# Clustering and Classification

Klaus Hechenbichler

Max-Planck-Institut für Psychiatrie

18.2.2004

# Clustering and Classification

- Cluster analysis (unsupervised learning)
  - the classes are unknown a priori
  - the goal is to discover these classes from the data

- Classification (class prediction, supervised learning)
  - the classes are predefined
  - the goal is to understand the basis for the classification from a set of labeled objects and build a predictor for future unlabeled observations

# Gene expression data

Statistical problems associated with tumor classification:

- the identification of new tumor classes using gene expression profiles – unsupervised learning
- the classification of malignancies into known classes – supervised learning
- the identification of marker genes that characterize the different tumor classes – feature selection

# Clustering

# Cluster analysis packages

- **mva**:
  - hierarchical clustering (**hclust**)
  - k-means (**kmeans**)
- **cluster**:
  - Partitioning Around Medoids (**PAM**)
- **som**:
  - Self-Organizing Maps (**SOM**)
- **mclust**:
  - model-based clustering (**mclust**)

- There are specialized summary, plot, and print methods for clustering results

# Cluster analysis

- Associated with each object is a set of $G$ measurements which form the feature vector $\mathbf{X} = (X_1, \ldots, X_m)$

- The task is to identify groups, or clusters, of similar objects on the basis of a set of feature vectors $\mathbf{X}_1, \ldots, \mathbf{X}_n$

- Clustering is a more difficult problem than classification:
  - there is no learning set of labeled observations
  - the number of groups is usually unknown
  - implicitly, one must have already selected both the relevant features and distance measure

- The goals can be quite vague: "*Find some interesting and important clusters in my data*"

# Cluster analysis

- Clustering involves several distinct steps:
  - a suitable distance between objects must be defined
  - a clustering algorithm must be selected and applied

- The results of a clustering procedure can include:
  - the number of clusters *k*
  - a set of cluster labels for the *n* objects

- Appropriate choices will depend on the questions being asked and available data

# Cluster analysis

Clustering procedures fall into two categories:

- Hierarchical methods, either divisive or agglomerative. These methods provide a hierarchy of clusters, from the smallest, where all objects are in one cluster, through to the largest set, where each observation is in its own cluster

- Partitioning methods. These usually require the specification of the number of clusters. Then, a mechanism for apportioning objects to clusters must be determined

Most methods used in practice are agglomerative hierarchical methods. In large part, this is due to the availability of efficient exact algorithms

# Distance measures

- The feature data are often transformed to an $n \times n$ distance or similarity matrix, $\mathbf{D} = (d_{ij})$, for the objects to be clustered

- Once a distance measure between individual observations has been chosen, one must often also define a distance measure between clusters (linkage methods), based on object dissimilarity between objects from the two clusters

- Different choices here can greatly affect the outcome

# Distance measures

- Euclidean: The distance between two vectors is the square root of the sum of the squared differences over all coordinates

- Manhattan: The distance between two vectors is the sum of the absolute (unsquared) differences over all coordinates

- Correlation: The distance between two vectors is 1-$\rho$, where $\rho$ is the Pearson correlation of the two vectors

# Distance measures

- Average linkage: Average pairwise distance between two objects of the different clusters

- Single linkage: Smallest pairwise distance between two objects of the different clusters

- Complete linkage: Largest pairwise distance between two objects of the different clusters

# Clustering gene expression data

- One can cluster genes and/or samples
- Clustering leads to readily interpretable figures
- Clustering can be helpful for identifying gene expression patterns in time or space
- Clustering is useful, perhaps essential, when seeking new subclasses of cell samples (tumors etc.)

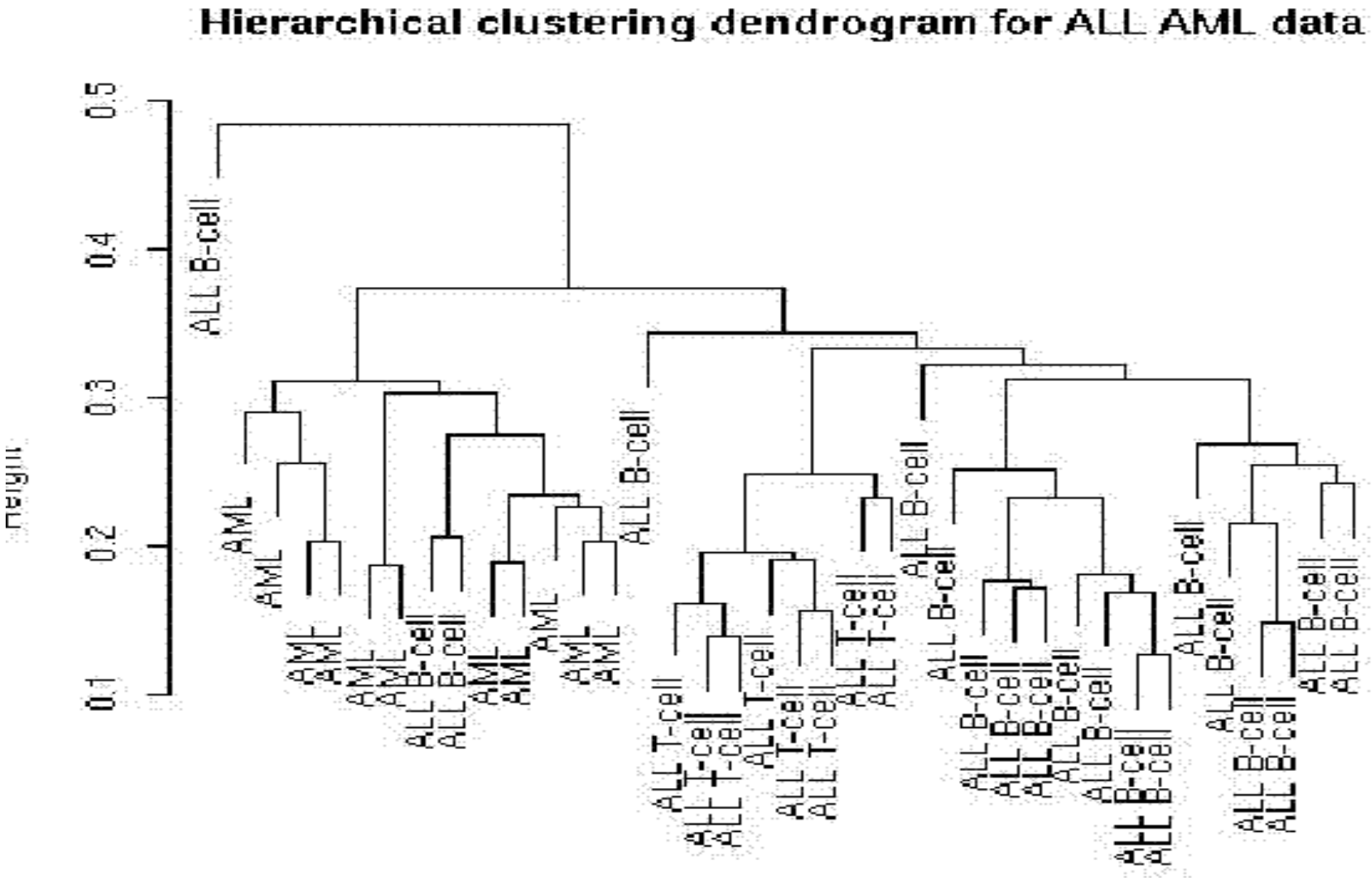
# Hierarchical methods

- Hierarchical clustering methods produce a tree or dendrogram
- They avoid specifying how many clusters are appropriate by providing a partition for each *k*
- The partitions are obtained from cutting the tree at different levels
- The tree can be built in two distinct ways:
  - bottom-up: agglomerative clustering
  - top-down: divisive clustering

# Hierarchical methods

- While dendrograms are quite appealing because of their apparent ease of interpretation, they can be misleading
- First, the dendrogram corresponding to a given hierarchical clustering is not unique, since for each merge one needs to specify which subtree should go on the left and which on the right – there are $2^{n-1}$ choices
- The default in the R function hclust is to order the subtrees so that the tighter cluster is on the left
- Second, they impose structure on the data, instead of revealing structure in these data
- Such a representation will be valid only to the extent that the pairwise dissimilarities possess the hierarchical structure imposed by the clustering algorithm

# **hclust** function from **mva** package



# Agglomerative methods

- Start with *n* mRNA sample clusters or *G* gene clusters
- At each step, merge the two closest clusters using a measure of between-cluster distance which reflects the shape of the clusters
- Computational simple to implement

# Divisive methods

- Start with only one cluster
- At each step, split clusters into two parts
- Advantages: Obtain the main structure of the data, i.e., focus on upper levels of dendrogram
- Disadvantages: Computational diffculties when considering all possible divisions into two groups

# Partitioning methods

- Partition the data into a prespecified number *k* of mutually exclusive and exhaustive groups

- Iteratively reallocate the observations to clusters until some criterion is met (e.g. minimize within-cluster sums-of-squares)

- Examples:
  - k-means clustering
  - Partitioning Around Medoids – PAM
  - Self-Organizing Maps – SOM
  - model-based clustering

# k-means clustering

- a partitioning algorithm with a prefixed number *k* of clusters, that tries to minimize the sum of within-cluster-variances

- a random sample of *k* different objects are chosen as initial cluster midpoints

- Alternation between two steps until convergence:
    1. assign each object to its closest of the *k* midpoints with respect to Euclidean distance
    2. Calculate *k* new midpoints as the averages of all points assigned to the old midpoints

# k-means clustering

- k-means is a randomized algorithm, two runs usually produce different results

- it has to be applied a few times to the same data set and the result with minimal sum of within-cluster variances should be chosen

# Partitioning around medoids

- PAM is a partitioning method which operates on a distance matrix

- For a prespecified number of clusters *k*, the PAM procedure is based on the search for *k* representative objects, or medoids $\mathbf{M} = (\mathbf{m}_1, \ldots, \mathbf{m}_k)$, among the observations

- The medoids minimize the sum of the distances of the observations to their closest medoid:

$$\mathbf{M}^* = \operatorname{argmin}_{\mathbf{M}} \sum_i \min_k d(\mathbf{x}_i, \mathbf{m}_k).$$

# Partitioning around medoids

- After finding a set of *k* medoids, *k* clusters are constructed by assigning each observation to the nearest medoid

- PAM can be applied to general data types and tends to be more robust than k-means

# Partitioning around medoids

- silhouette plots are graphical displays, which can be used to (i) select the number of clusters
  (ii) assess how well individual observations are clustered

- silhouette width of observation *i* is defined as

$$sil_i = (b_i - a_i) / \max(a_i, b_i)$$

where $a_i$ denotes the average distance between *i* and all other observations in the cluster to which *i* belongs, and $b_i$ denotes the minimum average distance of *i* to objects in other clusters

# Partitioning around medoids

- objects with large silhouette width are well-clustered, others tend to lie between clusters

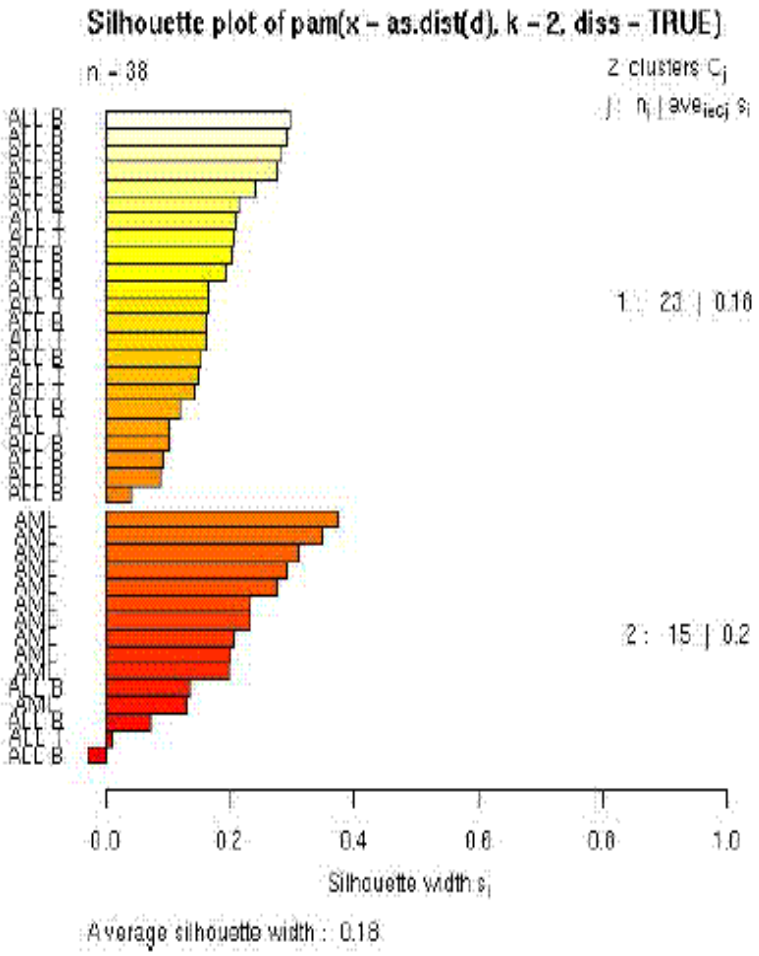
- For a given number of clusters *k*, the overall average silhouette width for the clustering is simply the average over all observations *i*:

$$\bar{sil} = \sum_i sil_i / n.$$

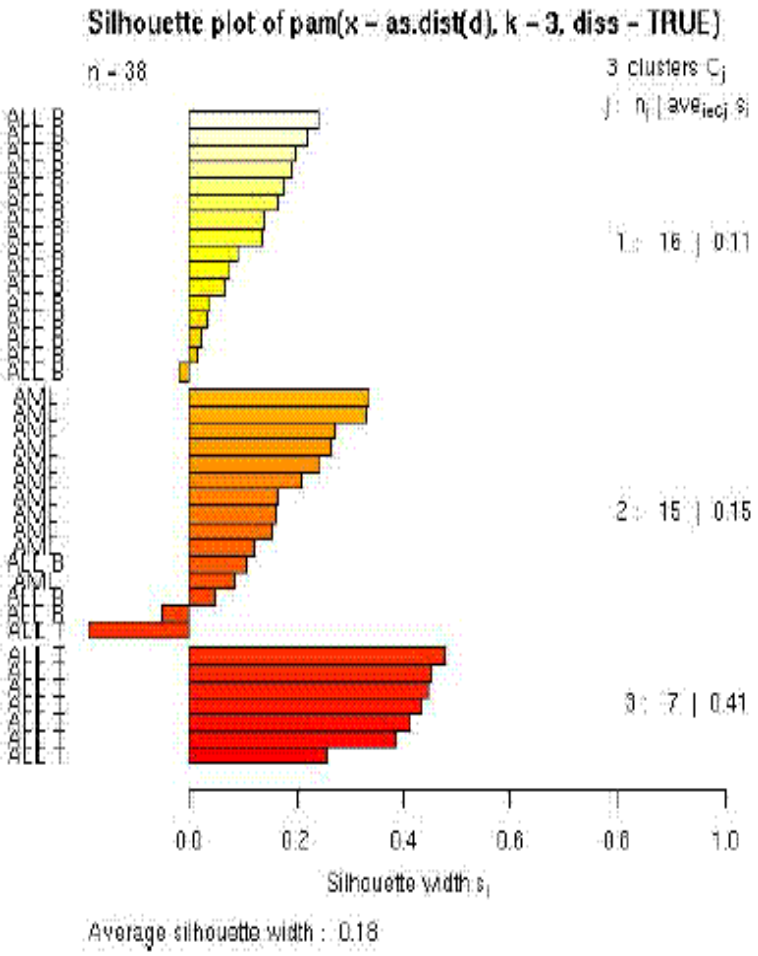
- The number of clusters *k* can be estimated by that which gives the largest average silhouette width

# **pam** and **plot** from **cluster** package

k=2

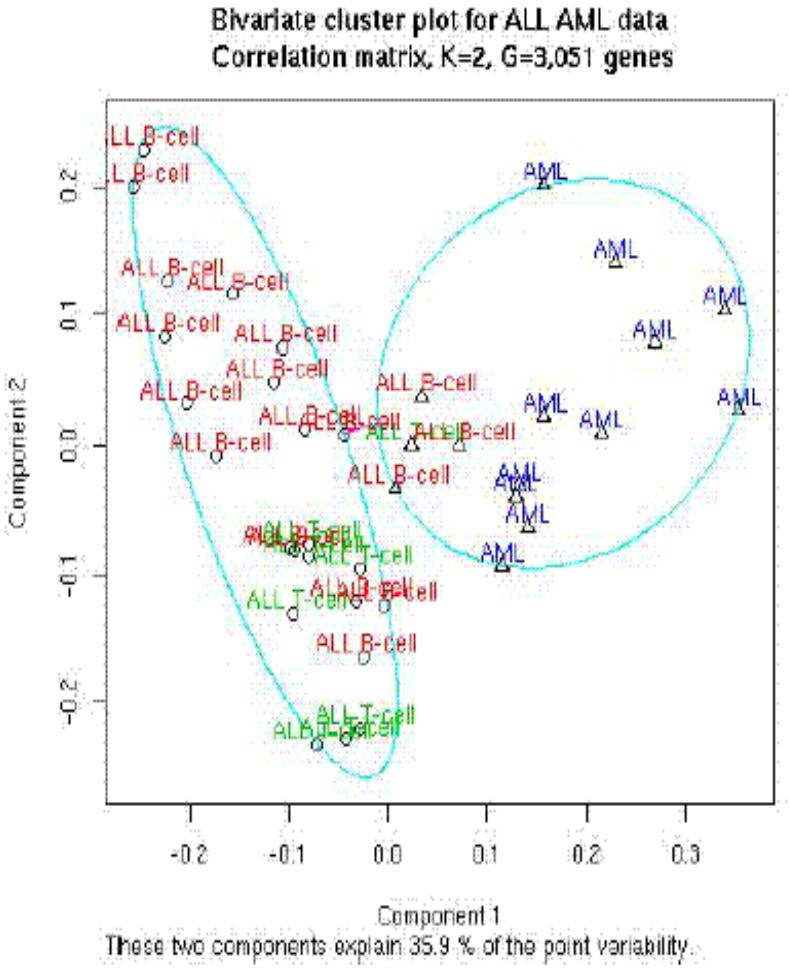



k=3

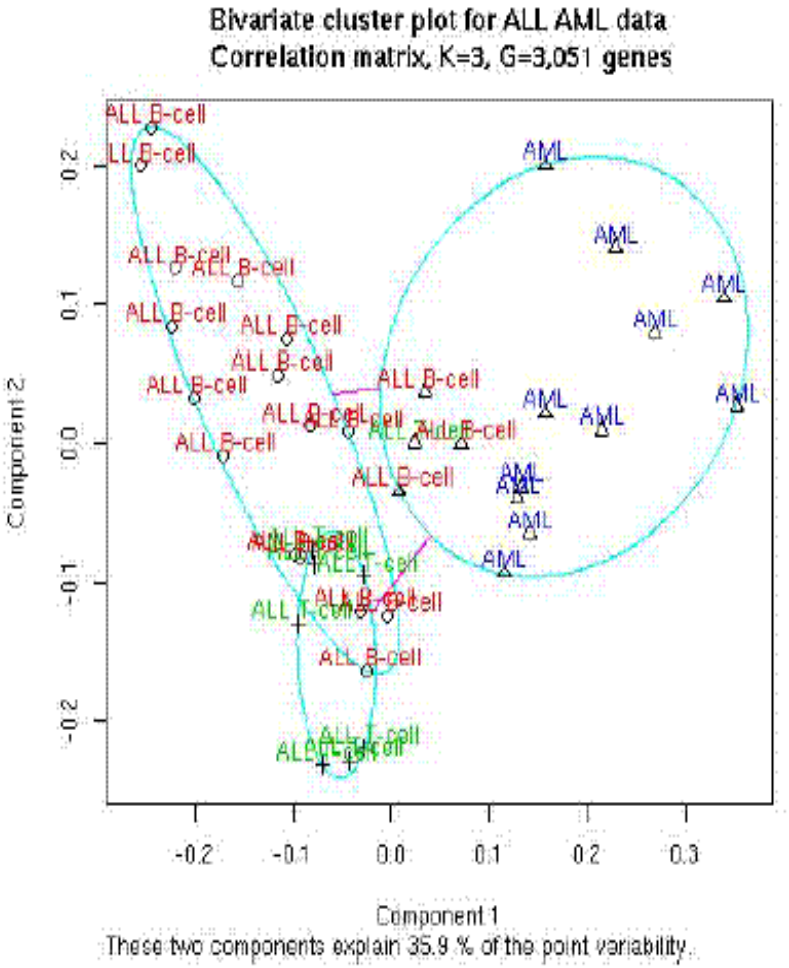


# **pam** and **clusplot** from **cluster** package

k=2




k=3



# Self-organizing maps

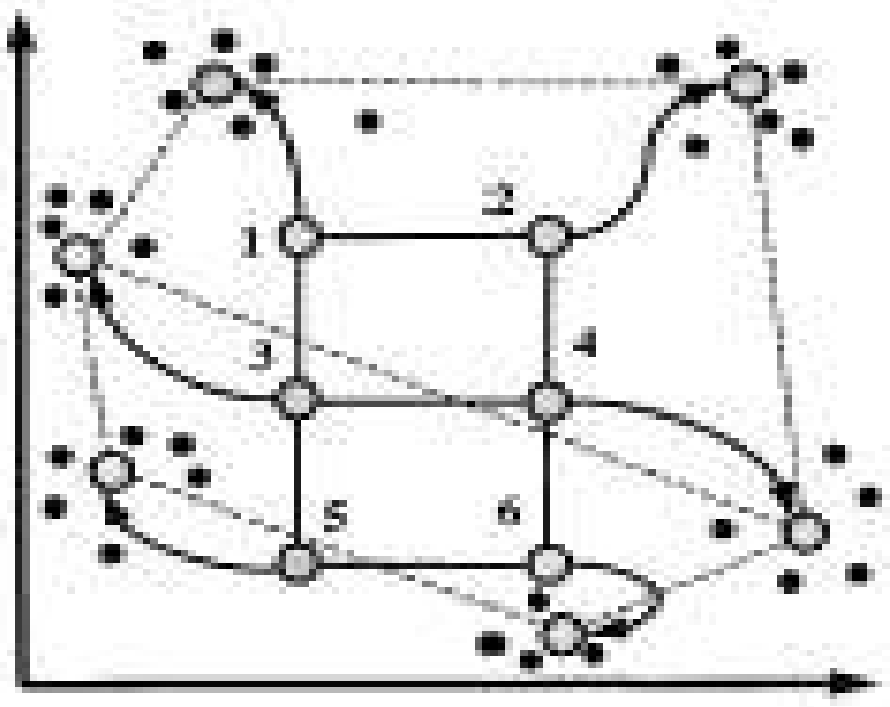
- SOMs are similar to k-means, but with additional constraints: The nodes in a SOM are ordered (results in a smooth transition between groups)

- Mapping from input space onto one- or two-dimensional array of *k* total nodes

- Iteration steps (20000-50000):
  - Pick data point P at random
  - Move all nodes in direction of P, the closest node most, the further a node is, the less
  - Decrease amount of movement with iteration steps

# Self-organizing maps

- Original motivation: First compute a one dimensional SOM, then use the ordering from the SOM to guide the flipping of nodes in a hierarchical tree (output ordering will come as close as possible to the ordering in the SOM without violating the structure of the tree)



# Discussion

- Hierarchical
  - Advantages: Fast computation, at least for agglomerative clustering
  - Disadvantages: Rigid, cannot correct later for erroneous decisions made earlier

- Partitioning
  - Advantages: Provide clusters that (approximately) satisfy an optimality criterion
  - Disadvantages: Need initial *k*, long computation time

# Classification

# Classification packages

- **MASS**:
  – linear and quadratic discriminant analysis (**lda**, **qda**)
- **class**:
  – k-nearest neighbor (**knn**)
- **rpart**:
  – classification and regression trees (**CART**)
- **pamr**:
  – prediction analysis for microarrays
- **nnet**:
  –neural networks and multinomial log-linear models
- **e1071**:
  – support vector machines (**svm**)
- **ranForest**:
  – random forests

# Classification

- Predict a biological outcome on the basis of observable features

- Outcome: tumor class, type of bacterial infection, survival, response to treatment, ...

- Features: gene expression measures, covariates such as age, sex, ...

# Classification

- Classification is a prediction or learning problem in which the variable to be predicted assumes one of *K* unordered values, corresponding to *K* predefined classes

- Associated with each object are a response variable Y (class label) and a set of *G* measurements which form the feature vector X

- The task is to classify an object into one of the *K* classes on the basis of an observed measurement, i.e., predict Y from X

# Classification

- Old and extensive literature on classification, in statistics and machine learning

- Examples of classifiers:
  - discriminant analysis
  - nearest neighbor classifiers
  - classification trees
  - neural networks
  - support vector machines
  - aggregated classifiers (bagging, boosting, forests)

- Comparison on microarray data: simple classifiers perform remarkably well

# Performance assessment

- Classification error rates, or related measures
  - to compare the performance of different classifiers
  - to support statements such as
    "*cancer Y can be predicted accurately based on gene expression measures X*"

- It is essential to take into account feature selection and other training decisions in the error rate estimation process (e.g. number of neighbors in k-NN, kernel in SVMs)

# Performance assessment

- Resubstitution estimation: Error rate on the learning set (Problem: can be severely biased downward)

- Test set estimation: Cases in the learning set are repeatedly randomly divided into two sets, the classifier is built using one and the error rate is computed for the other (Problem: reduces effective sample size)

- V-fold cross validation (CV) estimation: Cases in the learning set L are randomly divided into V subsets of as nearly equal size as possible. Classifiers are built on learning sets, error rates are computed on test sets and averaged

# Fisher linear discriminant analysis

- FLDA first was applied in 1935 and consists of
  - finding linear combinations (discriminant variables) of the gene expression profiles with large ratios of between groups to within groups sums of squares
  - predicting the class of an observation **x** by the class whose mean vector is closest to **x** in terms of the discriminant variables

- When the class densities have the same covariance matrix, the discriminant rule is based on the square of the Mahalanobis distance and given by

$$C(\mathbf{x}) = \text{argmin}_k \, (\mathbf{x} - \mu_k)\Sigma^{-1}(\mathbf{x} - \mu_k)'$$

# Fisher linear discriminant analysis

- Advantages:
  - Simple and intuitive: the predicted class of a test case is the class with the closest mean
  - Easy to implement, good performance in practice

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

## Example: Linear discriminant analysis

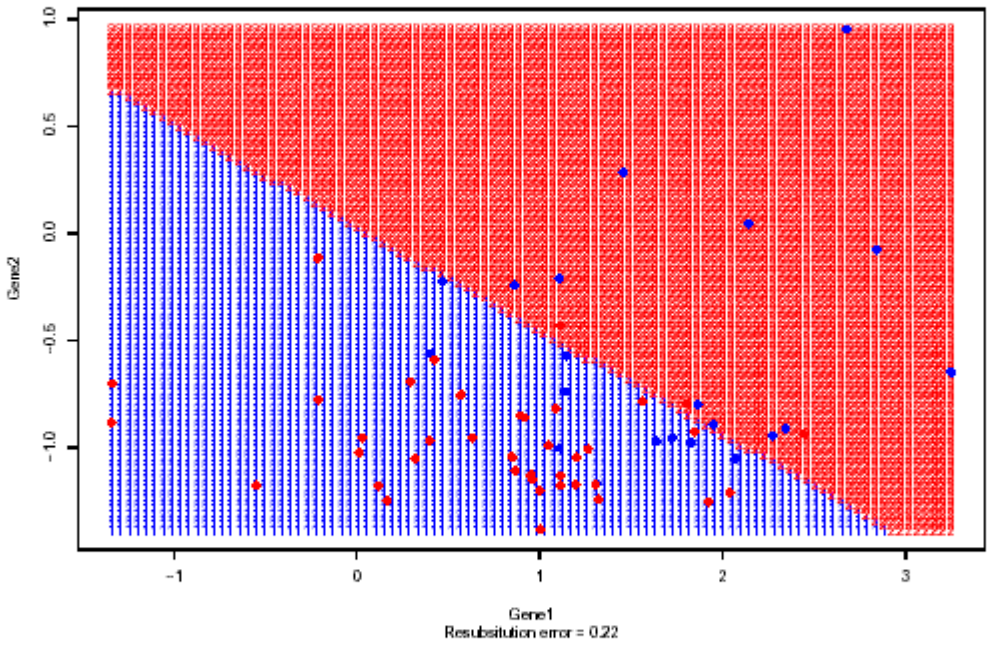



Figure 1: *Brain tumor MD survival dataset.* LDA partition for the two genes with the largest absolute $t$–statistics.

# Nearest neighbor classifiers

- Nearest neighbor methods are based on a measure of distance between observations, such as the Euclidean distance or one minus the correlation between two gene expression profiles

- The k-nearest neighbor rule (kNN) classifies an observation **x** as follows:
  - find the *k* observations in the learning set that are closest to **x**
  - predict the class of **x** by majority vote, i.e., choose the class that is most common among these *k* neighbors

# Nearest neighbor classifiers

- simple classifiers with $k = 1$ are generally quite successful
- Problem: large number of irrelevant or noise variables with little or no relevance can substantially degrade the performance of the classifier

# Example: Nearest neighbor classifier

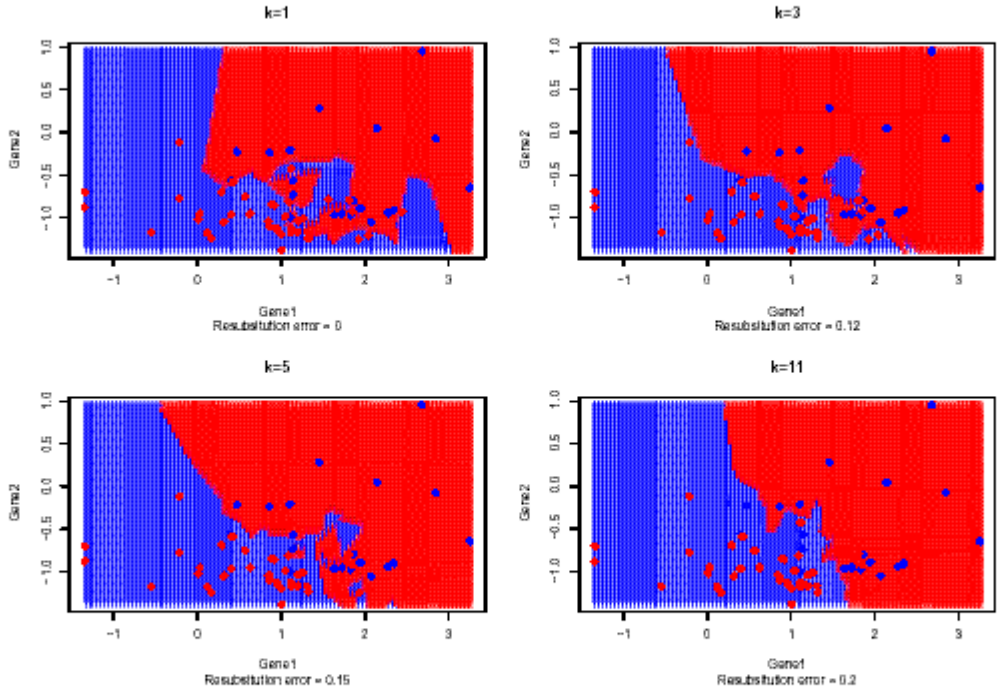



Figure 3: *Brain tumor MD survival dataset.* $k$–NN partitions ($k = 1, 3, 5, 11$) for the two genes with the largest absolute $t$–statistics

# Classification trees

- Binary tree structured classifiers are constructed by repeated splits of subsets (nodes) into two descendant subsets

- Each terminal subset is assigned a class label and the resulting partition corresponds to the classifier

- Three main aspects of tree construction:
  - selection of splits
  - decision to declare a node terminal or to continue splitting
  - assignment of each terminal node to a class

# Classification trees

- Splitting rule: At each node, choose the split that maximizes the decrease in impurity (e.g. Gini index, entropy)
- Split stopping rule: Grow a large tree, selectively prune the tree upward, getting a decreasing sequence of subtrees
- Class assignment rule: For each terminal node, choose the class that minimizes the resubstitution estimate of the misclassification probability

## Example: Classification tree

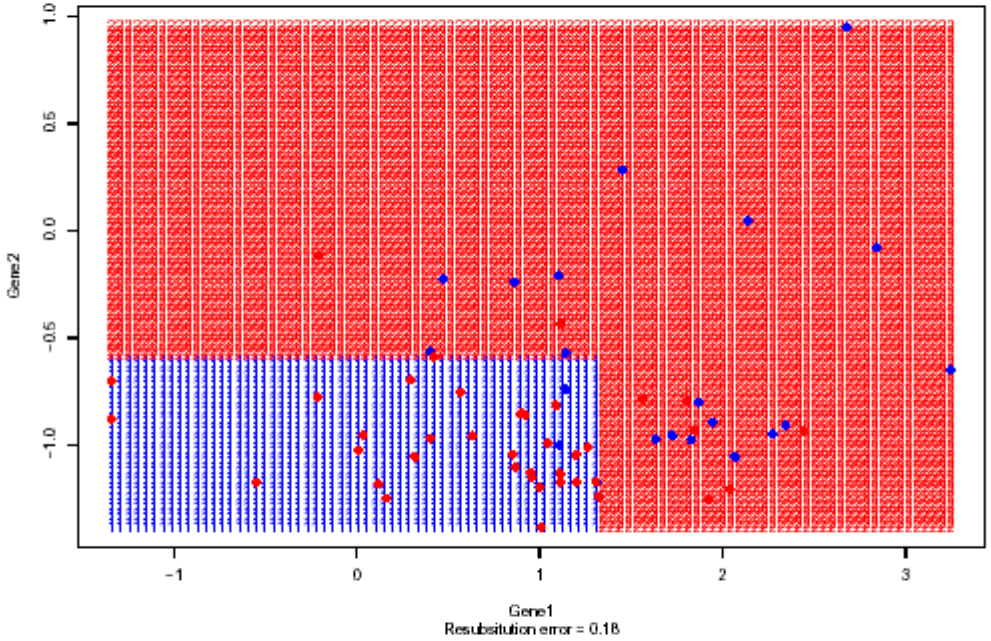



Figure 4: *Brain tumor MD survival dataset.* CART (10–fold CV) partition for the two genes with the largest absolute $t$–statistics.

# Distance and standardization

- Classification trees: Invariant under monotone transformations of individual features (genes); Not invariant to standardization of the observations (normalization in the microarray context)

- Linear discriminant analysis: Invariant to standardization of the variables (genes); Not invariant concerning the observations (arrays)

- Nearest neighbor classifiers: In general affected by standardization of both features and observations

# Prediction analysis for microarrays

- Nearest shrunken centroid methodology
- A standardized centroid for each class is computed (average gene expression for each gene in each class, divided by the within-class standard deviation for that gene)
- Each class centroid is shrunk toward the overall centroid for all classes by the so-called threshold (chosen by the user)
- Shrinkage consists of moving the centroid towards zero by threshold, setting it equal to zero if it hits zero (for example threshold = 2.0, a centroid of 3.2 is shrunk to 1.2, a centroid of -3.4 is shrunk to -1.4, a centroid of 1.2 is shrunk to zero)

# Prediction analysis for microarrays

- The gene expression profile of a new sample is compared to each of these shrunken class centroids and the class whose centroid is closest to (in squared distance) is the predicted class for that new sample

- Advantages:
  - Shrinkage makes the classifier more accurate by reducing the effect of noisy genes
  - It does automatic gene selection:
    - If a gene is shrunk to zero for all classes, then it is eliminated from the prediction rule
    - If a gene is shrunk to zero for all classes except one, then high or low expression for that gene characterizes that class

# Discussion

- There are two main goals:
  - Prediction: Predict biological outcomes for future samples using a collection of predictor variables
  - Information: Extract information about the underlying data generating mechanism, the relationship between responses and predictor variables

- Accuracy vs. simplicity: A simple predictor can provide reliable information about the relationship between responses and predictor variables (interpretability), a more complex one can provide accurate prediction

# Lunchtime ...