Some Things Every Biologist Should Know About Machine Learning

Artificial Intelligence is no substitute for the real thing

***W**e are drowning in information and starving for knowledge.*
Rutherford D. Roger

Robert Gentleman

Types of Machine Learning

- Supervised Learning
  - classification
- Unsupervised Learning
  - clustering
  - class discovery
- Feature Selection
  - identification of features associated with good prediction

Components of Machine Learning

- **features**: which variables or attributes of the samples are going to be used to cluster or classify
- **distance**: what method will we use to decide whether two samples are similar or not
- **model**: how do we cluster or classify
  - eg: kNN, neural nets, hierarchical clustering

Components of Machine Learning

Once these have been selected (or a set of candidates) we can use cross-validation to:

1. estimate the generalization error
2. perform model selection (could select distance or features as well)
3. feature selection (in a different way to 2)

The No Free Lunch Theorem

- the performance of all optimization procedures are indistinguishable when averaged over all possible search spaces
- hence there is **no** best classifier
- issues specific to the problem will be important
- human or domain specific guidance will be needed

The Ugly Duckling Theorem

- there is no canonical set of features for any given classification objective
- Nelson Goodman (Fact, Fiction, Forecasting)
  - any two things are identical in infinitely many ways
  - a choice of features, based on domain specific knowledge, is essential

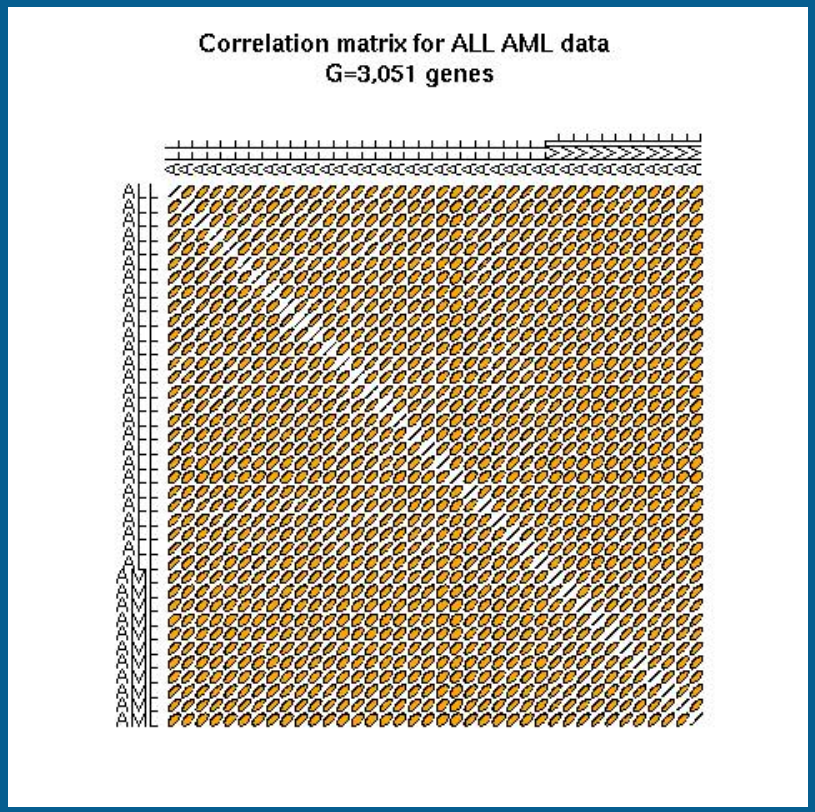
Distance

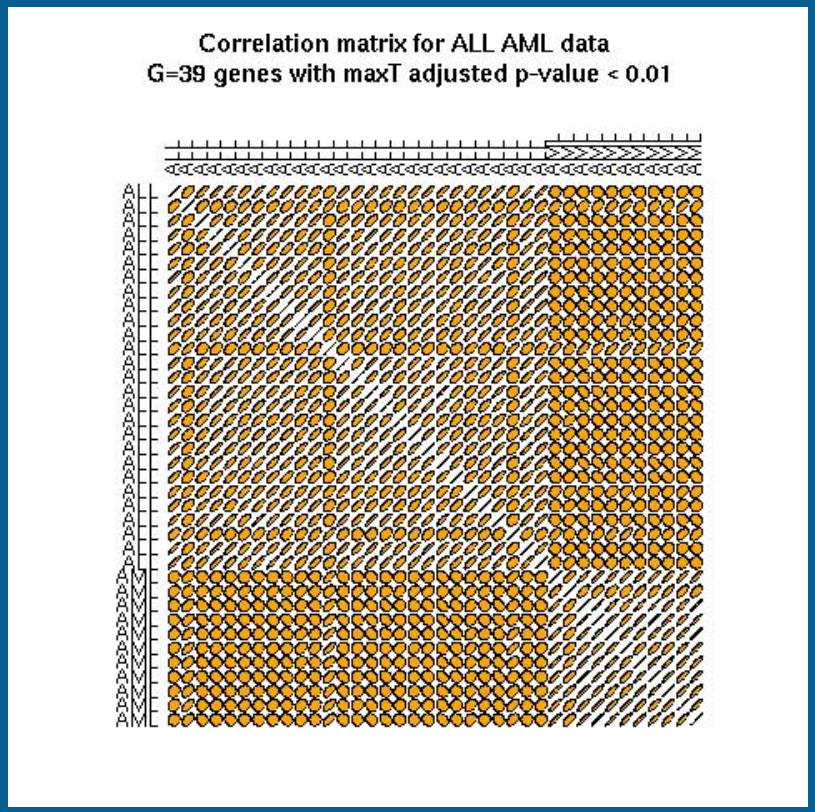
- **all** (every!) machine learning tool relies on some measure of distance between samples
- you **must** be aware of the distance function being used
- some ML algorithms have an implicit distance (but it is there none the less)

Getting to Know Your Data

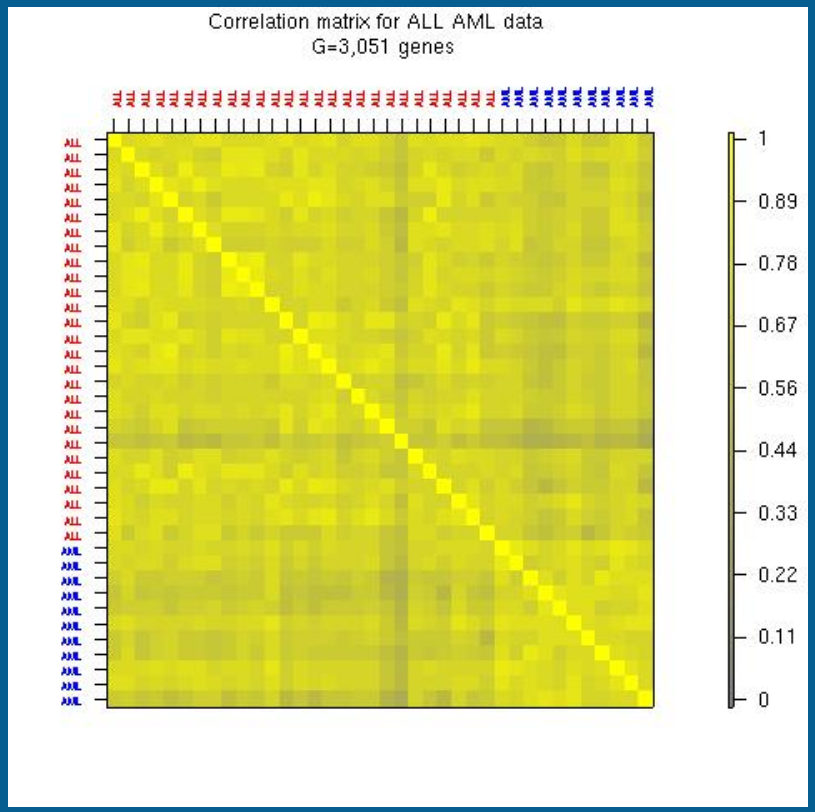
- statisticians call this EDA (Exploratory Data Analysis)
- it generally consists of some model free examinations of the data to ensure some general consistency with expectations

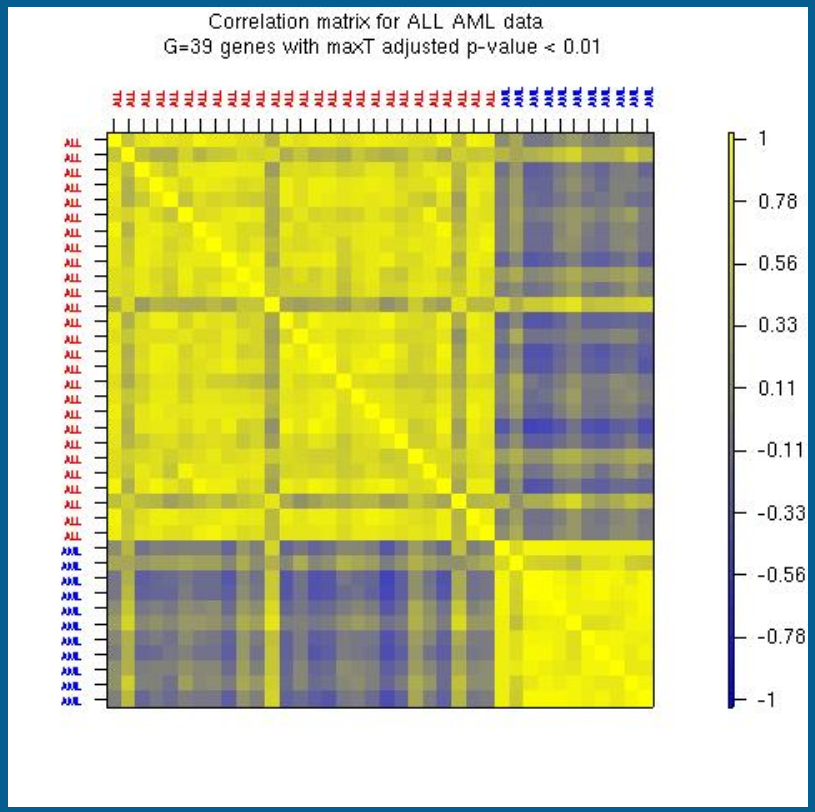
Correlation matrices





Correlation matrices





Distances

- inherent in all machine learning is the notion of distance
- there are very many different distances (Euclidean, Manhatten, 1-correlation)
- the choice of distance is **important** and in general substantially affects the outcome
- the choice of distance should be made carefully

Distances

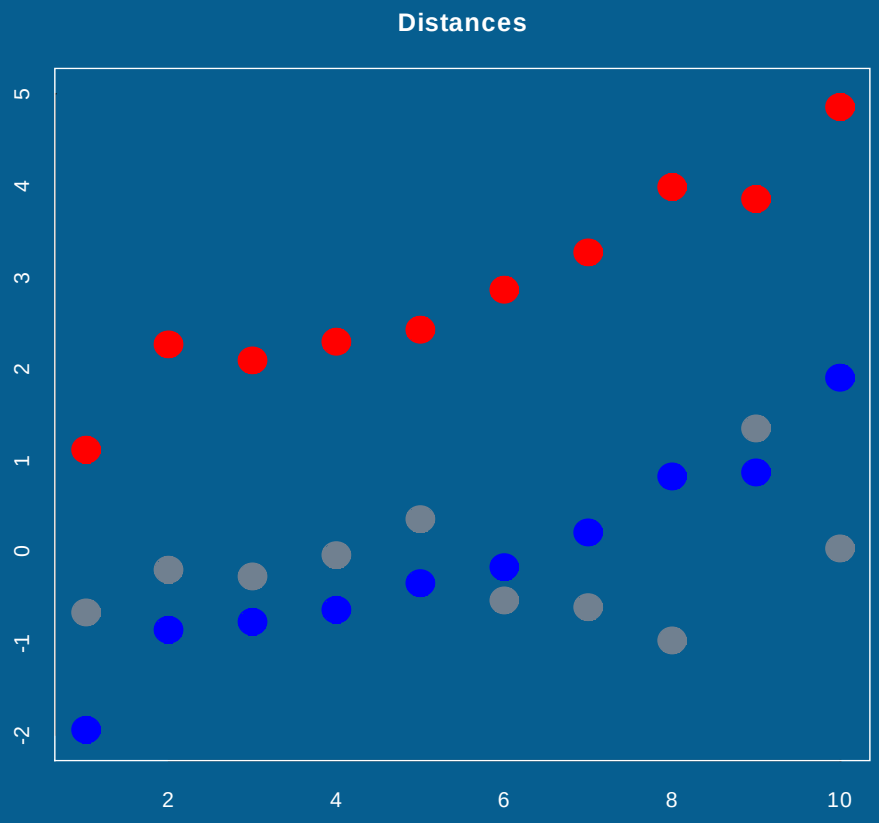
- distances can be thought of as matrices where the value in row *i* column *j* is the distance between sample *i* and sample *j* (or between genes *i* and *j*)
- these matrices are called distance matrices
- in most cases they are symmetric

Distances

- clustering methods work directly on the distance matrix
- Nearest-Neighbor classifiers use distance directly
- Linear Discriminant Analysis uses Mahalanobis distance
- Support Vector Machines are based on Euclidean distance between observations

Distances

- the Correlation distance
  - red-blue is 0.006
  - red-gray is 0.768
  - blue-gray is 0.7101
- Euclidean distance:
  - red-blue is 9.45
  - red-gray is 10.26
  - blue-gray is 3.29



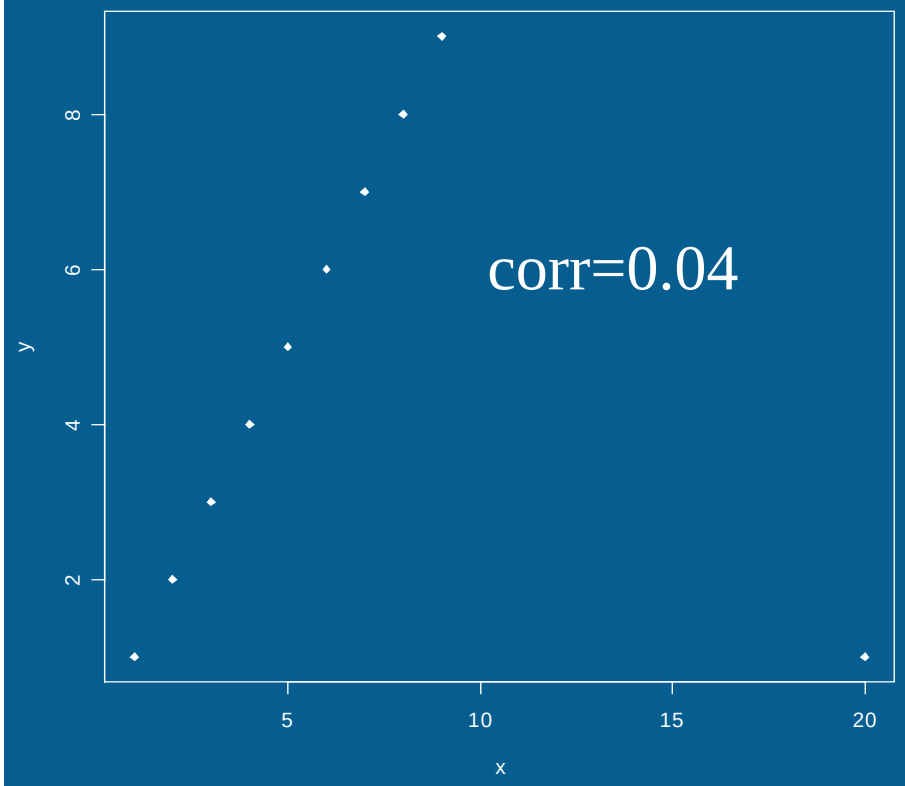
Distance

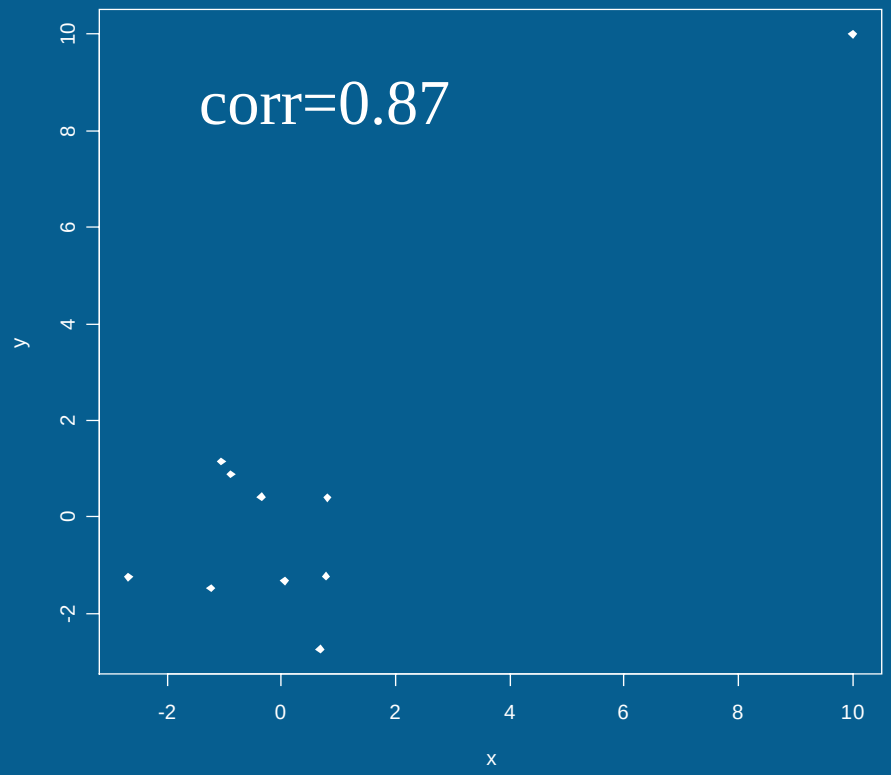
- it is not simple to select the distance function
- you should decide what you are looking for
  - patterns of expression in a time course experiment
  - genes related because they are affected by the same transcription factor
  - samples with known phenotypes and related expression profiles

Distances: Time-course

- you might want genes that are
  - correlated
  - anti-correlated
  - lagged
- 1-correlation is the correct distance only for the first one of these
- correlation measures linear association and is not resistant (one outlier can ruin it)

Correlations gone wrong





Distances: Transcription Factors

- suppose that we can induce a specific transcription factor
- we might want to find all direct targets
- does anyone know what the pattern of expression should be?
- use some known targets to help select a distance

Distances: Phenotype

- T-ALL can be classified according to their stage of differentiation (T1,T2,T3,T4)
- this is done on the basis of the detection of antigens on the surface of the cell
- these antigens can be directly associated with a gene
- look at the expression of those genes and use that to help find/select genes like the known ones

Multidimensional Scaling

- distance data is very high dimensional
- if we have N samples and G genes
- then distance between sample *i* and *j* is in G dimensional space
- this is very hard to visualize and hence methods that can reduce that dimensionality to two or three dimensions are interesting
- but only if they provide a reasonable reduction of the data

MDS

- three main ways of doing this
  - **classical MDS**
  - **Sammon mapping**

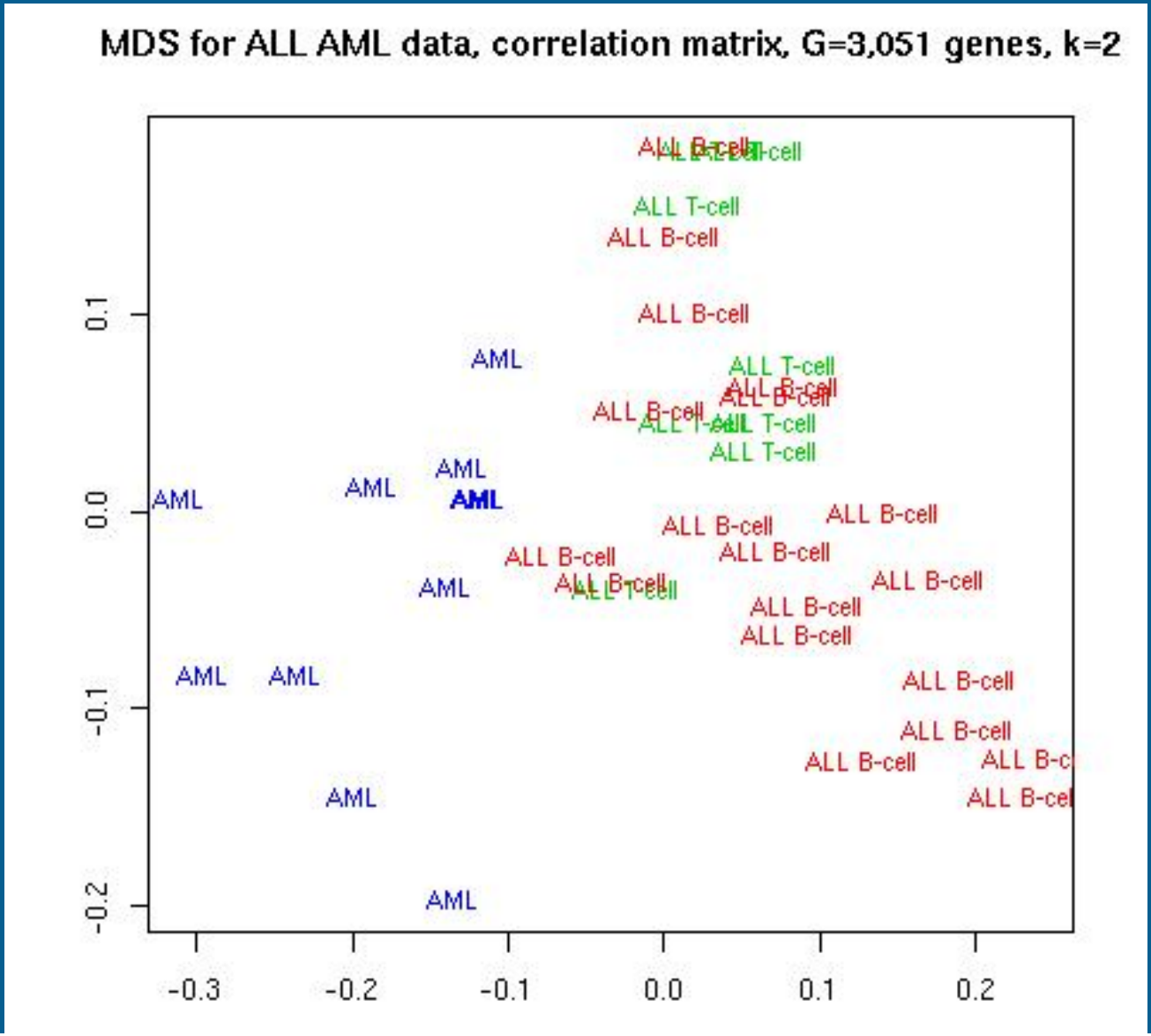
    places more emphasis on smaller dissimilarities
  - **Shepard-Kruskal non-metric scaling**

    based on the order of the distances not their values

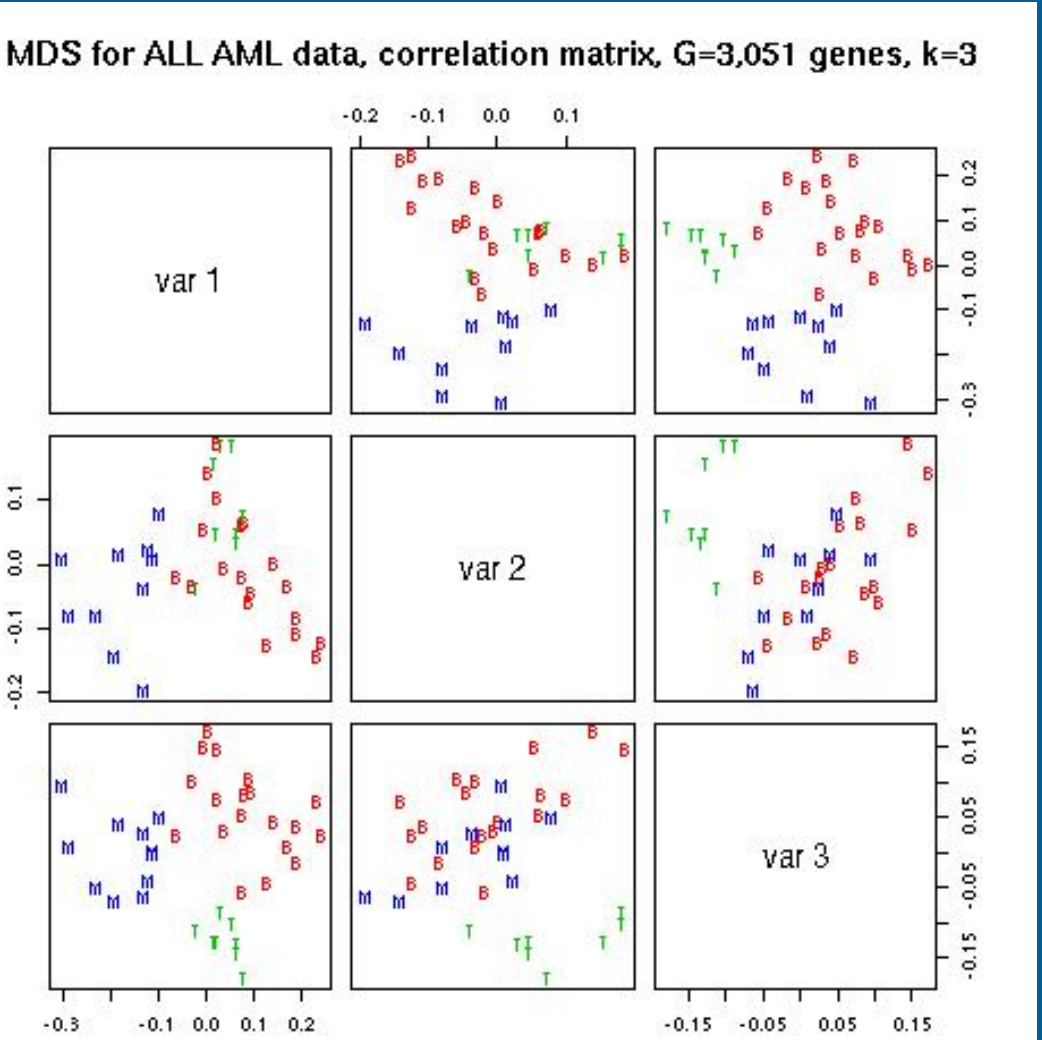
MDS

- the quality of the representation in *k* dimensions will depend on the magnitude of the first *k* eigenvalues.
- The data analyst should choose a value for *k* that is small enough for ease representation but also corresponds to a substantial "proportion of the distance matrix explained".

Classical MDS



Classical MDS



$$\frac{|\lambda_1| + |\lambda_2|}{\sum |\lambda_i|} = 0.43$$

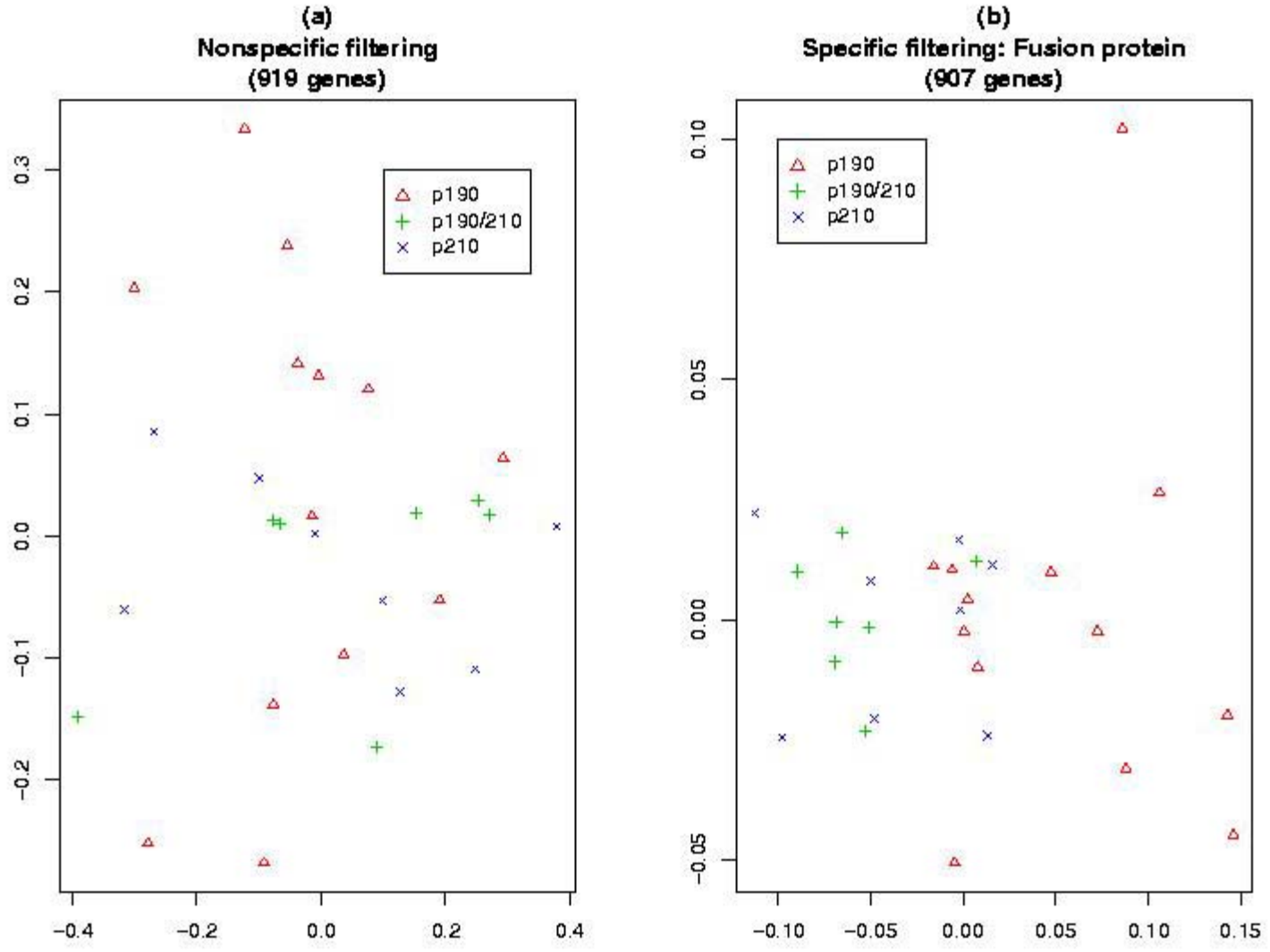
$$\frac{|\lambda_1| + |\lambda_2| + |\lambda_3|}{\sum |\lambda_i|} = 0.55$$

MDS

- **N.B.** The MDS solution reflects not only the choice of a distance function, but also the **features selected.**
- If features were selected to separate the data into two groups (e.g., on the basis of two-sample t-statistics), it should come as no surprise that an MDS plot has two groups. In this instance MDS is not a confirmatory approach.

$$\frac{|\lambda_1|+|\lambda_2|}{\sum|\lambda_i|}=0.63$$

$$\frac{|\lambda_1|+|\lambda_2|}{\sum|\lambda_i|}=0.88$$



Supervised Learning

- the general problem:

  Identify mRNA expression patterns that reliably predict phenotype.

Supervised Learning: 4 Steps

1. **feature selection**: includes transformation, eg: log(x), x/y, etc
2. **model selection**: involves distance selection
3. **training set**: used to determine the model parameters
4. **test set**: should be independent of the training set and it is used to assess the performance of the classifier from Step 2

Supervised Learning: Goal

To identify a set of features, a predictor (classifier) and all parameters of the predictor so that if presented (with a new sample we can predict its class with an error rate that is similar to that obtained in Step 4).

Supervised Learning: Problems

- to reliably estimate the error rate will require an enormous sample (if it is small)
- therefore the test set is wasteful in practice; samples are expensive and valuable
- if there are lots of features we cannot hope to explore all possible variants
- there are too many models
- there are too many distances

A Simpler Goal

- we want some form of generalizability
- we want to select features and a model that are appropriate for prediction of new cases

(not looking for Mr. Right but rather Mr. NotTooWrong)

- and in a slightly different form:

all models are wrong, but some models are useful

Supervised Learning

- **training error/prediction error**: this is the error rate on the training sample
- the training error is overly optimistic
- **the test error/generalization error:** is the error rate that will occur when a new independent sample is used (randomly chosen from the population of interest)

Supervised Learning

- there is sometimes benefit in considering class specific error rates
- some classes may be easy to predict and others hard
- especially if classes are not equally represented in the sample (or if we want to treat the errors differently)

Machine Learning: Mathematics

- Let Y denote the true class and X denote features chosen from the available set $\mathcal{X}$
- Suppose that $Y = f(X) + \varepsilon$
- so the true class is some function f of the features plus some random error
- so we must extract X from $\mathcal{X}$
- then estimate model parameters to get $\hat{f}$
- finally get $\hat{y} = \hat{f}(X)$

Machine Learning: Mathematics

- the training set gives us observations for which we know both *y* and *x* – the true class and the features
- we select the parameters of the model so that we minimize (in some way) the errors
- e.g. we want to find functions that minimize

$$\sum \left(y_i - \hat{f}(x_i) \right)^2$$

- there are an infinite number of functions that make this zero

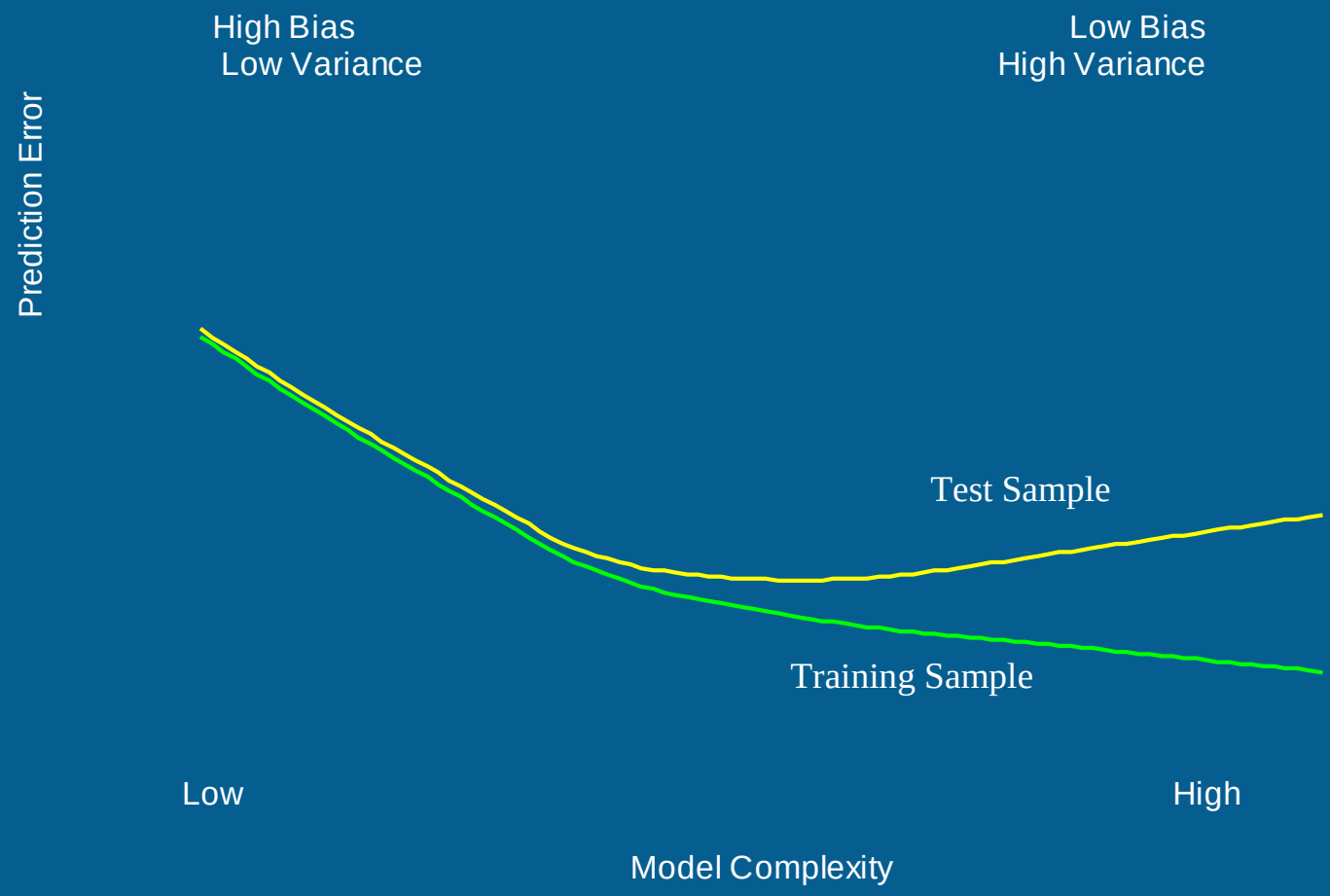
Supervised Learning

- so we must put some restrictions on the class of models that we will consider
- it is also worth observing at this time that model complexity is clearly an issue
- more complex models fit better
- in any comparison of models it is essential that the complexity be adjusted for
- Occam's Razor: *we prefer simple explanations to complex ones*

Supervised Learning

- **bias**: the difference between what is being predicted and the truth
- **variance**: the variability in the estimates
- generally low bias and low variance are preferred
- it is difficult to achieve this

Model Complexity



Supervised Learning

- The classifier can make one of three decisions:
  - classify the sample according to one of the phenotypic groups
  - **doubt**: it cannot decide which group
  - **outlier**: it does not believe the sample belongs to any group

Supervised Learning

- Suppose that sample ***i*** has feature vector ***x***
- The decision made by the classifier is called $\hat{f}(x)$ and the true class is ***y***
- We need to measure the cost of identifying the class as $\hat{f}(x)$ when the truth is ***y***
- this is called the **loss function**
- the loss will be zero if the classifier is correct and something positive if it is not

Loss Functions

- loss functions are important concepts because they can put different weights on different errors
- for example, mistakenly identifying a patient who will not achieve remission as one who will is probably less of problem than the reverse – we can make that loss/cost much higher

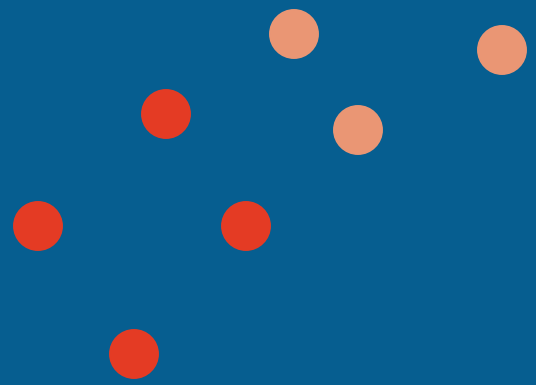
Feature Selection

- in most of our experiments the features must be selected
- part of what we want to say is that we have found a certain set of features (genes) that can accurately predict phenotype
- in this case it is important that feature selection be included in any error estimation process

Classifiers

- *k*-NN classifiers – the predicted class for the new sample is that of the *k*-NNs
- doubt will be declared if there is not a majority (or if the number required is too small)
- outlier will be declared if the new sample is too far from the original data

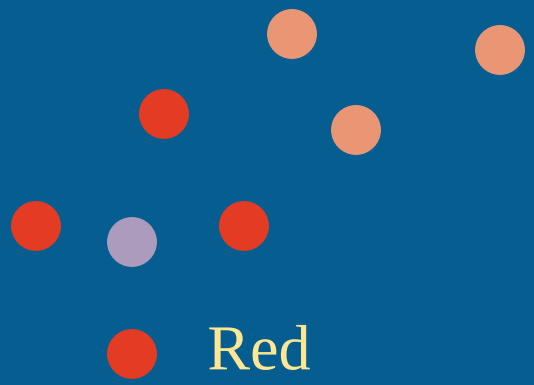
k-NN Classifier



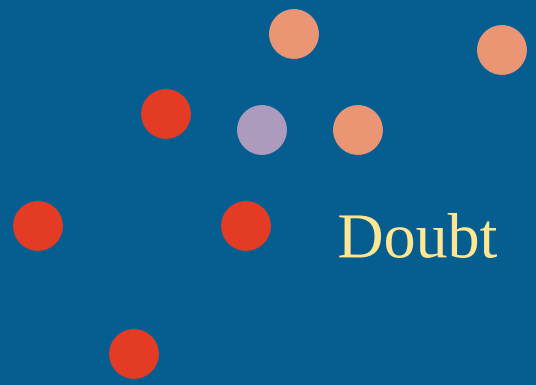
k-NN Classifier

Outlier

k-NN Classifier

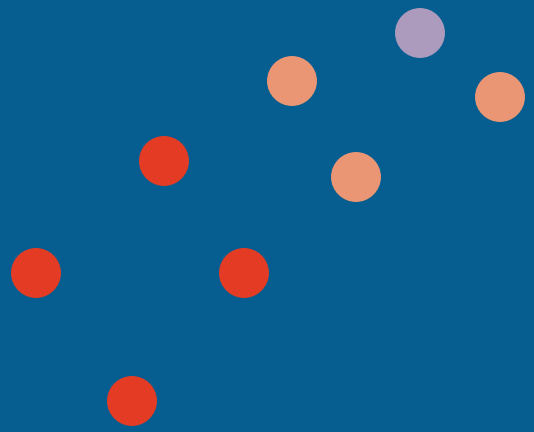


k-NN Classifier



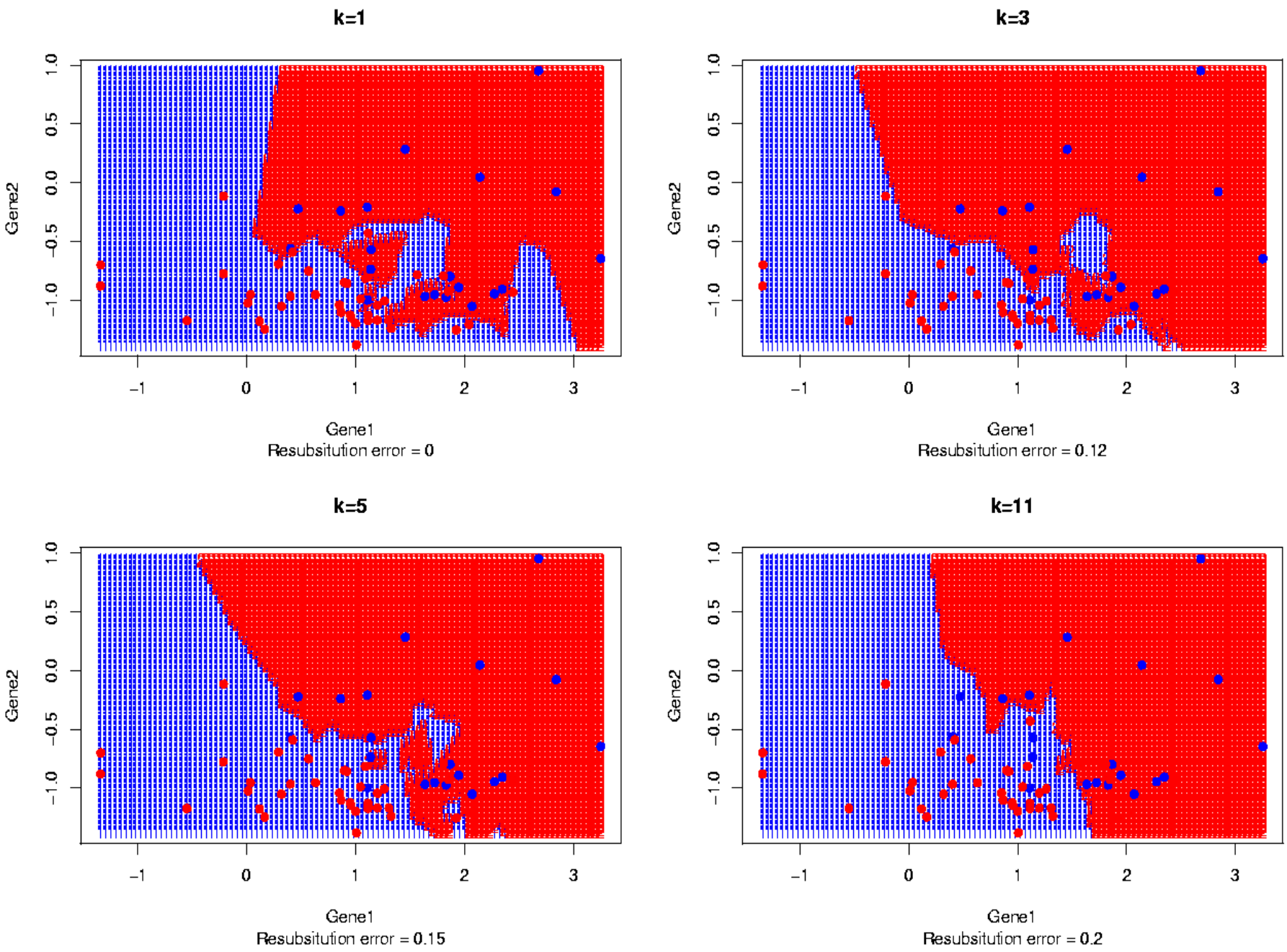
k-NN Classifier

Orange



k-NN

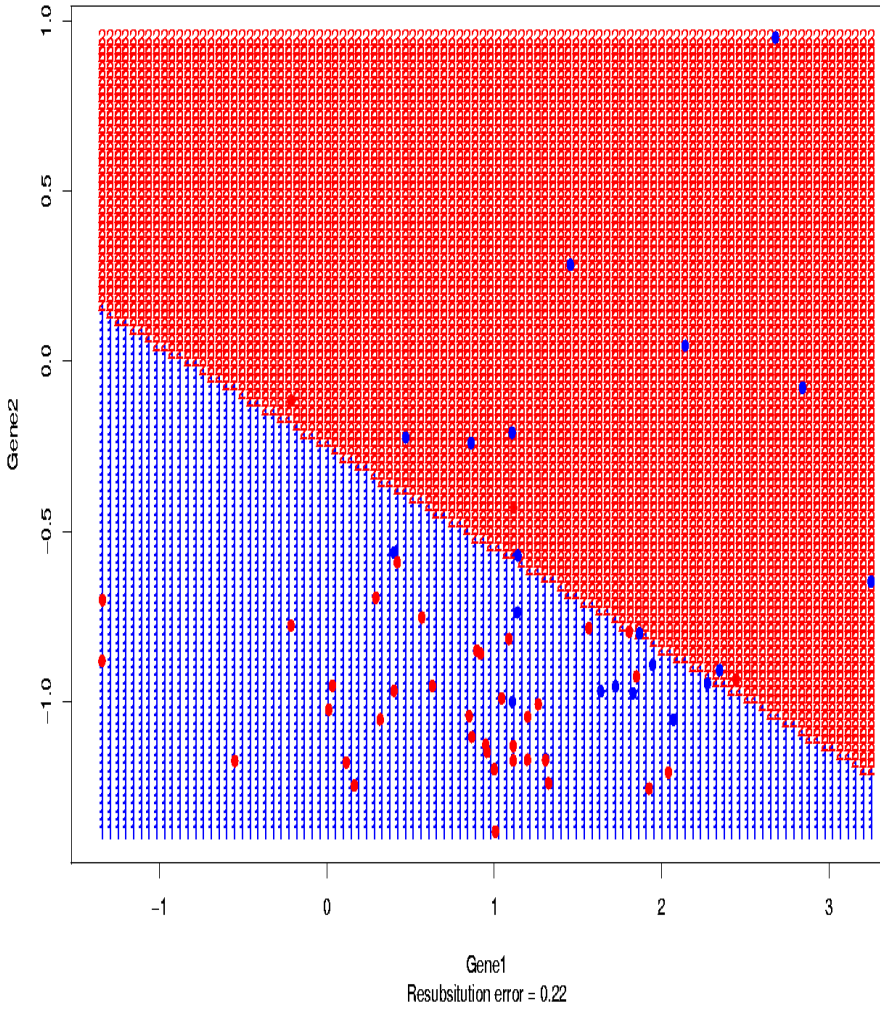
- larger values of *k* correspond to less complex models
- they typically have low variance but high bias
- small values of *k* (*k*=1) are more complex models
- they typically have high variance but low bias


k=1
Gene2
Gene1
Resubsitution error = 0
k=3
Gene2
Gene1
Resubsitution error = 0.12
k=5
Gene2
Gene1
Resubsitution error = 0.15
k=11
Gene2
Gene1
Resubsitution error = 0.2

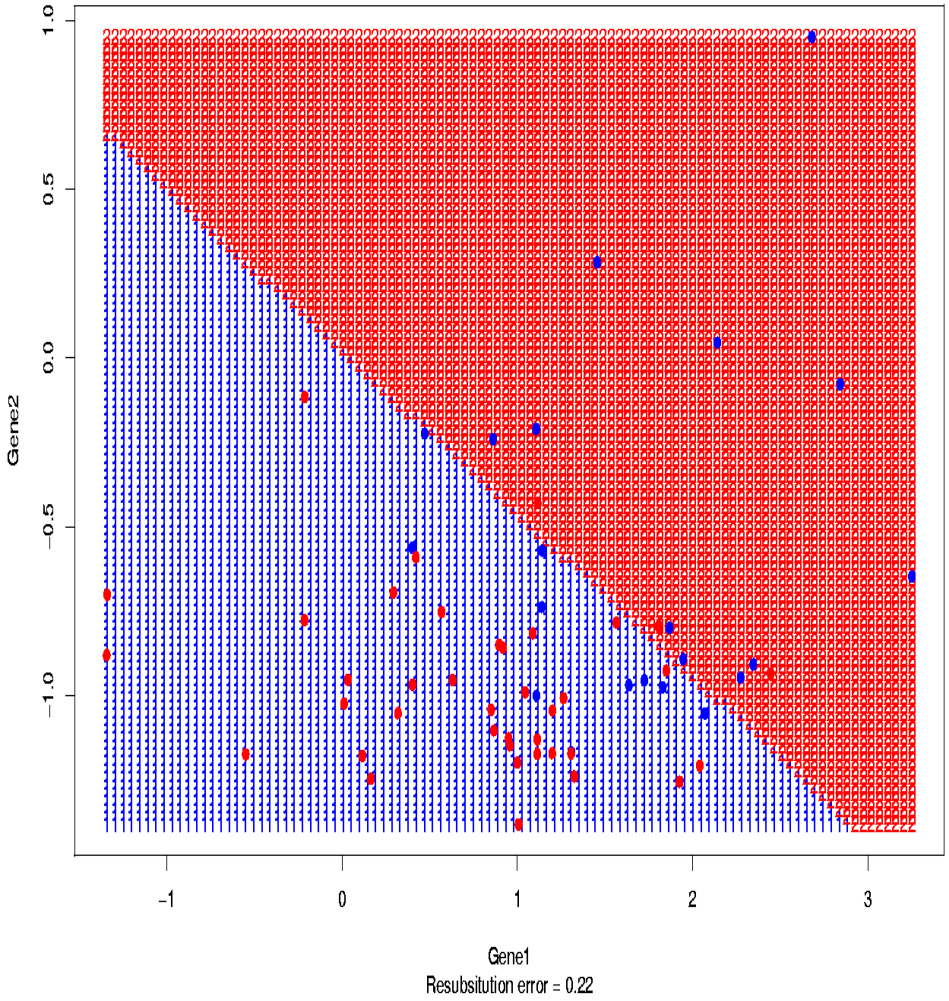
Discriminant Analysis

- we contrast the k-NN approach with linear and quadratic discriminant analysis (lda, qda)
- lda seeks to find a linear combination of the features which maximizes the ratio of its between-group variance to its within group variance
- qda seeks a quadratic function (and hence is a more complex model)

QDA


Gene2
1.0
0.5
0.0
−0.5
−1.0
−1
0
1
2
3
Gene1
Resubsitution error = 0.22

LDA


Gene2
1.0
0.5
0.0
−0.5
−1.0
−1
0
1
2
3
Gene1
Resubsitution error = 0.22

Cross-validation

- while keeping a separate test set is conceptually a good idea it is wasteful of data
- some sample reuse ideas should help us to make the most of our data without unduly biasing the estimates of the predictive capability of the model (if applied correctly)

Cross-validation

- the general principle is quite simple
  - our complete sample is divided into two parts
  - the model is fit on one part and the fit assessed on the other part
  - this can be repeated many times; each time we get an estimate of the error rate
  - the estimates are correlated, but that's ok, we just want to average them

Cross-validation

- leave-one-out is the most popular
- each sample is left out in turn, then the model fit on the remaining N-1 samples
- the left out sample is supplied and its class predicted
- the average of the prediction errors is used to estimate the training error

Cross-validation

- this is a low bias (since N-1 is close to N we are close to the operating characteristics of the test) but high variance
- there are arguments that suggest leaving out more observations each time would be better
- the bias increases but may be more than offset but the reduction in variance

Cross-validation

- Uses include
- ***estimating the error rate***
- ***model selection***: try a bunch of models choose the one with the lowest cross-validation error rate
- ***feature selection***: select features that provide good prediction in most of the subsamples

General Comments

- there is in general no best classifier (there are some theorems in this regard)
- it is very important to realize that if one classifier works very poorly and you try a different classifier which works very well, then someone has probably made a mistake!
- the advantages to SVM or *k*-NN, for example, are not generally so large that one works and the other doesn’t

Unsupervised Learning

- in statistics this is known as clustering
- in some fields it is known as class discovery
- the basic idea is to determine how many *groups* there are in your data and which variables seem to define the groupings
- the number of possible groups is generally huge and so some stochastic component is generally needed

What is clustering?

- Clustering algorithms are methods to divide a set of *n* observations into *g* groups so that within group similarities are larger than between group similarities
- the number of groups, *g*, is generally unknown and must be selected in some way
- implicitly we must have already selected both features and a distance!

Clustering

- the application of clustering is very much and art
- there are interactions between the distance being used and the method
- one difference between this and classification is that there is no training sample and the groups are unknown before the process begins
- unlike classification (supervised learning) there is no easy way to use cross-validation

Clustering

- class discovery: we want to find new and interesting groups in our data
- to do a good job the features, the distance and the clustering algorithm will have to be considered with some care
- the appropriate choices will depend on the questions being asked and the available data

Clustering

- probably some role for outlier
- any group that contained an outlier would probably have a large value for any measure of within cluster homogeneity
- fuzzy clustering plays the role of doubt
  - objects are assigned a weight (or probability of belonging to each cluster)

Clustering: QC

- one of the first things that a data analyst should do with normalized microarray data is to cluster the data
- the clusters should be compared to all known experimental features
  - when the samples were assayed
  - what reagents were used
  - any batch effects

Clustering: QC

- if the clusters demonstrate a strong association with any of these characteristics it will be difficult to interpret the data
- it is important, therefore, to design your experiment
- do not do all the type A samples on day 1 and all the type B on day 2

Aside: Experimental Design

- do not randomly decide which day to do a sample
- instead you should block (and randomize within blocks) to ensure proper balance across all important factors
- e.g half of the A's should be done on day 1 and half on day 2, the same as for the B's (but random assignment won't give you that)

Clustering

Two (and a half) types:

- **hierarchical –** generate a hierarchy of clusters going from 1 cluster to n
- **partitioning –** divide the data into g groups using some (re)allocation algorithm
- **fuzzy clustering**: each object has a set of weights suggesting the probability of it belonging to each cluster

Hierarchical Clustering

Two types

- **agglomerative** – start with n groups, join the two closest, continue
- **divisive** – start with 1 group, split into 2, then into 3,…, into n
- need both between observation distance and between group/cluster distance

Hierarchical Clustering

- between group distances
- *single linkage* – distance between two clusters is the smallest distance between an element of each group
- *average linkage* – distance between the two groups is the average of all pairwise distances
- *complete linkage* – distance is the maximum

Hierarchical Clustering

- agglomerative clustering is not a good method to detect a few clusters
- divisive clustering is probably better
- divisive clustering is not deterministic (as implemented)
- the space of all possible splits is too large and we cannot explore all
- so we use some approximations

Hierarchical Clustering

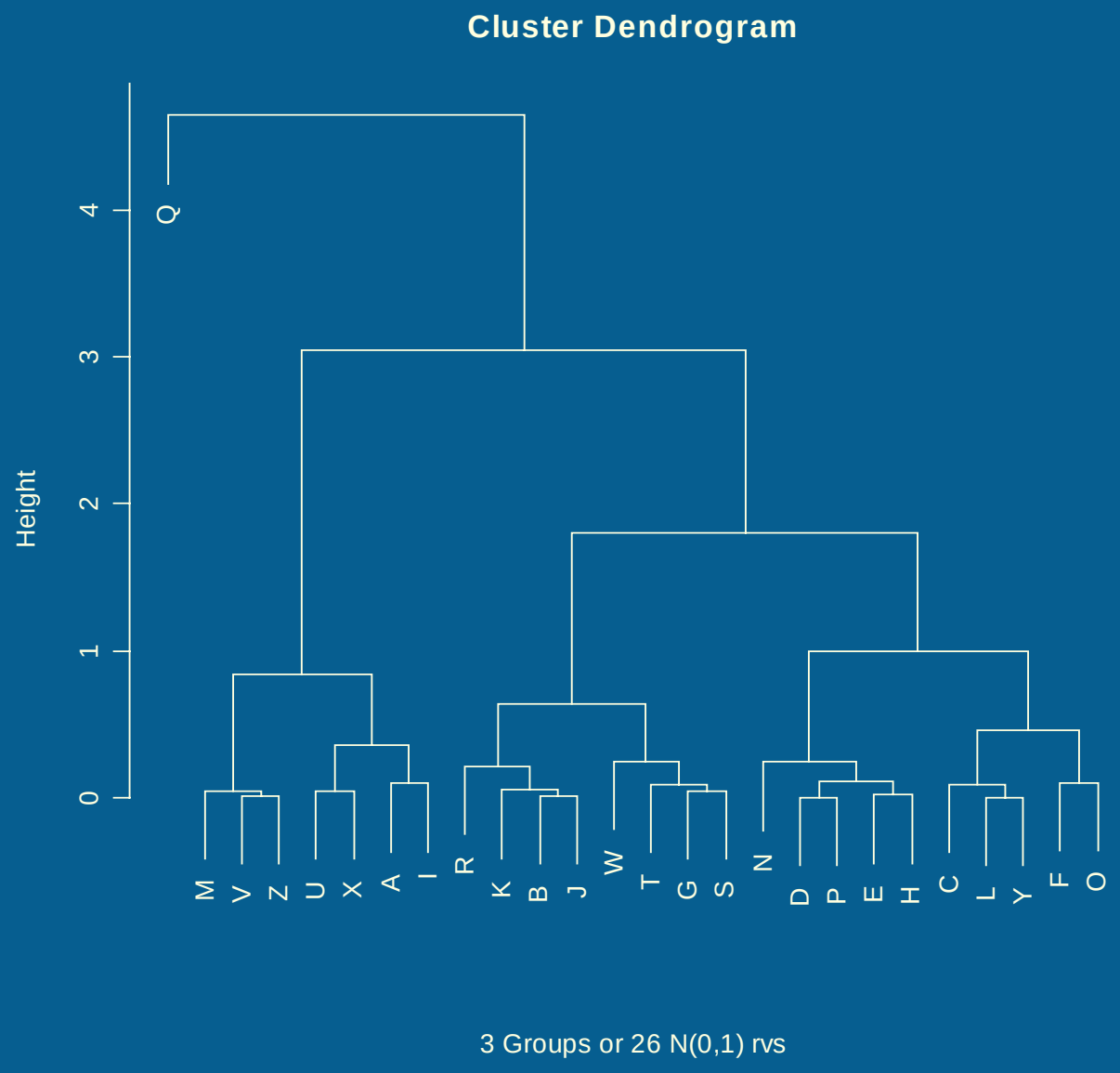
- agglomerative: start with all objects in their own cluster then gradually combine the closest to
- many ways to do this but there is an exact solution
- divisive: start with all objects in the same group, split into two, then three, then…until *n*

Dendrograms

- the output of a hierarchical clustering is usually presented as a dendrogram
- this is a tree structure with the observations at the bottom (the leafs)
- the height of the join indicates the distance between the left branch and the right branch

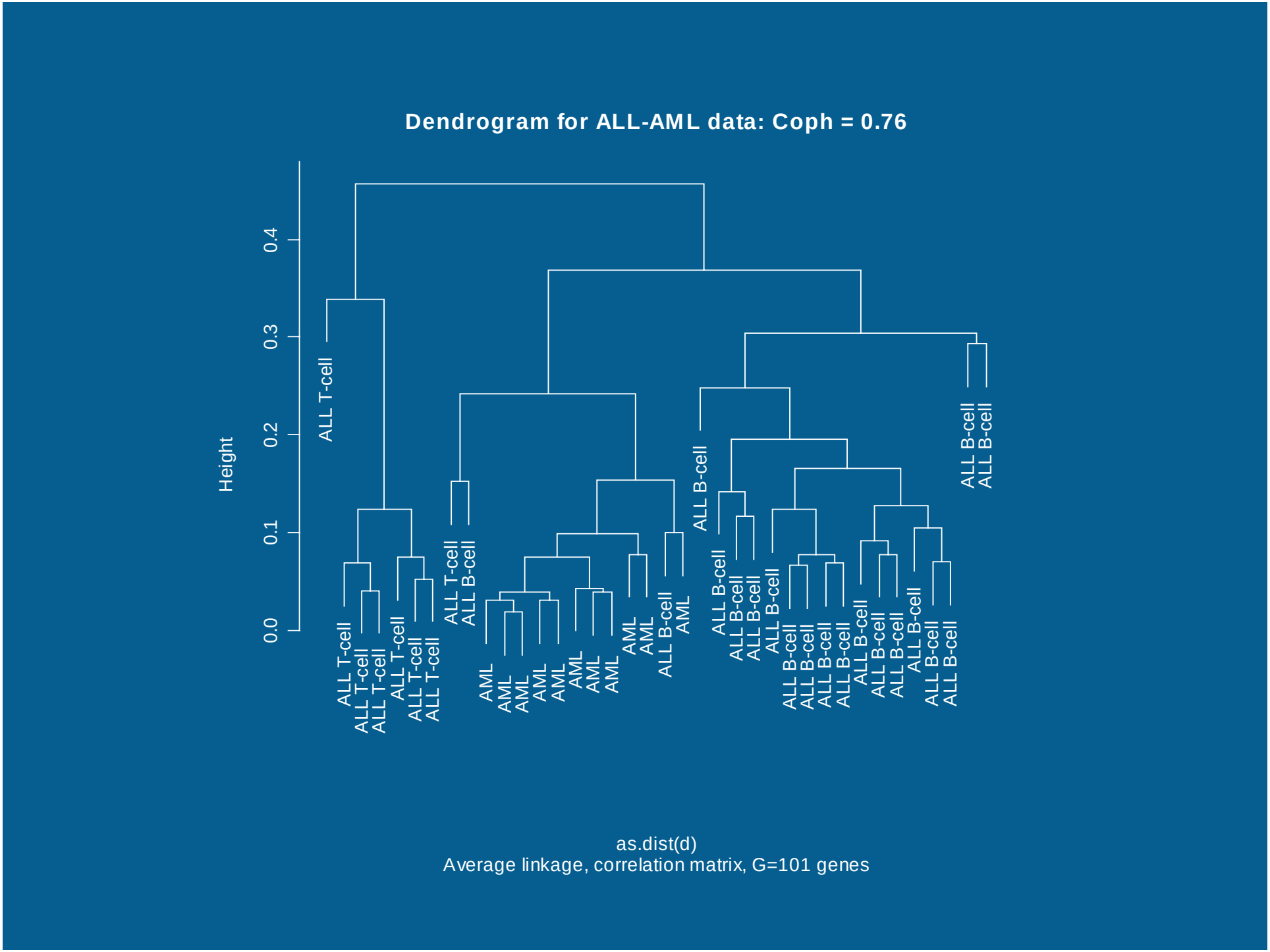
Dendrograms

- dendrograms are NOT visualization methods
- they do not *reveal* structure in data they *impose* structure on data
- the cophenetic correlation can be used to assess the degree to which the dendrogram induced distance agrees with the the distance measure used to compute the dendrogram


Cluster Dendrogram
Height
0
1
2
3
4
Q
M
V
Z
U
X
A
I
R
K
B
J
W
T
G
S
N
D
P
E
H
C
L
Y
F
O
3 Groups or 26 N(0,1) rvs

Dendrograms

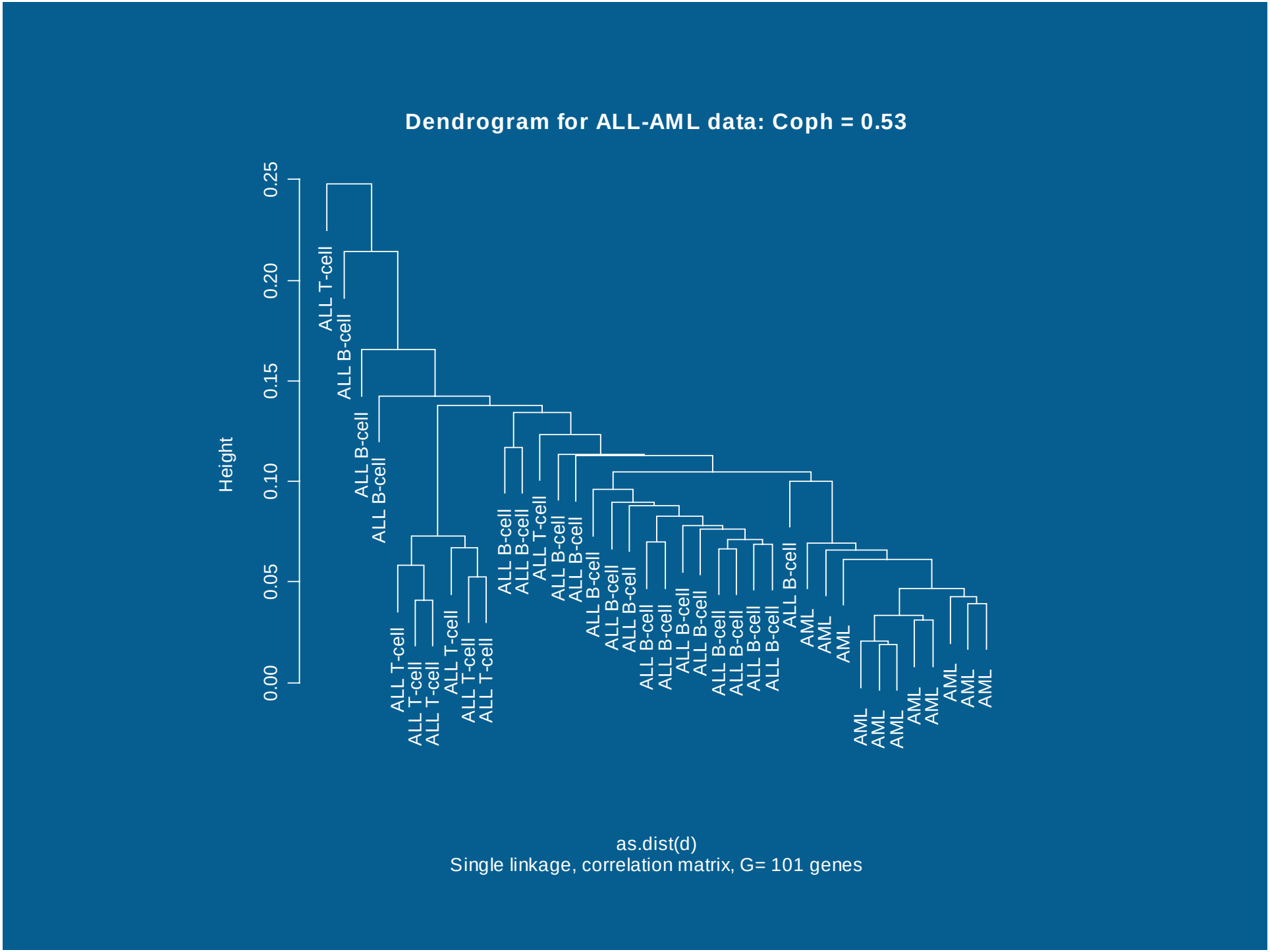
- the cophenetic correlation can help to determine whether the distances represented in the dendrogram reflect those used to construct it
- even if this correlation is high that is no guarantee that the dendrogram represents real clusters


Dendrogram for ALL-AML data: Coph = 0.76
Height
0.0
0.1
0.2
0.3
0.4
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL B-cell
AML
AML
AML
AML
AML
AML
AML
AML
ALL B-cell
AML
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
as.dist(d)
Average linkage, correlation matrix, G=101 genes

- the dendrogram was cut to give three groups

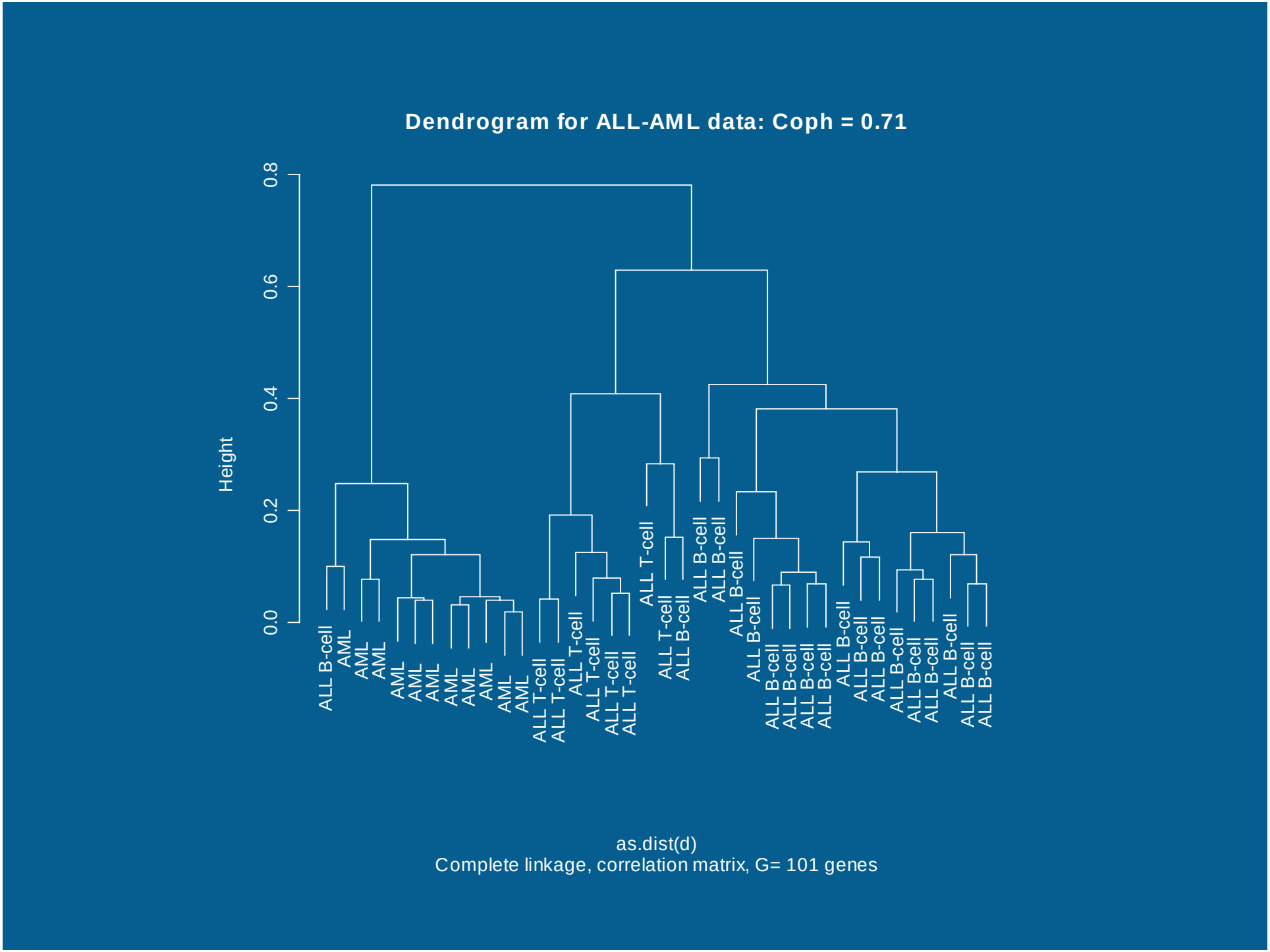
Average Linkage

Group	1	2	3
ALL B-cell	17	2	0
ALL T-cell	0	1	7
AML	0	11	0


Dendrogram for ALL-AML data: Coph = 0.53
Height
0.00
0.05
0.10
0.15
0.20
0.25
ALL T-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL B-cell
ALL B-cell
ALL T-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
AML
AML
AML
AML
AML
AML
AML
AML
AML
AML
AML
as.dist(d)
Single linkage, correlation matrix, G= 101 genes

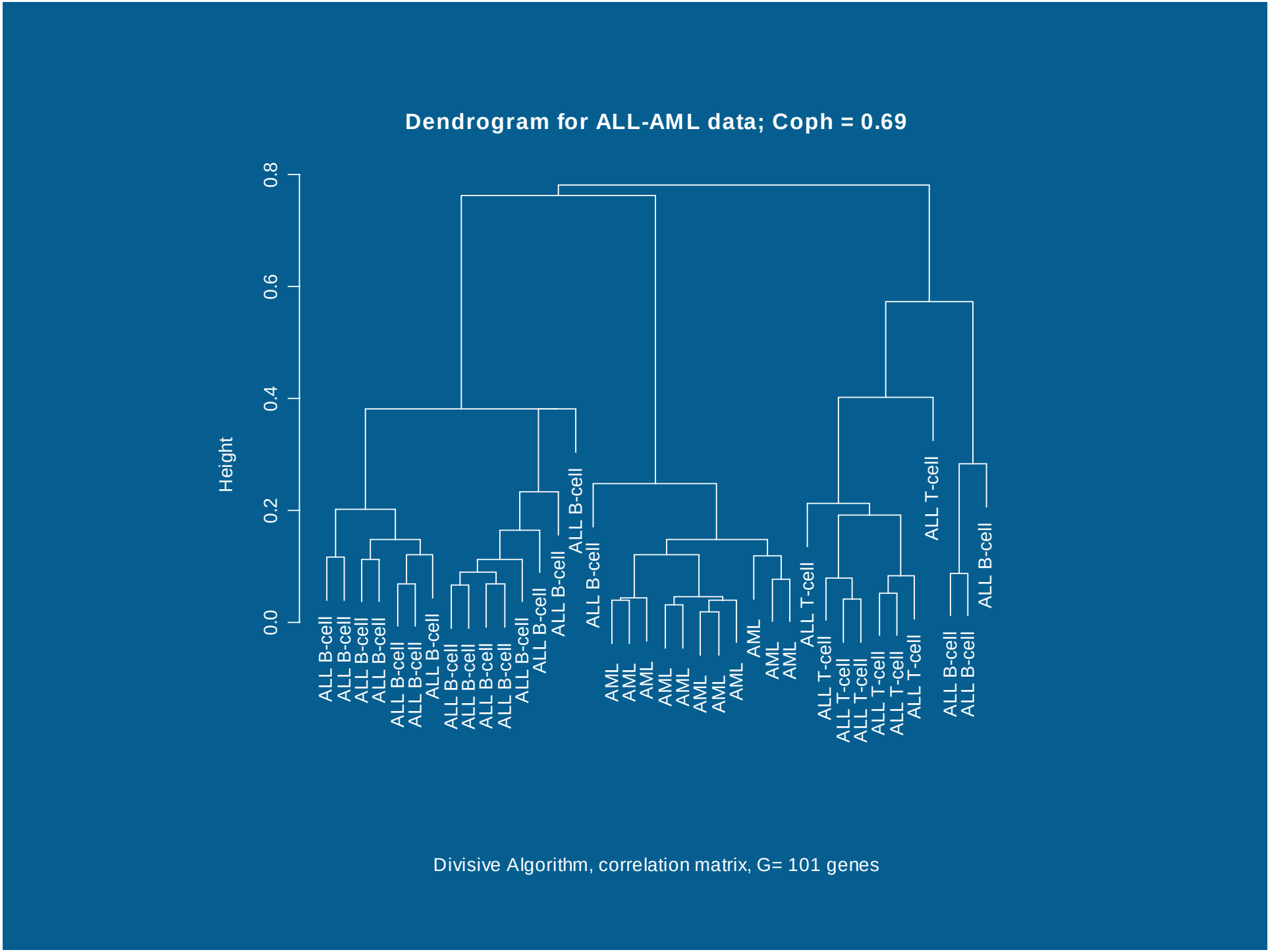
Single Linkage

Group	1	2	3
ALL B-cell	18	0	1
ALL T-cell	7	1	1
AML	11	0	0


Dendrogram for ALL-AML data: Coph = 0.71
Height
0.0
0.2
0.4
0.6
0.8
ALL B-cell
AML
AML
AML
AML
AML
AML
AML
AML
AML
AML
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
as.dist(d)
Complete linkage, correlation matrix, G= 101 genes

Complete Linkage

Group	1	2	3
ALL B-cell	17	1	1
ALL T-cell	0	8	0
AML	0	0	11


Dendrogram for ALL-AML data; Coph = 0.69
Height
0.0
0.2
0.4
0.6
0.8
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
ALL B-cell
AML
AML
AML
AML
AML
AML
AML
AML
AML
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL T-cell
ALL B-cell
ALL B-cell
ALL B-cell
Divisive Algorithm, correlation matrix, G= 101 genes

Divisive Clustering

Group	1	2	3
ALL B-cell	15	3	1
ALL T-cell	0	8	0
AML	0	0	11

Partitioning Methods

- the other broad class of clustering algorithms are the partitioning methods
- the user selects some number of groups, g
- group or cluster centers are determined and objects are assigned to some set of initial clusters
- some mechanism for moving points and updating cluster centers is used

Partitioning Methods

- many different methods for doing this but the general approach is as follows:
- select the number of groups, G
- divide the samples into G different groups (randomly)
- iteratively select observations and determine whether the overall gof will be improved by moving them to another group

Partitioning

- this algorithm is then applied to the data until some stopping criterion is met
- the solution is generally a local optimal not necessarily a global optimal
- the order in which the samples are examined can have an effect on the outcome
- this order is generally randomly selected

Partitioning Methods

- among the most popular of these methods are
  - k-Means
  - PAM
  - self-organizing maps

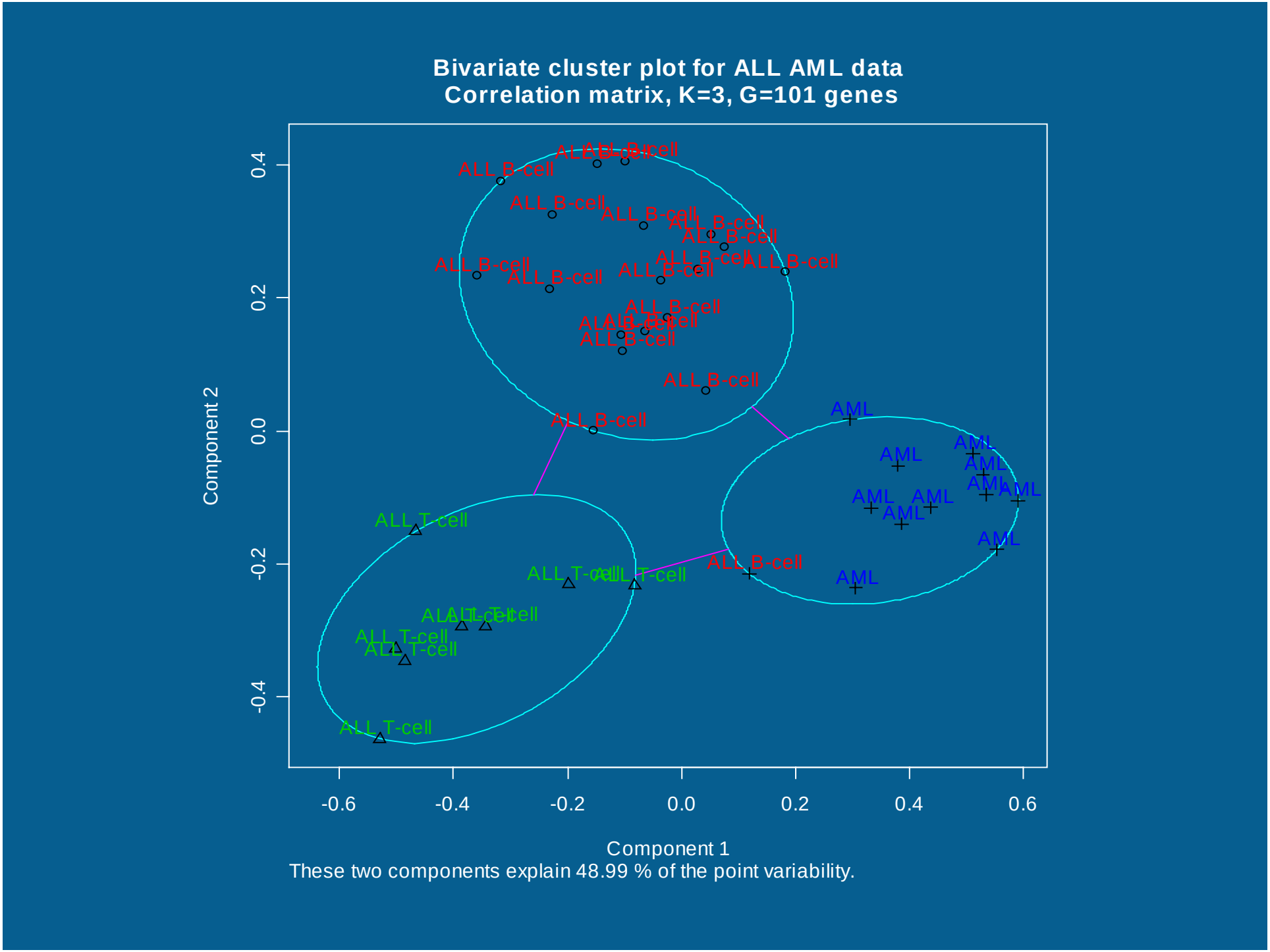
Partitioning Methods

- pam: partitioning around mediods
- cluster centers are actual examples
- we define a distance between samples and how many groups
- then we apply pam which sequentially moves the samples and updates the centers

PAM – ALL/AML

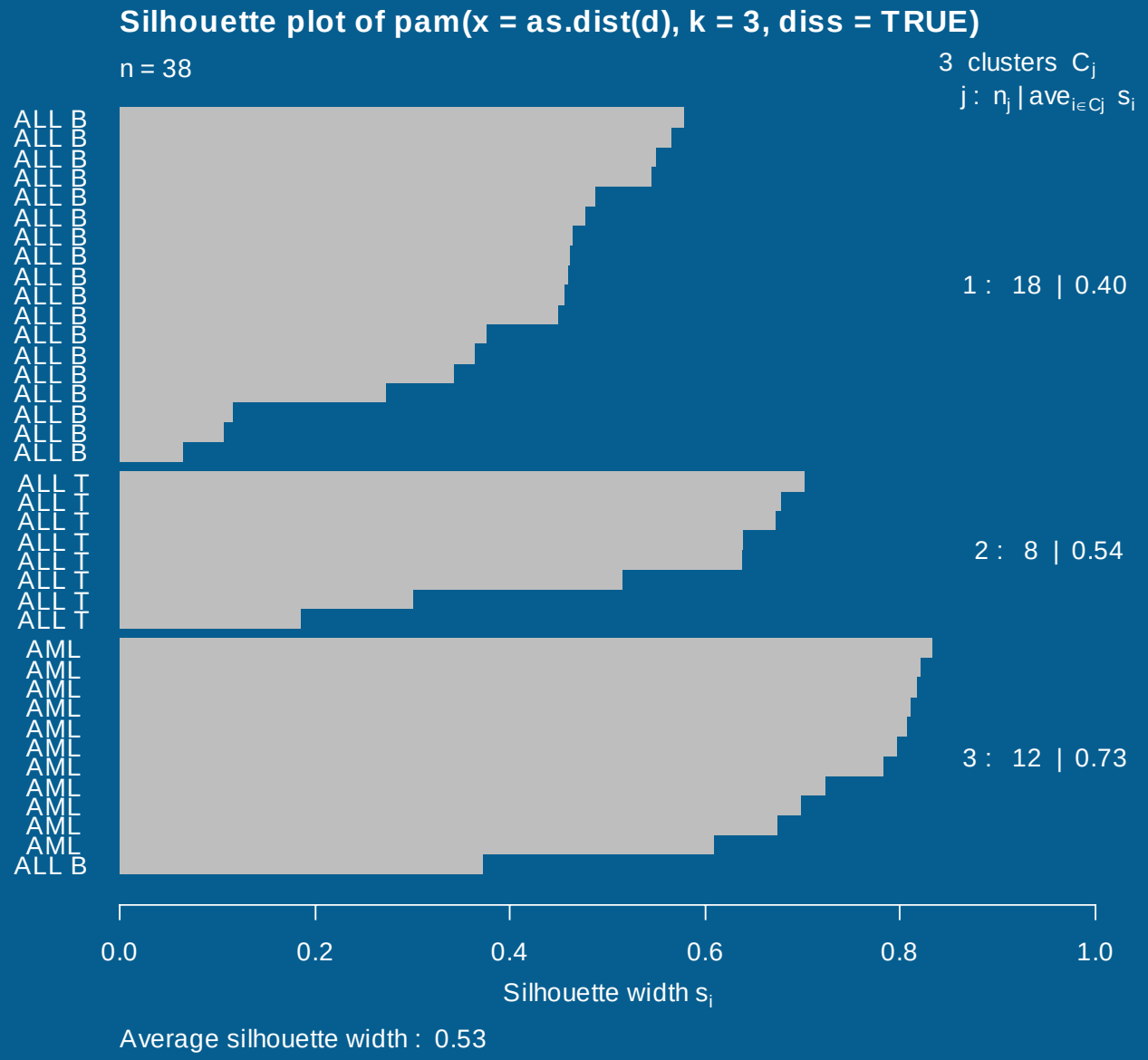
- pam was applied to the data from Golub et al.
- the results (for three groups) were:

Group	1	2	3
ALL B-cell	18	0	1
ALL T-cell	0	8	0
AML	0	0	11


Bivariate cluster plot for ALL AML data
Correlation matrix, K=3, G=101 genes
Component 2
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
Component 1
These two components explain 48.99 % of the point variability.
ALL B-cell
ALL T-cell
AML

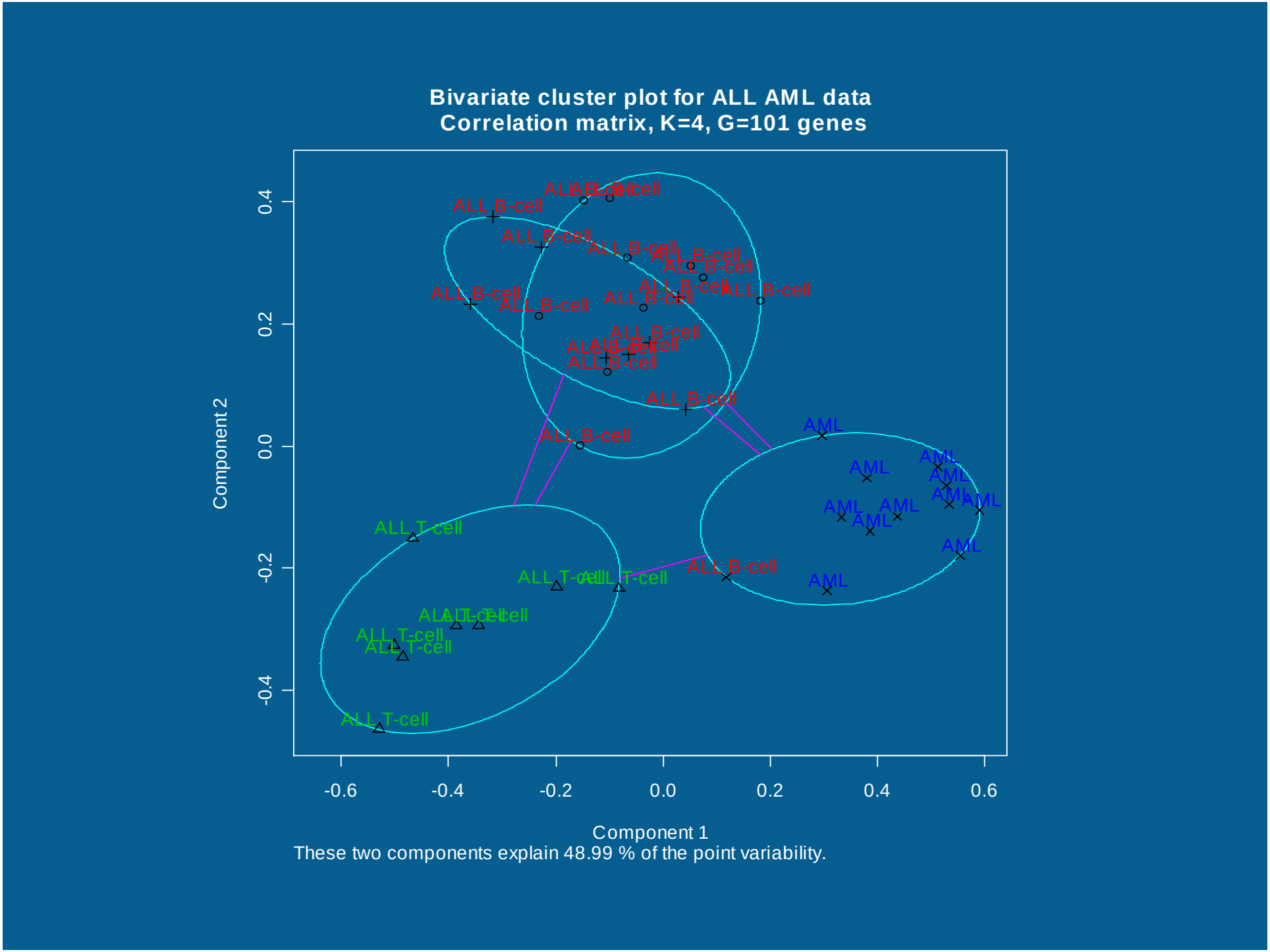
PAM

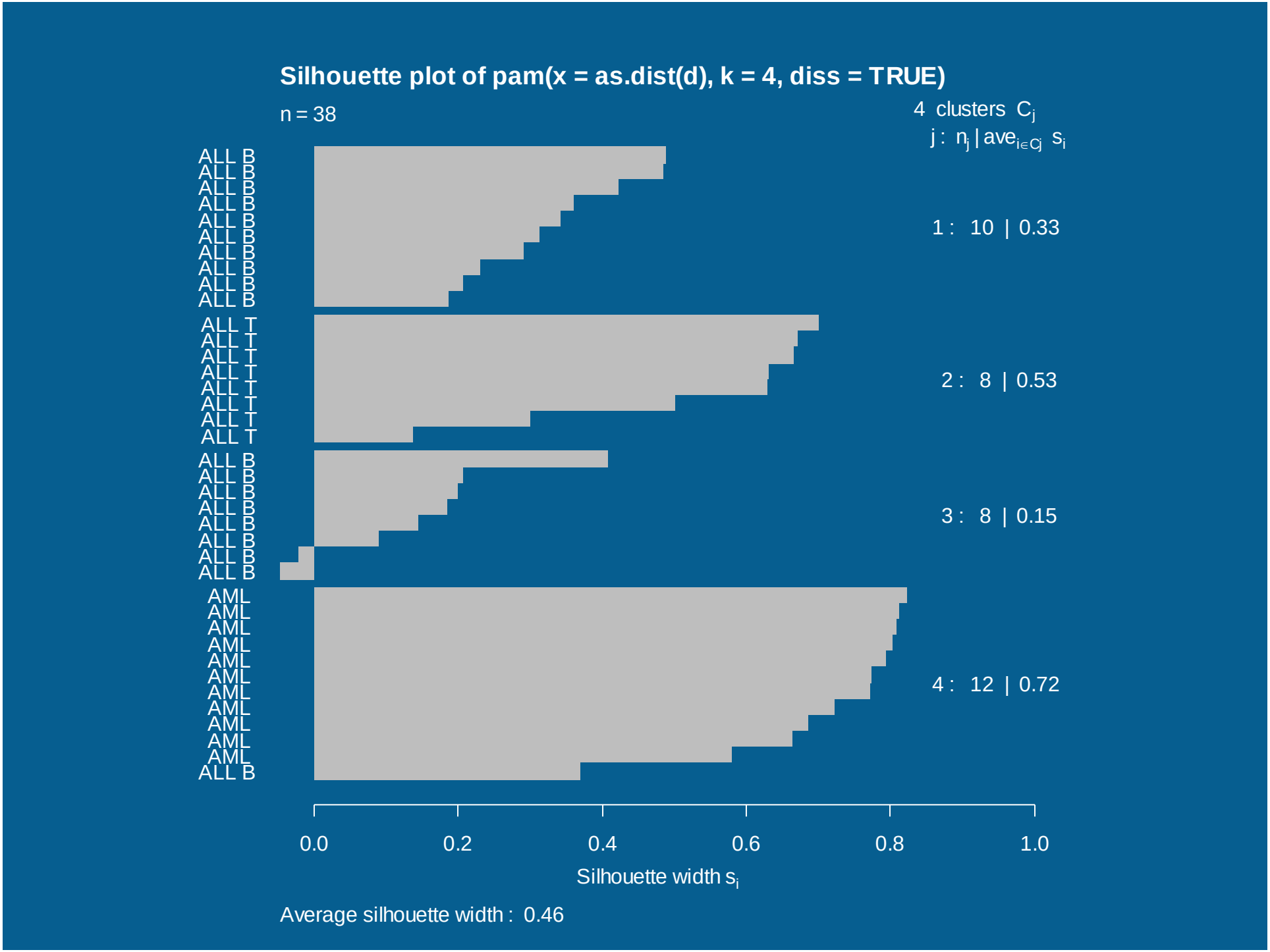
- the next plot is called a silhouette plot
- each observation is represented by a horizontal bar
- the groups are slightly separated
- the length of a bar is a measure of how close the observation is to its assigned group (versus the others)


Silhouette plot of pam(x = as.dist(d), k = 3, diss = TRUE)
n = 38
3 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL T
ALL T
ALL T
ALL T
ALL T
ALL T
ALL T
ALL T
AML
AML
AML
AML
AML
AML
AML
AML
AML
AML
AML
ALL B
1 : 18 | 0.40
2 : 8 | 0.54
3 : 12 | 0.73
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.53

How Many Groups do I have?

- this is a hard problem
- there are no known reliable answers
- you need to define more carefully what you mean by a group
- the next two slides ask whether there are four groups in the ALL/AML data


Bivariate cluster plot for ALL AML data
Correlation matrix, K=4, G=101 genes
Component 2
Component 1
These two components explain 48.99 % of the point variability.
ALL B-cell
ALL T-cell
AML
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6


Silhouette plot of pam(x = as.dist(d), k = 4, diss = TRUE)
n = 38
4 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
1 : 10 | 0.33
ALL T
ALL T
ALL T
ALL T
ALL T
ALL T
ALL T
ALL T
2 : 8 | 0.53
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
ALL B
3 : 8 | 0.15
AML
AML
AML
AML
AML
AML
AML
AML
AML
AML
AML
ALL B
4 : 12 | 0.72
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.46

How Many Groups

- for microarray experiments the question has often been stated more in terms of the samples by genes, false color displays
- there one is interested in finding relatively large blocks of genes with relatively large blocks of samples where the expression level is the same for all
- this is computationally very hard

Clustering Genomic Data

- in my examples (and in most applications I am aware of) I simply selected genes that looked like they differentiated the two major groups
- I could also do clustering on all 3,000-odd genes
- I could select genes according to pathway or GO category or … and do a separate clustering for each

Clustering Genomic Data

- it seems to me that there is a lot to be gained from thinking about the features and trying to use some known biology
- using subsets of the features rather than all of them to see whether there are interesting groups could be quite enlightening
- this requires collaboration between biologists and statisticians

Clustering

- one of the biggest problems here is a lack of a common interface
- many different software programs all are slightly different
- many tools are not yet implemented
- this is changing as both computational biology and data mining have spurred an interest in this field

Feature Selection

- this is perhaps the hardest part of the machine learning process
- it is also very little studied and there are few references that can be used for guidance
- the field of data-mining offers some suggestions

Feature Selection

- in most problems we have far too many features and must do some reduction
- for our experiment many of the genes may not be expressed in the cell type under examination
- or they may not be differentially expressed in the phenotype of interest

Feature Selection

- non-specific feature selection is the process of selecting features that show some variation across our samples without regard to phenotype
- for example we could select genes that show a certain amount of variability

Feature Selection

- specific feature selection is the process of selecting features that align with or predict a particular phenotype
- for example we may select features that show a large fold change when comparing two groups of interest (patients in remission versus those for whom cancer has returned)

Feature Selection

- most feature selection is done univariately
- most models are multivariate
- we know, from the simplest setting, that the best two variable model may not contain the best single variable
- improved methods of feature selection are badly needed

Feature Selection: CV

- there are two different ways to consider using CV for feature selection
- have an algorithm for selecting features
- obtain M different sets of features
- for each set of features (with the distance and model fixed) compute the CV error
- select the set of features with the smallest error

Feature Selection: CV

- a different method is to put the feature selection method into the algorithm
- for each CV subset perform feature selection
- predict those excluded
- could select those features that were selected most often

Feature Selection: CV

- a slight twist would be to weight the features according to the subsample prediction error
- give those features involved in models that had good predictive capabilities higher
- select the features with the highest combined weight

Feature Selection

- if we want to find those features which best predict the duration of remission we must also use supervised learning (classification) to predict duration of remission
- then we must use some method for determining which features provide the best prediction
- we will return to this interesting question a bit later

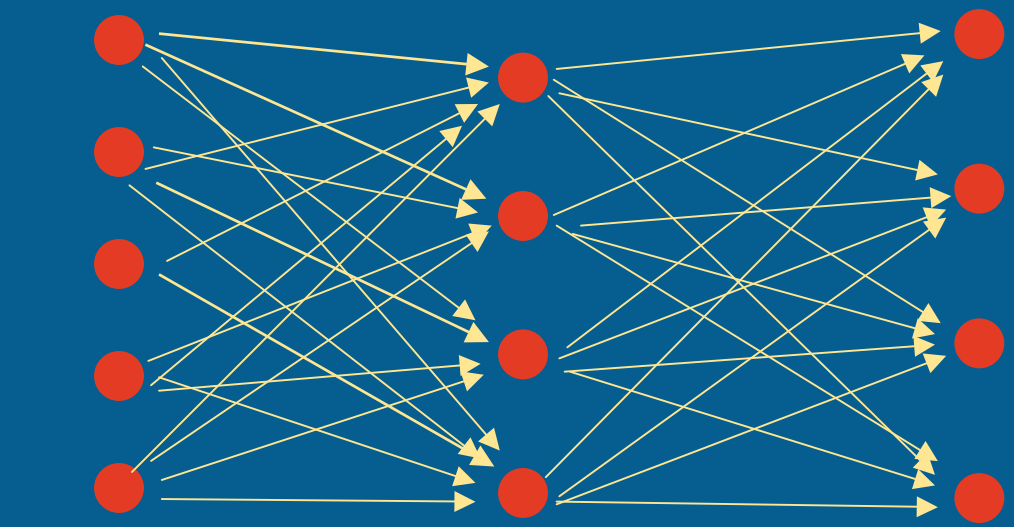
Some References

- *Classification, 2nd ed.*, A. D. Gordon, Chapman & Hall (it's about clustering), 1999
- *Pattern Recognition and Neural Networks*, B. D. Ripley, Cambridge Univ. Press, 1996
- *The Elements of Statistical Learning*, T. Hastie, R. Tibshirani, J. Friedman, Springer, 2001
- *Pattern Classification, 2nd ed.*, R. Duda, P. Hart and D. Stork, Wiley, 2000.
- *Finding Groups in Data*, L. Kaufman and P. J. Rousseeuw, Wiley, 1990.

Neural Networks

- a mechanism for making predictions
- they can be arbitrarily complex (some caution must be used when comparing to other methods)
- consist of a set of nodes arranged in layers

Neural Network

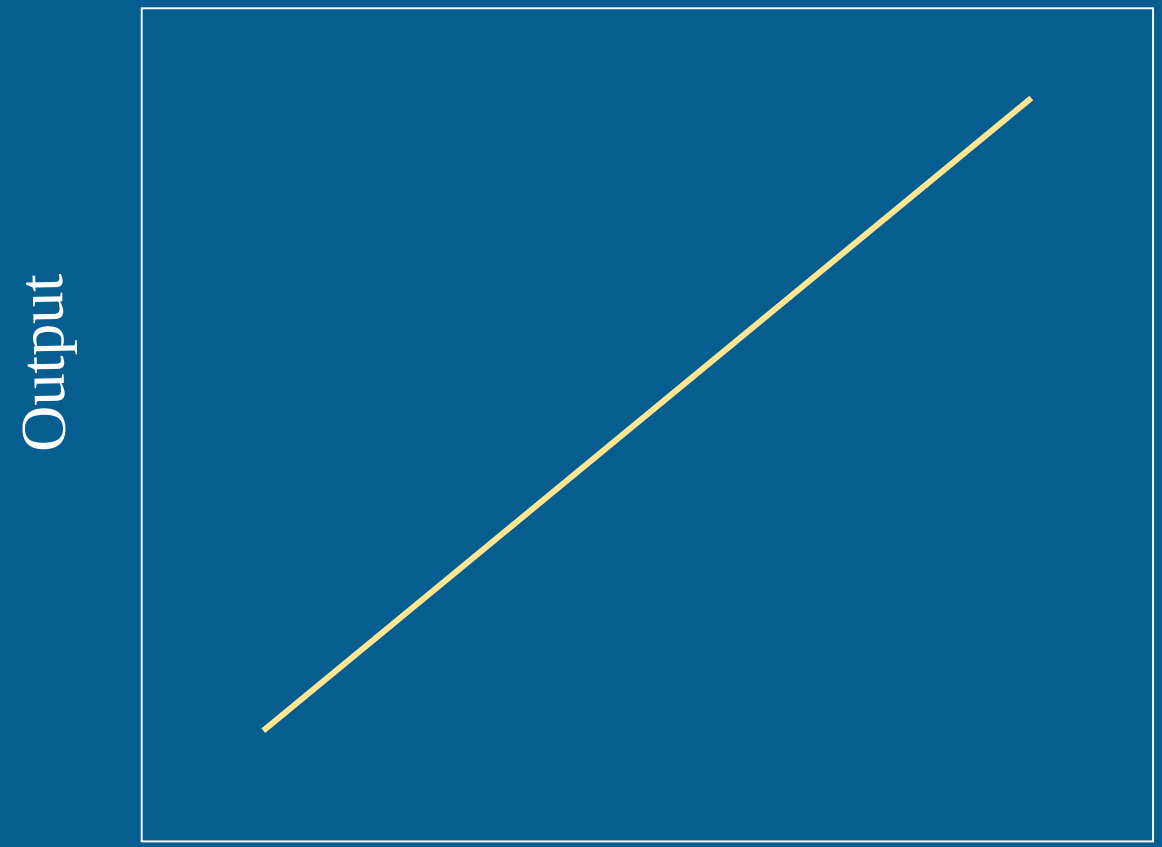


Neural Networks

- each node (unit) sums its inputs, adds a constant to form the total input
- a node specific function function $f_k()$ is then applied to the total input to yield the total output
- the output then becomes the input for the next layer
- the output from the final layer constitutes the prediction

Neural Networks

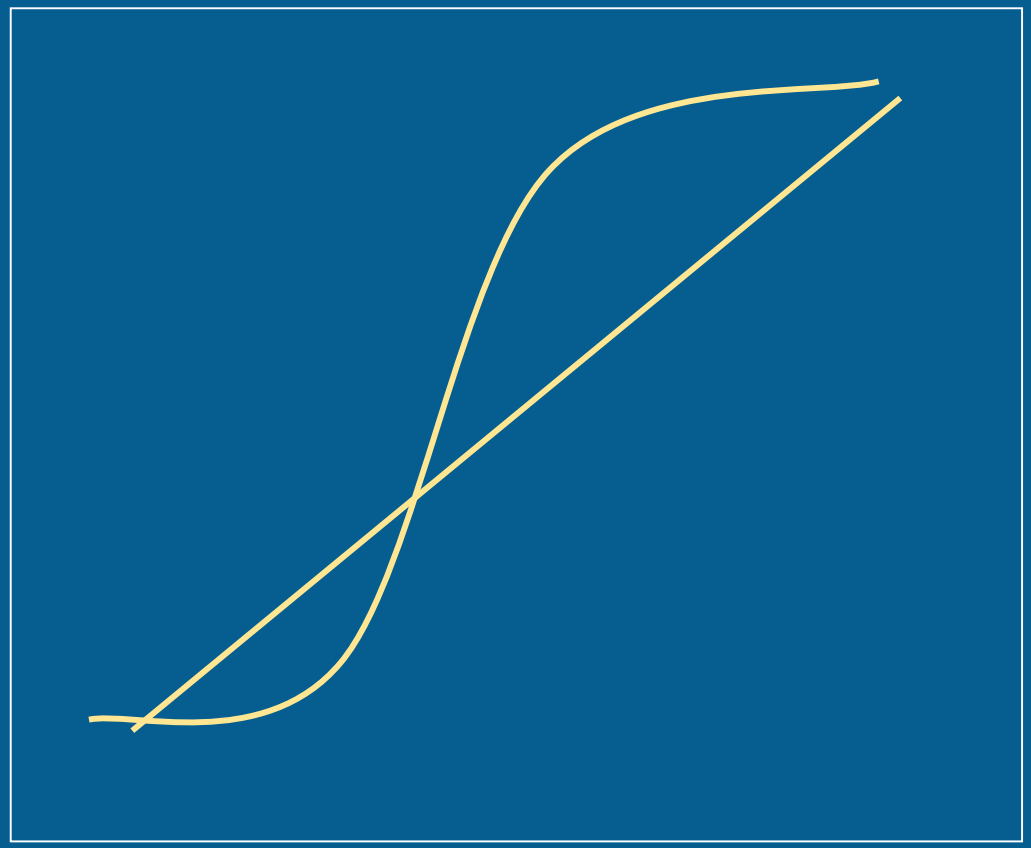
Linear



Neural Networks

Linear

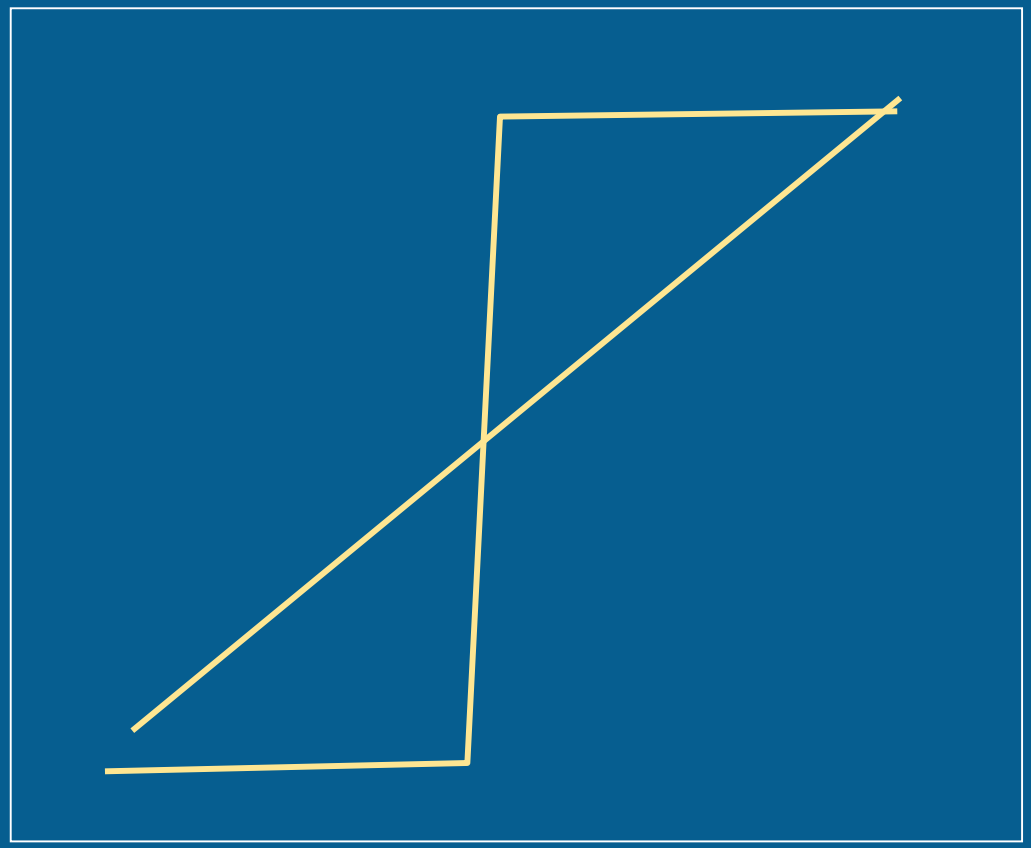
Sigmoid



Neural Networks

Linear

Threshold



Neural Networks

- for a unit k we assume the output is given by

$$y_k = f_k(\alpha_k + \sum_{j \to k} w_{jk} f_j(\alpha_j + \sum_{i \to j} w_{ij} x_i))$$

- to be useful we need to obtain values for the w_{ij}
- this is difficult and is usually based on the use of a training set

Neural Networks

- convergence is difficult to assess: even when you have an independent test set
- it seems that one seldom needs more than one hidden layer to accommodate the problems we are encountering with microarrays
- more hidden layers imply a more complex model

Thanks

- Sabina Chiaretti
- Vincent Carey
- Sandrine Dudoit
- Beiying Ding
- Xiaochun Li
- Denise Scholtens
- Jeff Gentry
- Jianhua Zhang
- Jerome Ritz
- Alex Miron
- J. D. Iglehart
- A. Richardson