Web Scraping

Thomas Schwarz, SJ

Important Preliminaries

- On your own machine:
  - Install pip3 (the python 3 version)
    - You can invoke pip3 also by python3 -m pip
  - Then install a number of packages:
    - pandas
      - sudo python3 -m pip install pandas
    - beautifulsoup4
      - sudo python3 -m pip install beautifulsoup4
    - requests

Scraping and Crawling

- Both involve automatic ('bot') access to a web-site
- Crawling tries to find and process all the information on all pages of the website
    - Typically used by search engines
- Scraping
    - Used to obtain data contained in certain web-pages

Legal and Ethical Issues

- Web-scrapping is sometimes considered a threat
  - Because it creates real problems
  - Because it accesses data for use against the business interests of the web service provider

Legal and Ethical Issues

- Web-scraping can run afoul of:
    - Existing and future laws
        - In the US:
            - Computer Fraud and Abuse Act, Digital Millennium Copyright Act,
    - Terms of Use / Breach of Contract e.g. those in robots.txt
    - Copyright
    - ...

Legal and Ethical Issues

- robots.txt gives conditions for automatic crawling
  - No crawling:

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

  - All crawling allowed:

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

  - Block twitbot from crawling the indicated directory

    ```
    User-agent: twitbot
    Disallow: /mysecrets/
    ```

Legal and Ethical Issues

- robots.txt
  - Needs to be called that (not Robots.txt)
  - Needs to be placed in the top-level of the hierarchy
  - needs to be publicly available
  - subdomains will have to use separate robots files
  - Can be used to provide a sitemap for crawlers (so that search engines will show your content)
    - Sitemap: `https://www.mysite.com/sitemap.xml`

Legal and Ethical Issues

- Aggressive scraping (and crawling) can become a Denial of Service Attack
    - Server busy to answer scraping demands and cannot serve other traffic
    - robots.txt can specify a desired back-off interval
        - In general: do not access web-pages on a site without an interval of at least 10 seconds

Legal and Ethical Issues

- Many sites provide APIs in order to allow users to make bulk-downloads of data
    - This usually means they do not want to have their site scraped, so they offer a simpler alternative

Legal and Ethical Issues

- Raw data is not protected by copy-right
- Exceptions can arise when scraping is used to obtain the same functionality as the original site
- Scraping needs to be done at a low level of intensity
- Using an agent that sends identifying information with each request is useful
    - Security pouring over logs can be put at ease with an explanation

Legal and Ethical Issues

- Websites are free to ban robots by using a black-list for IP addresses
    - Commercial crawling solutions exists that circumvent banning
        - Imitate human user behavior
        - Use many different IP addresses
        - Automatic throttling of requests
- The need and the existence of these automated crawlers show that:
    - Scraping is in a legal and ethical gray-zone

Techniques

- To download data from a website and prepare it for processing
  - We need to access the website
  - We need to find the data on the website and put it into a structure we can use
- Before we code, we need to first understand the source of the website
- After we obtained the data, we need to store it in a reasonable format

Understanding web sites

- Access the target website
- Use the developer tools or view the source
  - Browser dependent

Accessing web sites

- Selenium: Module for automatic web application tests
  - Automatically click links, pretend to be a certain browser, etc
  - Useful when data is accessed after ajax requests
  - Needs some downloads

Accessing Website

- Scrapy:
  - Framework to run scraping and web crawling
  - Developed by web-aggregation and e-commerce company Mydeco
  - Maintained by Scrapinghub
  - Interlaced with a commercial offering

Accessing websites

- Requests
  - Simple and basic translator for making url requests
    - `r = requests.get(address)`
    - Variable `r.content` now contains the contents of the web page (as a binary string)
    - Variable `r.text` contains the contents as a string
      - Requests will guess the encoding
      - But you can set the encoding with
        - `response.encoding = 'utf-8'`

Accessing websites

- Requests
  - Can use `r.headers` to obtain a dictionary-like object with various header values
  - Can use query string in requests:
    - Example:

```
requests.get('https://api.github.com/search/rep',
          params=[('q', 'requests+language:python')])
```

```
requests.put('https://httpbin.org/put',
          data={'key':'value'})
```

Understanding Websites

- Before we start downloading websites, let's first understand them
    - Each web browser has a way to view the source of a website
    - On Chrome, use Developer -> View Source
        - Easiest tool for web development

Understanding websites

- Inside the page source, find the data that you are interested in