Data Science for Public Policy From Econometrics to AI

Working with Big Data and Intro to ML

ETH Zurich

06/03/2024

Outline

Working with Big Data

Intro to Machine Learning

Working with Big Data

What does **Big Data** mean?

- Datasets too large or complex to be handled with traditional data-processing software
  - Data with *many* observations
  - Data where each observation is *big* (e.g., images or text)

```
print("Dataset Dimension: ", df.shape)
df[['filing_date', 'case_no', 'county', 'city', 'sex', 'race', 'party', 'first_name', 'middle_name', 'last_name', 'defendant_id', 'name']]
```

```
Dataset Dimension:  (3328979, 27)
```

	filing_date	case_no	county	city	sex	race	party	first_name	middle_name	last_name	defendant_id	name
0	2000-02-29	2000CF000017	1	Friendship	M	NaN	Defendant	robert	d	chitko	NaN	robertdchitko
1	2000-02-09	2000TR000178	1	Wisconsin Dells	F	NaN	Defendant	stacy	k	kallenbach	NaN	stacykkallenbach
2	2000-03-14	2000CV000043	1	Dellwood	NaN	NaN	Defendant	donald	a	kich	NaN	donaldakich
3	2000-02-17	2000CV000032	1	Wisconsin Dells	NaN	NaN	Defendant	michael	NaN	strack	NaN	michaelstrack
4	2000-03-08	2000CF000021	1	Baraboo	M	NaN	Defendant	kenneth	r	paddock	NaN	kennethrpaddock
...	...	...	...	...	...	...	...	...	...	...	...	...
3328974	2020-03-05	2020TR002293	70	Berlin	F	Caucasian	Defendant	brittany	e	clark	NaN	brittanyeclark
3328975	2020-03-26	2020TR000875	29	Mauston	M	Caucasian	Defendant	jesse	a	johnston	NaN	jesseajohnston
3328976	2020-12-10	2020SC001019	16	Superior	NaN	NaN	Defendant	lona	l	scouton	NaN	lonalscouton
3328977	2020-02-07	2020SC004567	40	South Milwaukee	NaN	NaN	Defendant	shana	NaN	inkman	NaN	shanainkman
3328978	2020-09-11	2020TR006367	30	Burlington	M	Caucasian	Defendant	daniel	henry	gauger	NaN	danielhenrygauger

3328979 rows × 12 columns

Working with Big Data – Challenges

- Require high computational power
- Very code-intensive $\implies$ hurdles to reproducibility
- Raw data are hard to handle $\implies$ we will learn some pre-processing techniques in the rest of the course

Working with Big Data – Computational Power

- Most laptops have between 8GB and 16GB of RAM

Working with Big Data – Computational Power

- Most laptops have between 8GB and 16GB of RAM
- However, large datasets require more just to load

```
import time
import psutil
start_time = time.time()
mem_before = psutil.Process().memory_info().rss >>30

df = pd.read_csv(os.path.join(data_path, "wisc_parties_names_clean_wemb_fastt_info.csv"), low_memory=False)
print("Time to load the dataset: ", round(time.time() - start_time, 3), " seconds")
mem_after = psutil.Process().memory_info().rss / (1024 ** 3)  # Convert to MB
print("RAM used: ", mem_after-mem_before)
```

```
Time to load the dataset:  326.89  seconds
RAM used:  11.74294662475586
```

Working with Big Data – Computational Power

- Most laptops have between 8GB and 16GB of RAM
- However, large datasets require more just to load
- How do we efficiently handle big data?
  - Monitor time and memory (or CPU) usage: use functions `time` and `psutil`, o with `cProfile`

Working with Big Data – Computational Power

- Most laptops have between 8GB and 16GB of RAM
- However, large datasets require more just to load
- How do we efficiently handle big data?
  - Monitor time and memory (or CPU) usage: use functions `time` and `psutil`, o with `cProfile`
  - Use a server to process your data, e.g., *ETH Euler Cluster*
  - Reduce data dimension

Working with Big Data – Computational Power

- Most laptops have between 8GB and 16GB of RAM
- However, large datasets require more just to load
- How do we efficiently handle big data?
  - Monitor time and memory (or CPU) usage: use functions `time` and `psutil`, o with `cProfile`
  - Use a server to process your data, e.g., *ETH Euler Cluster*
  - Reduce data dimension
  - Run tasks in parallel using multiple CPUs (parallelization)

    see here and here for an introduction to parallel programming
  - Use GPUs instead

    see here for an introduction to GPUs

Working with Big Data – Computational Power

- Most laptops have between 8GB and 16GB of RAM
- However, large datasets require more just to load
- How do we efficiently handle big data?
  - Monitor time and memory (or CPU) usage: use functions `time` and `psutil`, o with `cProfile`
  - Use a server to process your data, e.g., *ETH Euler Cluster*
  - Reduce data dimension
  - Run tasks in parallel using multiple CPUs (parallelization)

    see here and here for an introduction to parallel programming
  - Use GPUs instead

    see here for an introduction to GPUs

IMPORTANT: Use the amount of resources you need:

more does not always mean better!

Working with Big Data – Replicability I

- With code-intensive projects easy to mess up scripts
- Best practices for replicability (see Gentzkow and Shapiro, 2014)
  - At most four data directories: `data/raw`, `data/proc`, `data/final`, `data/aux` (often not needed)
  - Use `readme.md` files to annotate how each source of raw data was obtained
  - Keep scripts self-contained: one script = 1 task
  - Number scripts in order to execution (e.g., `00_scrape_data`, `01_clean_data`)
  - Comment scripts extensively (task of the script, describe functions)

Working with Big Data – Replicability II

- Packages and dependences can become obsolete very quickly
- Use **virtual environments**, `venv`

In the Command Line

Create the virtual environment in the directory of the project specifying the name (e.g., *myenv*)

```
$ cd myproject
myproject$ python -m venv -system-site-packages myenv
```

Activate the venv

```
myproject$ source myenv/bin/activate
```

Install needed packages

```
(myenv) myproject$ pip install packages
```

Collecting Data

- Simple downloads: easy, clean

Collecting Data

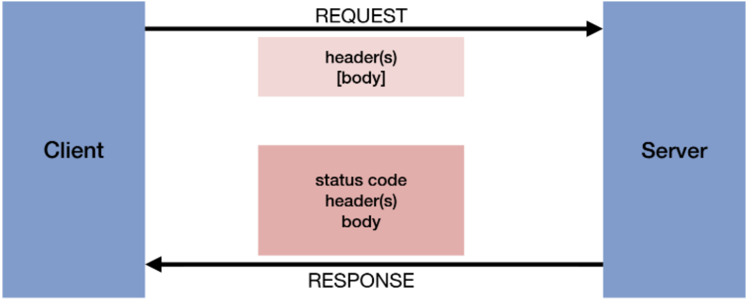
- Simple downloads: easy, clean, but hard for very large data

Collecting Data

- Simple downloads: easy, clean, but hard for very large data
- APIs: *online software* that allows to access data and its feature

Collecting Data

- Simple downloads: easy, clean, but hard for very large data
- APIs: *online software* that allows to access data and its feature
  - Mostly HTTP GET (only request data) and HTTP POST (request and send data) APIs



Collecting Data

- Simple downloads: easy, clean, but hard for very large data
- APIs: *online software* that allows to access data and its feature
  - Mostly HTTP GET (only request data) and HTTP POST (request and send data) APIs
- Web Scraping: automatic information retrieval from websites

Collecting Data

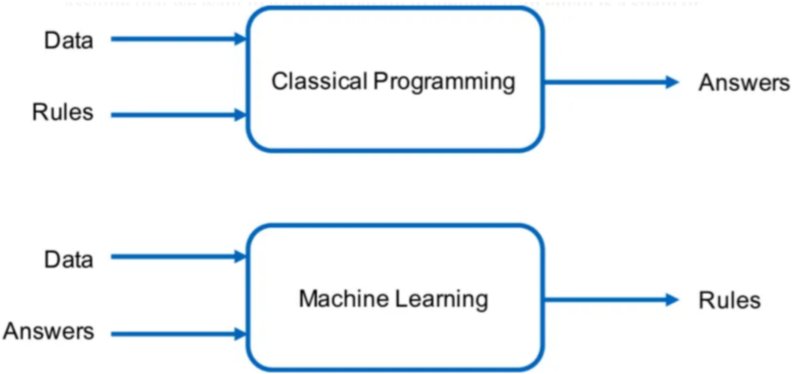
- Simple downloads: easy, clean, but hard for very large data
- APIs: *online software* that allows to access data and its feature
  - Mostly HTTP GET (only request data) and HTTP POST (request and send data) APIs
- Web Scraping: automatic information retrieval from websites
  - **Command line:** `curl`
  - **Python:** `request` to get webpage HTML content + `BeautifulSoup` for parsing HTML
  - If web pages use JavaScript or other interactive elements `selenium` works better

Outline

What is Machine Learning?

What is Machine Learning?

- In machine learning the objective is to learn rules from data and outcomes



What is Machine learning

Is a linear regression machine learning?

Machine Learning VS Econometrics

	Econometrics	**Machine Learning**
y	Dependent Variable, Outcome	Label
x	Independent/Explanatory Variable	Features
Objective		

Machine Learning VS Econometrics

	Econometrics	**Machine Learning**
y	Dependent Variable, Outcome	Label
x	Independent/Explanatory Variable	Features
Objective	**Causal Inference:** precisely estimating the **true parameters** of interest	**Prediction:** finding the **best rule** to predict y starting from x

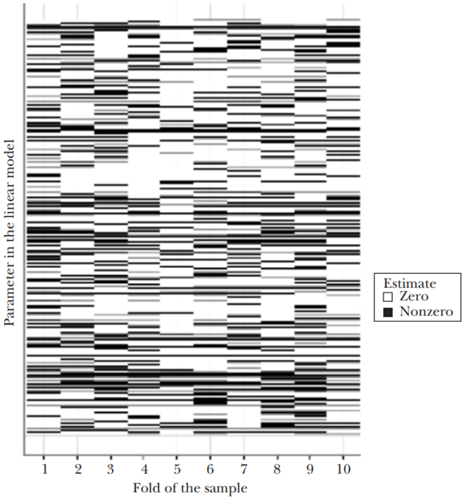
Machine Learning VS Econometrics

	Econometrics	**Machine Learning**
y	Dependent Variable, Outcome	Label
x	Independent/Explanatory Variable	Features
Objective	**Causal Inference:** precisely estimating the **true parameters** of interest	**Prediction:** finding the **best rule** to predict y starting from x

$\rightarrow$ Optimizing for one task doesn't solve the other: the parameters of the best model for prediction are not necessarily the causal effect

Machine Learning VS Econometrics

Selected Coefficients (Nonzero Estimates) across Ten LASSO Regressions



Mullainathan, S., and Spiess, J. (2017). Machine Learning: An Applied Econometric Approach, *Journal of Economic Perspectives*

Main Ingredients

What?		
Solving a Task		
Predict an outcome or learn patterns		

Main Ingredients

What?	**How? I**	
Solving a Task	Using Information	
Predict an outcome or learn patterns	Using labeled or unlabeled data: **training set**	

Main Ingredients

What?	**How? I**	**How? II**
Solving a Task	Using Information	Learning from Mistakes
Predict an outcome or learn patterns	Using labeled or unlabeled data: **training set**	Evaluating the predictions with performance metrics on the **test set**

Main Ingredients

What?	**How? I**	**How? II**
Solving a Task	Using Information	Learning from Mistakes
Predict an outcome or learn patterns	Using labeled or unlabeled data: **training set**	Evaluating the predictions with performance metrics on the **test set**
Filtering Spam	Set of emails labeled as *spam* or *not spam*	Check number of correct predictions

Solving a Task: Some Examples

- Predict house prices
- Assigning topics to articles
- Detect disease from medical reports
- Detect tax evasion
- Dimensionality reduction
- Detect faces in images
- Recommend products, songs, videos, etc.

Data Classifications

Labeled Data

- The training data contains information on y: *prices*, *spam/not spam*
- If every label is contained equally often then the data are **balanced** otherwise **unbalanced**
- To learn the existing labels use **Supervised Learning**

- **Continous** vs. **discrete** labels:
  - Regression for continuous labels
  - Classification for discrete labels

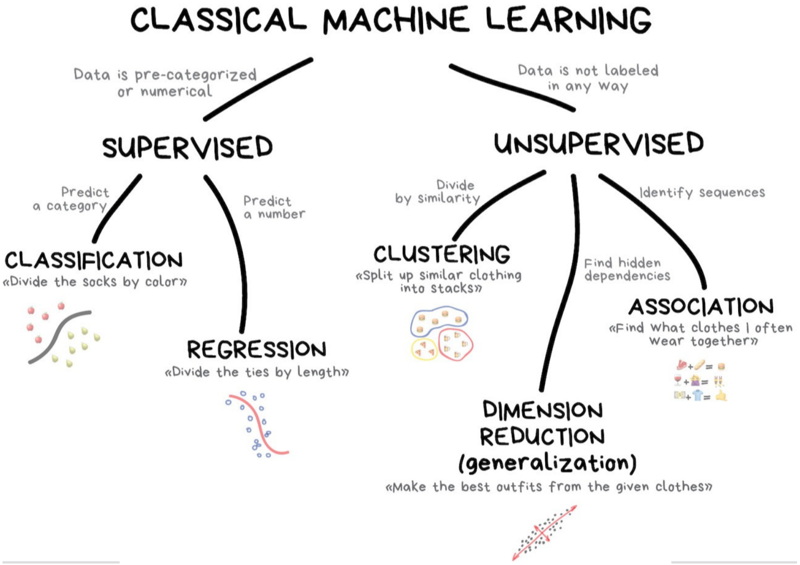
Data Classifications

Labeled Data

- The training data contains information on y: *prices*, *spam/not spam*
- If every label is contained equally often then the data are **balanced** otherwise **unbalanced**
- To learn the existing labels use **Supervised Learning**
- **Continous** vs. **discrete** labels:
  - Regression for continuous labels
  - Classification for discrete labels

Unlabeled Data

- The training data does not have y labels
- To learn labels and patterns in the data use **Unsupervised Learning**



Source: https://vas3k.com/blog/machine_learning/

Classifying Tasks

Back to our example tasks, which are supervised vs. unsupervised learning?

- Predict house prices
- Assigning topics to articles
- Detect disease from medical reports
- Detect tax evasion
- Dimensionality reduction
- Detect faces in images
- Recommend products, songs, videos, etc.

Evaluation: Performance Metrics

Once we train our ML model we need to evaluate how good it is . **How?**

- We need to evaluate the model on data it has not seen (**out-of-sample data**)
- Typically, we split the data into:
  - 70-80% training data
  - 30-20% test data, never seen during training
- Key question: how close the prediction is to the true label?
  - **Regression:** difference between predicted value ($\hat{y}$) and true value (y), e.g., mean squared error, mean absolute error (MSE, MAE)
  - **Classification:** share of correctly predicted labels (**accuracy**), share of *true positives* among all positive predictions (**precision**), share of *true positives* among all actual positives (**recall**)
  - **Unsupervised learning:** depends on the model/task, sometimes we'll need to be creative!
- Bad performances on the test set suggest under/over-fitting in the training set

Examples

Predicting recidivism rate – Ash, Goel, Li, Marangon, and Sun, 2023

One of the objectives of judges when deciding the sentencing outcome (incarcerating vs. other punishments) is to minimize the likelihood of committing another crime.

- The information is not available ex-ante
- However, we know whether past defendant re-offended after being sentenced, and their characteristics
- We use the universe of criminal cases in Wisconsin from 2000 to 2017 to predict recidivism rate starting from defendants characteristics and criminal history

Examples

Predicting recidivism rate – Ash, Goel, Li, Marangon, and Sun, 2023

- **Task:** predict recidivism episode (binary classification)
- **Labels:** Recidivism, defined as re-offense within 2 years from the date of disposition
- **Features:** Criminal history (using extended panel 1970-2019), case characteristics, gender, and age
- **ML algorithm:** XGBoost for classification tasks
- **Evaluation:** performance in out-of-sample test set, accuracy=0.65

Examples

Predicting recidivism rate – Ash, Goel, Li, Marangon, and Sun, 2023