# Data Analysis and Information Exploitation (ADEI)

**Bachelor Degree in Informatics Engineering**
**Information System Track**

**FIB-ADEI – 6 ECTS - Course 2020-21**

**BarcelonaTech - UPC**

# UNIT 3. CLUSTERING

Lecturer: Lídia Montero (DEIO)

FIB UPC UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

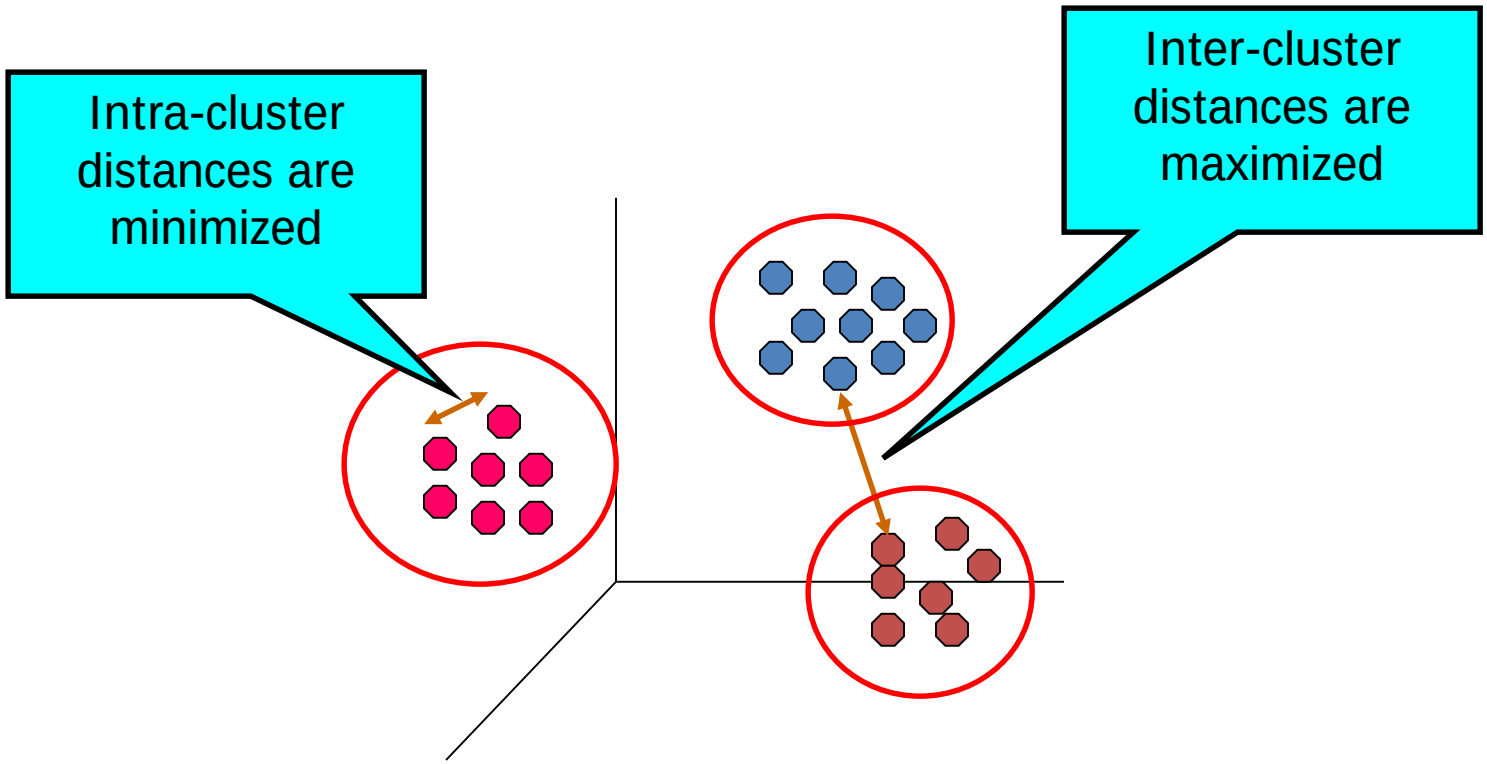
# Outline

1. Principles
2. Direct partitioning
3. Hierarchical clustering
4. Clustering and principal axes methods
5. Clusters description

- Two main families of clustering methods:
  - ✓Direct partitioning methods
  - ✓ Hierarchical methods
- **Classify either original data or factorial projected data?**
- Classification into factorial space reduces noisy and coordinates are alway real numbers.
- How many axes have to be taken?

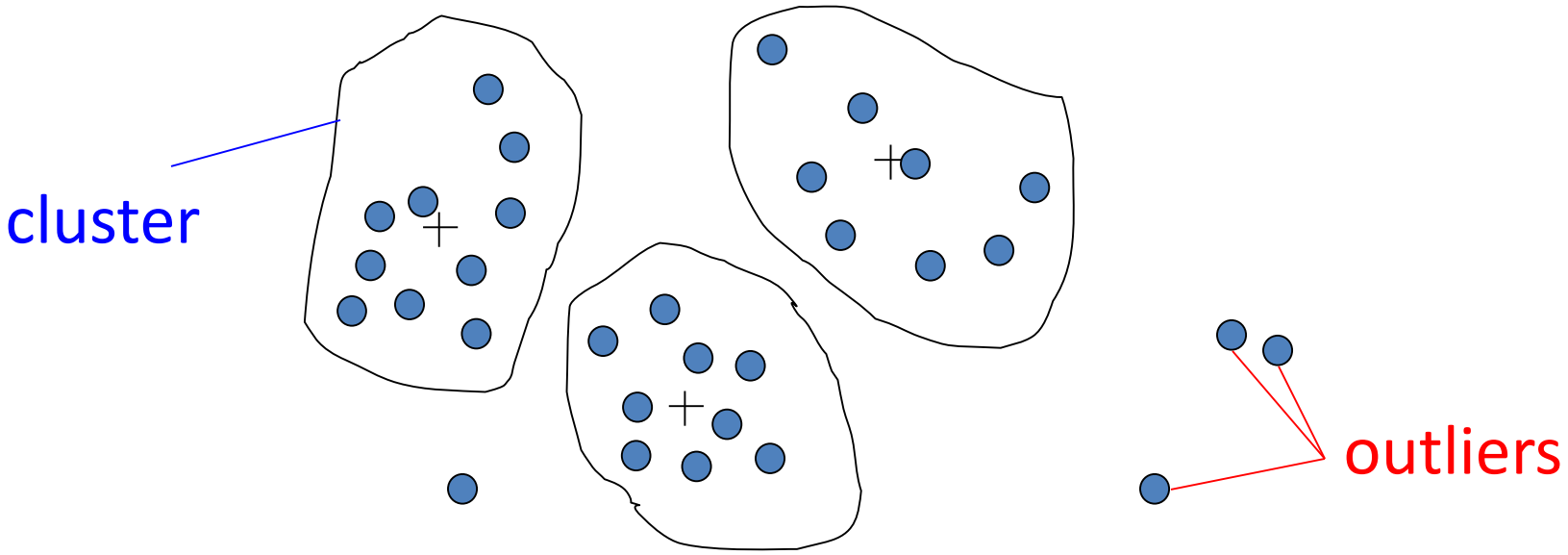
# What is clustering?

- A **grouping** of data objects such that the objects **within a group are similar** (or related) to one another **and different from (or unrelated to) the objects in other groups**



# Outliers

- **Outliers** are **objects that do not belong to any cluster** or form clusters of very small cardinality




- In some applications we are interested in discovering outliers, not clusters (outlier analysis)

# Why do we cluster?

- Clustering : given a collection of data objects group them so that
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters

- Clustering results are used:
  - As a stand-alone tool to get insight into data distribution
    - Visualization of clusters may unveil important information
  - As a preprocessing step for other algorithms
    - Efficient indexing or compression often relies on clustering

# Applications of clustering?

- Image Processing
  - cluster images based on their visual content
- Web
  - Cluster groups of users based on their access patterns on webpages
  - Cluster webpages based on their content
- Bioinformatics
  - Cluster similar proteins together (similarity wrt chemical structure and/or functionality etc)
- Many more…

# The clustering task

- Group observations into groups so that the observations belonging in the same group are similar, whereas observations in different groups are different

- **Basic questions:**
  - What does “similar” mean
  - What is a good partition of the objects? I.e., how is the quality of a solution measured
  - How to find a good partition of the observations

# Observations to cluster

## Machine Learning Issues:

- Real-value attributes/variables
  - e.g., salary, height
- Binary attributes
  - e.g., gender (M/F), has_cancer(T/F)
- Nominal (categorical) attributes
  - e.g., religion (Christian, Muslim, Buddhist, Hindu, etc.)
- Ordinal/Ranked attributes
  - e.g., military rank (soldier, sergeant, lieutenant, captain, etc.)
- Variables of mixed types
  - multiple attributes with various types

# Observations to cluster

- Usually data objects consist of a set of attributes (also known as **features/dimensions/variables**)

- If all **K** dimensions are ***real-valued*** then we can ***visualize*** each data point as points in a ***K-dimensional space***

- If all **d** dimensions are ***binary*** then we can think of each data point as a ***binary vector***

# Distance functions

- The distance **d(x, y)** between two objects **x** and **y** is a metric if
  - **d(i, j)≥0** (non-negativity)
  - **d(i, i)=0** (isolation)
  - **d(i, j)= d(j, i)** (symmetry)
  - **d(i, j) ≤ d(i, h)+d(h, j)** (triangular inequality) [**Why do we need it?**]

- The definitions of distance functions are usually different for real, boolean, categorical, and ordinal variables.

- Weights may be associated with different variables based on applications and data semantics.

# Data Structures

- *data* matrix

attributes/dimensions

tuples/objects

$$\begin{bmatrix} x_{11} & \cdots & x_{1\ell} & \cdots & x_{1K} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{i\ell} & \cdots & x_{iK} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{I1} & \cdots & x_{I\ell} & \cdots & x_{IK} \end{bmatrix}$$

- *Distance* matrix

objects

objects

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

# Distance functions for binary vectors

- **Not our topic !**
- **Jaccard similarity** between binary vectors **X** and **Y**

$$JSim(X,Y) = \frac{X \cap Y}{X \cup Y}$$

- **Jaccard distance** between binary vectors **X** and **Y**

  Jdist(X,Y) = 1- JSim(X,Y)

- Example:
  - JSim = 1/6
  - Jdist = 5/6

| | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 |
|---|---|---|---|---|---|---|
| X | 1 | 0 | 0 | 1 | 1 | 1 |
| Y | 0 | 1 | 1 | 0 | 1 | 0 |

# Distance functions for real-valued vectors

- $L_p$ norms or *Minkowski distance*:

$$L_p(x,y)=\left(|x_1-y_1|^p+|x_2-y_2|^p+\ldots+|x_K-x_K|^p\right)^{1/p}=\left(\sum_{i=1}^{K}(x_i-y_i)\right)^{1/p}$$

where ***K*** is a positive integer

- If **p = 1, $L_1$** is the *Manhattan (or city block)* distance:

$$L_1(x,y)=|x_1-y_1|+|x_2-y_2|+\ldots+|x_K-y_K|=\sum_{i=1}^{K}|x_i-y_i|$$

# Distance functions for real-valued vectors

- If **p = 2, $L_2$** is the **Euclidean distance**:

$$d(x,y)=\sqrt{(|x_1-y_1|^2+|x_2-y_2|^2+...+|x_K-y_K|^2)}$$

- Also one can use **weighted distance**:

$$d(x,y)=w_1|x_1-y_1|+w_2|x_2-y_2|+...+w_K|x_K-y_K|$$

$$d(x,y)=\sqrt{(w_1|x_1-x_1|^2+w_2|x_2-x_2|^2+...+w_K|x_K-y_K|^2)}$$

# Partitioning algorithms: basic concept

- Construct a partition of a set of ***n*** objects into a set of ***d*** clusters
  - Each object belongs to **exactly one** cluster
  - The number of clusters **d** is given in advance

# The k-means problem

- Given a set **X** of **n** points in a **K**-dimensional space and an integer **d**

- **Task:** choose a set of **d** points $\{c_1, c_2, \ldots, c_d\}$ in the **K**-dimensional space to form clusters $\{C_1, C_2, \ldots, C_d\}$ such that

$$Cost(C) = \sum_{i=1}^{d} \sum_{x \in C_i} L_2{}^2 (x - c_i)$$

  is minimized

- Some special cases: d = 1, d = n

# Algorithmic properties of the k-means problem

- NP-hard if the dimensionality of the data is at least 2 (**K>=2**)
- Finding the best solution in polynomial time is infeasible
- For **K=1** the problem is solvable in polynomial time (how?)
- A simple iterative algorithm works quite well in practice

# The k-means algorithm

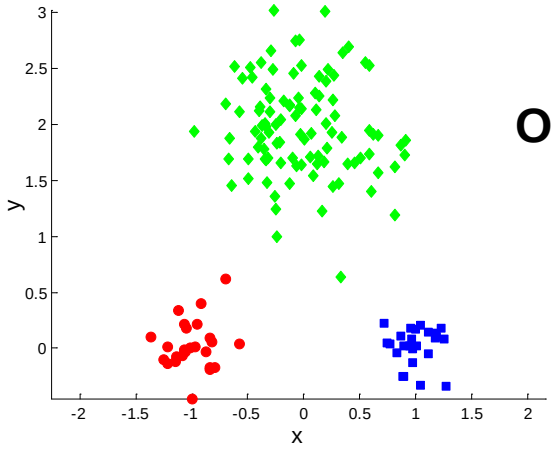
- One way of solving the **k**-means problem

  - ✓ Randomly pick **d** cluster centers $\{c_1,\ldots,c_d\}$
  - ✓ For each **i**, set the cluster $C_i$ to be the set of points in **X** that are closer to $c_i$ than they are to $c_j$ for all $i \neq j$
  - ✓ For each **i** let $c_i$ be the center of cluster $C_i$ (mean of the vectors in $C_i$)
  - ✓ Repeat until convergence

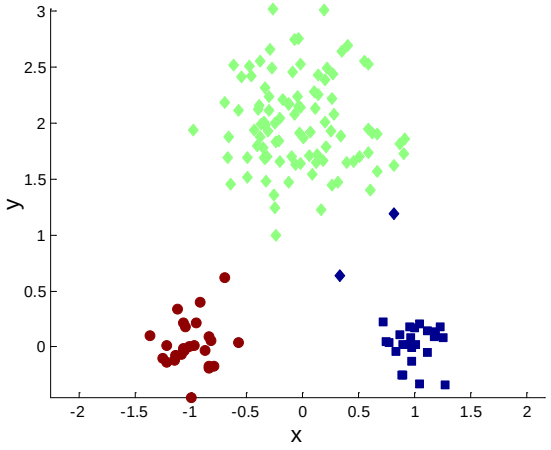
# Properties of the k-means algorithm

- Finds a local optimum
- Converges often quickly (but not always)
- The choice of initial points can have large influence in the result

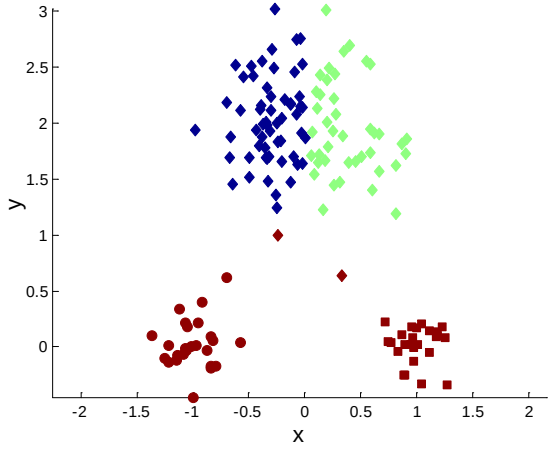
# Two different K-means Clusterings




**Original Points**




**Optimal Clustering**




**Sub-optimal Clustering**

# Discussion k-means algorithm

- Finds a local optimum
- Converges often quickly (but not always)
- The choice of initial points can have large influence
  - Clusters of different densities
  - Clusters of different sizes

- Outliers can also cause a problem (Example?)

# Some alternatives to random initialization of the central points

- Multiple runs
  - Helps, but probability is not on your side

- Select original set of  points by methods other than random . E.g.,  pick the most distant (from each other) points as cluster centers (kmeans++ algorithm)

# The k-median problem

- Given a set **X** of **n** points in a **K**-dimensional space and an integer **d**

- **Task:** choose a set of **d** points $\{c_1, c_2, \ldots, c_d\}$ from **X** and form clusters $\{C_1, C_2, \ldots, C_d\}$ such that

$$Cost(C) = \sum_{i=1}^{d} \sum_{x \in C_i} L_1(x, c_i)$$

is minimized

# The *k-medoids* algorithm

- *Or … PAM* (Partitioning Around Medoids, 1987)
  - Choose randomly **d** medoids from the original dataset **X**
  - Assign each of the **n-d** remaining points in **X** to their closest medoid
  - iteratively replace one of the medoids by one of the non-medoids if it improves the total clustering cost

# Discussion of PAM algorithm

- The algorithm is very similar to the k-means algorithm
- It has the same advantages and disadvantages
- How about efficiency?

# CLARA (Clustering Large Applications)

- It draws ***multiple samples*** of the data set, applies *PAM* on each sample, and gives the best clustering as the output
- Strength: deals with larger data sets than *PAM*
- Weakness:
  - Efficiency depends on the sample size
  - A good clustering based on samples will not necessarily represent a good clustering of the whole data set if the sample is biased

# The k-center problem

- Given a set **X** of **n** points in a **K**-dimensional space and an integer **d**

- **Task:** choose a set of **d** points from **X** as cluster centers $\{c_1, c_2, \ldots, c_d\}$ such that for clusters $\{C_1, C_2, \ldots, C_d\}$

$$R(C) = \max_{1 \le j \le d} \max_{x \in C_j} d(x, c_j)$$

is minimized

# Algorithmic properties of the k-centers problem

- NP-hard if the dimensionality of the data is at least 2 (**K>=2**)
- Finding the best solution in polynomial time is infeasible
- For **K=1** the problem is solvable in polynomial time (how?)
- A simple combinatorial algorithm works well in practice

# What is the right number of clusters?

- …or who sets the value of **d**?
- For n points to be clustered consider the case where **d=n**. What is the value of the error function
- What happens when **d = 1**?
- Since we want to minimize the error why don't we select always **d = n**?

# Occam's razor and the minimum description length principle

- Clustering provides a description of the data
- For a description to be good it has to be:
  - Not too general
  - Not too specific
- Penalize for every extra parameter that one has to pay
- Penalize the number of bits you need to describe the extra parameter
- So for a clustering **C**, extend the cost function as follows:
- **New Cost(C) = Cost( C ) + |C| x logn**
- **Criteria: Max $I_{partition}/I_{total}$**

# R Function kmeans

## kmeans(X, centers, iter.max= 10)

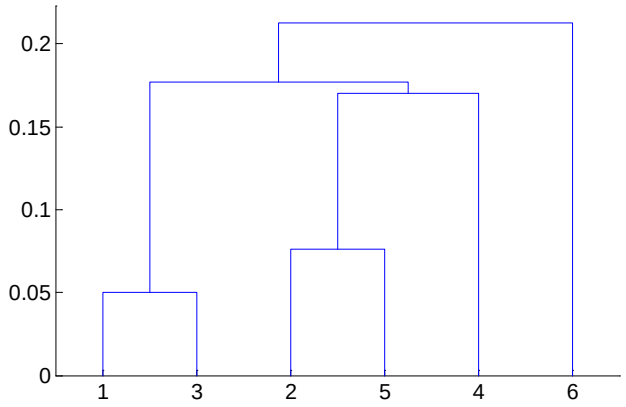
- X: A numeric matrix or a data frame with all numeric columns.
- centers: Either the number of clusters or a set of initial cluster centers.
- If the first, a random set of rows in 'x' are chosen as the initial centers.
- iter.max: The maximum number of iterations allowed.

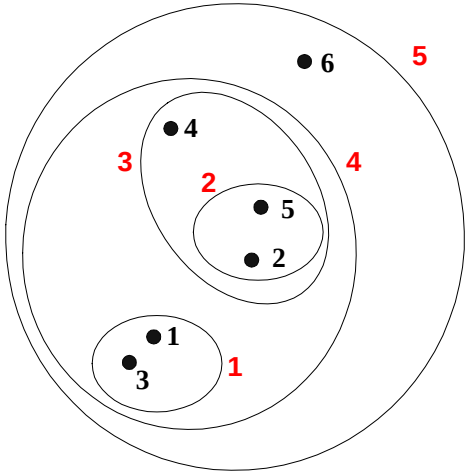
**Results:**

- cluster: A vector of integers indicating the cluster to which each point is allocated.
- centers: A matrix of cluster centres.
- withinss: The within-cluster sum of squares for each cluster.
- size: The number of points in each cluster.

# Hierarchical Clustering

- Produces a set of ***nested clusters*** organized as a hierarchical tree
- Can be visualized as a **dendrogram**
  - A tree-like diagram that records the sequences of merges or splits



# Strengths of Hierarchical Clustering

- No assumptions on the number of clusters
  - Any desired number of clusters can be obtained by ‘cutting’ the dendogram at the proper level

- Hierarchical clusterings may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., phylogeny reconstruction, etc), web (e.g., product catalogs) etc

# Hierarchical Clustering

- Two main types of hierarchical clustering
  - **Agglomerative:**
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or **d** clusters) left
  - **Divisive:**
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains a point (or there are **d** clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

# Complexity of hierarchical clustering

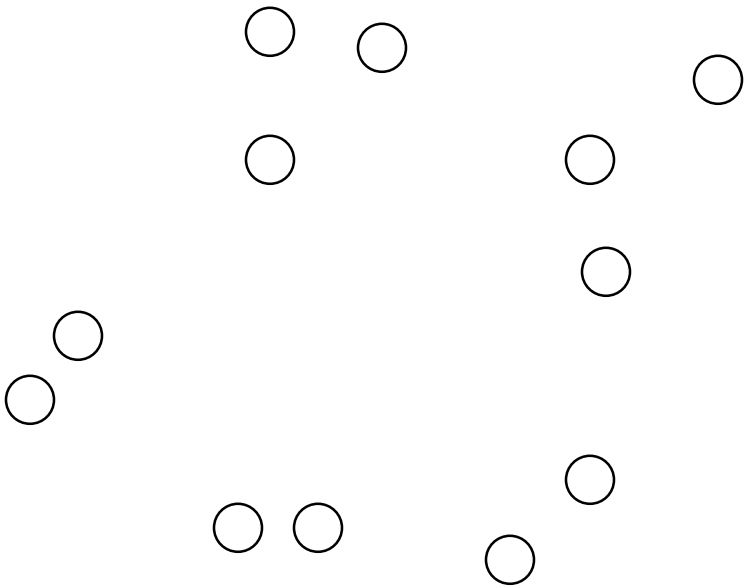
- Distance matrix is used for deciding which clusters to merge/split
- At least quadratic in the number of data points
- Not usable for large datasets

# Agglomerative clustering algorithm

- **Most popular hierarchical clustering technique**
- Basic algorithm
  1. Compute the distance matrix between the input data points
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the distance matrix
  6. **Until** only a single cluster remains
- Key operation is the computation of the distance between two clusters: **Different definitions of the distance between clusters lead to different algorithms**

# Input/ Initial setting

- Start with clusters of individual points and a distance/proximity matrix

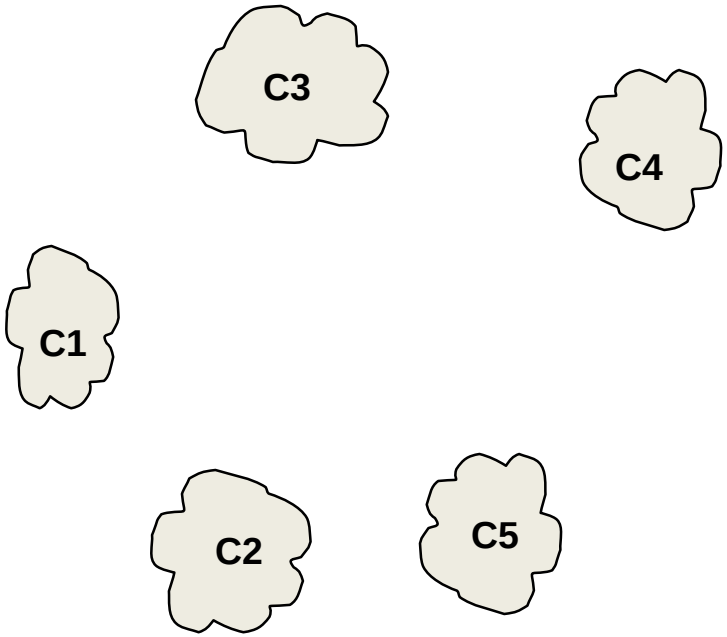


| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |

Distance/Proximity Matrix

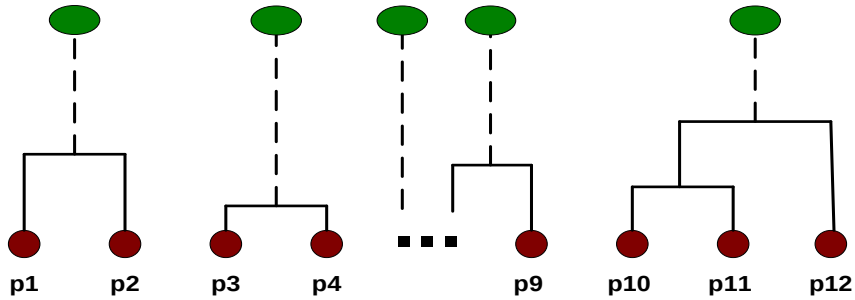
# Intermediate State

- After some merging steps, we have some clusters



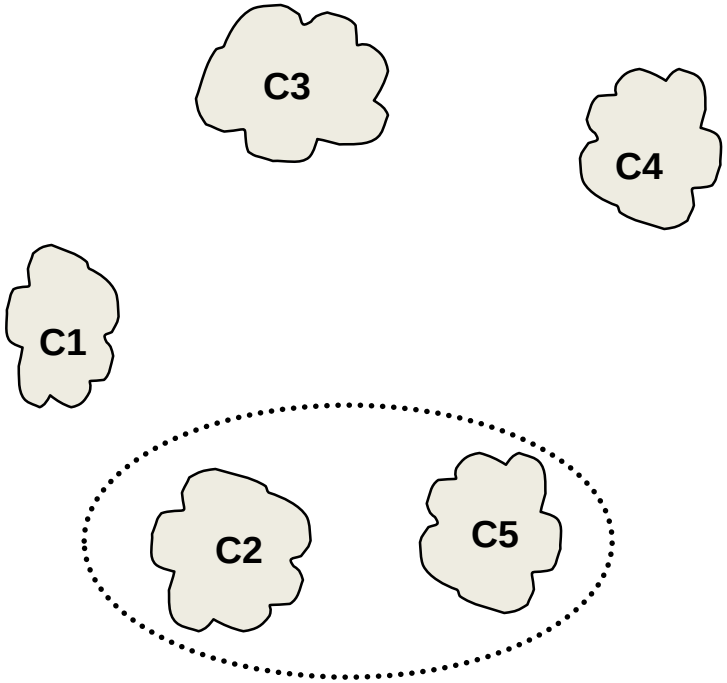

| | C1 | C2 | C3 | C4 | C5 |
|---|---|---|---|---|---|
| C1 | | | | | |
| C2 | | | | | |
| C3 | | | | | |
| C4 | | | | | |
| C5 | | | | | |

**Distance/Proximity Matrix**



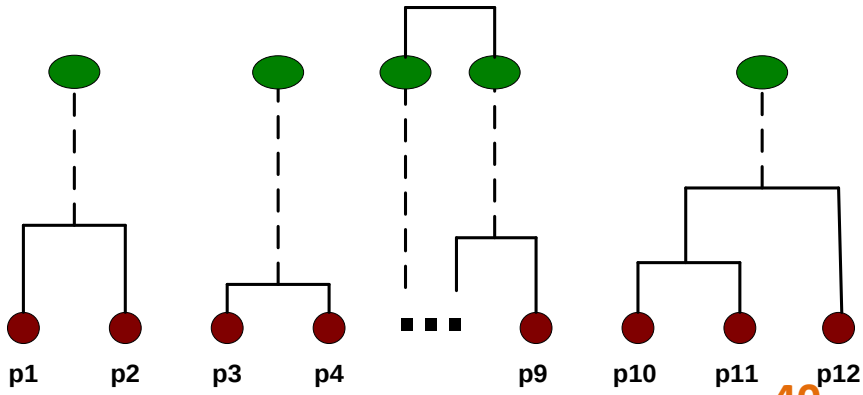
# Intermediate State

- Merge the two closest clusters (C2 and C5) and update the distance matrix.



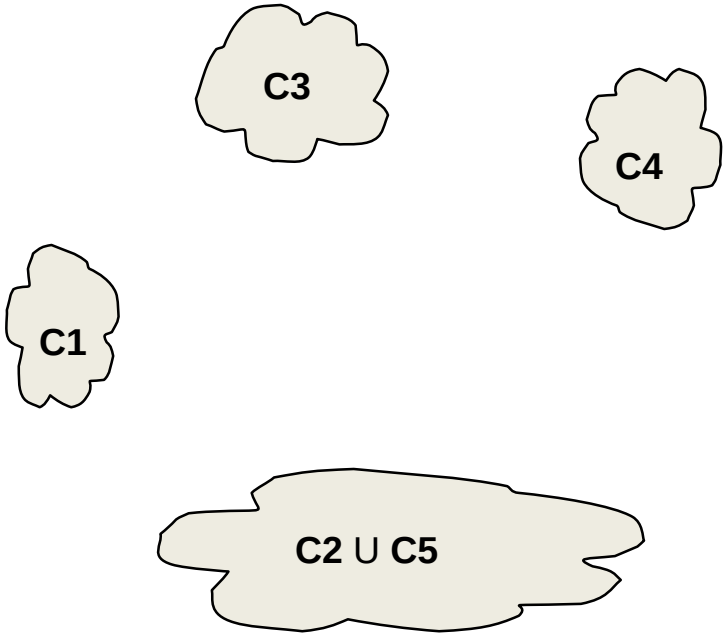

| | C1 | C2 | C3 | C4 | C5 |
|---|---|---|---|---|---|
| C1 | | | | | |
| C2 | | | | | |
| C3 | | | | | |
| C4 | | | | | |
| C5 | | | | | |

**Distance/Proximity Matrix**

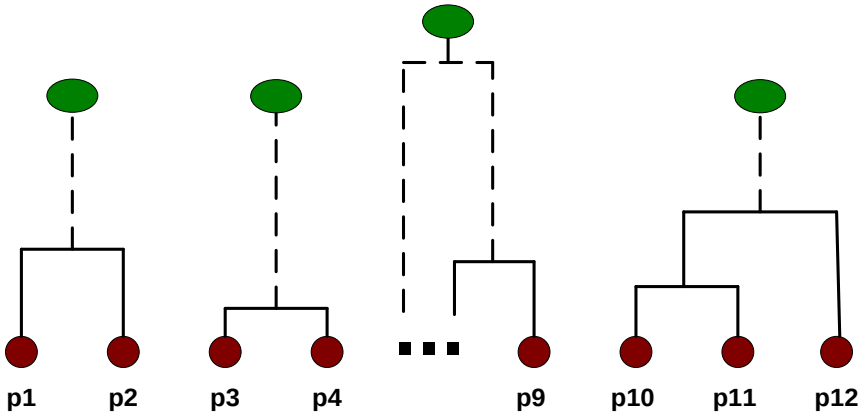


# After Merging

- "How do we update the distance matrix?"




| | C1 | C2 U C5 | C3 | C4 |
|---|---|---|---|---|
| C1 | | ? | | |
| C2 U C5 | ? | ? | ? | ? |
| C3 | | ? | | |
| C4 | | ? | | |



# Distance between two clusters

- Each cluster is a set of points

- How do we define distance between two sets of points
  - Lots of alternatives
  - Not an easy task

# Distance between two clusters

- **Single-link distance** between clusters $C_i$ and $C_j$ is the ***minimum distance*** between any object in $C_i$ and any object in $C_j$

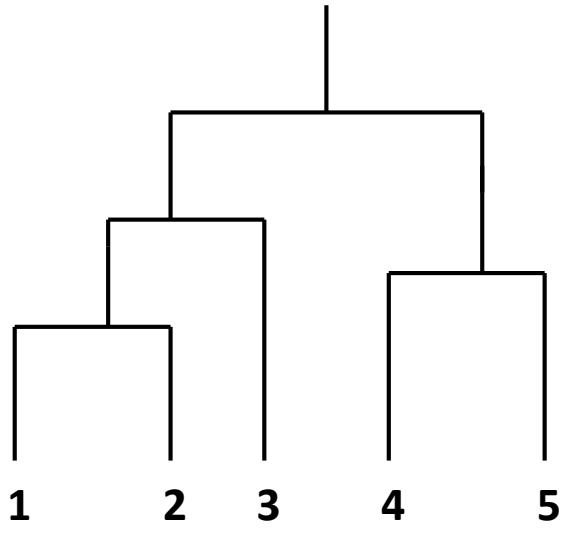
- The distance is **defined by the two most similar objects**

$$D_{sl}(C_i, C_j) = \min_{x,y} \{ d(x, y) \mid x \in C_i, y \in C_j \}$$

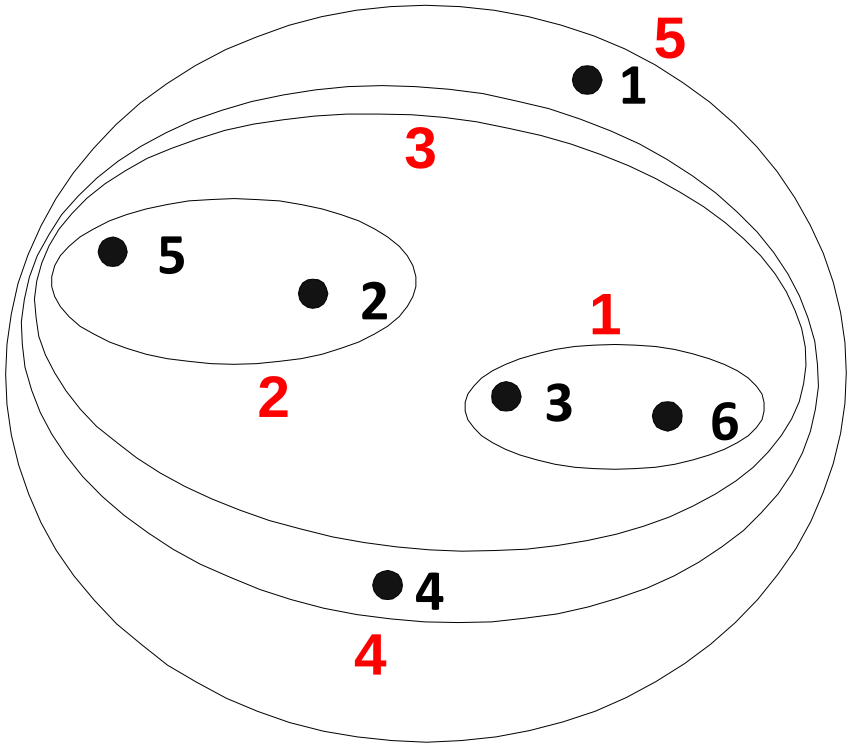
# Single-link clustering: example

- Determined by one pair of points, i.e., by one link in the proximity graph.

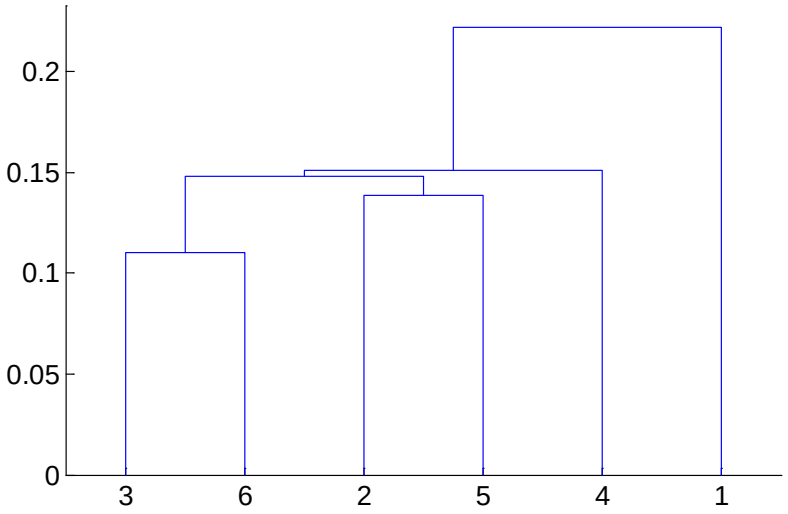
| | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| I1 | 1.00 | 0.90 | 0.10 | 0.65 | 0.20 |
| I2 | 0.90 | 1.00 | 0.70 | 0.60 | 0.50 |
| I3 | 0.10 | 0.70 | 1.00 | 0.40 | 0.30 |
| I4 | 0.65 | 0.60 | 0.40 | 1.00 | 0.80 |
| I5 | 0.20 | 0.50 | 0.30 | 0.80 | 1.00 |



# Single-link clustering: example

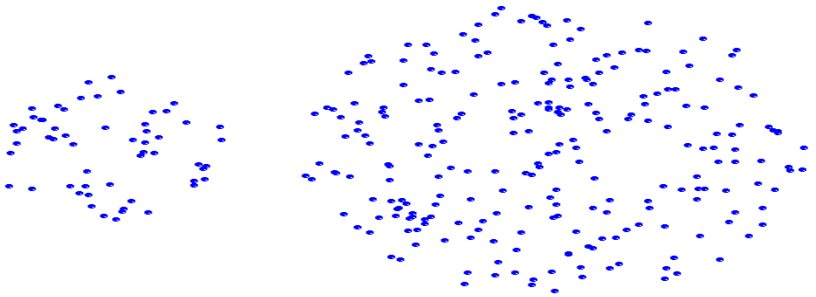



Nested Clusters

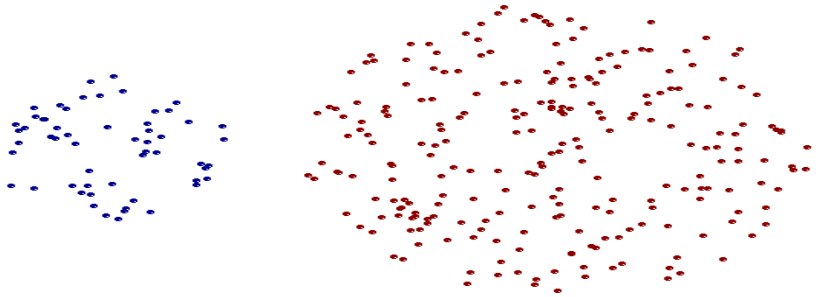



Dendrogram

# Strengths of single-link clustering



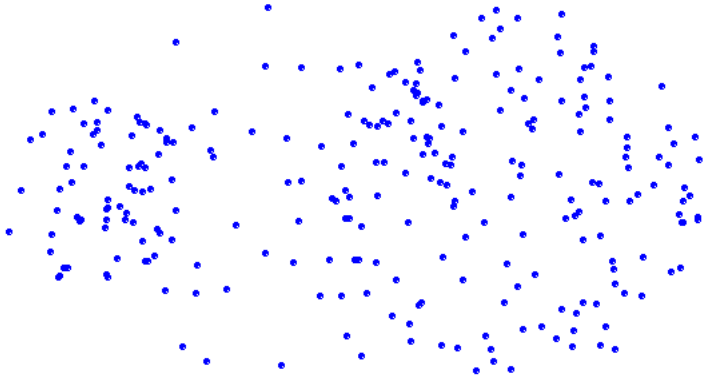
Original Points



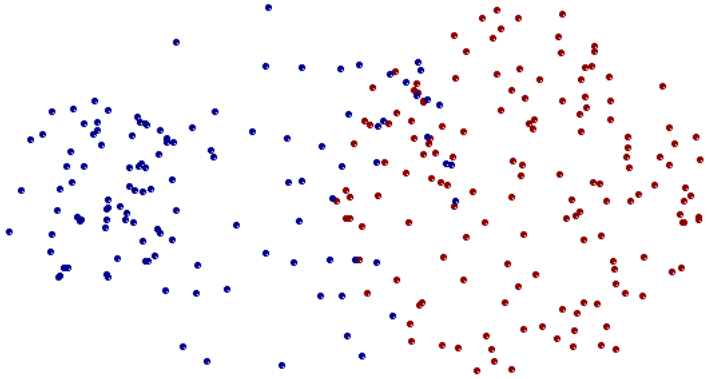
Two Clusters

- Can handle non-elliptical shapes

# Limitations of single-link clustering



Original Points



Two Clusters

- Sensitive to noise and outliers
- It produces long, elongated clusters

# Distance between two clusters

- **Complete-link distance** between clusters $C_i$ and $C_j$ is the ***maximum distance*** between any object in $C_i$ and any object in $C_j$

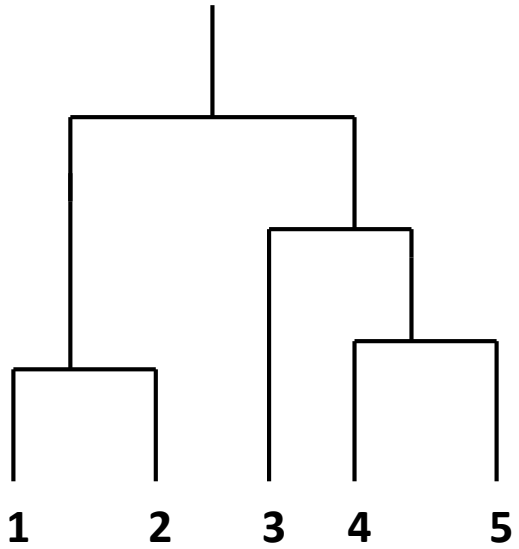
- The distance is **defined by the two most dissimilar objects**

$$D_{cl}(C_i, C_j) = \max_{x,y} \{ d(x, y) \mid x \in C_i, y \in C_j \}$$

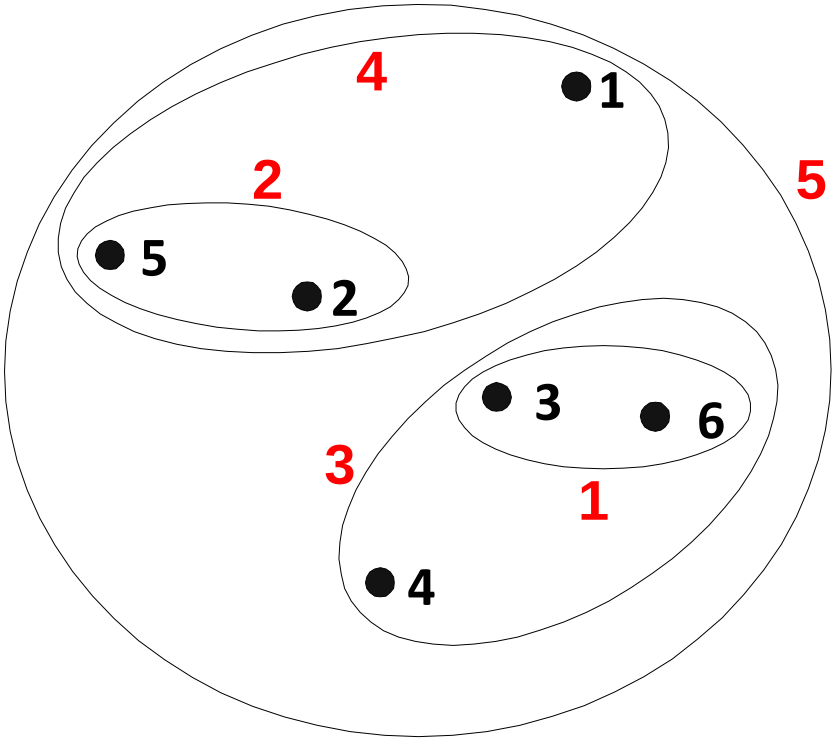
# Complete-link clustering: example

- Distance between clusters is determined by the two most distant points in the different clusters

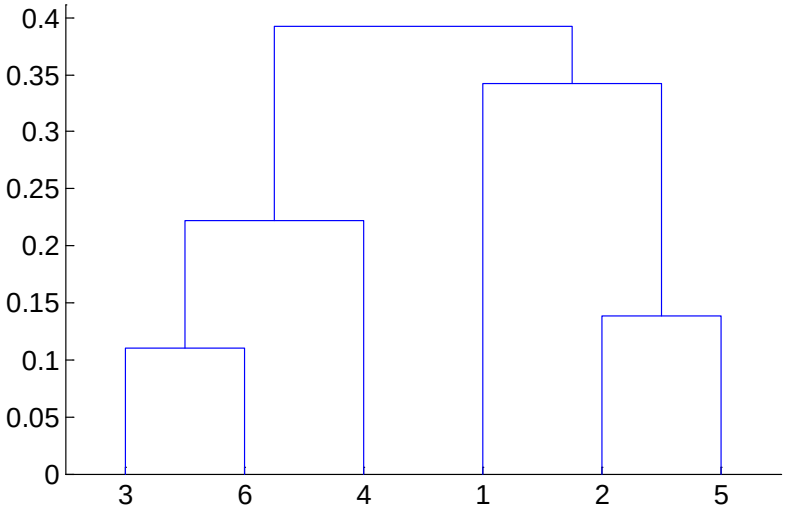
| | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| I1 | 1.00 | 0.90 | 0.10 | 0.65 | 0.20 |
| I2 | 0.90 | 1.00 | 0.70 | 0.60 | 0.50 |
| I3 | 0.10 | 0.70 | 1.00 | 0.40 | 0.30 |
| I4 | 0.65 | 0.60 | 0.40 | 1.00 | 0.80 |
| I5 | 0.20 | 0.50 | 0.30 | 0.80 | 1.00 |



# Complete-link clustering: example

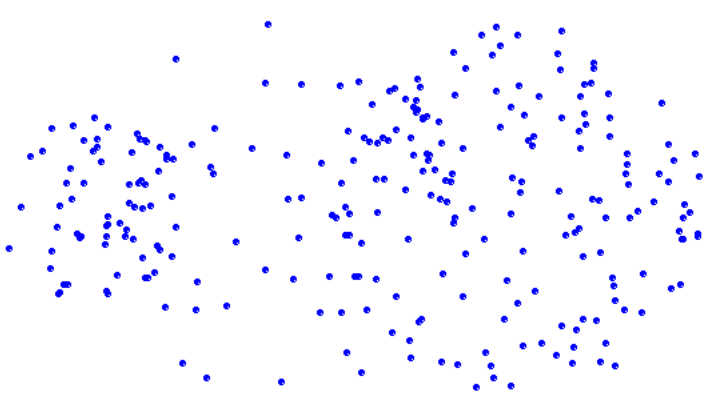



Nested Clusters

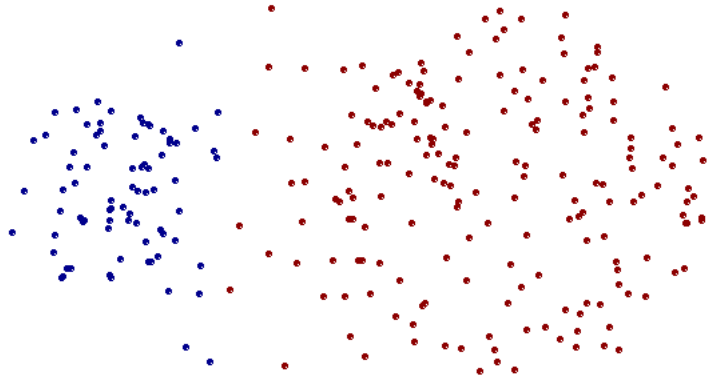



Dendrogram

# Strengths of complete-link clustering



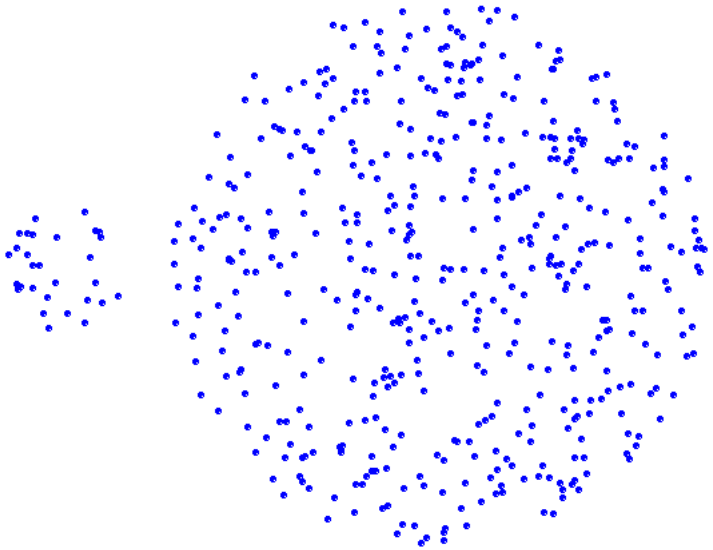
Original Points



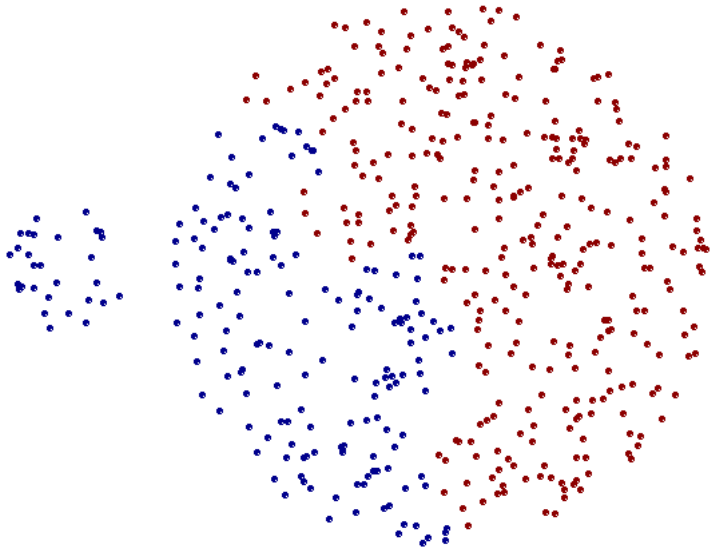
Two Clusters

- More balanced clusters (with equal diameter)
- Less susceptible to noise

# Limitations of complete-link clustering



Original Points



Two Clusters

- Tends to break large clusters
- All clusters tend to have the same diameter – small clusters are merged with larger ones

# Distance between two clusters

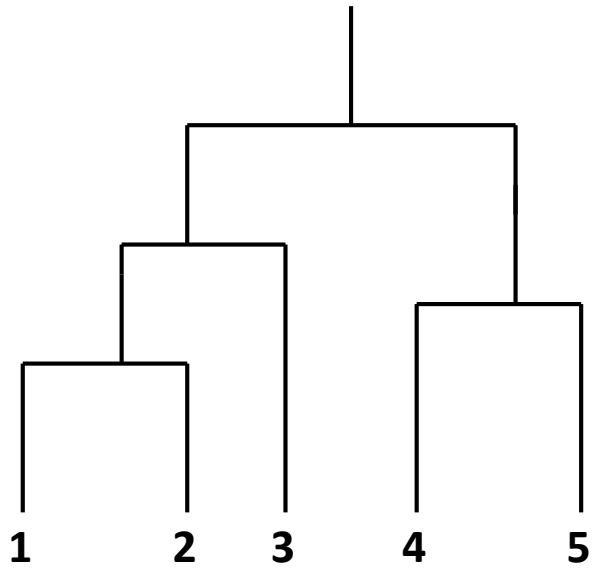
- **Group average distance** between clusters $C_i$ and $C_j$ is the ***average distance*** between any object in $C_i$ and any object in $C_j$

$$D_{avg}(C_i, C_j) = \frac{1}{|C_i| \times |C_j|} \sum_{x \in C_i, y \in C_j} d(x, y)$$

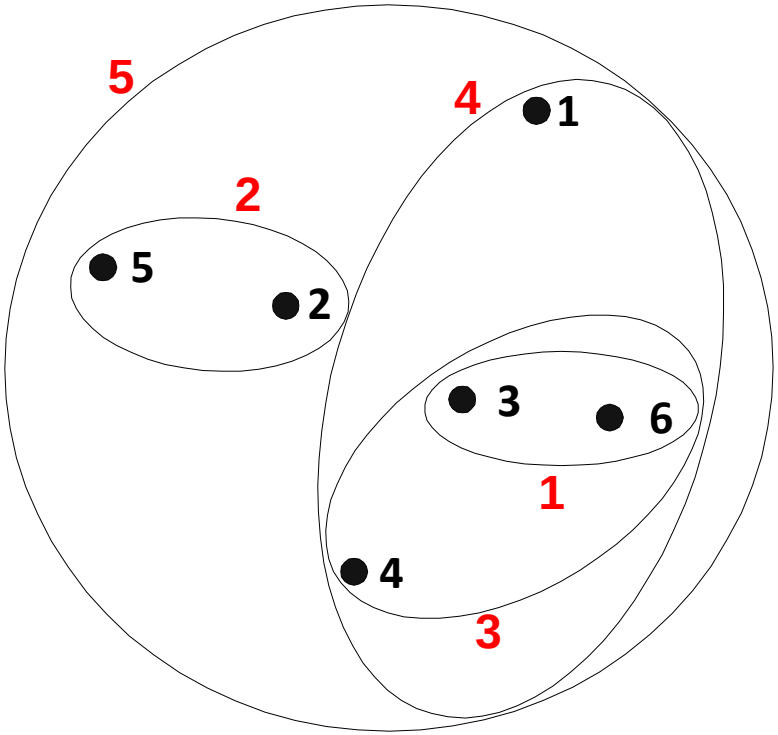
# Average-link clustering: example

- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

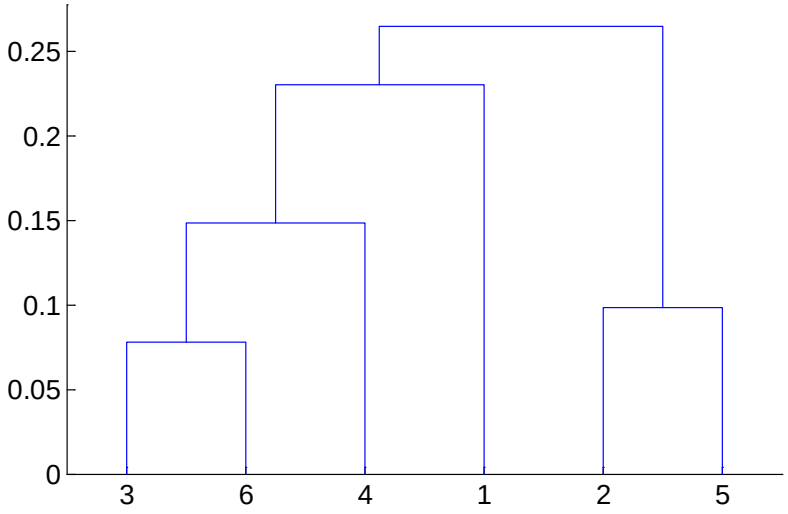
| | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| I1 | 1.00 | 0.90 | 0.10 | 0.65 | 0.20 |
| I2 | 0.90 | 1.00 | 0.70 | 0.60 | 0.50 |
| I3 | 0.10 | 0.70 | 1.00 | 0.40 | 0.30 |
| I4 | 0.65 | 0.60 | 0.40 | 1.00 | 0.80 |
| I5 | 0.20 | 0.50 | 0.30 | 0.80 | 1.00 |



# Average-link clustering: example




Nested Clusters




Dendrogram

# Average-link clustering: discussion

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise and outliers
- Limitations
  - Biased towards globular clusters

# Distance between two clusters

- **Centroid distance** between clusters $C_i$ and $C_j$ is the distance between the centroid $r_i$ of $C_i$ and the centroid $r_j$ of $C_j$

$$D_{centroids}(C_i, C_j) = d(r_i, r_j)$$

# Distance between two clusters

- **Ward's distance** between clusters $C_i$ and $C_j$ is the ***difference*** between the ***total within cluster sum of squares for the two clusters separately***, and the ***within cluster sum of squares resulting from merging the two clusters*** in cluster $C_{ij}$

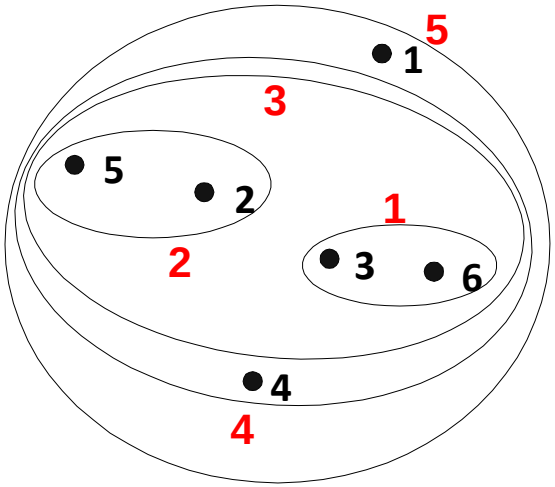
$$D_w(C_i, C_j) = \sum_{x \in C_i} (x - r_i)^2 + \sum_{x \in C_j} (x - r_j)^2 - \sum_{x \in C_{ij}} (x - r_{ij})^2$$

- $r_i$: centroid of $C_i$
- $r_j$: centroid of $C_j$
- $r_{ij}$: centroid of $C_{ij}$

# Ward's distance for clusters

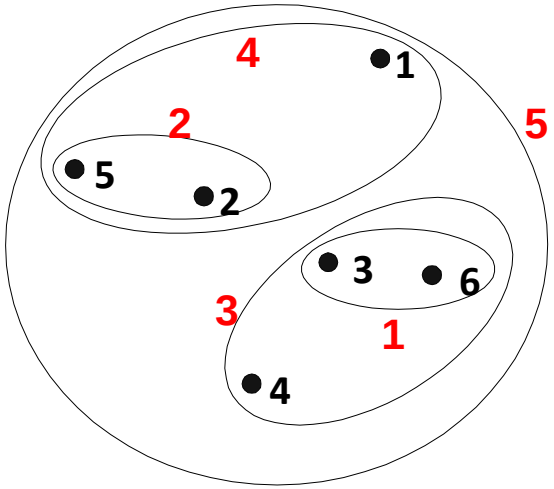
- Similar to group average and centroid distance
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of k-means
  - Can be used to initialize k-means

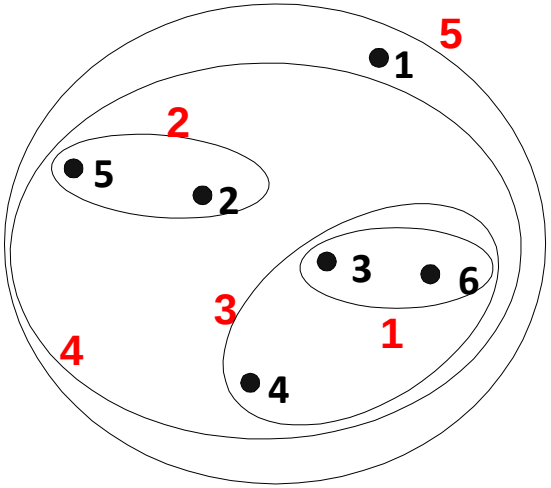
# Hierarchical Clustering: Comparison




MIN

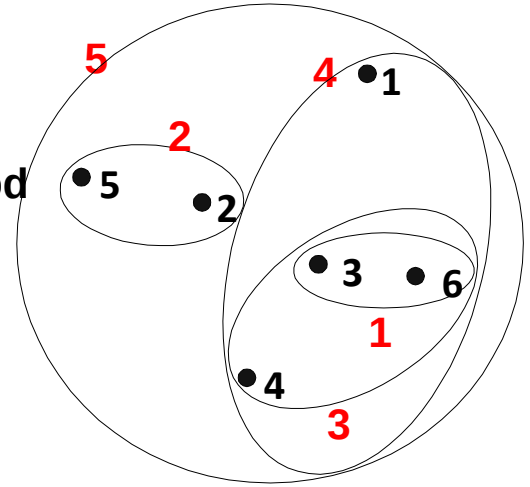
MAX




Group Average

Ward's Method



# Hierarchical Clustering: Time and Space requirements

- For a dataset **X** consisting of **n** points
- **$O(n^2)$ space**; it requires storing the distance matrix
- **$O(n^3)$ time** in most of the cases
  - There are **n** steps and at each step the size **$n^2$** distance matrix must be updated and searched
  - Complexity can be reduced to **$O(n^2 \log(n))$** time for some approaches by using appropriate data structures

# Data Analysis and Information Exploitation (ADEI)

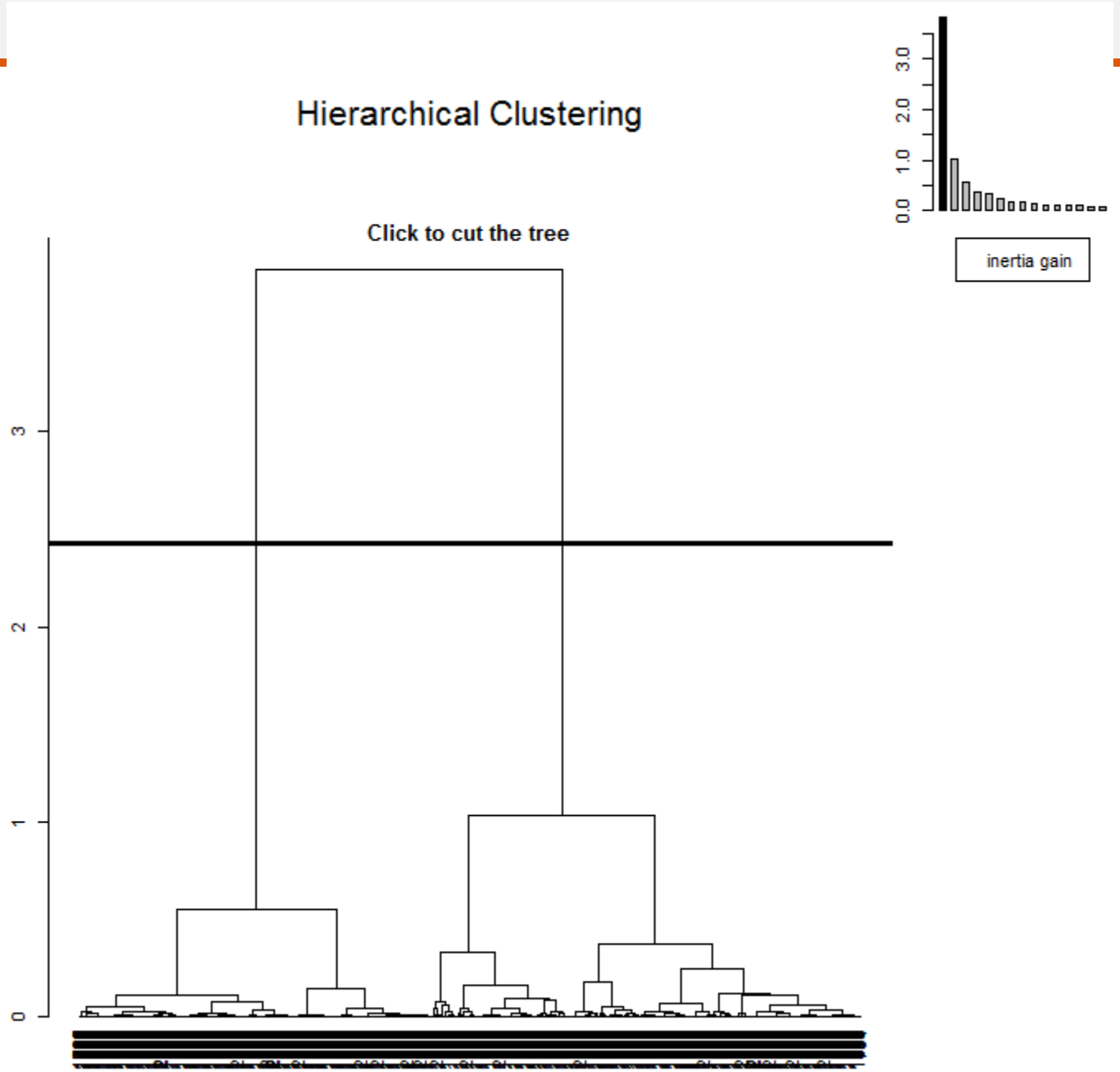
**Bachelor Degree in Informatics Engineering**
**Information System Track**

**FIB-ADEI – 6 ECTS - Course 2020-21**

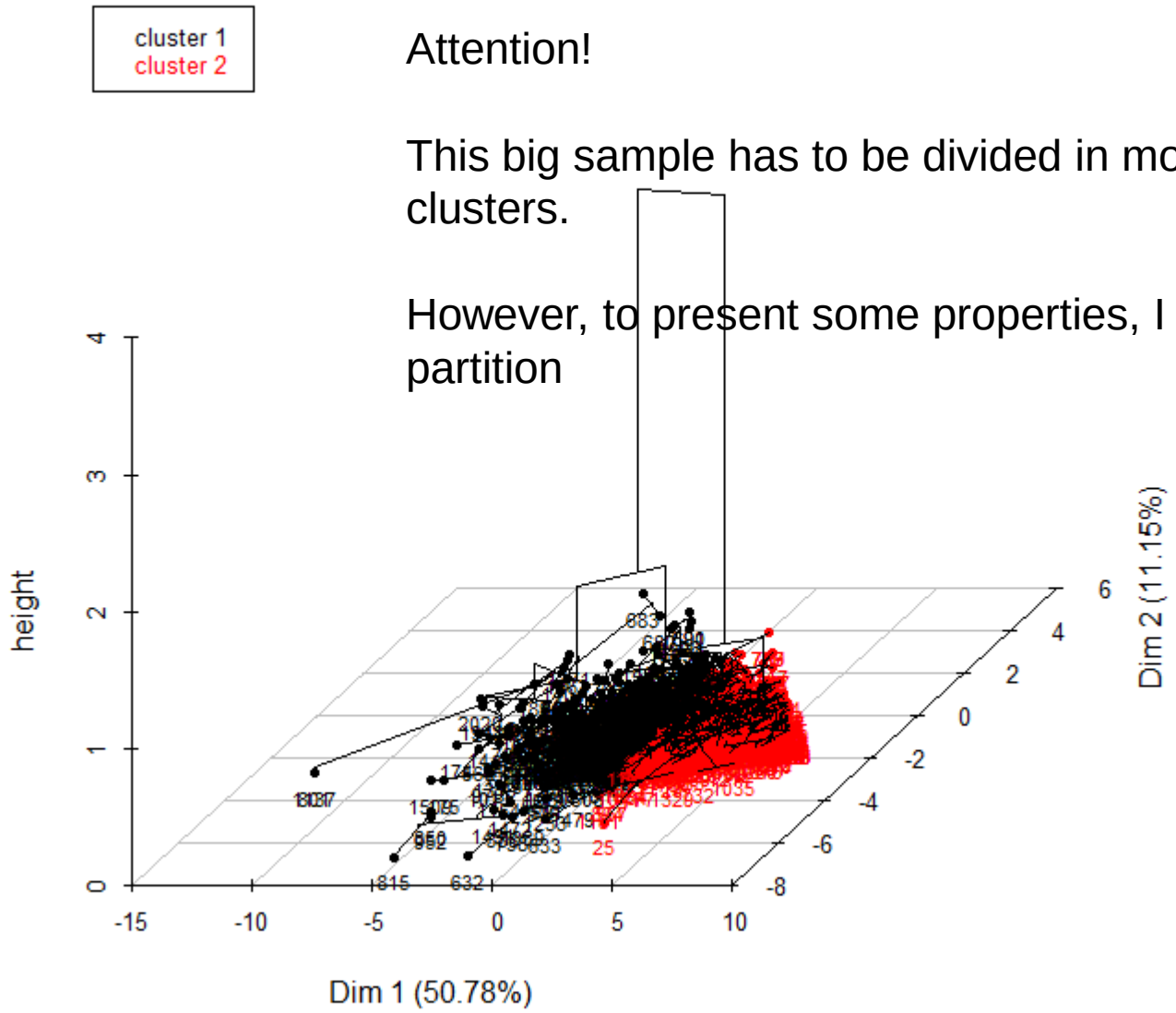
**BarcelonaTech - UPC**

# Clustering from principal components

FIB UPC UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH


Hierarchical Clustering
Click to cut the tree
0
1
2
3
0.0
1.0
2.0
3.0
inertia gain

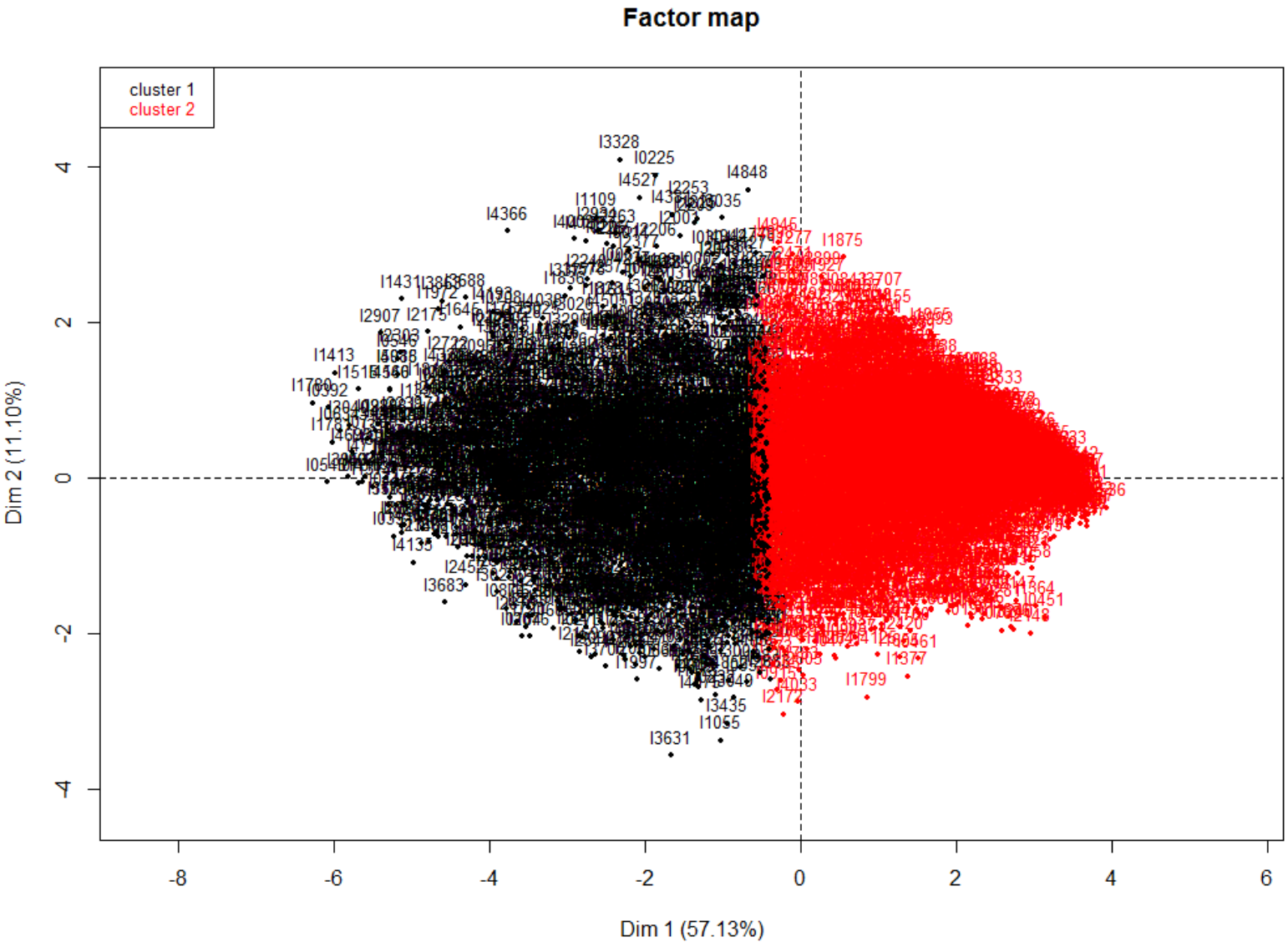
## Hierarchical clustering on the factor map




Attention!

This big sample has to be divided in more than 2 clusters.

However, to present some properties, I use this partition


Factor map
cluster 1
cluster 2
Dim 2 (11.10%)
Dim 1 (57.13%)
4
2
0
-2
-4
-8
-6
-4
-2
0
2
4
6
I3328
I0225
I4848
I4527
I2253
I1109
I4366
I1875
I1431
I2907
I2175
I1413
I4135
I3683
I1799
I2172
I3435
I1055
I3631

# Cluster description

## From the categorical variables

```
> res.hcpc$desc.var$test.chi2
                  p.value df
B1           0.000000e+00  4
Edad_Skol 3.245107e-156  8
Sex          2.257039e-12  1
```

## From the categories

```
> res.hcpc$desc.var$category
$`1`
                      Cla/Mod    Mod/Cla    Global       p.value     v.test
B1=health_poor      90.412272 48.0387163 20.706770  0.000000e+00        Inf
Edad_Skol=OldLow    66.666667 37.3917473 21.858249  5.403491e-99  21.118273
B1=health_fair      48.108926 32.3993887 26.245781  3.360483e-15   7.876734
Sex=female          42.852470 66.7345899 60.690887  1.850993e-12   7.045272
Edad_Skol=OldMedium 48.005908 16.5562914 13.440540  2.880527e-07   5.131101
Edad_Skol=MidLow    48.265896  8.5073867  6.869168  2.801679e-04   3.632979
Edad_Skol=OldHigh   46.255507  5.3489557  4.506651  2.254253e-02   2.281100
Edad_Skol=MidMedium 34.720571 14.8751910 16.696446  5.406843e-03  -2.781739
Edad_Skol=MidHigh   27.345845  5.1961284  7.405202  1.065718e-06  -4.879099
Edad_Skol=JovLow    17.801047  1.7320428  3.791940  1.345397e-10  -6.421950
Sex=male            32.979798 33.2654101 39.309113  1.850993e-12  -7.045272
Edad_Skol=JovHigh   11.673152  1.5282731  5.102243  2.449031e-23  -9.952915
B1=health_good      23.189466 16.1487519 27.139170  9.788030e-47 -14.355876
Edad_Skol=JovMedium 16.992188  8.8639837 20.329561  4.079413e-64 -16.905773
B1=health_excellent  1.694915  0.4075395  9.370657  7.632784e-93 -20.438316
B1=health_very good  7.082833  3.0056037 16.537622 1.428371e-114 -22.750203
```

# From the quantitative variables

```
> res.hcpc$desc.var$quanti.var
                 Eta2      P-value
PF_Phisica 0.30922824 0.000000e+00
RP_Role.li 0.50533202 0.000000e+00
RE_Role.li 0.29780836 0.000000e+00
SF_Social  0.46725019 0.000000e+00
….etc.


$`1`
              v.test Mean in category Overall mean sd in category Overall sd      p.value
Edad         8.061905         51.71472     49.08557       18.38630   18.49395 7.511481e-16
RE_Role.li -38.726772         44.86323     72.42412       44.61223   40.35830 0.000000e+00
PF_Phisica -39.462304         48.55833     69.92952       28.22450   30.71124 0.000000e+00
MH_Mental  -43.241226         46.73255     61.69069       17.88921   19.61691 0.000000e+00
P_Pain     -46.759448         39.82454     64.50947       24.40165   29.93736 0.000000e+00
EV_Energy  -47.369413         33.84870     51.83145       16.42762   21.52827 0.000000e+00
SF_Social  -48.508473         44.98724     64.83442       21.02165   23.20236 0.000000e+00
HP_General -48.525317         34.58839     54.26444       16.98673   22.99433 0.000000e+00
RP_Role.li -50.446527         25.01274     63.00377       34.73299   42.70719 0.000000e+00
```

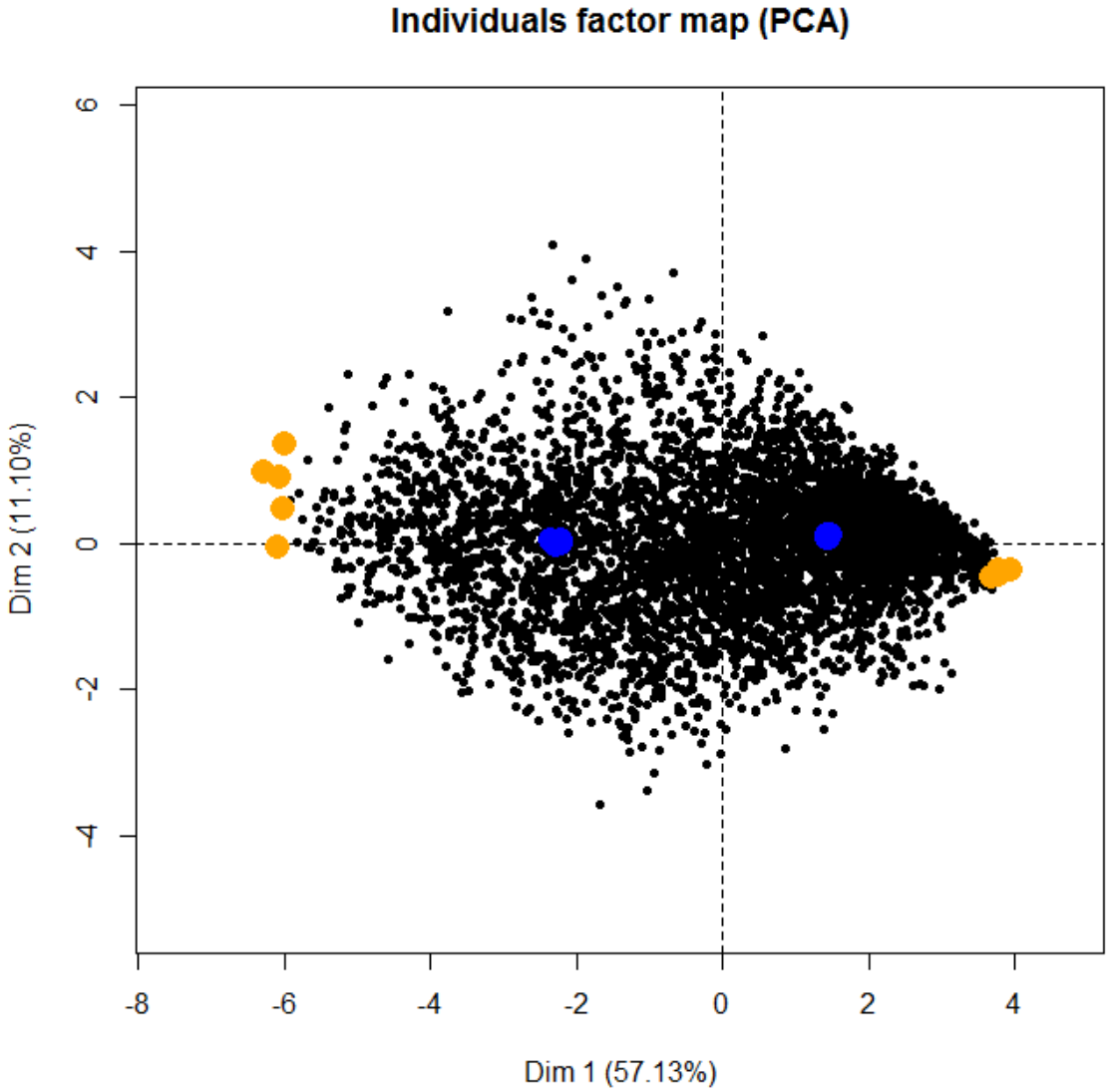
# From the individuals

```
> res.hcpc$desc.ind$para
Cluster: 1
    I3045     I2801     I3404     I4561     I2477
0.0140178 0.0454952 0.0748305 0.1016992 0.1131562
--------------------------------------------------------------------------------

Cluster: 2
     I3451      I3063      I3158      I4466      I1166
0.02458791 0.04874171 0.04874171 0.07030856 0.07354237
```

```
> res.hcpc$desc.ind$dist
Cluster: 1
   I1780    I0392    I1413    I0545    I1781
7.783213 7.566054 7.559595 7.543709 7.482944
--------------------------------------------------------------------------------
Cluster: 2
   I0536    I0977    I4932    I3460    I3806
6.219741 6.058947 6.047333 5.978893 5.978893
```

Individuals “para” and “dist” on the principal plane

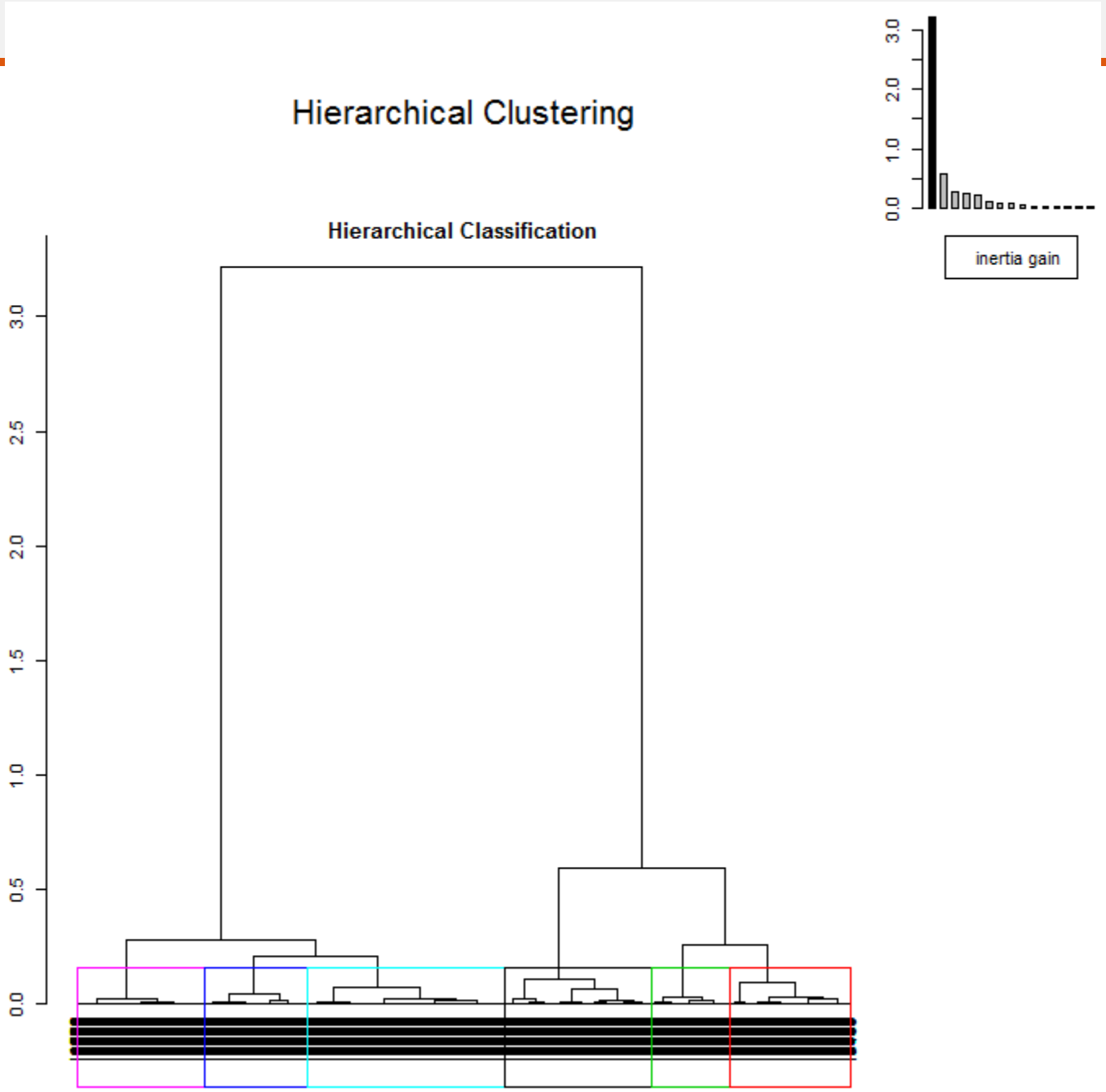


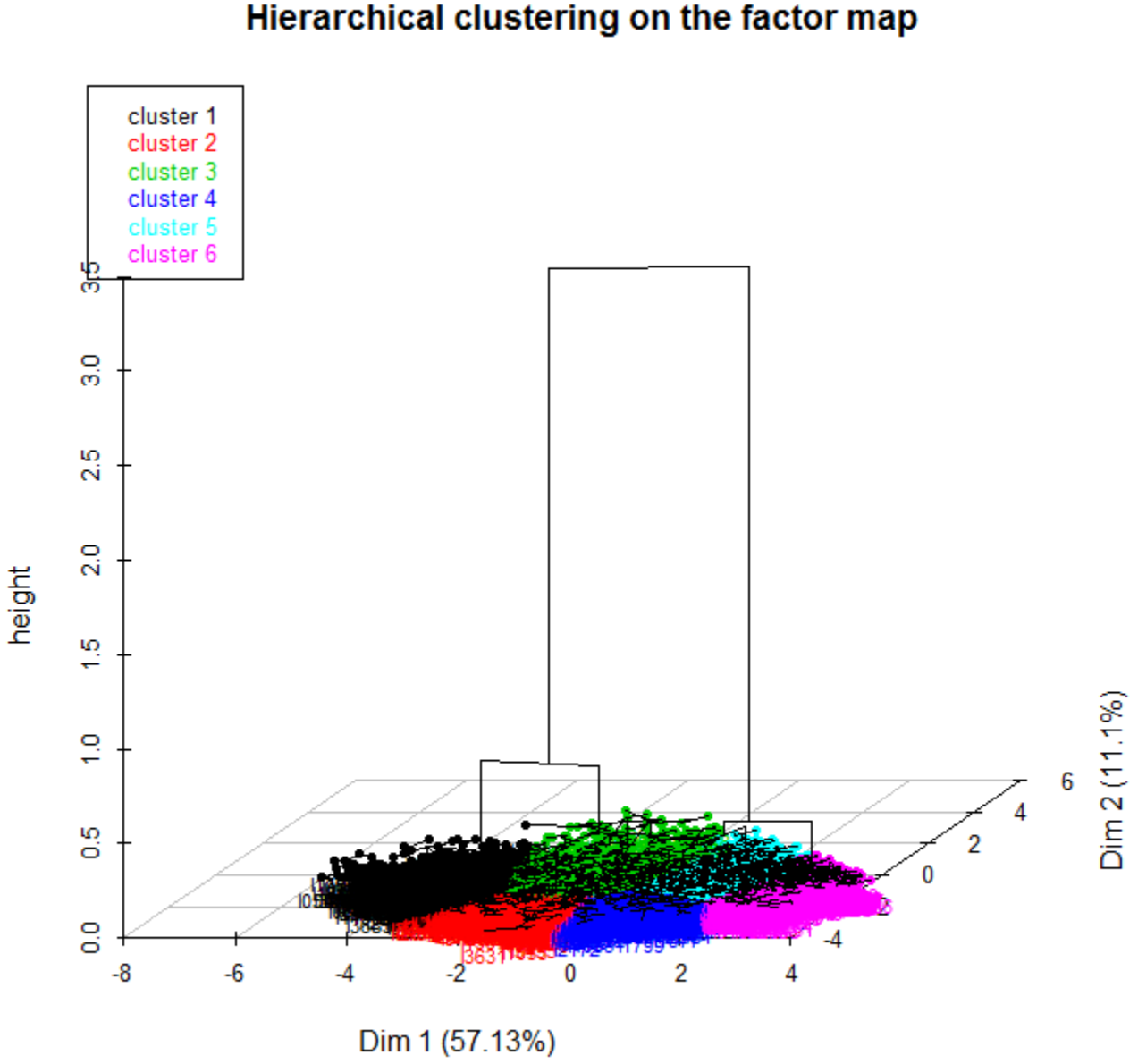
Variables values of the individuals “para” and “dist” (cluster 1)

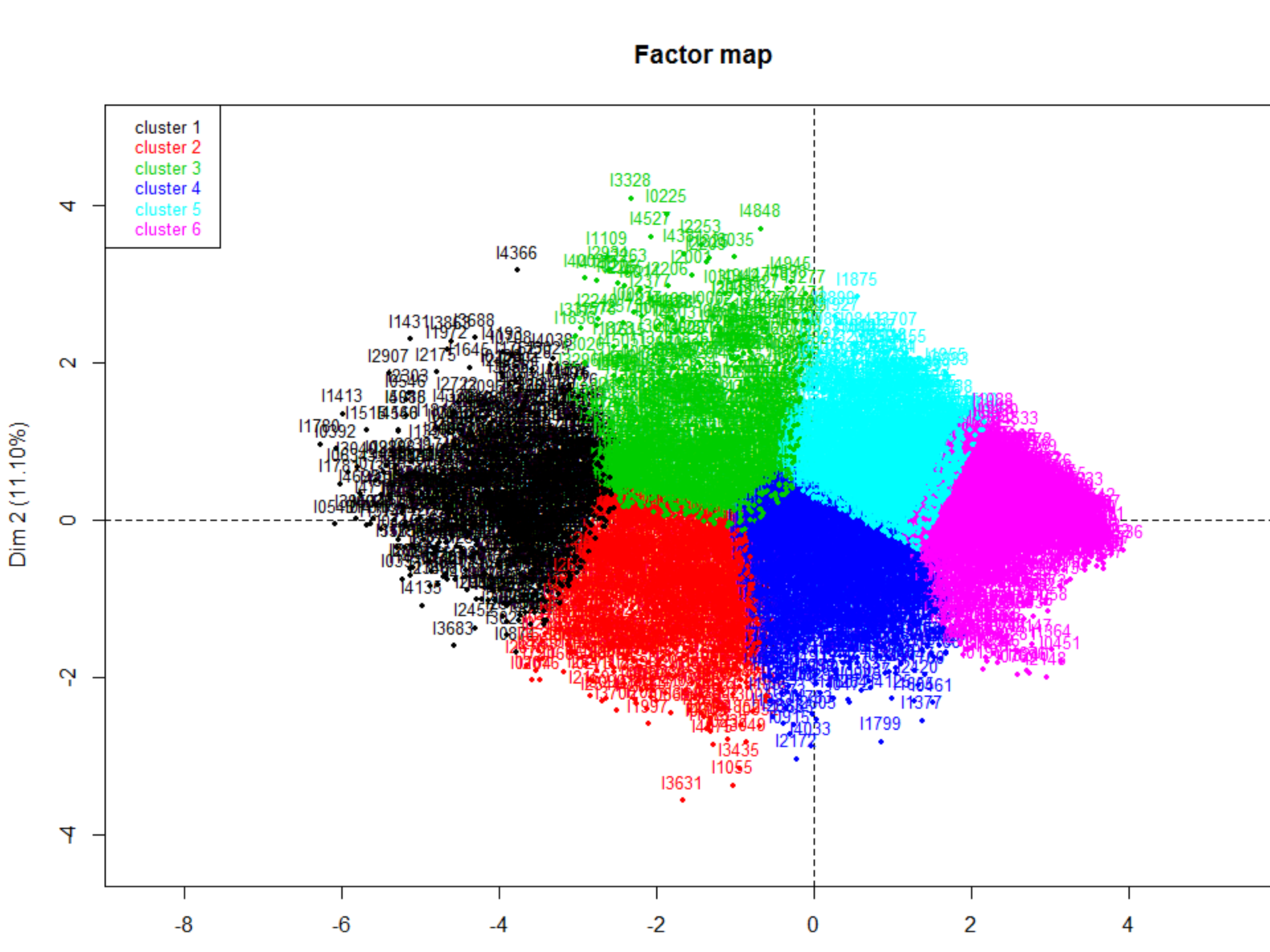
```
> res.hcpc$data.clust[which(rownames(res.hcpc$data.clust)%in%names(res.hcpc$desc.ind$para[[1]])),]
        Sex          B1 Edad PF_Phisica RP_Role.li RE_Role.li SF_Social MH_Mental EV_Energy P_Pain HP_General Edad_Skol clust
I2477 female health_poor   33         30         75      66.67     44.44        36         25  44.44         25    OldLow     1
I2801 female health_fair   27         40          0       0.00     44.44        52         50  44.44         45 OldMedium     1
I3045 female health_fair   75         35         25      66.67     44.44        28         45  55.56         30 MidMedium     1
I3404 female health_poor   87         90          0      33.33     44.44        68         25  22.22         30 MidMedium     1
I4561   male health_poor   71         50         25      33.33     44.44        56         30  55.56         25 OldMedium     1


> res.hcpc$data.clust[which(rownames(res.hcpc$data.clust)%in%names(res.hcpc$desc.ind$dist[[1]])),]
        Sex          B1 Edad PF_Phisica RP_Role.li RE_Role.li SF_Social MH_Mental EV_Energy P_Pain HP_General Edad_Skol clust
I0392 female health_poor   49         30          0          0      0.00         4          5   0.00          0    MidLow     1
I0545 female health_poor   67          0          0          0      0.00        24          0   0.00          0    OldLow     1
I1413   male health_poor   61         40          0          0      0.00         0          0  11.11          0    MidLow     1
I1780 female health_poor   26         25          0          0      0.00         0          0   0.00          0    MidLow     1
I1781 female health_poor   52          5          0          0     11.11         8          0  11.11          0 OldMedium     1
```

# Cluster description: Selection of a reasonable number of clusters


Hierarchical Clustering
Hierarchical Classification
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0.0
1.0
2.0
3.0
inertia gain


Hierarchical clustering on the factor map
cluster 1
cluster 2
cluster 3
cluster 4
cluster 5
cluster 6
height
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
-8
-6
-4
-2
0
2
4
Dim 1 (57.13%)
-4
0
2
4
6
Dim 2 (11.1%)


Factor map
cluster 1
cluster 2
cluster 3
cluster 4
cluster 5
cluster 6
Dim 2 (11.10%)
-8
-6
-4
-2
0
2
4
-4
-2
0
2
4
I3328
I0225
I4848
I4527
I2253
I1109
I4366
I1875
I1431
I2907
I2175
I1413
I1780
I4135
I3683
I3435
I1055
I3631
I1799
I1377
I2172

# Important Numerical Results

```
> dim(res.hcpc$data.clust)
[1] 5037   12

> summary(res.hcpc$data.clust[,12])
   1    2    3    4    5    6
 608  666  617  868  986 1292

> res.hcpc$call$t$within[1:5]
[1] 5.458357 2.244554 1.651811 1.369927 1.113741

> res.hcpc$call$t$inert.gain[1:5]
[1] 3.2138025 0.5927435 0.2818838 0.2561863 0.2117429

> sum(res.hcpc$call$t$inert.gain)
[1] 5.458357
```

**Cut quality:**

(res.hcpc$call$t$within[1]- res.hcpc$call$t$within[6])/ res.hcpc$call$t$within[1]

BetweenSS/TotalSS – Directed provided by k-means

# Description according to qualitative variables

```
> res.hcpc$desc.var$test.chi2
                  p.value df
B1           0.000000e+00 20
Edad_Skol 6.019521e-232 40
Sex          2.352911e-16  5
```

# Description according to qualitative factor levels

```
$`1`
                      Cla/Mod    Mod/Cla    Global       p.value     v.test
B1=health_poor      44.007670 75.4934211 20.706770 9.799110e-221  31.714522
Edad_Skol=OldLow    27.429609 49.6710526 21.858249  9.592124e-60  16.301741
Sex=female          13.052012 65.6250000 60.690887  7.578879e-03   2.670277
Edad_Skol=OldMedium 14.918759 16.6118421 13.440540  1.692725e-02   2.388284

$`2`
                      Cla/Mod    Mod/Cla    Global       p.value     v.test
B1=health_poor      33.0776606 51.8018018 20.706770 8.269209e-84  19.396447
Edad_Skol=OldLow    25.8855586 42.7927928 21.858249 2.536287e-39  13.119792
Edad_Skol=OldMedium 19.9409158 20.2702703 13.440540 1.324499e-07   5.275423
B1=health_fair      17.3222390 34.3843844 26.245781 5.677456e-07   5.001878
Sex=female          15.0474321 69.0690691 60.690887 1.524938e-06   4.807940
Edad_Skol=OldHigh   20.7048458  7.0570571  4.506651 1.354693e-03   3.204134


$`3`
                      Cla/Mod    Mod/Cla    Global      p.value    v.test
B1=health_fair      19.818457 42.4635332 26.245781 7.805536e-21  9.362251
Sex=female          13.444553 66.6126418 60.690887 1.205395e-03  3.237600
```

```
$`4`
                       Cla/Mod   Mod/Cla    Global      p.value    v.test
B1=health_fair       26.248109 39.976959 26.245781 1.607979e-22  9.763932
B1=health_good       21.872714 34.447005 27.139170 1.791435e-07  5.219772
Edad_Skol=OldMedium  23.781388 18.548387 13.440540 2.904027e-06  4.677494
Edad_Skol=OldHigh    23.788546  6.221198  4.506651 1.003363e-02  2.574668


$`5`
                       Cla/Mod   Mod/Cla    Global      p.value     v.test
B1=health_good       30.577908 42.393509 27.139170 3.289184e-31  11.619274
B1=health_very good  32.292917 27.281947 16.537622 6.309239e-22   9.624370
Edad_Skol=JovMedium  30.468750 31.643002 20.329561 3.533054e-21   9.445620
Edad_Skol=JovHigh    30.350195  7.910751  5.102243 2.261012e-05   4.237422
Edad_Skol=JovLow     26.701571  5.172414  3.791940 1.460394e-02   2.442054
Edad_Skol=MidHigh    23.860590  9.026369  7.405202 3.363408e-02   2.124431

$`6`
                        Cla/Mod   Mod/Cla    Global       p.value     v.test
B1=health_excellent  74.5762712 27.244582  9.370657 3.763011e-123  23.601012
B1=health_very good  49.6998800 32.043344 16.537622  3.668735e-61  16.500003
Edad_Skol=JovMedium  40.9179688 32.430341 20.329561  1.168573e-33  12.091690
Sex=male             31.6666667 48.529412 39.309113  5.507268e-15   7.814749
Edad_Skol=JovHigh    43.5797665  8.668731  5.102243  1.538311e-10   6.401526
Edad_Skol=JovLow     42.9319372  6.346749  3.791940  1.257359e-07   5.284955
Edad_Skol=MidHigh    30.5630027  8.823529  7.405202  2.624464e-02   2.222572
```

# Description according to numeric variables

```
> res.hcpc$desc.var$quanti.var
                Eta2 P-value
PF_Phisica 0.5540974       0
RP_Role.li 0.6037848       0
RE_Role.li 0.5429940       0
SF_Social  0.5979396       0
MH_Mental  0.6145947       0
EV_Energy  0.6281132       0
P_Pain     0.5805753       0
HP_General 0.6049045       0
```

# Description according to numeric variables

$`1`

| | v.test | Mean in category | Overall mean | sd in category | Overall sd | p.value |
|---|---|---|---|---|---|---|
| PF_Phisica | -30.08759 | 34.78618 | 69.92952 | 24.70291 | 30.71124 | 7.041828e-199 |
| RP_Role.li | -33.97905 | 7.81250 | 63.00377 | 19.03903 | 42.70719 | 4.543193e-253 |
| P_Pain | -34.73364 | 24.96169 | 64.50947 | 17.73485 | 29.93736 | 2.447572e-264 |
| HP_General | -35.40172 | 23.30428 | 54.26444 | 13.25947 | 22.99433 | 1.606760e-274 |
| EV_Energy | -37.63124 | 21.01974 | 51.83145 | 13.06095 | 21.52827 | 0.000000e+00 |
| MH_Mental | -37.69660 | 33.56579 | 61.69069 | 14.74773 | 19.61691 | 0.000000e+00 |
| RE_Role.li | -37.75437 | 14.47347 | 72.42412 | 30.20910 | 40.35830 | 0.000000e+00 |
| SF_Social | -39.19922 | 30.24304 | 64.83442 | 18.30446 | 23.20236 | 0.000000e+00 |

$`2`

| | v.test | Mean in category | Overall mean | sd in category | Overall sd | p.value |
|---|---|---|---|---|---|---|
| RE_Role.li | 3.262450 | 77.17733 | 72.42412 | 36.80135 | 40.35830 | 1.104536e-03 |
| MH_Mental | -4.711168 | 58.35435 | 61.69069 | 13.53121 | 19.61691 | 2.463012e-06 |
| SF_Social | -14.443856 | 52.73608 | 64.83442 | 18.77967 | 23.20236 | 2.740714e-47 |
| EV_Energy | -16.411812 | 39.07658 | 51.83145 | 14.43413 | 21.52827 | 1.574352e-60 |
| HP_General | -24.247403 | 34.13664 | 54.26444 | 14.82676 | 22.99433 | 7.042685e-130 |
| P_Pain | -25.606539 | 36.83529 | 64.50947 | 21.05721 | 29.93736 | 1.290045e-144 |
| PF_Phisica | -28.200871 | 38.66366 | 69.92952 | 22.35936 | 30.71124 | 5.705482e-175 |
| RP_Role.li | -28.667247 | 18.80631 | 63.00377 | 30.54377 | 42.70719 | 9.772936e-181 |

$`3`

| | v.test | Mean in category | Overall mean | sd in category | Overall sd | p.value |
|---|---|---|---|---|---|---|
| PF_Phisica | 4.573390 | 75.22690 | 69.92952 | 21.00854 | 30.71124 | 4.798949e-06 |
| P_Pain | -4.500828 | 59.42750 | 64.50947 | 23.00276 | 29.93736 | 6.768936e-06 |
| RP_Role.li | -7.645505 | 50.68882 | 63.00377 | 38.75247 | 42.70719 | 2.081264e-14 |
| HP_General | -9.619138 | 45.92220 | 54.26444 | 16.07661 | 22.99433 | 6.638531e-22 |
| EV_Energy | -16.527629 | 38.41167 | 51.83145 | 13.69342 | 21.52827 | 2.320971e-61 |
| SF_Social | -18.979137 | 48.22577 | 64.83442 | 17.71799 | 23.20236 | 2.537308e-80 |
| MH_Mental | -24.348605 | 43.67585 | 61.69069 | 14.20201 | 19.61691 | 5.997935e-131 |
| RE_Role.li | -27.704429 | 30.25371 | 72.42412 | 36.49884 | 40.35830 | 6.174987e-169 |

# Description according to numeric variables

$`4`

| | v.test | Mean in category | Overall mean | sd in category | Overall sd | p.value |
|---|---|---|---|---|---|---|
| RE_Role.li | 16.702296 | 93.24131 | 72.42412 | 19.86619 | 40.35830 | 1.261168e-62 |
| MH_Mental | 12.719324 | 69.39631 | 61.69069 | 12.08360 | 19.61691 | 4.618379e-37 |
| SF_Social | 9.955742 | 71.96819 | 64.83442 | 14.82190 | 23.20236 | 2.380394e-23 |
| EV_Energy | 7.356424 | 56.72235 | 51.83145 | 13.94939 | 21.52827 | 1.889030e-13 |
| P_Pain | -5.267258 | 59.63967 | 64.50947 | 22.99663 | 29.93736 | 1.384765e-07 |
| PF_Phisica | -13.001954 | 57.59793 | 69.92952 | 27.04465 | 30.71124 | 1.192560e-38 |

$`5`

| | v.test | Mean in category | Overall mean | sd in category | Overall sd | p.value |
|---|---|---|---|---|---|---|
| PF_Phisica | 24.631852 | 91.53651 | 69.92952 | 11.95823 | 30.71124 | 5.759604e-134 |
| RP_Role.li | 23.490443 | 91.65822 | 63.00377 | 19.32596 | 42.70719 | 5.107586e-122 |
| P_Pain | 20.486659 | 82.02746 | 64.50947 | 17.77253 | 29.93736 | 2.831708e-93 |
| HP_General | 15.062490 | 64.15720 | 54.26444 | 13.84032 | 22.99433 | 2.858148e-51 |
| SF_Social | 10.521685 | 71.80738 | 64.83442 | 14.15020 | 23.20236 | 6.863536e-26 |
| RE_Role.li | 5.446027 | 78.70199 | 72.42412 | 32.67579 | 40.35830 | 5.150727e-08 |
| EV_Energy | 4.031342 | 54.31034 | 51.83145 | 11.99565 | 21.52827 | 5.545920e-05 |

$`6`

| | v.test | Mean in category | Overall mean | sd in category | Overall sd | p.value |
|---|---|---|---|---|---|---|
| EV_Energy | 43.18690 | 74.13700 | 51.83145 | 12.417304 | 21.52827 | 0.000000e+00 |
| MH_Mental | 39.69956 | 80.37461 | 61.69069 | 9.713970 | 19.61691 | 0.000000e+00 |
| HP_General | 39.66114 | 76.14396 | 54.26444 | 13.436735 | 22.99433 | 0.000000e+00 |
| SF_Social | 36.52576 | 85.16657 | 64.83442 | 7.352870 | 23.20236 | 4.326001e-292 |
| P_Pain | 35.09416 | 89.71525 | 64.50947 | 15.488867 | 29.93736 | 8.275597e-270 |
| RP_Role.li | 33.12422 | 96.94272 | 63.00377 | 11.939403 | 42.70719 | 1.331644e-240 |
| PF_Phisica | 29.75074 | 91.84985 | 69.92952 | 16.836237 | 30.71124 | 1.696340e-194 |
| RE_Role.li | 27.04154 | 98.60692 | 72.42412 | 7.632298 | 40.35830 | 4.802356e-161 |