Dimension reduction and classification methods for the analysis of experimental data

Italia De Feis

Istituto per le Applicazioni del Calcolo “M. Picone”

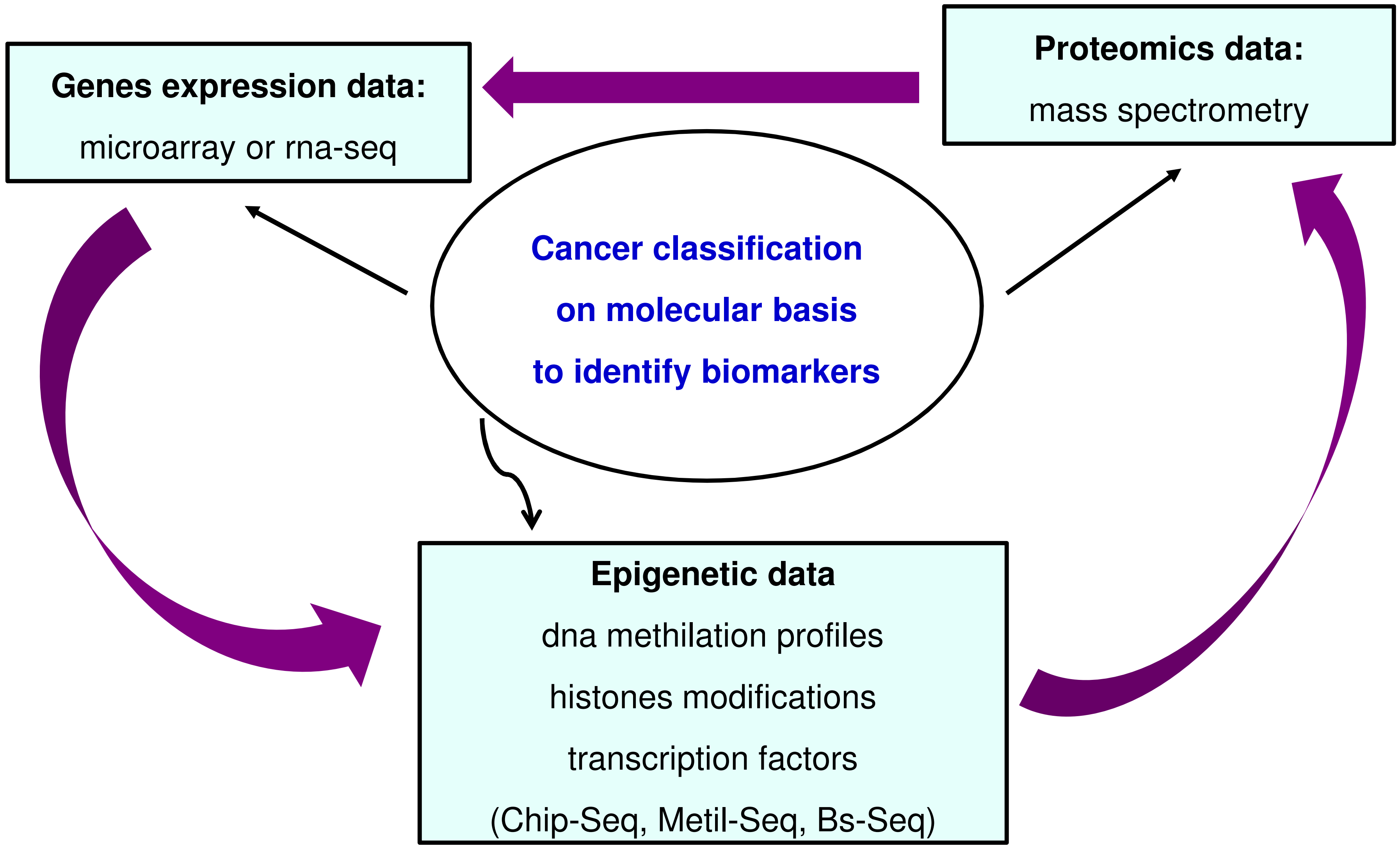
Classification

Supervised

To identify the class a new observation belongs to, starting from a **training dataset** and a rule built on this dataset

Unsupervised (clustering)

To identify the **hidden structures** in data with unknown a-prori information on the classes


Genes expression data:
microarray or rna-seq
Proteomics data:
mass spectrometry
Cancer classification
on molecular basis
to identify biomarkers
Epigenetic data
dna methilation profiles
histones modifications
transcription factors
(Chip-Seq, Metil-Seq, Bs-Seq)

Clinical Proteomic Tumor Analysis Consortium (CPTAC)

The **National Cancer Institute (NCI)**, part of the **National Institutes of Health (NIH)**, announced the launch of a **Clinical Proteomic Tumor Analysis Consortium (CPTAC)** in August 2011. CPTAC is a comprehensive and coordinated effort to accelerate the understanding of the molecular basis of cancer through the application of robust, quantitative, proteomic technologies and workflows.

Classification but…….

Neural Networks (NN)

Linear Discriminant Analysis (LDA)

Fuzzy methods

Learning Vector Quantization (LVQ)

Random Forests

Support Vector Machines (SVM)

Decision Tree

Multinomial Logistic Regression

Naive Bayes Classifier

K-Nearest Neighbours (KNN)

Quadratic Discriminant Analysis (QDA)

Given n observations (ex. # of spectra in a MS experiment)
each characterized by p characteristics/variables (ex. # of peaks)
Paradigm: $p >> n$

⬇

Dimension reduction / Variable Selection

⬇

Classification methods must keep into account !!!!

Dimension reduction / Variable Selection

Principal Component Analysis (PCA)

Correlation-Based Feature Selection (CFS)

Penalization

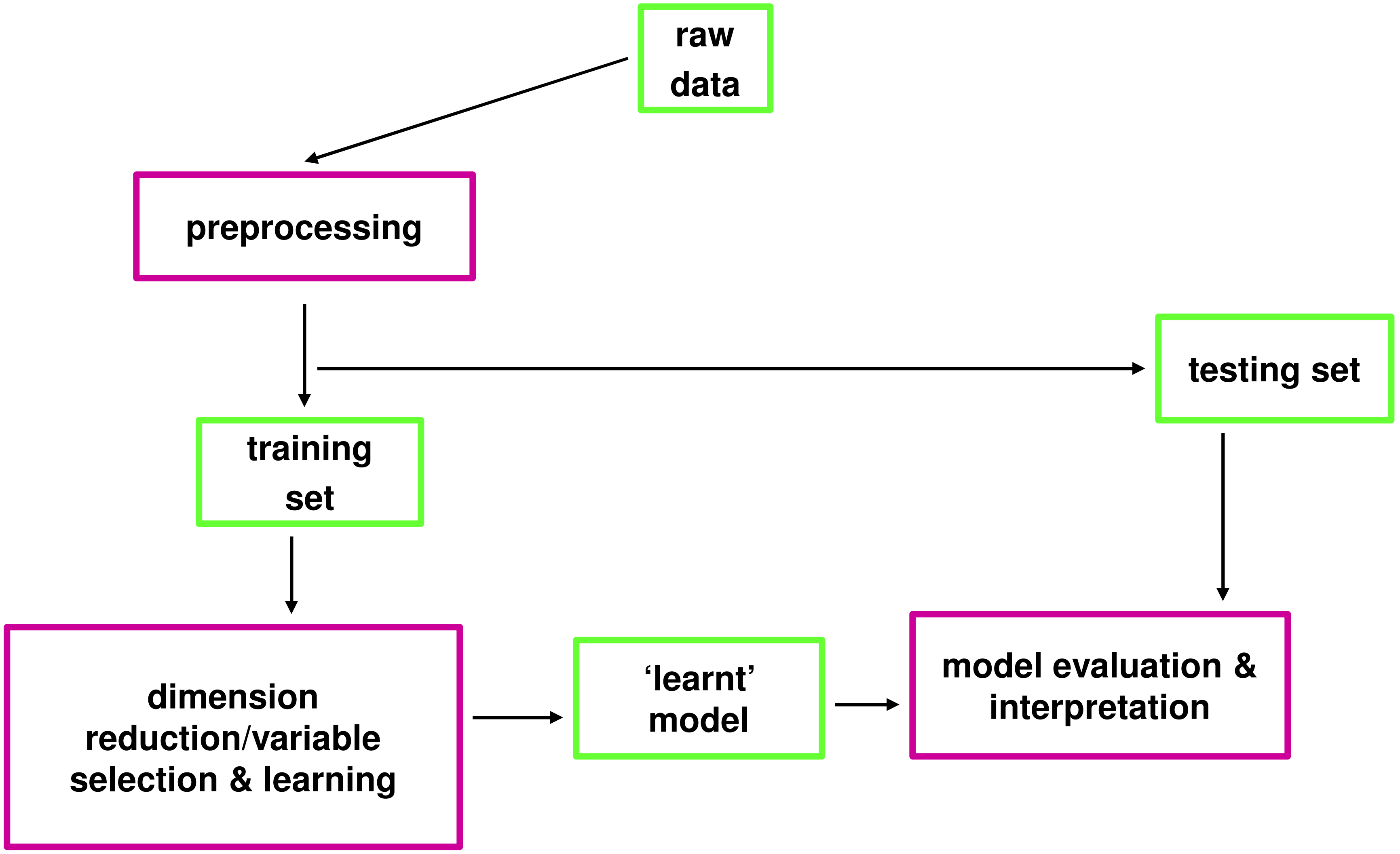
Partial Least Squares (PLS)

Hypothesis test

Genetic Algorithms (GA)

SVM-recursive feature elimination (SVM-RFE)

Classification and regression tree (CART)


raw data
preprocessing
testing set
training set
dimension reduction/variable selection & learning
'learnt' model
model evaluation & interpretation

$$\begin{tikzcd}
\boxed{\text{Dimension reduction}} \arrow[r, "\text{Ex PCA, PLS}"] & \boxed{\text{Given n samples, each characterized by p variables, ...}} \\
\boxed{\text{Variable selection}} \arrow[r, "\text{Ex hypothesis test, lasso}"] & \boxed{\text{Given n samples, each characterized by p variables, ...}}
\end{tikzcd}$$

Dimension reduction

Ex **PCA, PLS**

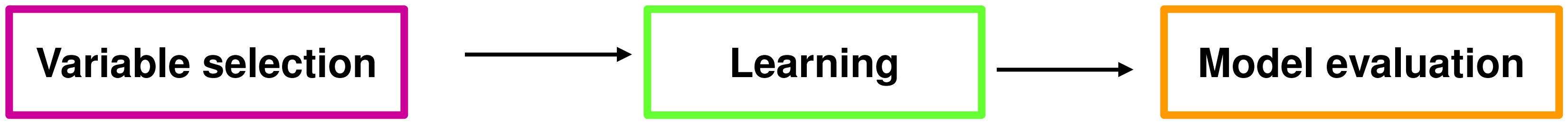
Given n samples, each characterized by p variables, it performs a combination of the p variables to preserve the maximum amount of information present in the original data and obtains n new samples of dimension p' <<p.

Variable selection

Ex **hypothesis test, lasso**

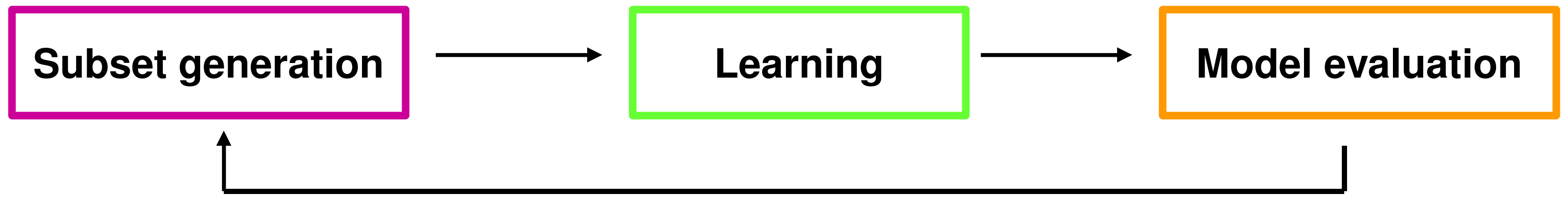
Given n samples, each characterized by p variables, it removes the 'noninformative' variables, to obtain the same n samples of dimension p' <<p.

Filter methods



The variable selection is independent of the classification algorithm and this is efficient from a computational point of view, but ignores that classification can depend from selection.

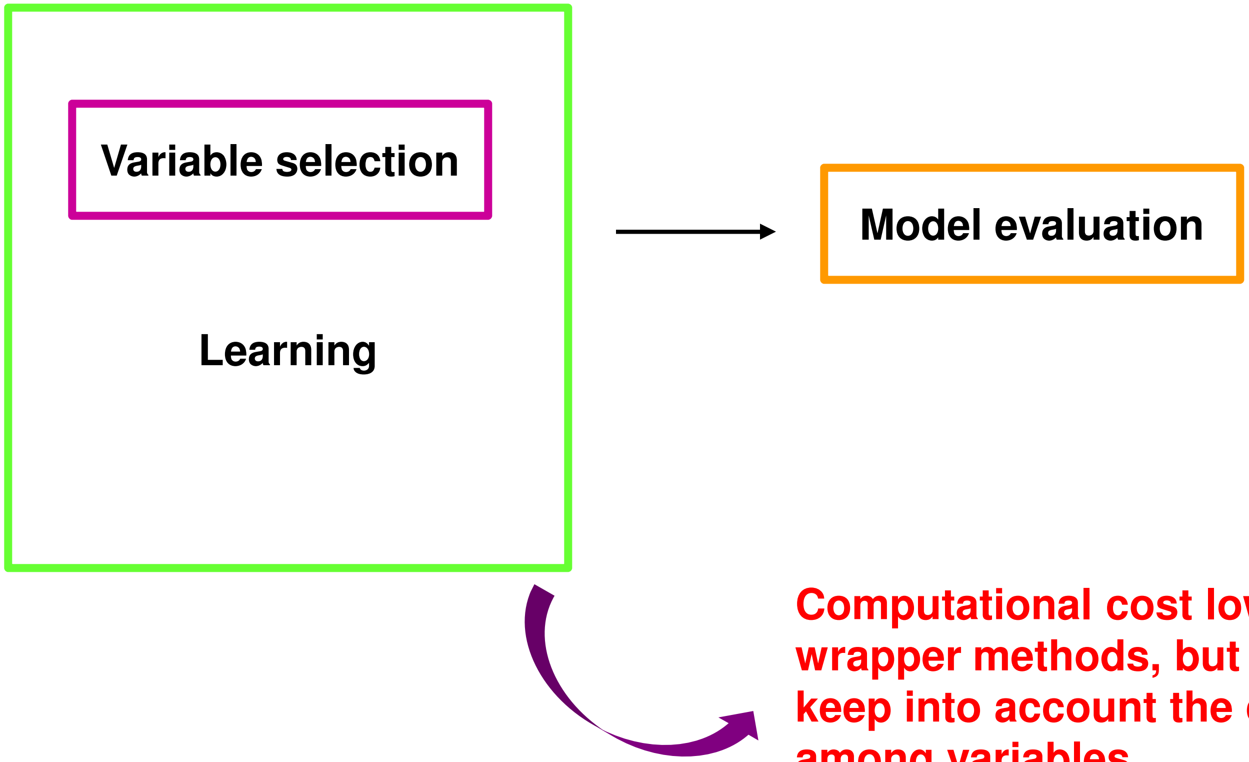
Wrapper methods



These are feedback methods that include the classification algorithm in the dimension reduction process. They search in the characteristics subsets space and evaluate the accuracy of a single classification algorithm for each characteristic that can be removed or added to the characteristics subset. The characteristics space exploration can be done by different strategies: forward (i. e., it add characteristics to a subset that is initially empty; backward (i. e., it starts from the complete set and remove a characteristic one by one → keep into account the correlation among the characteristics.

High computational cost

Embedded methods



Computational cost lower than wrapper methods, but they also keep into account the correlation among variables

Input data

$C_1, \dots, C_K$: classes;

X: matrix of dimension $n \times p$

Y: vector of dimension $n \times 1$

with values in $\{1,\dots,K\}$

It defines the classes labels:

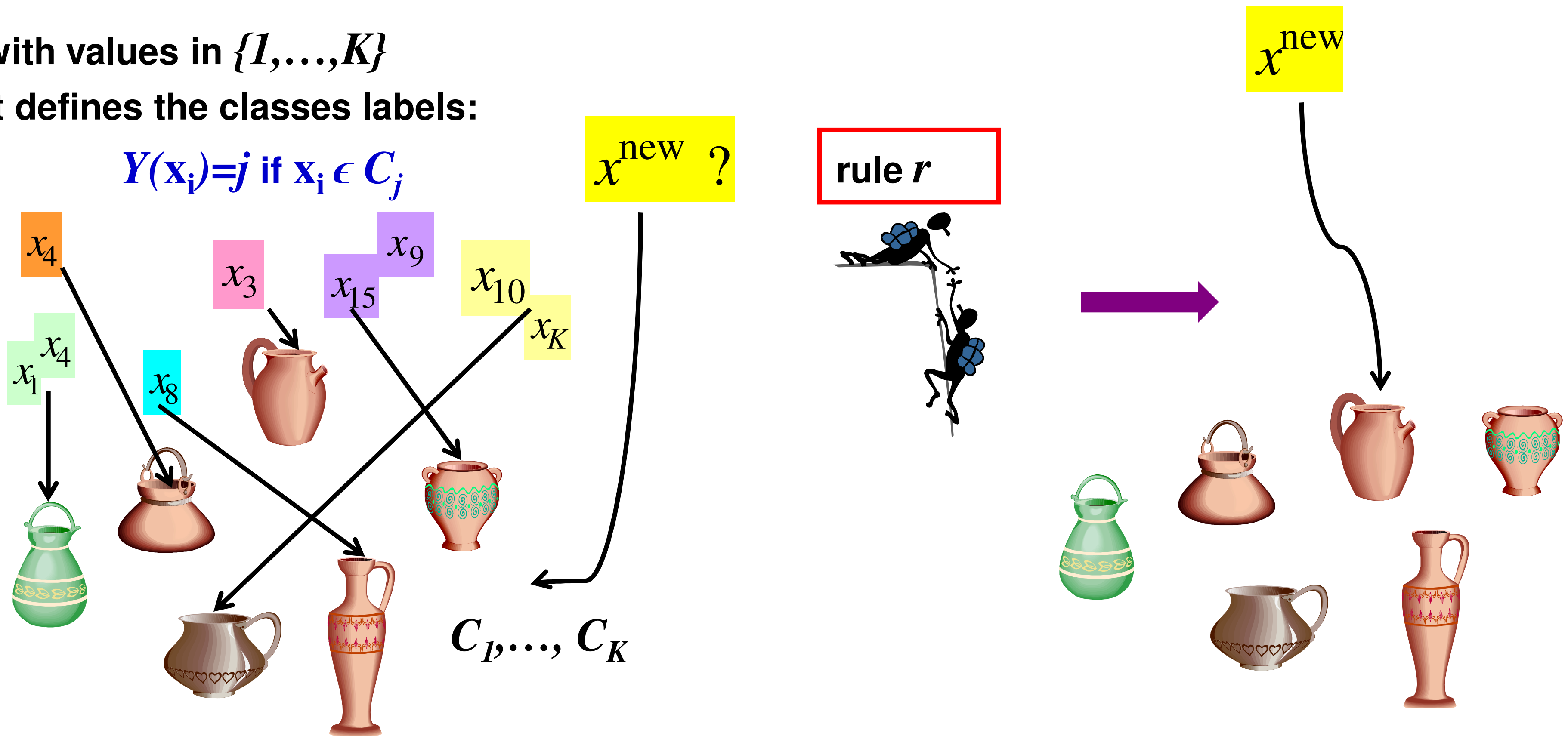
$Y(\mathrm{x_i}) = j$ if $\mathrm{x_i} \in C_j$

Learning

Building of the rule r

Prediction

Classification of new observations



Statistical decision theory

Bayes theorem

Information coming from data. It indicates the compatibility of this information with the given states of nature.

Degree of belief in states of nature, before data are observed

$$\boxed{\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}}$$

Posterior beliefs on states of nature, after experiment and data observation

Normalization costant: information on data averaged on all possible states of nature

Bayes theorem can be thought of as way of *coherently* updating our uncertainty in the light of new evidence.

$\mathbf{x}=(x_1,\ldots, x_p)$: observation from the training set, i.e. random sample from a r.v. X;

$C_1,\ldots, C_K$: classes.

Aim

$$\boxed{\mathbf{x} \xrightarrow{r(\cdot)} j, \quad j=1,\ldots k}$$

The rule $r(\cdot)$ depends on:

- **a priori information on classes probabilities: $\pi_1,\ldots, \pi_K$, $\Sigma_j \pi_j=1$**
- **information on the belonging of x to a class: $f_j(\mathbf{x})$: p.d.f. that x comes from C_j**
- **misclassification cost: to classify $x \in C_j$ in another class**

cost function 0-1

$$\boxed{Q(j,r(\mathbf{x})) = \begin{cases} 0, & \text{if } \ r(\mathbf{x}) = j \\ 1, & \text{if } r(\mathbf{x}) \neq j \end{cases}}$$

Bayes theorem

$$p\left(C_j / X = \mathbf{x}\right) = \frac{p\left(\mathbf{x} / C_j\right) p\left(C_j\right)}{\sum_{j=1}^{K} p\left(\mathbf{x} / C_j\right) p\left(C_j\right)} = \frac{f_j(\mathbf{x})\pi_j}{\sum_{j=1}^{K} f_j(\mathbf{x})\pi_j}$$

Posterior probability that $\mathbf{x} \in C_j$

Prior probability that $\mathbf{x}$ comes from C_j

$$\boxed{f(\mathbf{x}) = \sum_{j=1}^{n} f_j(\mathbf{x})\pi_j}$$

p.d.f. of X

rule r : assign $\mathbf{x}$ to the class having maximum posterior probability

Is *r* a "good" rule ?

Conditioned risk from x

$$R\left[r(\mathbf{x})=s/X=\mathbf{x}\right]=\sum_{j=1}^{K}p\left(C_j/X=\mathbf{x}\right)\cdot Q\left(j,r(\mathbf{x})=s\right)$$

$$\Downarrow$$

$$R\left[r(\mathbf{x})=s/X=\mathbf{x}\right]=\sum_{\substack{j=1\\ j\neq s}}^{K}p\left(C_j/X=\mathbf{x}\right)=1-p\left(C_s/X=\mathbf{x}\right)$$

Global risk: is the mean value of conditioned risk

Aim

to minimize the global risk $\leftrightarrow$ to minimize the conditioned risk for each x

$$\downarrow$$

r chooses the class that maximizes

$$\boxed{p\left(C_s/X=\mathbf{x}\right),\forall s=1,\ldots,K}$$

To apply the bayesian method it is necessary to know π_j and $f_j(x)$ or $p(C_j / X=x)$.

Non parametric methods: the form of the density is unknown and estimate it from the data, assuming some regularity condition it must satisfy.

Parametric methods: the form of the density is known and the training dataset is used to estimate the parameters.

The bayesian method belongs to the discriminant methods category

$$\boxed{\forall\, C_j : g_j(\mathbf{x}),\ j = 1,\ldots, K \quad \text{and} \quad \mathbf{x} \in C_i \ \ \text{if} \ \ g_i(\mathbf{x}) > g_j(\mathbf{x}),\ \forall j \neq i}$$

Discriminant functions

$$\boxed{g_i(\mathbf{x}) = g_j(\mathbf{x}),\ \forall j \neq i} \longrightarrow$$ Decision boundary

$$\boxed{\text{if} \ \ g_j(\mathbf{x}) = \mathbf{w}_j^t \cdot \mathbf{x} + w_{j0}, \quad j = 1,\ldots, K} \longrightarrow$$ Linear discriminant

$$\boxed{\text{if} \ \ g_j(\mathbf{x}) = \mathbf{x}^t \cdot \mathbf{W} \cdot \mathbf{x} + \mathbf{w}_j^t \cdot \mathbf{x} + w_{j0}, \quad j = 1,\ldots, K} \longrightarrow$$ Quadratic discriminant

$$\boxed{\text{se} \ \ g_j(\mathbf{x}) = \mathbf{a}_j^t \mathbf{y} \ \ j = 1,\ldots, K \ \ \text{with} \ \ \mathrm{y} = \left(\varphi_1(\mathbf{x}),\ldots,\varphi_{\hat{p}}(\mathbf{x})\right)^t} \longrightarrow$$ Generalized linear discriminant

For the bayesian method

$$g_j(\mathbf{x}) = P\left(C_j / X = \mathbf{x}\right)$$
$$\Leftrightarrow$$
$$g_j(\mathbf{x}) = f_j(\mathbf{x})\pi_j$$
$$\Leftrightarrow$$
$$g_j(\mathbf{x}) = \ln f_j(\mathbf{x}) + \ln \pi_j$$

Moreover to study

$$P\left(C_i / X = \mathbf{x}\right) \text{and} \ P\left(C_j / X = \mathbf{x}\right), \forall j \neq i$$
$$\Leftrightarrow$$
$$\log \frac{P\left(C_i / X = \mathbf{x}\right)}{P\left(C_j / X = \mathbf{x}\right)}$$

log-odds

Linear Discriminant Analysis (LDA) e Quadratic Discriminant Analysis (QDA)

$$f_j(\mathbf{x}) \approx N_p(\boldsymbol{\mu}_j, \Sigma_j) = (2\pi)^{-p/2} |\Sigma_j|^{-1/2} \exp\left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_j)^t \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) \right)$$

Discriminant functions

$$g_j(\mathbf{x}) = -\frac{1}{2}\log|\Sigma_j| - \cancel{\frac{p}{2}\ln 2\pi} - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) + \log \pi_j$$

It is constant

It is quadratic in x: QDA

Case $\Sigma_j = \Sigma$, for each j

$$g_j(\mathbf{x}) = \mathbf{x}^T \Sigma^{-1} \boldsymbol{\mu}_j - \frac{1}{2}\boldsymbol{\mu}_j^T \Sigma^{-1} \boldsymbol{\mu}_j + \log \pi_j$$
$$= \mathbf{x}^T \boldsymbol{\beta}_j + \beta_{j0}$$

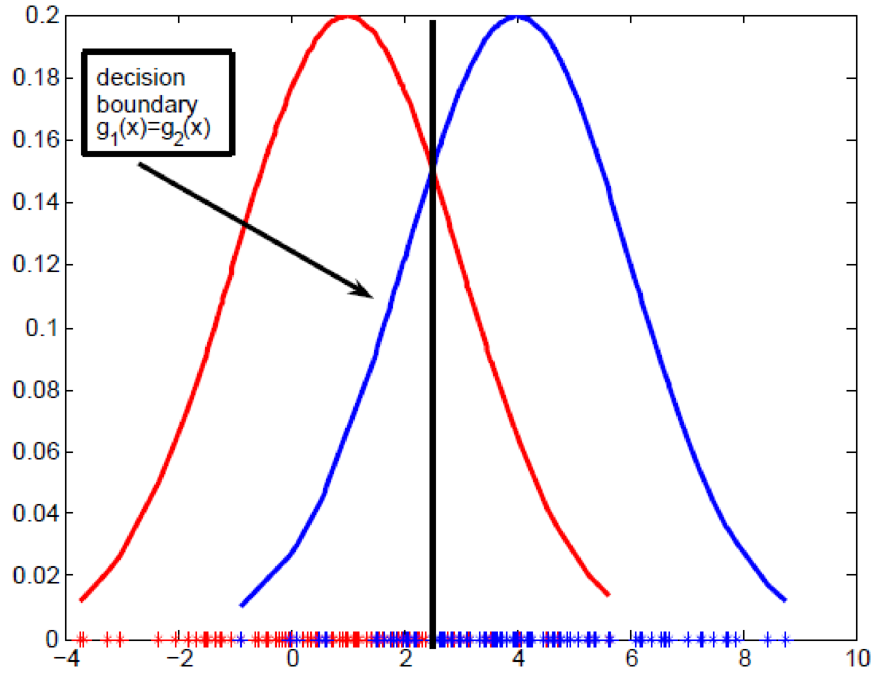
It is linear in x: LDA
It is a hyperplane
(for $p=1$ is a line)

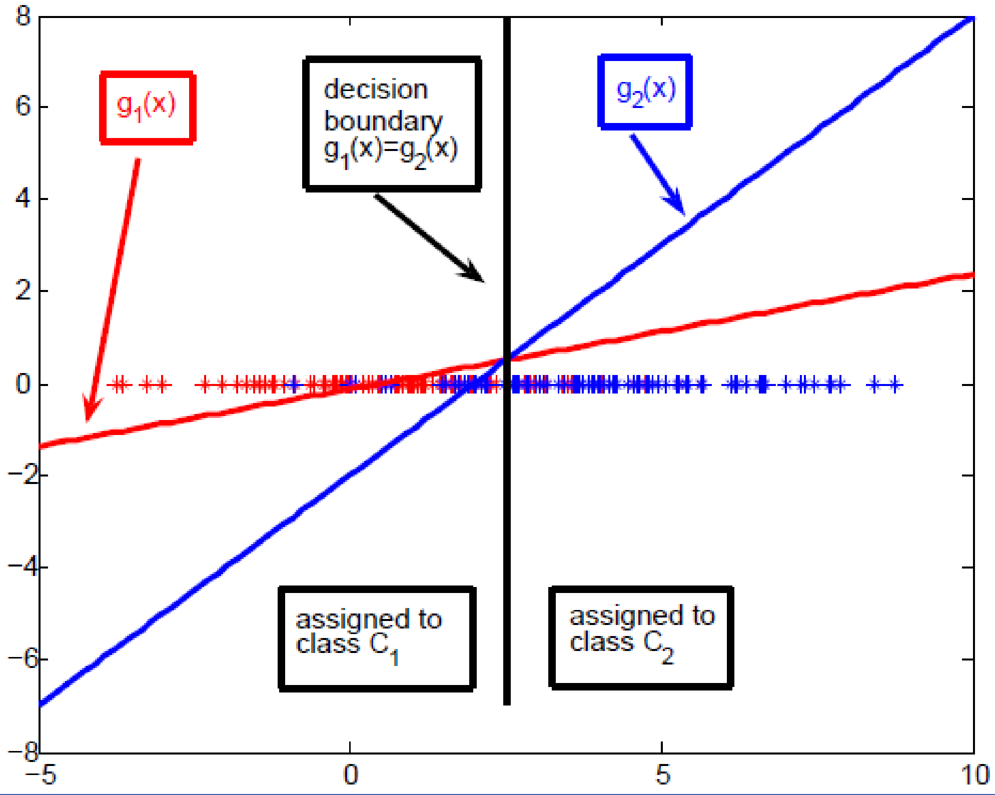
QDA is more flexible than LDA permitting non linear decision boundaries (ex.p=1 parabolas) but is more computationally expensive: μ_j and Σ_j must be estimated from data !!

Example

Let us consider $p=1, K=2, \pi_1 = \pi_2 = 1/2, n=100$:

$$\boxed{\begin{aligned} f(x) &= \pi_1 f_1(x) + \pi_2 f_2(x) = \\ &= \pi_1 N(1,4) + \pi_2 N(4,4) \end{aligned}}$$





Flexible discriminant analysis

It is a method to perform LDA on derived responses. The responses are obtained by assigning scores to the classes such that the transformed class labels are optimally predicted by regression on X. Let θ_l be a function that gives scores to the classes, *l=1,…,K-1*.

$$\boxed{\theta_l : \{1,\dots,K\} \to \Re \Leftrightarrow \theta_l(y_j) = s_{lj} \in \Re}$$

We then solve:

$$\boxed{\min_{\substack{\beta_1,\vartheta_1 \\ l=1,..K-1}} \frac{1}{n} \sum_{l=1}^{K-1} \sum_{j=1}^{n} \left(\theta_l(y_j) - \bigcirc\!\!\!\!\!\!\!\!\quad \mathbf{x}_j^t \boldsymbol{\beta}_l\right)^2}$$

can be replaced by non-parametric regression $\eta_l(x_j)$. Useful for overcoming the drawbacks of linear separations in LDA.

After solving the minimization, we can perform classification using a weighted nearest centroid rule, i.e. assign a new sample X to the class j that maximizes

$$\boxed{\sum_{l=1}^{K-1} w_l \left(\mathbf{x}^t \boldsymbol{\beta}_l - \overline{\eta}_l^j\right)^2 \text{ with } \overline{\eta}_l^j = \frac{1}{|C_j|} \sum_{\mathbf{z} \in C_j} \mathbf{z}^t \boldsymbol{\beta}_l}$$

Evaluated from the residuals of the minimization

Multinomial Logistic Regression

A multinomial distribution models an experiment of n independent repeated trials with K possible outcomes and probability p_j for the j-th outcome→ in classification n trials = n training samples, K outcomes = K classes, $p_j=p(C_j/X=x)$.

The link between x and p_j is expressed through the log odds that are assumed to be linear functions of x.

$$\log\frac{P(C_1/X=\mathbf{x})}{P(C_K/X=\mathbf{x})}=\beta_{10}+\mathbf{x}^{\mathbf{t}}\boldsymbol{\beta}_1$$
$$\log\frac{P(C_2/X=\mathbf{x})}{P(C_K/X=\mathbf{x})}=\beta_{20}+\mathbf{x}^{\mathbf{t}}\boldsymbol{\beta}_2$$
$$\vdots$$
$$\log\frac{P(C_{k-1}/X=\mathbf{x})}{P(C_K/X=\mathbf{x})}=\beta_{k-10}+\mathbf{x}^{\mathbf{t}}\boldsymbol{\beta}_{K-1}$$

$$P(C_j/X=\mathbf{x})=\frac{\exp(\beta_{j0}+\mathbf{x}^{\mathbf{t}}\boldsymbol{\beta}_j)}{1+\sum_{l=1}^{K-1}\exp(\beta_{l0}+\mathbf{x}^{\mathbf{t}}\boldsymbol{\beta}_l)}$$
$$j=1,\dots,K-1$$
$$P(C_K/X=\mathbf{x})=\frac{1}{1+\sum_{l=1}^{K-1}\exp(\beta_{l0}+\mathbf{x}^{\mathbf{t}}\boldsymbol{\beta}_l)}$$

The parameters estimation is done by log likelihood maximization

K=2 (binary case, logit model)

C_1 is encoded by $Y=1$ and C_2 is encoded by $Y=0$

$$P(Y=1/X=\mathbf{x})=\frac{\exp(\beta_0+\mathbf{x}^t\boldsymbol{\beta})}{1+\exp(\beta_0+\mathbf{x}^t\boldsymbol{\beta})}$$

$$P(Y=0/X=\mathbf{x})=1-P(Y=1/X=\mathbf{x})=\frac{1}{1+\exp(\beta_0+\boldsymbol{\beta}^t\mathbf{x})}$$

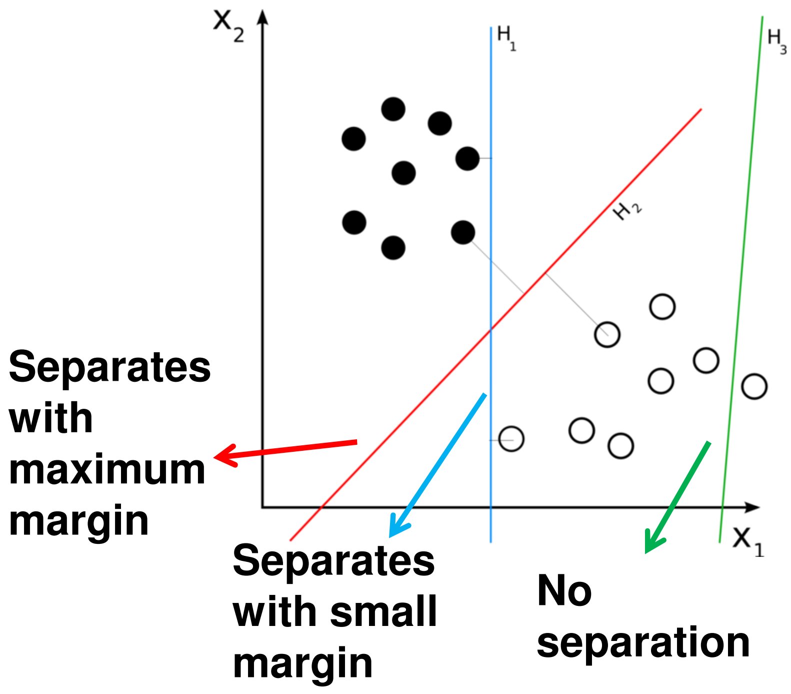
Log likelihood is

$$\sum_{j=1}^{n}\left[y_i\log\frac{\exp(\beta_0+\mathbf{x}_i^t\boldsymbol{\beta})}{1+\exp(\beta_0+\mathbf{x}_i^t\boldsymbol{\beta})}+(1-y_i)\log\left(\frac{1}{1+\exp(\beta_0+\mathbf{x}_i^t\boldsymbol{\beta})}\right)\right]$$

Support Vectors Machine (SVM), binary case $K=2$

A reasonable choice of linear discriminant function is the hyperplane having the largest distance to the nearest training data point of any class (so-called functional margin).

If the classes intersect we search a hyperplane that separates the classes in the "cleanest" way and, at the same time, maximizes the distance from the nearest and cleanest separation.



↓

Optimization problem:

$$\boxed{\begin{gathered}\sum_{i=1}^{n}\left[1-\tilde{y}_i\left(\beta_0+\mathbf{x}_i^t\boldsymbol{\beta}\right)\right]_+ +\tau\|\boldsymbol{\beta}\|_2^2 \\ \tilde{Y}=2Y-1\end{gathered}}$$

For $K>2$ we need to solve many binary problems:

✓ one-versus-all → strategy winner-takes-all

✓ one versus one → strategy max-wins voting

Penalization methods (linear case)

The penalization methods are embedded methods, i.e. they perform simultaneously the variable selection and the classification by minimizing a penalized objective function that estimates β:

$$\boxed{\hat{\boldsymbol\beta} = \arg\min_{\beta} \{ m(\boldsymbol\beta; \mathbf{D}) + \lambda \, \mathrm{pen}(\boldsymbol\beta) \}}$$

Objective function for classification

dataset (X,Y)

penalty: controls the complexity of the model, the choice of the penalty permits to put 0 the uninformative components of β

Parameter that balances the goodness of fit and the model complexity:

- **$\lambda \to 0$ the goodness of fit is better, but the classifier is too complex, it has a small predictive capability and less interpretability;**
- **$\lambda \to \infty$ the classifier has less variables, and $\lambda = \infty$ indicates that no variables is informative ($\beta = 0$).**

Some common objective functions (binary case):

$$m(\boldsymbol{\beta};\mathbf{D}) = \begin{cases} -\dfrac{1}{n}\displaystyle\sum_{i=1}^{n}\left[y_i \log P(Y=1/X=\mathbf{x}) + (1-y_i)\log P(Y=0/X=\mathbf{x})\right] \\ \dfrac{1}{n}\displaystyle\sum_{i=1}^{n}\left(\theta(y_i)-\mathbf{x}_i^t\boldsymbol{\beta}\right)^2 \longrightarrow \text{FDA} \\ \dfrac{1}{n}\displaystyle\sum_{i=1}^{n}\left[1-\tilde{y}_i\left(\beta_0+\mathbf{x}_i^t\boldsymbol{\beta}\right)\right]_+ + \tau\left\|\boldsymbol{\beta}\right\|_2^2 \longrightarrow \text{SVM} \end{cases}$$

$\downarrow$ Logistic regression

Penalty for variable selection

$$
pen(\boldsymbol{\beta}) = \begin{cases}
\sum_{j=1}^{p} |\beta_j| \longrightarrow \text{LASSO} \\
\sum_{j=1}^{p} \frac{1}{|b_j|} |\beta_j| \longrightarrow \text{Adaptive LASSO} \\
\sum_{j=1}^{p} |\beta_j|^{\gamma}, \; 0 < \gamma \leq 1 \longrightarrow \text{Bridge} \\
\sum_{j=1}^{p} |\beta_j|^{\gamma} + \left(\sum_{j=1}^{p} \beta_j^2 \right)^{\eta}, \; 0 < \gamma \leq 1 \text{ e } \eta \geq 1 \longrightarrow \text{Elastic net}
\end{cases}
$$

The lasso method needs a strong condition for selection consistency (in real cases it is difficult to have it) but it is a convex optimization problem with a low computational cost. A lot of bioinformatics papers have shown the good performances of lasso in classification problem with high dimensionality.

The adaptive lasso method needs light conditions than lasso and has its simplicity, also if the weights choice is still not optimal.

The bridge method is consistent if the correlations among discriminant features of the classes are weak, but there is still not a satisfying algorithm.

The elastic net works for high correlated characteristics, but can be inconsistent as lasso if $\gamma=1$ or can have the computational difficulties of the bridge if $\gamma<1$.

$$pen(\boldsymbol{\beta}) = \sum_{j=1}^{p} \begin{cases} |\beta_j| & \text{se } |\beta_j| \le \lambda \\ -\dfrac{|\beta_j|^2 - 2a\lambda|\beta_j| + \lambda^2}{2\lambda(a-1)} & \text{se } \lambda < |\beta_j| \le a\lambda \\ (a+1)\lambda/2 & \text{se } |\beta_j| > a\lambda \end{cases} \longrightarrow$$

SCAD,

a is a tuning parameter,

suggested $a=3.7$

The SCAD method is consistent, and in some papers it has been shown its good performance with respect to lasso-type penalties, but from a computational point of view it is less simple to treat.

Measures for classifier performance

Binary case

- **TP = # true positives,**
  number of molecular measures that are biomarkers and are classified as biomarkers

- **FP = # false positives,**
  number of molecular measures that are not biomarkers and are classified as biomarkers

- **TN = # true negatives,**
  number of molecular measures that are not biomarkers and are classified as not biomarkers

- **FN = # false negatives,**
  number of molecular measures that are biomarkers and are classified as not biomarkers

Confusion matrix

Predicted class (observation)	Actual class (expectation)	
	TP	FP
	FN	TN

Predicted positive class

Predicted negative class

True positive class

True negative class

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

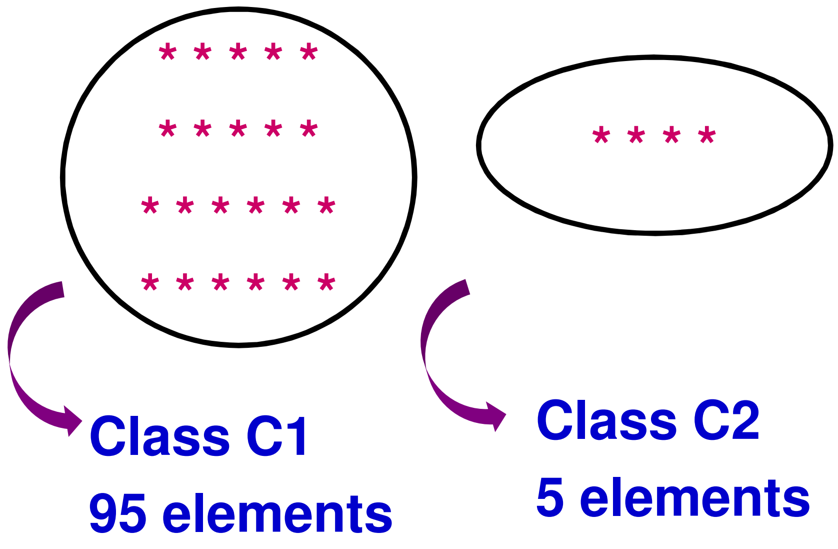
It measures the proportion of test cases correctly classified.

Ideal value: accuracy→ 1

Error rate = 1-accuracy
(Missclassification error)

If the class are unbalanced: problem with accuracy !!!!

Let us suppose to classify 100 samples:



We consider the following rule:
Each sample assigned to class C1

$\boxed{\text{Accuracy = 95\% !!!!!}}$

This index fails because gives no indication that the 0% of the elements in class C2 are identified !!!

Besides accuracy, other indexes are considered

- **Sensitivity/Recall** $\boxed{\dfrac{TP}{TP+FN}}$ $\rightarrow$ **Completeness measure. It is the fraction of true positive class that is retrieved. Ideal value: sensitivity→ 1**

- **Specificity** $\boxed{\dfrac{TN}{TN+FP}}$ $\rightarrow$ **Completeness measure. It is the fraction of true negative class that is retrieved. Ideal value: specificity→ 1**

They are not sensitive to data distribution

- **Precision /Positive Predicted Value (PPV)** $\boxed{\dfrac{TP}{TP+FP}}$ $\rightarrow$ **Accuracy measure. It is the fraction of predicted positives that are true positives. Ideal value: PPV→ 1**

- **Negative Predicted Value (NPV)** $\boxed{\dfrac{TN}{TN+FN}}$ $\rightarrow$ **Accuracy measure . It is the fraction of predicted negatives that are true negatives. Ideal value: NPV → 1**

- **F-measure**

$$\boxed{\frac{(1+\beta^2)\cdot(recall \times precision)}{\beta^2 \cdot precision + recall}}$$

→ **Efficacy measure. Ideal value: F-measure → 1**

β tunes the relative importance of precision with respect to recall.

- **G-measure**

$$\boxed{\sqrt{specificity \times recall}}$$

It is a measure of inductive bias, i.e. the assumptions on the objective function linking input x/ output j (class label)

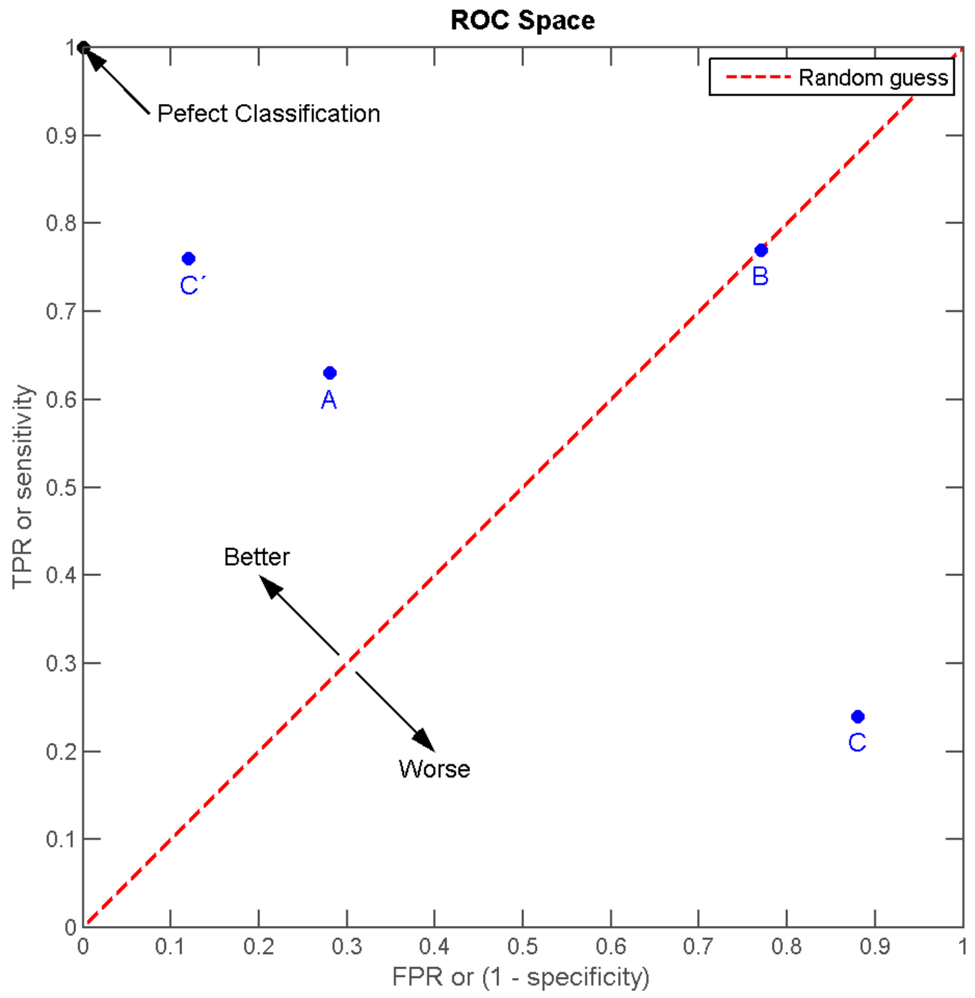
ROC curves (Receiver Operating Characteristics)

They are a graphical scheme for a binary classifier. The y-axes represents the sensitivity (*True Positive Rate*) and the x-axes represents the (1-specificity) (*False Positive Rate*).

It is a useful instrument because it visualizes the relative balance benefits (TP) / costs (FP).

Hard-type classifier: output is class labels → each classifier will output a single point (TP rate; FP rate) in the ROC space:

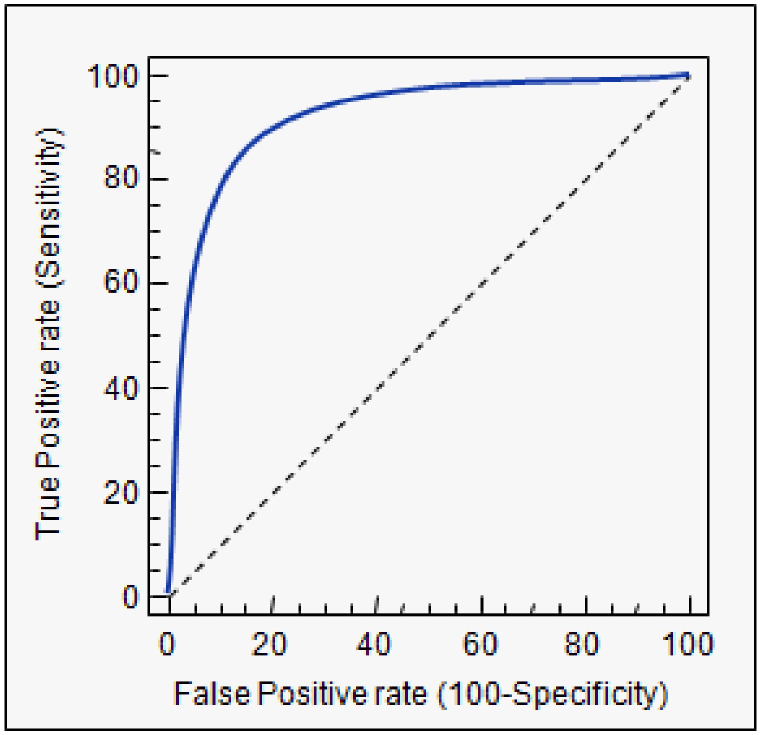
- **Point (0,1): perfect classifier classifies all the samples (positives/negatives) correctly.**
- **Point (0,0): classifier that predicts all samples as negatives.**
- **Point (1,1): classifier that predicts all samples as positives.**
- **Point (1,0): classifier completely wrong for all samples.**



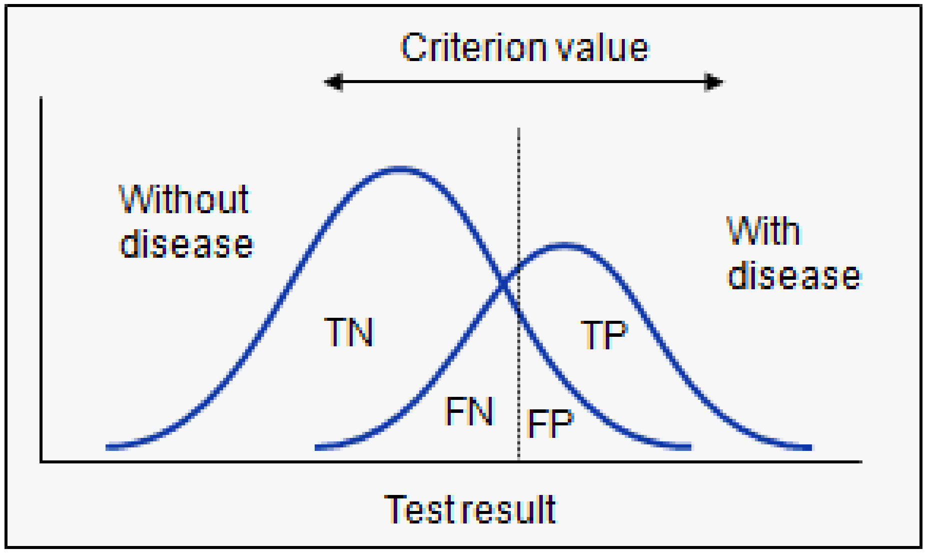
- **A classifier is better than another one if the corresponding ROC point is closer to (0,1) than the ROC point corresponding to the other classifier.**
- **A classifier whose ROC point is on the diagonal represents a 'random guess' classifier (i.e. it is equivalent to flip a coin).**

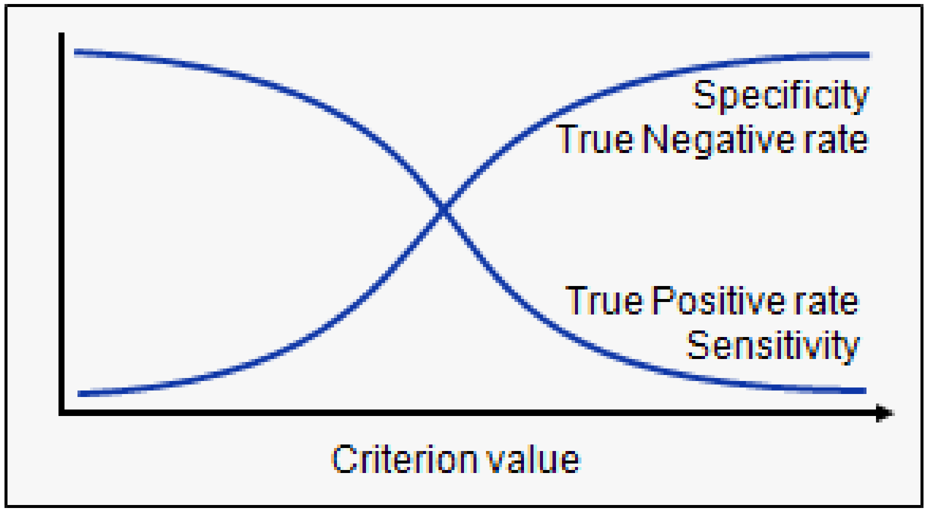
- **Soft-type classifier:** the output is a continuos numerical value representing the confidence (probability) that a sample belongs to a predicted class. In this case we can use a threshold to produce a series of points in the ROC space.

We obtain curves and not single points.



Ex imagine that the blood protein levels in diseased people and healthy people are normally distributed with means of 2 g/dL and 1 g/dL respectively. A medical test might measure the level of this protein in a blood sample and classify any number above a certain threshold as indicating disease. The experimenter can adjust the threshold (black vertical line in the figure), which will in turn change the false positive rate





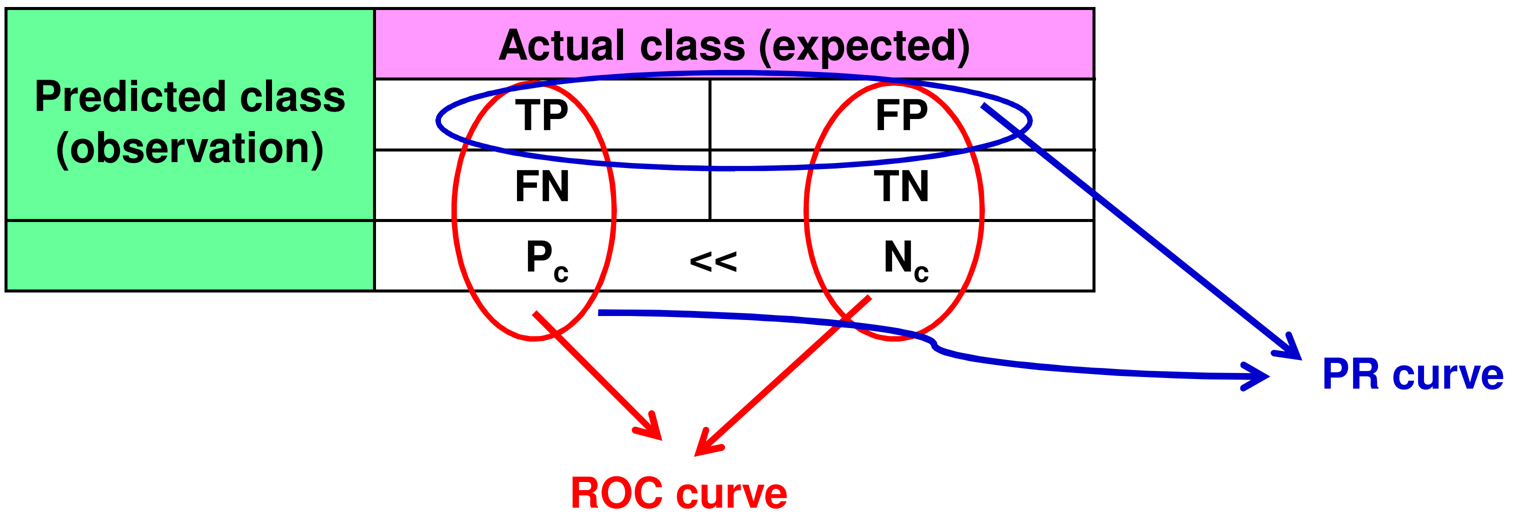
A perfect classifier has a ROC curve that passes through the left upper angle (100% sensitivity, 100% specificity).

Strategy to evaluate the classifier efficiency: AUC (Area Under the Curve).

PR curve (Precision/Recall)

It is another graphical scheme for binary classifier more informative than ROC curve when classes are strongly unbalanced. Along the axes we can represent precision/PPV and Recall/Sensitivity.

Ex: let us suppose $N_c >> P_c$. In this case varying the number of false positives, the FP rate (=FP/ N_c) will not change significantly because the negative class is large, and the ROC curve will not capture this phenomenon. On the contrary the PR curve will take into account this because it considers TP/(TP+FP)=precision/PPV.



Predicted class (observation)	Actual class (expected)	
	TP	FP
	FN	TN
	P_c <<	N_c

How to evaluate a classifier

Independently from the chosen metrics, we have to consider the strategy to evaluate the performance of a method on available data. A good classifier must be "generalizable" from training set to testing set↔ "unseen data", independent from the ones in the training set.

Ideal training and testing sets

✓ **for both the classification must be known;**

✓ **both must be representative samples of the phenomenon under study, and the sample proportions reflect the proportion of real population;**

✓ **both the dataset must be "large"**

- **the larger the training dataset the better is the classifier**
- **the larger is the testing dataset the more confident is the estimated accuracy**

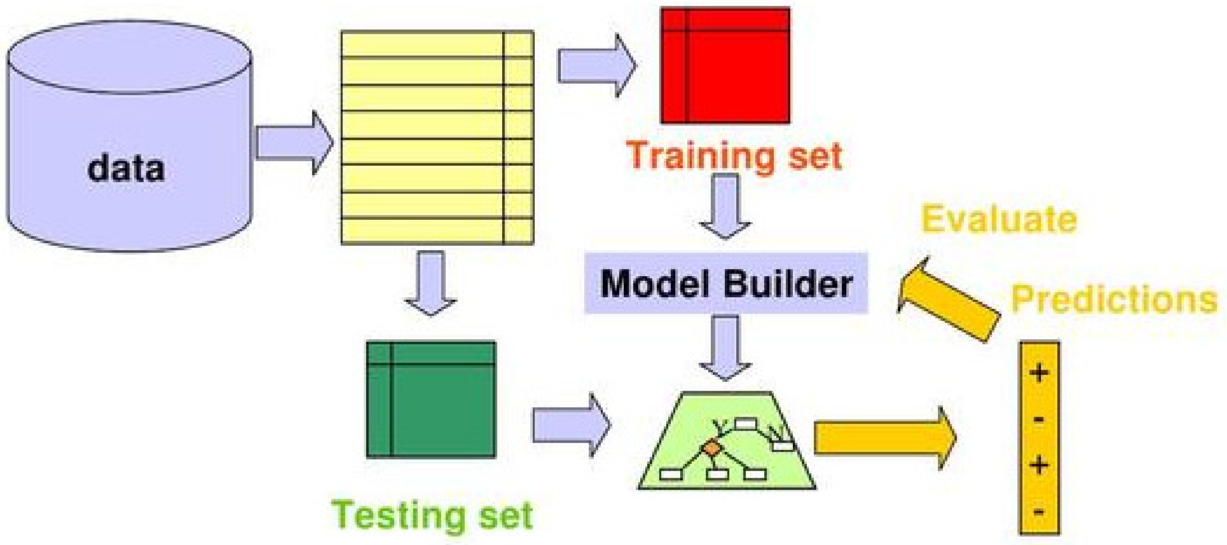
It is difficult to collect two independent and large dataset, representative of instances whose classification is known !!!!

How to evaluate a classifier

Let us suppose to have available only a dataset $\mathcal{D}$ whose classification is known (labels), there exist different methods to build training and testing dataset to evaluate the classifier performance.

✓ Hold out → the dataset is randomly divided in 2 parts: one is used as training and the other one as testing. Generally:

1/2 for training dataset and 1/2 for testing dataset
2/3 for training dataset and 1/3 for testing dataset



Note that

✓ It is fundamental that the testing dataset doesn't contain samples used for the training phase !!

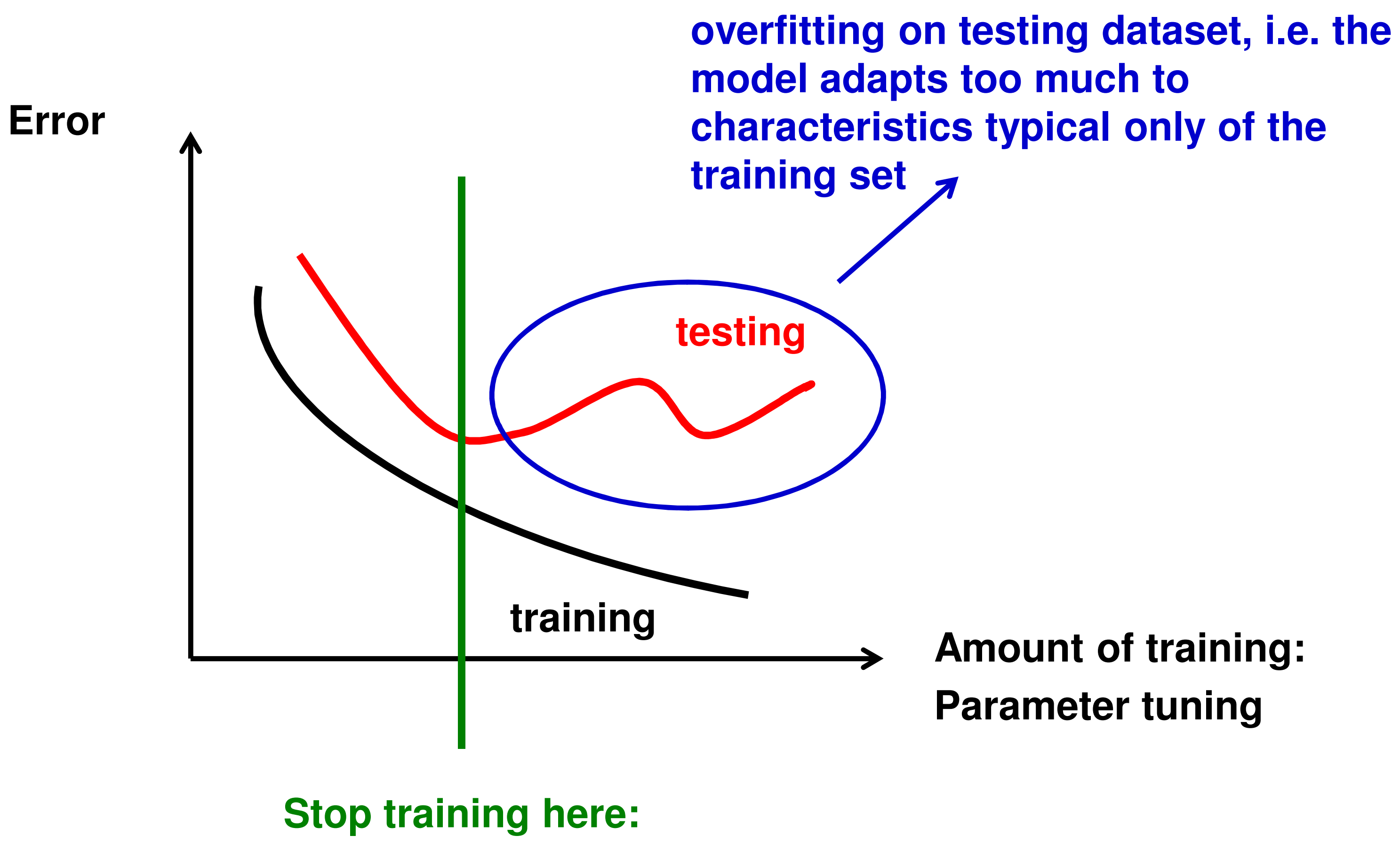
✓Some classification methods need parameters estimation in the training phase: the tuning of these parameters must be independent of the testing dataset !!

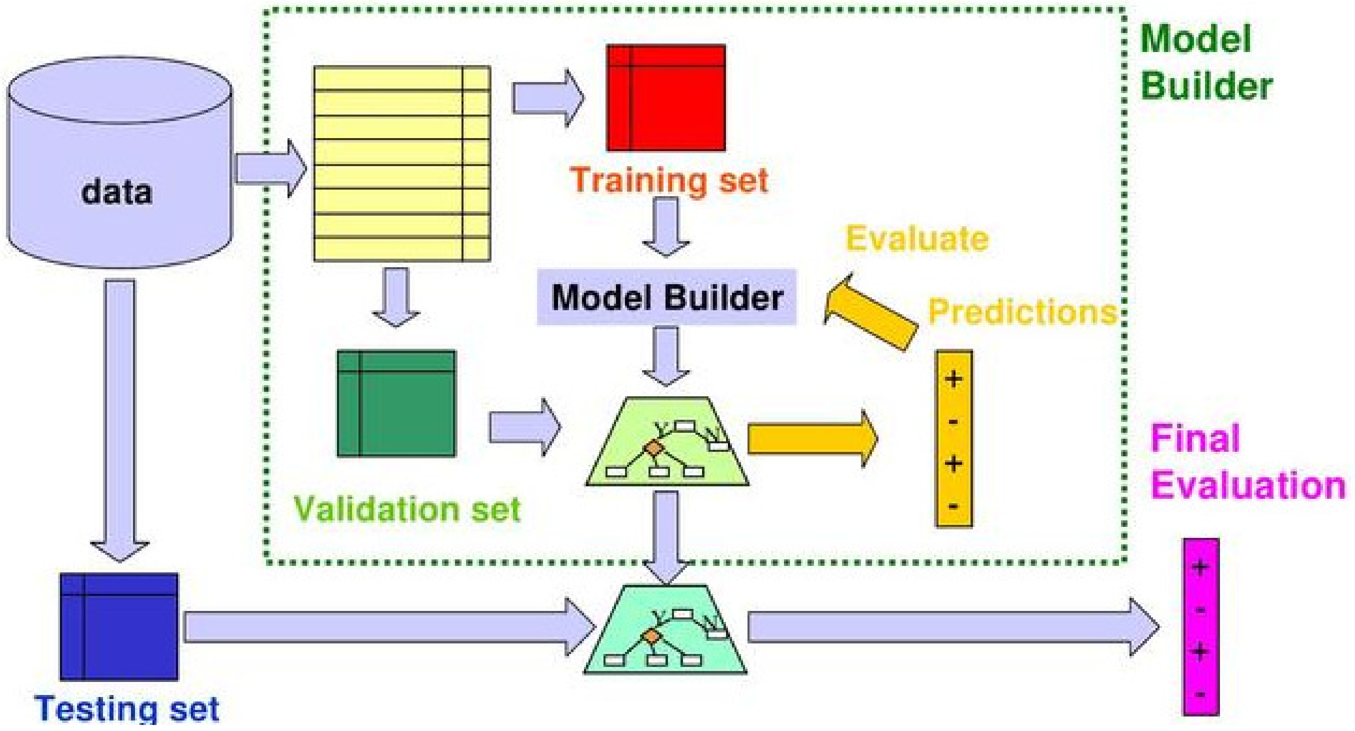
$$\Downarrow$$

$$\boxed{\text{Training dataset, Validation dataset, Testing dataset}}$$

$$\searrow$$

Parameters tuning


overfitting on testing dataset, i.e. the model adapts too much to characteristics typical only of the training set
Error
testing
training
Amount of training:
Parameter tuning
Stop training here:

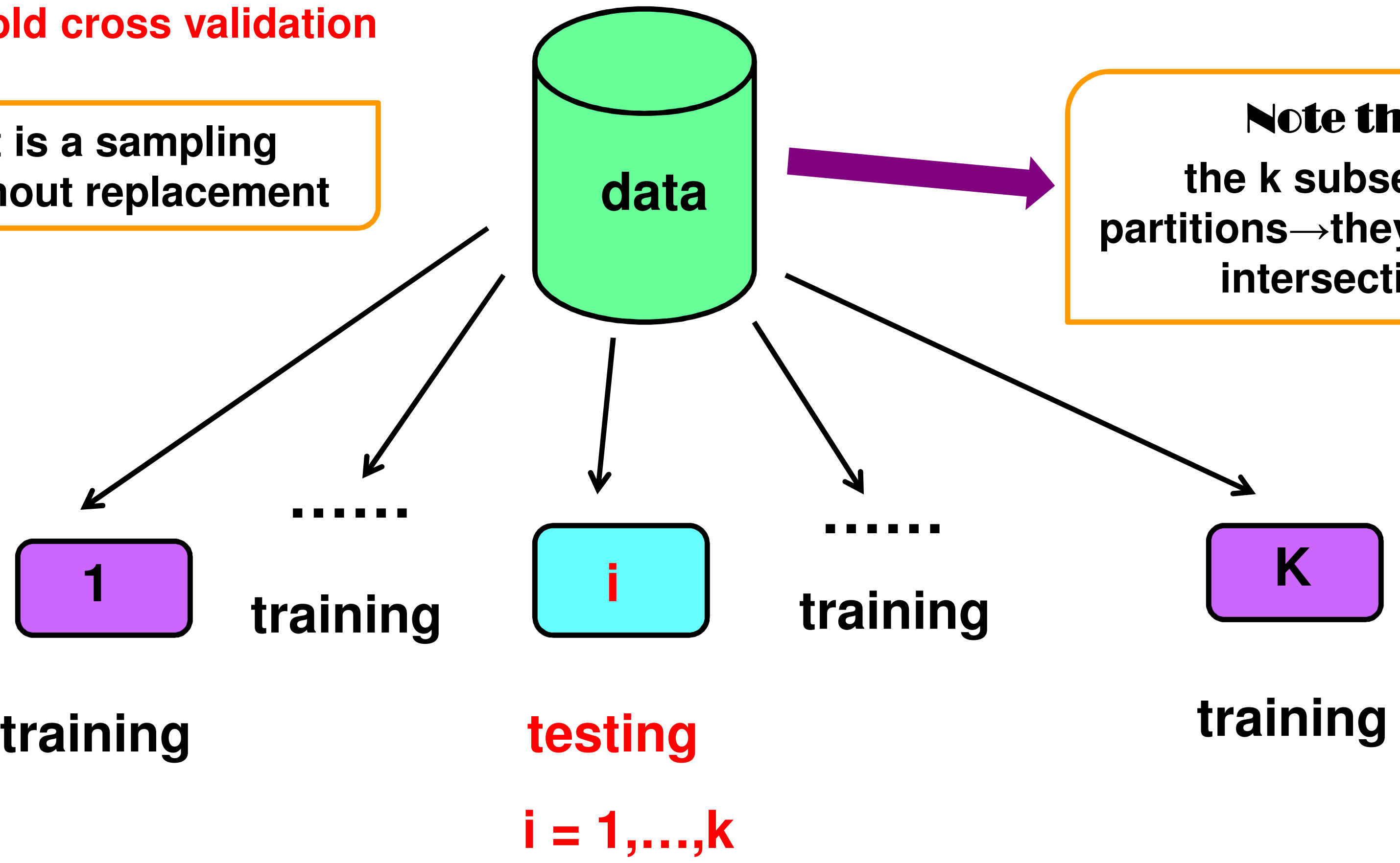


✓ repeated holdout → the method is repeated different times sampling randomly. The error obtained in each iteration is averaged with respect to the iterations to have a global accuracy measure. It is not optimal:

- the different testing set can have intersections
- the testing set varies at each iteration.

✓ **k-fold cross validation**

It is a sampling without replacement



Note that
the k subsets are partitions→they have not intersections

- The error obtained on the single folds is averaged. This method avoids the problem of the intersections of different testing sets.

- Moreover it is often used the stratified k-fold cross-validation, where the samples of the different classes within the single fold are balanced .

- if k = | $\mathcal{D}$ | → leave-one-out cross validation: better use of the dataset, no random sampling, unbiased, but with high variability and very expensive from a computational point of view.

- other common choices → k=5 or 10: more bias but less variability.

- to reduce variability in cross validation → repeated k-fold cross validation.

Correct use of cross validation in classification problems at high dimensionality

- Select the predictors having high correlation with class labels.

- Use the selected predictors to build a classifier

- Use the cross validation for parameters tuning and estimate of prediction error on the final model

Problem: the predictors have been chosen using all the dataset. When we leave the samples out after this selection we are not applying the classifier to a test set completely independent , because these predictors “have just seen” the samples left out.

- randomly divide the samples in K cross-validation folds.
- for each fold k = 1, 2, . . . ,K

Select the predictors having high correlation with class labels using all the samples except those in fold k

Use the selected predictors to build a classifier considering all the samples except those in fold k

Use the classifier to predict the class labels of samples in fold k.

Please note that

Only the unsupervised dimension reduction/variable selection methods, i.e. those that don’t use the information on class labels can be applied before classification !!!

✓Bootstrap → uses a sampling with replacement to build the training dataset.
Let us assume $|\mathcal{D}| = n$ and let us sample the dataset $\mathcal{D}$ n times with replacement

- from dataset $\mathcal{D}$ let us choose x randomly (but we don't remove x from $\mathcal{D}$)
- let $\mathcal{D}$*_train* = $\mathcal{D}$*_train* $\cup$ *x*
- let us repeat this process n times
- let us use $\mathcal{D}$*_train* as training set
- let the testing set be $\mathcal{D}$*_test* = $\{z \in D; z \notin \mathcal{D}\_train\}$

Note that

At each iteration: a sample has a probability of not being chosen for the training set= (1-1/n) → at the end of the sampling process the probability of a sample to be in the testing set is

$$\boxed{\left(1-\frac{1}{n}\right)^{n} \approx e^{-1} \approx 0.368}$$

$\mathcal{D}$*_train* (cardinality n) will contain about the 63.2% of the samples in $\mathcal{D}$, and a sample in $\mathcal{D}$ can have more than one occurrence in $\mathcal{D}$*_train* .

$\mathcal{D}$*_test* (cardinality < n) will contain about the 36.8% of samples in $\mathcal{D}$, and one sample in $\mathcal{D}$ can have a most one occurrence in $\mathcal{D}$*_test*.

Bootstrap is good for small dataset !!

Which classification algorithm?

No Free Lunch Theorem

There not exists a learning algorithm better than another one independent of the problem

↔

without a-priori knowledge there is no reason to prefer a classification method to another one.

⬇

The improvment of the performance depends on the use of a-priori information to adapt the procedures to the problems,

the theory and the algorithms alone are not sufficient, classification is an empirical object.

Conclusions

Things to consider when we choose and apply a classification algorithm for high dimensional data

- ***bias - variance* tradeoff**. Let us assume to have different training dataset.

A classification algorithm:

- is biased for a particular input if, when trained on each of these data sets, it is systematically incorrect when predicting the correct output for x.

- has high variance if it predicts different output values when trained on different training sets.

$$\text{Error} \approx \text{bias}^2 + \text{variance}.$$

A classification algorithm must be flexible (small bias) to ‘follow’ the data, but not too much, otherwise it will have high variability !!!

- **quantity of available training data with respect to the complexity of the ‘true’ classification function:**

  if it is simple the algorithm will need few data and no “flexibility” (high bias and small variance);

  if it is complex the algorithm will need large data and “flexibility” (small bias and high variance).

- Data heterogeneity
- Data redundancy (e.g. high correlated characteristics)
- Interactions and non linearities

But especially……..

The research of biomarkers significantly associated to the pathology under study must be guided from the current biological knowledge

(genes/proteins or pathogenetical pathways and their interactions)

$\Downarrow$

It is necessary to compare different classification/dimension reduction algorithms to determine experimentally what is the ‘best’ and not to use them as a black box !!!!

References

Books

- G.J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, New York: John Wiley & sons, 1992.
- R.O. Duda, P.E. Hart, and D.G. Stork, Pattern Classification, New York: John Wiley & Sons, 2001
- T. Hastie, R. Tibshirani, J. H. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer Series in Statistics, 2009.

Papers

- Hand D.J. and Till R.J. (2001) A simple generalisation of the area under the ROC curve for multiple class classification problems. *Machine Learning*, 45, 171-186.
- Competition on Clinical Mass Spectrometry Based Proteomic Diagnosis, Statistical Applications in Genetics and Molecular Biology, Volume 7, Issue 2, Gennaio 2008.

Web pages

- Trevor Hastie: http://www.stanford.edu/~hastie/
- Robert Tibshirani: http://www-stat.stanford.edu/~tibs/
- Jerome Friedman: http://www-stat.stanford.edu/~jhf/
- Geoff McLachlan: http://www.maths.uq.edu.au/~gjm/

Inter@mics
Flagship Project

1-year Research Grant (renewable)

- **Job Description:** Development statistical methods for the analysis of "-Omics" data

- **Location:** Istituto per le Applicazioni del Calcolo "Mauro Picone", Napoli, Italy

- **Number of positions available:** 1

- **Call** (bando):http://www.na.iac.cnr.it/assegni/IAC-003-2012-NA.PDF

- **Deadline for Application:** 28 September 2012

- **Interview** : 8 October 2012