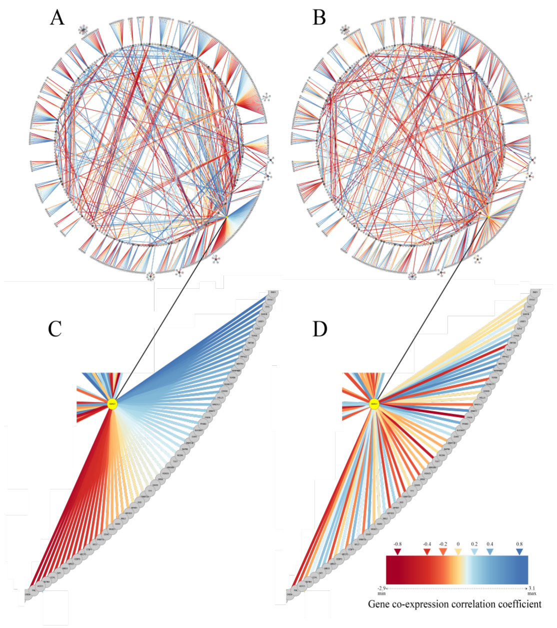
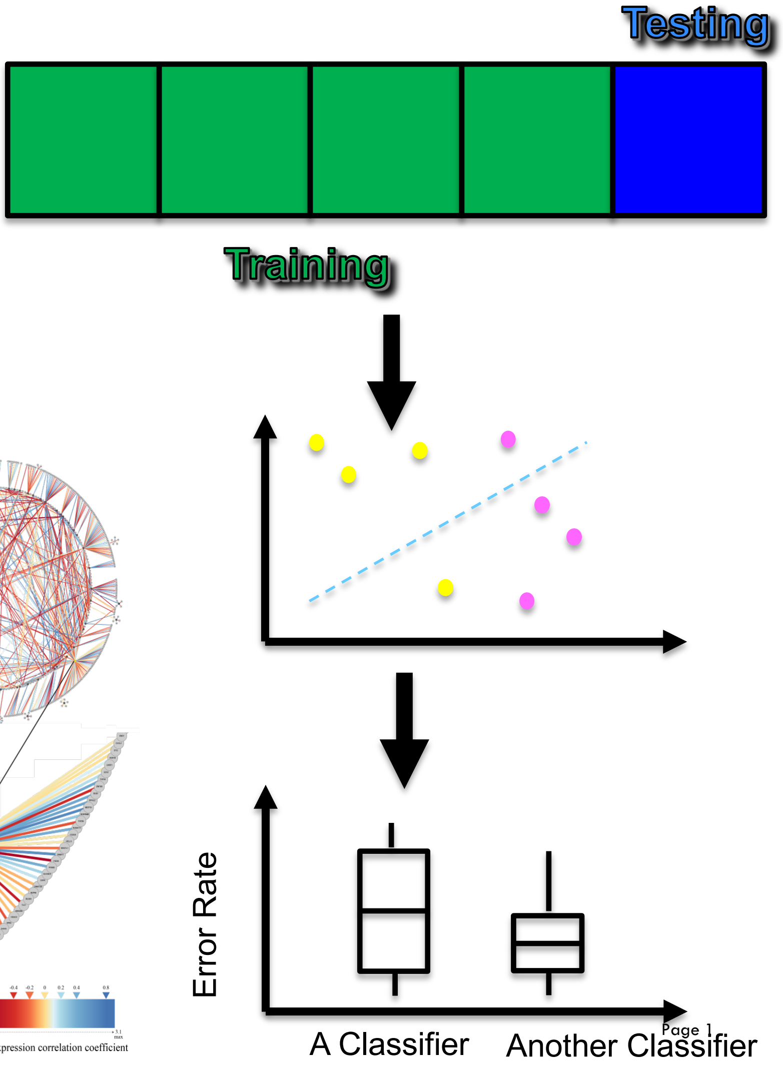
Introduction to Classification

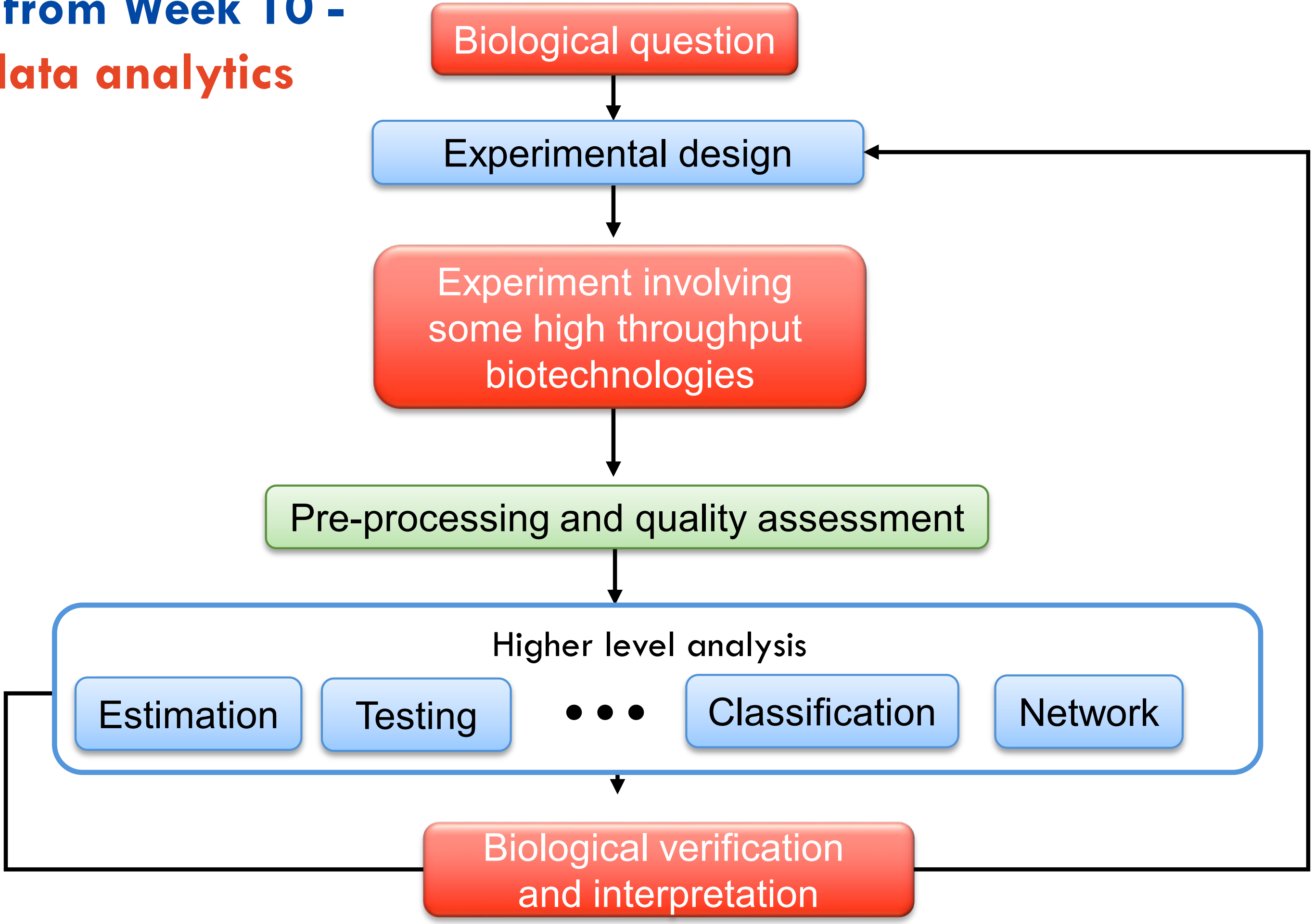
Omics analysis – cross validation

AMED3002

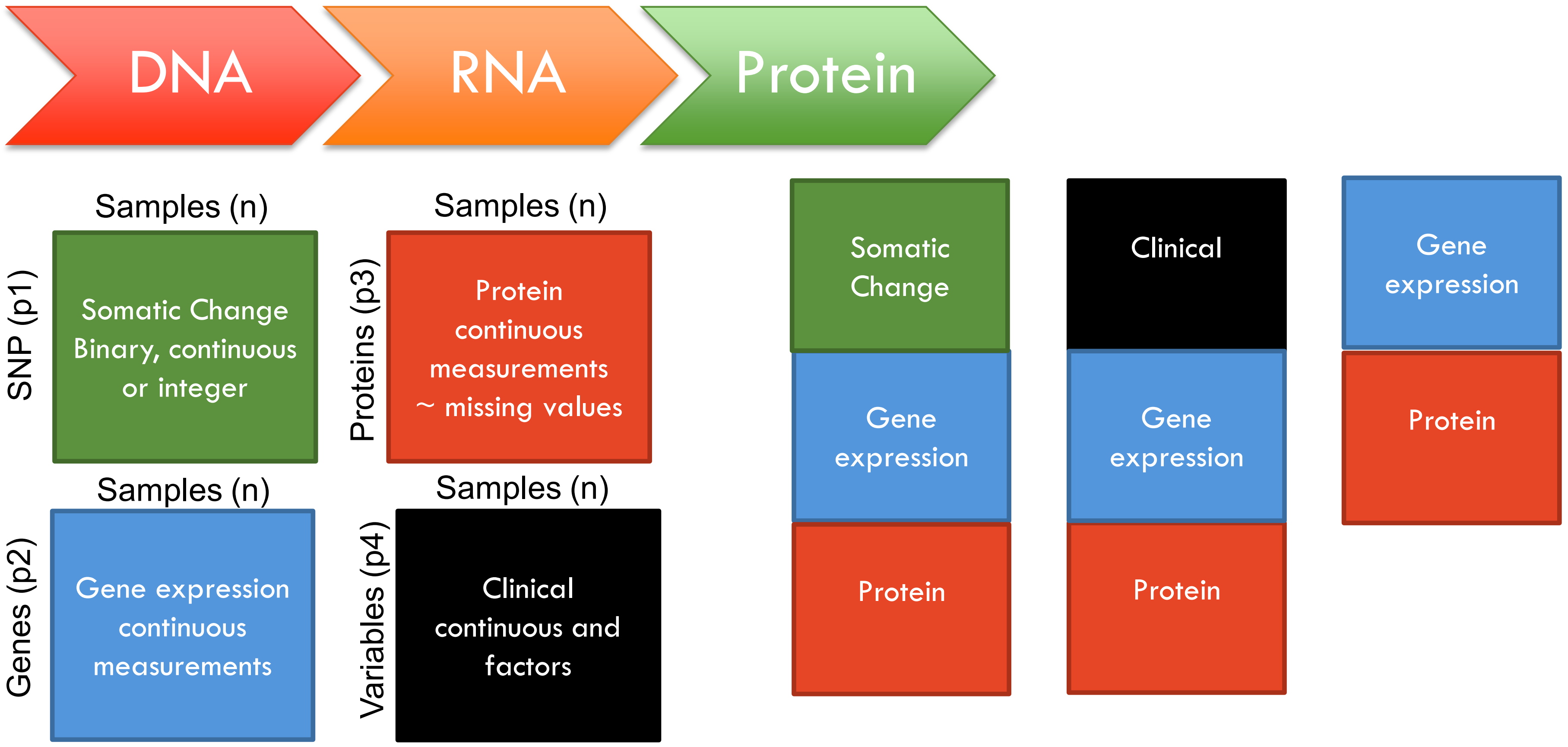




Review from Week 10 - Omics data analytics


Biological question
Experimental design
Experiment involving some high throughput biotechnologies
Pre-processing and quality assessment
Higher level analysis
Estimation
Testing
• • •
Classification
Network
Biological verification and interpretation

Possible input data?

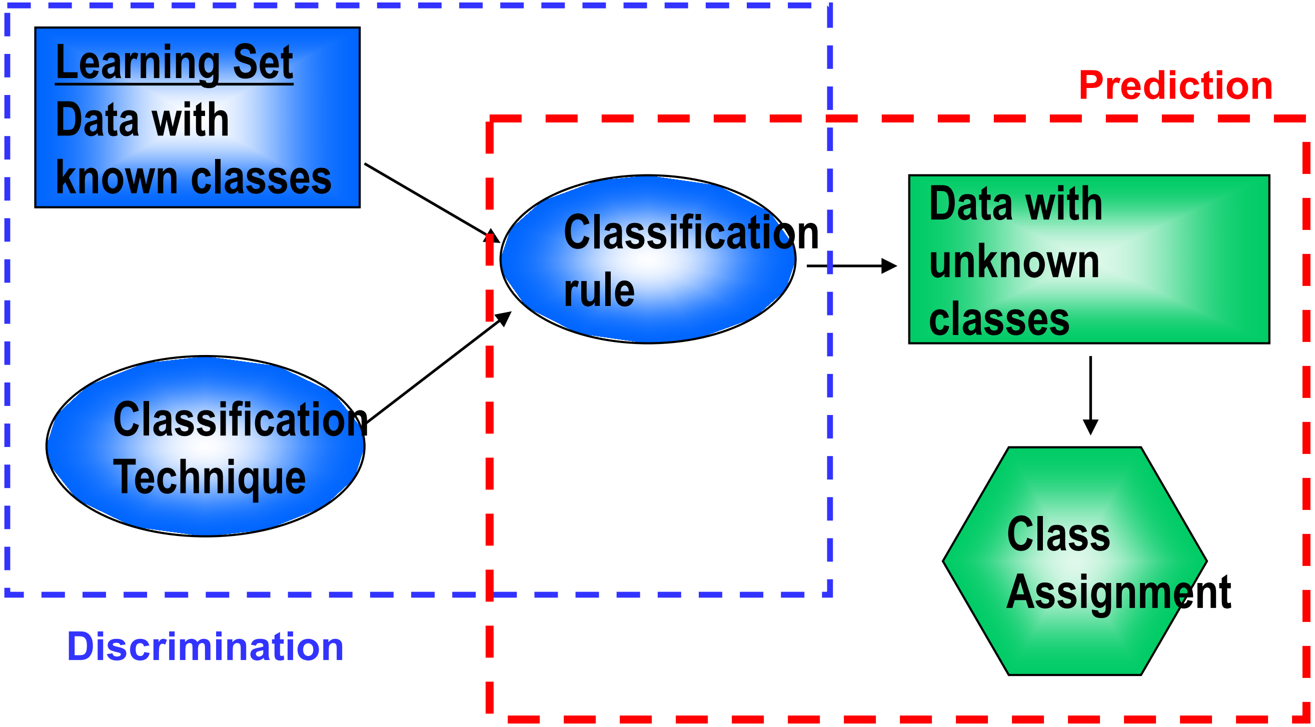


Clustering and discrimination

- **Unsupervised:** classes unknown, want to discover them from the data (cluster analysis)

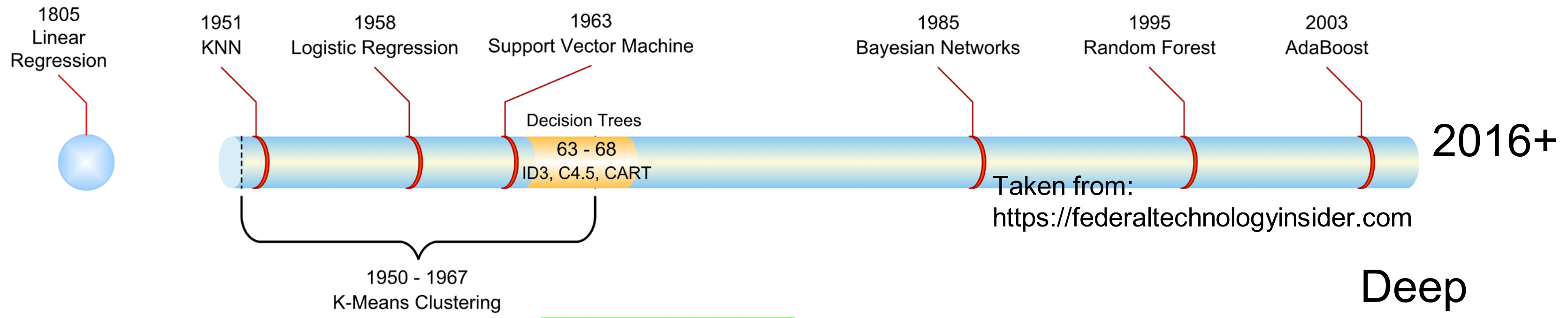
- **Supervised:** classes are predefined, want to use a (training or learning) set of labeled objects to form a classifier for classification of future observations

Classification – supervise learning



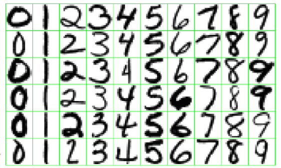
History

Time Line of Machine Learning Algorithms



Taken from:
https://federaltechnologyinsider.com

Deep learning



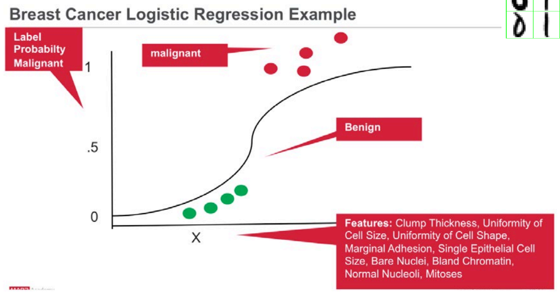
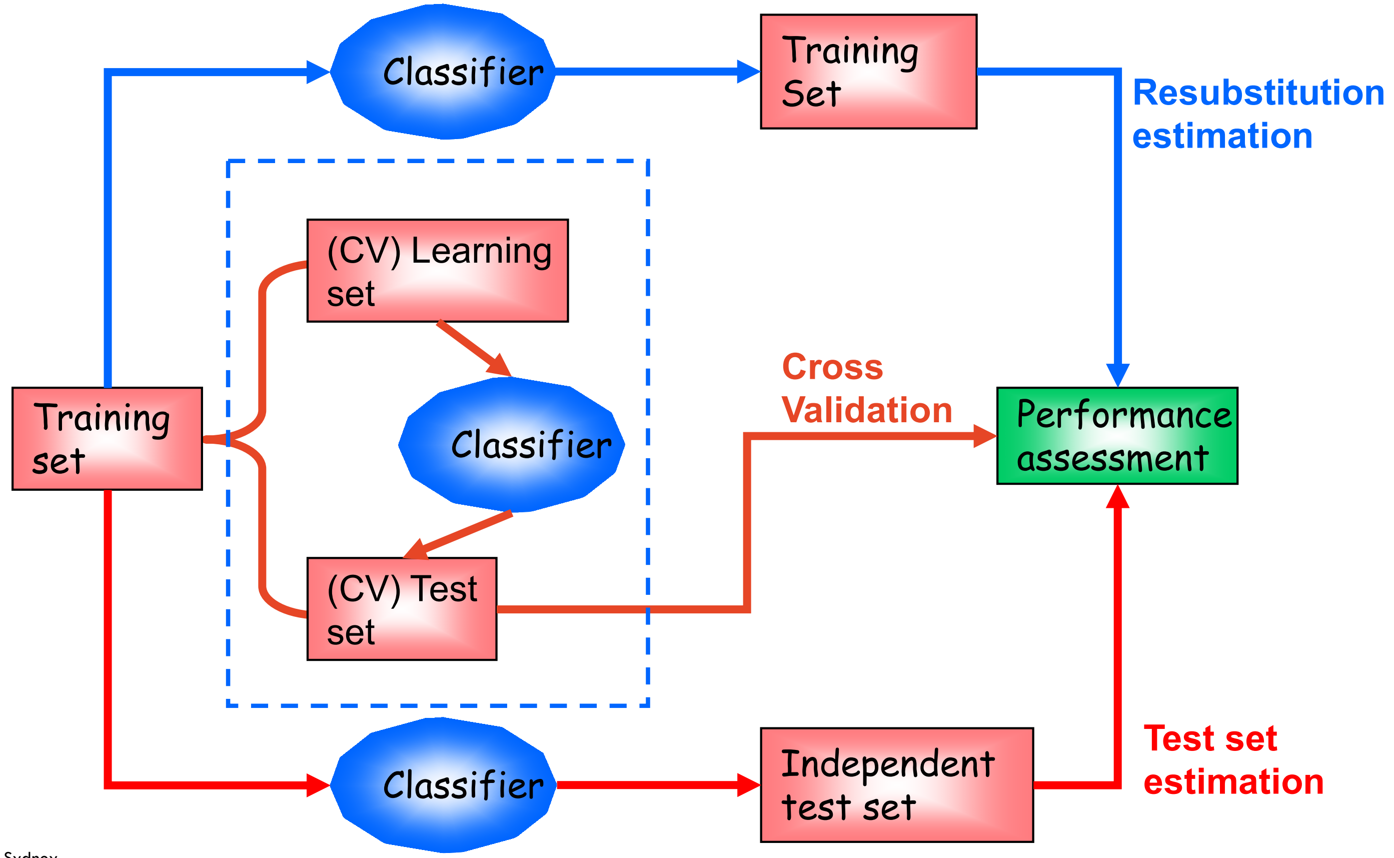

Breast Cancer Logistic Regression Example
Label Probabilty Malignant
malignant
Benign
1
.5
0
X
Features: Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli, Mitoses

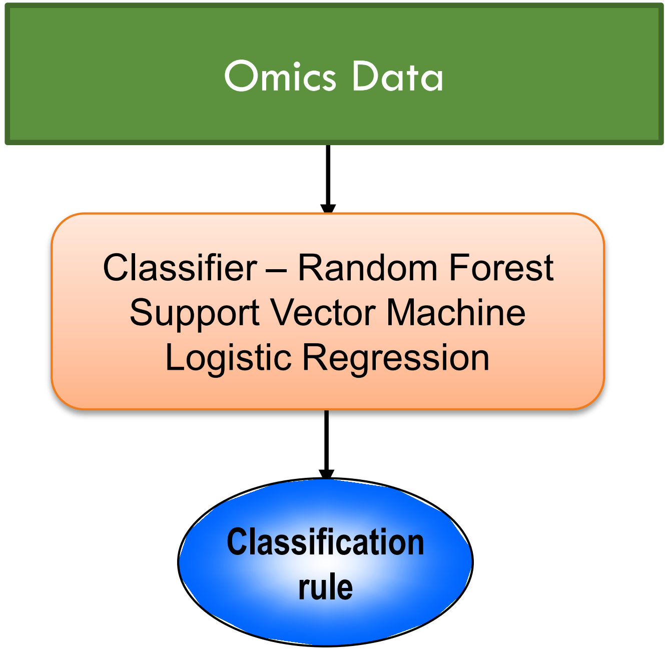
Diagram of performance assessment



Machine learning cures cancer… really?

THE UNIVERSITY OF SYDNEY

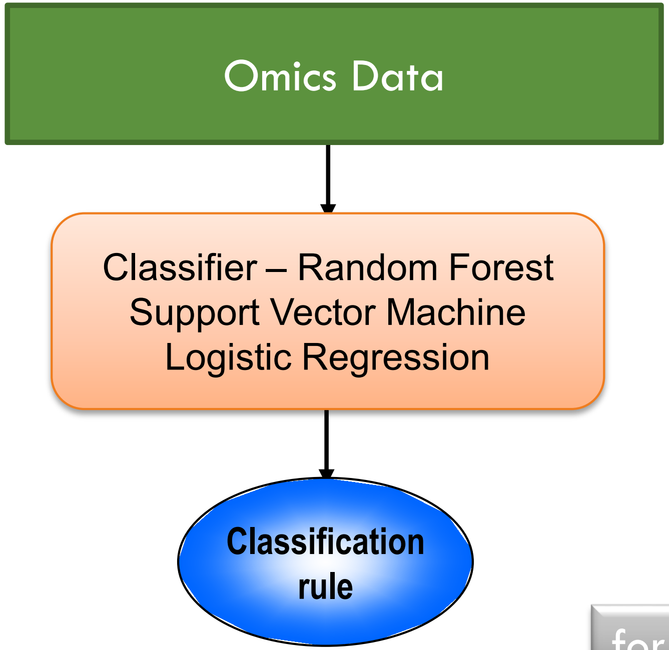
Supervised learning in practice

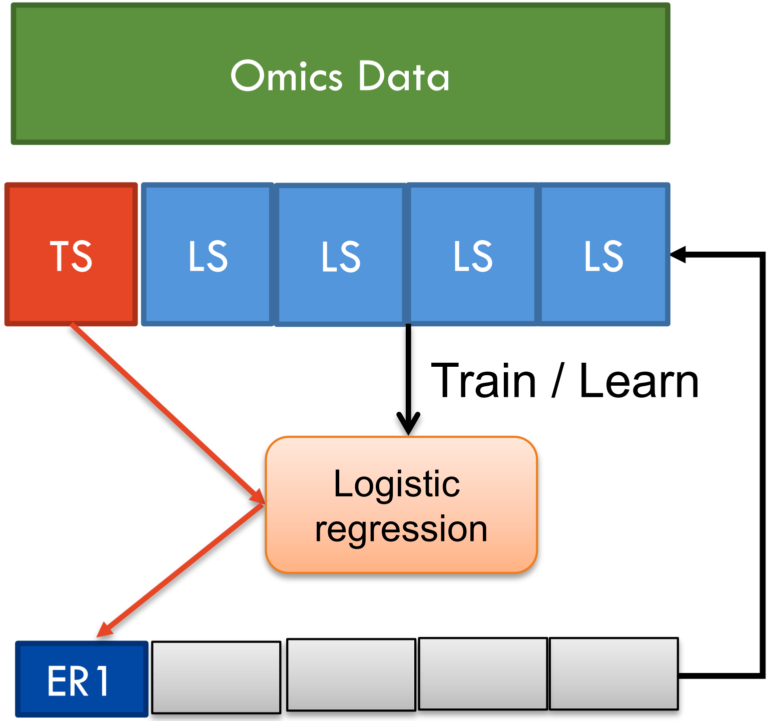


How good is my classifier ?

How good is my model ?

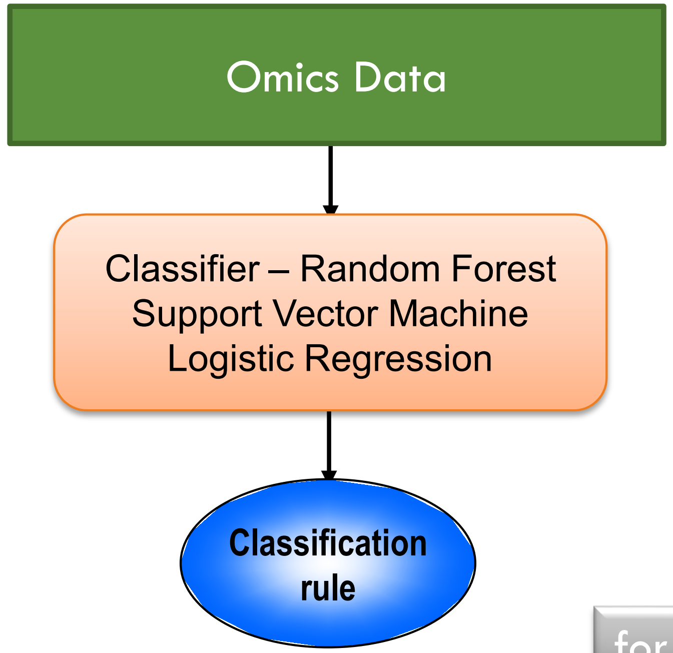
5-fold CV



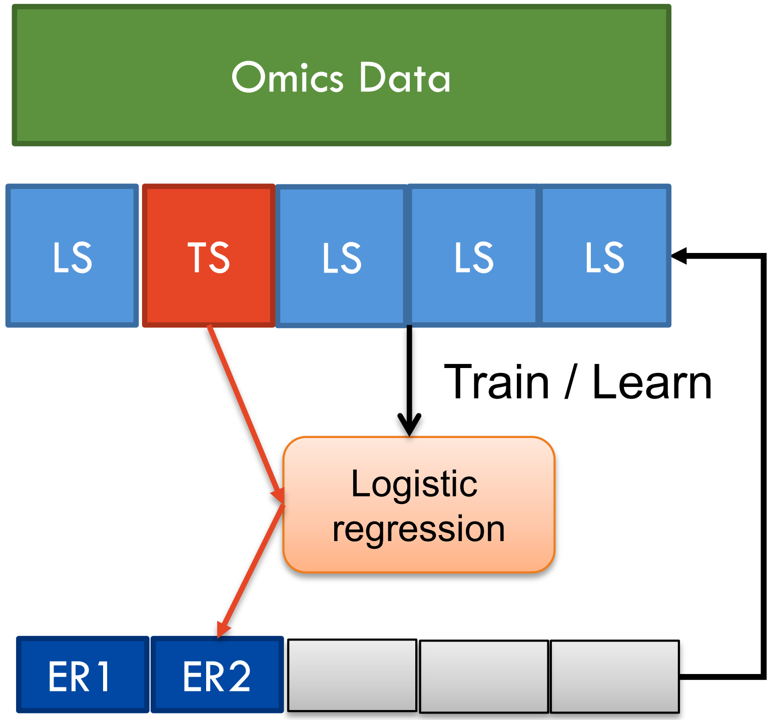


for (i in 1:5) { CV }
i = 1

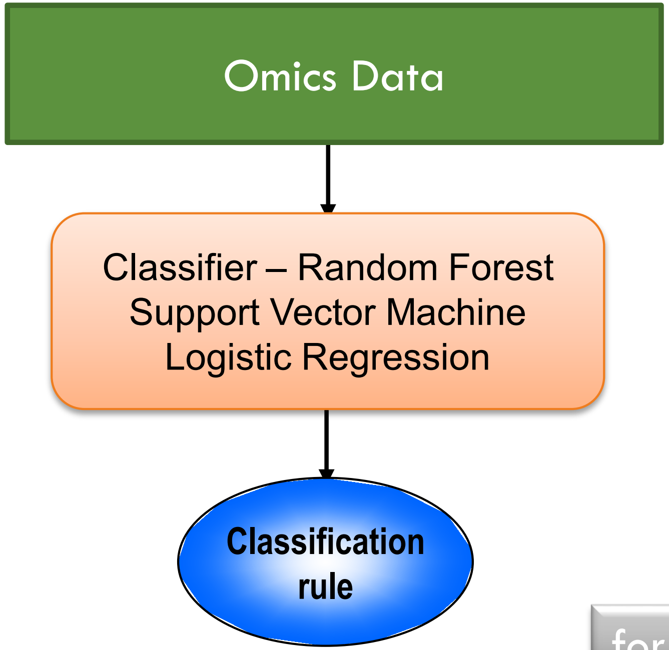
5-fold CV

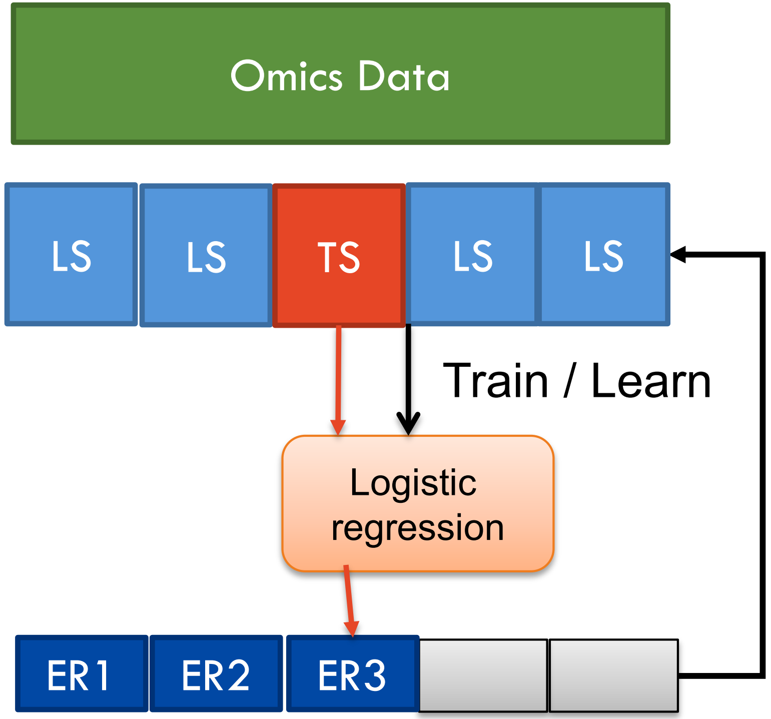


```
for (i in 1:5) { CV }
```
i = **2**



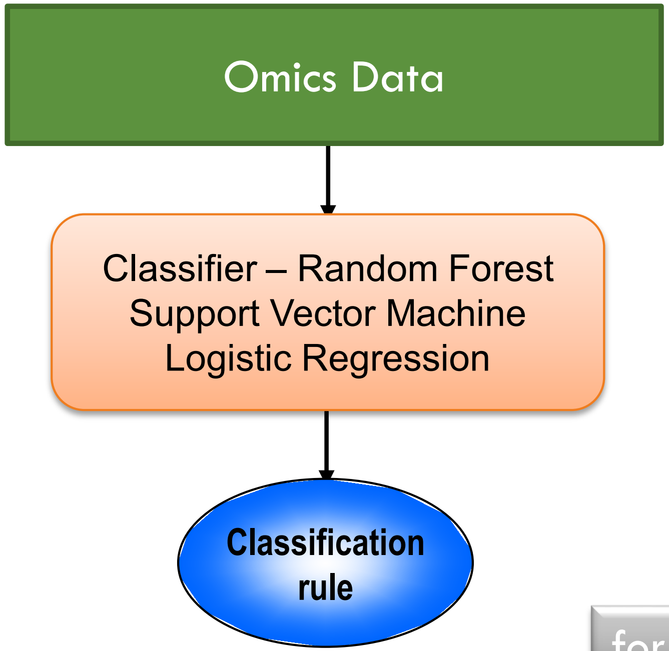
5-fold CV



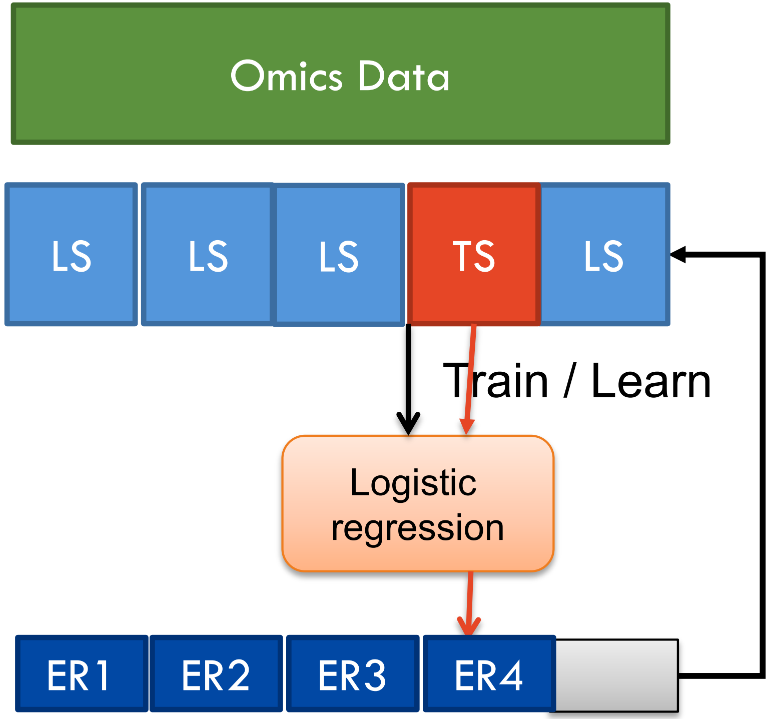


for (i in 1:5) { CV }
i = 3

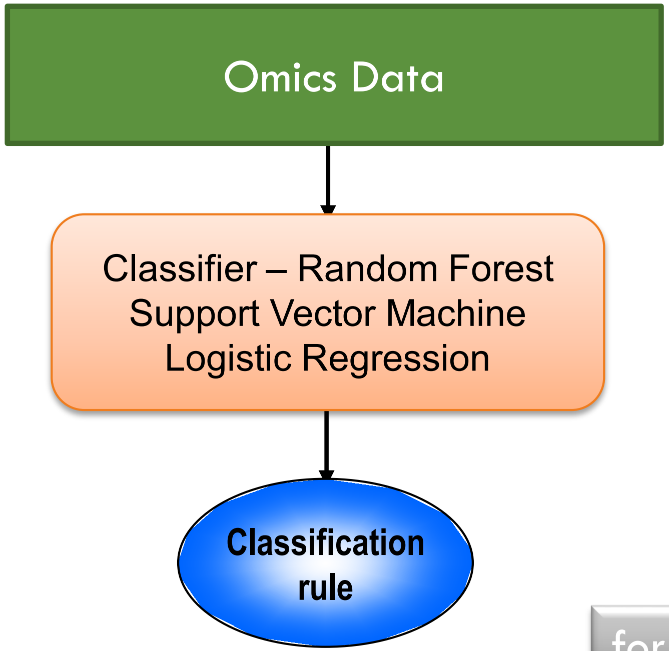
5-fold CV



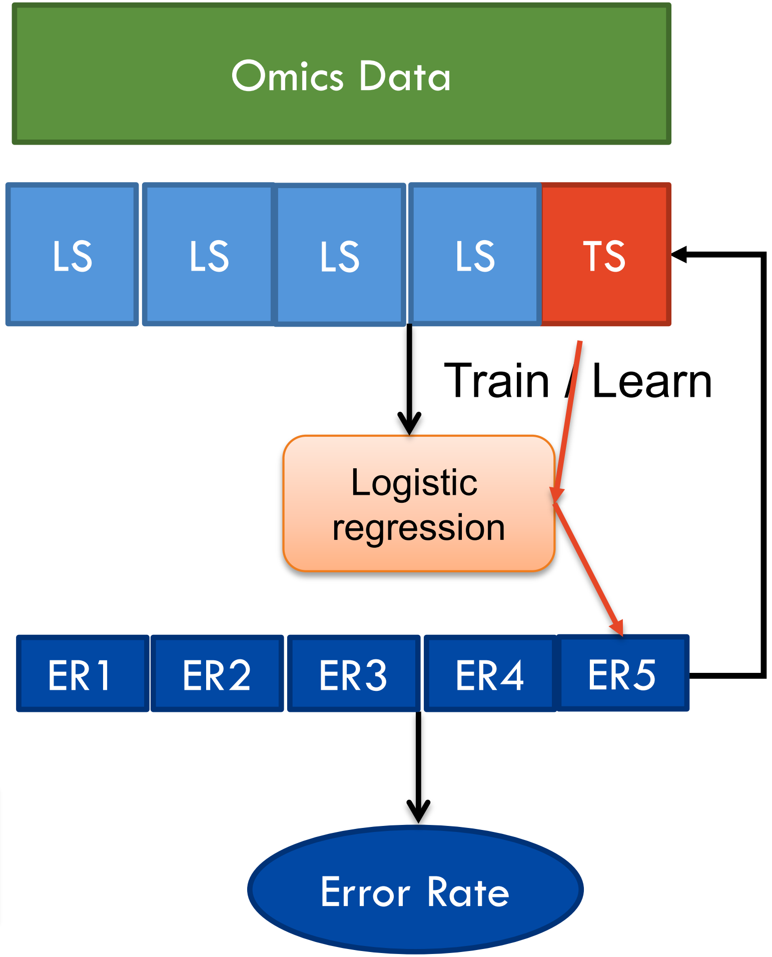
for (i in 1:5) { CV }
i = 4



5-fold CV

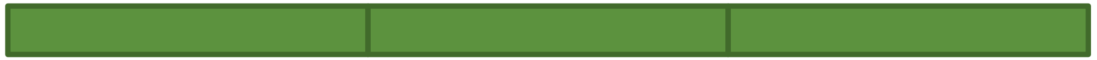


for (i in 1:5) { CV }
i = 5

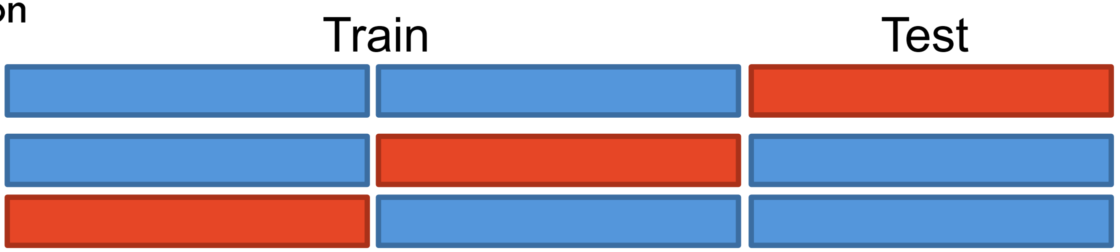


Common Splitting Strategies

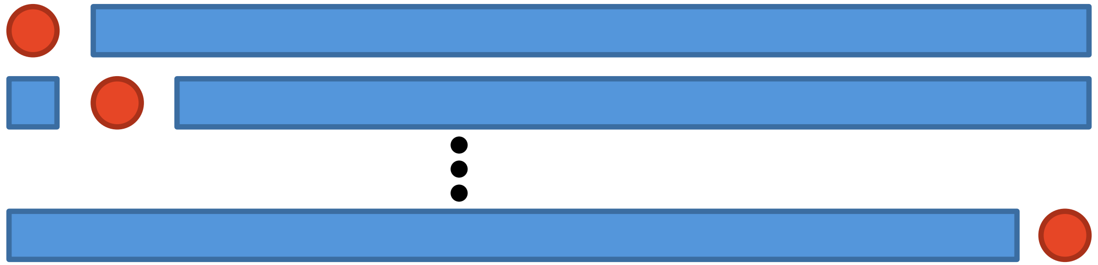
Dataset



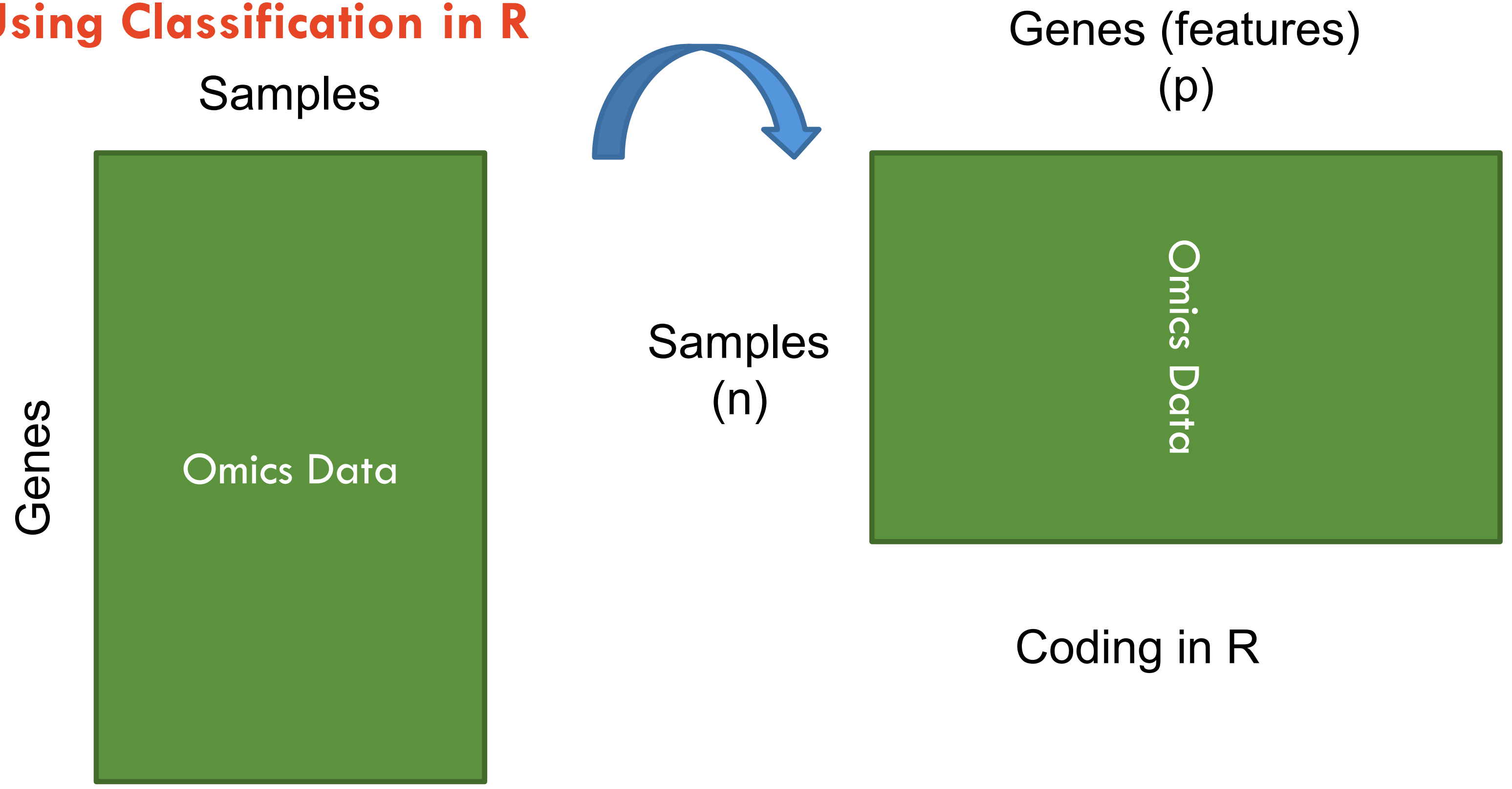
k-fold cross-validation



Leave-one-out (n-fold cross validation)



Using Classification in R



Coding in R

A little bit more on classifier

Maximum likelihood (ML) discriminant rule

Consider the extremely rare situation where the exact distributions of the populations are known.

For known class conditional densities $\mathbf{p}_g(x) = \mathbf{p}(x \,|\, \mathbf{Y} = \mathbf{g})$, the ML rule predicts the class of an observation x by that which gives the largest likelihood to

$$x: \mathbf{C}(x) = \mathrm{argmax}_g\, \mathbf{p}_g(x).$$

That is, if we write the p.d.f. of the gth population as $\mathbf{Lg}(\mathbf{x})$, than the ML rule says that one should allocate **x** to population **g** where

$$\mathbf{L}_g(\mathbf{x}) = \mathbf{max}_i \mathbf{L}_i(\mathbf{x}):$$

Special case when features have Gaussian (Normal) distribution.

Special case when G=2 (two classes)

For G = 2 classes, ML discriminant rule becomes:

– allocate x to population 1,$\mathbf{C}(x) = 1$ if $\log L_1(x) > \log L_2(x)$ and

– allocate x to population 2,$\mathbf{C}(x) = 1$ if $\log L_1(x) < \log L_2(x)$.

ML discriminant rules – special cases

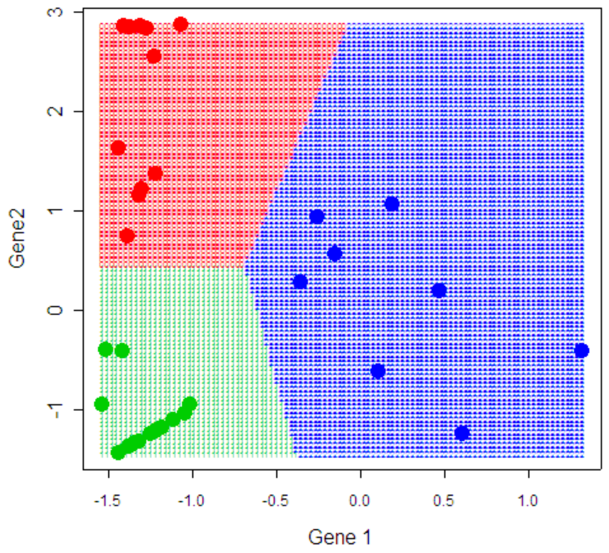
Gaussian ML discriminant rules

For multivariate Gaussian (normal) class densities $\mathbf{X}|Y = k \sim N(\mu_k, \Sigma_k)$, the ML classifier is

$$C(\mathbf{X}) = \mathrm{argmin}_k \{(\mathbf{X} - \mu_k)\, \Sigma_k^{-1} (\mathbf{X} - \mu_k)' + \log|\Sigma_k|\}$$

In general, this is a quadratic rule (Quadratic discriminant analysis, or QDA)

In practice, population mean vectors μ_k and covariance matrices Σ_k are estimated by corresponding sample quantities

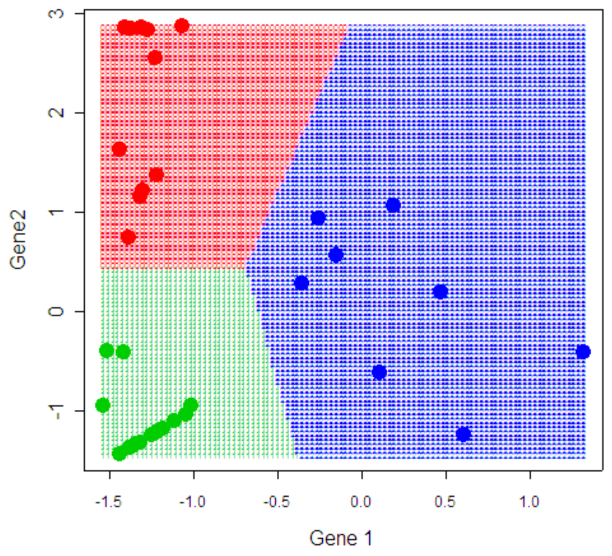


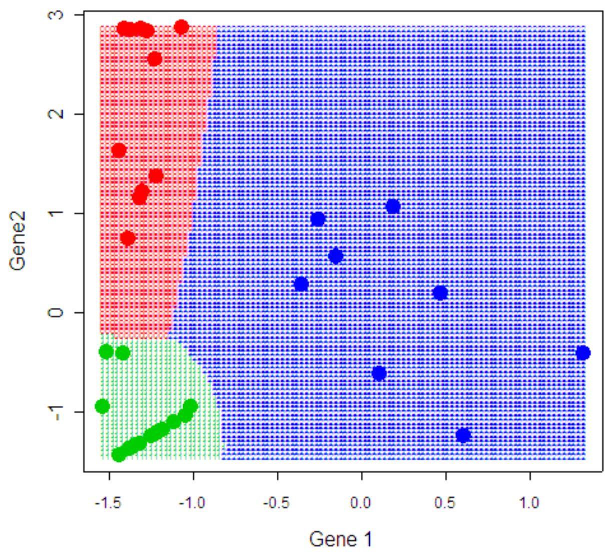
[DLDA]

Diagonal linear discriminant analysis

class densities have the same diagonal covariance matrix $\nabla = \mathrm{diag}(s_1^2, \ldots, s_p^2)$

ML discriminant rules - special cases





[DLDA]
Diagonal linear discriminant analysis
class densities have the same diagonal covariance matrix ∇= diag(s_1^2, …, s_p^2)

[DQDA]
Diagonal quadratic discriminant analysis)
class densities have different diagonal covariance matrix $\nabla_{\mathbf{k}}$= diag(s_{1k}^2, …, s_{pk}^2)

ML and LDA

The linear discriminant rules described above are classical and widely used classification tools.

- Simple and intuitive: the predicted class of a test case is the class with the closest mean (using the Mahalanobis metric).
- Easy to implement: the partition has linear boundaries.

However, LDA has a number of obvious limitations:

- Linear or even quadratic discriminant boundaries may not be flexible enough.
- Features may have mixture distributions within classes.
- In the case of too many features, performance may degrade rapidly due to over parameterization and high variance of parameter estimates.

Classification tree

Binary tree structured classifiers are constructed by repeated splits of subsets (nodes) of the measurement space X into two descendant subsets, starting with X itself. Each terminal subset is assigned a class label and the resulting partition of X corresponds to the classier.

Three main aspects of tree construction:

(i) the selection of the splits;

(ii) the decision to declare a node terminal or to continue splitting;

(iii) the assignment of each terminal node to a class.

Different tree classifiers use different approaches to deal with these three issues. Here, we use CART **Classification And Regression Trees** of Breiman et al. (1984).

Classification trees

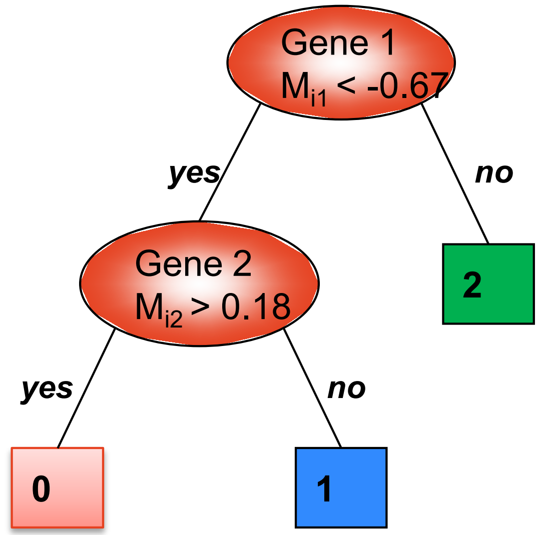
1. Splitting rule. At each node, choose the split that maximizes the decrease in impurity.

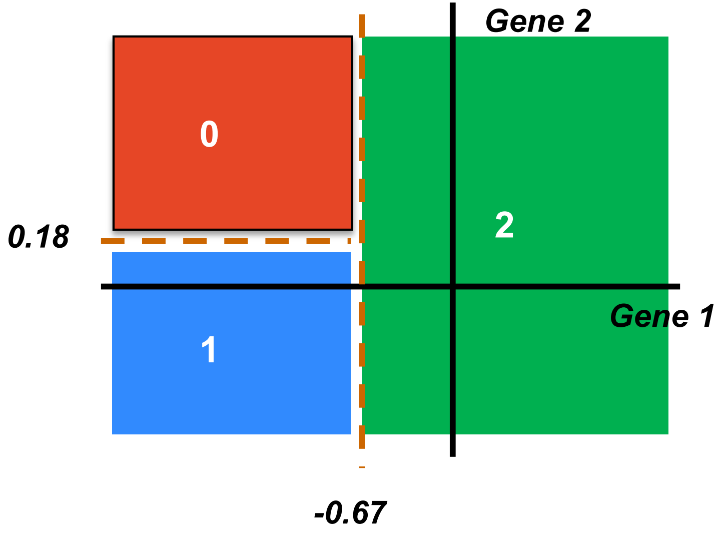
E.g. of impurity functions:
Gini index $\phi(p_1, \ldots, p_G) = \sum_{k \neq l} p_k p_l = 1 - \sum_k p_k^2$,
entropy, and twoing rule.

2. Split stopping rule. Grow a large tree, selectively **prune** the tree upward, getting a decreasing sequence of subtrees. Use **cross validation** to identify the subtree having the lowest estimated misclassification rate.

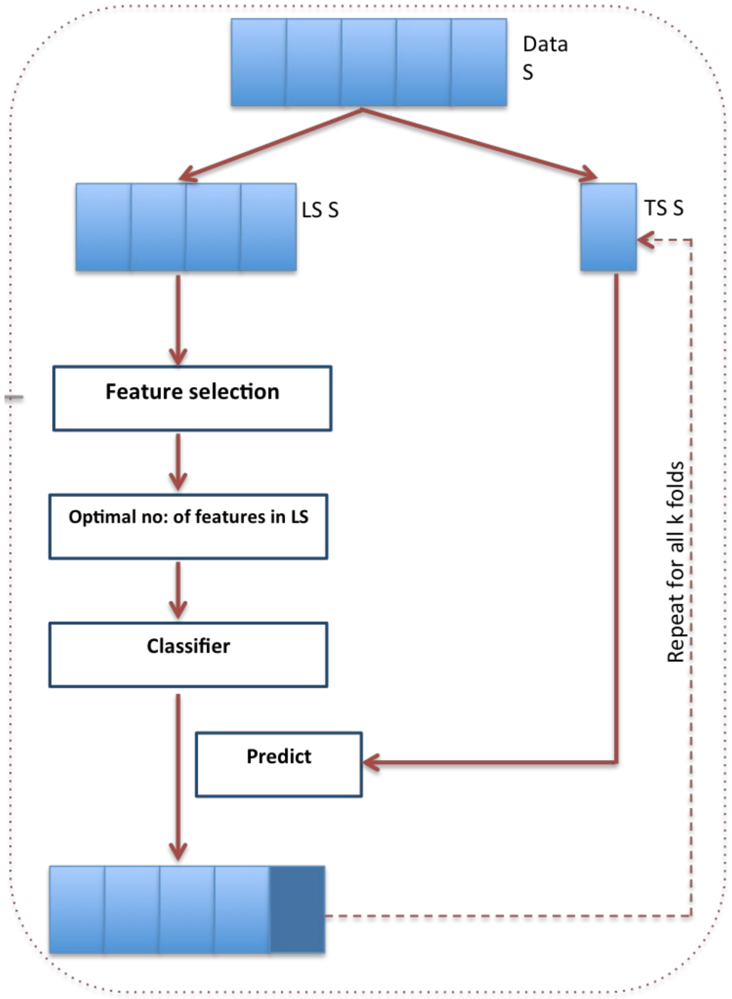
3. Class assignment rule. For each terminal node, choose the class that minimizes the resubstitution estimate of the misclassification probability, given that a case falls into this node.

Classification tree

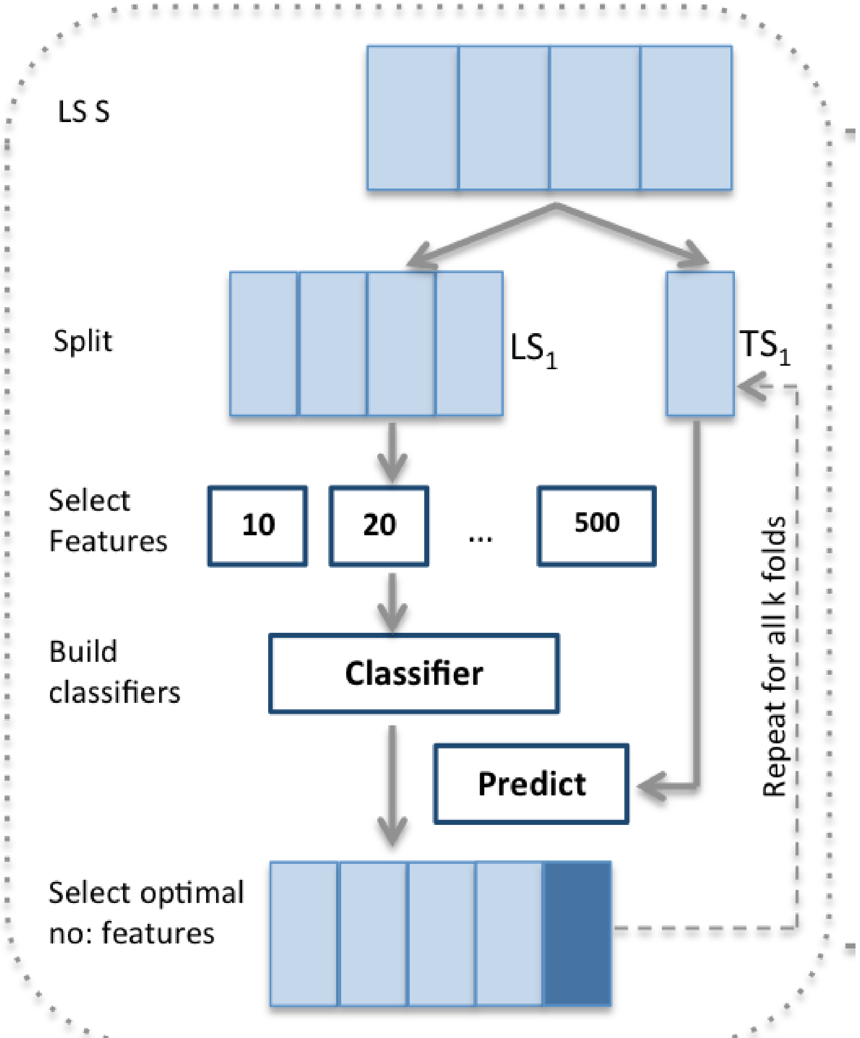




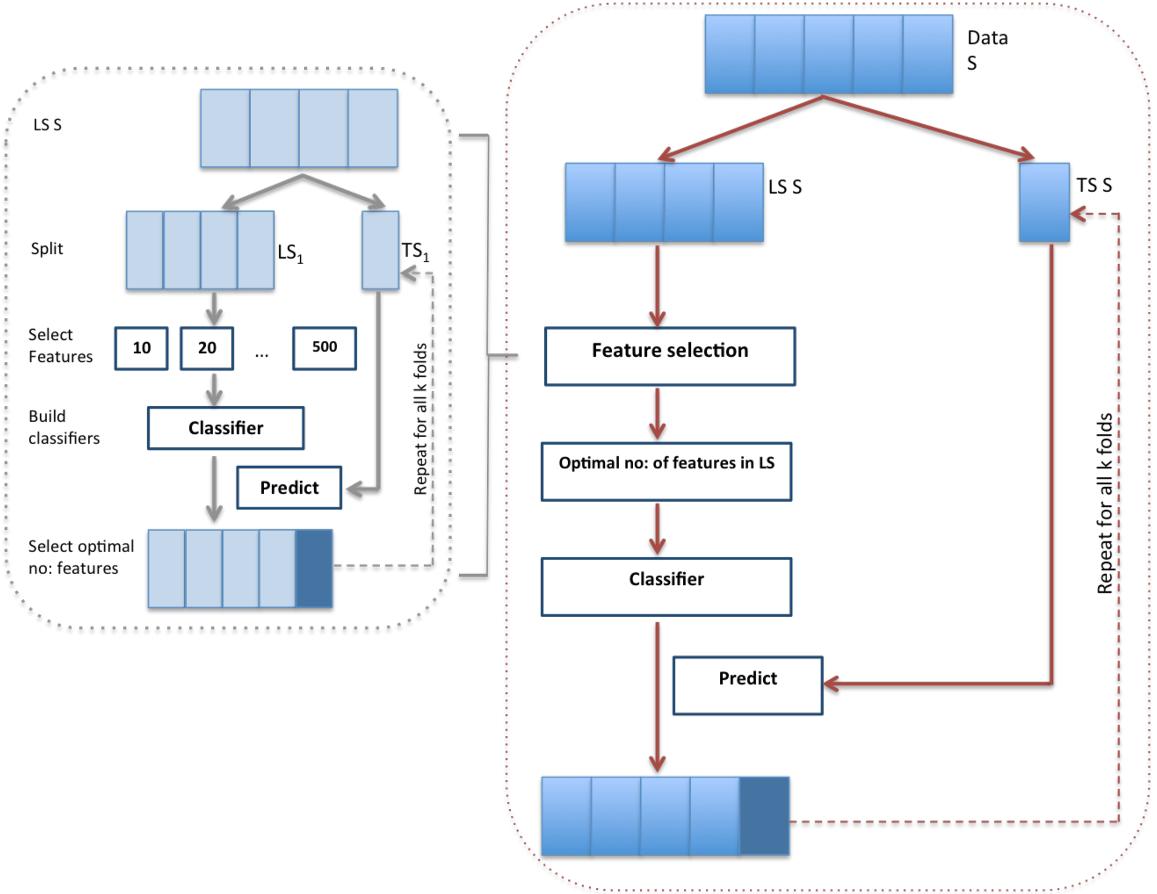
Draw your CV loop



Parameter selection



Full CV


LS S
Split
LS1
TS1
Select Features
10
20
...
500
Build classifiers
Classifier
Predict
Select optimal no: features
Repeat for all k folds
Data S
LS S
TS S
Feature selection
Optimal no: of features in LS
Classifier
Predict
Repeat for all k folds