Model based clustering with the packages mclust - teigen - mixsmsn

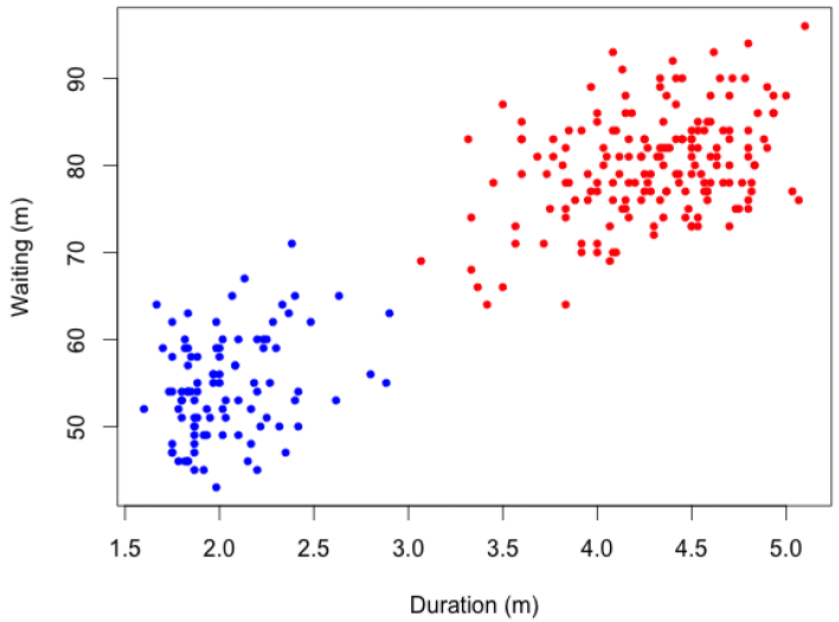
Marta Nai Ruscone

University of Genova, Italy

Stralsund - Germany, November 6, 2023

Preface: clustering

- Clustering is the (statistical) process of uncovering groups in the data
- According to some criterion of similarity or dissimilarity, objects in the same cluster are more similar to each other than objects in other clusters



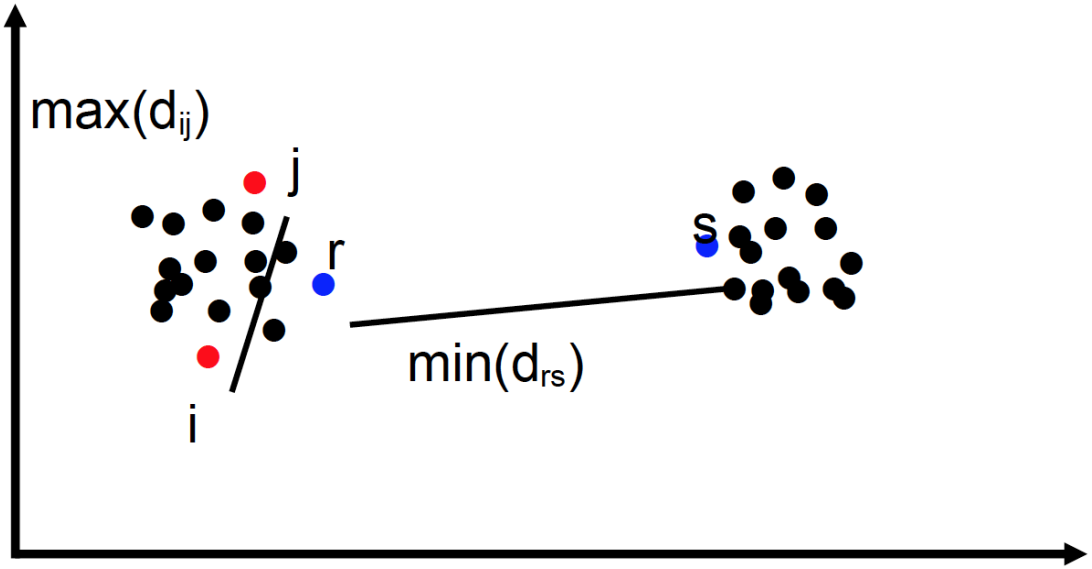
Applications

- **Medicine or Psychiatry**: clusters of patients on the basis of the symptoms
- **Biology**: clusters of tissues on the basis of the gene mutations
- **Environmental sciences**: clusters of species
- **Business**: clusters of firms
- ...

Applications

- **There are many cluster analysis methods**. On many datasets these may produce many different clusterings.
- **Clustering may have different aims**. Different clusterings on the same data may be appropriate for different aims.
- **Particularly**: Within-cluster homogeneity vs. between-cluster separation.
  Both are often relevant, but may be in conflict.

Cluster analysis: Graphically....



In case of 2 groups G_1 and G_2

$$max(d_{ij}) < min(d_{rs})$$

with $i,\ j\ \in G_1,\ r \in G_1,\ s\ \in G_2$.

Sum of Squares decomposition

Objective way to express the concept of similar units (given the clusters!)

$$T = B + W$$

where

- T $\rightarrow$ **Total sum of squares** (total variation - available information)
- B $\rightarrow$ **Between sum of squares** (between-cluster variation - separation)
- W $\rightarrow$ **Within sum of squares** (within-cluster variation - compactness)

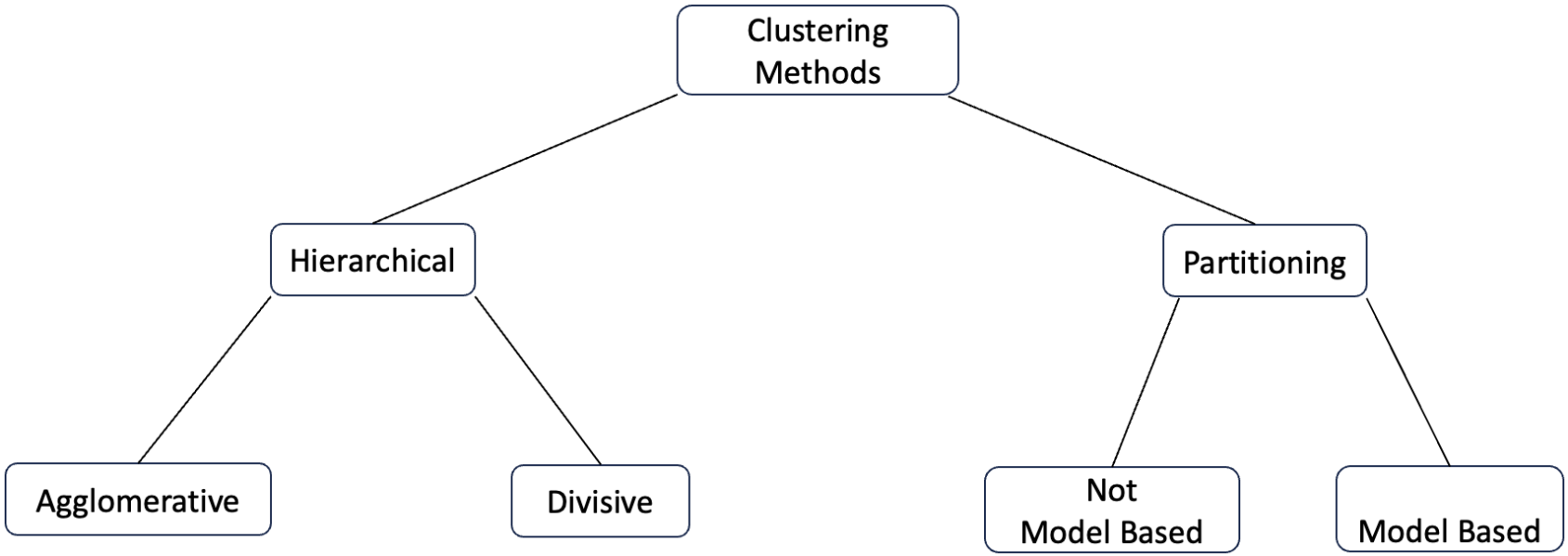
Sum of Squares decomposition

Given p variables observed on n units partitioned in k groups (clusters), each one containing n_g units

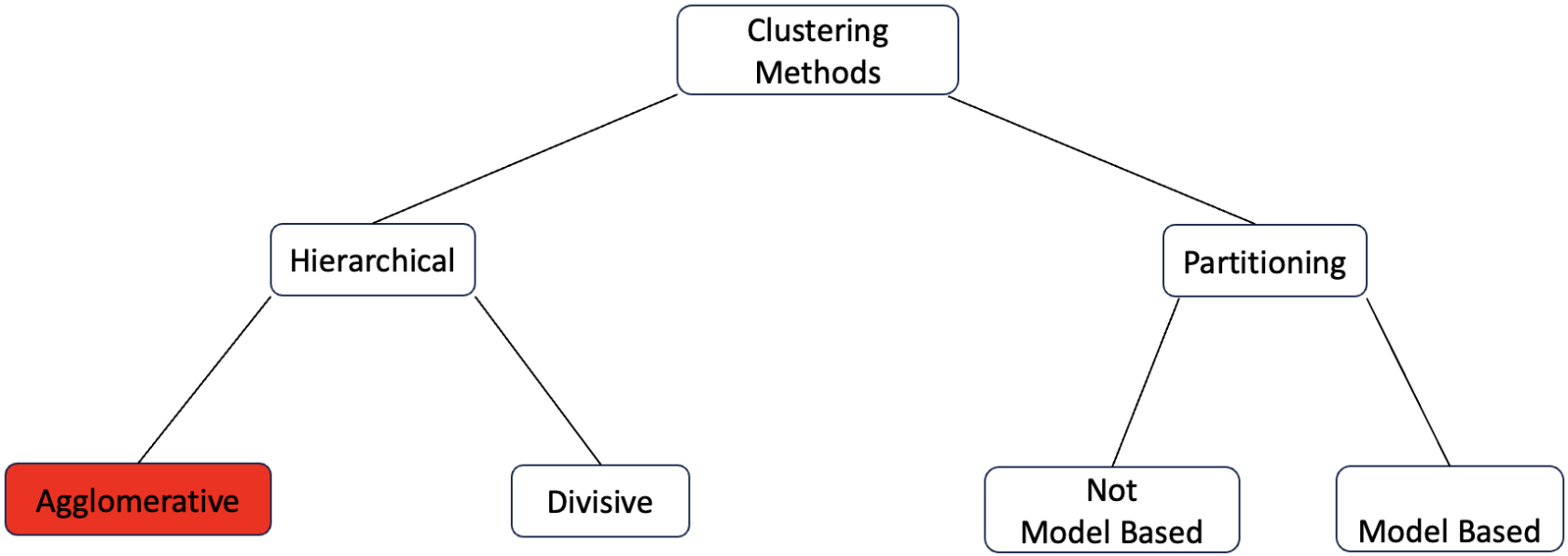
$$
\begin{aligned}
\boldsymbol{T} &= \sum_{i=1}^{n}\sum_{j=1}^{p}(x_{ij} - \bar{x}_j)^2 \\
&= \sum_{g=1}^{k}\sum_{j=1}^{p} n_g(\bar{x}_{j,g} - \bar{x}_j)^2 + \sum_{g=1}^{k}\sum_{j=1}^{p}\sum_{i=1}^{n_g}(x_{ij} - \bar{x}_{j,g})^2 \\
&= \boldsymbol{B} + \boldsymbol{W}
\end{aligned}
$$

- $\bar{x}_j$ is the mean of variable j
- $\bar{x}_{j,g}$ is the mean of variable j for cluster g $(g = 1, \cdots, k)$

Clustering methods



Clustering methods



Aggregative methods

- We start the clustering process by considering the partition with n clusters (each unit represents one cluster) until the partition with 1 cluster (all the units assigned to the single cluster). [from 1 to n clusters: divisive methods]
- Hierarchical clustering methods are criteria aimed to create nested partitions that allow investigations with respect to different within clusters homogeneity levels.
- Those relationships are often represented by dendrograms; by cutting the graph at a certain level of dissimilarity we obtain a partition of objects into different groups.
- Each partition in k groups is optimal in a step-wise sense. It is the best partition in k group one can obtain, given the partition in $k + 1$ group defined at the previous step.

Distance/dissimilarity matrix

- Starting from the dissimilarity matrix, agglomerative hierarchical clustering methods initially consider each unit as a separate cluster, and they subsequently merge in one cluster those units that have the lowest distance/dissimilarity.
- Then, they evaluate dissimilarities between this new cluster and the others, merging again the two groups that have the lowest value in the distance/dissimilarity matrix.
- Different clustering methods in this class refer to the different ways the dissimilarity between clusters is defined and evaluated.

Distance/dissimilarity matrix

Distance measures and functions for quantitative variables are implemented in several functions.

The most common are `dist` of the package **stats** and `daisy` of the package **cluster**.

Function	Package	Option	Distance
`dist`	**stats**	`method = "euclidean"`	Euclidean
		`method = "minkowski"`	Minkowski
		`method = "manhattan"`	Minkowski with $q = 1$
		`method = "maximum"`	Minkowski with $q = +\infty$
`daisy`	**cluster**	`metric = "euclidean"`	Euclidean
		`metric = "manhattan"`	Minkowski with $q = 1$

- Both functions require an object of class matrix or data.frame as input argument x.
- The function daisy offers the possibility of standardizing the data before computing the distance matrix (`stand = TRUE`).
- If one is interested in computing the distance matrix on standardized data by using `dist`, a possible way is to set `x = stand(dataset)` where dataset is the object containing the data at hand.

Distances between two groups: examples

Given two groups, I and J, we can define different distances between I and J, for example:

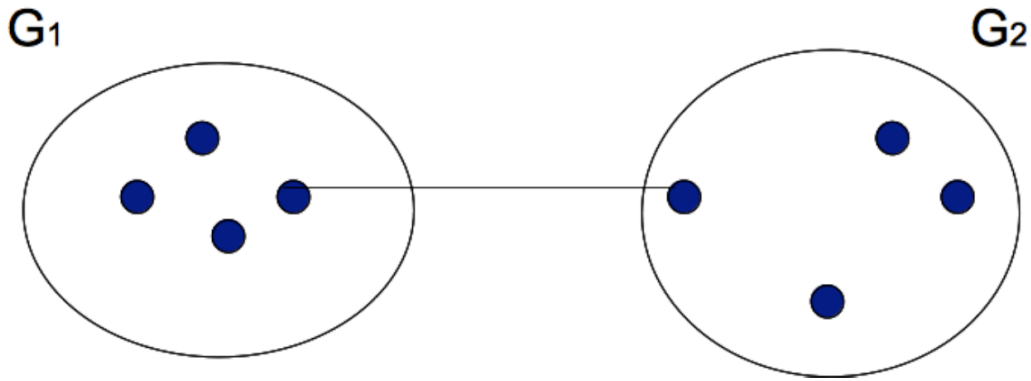
$$d_{SL}(I, J) = \min_{i \in I, j \in J} d(i, j)$$

$$d_{CL}(I, J) = \max_{i \in I, j \in J} d(i, j)$$

$$d_{AL}(I, J) = mean(d(i, j))$$

Single Linkage

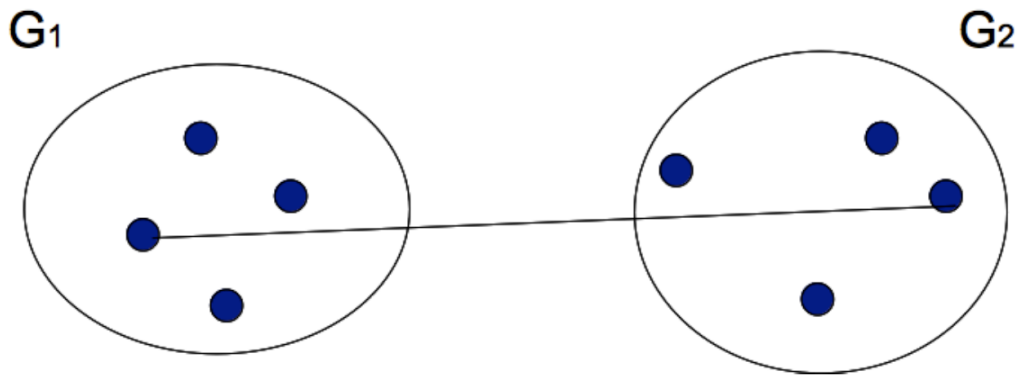
The distance between two groups G_1 and G_2 is defined as the minimum of the distances between any element i of G_1 and any element j of G_2:



Clusters obtained with the Single Linkage method tend to have a long and narrow shape and to be internally poorly homogenous. For this reason, this method is not able to identify overlapping clusters, but it is good when clusters look well separated.

Complete Linkage

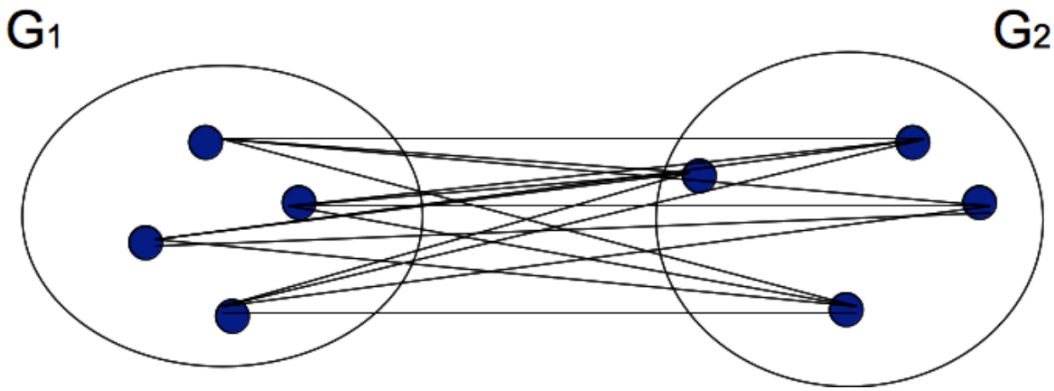
The distance between two groups G_1 and G_2 is defined as the maximum of the distances between any element i of G_1 and any element j of G_2:



Clusters yielded by this procedure tend to be spherical and compact; nothing can be said about their separation.

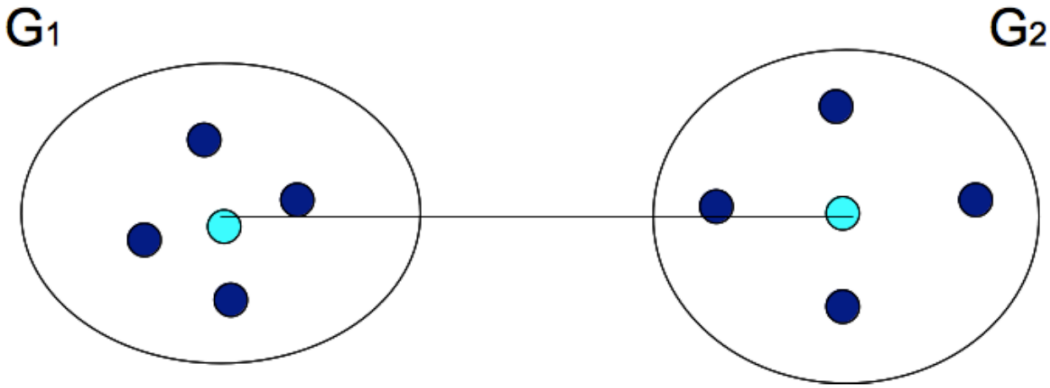
Average Linkage

The distance between two groups G_1 and G_2 is defined as the arithmetic mean of the distances between any element i of G_1 and any element j of G_2:



Centroid Linkage

The distance between two groups G_1 and G_2 is defined as the Euclidean distance between the two cluster centers (the so-called centroids or prototypes), i.e., the centroid or prototype of a cluster is the mean vector of the observed variables computed using the units assigned to the cluster.



A problem that may be encountered with this approach is that dissimilarity of a union is lower than the one measured at the previous merge and this makes the results less interpretable.

Ward method

- The previous methods produce clusters composed by homogenous units. Thus, the within sum of squares W is small, but not the minimum one.
- To minimize W, take it into account in an explicit way when merging the clusters.
- Distance between two clusters is defined in terms of W.
- As usual the classification process starts by considering each unit as a cluster ($W = 0$). Each merge leads to an increase of W; the algorithm is looking for the merge that will imply the smallest increase of W for the whole partition (i.e. the groups that will be merged are those having the minimum between sum of squares).

Distances between two groups

All the available methods implemented in the package **hclust**

method	Method
"single"	Single linkage method
"average"	Average linkage method
"complete"	Complete linkage method
"ward.D"	Variant of Ward's method
"ward.D2"	Ward's method
"mcquitty"	McQuitty method
"median"	Median method
"centroid"	Centroid method

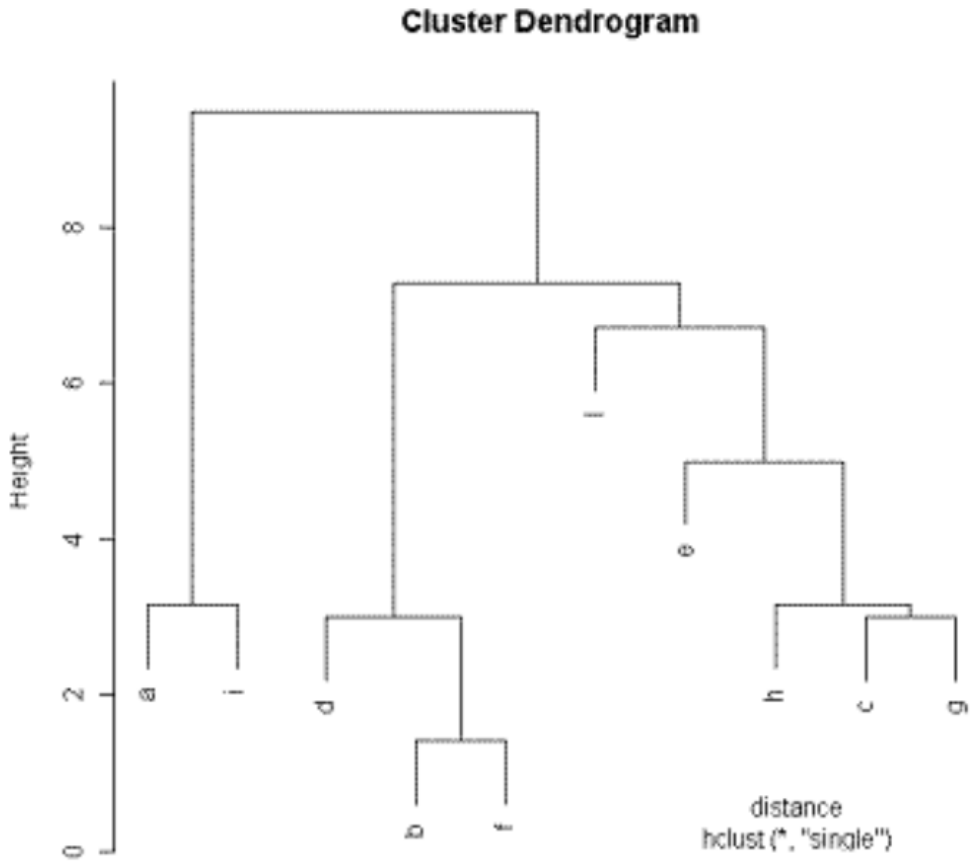
Distances between two groups

All the available methods implemented in the package **agnes**

method	Method
"single"	Single linkage method
"average"	Average linkage method
"complete"	Complete linkage method
"ward.D"	Variant of Ward's method
"ward.D2"	Ward's method
"mcquitty"	McQuitty method
"median"	Median method
"centroid"	Centroid method

Dendrogram

A dendrogram (from Greek dendro "tree" and gramma "drawing") is a tree diagram used to illustrate the arrangement of the clusters produced by hierarchical clustering methods.

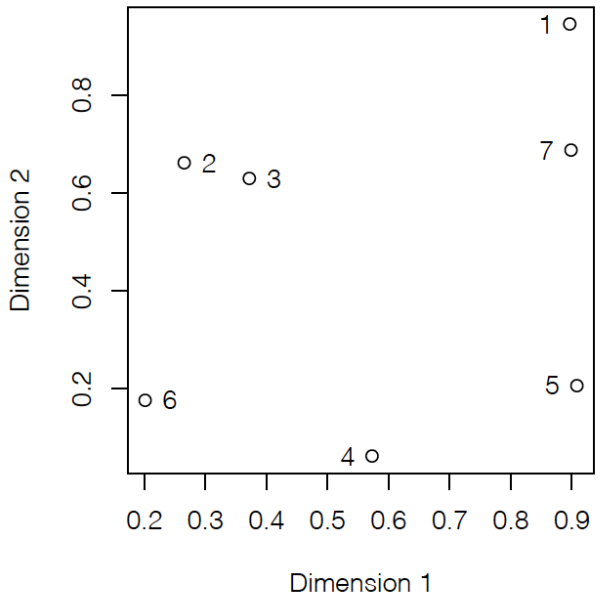


Dendrogram

- The x-axis refers to the units/clusters.
- The y-axis refers to the distance between merged units/clusters.
- Each node represents a group
- Each leaf node is a singleton (i.e., a group containing a single data point)
- Root node is the group containing the whole data set
- Each internal node has two daughter nodes (children), representing the the groups that were merged to form it
- By cutting the graph at a certain level of dissimilarity we obtain a partition of units into different groups.
- Which level? Seek "high jumps" (k groups with low W, $k+1$ groups with high W).

Simple Example

Given these data points, an agglomerative algorithm might decide on a clustering sequence as follows:



Step 1: $\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\}$;

Step 2: $\{1\},\{2,3\},\{4\},\{5\},\{6\},\{7\}$;

Step 3: $\{1,7\},\{2,3\},\{4\},\{5\},\{6\}$;

Step 4: $\{1,7\},\{2,3\},\{4,5\},\{6\}$;

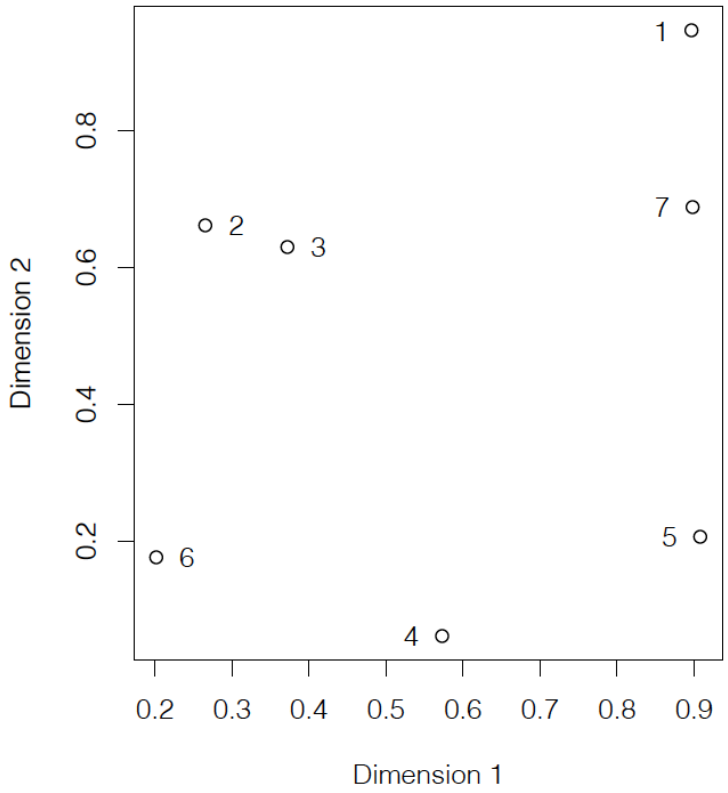
Step 5: $\{1,7\},\{2,3,6\},\{4,5\}$;

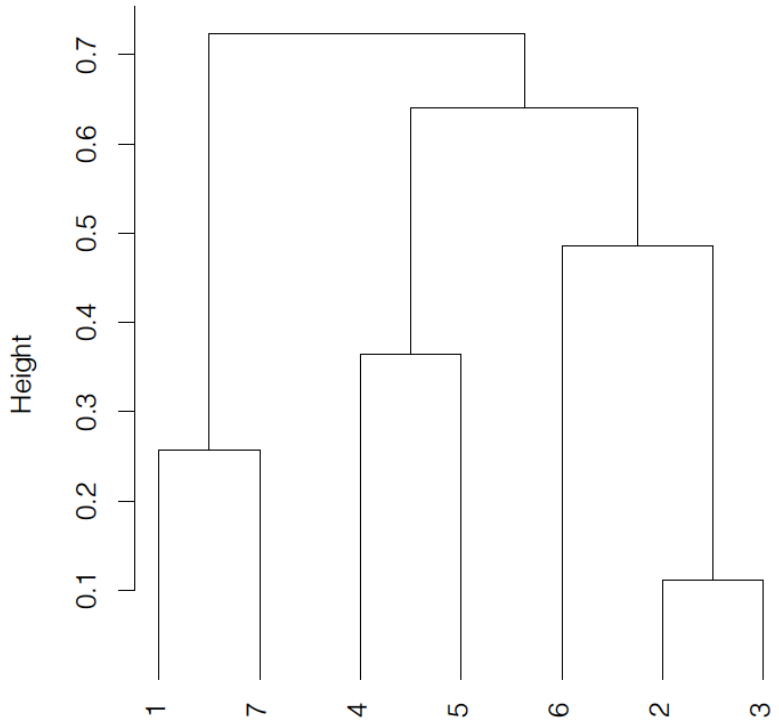
Step 6: $\{1,7\},\{2,3,4,5,6\}$;

Step 7: $\{1,2,3,4,5,6,7\}$.

Simple Example

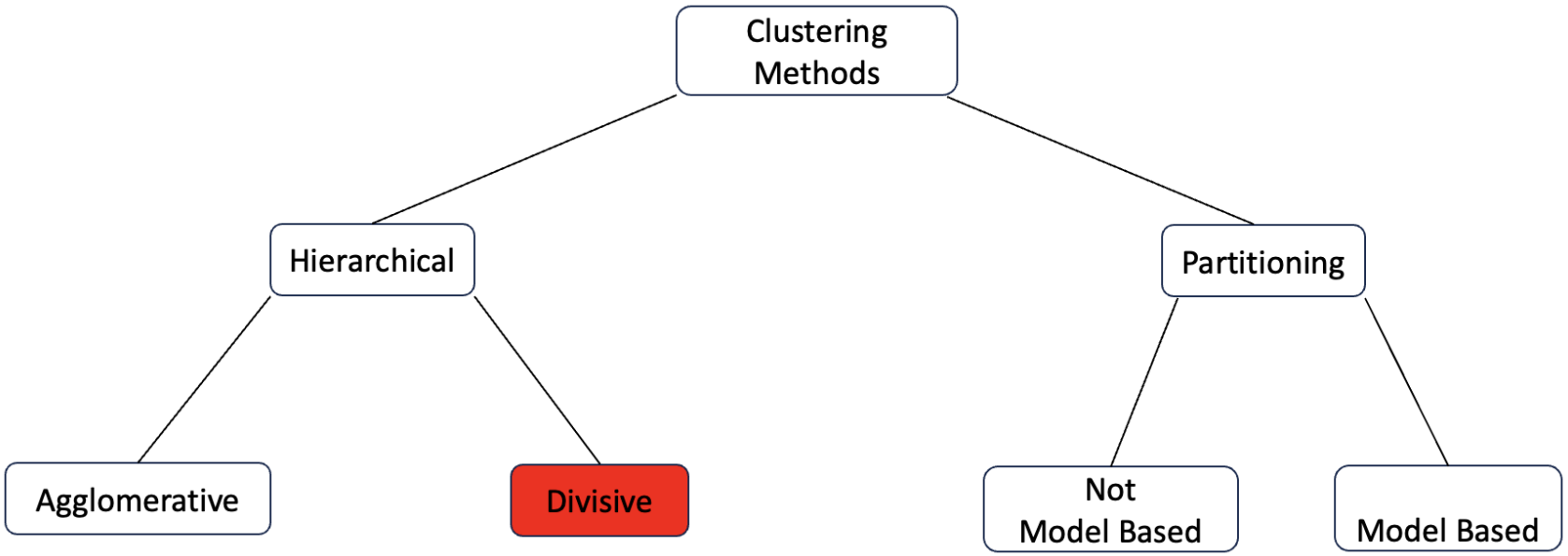
We can also represent the sequence of clustering assignments as a dendrogram:





Note that cutting the dendrogram horizontally partitions the data points into clusters

Clustering methods



Agglomerative vs divisive

Two types of hierarchical clustering algorithms

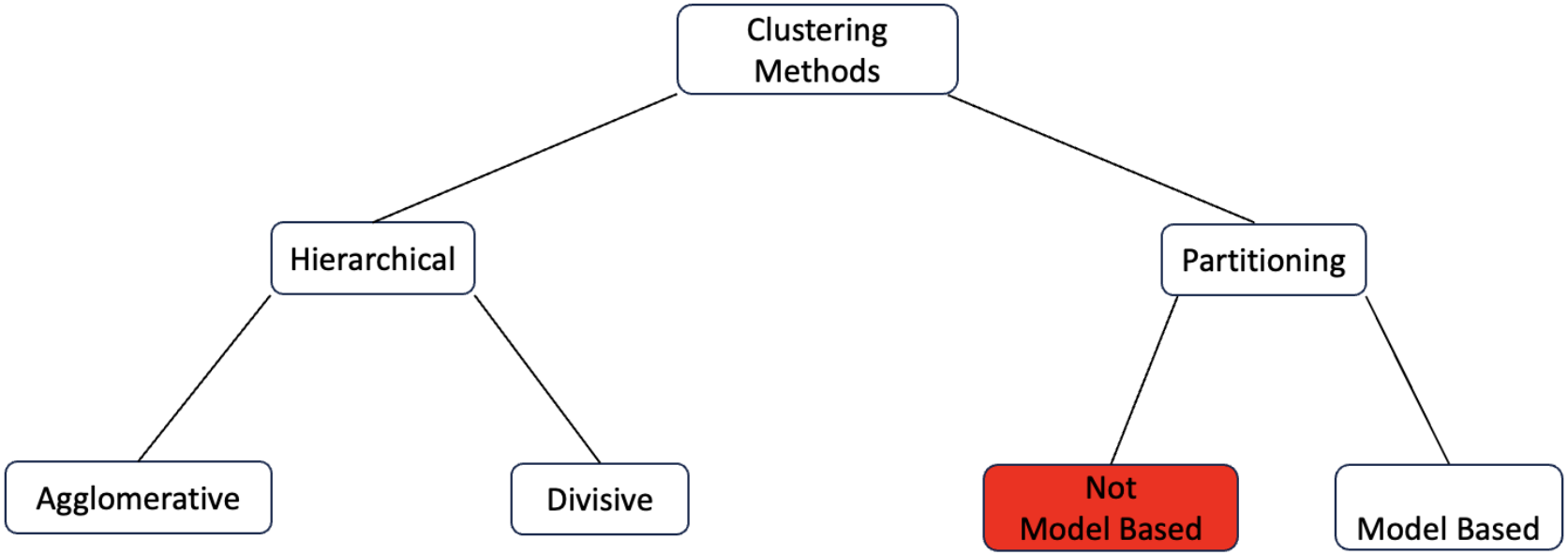
Agglomerative (i.e., bottom-up):

- Start with all points in their own group
- Until there is only one cluster, repeatedly: merge the two groups that have the smallest dissimilarity

Divisive (i.e., top-down):

- Start with all points in one cluster
- Until all points are in their own cluster, repeatedly: split the group into two resulting in the biggest dissimilarity

Clustering methods



Non-hierarchical clustering

- Hierarchical methods generate n different nested classifications that go from n one-element clusters to one cluster which contains all the n observations.
- Non-hierarchical methods give rise to a single partition with k groups, where k is previously specified.

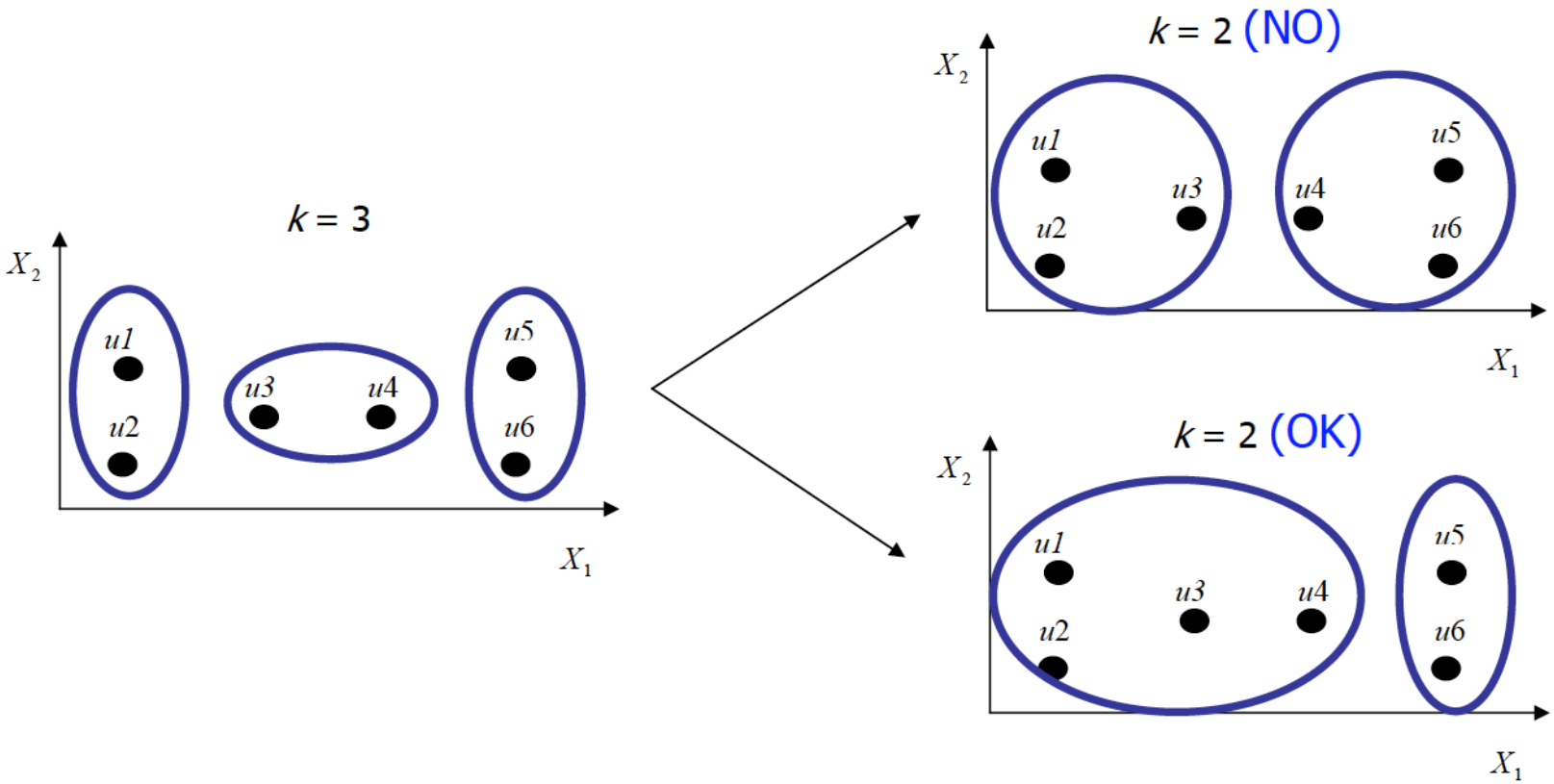
! New problem: we do not know if clusters exist, but we must specify their number!

Do we really need Non-hierarchical methods?

Yes, we do..... Why?

- Low computational complexity (no need for dissimilarity/distance matrix $\mathbf{D}$, simply use $\mathbf{X}$) (when n is large, the computation of $\mathbf{D}$ is cumbersome)
- Low computational complexity (no need for finding n partitions)
- The dendrogram is useless when n is large
- The hierarchy (units merged at a certain step remain in the same groups in the following steps) may represent a relevant problem leading spurious classification:
  - just one bad merging may produce useless results
  - the obtained classification mainly depends on the methods and NOT on the data

Example of spurios classification



It would be reasonable to have two clusters ($u1 - u3$ and $u4 - u6$) but it is not possible: $u3 - u4$ are already merged!

Non-hierarchical methods

- Fixed the number of groups k, we look for the optimal partition among all possible ones.
- Unfortunately, the set of all possible solutions is very very big (for example, if $n = 20$ and $k = 4$, the possible solutions are more than 45 billions!)
- Since we use X, we can compute the sum of squares decomposition
- Given $k(< n)$, we look for the partition such that $W = min$.
- How to do it? By means of an iterative algorithm

Iterative algorithm

Step 0 Randomly choose k centroids (random numbers or a random sample of size k from the n units) The centroids are usually denoted by

$$\mathbf{h}_g = (h_{g1}, h_{g2}, ..., h_{gp}) \qquad (g = 1, \cdots, k)$$

Step 1 Compute the Euclidean distance between every unit and every centroid

$$d(\mathbf{x}_i, \mathbf{h}_g) \qquad (i = 1, \cdots, n, \quad g = 1, \cdots, k)$$

Note: more efficient than hierarchical methods (computation of $n \times k$ distances, $n \times k << n^2$)

Iterative algorithm

Step 2 Assign unit i to the cluster with minimum distance (w.r.t. the centroid)

$$i \in \text{cluster g if } d(\mathbf{x}_i, \mathbf{h}_g) = min\{d(\mathbf{x}_i, \mathbf{h}_1), \cdots, d(\mathbf{x}_i, \mathbf{h}_k)\}$$

Step 3 Compute the new centroids as the average values of the units assigned to the clusters

$$\mathbf{h}_g = \left(\frac{\sum_{i=1}^{n_g} x_{i1}}{n_g}, \frac{\sum_{i=1}^{n_g} x_{i2}}{n_g} \cdots, \frac{\sum_{i=1}^{n_g} x_{ip}}{n_g} \right)$$

being n_g the number of units assigned to cluster g.

Step 4 Repeat Step 1 - Step 3 until convergence.

Convergence is attained whenever, in two consecutive iterations, the centroids remain the same (i.e., the assignments of the units to the clusters do not vary).

(Hard) k-Means clustering algorithm (HkM)

It can be shown that HkM consists in solving a constrained optimization problem.

By introducing the membership degree matrix $\mathbf{U}$ of order $(n \times k)$ with generic element u_{ig} such that

- u_{ig}=1, if i is assigned to cluster g
- u_{ig}=0, otherwise

and, since every unit can be assigned to one and only one cluster,

$$\sum_{g=1}^{k} u_{ig} = 1, \qquad \forall i = 1, \cdots, n$$

HkM: optimization problem

$$\min_{\mathbf{U},\mathbf{H}} J_{\mathrm{H}k\mathrm{M}} = \sum_{i=1}^{n}\sum_{g=1}^{k} u_{ig} d^2\left(\mathbf{x}_i, \mathbf{h}_g\right)$$

$$\text{s.t.} \quad u_{ig} \in \{0,1\}, i = 1, \cdots, n \quad g = 1, \cdots, k$$

$$\sum_{g=1}^{k} u_{ig} = 1, i = 1, \cdots, n$$

where $\mathbf{H}$ is the centroid matrix of order $(k \times p)$

HkM: iterative algorithm

Step 0 Randomly select the centroid matrix $\mathbf{H}^{(t)}$ with $t = 0$.

Step 1 Update the membership matrix $\mathbf{U}^{(t+1)}$ by

- u_{ig}=1, if $g = \text{argmin}_{g=1,\cdots,k} d\left(\mathbf{x}_i, \mathbf{h}_g^{(t)}\right)$
- u_{ig}=0, otherwise

Step 2 Update the centroid matrix $\mathbf{H}^{(t+1)}$ by

$$\mathbf{h}_g^{(t+1)} = \left(\frac{\sum_{i=1}^{n_g} x_{i1}}{n_g}, \frac{\sum_{i=1}^{n_g} x_{i2}}{n_g} \cdots, \frac{\sum_{i=1}^{n_g} x_{ip}}{n_g} \right)' = \frac{\sum_{i=1}^{n_g} \mathbf{x}_i}{n_g}$$

Step 3 Check convergence, i.e. if $\|\mathbf{H}^{(t+1)} - \mathbf{H}^{(t)}\|^2 < \varepsilon$, the algorithm has converged (no changes in $\mathbf{H}$), otherwise set $t = t + 1$ and go to Step 1.

HkM: iterative algorithm

Step 0 Randomly select the membership matrix $\mathbf{U}^{(t)}$ with $t = 0$.

Step 1 Update the centroid matrix $\mathbf{H}^{(t+1)}$ by

$$\mathbf{h}_g^{(t+1)} = \left(\frac{\sum_{i=1}^{n_g} x_{i1}}{n_g}, \frac{\sum_{i=1}^{n_g} x_{i2}}{n_g} \cdots, \frac{\sum_{i=1}^{n_g} x_{ip}}{n_g} \right) = \frac{\sum_{i=1}^{n_g} \mathbf{x}_i}{n_g}$$

Step 2 Update the membership matrix $\mathbf{U}^{(t+1)}$ by

- u_{ig}=1, if $g = \text{argmin}_{g'=1,\cdots,k} d\left(\mathbf{x}_i, \mathbf{h}_{g'}^{(t)}\right)$
- u_{ig}=0, otherwise

Step 3 Check convergence, i.e. if $\|\mathbf{U}^{(t+1)} - \mathbf{U}^{(t)}\|^2 < \varepsilon$, the algorithm has converged (no changes in $\mathbf{U}$), otherwise set $t = t + 1$ and go to Step 1.

How does HkM work?

- It does not explore all the possible partitions of n units in k groups.
- Rather, it explores a limited subset of potential optimal partitions where the concept of optimality is expressed in terms of *minW* (i.e, the loss function).
- Two fundamental comments (closely related):
  1. The limited subset of potential optimal partitions depends on *Step* 0.
  2. It is not guaranteed to obtain the partition such that *minW* is attained (LOCAL MINIMAL...).

What can we do?

- Run the HkM algorithm more than one time (5? 10?, more?) in order to increase the chance of getting the optimal partition.
- Keep and interpret the solution with the minimum loss function value (among all the solutions obtained considering different random starts (RS)).

Example (5 starting points):

[The loss function value upon convergence is recorded]

Case 1: 30.54 30.54 30.54 30.54 30.54

Same loss function value (same matrices **H** and **U**). High confidence that we got the optimal partition (GLOBAL OPTIMUM) (different starting points led to the same conclusion)

Example (5 starting points):

Case 2: 30.54 30.54 30.54 37.48 30.54

Four times (out of five) same **minimum** loss function value.

For the starting point n. 4, we got a higher value (LOCAL OPTIMUM).

Good confidence that we got the GLOBAL OPTIMUM in the remaining runs.

Case 3: 45.11 32.48 39.22 49.49 30.54

Different loss function values.

We cannot assess whether 30.54 is the GLOBAL or a LOCAL OPTIMUM. Increase the number of random starts hoping that Case 2 occurs.

(Hard) k-Medoids clustering algorithm (HkMed)

Similar to HkM algorithm, but simpler in the cluster interpretation.....why?

- The obtained clusters are interpreted in terms of the centroids.
- The centroids are fictitious units (average values for the p variables computed using the units belonging to the same clusters).
- The concept of centroids is not simple for a non-statistical audience!

Replace the fictitious centroids with real units

- In this way, we can interpret the clusters on the basis of the observed units.
- We therefore select the centroids as a subset of size k of the n units. These are no longer called centroids, but medoids. We get

$$\min_{\mathbf{U},\mathbf{M}} J_{\mathrm{HkMed}} = \sum_{i=1}^{n} \sum_{g=1}^{k} u_{ig} d^2(\mathbf{x}_i, \mathbf{m}_g)$$

$$\text{s.t.} \quad u_{ig} \in \{0,1\}, i = 1, \cdots, n \quad g = 1, \cdots, k$$

$$\sum_{g=1}^{k} u_{ig} = 1, i = 1, \cdots, n \qquad \{\mathbf{m}_1, \cdots, \mathbf{m}_k\} \subseteq \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$$

- HkMed algorithm is often referred to as Partitioning Around Medoids (PAM)

PAM: comments

As for HkM algorithm, also for PAM:

- I Constrained optimization problem
- Risk of local optima
- More than one random start is recommended
- Iterative algorithm

Main difference:

- PAM can be computationally cumbersome (especially for large values of n). To understand the reason way, let us see the iterative algorithm (similar to the HkM one except for *Step* 1).

PAM: iterative algorithm

Step 0 Randomly select the membership matrix $\mathbf{U}^{(t)}$ with $t = 0$.

Step 1 Update the medoid matrix $\mathbf{M}^{(t+1)}$ by

$$\mathbf{m}_g^{(t+1)} = \text{argmin}_{\ i=1,\cdots,n} \sum_{i'=1}^{n_g} d(\mathbf{x}_i, \mathbf{x}_{i'})$$

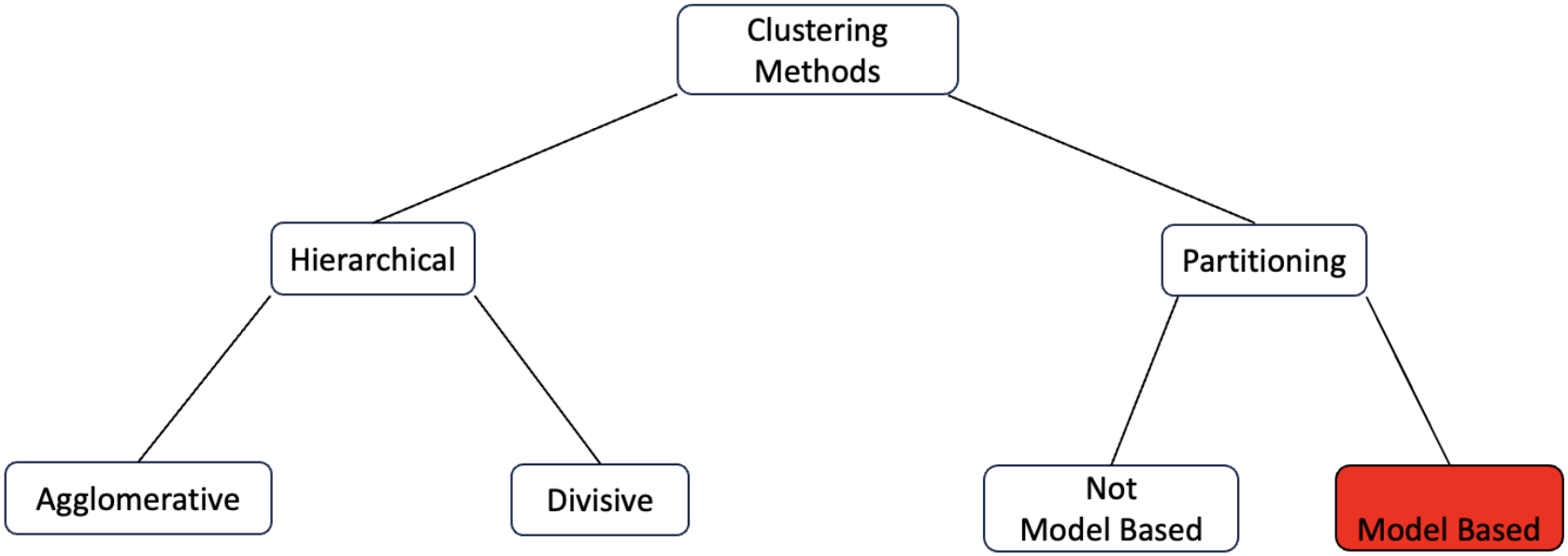
(In each cluster, the medoid minimizes the sum of distances within the cluster)

Step 2 Update the membership matrix $\mathbf{U}^{(t+1)}$ by

- $u_{ig}^{(t+1)}=1$, if $g = \text{argmin}_{\ g'=1,\cdots,k} d\left(\mathbf{x}_i, \mathbf{m}_{g'}^{(t+1)}\right)$
- $u_{ig}^{(t+1)}=0$, otherwise

Step 3 Check convergence, i.e. if $\|\mathbf{U}^{(t+1)} - \mathbf{U}^{(t)}\|^2 < \varepsilon$, the algorithm has converged (no changes in $\mathbf{U}$), otherwise set $t = t + 1$ and go to Step 1.

Clustering methods

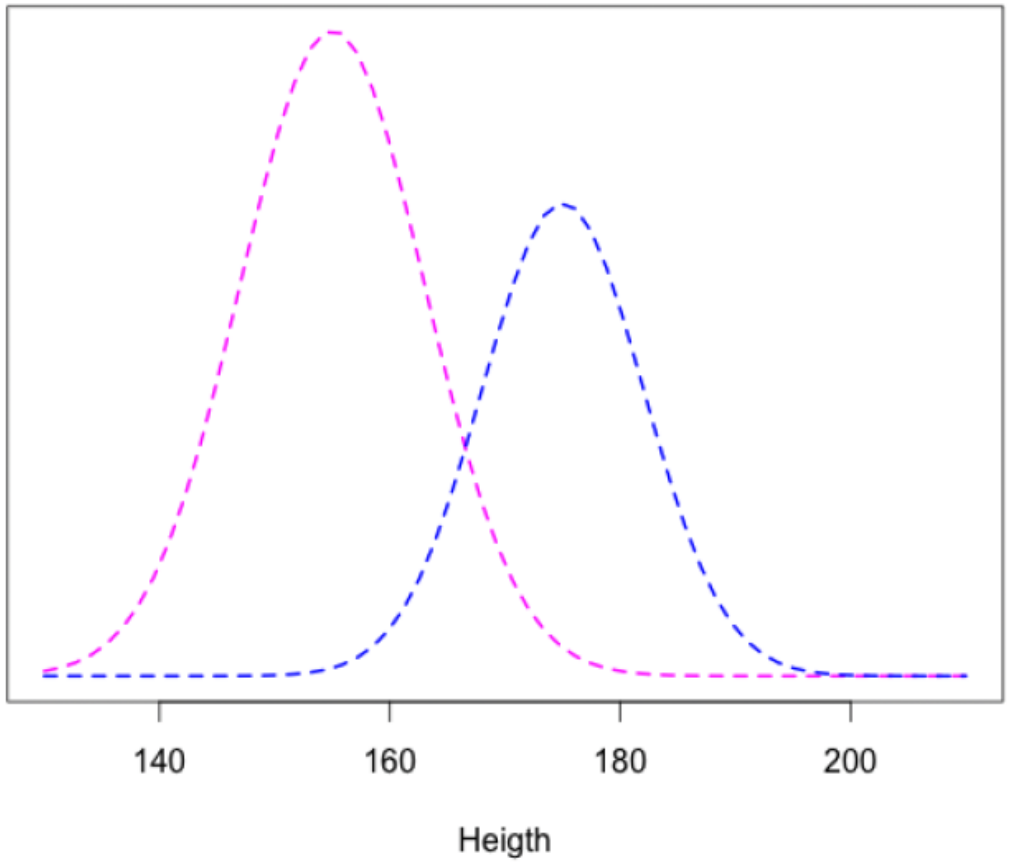


Model-based clustering

- One disadvantage of hierarchical clustering algorithms, k-means algorithms and others is that they are largely heuristic and not based on formal models. Formal inference is not possible.
- Not necessarily a disadvantage since clustering is largely exploratory.
- Model-based clustering is an alternative. Banfield and Raftery (1993, *Biometrics*) is the classic reference. A more comprehensive and up-to-date reference is Melnykov and Maitra (2010, *Statistics Surveys*) also available on Professor Maitra's "Manuscripts Online" link.
- SAS will not implement model-based clustering algorithms.
- With R, you need to load a package called `mclust` and accept the terms of the (free) license. `mclust` is a very good package, but it can have issues with initialization.

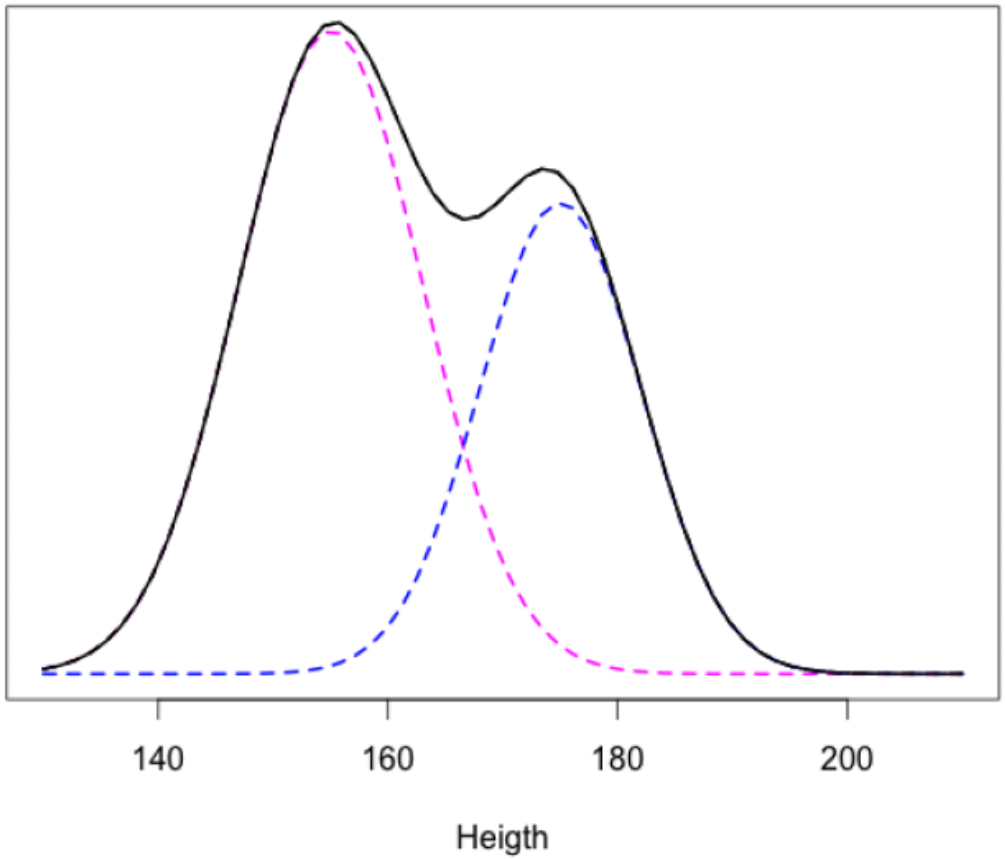
Model-based clustering

- A population is a mixture of sub-populations
- Each component is modelled through a probability density function
- A component can be considered a cluster



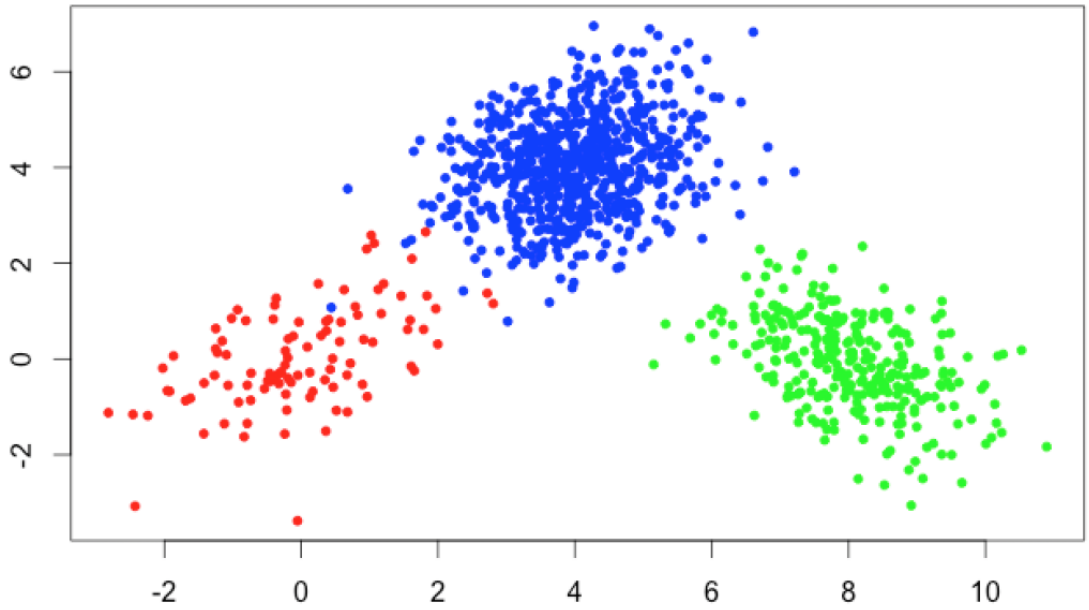
Model-based clustering

- A population is a mixture of sub-populations
- Each component is modelled through a probability density function
- A component can be considered a cluster



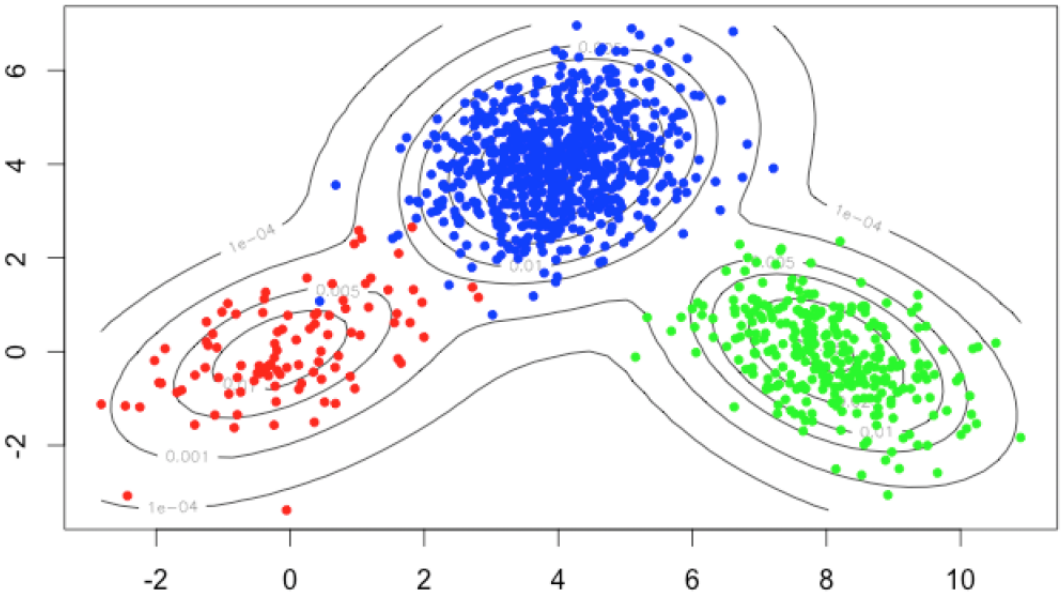
Model-based clustering

- A population is a mixture of sub-populations
- Each component is modelled through a probability density function
- A component can be considered a cluster



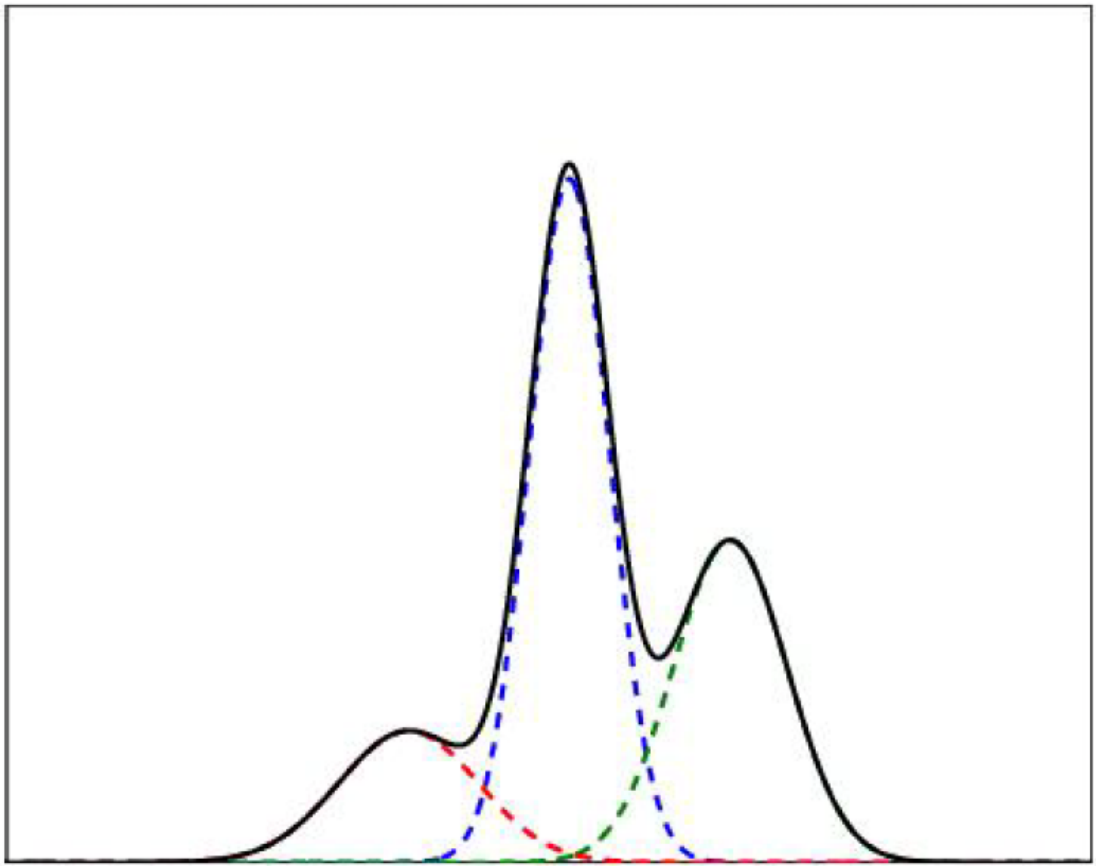
Model-based clustering

- A population is a mixture of sub-populations
- Each component is modelled through a probability density function
- A component can be considered a cluster



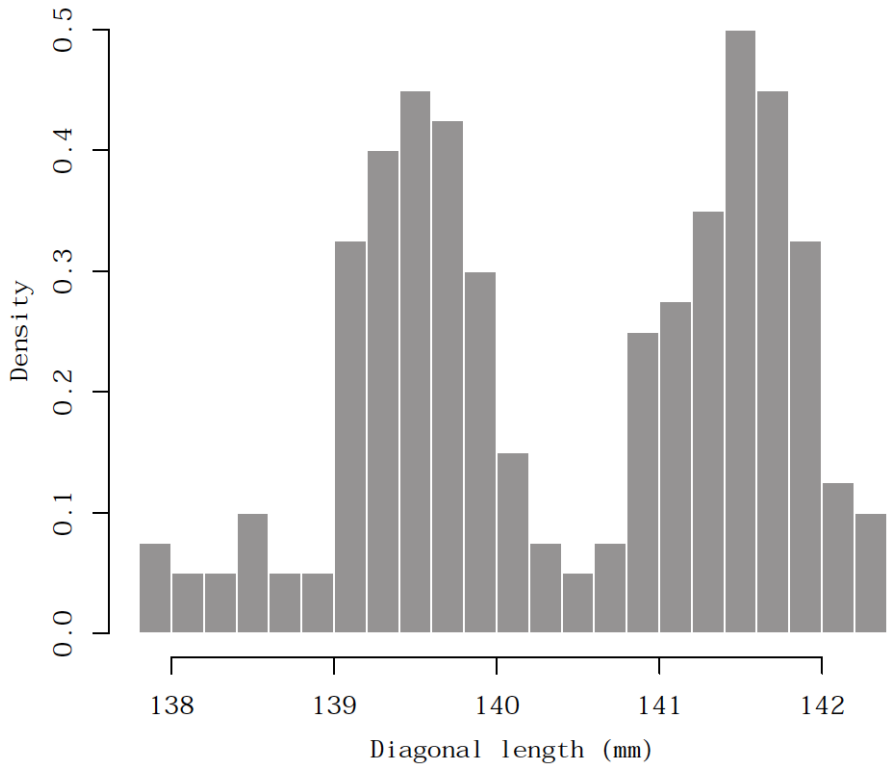
Model-based clustering

- A population is a mixture of sub-populations
- Each component is modelled through a probability density function
- A component can be considered a cluster



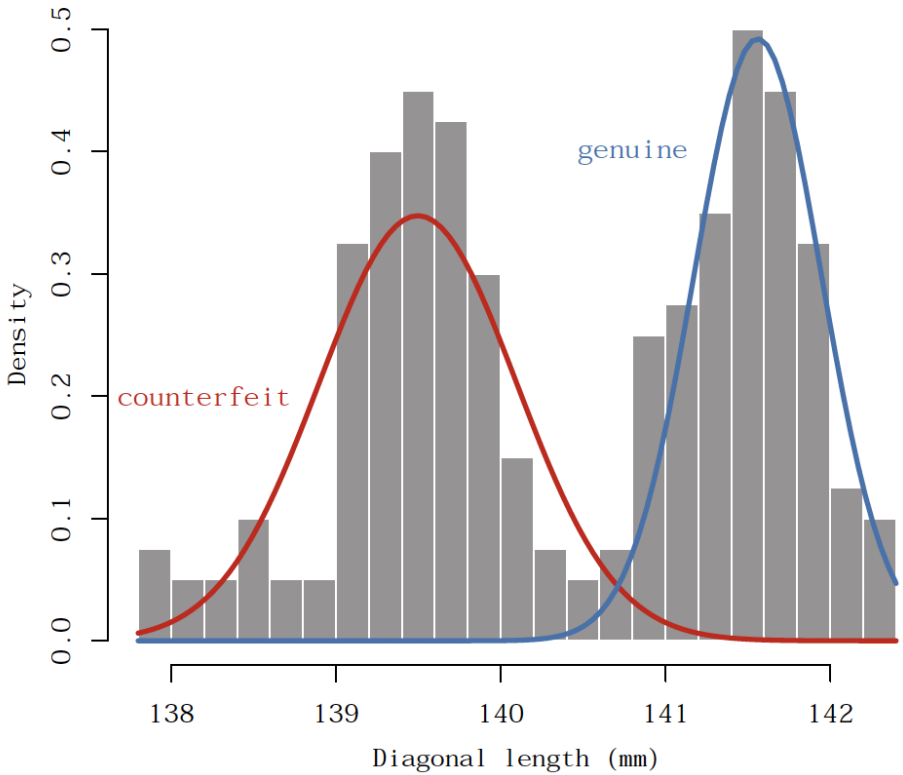
Model-based clustering - Swiss banknotes classification

- Data on geometrical measurements of 200 Swiss banknotes
- Are the observations coming from the same populations?



Model-based clustering - Swiss banknotes classification

- Data on geometrical measurements of 200 Swiss banknotes
- Are the observations coming from the same populations?

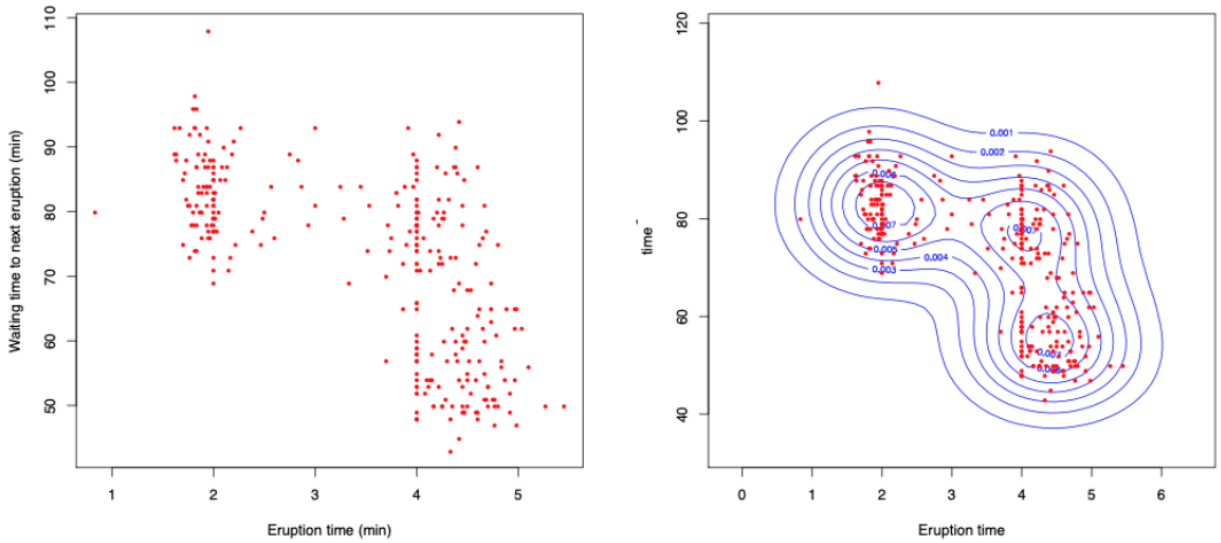


Basic idea behind Model-based Clustering

- Sample observations arise from a distribution that is a mixture of two or more components.
- Each component is described by a density function and has an associated probability or "weight" in the mixture.
- In principle, we can adopt any probability model for the components, but typically we will assume that components are p-variate normal distributions. (This does not necessarily mean things are easy: inference in tractable, however.)
- Thus, the probability model for clustering will often be a mixture of multivariate normal distributions.
- Each component in the mixture is what we call a cluster.

Example: Old Faithful eruptions

Data: 272 observations of the waiting time between eruptions and the duration of the eruptions of Old Faithful (`geyser` in `MASS`)




```
library(KernSmooth)
est <- bkde2D(geyser[, 2:1], bandwidth=c(0.7, 7))
contour(est$x1, est$x2, est$fhat, col = "blue", ylab = "Waiting
time", xlab = "Eruption time")
points(geyser[,2:1], col = "red", pch = 16)
```

Old Faithful Geyser Data (cont'd)

- There seem to be at least three components (perhaps four) in the mixture.
- Contours suggest a mixture of three approximately bivariate normal distributions.
- Since the clouds of points form ellipses that appear to be "similar" in terms of volume, shape and orientation, we might anticipate that the three components (or four?) of this mixture might have homogeneous covariance matrices.

Model-based clustering (continued)

- Set-up: n p-dimensional observations $x_1, \cdots, x_n$. We assume that the joint distribution is a mixture of G components, each of which is multivariate normal with density $f_k(x|\mu_k, \Sigma_k), k = 1, ..., G$.

- The mixture model is then

$$
\begin{aligned}
f(x|\pi, \mu, \Sigma) &= \prod_{i=1}^{n} \sum_{k=1}^{G} \pi_k f_k(x_i|\mu_k, \Sigma_k) \\
&= \text{const.} \prod_{i=1}^{n} \sum_{k=1}^{G} \pi_k |\Sigma_k|^{-\frac{1}{2}} \exp\{-\frac{1}{2}(x_i - \mu_k)' \Sigma_k^{-1} (x_i - \mu_k)\},
\end{aligned}
$$

  where π_k = probability that x_i belongs to the kth component ($0 < \pi_k < 1$, $\sum_k \pi_k = 1$).

- Difficult to find MLEs of these parameters directly: use EM.

Parameter Estimation

- Suppose we observed the random variables which indicates which component each $\boldsymbol{x}_i$ belongs to.
- Let $\zeta_{ik} = 1$ if ith observation belongs to the kth group, zero otherwise.
- With complete data, obtaining the MLE of the component parameters would be trivial:

$$\hat{\boldsymbol{\mu}}_k = \frac{\sum_{i \in E_k} \boldsymbol{x}_i}{n_k}, \quad \hat{\pi}_k = \frac{n_k}{n}, \quad \hat{\boldsymbol{\Sigma}}_k = \frac{\sum_{i \in E_k} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)'}{n_k},$$

  where $E_k = \{i : \gamma_i = k\}$ and n_k is the number of elements in E_k.
- Unfortunately, we do not know the ζ_{ik}s, so we treat these as missing observations.

Parameter Estimation via EM

- Note: If the group identifiers were known, estimation would be easy.
- Then the complete loglikelihood and corresponding Q-function are:

$$\ell(\boldsymbol{\vartheta}; \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{K}\zeta_{ik}\left\{\log|\boldsymbol{\Sigma}_k| + (\boldsymbol{x}_i - \boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)\right\}$$
$$+\sum_{i=1}^{n}\sum_{k=1}^{K}\zeta_{ik}\log\pi_k - \frac{pn}{2}\log 2\pi.$$

$$Q(\boldsymbol{\vartheta}; \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{K}\pi_{ik}\left\{\log|\boldsymbol{\Sigma}_k| + (\boldsymbol{x}_i - \boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)\right\}$$
$$+\sum_{i=1}^{n}\sum_{k=1}^{K}\pi_{ik}\log\pi_k - \frac{pn}{2}\log 2\pi.$$

where $\pi_{ik} = E(\zeta_{ik}|\boldsymbol{x}_i)$ is calculated at the E-step.

The E- and M-steps in Parameter Estimation

- The E-step at the sth iteration:

$$\pi_{ik}^{(s)} = \frac{\pi_k^{(s-1)} \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k^{(s-1)}, \boldsymbol{\Sigma}_k^{(s-1)})}{\sum_{k'=1}^{K} \pi_{k'}^{(s-1)} \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_{k'}^{(s-1)}, \boldsymbol{\Sigma}_{k'}^{(s-1)})}.$$

- M-step at the sth iteration:

$$\pi_k^{(s)} = \frac{1}{n} \sum_{i=1}^{n} \pi_{ik}^{(s)}, \qquad \boldsymbol{\mu}_k^{(s)} = \frac{\sum_{i=1}^{n} \pi_{ik}^{(s)} \boldsymbol{x}_i}{\sum_{i=1}^{n} \pi_{ik}^{(s)}},$$

$$\text{and } \boldsymbol{\Sigma}_k^{(s)} = \frac{\sum_{i=1}^{n} \pi_{ik}^{(s)} (\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(s)})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(s)})'}{\sum_{i=1}^{n} \pi_{ik}^{(s)}}.$$

- iterate from initial values (**need to choose carefully**) till convergence.

Modeling Σ_k

- The components or clusters in the mixture model have elipsoidal shape and are centered at μ_k.
- However, the G groups need not have homogenous covariance matrices.

- Since attributes of Σ_k determine the geometric features of the kth component, we can parameterize each Σ_k in the mixture model as

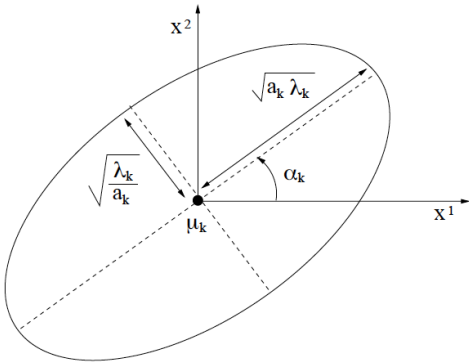
$$\Sigma_k = \lambda_k D_k A_k D_k',$$

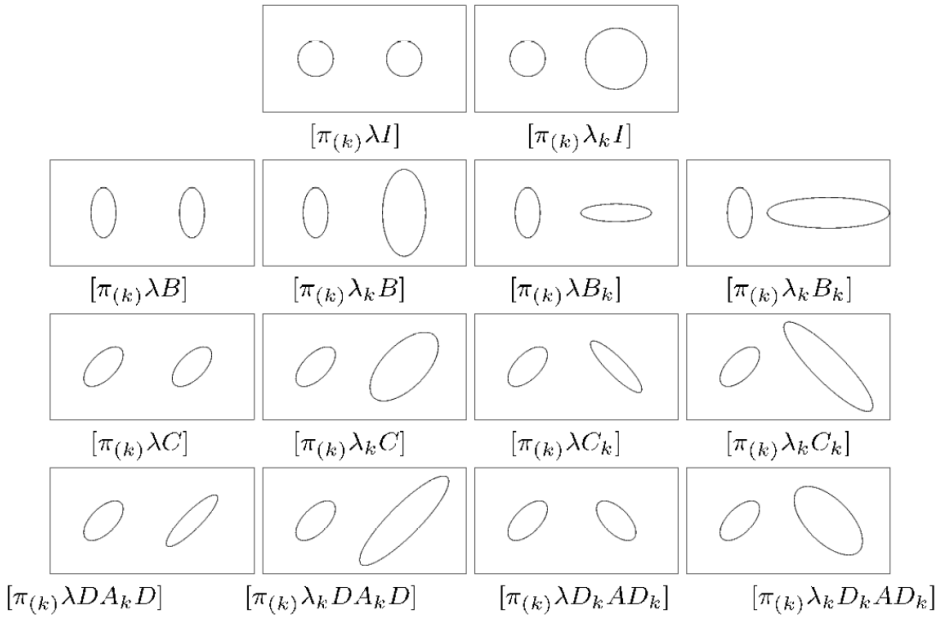
where D_k is the orthogonal matrix of eigenvectors of Σ_k, A_k is a diagonal matrix with elements proportional to the eigenvalues of Σ_k and λ_k is a scalar.

- D_k determines the orientation of the principal components of Σ_k, A_k determines the shape of the density countours, and λ_k determines the volume of the ellipsoid, which is proportional to $\lambda_k^p |A_k|$.

Geometric interpretation of Σ_k

$$\Sigma_k = \underbrace{\lambda_k}_{\text{volume}} \cdot \underbrace{\mathbf{D}_k}_{\text{orientation}} \cdot \underbrace{\mathbf{A}_k}_{\text{shape}} \cdot \mathbf{D}_k'$$





Mclust to implement algorithm

- The model-based clustering algorithm can be implemented using mclust package (`Mclust` function) in `R`.
- Mclust uses an identifier for each possible parametrization of the covariance matrix that has three letters: `E` for "equal", `V` for "variable" and `I` for "coordinate axes".
- The first identifier refers to volume, the second to shape and the third to orientation. For example:
  - `EEE` means that the G clusters have the same volume, shape and orientation in $p-$dimensional space.
  - `VEI` means variable volume, same shape and orientation equal to coordinate axes.
  - `EIV` means same volume, spherical shape and variable orientation.
- There are a total of 10 combinations of volume, shape and orientation included in the package.

Choosing the best model

- Given G, how do we choose a model?
- Any model selection criterion (AIC, likelihood ratio, BIC) can be used to select the best fitting model given G.
- Mclust uses the Bayesian Information Criterion (BIC, or Schwarz Information Criterion) to choose the best model given G.

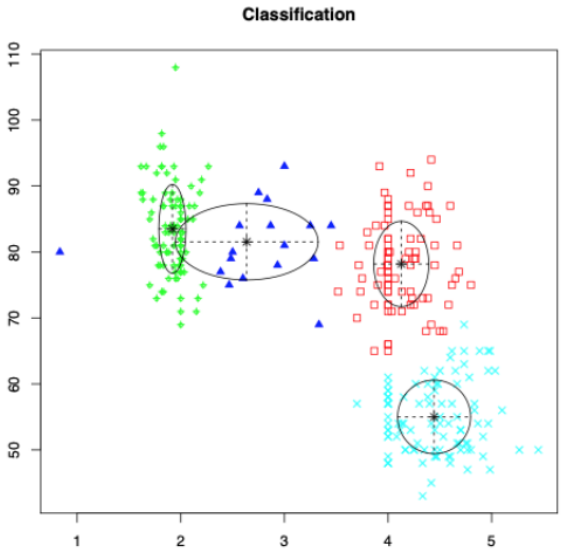
$$BIC = -2\log(L) + m\log(n),$$

where L is the likelihood function and m is the number of free parameters to be estimated. A model with low BIC fits the data better than one with high BIC.

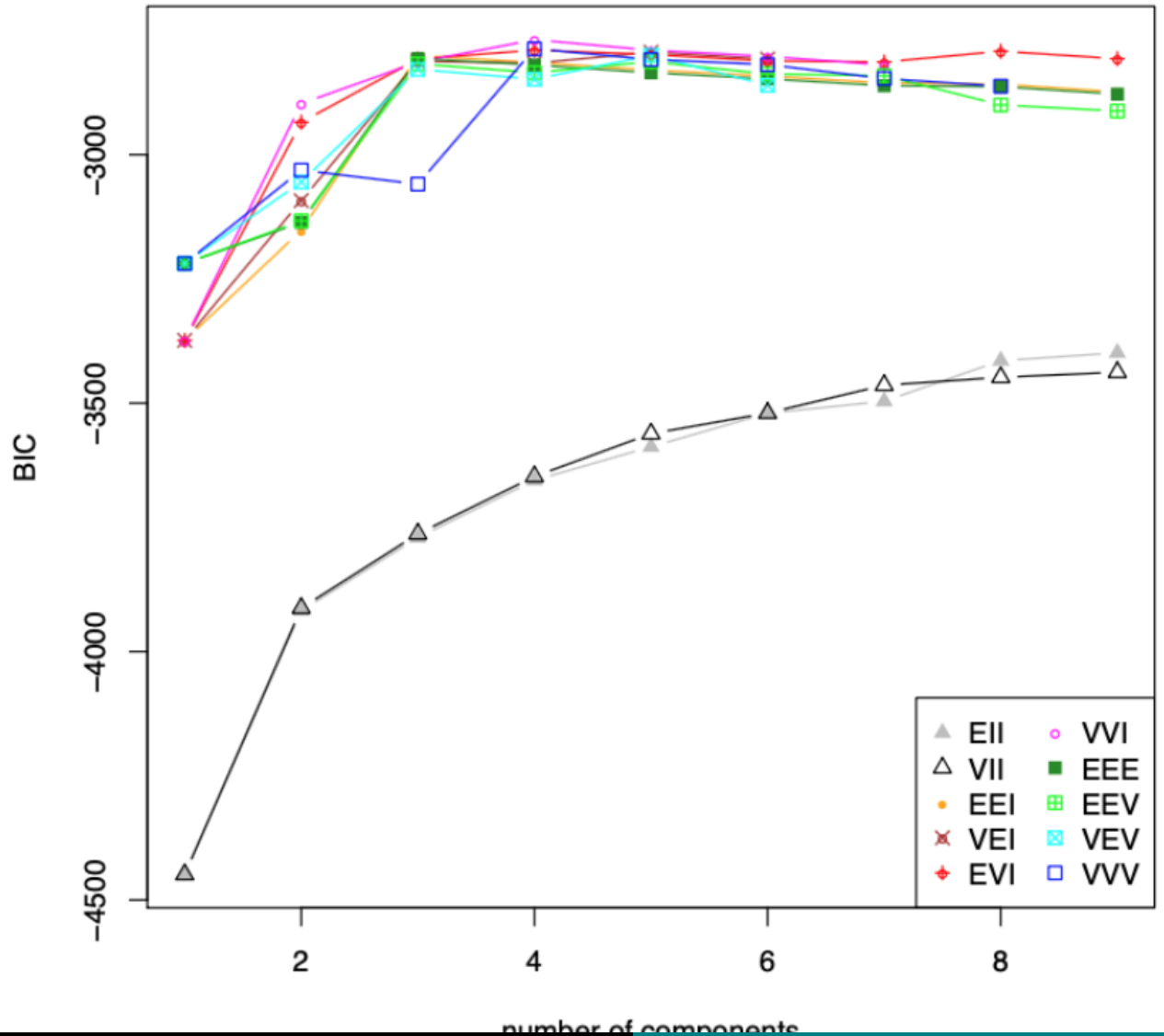
- BIC penalizes large models (large m) more than AIC for which the penalty is $2m$.

Back to Old Faithful

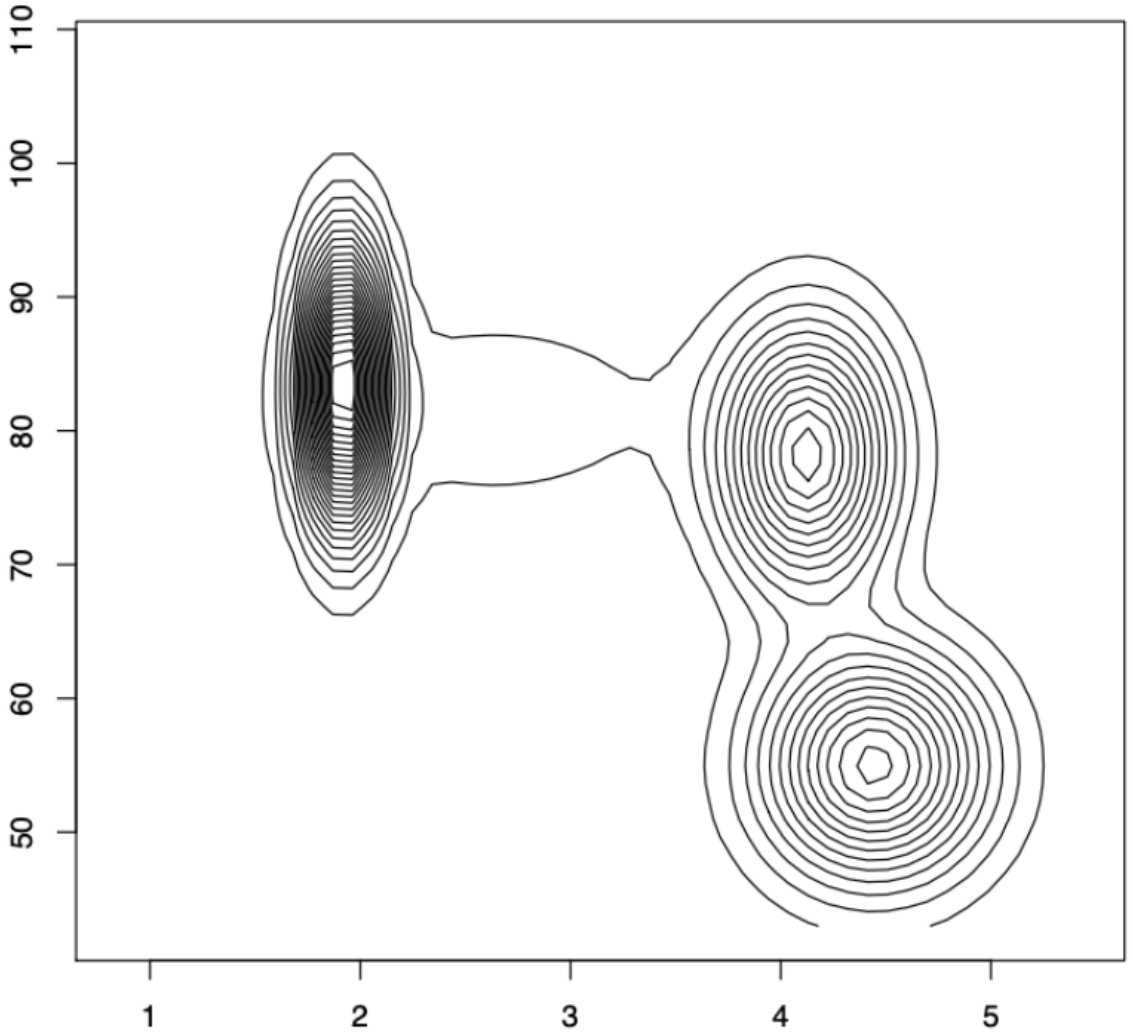
- We used `Mclust` to fit a mixture model to the Old Faithful data.
- The 4-component `VVI` model provided the best fit (although the 3-component model was also good). A `VVI` model is one where components have variable volume and shape but orientation in the direction of the main coordinate axes.



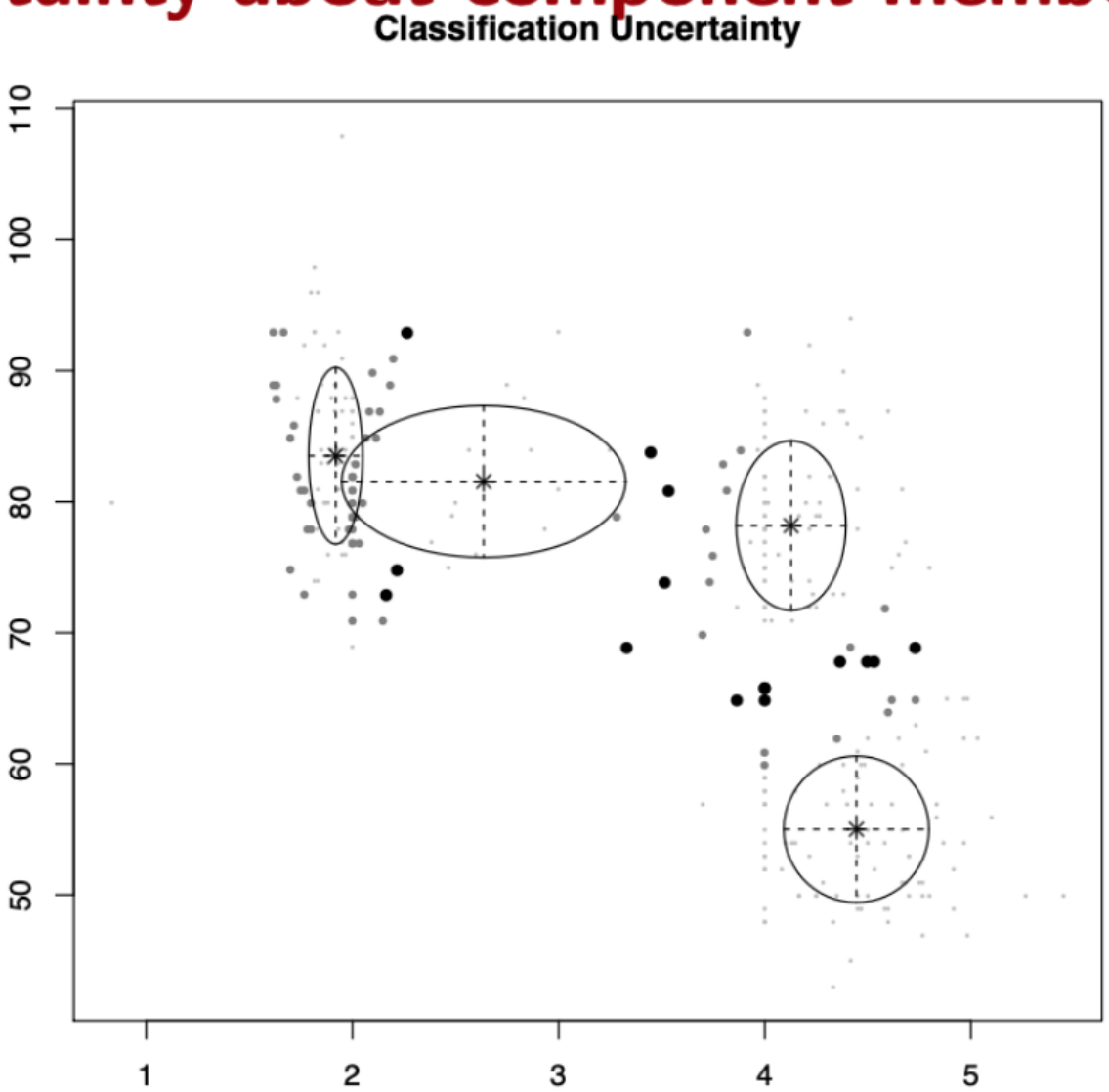
BIC values


BIC
−3000
−3500
−4000
−4500
2
4
6
8
number of components
EII
VII
EEI
VEI
EVI
VVI
EEE
EEV
VEV
VVV

Countours of estimated normal components



Uncertainty about component membership



[1] Bouveyron, C., Celeux, G., Murphy, T. B., Raftery, A. E.: Model-based Clustering and Classification for Data Science. Cambridge University Press (2019)

[2] Giordani, P., Ferraro, M.B., Martella, F.: datasetsICR: Datasets from the Book 'An Introduction to Clustering with R'. R package version 1.0 (2020) `https://CRAN.R-project.org/package=datasetsICR`

[3] Hartigan, J.A., Wong, M.A.: A k-means clustering algorithm. Applied Statistics 28, 100-108 (1979)

[4] Hennig, C.: What are the true clusters? Pattern Recognition Letters 64, 53-62 (2015)

[5] Kaufman, L., Rousseeuw, P.: Finding Groups in Data. Wiley, New York (1990)

[6] Hennig, C.: fpc: Flexible Procedures for Clustering. R package version 2.2-5 (2020) `https://CRAN.R-project.org/package=fpc`

[7] Melnykov, V., Maitra, R.: Finite mixture models and model-based clustering. Statistics Surveys 4, 80-116 (2010)

[8] Peel, D., McLachlan, G.: Finite Mixture Models. John Wiley and Sons, Inc., Wiley Series in Probability and Statistics (2000)

[9] Scrucca, L., Fop, M., Murphy, T.B., Raftery, A.E.: mclust 5: clustering, classiâcation and density estimation usingGaussian âfinite mixture models. R J., 8(1): 205–233 (2016)

[10] Ward, J.H., Hierarchical grouping to optimize an objective function. J. Am. Stat. Assoc., 58: 236–244 (1963)