Pattern recognition

3. Feature selection and extraction

Lodewyk Wessels (*The Netherlands Cancer Institute*)
Marcel Reinders (*Delft University of Technology*)
Perry Moerland (UAMC, *University of Amsterdam*)

Some material courtesy of Robert Duin and David Tax

Overview

- Feature extraction
- Feature selection
- Regularized classifiers

Overview

- **Feature extraction**
  - Linear:
    - PCA
    - Fisher
  - Non-linear
    - MDS (Multi-dimensional scaling)

Overview

- **Feature selection**
  - Criteria
  - search algorithms
    - Forward selection
    - Backward selection
    - Branch & Bound search

Overview

- **Regularized classifiers**
  - PAM (Prediction Analysis of Micro-arrays = shrunken centroids)
  - Ridge regression
  - LASSO (Least Absolute Shrinkage and Selection Operator)

BioSB

Dimensionality reduction

Aim of Feature Extraction and Selection: reduce dimensionality

Dimensionality reduction

Aim of Feature Extraction and Selection: reduce dimensionality

Why is reducing dimensionality useful?

Dimensionality reduction

Aim of Feature Extraction and Selection: reduce dimensionality

Why is reducing dimensionality useful?

1. **Fewer parameters**: faster, easier to estimate – possibly better performance
2. **Explain** which measurements (features) are useful and which are not (reduce redundancy)
3. **Visualisation**

Molecular data (e.g. RNAseq data)

- **Curse of dimensionality (# features / # samples):**
  - for **fixed** sample size
  - and **increasing** number of features (number of parameters)
  - performance **decreases**
  - (There are fewer samples per parameter, i.e. worse estimates)

BioSB

Molecular data (e.g. RNAseq data)

- **Curse of dimensionality (# features / # samples):**
  - for **fixed** sample size
  - and **increasing** number of features* (number of parameters)
  - performance **decreases**
  - (There are fewer samples per parameter, i.e. worse estimates)
- **Traditional assumption in pattern recognition:**
  - need 5-10 times as many samples as there are parameters
  - with regularization we can do with fewer

* measurements

BioSB

Molecular data (e.g. RNAseq data)

- **Curse of dimensionality (# features / # samples):**
  - for **fixed** sample size
  - and **increasing** number of features (number of parameters)
  - performance **decreases**
  - (There are fewer samples per parameter, i.e. worse estimates)
- **Traditional assumption in pattern recognition:**
  - need 5-10 times as many samples as there are parameters
  - with regularization we can do with fewer
- **But genomic data (e.g. RNAseq) is extreme:**
  - 100-1000 times *fewer* samples than parameters!

* measurements

BioSB

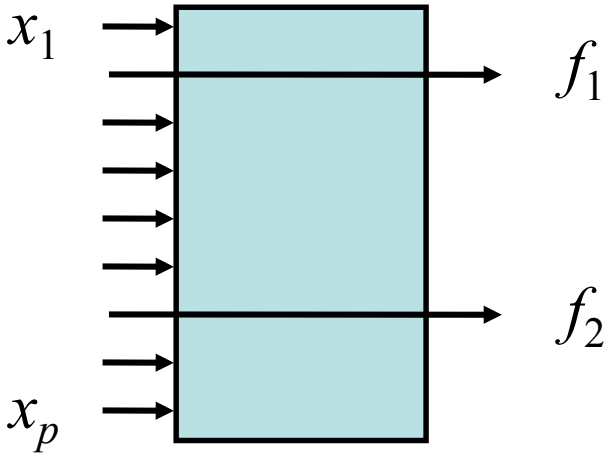
Molecular data (e.g. RNAseq data)

- **Curse of dimensionality (# features / # samples):**
  - for **fixed** sample size
  - and **increasing** number of features (number of parameters)
  - performance **decreases**
  - (There are fewer samples per parameter, i.e. worse estimates)
- **Traditional assumption in pattern recognition:**
  - need 5-10 times as many samples as there are parameters
  - with regularization we can do with fewer
- **But genomic data (e.g. RNAseq) is extreme:**
  - 100-1000 times *fewer* samples than parameters!
- **For example: nearest mean classifier on Golub data**
  - $p = 3051$, $k = 2 \rightarrow$ number of parameters = 6102
  - Number of samples, $n = 38$

* measurements

Feature selection vs. extraction

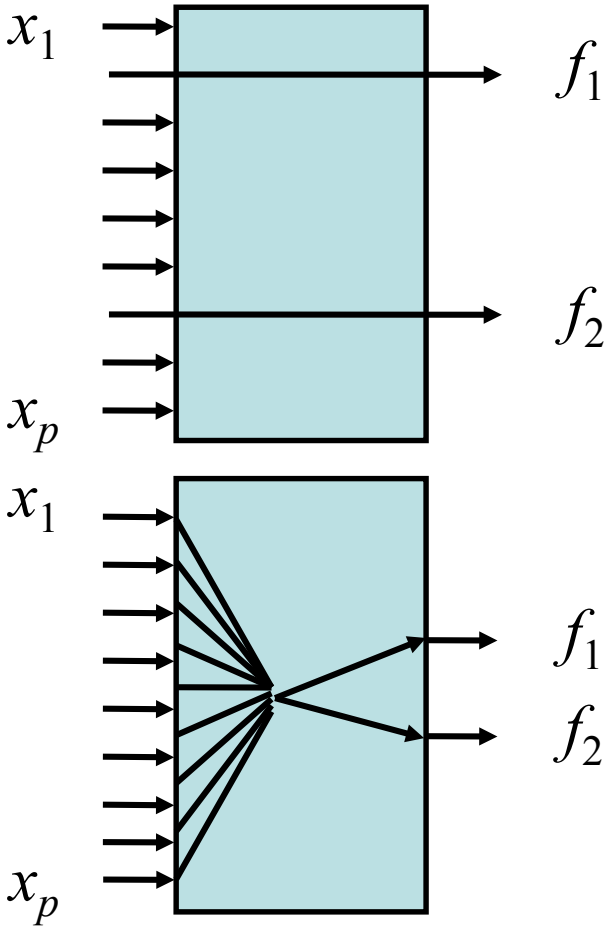
- **Feature selection:**
  select d out of p features



Feature selection vs. extraction

- **Feature selection:**
  select d out of p features

- **Feature extraction:**
  map p features
  to d features
  (e.g. PCA)



Feature selection v extraction (2)

	Advantage	Disadvantage
Selection	cut in features	expensive
	easy interpretation	often approximate

Feature selection v extraction (2)

	Advantage	Disadvantage
Selection	cut in features easy interpretation	expensive often approximate
Extraction	cheap can be nonlinear not axis aligned	need all features criterion sub-optimal

Feature extraction (2)

- **Linear, unsupervised (= no class labels):**
  - Principal Component Analysis (PCA)

- **Linear, supervised (= use class labels):**
  - Linear Discriminant Analysis (LDA)

BioSB

Principal component analysis
(Unsupervised feature extraction)

- **Principal component analysis (PCA, 1901):**
  Goal: find directions in data...

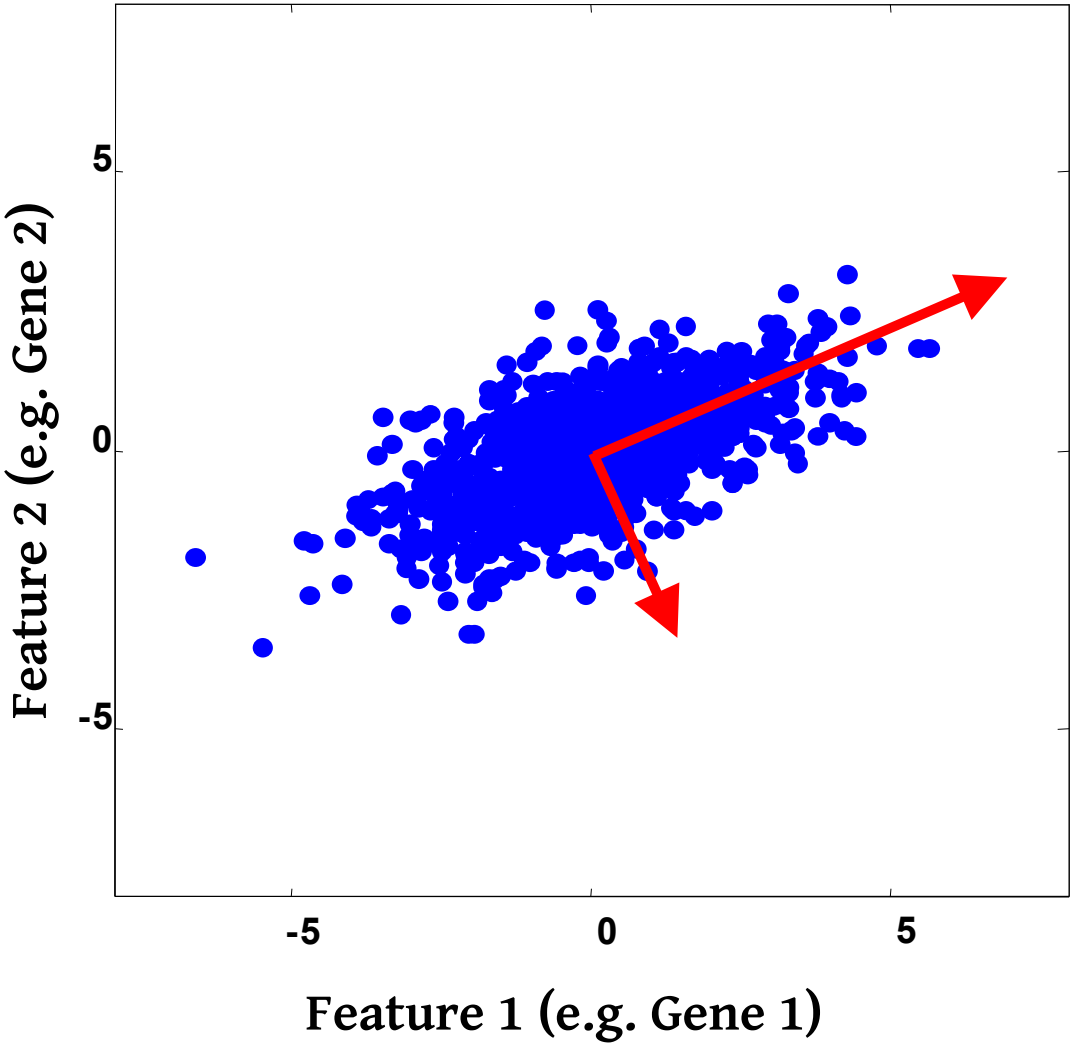
Principal component analysis
(Unsupervised feature extraction)

- **Principal component analysis (PCA, 1901):**
  Goal: find directions in data...
  - which retain as much *variation* as possible

Principal component analysis
(Unsupervised feature extraction)

- **Principal component analysis (PCA, 1901):**
  Goal: find directions in data...
  - which retain as much *variation* as possible
  - which minimise squared *reconstruction error*

Principal component analysis (Unsupervised feature extraction)



Steps:

1. Center data
2. Compute covariance, C
3. Perform PCA on C

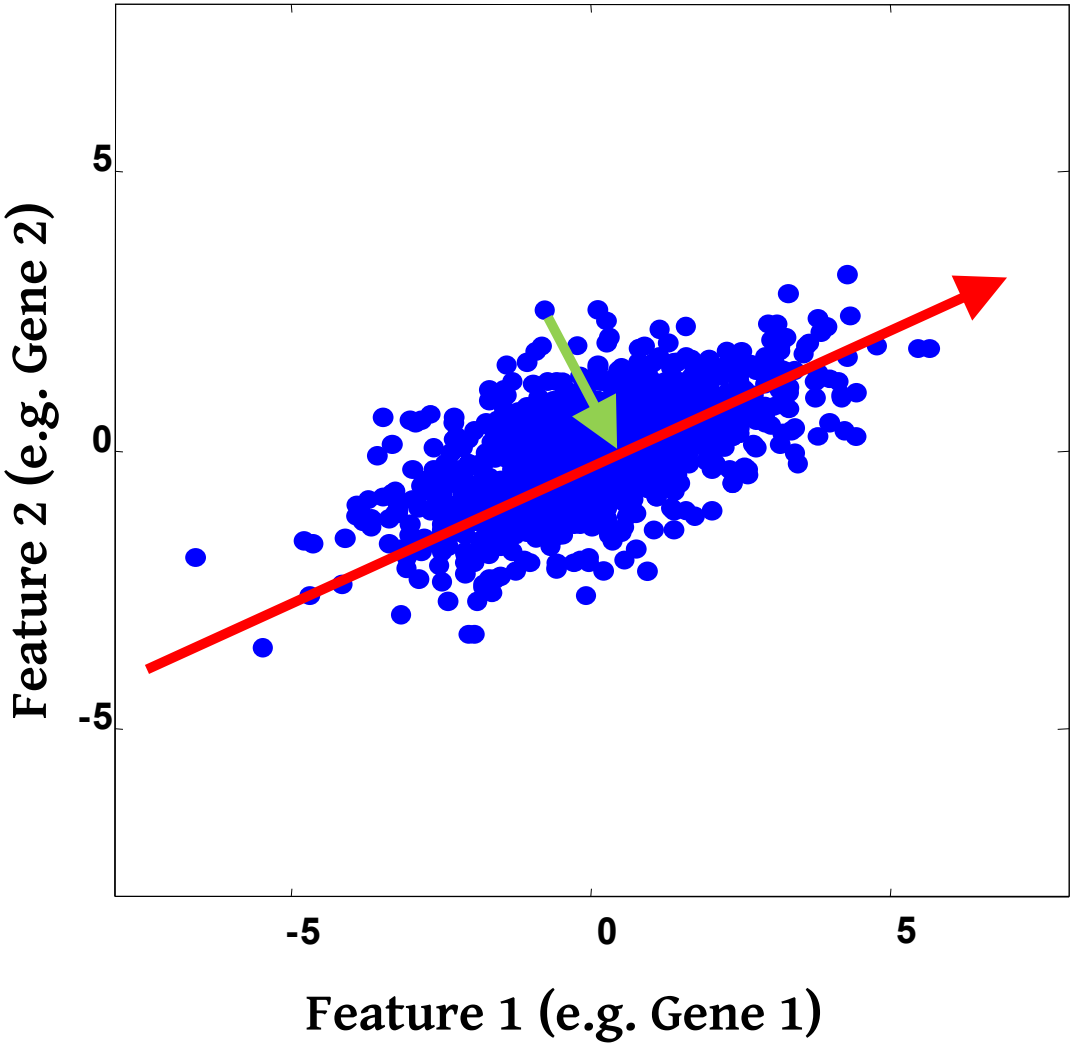
Output:

1. Eigenvectors: $\boldsymbol{e}_1$, $\boldsymbol{e}_2$
2. Eigenvalues: λ_1, λ_2

Reducing dimensions:

Choosing '*d*'

Principal component analysis (Unsupervised feature extraction)



Steps:

1. Center data
2. Compute covariance, C
3. Perform PCA on C

Output:

1. Eigenvectors: $\boldsymbol{e}_1$, $\boldsymbol{e}_2$
2. Eigenvalues: λ_1, λ_2

Reducing dimensions:

1. Choosing $\boldsymbol{d = 1}$
2. *Project data on* $\boldsymbol{e}_1$

Choosing reduced dimensionality

- To choose d inspect the retained variance,

$$\sum_{i=1}^{d} \lambda_i$$

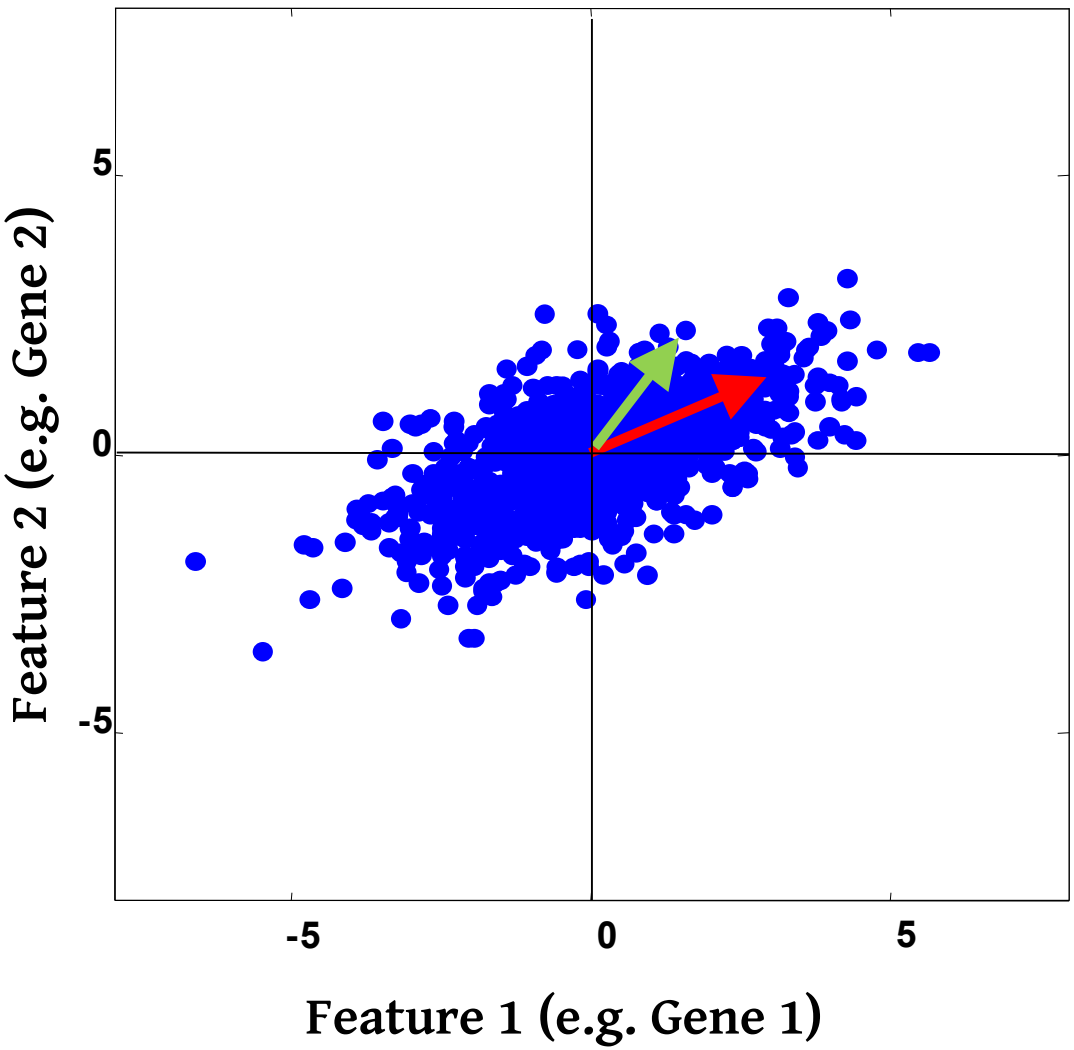
- or the ratio of retained variance,

$$\sum_{i=1}^{d} \lambda_i \Big/ \sum_{j=1}^{p} \lambda_j$$

- Rule of thumb: Select d for which 80-90% variance is retained

- Reduced dimensionality data set
  - $[\boldsymbol{x}_1^{\mathrm{T}}; \boldsymbol{x}_2^{\mathrm{T}}; \ldots; \boldsymbol{x}_2^{\mathrm{T}}][\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_{\mathrm{d}}]$

Principal component analysis (Unsuperrvised feature extraction)



Steps:

1. Center data
2. Compute covariance, C
3. Perform PCA on C

Output:

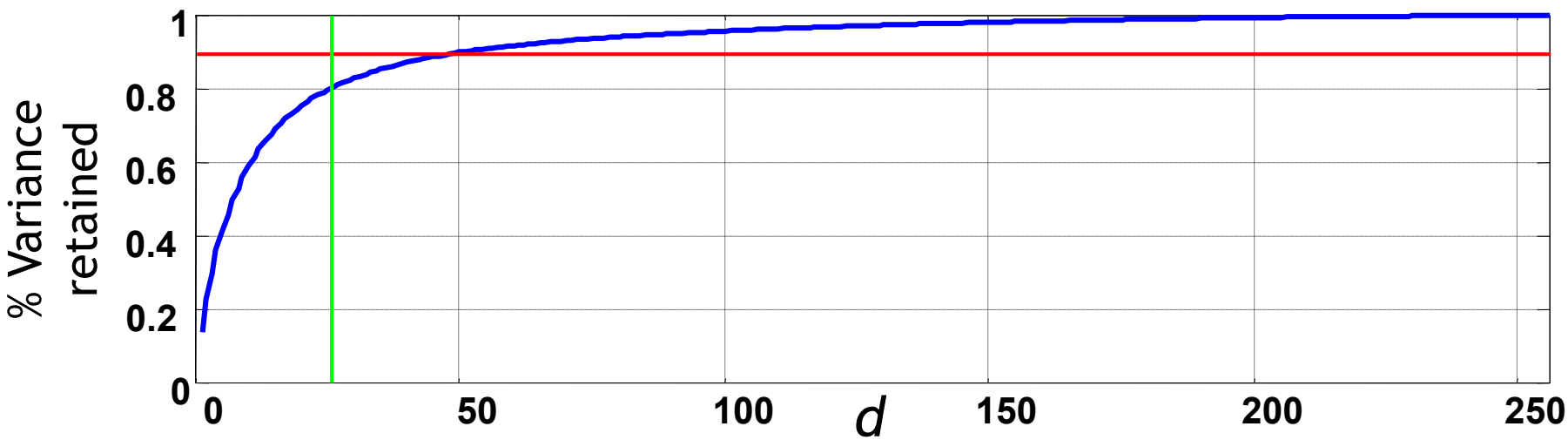
1. Eigenvectors: $\boldsymbol{e}_1$, $\boldsymbol{e}_2$
2. Eigenvalues: λ_1, λ_2

Reducing dimensions:

1. Choosing $\boldsymbol{d = 1}$
2. *Project data on* $\boldsymbol{e}_1$

PCA example

- *e.g.* NIST digits: 2000 samples, p = 256 (16 X 16)



PCA tips

- Ensure data is centered (mean of each feature is zero):
  $\boldsymbol{x}' \leftarrow (\boldsymbol{x} - \boldsymbol{\mu})$

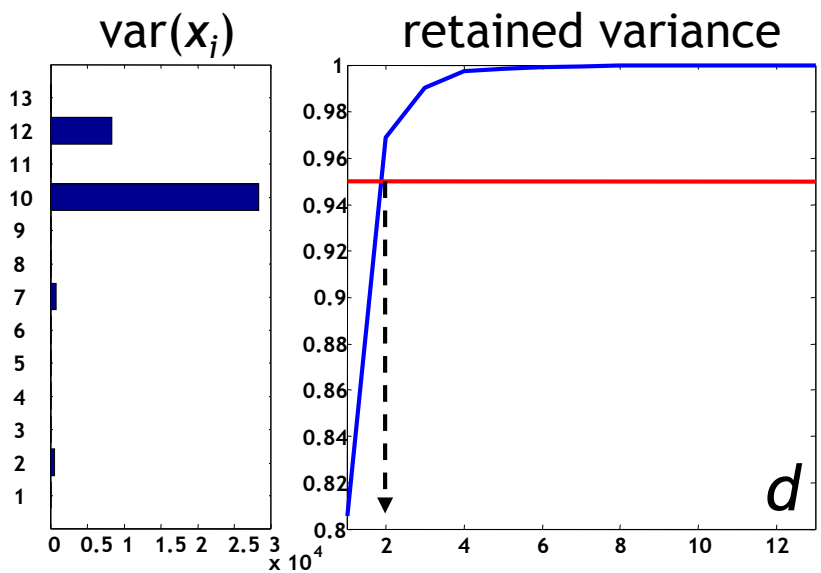
PCA tips

- Ensure data is centered (mean of each feature is zero):
  $\boldsymbol{x}' \leftarrow (\boldsymbol{x} - \boldsymbol{\mu})$
- PCA is sensitive to scaling
  - length in cm has a much larger variance than length in m
  - best to standardise: $\boldsymbol{x}' \leftarrow (\boldsymbol{x} - \boldsymbol{\mu}) / \sigma$

BioSB

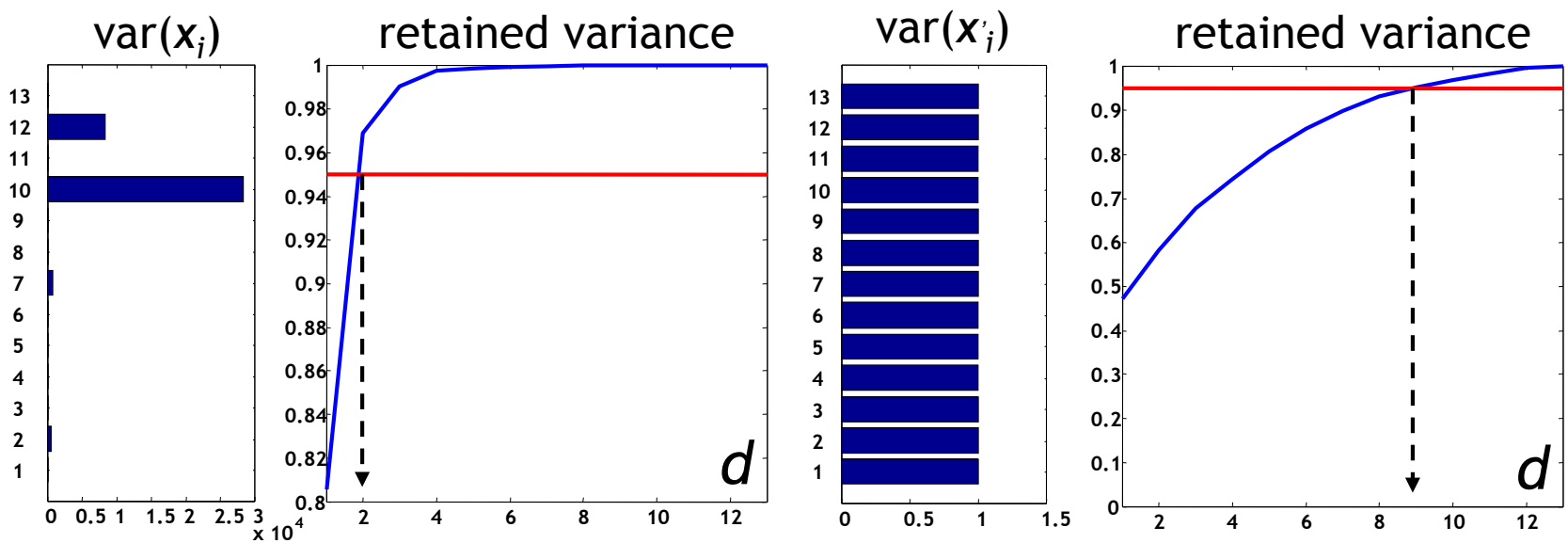
PCA tips

- Ensure data is centered (mean of each feature is zero): $\boldsymbol{x}' \leftarrow (\boldsymbol{x} - \boldsymbol{\mu})$
- PCA is sensitive to scaling
  - length in cm has a much larger variance than length in m
  - best to standardise: $\boldsymbol{x}' \leftarrow (\boldsymbol{x} - \boldsymbol{\mu}) / \sigma$



PCA tips

- Ensure data is centered (mean of each feature is zero): $\boldsymbol{x}' \leftarrow (\boldsymbol{x} - \boldsymbol{\mu})$
- PCA is sensitive to scaling
  - length in cm has a much larger variance than length in m
  - best to standardise: $\boldsymbol{x}' \leftarrow (\boldsymbol{x} - \boldsymbol{\mu}) / \sigma$



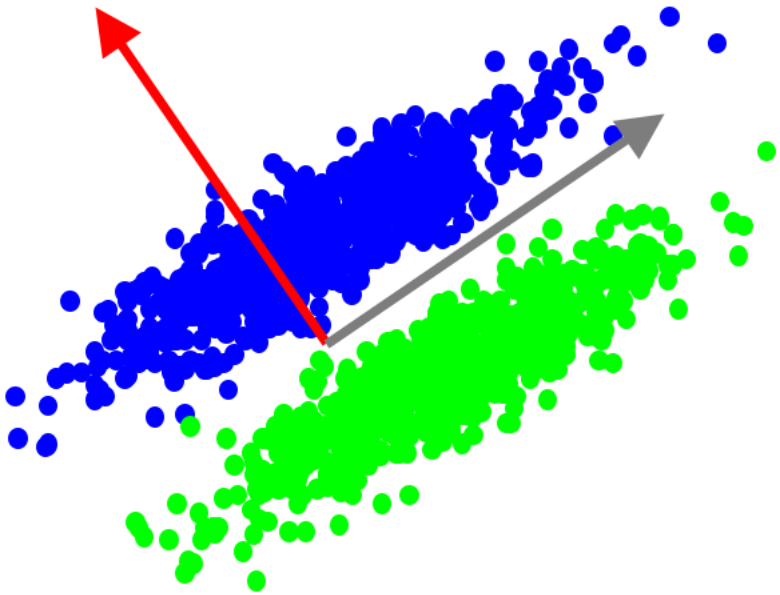
PCA conclusions

- PCA:
  - Is **global** and **linear**
  - Is **unsupervised** (but we can do PCA on each class)
  - Needs a **lot of data** to estimate $\boldsymbol{\Sigma}$ well.

PCA conclusions

- PCA:
  - Is **global** and **linear**
  - Is **unsupervised** (but we can do PCA on each class)
  - Needs a **lot of data** to estimate $\mathbf{\Sigma}$ well.

- Danger:
  - Criterion is not necessarily related to the goal;
  - Might discard important directions

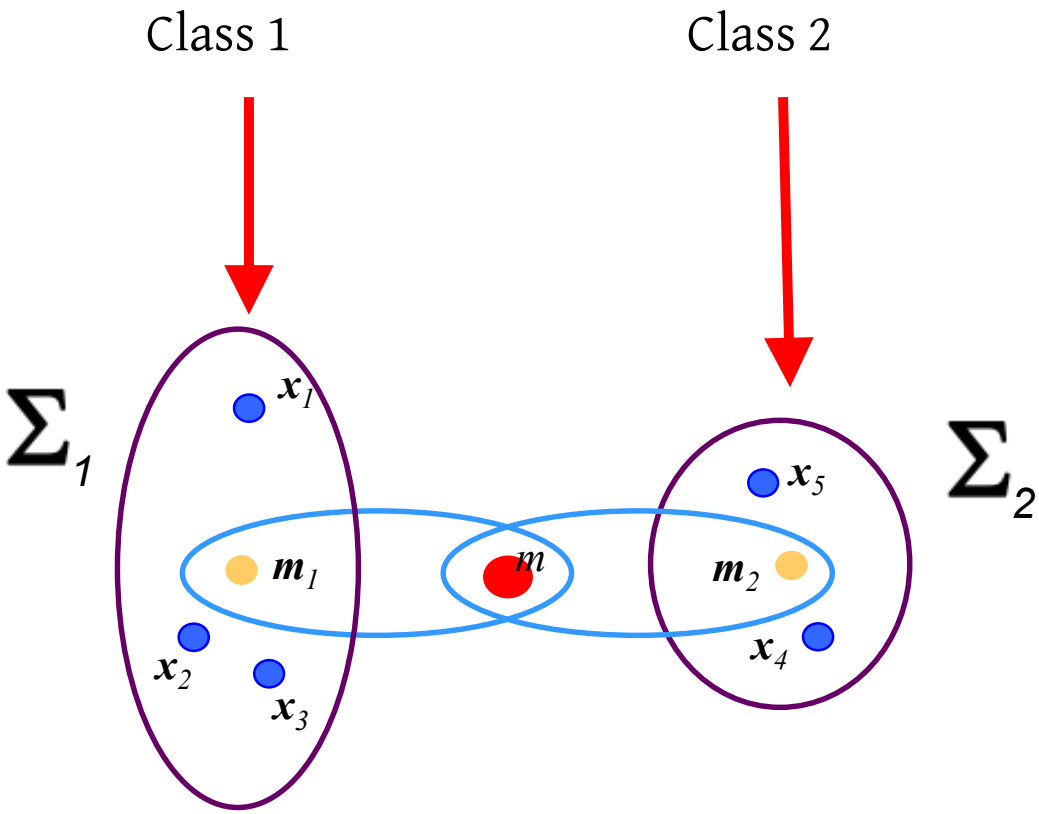


Supervised, linear feature extraction

- If class label ω (or $\boldsymbol{y}$) is given, supervised extraction
- Examples: Fisher mapping; Linear Discriminant Analysis (Day 2)

Supervised feature extraction (2)

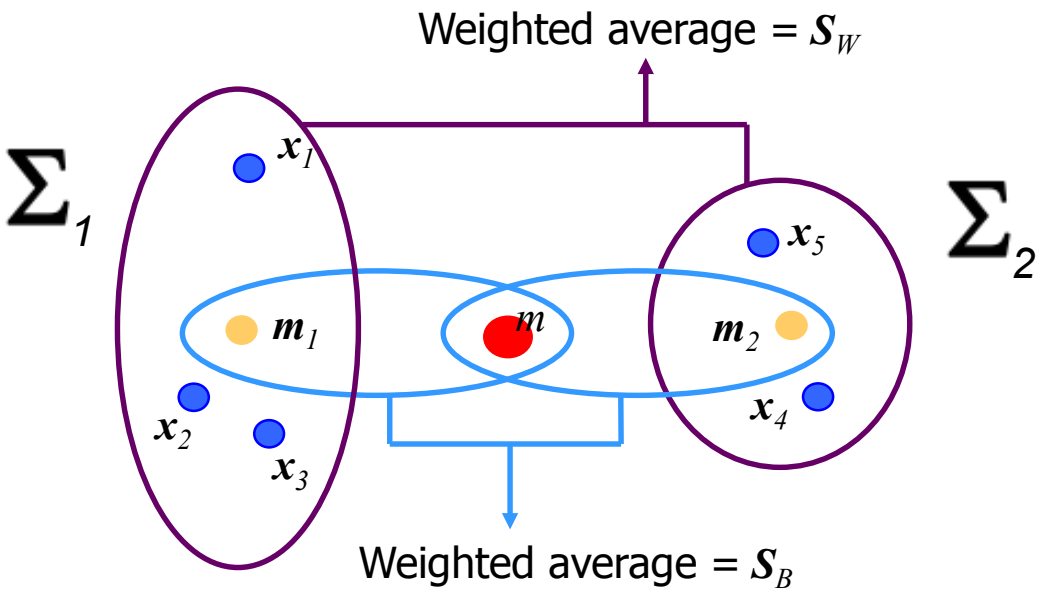
(supervised = we know the class labels)



Supervised feature extraction (2)

Within-class and between-class scatter matrices:

- Within-class: $$\boldsymbol{S}_w = \sum_{i=1}^{C} \frac{n_i}{n} \boldsymbol{\Sigma}_i$$

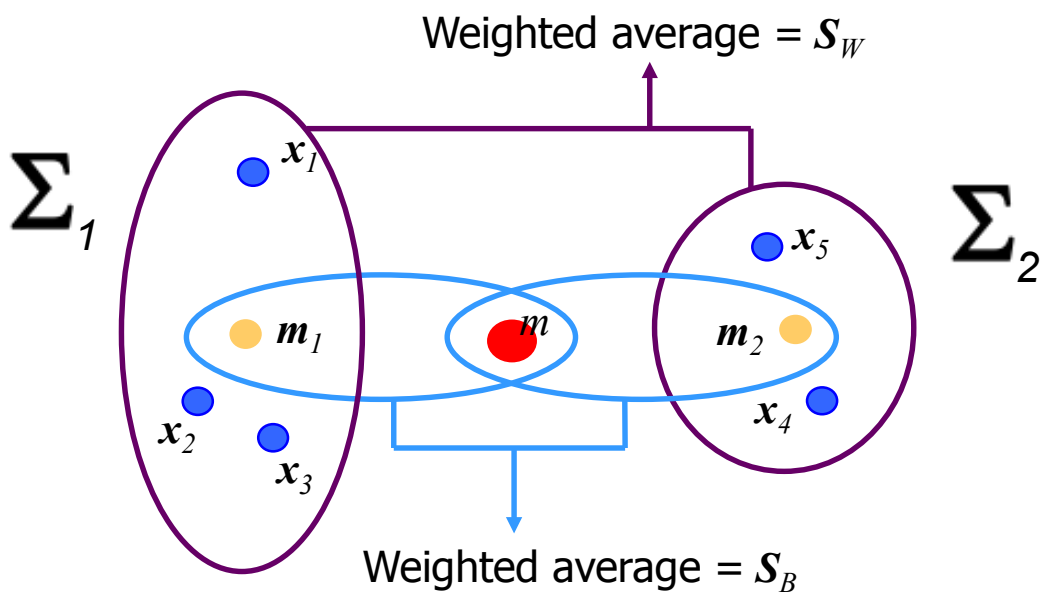


Supervised feature extraction (2)

Within-class and between-class scatter matrices:

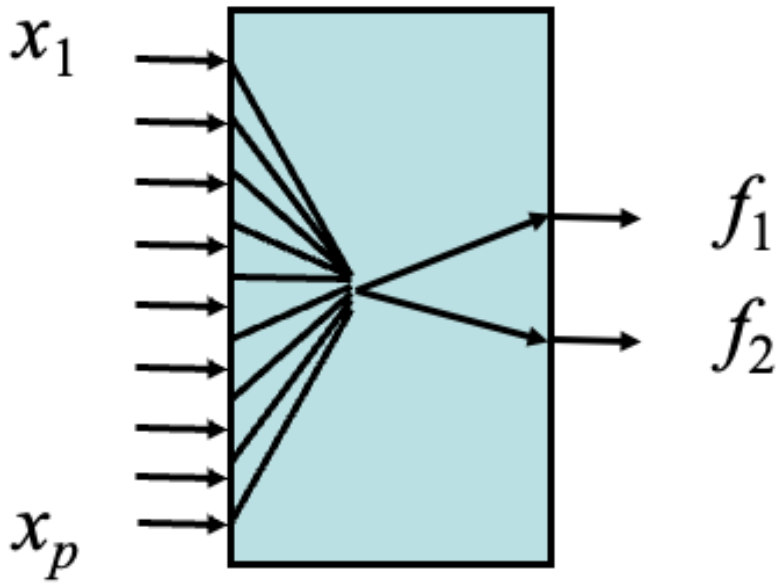
- Within-class: $$\boldsymbol{S}_w = \sum_{i=1}^{C} \frac{n_i}{n} \boldsymbol{\Sigma}_i$$

- Between-class: $$\boldsymbol{S}_B = \sum_{i=1}^{C} \frac{n_i}{n} (\boldsymbol{m}_i - \boldsymbol{m})(\boldsymbol{m}_i - \boldsymbol{m})^T$$

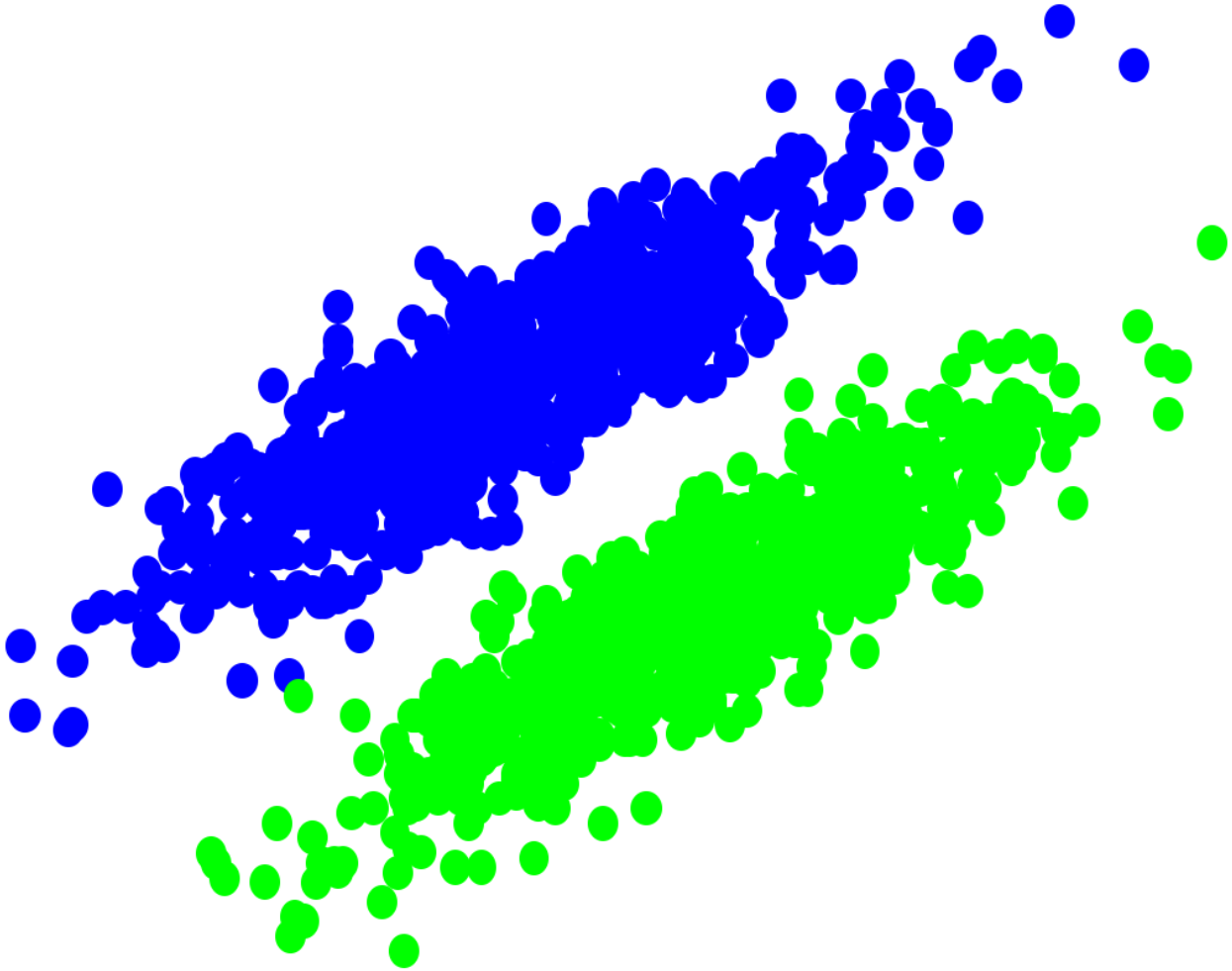


Supervised, linear feature extraction

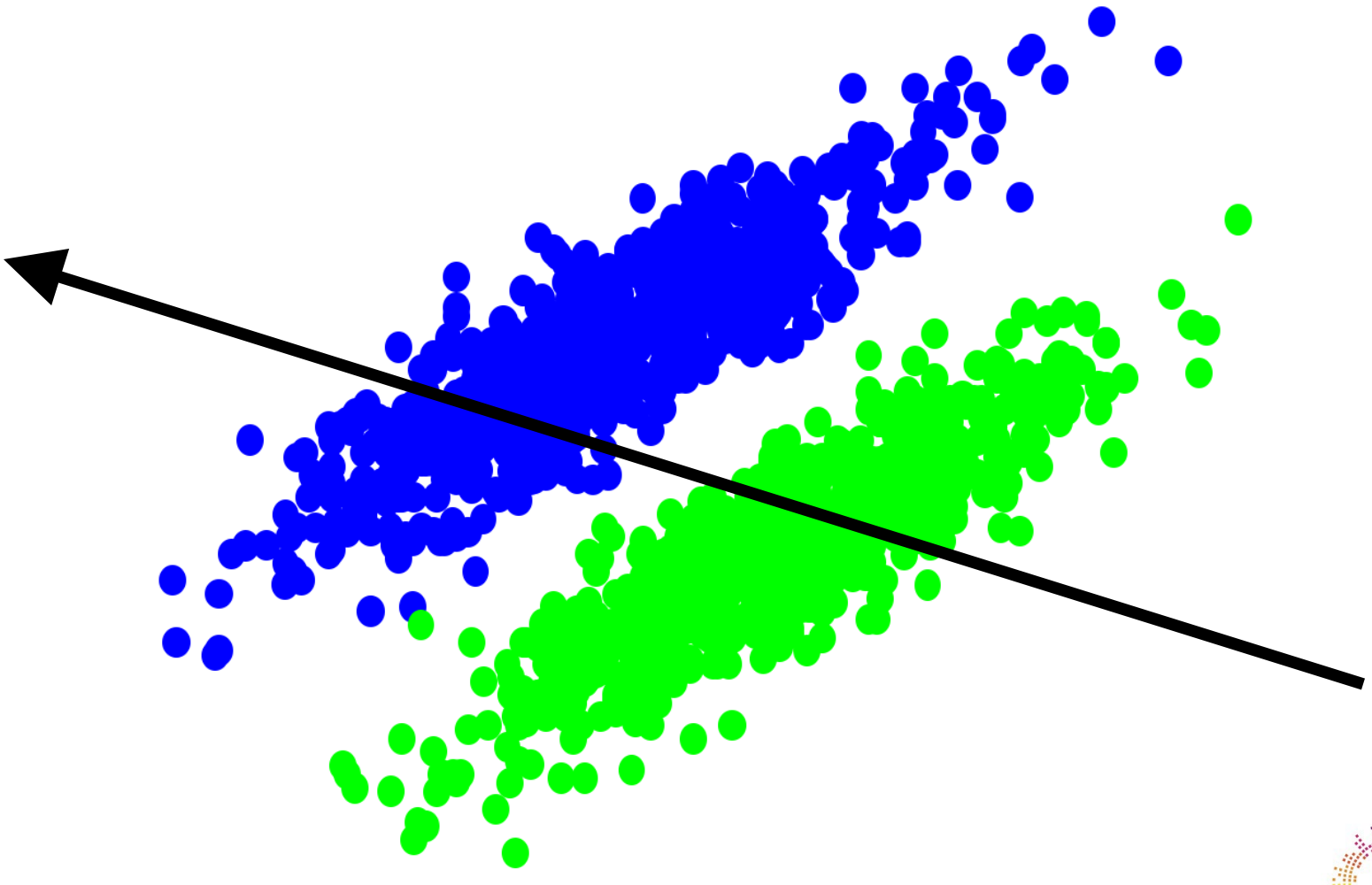
- Example: Fisher mapping
- Supervised: we know the class labels
- Extraction: mapping of features to new (sub)space
- This (sub)space gives the best class separation



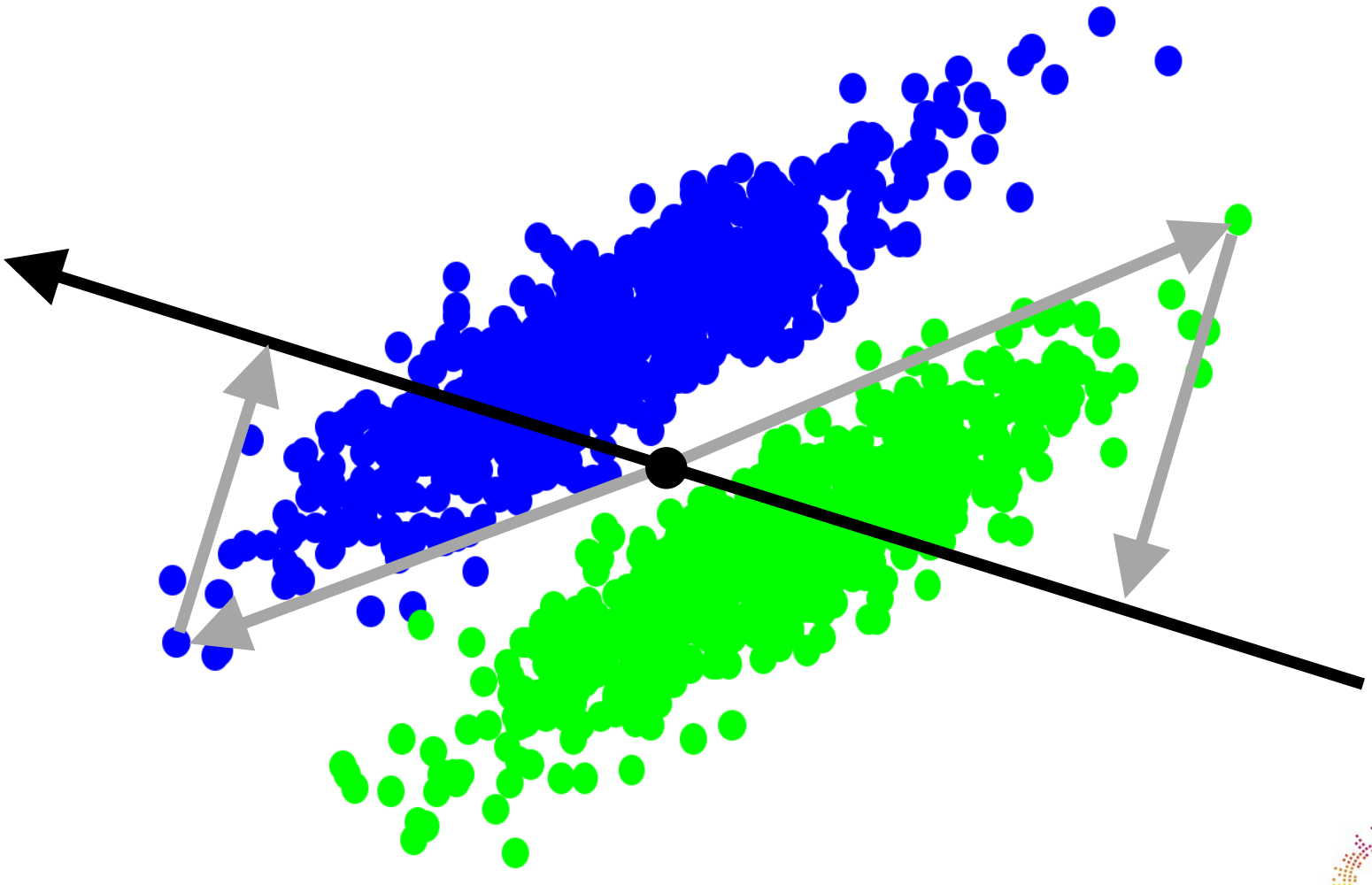
Fisher mapping: finding the direction (subspace) to project onto for the best class separation



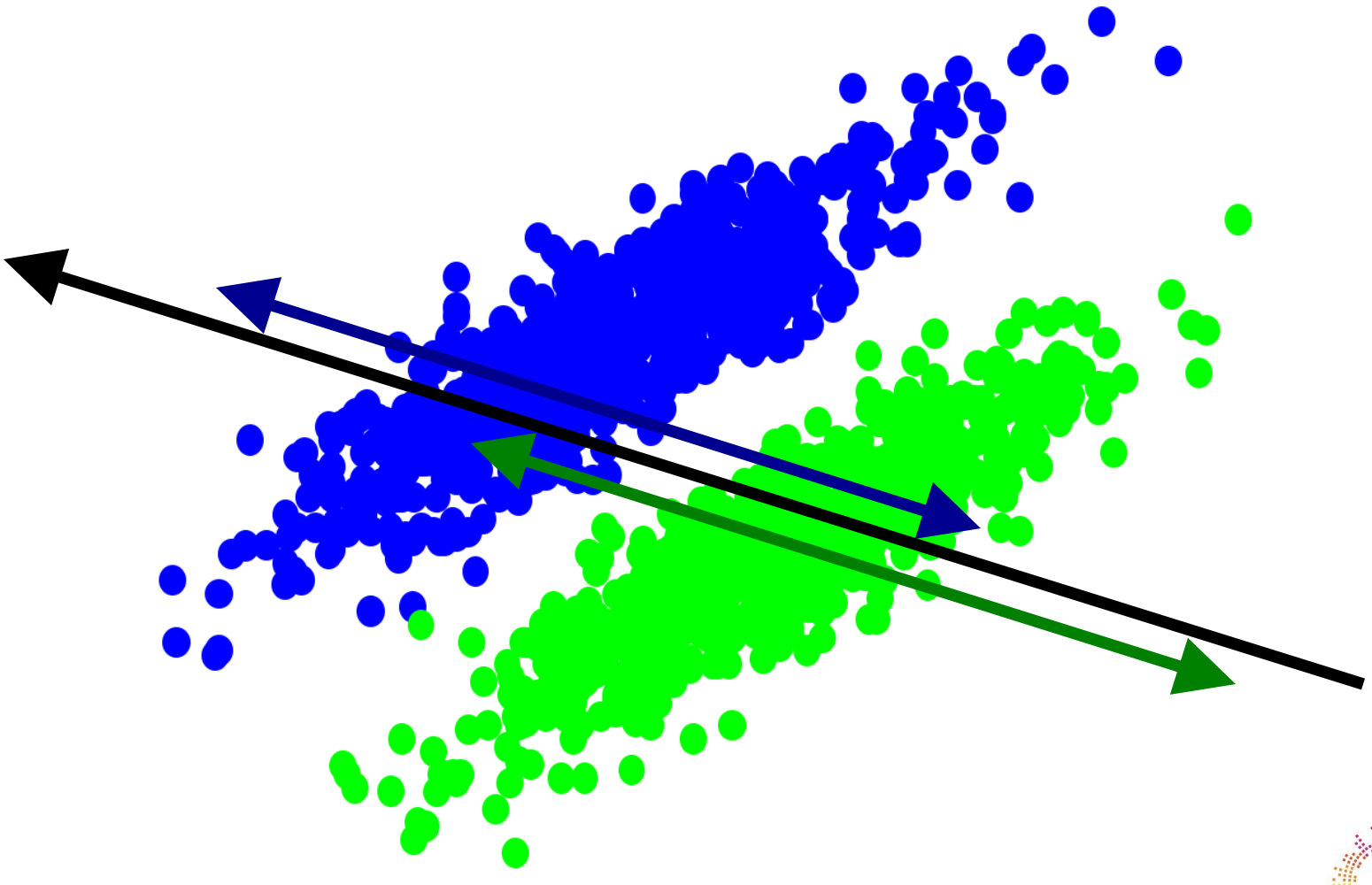
Fisher mapping: defining the Fisher criterion



Fisher mapping: defining the Fisher criterion

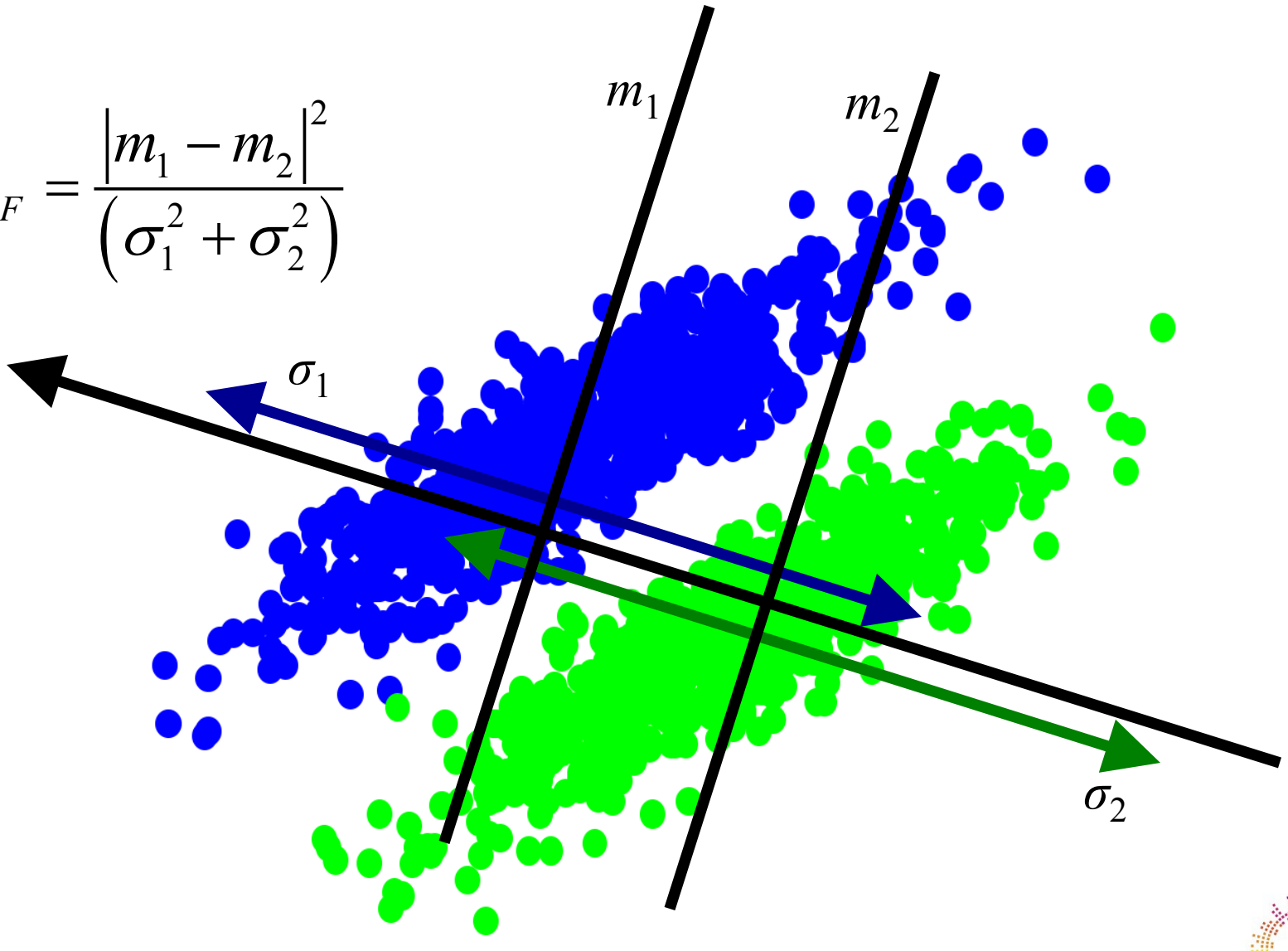


Fisher mapping: defining the Fisher criterion



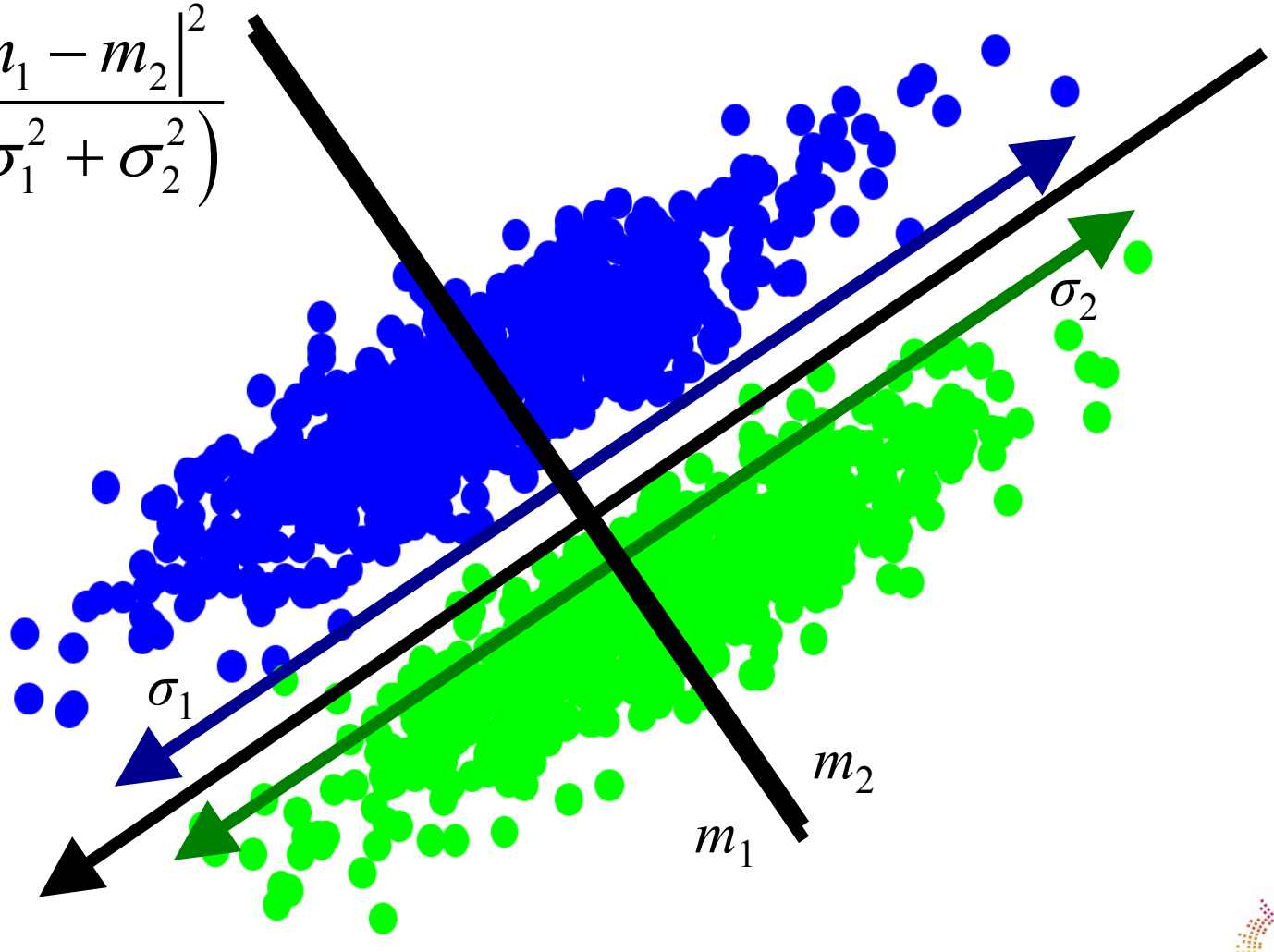
Fisher mapping (Fisher criterion)

$$J_F = \frac{\left|m_1 - m_2\right|^2}{\left(\sigma_1^2 + \sigma_2^2\right)}$$



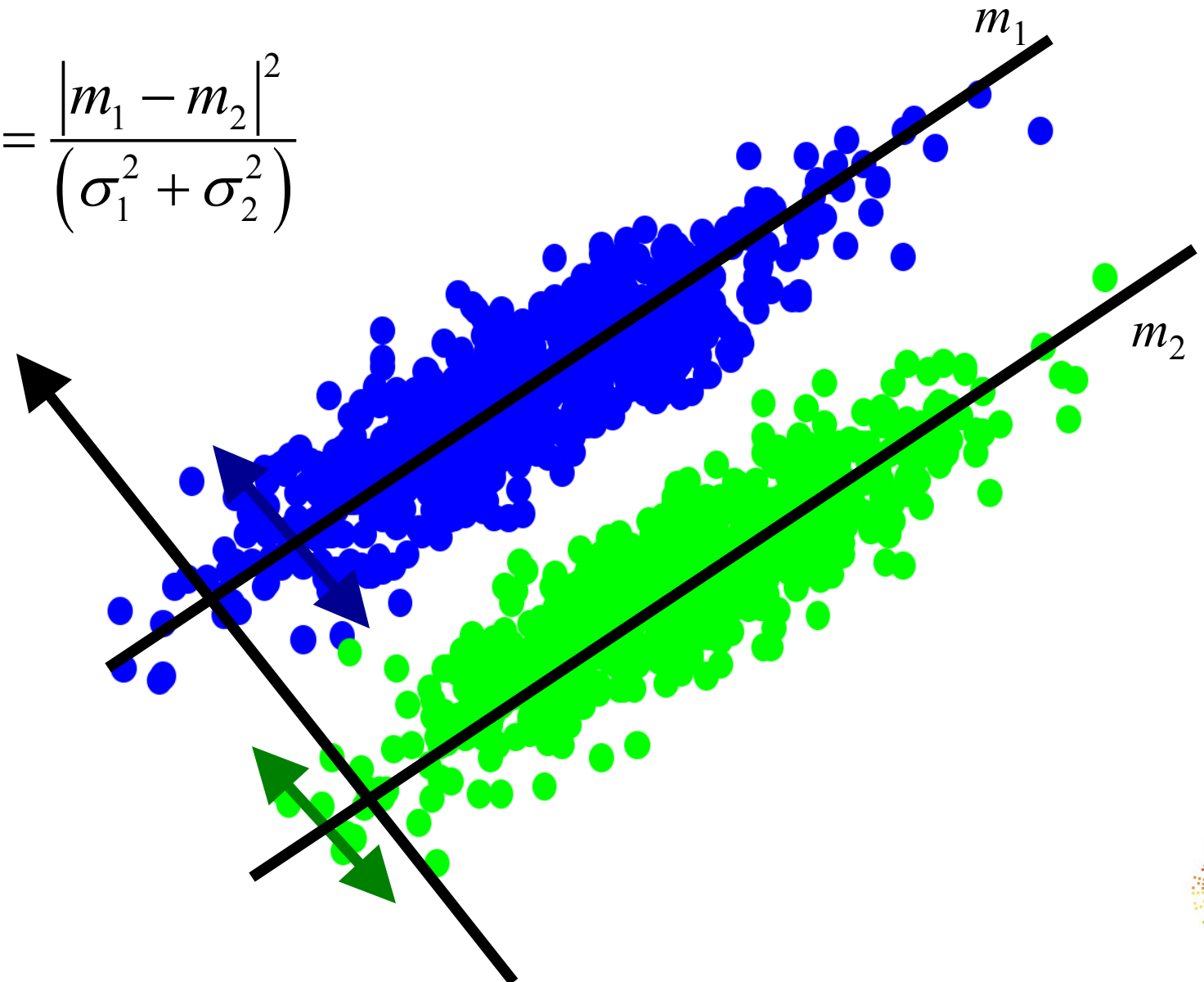
Fisher mapping: defining the Fisher criterion

$$J_F = \frac{\left|m_1 - m_2\right|^2}{\left(\sigma_1^2 + \sigma_2^2\right)}$$



Fisher mapping: defining the Fisher criterion

$$J_F = \frac{\left|m_1 - m_2\right|^2}{\left(\sigma_1^2 + \sigma_2^2\right)}$$



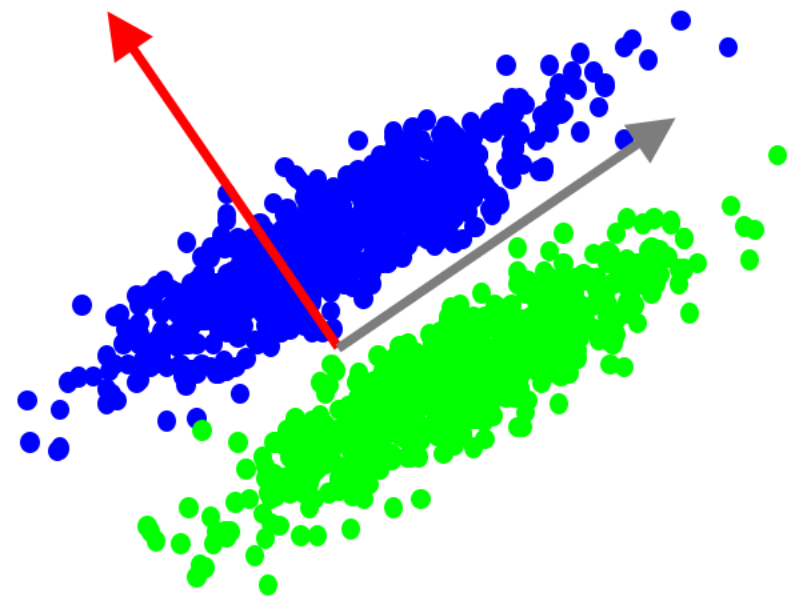
Fisher mapping

- Find basis vector $\boldsymbol{a}_1$ for $\{\boldsymbol{x}\}$ such that in the projections, the classes are maximally separated
- Choose $\boldsymbol{a}_1$ to maximise Fisher criterion:

$$J_F(\boldsymbol{a}_1) = \frac{\boldsymbol{a}_1^T \boldsymbol{S}_B \boldsymbol{a}_1}{\boldsymbol{a}_1^T \boldsymbol{S}_W \boldsymbol{a}_1}$$

- Maximize between class variance

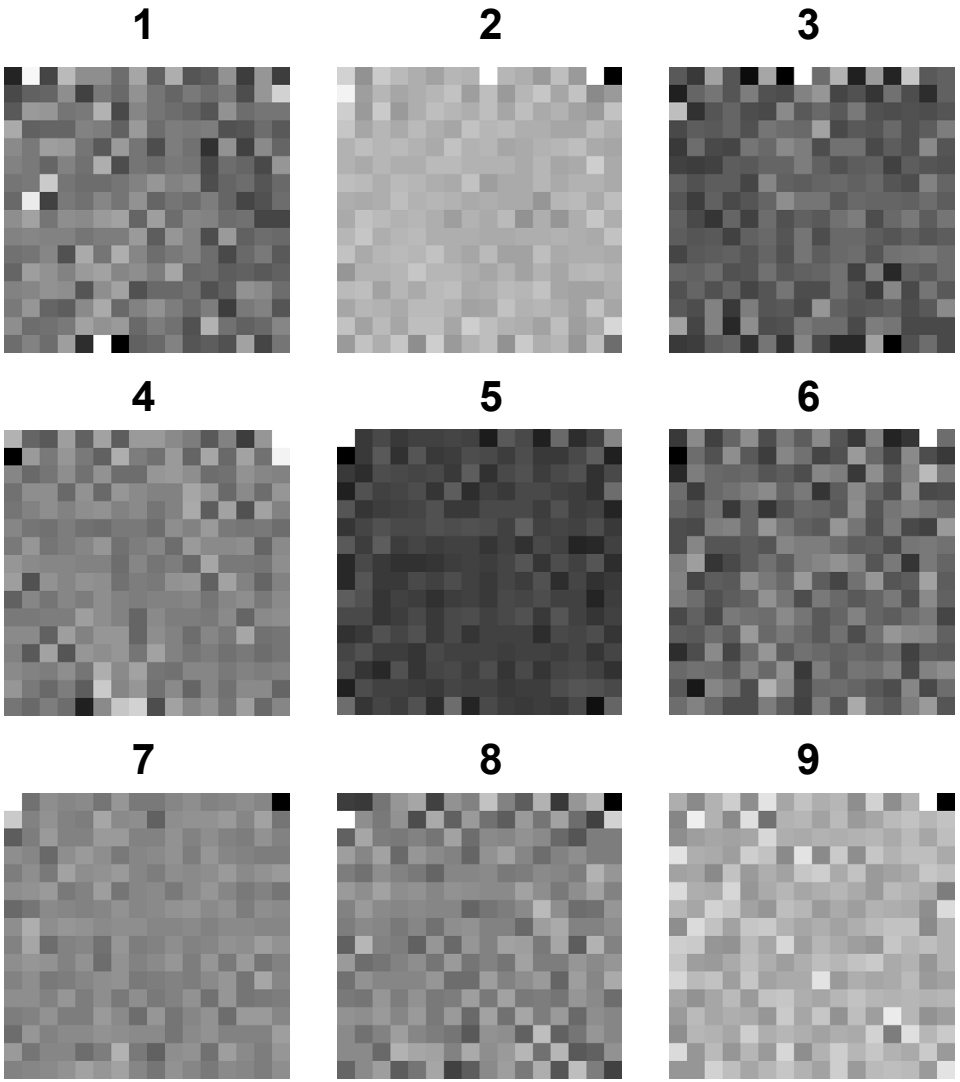
  Minimize within class variance



- Solution:
  - eigen-analysis on $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$
  - select c-1 (# classes – 1) dimensions for final classifier

Fisher mapping (3)

- Map down to a maximum of c - 1 dimensions
- Example: NIST digits



Fisher mapping (4)

- To avoid fitting noise, can do PCA first
- If system is underdetermined ($n \leq p$), first doing PCA is required, otherwise matrix inversion results in singularity
- But then... ?

Fisher mapping (4)

- To avoid fitting noise, can do PCA first
- If system is underdetermined ($n \leq p$), first doing PCA is required, otherwise matrix inversion results in singularity
- But then we might be destroying the class separation as PCA is *unsupervised*

Summary

- Discussed:
  - Linear feature extraction
    - Unsupervised: Principal Component Analysis (PCA)
    - Supervised: Fisher mapping

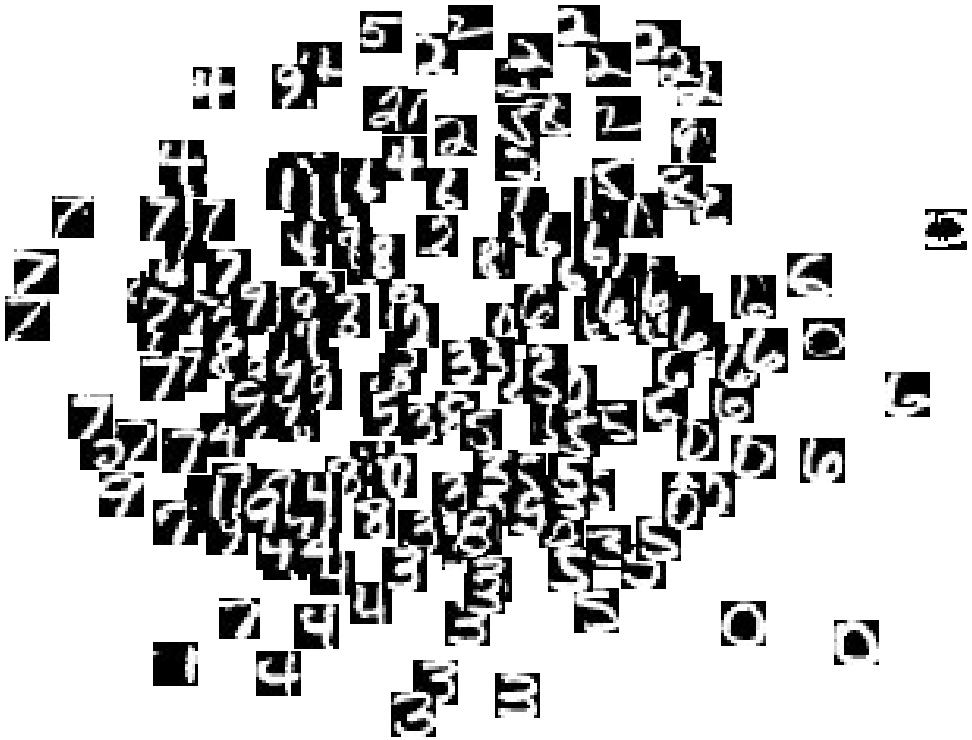
Nonlinear, unsupervised feature extraction

- **Multidimensional scaling (MDS):**
  - Nonlinear:
    - Sammon mapping
    - t-SNE / UMAP

Nonlinear feature extraction (3)

Example: embedding

- Find new representation such that distances between samples are preserved as well as possible

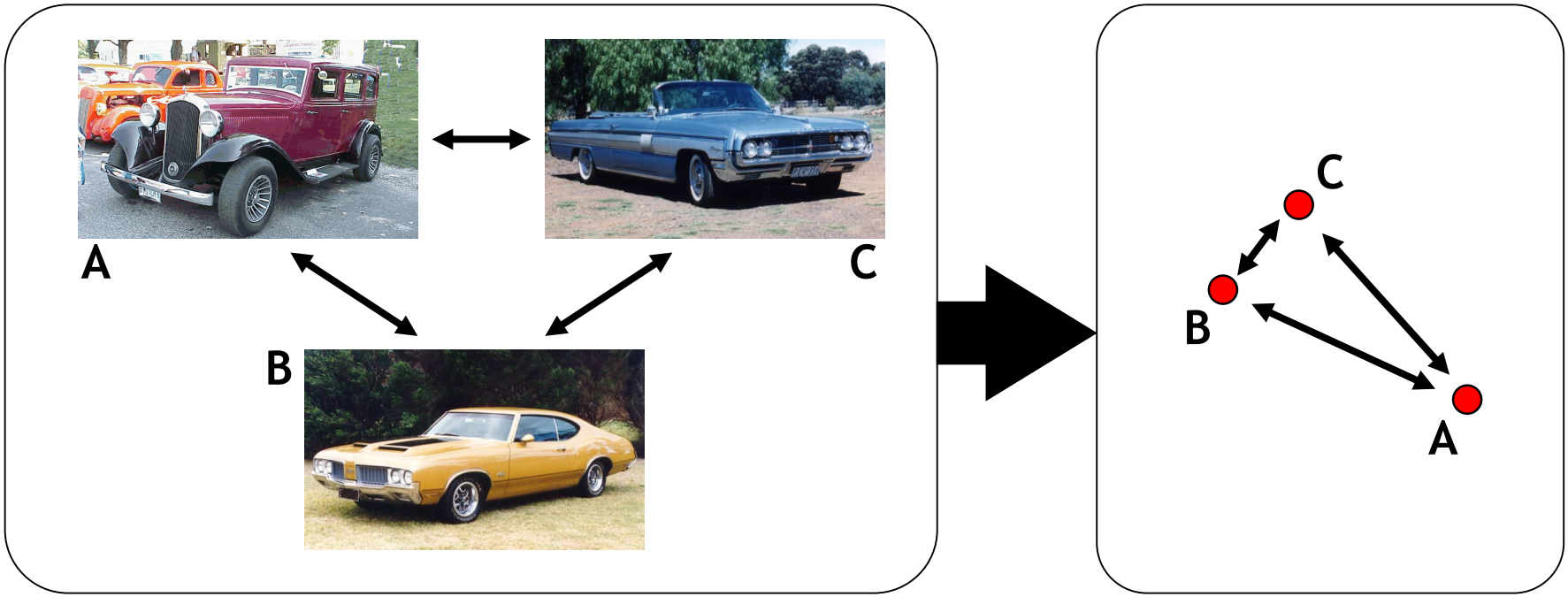


Multidimensional scaling (MDS)

- Criterion: preserve all inter-sample distances
- Needed: n x n distance matrix between all samples
- Map samples to a new (lower dimensional) space

Multidimensional scaling (MDS)

- Criterion: preserve all inter-sample distances
- Needed: n x n distance matrix between all points
- Map samples to a new (lower dimensional) space



MDS (2)

- Advantages of using distances:
  - do not necessarily need original data

MDS (2)

- Advantages of using distances:
  - do not necessarily need original data
  - allows inclusion of **knowledge on objects** (e.g. characteristics of amino acids when comparing proteins)

MDS (2)

- Advantages of using distances:
  - do not necessarily need original data
  - allows inclusion of **knowledge on objects** (e.g. characteristics of amino acids when comparing proteins)
  - allows inclusion of **knowledge of relations** (e.g. invariances) in distance measure (e.g. Pearson correlation being shift and scale invariant when comparing expression profiles)

MDS (2)

- Advantages of using distances:
  - do not necessarily need original data
  - allows inclusion of **knowledge on objects** (e.g. characteristics of amino acids when comparing proteins)
  - allows inclusion of **knowledge of relations** (e.g. invariances) in distance measure (e.g. Pearson correlation being shift and scale invariant when comparing expression profiles)
  - easy to introduce nonlinearity

MDS (2)

- Advantages of using distances:
  - do not necessarily need original data
  - allows inclusion of **knowledge on objects** (e.g. characteristics of amino acids when comparing proteins)
  - allows inclusion of **knowledge of relations** (e.g. invariances) in distance measure (e.g. Pearson correlation being shift and scale invariant when comparing expression profiles)
  - easy to introduce nonlinearity

- Algorithms should find:
  - new, low-dimensional coordinates for each object
  - the number of dimensions to embed the data in

MDS: Non-linear mappings (5)

- d_{ij} : distance $\| \boldsymbol{x}_i - \boldsymbol{x}_j \|$ in original space (? - dimensional)
- δ_{ij} : distance $\| \boldsymbol{y}_i - \boldsymbol{y}_j \|$ in new space (d - dimensional)

MDS: Non-linear mappings (5)

- d_{ij} : distance $\| \boldsymbol{x}_i - \boldsymbol{x}_j \|$ in original space (? - dimensional)
- δ_{ij} : distance $\| \boldsymbol{y}_i - \boldsymbol{y}_j \|$ in new space (d - dimensional)

$$Stress(\boldsymbol{y}) = \frac{1}{\sum_i \sum_{j>i} d_{ij}^{(q+2)}} \sum_i \sum_{j>i} d_{ij}^q (\delta_{ij} - d_{ij})^2$$

- weight factor q = ..., -2, -1, 0, 1, 2,...

  $q > 0$: emphasise large distances

  $q < 0$: de-emphasise large distances (smaller more important)

  Sammon mapping: $q = -1$

MDS: Non-linear mappings (6)

- **Procedure**:
  - Initialize positions of samples in lower dimensional space ($\boldsymbol{y}_i$)

MDS: Non-linear mappings (6)

- **Procedure**:
  - Initialize positions of samples in lower dimensional space ($\boldsymbol{y}_i$)
  - Compute stress

$$Stress(\boldsymbol{y}) = \frac{1}{\sum_i \sum_{j>i} d_{ij}^{(q+2)}} \sum_i \sum_{j>i} d_{ij}^{q} (\delta_{ij} - d_{ij})^2$$

MDS: Non-linear mappings (6)

- **Procedure:**
  - Initialize positions of samples in lower dimensional space ($\boldsymbol{y}_i$)
  - Compute stress

$$Stress(\boldsymbol{y}) = \frac{1}{\sum_i \sum_{j>i} d_{ij}^{(q+2)}} \sum_i \sum_{j>i} d_{ij}^{q} (\delta_{ij} - d_{ij})^2$$

  - Compute derivative of the stress with respect to positions of samples in new space

MDS: Non-linear mappings (6)

- **Procedure:**
  - Initialize positions of samples in lower dimensional space ($\boldsymbol{y}_i$)
  - Compute stress

$$Stress(\boldsymbol{y}) = \frac{1}{\sum_i \sum_{j>i} d_{ij}^{(q+2)}} \sum_i \sum_{j>i} d_{ij}^q (\delta_{ij} - d_{ij})^2$$

  - Compute derivative of the stress with respect to positions of samples in new space
  - Adapt the positions of samples in lower dimensional space

$$\boldsymbol{y}' = \boldsymbol{y} - \alpha \frac{\partial Stress(\boldsymbol{y})}{\partial \boldsymbol{y}}$$

MDS: Non-linear mappings (6)

- **Procedure:**
  - Initialize positions of samples in lower dimensional space ($\boldsymbol{y}_i$)
  - Compute stress

$$Stress(\boldsymbol{y}) = \frac{1}{\sum_i \sum_{j>i} d_{ij}^{(q+2)}} \sum_i \sum_{j>i} d_{ij}^q (\delta_{ij} - d_{ij})^2$$

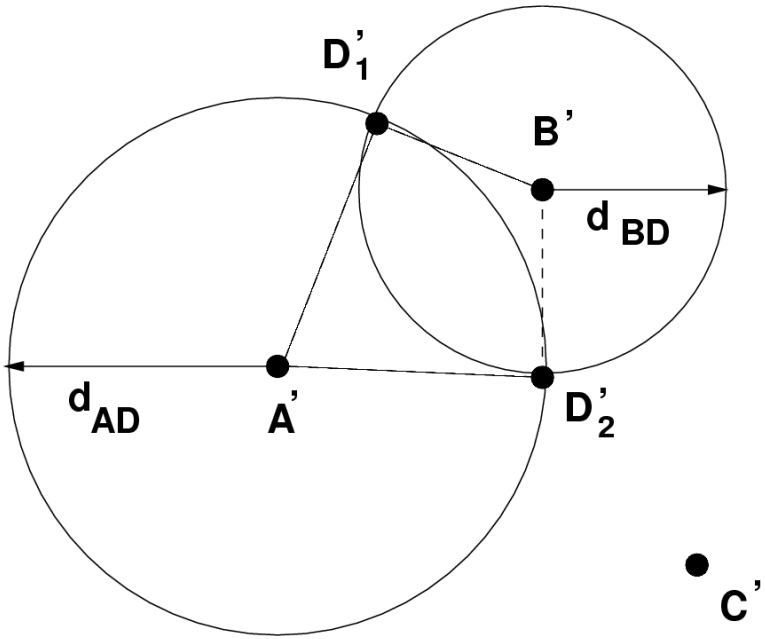
  - Compute derivative of the stress with respect to positions of samples in new space
  - Adapt the positions of samples in lower dimensional space

$$\boldsymbol{y}' = \boldsymbol{y} - \alpha \frac{\partial Stress(\boldsymbol{y})}{\partial \boldsymbol{y}}$$

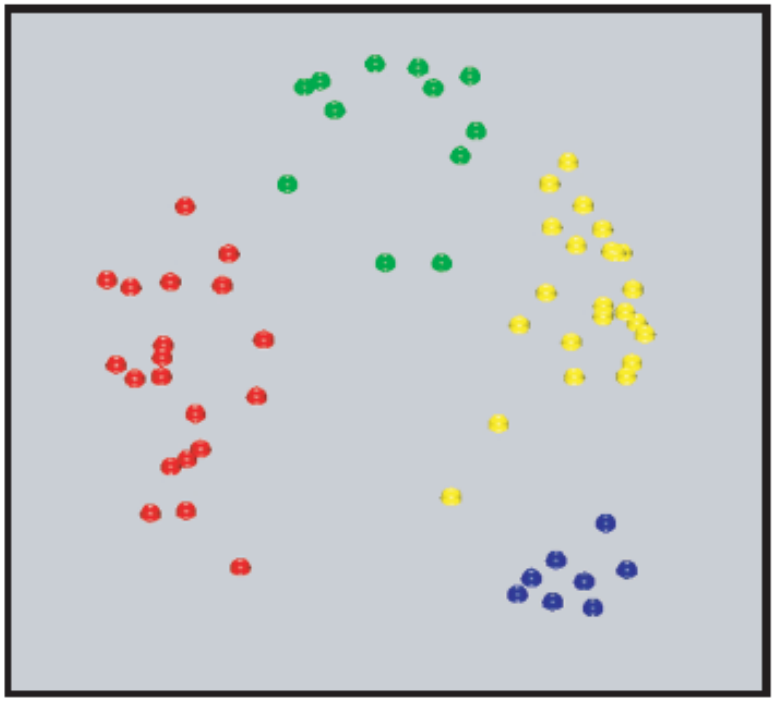
  - Repeat till convergence (positions of samples do not change)

Embedding new points

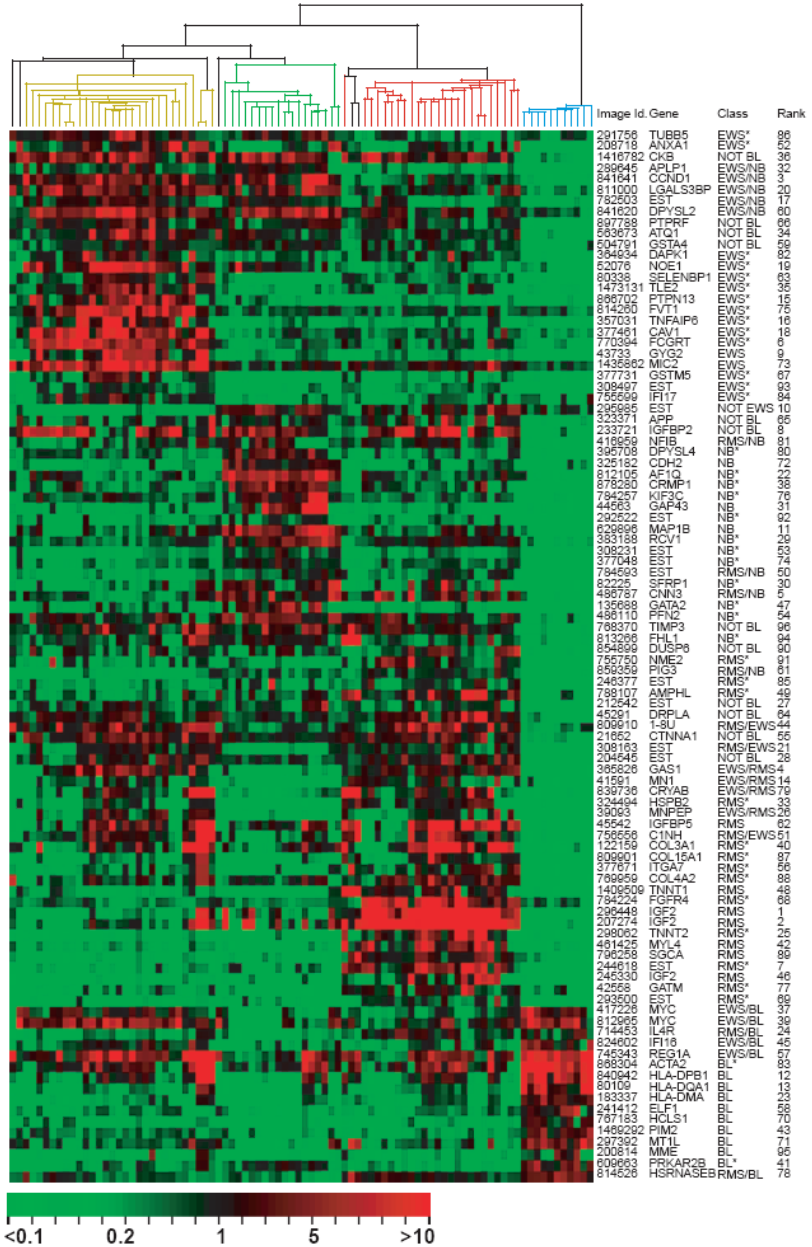
- Problematic: re-run entire algorithm…
- Sub-optimal solution: triangulation
  - Embed new point **D**
  - **D** has **A** and **B** as neighbors in original space
  - Preserve distance to two embedded neigbours **A'**, **B'** exactly
  - Use **C'** to decide which of the two candidates $\mathbf{D_1}'$, $\mathbf{D_2}'$ to use



MDS example



- Neuroblastoma (NB)
- Rhabdomyosarcoma (RMS)
- Burkitt lymphoma (BL)
- Ewing family of tumors (EWS),



Khan et al, Nature Medicine, 2001

t-SNE (t-distributed stochastic neighbor embedding) (van der Maaten et al, 2008)

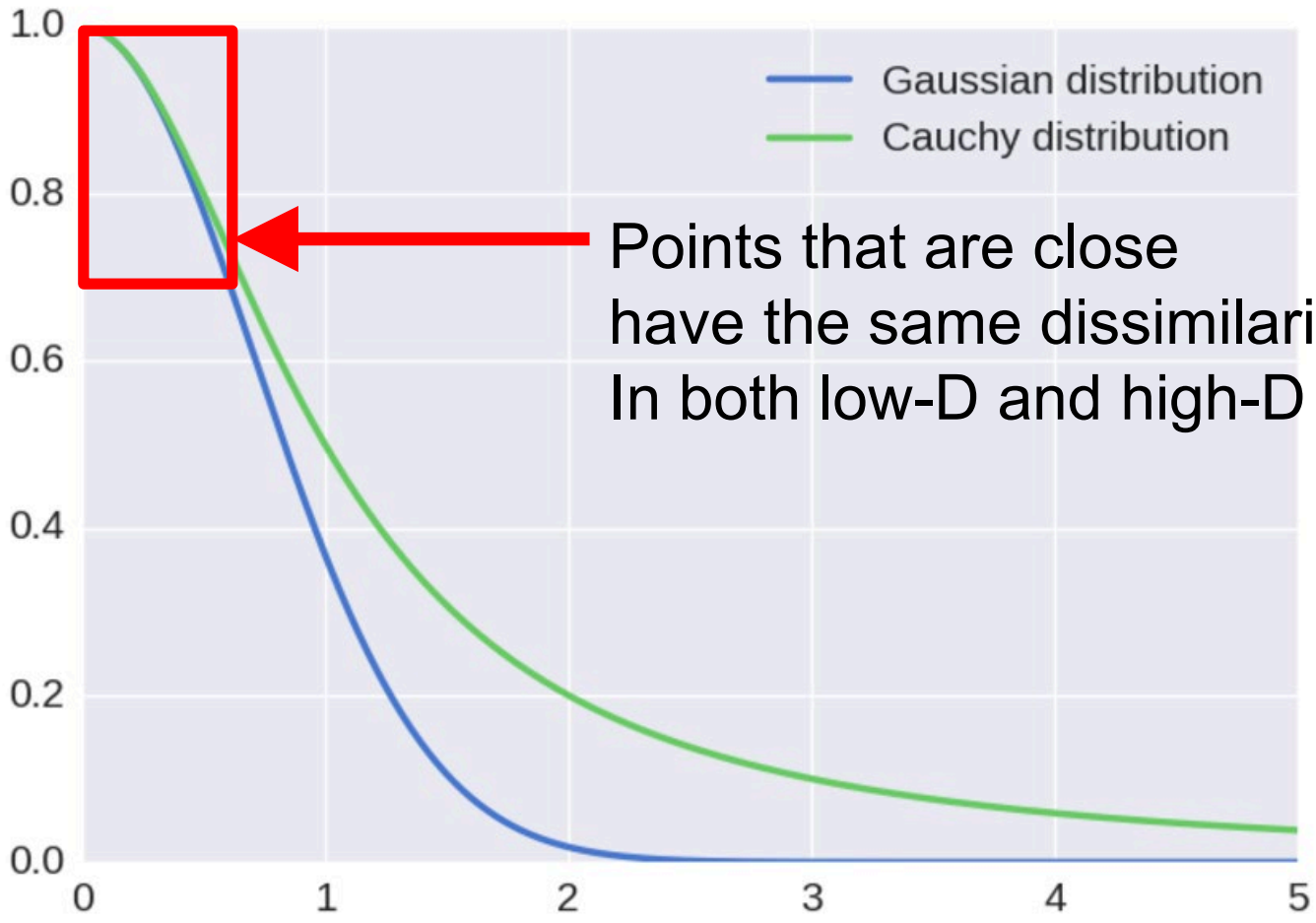
- In the input (high-D) space, X: compute dissimilarities between all pairs of points using a gaussian dissimilarity measure, pij

- In the output (low-D) space, Y: compute dissimilarities between all pairs of points using a t-distribution (with 1 d.o.f. (Cauchy)) dissimilarity measure, qij

- Minimize the Kullback-Leibler distance between these two distributions

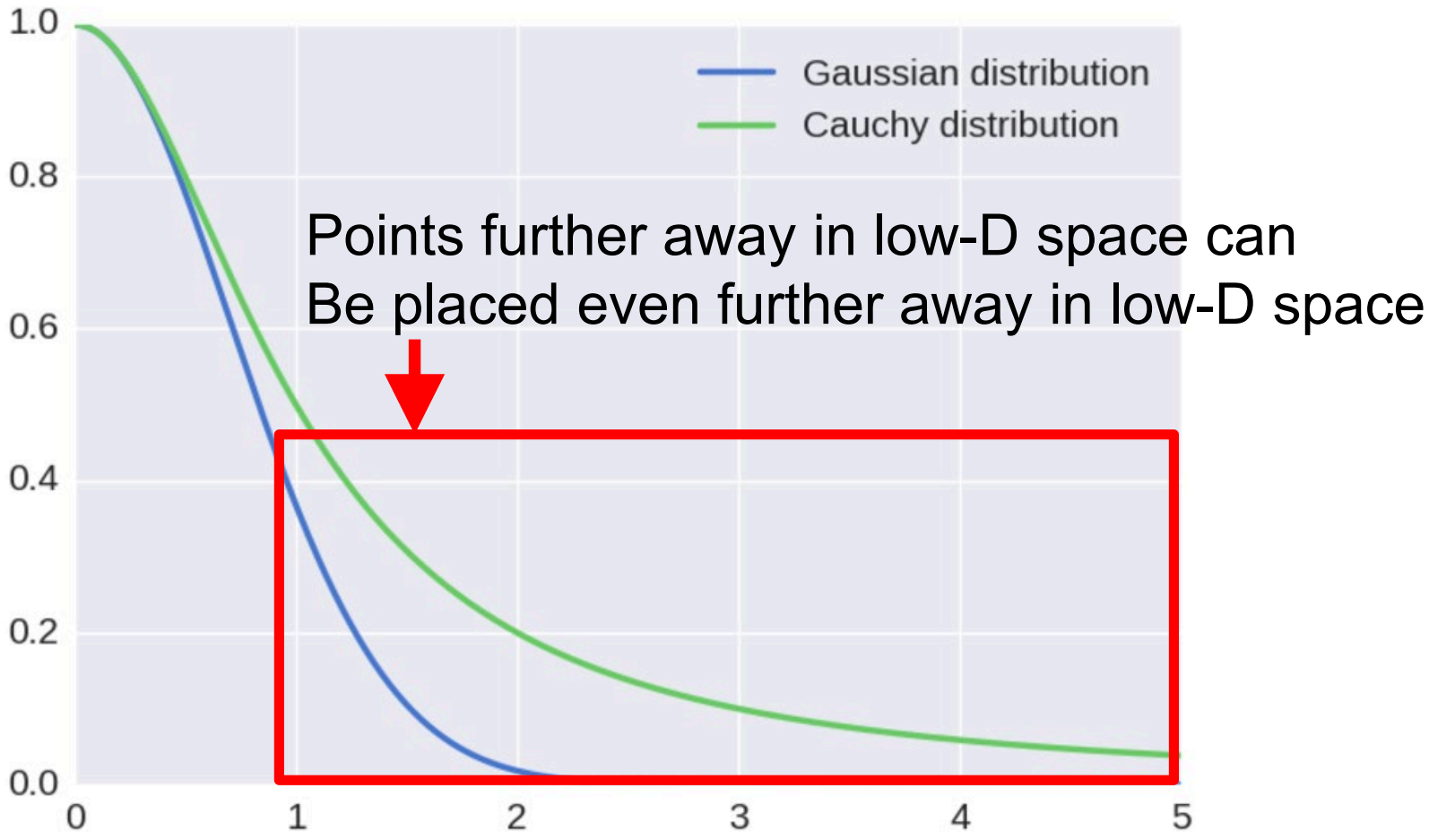
BioSB

t-SNE: Cauchy and Gaussian distribution



Points that are close
have the same dissimilarity
In both low-D and high-D spaces

t-SNE: Cauchy and Gaussian distribution

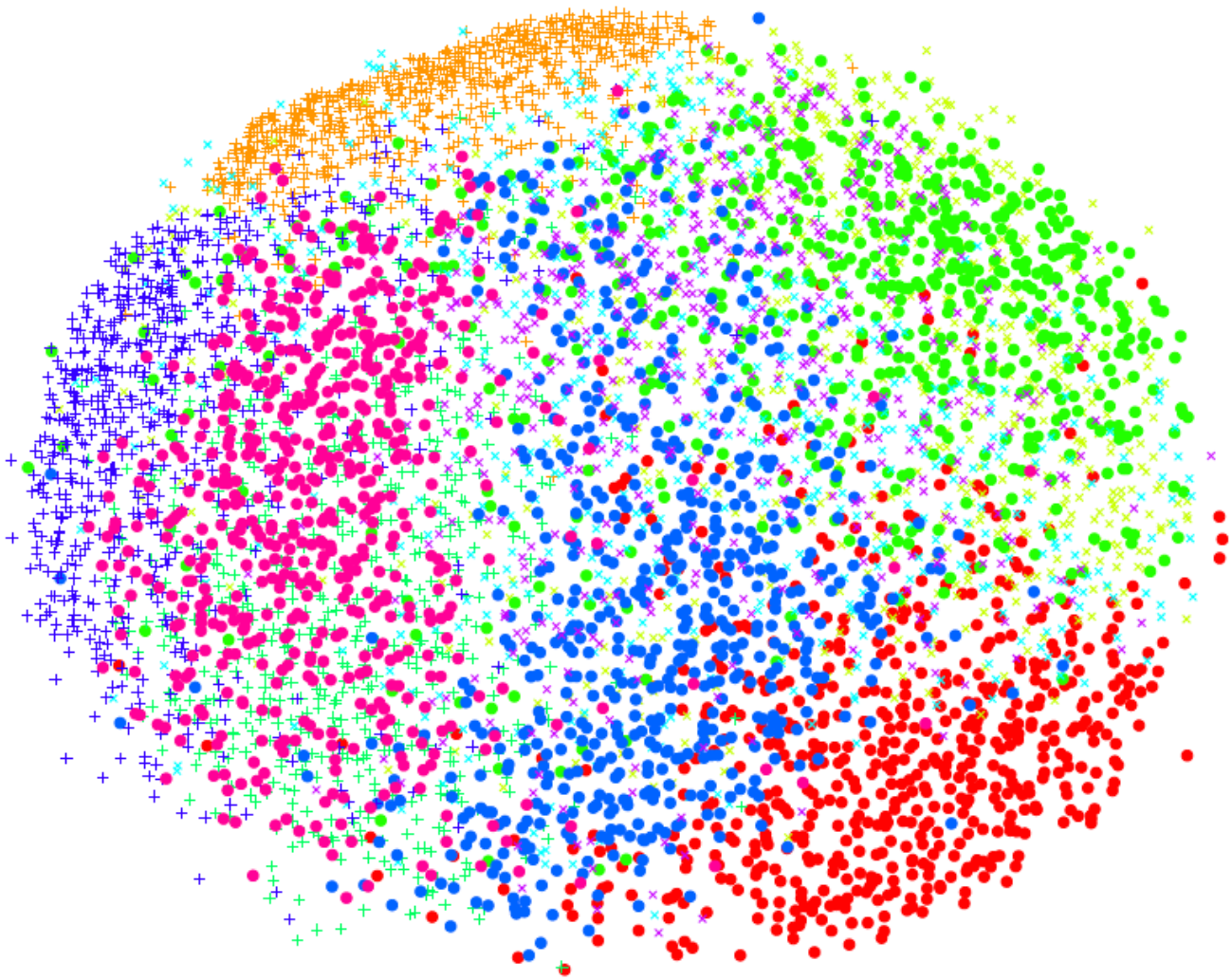


t-SNE (t-distributed stochastic neighbor embedding) (van der Maaten et al, 2008)

- In the input (high-D) space, X: compute dissimilarities between all pairs of points using a gaussian dissimilarity measure, pij
- In the output (low-D) space, Y: compute dissimilarities between all pairs of points using a t-distribution (with 1 d.o.f. (Cauchy)) dissimilarity measure, qij
- Minimize the Kullback-Leibler distance between these two distributions (P and Q)
- t-SNE faithfully retains small distances

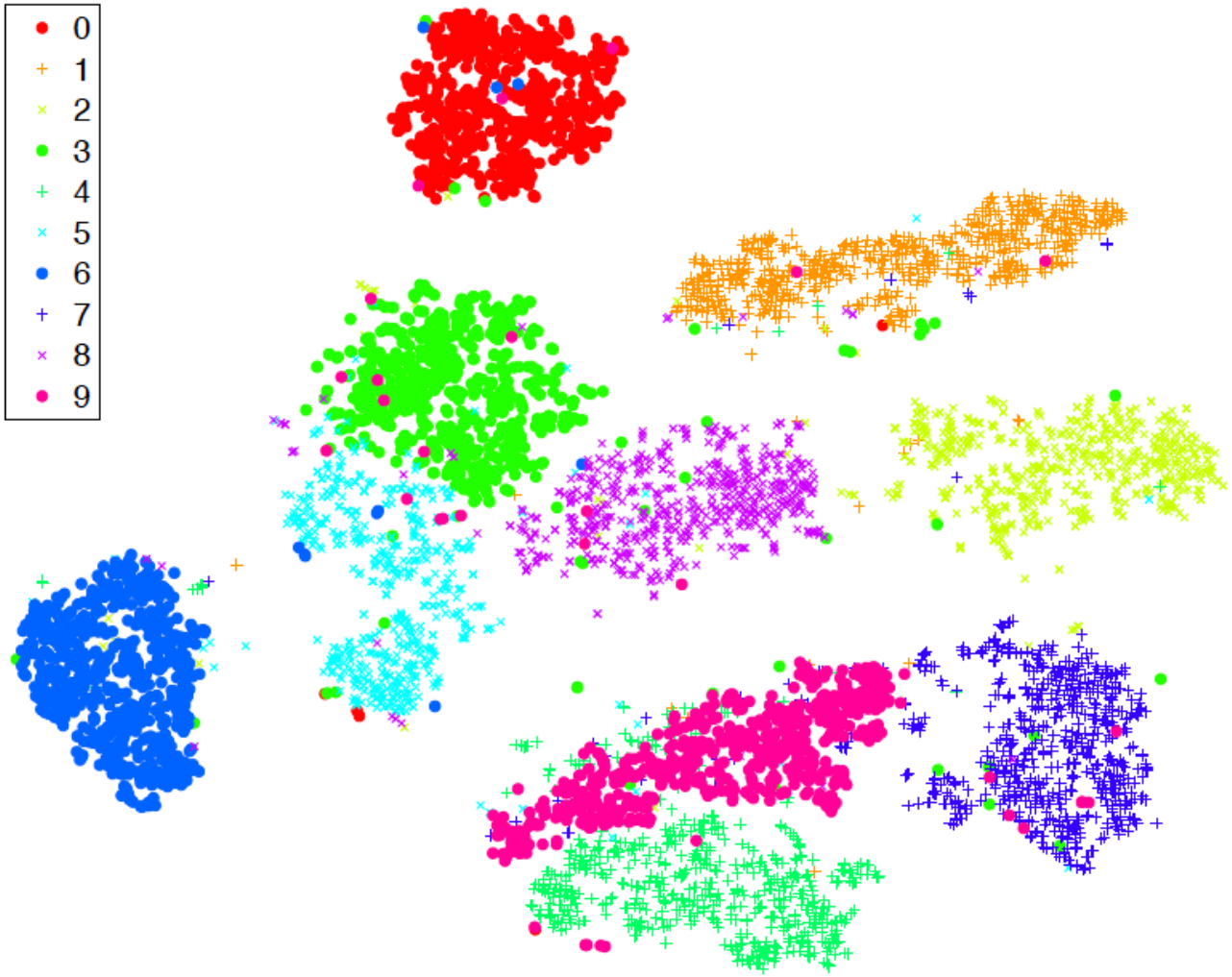
BioSB

t-SNE: Sammon map of digit data



BioSB

t-SNE: t-SNE map of digit data



MDS conclusions

- Experts or measurements give distances
- Optimise a *stress-function* (MDS) or KL distance (t-SNE)
- Important:
  - *the distance measure used:* is it representative?
  - *the weighting of distances (q):* can influence outcome heavily.
  - t-SNE run with defaults is quite reliable
- Largest risk: seeing structure in the data that is not really there
- Remaining problem: embedding new data points
- t-SNE (and now UMAP) are modern techniques to perform representation of data in high-D space in 2D

Feature selection

- For feature selection, we need:
  - A **criterion** function
    e.g. error, class overlap, information loss
  - A **search algorithm**
    e.g. pick the best single feature at each time

BioSB

Criteria

1. **Wrapper**: exact performance measure
    - base performance estimate on classifier;
    - estimate performance using cross-validation:
    - very expensive!

BioSB

Criteria

1. **Wrapper:** direct performance measure
    - base performance estimate on classifier;
    - estimate performance using cross-validation:
    - very expensive!

Note:
we should never use the training set to calculate performance; this will give a biased estimate!

BioSB

Criteria

1. **Wrapper**: direct performance measure
   - base performance estimate on classifier;
   - estimate performance using cross-validation:
   - very expensive!

Note:
we should never use the training set to calculate performance; this will give a biased estimate!

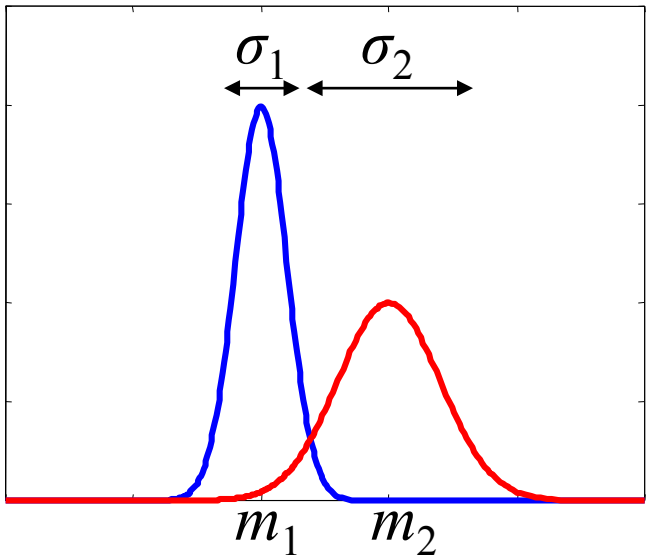
2. **Filter**: approximate performance predictors:
   - calculate the performance of an easy-to-use/'cheap' model
   - indication of how well a more powerful model may perform
   - is much faster to compute.

BioSB

Criteria (2)

- Example
  - Simple measure of the 'separability' of classes given a feature
  - 1D case: Signal-to-Noise Ratio (SNR) or Fisher criterion:

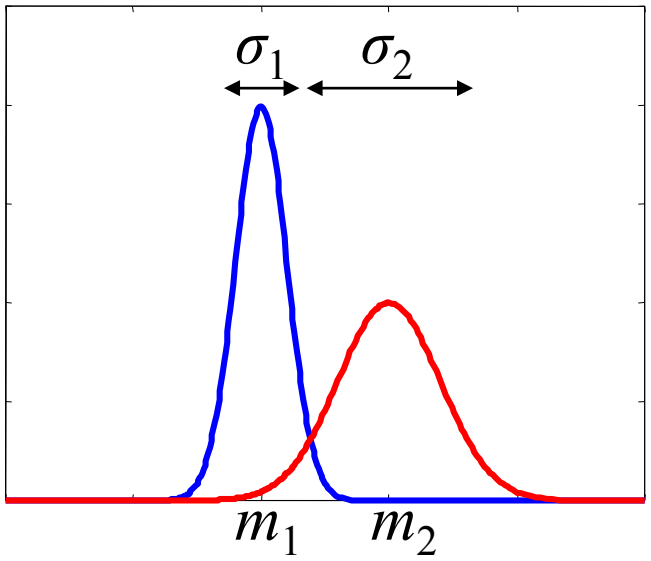
$$J_F = \frac{\left|m_1 - m_2\right|^2}{\left(\sigma_1^2 + \sigma_2^2\right)}$$



Criteria (3)

- Example
  - Simple measure of the 'separability' of classes given a feature
  - 1D case: Signal-to-Noise Ratio (SNR) or Fisher criterion:

$$J_F = \frac{\left|m_1 - m_2\right|^2}{\left(\sigma_1^2 + \sigma_2^2\right)}$$



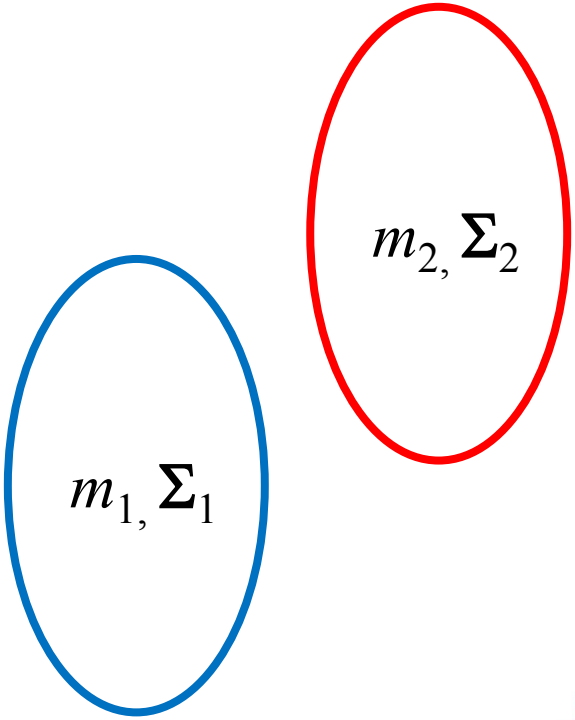
  - If J_F is large: good separability
  - If J_F is small: poor separability

Criteria (4)

- The multi-variate equivalent of the Fisher criterion is the
- Mahalanobis distance:
  - assumes
    - Gaussian distributions with
    - **equal** covariance matrix Σ:

$$D_M = (m_1 - m_2)^T \Sigma^{-1} (m_1 - m_2)$$

$$\boldsymbol{\Sigma} = \sum_{i=1}^{C} \frac{n_i}{n} \boldsymbol{\Sigma}_i$$



Search algorithms

- **Feature selection**: select a subset of d out of p features which optimises the criterion

- Simplest solution: look at all possible subsets

- Problem: there are $\binom{p}{d} = \frac{p!}{(p-d)!d!}$ subsets

- *e.g.* $p = 50$ features,

  $d = 2$: 1225 subsets

  $d = 5$: 2.1×10^6 subsets

  $d = 25$: 1.3×10^{14} subsets

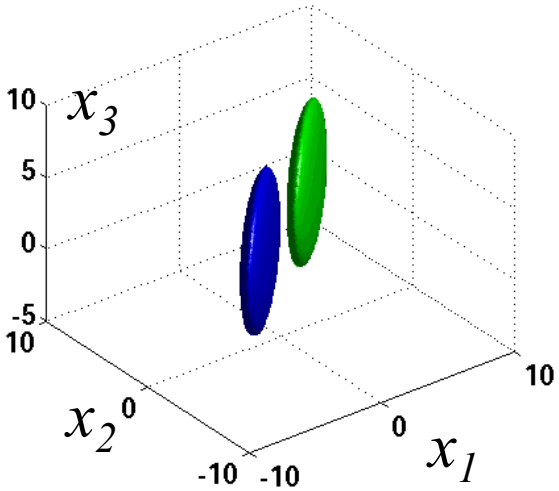
Search algorithms (2)

- Sub-optimal algorithms: select or deselect one feature (or a few features) at a time

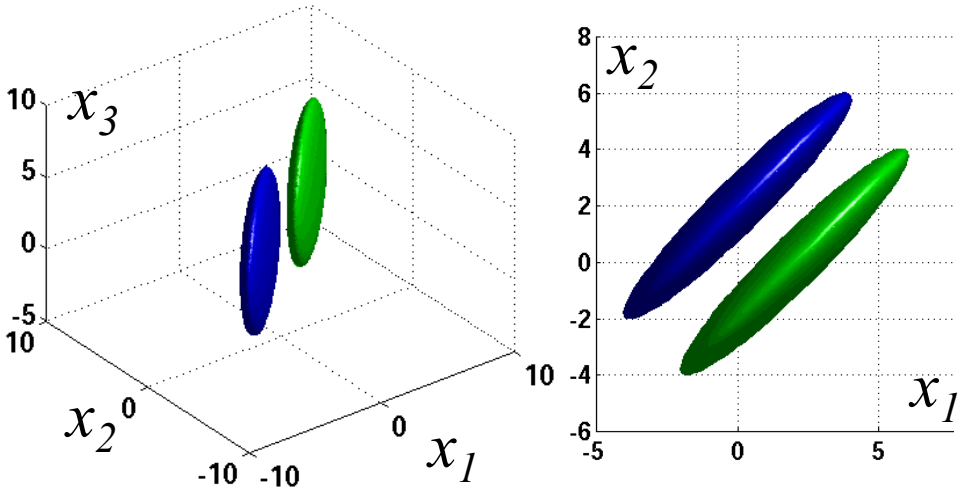
- Simplest: best individual d
  but these are not necessarily the best d !

- Demonstration: two Gaussians;
  select 2 features out of 3 for classification

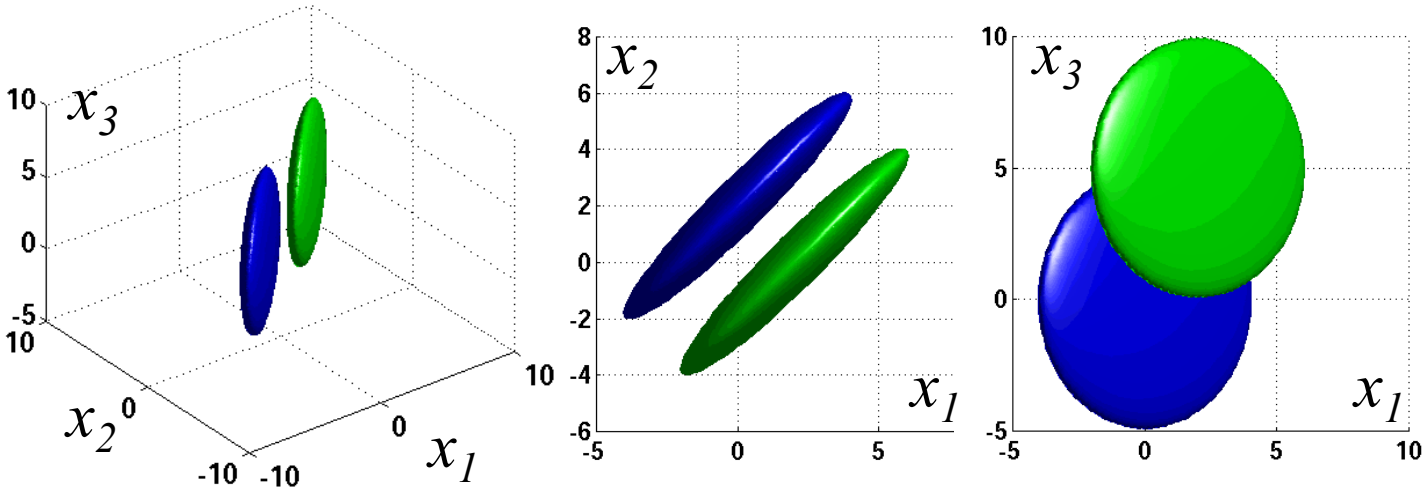
Search algorithms (3)



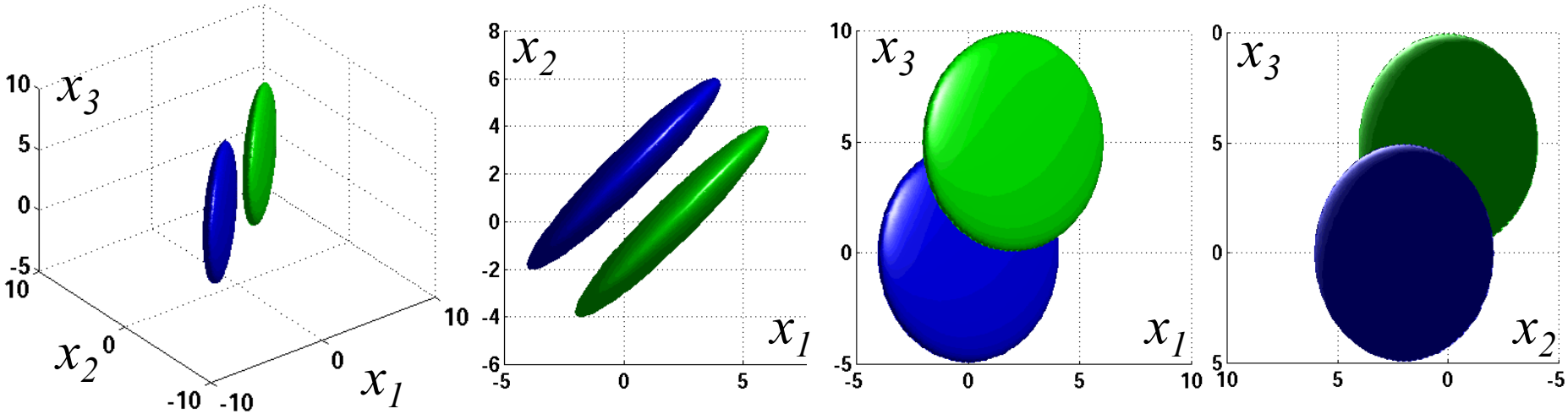
Search algorithms (3)



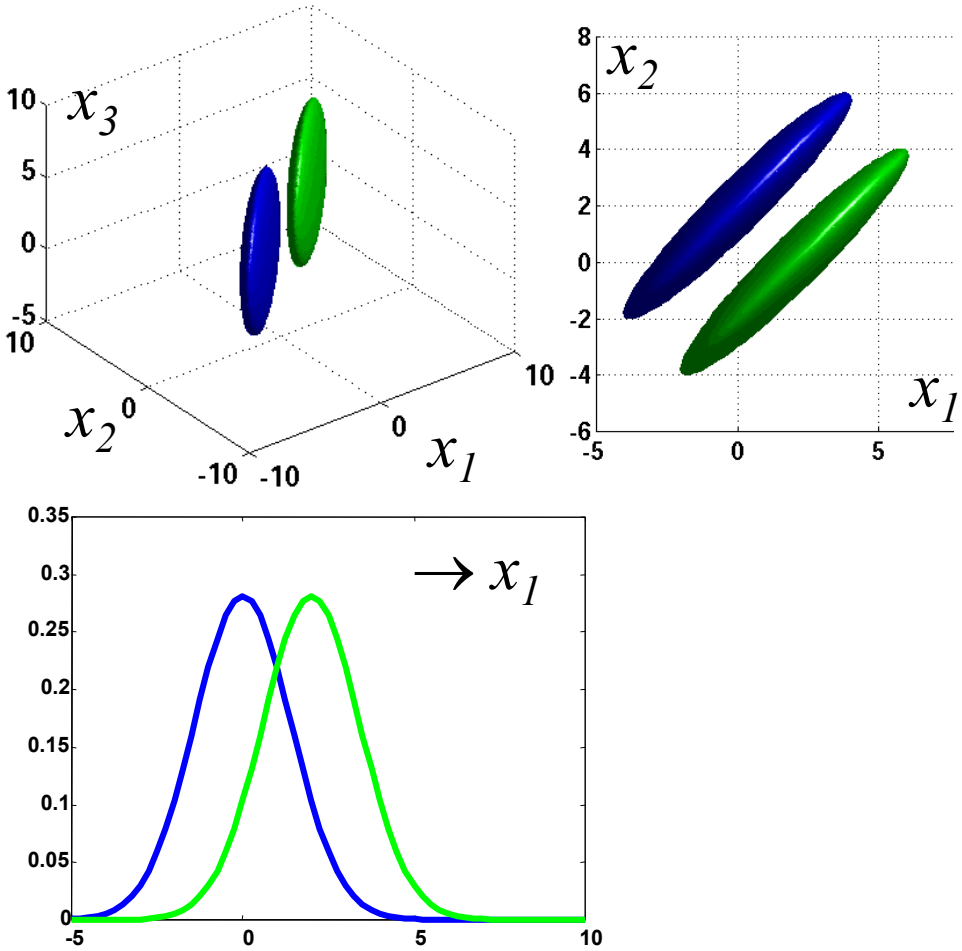
Search algorithms (3)



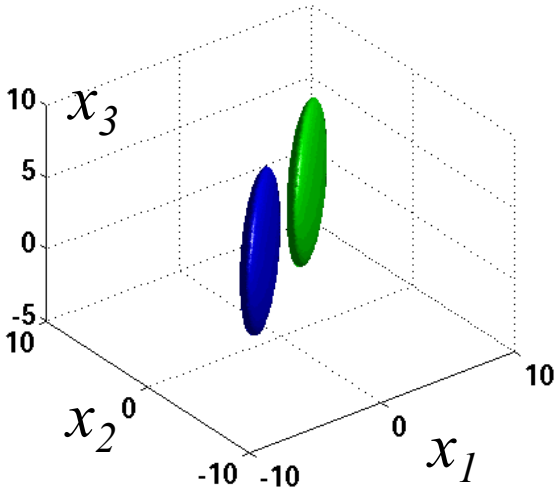
Search algorithms (3)

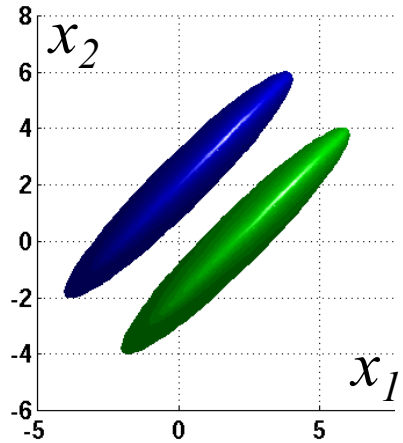


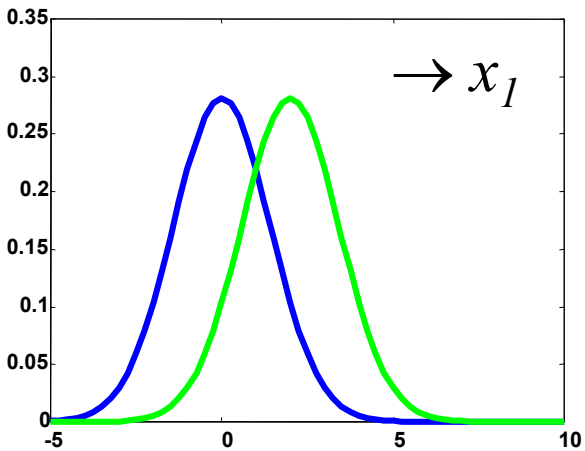
Search algorithms (3)

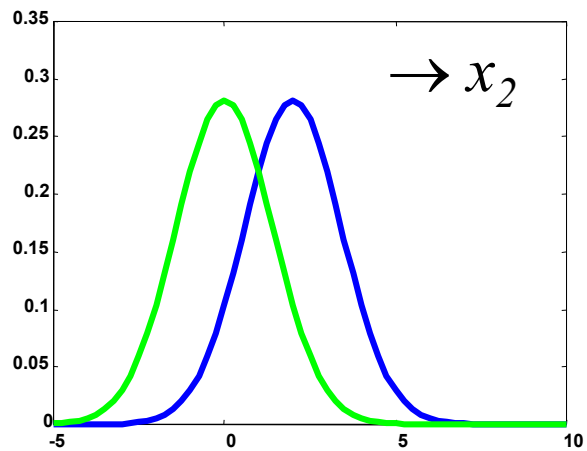


Search algorithms (3)

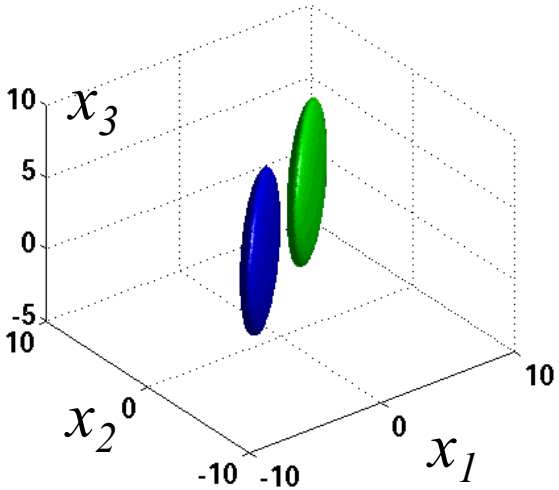


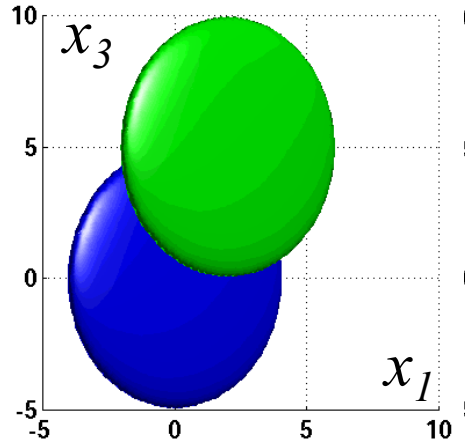


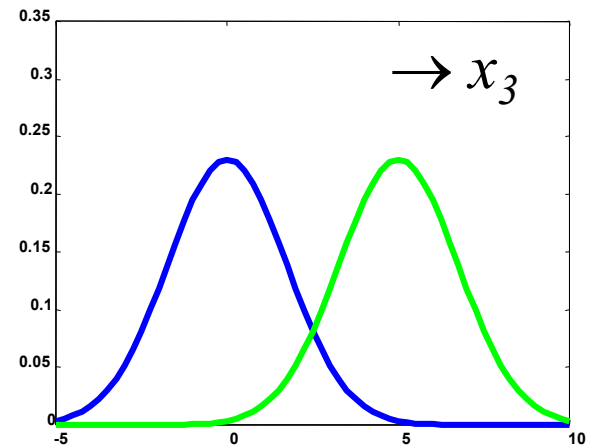




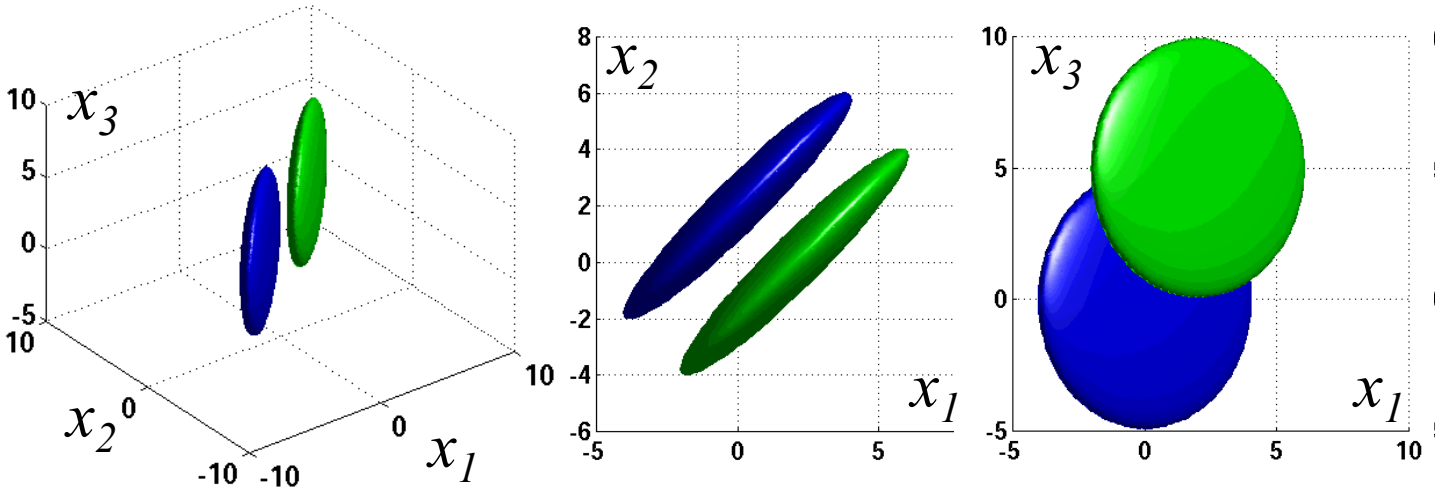
Search algorithms (3)

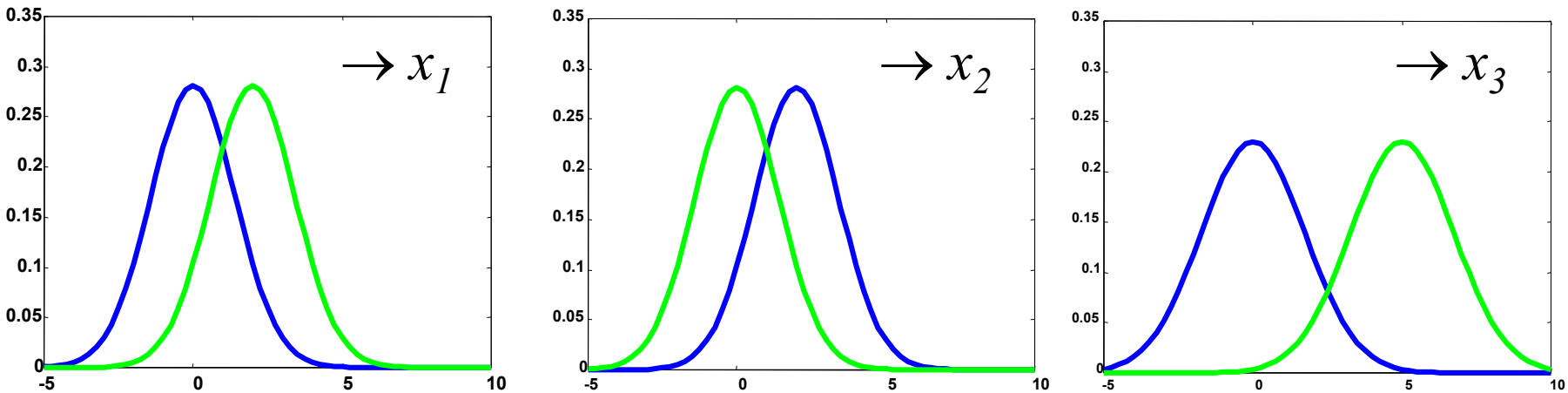






Search algorithms (3)





Search algorithms (4)

- Other sub-optimal algorithms:
  - Forward selection (for when d is low)
    - start with empty set
    - keep adding one feature at a time
      so that the entire subset so far performs best

Search algorithms (4)

- Other sub-optimal algorithms:
  - Forward selection (for when d is low)
    - start with empty set
    - keep adding one feature at a time
      so that the entire subset so far performs best
  - Backward selection (for when d is high)
    - start with entire set
    - keep removing one feature at a time
      so that the entire subset so far performs best

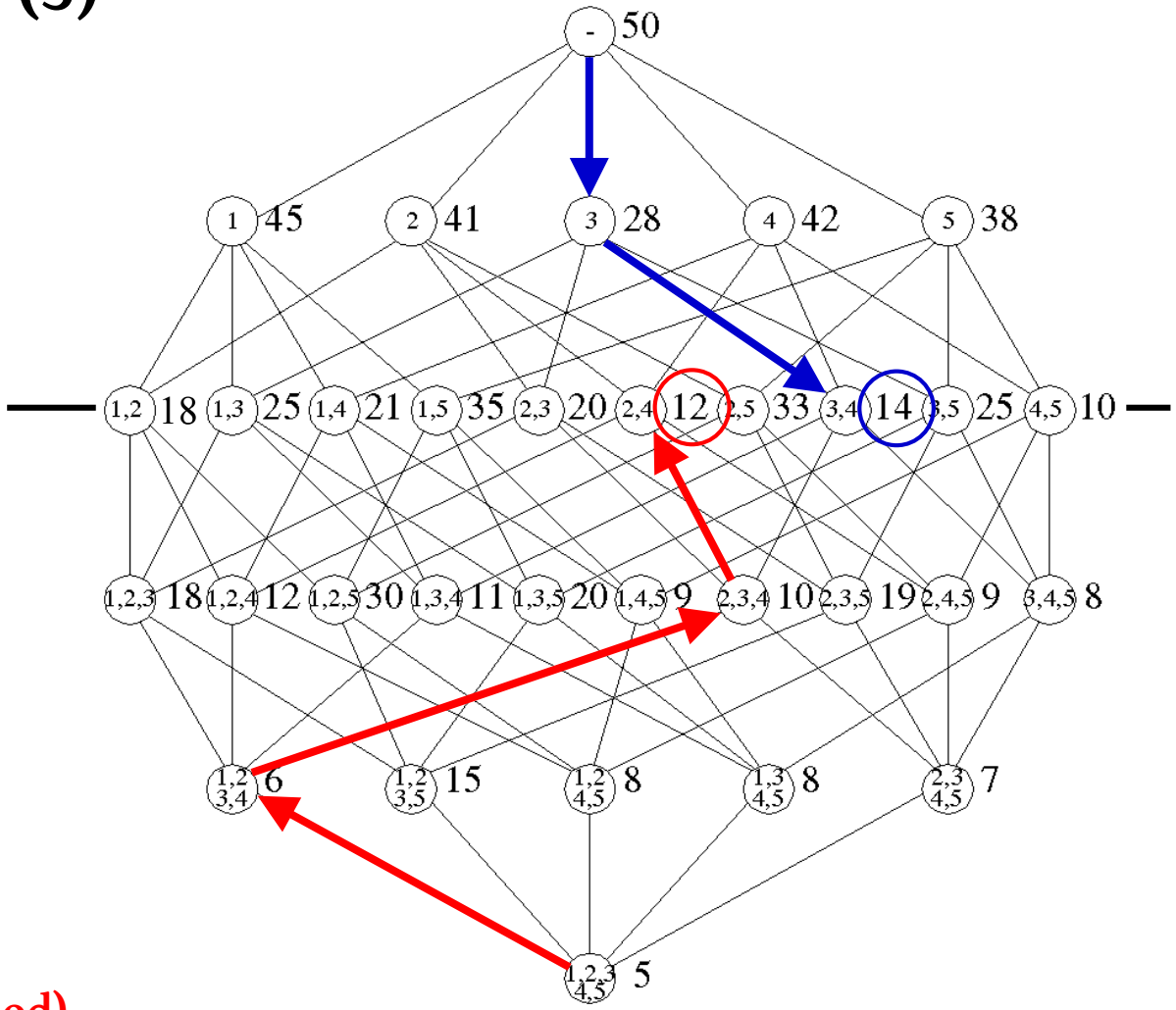
BioSB

Search algorithms (4)

- Other sub-optimal algorithms:
  - Forward selection (for when d is low)
    - start with empty set
    - keep adding one feature at a time
      so that the entire subset so far performs best
  - Backward selection (for when d is high)
    - start with entire set
    - keep removing one feature at a time
      so that the entire subset so far performs best
  - Plus-l-takeaway-r (may be slightly better)
    - start with empty set (if $l > r$) or entire set (if $l < r$)
    - keep adding best l and removing worst r

Search algorithms (5)

- Select $d = 2$ out of $p = 5$ features
- Sub-optimality illustrated:
  - forward
  - backward



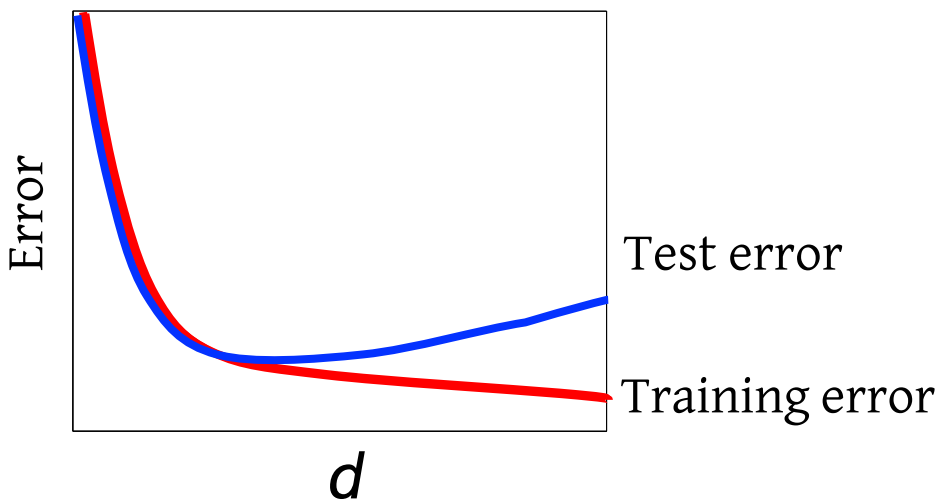
Feature subset

Criterion value

(Low is good)

Search algorithms (8)

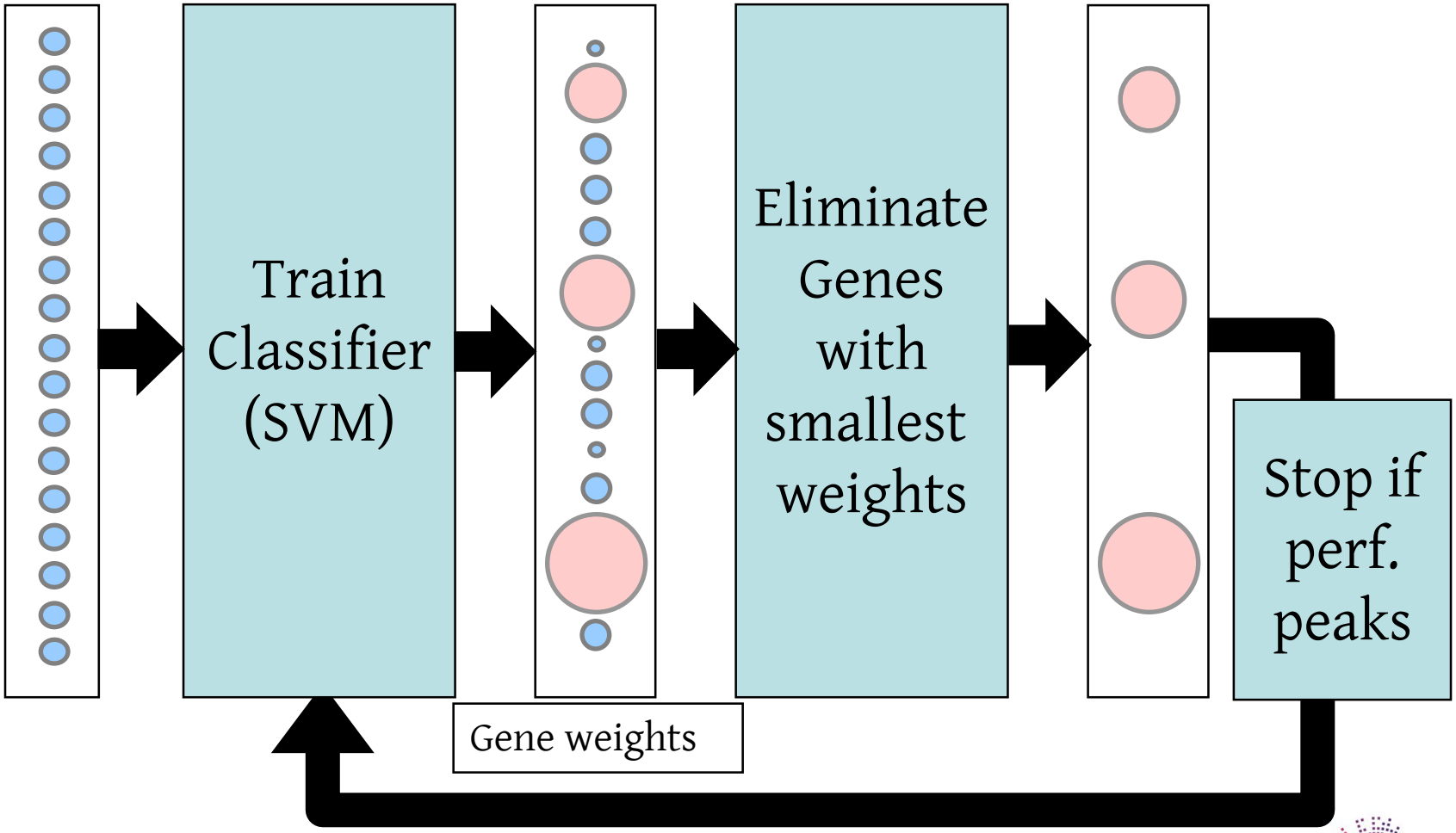
- When should we stop?
  - Due to estimation problems (*e.g.* covariance matrix), we may be overtraining on training set
  - This is revealed by increasing error on the test set



  - Otherwise (with very large sample sizes), we will have to specify a desired number of measurements

Example: Recursive feature elimination (RFE)

Wrapper, Backward search

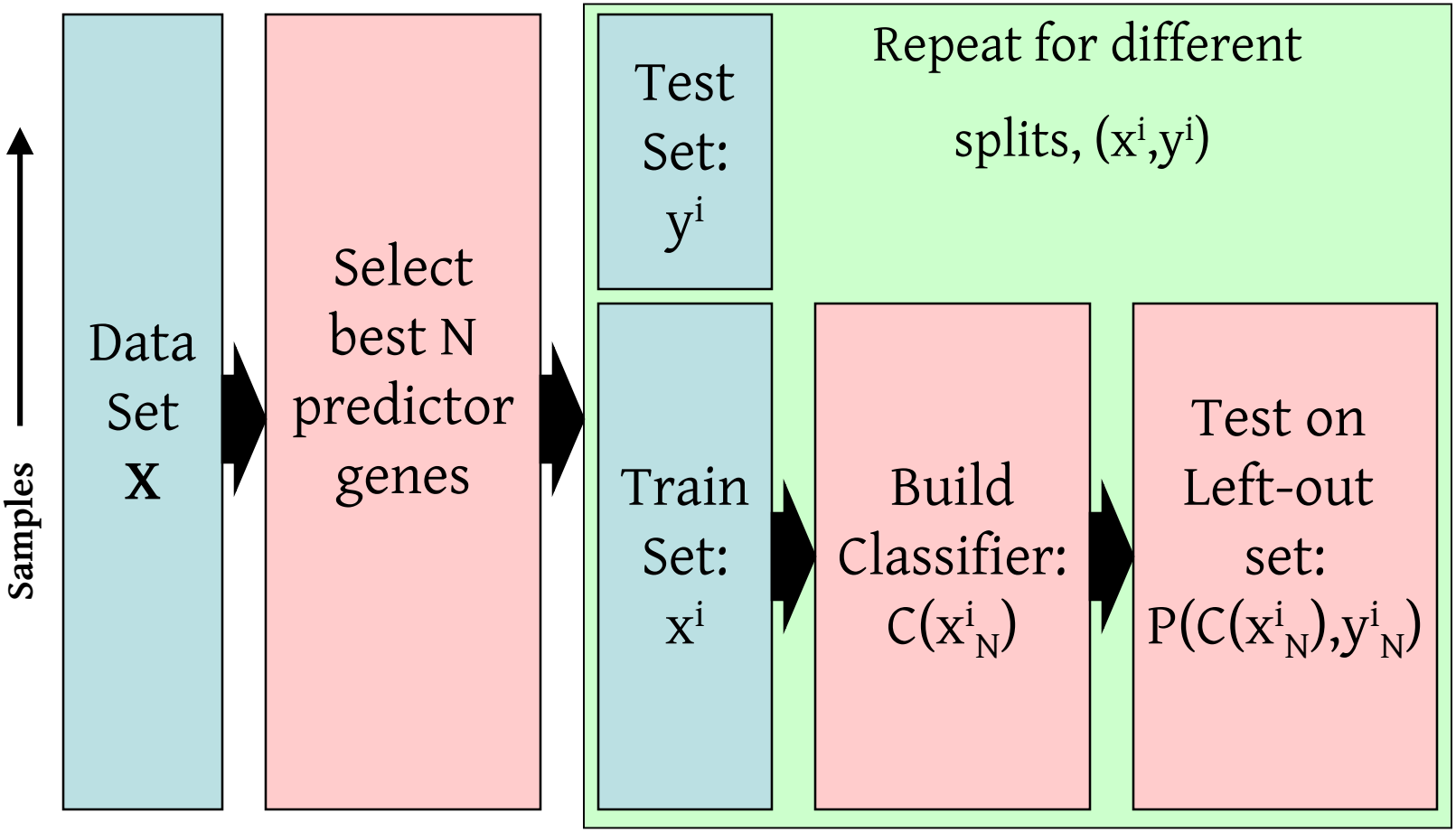


What can go wrong?

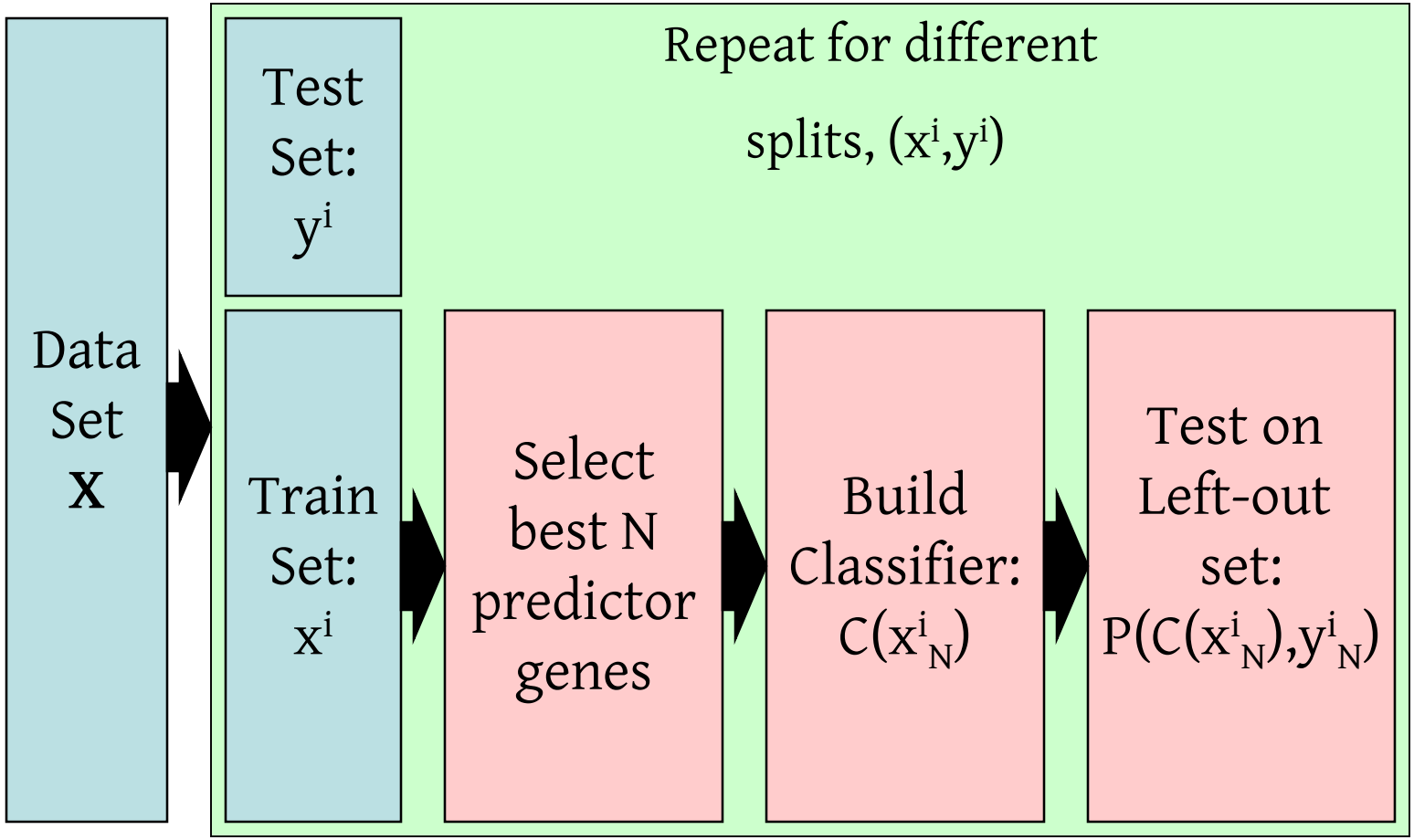
Selection bias...

- Guyon et al. (2002). Machine Learning **46**, 389 – 422.
- Ambroise and McLachlan (2002). PNAS **99**, 6562-6566.

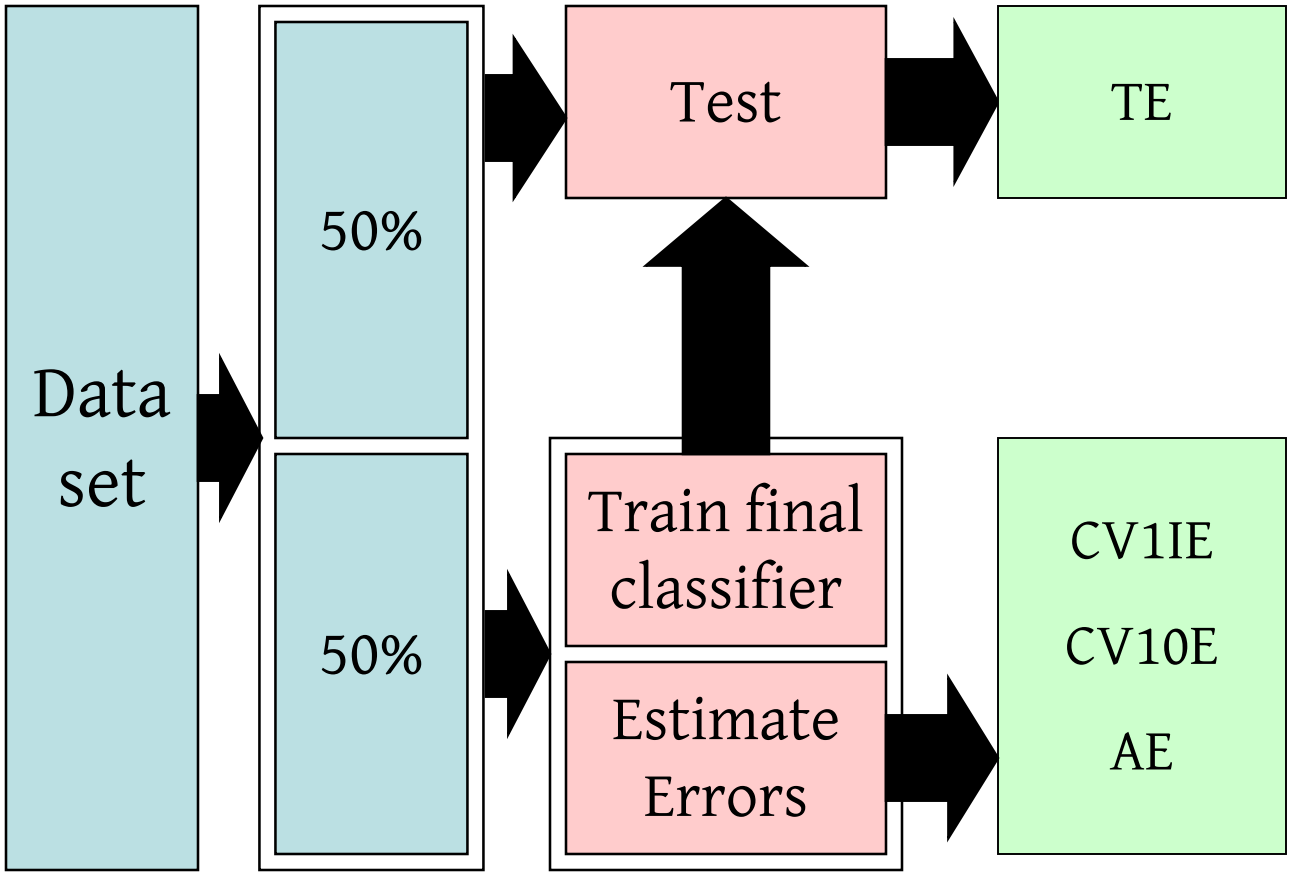
Biased selection



Unbiased selection

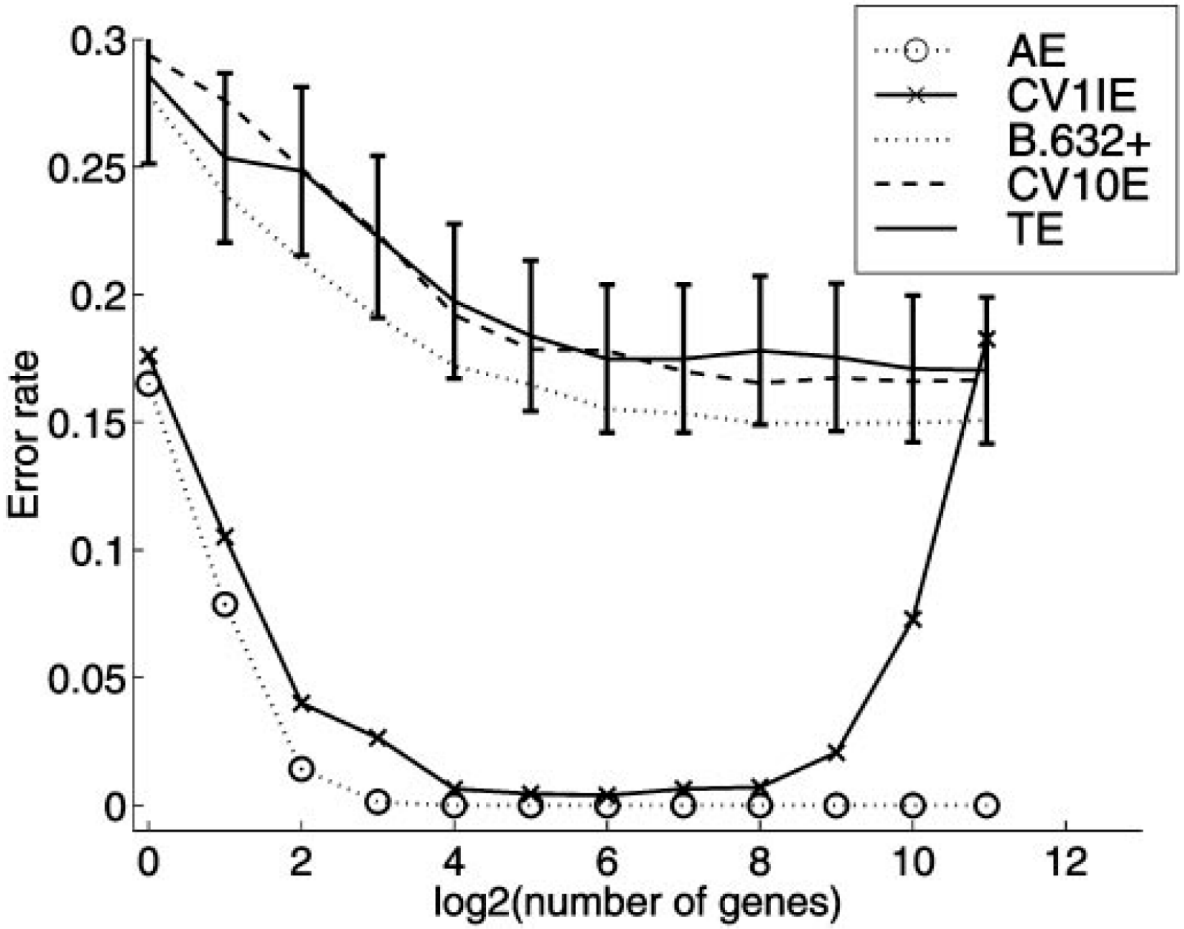


Ambroise & McLachlan experiments



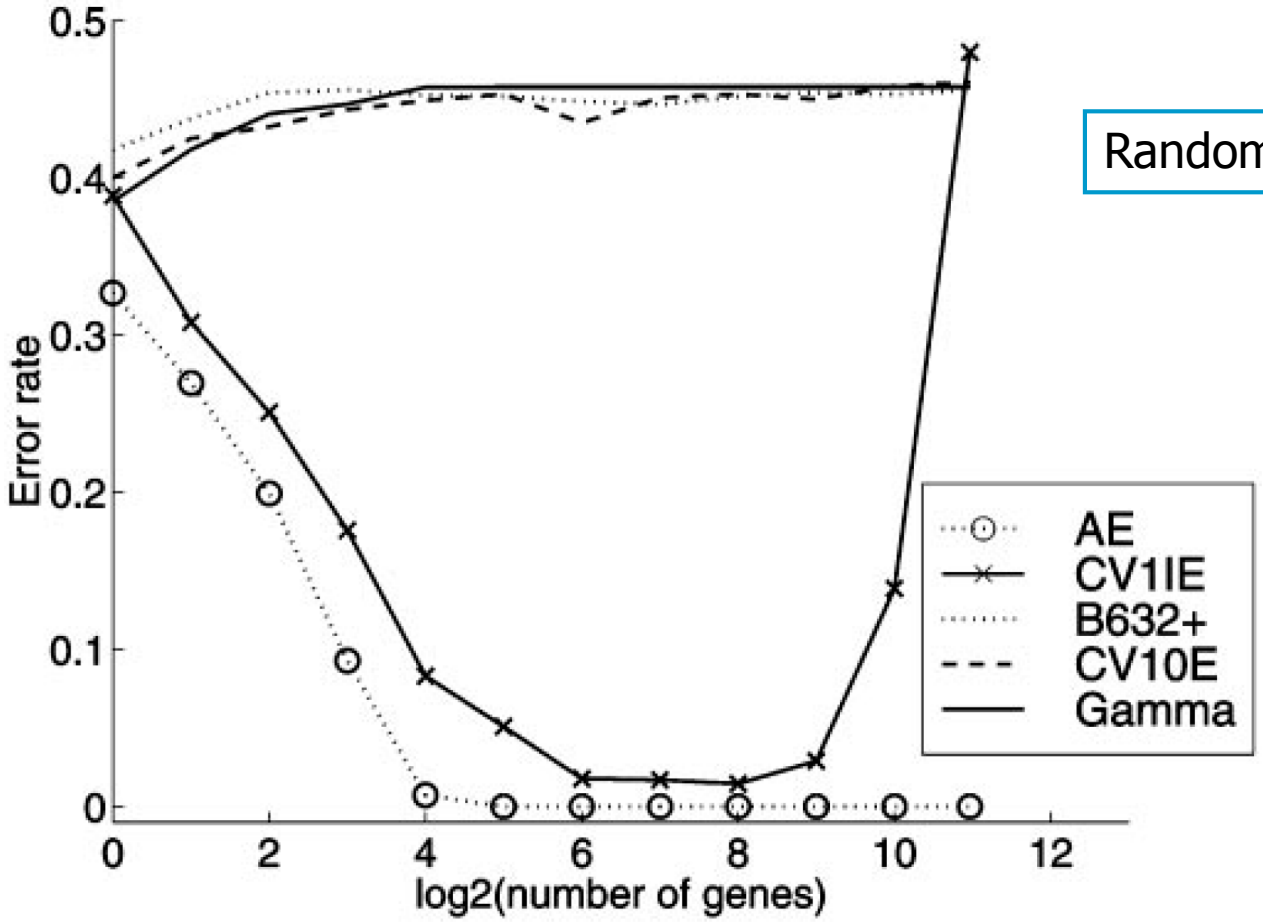
Ambroise & McLachlan experiments

Colon vs. normal data



Ambroise & McLachlan experiments

Random data



Random data

Cross-validation

- Remember:

Note:
we should never use the training set to calculate performance; this will give a biased estimate!

  - for small sample size: use cross-validation
- Cross-validation should be applied to *every choice* made, including:
  - the number of features to use
  - the features to use
  - the type of classifier to use
  - ...

BioSB

Feature selection: summary

- Feature selection can improve performance and help interpretation
- Requirements: a criterion and a search algorithm
- Methodology (cross-validation) is very important, especially for RNAseq data ('p >> n')
- There seems to be some evidence that the simplest methods (individual selection) work best

BioSB

Shrinkage

- Feature selection: selects a subset of features (1/0)
- Feature extraction: combinations of features are constructed based on variance and accuracy arguments
- Regularization 1: control contribution of genes to classifier based on individual quality and control degree of contribution with cross-validated classification error
- Regularization 2: combines accuracy (error) and penalty on large weights (= simple models) in one criterion.

BioSB

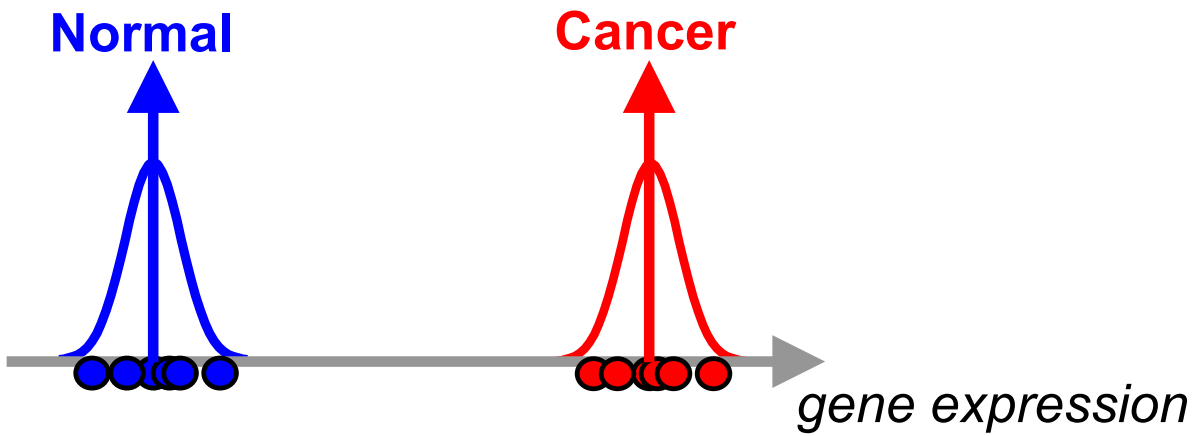
Shrunken centroids

- Same principle as forward filtering
- Genes are evaluated *individually*
- BUT, do not start with the best and keep adding;
- RATHER, start removing worst genes from the back
- In PAM* genes can participate 'partially', in forward filtering a gene is either 100% in or out.

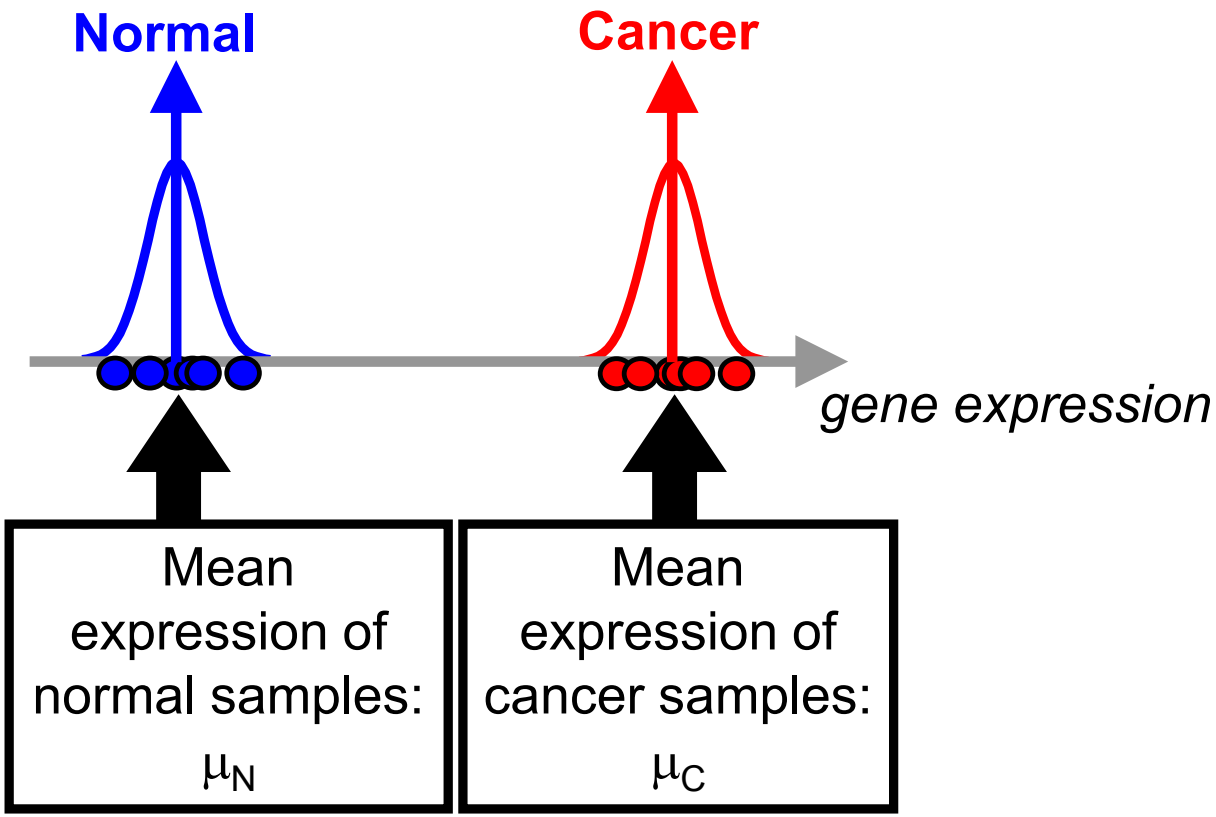
* PAM: Prediction analysis of micro-arrays; R. Tibshirani, T. Hastie, B. Narasimhan and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. PNAS 99(10):6567 6572, 2002.

BioSB

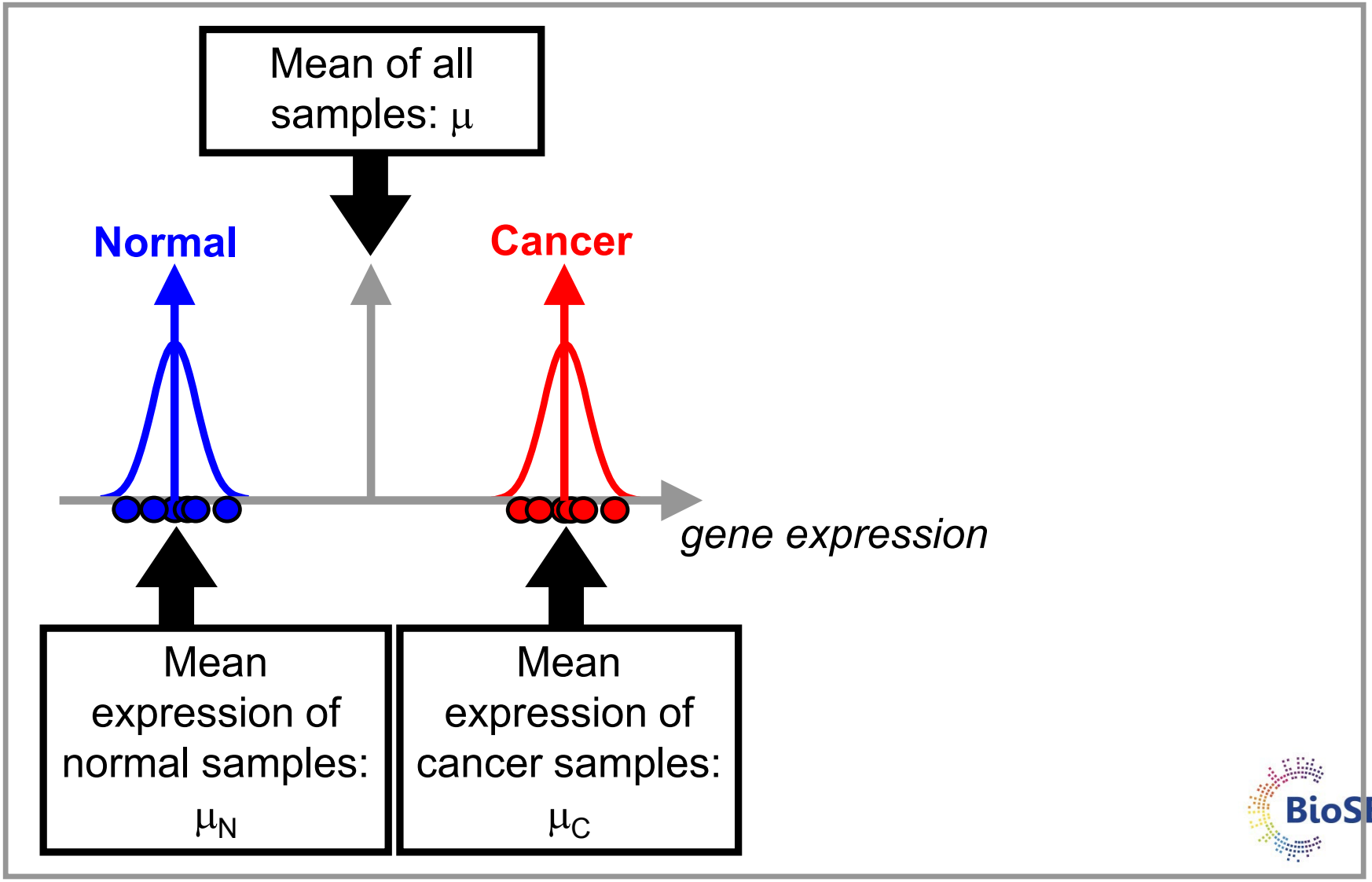
Shrunken centroids (1)
Step 1: Compute class centroids per gene



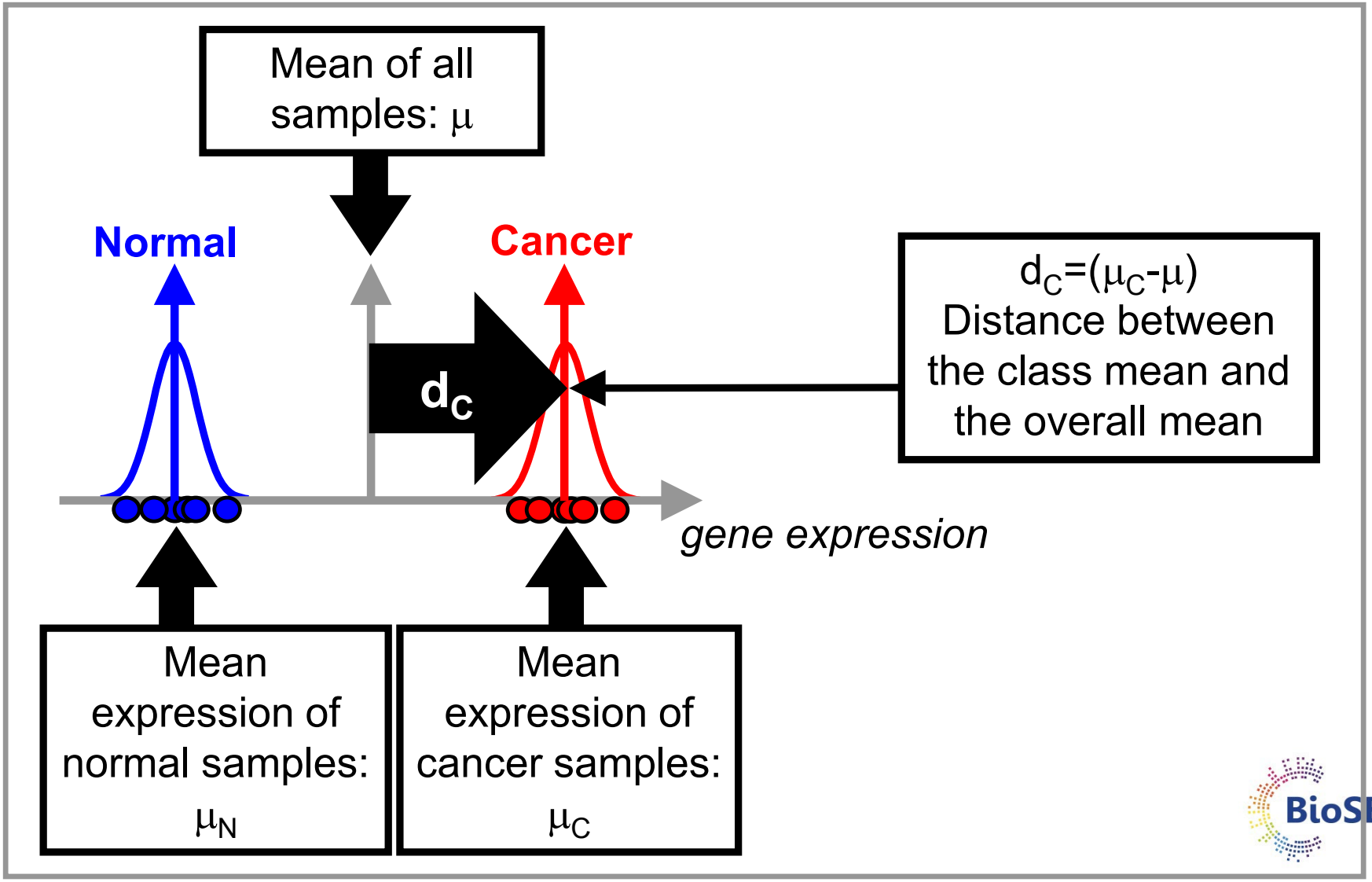
Shrunken centroids (2)
Step 1: Compute class centroids per gene



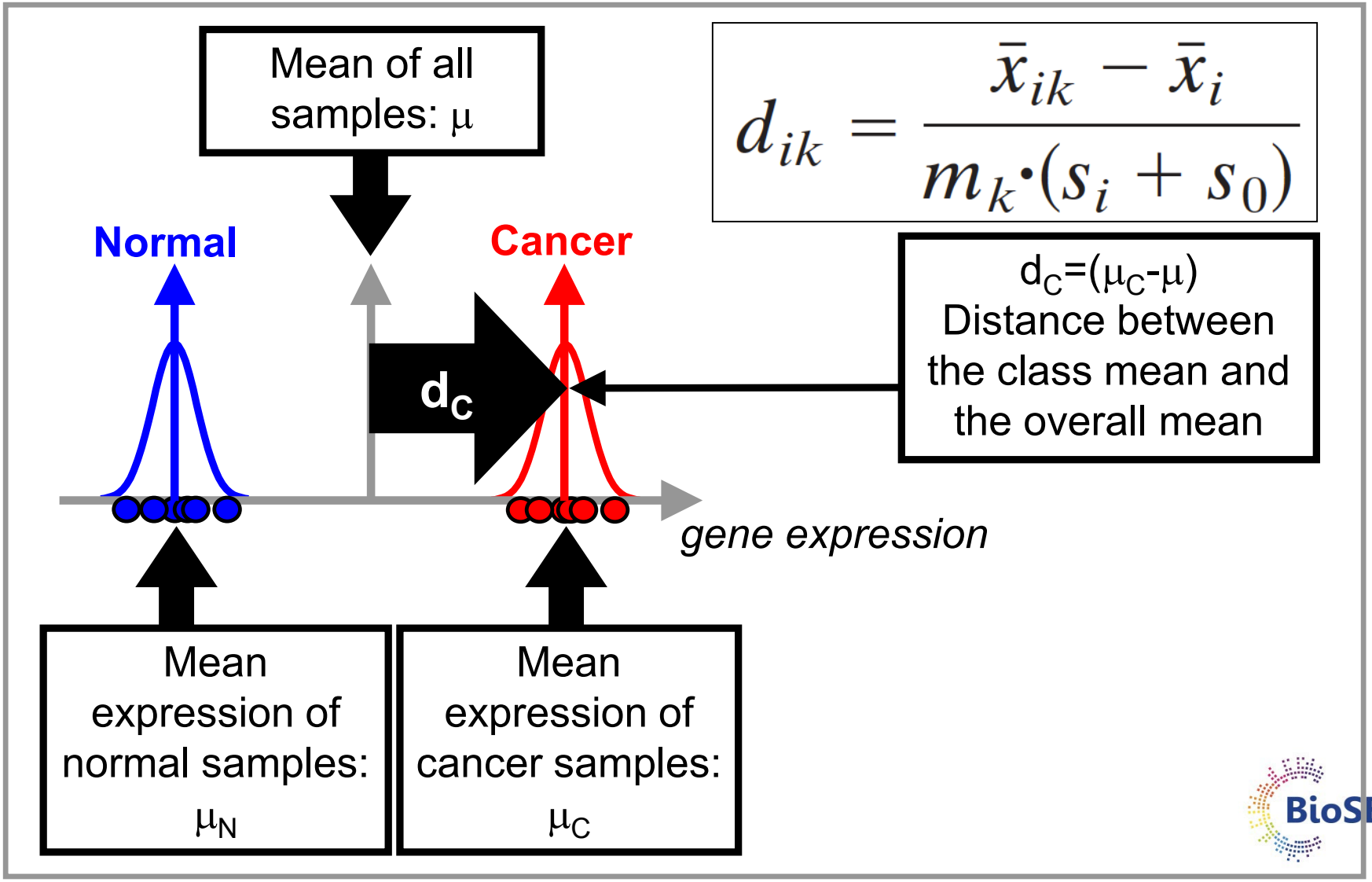
Shrunken centroids (3)
Step 2: Compute overall centroids per gene



Shrunken centroids (4)
Step 3: Compute d per gene

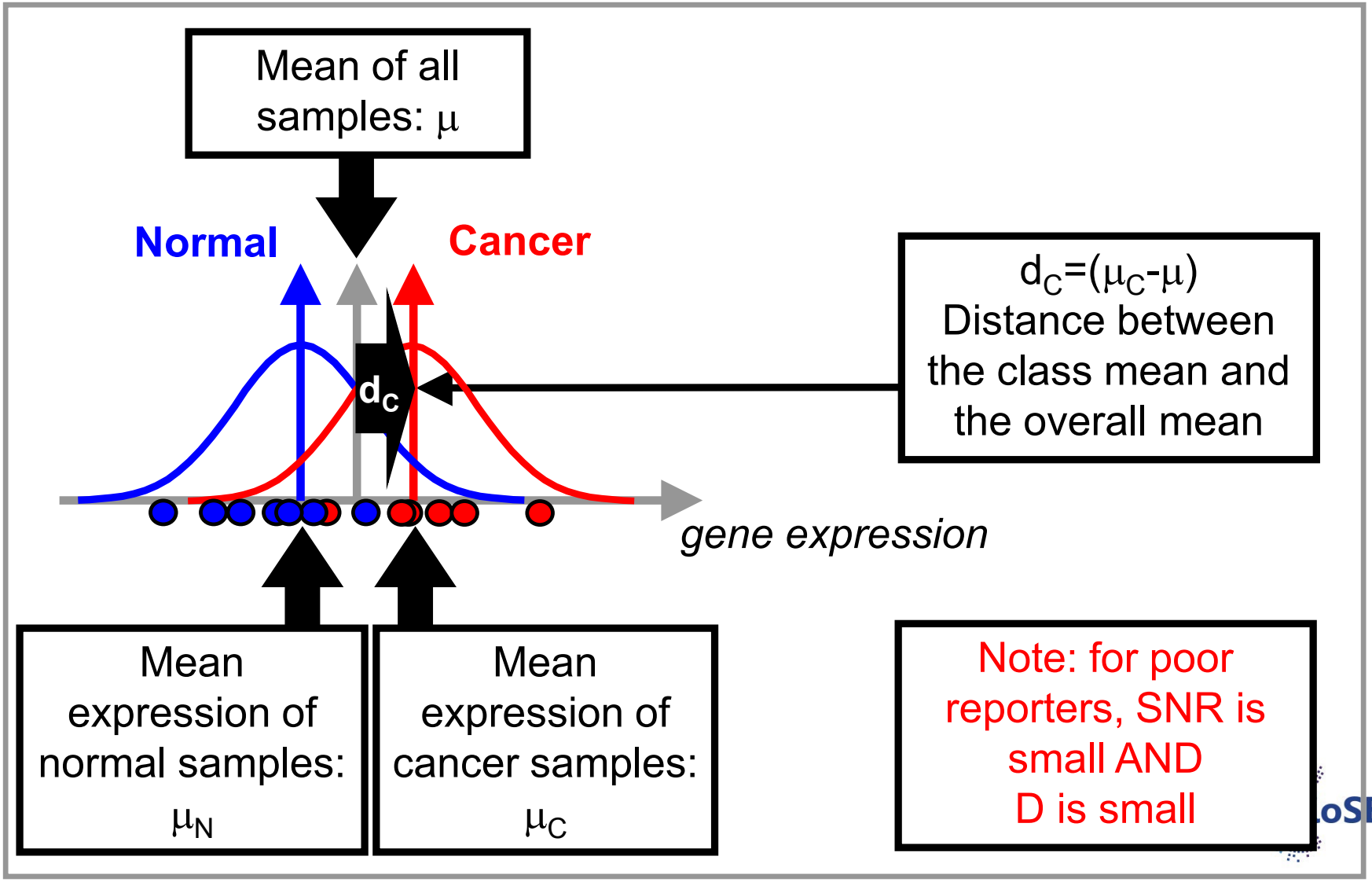


Shrunken centroids (4)
Step 3: Compute d per gene

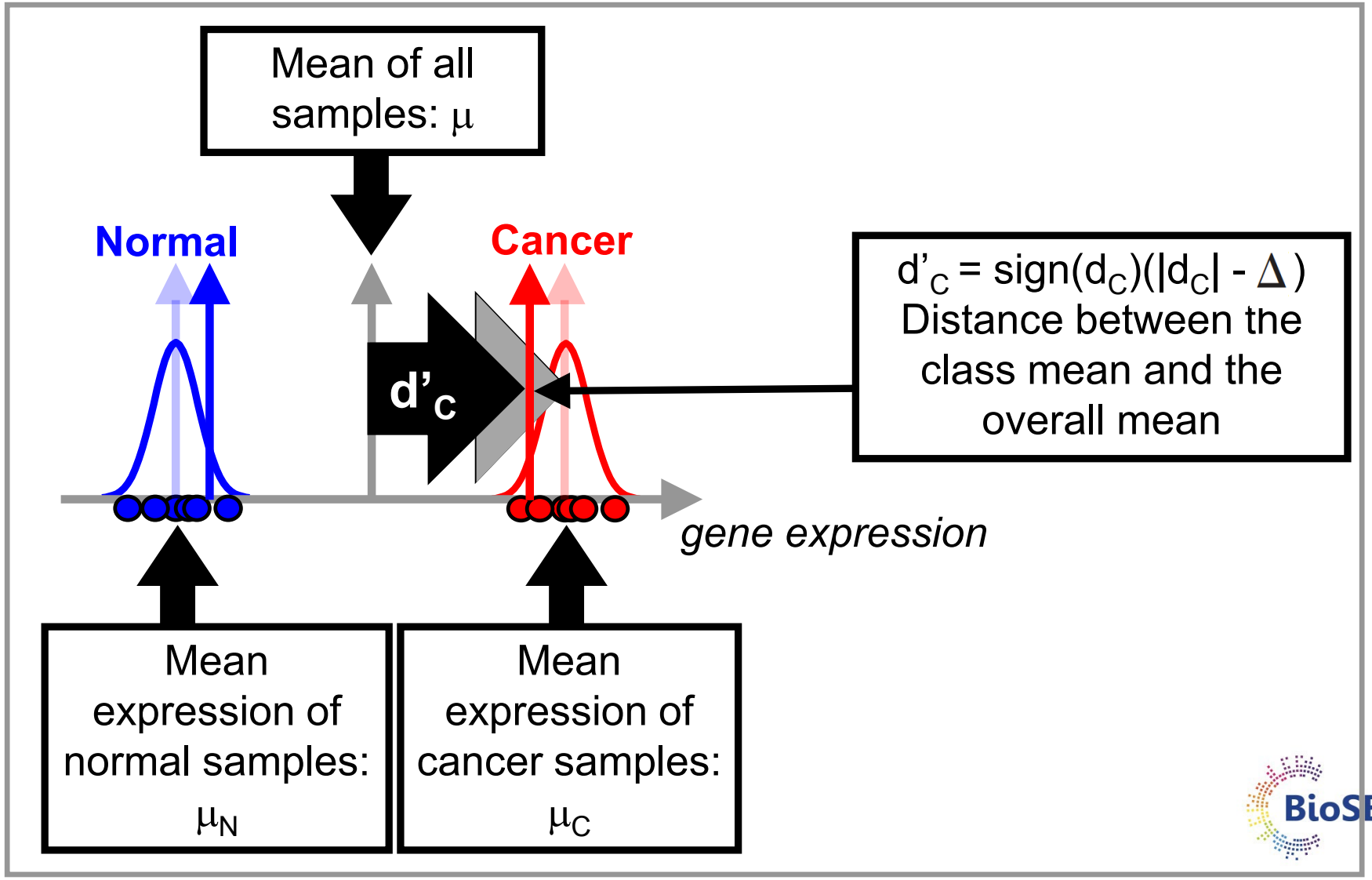


$$d_{ik} = \frac{\bar{x}_{ik} - \bar{x}_i}{m_k \cdot (s_i + s_0)}$$

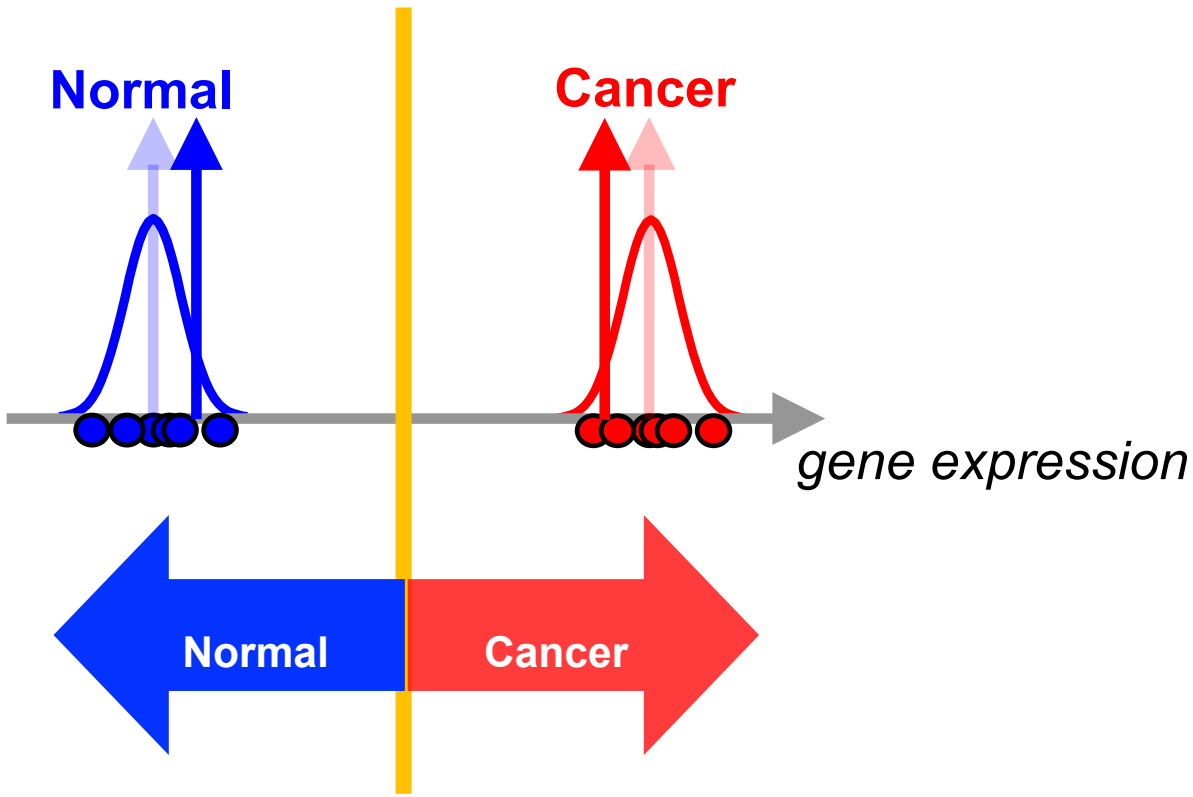
Shrunken centroids (5)
Step 3: Compute d per gene



Shrunken centroids (6)
Step 4: Shrink the centroids

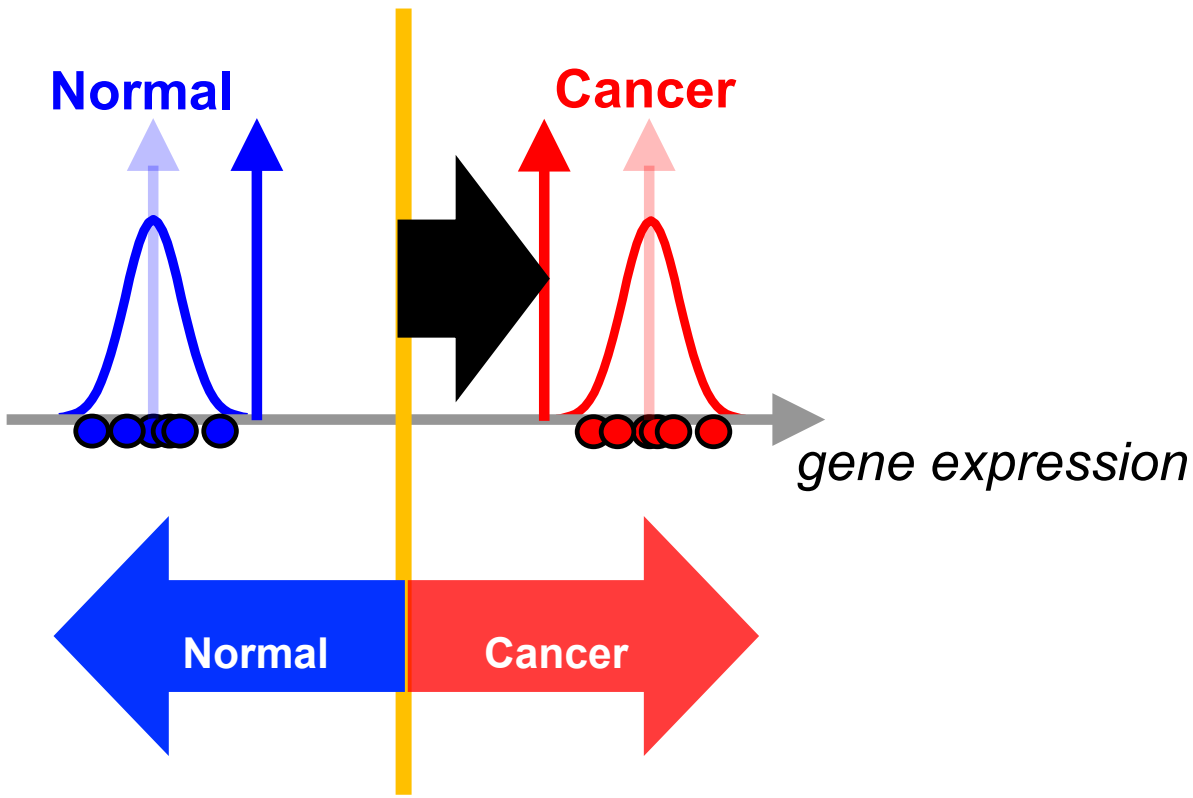


Shrunken centroids (7)
Step 5: Classify with shrunken centroids / perf.



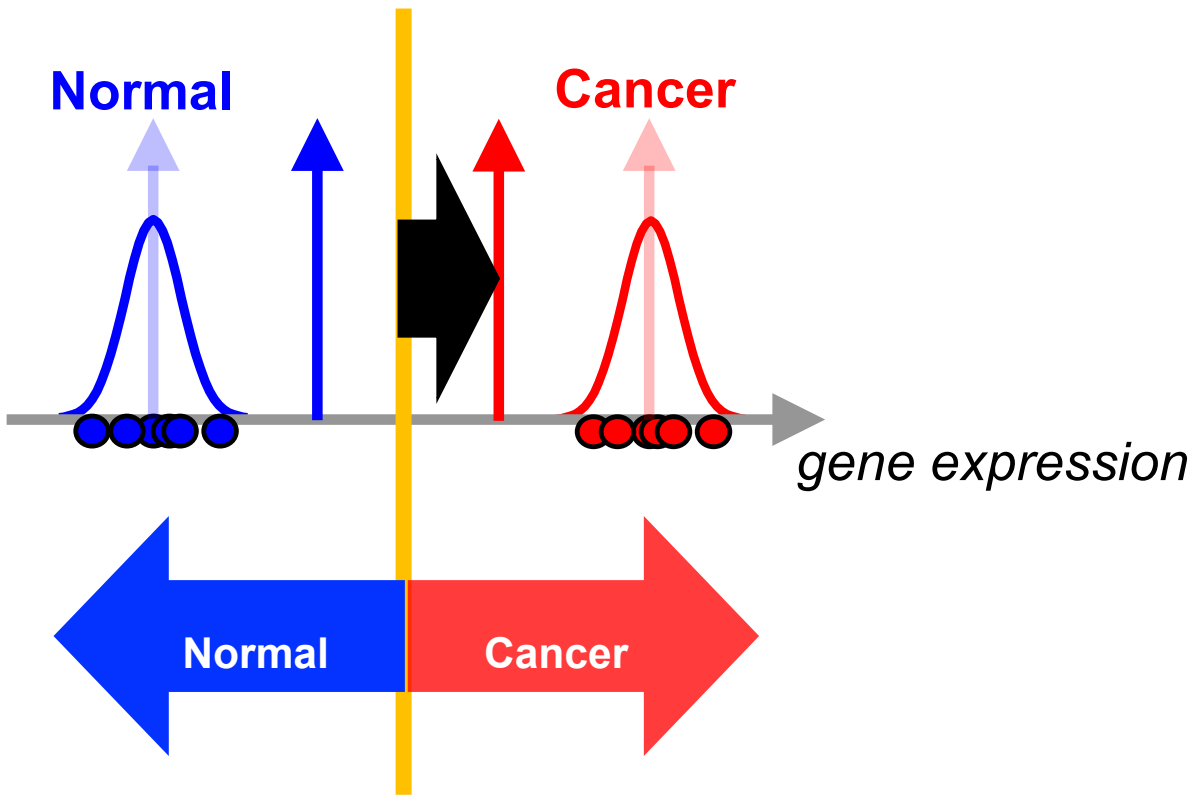
BioSB

Shrunken centroids (8)
Repeat shrinking and evaluation of classifier till all genes shrunk away



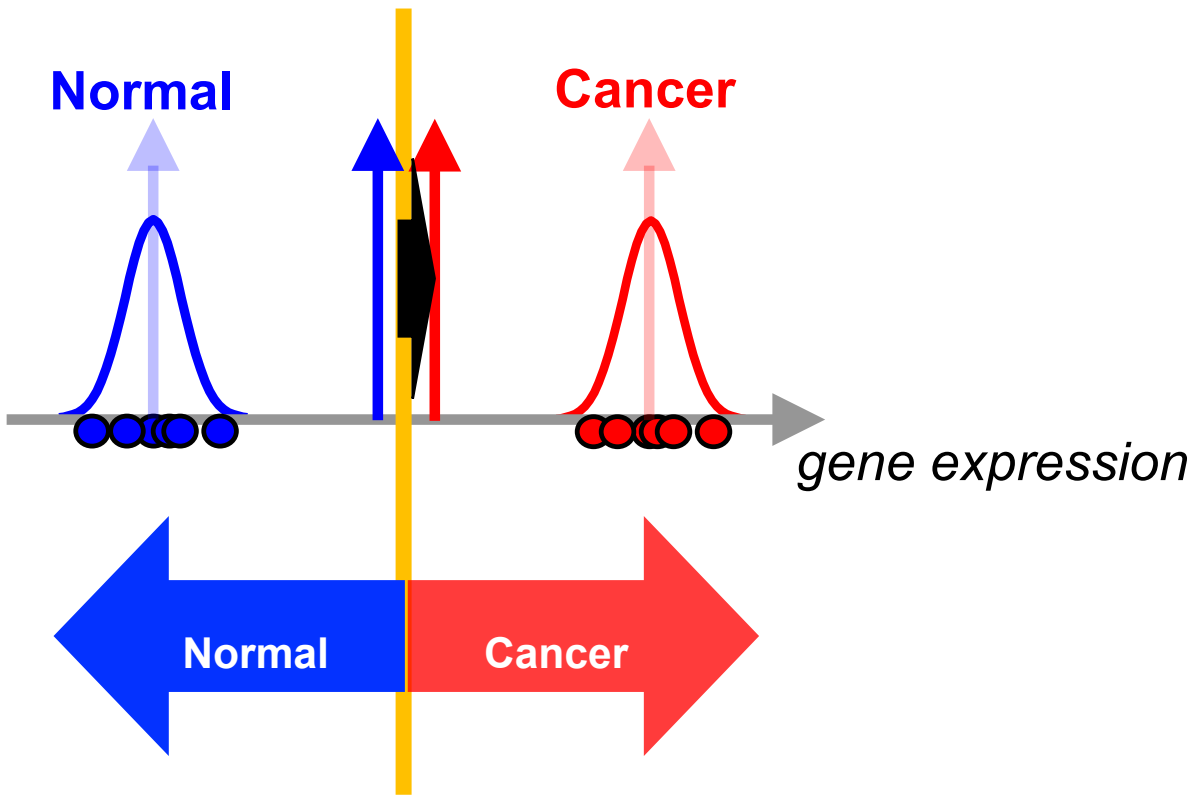
BioSB

Shrunken centroids (8)
Repeat shrinking and evaluation of classifier till all genes shrunk away



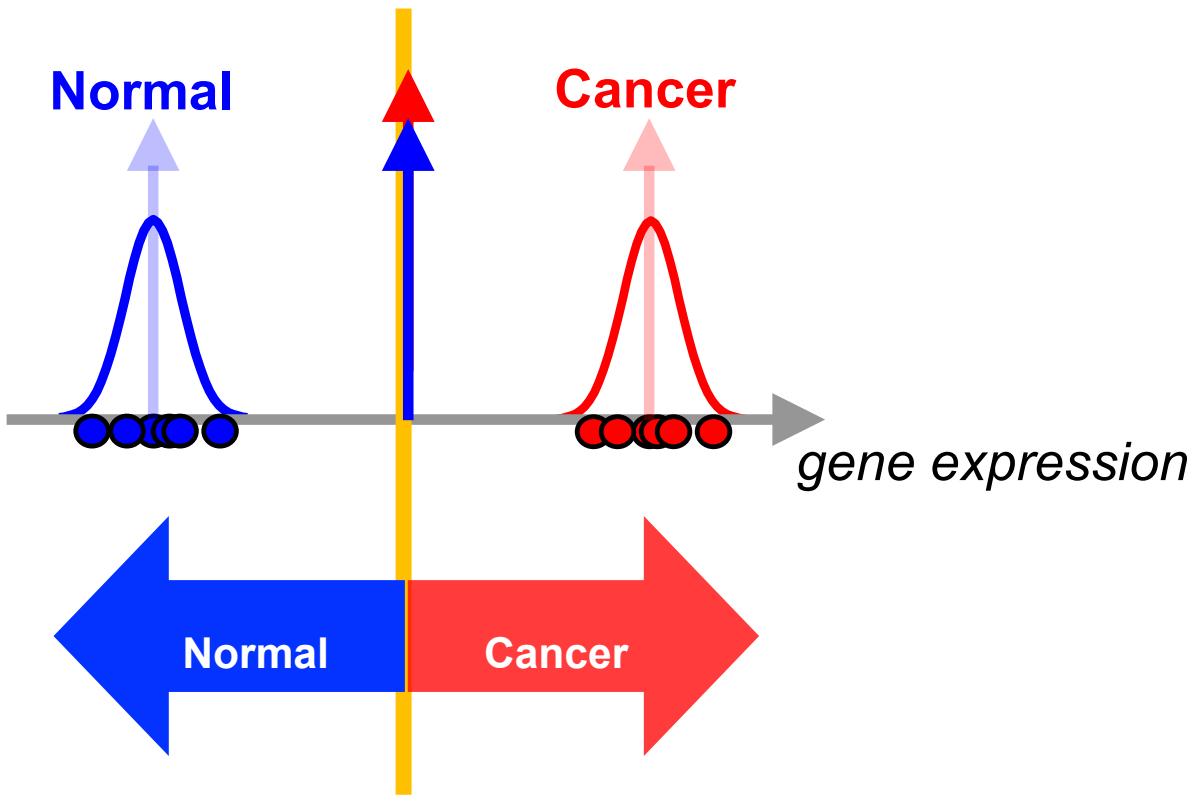
BioSB

Shrunken centroids (8)
Repeat shrinking and evaluation of classifier till all genes shrunk away



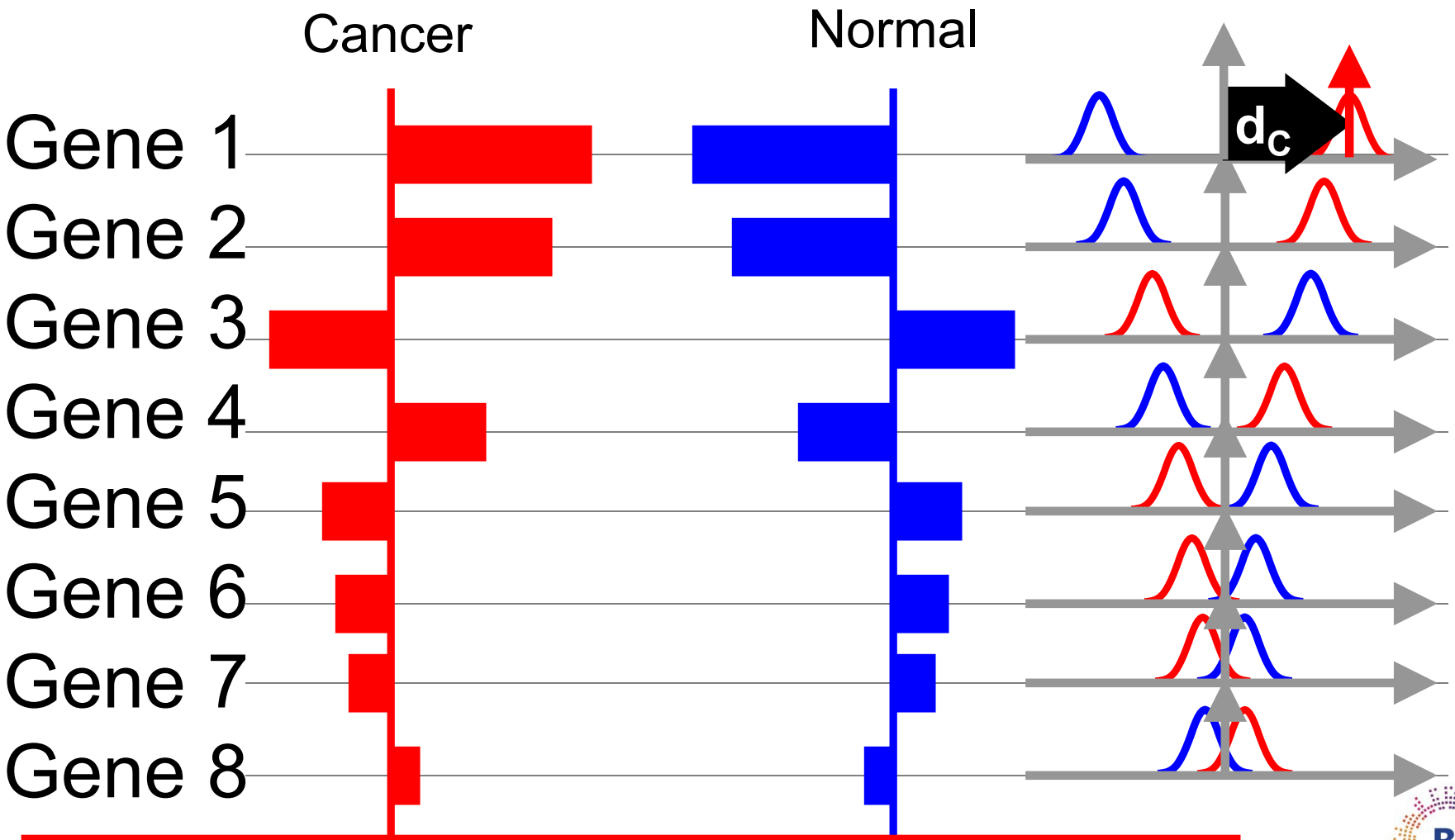
BioSB

Shrunken centroids (8)
Repeat shrinking and evaluation of classifier till all genes shrunk away



BioSB

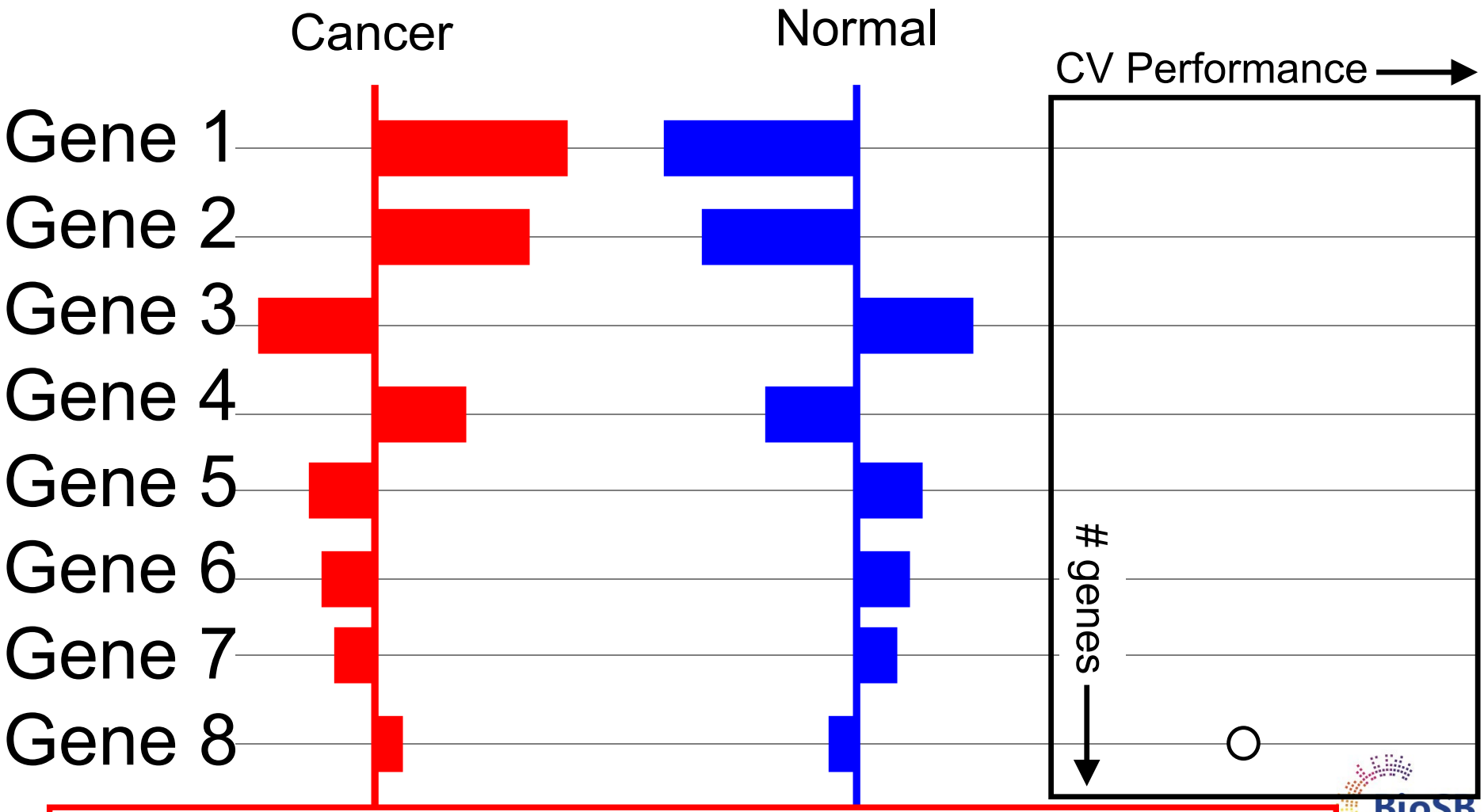
Shrunken centroids: selecting the genes



Genes sorted based on D-measure: best to worse

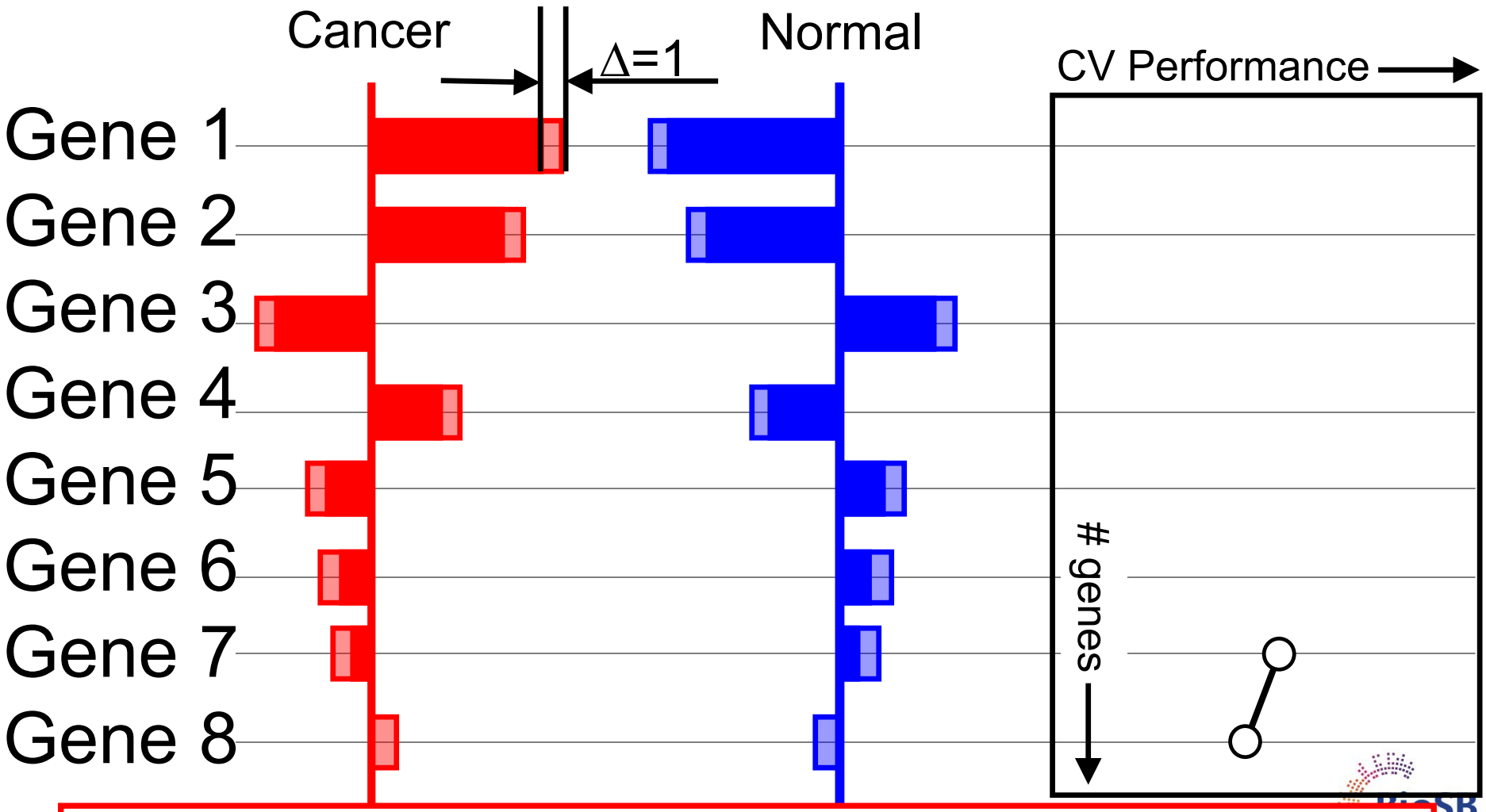
BioSB

Shrunken centroids: selecting the genes



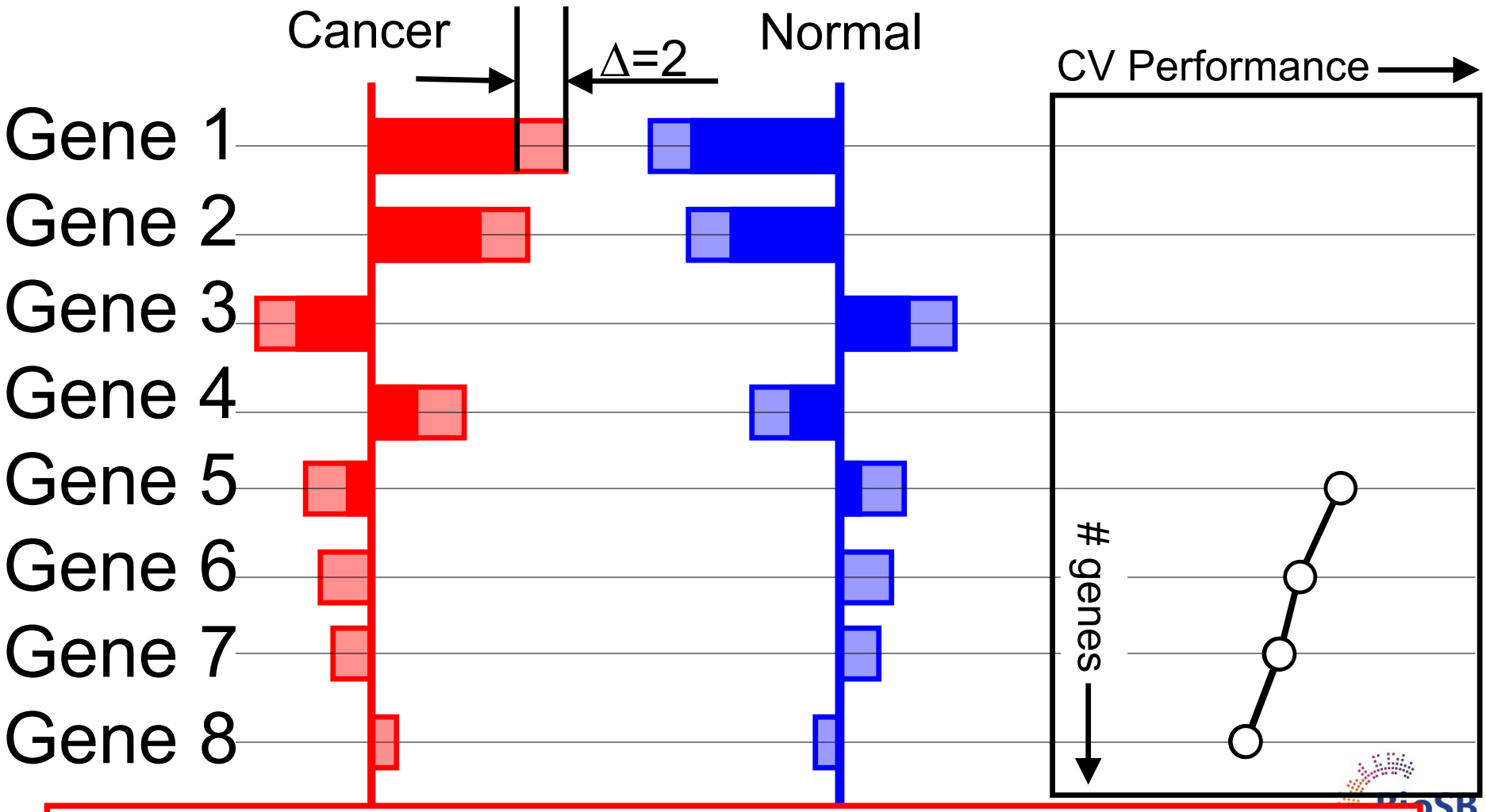
Train classifier on all 8 genes; estimate CV performance

Shrink all d by Δ=1: reduce length by 1



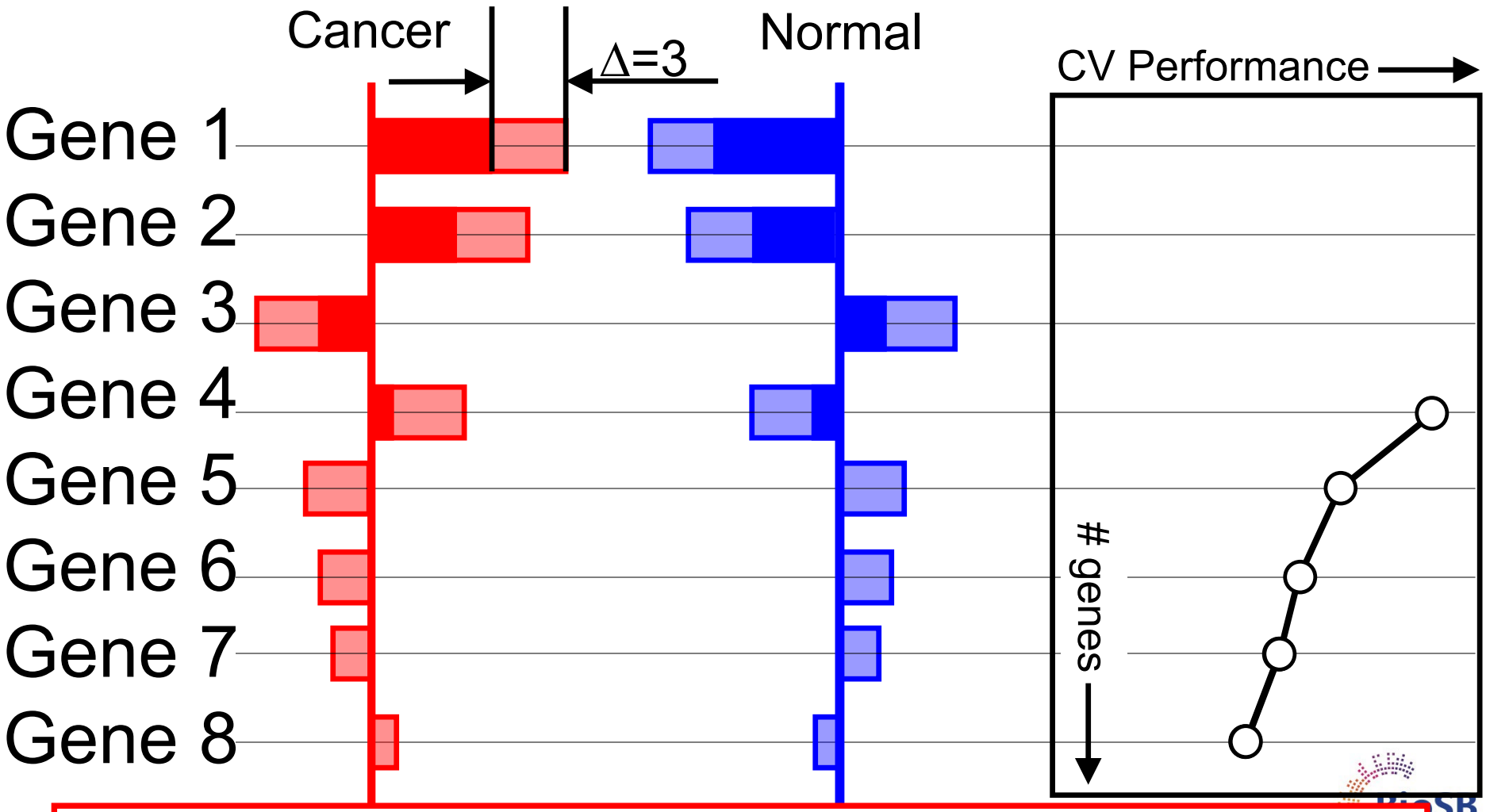
Train classifier on 7 genes (d>0); estimate CV performance

Shrink all d by Δ=2: reduce length by 2



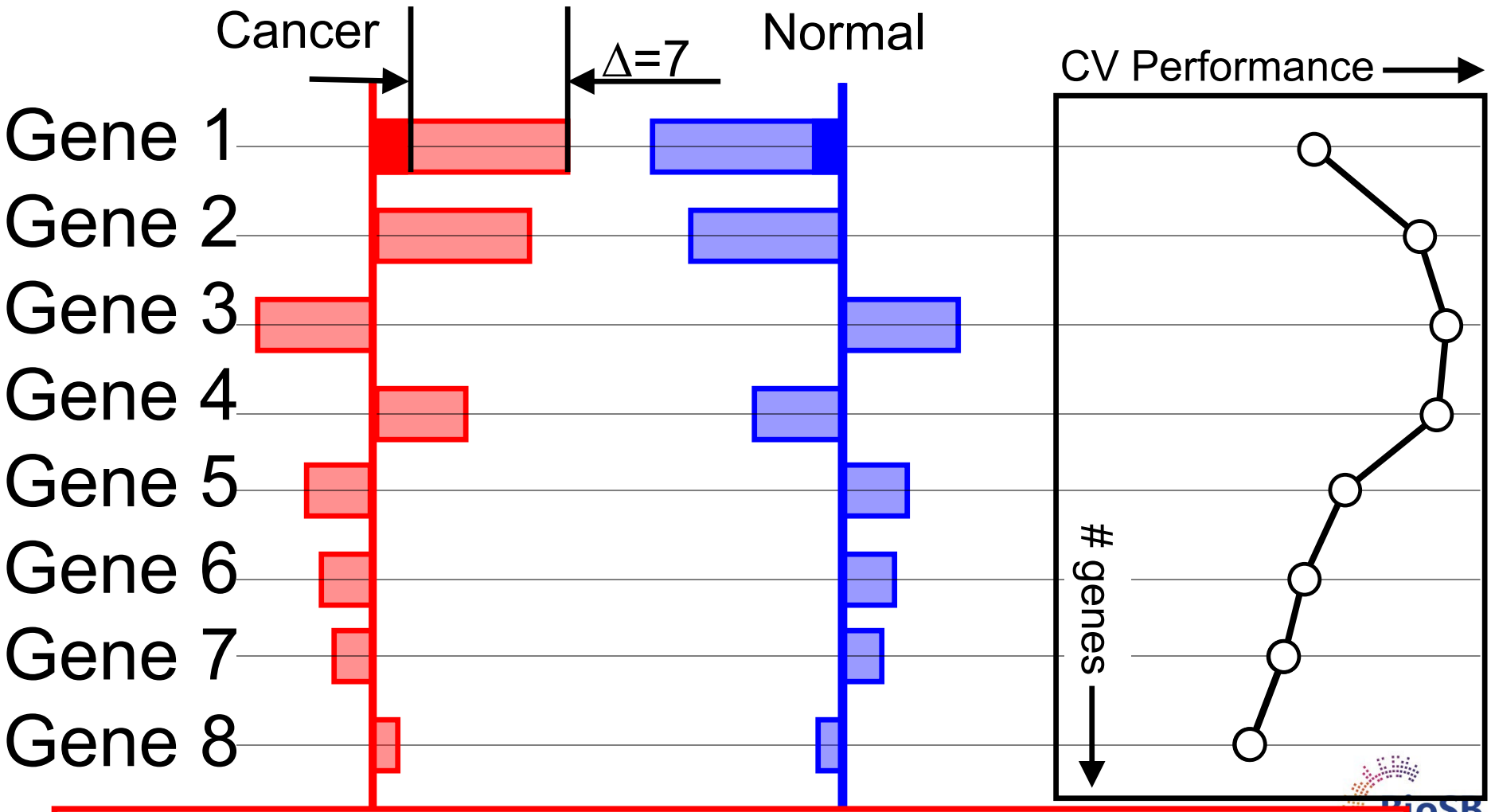
Train classifier on 5 genes (d>0); estimate CV performance

Shrink all d by Δ=3: reduce length by 3



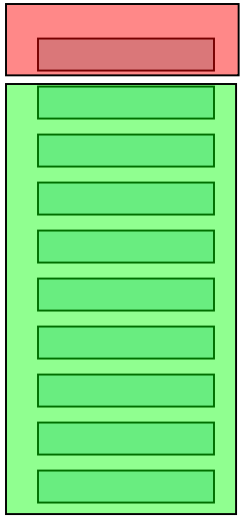
Train classifier on 4 genes (d>0); estimate CV performance

Shrink all d by Δ=7: reduce length by 7

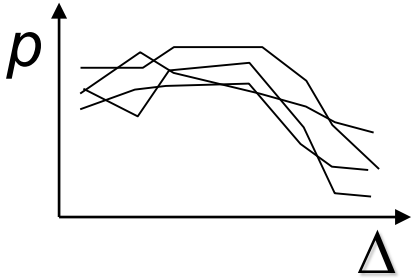


Train classifier on 1 gene (d>0); estimate CV performance

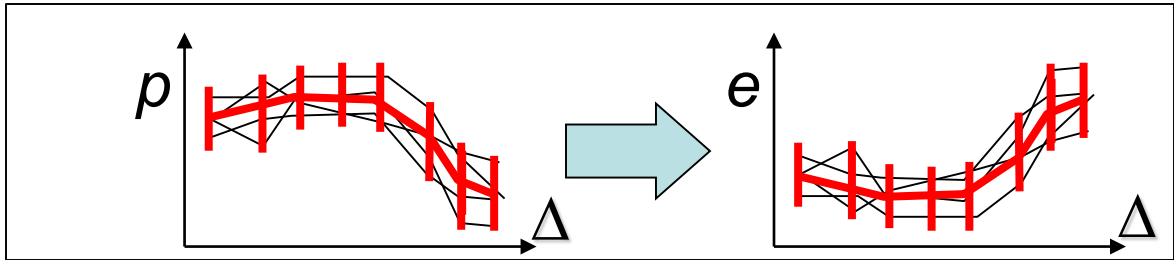
Determining the optimal Δ



1. Split the data (X) in 10 equal parts (x_1,...x_{10})
2. For each of the 10 folds (i=1,2,...,10)
3. On the training set ($X \backslash x_i$)
   1. Compute the class and overall centroids
   2. For a range of Δ (Δ = [0,0.5,...,7])
      i. Shrink d for all genes
      ii. Compute 'shrunken centroids' on training set
      iii. Test the resulting classifier on the test set (x_i)
   3. Result: 10 Curves of performance vs. Δ

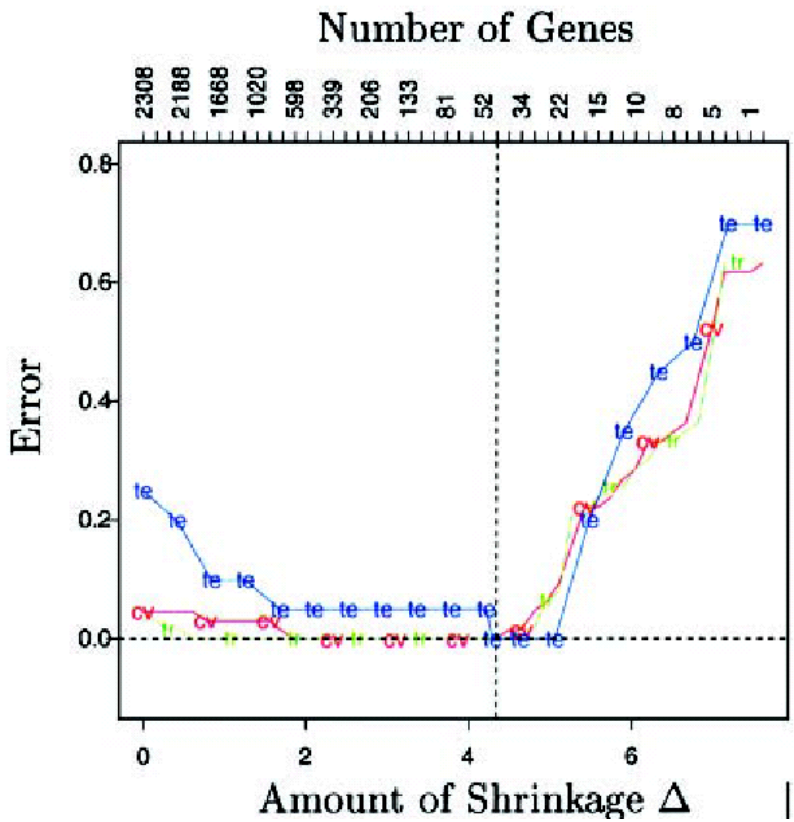


4. Average all 10 curves and compute std. dev. at each Δ
5. Pick the Δ where the performance is maximal (error min.)



PAM

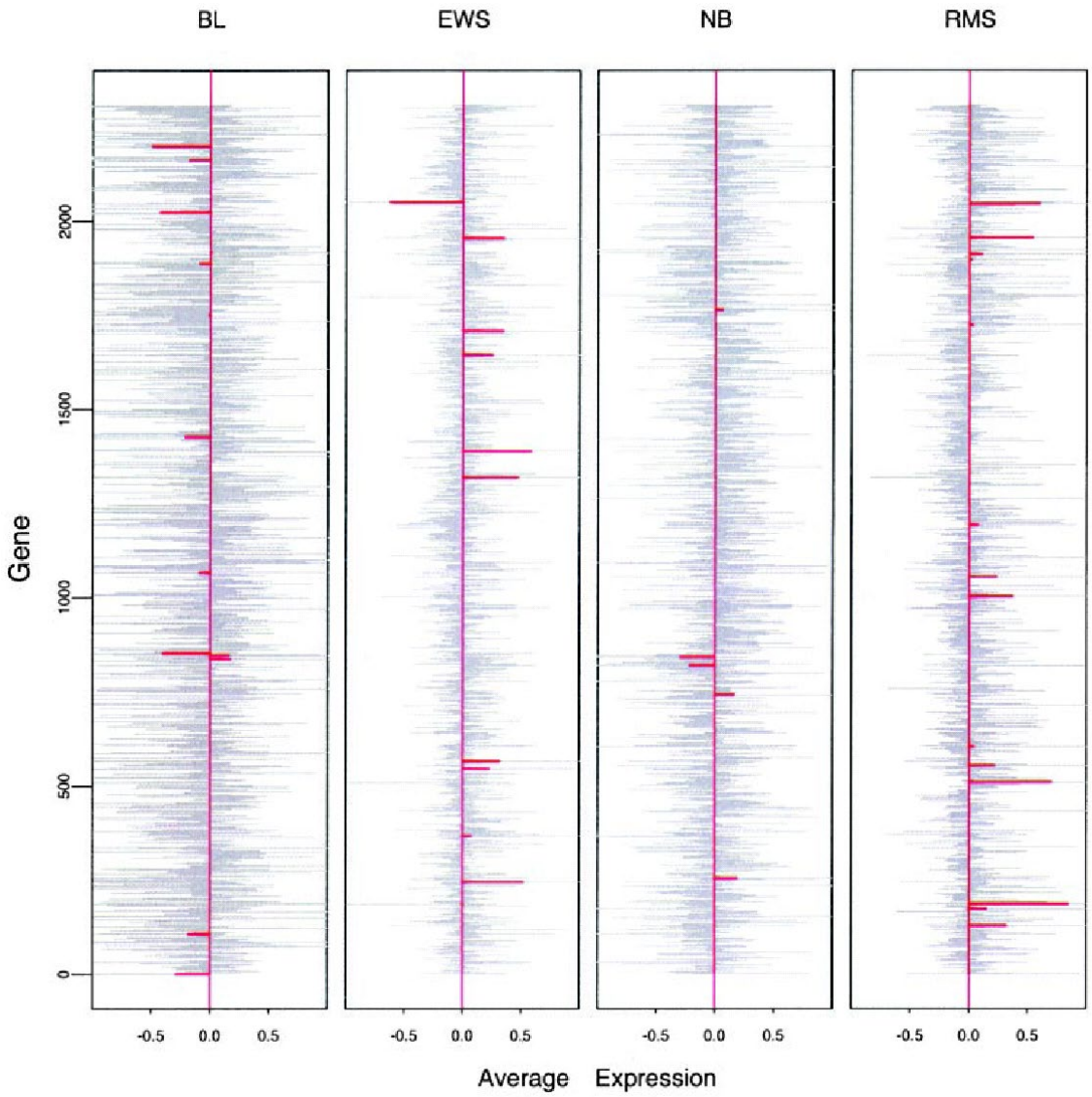
- For the Khan datatset; 4 classes: BL, EWS, NB, RMS
- At optimal Δ : 43 genes *not* shrunk away



Neuroblastoma (NB)
Rhabdomyosarcoma (RMS)
Burkitt lymphoma (BL)
Ewing family of tumors (EWS),

R. Tibshirani *et al.* (2002) PNAS 99(10):6567-6572, 2002.

PAM (2)

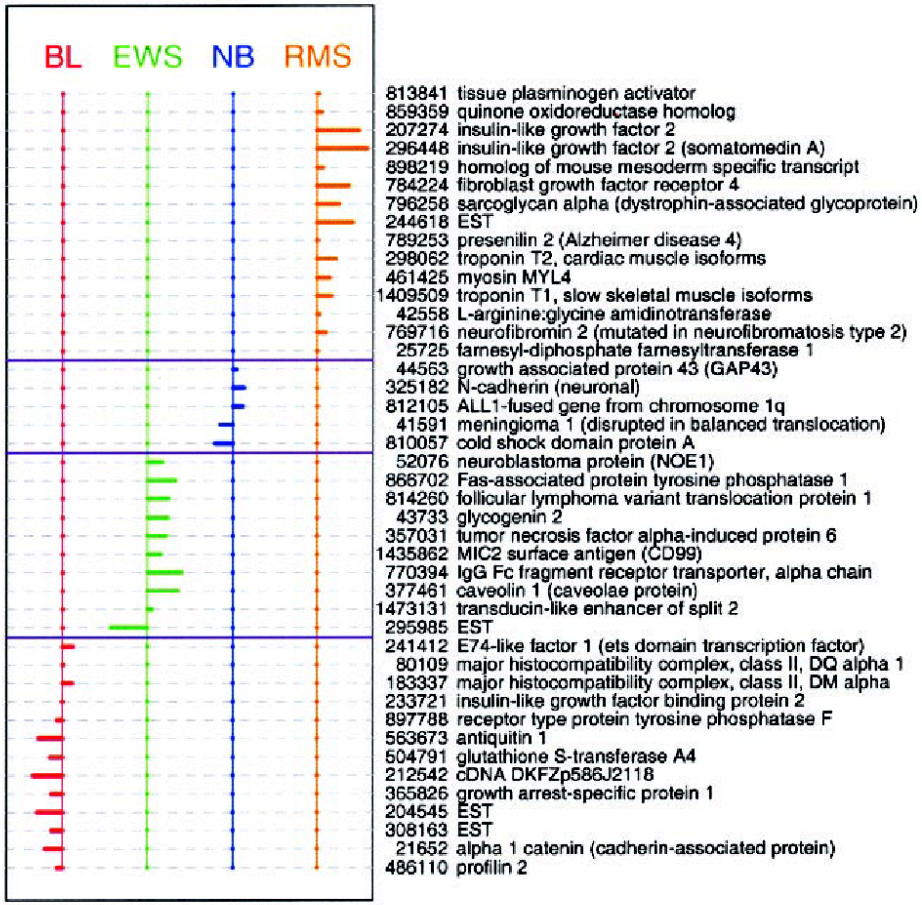


shrunk

unshrunk

PAM (3)

At optimal Δ : 43 genes *not* shrunk away

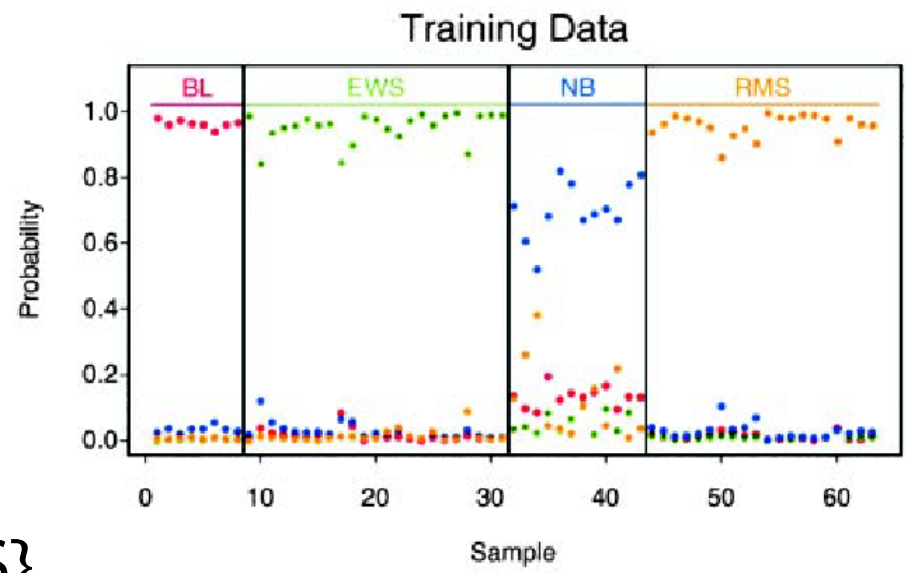


- Neuroblastoma (NB)
- Rhabdomyosarcoma (RMS)
- Burkitt lymphoma (BL)
- Ewing family of tumors (EWS),

R. Tibshirani *et al*. (2002) PNAS 99(10):6567-6572, 2002.

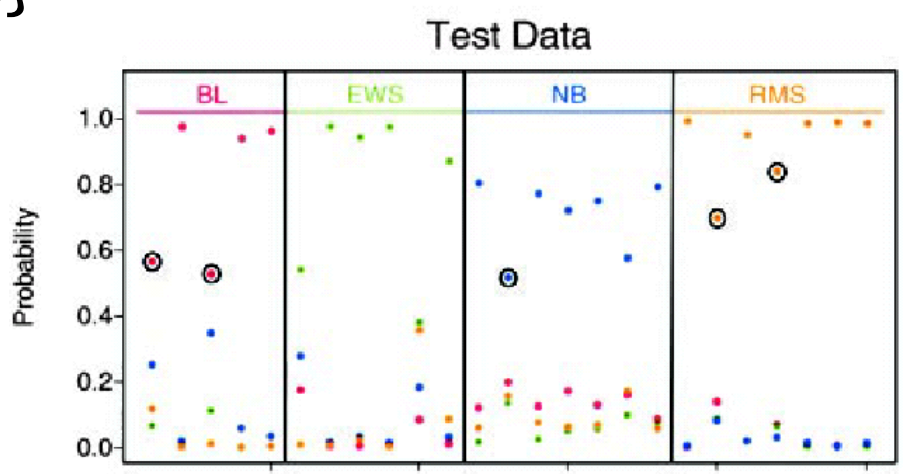
Scoring samples by posterior prob's

$$\widehat{p}(k|x^*)$$



k = {BL,EWS,NB,RMS}

$$\widehat{p}(k|x^*)$$



Shrinkage

- PAM: controls contribution of genes to classifier based on individual quality (d-measure) and controls degree of contribution with cross-validated classification error
- Other approach: regularisation, combine error and penalty for number of genes explicitly

BioSB

Shrinkage (2)

- Model: $y = \beta_0 + \sum_{i=1}^{p} \beta_i x_i + \varepsilon$

- Penalised (*aka* regularised) least squares:
  - Ridge regression:

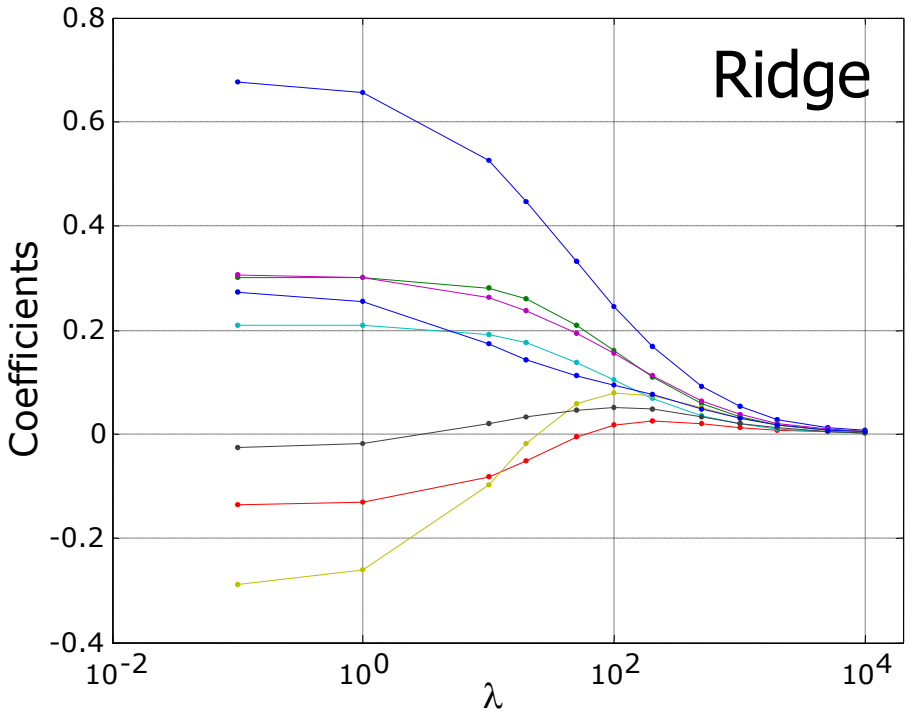
$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left[\sum_{j=1}^{n} \left(y_j - \beta_0 - \sum_{i=1}^{p} \beta_i x_{j,i} \right)^2 + \lambda \sum_{i=1}^{p} \beta_i^2 \right]$$

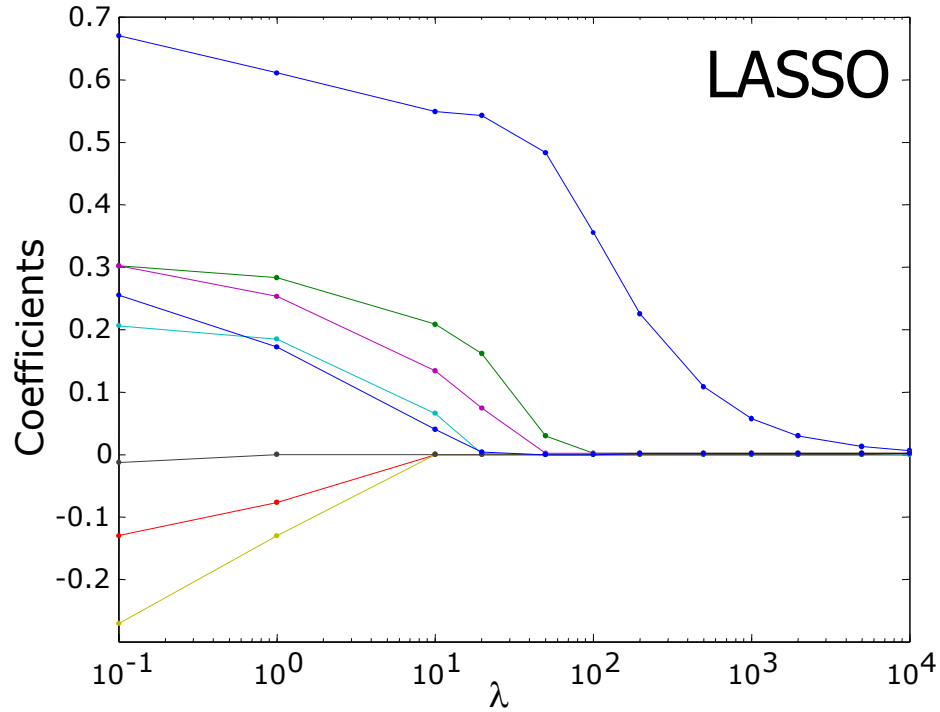
  - LASSO: minimise

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left[\sum_{j=1}^{n} \left(y_j - \beta_0 - \sum_{i=1}^{p} \beta_i x_{j,i} \right)^2 + \lambda \sum_{i=1}^{p} |\beta_i| \right]$$

LASSO

- Difference seems small, but effect of LASSO is that genes are no longer used (like in PAM!)





Final summary

- Feature extraction:
  - Linear:
    - PCA,
    - Fisher
  - Non-linear
    - MDS
- Feature selection:
  - Criteria
  - search algorithms
    - forward,
    - backward,
    - branch & bound.
- Sparse classifiers:
  - Ridge,
  - LASSO