MVE441/MSA220 - Statistical Learning for Big Data

Lecture 7

Rebecka Jörnsten

Mathematical Sciences
University of Gothenburg and Chalmers University of Technology

Cluster prediction strength

Another method for choosing the number of clusters in a data set draws upon classification methods.
The idea is as follows: A clustering with the "correct" number of clusters is something that should be based on non-random structures in the data. Therefore the finding of groups should be reproducible - similar groups should be found if you were able to obtain a new, independent draw of data from the same data generative distribution.

Cluster prediction strength

For a given number of clusters, K

1. Divide the set of observations into two parts: A and B
2. On each of the data sets cluster the observations into K groups. Call these partitions C_A and C_B respectively
3. The partitions results in a labeling of the data sets. Treat these labels as "true" labels and learn a classification rule for each of the data sets: rule c_A is learnt from data set A with class labels C_A, rule c_B is learnt from data set B with class labels C_B.
4. Use data set B as a test set for the classifier c_A and data set A as the test set for classifier c_B. That is, the rule c_A applied to data set B results in a new labels $c_A(B)$ to be compared to the cluster label C_B and v.v. for data set B. Note, you may need to do a label-matching/permuting of labels first. Since the order of labels is arbitrary, group 1 in data set A might correspond to group 4 in data set B etc.
5. Compute the overall test error rate as the average of the test error rate in data set A and data set B

Cluster prediction strength

The optimal number of clusters is the K that makes the classifier from each data set predict the cluster label on the other data set as best as possible (reproducible groups).
If you try to find more clusters than is supported by data, the clusters are not reproducible since the "extra" clusters will correspond to some kind of random division of a group which will not line up for the different data sets.

Cluster prediction strength

Some things to consider:

- You need a lot of observations for this to work - enough such that cluster structure can be seen with only 50% of the data.
- Think about which clustering method and which classifier goes together. Kmeans and nearest-centroids are a good match. kmeans and LDA is also an OK match. If you use cluster methods based on non-euclidean distance remember you have to have a classifier that works like that also (kNN for example).

Modelbased clustering - continued

More on model selection....

Cautionary remarks:

- EM is sensitive to the choice of starting values and can converge to a local optimum or exhibit very slow convergence if starting values are poorly chosen. Track the likelihood as a function of the iterations to catch this and try a couple of different starting values.
- EM applied to MVN mixture modeling has a tendency to create empty clusters or singleton clusters which creates singular Σ_k. Once a cluster starts growing, Σ_k often increases in the next M-step which makes it even easier for an observation to be allocated to cluster k in the next E-step, etc.

We deal with the singularities by regularizing the estimates of the Σ_k. That is, we estimate

$$\tilde{\Sigma}_k = \hat{\Sigma}_k + \lambda\Lambda$$

where Λ is a covariance matrix you regularize Σ_k toward, usually taken as a scaled version of the global covariance (covariance of data without clustering) and λ controls how much you regularize. This form of regularization has a Bayesian motivation, and is the Bayesian covariance estimate if you make a prior assumption on $\Sigma_k = \Lambda$ with an Inverse-Wishart prior distribution.
Don't regularize too much - keep λ as small as possible so that the data dominates the likelihood and not the prior.

The EM-algorithm outputs parameter estimates and posterior probabilities η_{ik}. A final cluster allocation is achieved as

$$C(i) \arg\max_k \eta_{ik}$$

but you can also use the η directly as your "soft-clustering" output. How many clusters should you use?

As in all model-based methods, the likelihood cannot be used to select the model (number of clusters) as the likelihood is always increased by adding more and more model parameters to the description of the data - i.e. using the likelihood to select the number of clusters would just lead you to choose the largest number of clusters you try.
However, as we are in a standard modeling setting we can use the off-the-shelf model selection criteria that you may be familiar with from regression. The most commonly used criterion in mixture modeling is the Bayesian information criterion, BIC.

$$BIC(K) = -2 * \log - likelihood + \log(n) * p$$

where

$$p = (K - 1) + K * D + K * D(D + 1)/2$$

is the number of model parameters, where the three terms refer to the number of π's, μ's and Σ's respectively.
The BIC measures the trade-off between the model fit to the data (the loglikelihood term) and the complexity of the model (the number of parameters).

You pick the number of clusters K that minimizes the BIC.
In practice, BIC is usually rather flat-looking so there are often a couple of different values for K that gives you an almost equally good fit.
The number of parameters in your model can be altered if you are willing to make simplifying assumption on the cluster shapes, sizes or correlation structures (i.e. on the form of Σ_k). If you simplify the cluster shapes you save on the number of parameters and may "afford" more clusters.

This idea of simplifying the cluster shapes, sizes and correlations (orientations) was the basis for the Mclust procedure of Raftery et al (2006). This modelbased clustering method has been implemented in a very easy-to-use R-package (`mclust()`).

Setup:

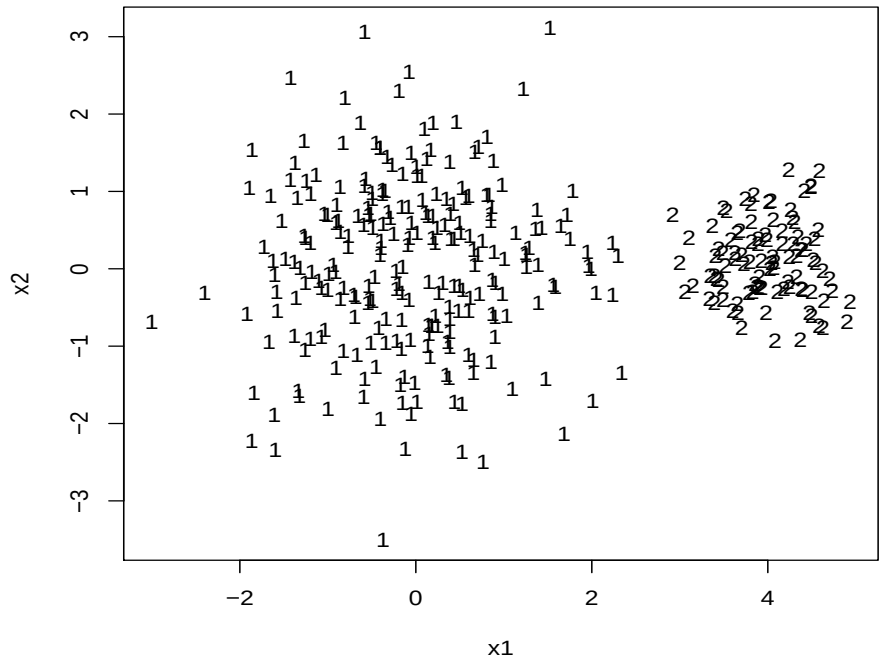
$$x_i \sim \sum_{k=1}^{K} \pi_k \Phi(x_i \mid \mu_k, \Sigma_k)$$

where we write

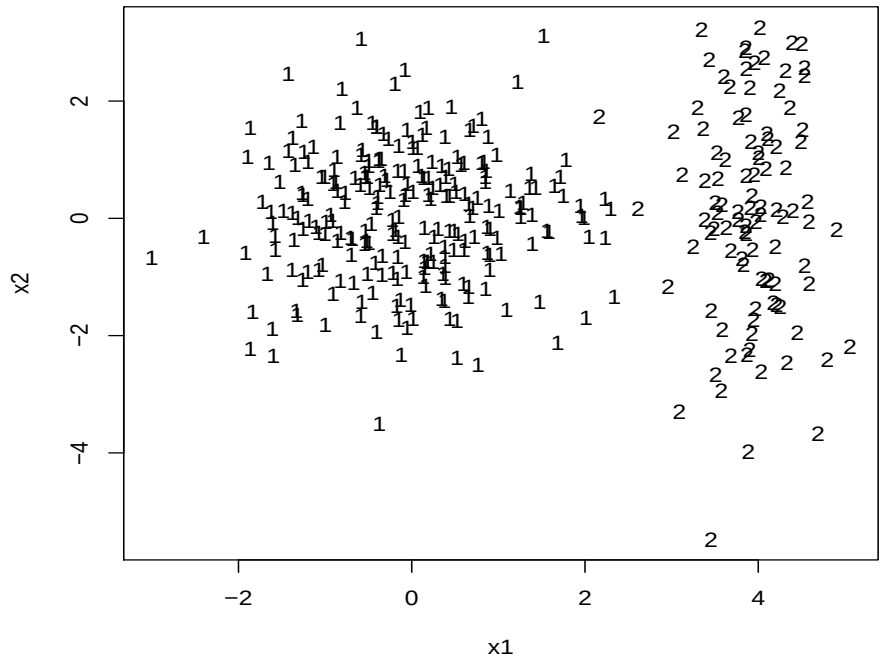
$$\Sigma_k = \lambda_k D_k A_k D_k'$$

This is the eigenvalue decomposition of Σ_k.

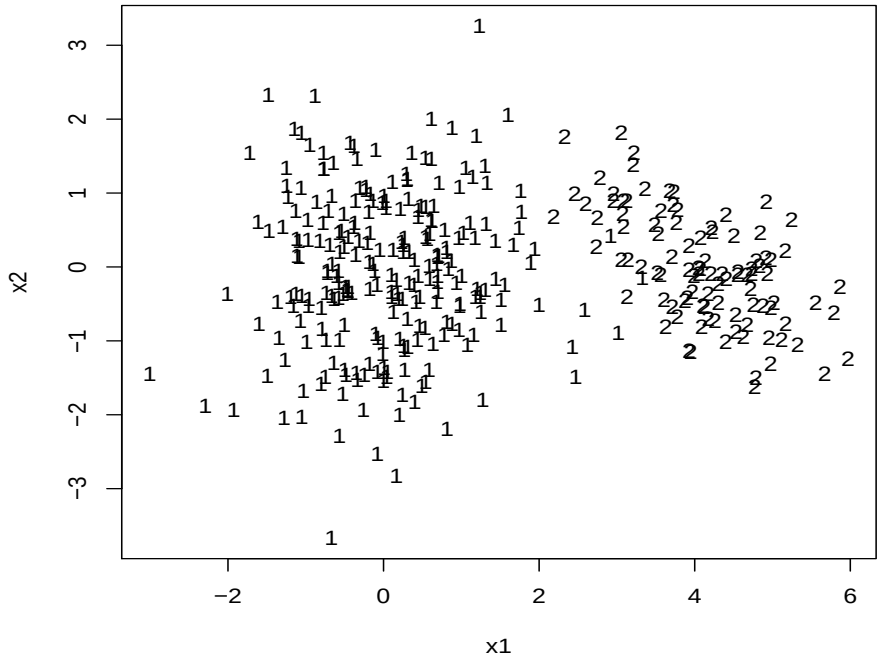
- λ_k is a scalar that controls the *volume* of the cluster
- D_k is a matrix that controls the *orientation* (correlation structure) of the cluster
- A_k is a diagonal matrix that controls the *shape* of the cluster, i.e. the relative spread of each feature.



$\lambda_1 > \lambda_2$ so the first cluster has a bigger volume than the second.



For cluster 1: $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ so both features have equal spread, and for cluster 2: $A = \begin{pmatrix} 1 & 0 \\ 0 & 3 \end{pmatrix}$, i.e. much more spread in feature 2 than feature 1.



For cluster 1: both D and $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ so both features have equal spread and no correlation, and for cluster 2: $A = \begin{pmatrix} 1 & 0 \\ 0 & .25 \end{pmatrix}$, $D = \begin{pmatrix} 0.7 & -0.7 \\ 0.7 & 0.7 \end{pmatrix}$ i.e. $\Sigma \simeq \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$, i.e. the features are negatively correlated in cluster 2.

In Mclust you can set some or all of the λ's, A's and D's to be equal across clusters. This can save a lot of parameters but of course also corresponds to making assumptions about the clustering structure in the data (e.g. that feature dependencies is the same for all clusters, $D_k = D$).
Some examples:

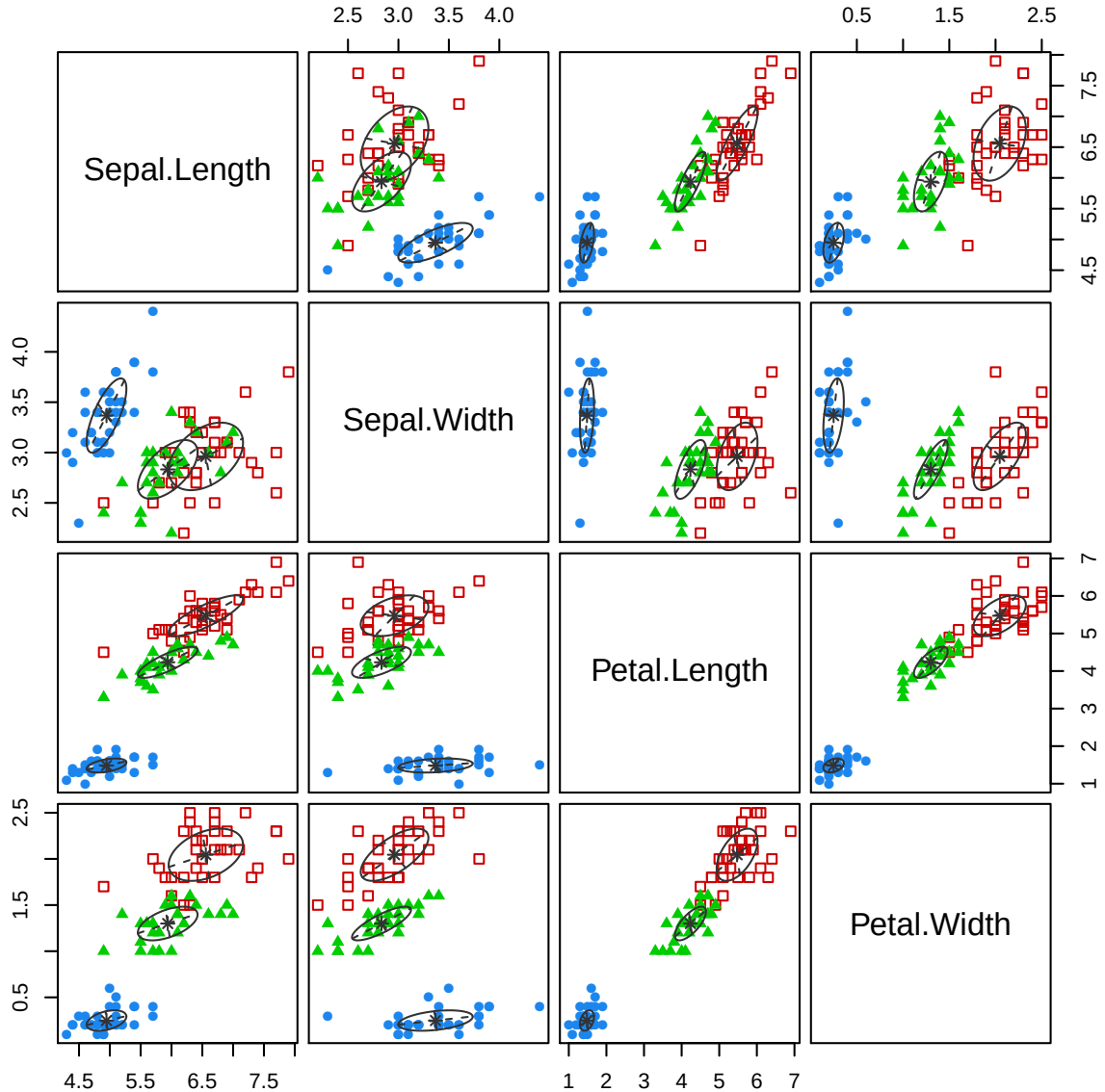
- $\Sigma_k = \Sigma = \lambda I$ assumes that clusters are equal volume, spherical blobs in x-space
- $\Sigma_k = \lambda_k I$ assumes that clusters are different volume but spherical for all clusters
- $\Sigma_k = \lambda A$ assumes that all clusters have the same volume, spread can vary for different features but in the same way for all clusters, and there are not feature-feature dependencies

Some more examples:

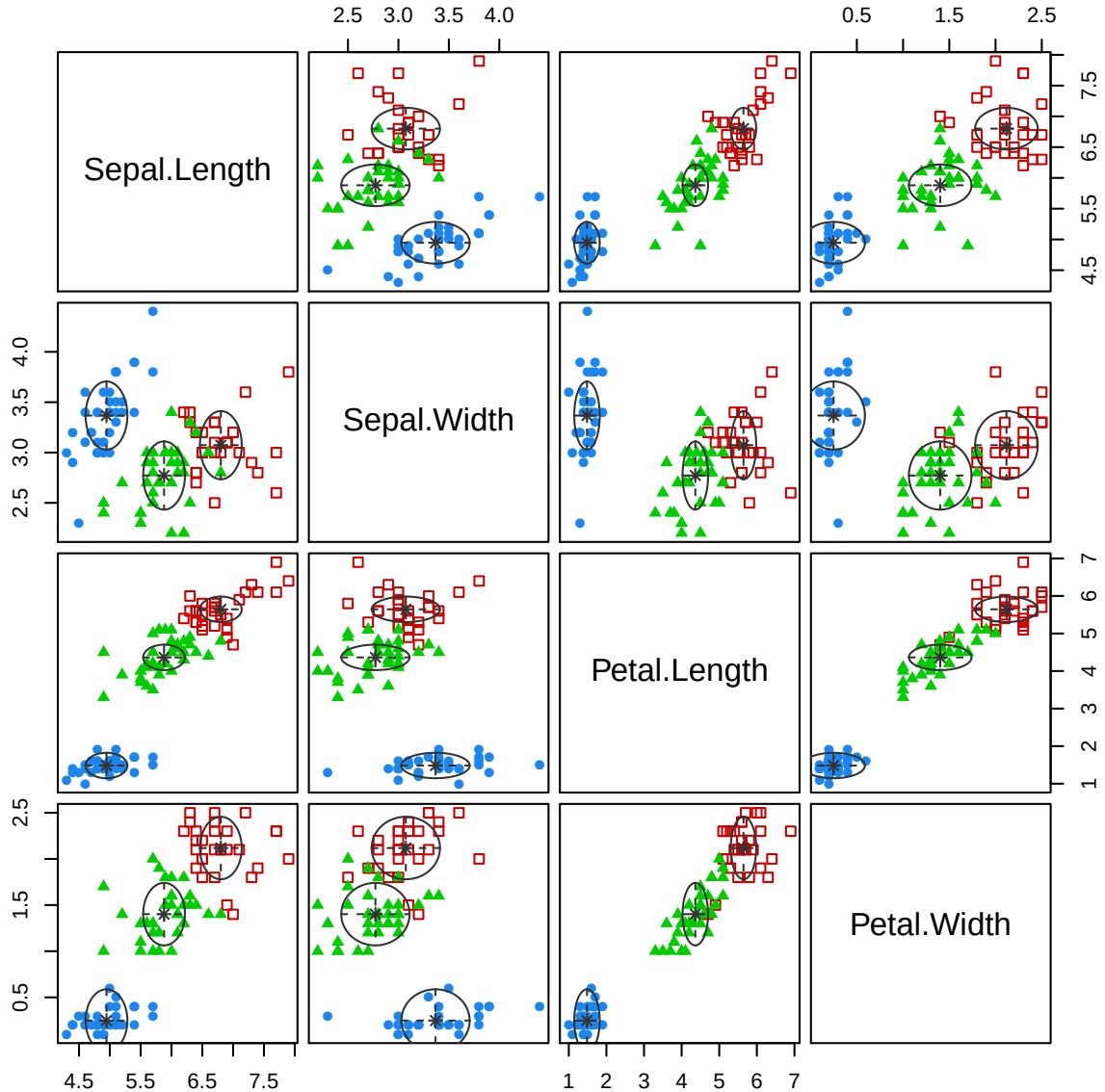
- $\Sigma_k = \lambda A_k$ assumes all clusters have the same volume but can have different spread for different features (e.g. feature 2 more variable than feature 1 in cluster 1, and the opposite is true for cluster 2.)
- $\Sigma_k = \lambda DAD'$ assumes that the volume, shape (feature spread) and orientation (feature dependencies) are the same for all clusters

We can use BIC to select between these special cases for the clusters since we simply count the number of parameters in each model and add to the goodness-of-fit (negative log-likelihood).

Mclust with varying volume, shape and orientation



Mclust with equal volume, spherical distributions



BIC selects varying volume, equal shape and varying orientation for the clusters.