Web Data Collection with R

Session 9: Collecting Data from Dynamic Websites

Sascha Göbel
sascha.goebel@uni-konstanz.de

January 13th, 2021

Today: Dynamic Websites

- Handling Forms
- AJAX
- Automating a browser session with Selenium

Handling HTML Forms

User-server interaction

- forms occur with search bars, username and password inputs
- recognizable by <form> or <input> elements
- server-side dynamic
- collect user input and return response from server
- via http
- easily dealt with using functions from the `rvest` package

AJAX

Client-side dynamic

- initial source code still contains all data
- page modified by browser

Asynchronous JavaScript and XML

- client- and server-side dynamic
- content is iteratively loaded in the DOM upon request (e.g., scrolling, clicking, or hovering over specific elements)
- without reloading the page
- previously learned methods for web data collection do not work here

Detecting JavaScript

- code between <script> tags
- external code referenced by ‘src’ attribute of ‘script’ element
- in an attribute of an HTML element
- programming language is described in ‘type’ attribute

Selenium

Selenium WebDriver

- standalone Java program
- built for web application testing
- automates a session in the browser
- to show and make content in the live DOM accessible

`RSelenium`

- R binding for Selenium 2.0 WebDriver
- allows to render dynamic content from R
- then collect via `rvest`

Install Selenium WebDriver

- download Java
- Download Selenium standalone server at `http://selenium-release.storage.googleapis.com/index.html?`,
- pick 'selenium-server-standalone-3.9.1.jar'
- in command line:
  - navigate to where the file is stored
  - run 'java -jar selenium-server-standalone-3.9.1.jar'
- for further information see `https://cran.r-project.org/web/packages/RSelenium/vignettes/basics.html`
- make sure to grant firewall exception