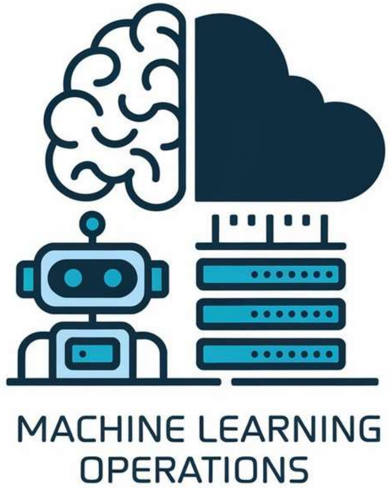
1st AI-INFN Advanced Hackathon
28 November 2024, Padua

Machine Learning Operations (MLOps)

Introduction to MLOps for experiment tracking

Luca Clissa

Researcher @ DIFA - UNIBO
luca.clissa2@unibo.it



ML model in production

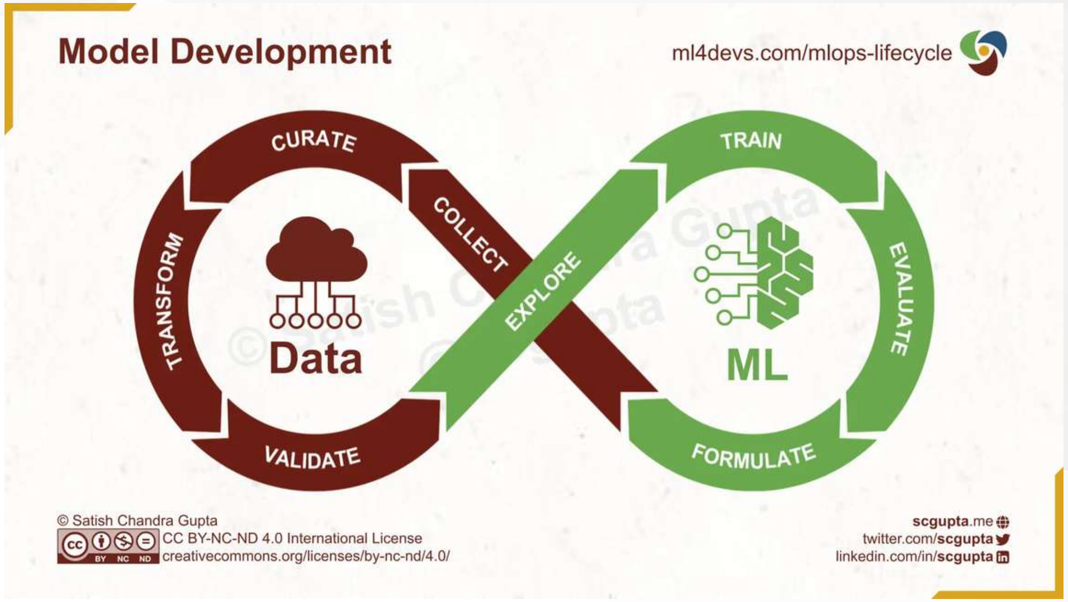
- Many ML projects fail before being deployed to production

- **Why is it so difficult?**

- **What can we do?**

ML projects pipeline

Machine Learning projects involve several **stages** that share a common **goal**:



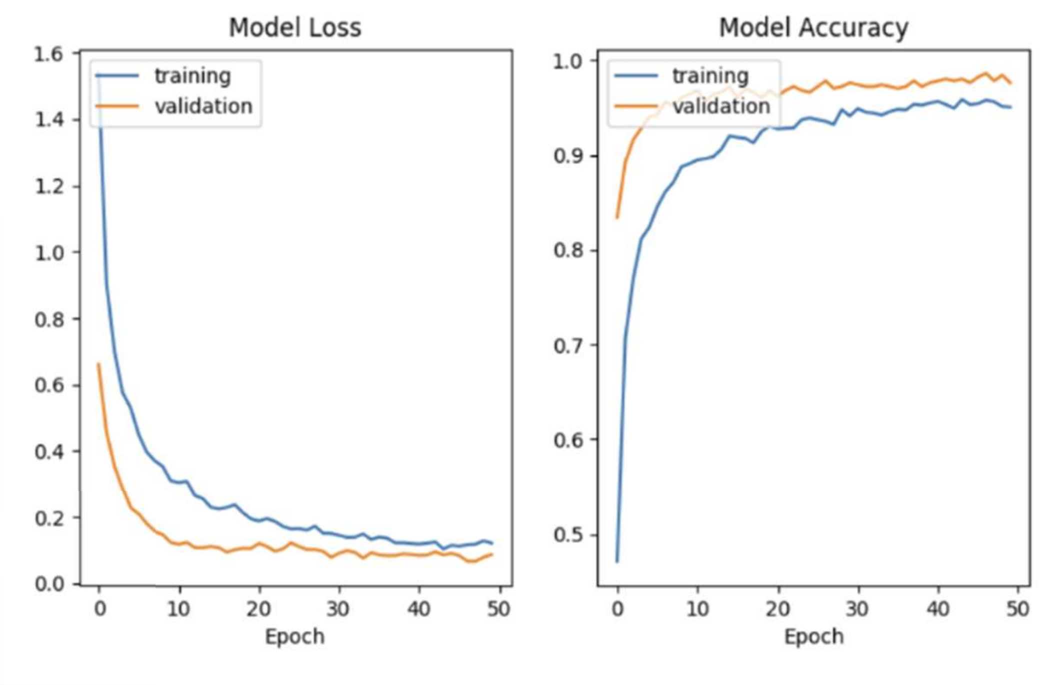
Stages

Data processing: collection, transformation, validation, exploration
Model development: training, evaluation, formulate new hypoothesis to test...

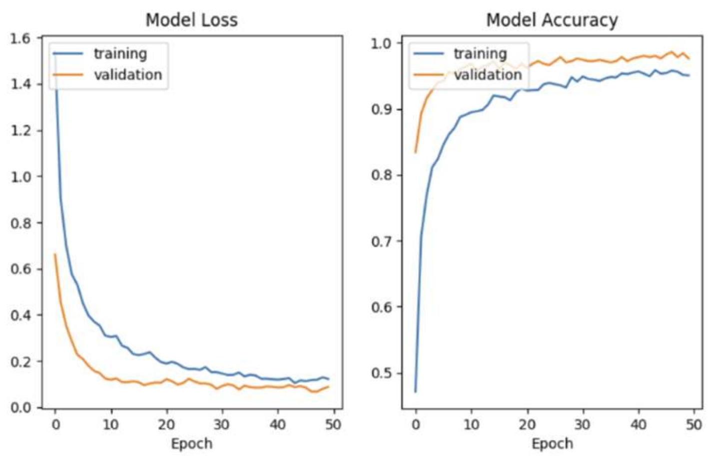
Goal

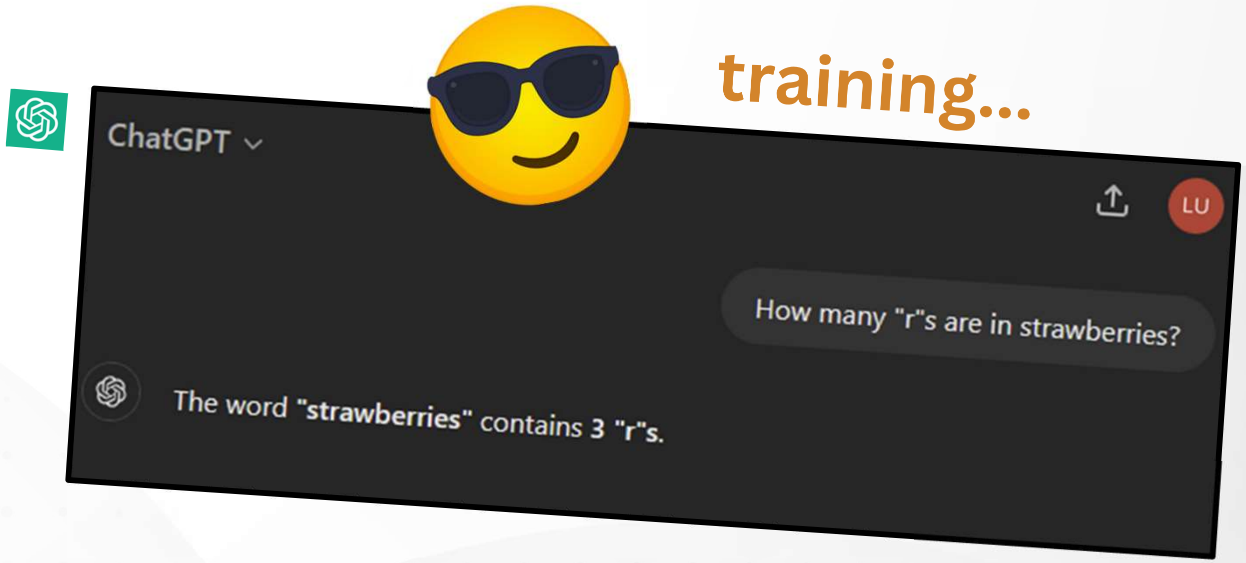
Build model that **works well in production**, is **portable** and **reproducible**

What do we need?

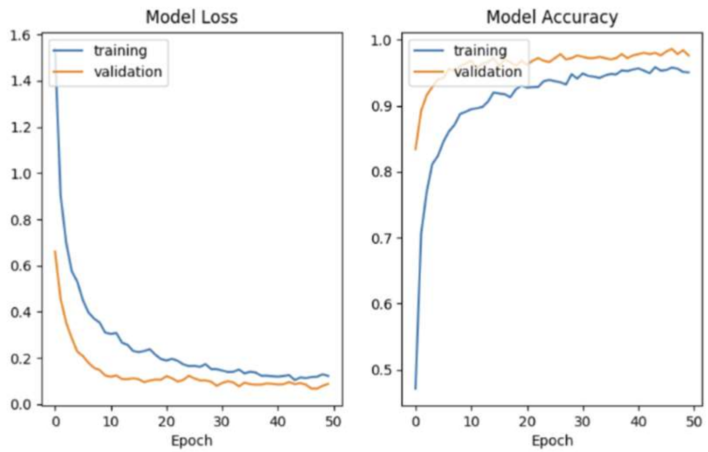


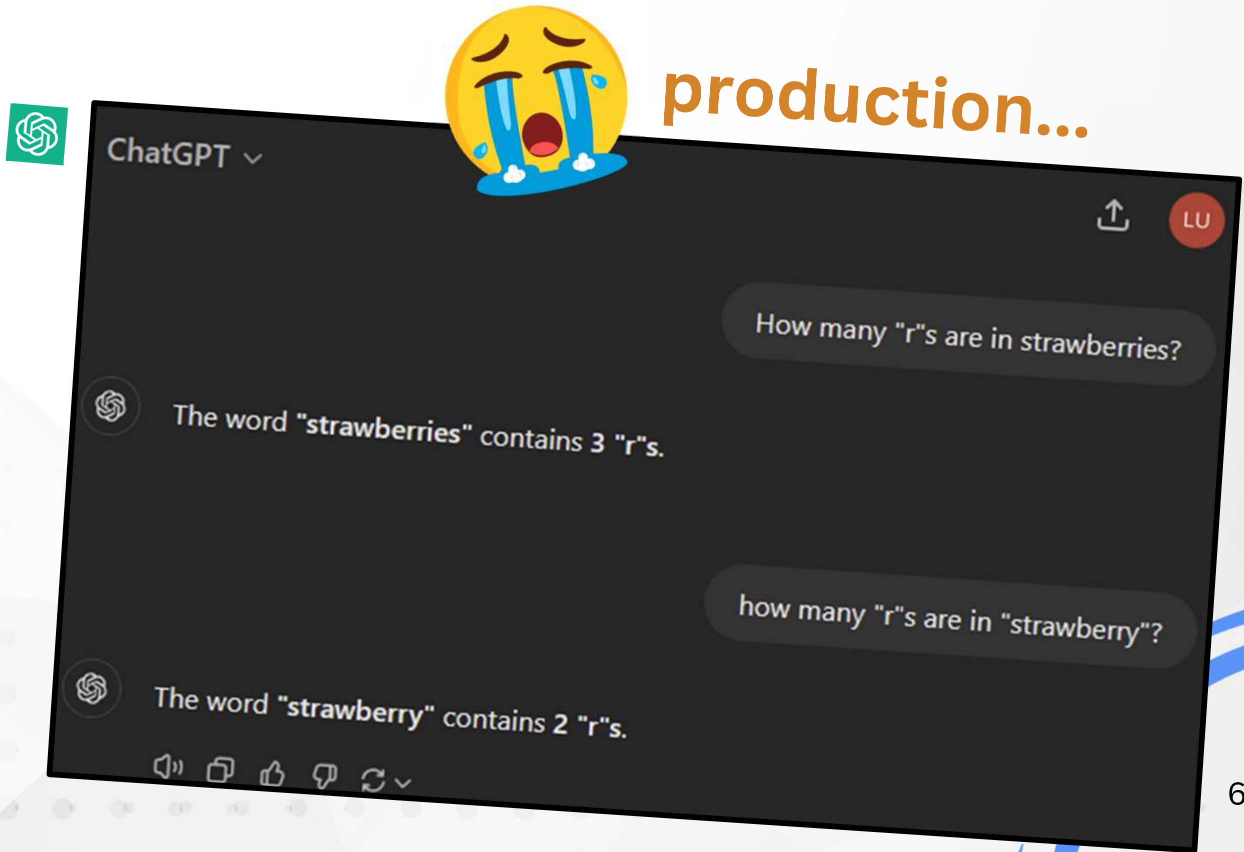
What happens in practice





What happens in practice





What is MLOps?

> MLOps is about maintaining the trained **model performance*** in production.
>
> This may deteriorate due factors we cannot control, so it is key to monitor, update and roll out new models when necessary

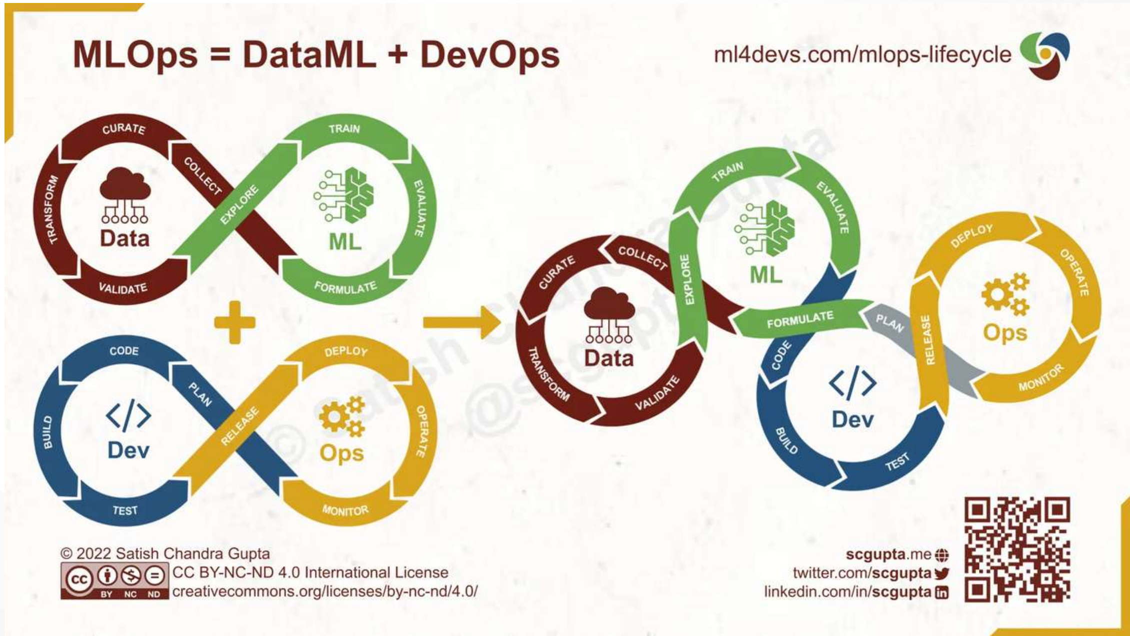
model performance* = metrics, but also latency, SLA, …

MLOps components

DataML = Data + ML/Code

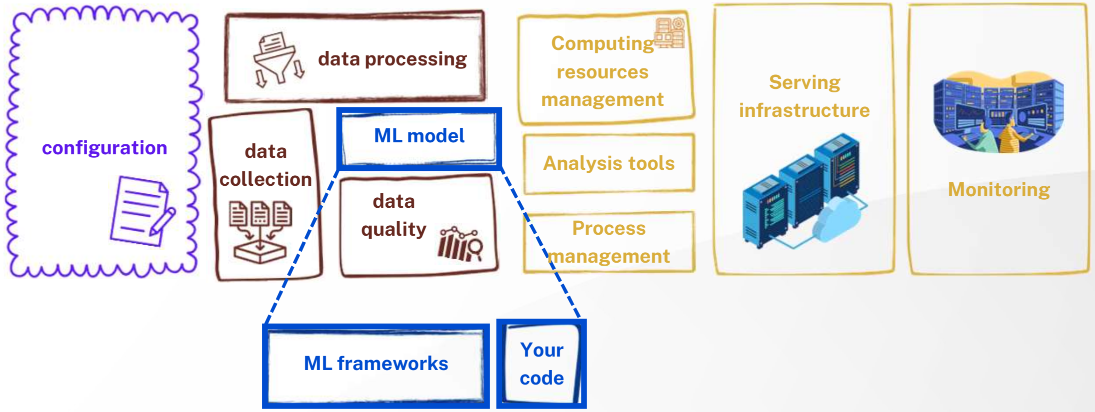
MLOps = DataML + DevOps

+ Algorithm	+ Software
+ Weights	+ Infrastructure
+ Hyperparameters	+ CI/CD



MLOps = ML model + Software

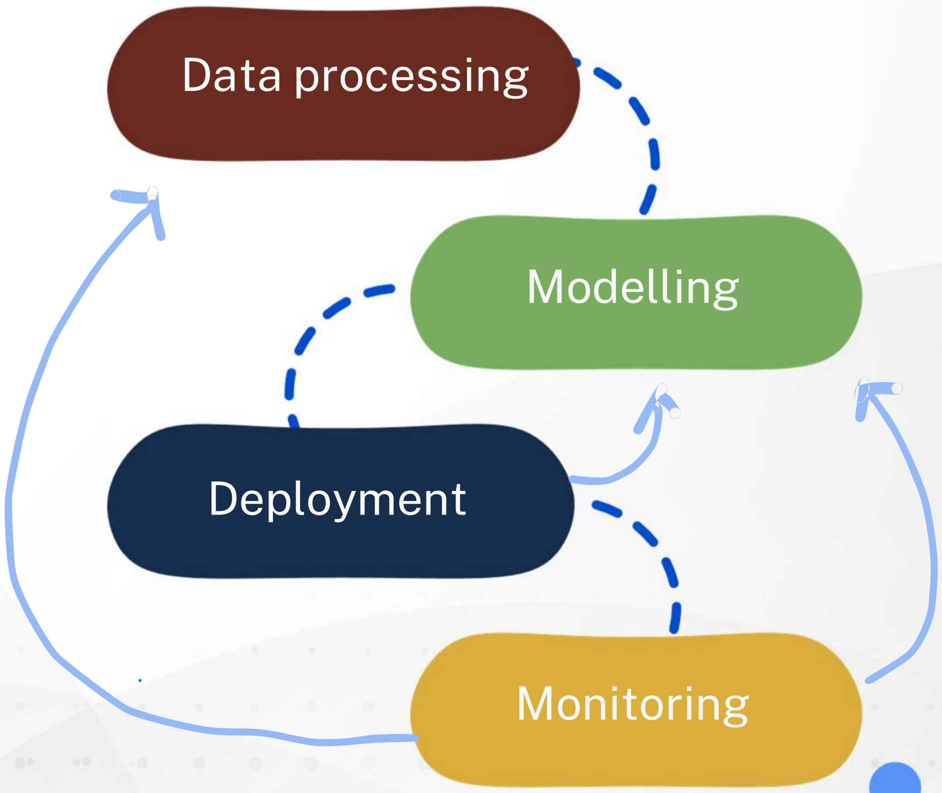
- MLOps combines two key elements:
  - ML model
  - software engineering
- Good news: ready-to-use frameworks for most components
- Downside: hard to keep up with new tools
  ⟶ technical debt [1]



[1] D. Sculley et. al. Hidden Technical Debt in Machine Learning Systems, NIPS 2015

MLOps pipeline

MLOps is a multi-stage, iterative process



Data processing: best practice

- Data processing is key to ML success
  - quality control: garbage in, garbage out 
  - Exploratory Data Analysis (EDA) enables understanding content and challenges
  - tracking, monitoring and reproducibility
- [Paradigm shift: model-centric to data-centric AI [2]]

	Model-driven ML	Data-driven ML
Fixed component	Dataset	Model Architecture
Variable component	Model Architecture	Dataset
Objective	High accuracy	Fairness, low bias
Explainability	Limited	Possible

Data processing pipeline

Data ingestion
- Load file
- Read from database

Data validation
- Check schema (structured data)
- More complex for unstructured data (e.g. audio, image)

Data cleaning
- Fill NaNs
- Filtering
- Transforms (e.g. normalization, standardization, ...)

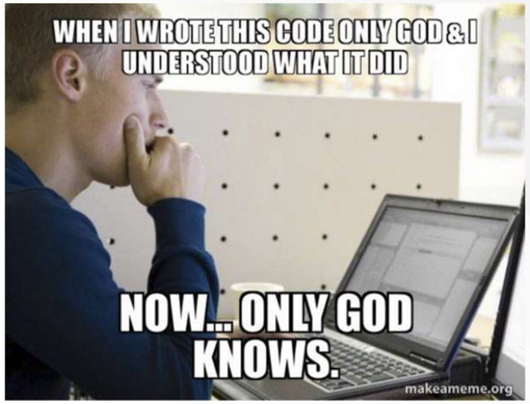
Feature engineering
- Feature selection
- Dimensionality reduction

reproduce steps in production!

Reproducibility:

[3] https://sites.google.com/princeton.edu/rep-workshop/

Paramount to keep track and document every step of our processing to ensure reproducibility!



Documentation

- track every design decision
- make sure to include full descriptions! --> easy to forget, soon out of control
- **Provenance**: where does data come from?
- **Lineage**: how data is manipulated?

Versioning

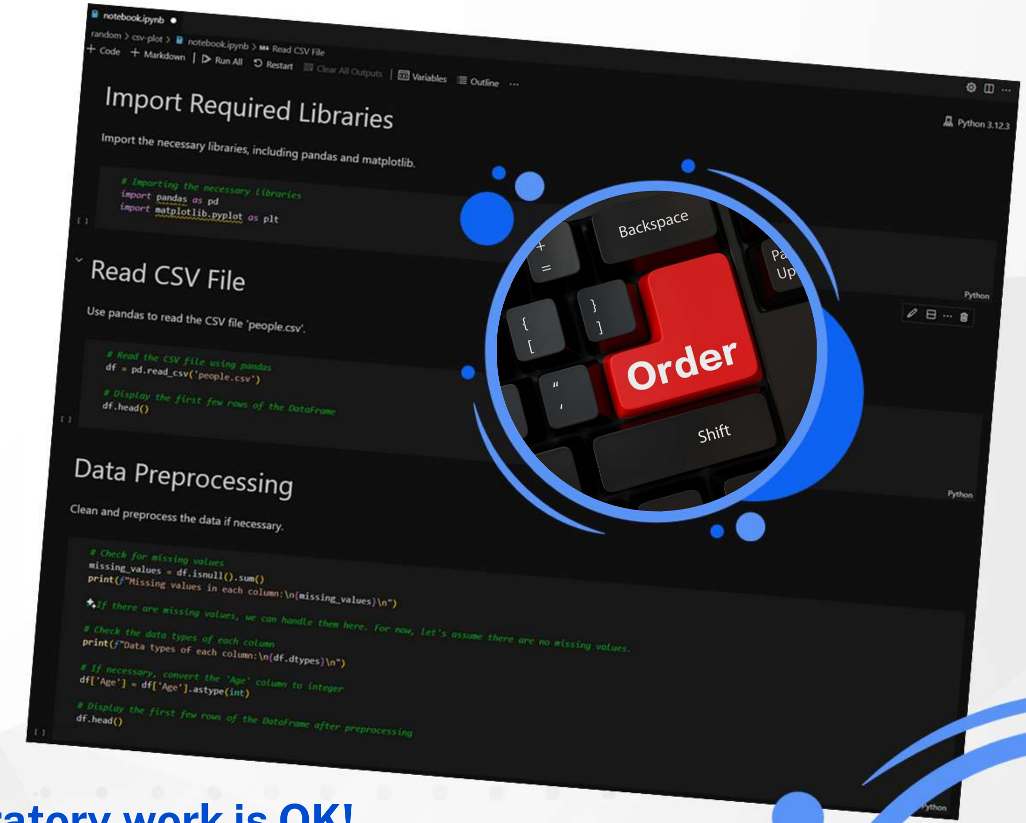
- input data - DVC
- code - git/GitHub

Controlled software envs

- software libraries - e.g. conda/pip
- computing environment - containers

Notebooks: good practice

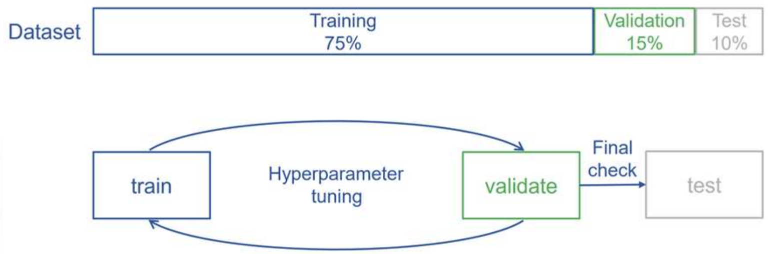
- Linear flow of execution
- Cells as logic units: little amount of code
- Use markdown cells for documentation
- Refactor reusable code into packages
- Set parameters on top: easy to find and edit
  --> notebooks as a function
- Clean notebook before commit to repo



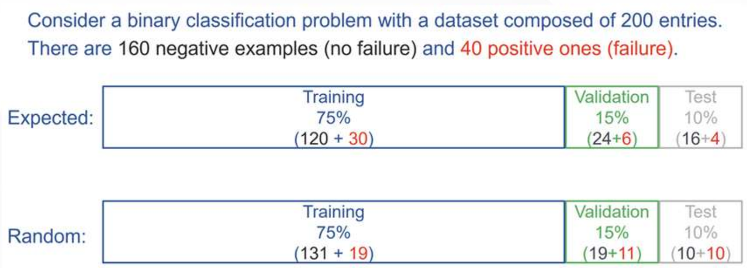
"quick&dirt" exploratory work is OK!
...but remember to tidy up when sharing

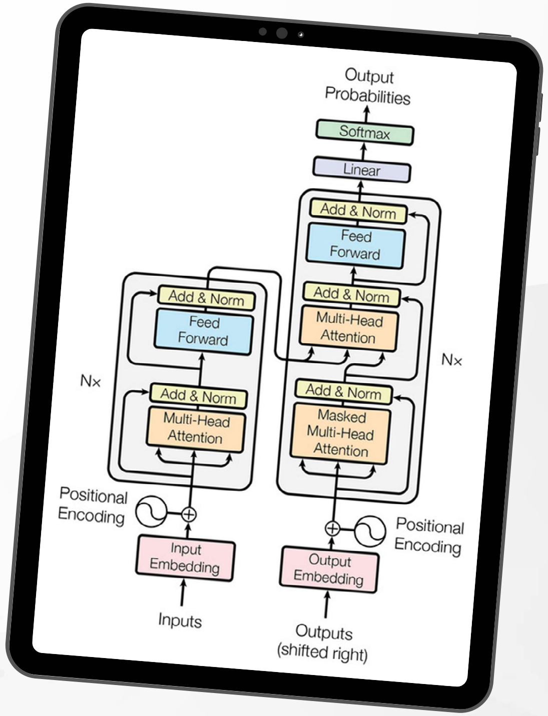
Modelling: good practice

- Data splitting: train, validation, test



- Balanced sampling useful for preserving distributions (fairness)



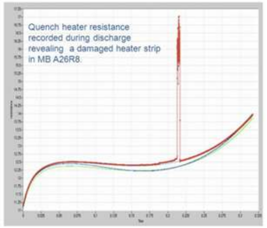


Modelling: good practice

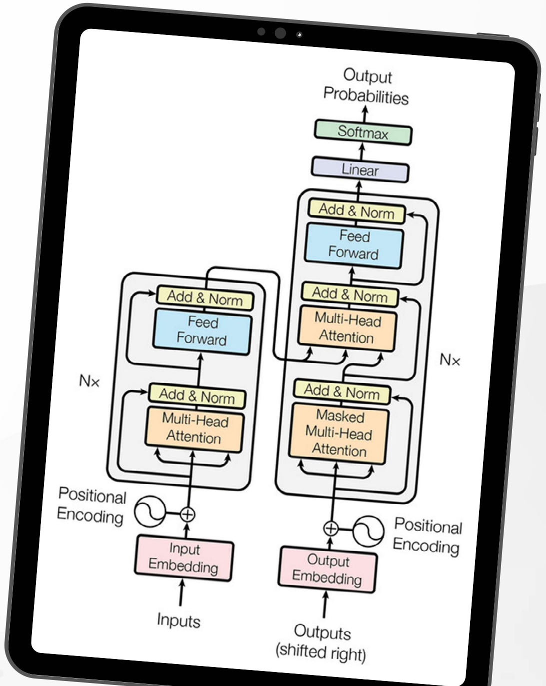
- Data splitting: train, validation, test
- Balanced sampling useful for preserving distributions (fairness)

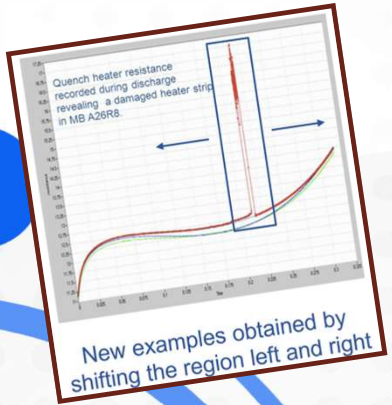
- EDA is key to understand training requirements & challenges
  - class imbalance
  - rare events
  - metrics

e.g. binary classification: 3130 healty signals (Y=0) VS 112 failures (Y=1)



[4] C. Obermair, Extension of Signal Monitoring Applications with Machine Learning, Master Thesis, TU Graz





New examples obtained by shifting the region left and right

--> **naive classifier** (always predict 0) would be **97% accurate!**

--> **better look at precision/recall/F1-score instead**

--> resort to sampling (upsampling/downsampling) or collect new data

--> **data augmentation** can also help

		Ground truth	
		Y = True	Y = False
Model	Y = True	0 *true positive*	0 *false positive*
	Y = False	112 *false negative*	3130 *true negative*

$$\text{Avg accuracy} = \frac{\text{TN}}{\text{TN} + \text{FN}} = 97\%$$

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{0}{0}$$

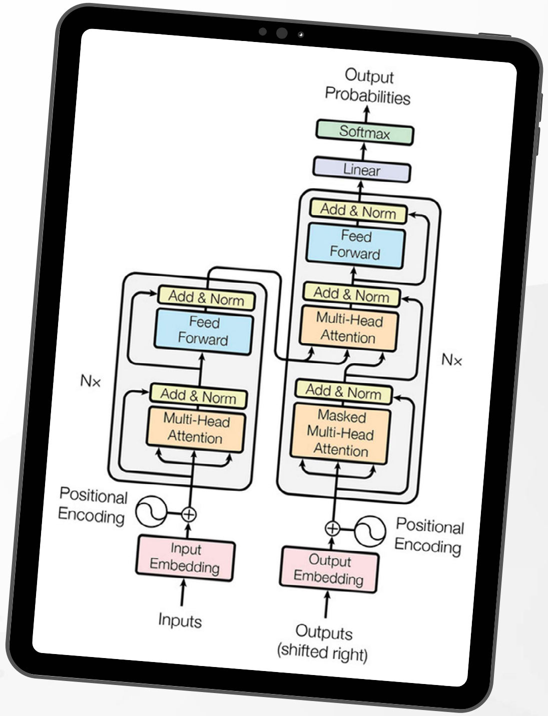
$$\text{Recall} = \frac{TP}{TP + FN} = \frac{0}{0 + 112} = 0$$

$$\text{F1}_{\text{score}} = \frac{2}{1/\text{Precision} + 1/\text{Recall}}$$

Modelling: good practice

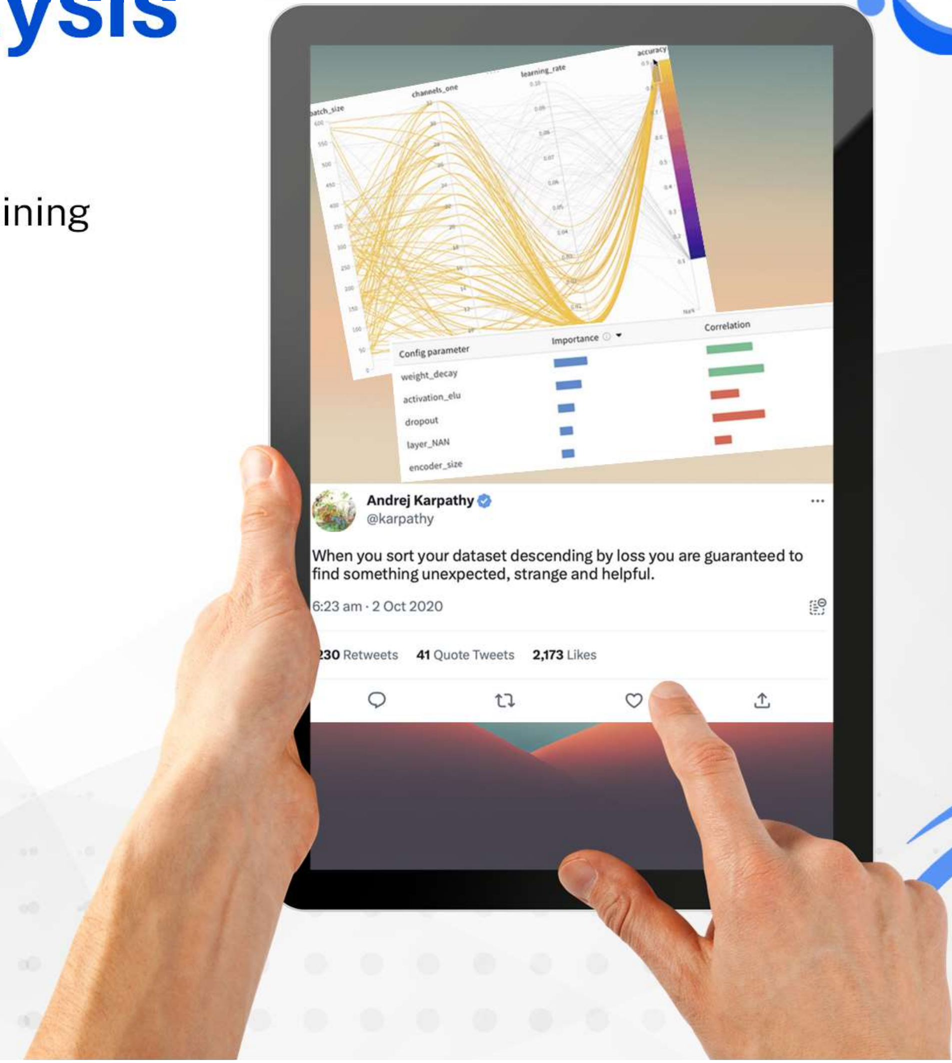
- Data splitting: train, validation, test
- Balanced sampling useful for preserving distributions (fairness)

- EDA is key to understand training requirements & challenges
  - class imbalance
  - rare events
  - metrics
- Experiment tracking
  - pen & paper
  - spreadsheets
  - dedicated frameworks



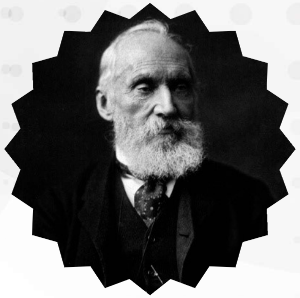
Modelling: error analysis

- Looking at wrong predictions help debugging training
- Set reference metrics to compare experiments
- Exploit tracking framework functionalities
  - Bayesian search
  - parameter importance
  - parallel coordinates plot



> If you can't measure it,
> you can't improve it

William Thomson, Lord Kelvin



Deployment & Monitoring

- When the model is ready we can finally release it in production!
- However, this is not the final step --> MLOps is a cycle!

CI/CD

- Production is only a checkpoint between development stages
- may want to keep improving the model
  - try different configs/architectures
  - re-train as new data comes in

Monitoring

- always keep an eye on performance, as data shift may impact your application
- model metrics
- infrastructure metrics
  - errors, resource utilization, ...

Conclusion

- MLOps is key to ensuring solid ML development and reproducibility
  - It seems overdoing at first, but it pays in the long term
- Many tools can help, difficult to choose one…hard to say what is best
  - Pick one and get proficient with that
  - …. But leave a door open to exploration in case you hit roadblocks

	Development ML	Production ML
Objective	High-accuracy model	Efficiency of the overall system
Dataset	Fixed	Evolving
Code quality	Secondary importance	Critical
Model training	Optimal tuning	Fast turn-arounds
Reproducibility	Secondary importance	Critical
Traceability	Secondary importance	Critical

Any questions?

luca.clissa2@unibo.it academic homepage

mlops-handson LinkedIn blog