just leaving them here in case you want more practice.

Now let's try to scrape data over a custom date range. Go back to the historical data and use the header to customize the frequency to Monthly and time period from Nov 20, 2020 to Nov 20, 2025. Click Show. Scroll down to the bottom of the page and right click on the link "Download data in csv file..." to copy the link. Paste the updated url into the code chunk below. Modify your code from question 4 to use this link in conjunction with `read_csv()` in order to to create a data frame of monthly prices for each ticker. Finally, plot the Open prices over time for each company in facets using the option `scales = "free_y"`.

What if we want to focus on a different time period? Customize your "just for fun" function to take three arguments: the ticker, start date, and end date (in date format). Use the function to make a data frame of monthly prices for the ticker `GS` from January 1, 2000 through December 31, 2024, and make a plot of Open prices analogous to the one from question 7 (but for just Goldman Sachs).