Data Science
Introduction to classification

Séverine Affeldt

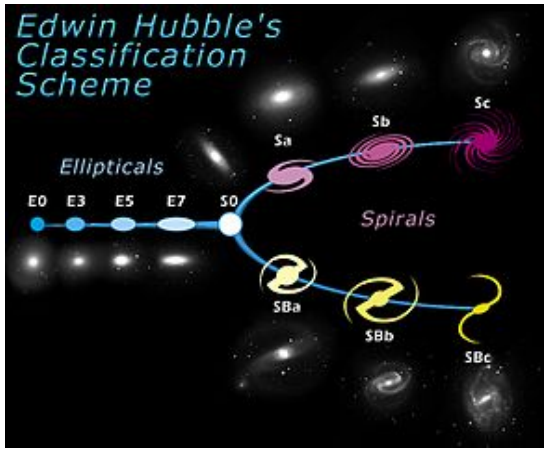
Université Paris Cité
Centre Borelli, UMR 9010, Equipe AI-DSCy
UFR Sciences Fondamentales et Biomédicales

2023 – 2024

Classification task *(general idea)*

Assigning objects to one of several predefined categories

Classifying galaxies upon their shapes



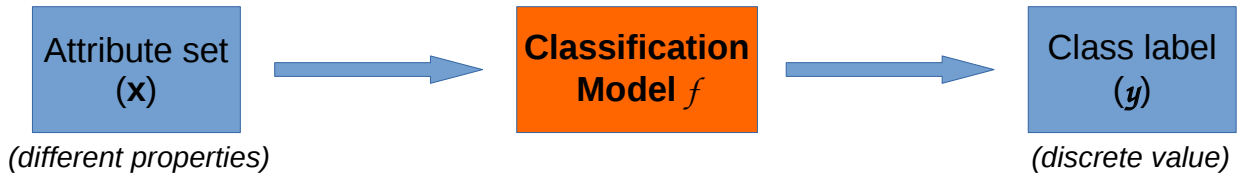
Elliptical galaxy

Spiral galaxy

Classification task *(formally)*

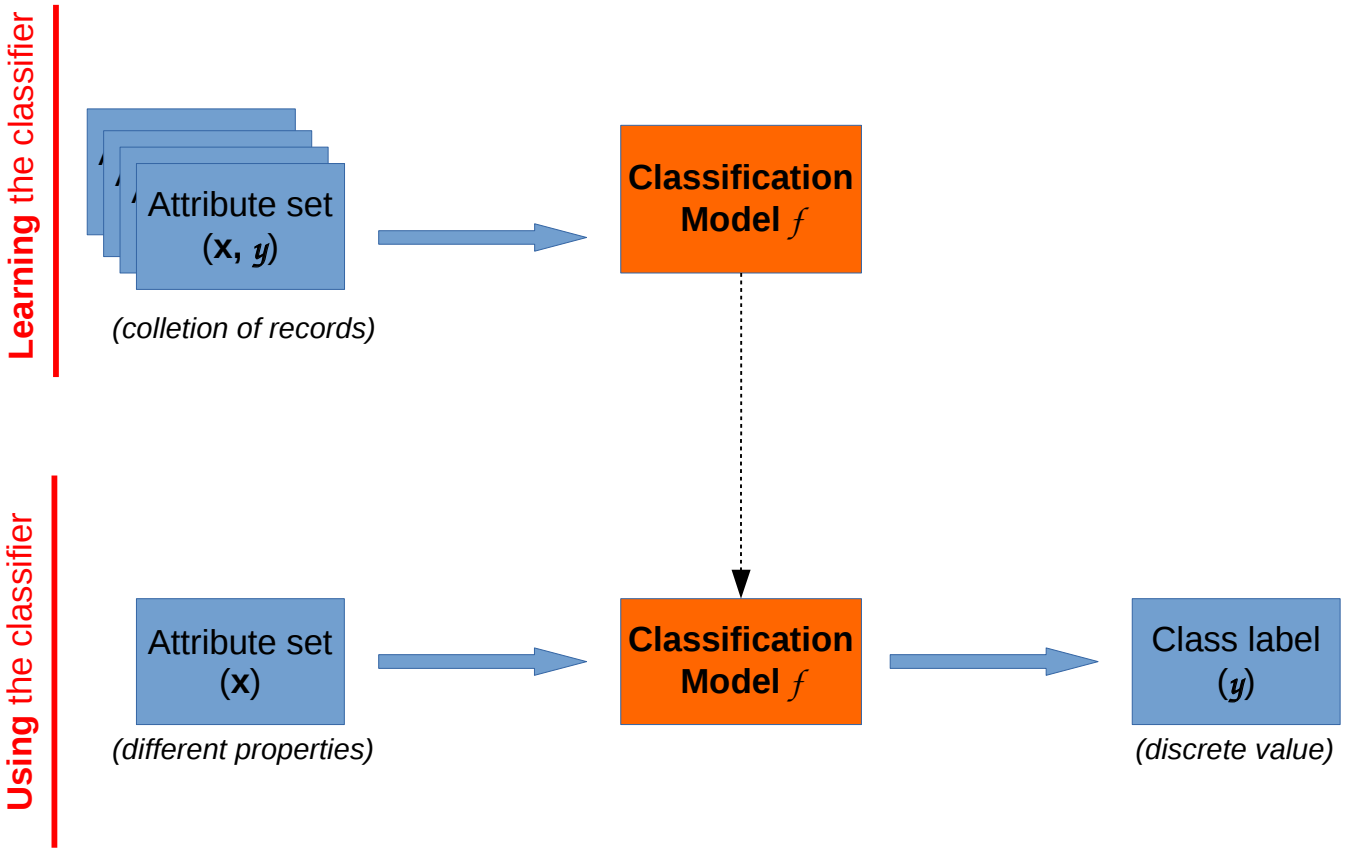
Learning a function f that maps an attribute set **x** to one of the predefined class labels y

Using the classifier



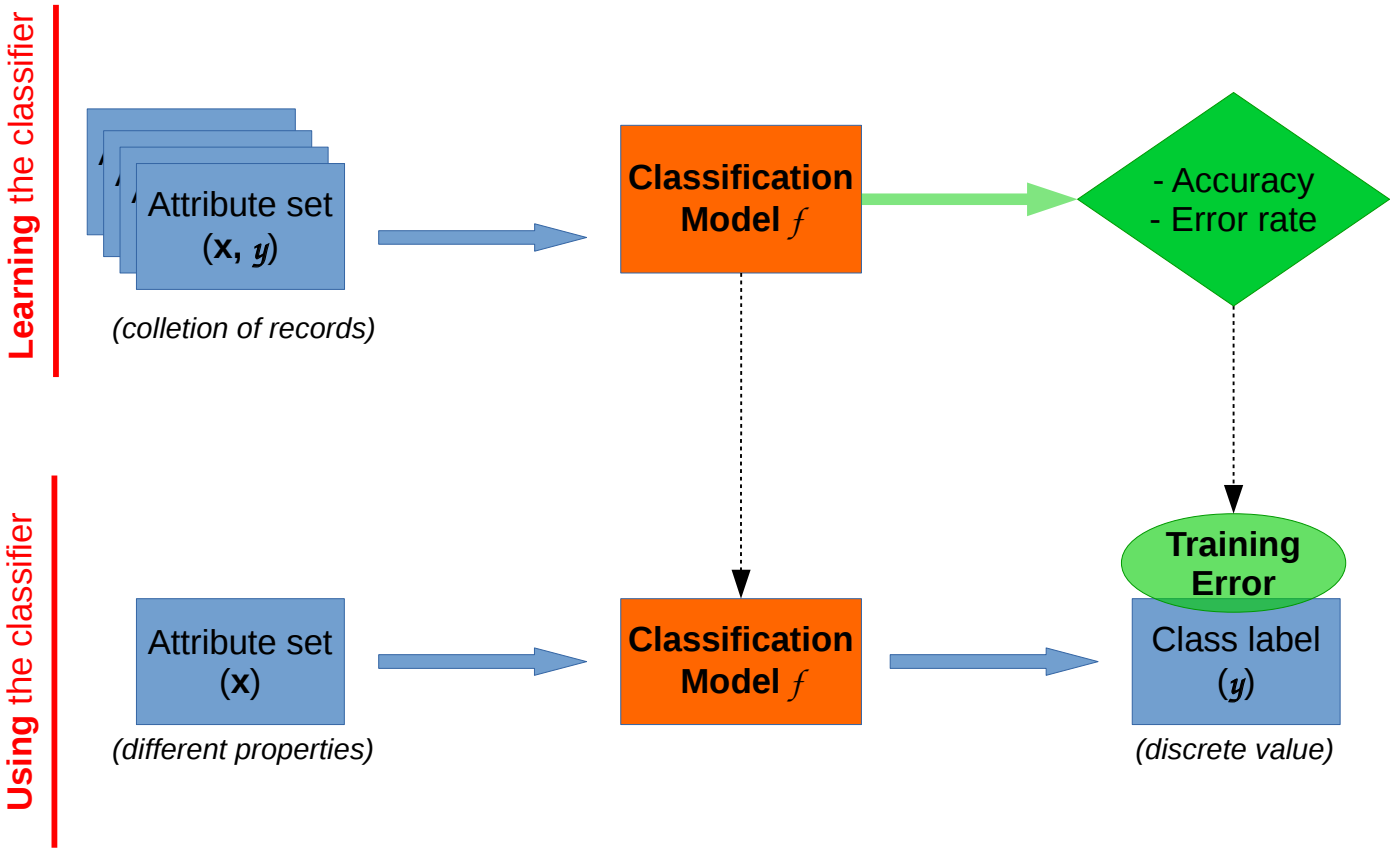
Classification task *(formally)*

Learning a function f that maps an attribute set **x** to one of the predefined class labels y



Classification task *(formally)*

Learning a function f that maps an attribute set **x** to one of the predefined class labels y



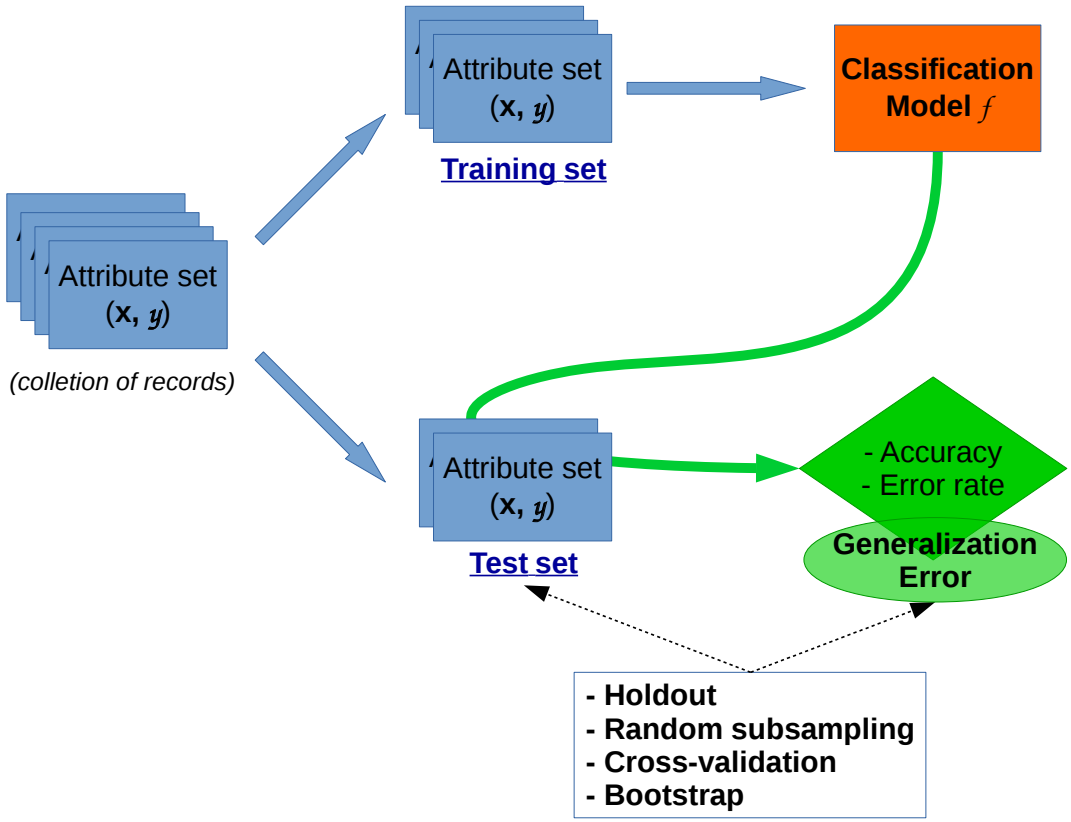
x	Properties							y
Name	Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hibernates	**Class Label**
human	warm-blooded	hair	yes	no	yes	yes	no	mammal
python	cold-blooded	scales	no	no	no	no	yes	reptile
salmon	cold-blooded	scales	no	yes	no	no	no	fish
whale	warm-blooded	hair	yes	yes	no	no	no	mammal
frog	cold-blooded	none	no	semi	yes	yes	yes	amphibian
komodo dragon	cold-blooded	scales	no	no	yes	yes	no	reptile
bat	warm-blooded	hair	yes	no	yes	yes	yes	mammal
pigeon	warm-blooded	feathers	no	no	yes	yes	no	bird
cat	warm-blooded	fur	yes	no	yes	yes	no	mammal
leopard shark	cold-blooded	scales	yes	yes	no	no	no	fish
turtle	cold-blooded	scales	no	semi	no	yes	no	reptile
penguin	warm-blooded	feathers	no	semi	no	yes	no	bird
porcupine	warm-blooded	quills	yes	no	no	yes	yes	mammal
eel	cold-blooded	scales	no	yes	no	no	no	fish
salamander	cold-blooded	none	no	semi	no	yes	yes	amphibian

A classification model is

- **descriptive**: summarizes data and distinguishes between objects of different classes
- **predictive**: predicts the class label of unknown records

x	Properties							y
Name	Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hibernates	**Class Label**
gila monster	cold-blooded	scales	no	no	no	yes	yes	?

$\Rightarrow$ Need a classification model with good **generalization capability**


Attribute set
(x, y)
Training set
Classification
Model f
Attribute set
(x, y)
(colletion of records)
Attribute set
(x, y)
Test set
- Accuracy
- Error rate
Generalization
Error
- Holdout
- Random subsampling
- Cross-validation
- Bootstrap

Confusion matrix

f_{ij}, number of records from class i predicted to be of class j

		Predicted class	
		Class = 1	Class = 0
Actual class	Class = 1	f_{11} (TP)	f_{10} (FN)
	Class = 0	f_{01} (FP)	f_{00} (TN)

Correct predictions: $(f_{11} + f_{00})$
Incorrect predictions: $(f_{01} + f_{10})$

Accuracy $= \frac{\textit{Number of correct predictions}}{\textit{Total number of predictions}} = \frac{f_{11}+f_{00}}{f_{11}+f_{00}+f_{01}+f_{10}}$

Error rate $= \frac{\textit{Number of wrong predictions}}{\textit{Total number of predictions}} = \frac{f_{01}+f_{10}}{f_{11}+f_{00}+f_{01}+f_{10}}$

Precision $\rightsquigarrow$ Prop. of correctly predicted '1' among all the predicted '1'

$\textit{Precision} = \frac{\textit{Number of true positive}}{\textit{true \& false positive}} = \frac{f_{11}}{f_{11}+f_{01}} = \frac{\text{TP}}{\text{TP}+\text{FP}}$

Recall $\rightsquigarrow$ Prop. of correctly predicted '1' among all the true '1'

$\textit{Recall} = \frac{\textit{Number of true positive}}{\textit{true positive \& false negative}} = \frac{f_{11}}{f_{11}+f_{10}} = \frac{\text{TP}}{\text{TP}+\text{FN}}$

F_1 score $\rightsquigarrow$ Harmonic mean of Precision and Recall

$F_1 \textit{ score} = \frac{2 \times \textit{Precision} \times \textit{Recall}}{\textit{Precision}+\textit{Recall}}$

Macro averaging

Macro averaging reduces the multiclass predictions to multiple sets of binary predictions (one vs. all others). In practice, the macro averaging approach calculates a metric (eg. precision, recall) for each of the binary case and then average the results together.

For k classes, the *macro average precision* is:

$$Pr_{macro} = \frac{Pr_1 + Pr_2 + ... + Pr_k}{k} = Pr_1 \frac{1}{k} + Pr_2 \frac{1}{k} + ... + Pr_k \frac{1}{k}$$

With macro averaging, all classes contributes equally to the metric. When the data contains imbalance classes, the *weighted macro average* should be preferred.

$$Pr_{weighted-macro} = Pr_1 \frac{\#Obs_1}{N} + Pr_2 \frac{\#Obs_2}{N} + ... + Pr_k \frac{\#Obs_k}{N}$$

Micro averaging

Micro averaging aggregates the results in one metric. For precision, this approaches computes all the true positive and false positive of the multiple sets of binary predictions (one vs. all others) and some them to compute a unique precision.

For k classes, the *macro average precision* is:

$$Pr_{micro} = \frac{TP_1 + TP_2 + ... + TP_k}{(TP_1 + TP_2 + ... + TP_k) + (FP_1 + FP_2 + ... + FP_k)}$$

With micro averaging, each observation gets equal weight. This gives the classes with the most observations more power.

Evalutation principle

Finding the model of the **right complexity**
that is **not susceptible to overfitting**
based on the estimated **generalization error**.

Generalization error

An estimation of the generalization error is obtained from a **test set**.

Possible methods of evaluation

1. Holdout
2. Random sampling
2. Cross-validation
2. Bootstrap

Holdout steps

1. Partition the samples in two **disjoint** sets, the **training** and the **test** sets (e.g. $50\%-50\%, 60\%-40\%, 80\%-20\%...$)
2. Estimate the **accuracy** of your classifier from the **test set**

Holdout method limitations

- **Reduction of the training set** because of the train/test partition
  $\Rightarrow$ Your model might not be as good as with all the data!
- Your model might **depend on the partition** composition
  $\rightarrow$ A small training set $\Rightarrow$ a large variance of the model
  $\rightarrow$ A large training set $\Rightarrow$ a less reliable accuracy from the test set
- The training and test sets are **not independent**
  $\Rightarrow$ An under-represented class in the training set is over-represented in the test set !

Random Subsampling steps

1. Repat **k Holdout** evaluations
2. Estimate the **accuracy** of your classifier by taking the average over the **k Holdout**

Random Subsampling method limitations

- **Reduction of the training set** because of the train/test partition
  $\Rightarrow$ Your model might not be as good as with all the data!
- Your model might **depend on the partition** composition
  $\rightarrow$ A small training set $\Rightarrow$ a large variance of the model
  $\rightarrow$ A large training set $\Rightarrow$ a less reliable accuracy from the test set
- The training and test sets are **not independent**
  $\Rightarrow$ An under-represented class in the training set is over-represented in the test set !
- No control over the number of times a sample is used
  $\Rightarrow$ Some samples might be used for training more often !

2-fold Cross-Validation steps

1. Partition the data into **two equal-sized subsets**, the training and the test sets
2. Estimate the **accuracy** of your classifier from the **test set**
3. **Swap** the roles of the training and test sets
4. Estimate the **accuracy** of your classifier from the **test set**
5. Sum the errors from the 1 & 2

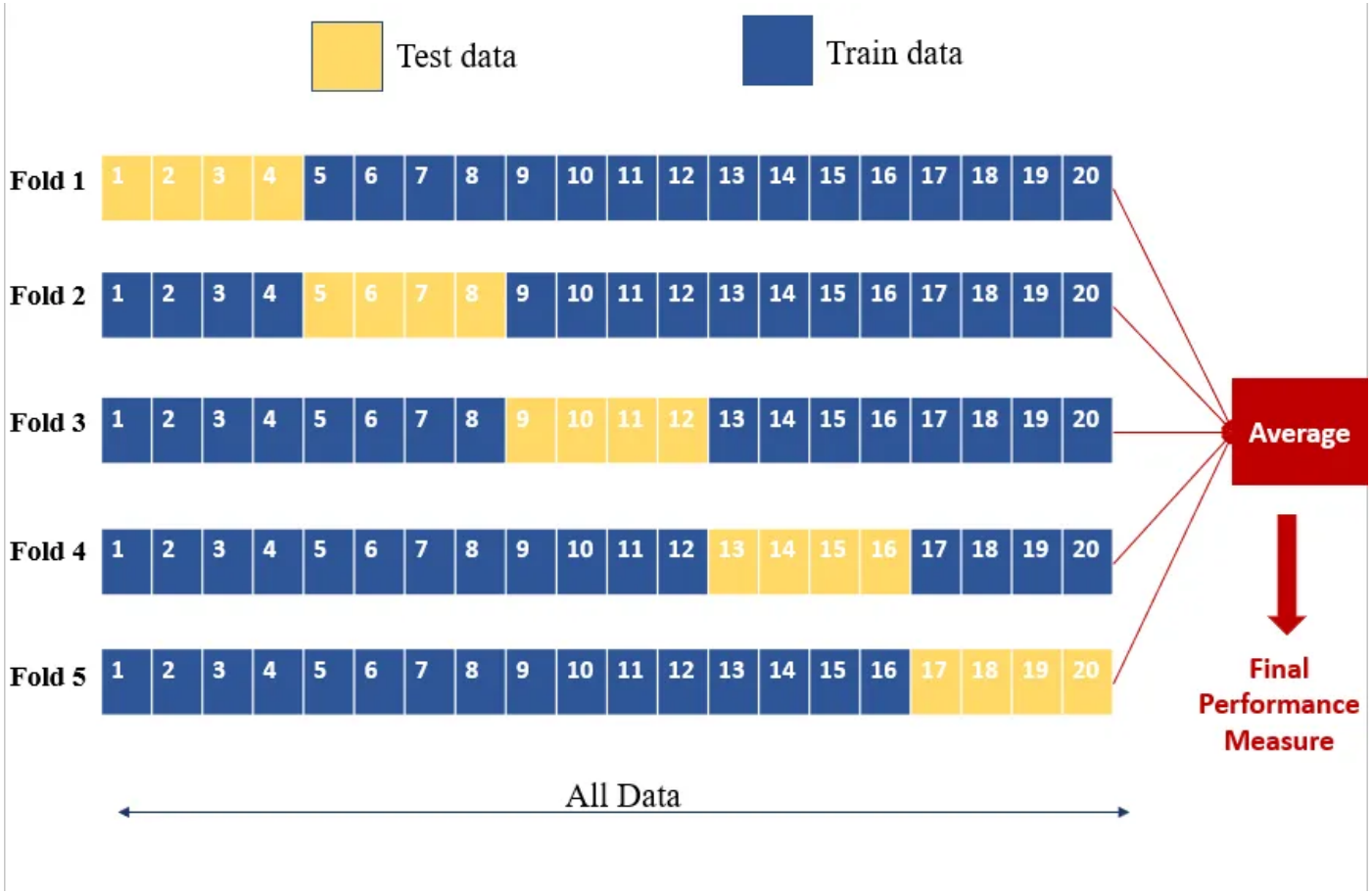
k-**fold** Cross-Validation steps

1. Partition the data into **k equal-sized subsets**, the training and the test sets
2. $\forall k$, pick **one subset** for the test set and the $\mathbf{k-1}$ others for the training set...
3. ... and proceed as for the 2-fold Cross-Validation
4. Sum the errors over the k runs

Stratified *k*-**fold** Cross-Validation steps

1. Partition the data into **k equal-sized subsets**, with similar a class distribution similar to the whole dataset...then same as *k*-**fold** Cross-Validation!

Each sample is used exactly once for testing



Advantages

- Each data point is used for both training and testing purpose.
- High accuracy of the method

Special case of K-Fold

The **leave-one-out** method, when $k = N$ (the number of samples)

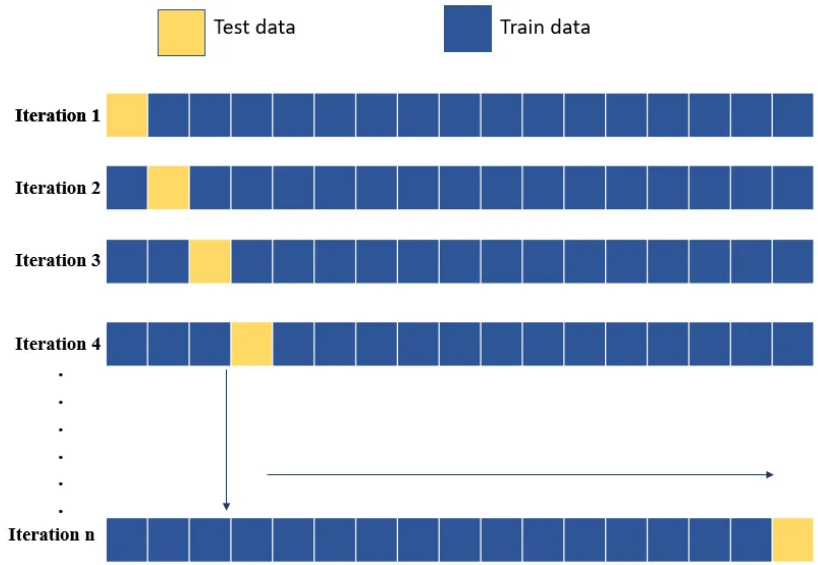
Advantages

- Use as much samples as possible for the training
- Mutually exclusive test sets that cover all the data

Drawbacks

- Computationally expensive (N runs)
- Small test set $\Rightarrow$ large variance of the error estimate

NB: not really used nowadays



For each bootstrap round, training samples are pooled **with replacement**!

Bootstrap steps

Start from a dataset of size N.

For b bootstrap rounds

1. Draw one instance from the data and assigned it to the current *boot sub-set*
1. Repeat **with replacement** until the size of the current *boot sub-set* is N (so, the *boot sub-set* contains duplicates!, and some instance are not chosen!)
2. Fit the model on the current *boot sub-set* $\rightsquigarrow ACC_{r,i}$
3. Test the model on the unselected instances (*out-of-bag* instances) $\rightsquigarrow ACC_{h,i}$

Bootstrap accuracy computation

$$acc_{boot} = \frac{1}{b}\sum_{i=1}^{b}(0.632 \times ACC_{h,i} + 0.368 \times ACC_{r,i})$$

where $ACC_{h,i}$ is the *out-of-bag* instances accuracy and $ACC_{r,i}$ is the accuracy from the training *boot sub-set*

NB: If the original data set has N samples, it can be shown that a bootstrap sample of size N contains about 63.2% of the samples, as the probability of chosing a sample is $1 - (1 - \frac{1}{N})^N$

General description

A performance measurement for the classification problems at various threshold settings.

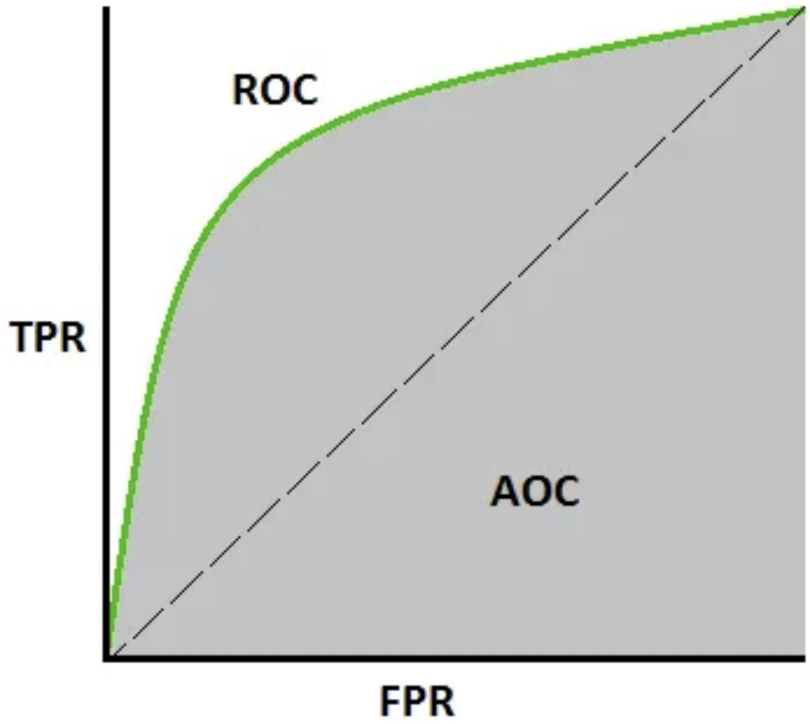
- ROC $\sim$ probability curve
- AUC $\sim$ degree or measure of separability
- ROC = Receiver Operating Characteristic
- AUC = Area Under the Curve

  → AUC $\rightsquigarrow 1 \Rightarrow$ excellent separability
  → AUC $\rightsquigarrow 0.5 \Rightarrow$ random classification
  → AUC $\rightsquigarrow 0 \Rightarrow$ classification reciprocates the results

$$\textbf{TPR}|\textbf{Recall}|\textbf{Sensitivity} = \frac{TP}{TP + FN}$$

$$\textbf{Specificity} = \frac{TN}{TN + FP}$$

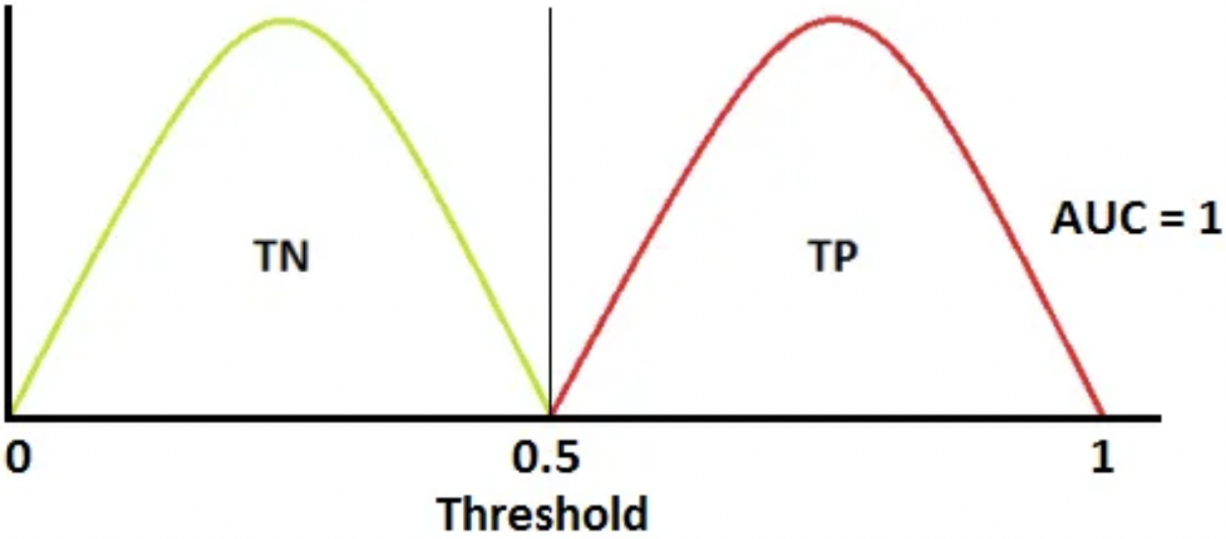
$$\textbf{FPR} = 1 - \textit{Specificity} = \frac{FP}{TN + FP}$$

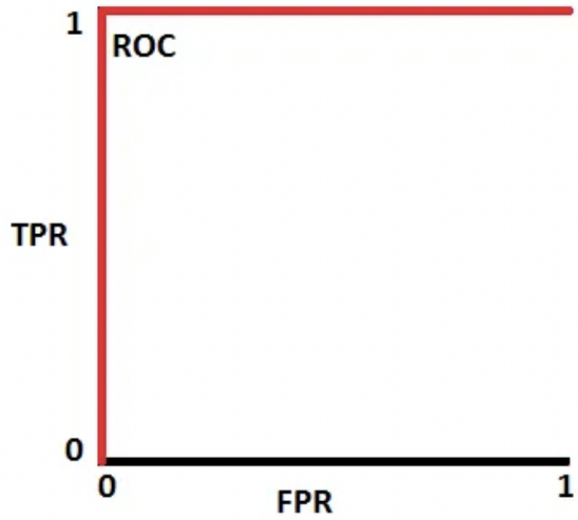


When the separation is perfect

- Red curve distribution is the positive class (eg. patient with disease)
- Green curve distribution is the negative class (eg. patient without disease)

$\rightarrow \Rightarrow$ The model can perfectly separate the classes at 0.5.

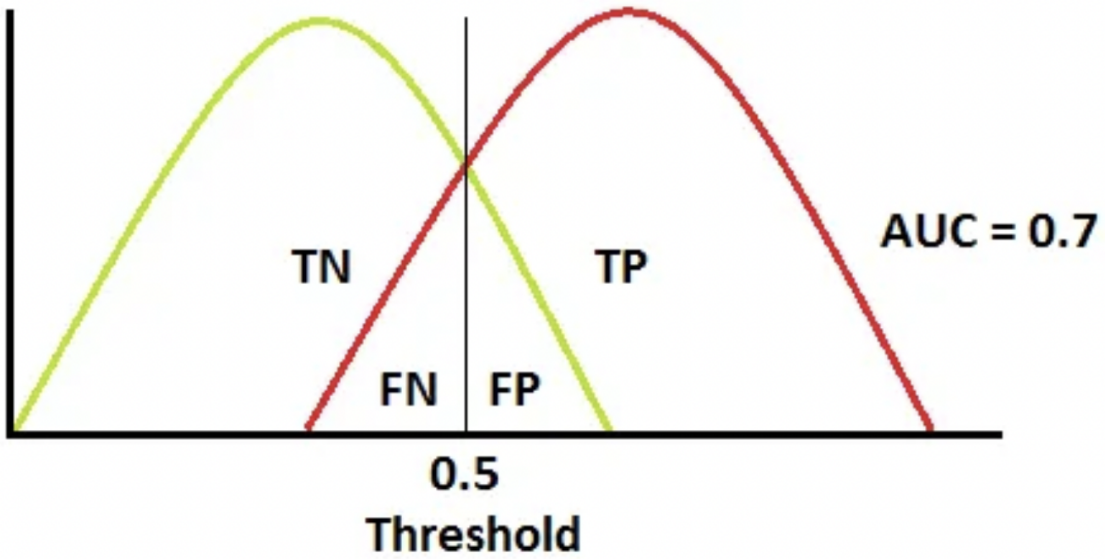


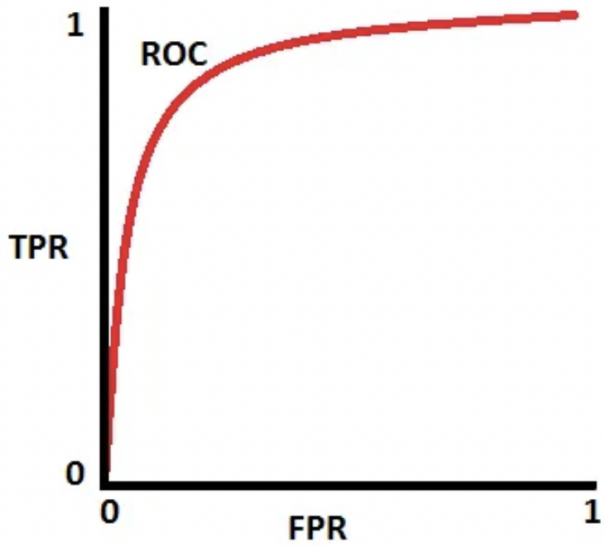


When the type of errors can be tuned with the threshold

As the two distributions overlap, we introduce type 1 (FP) and type 2 errors (FN).

→ AUC= 0.7 ⇒ there is 70% chance that the model will be able to distinguish between positive class and negative class.

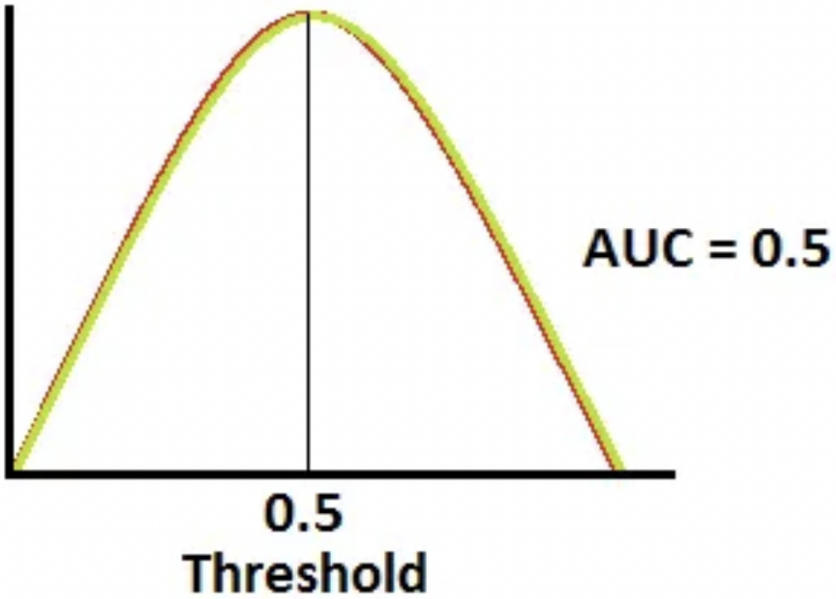


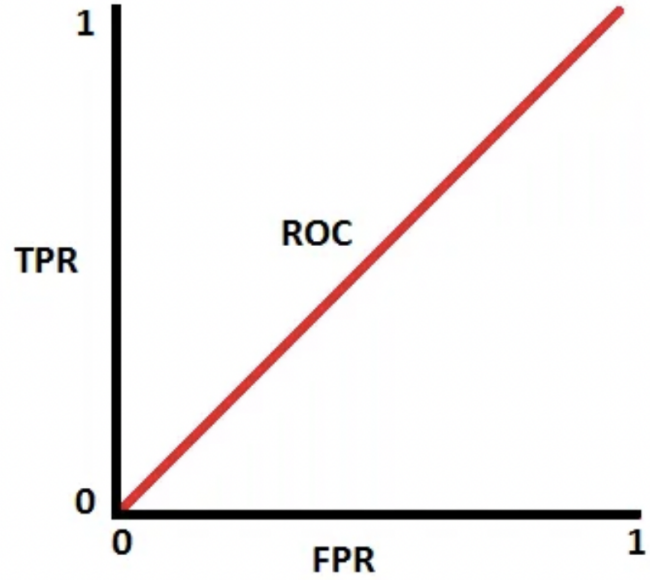


When the model is as good as random...

When the two distributions fully overlap, the model has no discrimination capacity to distinguish between positive class and negative class.

→ AUC= 0.5

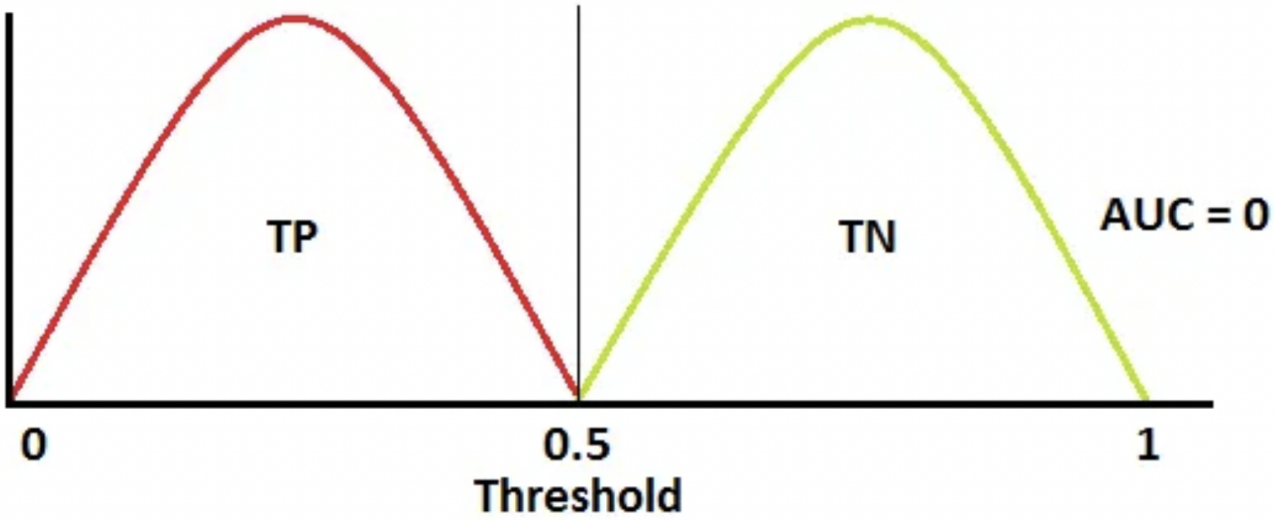


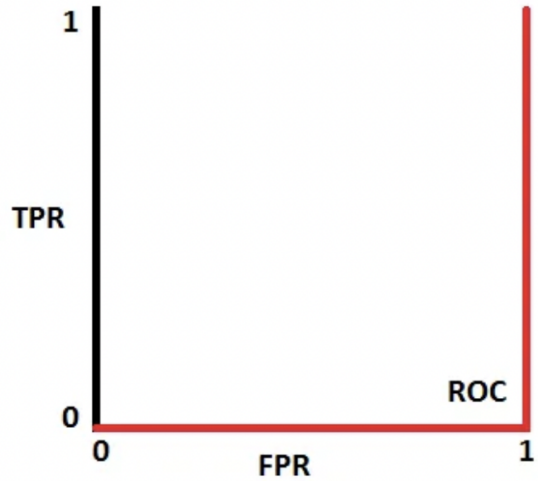


When the model inverts both classes...

The model is reciprocating the classes. In other words, the model is predicting a negative class as a positive class and vice versa.

→ AUC≈ 0.5





Sensitivity and Specificity

These metrics are inversely proportional to each other:

- When Sensitivity $\nearrow$, then Specificity $\searrow$
- When Sensitivity $\searrow$, then Specificity $\nearrow$

$$\textbf{TPR}|\textbf{Recall}|\textbf{Sensitivity} = \frac{TP}{TP + FN} \qquad \textbf{Specificity} = \frac{TN}{TN + FP}$$

Relation to threshold?

When we decrease the threshold, we get more positive values:

$\rightarrow$ it decreases the FN and increases FP
$\rightarrow$ it decreases Specificity and increases Sensitivity

FPR and TPR

The FPR is $1 - \textit{Specificity}$. So, when Specificity increases, the FPR decreases, as well as the TPR (which is the Sensitivity).
So...

- When FPR $\nearrow$, then TPR $\nearrow$
- When FPR $\searrow$, then TPR $\searrow$