Web scraping for public health

Workshop for DLSPH 15th Annual Student-Led Conference
November 19, 2022

Jean-Paul R. Soucy (jprs.me)
PhD candidate in epidemiology
Dalla Lana School of Public Health

Requirements for this workshop

To participate in the interactive portion of this workshop, you should have installed R and RStudio on your computer and possess a basic working knowledge of R.

For help installing R and RStudio:
https://rstudio-education.github.io/hopr/starting.html

Requirements for this workshop

To follow along with the code in the interactive portion of this workshop, please download the required files from GitHub:

https://github.com/jeanpaulrsoucy/dlsph-student-conf-2022-web-scraping-workshop (short link: https://tinyurl.com/dlsph-scraping)

Click the big green “Code” button then “Download ZIP”. Save the ZIP file to your desktop and extract.

You must also **install the packages** listed in *install-required-packages.R*.

Outline

1. What is web scraping?
2. Project 1: Canada's international vaccine distribution
3. Project 2: Who owns Ontario's long-term care homes?
4. Project 3: *Toronto Star* headlines throughout the pandemic
5. Wrap-up
6. Q&A

What is web scraping?

- The Internet contains an enormous amount of information, but rarely is it in a format suitable for easy re-use or analysis.

- Web scraping is the process of extracting data from websites.

- While a variety of (expensive) “no code” tools exist, the easiest way to get started with web scraping is using the free programming languages R or Python.

What is web scraping for?

- Web scraping is used to automate the collection of data that would be tedious or impossible to collect manually.

- For example, I use web scraping to collect COVID data from dozens of sources to maintain the pan-Canadian COVID-19 Canada Open Data Working Group dataset: https://github.com/ccodwg/CovidTimelineCanada

How do I get started with web scraping?

- The easiest packages for web scraping are *rvest* in R and *Beautiful Soup* in Python.

- Basic knowledge of how websites work is essential. Your web browser’s developer tools are your best friend (right click on a webpage → Inspect).

- Today, we will go through three projects to demonstrate the fundamentals of scraping basic HTML websites.

Setting up RStudio for today’s workshop

- If you haven’t already, download the required files from GitHub as described on the slide “Requirements for this workshop”

- Open the project in RStudio: Project → Open Project… → *dlsph-student-conf-2022-web-scraping-workshop.Rproj*

- Make sure you have the required packages installed by running the script *install-required-packages.R*

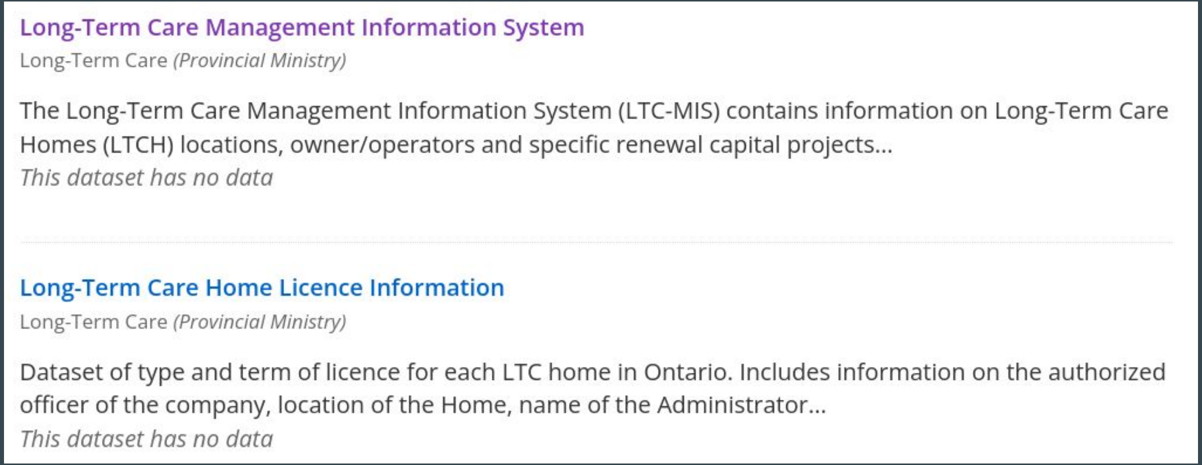
Project 1: Canada's international COVID-19 vaccine distribution

The following website provides data on Canada's international donations of COVID-19 vaccines: https://www.canada.ca/en/public-health/services/diseases/coronavirus-disease-covid-19/vaccines/supply-donation.html

However, there is no option to download the data from the table detailing the distribution of Canada's surplus vaccine doses. In this project, we learn to extract an HTML table using the function *rvest::html_table*.

Project 2: Who owns Ontario's long-term care homes?

The Ontario Data Catalogue provides no information on the ownership of the province's long-term care homes: https://data.ontario.ca/en/dataset?organization=long-term-care



Long-Term Care Management Information System

Long-Term Care *(Provincial Ministry)*

The Long-Term Care Management Information System (LTC-MIS) contains information on Long-Term Care Homes (LTCH) locations, owner/operators and specific renewal capital projects...

This dataset has no data

Long-Term Care Home Licence Information

Long-Term Care *(Provincial Ministry)*

Dataset of type and term of licence for each LTC home in Ontario. Includes information on the authorized officer of the company, location of the Home, name of the Administrator...

This dataset has no data

Project 2: Who owns Ontario’s long-term care homes?

The Ministry of Long Term care maintains a website with information on ownership, inspections and other data: http://publicreporting.ltchomes.net/en-ca/Search_Selection.aspx

However, each home has a separate webpage. In this project, we learn to extract information spread out across many different webpages and assemble it into a single dataset.

Project 3: *Toronto Star* headlines throughout the pandemic

The Internet Archive's Wayback Machine captures "snapshots" of what a website looked like at a specific date and time in the past: https://web.archive.org/

The Wayback Machine API lets us request the closest "snapshot" of a webpage to a given date and time:

https://archive.org/help/wayback_api.php

Project 3: *Toronto Star* headlines throughout the pandemic

The main page of the *Toronto Star* has been captured thousands of times in the Wayback Machine. This means we can see what the headlines were on any given day:

https://web.archive.org/web/20220000000000*/thestar.com

For example, here is the main page on the day Ontario declared a state of emergency over COVID-19:

https://web.archive.org/web/20200317181619/https://www.thestar.com/

Project 3: *Toronto Star* headlines throughout the pandemic

In this project, we will extract "front page" (editor's picks) headlines on *thestar.com* from snapshots taken for each day of October 2020, October 2021 and October 2022 and use pattern matching to compare the fraction of headlines that may be related to the COVID-19 pandemic.

Some considerations when web scraping

- Before scraping a website, check to see if they have downloadable datasets or an API that will get you the data you need.

- Be polite. If making a large number of requests or downloading a lot of data, try to space out your queries.

- Consider if what you are doing is legal and ethical.

Advanced web scraping

- The projects we looked at today dealt with static HTML websites. However, many websites generate content dynamically using JavaScript. Many “fancier” dashboards do this, but it makes data harder to extract.

- Dynamic websites require additional tools to collect data before you can begin parsing it. *Selenium*, which has interfaces in both R and Python, is a good tool to learn.

Conclusion

- In today’s workshop, we learned the basics of web scraping using R and went through three example projects.
- If you regularly work with data from the web, there’s a good chance web scraping can be used to make your job easier.
- “Inspect” is your best friend!

Questions?

Jean-Paul R. Soucy

Website: jprs.me

Twitter: @JPSoucy

Email: jeanpaul.soucy@mail.utoronto.ca

GitHub: https://github.com/jeanpaulrsoucy