Web Scraping: Python Data Playbook

SETTING UP BEAUTIFULSOUP

Ian Ozsvald
INTERIM CHIEF DATA SCIENTIST

@ianozsvald   ianozsvald.com

Overview

Scrape HTML with the requests module

Parse the HTML with BeautifulSoup4

- Check for success
- Inspect the processed HTML

Why Scrape Web Pages?

Storage and Queries

Data Augmentation

Analysis and Communication

Dynamic vs. Static Websites

Static pages

Easy to scrape

Data dumps are often static

Requests and BeautifulSoup4

Dynamic pages

More difficult to scrape

Modern sites often dynamic

Requires a tool like Selenium

Demo

Attribution for “auto-mpg” dataset

Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Demo

Serve HTML with Python’s http.server

What relationships might we see?

- Weight and MPG?
- Cylinders and Displacement?
- MPG improvements over time?

Demo

Interactive Python Shell (IPython)

Fetching HTML

Check our content

Web Scraping Strategies

Processing Online

Easy to develop

Great for fewer pages

Use for research

Processing Offline

More complex

Great for larger volumes

Use for engineered solutions

Demo

Dynamic vs. static HTML scraping

Fetch two pages using requests module

One will be mostly empty!

Verify using wget – great for debugging

Summary

Web page successfully downloaded

Parsed using BeautifulSoup4

We are ready to extract elements