FMUL · ISTARS · WORKSHOP · 2025–26

A full-day workshop

Machine Learning for Biomedical Data

From research questions to interpretable models. Concepts, methods, and a full pipeline you can take home.

PRESENTER

Laura Quintas

FMUL · iSTARS

The day, in eight blocks.

Concepts → Python → Data → Unsupervised → Supervised & Evaluation → CV, Leakage & Pipelines

(if we have time) → Challenge

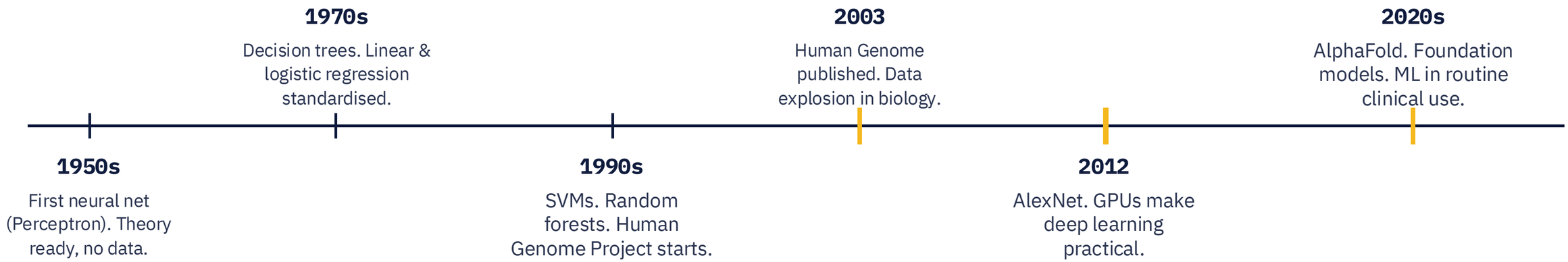
Concepts & foundations · Hands-on notebooks

History, framing, AI vs ML vs DL, the research mindset, supervised vs unsupervised.

With Six Jupyter notebooks, building from `2.1 print()` to a full ML pipelines.

Why are we talking about ML now?

The math is old. What changed is the access. To data, to compute, and to open-source software.



DRIVER 01 · DATA

Cheap to collect.

Sequencing: $3B → ~$200. Imaging digital. Records electronic.

DRIVER 02 · COMPUTE

Cheap to run.

A consumer GPU does work that needed a supercomputer in 2005.

DRIVER 03 · SOFTWARE

Free & open.

scikit-learn, PyTorch, pandas. Documented, on every laptop.

Technology changed the scale of biomedical science.

Twenty years ago: one cohort, one assay. Today: every patient is a multi-modal dataset.

EHR

Diagnoses, labs, meds, vitals.

Omics

Genome, transcriptome, proteome.

Imaging

MRI, CT, ultrasound, pathology.

Wearables

Heart rate, ECG, activity, sleep.

Sensors

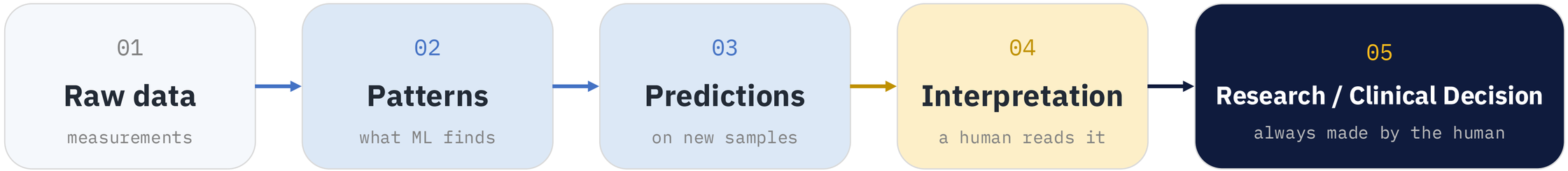
Continuous glucose, pulse-ox, IoT.

Trials

Structured outcomes & endpoints.

The bottleneck moved. It used to be: how do I get the data. Now it is: what do I ask of the data?

From data to decisions.



The model never makes the decision. It surfaces information for the person who does.

What is artificial intelligence?

Systems designed to perform tasks we would associate with human intelligence. Recognising speech, planning, reading a scan, holding a conversation.

> **AI is a goal, not a method.** Symbolic systems from the 1960s and modern large language models both count.

speech recognition · vision · planning · language · robotics · games

A USEFUL FRAMING

“AI is whatever hasn't been done yet.”

, the moving-target problem in AI research

Once a task works, we stop calling it AI. Spam filters were AI in 1995. Autopilot was AI in 2005. The label slides forward.

What is machine learning?

Algorithms that learn rules from examples, instead of being told the rules.

TRADITIONAL PROGRAMMING

You write the rules.

```
def is_spam(email):
if "lottery" in email: return True
if "danger" in email: return True
... return False
```

Brittle. Every new pattern requires a new rule. Does not scale past a handful of features.

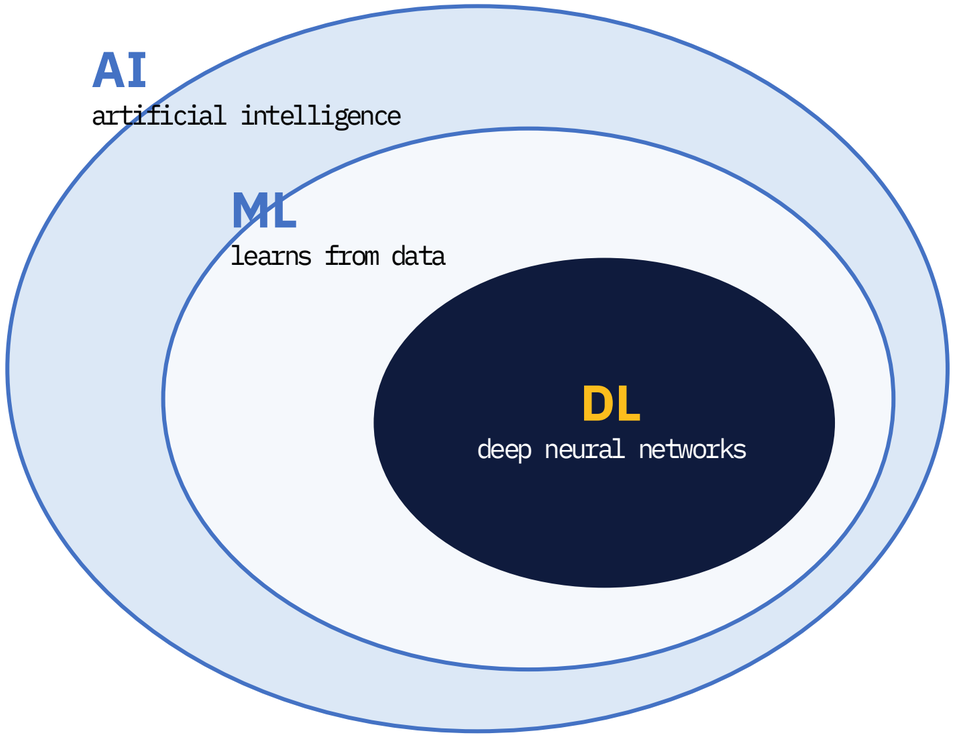
MACHINE LEARNING

The system writes the rules.

```
examples = [(email_1, "spam"), (email_2,
"okay"), …]
model = train(examples)
model.predict(new_email)
```

Scales to thousands of features. Less obvious how it works. Which is exactly why we will spend time on interpretation.

Where does deep learning fit?



AI is the goal.

Any system that does something intelligent.

ML is one approach.

Learn rules from examples. Logistic regression. Trees. Forests. SVM. KMeans.

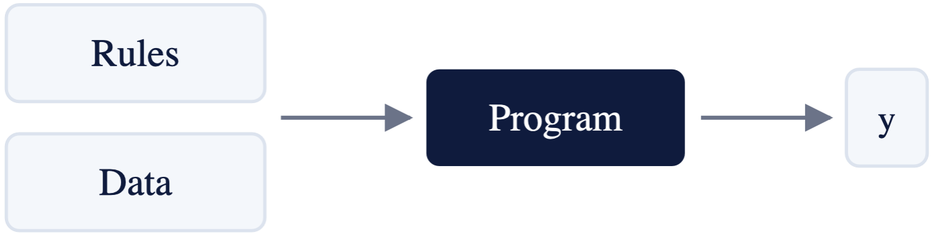
DL is a subset of ML.

Multi-layer neural networks. Excellent on images, audio, language. Overkill for most tabular biomedical work.

ML inverts the classical programming flow.

TRADITIONAL

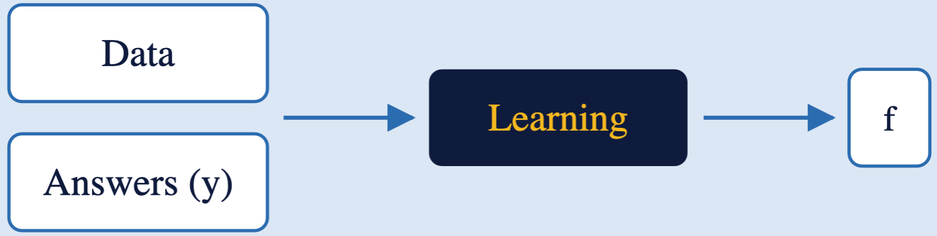
Rules + data → answers.



You design the logic. The computer applies it. Predictable, brittle, hard to scale.

MACHINE LEARNING

Data + answers → rules.



The model discovers the logic. Scales to thousands of features. Requires data and discipline.

Deep learning is powerful. It is also often the wrong choice.

Reach for DL when:

- Inputs are **images, audio, text, sequences**.
- You have **thousands** of labelled samples.
- Simpler models hit a ceiling you can measure.
- You have GPU access and time to tune.

Stick with classical ML when:

- Data is **tabular**. Rows of patients, columns of features.
- You have **hundreds**, not thousands, of samples.
- Interpretability matters for downstream use.
- You need a baseline before anything else.

Default rule: start with logistic regression or a random forest. Justify deep learning with a measured gap, not a guess.

Where ML is already moving the needle.

Medical imaging

Dermatology, radiology, retinal screening. Reading scans at scale.

Genomics / omics

Variant calling, expression analysis, multi-omics integration.

Pathology

Tumour grading and tissue classification from whole-slide images.

Risk prediction

Readmission, sepsis, deterioration, post-op complications.

Drug discovery

Protein folding, target ID, molecular property prediction.

Patient stratification

Discovering subgroups for trials and personalised treatment.

Wearables

Arrhythmia detection, gait, sleep, continuous monitoring.

Clinical text

Extracting structured info from notes, reports, and literature.

What can ML do? Seven verbs.

01

Classify

Assign a label. Ductal or lobular? Responder or non-responder?

02

Predict

Estimate a number. Risk score, age, drug response.

03

Cluster

Group similar samples without labels.

04

Reduce dimensions

Flatten high-D data so a human can see it.

05

Detect anomalies

Flag the unusual sample, the failed assay, the outlier.

06

Prioritise

Rank candidates. Biomarkers, drugs, patients.

07

Support decisions

Give a human extra information. Not a verdict.

Every task today → one verb

SECTION 3 · HONEST LIMITS

What ML cannot do alone.

Read this before the next demo from a vendor.

✕

Prove causality

Association is not cause. ML finds patterns, not mechanisms.

✕

Replace expertise

The model does not know biology. You do. It needs you.

✕

Fix bad data

Garbage in, confident garbage out. Cleaning is not optional.

✕

Guarantee usefulness

High accuracy ≠ clinical value. Validation is multi-stage.

✕

Generalise for free

Trained at one hospital, often broken at another. External validation matters.

Heuristic: if a claim sounds too good for a 100-patient cohort, it probably is.

Biomedical ML requires extra care.

Six properties of biomedical data that quietly break textbook ML.

Small datasets

Hundreds of patients, not millions. Generalisation is the central challenge.

Class imbalance

5% of patients have the rare subtype. Accuracy lies. Use stratified metrics.

Missing data

Patients drop out, instruments fail, fields are blank. Imputation is a decision, not a fix.

Batch effects

Run 1 vs run 2 vs another lab. Models can learn the batch, not the biology.

Data leakage

Test info slips into training. Numbers look great. Reality does not. We will drill this.

Confounding

Age, site, sex, batch. The model picks up the easiest signal, often the wrong one.

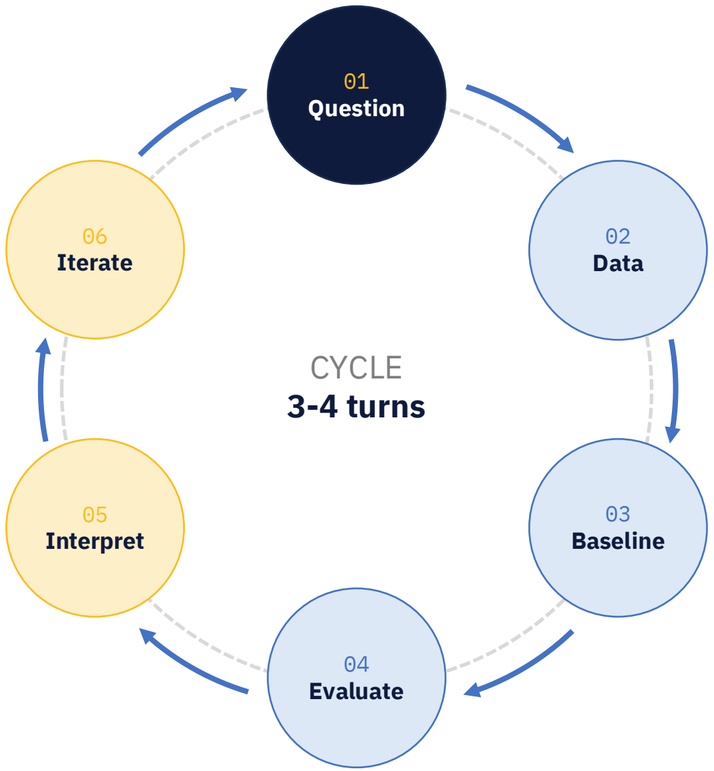
Start with the question.

Given [data] , can we predict / explain / discover [outcome or structure]?

If you cannot write that sentence, you do not have a project yet. You have an interest. A model can only optimise what you define.

Take a question from your own work and write it in this form. What is missing?

The ML research loop.



The arrow goes back. The first answer reshapes the next question. Real ML is iterative, not linear.

Do not start with the model.

ANTI-PATTERN

“I want to use a random forest.”

A tool looking for a job. Months get spent tuning a model that was never the right answer to a question that was never sharply defined.

`model picked` → `data hunted` → `question retrofitted`

PATTERN

“Given x, can we predict y?”

Question first. Data second. Method last. The model is a downstream consequence, not the starting point.

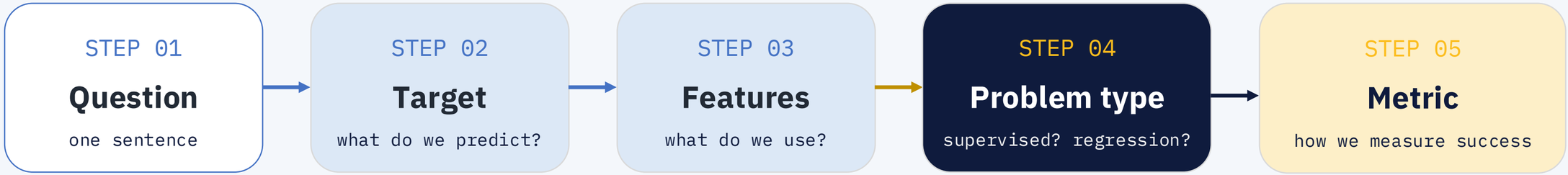
`question` → `data` → `method`

Rule of thumb: if you cannot say in one sentence what you are predicting and from what, do not fit a model yet.

The questions you will actually ask.

BIOMEDICAL QUESTION	SHAPE	WHAT ML RETURNS
Can we predict tumour subtype from omics?	classification	A label per patient, plus a probability.
Can we identify patient subgroups in a cohort?	clustering	A group assignment, no labels needed.
Which features drive the prediction?	interpretation	A ranked list of important features, with caveats.
Can we predict a continuous clinical score?	regression	A number on a continuous scale per patient.

From question to ML task, in five steps.



The metric is part of the question. A wrong metric is a wrong project. Even if the model is good.

Why Python for ML?

- **The ML ecosystem lives there.** scikit-learn, PyTorch, TensorFlow, pandas, numpy.
- **It reads like pseudo-code.** For people coming from biology, that matters.
- **It is free and open.** Runs on your laptop, on a server, in a notebook.
- **Active community.** Almost any problem has a Stack Overflow answer.

Today's stack. Everything else is optional.

`python · pandas · numpy · matplotlib · scikit-learn`

HELLO BIOMEDICAL ML

```
import pandas as pd
from  sklearn.linear_model import LogisticRegression

X = pd.read_csv( "omics.csv" )
y = pd.read_csv( "labels.csv" )

model = LogisticRegression()
model.fit(X, y)
model.predict(X_new)
```

Python and R. Not Python vs R.

Py Python wins on...

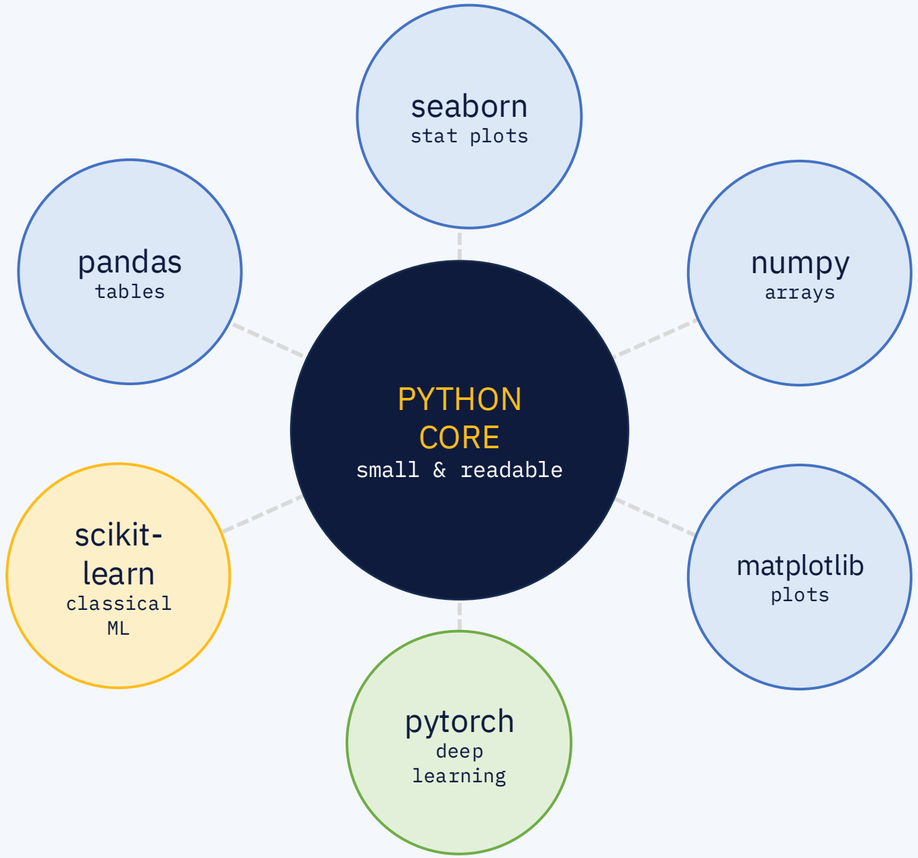
- ML pipelines & deployment
- General-purpose programming
- Deep learning (PyTorch, TF)
- Production & integration

R R wins on...

- Classical statistics & mixed models
- Exploratory data analysis
- Bioconductor (genuinely better)
- Publication-quality figures

Reality: most working biomedical researchers use both. We do Python today because the workshop is about ML. None of this invalidates R.

Python is small. Its packages make it powerful.



Mental model: Python is the language; packages are the toolboxes. `import sklearn` means "load that toolbox into this notebook".

NOTEBOOK · 2.1

Intro to Python.

`2.1_intro_python.ipynb`

The warm-up. By the end you can read every line of code in this workshop without panic.

YOU WILL LEARN TO…

- **Print** things. Yes, that is a real first step.
- **Variables & types**. Strings, numbers, lists, dicts.
- **Functions**. Write one, call one, read one.
- **Import packages** , `import pandas as pd`.
- **Pandas DataFrames**. The biomedical table format.
- **Simple plots**. Your first scatter and histogram.

YOUR FIRST CELL

```
print("Hello, ML")
```

The easiest line of code you will ever write. Press Shift+Enter to run it.

PREPROCESSING · BEFORE NOTEBOOK 2.2

Why we do not start with the model.

Most ML projects fail at the data layer, not the model layer. Exploration and preprocessing is where the project gets honest.

THE 80/20 REALITY

80%

of an ML project is data work. The model is the easy part.

A clean dataset with a sloppy model usually beats a clean model with a sloppy dataset.

WHAT YOU DO IN THIS STAGE

Understand the shape

How many samples, how many features, what is in each column.

See what is missing

Which fields are blank, where, and most importantly: why.

Check class balance

50/50 or 95/5? It changes the entire evaluation.

Spot outliers

Real biology, instrument glitch, or data-entry error?

Look for batch effects

Do samples cluster by run or site instead of by biology?

Match features to the question

Drop what the question does not need. Keep what it does.

PREPROCESSING · BEFORE NOTEBOOK 2.2

The model only sees numbers.

Two beginner traps come from forgetting that. Both have a one-line fix. Skip either and the model will learn the wrong thing, confidently.

TRAP 01 A category is not a number.

NAÏVE

ductal	1
lobular	2
other	3

If we encode tumor type as **`ductal = 1`**, **`lobular = 2`**, **`other = 3`**, the model may interpret this as an ordered scale.
That creates fake relationships: lobular is not “between” ductal and other.

FIX · ONE-HOT ENCODING

DUC	LOB	OTH
1	0	0
0	1	0
0	0	1

One column per category. Exactly one 1, the rest 0 .

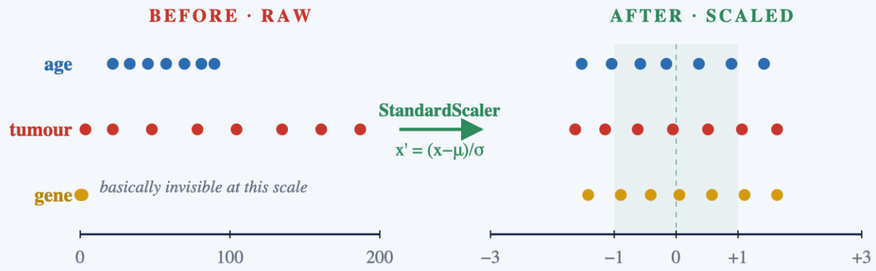
No order, no fake distances.

Rule. Use one-hot encoding for nominal/unordered categories. Use ordinal encoding only for truly ordered categories, and only when treating the order as meaningful for the model. For example, stage I/II/III can be ordinal, but one-hot may still be preferable if the effect is not expected to be linear or monotonic..

TRAP 02 A range is not an importance.

SAME DATA — RAW VS. STANDARDISED

For distance-based models, range alone decides influence.



On the left, tumour-mm dominates the space and gene-expression is a single dot. On the right, every feature is centred at 0 with std 1, now the model decides on signal, not on units.

Rule. Distance-based models *need* it (KMeans, KNN, SVM, PCA, linear/logistic, neural nets). Tree-based models do not.

NOTEBOOK · 2.2

A preprocessing pipeline, step by step.

`3.2_data_exploration_and_preprocessing.ipynb` `~30 min`

The notebook walks the data through five transforms, in order, then wraps the whole thing in one reusable object.

01

LOAD & LOOK

Read the data.

Read the CSV into a DataFrame. Print `.shape`, `.dtypes`, `.head()`. Know what you are holding before you touch it.

02

AUDIT

Find the gaps.

Where are the missing values, and how much is missing per column? Check class balance and outliers while you are here.

03

ENCODE

Numbers, not labels.

Turn every categorical column into one-hot binaries (or ordinal if it is truly ordered). Trap 1 fixed.

04

IMPUTE

Fill the gaps.

Replace missing values with a choice you can defend, mean, median, or KNN. Never the truth, always a stated assumption.

05

SCALE

Same playing field.

`StandardScaler` on numeric features so range no longer decides importance. Trap 2 fixed.

WRAP · THE PAYOFF

One Pipeline object.

```
Pipeline([ encode → impute → scale → model ])
```

One `.fit()` on training data. The exact same transforms re-applied automatically to test data. Skip the wrap and you will leak.

Two families: with labels, and without.

Supervised

labels available

Learn an input → output mapping from labelled examples. You know the answer for the training data.

classification regression

Unsupervised

no labels

Discover structure in data when you do not know the answer. Groups, gradients, outliers.

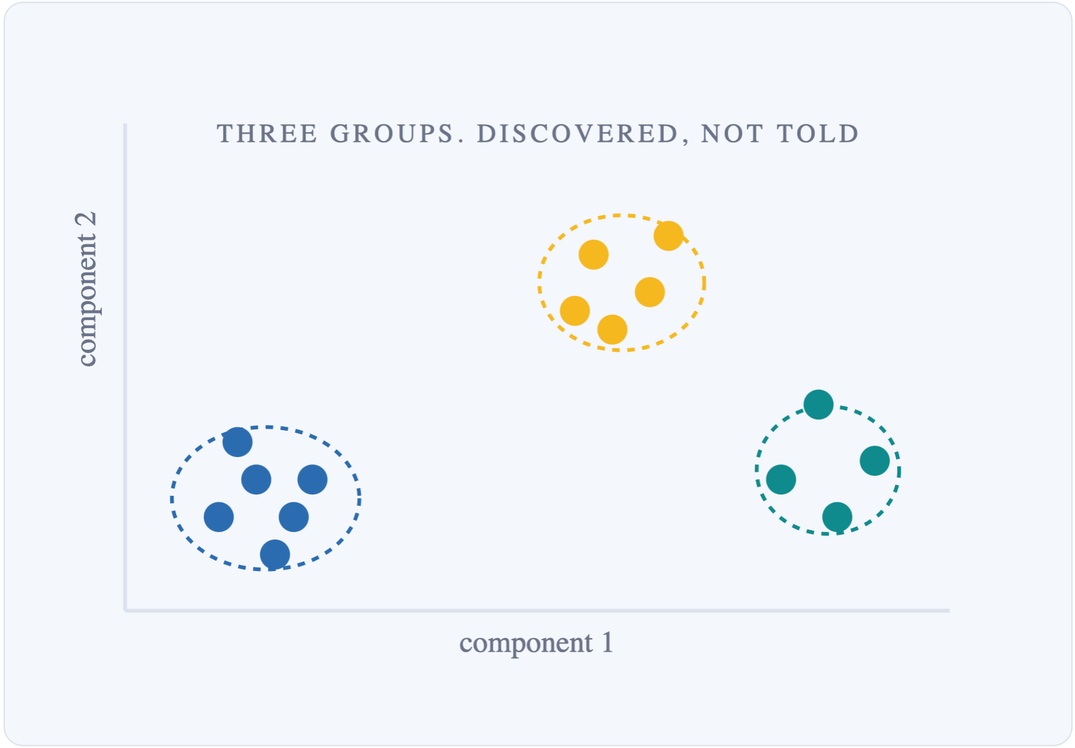
clustering dim. reduction

Unsupervised learning: no answer key.

You give the algorithm features only. It returns structure . Groups, projections, anomalies, gradients.

- **You decide afterwards** if the structure is real biology or an artefact.
- **Useful for exploration** . QC, batch detection, hypothesis generation.
- **Hard to evaluate**. No ground truth, internal metrics only.

Watchword: a cluster is not a subtype until a biologist validates it. The algorithm cannot tell you what is real.



UNSUPERVISED · BEFORE NOTEBOOK 2.3

How KMeans actually works.

Four steps. You pick K. The algorithm does the rest by minimising within-cluster variance.

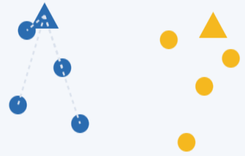
STEP 01

Initialise

Pick K centroids at random (or with k-means++ seeding).

STEP 02

Assign



Each point joins the nearest centroid (Euclidean distance).

STEP 03

Update

Move each centroid to the mean of its assigned points.

STEP 04

Repeat

Steps 2–3 until centroids stop moving (convergence).

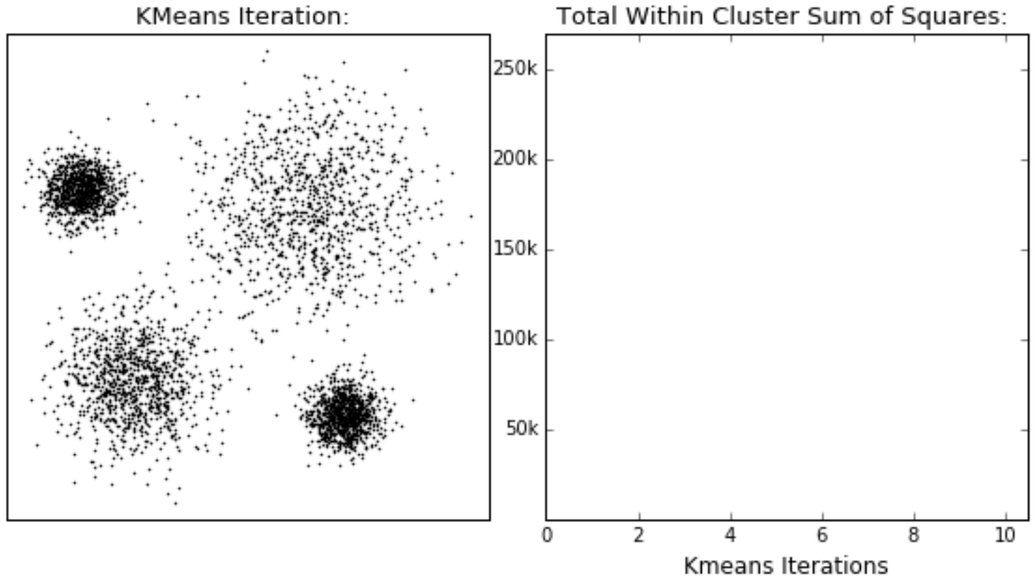
WHAT IT IS MINIMISING

$$\min_{\mu} \sum_{i=1..K} \sum_{x \in C_i} \| x - \mu_i \|^2$$

Total within-cluster variance. KMeans converges to a local minimum, different seeds, different answers.

WATCH OUT

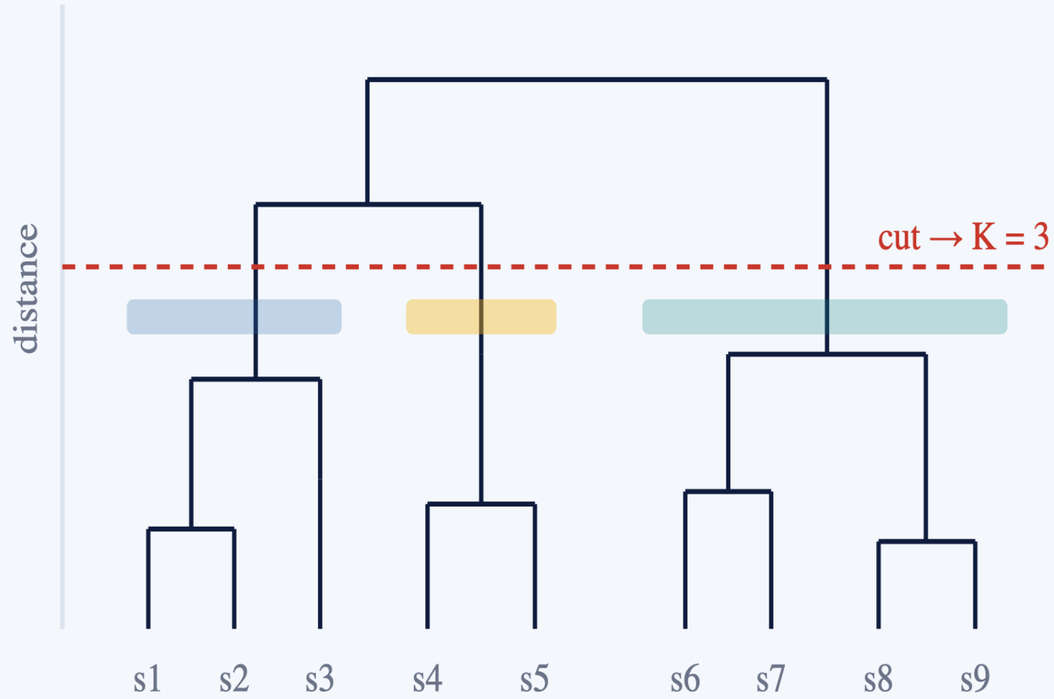
- You must pick K up front.
- Assumes round, equal-sized clusters.
- Sensitive to outliers and feature scale.



How hierarchical clustering works.

Bottom-up: start with each sample alone, merge the two closest groups, repeat until everything is one tree. You decide the cut.

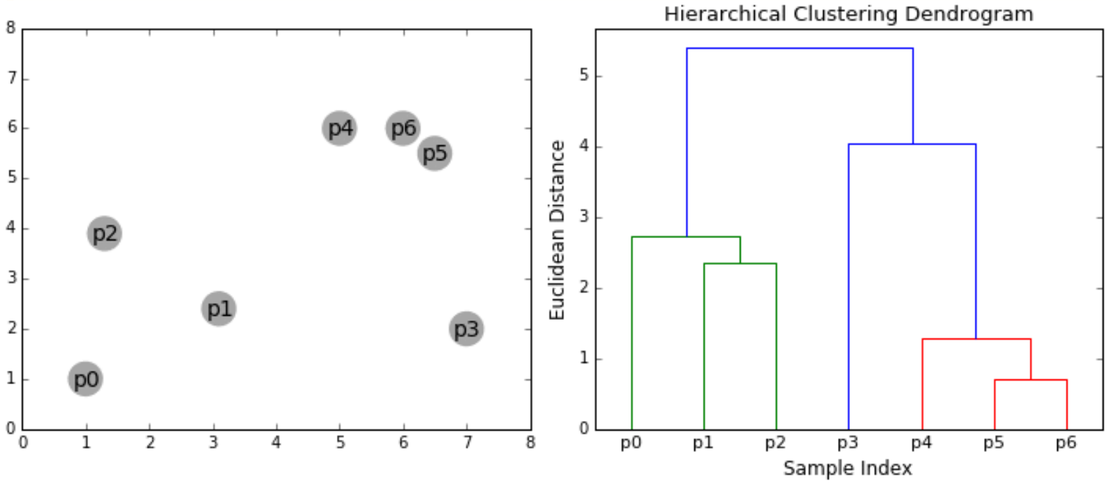
THE DENDROGRAM



Linkage: how do you measure "distance" between two *groups*?

`single`	distance of the *closest* pair across the two groups
`complete`	distance of the *farthest* pair
`average`	mean distance over all pairs
`Ward`	merge that grows variance least

You do not pick K up front. You pick where to *cut* the tree. Different cuts give different K. That is the strength, and the responsibility.

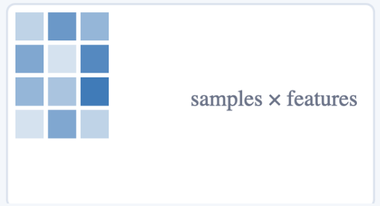


EXAMPLE 1 · BEFORE NOTEBOOK 2.3

Can unsupervised learning reveal meaningful patient groups?

STEP 01

Data matrix

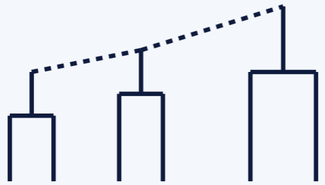


STEP 02

Clustering

STEP 03

Dendrogram

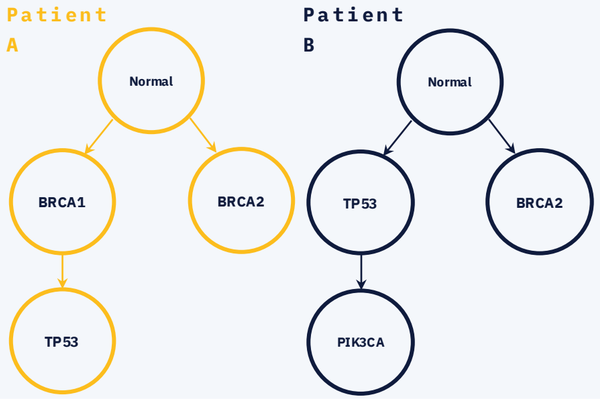


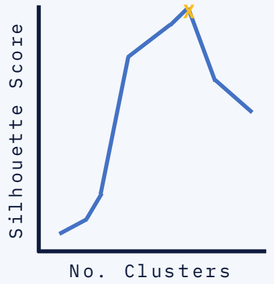
STEP 04

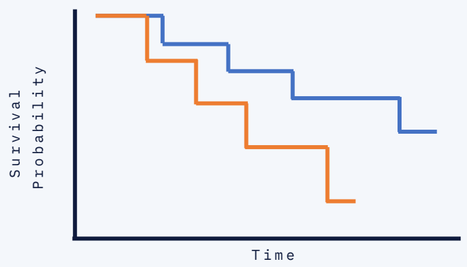
Annotate clinically

Do clusters map onto known clinical variables? Does any cluster suggest a new subgroup?

PROJECT







NOTEBOOK · 2.3

Unsupervised ML.

`3.3_unsupervised_ml.ipynb`

Your first models. Exploring structure without labels.

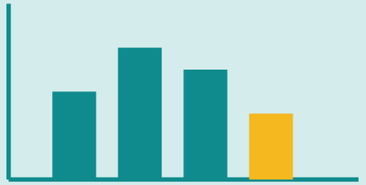
Hierarchical

Build a tree of sample similarity. Dendrograms.

KMeans

Group samples into K clusters by similarity.

Silhouette



Are your clusters any good? Score them.

Honest warning: a cluster is not a subtype. It is a hypothesis you validate with biology.

Supervised learning: an answer key.

Each sample is a pair: (X, y) . Features X, known outcome y. The model learns a function so that $f(X) \approx y$.

- **Training**. Show the model many pairs.
- **Generalisation**. Test it on pairs it has never seen.
- **Labels are everything** : wrong labels → confidently wrong model.

> **Biomedical reality:** labels come from charts, pathology, follow-up. They are noisier than they look.

EXAMPLE DATASET (BRCA)

ID	BRCA1	ESR1	AGE	Y
001	5.2	8.1	54	lobular
002	7.9	3.2	61	ductal
003	6.1	7.4	49	lobular
004	4.8	2.9	67	ductal
…	…	…	…	…

Features on the left. The label y on the right. That is the answer key.

Classification: the target is a category.

OUTPUT

A label per sample.

patient_042	lobular · p = 0.87
patient_043	ductal · p = 0.61
patient_044	other · p = 0.52

Most classifiers actually return a probability per class. A threshold turns probability into label.

Biomedical examples

- Tumour subtype (`ductal / lobular`)
- Vital status (`alive / deceased`)
- Sepsis in next 6h (`yes / no`)
- Drug responder (`R / NR`)

ALGORITHMS YOU WILL MEET

`logistic regression` `decision tree` `random forest`

Regression: the target is a number.

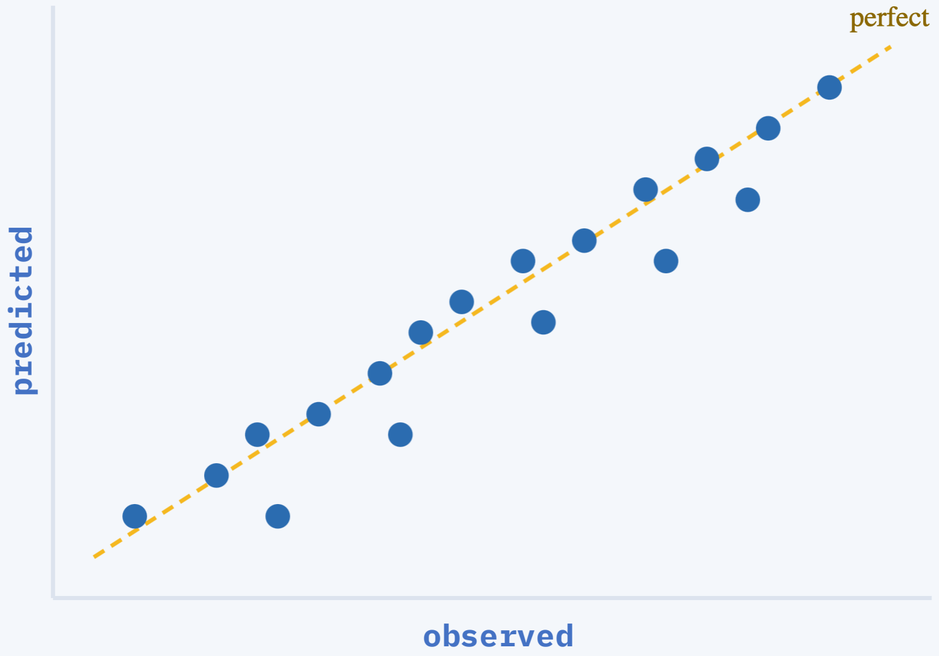
Biomedical examples

- Patient age (years)
- Risk score (0–1)
- Oxygen saturation (%)
- Gene expression level
- Drug dose

ALGORITHMS YOU WILL MEET

`linear regression`

PREDICTED VS OBSERVED



Can supervised models predict clinically meaningful groups from multi-omics?

STEP 01
BRCA data
Multi-omics + clinical features.

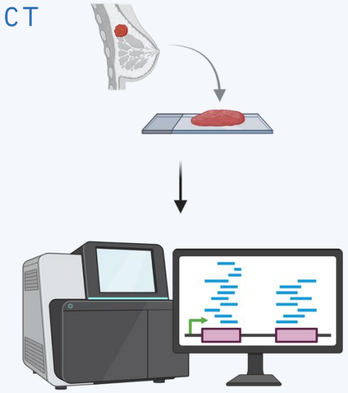
STEP 02
Preprocess
Impute, scale, select. On train only.

STEP 03
Model
Linear, Logistic, DT, RF.

STEP 04
Performance
CV + held-out test, with AUC.

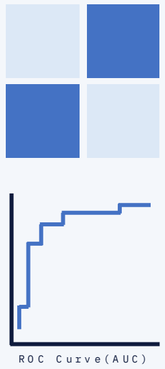
STEP 05
Interpret
Important features → biology.

PROJECT



Data:
- WES
- RNA-Seq
- Clinical

Logistic Regression with **LASSO**



What it teaches

— A solid pipeline beats a fancy model.

— Cross-validation is non-negotiable.

— Important features are *hypotheses*, not findings.

BRIDGE · BEFORE NOTEBOOK 2.4

Same concepts. Different questions.

You do not need to invent new methods to do good biomedical ML. You need to ask a good question, prepare data carefully, pick an appropriate model, and validate honestly.

methods are shared

questions are yours

NEXT →

The four supervised algorithms you will meet in Notebook 2.4: linear regression, logistic regression, decision trees, random forest.

SUPERVISED · BEFORE NOTEBOOK 2.4

Linear regression: fit a straight line through the cloud.

The simplest supervised model. A weighted sum of features, plus an intercept. Old, transparent, and surprisingly hard to beat.

THE MODEL

$$\hat{y} = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$$

OBJECTIVE – MINIMISE

$$RSS(\beta) = \sum_{i=1..n} (y_i - \hat{y}_i)^2$$

Ordinary least squares.

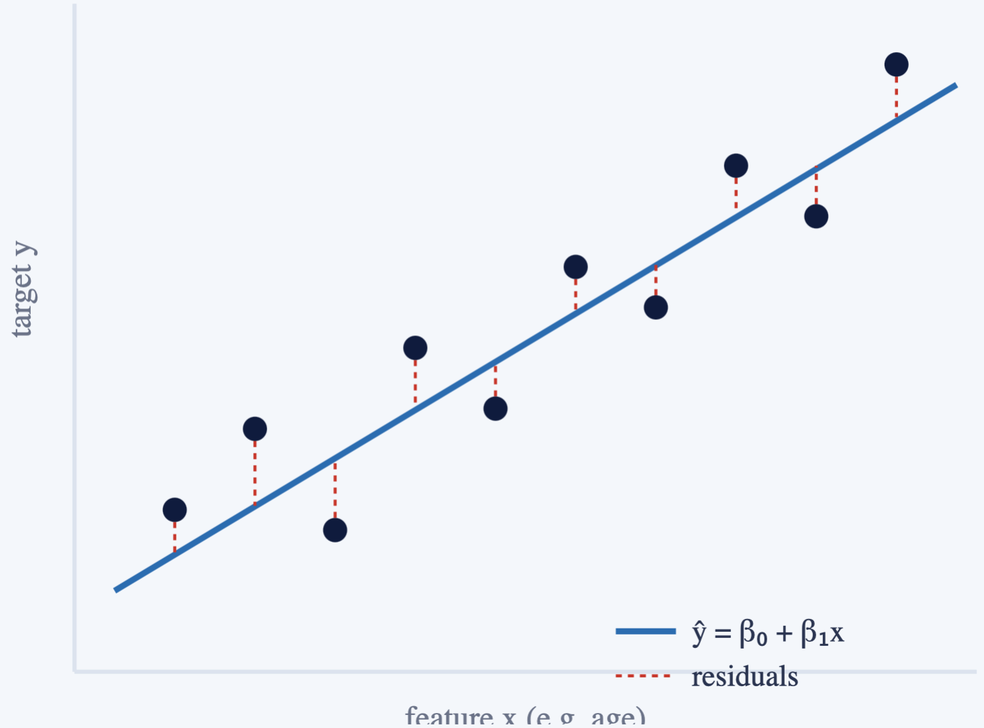
Why it is the right baseline

- One coefficient per feature, you can read the model off the page.
- Fast to fit.
- If a fancy model cannot beat it, the fancy model is not earning its complexity.

WATCH OUT

- Assumes a linear relationship. Real biology often is not.
- Sensitive to outliers and to feature scale.
- With many correlated features, use ridge or lasso.

FIT ON A SCATTER



Logistic regression: a line, squashed into a probability.

The classification cousin of linear regression. Same weighted sum of features — passed through a sigmoid so the output lives in [0, 1].

THE MODEL

$$\hat{p}_i = \sigma(\beta_0 + \beta^T x_i), \quad \sigma(z) = 1 / (1 + e^{-z})$$

OBJECTIVE — MINIMISE

$$L(\beta) = -\Sigma_i [\, y_i \log \hat{p}_i + (1-y_i) \log(1-\hat{p}_i) \,]$$

Negative log-likelihood (a.k.a. binary cross-entropy). Convex — one global optimum, solved by gradient descent.

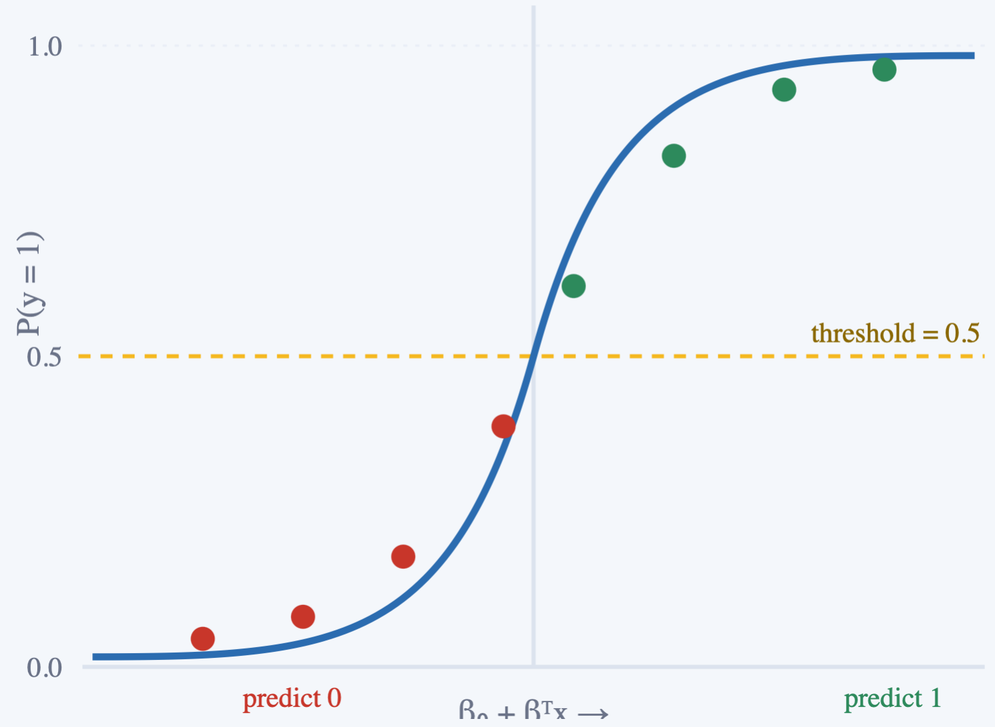
The right default for classification

- Calibrated probabilities - useful for clinical thresholds and ROC curves.
- Coefficients are log-odds: a positive β_j means feature j pushes toward the positive class.
- Add L1 or L2 regularisation when features outnumber samples.

WATCH OUT

- The decision boundary is still *linear* in the features.
- Needs feature scaling for fair coefficients.
- Probability ≠ causality. Same warning as every model today.

THE SIGMOID & THE THRESHOLD



Decision trees: a series of yes / no splits.

At every node, pick the feature and threshold that best separates the classes. Recurse on each side until the leaves are pure (or small) enough.

NODE IMPURITY

$$G(\text{node}) = 1 - \sum_k p_k^2$$

OBJECTIVE — AT EACH SPLIT, MINIMISE

$$\frac{n_L}{n} G_L + \frac{n_R}{n} G_R$$

Greedy. Try every feature and threshold, keep the one with the lowest weighted child impurity. Recurse.

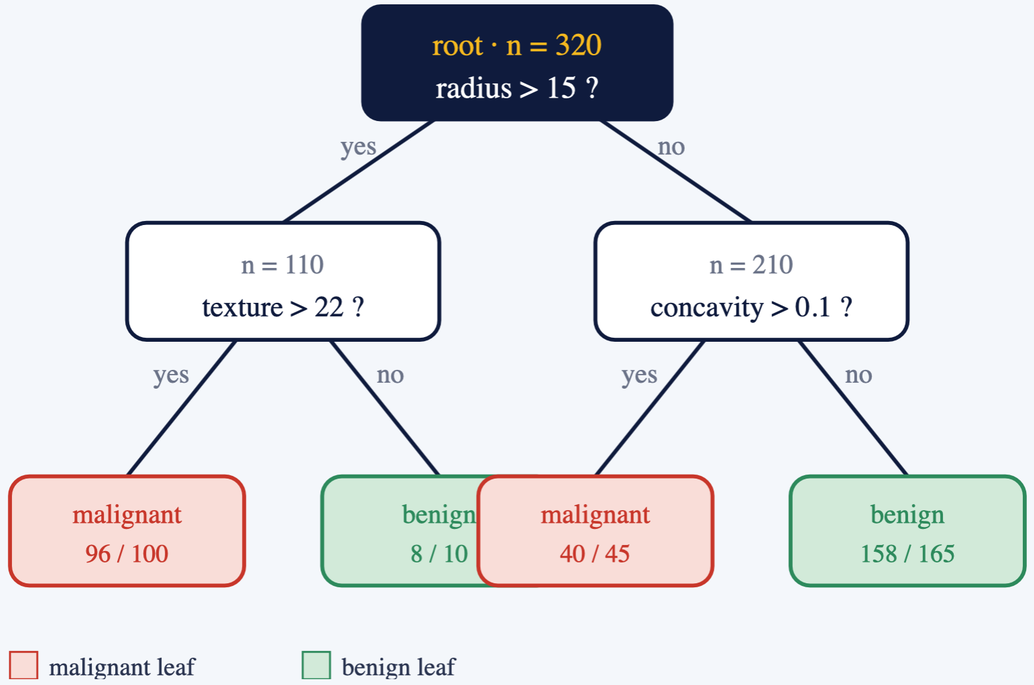
Why people reach for trees

- Reads like a clinical decision rule. You can draw the whole model.
- Handles numeric and categorical features without scaling.
- Captures non-linear interactions.

WATCH OUT

- Unconstrained trees *memorise* the training data.
- Tiny changes in data → very different tree. High variance.
- Constrain with `max_depth`, `min_samples_leaf`, or use a forest.

TOY TREE · IS THE TUMOUR MALIGNANT?



Random forest: many trees voting.

Grow hundreds of decision trees, each on a random sample of the data and a random subset of the features. Average their predictions. Variance collapses.

THE RECIPE

1. Bootstrap: resample rows with replacement.
2. At each split, consider only a random subset of features.
3. Grow each tree deep. Repeat for T trees.
4. Predict by majority vote (classification) or average (regression).

OBJECTIVE — EACH TREE MINIMISES

`weighted Gini at every split` (same as a single tree)

No global loss for the forest. Diversity + averaging is what cancels variance.

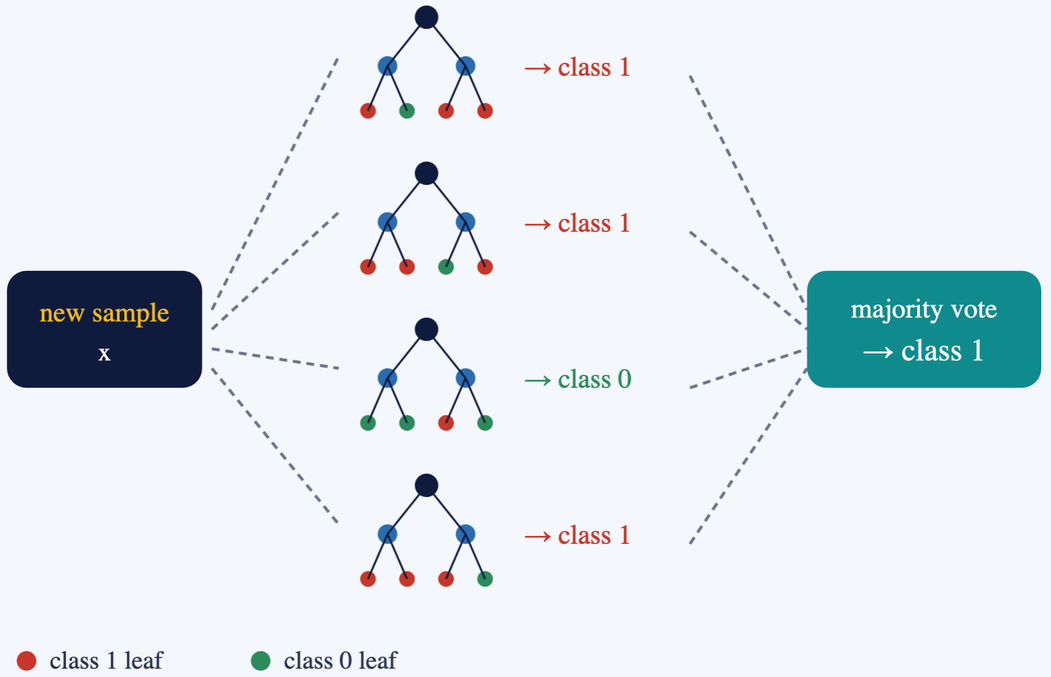
Why it is the strong default

- Trees disagree in different ways, their errors cancel.
- Robust out of the box. Few hyperparameters that really matter.
- Built-in *feature importance* and out-of-bag error estimates.

WATCH OUT

- You can no longer read the model as a single rule.
- Importance ranks features, it does not explain causality.

BAG OF TREES → ONE VOTE



How well does it work? Split, then measure.

A model that has only seen its training data tells you nothing about new patients. Hold out data the model never sees, then score on it.

WHY NOT JUST SCORE ON TRAINING?

A flexible model can *memorise* the training set and report 100%. That number says nothing about a new patient walking in tomorrow.

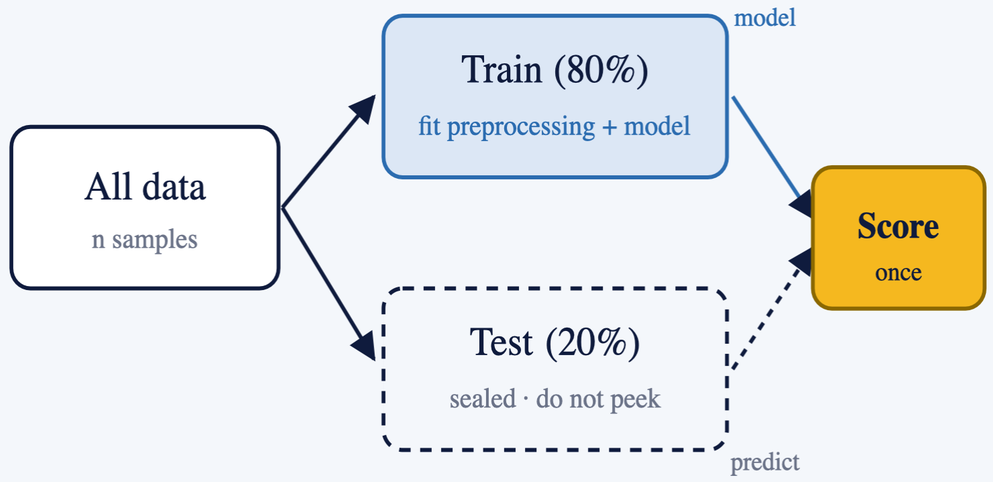
The one-line rule

- Split *before* you touch the data. 80 / 20 or 70 / 30 is typical.
- Stratify the split — same class balance in both halves.
- Fit everything (preprocessing, model) on *train*.
- Score on *test* exactly *once*, at the very end.

WHY ONE NUMBER IS NOT ENOUGH

"Accuracy" hides which patients you got wrong. A screening model and a confirmatory model can have the same accuracy and very different clinical value. The next slides unpack the metrics.

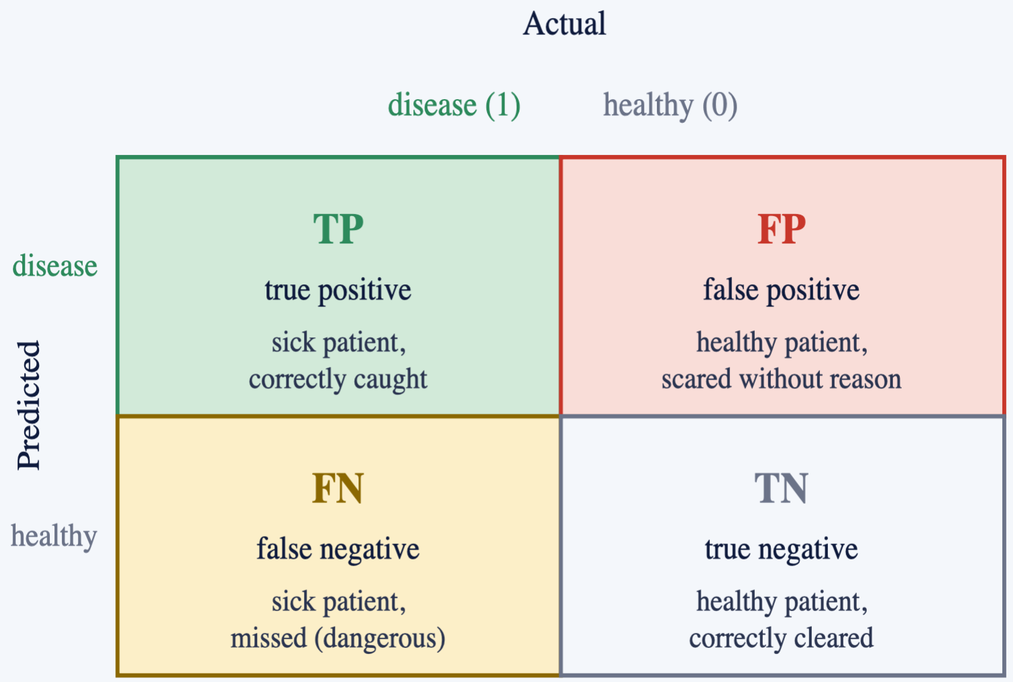
THE HONEST SPLIT



Every classification metric starts here.

Cross-tabulate predictions against truth. Four cells. Every number you will ever report is just a way of squeezing them.

2 × 2 — BINARY CASE



All four costs are different. The metric you pick decides which one matters.

READ IT AS A CLINICIAN

- **FN is usually the worst** — a missed cancer.
- **FP is the next worst** — a healthy patient told they are sick.
- TP and TN are the cells you want to grow.

WHY WE NEVER REPORT A SINGLE NUMBER

A model that predicts "healthy" for everyone has TP = 0, FN = all sick patients, and very high TN. Accuracy looks fine. The model is useless. The matrix tells you that. Accuracy does not.

Multiclass works the same way

For K classes you get a K×K matrix. Diagonal = correct. Off-diagonal = which classes get confused for which.

EVALUATION · BEFORE NOTEBOOK 2.5

The four numbers everyone reports.

Same confusion matrix, four ways to squeeze it. Each answers a different clinical question. Pick the one that matches your cost of being wrong.

ACCURACY

Overall hit rate

$$TP + TN \,/\, all$$

Fine when classes are *balanced*.

Trap: 95% accuracy on a 95/5 class split is the trivial "predict majority" model.

RECALL — SENSITIVITY

Catch all the sick

$$TP \,/\, (TP + FN)$$

Of all the truly sick patients, what fraction did we catch?

Use for screening. Missing a real case is the costly error.

PRECISION — PPV

Trust the alarms

$$TP \,/\, (TP + FP)$$

Of the patients we flagged as sick, what fraction really were?

Use for confirmation. A false alarm scares a healthy patient.

F1 SCORE

Balance the two

$$2 \cdot P \cdot R \,/\, (P + R)$$

Harmonic mean. Punishes you for being good at only one.

Use when classes are imbalanced and you care about both errors.

In medicine, recall and precision pull in opposite directions.

A screening test should miss nobody , so we push recall up — even if precision drops and many healthy patients get a scary follow-up call. A confirmatory test does the opposite: we will not tell a patient they have cancer unless we are sure, so precision goes up and recall accepts a few misses. The same model, two different thresholds, two different clinical roles.

The whole picture: ROC and AUC.

Recall and precision depend on the threshold you pick. Sweep the threshold from 0 to 1, plot the trade-off, and you get one curve and one number that summarise the whole model.

THE TWO AXES

`TPR = TP / (TP + FN)` `(= recall)`

`FPR = FP / (FP + TN)` `(= 1 − specificity)`

For every threshold from 0 to 1, count both rates. Plot them. Connect. That is the ROC curve.

What AUC actually means

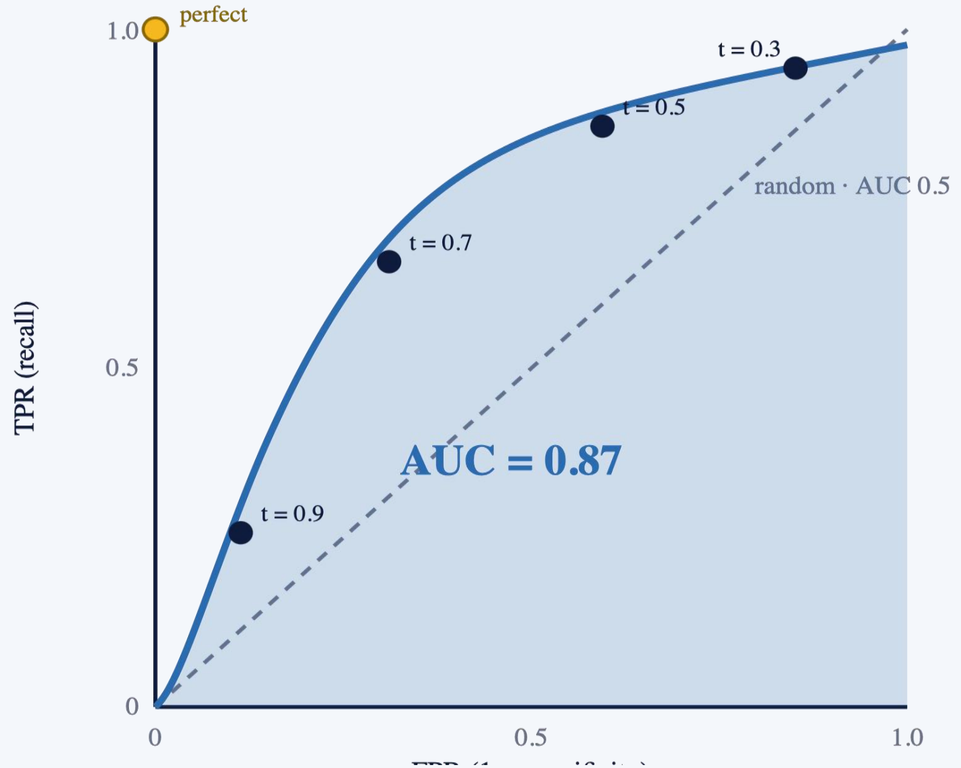
The probability that a random *positive* patient is ranked higher by the model than a random *negative* one.

- `AUC = 1.0` — perfect ranking.
- `AUC = 0.5` — coin flip.
- `AUC < 0.5` — usually a flipped-label bug.

WATCH OUT

- AUC is *threshold-free* . It tells you ranking quality, not which threshold to pick.
- On imbalanced data, the *precision-recall* curve is more honest.

AN ROC CURVE



Leakage. The silent performance killer.

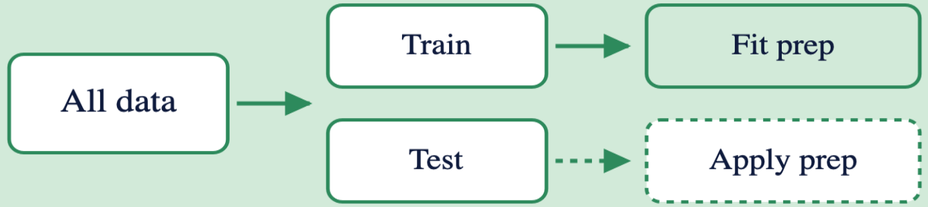
WRONG

Preprocess → split.

Test info leaks into training. Numbers look great. Reality does not.

RIGHT

Split → preprocess (train only).



Test stays sealed. Preprocessing learns from train only.

Split. Then preprocess. Always. No exceptions. Most published biomedical ML failures trace back here.

Interpretation is not causality.

Important feature =

…a variable the *model* uses to predict.

…not necessarily =

…a variable that *causes* the outcome.

It might be:

A marker

A confounder

A leak you missed

Important features are hypotheses. For the wet lab, not the conclusions section.

NOTEBOOK · 2.4

Supervised ML.

3.4_supervised_ml.ipynb

~25 min

The main event. Fit five models. Compare. Pick a winner. For a reason you can defend.

MODEL 01

Linear regression

Your baseline. Linear, interpretable, often hard to beat. (continuous)

MODEL 02

Logistic Regression

Your baseline. Linear, interpretable, often hard to beat. (binary)

MODEL 03

Decision tree

A series of yes/no splits. Easy to read, easy to overfit.

MODEL 04

Random forest

Many trees voting. Strong default on tabular data.

You will also start asking: which features matter?. And we will be careful about how much we believe the answer.

When would you trust a simpler model over a stronger one?

NOTEBOOK · 2.4

Performance metrics & pitfalls.

3.4_supervised_ml.ipynb ~25 min

The discipline layer. We will deliberately build a leaking pipeline, watch it cheat, and fix it.

Honest evaluation

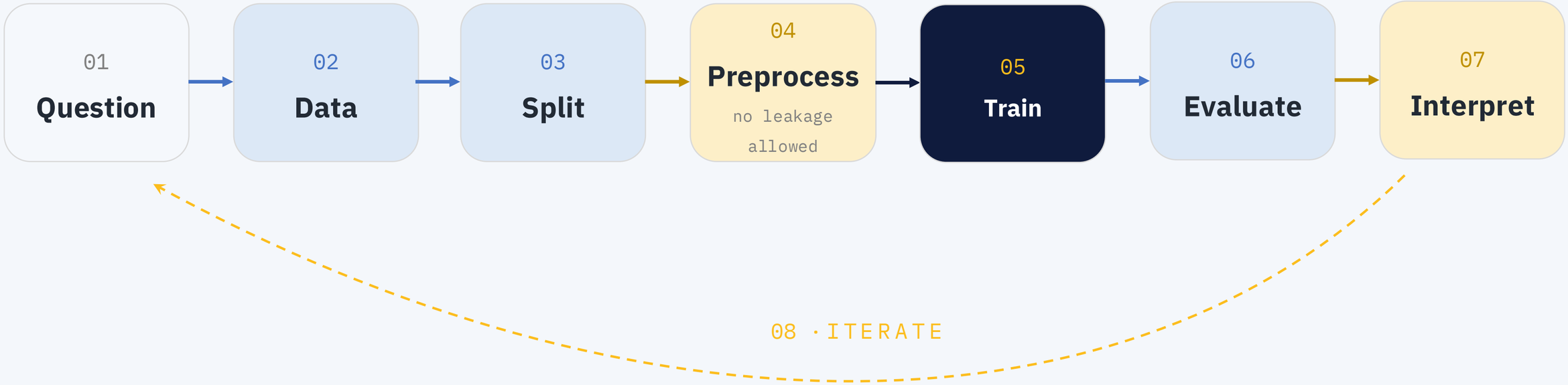
- Train / test split. No peaking.
- Confusion matrix. The source of truth.
- Accuracy · precision · recall · F1.
- ROC Curve and AUC.

Pitfalls to break on purpose

- **Leakage**. Preprocess before splitting.
- **Confounding**. Site, batch, age.
- **Overfitting**. Memorising noise.
- **Causality confusion**. Important ≠ causal.

Promise: once you have seen a leaking pipeline live once, you will spot it for the rest of your career.

The golden ML workflow.



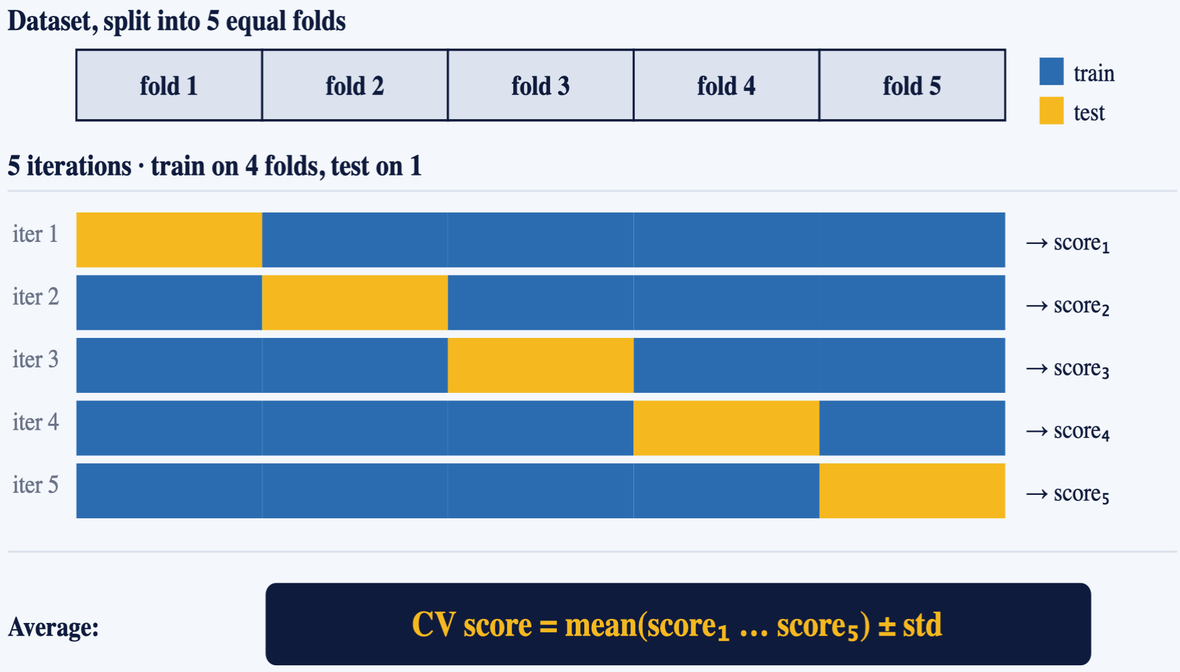
VALIDATION · BEFORE NOTEBOOK 2.6

A single split is not enough. Use k-fold CV.

One train-test split tells you how the model did on *that one* test set. CV repeats the experiment K times so every sample gets a turn on the bench.

5-FOLD CROSS-VALIDATION

Every sample tested exactly once.



WHY BOTHER

- **Less luck.** One bad split can flatter or punish the model. K splits average that out.
- **Uncertainty.** The std across folds tells you how stable the score is.
- **More data used.** Every sample is used for both training and evaluation.

TWO GROUND RULES

- **Stratify** on the target when classes are imbalanced.
- Fit preprocessing *inside* each fold — never on the full set first, or you leak.

VALIDATION · BEFORE NOTEBOOK 2.6

Picking hyperparameters with grid search + CV.

Every model has knobs you have to set *before* training. Tuning honestly means: try a grid of values, score each with CV, pick the winner.

GRIDSEARCHCV · EXAMPLE: A RANDOM FOREST

3 × 3 = 9 combinations, each scored by 5-fold CV.

n_estimators \ max_depth	3	5	10
50	0.81 ± 0.04	0.84 ± 0.03	0.83 ± 0.05
100	0.85 ± 0.03	**0.89 ± 0.02**	0.86 ± 0.04
200	0.84 ± 0.05	0.87 ± 0.04	0.82 ± 0.06

winner
depth=5, n=100

each cell = mean(CV scores)

Then once, at the end:

Refit on full train
with the winning combo

→

Score on held-out test
touched exactly once

→

Honest score
the number you report

THE THREE-STEP RECIPE

01 **Define a grid** of hyperparameter values. Coarse first, then refine.

02 **Score every combination** with K-fold CV on the training data. Same fold split for all.

03 **Pick the winner**, refit on the full training set, evaluate once on held-out test.

MODEL SELECTION · SAME LOOP

Comparing *across* model families (logistic vs. RF vs. gradient boosting) is the same procedure, just put **model** in the grid alongside its hyperparameters. The CV score decides.

WATCH OUT

- The held-out test set is **never** in the grid search. Use it once.
- Tune inside a Pipeline so preprocessing is refit per fold.

NOTEBOOK · 2.6

Hyperparameter tuning & pipelines.

`3.5_cv_leakage_tuning.ipynb` `~30 min`

Put the pieces together into a single reusable object.

TOOL 01

Pipeline object

Bundle preprocessing + model. Fit once. Reuse safely.

TOOL 02

Cross-validation

K-fold estimate of performance. Without burning extra data.

TOOL 03

GridSearchCV

Systematic search over hyperparameters with proper CV.

TOOL 04

Random seed

random_state=42 . Reproducible runs. Paper-ready.

The contract: touch the held-out test set *once*. After all tuning. Never again.

QUESTIONS · DISCUSSION

Thank you. Questions?

The notebook link is at the top of each section. We are here for the rest of the week.
Ask anything. Especially the dumb questions.

CONTACT

Laura Quintas

FMUL · iSTARS