Data Collection and Modeling

Course 6 - Web Scraping

Definition

“Web scraping is a term for various methods used to collect information from across the Internet. Generally, this is done with software that simulates human Web surfing to collect specified bits of information from different websites. Those who use web scraping programs may be looking to collect certain data to sell to other users, or to to use for promotional purposes on a website.

Web scraping is also called Web data extraction, screen scraping or Web harvesting.” (Techopedia)

“the extraction and copying of data from a website into a structured format using a computer program” (dictionary.com)

Is it legal?

The action of web scraping isn't illegal if certain rules are followed.

It becomes illegal when extracting non publicly available data.

Data on the internet is considered publicly available if:

- the publisher stated that data is public.
- data access doesn't involve an account creation nor login.
- the robots.txt file for the website doesn't block web scrapers or spiders.

(Legality article)

Is it ethical?

Depends on the purpose of the action; it is considered to be ethical when extracting data from websites that are public and using it without harm and invasion of privacy.

- Request strictly necessary data
- Use a reasonable rate for requests
- Use a User-Agent string
- Use data only to create new data/information

How Web Scrapers Work?

1. Make a HTTP request to a web server: define the URL or set of URLs that the scraper needs to load; the scraper loads the HTML (some advanced scrapers can load also CSS and Javascript elements)
2. Extract and parse web server’s response (website code): it can extract all the data or data selected by the user
3. Save extracted data in desired format: most scrapers can output data in csv or spreadsheet (Excel, OpenOffice) format, but also in JSON

Web Scraper Types

There may be several types of such software; they can be grouped based on 4 main perspectives:

- Pre-built or custom-built
- With or without (comprehensive) user interface
- Software or browser extension
- Local running or cloud based

Pre-built or custom-built

There are a plethora of web scrapers ready to be downloaded (free or commercial ones), with options that would suit all needs. They can allow inexperienced users to just click and select site components they want to scrape, others need some technical knowledge to define how and what to select (Winautomation vs Easy Web Extract).

As the above ones were created, anyone can create its own web scraper provided that has some advanced programming and communication protocols knowledge; more features needed, deeper knowledge is requested.

Custom-built

In order to build a web scraper, there are some generic steps to be followed:

- Define the URLs to be used
- Inspect the page (view page source)
- Identify the components to be extracted
- Write the code (use a library for good productivity)
- Execute the code
- Store the data

User Interface

The range of features offered to the users can vary widely:

- Minimal UI with command line; appropriate for experienced users that use command line regularly
- Powerful UI with entire site rendered so the user can click the components it wants to scrape
- Suggestions and tips to guide the user during the actions to be performed

Software or Browser Extension

Depending on the need for the scraper to be integrated with the browser or not, we can identify:

- Browser extensions; offer features for ad blockers, themes, etc. They can be lighter and simpler to run. They are limited to what a browser can offer.
- Stand alone application: can be downloaded and installed locally, offering advanced features like IP rotation, multithreading, scheduling

Local Running or Cloud Based

The scraping session can run from local computer or from cloud.

- When running locally, it uses the resources of local system and is limited to local network card and connection bandwidth; when having large set of URLs, it may have impact on system’s performance for a while
- Some companies provide scraper that run directly in their cloud environments; the impact on local system is nonexistent; advanced features can be provided (IP rotation, multithreading, etc)

Web Scrapers Use

There are several use cases that span across different industries:

- Price monitoring
- Sentiment analysis
- Real estate listings consolidation
- Lead generation
- Market research
- Email marketing

Scraping and Security

Data scraping can be used to expose sensitive data. The website may not be aware of its data being collected or the scraper may not secure stored data, being accessible by attackers.

- Phishing attacks
- Password cracking

Scraping Techniques

- HTML Parsing: JavaScript is used to extract text and links from HTML pages
- DOM Parsing: it describes the structure and content of an XML; used to access nodes with information of interest
- XPATH: query language for XML documents; can be used to select nodes; can be combined with DOM parsing
- Vertical Aggregation: used to scrape certain business domains (verticals) and aggregate this information; usually they use massive computing power (cloud)

Mitigating Web Scraping

In order to reduce the amount of data that is scrapable, there are few techniques to be used:

- Limit user requests rate: limit the number of requests made from same IP per time interval according to the maximum rate a human is capable to perform (1 page in few seconds vs 100 pages per second)
- Modify HTML regularly: randomized HTML elements modifications
- Apply CAPTCHA: add a task completion easy to perform for humans
- Embed content in media objects: insert content as image instead of text

Robots.txt

A file that is used by crawlers to know what they can access on that specific website.

It manages the crawler traffic based on 3 file types:

- Web page: blocks crawling to specific page, but URL can still be reached via search results; non HTML pages are excluded
- Media file
- Resource file

Robots.txt

It is placed in the root folder of the website.

Consists of at least one rule.

Each defined rule allows or blocks access for specified crawlers to given paths.

All files are implicitly accessible for crawling.

Can be consulted by accessing the website URl and appending /robots.txt to it.

Robots.txt Rules

- Allow access to a single crawler

```
User-agent: My-crawler
Allow: /
User-agent: *
Disallow: /
```

- Allow access to all except one crawler

```
User-agent: My-crawler
Disallow: /
User-agent: *
Allow: /
```

- Disallow crawling for the entire website

```
User-agent: *
Disallow: /
```

- Disallow crawling for an entire folder

```
User-agent: *
Disallow: /events/
Disallow: /personal/work/
```

Robots.txt Rules

- Disallow crawling for a specific web page

```
User-agent: *
Disallow: /myfile.html
```

- Disallow crawling for the entire website except a subfolder

```
User-agent: *
Disallow: /
Allow: /public/
```

- Disallow crawling for files of a specific file type

```
User-agent: *
Disallow: /*.png$
```

- Disallow crawling for URLs that match a string with wildcards

```
User-agent: Googlebot
Disallow: /*.xls$
```

Tutorials

Webscraper.io

Bardeen.ai