ETH zürich

Working with (big) data II

Big Data for Public Policy

Christoph Goessmann
Law, Economics, and Data Science Group (Prof. Elliott Ash)

Course: 860-0033-00L Big Data for Public Policy, Spring 2023

16 March 2023

Student Presentations

Reminder if you're taking the course for credit

By **19 March**, please:

- **Organize into a group** / let us know if you want us to assign you to a group,
- **Choose a paper,** and
- **Sign up for a presentation slot.**

Repetition

What did we do last time?

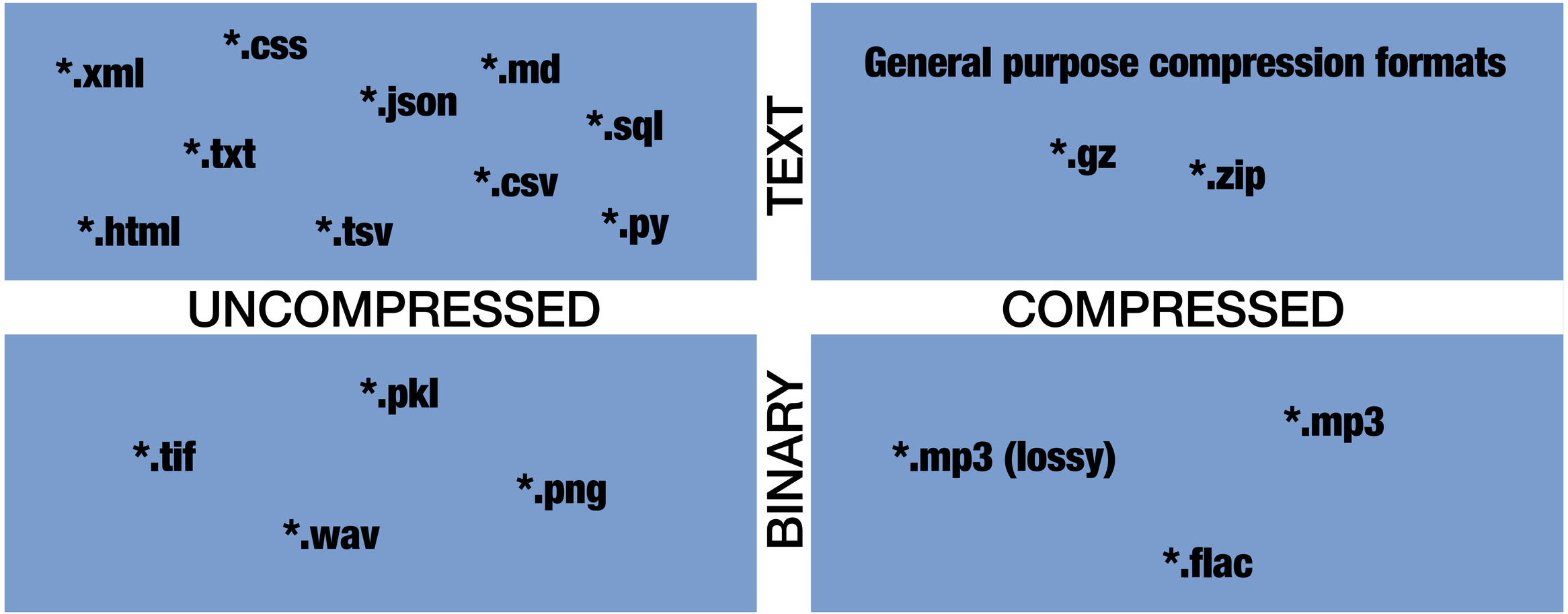
- I/O bound vs. CPU bound processes
- Basic command line usage
- Python virtual environments (`pipenv`)
- Automatic logging with timestamps (`import logging`)
- Estimating memory requirements
- Timing/profiling (`time`, cProfile, SnakeViz)
- Multi threading vs. single threading
- CPU → GPU (`cupy` as `numpy` GPU drop-in replacement → 24x faster)

Tentative Outline for Today

- **Shapes and forms of data**
- **APIs (and scraping)**
  - APIs (HTTP GET and HTTP POST)
  - Proxies vs. VPN
  - Parsing
- **Hands-on**
  - (Multi-threaded) API requests
  - OLS regression

Data

Some examples



Data

Stick to human-readable and open source formats

- **Flat/tabular data: `*.csv`**
  - the open source and universal standard
  - databases are optimized for ingestion
- **Hierarchical data: `*.json`**
- **Arbitrary data structures: `*.pkl`**
  - only use as last resort / if there is no other option

Use the integrated modules (`import json`, `import csv`) to read and write files. Do not escape/delimit manually … you'll almost certainly mess up.

Character encoding: use **UTF-8** whenever possible, it is the de-facto standard and works well internationally.

APIs

Formalized way of exchanging data → Automation :)

Mostly HTTP GET and POST APIs:

HTTP GET

- query parameters in **URL**
- **header** for additional information (e.g., authentication for non-public APIs)
- length of URL is limited to 2048 characters (limiting when there are many query parameters)
- can only be use to **request** data

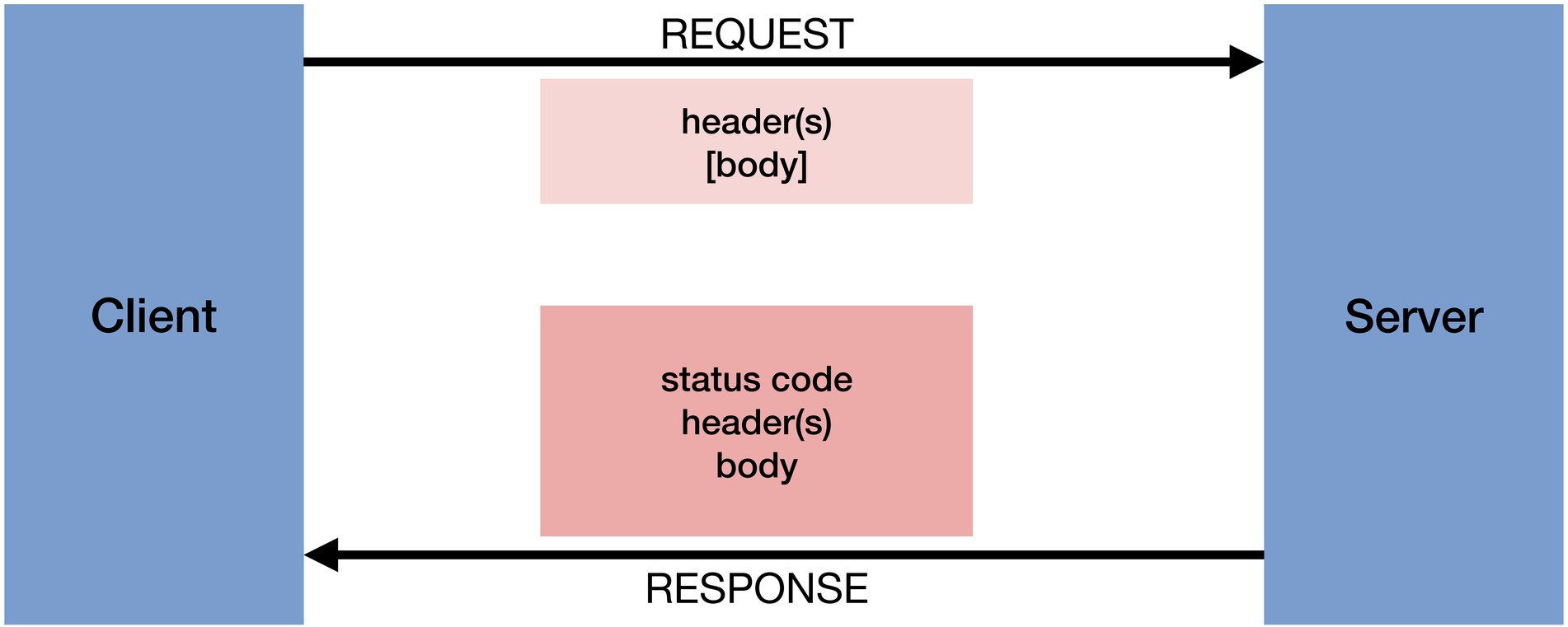
HTTP POST

- query parameters in **body**/payload
- **header** for additional information (e.g., authentication for non-public APIs)
- secure if HTTPS/TLS is used
- can be used to **request** and **send** data

Both methods return a HTTP status code, header(s), and normally also body.

APIs

Formalized way of exchanging data → Automation :)



APIs and Scraping

Appropriate tools

Command line: `curl`

Python: `import requests`

Web scraping is not inherently different from using APIs. Your browser is effectively doing the same as you in the command line. But: JavaScript and other interactive elements might require you to go beyond simple requests and opt for something like **selenium**.

Proxies and VPNs can mask your real IP address; mostly used to change client geolocation or rotate through IPs. Proxies normally better suited for scraping (less overhead, faster rotation).

For **parsing**, `BeautifulSoup` mostly is sufficient. For more complicated cases, you can go for `xml.etree.ElementTree`.

Hands-on

1. Create a virtual environment

```
$ pip install pipenv
$ mkdir bdpp_session2
$ cd bdpp_session2
$ pipenv install requests
$ pipenv install ipykernel
$ pipenv install matplotlib
$ pipenv install statsmodels
$ pipenv install tenacity
```

2. Open this lecture's jupyter notebook so that we can:

- execute an API request and examine its response
- performe an OLS regression in python
- optimize an I/O bound process (vs CPU bound in last session)
- increase scraping robustness via `@retry` decorator

End-of-Lecture Survey

ETH Edu App

Web app

https://eduapp-app1.ethz.ch/

iOS

Android