ECON 370: Assignment 4

Drew Van Kuiken

Due Wednesday, November 20, 2024

Note: **AI and LLMs are not allowed on this assignment.**

Part One: Server-side Webscraping (Total Points: 75)

We're going to kick things off by scraping a fake bookstore. Note that this website is designed for pedagogy, so it should theoretically be prepared for you to try to scrape it. Feel free to ping it to your heart's content.

Our website is `https://books.toscrape.com`. If you navigate to the homepage, you should see a navigation bar on the left with various book genres. The main portion of the page shows 20 different books "for sale." Each book title is a link to a separate landing page with more detail about that book. If you click on, for example, *A Light in the Attic*, you'll see what one of these landing pages looks like. There's lots of information in here. I want to call your attention to 3 objects: the Book's name, the Book's description (a paragraph of text in the middle of the page), and the Product Information table at the bottom of the page.

Your goal in this question is to scrape each of those elements for all 1,000 books in the website's main collection. This means you must scrape information from each of the 50 pages that pops up when you click next page. You do not have to use the navigation bar on the left at all. Note that any piece of information that is missing for a given book can be left blank.

Your output should consist of a dataframe where each row represents one book that the website "sells." You should have 9 columns: name, description, upc, product_type, price_excl_tax, price_incl_tax, tax, availability, and number_of_reviews.

Grading:

- Correctly scrapes information on book names, book descriptions, and product information (15 pts each)
- Correctly scrapes links to each book (15 pts)

- Correctly scrapes information for at least one page on the bookseller's website (10 pts)
- Correctly loops through all pages to get information (5 pts)

Hints:

- Internal links on this website are *relative*. To get R to follow a link, you'll need to paste the relative link to an absolute link (i.e., a link to the website itself).
- You can store dataframes in a list. To pre-allocate a list that can store dataframes, you'll need to use the following command: vector(mode = "list", length = XXX) which will create an empty list for you.
- **bind_rows** from **dplyr** will call rbind on all elements of a list.

Part Two: Close Calls (Total Points: 40)

NASA has a cool database called the small-body database. It contains all of the small-bodied asteroids that NASA has tracked and which have flown close to the Earth or other planets in the solar system. You can find a link to the api documents for this database here: `https://ssd-api.jpl.nasa.gov/doc/cad.html#cad_body_table`.

For this question, I'd like you to query the small-body database API. Note that you do not need an API key to query this database.

Use default parameters unless otherwise specified.

1. Using the total-only parameter and other parameter options, determine the max number of observations that we can retrieve from this API. (10pts)
2. Pull all observations up to a max distance of 100aus and for all available dates, store them in a data frame. (10pts)
3. Create a graph that shows, for each object which passed by Earth in 2024, the relationship between the size of the object, its relative velocity, and its nominal approach distance from Earth. (10 pts)
4. Does relative velocity increase or decrease with asteroid size? Provide evidence to support your conclusion. (10 pts)

Part Three: Fiber Expansion in NC (Total Points: 65 pts)

For this question, we'll expand upon our analysis of the Google Fiber Rollout data from class. We're going to use data from 3 sources: first, the Google Fiber website for all of

NC; second, we'll use the Zillow average rent data from class; and finally, we'll use zipcode level shapefiles from the US Census for mapping purposes.

Our goal in this analysis is to assess whether Google Fiber has rolled out services to fancy neighborhoods first. Please complete the following steps:

- Access the hidden API for Google Fiber in the Triangle and in Charlotte, NC. Perform GET request and save data for each area in a dataframe. (5 pts each)
- Read in Zillow and zipcode data:
  1. Read in the Zillow average rent data (5 pts)
  2. Using the **zctas** function from the **Tigris** package that we covered in class, download zipcodes for the state of North Carolina from the 2010 census (5 pts)
  3. Merge the Zillow rent data and the zipcode data (5 pts)
- Create a graph of all zip codes in North Carolina using **geom_sf**. Color zip codes based on average Zillow rent in that zip code from September 30, 2024. Add a scatterplot layer which has data on Google Fiber status in apartment buildings, where points are colored based on whether Google Fiber is available in an apartment building. (30 pts)
- Create a graph with the above features, but which shows data only for zip codes that appear in the Google Fiber Charlotte API results. (5 pts)
  - Hint: to do this, you'll probably want to merge the Zillow-zipcode data and the Google Fiber data for Charlotte
- Create a graph with the above features, but which shows data only for zip codes that appear in the Google Fiber API results for the Triangle. (5 pts)