Workplace Activity 04

Web Scraping + AI

Introduction

Web scraping is the process of extracting data from websites. This is typically done through automated software that simulates human web surfing to collect specific pieces of information from web pages. It's a process / concept that's been around since the beginning of the World Wide Web.

Here's how it usually works:

1. **Crawling:** The software, often referred to as a crawler or a spider, accesses the website to read its data. It might scan through multiple pages of the website, following links to access different sections. Google and every other search engine rely on crawler processes to collect the information that makes their algorithms useful.
2. **Parsing:** The software then parses the HTML content of the pages to identify specific data elements. This might involve looking for specific tags, attributes, or content patterns.
3. **Extraction:** The identified data is extracted from the page and typically stored in a structured format such as a CSV file, a database, or a spreadsheet.
4. **Data Processing:** After extraction, the data can be cleaned, processed, and analyzed depending on the requirements.

Web scraping is used in various fields like market research, price monitoring, real estate listings, social media analysis, and more. It's important to note, however, that web scraping can raise legal and ethical issues, particularly concerning data privacy, security, and violating terms of service of websites. Thus, it's crucial to be aware of the legal landscape and ethical considerations before engaging in web scraping.

Examples of Legal / Ethical v. Illegal / Unethical

Legal / Ethical	Illegal / Unethical
Academic Research	Copyright Infringement
Competitive Analysis	Violation of Terms of Service
Public Data Aggregation	Data Theft: Using web scraping tools to extract personal or proprietary information without consent, such as harvesting personal details from social media profiles or stealing trade secrets from business websites, is both illegal and unethical.

How does this relate to AI?

Web scraping relates to AI in (at least) two ways.

A. LLM models are usually trained (at least in part) from data scraped from the web.
B. We can use ML models to evaluate the data returned from scraping.

IMPORTANT NOTE: It seems to me that the "parsing" operation (see step 2 in the prior section) seems to rely heavily on the correct semantic use of HTML tags (that is to say, the web page must have paragraph text inside a (paragraph tag) and header text inside a <h1></h1> (header tag) and so on. A natural next step in the

evolution of web scraping will be to use AI to literally read web page like a human would, without regard for the correct semantic use of the HTML tags.

As AI integrates more deeply into web scraping, it will likely make the process more robust and less dependent on specific HTML implementations.

This would be a third and very important way in which AI relates to Web Scraping.

However, this approach will bring challenges:

- **Ethical and Legal Concerns:** As AI becomes more capable of extracting information from web pages, the potential for misuse increases.
- **Complexity and Resource Requirements:** AI-driven web scraping can be more resource-intensive and complex to implement compared to traditional methods.
- **Resistance from Web Hosts:** Websites may implement more sophisticated measures to detect and block AI-driven scraping.

Let's give it a try

First, let's see an "old school" technique for grabbing the content from a public website. Your instructor will demonstrate the use of HTTrack. HTTrack is a free and open-source Web crawler and offline browser, initially released in ***1998.*** Yes, this thing is a little over 25 years old and is still being maintained!

Now it's your turn:

1. Visit https://tinyurl.com/shaferaicourse
2. Download the notebook called webscraping.ipynb found in the workspace4 folder.
3. Open up Anaconda Navigator / Jupyter notebooks.
4. Together as class we will run the script and scrape the news found on https://cnn.com and https://www.theonion.com
5. DISCUSS – which of these option best describes what we are doing here?
    a. LLM models are usually trained (at least in part) from data scraped from the web.
    b. We can use ML models to evaluate the data returned from scraping.
    c. Using AI to read the web page (like a human would)
6. DISCUSS – How might a business or organization use this technology?
7. DISCUSS – How might you go about finding the most suitable pre-trained transformer?
8. DISCUSS – Based on what we saw in the last few classes, is there another way we could have done this?

Other important notes

This script uses an AI model called https://huggingface.co/jy46604790/Fake-News-Bert-Detect

There are many, many free-to-use AI models available at huggingface.co (the same folks who give us Mistral AI)

Overall, the script scrapes headlines from a given URL, classifies them as "real" or "fake" using this pre-trained model, and writes the classified headlines to a CSV file.

IMPORTANT: In our script the classification action is being run ***locally***. The script downloads the pre-trained model and tokenizer from the Hugging Face model hub and loads them into memory on the local machine. No data is sent to a remote server for classification.

Here's a high-level summary of how our script works:

1. The script imports the necessary libraries, including `requests`, `BeautifulSoup`, `torch`, `transformers`, and `csv`.
    a. `requests` is a popular Python library used for making HTTP requests.
    b. `BeautifulSoup` is a Python library used for parsing and manipulating HTML and XML documents. It's perfect for "traditional" web scraping.
    c. `torch` is short for PyTorch. PyTorch is a library developed by Meta that provides a way to represent and manipulate data in a format that's suitable for machine learning.
    d. `transformers` is a library developed by Hugging Face for working with transformer-based models.
    e. `csv` is a built-in Python library for working with CSV (Comma-Separated Values) files.
2. The `scrape_headlines` function is defined, which takes a URL as input and uses `requests` and `BeautifulSoup` to extract headlines from the HTML content of the webpage at that URL.
3. The `classify_headline` function is defined, which takes a headline string, a pre-trained model, and a tokenizer as input, and uses the model to classify the headline as either "real" or "fake".
4. The `main` function is defined, which first prompts the user to enter a URL to scrape. It then calls the `scrape_headlines` function to extract headlines from the webpage at that URL.
5. The `main` function loads a pre-trained model and tokenizer.
6. The `main` function loops through each headline in the list of headlines extracted from the webpage and calls the `classify_headline` function to classify it as "real" or "fake".
7. The classified headlines are written to a CSV file using the `csv` library.