


Classification with microarray data

Aron C. Eklund
eklund@cbs.dtu.dk

DNA Microarray Analysis
January 6, 2012

CENTER FOR
RIBOBIOMASS
CALCULATION
ANALYSIS CBS

 DTU Systems Biology
Department of Systems Biology

Outline

1. The concepts of classification and prediction

and how they relate to microarrays and biological/medical problems

2. Machine learning methods for deriving classifiers

LDA, kNN, SVM, etc.

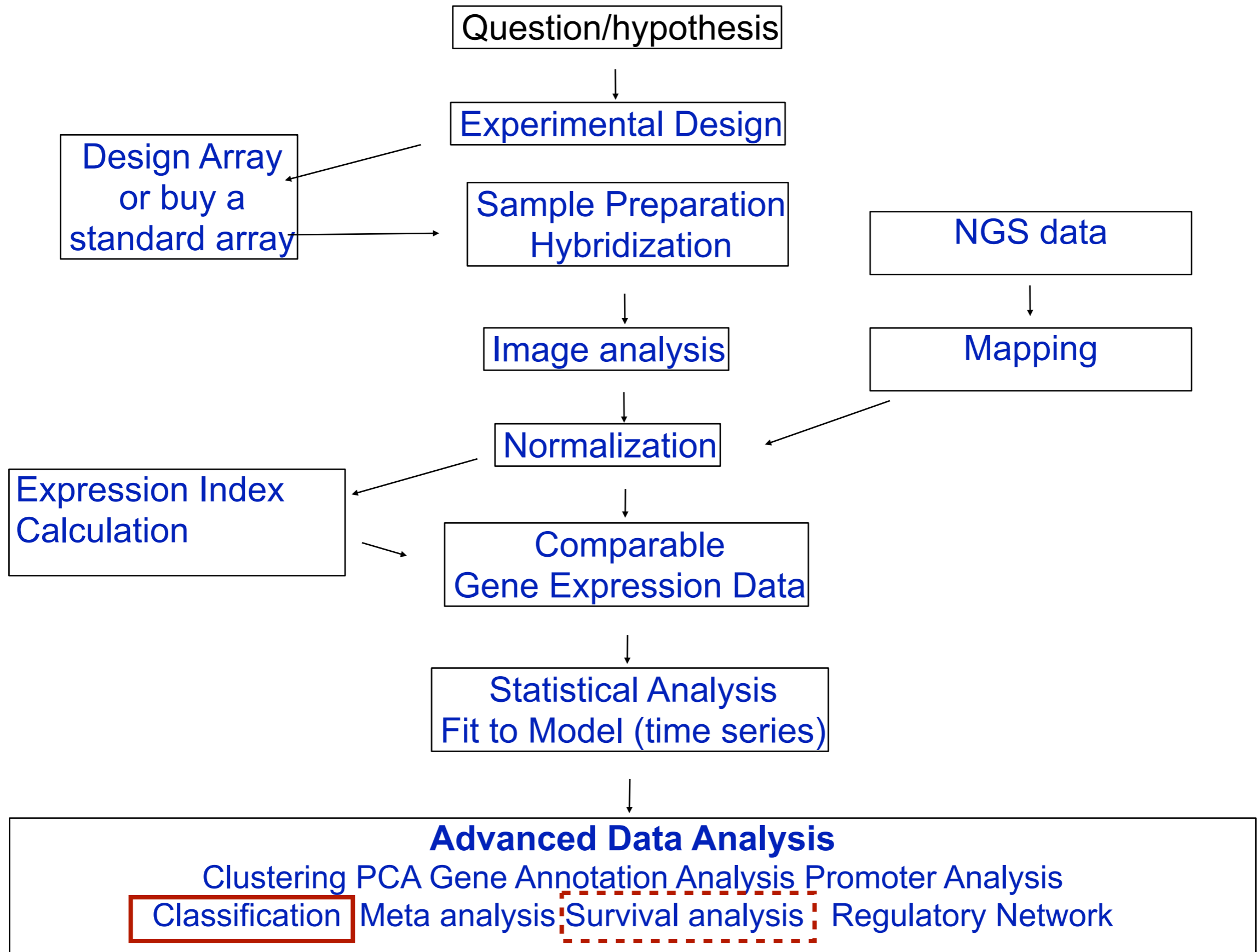
3. Assessment of classifier performance

Accuracy, specificity, sensitivity, survival analysis, ROC curves, ...

4. Other considerations

Feature selection, overfitting, validation and cross-validation.

Track A: The DNA Array Analysis Pipeline



Classification with gene expression profiles

expression profile of a tumor specimen

| | patient #7 |
|-------|------------|
| ESR1 | 876.4 |
| ERBB2 | 100.1 |
| BRCA1 | 798.8 |
| ... | ... |

features (genes)



classifier



predicted class

low-risk tumor

or

aggressive tumor

classifier algorithm:

```

if (ESR1 > 100) AND (ERBB2 > 550)
  tumor = low-risk
else
  tumor = aggressive
  
```

Why use classification with microarrays?

... mostly medical applications (?)

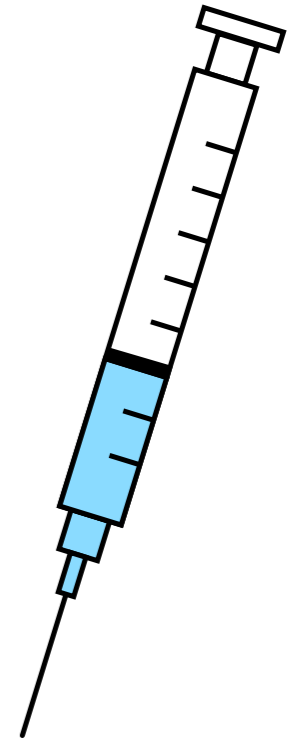
SNP profile of healthy tissue >> **predict risk of disease**

Expression profile of tissue, blood, etc. >> **diagnose disease**

Expression profile of tumor >> **select optimal chemotherapy**

Genomic profile of bacterial sample >> **identification of species**

... *others?*



Classifiers sound great

Where do I get one?

Two approaches to classification

1. Simple methods (~ univariate)

- easy to understand
- easy to derive
- easy to evaluate

2. Machine learning methods (~ multivariate)

- complicated

Even though the exercises emphasize #2, I recommend using #1 if possible!

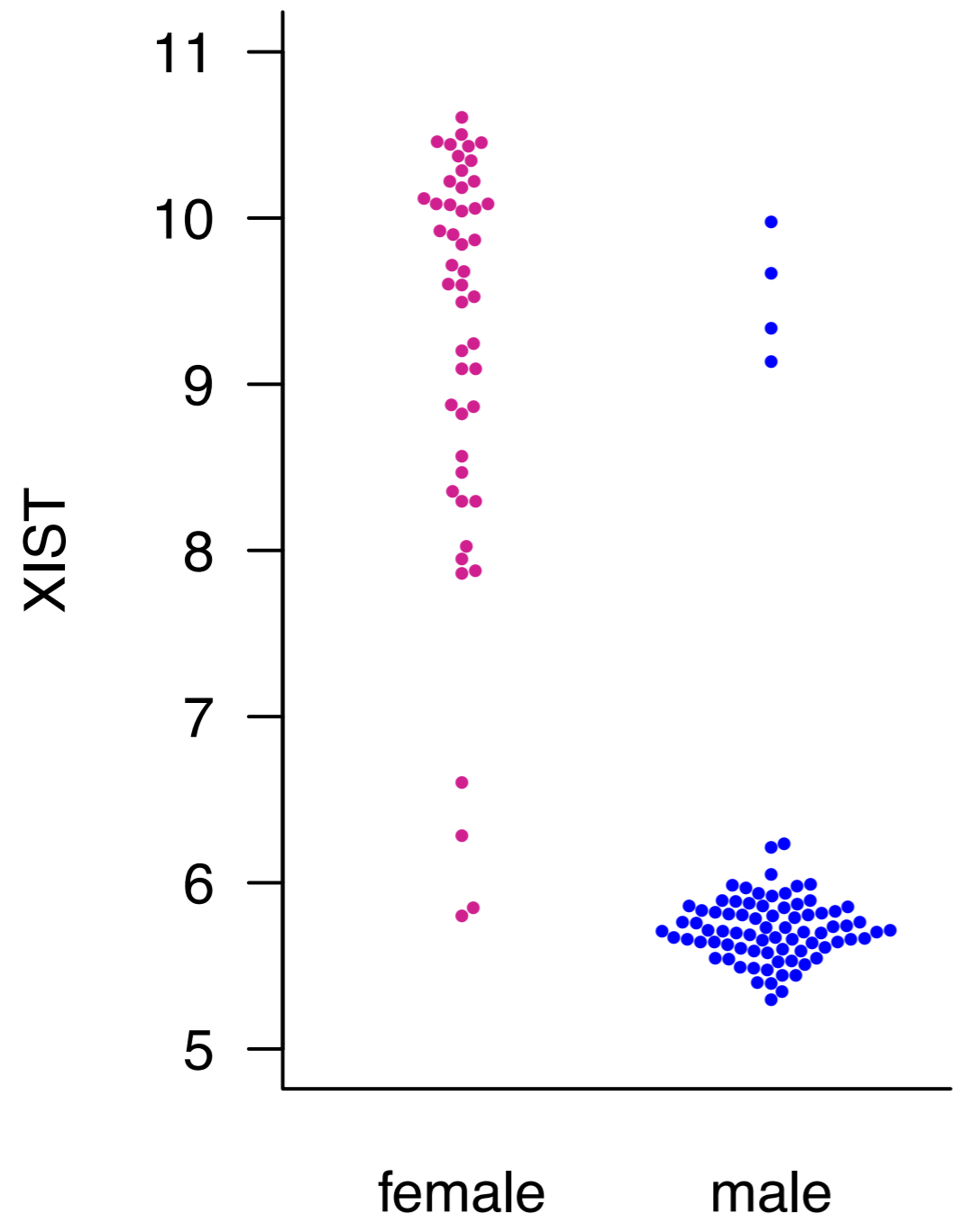
Hypothetical classification problem: gender

Input:
Expression profile



Output:
Gender (male/female)

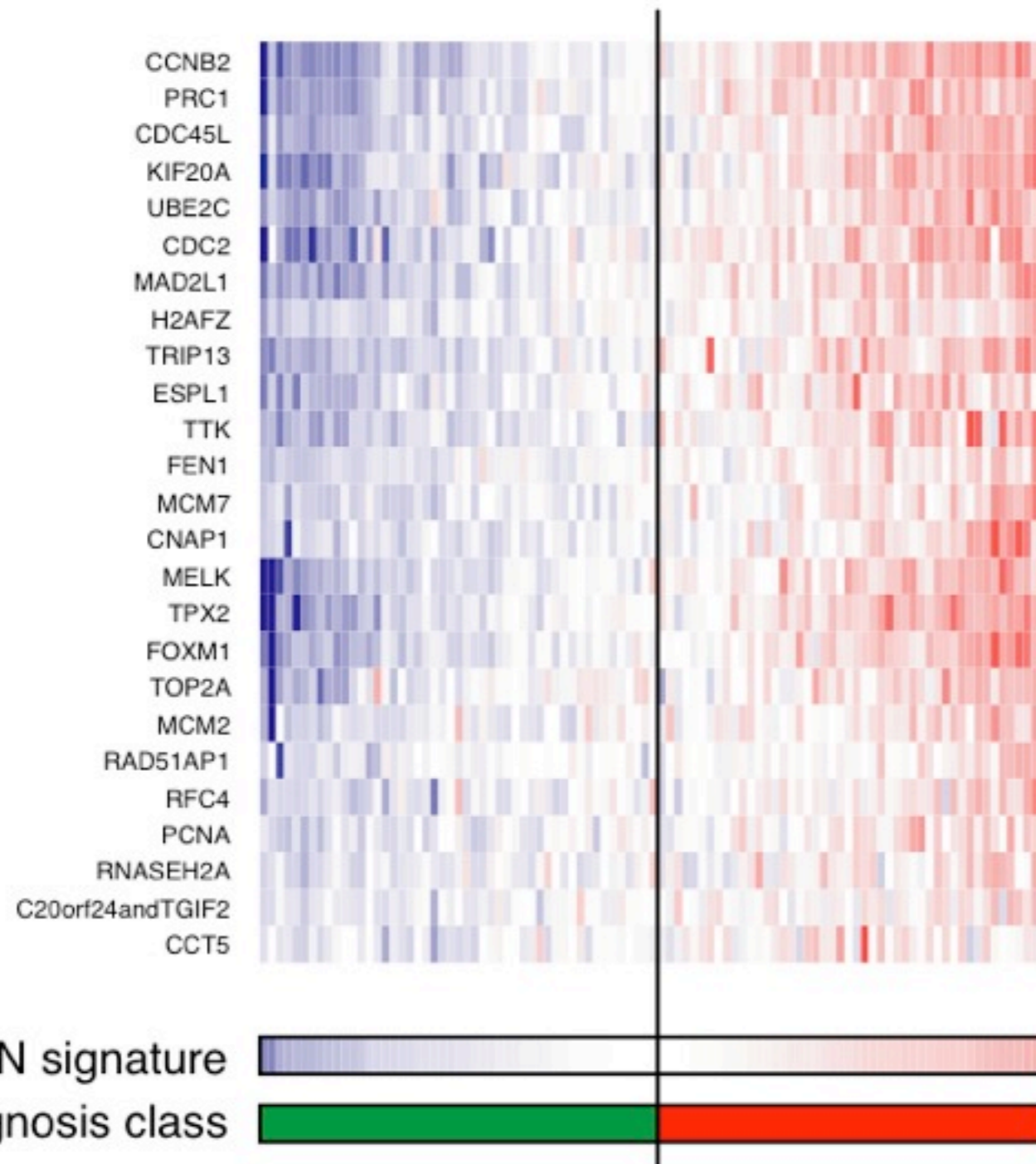
A very simple classifier



One gene!

```
If XIST > 7  
  sex = female  
else  
  sex = male
```

Another simple classifier



The average of 25 correlated genes

High = bad prognosis
 Low = good prognosis

Recipe for a simple univariate classifier

1. Choose a set of genes

a. Genes that you already know are relevant.

or

b. Genes that are differentially expressed between the two groups.

(*t* test, etc.)

2. Choose a simple rule for combining the gene measurements (mean, median, etc.)

3. Choose a decision threshold (or cutoff).

Now, a complicated classifier

A 21-gene recurrence score for breast cancer

The recurrence score on a scale from 0 to 100 is derived from the reference-normalized expression measurements in four steps. First, expression for each gene is normalized relative to the expression of the five reference genes (ACTB [the gene encoding b-actin], GAPDH, GUS, RPLPO, and TFRC). Reference-normalized expression measurements range from 0 to 15, with a 1-unit increase reflecting approximately a doubling of RNA. Genes are grouped on the basis of function, correlated expression, or both. Second, the GRB7, ER, proliferation, and invasion group scores are calculated from individual gene-expression measurements, as follows: **GRB7 group score = $0.9 \times \text{GRB7} + 0.1 \times \text{HER2}$** (if the result is less than 8, then the GRB7 group score is considered 8); **ER group score = $(0.8 \times \text{ER} + 1.2 \times \text{PGR} + \text{BCL2} + \text{SCUBE2}) \div 4$** ; **proliferation group score = $(\text{Survivin} + \text{KI67} + \text{MYBL2} + \text{CCNB1}$ [the gene encoding cyclin B1] + $\text{STK15}) \div 5$** (if the result is less than 6.5, then the proliferation group score is considered 6.5); and **invasion group score = $(\text{CTSL2}$ [the gene encoding cathepsin L2] + MMP11 [the gene encoding stromolysin 3]) $\div 2$** . The unscaled recurrence score (RSU) is calculated with the use of coefficients that are predefined on the basis of regression analysis of gene expression and recurrence in the three training studies: **$\text{RSU} = +0.47 \times \text{GRB7 group score} - 0.34 \times \text{ER group score} + 1.04 \times \text{proliferation group score} + 0.10 \times \text{invasion group score} + 0.05 \times \text{CD68} - 0.08 \times \text{GSTM1} - 0.07 \times \text{BAG1}$** . A plus sign indicates that increased expression is associated with an increased risk of recurrence, and a minus sign indicates that increased expression is associated with a decreased risk of recurrence. Fourth, **the recurrence score (RS) is rescaled from the unscaled recurrence score, as follows: $\text{RS} = 0$ if $\text{RSU} < 0$; $\text{RS} = 20 \times (\text{RSU} \times 6.7)$ if $0 \leq \text{RSU} \leq 100$; and $\text{RS} = 100$ if $\text{RSU} > 100$.**

...a complex function of several variables!!

Paik et al (2004) NEJM 351:2817

Machine learning

Let the computer do the work developing the classifier!

The goal: Given a set of *training data* (data of known class), develop a classifier that accurately predicts the class of novel data.

tumor specimen number

| gene | #1 | #2 | #3 | #4 | #5 | #6 | #7 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| ESR1 | 143.5 | 165.0 | 134.2 | 522.4 | 599.1 | 288.3 | 209.0 |
| ERBB2 | 801.1 | 291.7 | 293.1 | 827.3 | 980.0 | 155.5 | 128.1 |
| BRCA1 | 129.1 | 238.0 | 883.3 | 281.0 | 95.1 | 385.2 | 383.4 |

| | | | | | | | |
|------------|----|-----|-----|----|----|-----|-----|
| aggressive | no | yes | yes | no | no | yes | ??? |
|------------|----|-----|-----|----|----|-----|-----|

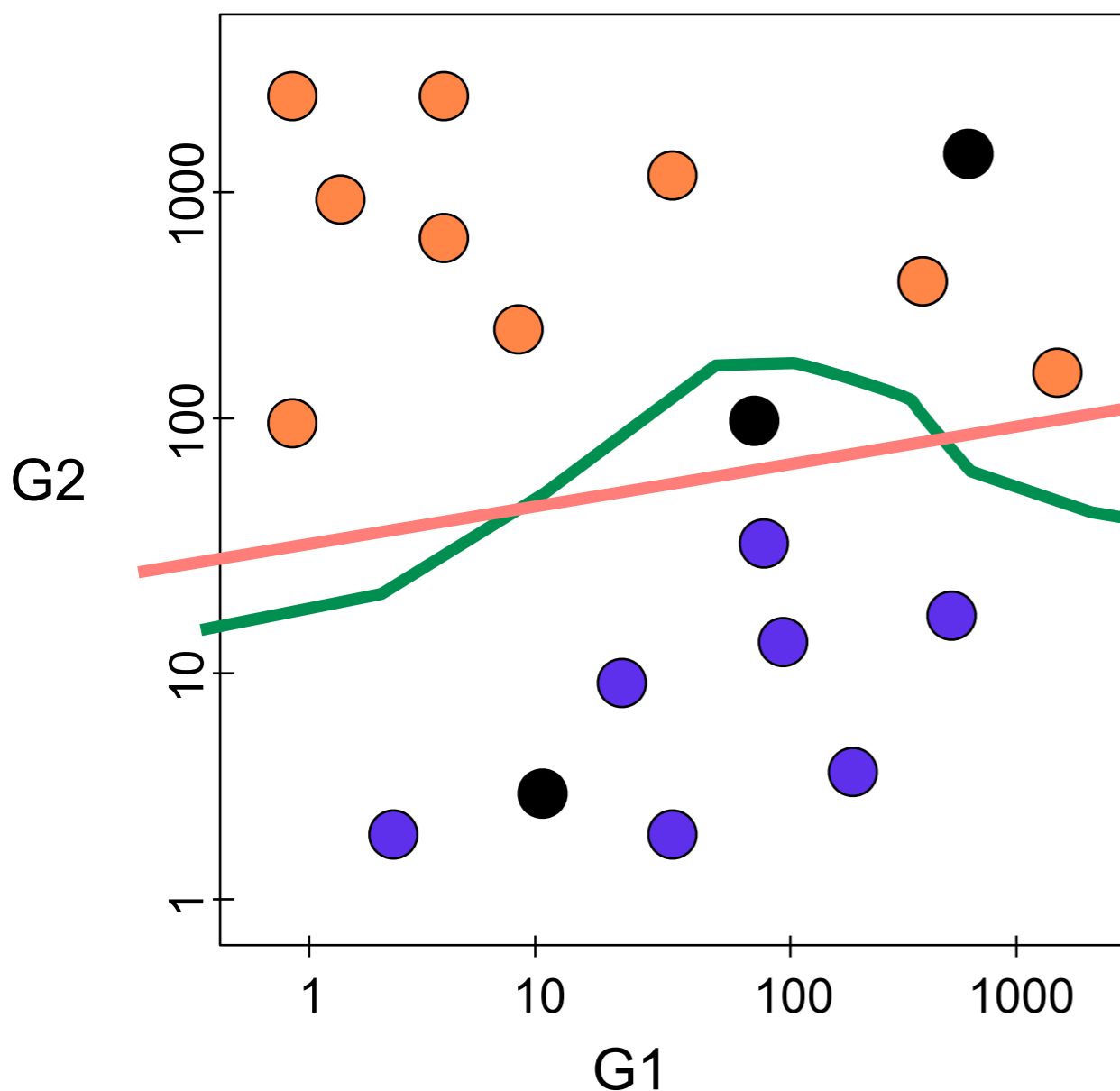
Training set

Classification as mapping

Suppose we want to develop a classifier based on the expression levels of two genes, G1 and G2:

Training data set:

- aggressive tumor
- friendly tumor
- unknown



>>> How do we do this algorithmically?

Linear vs. nonlinear

Multiple dimensions...?

Recipe for classification with machine learning

1. Choose a classification method (simpler is probably better)
2. Feature selection / dimension reduction (fewer is probably better)
3. Train the classifier
4. Assess the performance of the classifier

1. Choose a classification method

Linear:

- Linear discriminant analysis (LDA)
- Nearest centroid

Nonlinear:

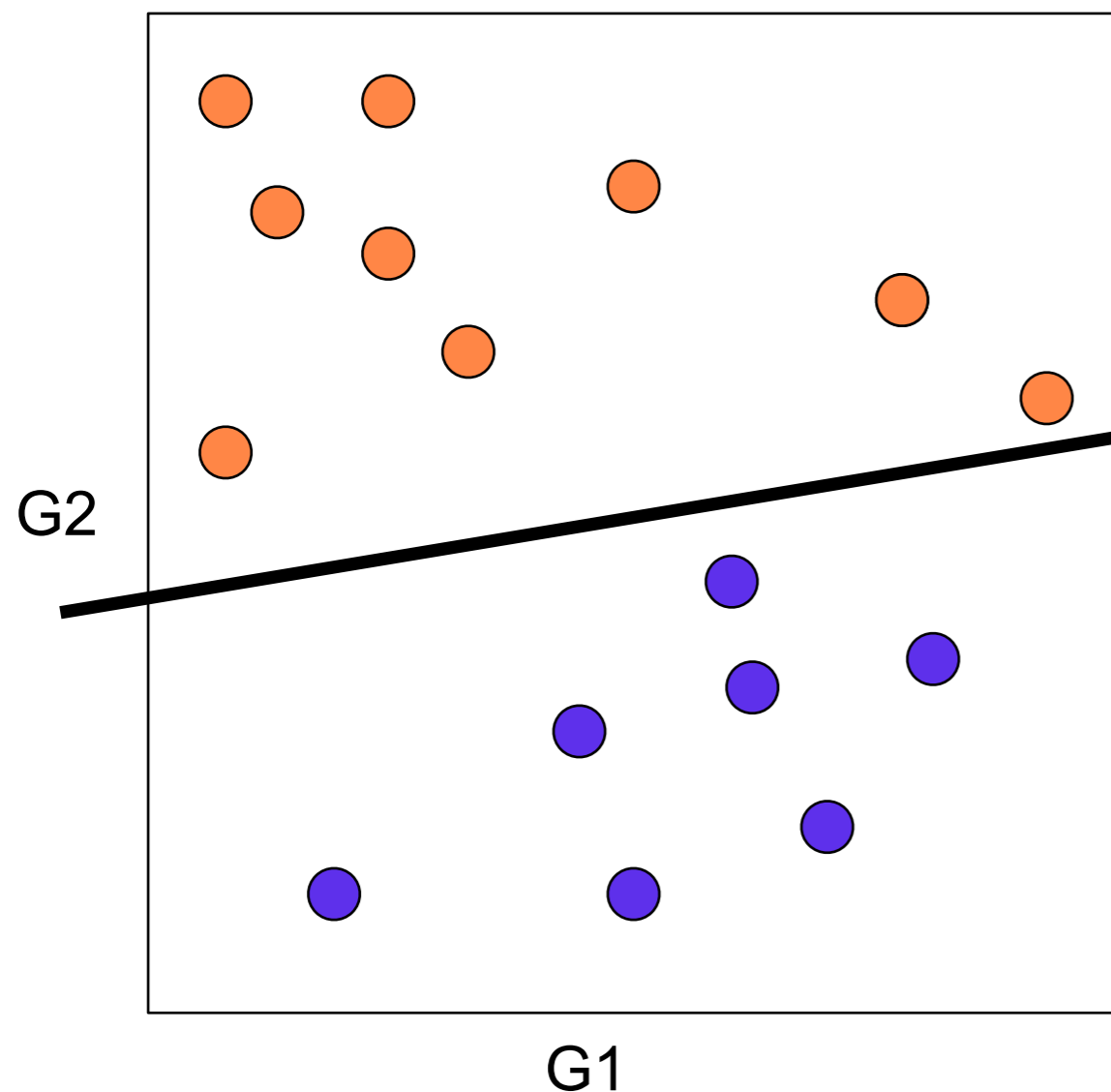
- k nearest neighbors (kNN)
- Artificial neural network (ANN)
- Support vector machine (SVM)

(many others also exist!)

Linear discriminant analysis (LDA)

Model the density of points as Gaussian.

Find the optimal separating line (or hyperplane, in higher dimensions)



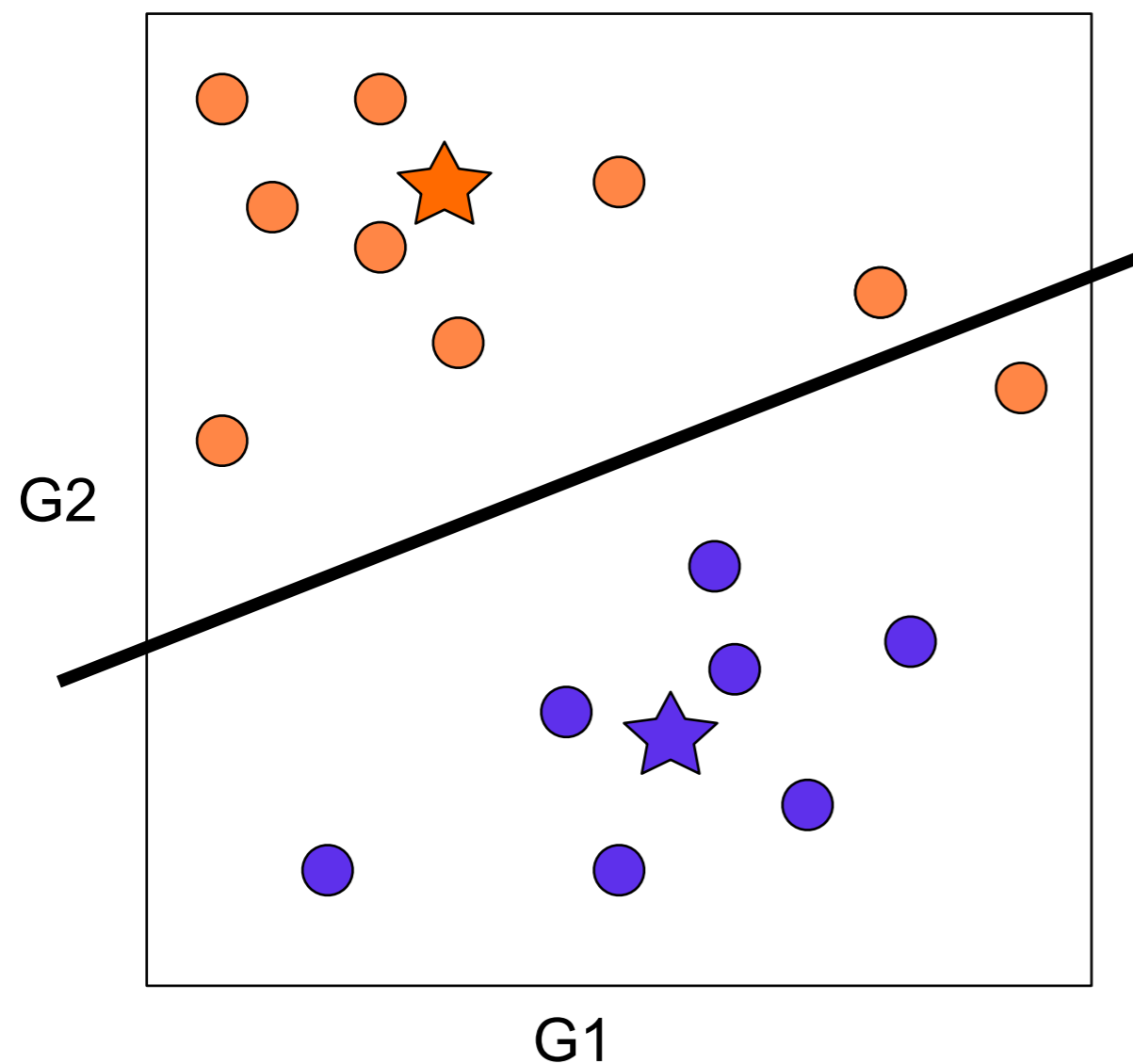
Assumptions:

- sampled from a normal distribution
- variance/covariance same in each class

R: lda (MASS package)

Nearest centroid

Calculate centroids for each class.

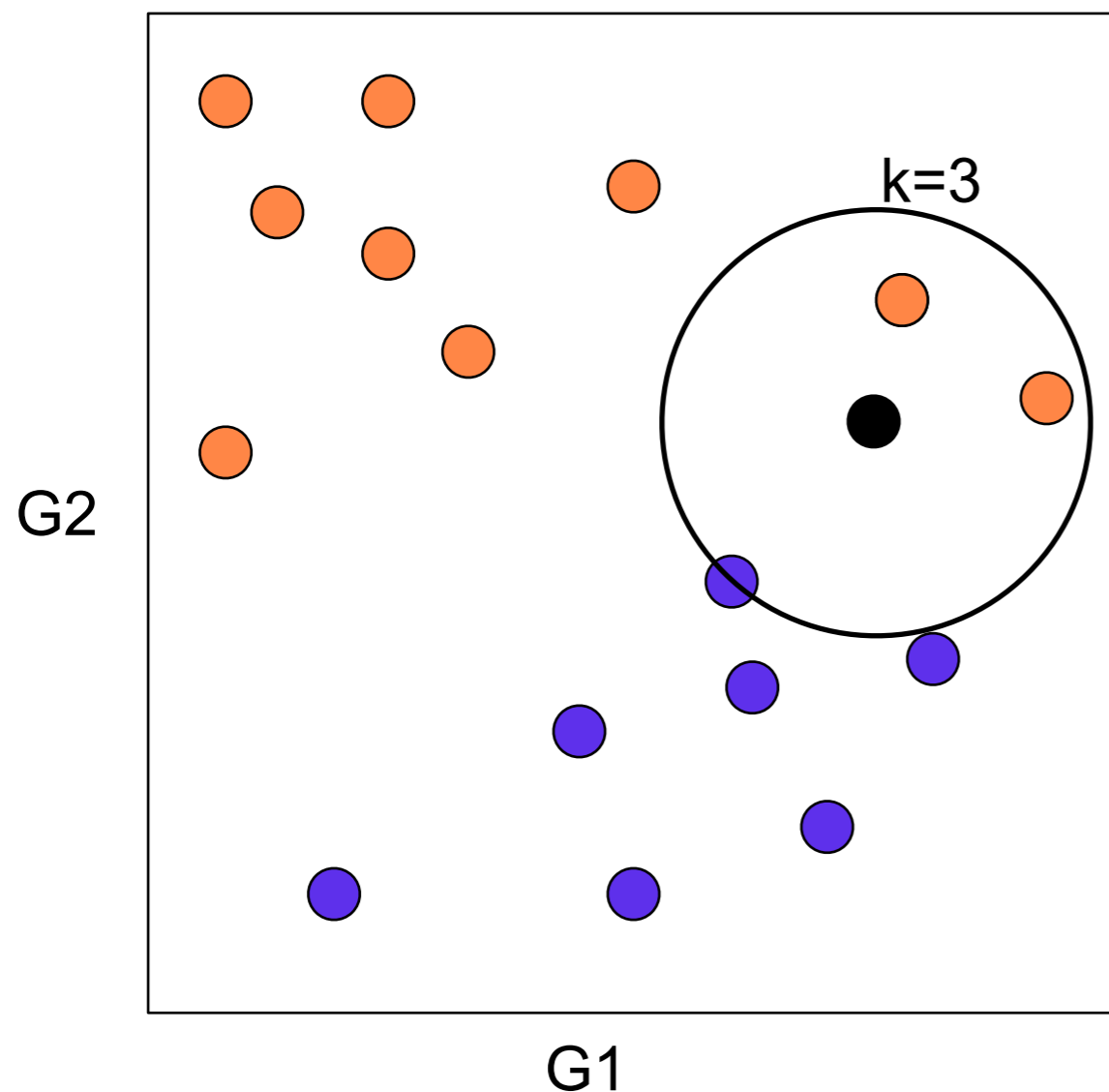


Similar to LDA.

Can be extended to multiple ($n > 2$) classes.

k nearest neighbors (kNN)

For a test case, find the k nearest samples in the training set, and let them vote.

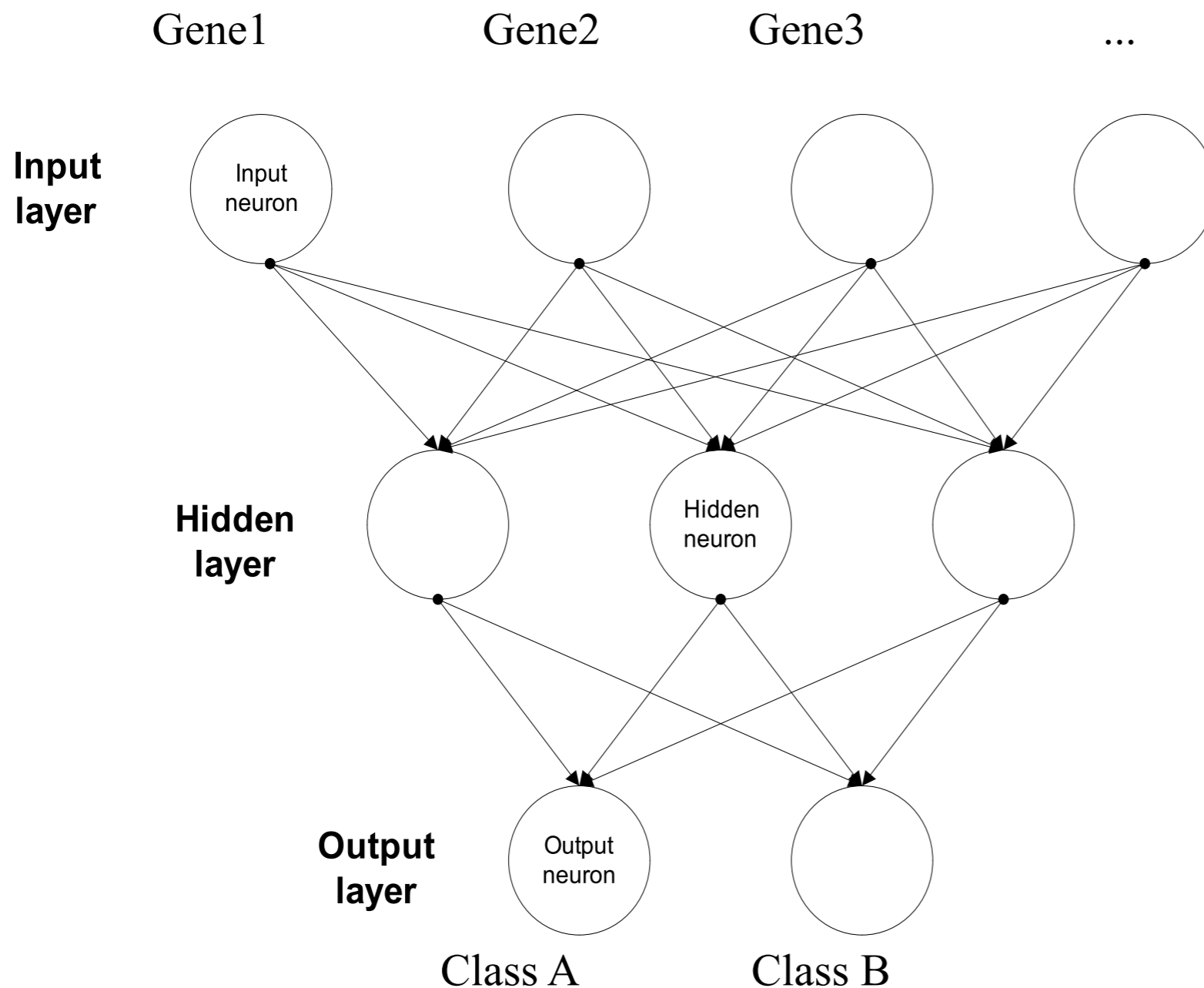


- need to choose k (hyperparameter)
- k must be odd
- need to choose distance metric

No real “learning” involved - the training set defines the classifier.

R: `knn` (*class package*)

Artificial neural network (ANN)



Each “neuron” is simply a function (usually nonlinear).

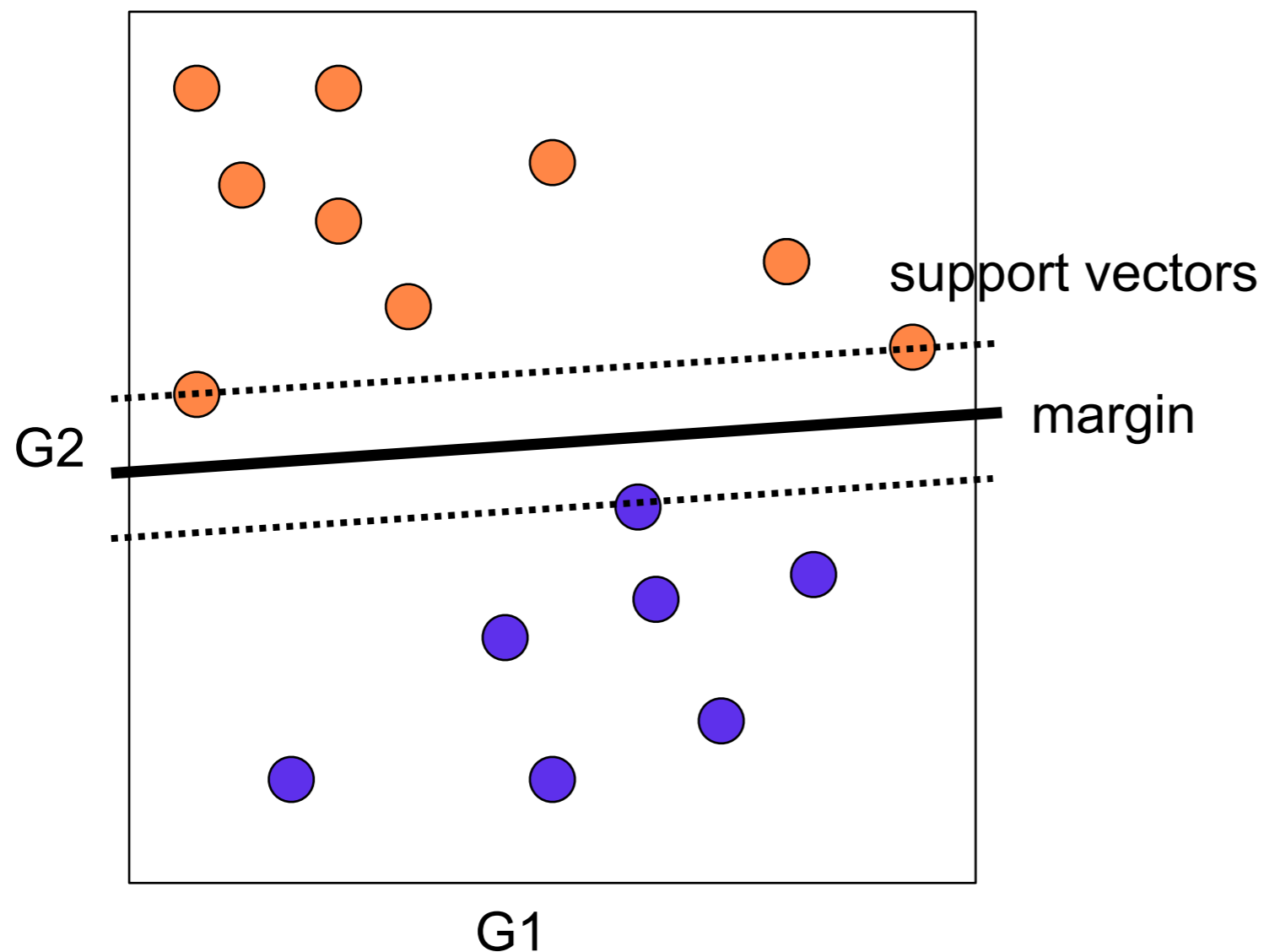
The “network” to the left just represents a series of nested functions.

Intepretation of results is difficult.

R: `nnet` (*nnet* package)

Support vector machine (SVM)

Find a line / plane / hyperplane that maximizes the margin.
 Often uses the “kernel trick” to map data to higher dimensions.



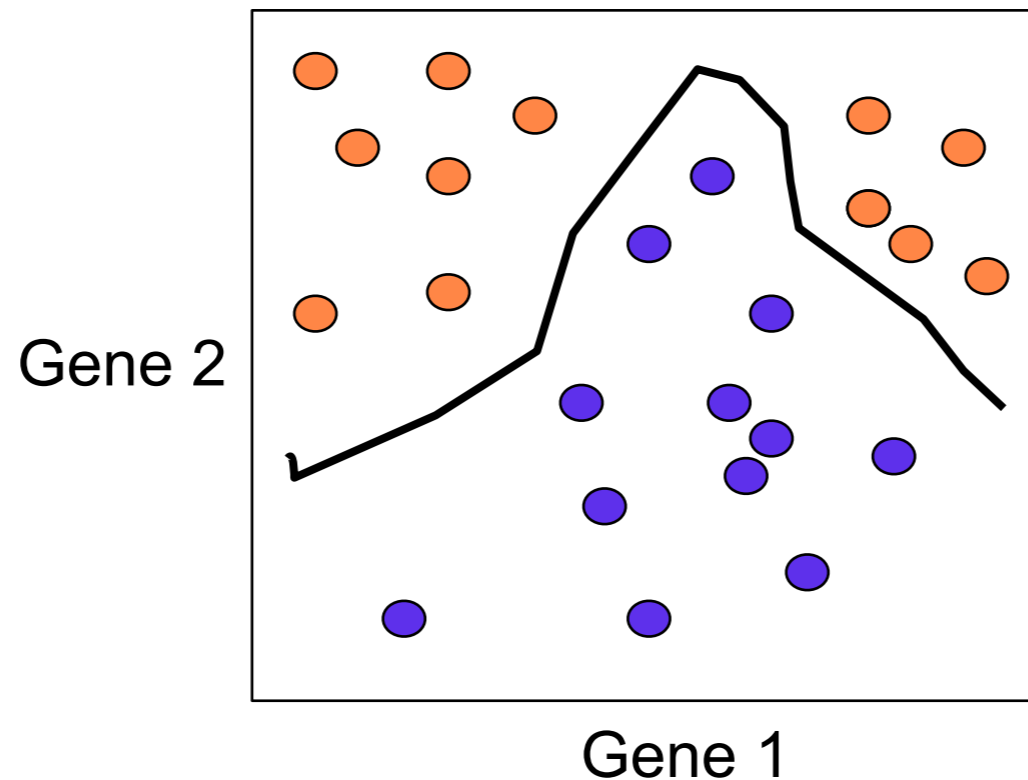
- very flexible
- many parameters

Results can be difficult to interpret.

R: `svm` (*e1071* package)

Comparison of machine learning methods

| | |
|--|--|
| <p>Linear discriminant analysis Nearest centroid kNN</p> | <p>Neural networks Support vector machines</p> |
| <p>Relatively simple Based on distance calculation Good for simple problems Good for few training samples Distribution of data assumed</p> | <p>Relatively complicated Involve machine learning Several adjustable parameters Many training samples required (e.g.. 50-100) Flexible methods</p> |



Recipe for classification with machine learning

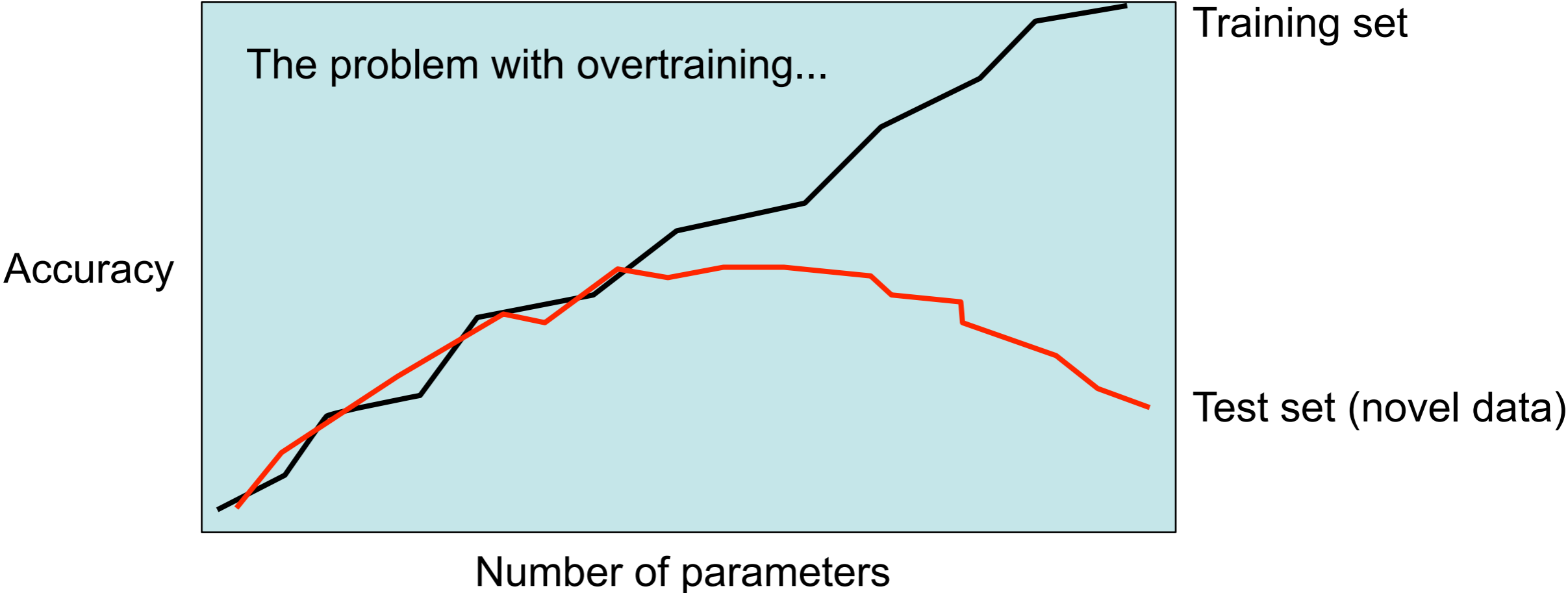
1. Choose a classification method
2. Feature selection / dimension reduction
3. Train the classifier
4. Assess classifier performance

Why do dimension reduction?

Previous slides: 2 genes X 16 samples.

Your data: **22000** genes X 31 samples.

More genes = more parameters in classifier. **Potential for over-training!!** aka *The Curse of Dimensionality*.



Dimension reduction

Two approaches:

1. Data transformation

- Principal component analysis (PCA); use top components
- Find clusters of genes; use their average.

2. Feature selection (gene selection)

- Significant genes: t-test / ANOVA
- High variance genes
- Hypothesis-driven
-

Recipe for classification with machine learning

1. Choose a classification method
2. Feature selection / dimension reduction
- 3. Train the classifier**
- 4. Assess classifier performance**

Train the classifier

If you know the exact classifier you want to use: **just do it.**

but...

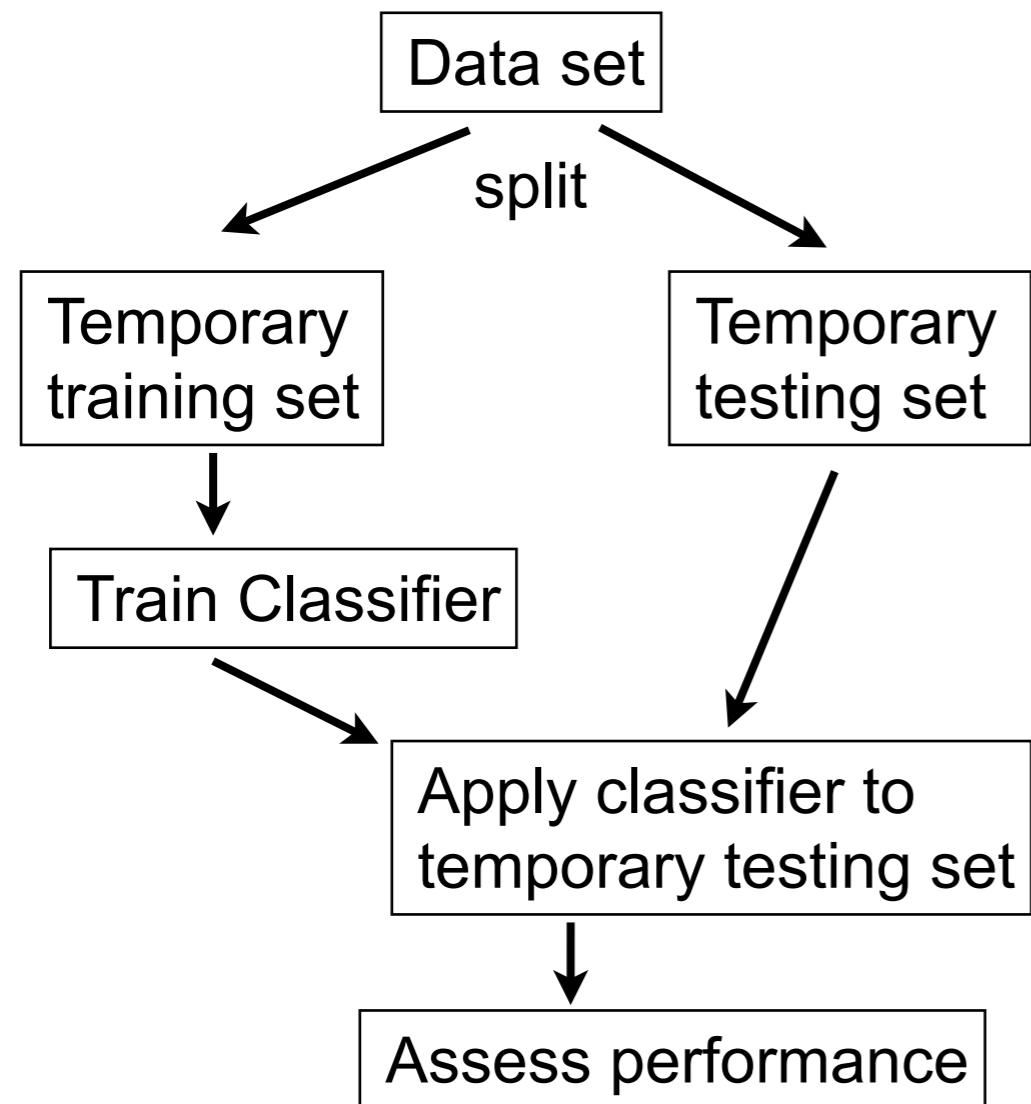
Usually, we want to try several classifiers *or* several variations of a classifier. e.g. number of features, k in kNN, etc. (hyperparameters)

The problem:

Testing several classifiers, and then choosing the *best one* leads to selection bias (= **overtraining**). This is bad.

Instead we can use **cross-validation** to choose a classifier.

Cross-validation



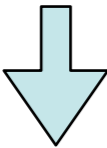
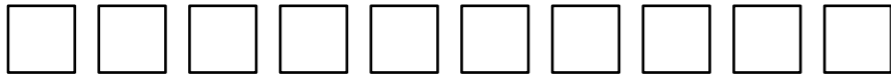
Do this several times!

Each time, select different sample(s) for the temporary testing set.

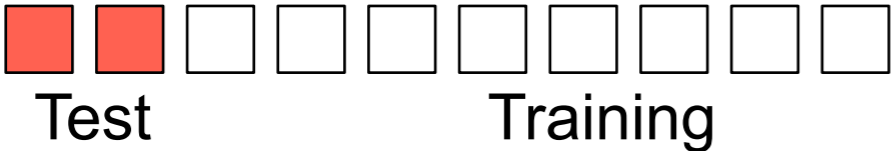
If you are using cross-validation as an optimization step, choose the classifier (or classifier hyperparameters) that results in best performance.

Cross-validation

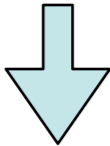
10 specimens of known class



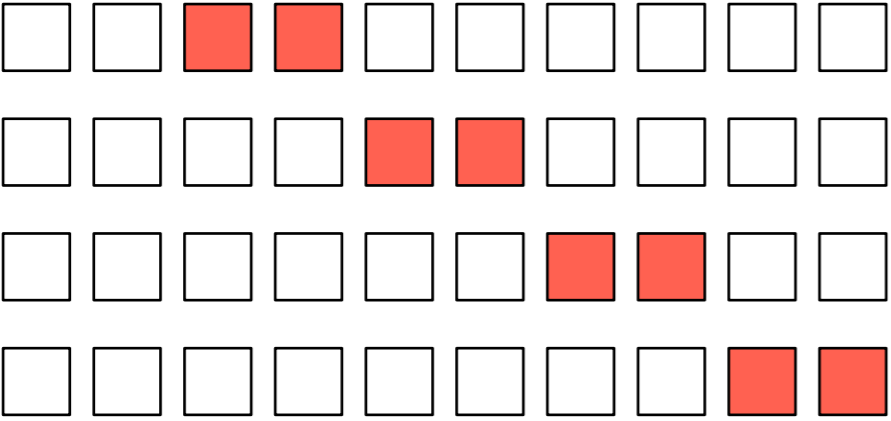
Split into training/test sets



Accuracy = 0.5



Repeat for other combinations



1.0

0.0

0.5

1.0

Average accuracy = 0.6

Cross-validation example

Given data with 100 samples

5-fold cross-validation:

Training: 4/5 of data (80 samples)

Testing: 1/5 of data (20 samples)

- 5 different models and performance results

Leave-one-out cross-validation (LOOCV)

Training: 99/100 of data (99 samples)

Testing: 1/100 of data (1 sample)

- 100 different models and performance results

Problems with cross-validation

1. You cannot use cross-validation to both optimize and evaluate your classifier.
2. The classifier obtained at each round of cross-validation is different, and is different from that obtained using all data.
3. Cross-validation will overestimate performance in the presence of experimental bias.

.... therefore an independent test set is *always* better!

Recipe for classification

1. Choose a classification method
2. Feature selection / dimension reduction
3. Train the classifier
- 4. Assess classifier performance**

Assess performance of the classifier

How well does the classifier perform on *novel* samples?

Bad: Assess performance on the training set.

Better: Assess performance on the training set using cross-validation.

Alternative: Set aside a subset of your data as a test set.

Best: *Also* assess performance on an entirely independent data set
(e.g. data produced by another lab).

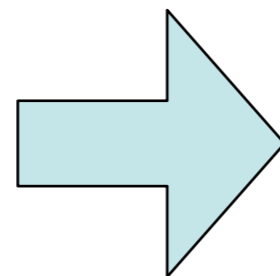
Assessing classifier performance

predict relapse of cancer

| Patient ID | predicted relapse | actual relapse | |
|------------|-------------------|----------------|-----|
| #001 | yes | no | FP |
| #002 | no | no | TN |
| #003 | yes | yes | TP |
| #004 | no | yes | FN |
| ... | ... | ... | ... |

TP = True Positive
 TN = True Negative
 FP = False Positive
 FN = False Negative

confusion matrix



| | predict yes | predict no |
|------------|-------------|------------|
| actual yes | 131 TP | 21 FN |
| actual no | 7 FP | 212 TN |

Classifier performance: Accuracy

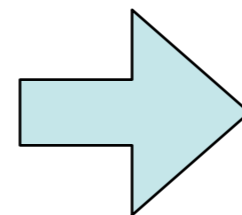
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

This is the fraction of predictions that are correct

range: [0 .. 1]

confusion matrix

| | predict yes | predict no |
|------------|-------------|------------|
| actual yes | 131 TP | 21 FN |
| actual no | 7 FP | 212 TN |



Accuracy = 0.92

Classifier performance: sensitivity/specificity

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

the ability to detect positives

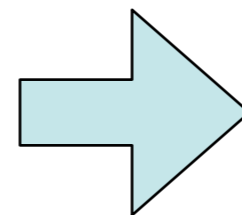
$$\text{Specificity} = \frac{TN}{TN + FP}$$

the ability to reject negatives

range: [0 .. 1]

confusion matrix (cancer)

| | predict yes | predict no |
|------------|-------------|------------|
| actual yes | 131 TP | 21 FN |
| actual no | 7 FP | 212 TN |



Sensitivity = 0.86

Specificity = 0.97

Classifier performance: Matthews correlation coefficient

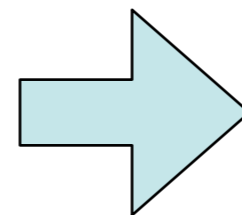
$$MCC = \frac{(TP \cdot TN) - (FN \cdot FP)}{\sqrt{(TN + FN)(TN + FP)(TP + FN)(TP + FP)}}$$

A single performance measure that is less influenced by unbalanced test sets

range: [-1 .. 1]

confusion matrix (cancer)

| | predict yes | predict no |
|------------|-------------|------------|
| actual yes | 131 TP | 21 FN |
| actual no | 7 FP | 212 TN |



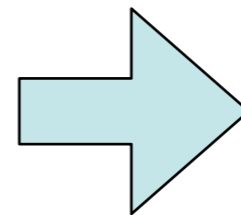
MCC = 0.84

The problem with unbalanced test data

A new classifier: does patient have tuberculosis?

confusion matrix

| | predict yes | predict no |
|------------|-------------|------------|
| actual yes | 5345 TP | 21 FN |
| actual no | 113 FP | 18 TN |



Accuracy = 0.98 (?)

For patients who do not really have tuberculosis:
 this classifier is usually wrong!

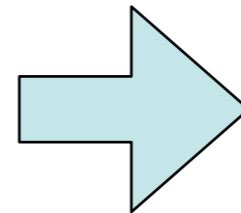
>>> Accuracy can be misleading, especially when the test cases
 are unbalanced

Evaluating unbalanced test data

The tuberculosis classifier

confusion matrix

| | predict yes | predict no |
|------------|-------------|------------|
| actual yes | 5345 TP | 21 FN |
| actual no | 113 FP | 18 TN |



Accuracy = 0.98



Sensitivity = 0.996

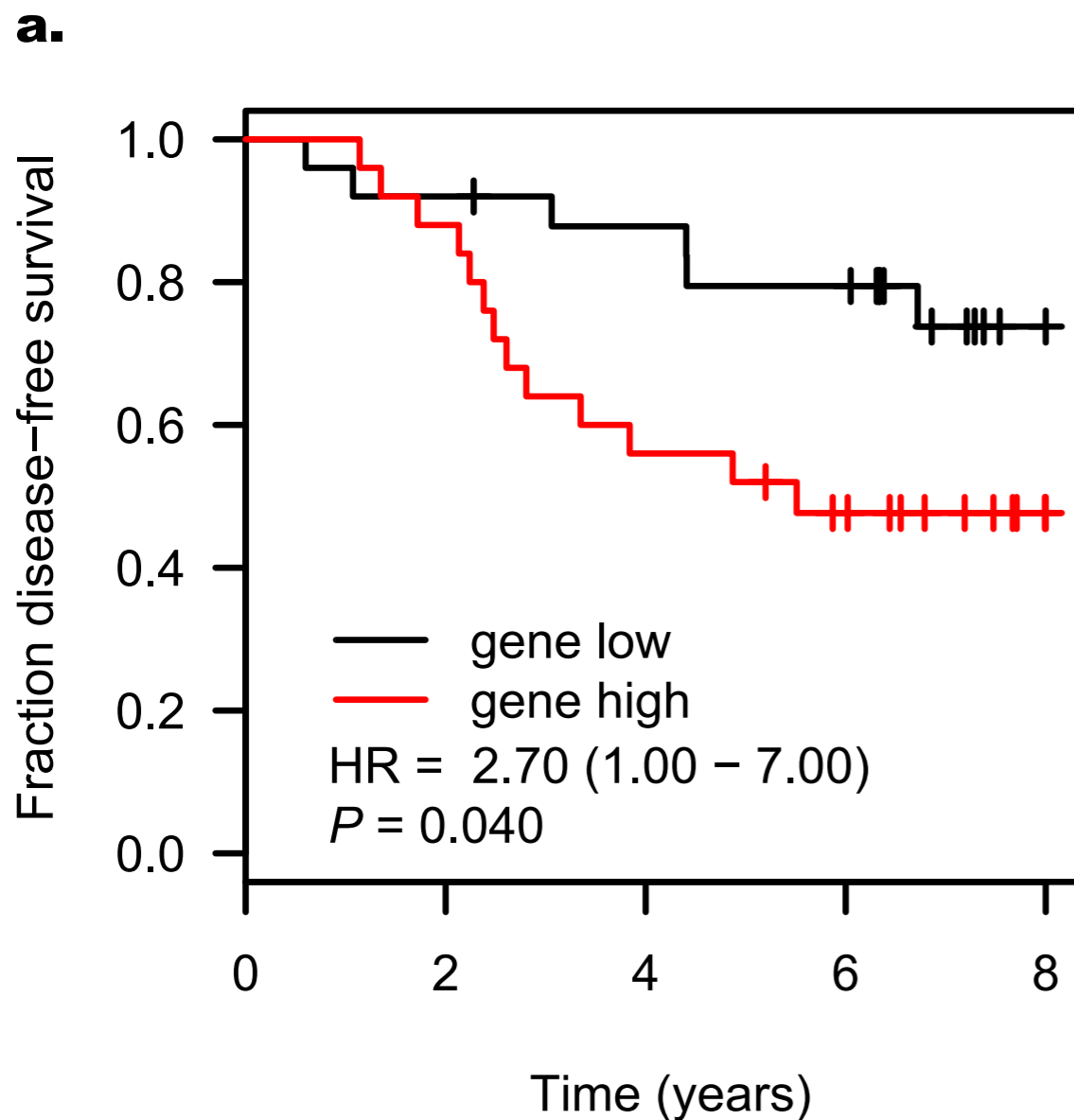
Specificity = 0.14



MCC = 0.24

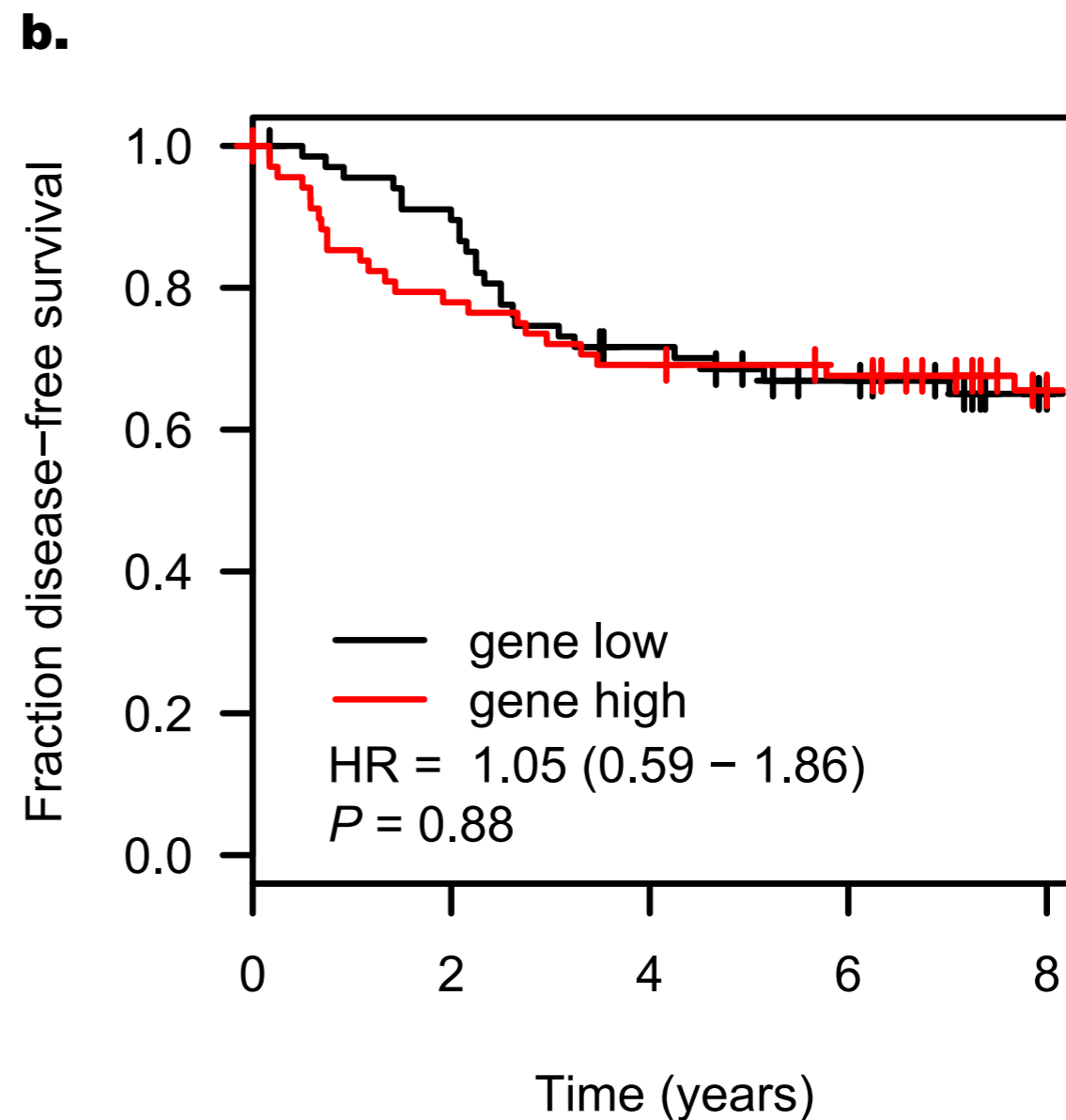


Other performance metrics: Survival



Number at risk

| | | | | |
|----|----|----|----|---|
| 25 | 23 | 21 | 19 | 8 |
| 25 | 22 | 14 | 10 | 1 |

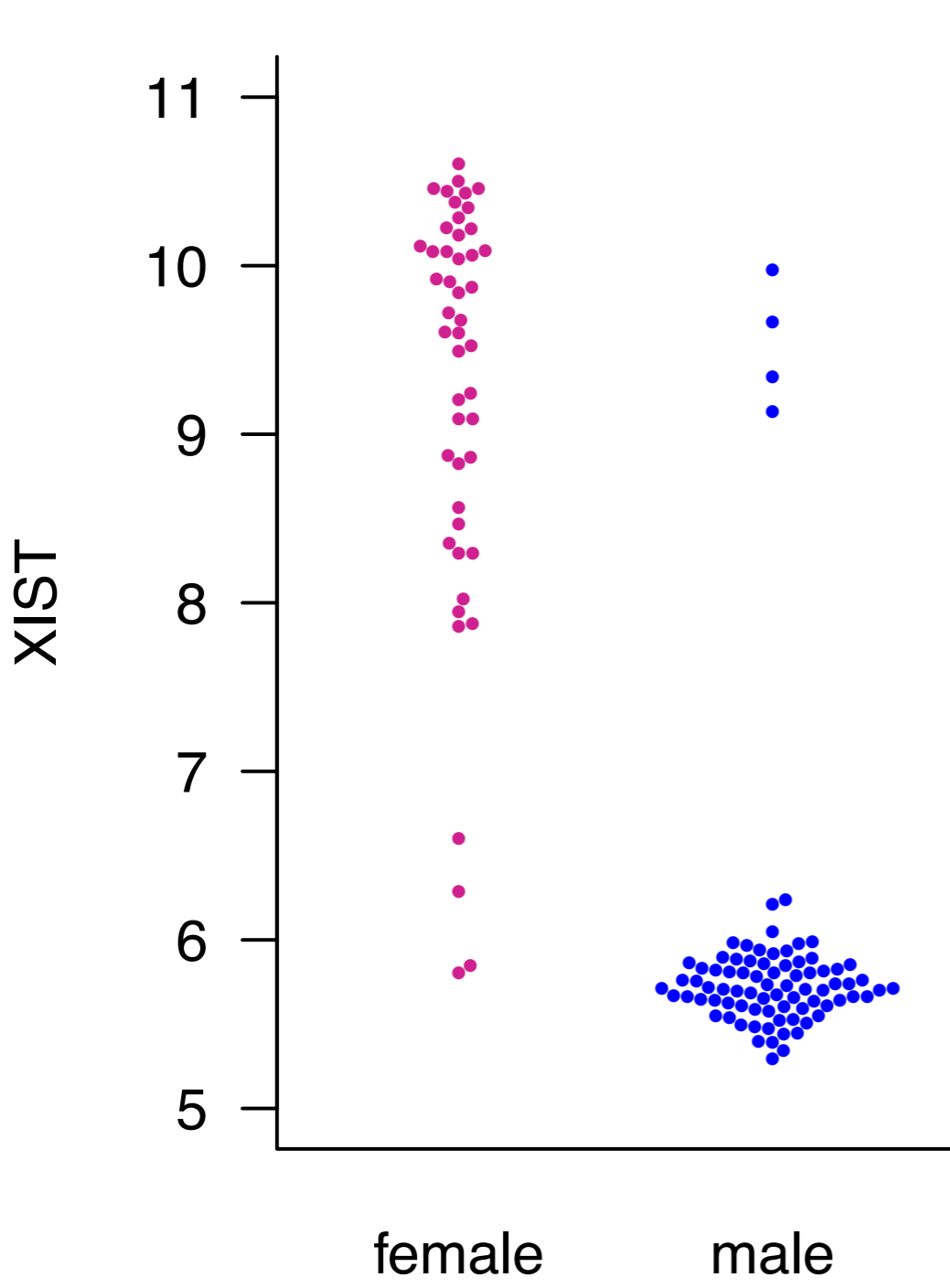


Number at risk

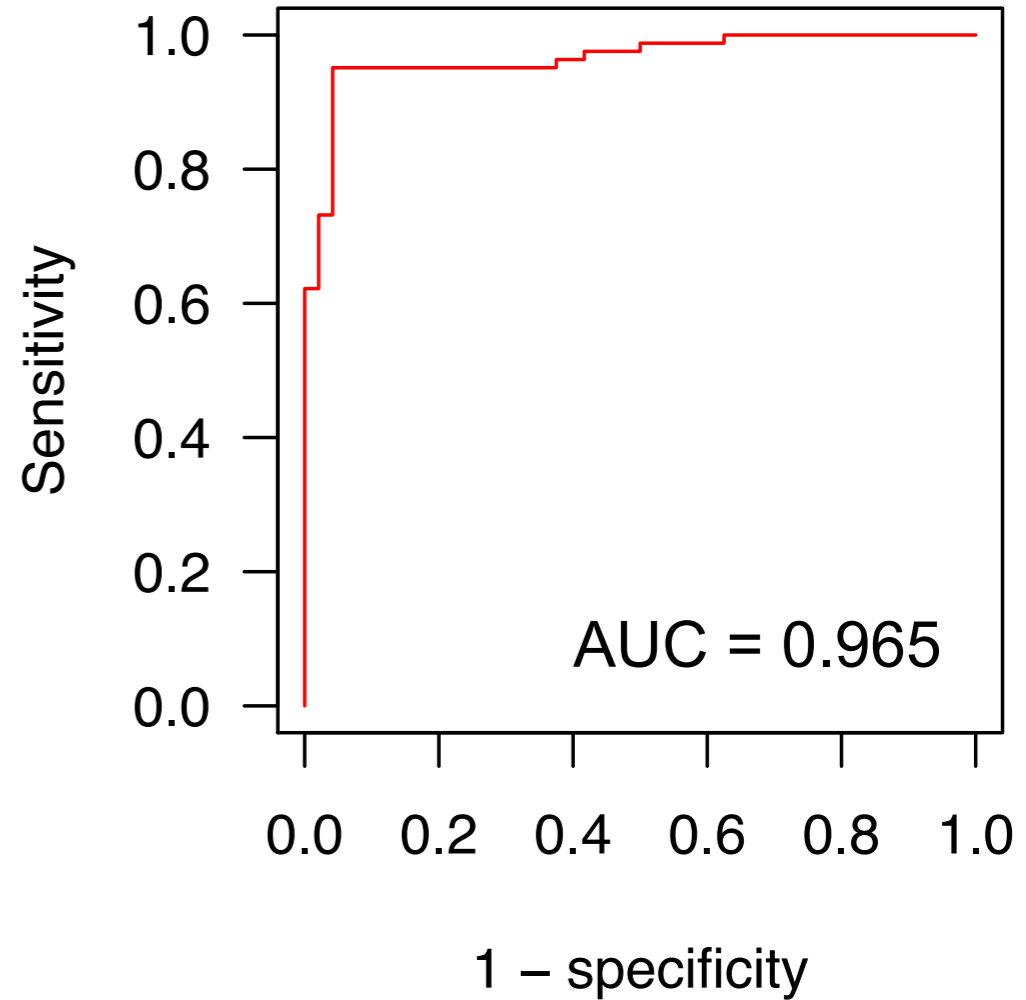
| | | | | |
|----|----|----|----|----|
| 68 | 60 | 46 | 39 | 29 |
| 69 | 53 | 47 | 44 | 31 |

Other performance metrics: ROC curve

What if we don't know where to set the threshold?



ROC = "Receiver operating characteristics"



AUC = "Area under the (ROC) curve"

Summary: classification with machine learning

1. Choose a classification method
 - kNN, LDA, SVM, etc.

2. Feature selection / dimension reduction
 - PCA, possibly t-tests

3. Train the classifier
 - use cross-validation to optimize parameters

4. Assess classifier performance
 - use an independent test set if possible
 - determine sensitivity and specificity when appropriate

The data set in the exercise

Genome-wide expression profiling of human blood reveals biomarkers for Huntington's disease.

Borovecki F, Lovrecic L, Zhou J, Jeong H, Then F, Rosas HD, Hersch SM, Hogarth P, Bouzou B, Jensen RV, Krainc D.

Proc Natl Acad Sci U S A. 2005 Aug 2;102(31):11023-8.

31 samples:

- 14 normal
- 5 presymptomatic
- 12 symptomatic

Affymetrix HG-U133A arrays

Huntingon's disease:

- neurological disorder
- polyglutamine expansion in the huntingtin gene

Why search for marker of disease *progression* (not diagnosis)?

- assess treatment efficacy
- surrogate endpoint in drug trials

Break 09:45 – 10:00