Web Data

1405 Instructor: Ruiqing (Sam) Cao

Required Python Libraries for Today

Main libraries: json, requests, csv, Beautiful Soup (bs4), pyjsonviewer

```
import requests
from bs4 import BeautifulSoup
import pyjsonviewer
import json
import csv
```

Recurring libraries (we'll see a lot more of later): numpy, pandas, matplotlib

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
```

If a `module` is not yet installed, you can type `%pip install module` inline in your Jupyter Notebook to install it

Collecting Web Data: Formats

From cleaner (structured) to messier (unstructured):

- Data dumps: usually well-formatted CSV files, or *.parquet which is common for larger data sets
- APIs (requests and response): output format is often JSON
- HTML raw page source: tree structure through opening/closing tags (e.g., <p>), attributes and elements contain useful data

Read Data in Pandas

- Pandas provides a powerful library for directly ingesting many types of data (e.g., tabular, JSON) from a file

```
>> import pandas as pd
>> data = pd.read_json('example.json')
```

- Similar for other common file types, e.g., *.csv, *.parquet, *.dta, *.xlsx, etc.

Read & Write CSV Files

- Read the CSV file line by line using the Python `csv` module

```
>> import csv
>> with open('crunchbase_europe_2016_2020.csv','r') as f:
>>     f_csv = csv.reader(f, delimiter=',')
>>     for row in f_csv:
>>         print(row)
```

Read & Write JSON Data

Suppose you have some raw JSON data stored in `raw_string` which is just a piece of text that satisfies the formatting requirements for JSON ***parsing*** (e.g., it looks like nested lists and dictionaries)

- To load the data into a JSON object `data`

```
>> import json
>> data = json.loads(raw_string)
```

- To convert the JSON object into

```
>> raw_string = json.dumps(data)
```

- HTML/XML parsing quite similar (follows the same tree structure broadly defined), but may require additional “data janitorial” work

JSON Objects: Basic Rules

- JavaScript Object Notation (JSON) is a simple and flexible text-based format for representing semi-structured data
- JSON is widely used in web development for data exchange between applications: Data retrieved from API calls are often in JSON format
- A raw JSON string is parsed into a JSON object, which *roughly* consists of nested dictionaries (with some differences) and lists

JSON Objects: Basic Rules

JSON data are represented by *key-value pairs* and *elements in arrays*, separated by *commas*

- Keys are always strings and placed in double quotes
- Values can be strings (e.g.,"text"), numbers (e.g.,123, 12.3), Booleans (true/false), null value (null), objects (e.g., nested dictionaries {"key":"value"}), arrays (e.g., lists of values [v1,v2])

➔Curly Braces {}: Represent a dictionary

➔Square Brackets []: Represent a list/array


```
{"name":"Astrid","age":35,"is_student":false,"skills":["Python","Java"],"address":{"city":"Stockholm","zip":"10055"},"phone":null}
```

JSON Objects vs. Dictionaries

JSON strings are stored in text format and look like `dict()` in Python, but they are fundamentally quite different:

- JSON can only contain double quotes "", but ***not single quotes '***
- ***Tuples*** cannot be part of a JSON string (almost anything else is fine: lists, nested dictionaries, strings, Booleans, numbers)
- Boolean values must be written as ***true*** and ***false***, and missing values as ***null*** (not True/False and None)

Web Data Sources: Structured Databases

Publicly listed firms (e.g., Spotify, Volvo, Ericsson) usually publish a lot of information in their mandatory regulatory filings

- Check out Compustat North American & Compustat Global

➔Get a Wharton Research Data Services (WRDS) account from SSE library (https://wrds-www.wharton.upenn.edu/pages/get-data/compustat-capital-iq-standard-poors/)

- Hundreds of variables, directly downloadable data in CSV format, already cleaned and standardized
- Doesn’t cover startup companies and other larger companies that are not publicly traded (e.g., Klarna)

Web Data Sources: General Search

If you are starting a **general data search**

- Before ChatGPT:
  - Google dataset search: https://datasetsearch.research.google.com
  - Kaggle: https://www.kaggle.com/datasets/rajugc/kaggle-dataset
- Now you can also ask ChatGPT (or another LLM), plain & simple

- Don’t forget to ask your professors and industry friends

Web Data Sources: Open Data & APIs

Open data archives and APIs

- GitHub: e.g., https://github.com/collections/open-data, https://github.com/DATASETS
- Open APIs: e.g., https://github.com/public-apis/public-apis?tab=readme-ov-file, https://rapidapi.com/collection/list-of-free-apis

… and A LOT MORE !

Web Data Sources: Web Crawling

Web crawling (brute force, not recommended unless you're Batman)

- Send HTTP requests and parse source page HTML
- Dynamic web-scraping (e.g., using Selenium)

Disadvantages relative to APIs and downloadable clean data:

- Usually takes longer: each HTTP call takes longer than an API call for the same amount of information (lots of tags instead of actual data)
- Usually need more cleaning: additional regular expression parsing
- HTML structures change across pages over time in unpredictable ways
- Sending fake traffic to website can overload the server (DDoS attacks), and many websites now use bot blockers to thwart web crawlers

Web Data Sources: Large Platforms

Look for the glue (software tools to facilitate data collection)

- Widely-used large platforms have a lot of data, but they are not easy to retrieve or automatically download at scale: e.g., YouTube videos, Amazon products, Google search, Reddit posts, and more
- They often contain useful unstructured data: e.g., images, audio

- There tend to be available tools (e.g., Python module) to facilitate data collection at scale for commonly used platforms

Web Data Sources: Large Platforms

- For example, you can download YouTube videos

```
>> from pytubefix import YouTube
>> from pytubefix import Playlist
>> from pytubefix.cli import on_progress
```

- Transcription tools to turn audio into text
  - e.g., OpenAI Whisper (https://openai.com/index/whisper/), VOSK (https://github.com/alphacep/vosk-api?tab=readme-ov-file)

- Download GitHub repositories (data, code, and documentation)
  In command line, “git clone [URL]”