ARTIFICIAL INTELLIGENCE 2

Machine Learning Basics

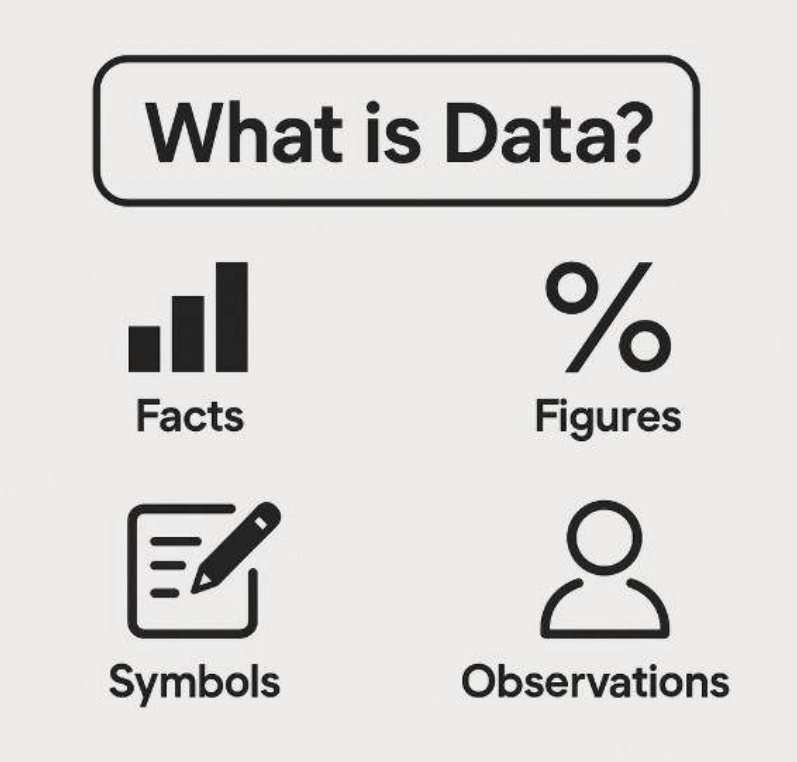
Francesco Spinnato

* francesco.spinnato@unipi.it
University of Pisa

Data

What is Data? What is Healthcare Data?

Data



Data is **information** collected, measured, or observed to support analysis, decision-making, and knowledge creation.

It can exist in various forms and levels of organization.

Structured and Unstructured Data

Structured

- Organized with a predefined schema
- Stored in tabular formats
- Common in relational databases
- Examples: transaction logs, customer tables, patient records

Unstructured

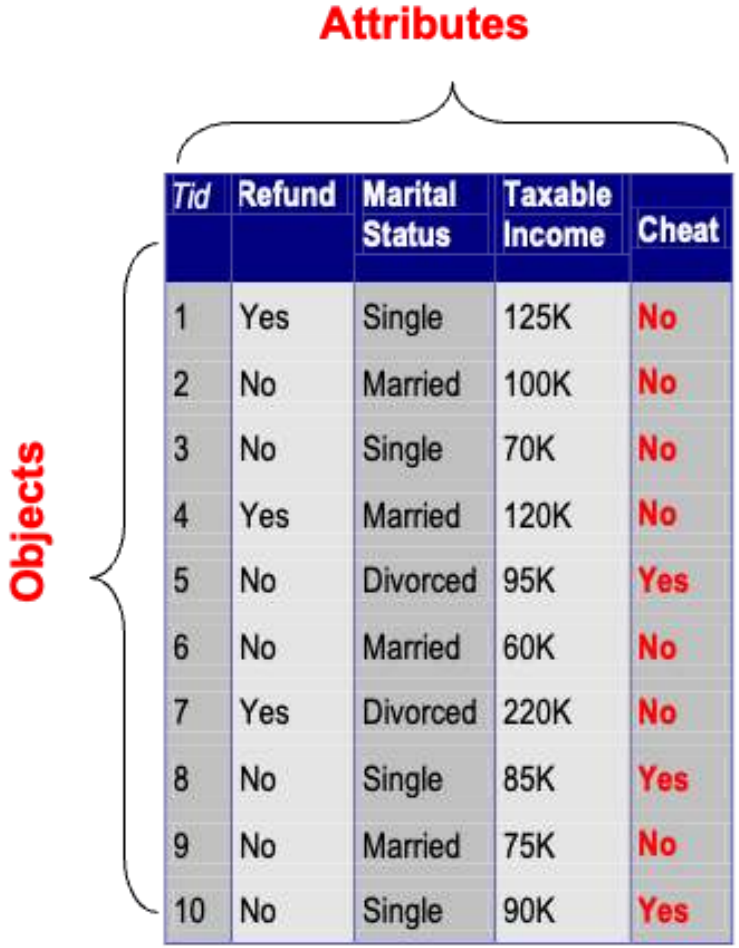
- Lacks a defined structure or schema
- More complex to store and analyze
- Examples: text documents, images, audio, video, social media content

Both kinds of data are present in the electronic health records (EHRs)

What is a (tabular) Dataset?

Structured collection of **data objects** and their **attributes**.

- An attribute is a property or characteristic of an **object**:
  - E.g. if the "object" is a patient, attributes could be age, sex, income...
- Attribute is also known as variable, field, characteristic, dimension, or feature. A collection of attributes describes an object
- Object is also known as record, point, case, sample, entity, or instance



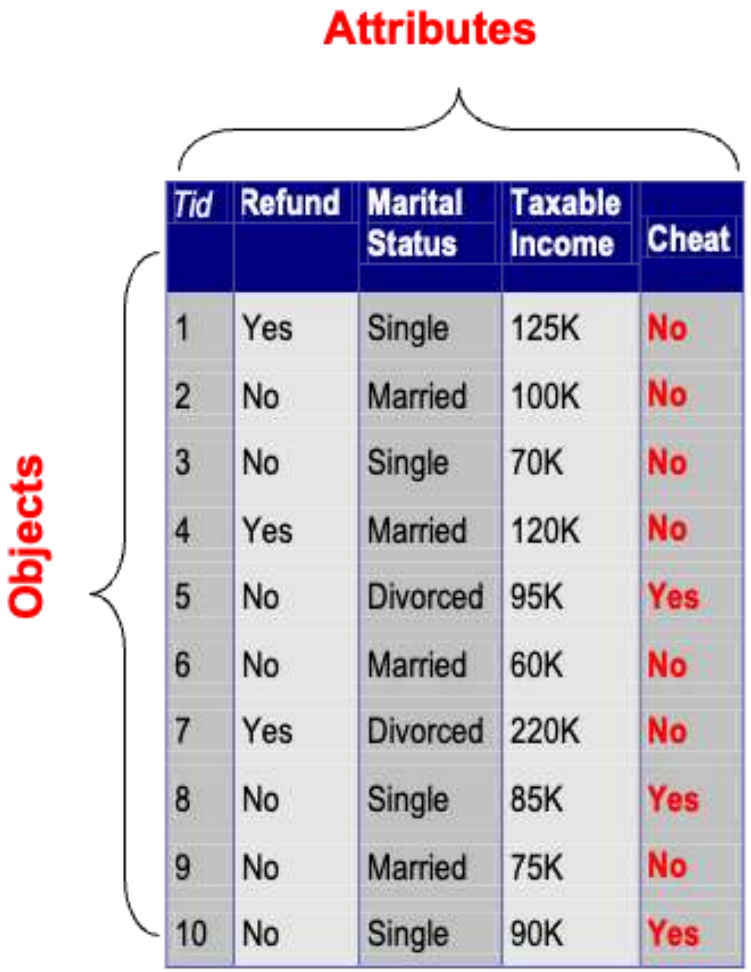
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Notation for (tabular) Datasets

- Dataset: $\mathbf{X}$ (uppercase bold)
- Instance: $\mathbf{x}_i$ (lowercase bold),
  - e.g. $\mathbf{x}_1 = [\text{yes}, \text{Single}, 125\text{K}, \text{No}]$
  - e.g. $\mathbf{x}_6 = [\text{No}, \text{Married}, 60\text{K}, \text{No}]$
- Entry: $x_{i,j}$
  - e.g. $x_{6,2} = \mathbf{X}[6, 2] = \text{Married}$

Usually, the first index is denoted as $i \in [1, N]$ (N number of instances).

While the second is $j \in [1, M]$ (M number of attributes).

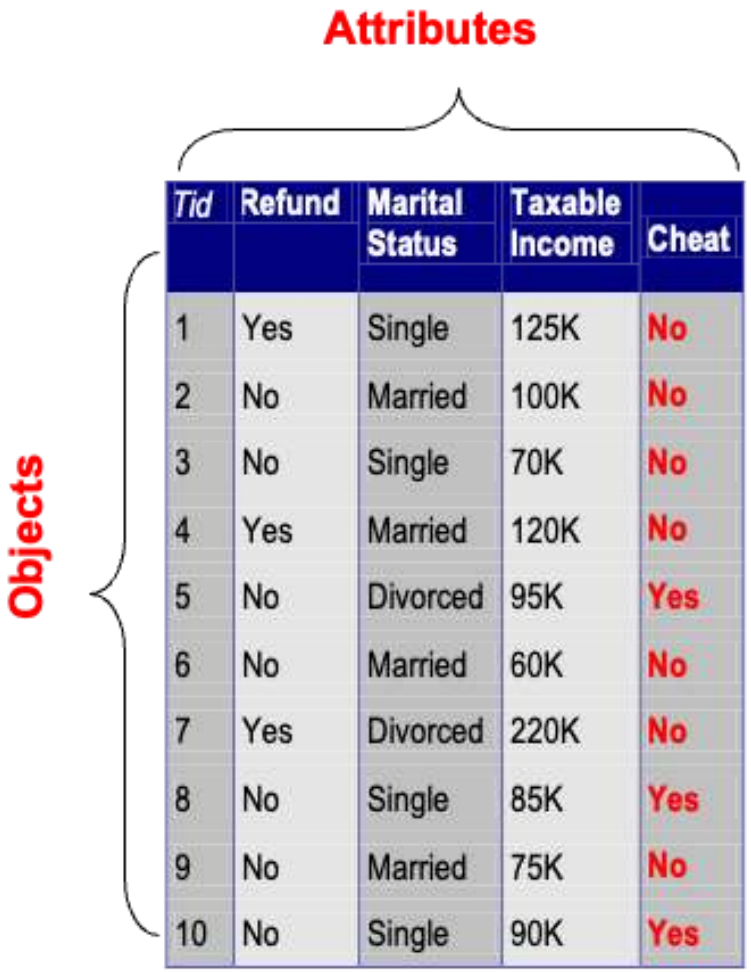


Notation for (tabular) Datasets

Some attributes can be a **target** variable (i.e. something you want to predict).

In this case they are often denoted as $\mathbf{y}$. In the example the target variable is *Cheat*.

So, for example, $y_6 = \text{No}$.



Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Structured data in electronic healthcare

Structured data are common in healthcare and are often represented using medical codes for diagnoses, procedures, medications...

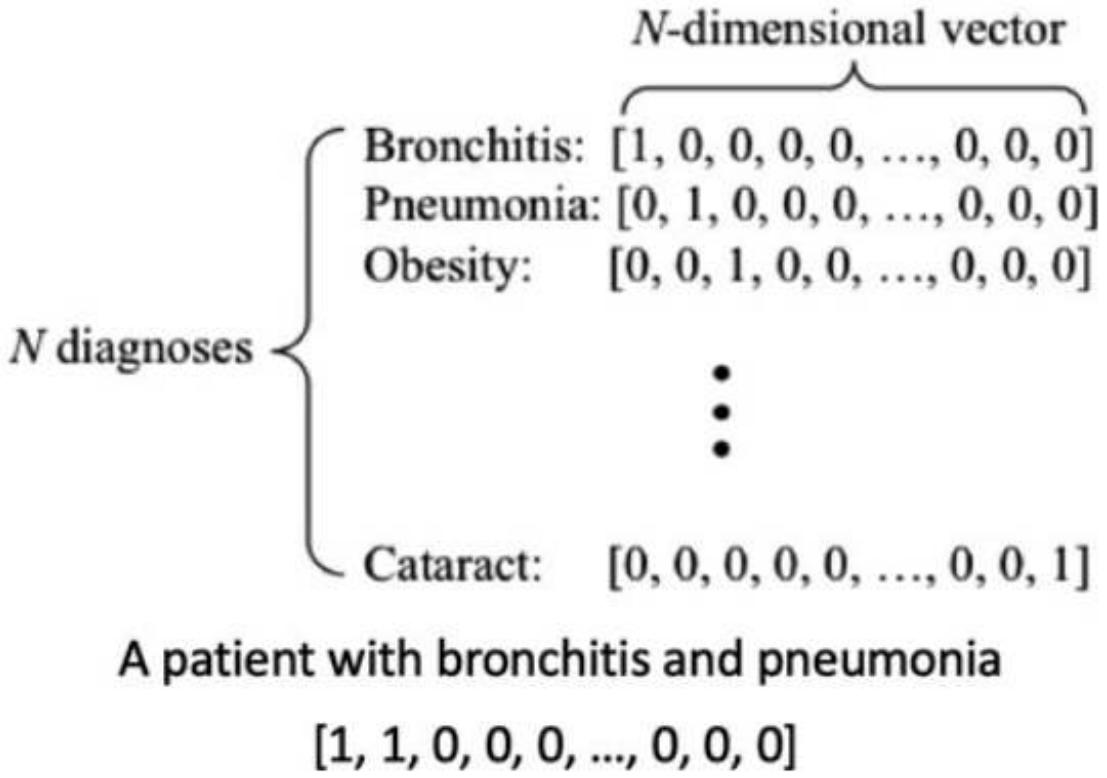
Type	Standard	Example
Diagnoses	International Classification of Disease (ICD)	S06.0x1A
Procedures	Current Procedural Terminology (CPT)	70010
Labs	LOINC (Logical Observation Identifiers Names & Codes)	5792-7
Medication	RxNorm, ATC	—
Demographics	NA	Gender, age
Vital signs	NA	Numeric
Behavioral status	NA	Smoking status

Challenges of structured healthcare data

Several challenges arise when analyzing structured healthcare data:

Sparsity: medical codes and similar features are often very high-dimensional and extremely sparse.

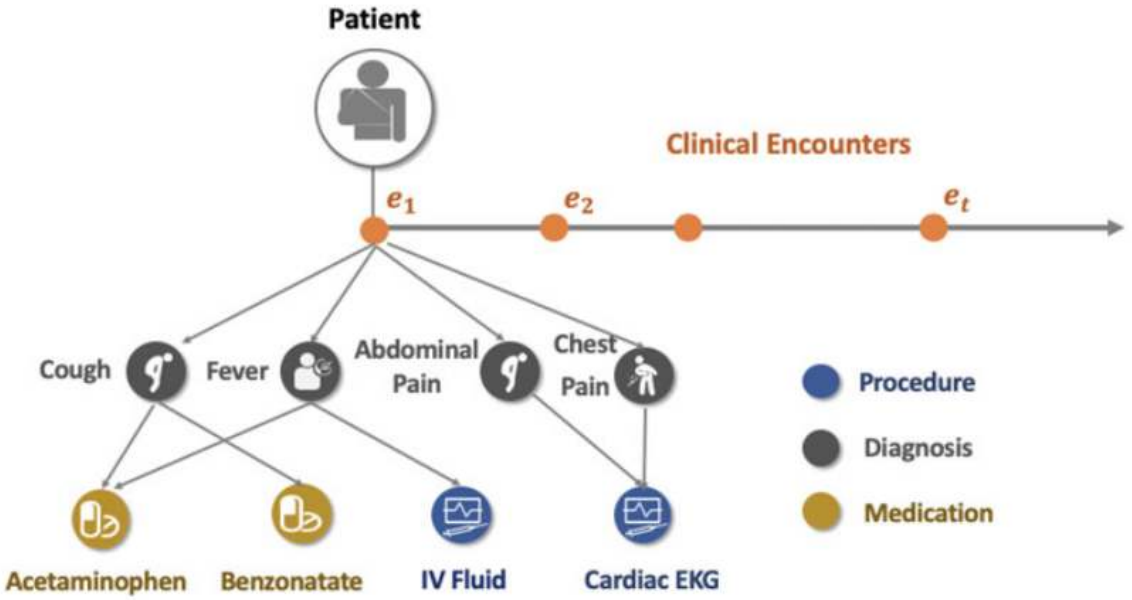
Example: with tens of thousands of possible ICD codes, each patient record contains only a few active entries.



Challenges of structured healthcare data

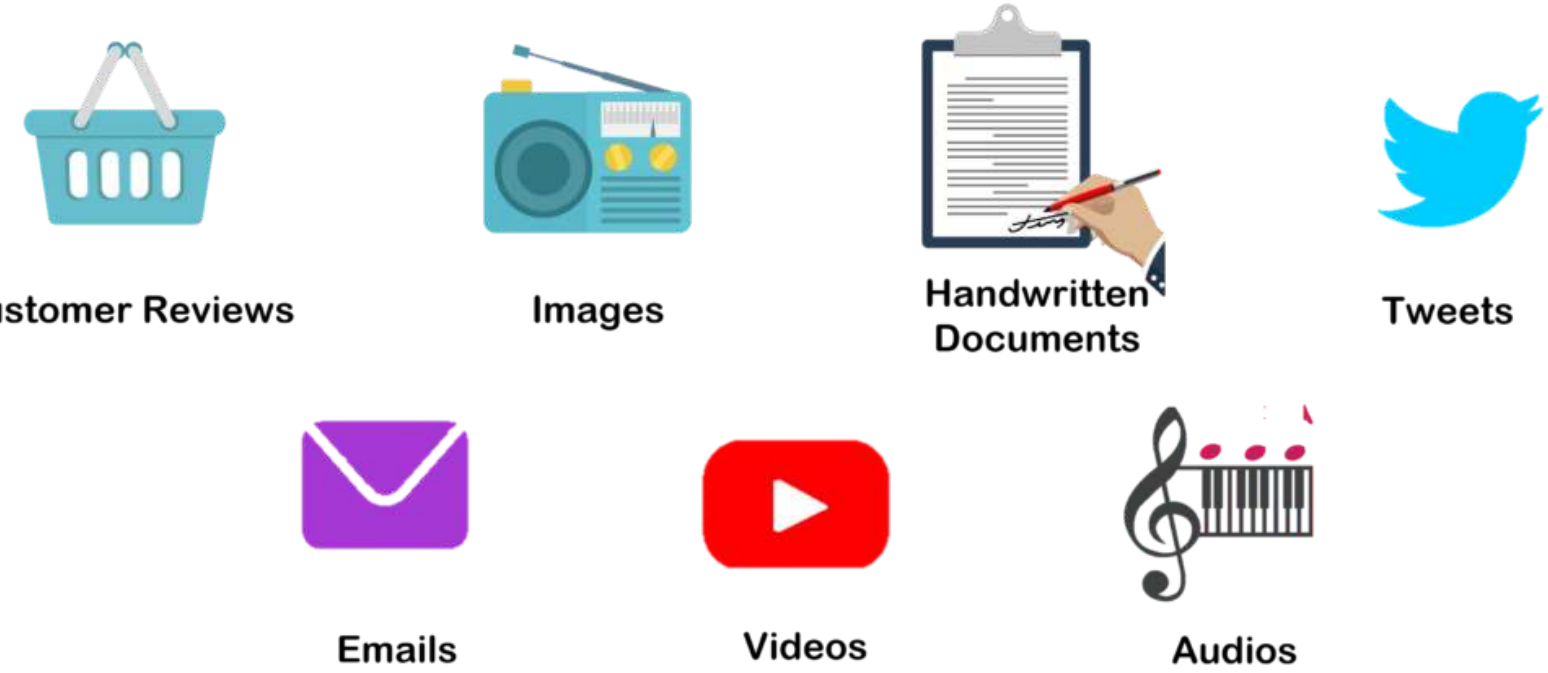
Temporality: healthcare data has an important time component.

Example: a patient has multiple visits over time, and the order and timing of events matter for prediction.



Unstructured Datasets

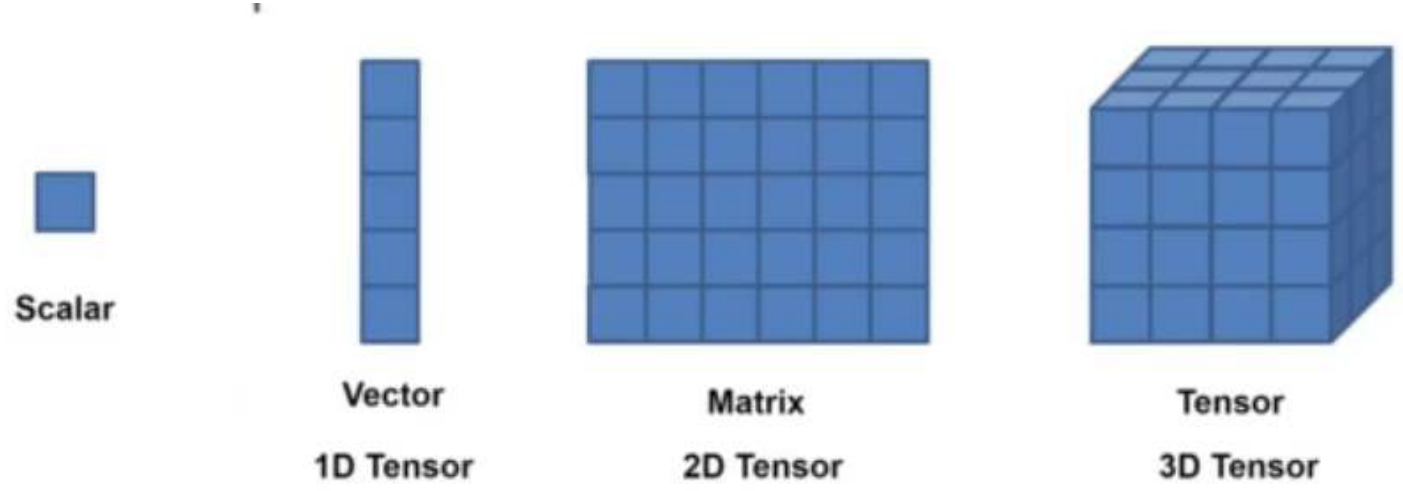
Unstructured data can come in different formats: text, images, time series, graphs...



Notation for Unstructured Datasets

Notation is more arbitrary for unstructured datasets but in general input data is denoted with the letter x:

- x (scalar, i.e., a number)
- $\mathbf{x}$ (a vector, i.e., a univariate time series)
- $\mathbf{X}$ (a matrix, i.e., a black and white image)
- $\mathcal{X}$ (a tensor, i.e., a colored image)



Unstructured data in electronic healthcare

Data type	Examples	Format
Clinical text	Discharge summaries, progress notes, reports	Text
Medical imaging	X-ray, MRI, CT, microscopy	Image
Physiological signals	ECG, EEG, vital signs, lab trajectories	Time series
Biological networks	Gene regulation, protein interactions, cells	Graph
Omics data	Genomics, transcriptomics, proteomics	Sequence / Table
Multimodal	Patient record: text + image + signals + labs	Mixed

Healthcare Data

Different players:

Different data sources:

Providers

- Deliver care in hospitals, clinics, and health systems
- Capture visit details in EHRs: diagnosis, procedures, medications, labs, imaging, clinical notes
- Exchange data with payers, pharmacies, labs
- Generate structured and unstructured clinical data

Payers

- Private or public insurers
- Receive claims from providers and pharmacies
- Process structured data: diagnoses, procedures, medications, costs
- Reimburse services fully or partially

Patients

- Central actors interacting with all healthcare players
- Access EHR data through portals and apps
- Generate self-tracked data via wearables (e.g., activity, heart rate)
- Use insurance benefits administered by payers

Pharmacies

- Dispense prescribed medications
- Know what and when patients fill prescriptions
- Produce pharmacy claims submitted to payers
- Contribute medication utilization data

Pharmaceutical Companies and Contract Research Organizations

Pharmaceutical Companies

- Discover, develop, and market drugs
- Run clinical trials to validate efficacy and safety
- Generate experimental, preclinical, and clinical trial datasets
- Submit trial results to regulators for approval

CROs

- Provide outsourced preclinical and clinical research services
- Manage clinical trials for pharmaceutical companies
- Produce datasets supporting regulatory submissions and drug development

Government Agencies

- Regulate drugs, approve new therapies, and oversee post-market safety
- Monitor, control, and prevent diseases
- Maintain spontaneous reporting systems for adverse event surveillance

Researchers

- Conduct basic, translational, and clinical research
- Use EHRs, trials, biological data, and real-world evidence
- Produce medical literature and clinical guidelines

The Life Cycle of Health Data

- Clinical encounter recorded in EHR
- Notes, diagnoses, procedures, medications, labs, imaging generated
- Claims submitted to payers; reimbursement processed
- Pharmacies dispense medications and submit pharmacy claims
- Pharma and CROs collect trial data for drug development
- Government Agencies review trial results for approval

Models

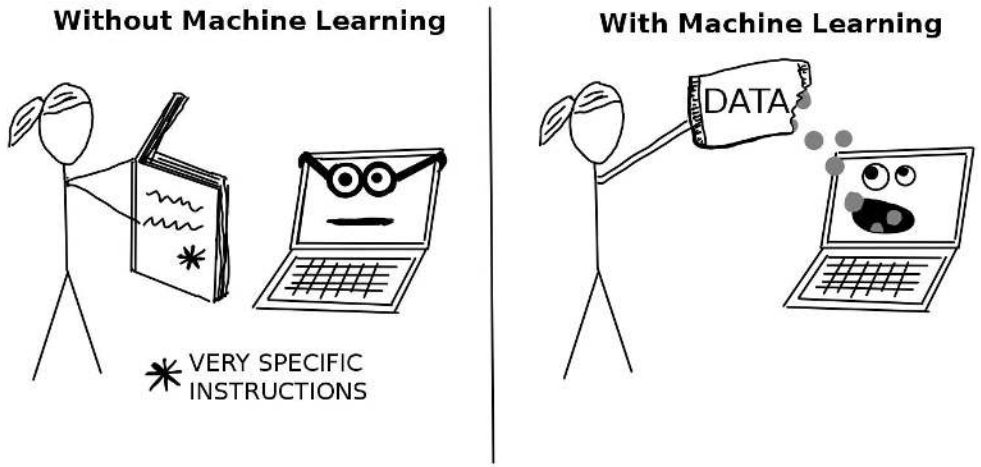
Learning from Data

Learning from Data

Learning from data is a central idea in Machine Learning.

A **machine learning model is a parameterized function** that represents a relationship between inputs and outputs.

A learning algorithm **adjusts the parameters of the model** using data.



Learning and Inference

Learning (training)

- The model is fit to data (training set) by an optimization procedure.
- In this phase the model changes, i.e., its parameters are learned.

Inference (prediction)

- The trained model is applied to (usually) new data.
- In this phase the model is fixed and does not change.

Loss function

To learn a model, we define a **loss function** $\mathcal{L}$ that measures how well (or not) the model is doing.

Learning consists in finding parameters that minimize the loss on the training data.

The choice of loss function depends on the task.

Supervised and Unsupervised Learning

Supervised

In supervised learning, each training example $\mathbf{x}$ is associated with a ground-truth label (or target) y.

Unsupervised

In unsupervised learning, no ground-truth labels are provided.

Learning and Inference (Supervised)

Learning (training)

Given training data $(\mathbf{X}, \mathbf{y})$, the learning algorithm produces a trained model, $f_{\hat{\boldsymbol{\theta}}}$:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \mathbf{X}, \mathbf{y})$$

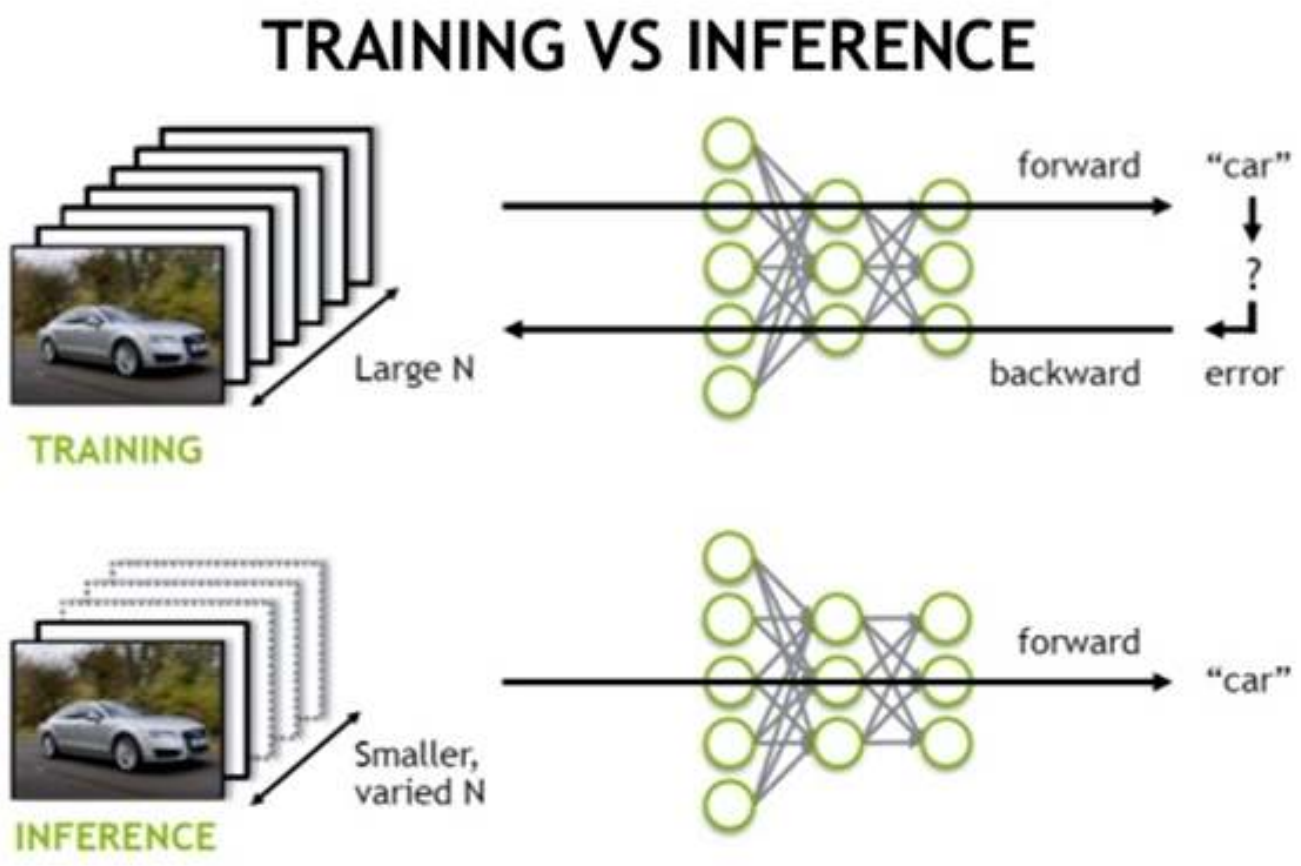
where $\mathcal{L}$ is a loss function measuring how well the model is doing.

Inference (prediction)

The trained model is applied to a (possibly new) input:

$$\hat{y} = f_{\hat{\boldsymbol{\theta}}}(\mathbf{x})$$

Learning and Inference (Supervised)



Learning and Inference (Unsupervised)

Learning (training)

Given training data $\mathbf{X}$, the learning algorithm produces a trained model, $f_{\hat{\boldsymbol{\theta}}}$:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \mathbf{X})$$

where $\mathcal{L}$ is a loss function measuring how well the model is doing.

Inference (application)

The trained model is applied to a (possibly new) input:

$$\hat{\mathbf{z}} = f_{\hat{\boldsymbol{\theta}}}(\mathbf{X})$$

Tasks

Supervised

- **classification**
- **regression**
- forecasting
- anomaly detection

Unsupervised

- clustering
- pattern mining
- **data generation**
- dimensionality reduction
- anomaly detection

Some tasks only make sense for certain kinds of data.

Classification

The aim of a classification is to predict a **categorical** variable for each instance in a dataset.

$$f(\mathbf{X}) = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)] = [\hat{y}_1, \ldots, \hat{y}_N] = \hat{\mathbf{y}}$$
$$\hat{\mathbf{y}} \in \{1, \ldots, C\}^N$$

where C is the number of classes. If $C = 2$ it is **binary** classification, otherwise it is **multiclass** classification.

Binary: if f predicts whether the patient is ill based on a test result, e.g., $f(\text{patient\_1}) = \text{ill}$
Multiclass: if f detects low/mid/high blood pressure: $f(\text{patient\_1}) = \text{high}$

Classification: patient outcome prediction

Based on patient attributes, predict the result of a clinical test

PatientID	Gender	Age	Zip code	Test
55998	M	19	15723	Negative
88557	F	35	15674	Positive
55868	F	35	15674	Positive
44551	M	45	15623	Negative
58524	M	45	15623	Negative
25584	F	61	15633	Negative
58744	F	61	15643	Positive
87524	M	19	15762	Positive
87384	M	19	15762	Negative
17583	F	19	15762	Positive

M: male; F: female

Input

- Tabular data

Output

- Binary Class (Positive/Negative)

Classification: patient outcome prediction

Based on patient attributes, predict the result of a clinical test

PatientID	Gender	Age	Zip code	Test
55998	M	19	15723	Negative
88557	F	35	15674	Positive
55868	F	35	15674	Positive
44551	M	45	15623	Negative
58524	M	45	15623	Negative
87744	F	61	15633	Negative
58744	F	61	15643	Positive
87524	M	19	15762	Positive
87384	M	19	15762	Negative
17583	F	19	15762	Positive

M: male; F: female

BINARY TABULAR CLASSIFICATION

Input

- Tabular data

Output

- Binary Class (Positive/Negative)

From probabilities to classes

Some classification models do not directly output a class label, but **class probabilities**. For a single instance:

$$f(\mathbf{x}) = \hat{\mathbf{p}}, \quad \hat{\mathbf{p}} \in [0,1]^C, \quad \sum_{c=1}^{C} \hat{p}_c = 1$$

Example: for a patient, the model outputs probabilities: $[\ \underbrace{0.1}_{\text{negative}}\ ,\ \underbrace{0.9}_{\text{positive}}\ ]$

The final predicted class is obtained by:

$$\hat{y} = \arg \max_{c \in \{1,\ldots,C\}} \hat{p}_c$$

Classification loss (cross-entropy)

For classification we want to train the model such that it makes the observed (ground-truth) labels as likely as possible under the model.

Given the true class $y \in \{1, \ldots, C\}$, the cross-entropy loss for one example is:

$$\ell(\hat{\mathbf{p}}, y) = -\log \hat{p}_y$$

Learning minimizes the average loss over the training set.

Classification loss (cross-entropy)

For example, if the ground truth class is $y = 1 = \text{positive}$, if the model predicts:

$$\hat{\mathbf{p}} = [0.1, 0.9]$$

Then the cross-entropy loss is:

$$\ell = -\log \hat{p}_y = -\log(0.9)$$

Regression

The aim of regression is to predict a **continuous** variable for each instance in a dataset.

$$f(\mathbf{X}) = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)] = \hat{\mathbf{y}}$$
$$\hat{\mathbf{y}} \in \mathbb{R}^N$$

Regression: patient age prediction

Based on patient attributes, predict their age

PatientID	Gender	Age	Zip code	Test
55998	M	19	15723	Negative
88557	F	35	15674	Positive
55868	F	35	15674	Positive
44551	M	45	15623	Negative
58524	M	45	15623	Negative
25584	F	61	15633	Negative
58744	F	61	15643	Positive
87524	M	19	15762	Positive
87384	M	19	15762	Negative
17583	F	19	15762	Positive

M: male; F: female

Input

- Tabular data

Output

- Age

Regression: patient age prediction

Based on patient attributes, predict their age

PatientID	Gender	Age	Zip code	Test
55998	M	19	15723	Negative
88557	F	35	15674	Positive
55868	F	35	15674	Positive
44551	M	45	15623	Negative
58524	M	45	15623	Negative
25584	F	61	15633	Negative
58744	F	61	15643	Positive
87524	M	19	15762	Positive
87384	M	19	15762	Negative
17583	F	19	15762	Positive

M: male; F: female

TABULAR REGRESSION

Input

- Tabular data

Output

- Age

Regression losses (MSE and MAE)

Given the true target $y \in \mathbb{R}$, and the prediction $\hat{y}$:

Mean Squared Error (MSE):

$$\ell_{\text{MSE}}(\hat{y}, y) = (\hat{y} - y)^2$$

Mean Absolute Error (MAE):

$$\ell_{\text{MAE}}(\hat{y}, y) = |\hat{y} - y|$$

Learning minimizes the average loss over the training set.

Regression losses (MSE and MAE)

Example: the true age of a patient is $y = 50$ years, the model predicts $\hat{y} = 45$ years.

Mean Squared Error (MSE):

$$\ell_{\text{MSE}}(\hat{y}, y) = (\hat{y} - y)^2 = (45 - 50)^2 = 25$$

Mean Absolute Error (MAE):

$$\ell_{\text{MAE}}(\hat{y}, y) = |\hat{y} - y| = |45 - 50| = 5$$

Learning minimizes the average loss over the training set.

Forecasting

Forecasting is a **special case of regression** that applies to sequences (e.g. time series), where the goal is to predict **future values** (**sequence-to-sequence**).

$$f([x_1, x_2, \ldots, x_T]) = [\hat{x}_{T+1}, \hat{x}_{T+2}, \ldots]$$

where T is the number of points.

We use T instead of M to denote time

Clustering

Clustering is an unsupervised learning task whose goal is to assign each instance to a cluster.

Given a dataset $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, the model produces:

$$f(\mathbf{X}) = \hat{\mathbf{z}}, \quad \hat{\mathbf{z}} \in \{1, \ldots, K\}^N$$

where K is the number of clusters and $\hat{z}_i$ is the cluster assignment of $\mathbf{x}_i$.

Generative Models

Generative models learn a model of the data and can generate new samples similar to the training data.

Given training data $\mathbf{X}$, learning produces a model $f_{\hat{\boldsymbol{\theta}}}$.

Sampling generates new data:

$$\tilde{\mathbf{x}} \sim f_{\hat{\boldsymbol{\theta}}}$$

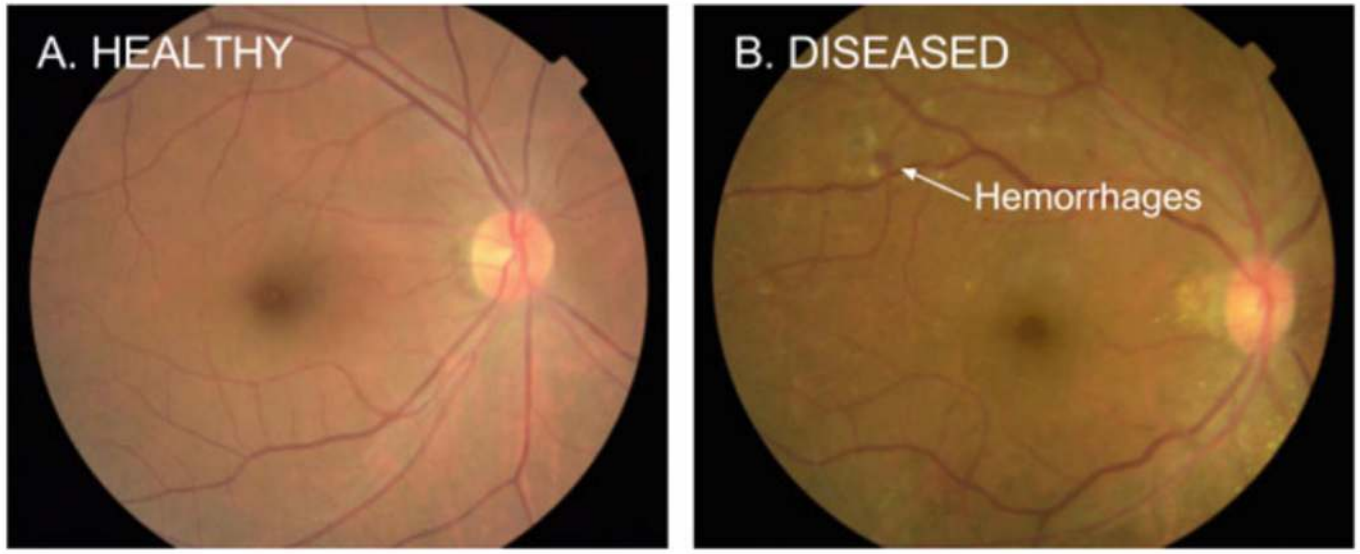
where $\tilde{\mathbf{x}}$ denotes a newly generated sample.

Quiz Time!

Categorize the following tasks

Task: Diabetic Retinopathy Detection

Predict health based on pictures of the retina.



Input

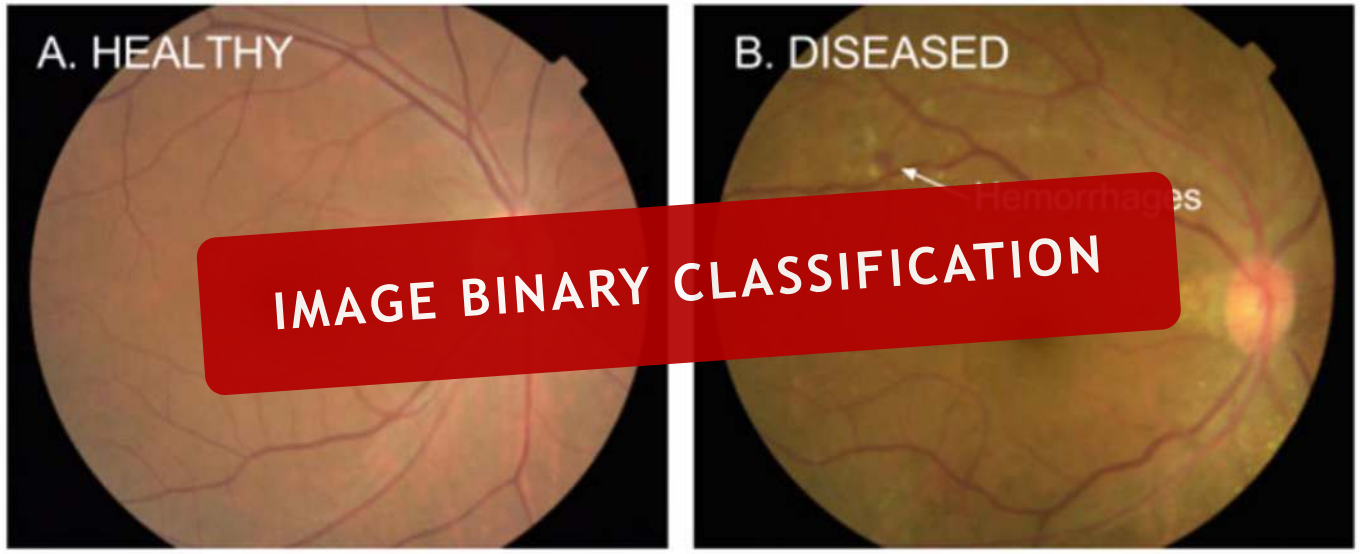
- Image data

Output

- Healthy/Diseased

Task: Diabetic Retinopathy Detection

Predict health based on pictures of the retina.



Input

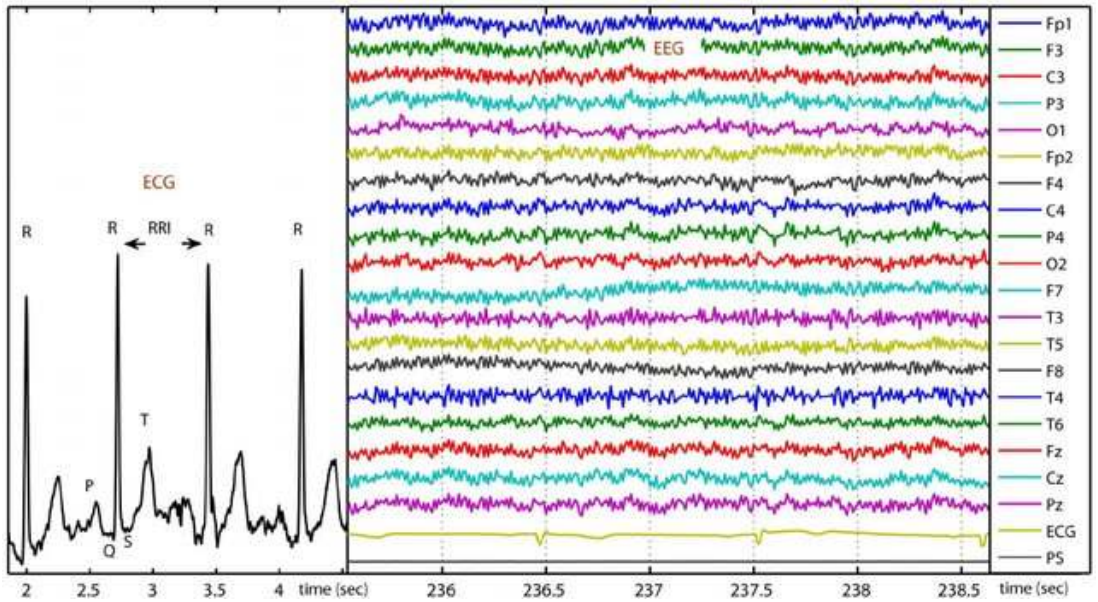
- Image data

Output

- Healthy/Diseased

Task: Early Detection of Heart Failure

Detect heart failure as early as possible.



Input

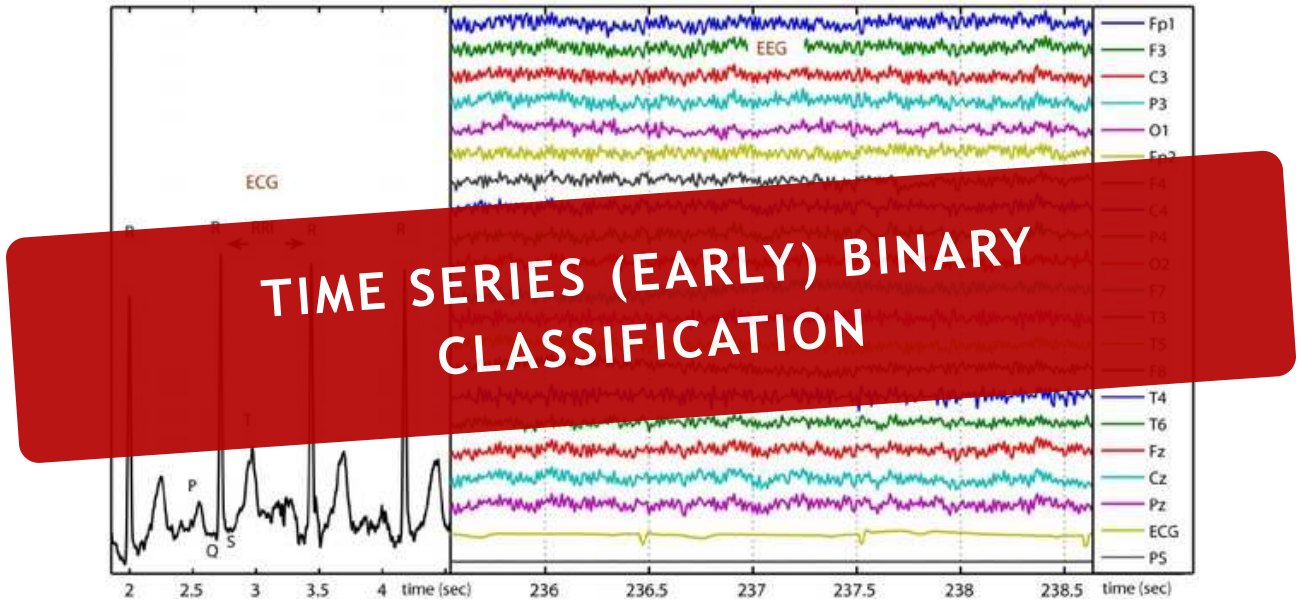
- ECG, (multivariate) time series

Output

- No-Failure/Failure

Task: Early Detection of Heart Failure

Detect heart failure as early as possible.



Input

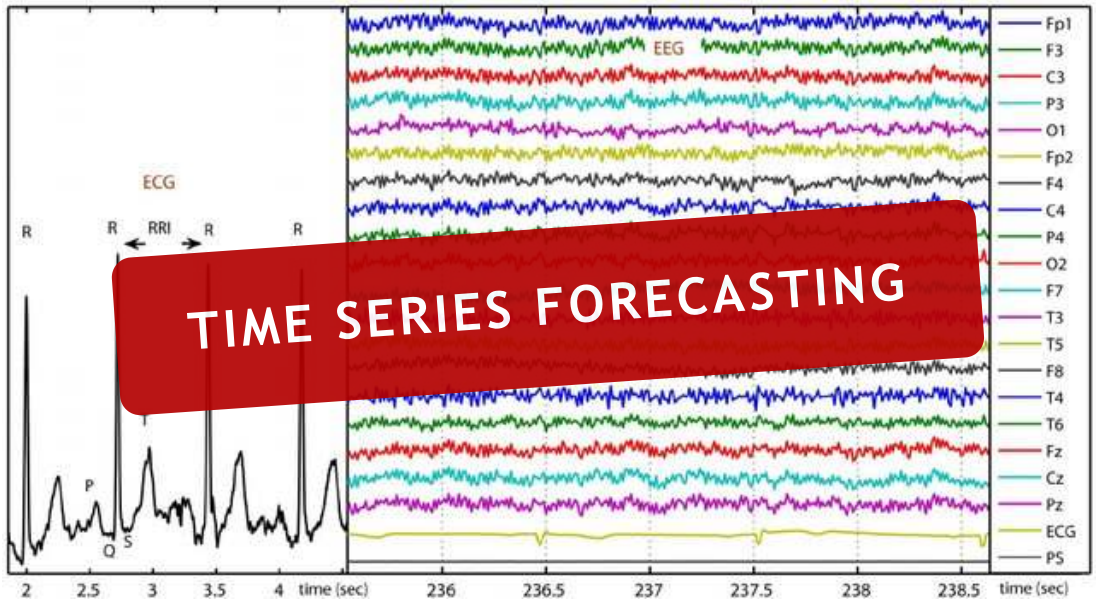
- ECG, (multivariate) time series

Output

- Binary Class (No-Failure/Failure)

Task: Early Detection of Heart Failure

Detect heart failure as early as possible.



Input

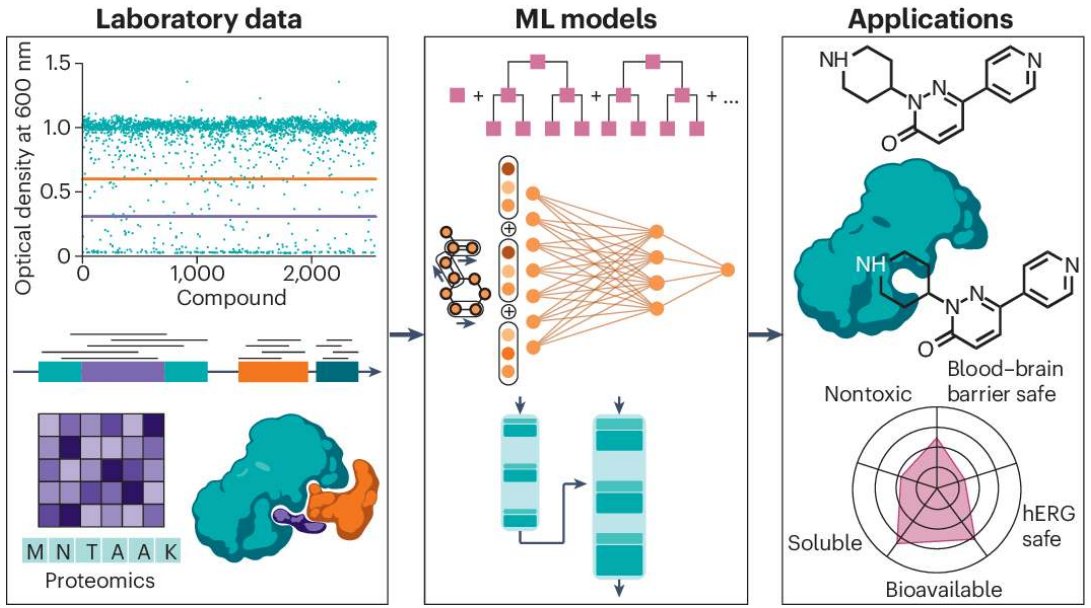
- ECG, (multivariate) time series

Output

- Future pulse behavior

Drug discovery

Drug discovery is not a single task but one very common task is Quantitative Structure–Activity Relationship (QSAR)



Input

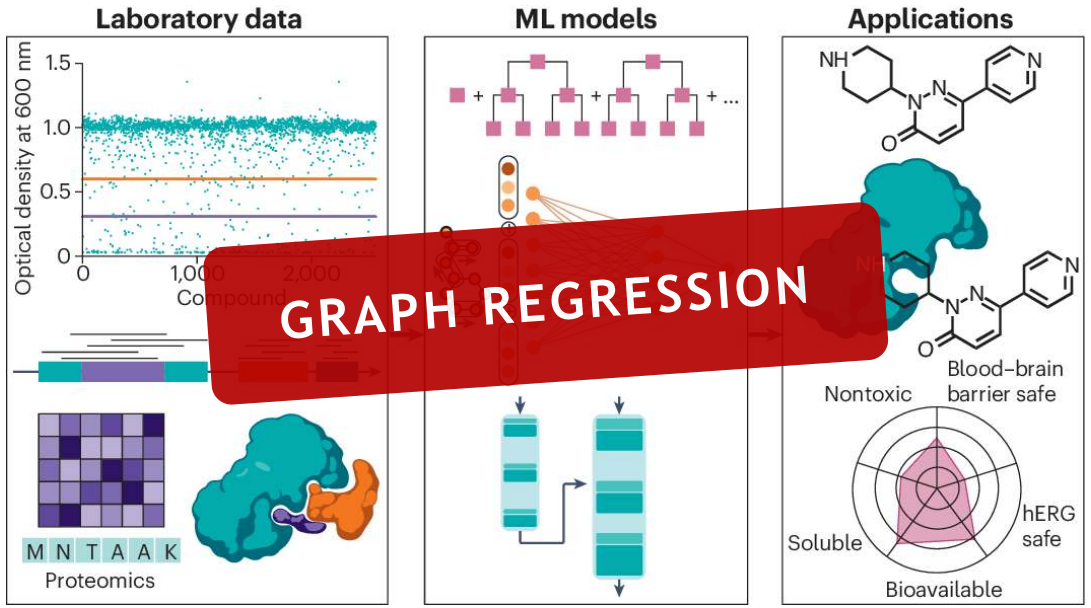
- Molecule structure (graph)

Output

- biological effect (continuous activity/affinity values)

Drug discovery

Drug discovery is not a single task but one very common task is Quantitative Structure–Activity Relationship (QSAR)



Input

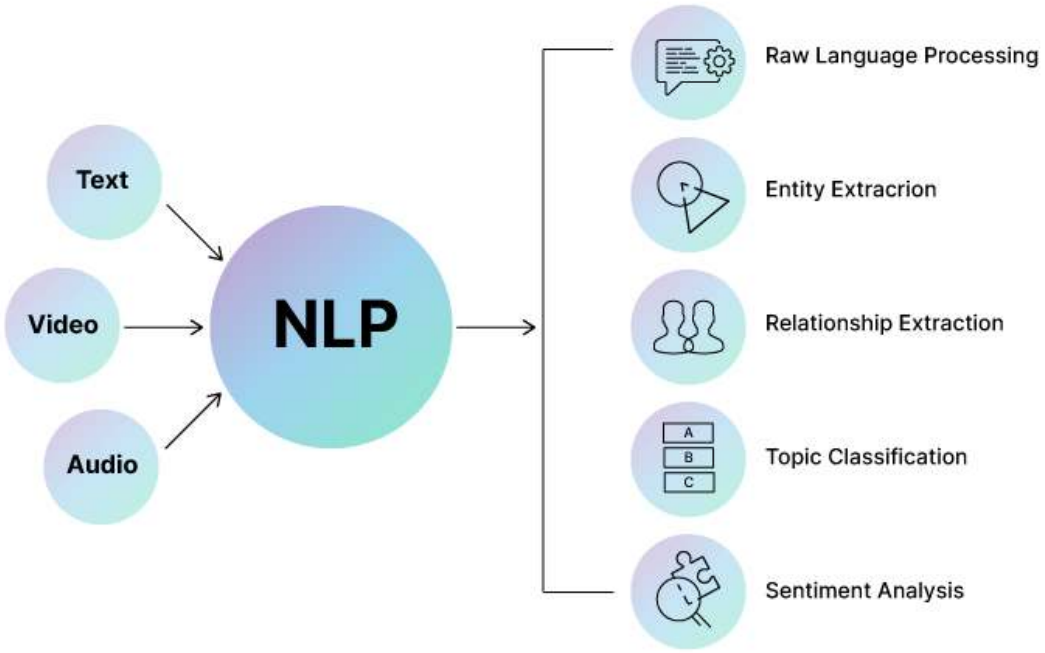
- Molecule structure (graph)

Output

- biological effect (continuous activity/affinity values)

Diagnosis to Prediction

Analyze natural language doctor prescription and predict diagnosis.



Input

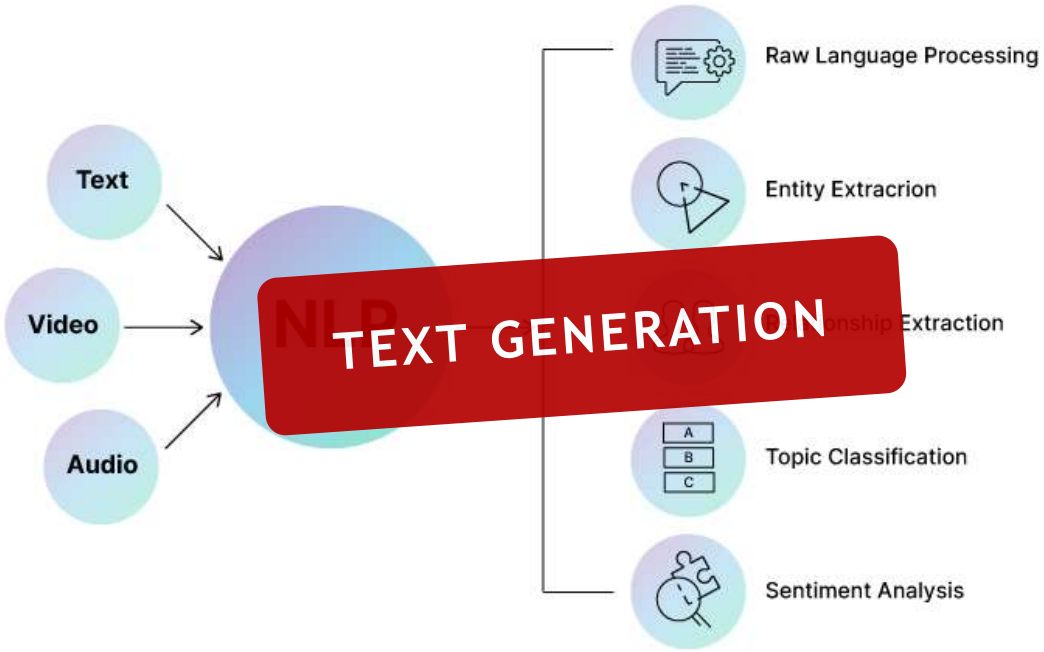
- Diagnosis (text)

Output

- Prescription (text)

Diagnosis to Prediction

Analyze natural language doctor prescription and predict diagnosis.



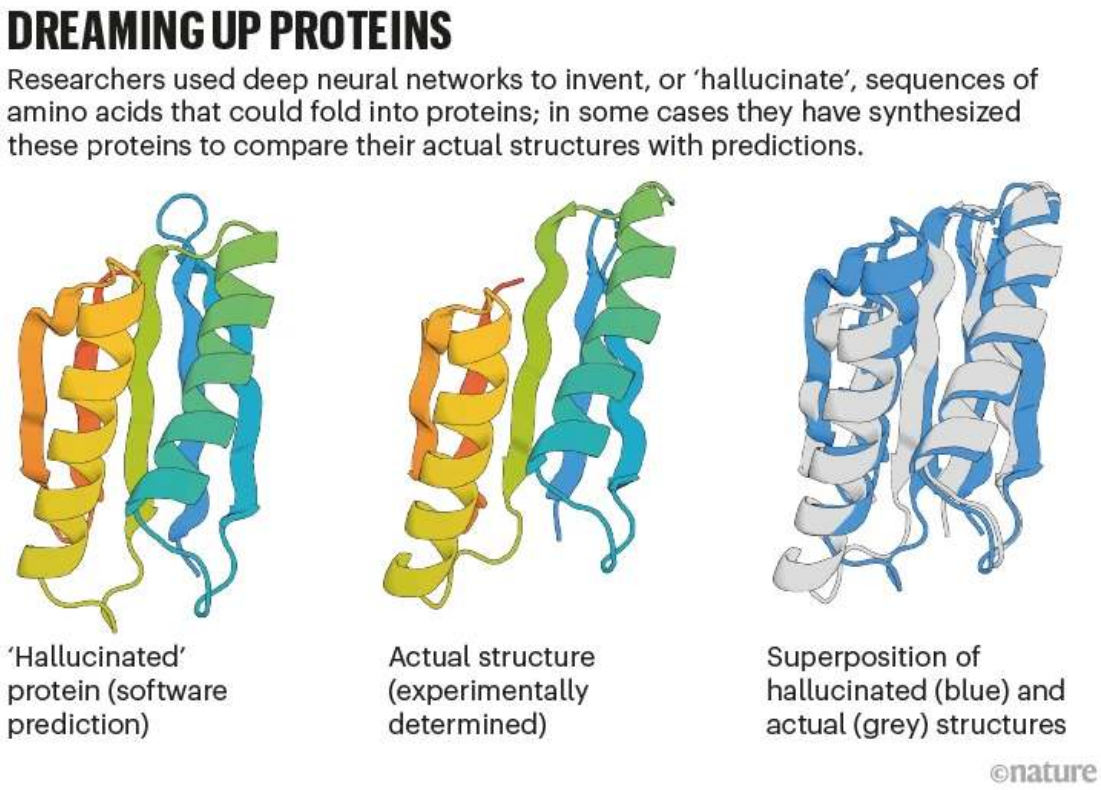
Input

- Diagnosis (text)

Output

- Prescription (text)

Task: Protein Folding



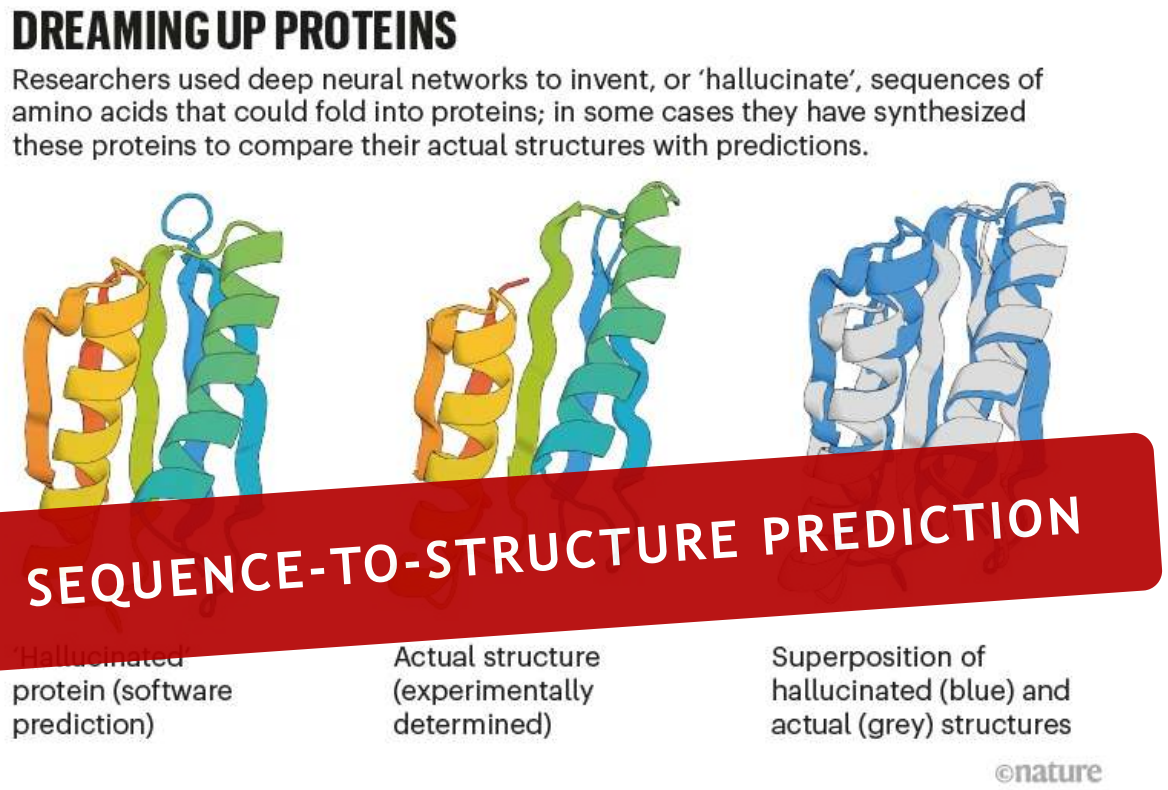
Input

- Amino acid sequence (optionally with evolutionary features, MSA, templates)

Output

- 3D structure of the protein (atom coordinates, distance maps, torsion angles, etc.)

Task: Protein Folding



Input

- Amino acid sequence (optionally with evolutionary features, MSA, templates)

Output

- 3D structure of the protein (atom coordinates, distance maps, torsion angles, etc.)

Evaluation

Choose the right metric for the task

Loss vs Metric

- A **loss function** is used during training to fit the model.
- An **evaluation metric** is used to measure performance after training.

They answer different questions:

- Loss: *“What does the model optimize?”*
- Metric: *“How good is the model?”*

They are often related, but not necessarily the same.

Metrics depend on the task

The choice of evaluation metric depends on:

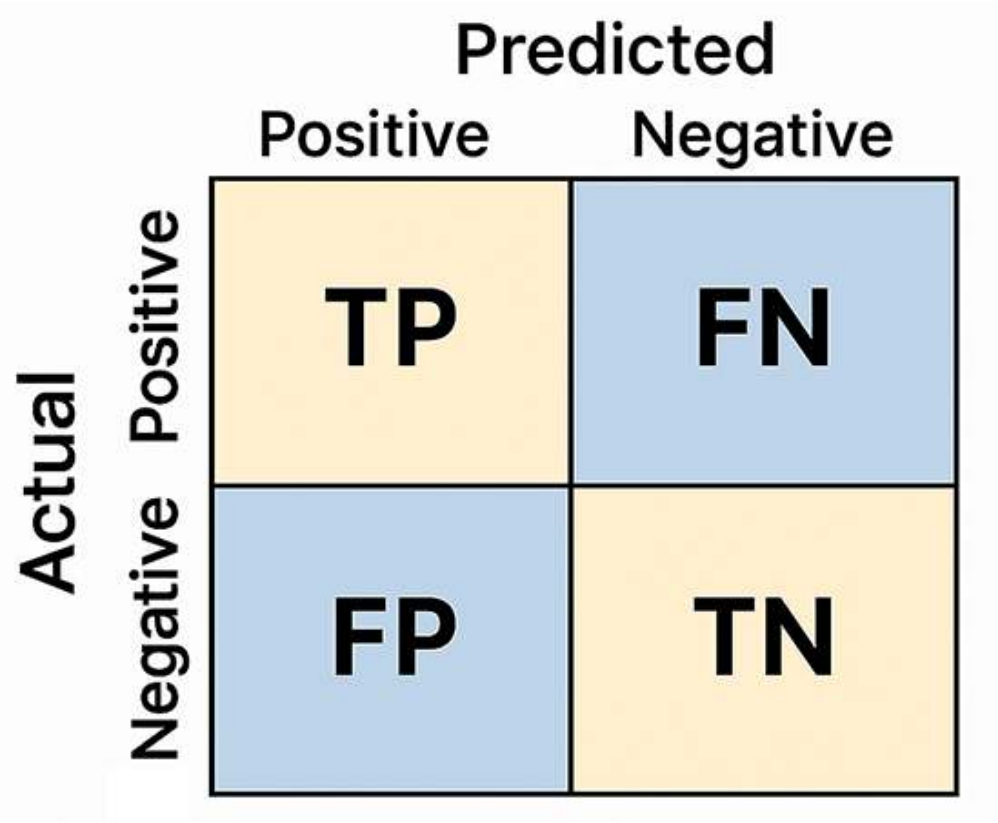
- the task (classification, regression, forecasting, ...)
- the data (e.g. class imbalance)
- the application (what kinds of errors matter more)

Confusion matrix (binary classification)

For binary classification:

- TP = true positives
- TN = true negatives
- FP = false positives
- FN = false negatives

Most classification metrics are defined in terms of these four numbers.



Classification metric: Accuracy

Accuracy measures the fraction of correct predictions:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

It treats all errors in the same way.

Imbalanced data

In many biological and medical problems, classes are **imbalanced** (e.g. 99% negative, 1% positive).

In this case:

- A model that always predicts “negative” can have very high **accuracy**
- But it is useless in practice

Therefore:

- Accuracy can be misleading
- Precision, recall, F1-score, or ROC-AUC are usually more informative

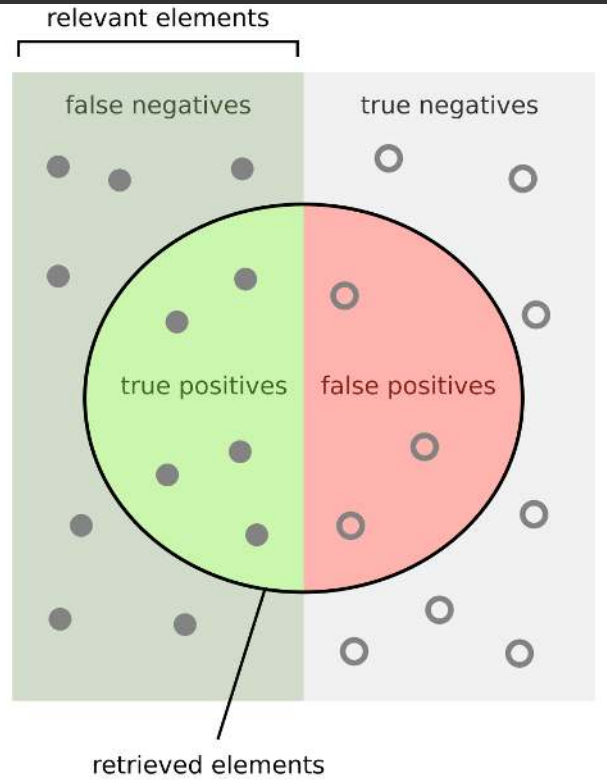
Classification metrics: Precision and Recall

Precision (how many predicted positives are correct with respect to all predicted positives):

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

Recall (how many true positives are found with respect to all true positives):

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



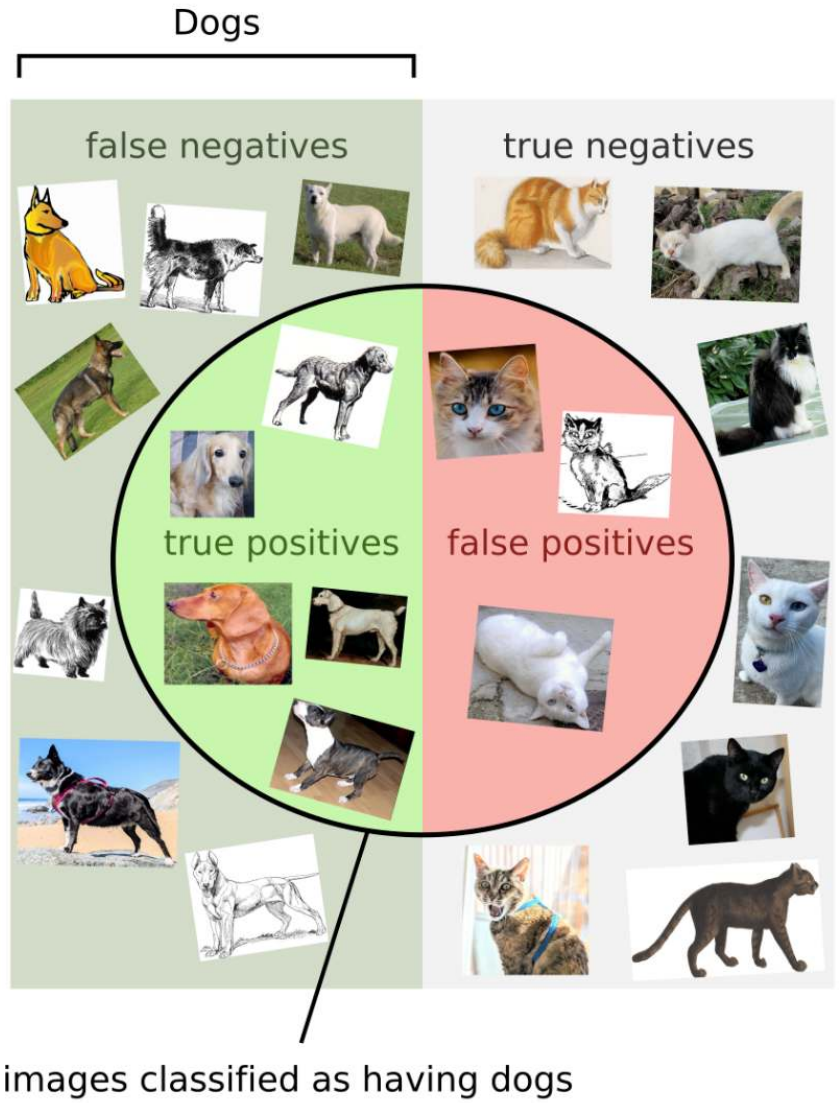
Classification metrics: Precision and Recall

For example, let's say you are building a classifier to **recognize dogs in an image:**

- The set of animals **classified** as dogs is the **retrieved** set ($TP + FP$)
- The set of all **true** dogs is the **relevant** set ($FN + TP$)

$$\text{Precision} = \frac{5\,TP}{5\,TP + 3\,FP}$$

$$\text{Recall} = \frac{5\,TP}{5\,TP + 7\,FN}$$



Classification metric: F1-score

The F1-score is the harmonic mean of precision and recall:

$$\mathrm{F1} = 2 \cdot \frac{\mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$$

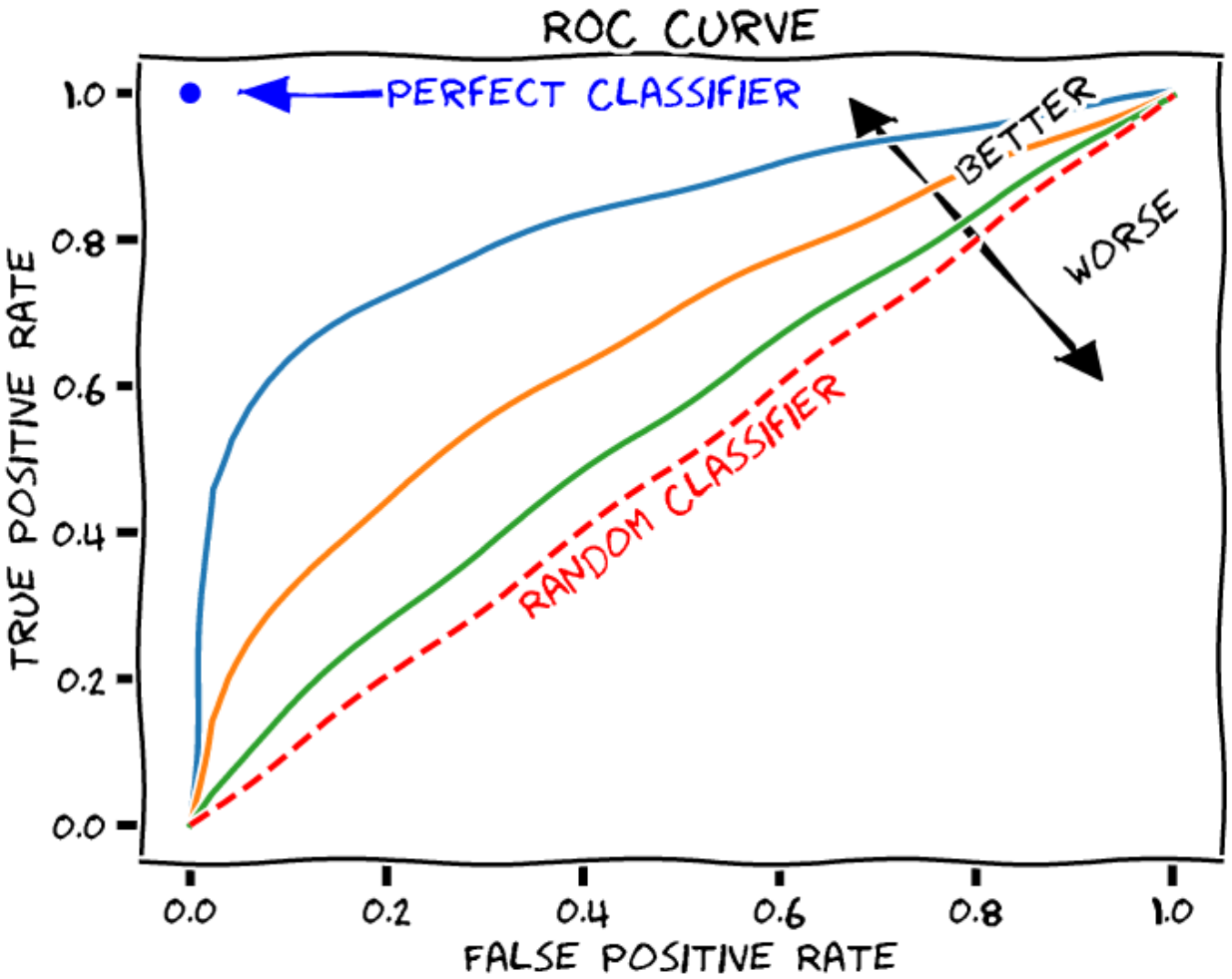
It balances precision and recall.

Classification metric: ROC-AUC

The ROC curve plots True Positive Rate (TPR) vs False Positive Rate (FPR)

$$\text{TPR} = \frac{TP}{TP + FN}, \quad \text{FPR} = \frac{FP}{FP + TN}$$

AUC is the area under this curve and measures ranking quality.



Example: ROC-AUC

Instance	y	$\hat{p}_1$ (prob positive class)	Predicted $\hat{p}_1 > 1$
A	1	0.90	0
B	0	0.80	0
C	1	0.60	0
D	0	0.40	0
E	0	0.20	0

Counts:
TP = 0, FP = 0, FN = 2, TN = 3

$$\text{TPR} = \frac{TP}{TP + FN} = \frac{0}{2}, \quad \text{FPR} = \frac{FP}{FP + TN} = \frac{0}{3}$$

ROC point: (0, 0)

Example: ROC-AUC

Instance	y	$\hat{p}_1$ (prob positive class)	Predicted $\hat{p}_1 \geq 0.9$
A	1	0.90	1
B	0	0.80	0
C	1	0.60	0
D	0	0.40	0
E	0	0.20	0

Counts:

TP = 1, FP = 0, FN = 1, TN = 3

$$\text{TPR} = \frac{TP}{TP + FN} = \frac{1}{2}, \quad \text{FPR} = \frac{FP}{FP + TN} = \frac{0}{3}$$

ROC point: (0, 0.5)

Example: ROC-AUC

Instance	y	$\hat{p}_1$ (prob positive class)	Predicted $\hat{p}_1 \geq 0.8$
A	1	0.90	1
B	0	0.80	1
C	1	0.60	0
D	0	0.40	0
E	0	0.20	0

Counts:

TP = 1, FP = 1, FN = 1, TN = 2

$$\text{TPR} = \frac{1}{2}, \quad \text{FPR} = \frac{1}{3}$$

ROC point: (0.33, 0.5)

Example: ROC-AUC

Instance	y	$\hat{p}_1$ (prob positive class)	Predicted $\hat{p}_1 \geq 0.6$
A	1	0.90	1
B	0	0.80	1
C	1	0.60	1
D	0	0.40	0
E	0	0.20	0

Counts:
TP = 2, FP = 1, FN = 0, TN = 2

$$\text{TPR} = \frac{2}{2} = 1, \quad \text{FPR} = \frac{1}{3}$$

ROC point: (0.33, 1)

Example: ROC-AUC

Instance	y	$\hat{p}_1$ (prob positive class)	Predicted $\hat{p}_1 \geq 0.4$
A	1	0.90	1
B	0	0.80	1
C	1	0.60	1
D	0	0.40	1
E	0	0.20	0

Counts:
TP = 2, FP = 2, FN = 0, TN = 1

$$\text{TPR} = 1, \quad \text{FPR} = \frac{2}{3}$$

ROC point: (0.66, 1)

Example: ROC-AUC

Instance	y	$\hat{p}_1$ (prob positive class)	Predicted $\hat{p}_1 \geq 0.2$
A	1	0.90	1
B	0	0.80	1
C	1	0.60	1
D	0	0.40	1
E	0	0.20	1

Counts:
TP = 2, FP = 3, FN = 0, TN = 0

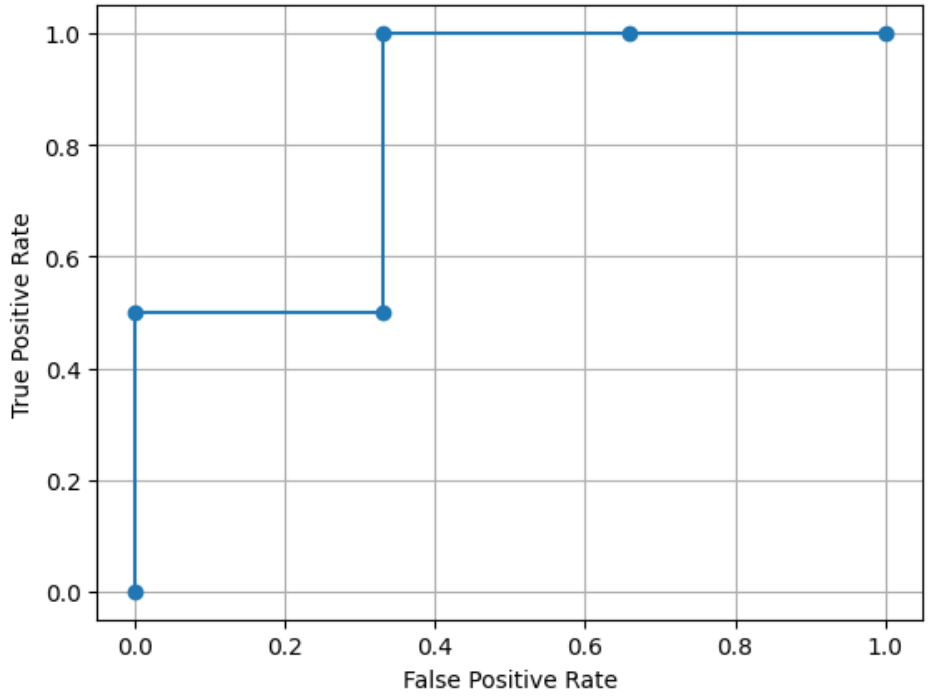
$$\text{TPR} = 1, \quad \text{FPR} = 1$$

ROC point: (1, 1)

Example: ROC-AUC

So the ROC is:

$$(0,0) \rightarrow (0,0.5) \rightarrow (0.33,0.5) \rightarrow (0.33,1) \rightarrow (0.66,1) \rightarrow (1,1)$$



Metrics for regression and forecasting

Let y_i be the true value and $\hat{y}_i$ the prediction.

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|$$

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

$$\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$$

No single best metric

There is no single “best” metric.

Always choose the metric that matches:

- the scientific question
- the cost of different types of errors
- the downstream use of the predictions

Why do we need evaluation?

The goal of a machine learning model is not to perform well on the training data, but to **generalize to unseen data**.

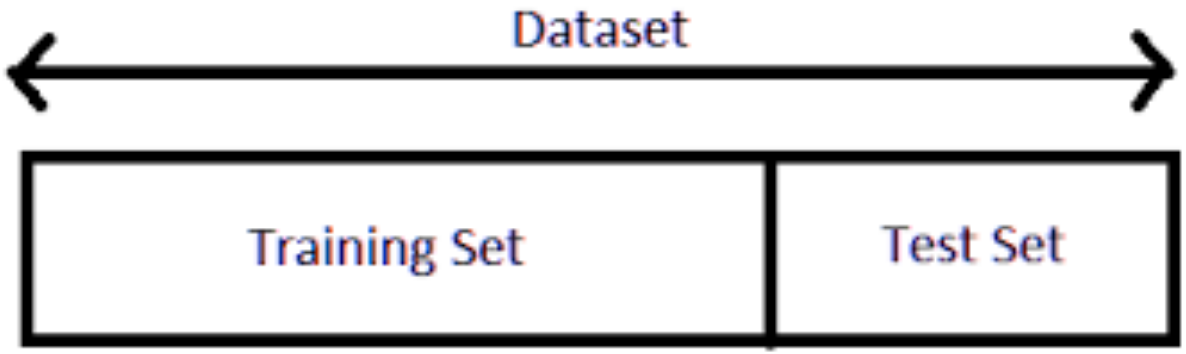
To estimate how well a model generalizes, we must evaluate it on data that was not used for learning.

Independently of the type of data, the general approach is the following:

Train and Test

The simplest evaluation setup is to split the data into:

- a **training set**, used to fit the model
- a **test set**, used to estimate generalization performance

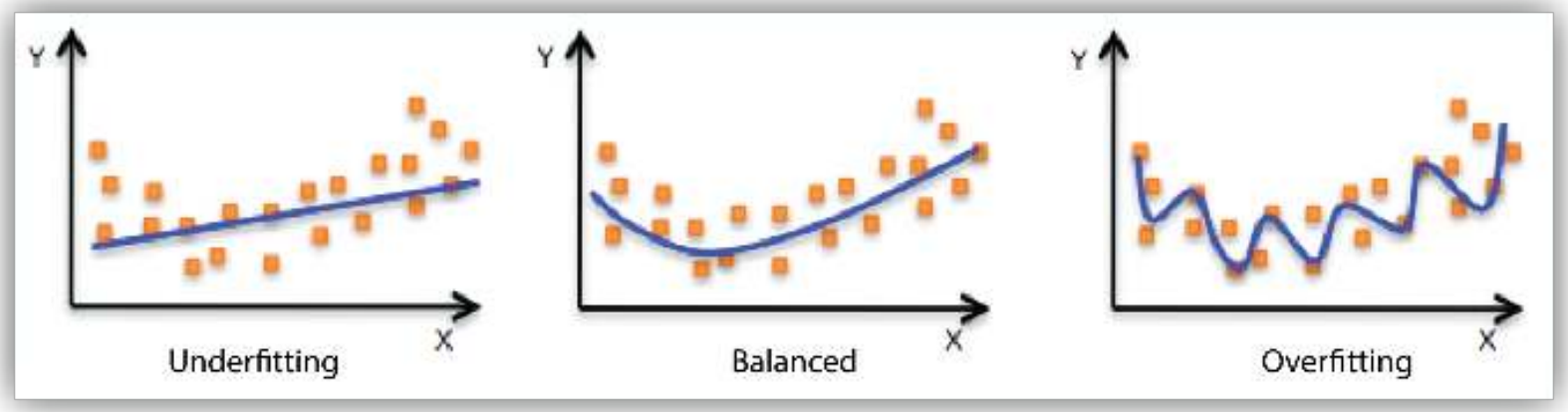


However, this is often not sufficient to build good models.

Typical problems

When training models, two main problems can occur:

- **underfitting**: the model is too simple
- **overfitting**: the model is too complex



Underfitting

A model is **underfitting** when it cannot capture the structure of the data.

Typical situation:

- high training error
- high test error

Underfitting usually means the model is too simple or not trained enough.

Overfitting

A model is **overfitting** when it fits the training data very well but performs poorly on new, unseen data.

Typical situation:

- very low training error
- much higher test error

Overfitting means the model has learned noise and dataset-specific details instead of general patterns.

Why do we need a validation set?

If we only use a training set and a test set, we are tempted to:

- try many models
- try many hyperparameters
- and keep the one that works best on the test set

This makes the test set no longer a fair estimate of generalization.

An important distinction!

- A parameter is a value learned from the data during training.
- A hyperparameter is a value set before training that controls how the model is trained or its structure.

Train, Validation, Test

Training set

Used to fit the model **parameters** (learned by the model)

Validation set

Used to select models and **hyperparameters** (selected by the user).

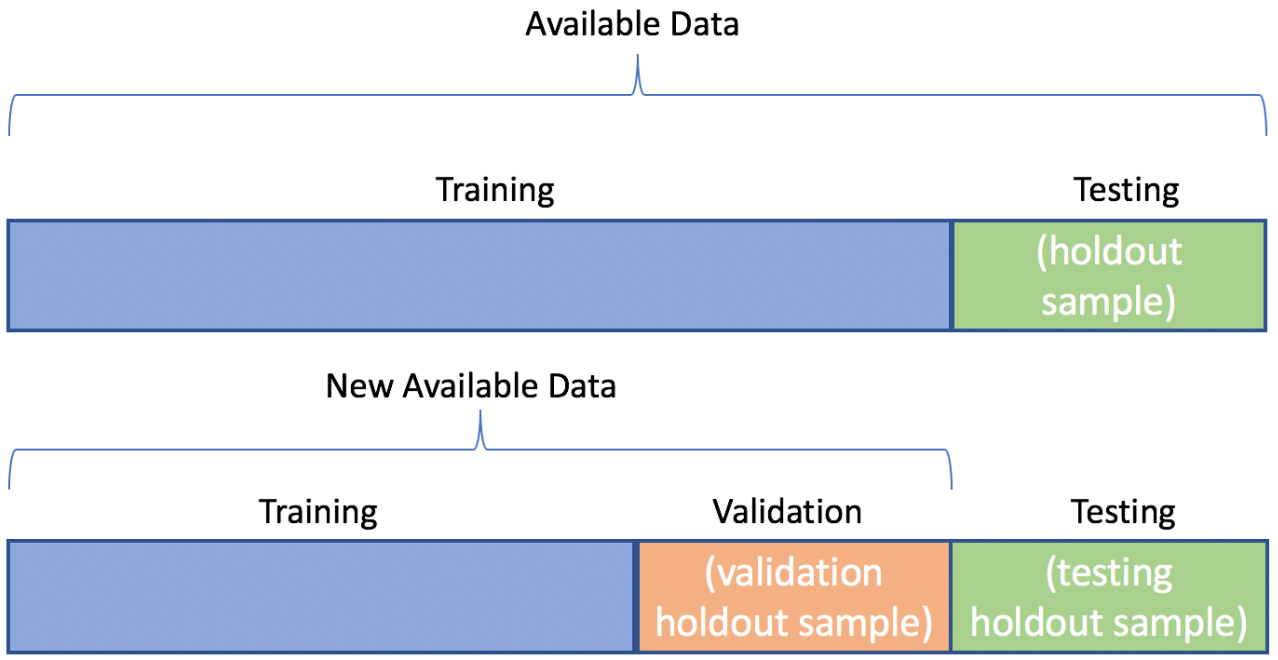
Test set

Used **only once** to estimate the final generalization performance.

Hold-out validation

The simplest evaluation strategy is to split data into train, validation, test sets

The model is trained on the training set, tuned on the validation set, and evaluated on the test set.



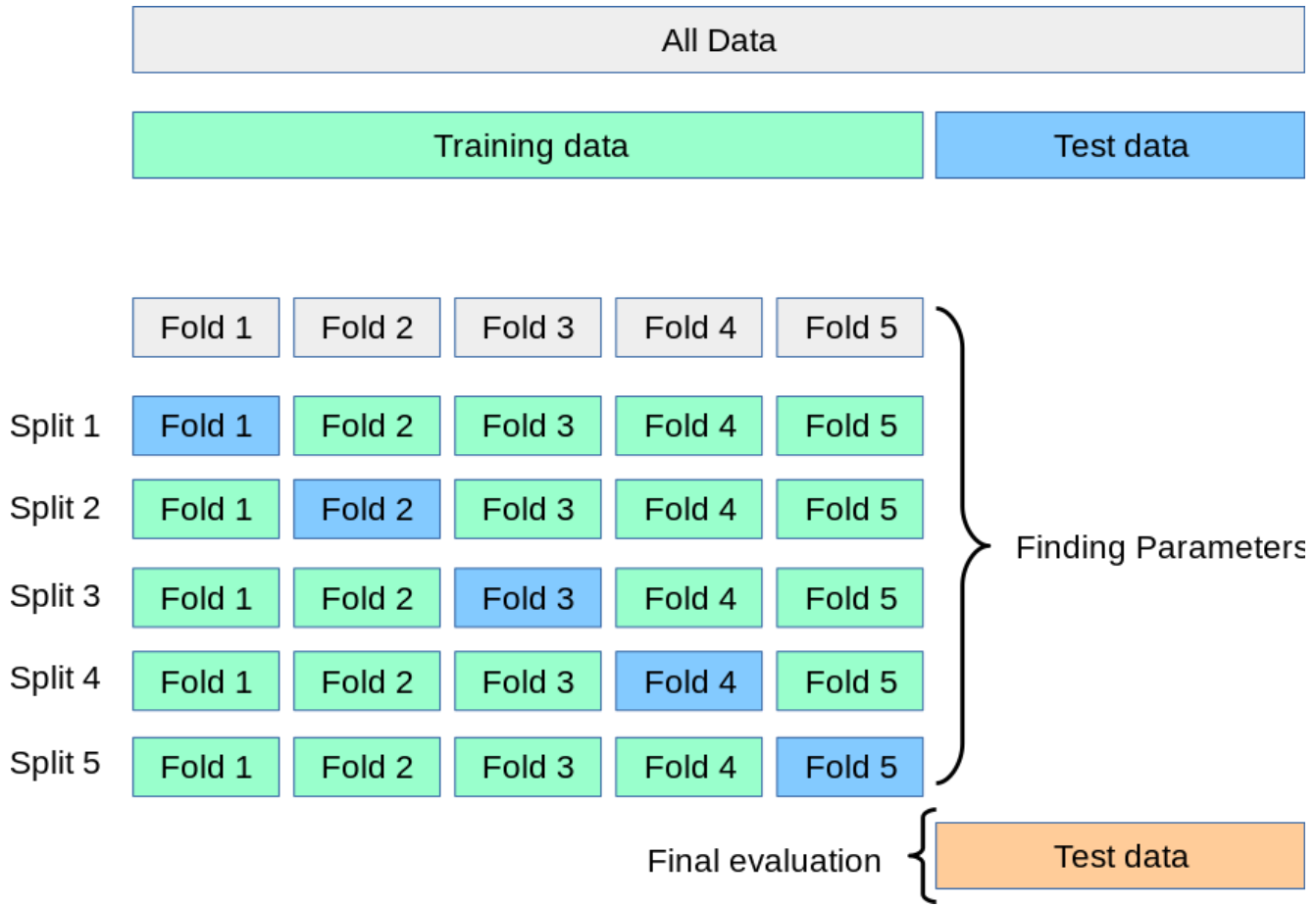
Cross-validation

In cross-validation, the training data is split into K folds.
For each fold:

- the model is trained on $K - 1$ folds
- evaluated on the remaining fold

The final validation score is the average over the K runs.

This gives a more stable estimate of performance than a single hold-out split.



How should the test set be used?

Cross-validation is used for:

- model selection
- hyperparameter tuning

The test set is:

- not used in training
- not used in cross-validation
- used only once, at the very end

Otherwise, the test performance is no longer an unbiased estimate of generalization.

Data leakage

Data leakage happens when information from outside the training set is used during training.

Examples:

- normalizing using the full dataset before splitting
- feature selection using all samples
- using test data to choose hyperparameters

Consequence:

The evaluation becomes overly optimistic and does not reflect real-world performance.