Université de Strasbourg

Hierarchical clustering

Maja Temerinac-Ott
temerinacott@unistra.fr

ICUBE

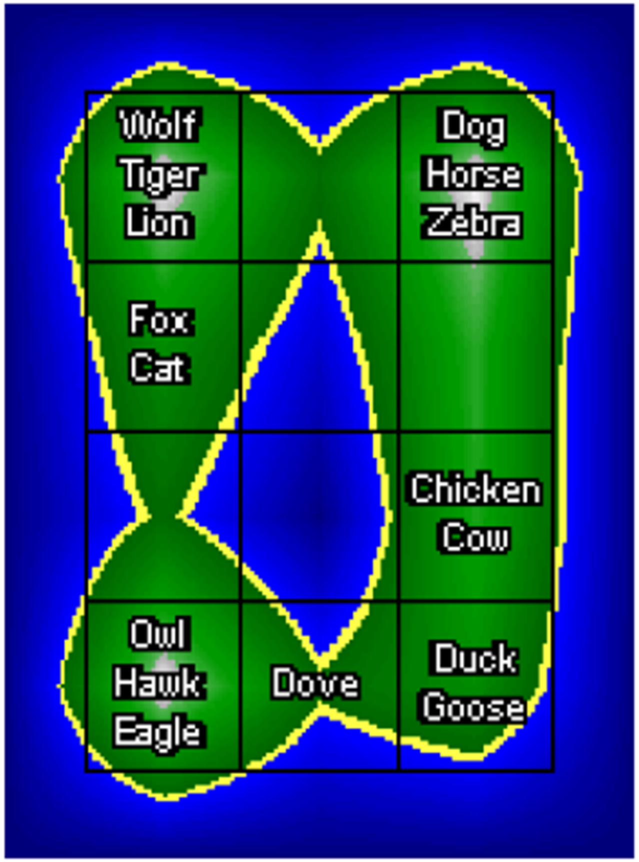
Slides adopted from Germain Forestier

Introduction

- **Agglomerative clustering**:
  - Classification hiérarchique ascendante (CHA)
- **Principle**: create, at each step, a partition obtained by aggregating the closest elements two by two.

- Elements:
  - Individuals or objects to be classified
  - Groups of individuals generated by the algorithm
- Each individual or cluster is gradually absorbed by the closest cluster

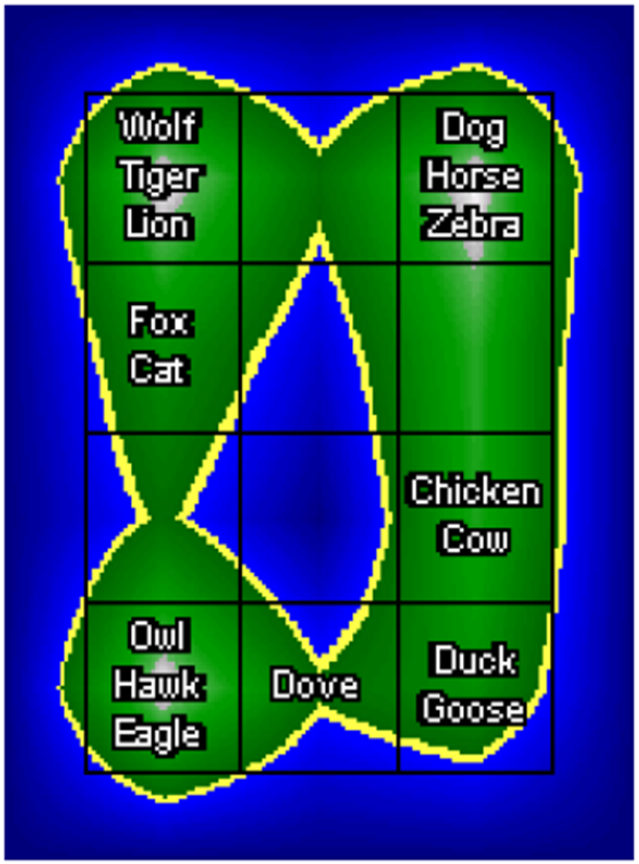
Clustering Example



Clustering Example

Université de Strasbourg

Birds/Mammals



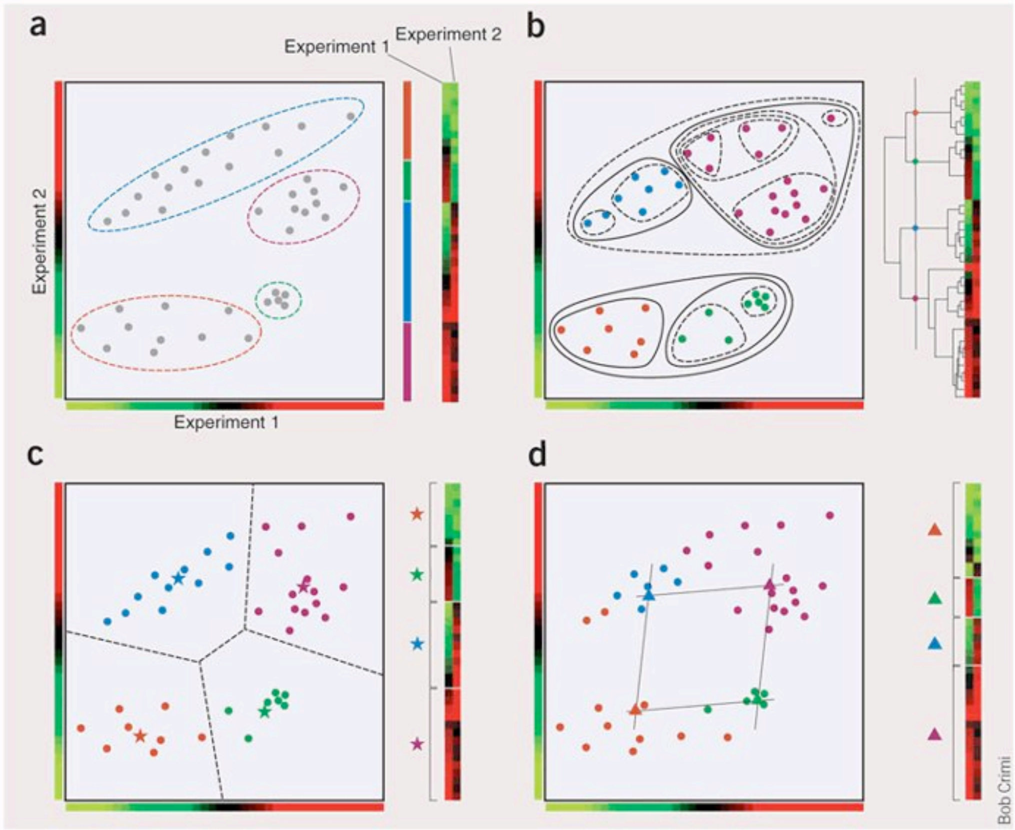
Predators/Non-Predators

User‘s Dilemma

- What is a cluster?
- Which features and normalization should be used?
- How to define pair-wise similarity?
- How many clusters?
- Which clustering method?
- Does the data have any clustering tendency?
- Are the discovered clusters & partition valid?

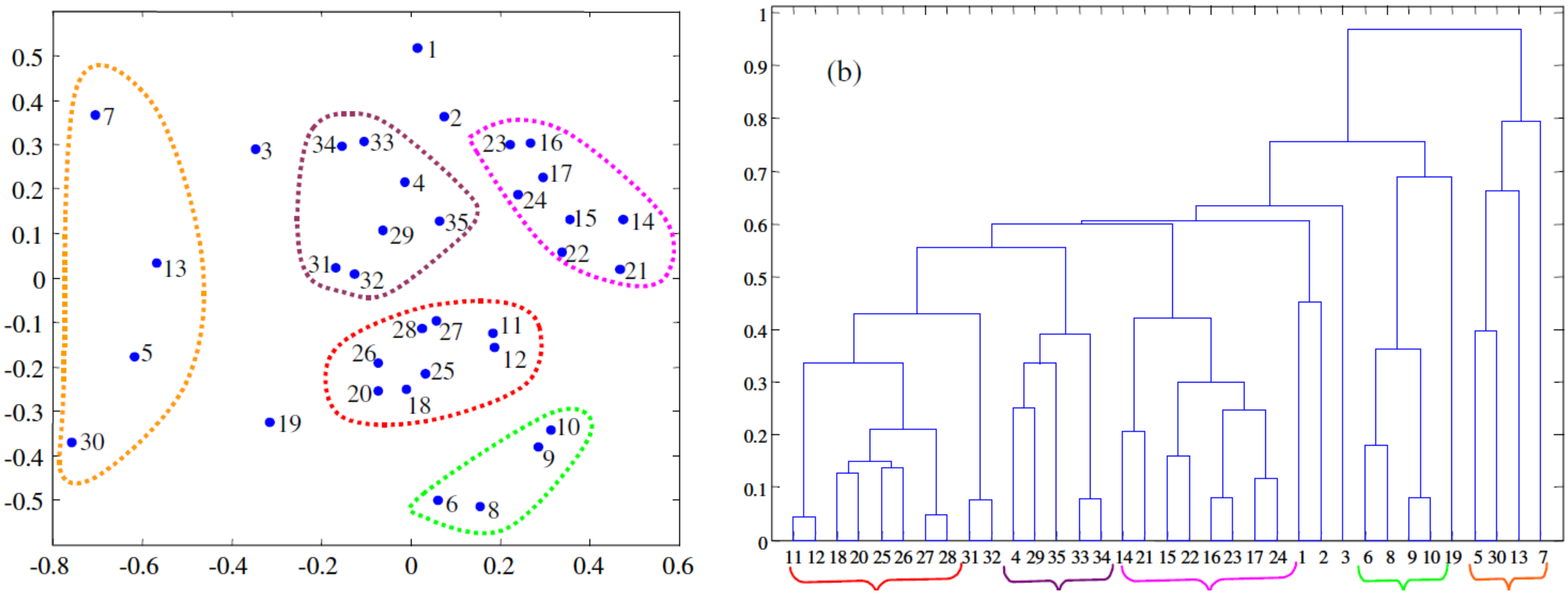
Clustering Example

- 40 Genes measured under two different conditions



http://www.nature.com/nbt/journal/v23/n12/full/nbt1205-1499.html

Hierarchical vs. Partitional Clustering



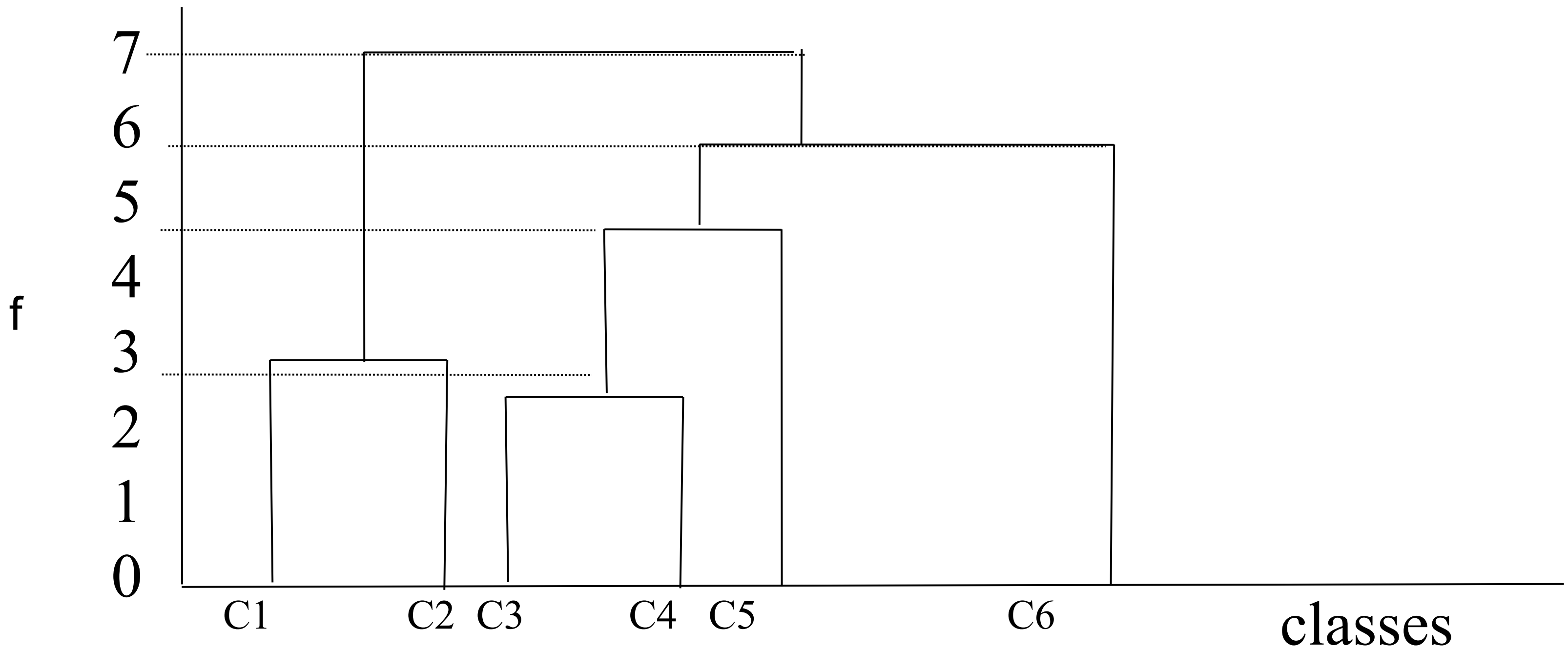
Agglomerative Clustering

Université de Strasbourg

- **Agglomerative clustering**:
  - Classification hiérarchique ascendante (CHA)
- **Result**: partition hierarchy, in the form of trees containing n - 1 partitions.
- **Defintion**: The set D of data, partitioned into K classes, is a hierarchy H iff:

$$
\begin{aligned}
&1.\ D \in H \\
&2.\ \forall x \in D,\ \{x\} \in H \\
&3.\ \forall h, h' \in H,\ h \cap h' = \varnothing \\
&\quad\ \ ou\ \ h \subset h'\ \ ou\ \ h' \subset h
\end{aligned}
$$

- **Agglomerative clustering**:

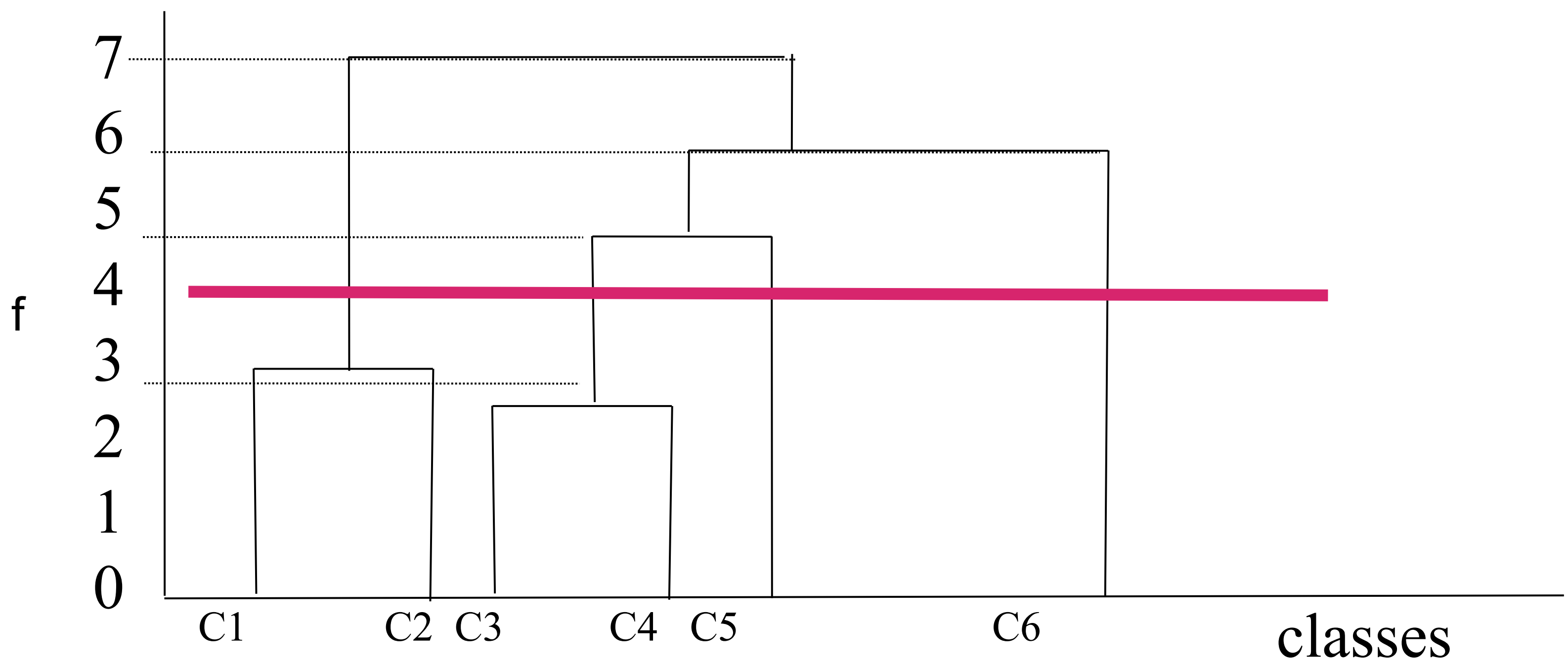


Agglomerative Clustering

- **Agglomerative clustering**:
  - these trees give an idea of the number of classes actually existing in the population.
  - By "cutting" the tree by a horizontal line, we obtain a partition
  - The closer the line is to the terminal elements the finer the partition is.

Agglomerative Clustering

- **Agglomerative clustering**:
  - *P ={{ C1, C2}, {C3,C4},{C5},{C6}}*



Agglomerative Clustering

- **Agglomerative clustering**:
  - these trees give an idea of the number of classes actually existing in the population.
  - By "cutting" the tree by a horizontal line, we obtain a partition
  - The closer the line is to the terminal elements the finer the partition is.
  - A hierarchy therefore makes it possible to provide a chain of n partitions having from 1 to n classes.

Agglomerative Clustering

- **Agglomerative clustering**:
  - At the beginning all the individuals have a distance: how is it defined?
    - The distance between an individual and a group
    - The distance between two groups
  - Define a strategy of grouping the elements: calculating the distances between disjoint groups of individuals
    - Aggregation criterion

Agglomerative Clustering

- **Agglomerative clustering**:
  - Example:
    - **Single linkage**: define the distance from H to y by the **smallest** distance of the elements from H to y:

$$d(H,y) = min\{d(x_i, y)\}\ x_i \in H$$

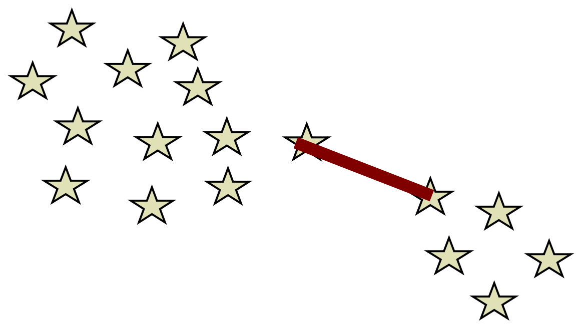
    - and the distance between two clusters H_1 and H_2 can be defined by:

$$d(H_1, H_2) = \min\{ d(x_i, y_j) \}\ \text{où}\ x_i \in H_1,\ y_j \in H_2$$

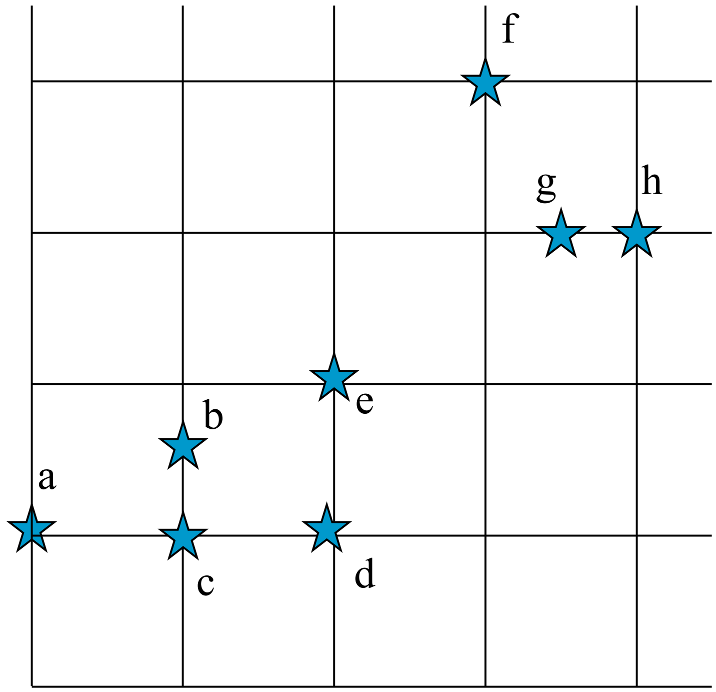
Agglomerative Clustering

- **Agglomerative clustering**:
  - Example:
    - minimum link index (nearest neighbor)

$$D(h_1,h_2) = \min_{x_i \in h_1, x_j \in h_2} d(x_i, x_j)$$

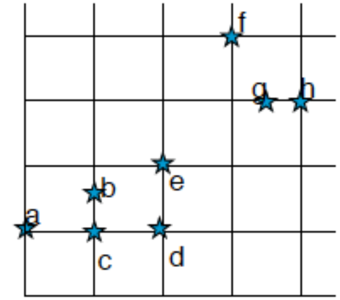


- **Agglomerative clustering**:
  - Example: Single Linkage



Agglomerative Clustering

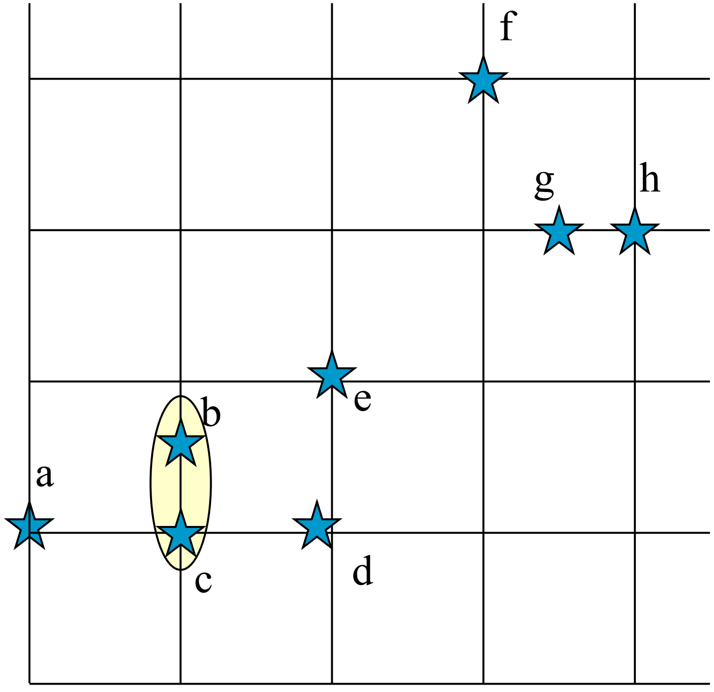
- Example: **Single Linkage**
  - Distance matrix

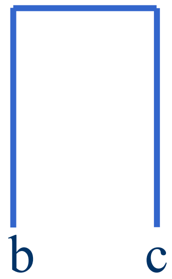


	a	b	c	d	e	f	g	h
a	0	1.118	1	2	2.23	4.24	4.03	4.47
b	1.118	0	0.707	1.118	1.118	3.201	2.91	3.35
c	1	0.707	0	1	1.414	3.605	3.201	3.605
d	2	1.118	1.118	0	1	3.162	2.121	2.828
e	2.23	1.118	1.414	1	0	2.236	1.802	2.236
f	4.24	3.20	3.605	3.162	2.236	0	1.118	1.414
g	4.03	2.91	3.201	2.121	1.802	1.118	0	0.707
h	4.47	3.35	3.605	2.828	2.236	1.414	0.707	0

Agglomerative Clustering

- Example: **Single Linkage**
  - Dendogram

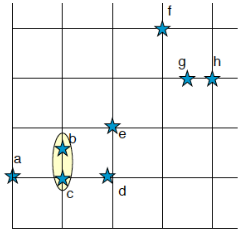




Agglomerative Clustering

Université de Strasbourg

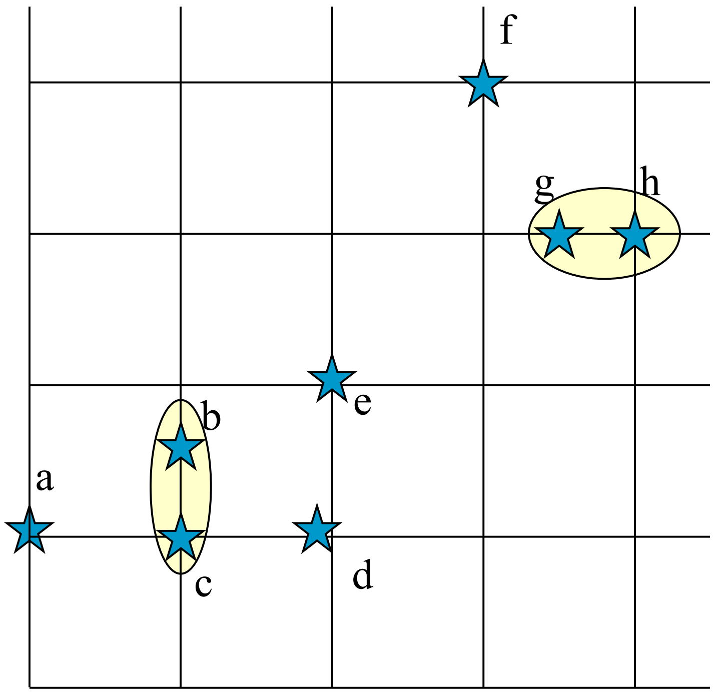
- Example: **Single Linkage**
  - Distance matrix



	a	{b,c}	d	e	f	g	h
a	0	1	2	2.23	4.24	4.03	4.47
{b,c}		0	1	1.118	3.201	2.91	3.35
d			0	1	3.162	2.121	2.828
e				0	2.236	1.802	2.236
f					0	1.118	1.414
g						0	$\boxed{0.707}$
h							0

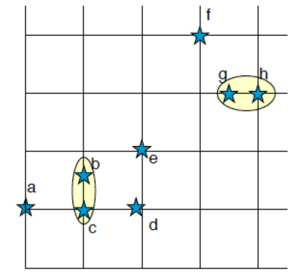
Agglomerative Clustering

- Example: **Single Linkage**
  - Dendogram



Agglomerative Clustering

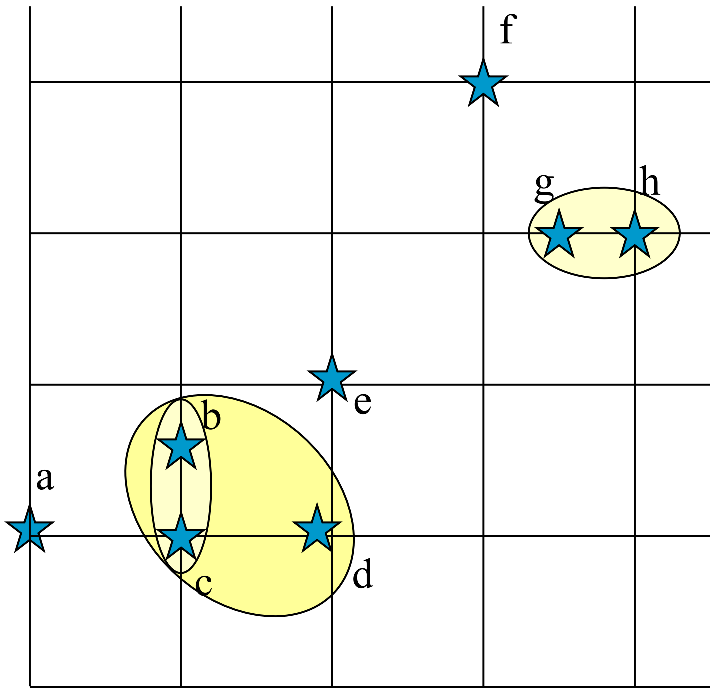
- Example: **Single Linkage**
  - Distance matrix

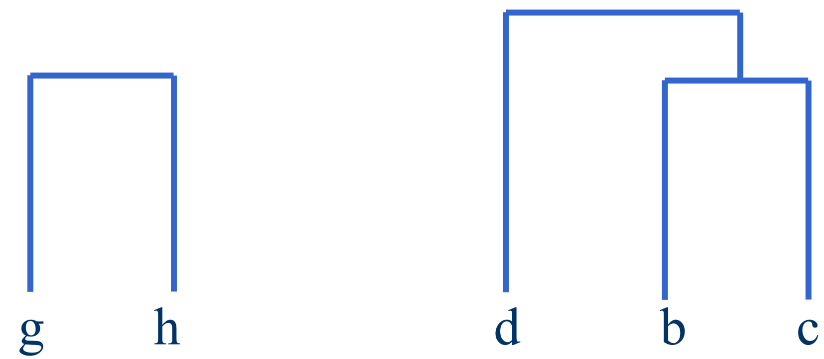


	a	{b,c}	d	e	f	{g,h}
a	0	1	2	2.23	4.24	4.03
{b,c}		0	1	1.118	3.201	2.91
d			0	1	3.162	2.121
e				0	2.236	1.802
f					0	1.118
{g,h}						0

Agglomerative Clustering

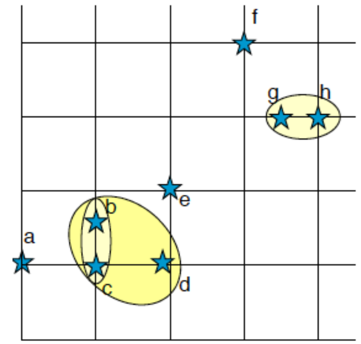
- Example: **Single Linkage**
  - Dendogram





Agglomerative Clustering

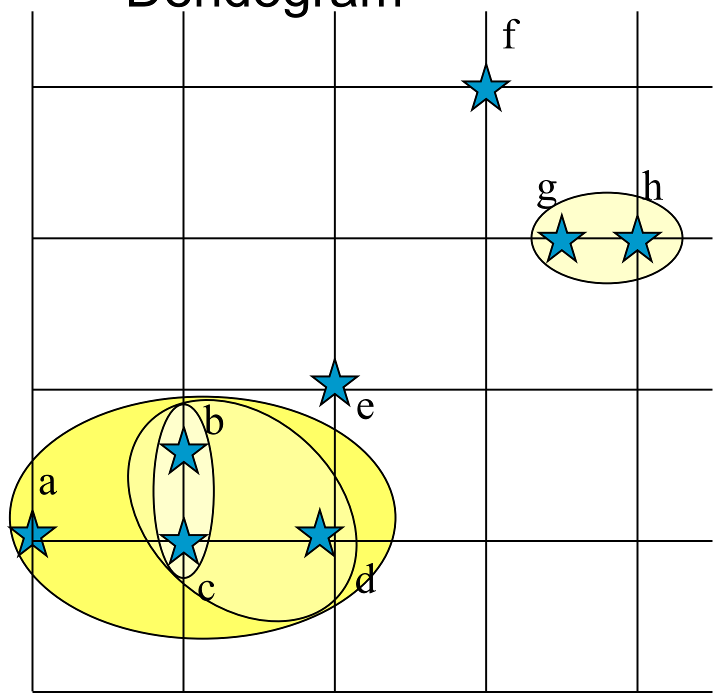
- Example: **Single Linkage**
  - Distance matrix

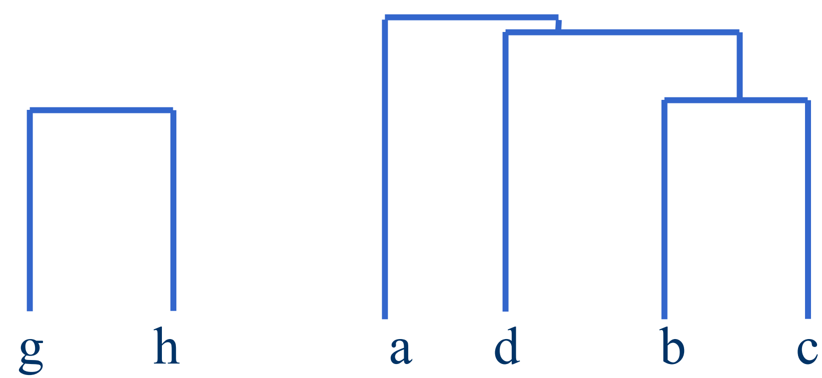


	a	{{b,c},d}	e	f	{g,h}
a	0	1	2.23	4.24	4.03
{{b,c},d}		0	1.118	3.201	2.121
e			0	2.236	1.802
f				0	1.118
{g,h}					0

Agglomerative Clustering

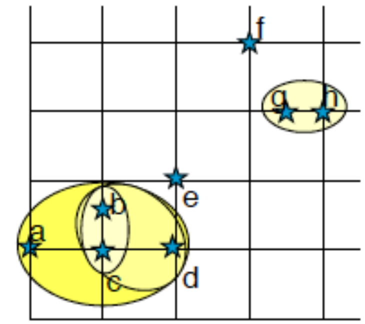
- Example: **Single Linkage**
  - Dendogram





Agglomerative Clustering

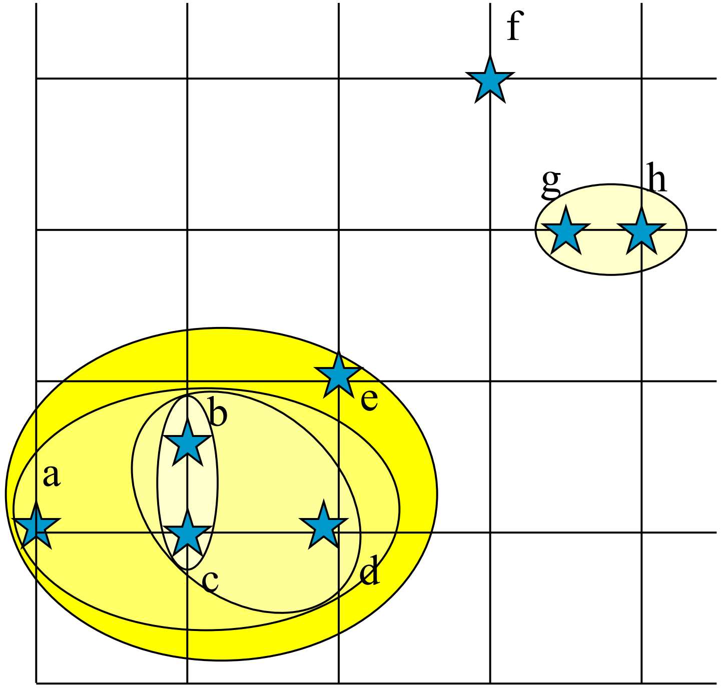
- Example: **Single Linkage**
  - Distance matrix

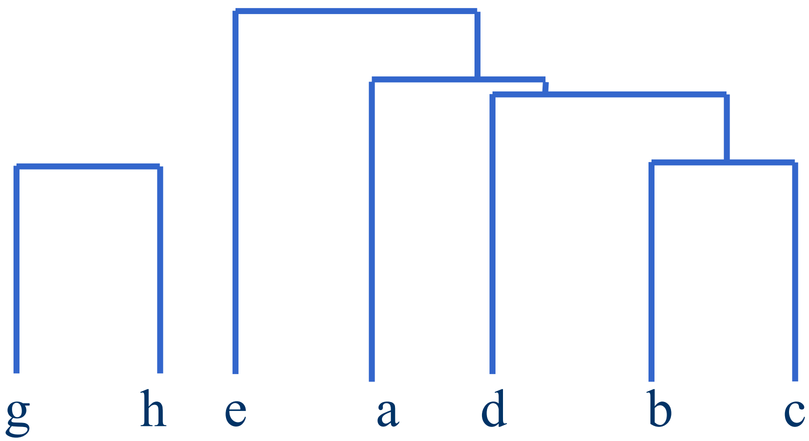


	{a,{{b,c},d}}	e	f	{g,h}
{a,{{b,c},d}}	0	1.118	3.201	2.121
e		0	2.236	1.802
f			0	1.118
{g,h}				0

Agglomerative Clustering

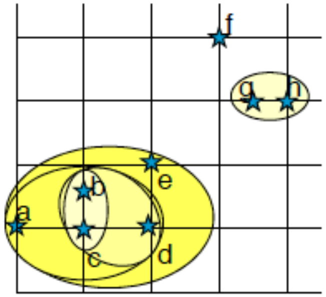
- Example: **Single Linkage**
  - Dendogram





Agglomerative Clustering

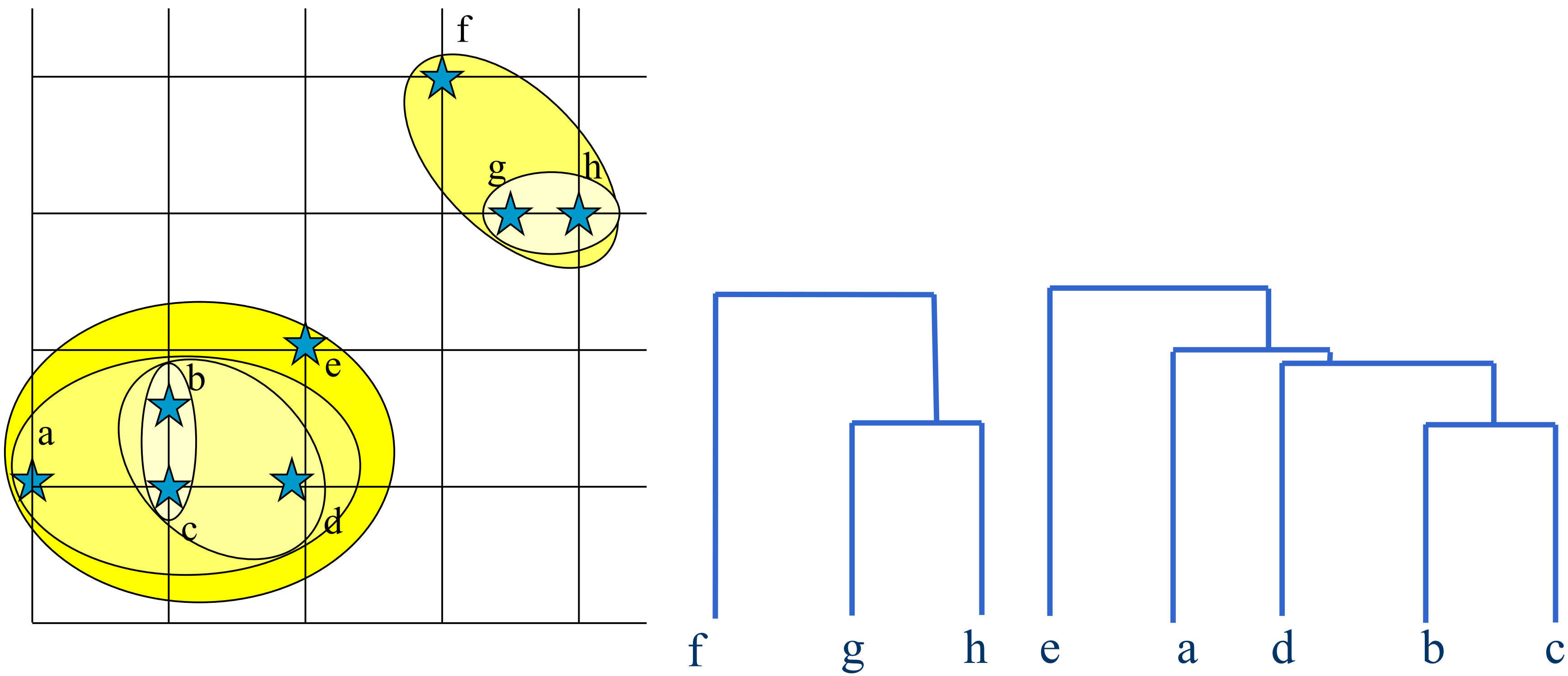
- Example: **Single Linkage**
  - Distance matrix



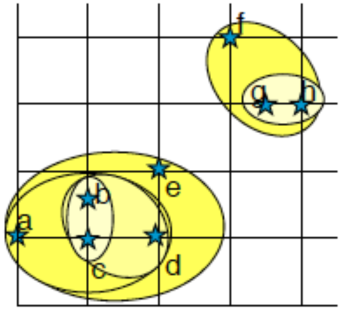
	{{a,{{b,c},d}},e}	f	{g,h}
{{a,{{b,c},d},e}	0	3.201	2.121
f		0	1.118
{g,h}			0

Agglomerative Clustering

- Example: **Single Linkage**
  - Dendogram



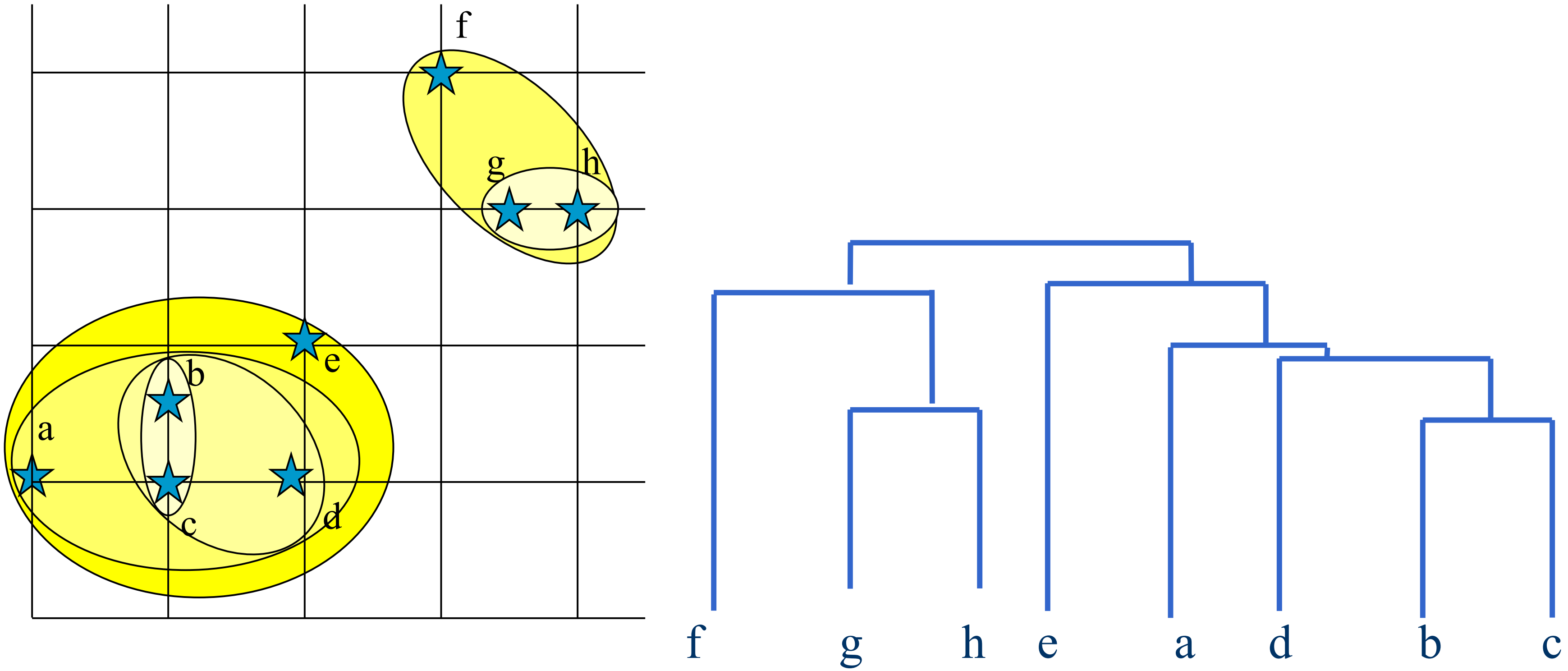
- Example: **Single Linkage**
  - Distance matrix



	{{a,{{b,c},d}},e}	{f,{g,h}}
{{a,{{b,c},d},e}	0	1.802
{f,{g,h}}		0

Agglomerative Clustering

- Example: **Single Linkage**
  - Dendogram



- **Agglomerative clustering**:
  - Example:
    - **Complete linkage**: define the distance from H to y by the **bigest** distance of the elements from H to y:

$$d(H_1, H_2) = \max \{ d(x_i, y_j) \} \text{ où } x_i \in H_1, y_j \in H_2$$

    - **Attention**: We still link the classes with the smallest distance!!

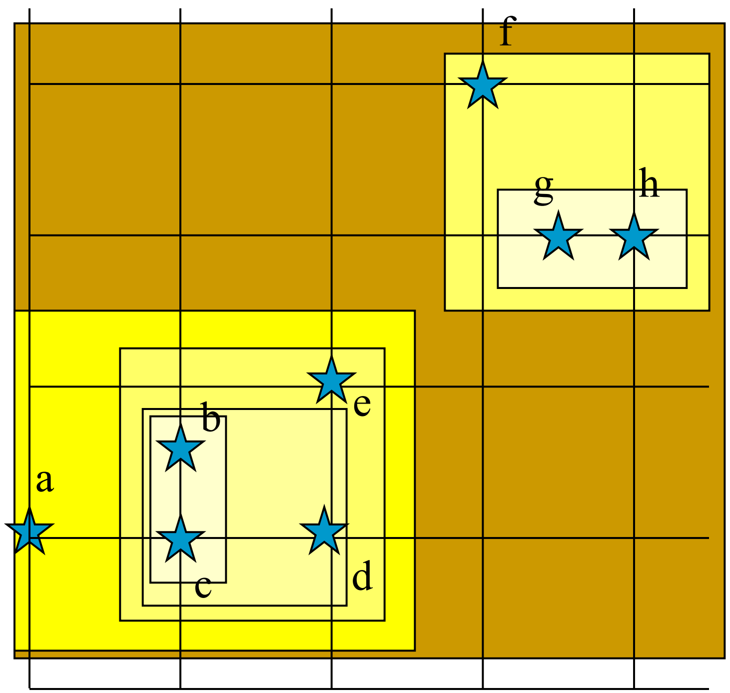
Agglomerative Clustering

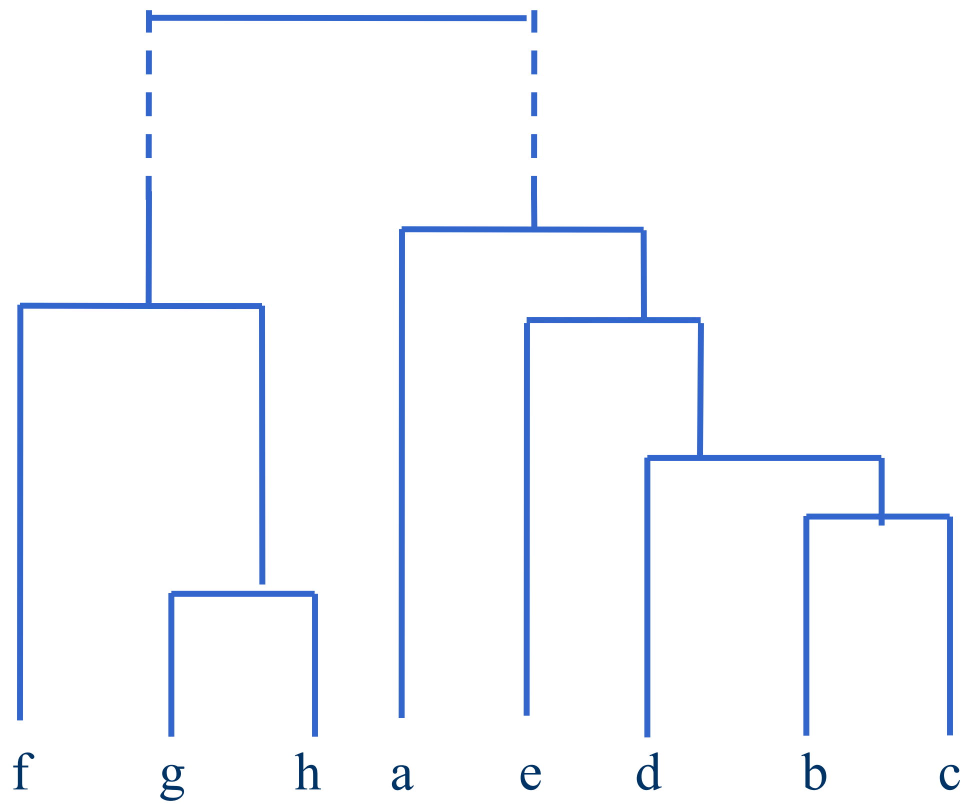
- **Agglomerative clustering**:
  - Example:
    - maximum link index (maximum diameter)

$$D(h_1, h_2) = \max_{x_i \in h_1, x_j \in h_2} d(x_i, x_j)$$

Agglomerative Clustering

- Example: Complete Linkage

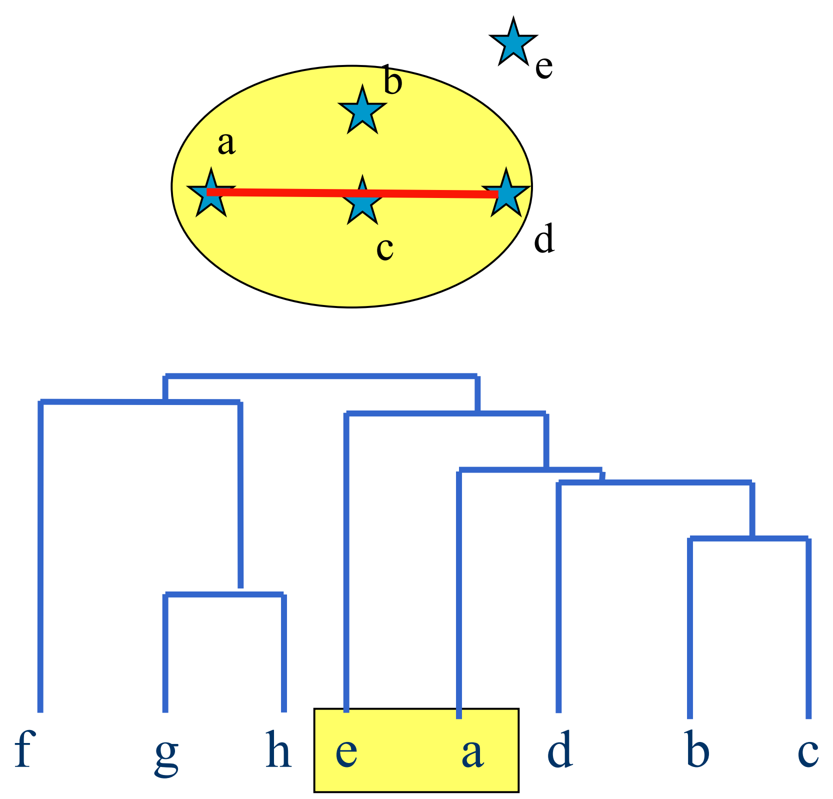


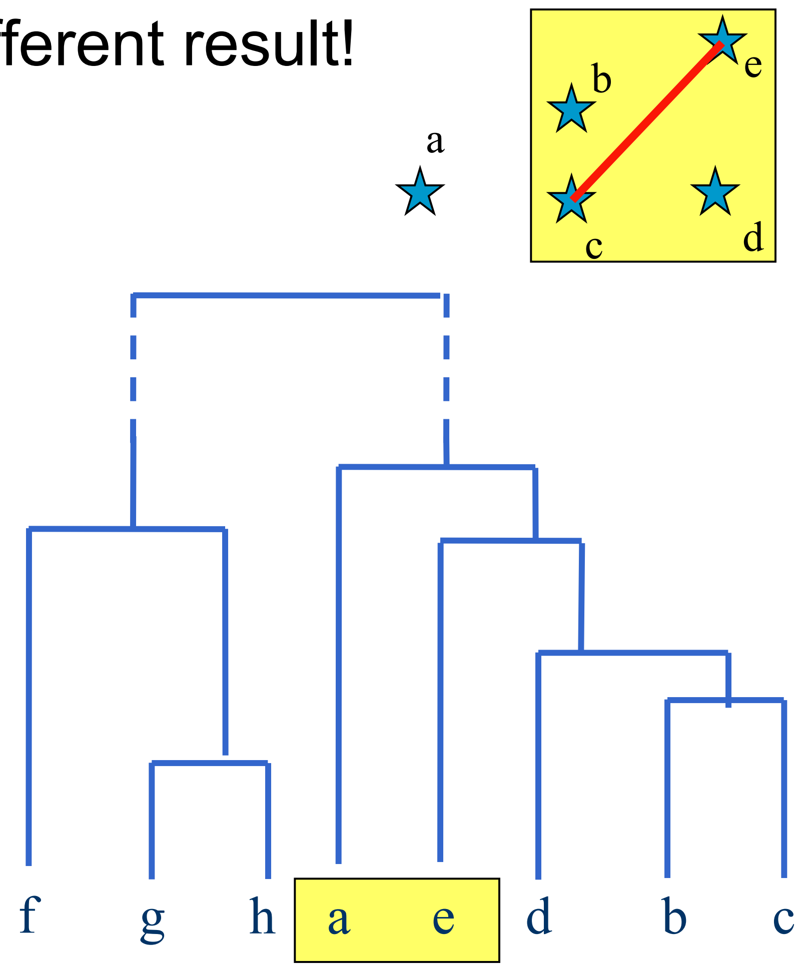


d({e},{ b,c,d}) = d(e,c) = 1,4
< d({a},{ b,c,d}) = d(a,d) = 2

Agglomerative Clustering

single vs complete linkage: different result!



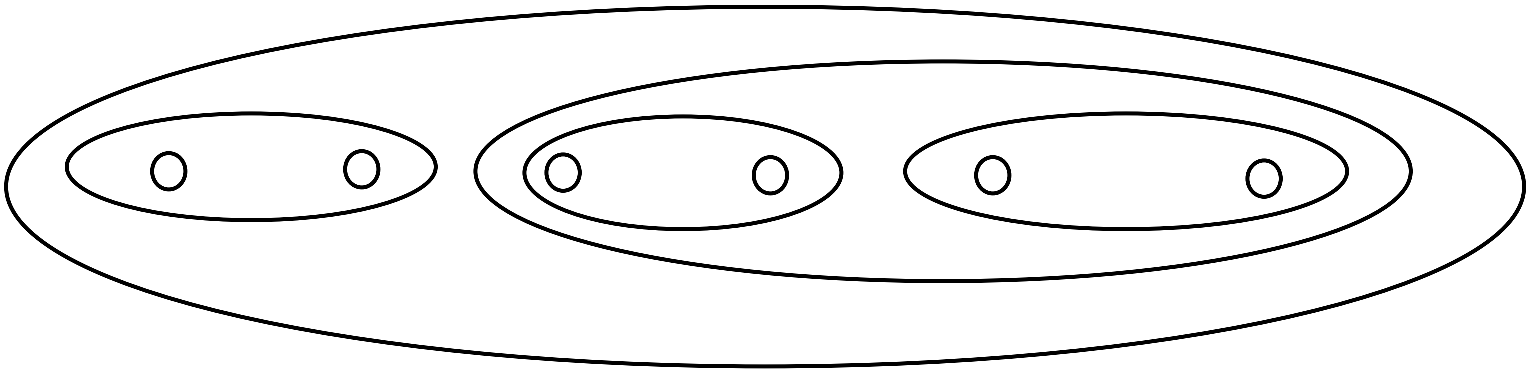


Agglomerative Clustering

- Chain problem in the classification!

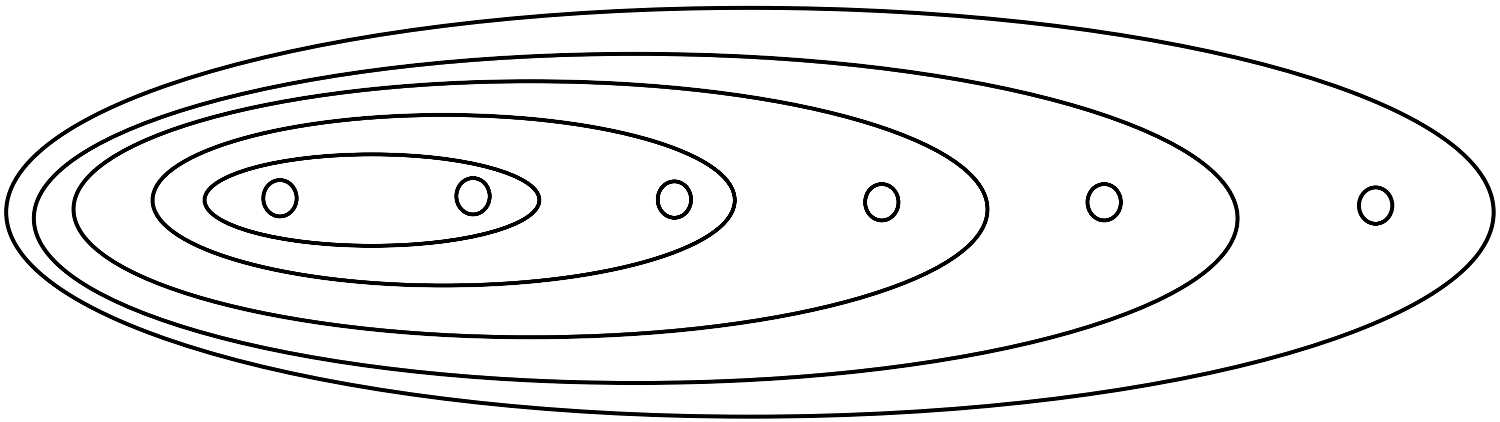
Agglomerative Clustering

- Chain problem in the classification!



Agglomerative Clustering

- Chain problem in the classification!



Agglomerative Clustering

- Chain problem in the classification!
  - Consider the variance of the classes: Ward index
    - Updating:
      - After merging h1 and h2: the distance is calculated with another block

$$\delta_1(h_1 \cup h_2, h) = \frac{|h| + |h_1|}{|h| + |h_1| + |h_2|}\delta_1(h_1, h)$$

$$+\frac{|h| + |h_2|}{|h| + |h_1| + |h_2|}\delta_1(h_2, h) + \frac{|h_1| + |h_2|}{|h| + |h_1| + |h_2|}\delta_1(h_1, h_2)$$

      - it can be shown that at each step the new partition is the one that limits the increase in intra-class inertia

Agglomerative Clustering

- Distance measure:
  - centers of gravity (average distance)

$$D(h_1, h_2) = d(g_1, g_2)$$

Agglomerative Clustering

- Distance measure:
  - Aggreagation of variation of inertia

$$\delta_1(h_1, h_2) = \frac{p(h_1).p(h_2)}{p(h_1) + p(h_2)} d(g_1, g_2)$$

  - Compute the likelihood of the link

$$\delta_1(h_1, h_2) = -\log\left(-\log\left(\left[d(h_1, h_2)\right]^{(n_1, n_2)^{\varepsilon}}\right)\right)$$

    - where d (h1, h2) is the index of the simple link
    - facilitates the fusion of low variance classes

Ultrametric Space

- In mathematics, an **ultrametric space** is a special kind of metric space in which the triangle inequality is replaced
- Definition: $d : M \times M \rightarrow R^+$
- It has the following properties:

$$\forall (x,y) \in M^2 , d(x,y) = 0 \leftrightarrow x = y$$

$$\forall (x,y) \in M^2 , d(x,y) = d(y, x)$$

and

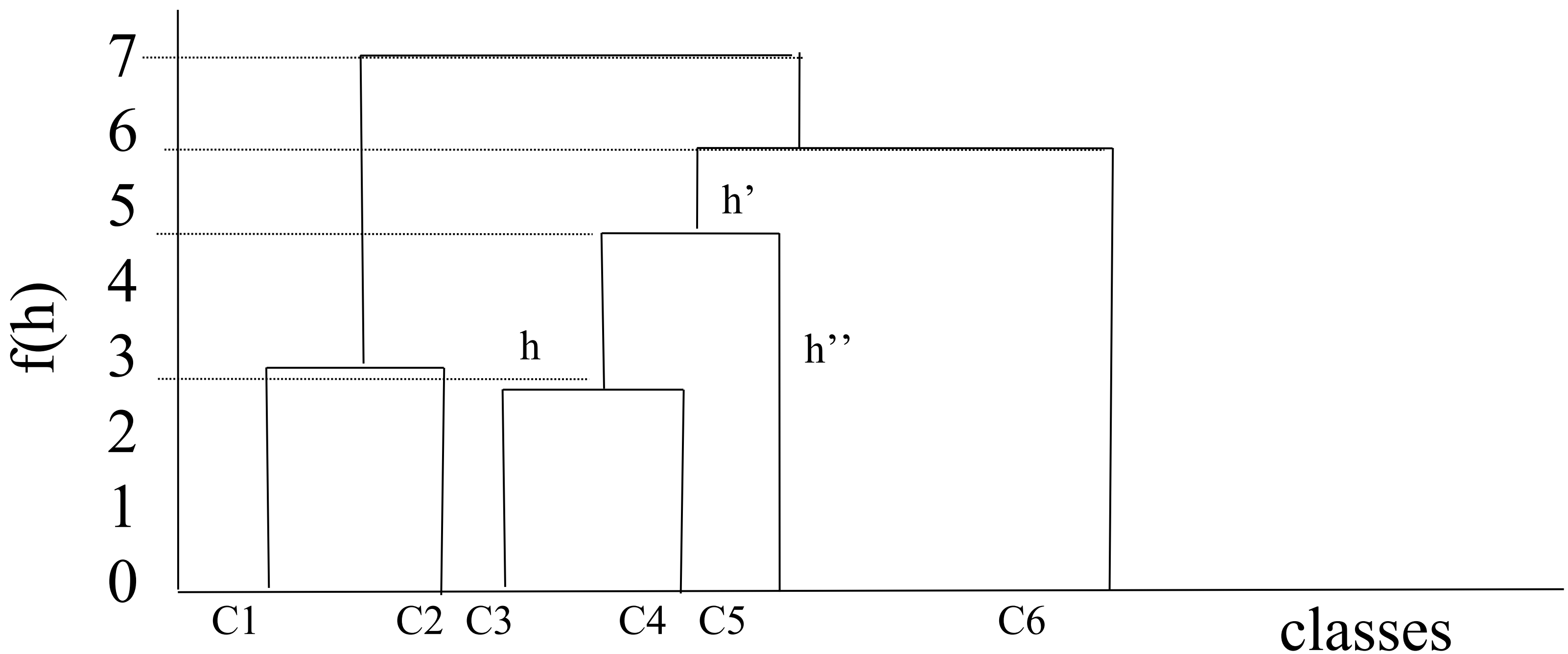
$$\forall (x,y,z) \in M^3 , d(x,z) \leq \max \{d(x,y), d(y,z)\}$$

Ultrametric Space

- A hierearchy is indexed if there is f such that

$$\forall\, x \in H,\ f(\{x\}) = 0$$

$$\forall\, h, h' \in H,\ h \neq h',\ h' \subset h \Rightarrow f(h') < f(h)$$



Ultrametric Space

- It can be shown:
  - any indexed hierarchy makes it possible to define an ultrametric
  - any ultrametric allows to define an indexed hierarchy

Ultrametric Space

{b};{c} → 0.707 / {h},{g} → 0.707{b,c}.{d} → 1

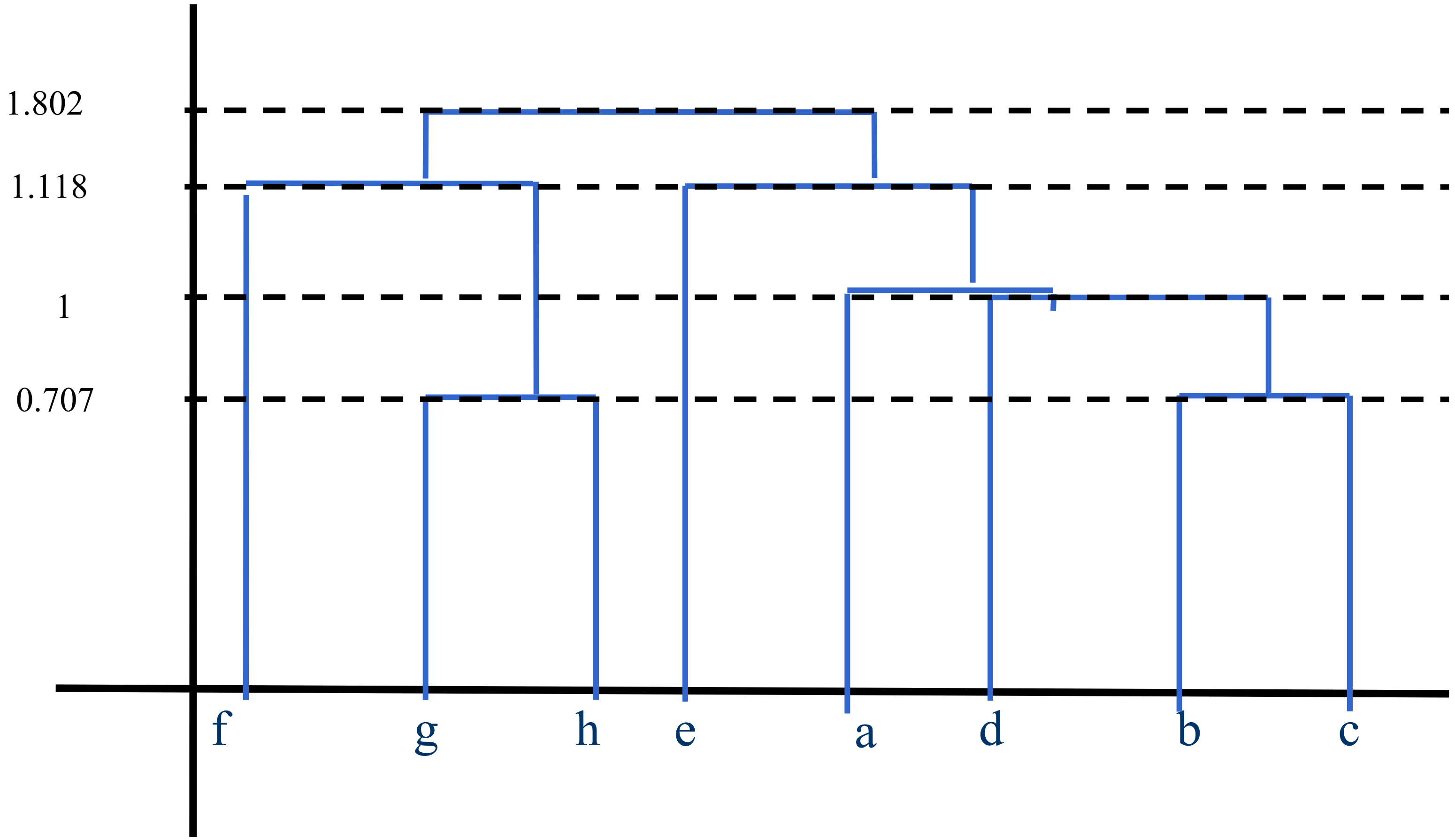
{{b,c},{d}}.{a} → 1 {{a},{b,c},{d}}.{e} → 1.118

{g,h}.{f} → 1.118 {{a},{b,c},{d}.{e}}{{f},{g,h}} → 1.802

	a	b	c	d	e	f	g	h
a	0	1	1	1	1.118	1.802	1.802	1.802
b	1	0	0.707	1	1.118	1.802	1.802	1.802
c	1	0.707	0	1	1.118	1.802	1.802	1.802
d	1	1	1	0	1.118	1.802	1.802	1.802
e	1.118	1.118	1.118	1.118	0	1.802	1.802	1.802
f	1.802	1.802	1.802	1.802	1.802	0	1.118	1.118
G	1.802	1.802	1.802	1.802	1.802	1.118	0	0.707
H	1.802	1.802	1.802	1.802	1.802	1.118	0.707	0

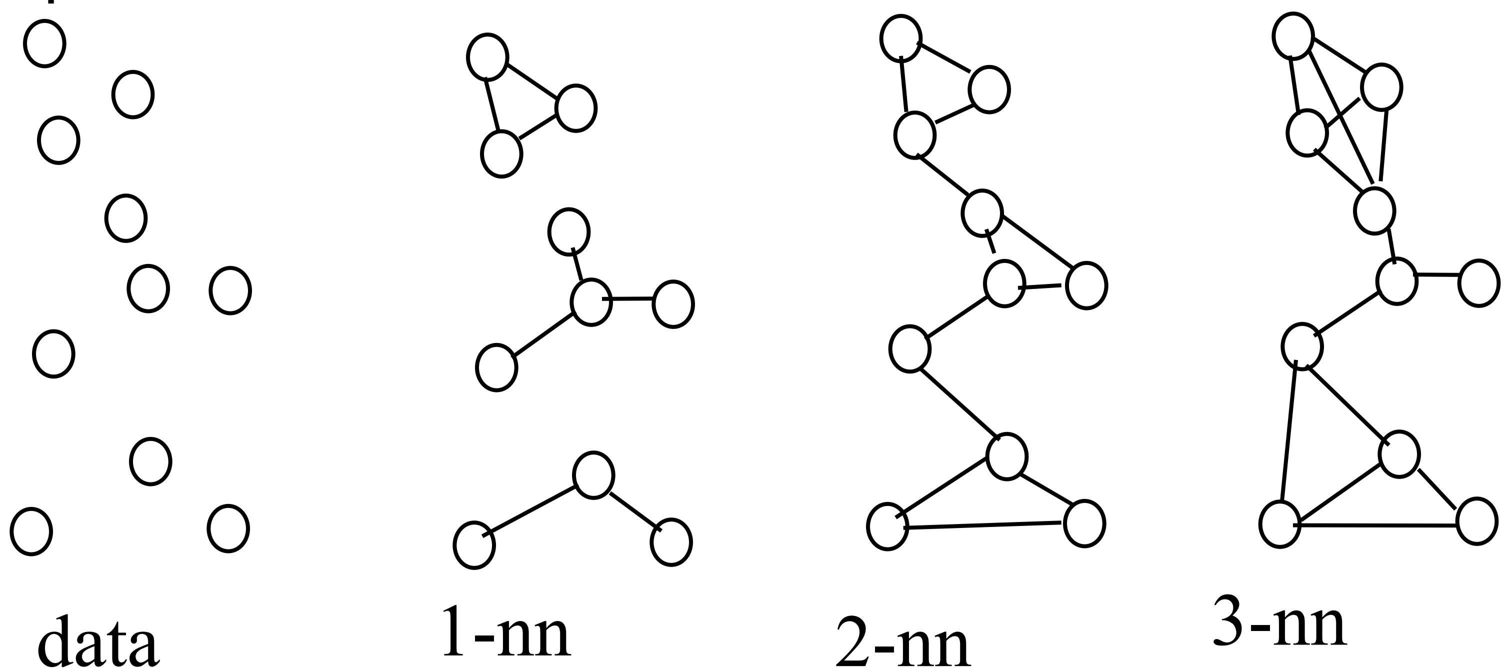
Ultrametric Space

- Minimum likelihood clustering



Chameleon algorithm

- Combines initial partition of data with hierarchical clustering techniques it modifies clusters dynamically
- **Principle**: estimate the intra-cluster and extra-cluster density from the k nearest neighbors (k-nn) graph



Chameleon algorithm

- Step 1:
  - Find the initial clusters
    - By partitioning the graph **k-nn** into **m** "solid" partitions (where the distance between points is minimized)
- Step 2:
  - Dynamically merge sub-clusters
    - Depending on two subcriteria:
      1. RI(C,C’): relative inter-connectivity
      2. RC(C,C’) : relative proximity

Chameleon algorithm

Université de Strasbourg

$$RC(C,C') = \frac{(|C|+|C'|)DC(C,C')}{|C|DC(C)+|C'|DC(C')}$$

$$RI(C,C') = \frac{2\times|EC(C,C')|}{|EC(C)|\times|EC(C')|}$$

EC(C,C’): set of edges that connect C and C’ (absolute interconnectivity between 2 clusters)

EC(C): smallest set of arrays that partition C into 2 clusters of proximally same size (internal interconnectivity)

DC(C,C’): average distance between points of C and C’

DC(C): average distance between points in C

Chameleon algorithm

- Problems:
  - Costs are too high to compute all distances $O(n^3)$
  - Binary tree:
    - We need to find k
    - Verify that the indices between every pair of k classes are minimal

Descending hierarchical classification

- **Principle**: by division of classes, we build a sequence of nested partitions whose classes form the hierarchy H sought

- Problems:
  - How to select which class to divide?
  - How many subclasses are there?

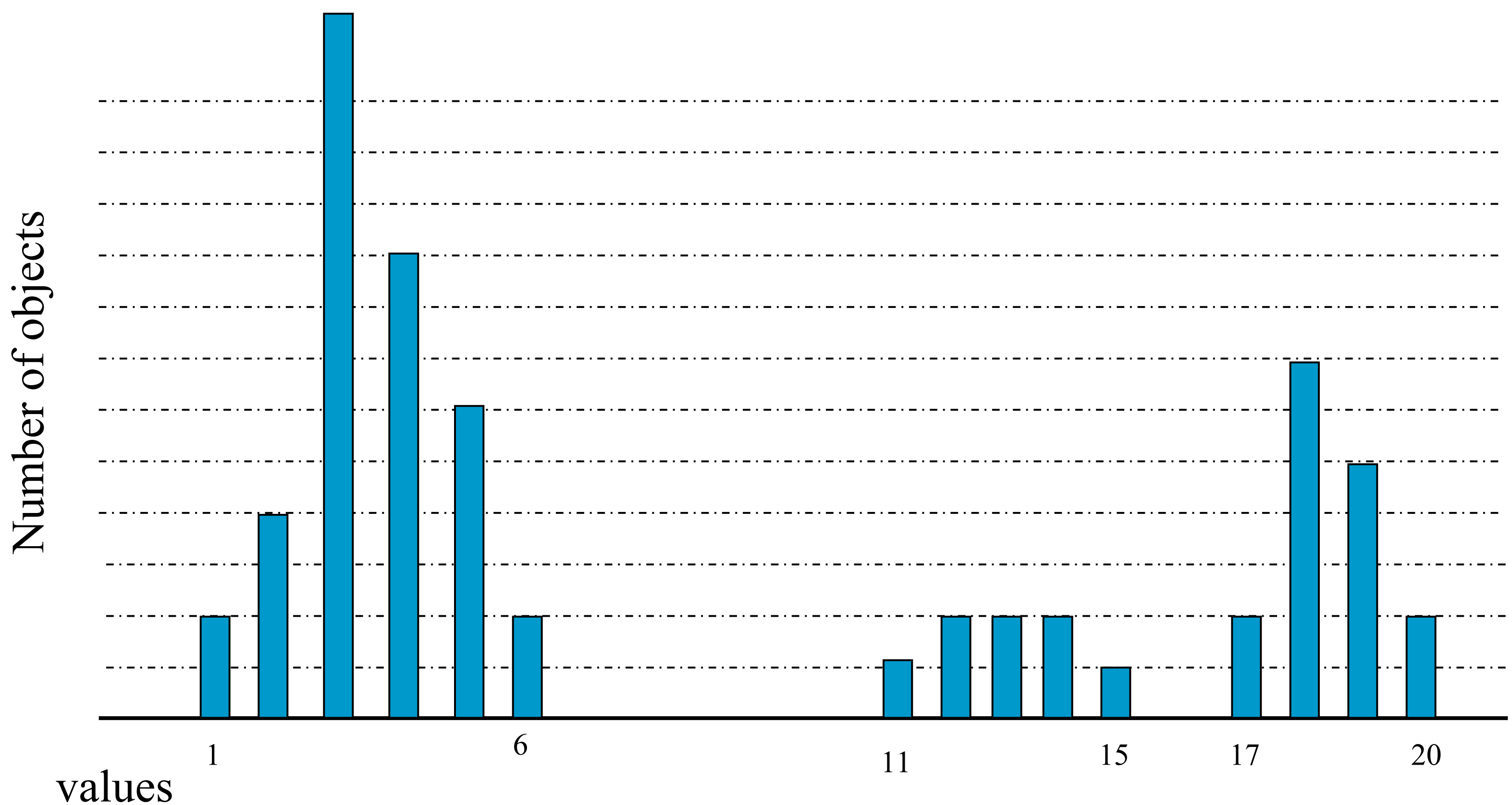
Descending hierarchical classification

- How to select which class to divide?
  - at each level, all classes that can be developed according to a certain criterion
  - only the class with the strongest criterion is developed

- Possible criteria
  - minimum variance
  - Number of objects
  - study of the histograms of values taken from the data

Descending hierarchical classification

- Using the distribution of the vlaues

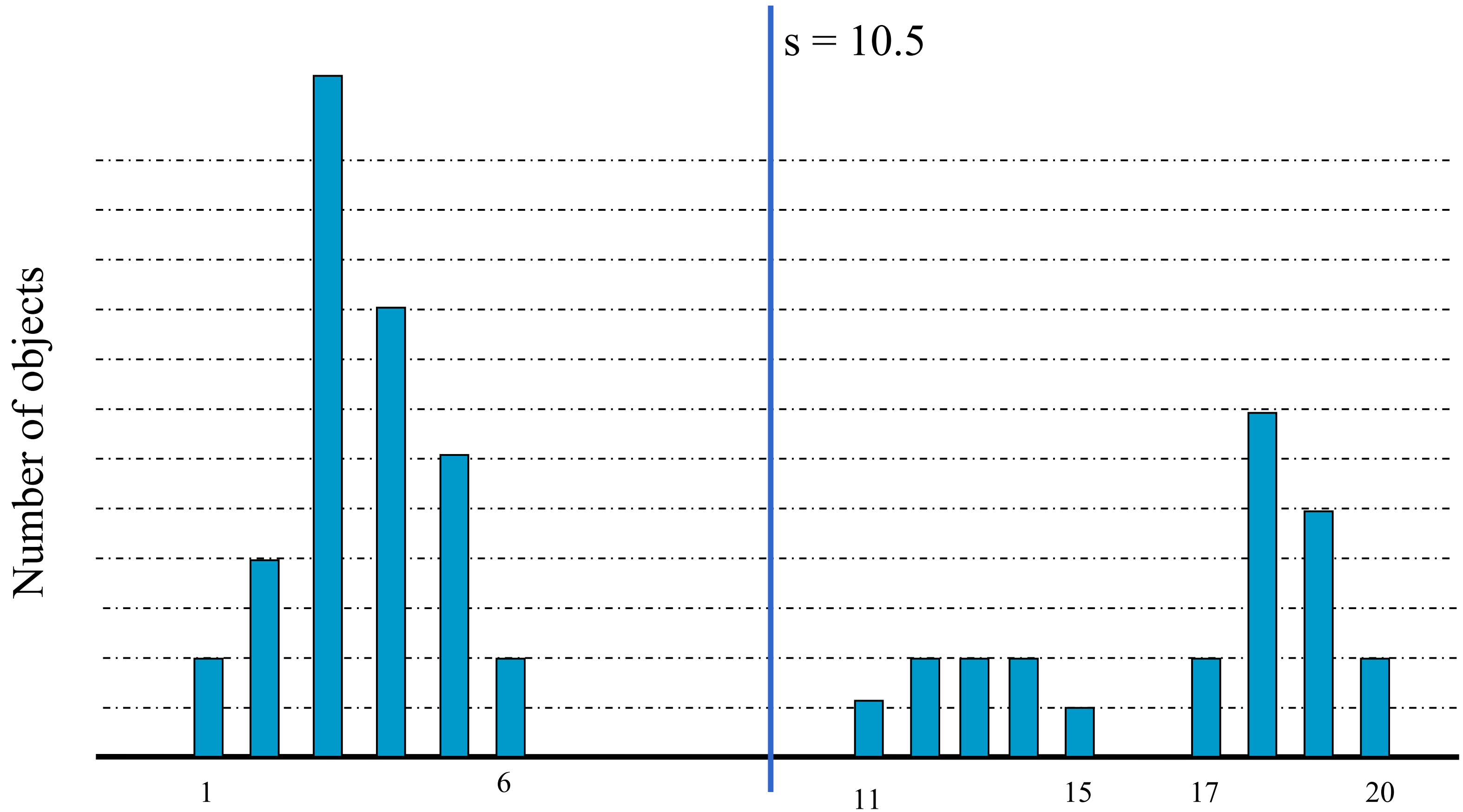


Descending hierarchical classification

- **Principle**: for each attribute
  - find a threshold separating the histogram into two "subclasses »
  - separate these two subclasses if necessary
  - iterate

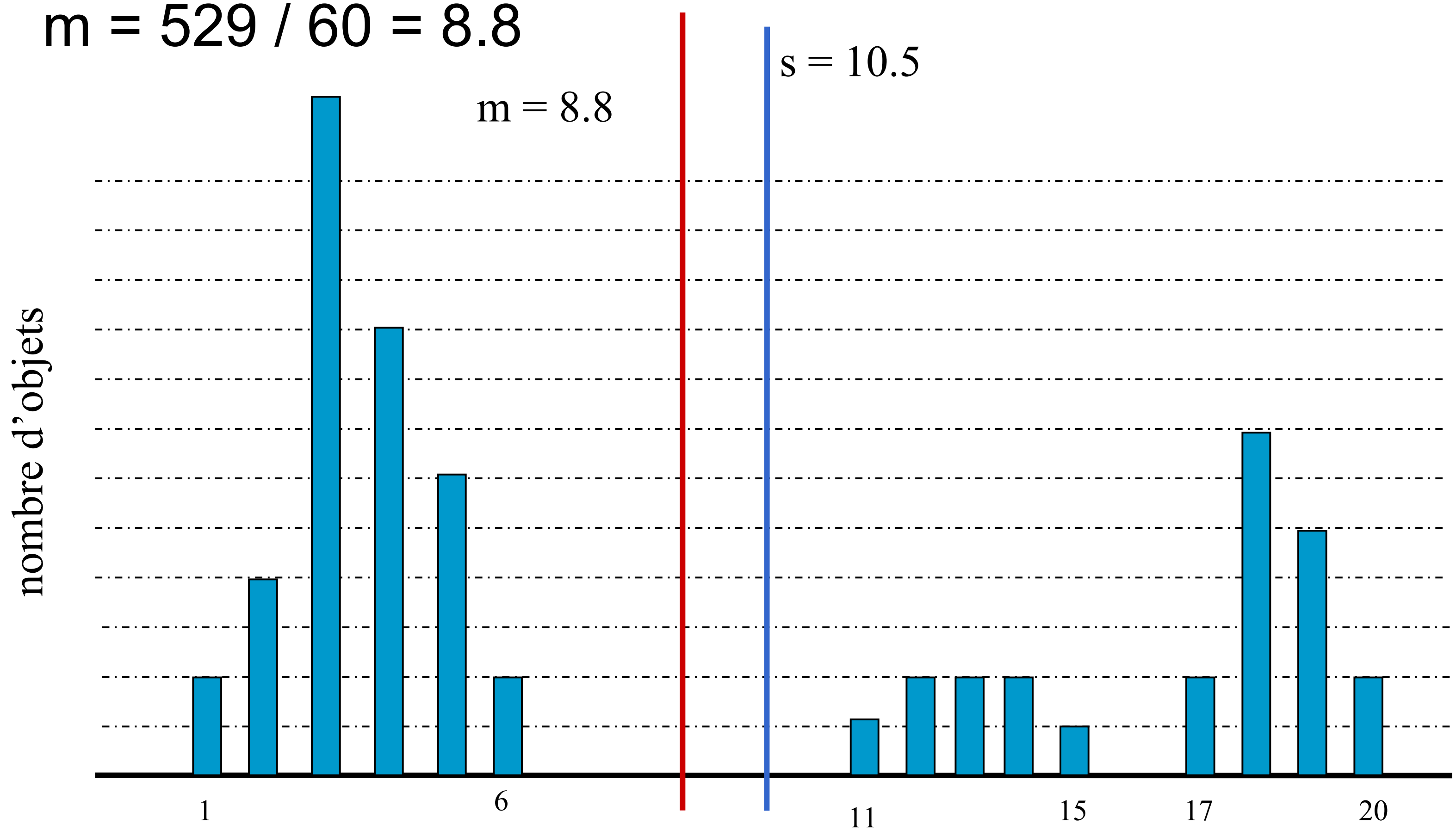
Descending hierarchical classification

- Calculation of a theoretical threshold:
  - s = (max + min)/2 = 10,5



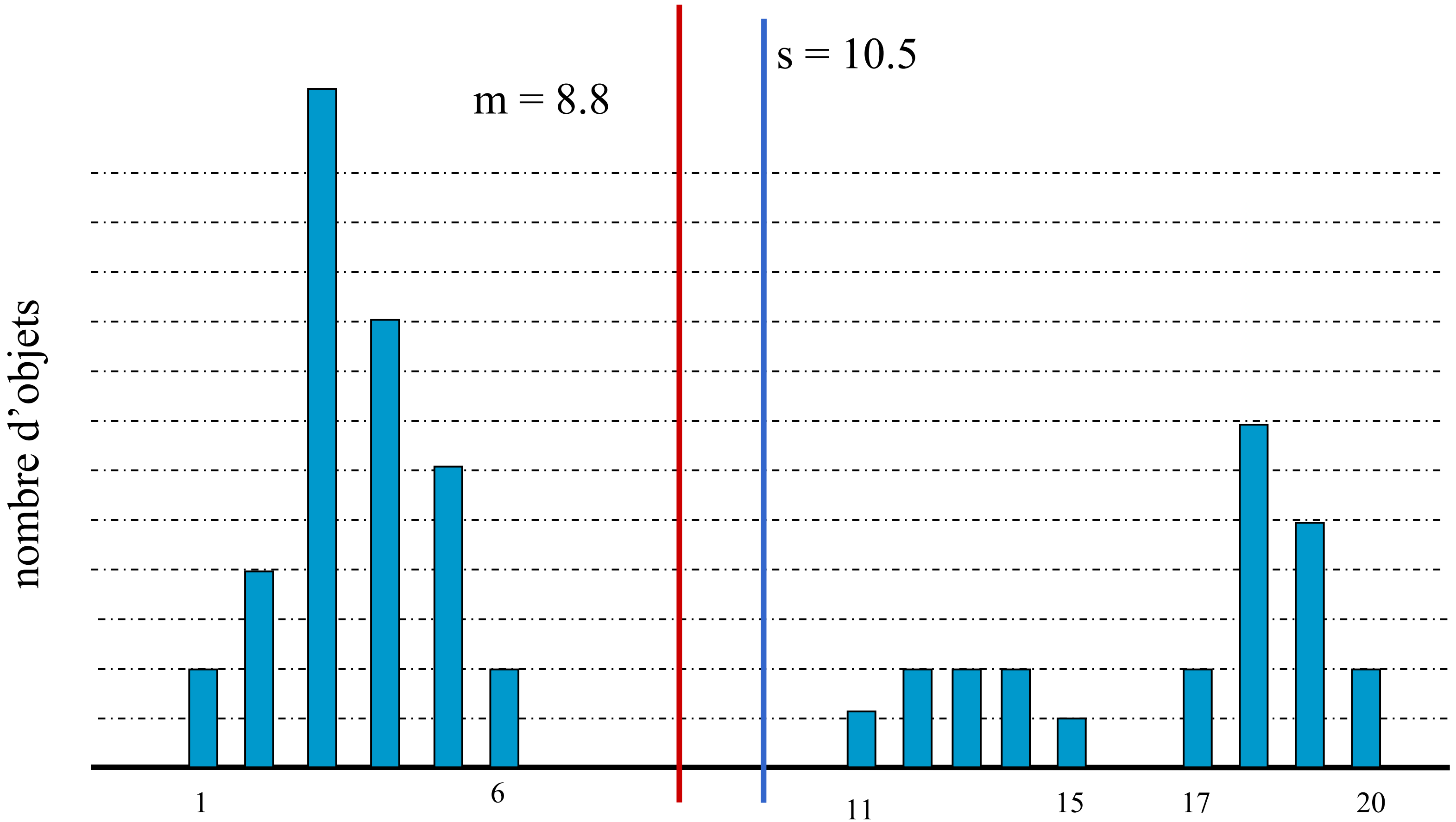
Descending hierarchical classification

- Calculation of the weighted average:
  - m = 529 / 60 = 8.8



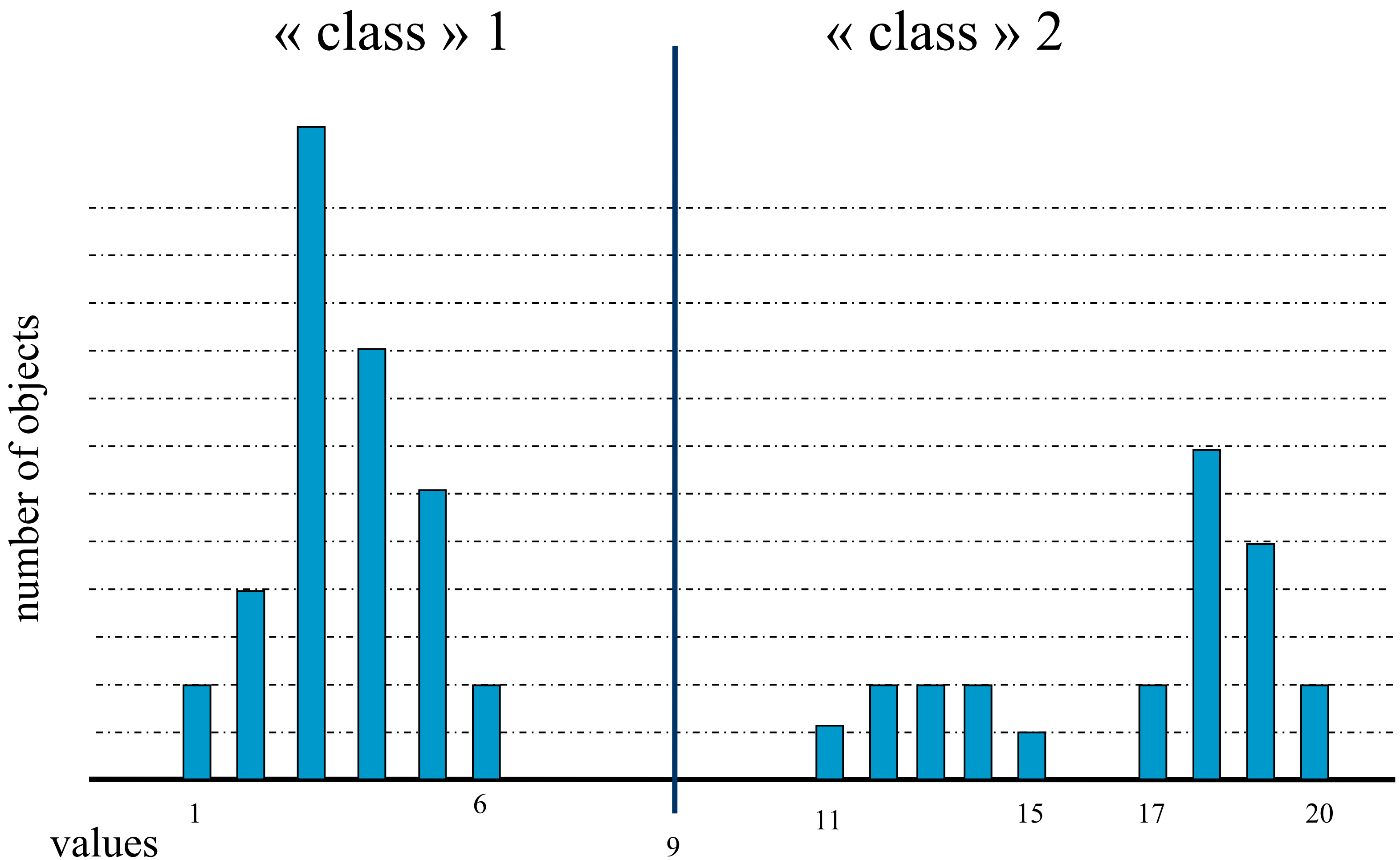
Descending hierarchical classification

10,5 ≠ 8,8 => we divide at 9



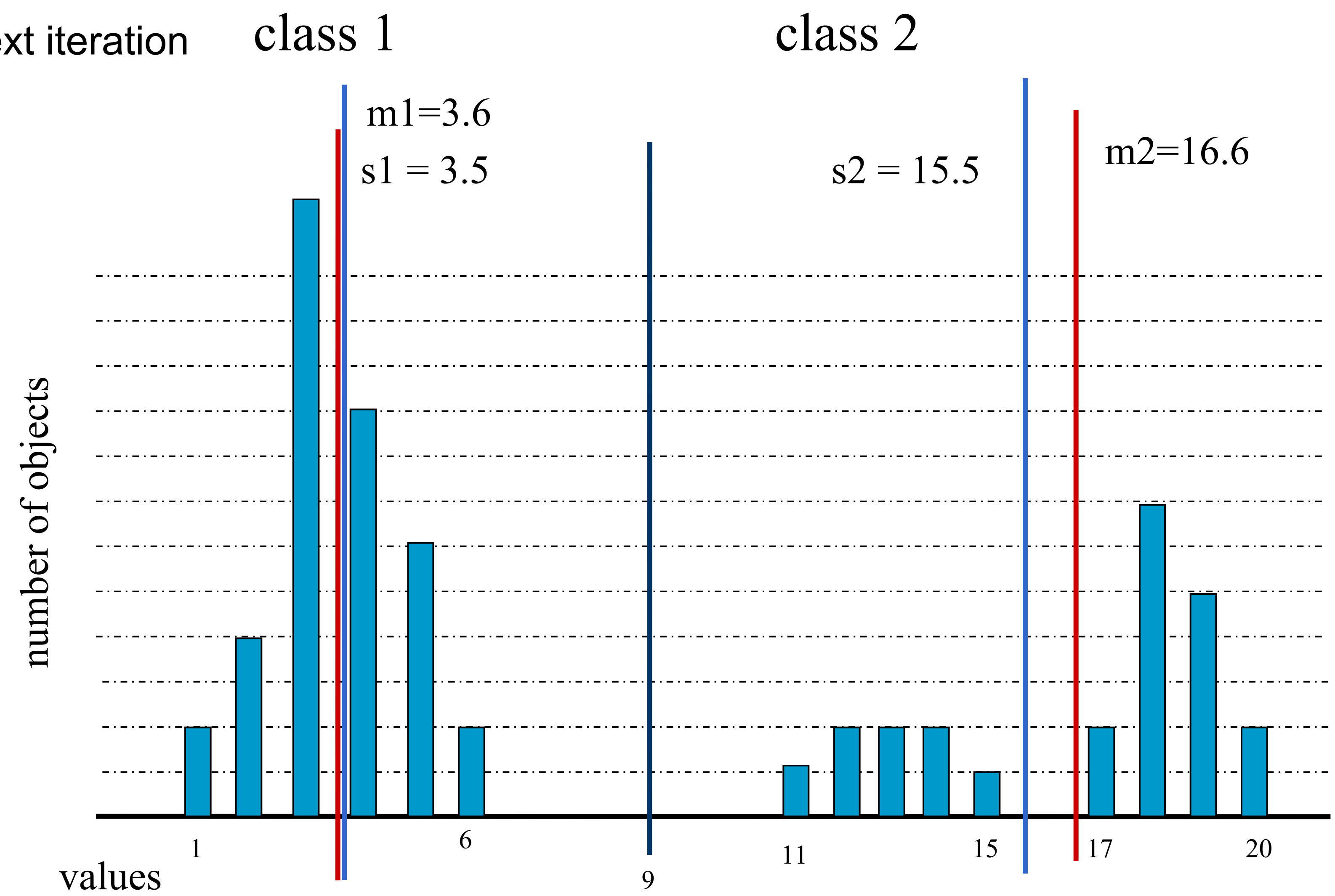
Descending hierarchical classification

10,5 ≠ 8,8 => we divide at 9

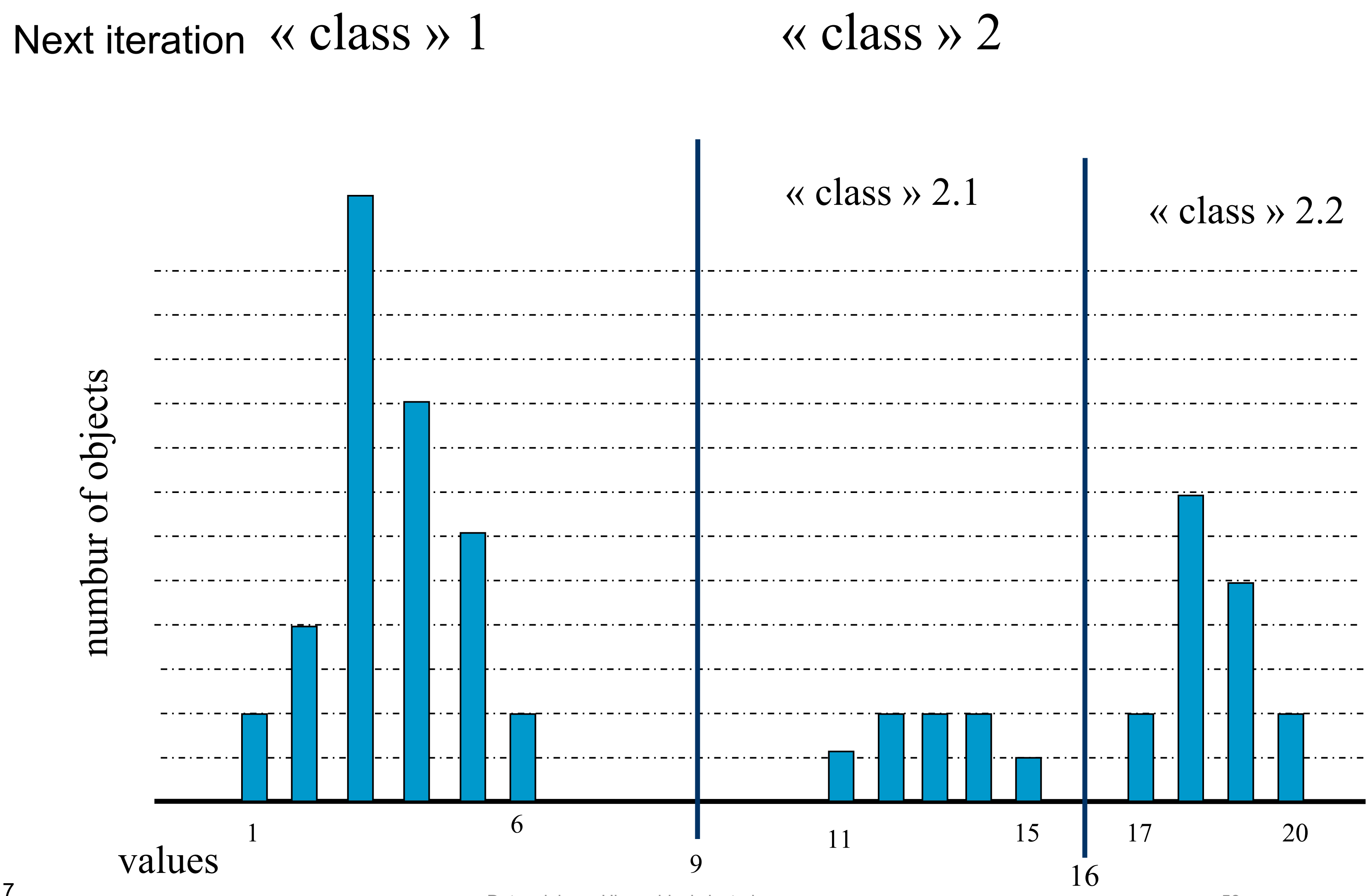


Descending hierarchical classification

Next iteration

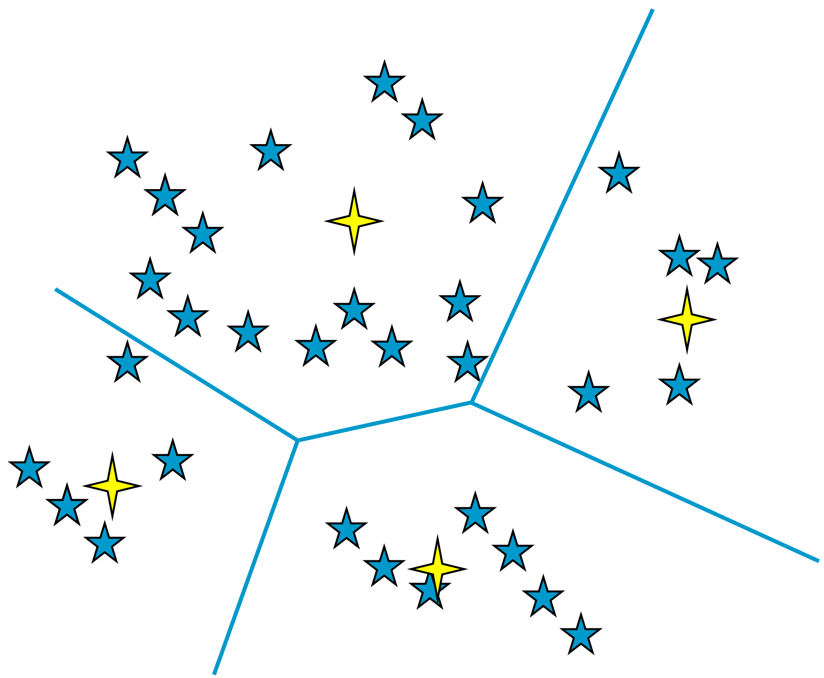


Descending hierarchical classification



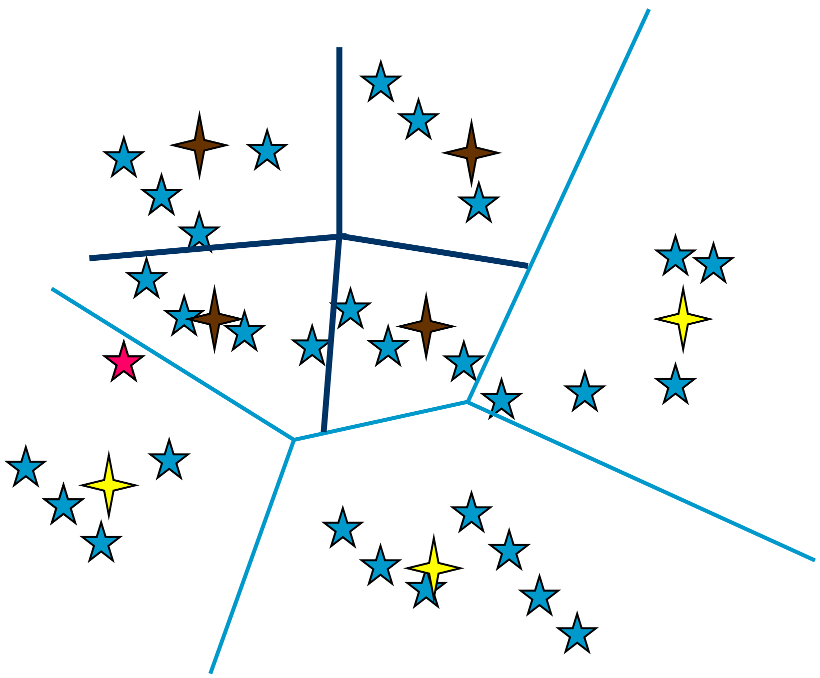
Descending hierarchical classification

- K-means example:
  - two approaches when developing a class
    - Either **only the objects inside the class are reclassified**



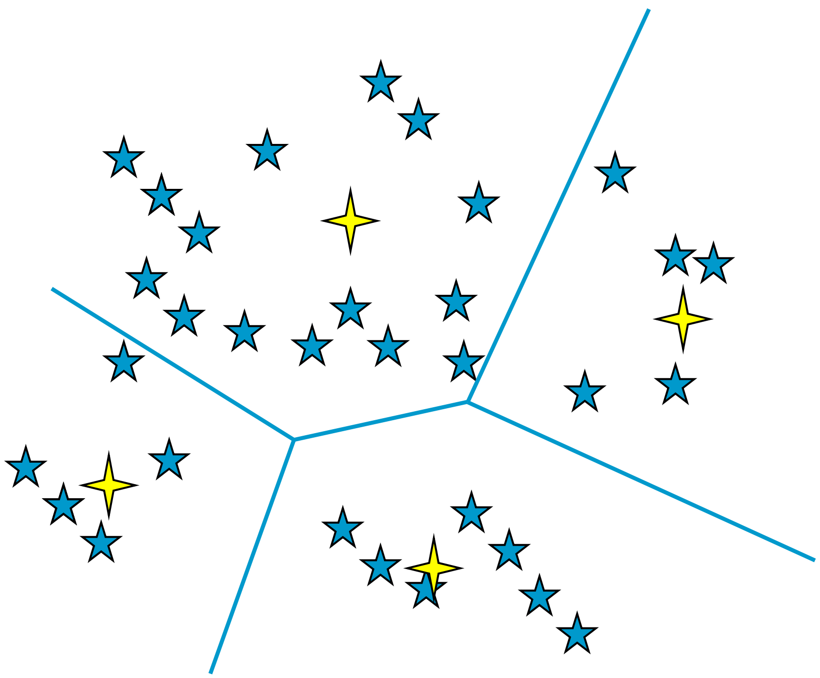
Descending hierarchical classification

- K-means example:
  - two approaches when developing a class
    - Either **only the objects inside the class are reclassified** and new centers computed



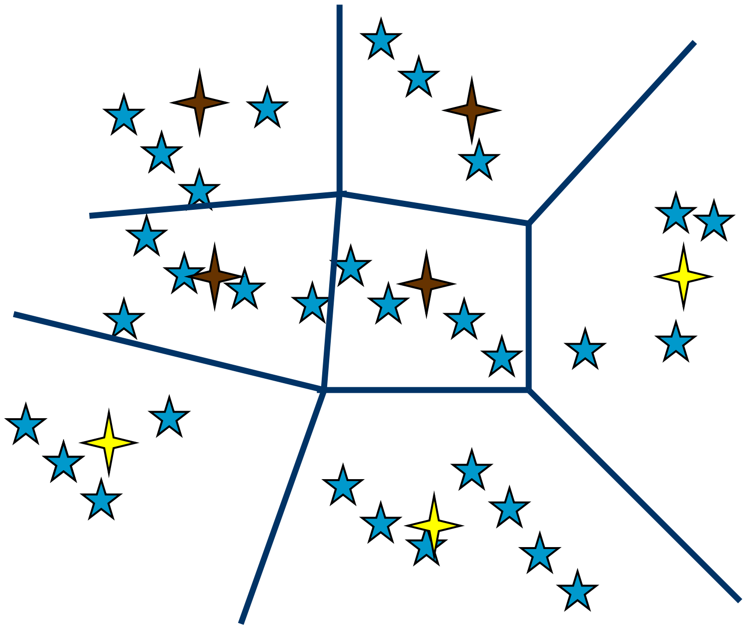
Descending hierarchical classification

- K-means example:
  - two approaches when developing a class
    - Or **all objects are reclassified from all centers**



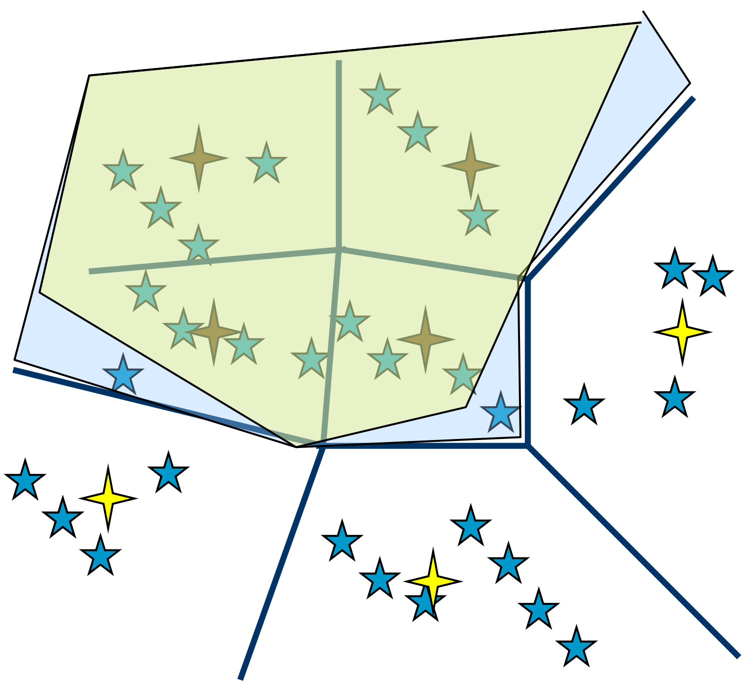
Descending hierarchical classification

- K-means example:
  - two approaches when developing a class
    - Or **all objects are reclassified from all centers**



Descending hierarchical classification

- K-means example:
  - inconvenient
    - the initial class does not cover all the objects assigned to the new kernels -> we use the property of a hierarchy



Descending hierarchical classification

- How to determine K:
  1. K is found using statistical criteria
  2. K is found by studying the histograms
  3. K is fix: in general K=2

- K-means example:
  - The first two cases correspond to ISODATA
  - for the third case, the problem of the initialization of new centers remains

Descending hierarchical classification

- Hierarchical K-means:
  - Generate in the class you would like to split two points by slightly perturbing the center of gravities
  - Apply the K-means algorithm to the class you would like to split
  - iterate