

Preparing for machine learning

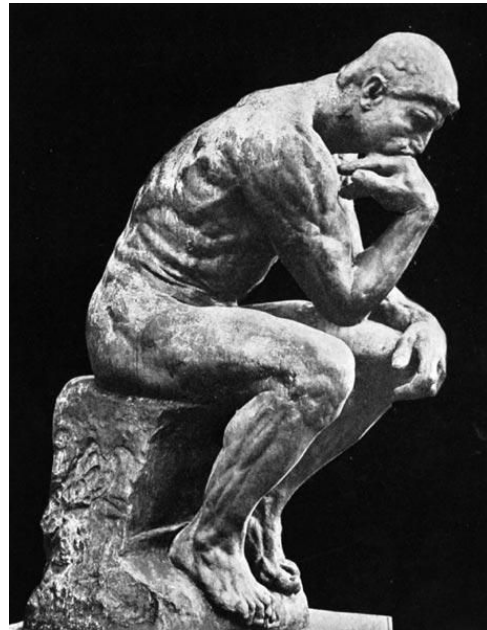
Machine learning

What does it mean “learning” for a machine?

What does it learn?

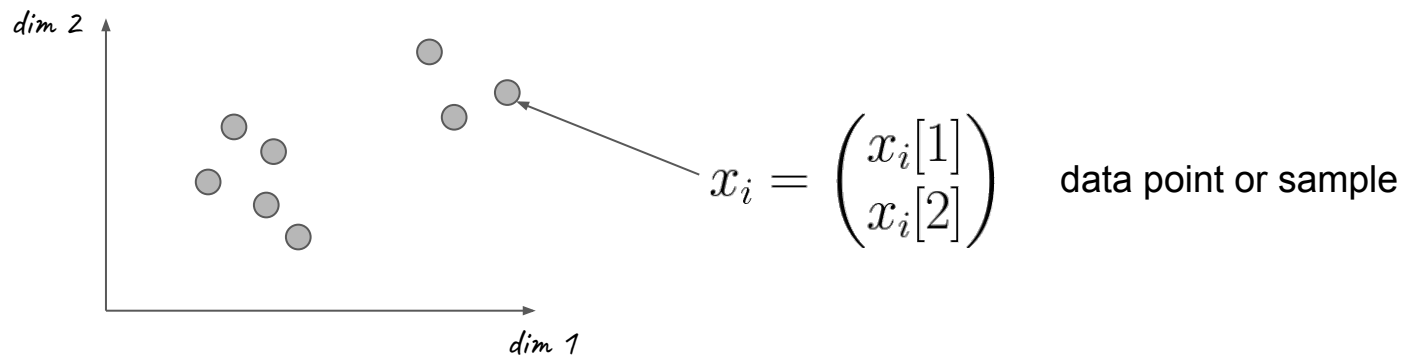
How to learn?

What do we really want to do?



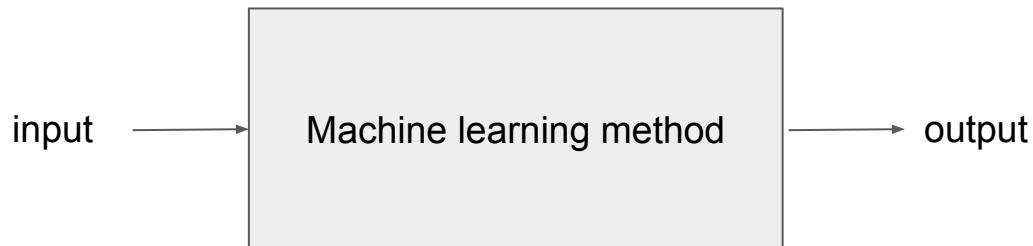
Learning from data

Dataset: a set of points in a space. Each point is a vector.



More generally, $x_i \in \mathbb{R}^N$ dataset: $\{x_i\}_{i \in [1, N]}$

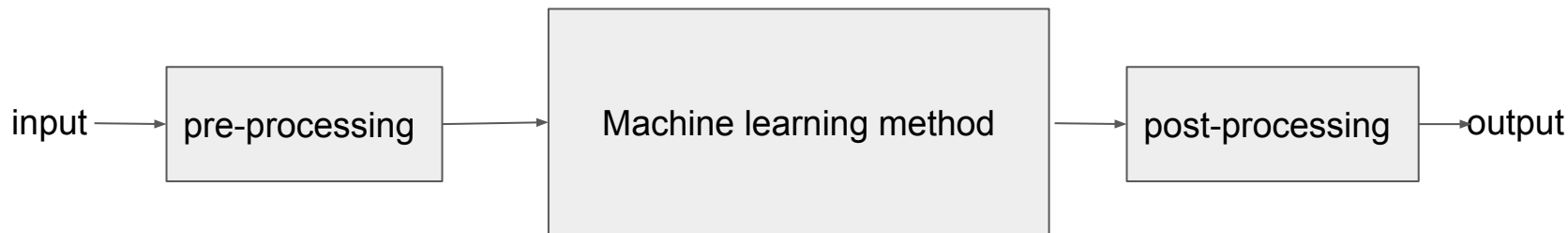
Learning structure



General setting

- input: numerical values,
- output: numerical values

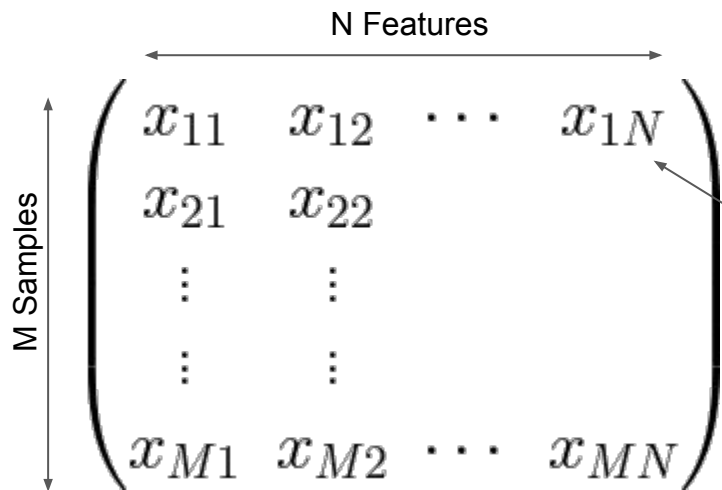
In reality, there is often a data processing “pipeline”:



Data format

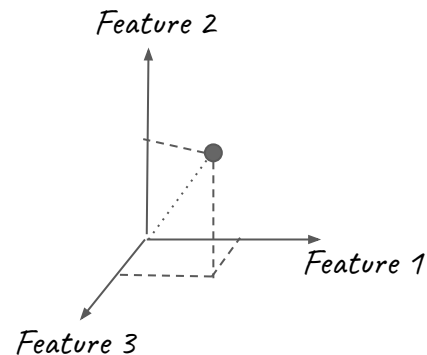
Dataset: samples (a lot of them) with features.

- each sample is a data point of the dataset
- each sample has N features
- a feature is a property of the data point, a dimension in the data space.



A diagram showing a matrix of data points. The matrix is enclosed in large parentheses and contains elements x_{11} , x_{12} , ..., x_{1N} in the first row, x_{21} , x_{22} in the second row, vertical ellipses in the third and fourth rows, and x_{M1} , x_{M2} , ..., x_{MN} in the last row. A horizontal double-headed arrow above the matrix is labeled "N Features". A vertical double-headed arrow to the left of the matrix is labeled "M Samples". An arrow points from the element x_{1N} in the matrix to a 3D coordinate system on the right.

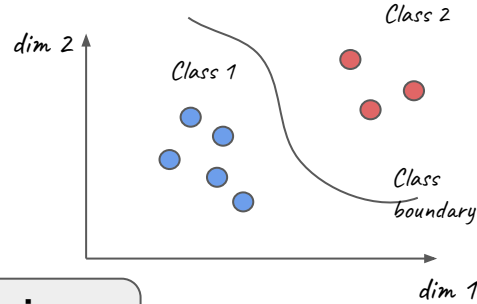
Value of feature N
for sample 1



We will work with array of numbers: matrices

Different learning tasks

- 2 main tasks



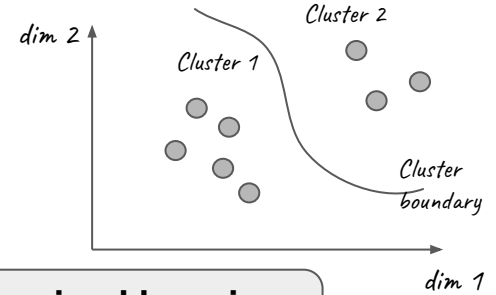
Supervised learning
learning with labels

Classification

associate each
sample with a
class

Regression

associate each
sample with a
number



Unsupervised learning
learning without labels

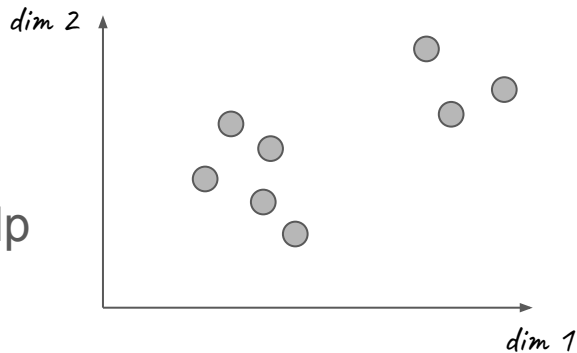
Clustering

Classification

associate each sample with a
cluster
(group similar samples together)

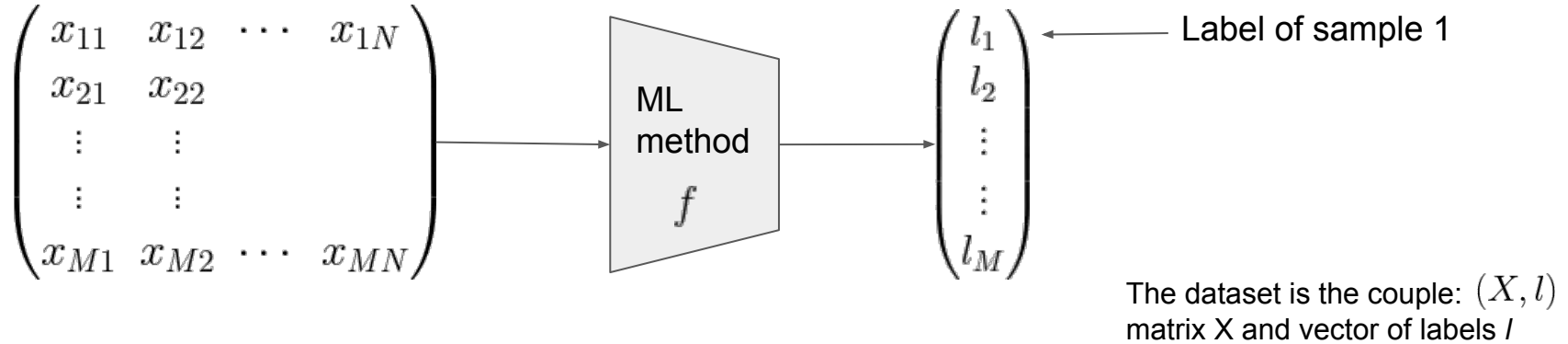
Other learning tasks

- **Semi-supervised learning:** learn with the help of a few labels,
- **self-supervised learning:** uses the data as “labels”. ex: learn to denoise an image, learn to guess a word in a sentence,
- **few-shot learning, zero-shot learning:** learn on a task and apply on a different task with different classes,
- **Reinforcement learning:** agents getting rewards for their actions.



Supervised learning

Learning with labels



Given the features, the method associates a label

$$l_1 = f(x_{11}, x_{12}, \dots, x_{1N})$$

The labels can be real values (regression) or discrete values (classification)

Training process

- We learn the mapping f from the features to the labels
- More precisely, f depends on parameters $\{w_i\}$

$$f(x_{11}, x_{12}, \dots, x_{1N}; w_1, w_2, \dots, w_k) = f_w(x_{11}, x_{12}, \dots, x_{1N})$$

The parameters or weights w are learnable:

the weights w will be tuned to minimize the prediction error

We have the prediction for all samples i :

$$\hat{l}_i = f_w(x_{i1}, x_{i2}, \dots, x_{iN})$$

We want the smallest error:

$$E_i = |l_i - \hat{l}_i|^2$$

The prediction is compared to the true value and the weights are updated to better match it.

The loss function

A mathematical function that measure the prediction error made by the system.

There are many different kinds of loss functions

This is one:
$$E = \sum_i |l_i - \hat{l}_i|^2$$

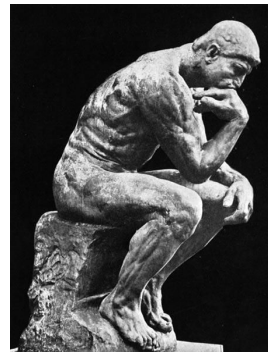
The loss function

A mathematical function that measure the prediction error made by the system.

There are many different kinds of loss functions

This is one: $E = \sum_i |l_i - \hat{l}_i|^2$

Learning is minimizing the loss function



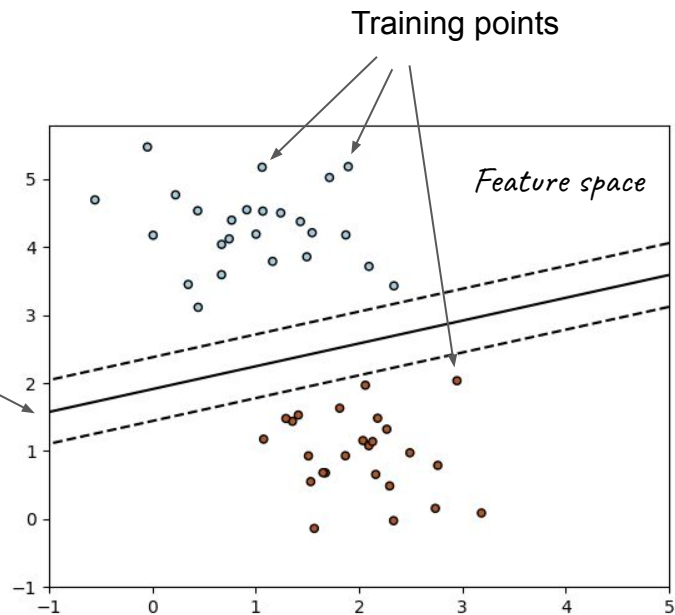
Separating classes - Generalization

Result of the learning:

Cutting the feature space into regions

Any point above this line belongs to class “blue”,
any point below belongs to class “red”

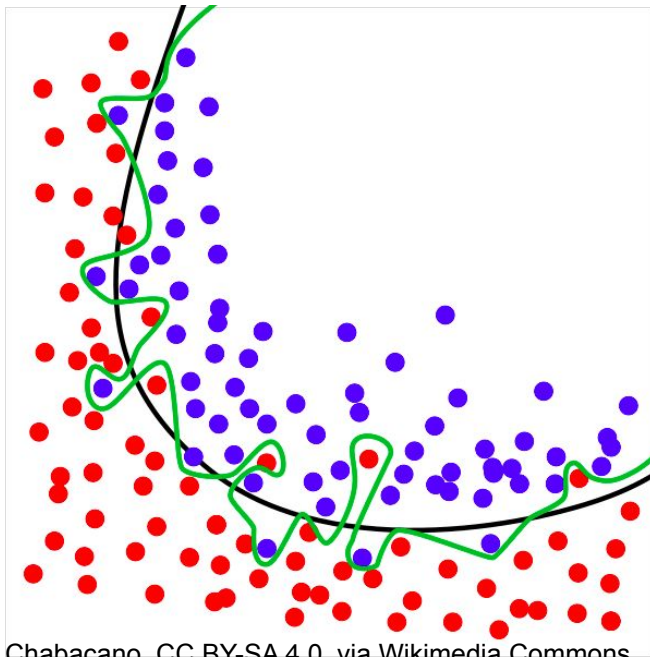
Generalization: ability to classify new, unseen, samples



From <https://scikit-learn.org/stable/modules/sgd.html>

Overfitting

Optimizing too much on the training points may be problematic

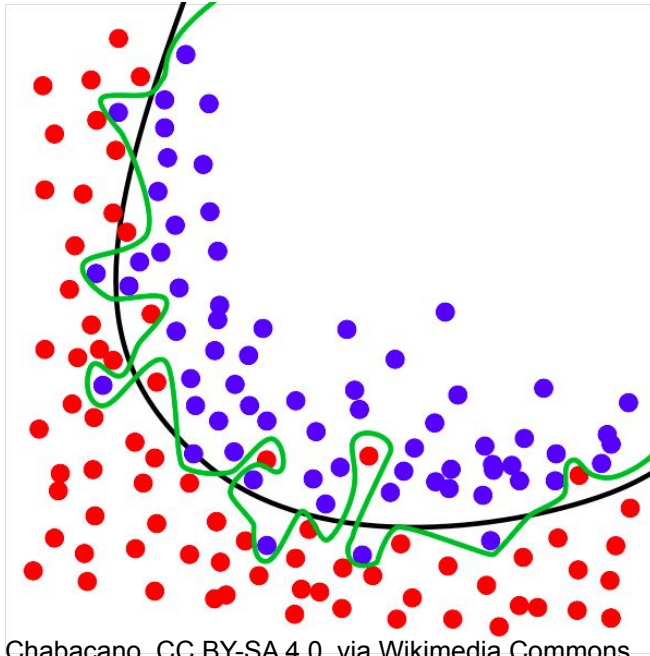


We may have

- noise in the data,
- some errors in the labelling

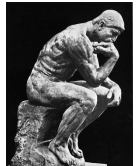
Overfitting

Optimizing too much on the training points may lead to overfitting



Overfitting: the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit to additional data or predict future observations reliably
(from Wikipedia, “overfitting”)

- Equivalent to learning by heart something
- It is unable to generalize well

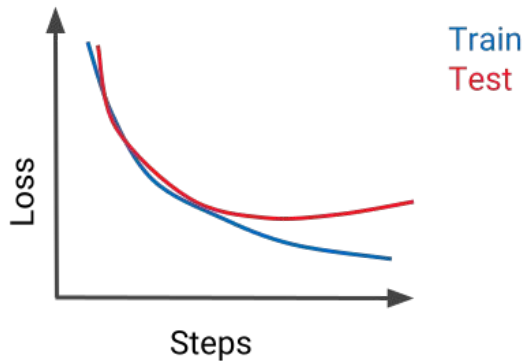


How to measure the ability to generalize?

How well does the trained model perform on unseen data?

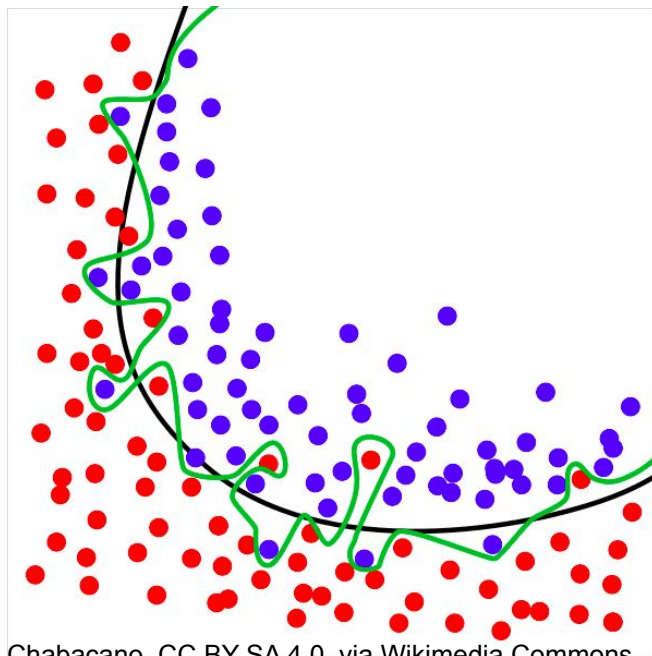
- The dataset is split into 2 parts: a train set and a test set, usually 80% / 20%.
- The machine learning model is trained only on the **train set**.
- The model is evaluated on the **test set** (unseen data during training).

The performance of 2 models can be compared **on the test set**.



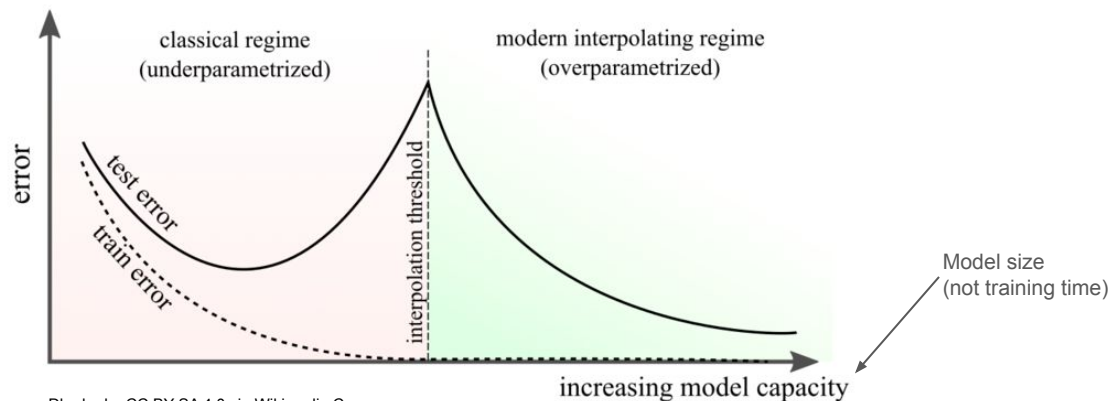
Interesting phenomenon: Double descent

Optimizing too much on the training points may lead to overfitting



Chabacano, CC BY-SA 4.0, via Wikimedia Commons

Large ML models can overfit but they don't.
The double descent phenomenon



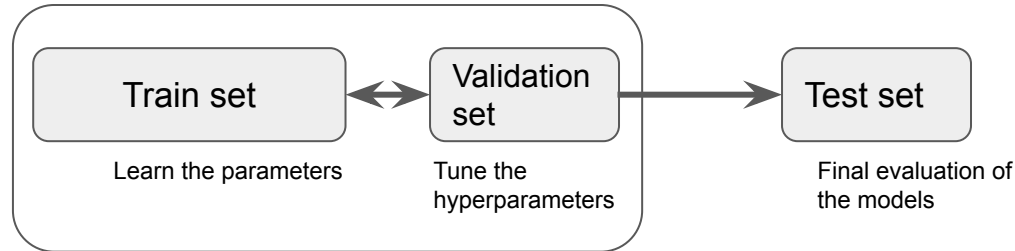
Dkarkada, CC BY-SA 4.0 via Wikimedia Commons

<https://mlu-explain.github.io/double-descent/>

Train/test/validation set

Better evaluation: introduce a validation set

- The ML model has some **hyperparameters** tuned by the experimenter.
- hyperparameters are tuned so that the score on the validation set is the highest -> there is a risk of overfitting to the test set.
- After all the experiments are performed: the model is evaluated on new samples, the test set.



But: the train set is reduced; problematic if we have a small dataset.

Hyperparameter: parameter tuned to control the model but not learned. Ex: number of neurons in a neural network.

Data leakage

Data or information used for training that is also present in the test set.

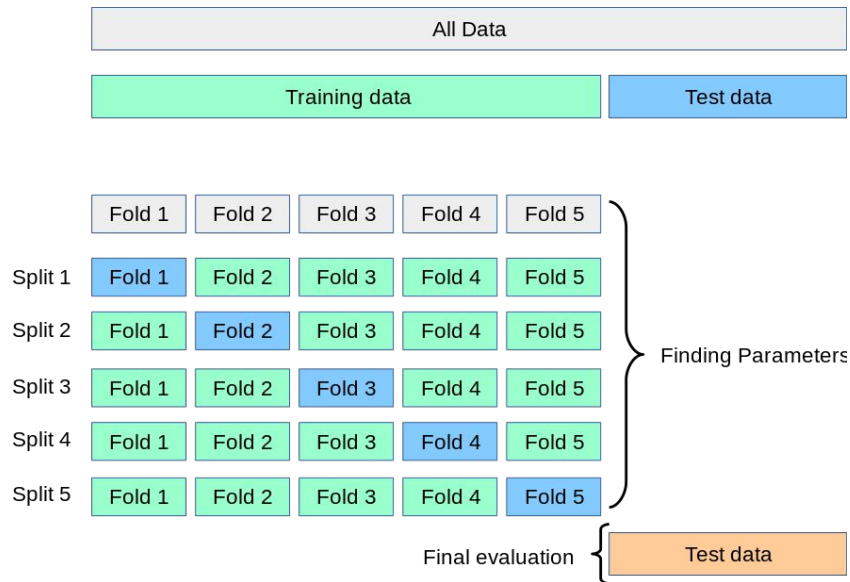
Leakage: the test set is not any more independent of the training

Example 1: You want to classify healthy/non-healthy patients from medical images. Each patient have several images. There is a leakage if for the same patient, some images are in the train set and some in the test set. The ML method could learn to detect the patient and not the state healthy/non-healthy.

Example 2: Some samples are duplicated and end up both in the train and test set.

Example 3: Selecting hyperparameters values that give the best results on the test set. We obtain the best model for this train and test set, but maybe not in general!

Cross validation



k-fold cross validation

- model trained using k-1 folds
- model tested on the remaining fold
- accuracy is averaged over splits

Good solution for small datasets.

- + we have a mean and variance on the accuracy

From

https://scikit-learn.org/stable/modules/cross_validation.html

Leave-one-out (LOO)

Extreme case of cross validation:

N samples in the dataset: train on N-1 samples and test on the remaining one.
Train N times and average the scores.

- almost no loss of data for training
- more computationally intensive than k-fold (training N times with N-1 samples)
- the variance of the averaged score can be high.

Unbalanced dataset

In a classification problem:

an unbalanced dataset have classes with different number of samples each.

Example: a dataset with 90% of samples with class 1 and 10% with class 2.

- Problem 1: when choosing randomly samples for the test set, there may not be any of class 2
- Problem 2: if the model label everything as class 1, there a 90% accuracy !

Solution for problem 1: stratified-k-fold, keep approximately the same proportion of each class in the train and test sets.

Solution for problem 2: we need a different score to evaluate the model (we will see that later on)

Learning demo

Classification with a neural network (Bonus as training neural networks is out of the scope of this course):

<https://playground.tensorflow.org/>