Advanced Web Scraping: Playwright

CS 203: Software Tools and Techniques for AI

Prof. Nipun Batra, IIT Gandhinagar

Part 6: Playwright - JavaScript-Heavy Sites

When requests/BeautifulSoup isn't enough

The JavaScript Problem

Problem: Many modern sites load data dynamically with JavaScript

Example: Netflix loads movie thumbnails after page loads

When you use `requests.get()`:

- You get initial HTML
- JavaScript hasn't run yet
- Data you want isn't in the HTML!

Solution: Use a real browser that runs JavaScript

What is Playwright?

Playwright: Library to control a real browser programmatically

Features:

- Supports Chrome, Firefox, Safari
- Runs JavaScript, waits for elements
- Can screenshot, interact with page
- Much slower than requests!

When to use: Only when necessary (JS-heavy sites)

Installing Playwright

```
pip install playwright
playwright install chromium
```

This downloads Chromium browser (100+ MB)

Basic Playwright Example

```python
from playwright.sync_api import sync_playwright

with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page()
    page.goto("https://www.example.com")
    page.wait_for_selector('h1')
    html = page.content()
    browser.close()
```

Playwright with BeautifulSoup

```python
from playwright.sync_api import sync_playwright
from bs4 import BeautifulSoup

def scrape_js_site(url):
    with sync_playwright() as p:
        browser = p.chromium.launch(headless=True)
        page = browser.new_page()
        page.goto(url)
        page.wait_for_selector('.movie-card')
        html = page.content()
        browser.close()
    soup = BeautifulSoup(html, 'lxml')
    return soup
```

When to Use Playwright vs requests

Use requests + BeautifulSoup:

- Simple static websites
- Speed is important
- Scraping many pages

Use Playwright:

- JavaScript-heavy sites
- Need to interact (click, scroll, type)
- Data loads after page load

Rule of thumb: Try requests first, use Playwright if needed