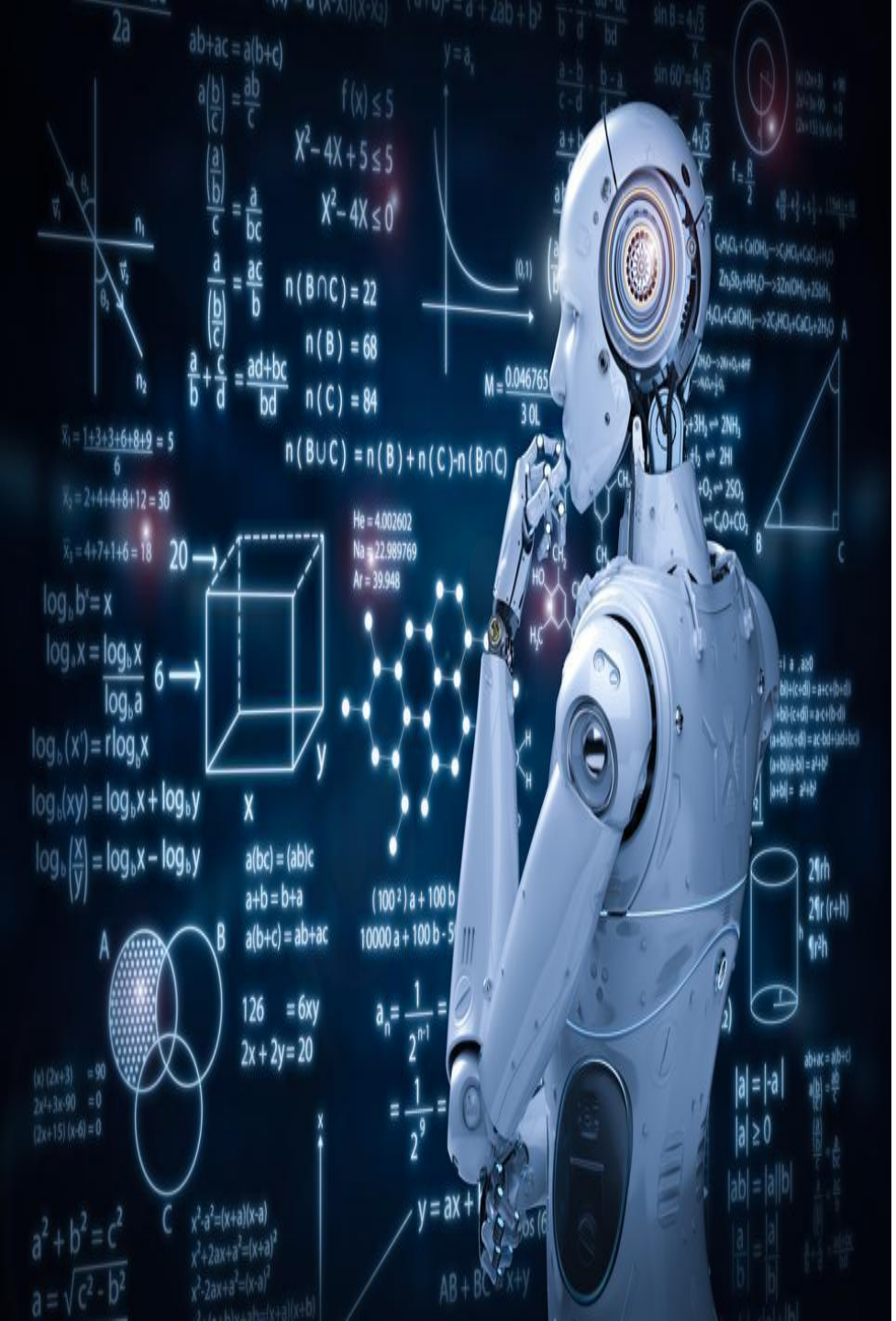


Practical Machine Learning

Day 11: SEP23 DBDA

Kiran Waghmare

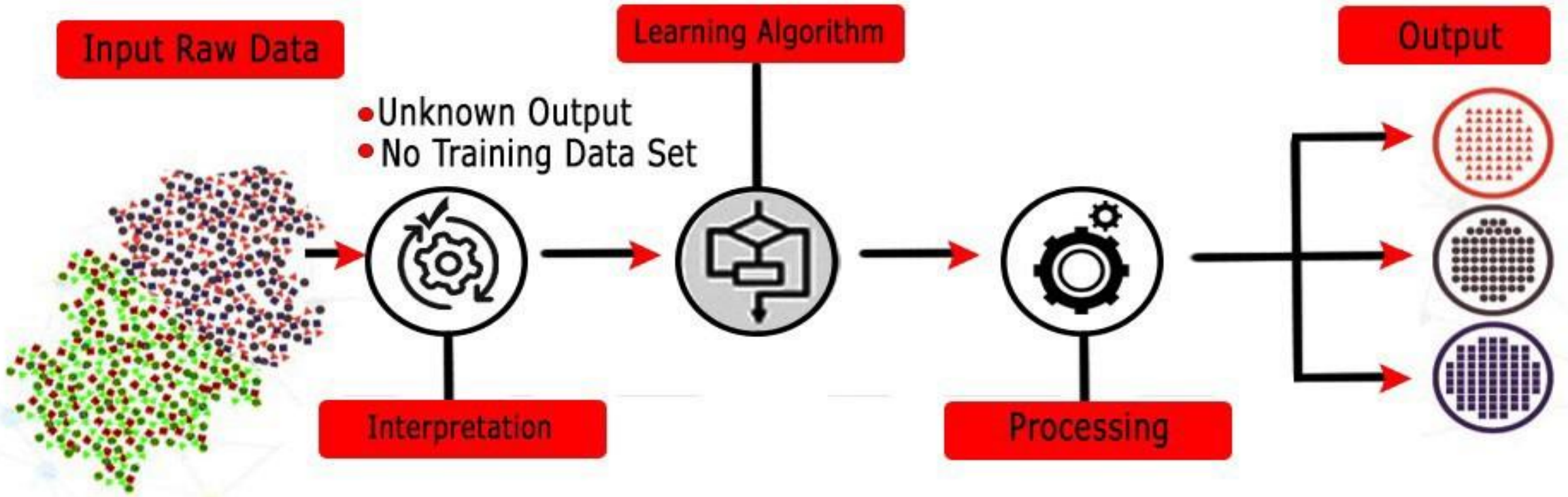
Agenda

- Clustering
- K-Means
- Hierarchical
- DB-SCAN

Machine learning:

- **Supervised vs Unsupervised.**
  - ***Supervised learning*** - the presence of the outcome variable is available to guide the learning process.
    - there **must** be a training data set in which the solution is already known.
  - ***Unsupervised learning*** - the outcomes are unknown.
    - cluster the data to reveal meaningful partitions and hierarchies

Unsupervised Learning



Clustering

Clustering:

- Unsupervised learning
- Requires data, but no labels
- Detect patterns e.g. in
  - Group emails or search results
  - Customer shopping patterns
  - Regions of images
- Useful when don't know what you're looking for
- But: can get gibberish

Machine Learning: Clustering

By color

By shape

By size

etc...

Cluster by Type

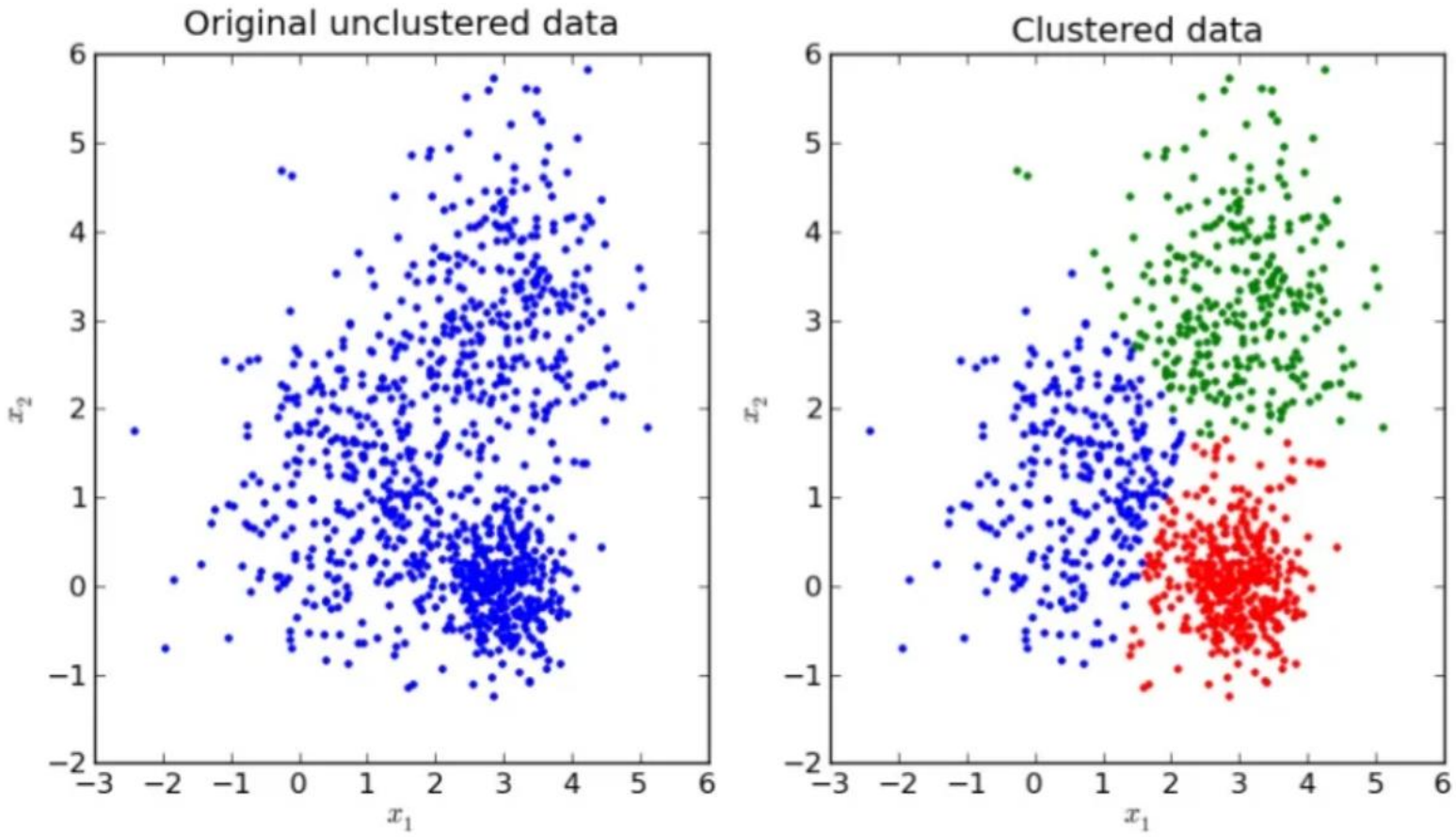
Clustering:

- Clustering is the task of gathering samples into groups of similar samples according to some predefined similarity or dissimilarity measure

sample

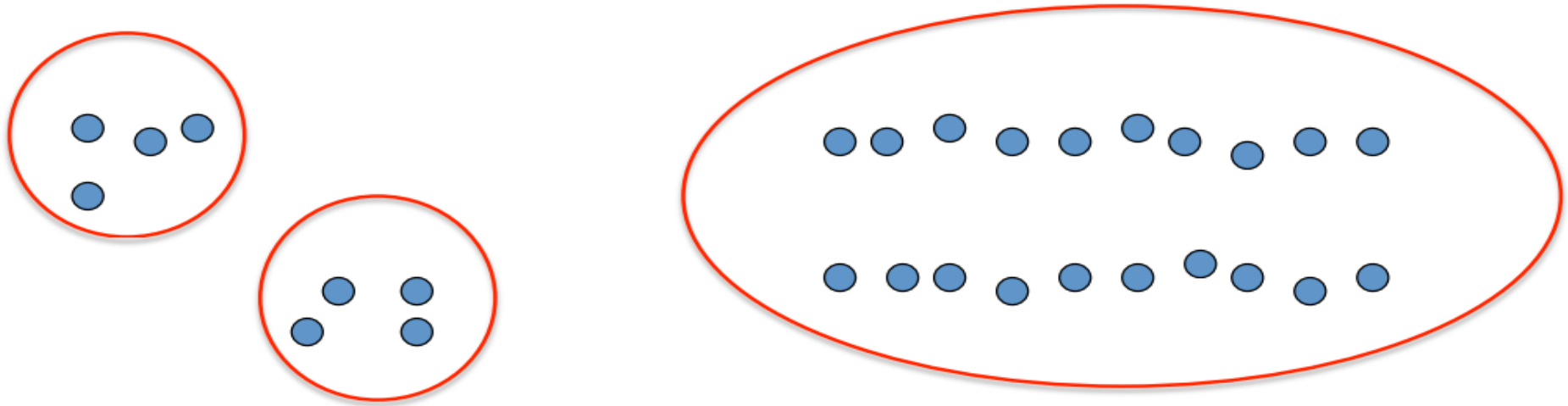
Cluster/group

- In this case clustering is carried out using the Euclidean distance as a measure.



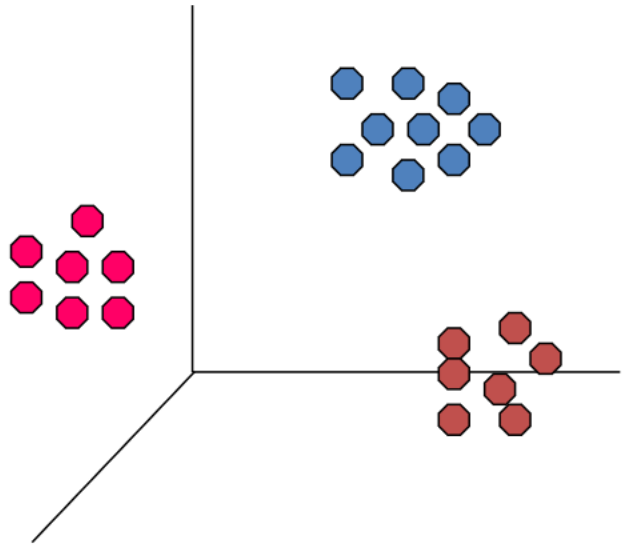
Clustering

- Basic idea: group together similar instances
- Example: 2D point patterns



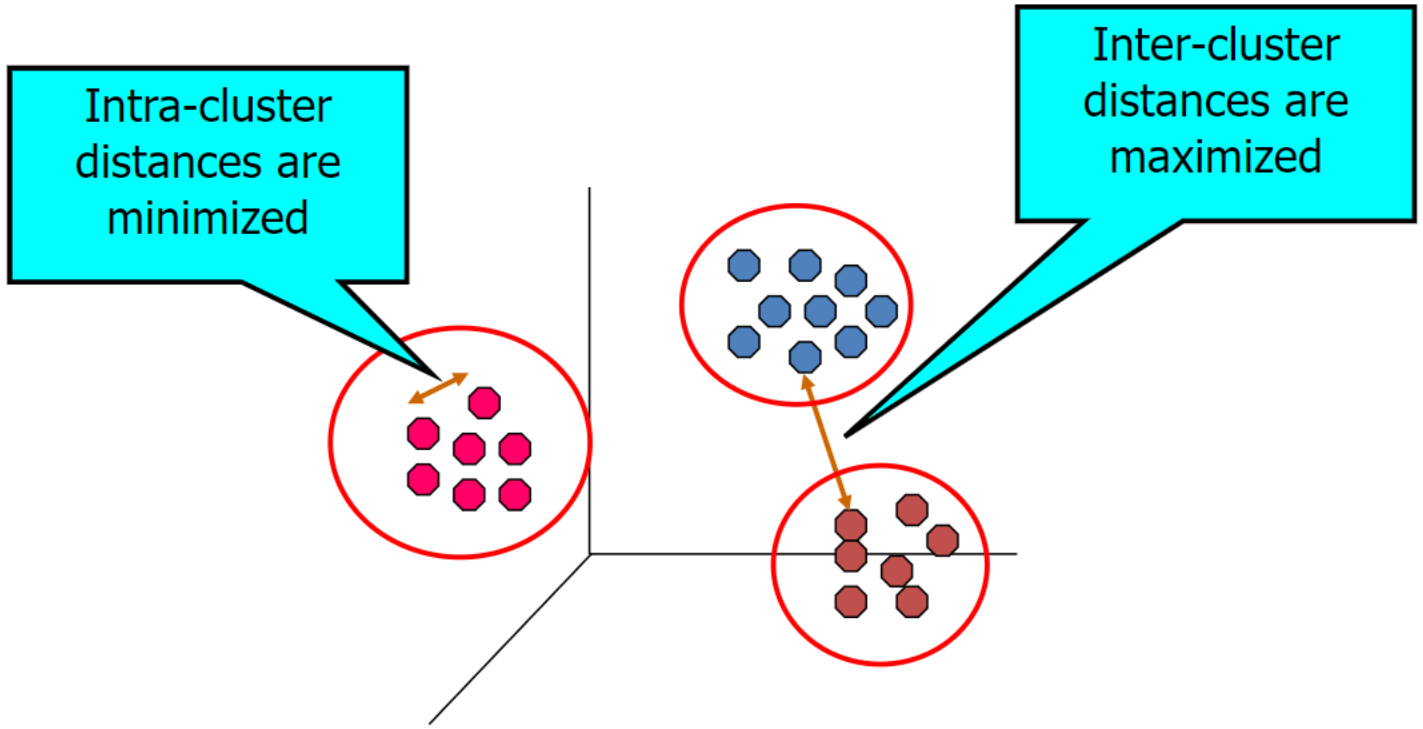
What is clustering?

- 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**



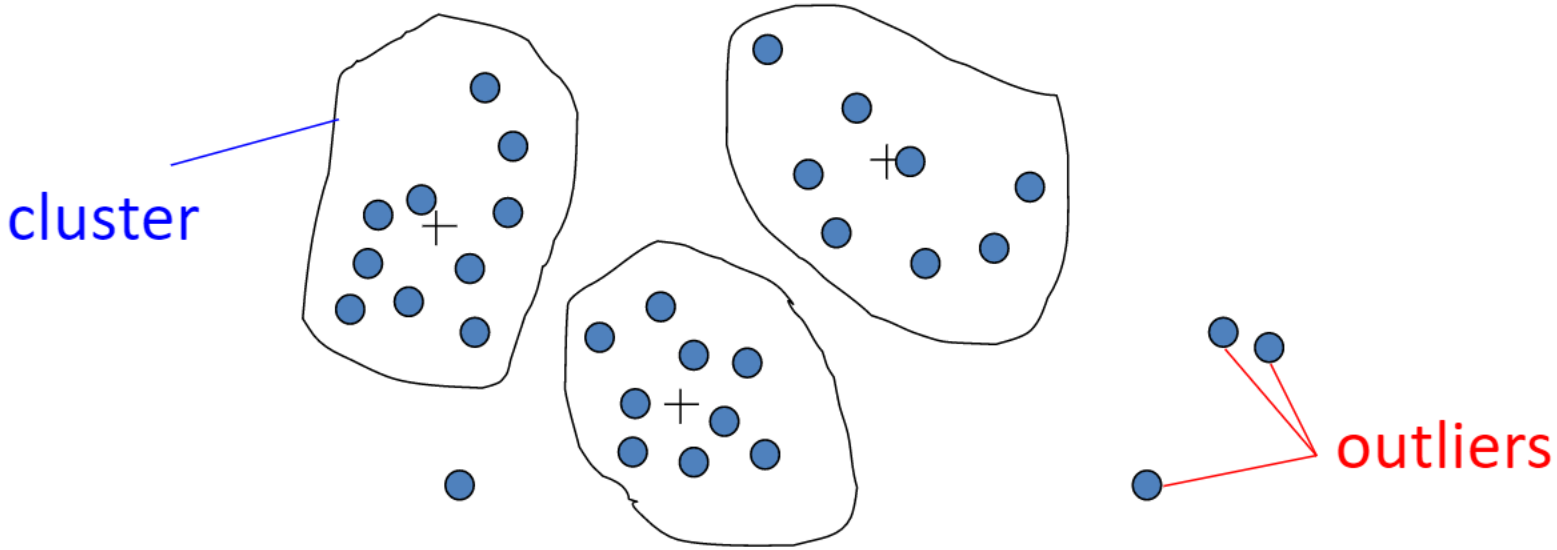
What is clustering?

- 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**

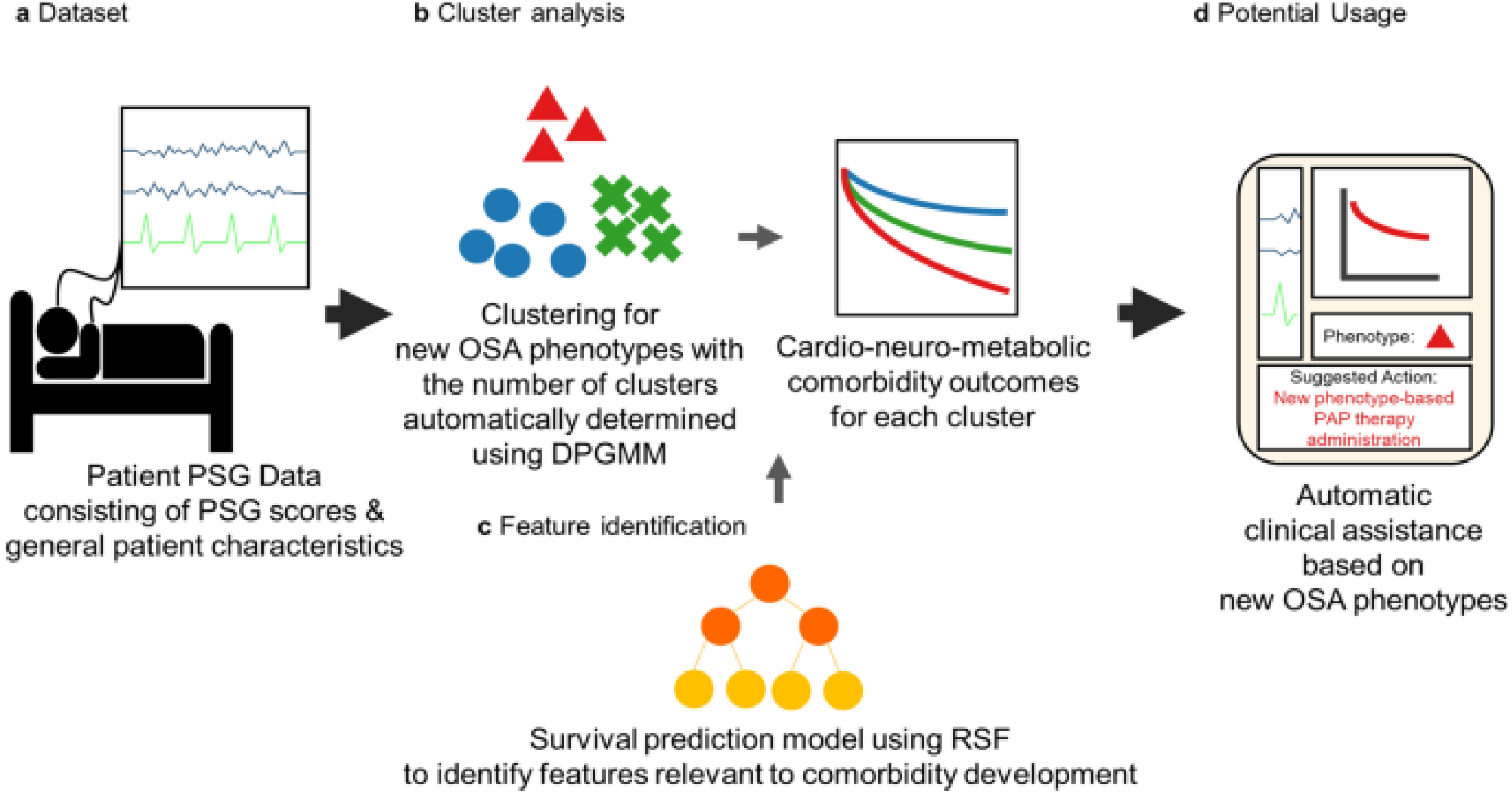


Outliers

- **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)


a Dataset
b Cluster analysis
d Potential Usage
Patient PSG Data
consisting of PSG scores &
general patient characteristics
Clustering for
new OSA phenotypes with
the number of clusters
automatically determined
using DPGMM
Cardio-neuro-metabolic
comorbidity outcomes
for each cluster
Phenotype:
Suggested Action:
New phenotype-based
PAP therapy
administration
Automatic
clinical assistance
based on
new OSA phenotypes
c Feature identification
Survival prediction model using RSF
to identify features relevant to comorbidity development

Feature Extraction → **Graph Contruction** → **Graph Embedding** → **Clustering**

- i) L2-norm
- ii) PCA
  - w/ optional ICA BSS
- iii) UFL
  - RICA
  - SFT

- i) Similarity graph
  - scaling factor
  - similarities
  - kNN criterion

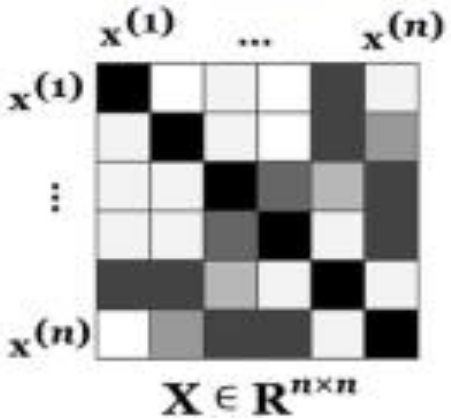
- i) Eigendecomp.
- ii) GNMF
- iii) ICA
  - used as BSS

- i) K-means
  - check clusters
  - NMI, ACC

SFT features

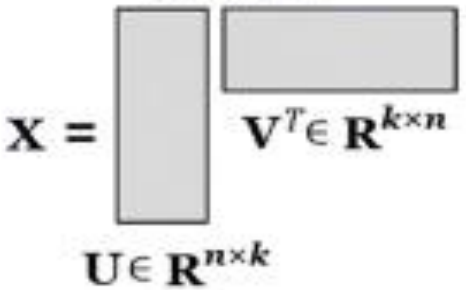
→

Similarity matrix



→

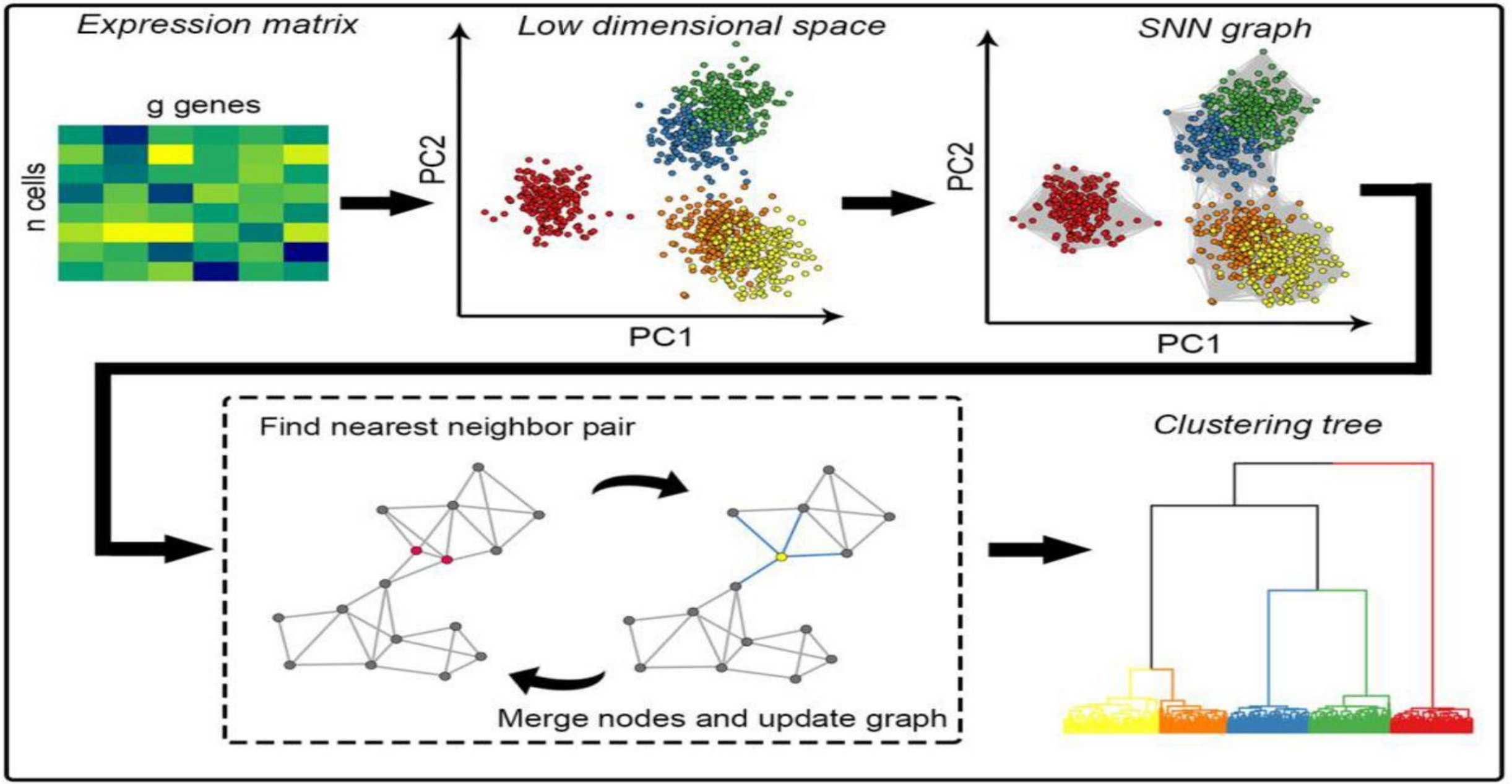
GNMF



→

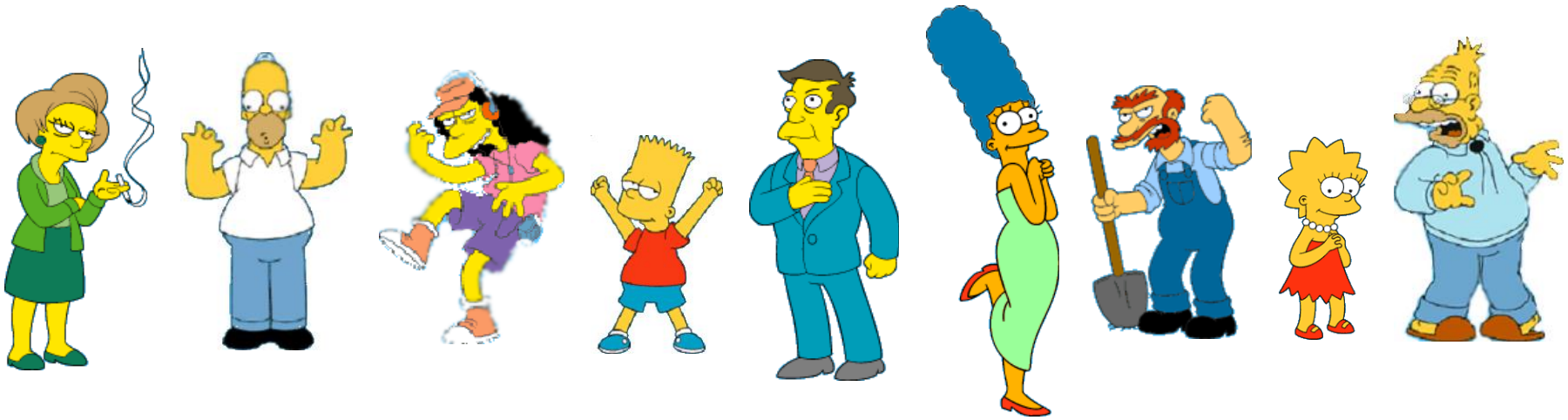
Clusters

Workflow of HGC



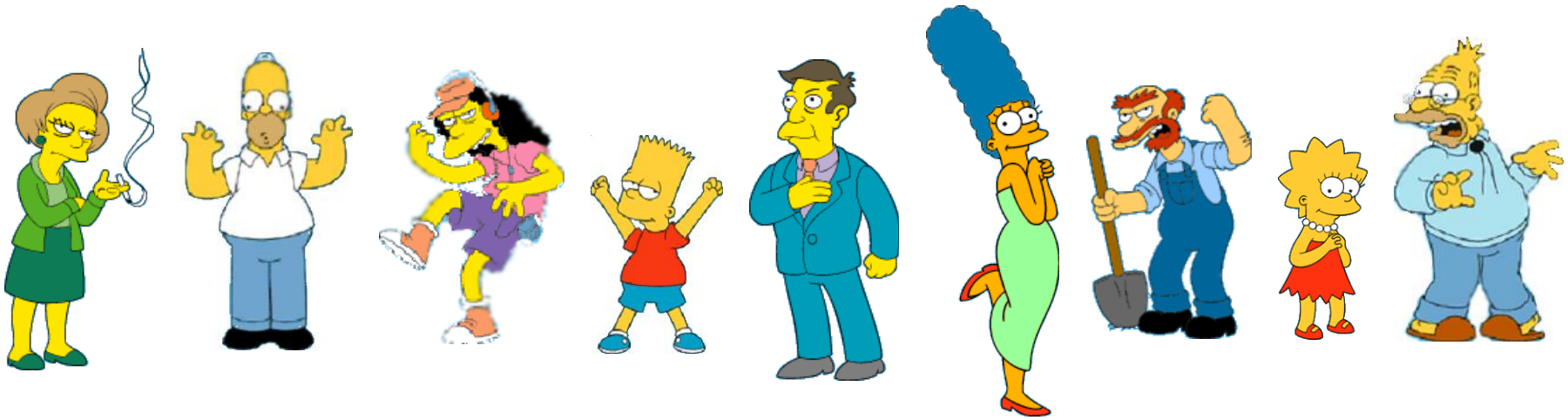
What is a natural grouping of these objects?

Slide from Eamonn Keogh

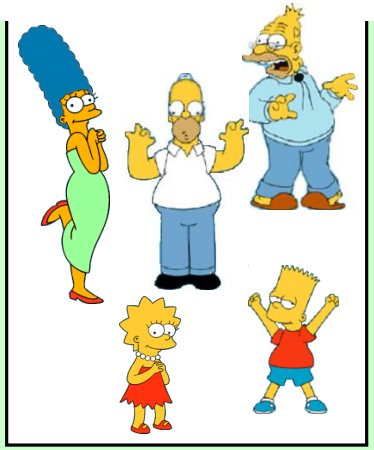


What is a natural grouping of these objects?

Slide from Eamonn Keogh

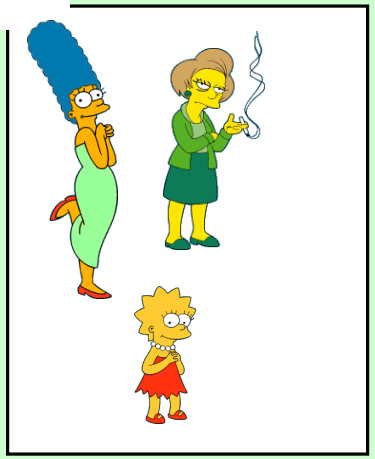


Clustering is subjective



Simpson's Family

School Employees



Females

Males

What is Similarity?

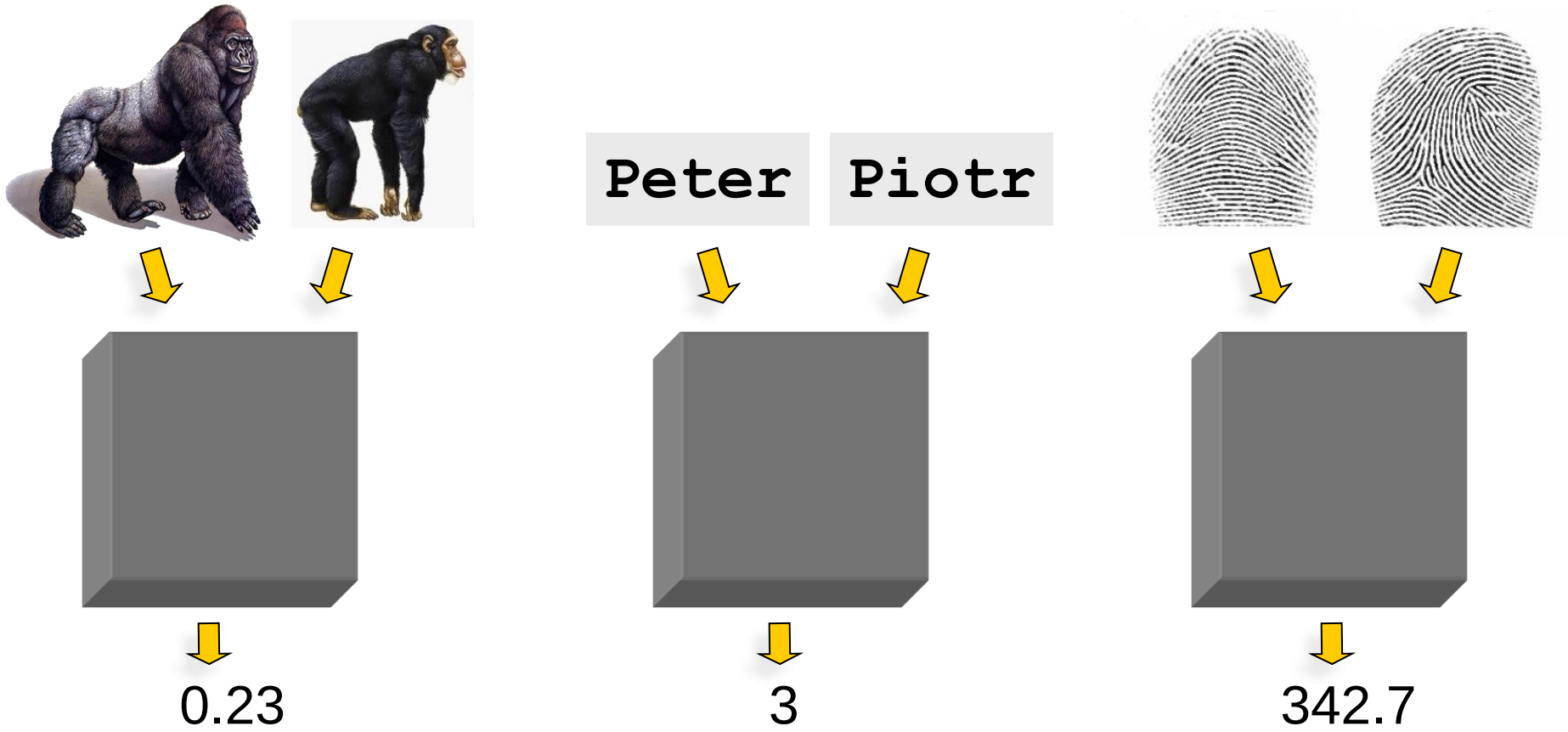
Slide based on one by Eamonn Keogh

Similarity is hard to define, but…
"*We know it when we see it*"

Defining Distance Measures

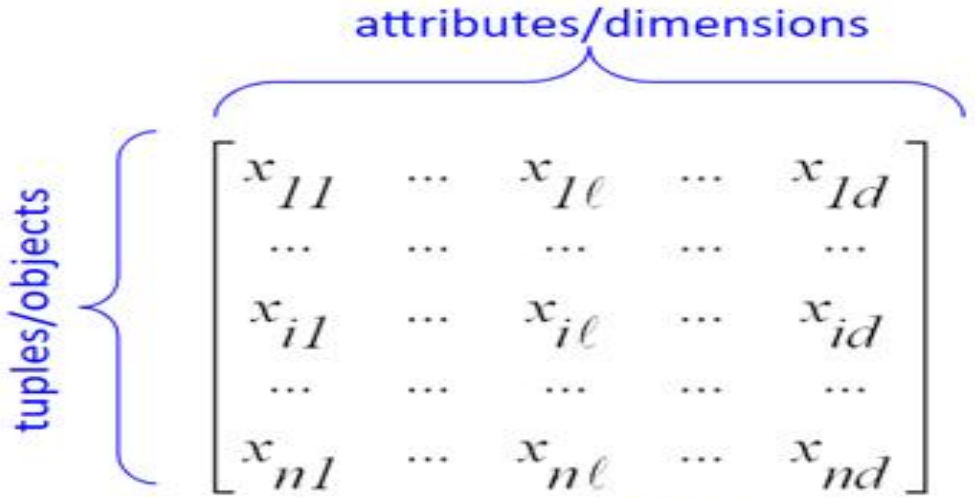
Slide from Eamonn Keogh

Definition: Let O_1 and O_2 be two objects from the universe of possible objects. The distance (dissimilarity) between O_1 and O_2 is a real number denoted by $D(O_1,O_2)$

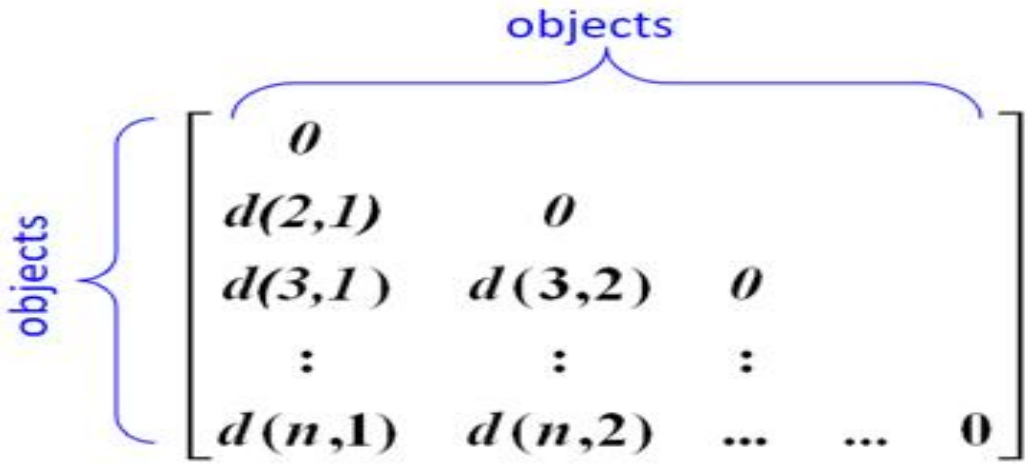


Data Structures

- *data* matrix



- *Distance* matrix



Distance functions for real-valued vectors

- L_p norms or *Minkowski distance*:

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

where ***p*** is a positive integer

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

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

Distance functions for real-valued vectors

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

$$d(x,y)=\sqrt{(|x_1-y_1|^2+|x_2-y_2|^2+\ldots+|x_d-y_d|^2)}$$

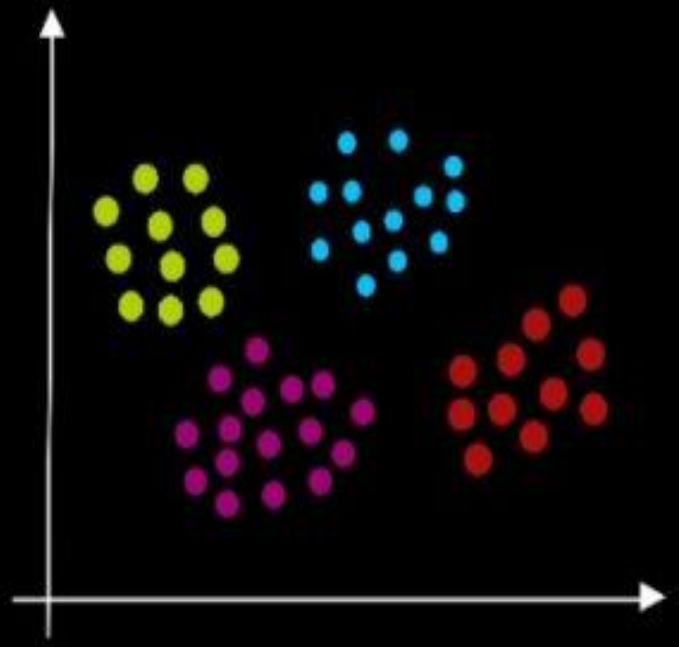
- Also one can use **weighted distance** :

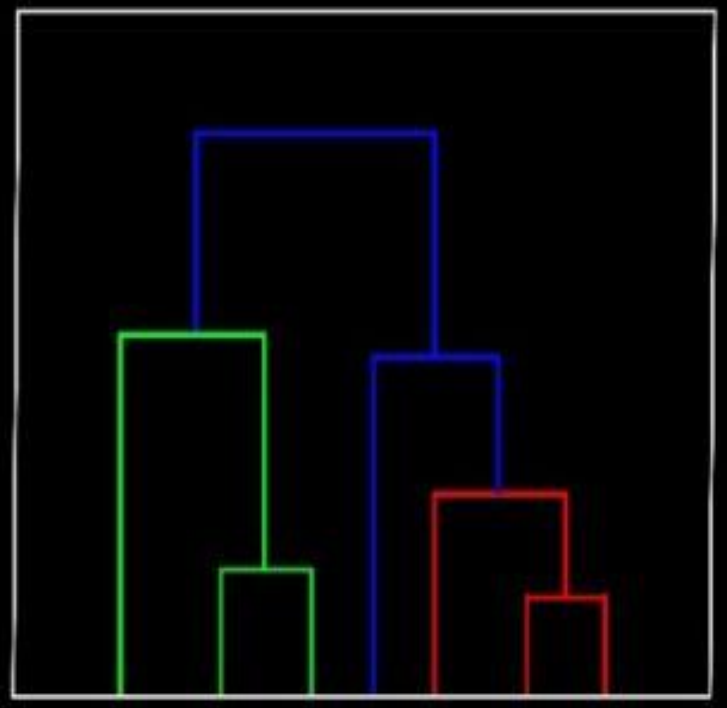
$$d(x,y)=\sqrt{(w_1|x_1-x_1|^2+w_2|x_2-x_2|^2+\ldots+w_d|x_d-y_d|^2)}$$

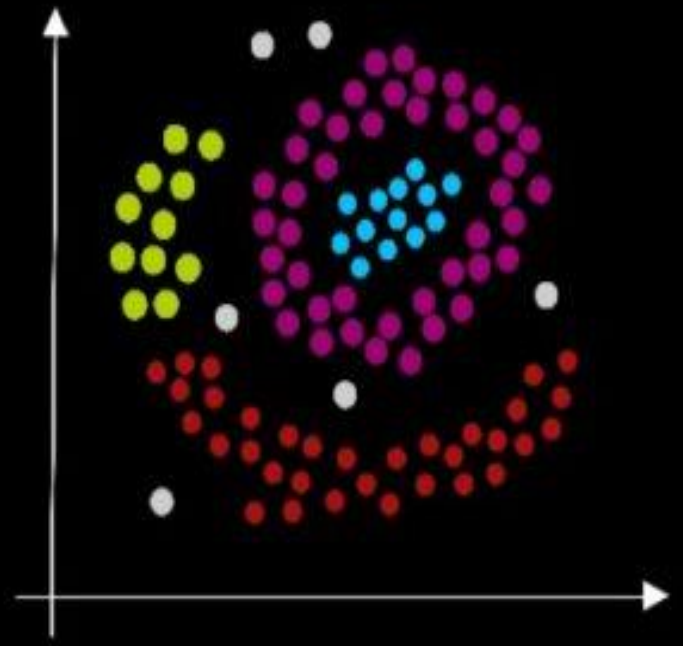
$$d(x,y)=w_1|x_1-y_1|+w_2|x_2-y_2|+\ldots+w_d|x_d-y_d|$$

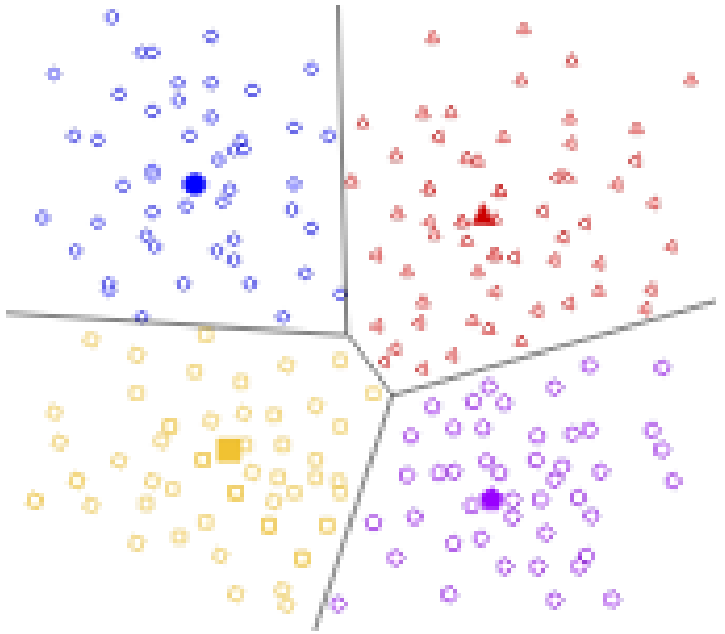
- Very often **L_p^p** is used instead of L_p (why?)

CLUSTERING IN MACHINE LEARNING

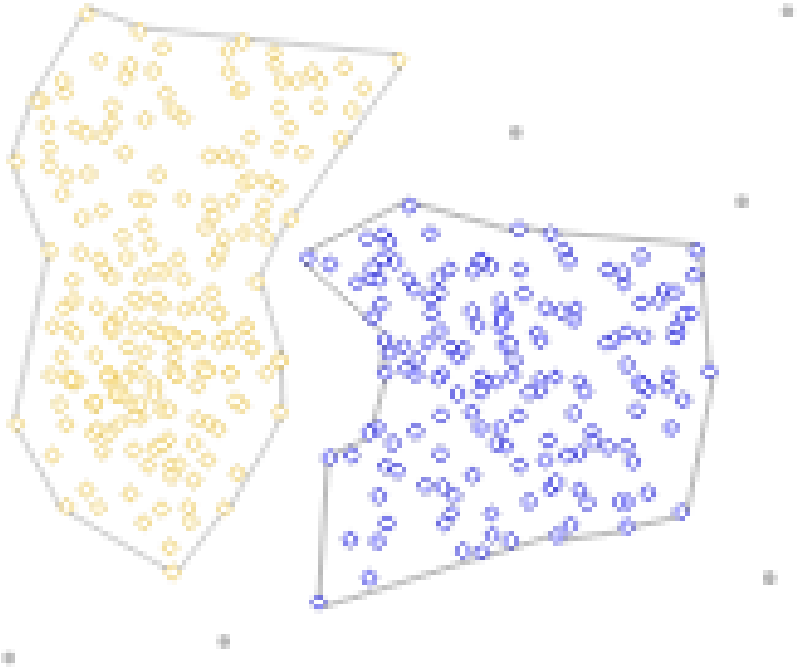




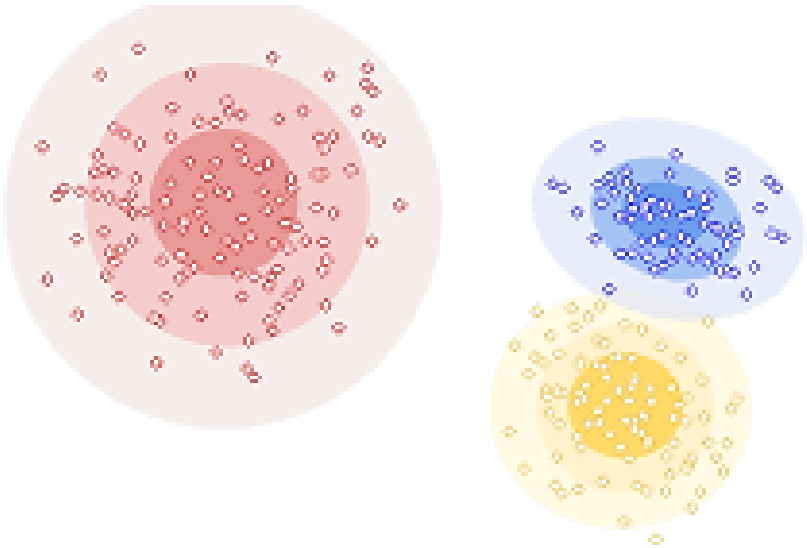




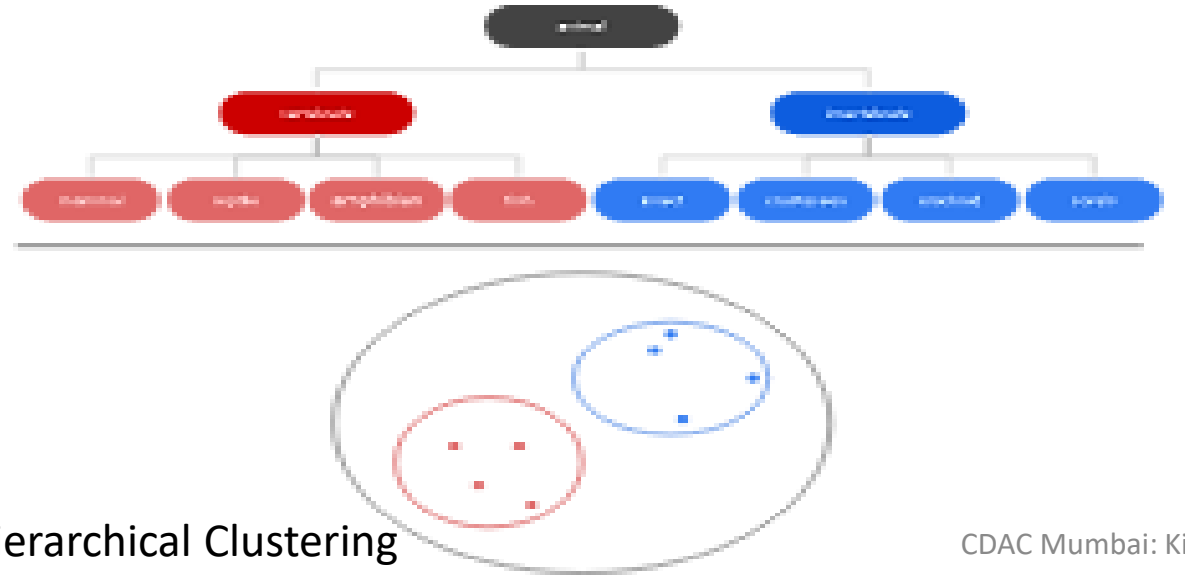
Centroid-based Clustering



Density-based Clustering



Distribution-based Clustering



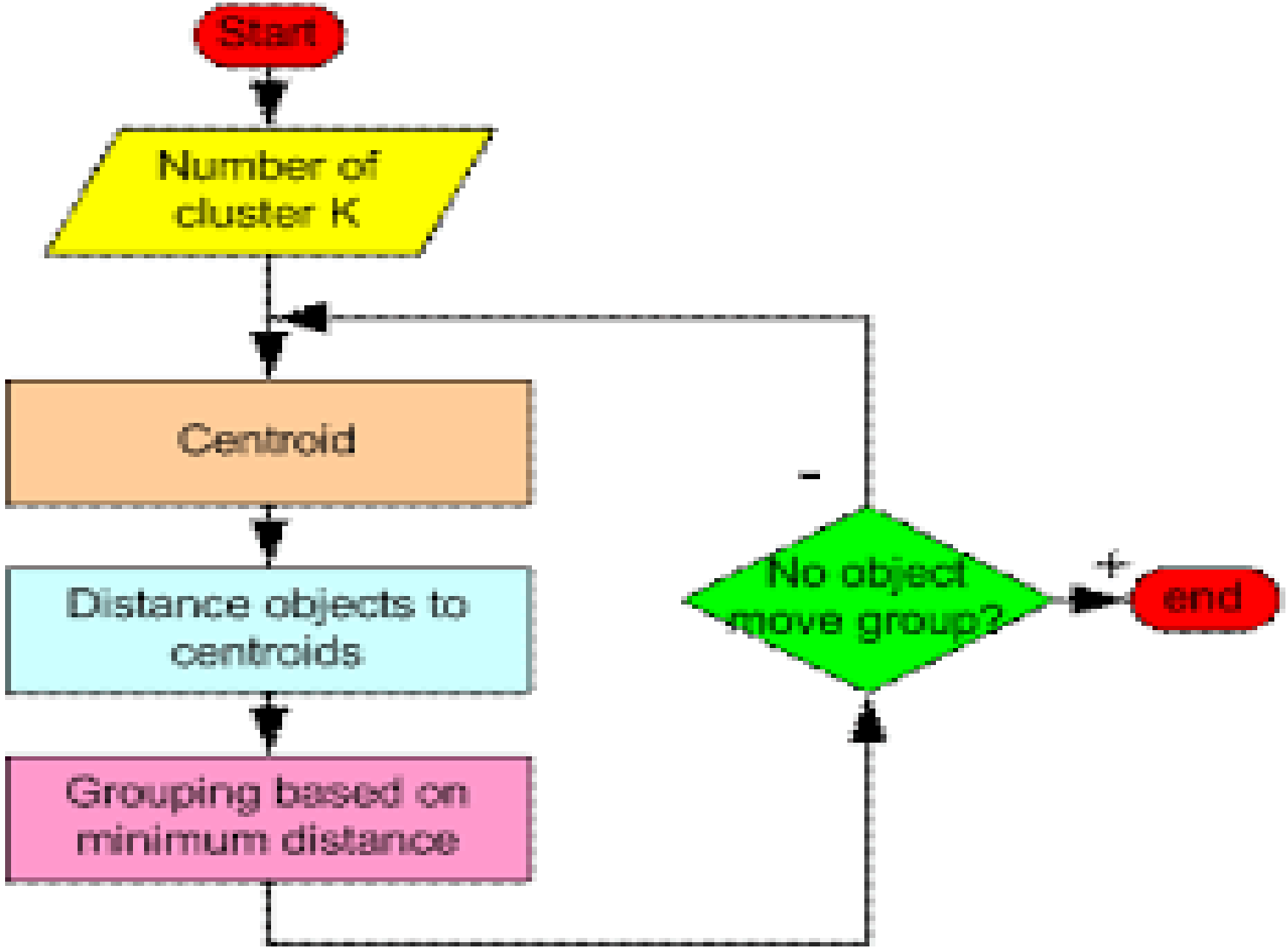
Hierarchical Clustering

Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database ***D*** of ***n*** objects into a set of ***k*** clusters, s.t., min sum of squared distance

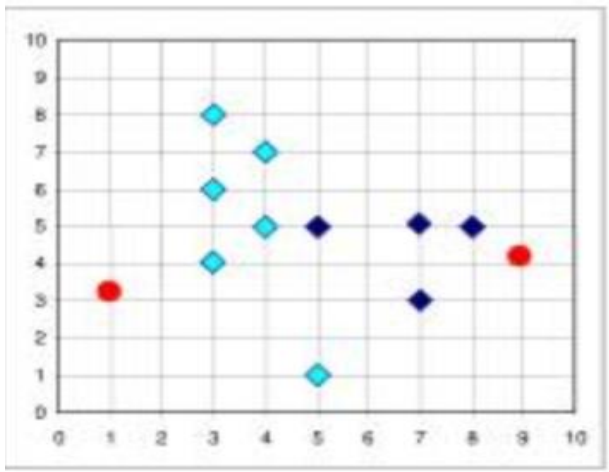
$$\sum_{m=1}^{k}\sum_{t_{mi}\in Km}(C_m - t_{mi})^2$$

- Given a *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - *k-means* (MacQueen'67): Each cluster is represented by the center of the cluster
  - *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects in the cluster


Start
Number of cluster K
Centroid
Distance objects to centroids
Grouping based on minimum distance
No object move group?
-
+
end

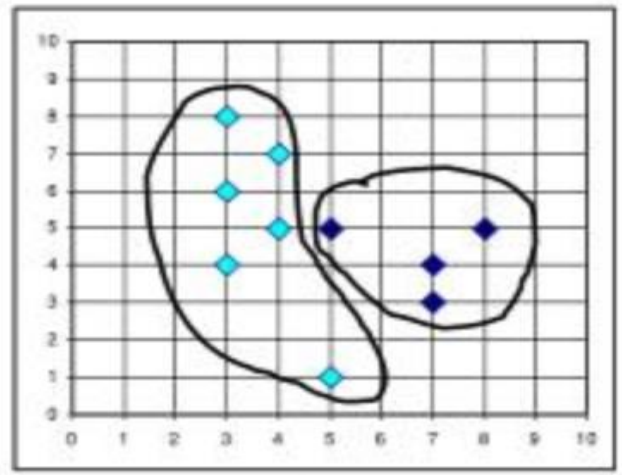
The *K-Means* Clustering Method

- Example

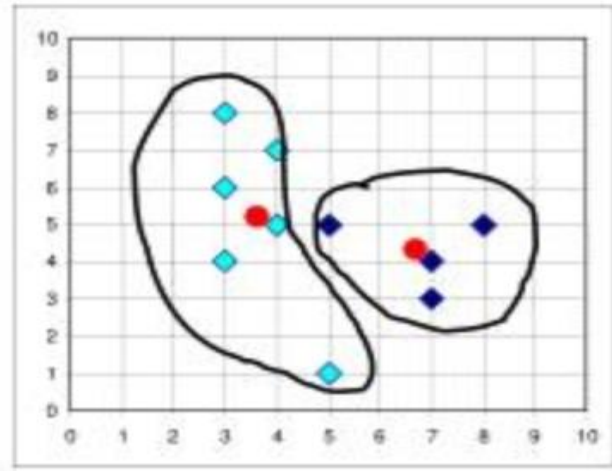


K=2

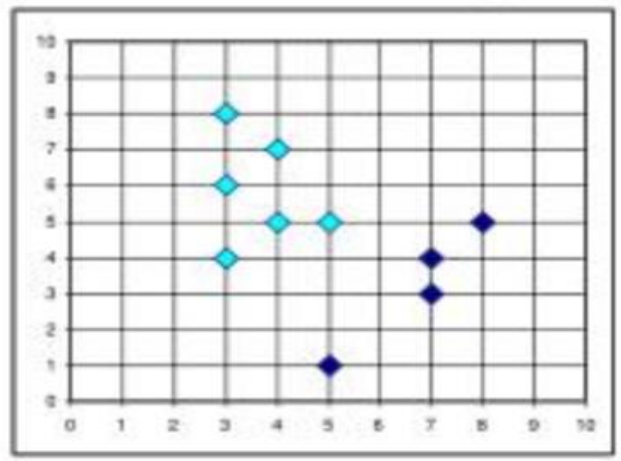
Arbitrarily choose K object as initial cluster center



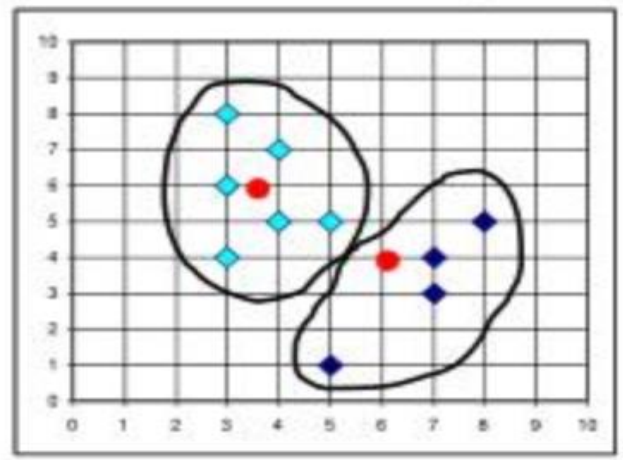
Assign each objects to most similar center



Update the cluster means



reassign

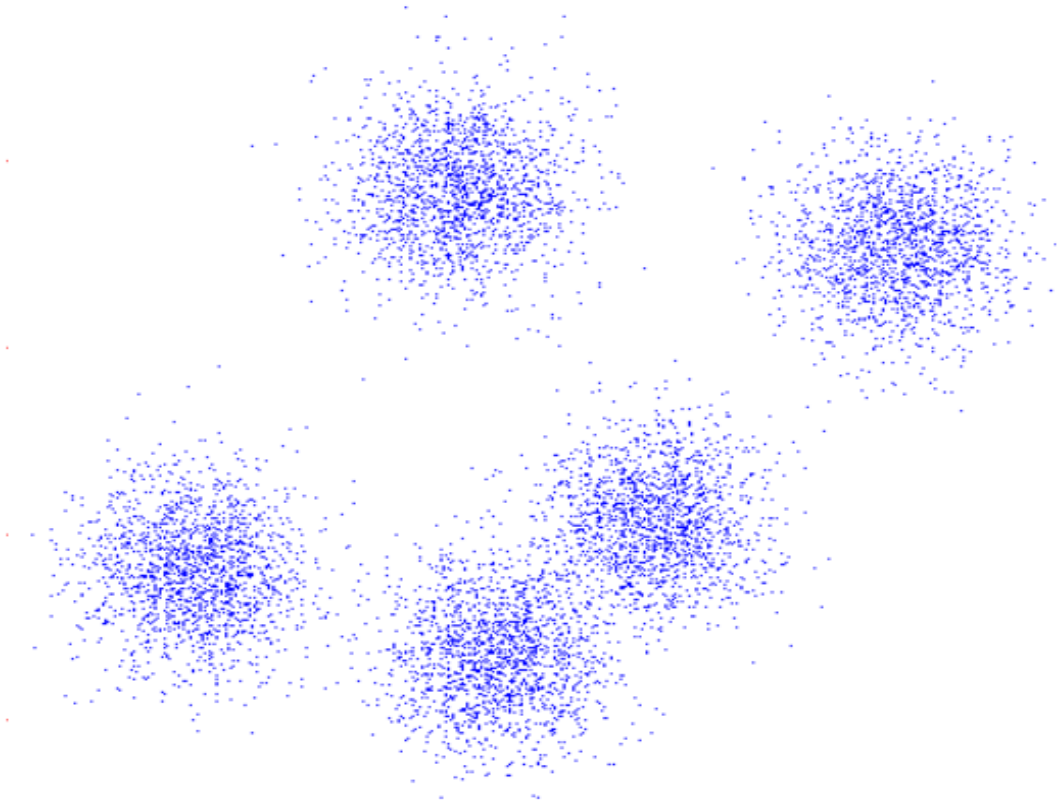


reassign

Update the cluster means

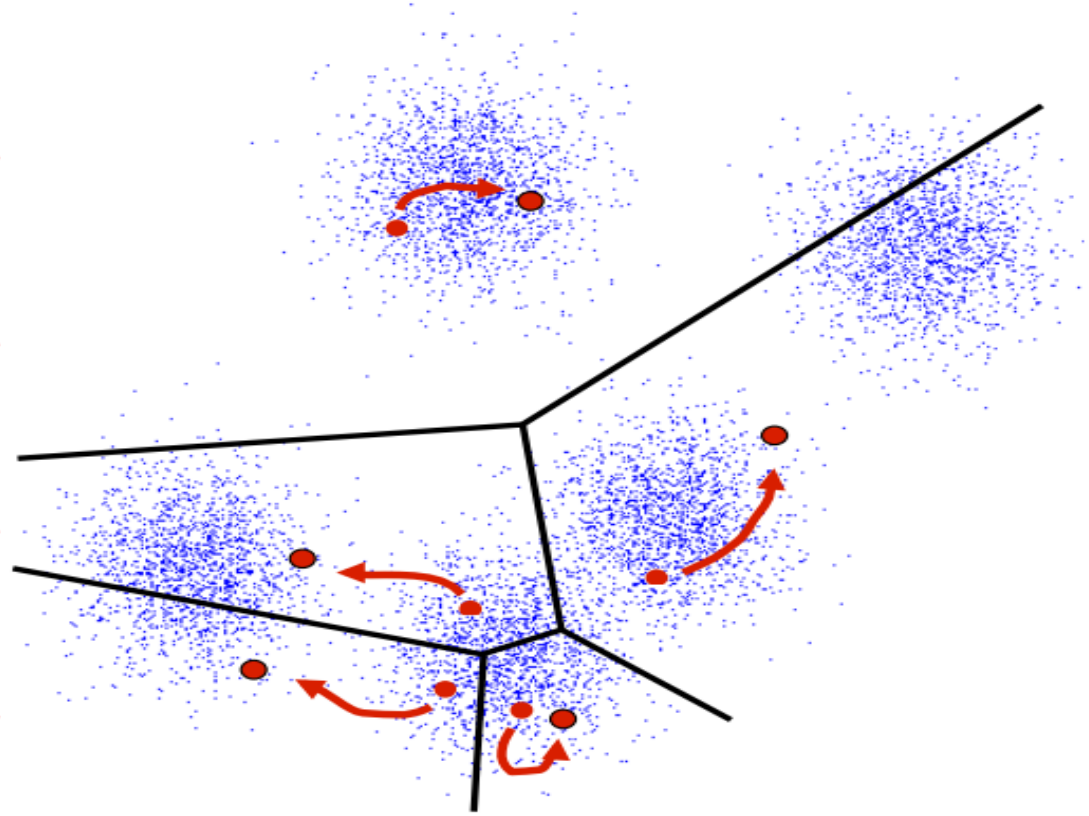
K-Means

- An iterative clustering algorithm
  - Initialize: Pick *K* random points as cluster centers
  - Alternate:
    1. Assign data points to closest cluster center
    2. Change the cluster center to the average of its assigned points
  - Stop when no points' assignments change

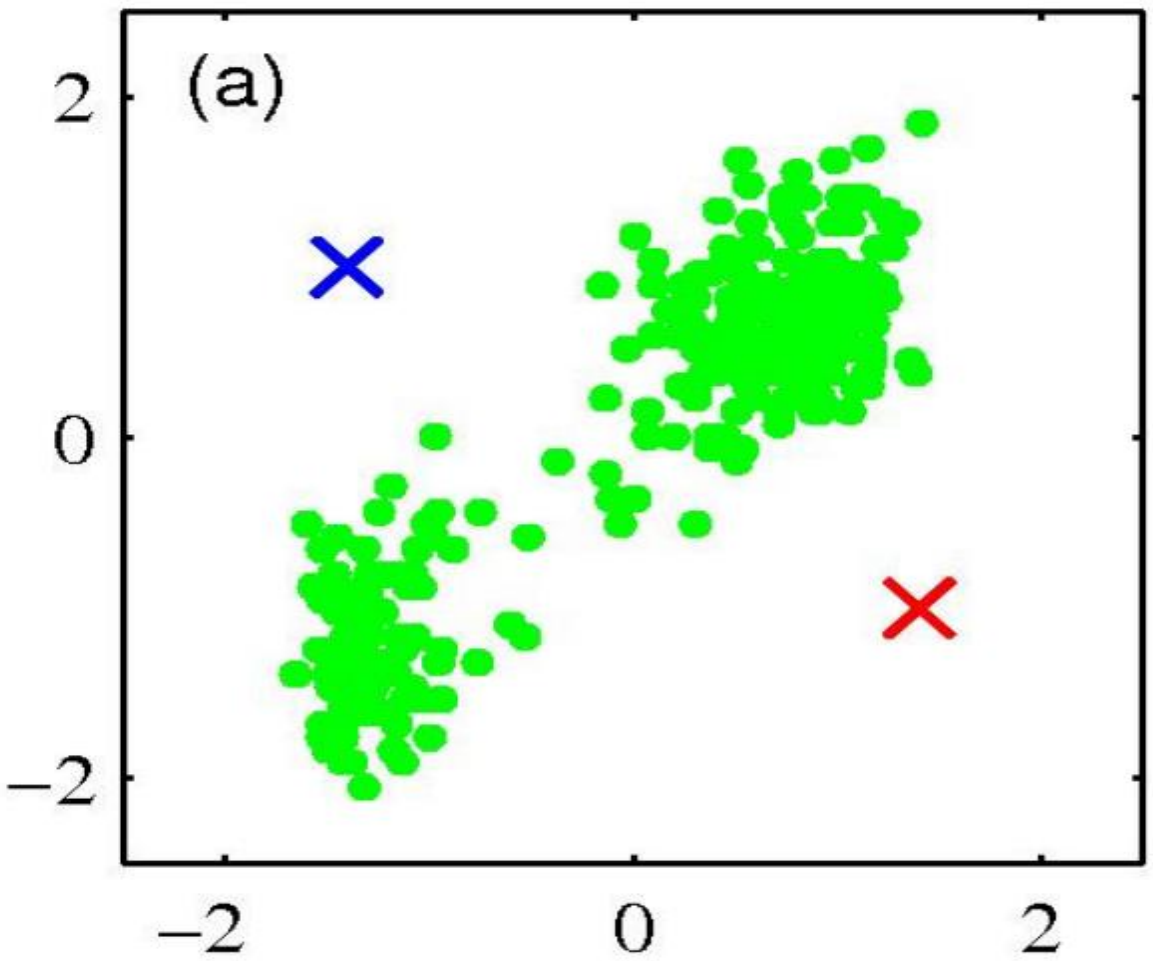


K-Means

- An iterative clustering algorithm
  - Initialize: Pick *K* random points as cluster centers
  - Alternate:
    1. Assign data points to closest cluster center
    2. Change the cluster center to the average of its assigned points
  - Stop when no points' assignments change



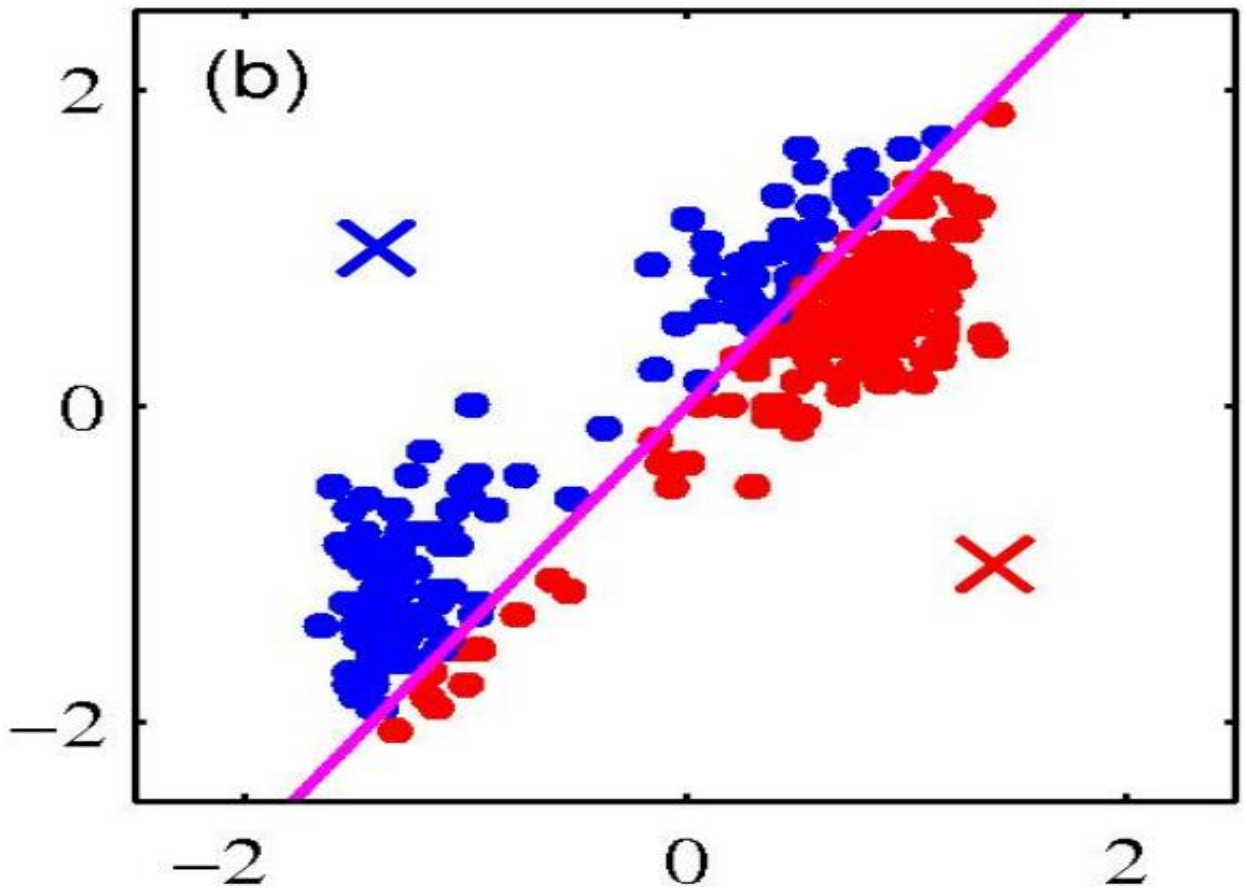
K-means clustering: Example



- Pick *K* random points as cluster centers (means)

Shown here for *K*=2

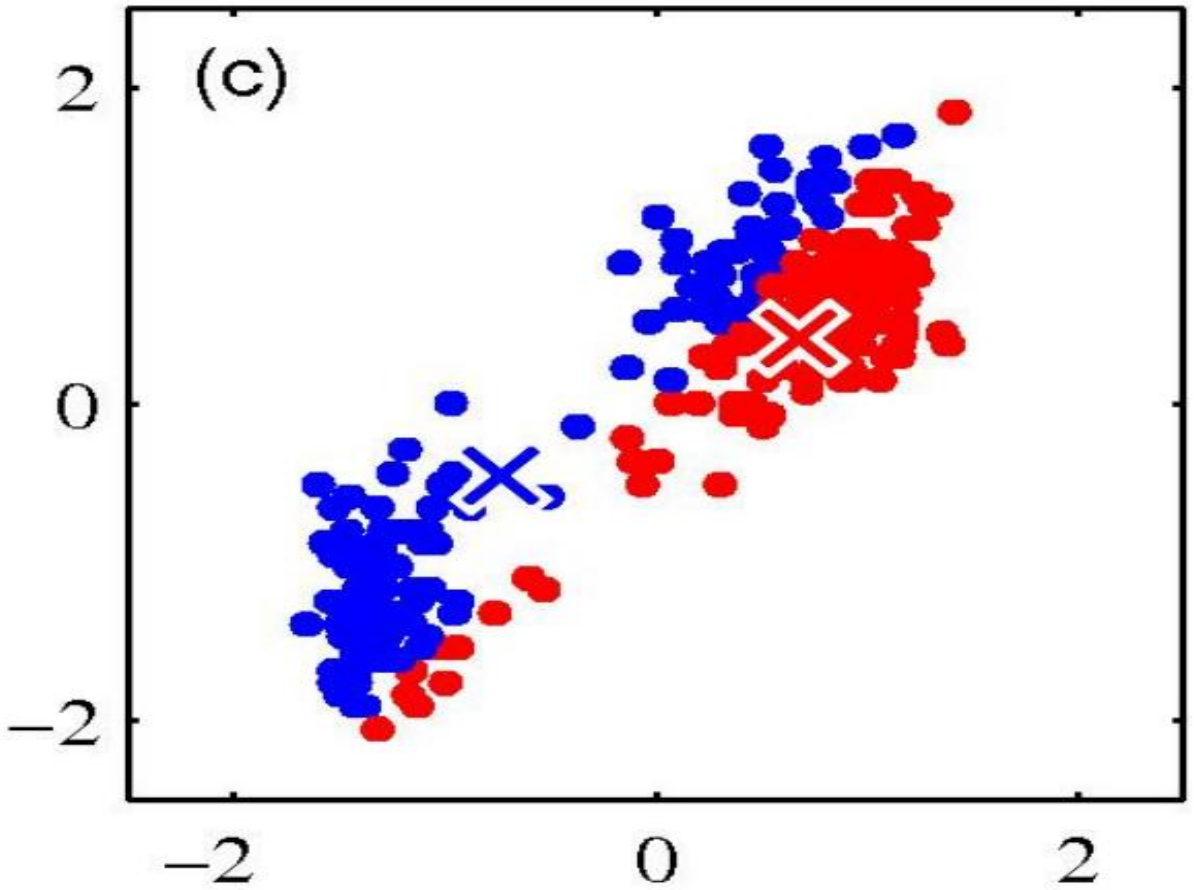
K-means clustering: Example



Iterative Step 1

- Assign data points to closest cluster center

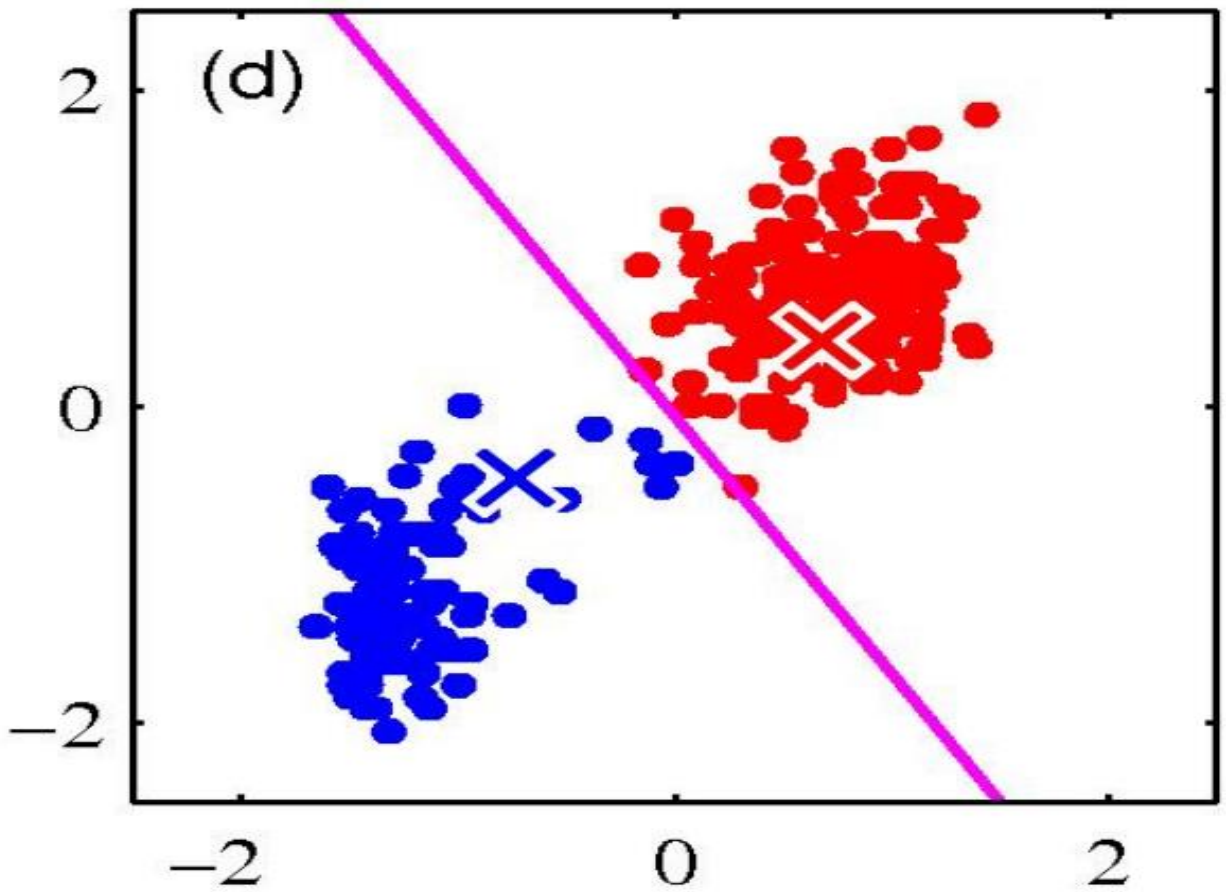
K-means clustering: Example



Iterative Step 2

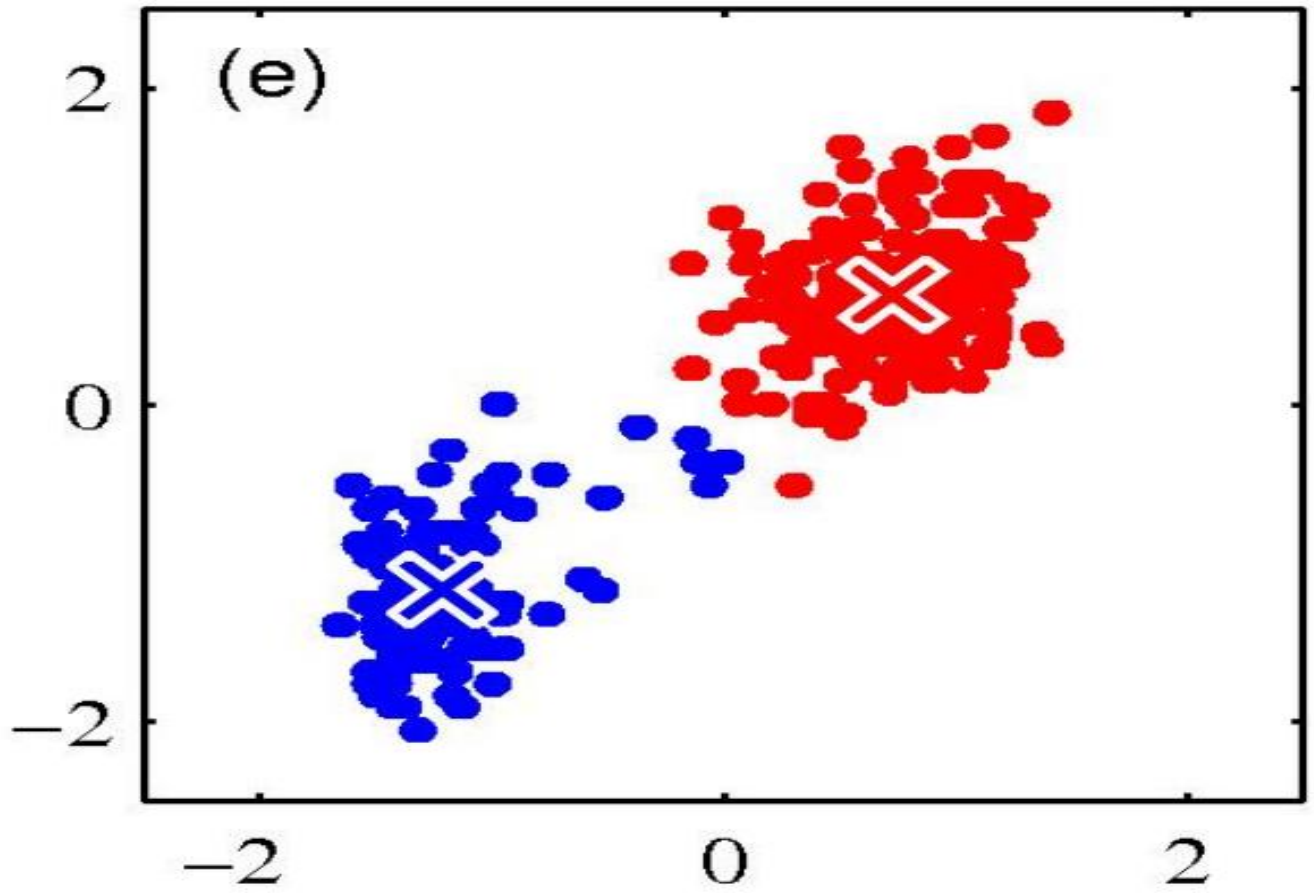
- Change the cluster center to the average of the assigned points

K-means clustering: Example

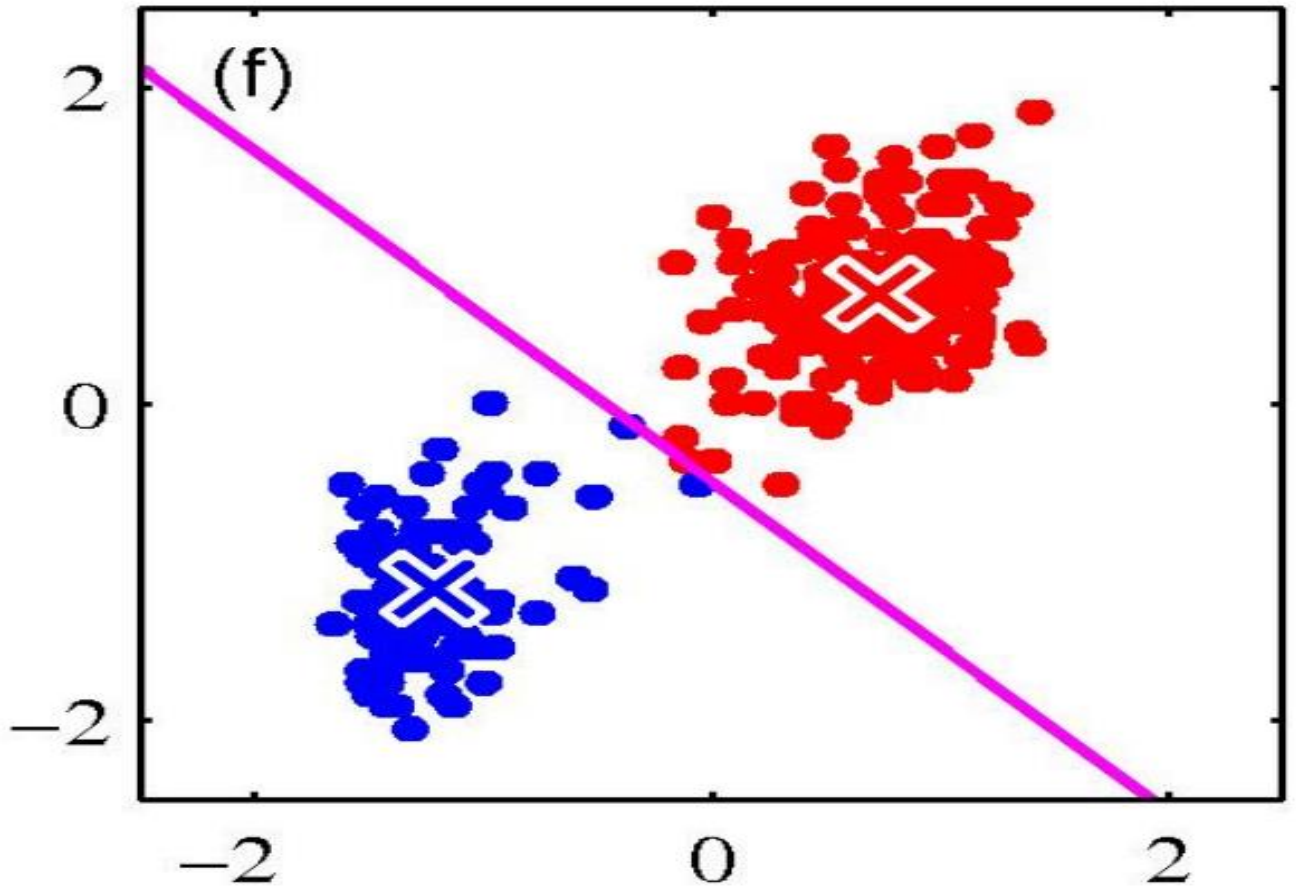


- Repeat until convergence

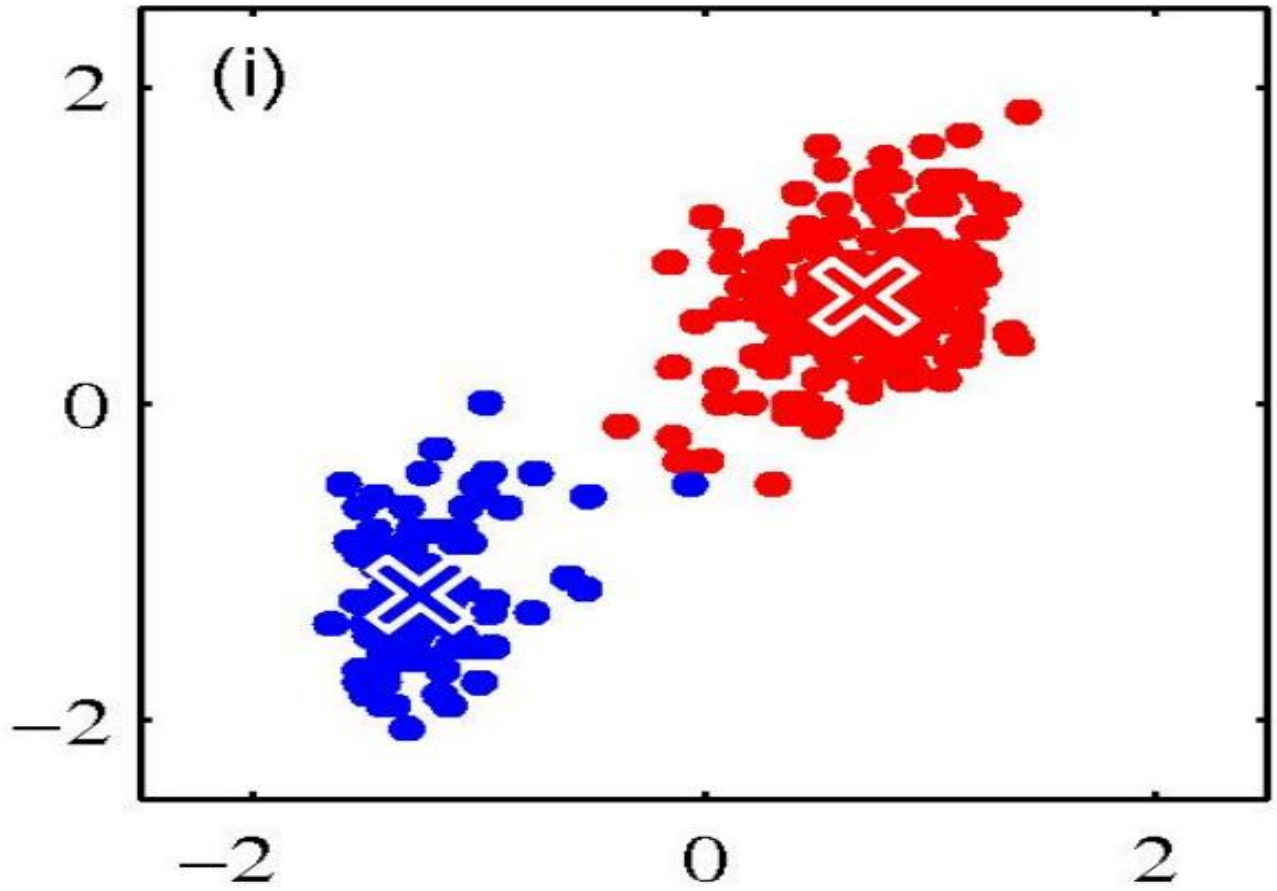
K-means clustering: Example



K-means clustering: Example

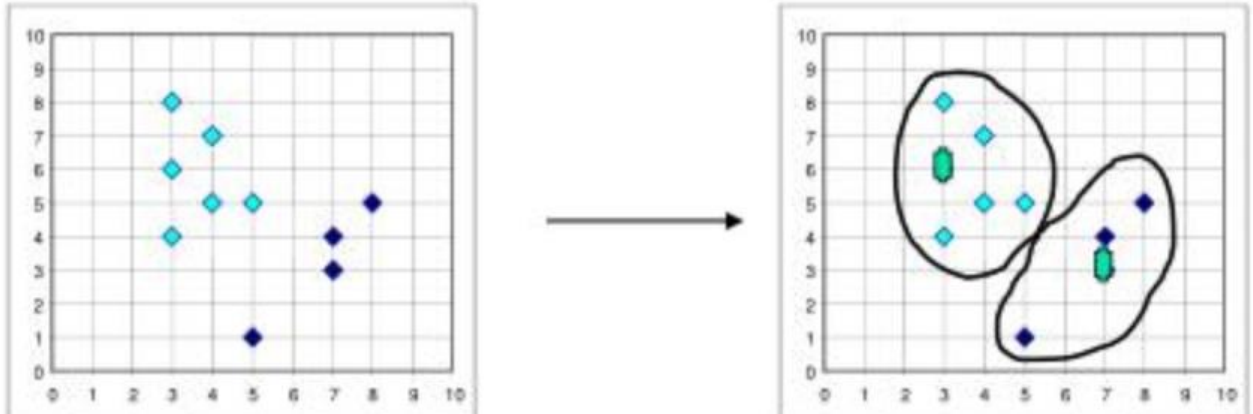


K-means clustering: Example

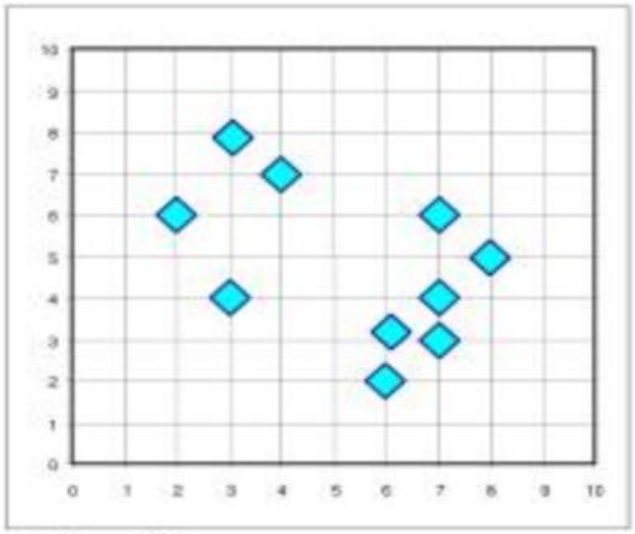


What Is the Problem of the K-Means Method?

- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data.
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.

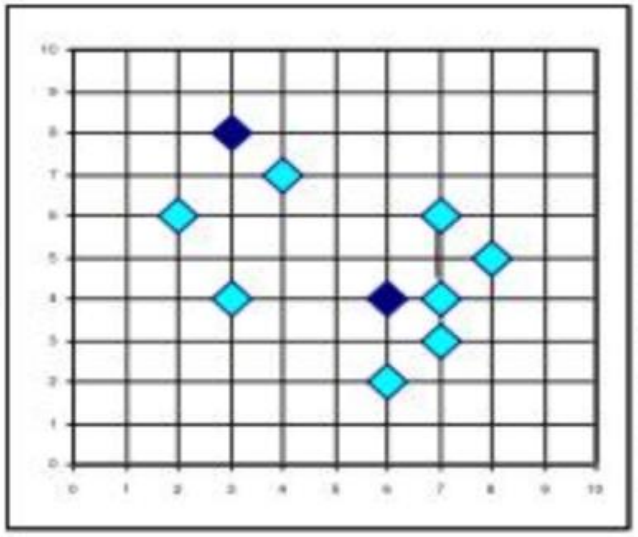


A Typical K-Medoids Algorithm (PAM)



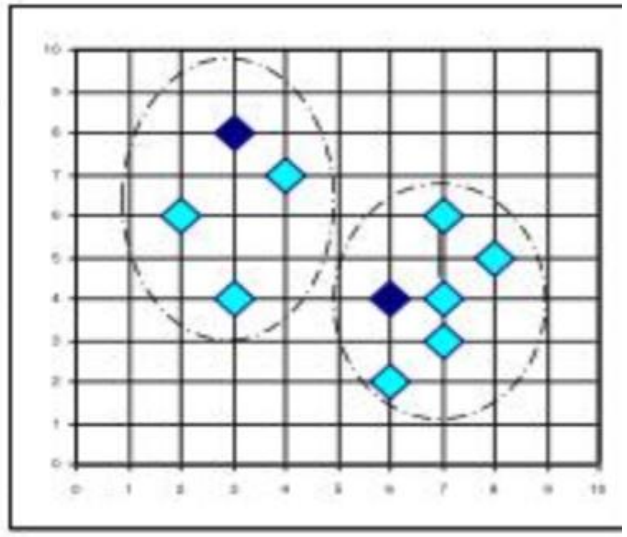
K=2

Arbitrary choose k object as initial medoids

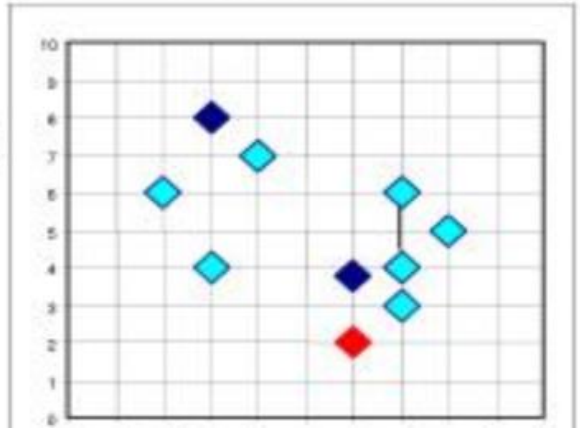


Total Cost = 20

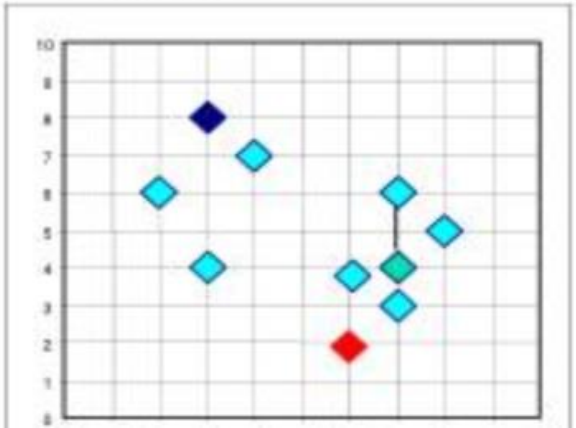
Assign each remaining object to nearest medoids



Randomly select a nonmedoid object, O_{ramdom}



Compute total cost of swapping



Total Cost = 26

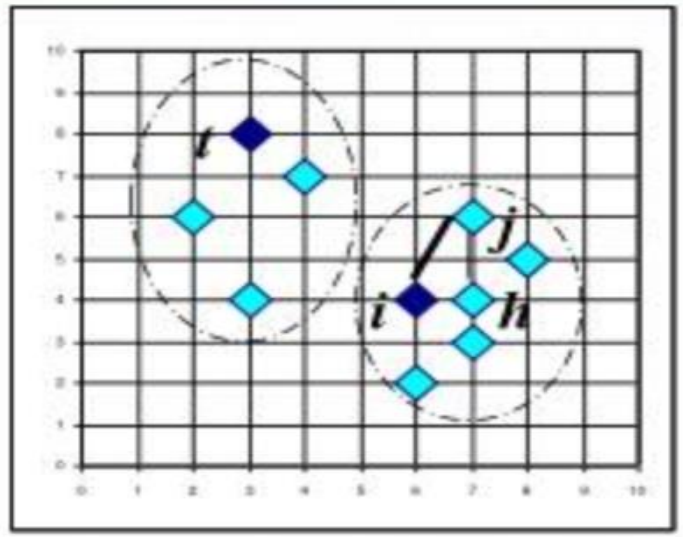
Swapping O and O_{ramdom}

If quality is improved.

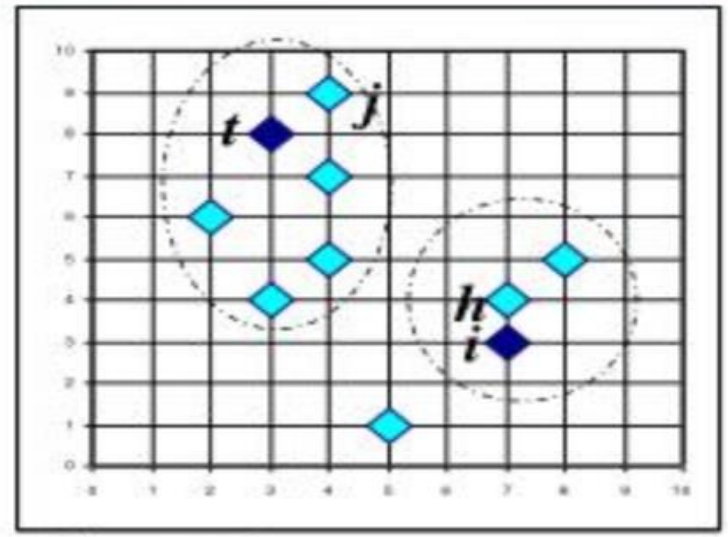
Do loop

Until no change

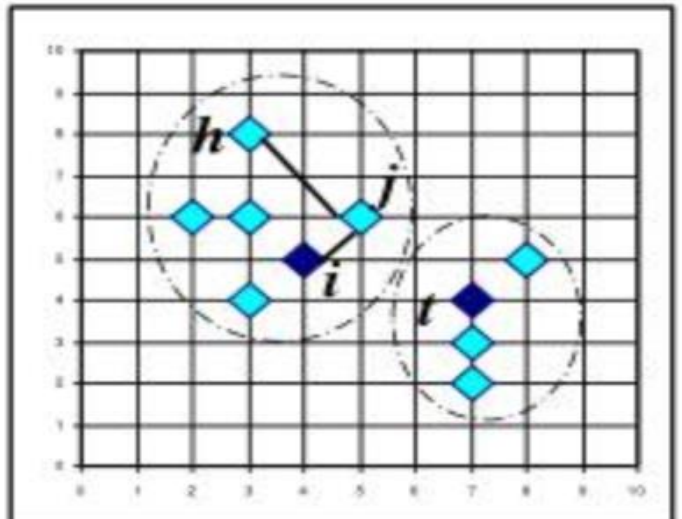
PAM Clustering: Total swapping cost $TC_{ih}=\sum_j C_{jih}$

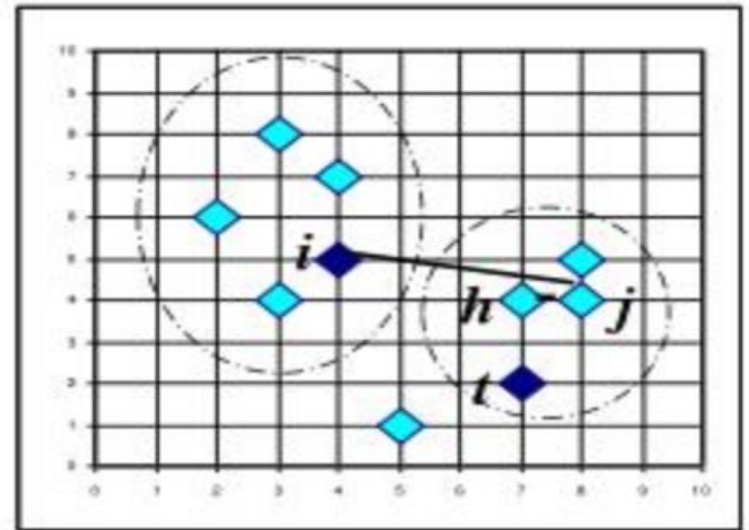


$C_{jih} = d(j, h) - d(j, i)$



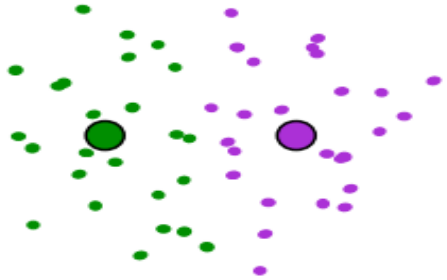
$C_{jih} = 0$



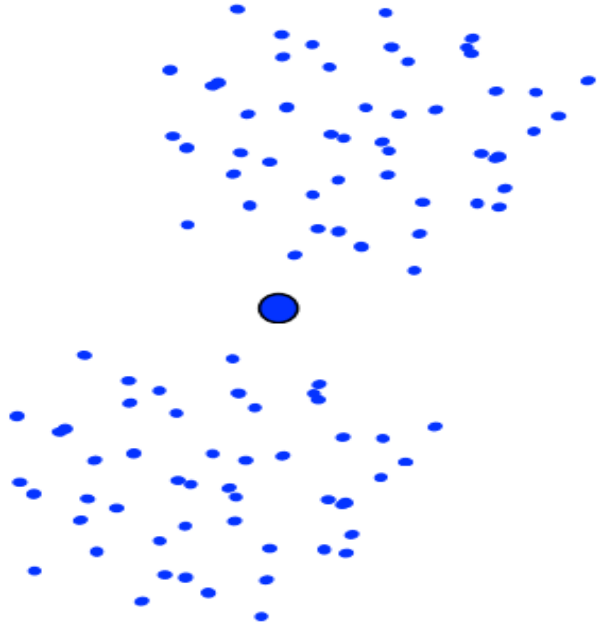


K-Means Getting Stuck

A local optimum:

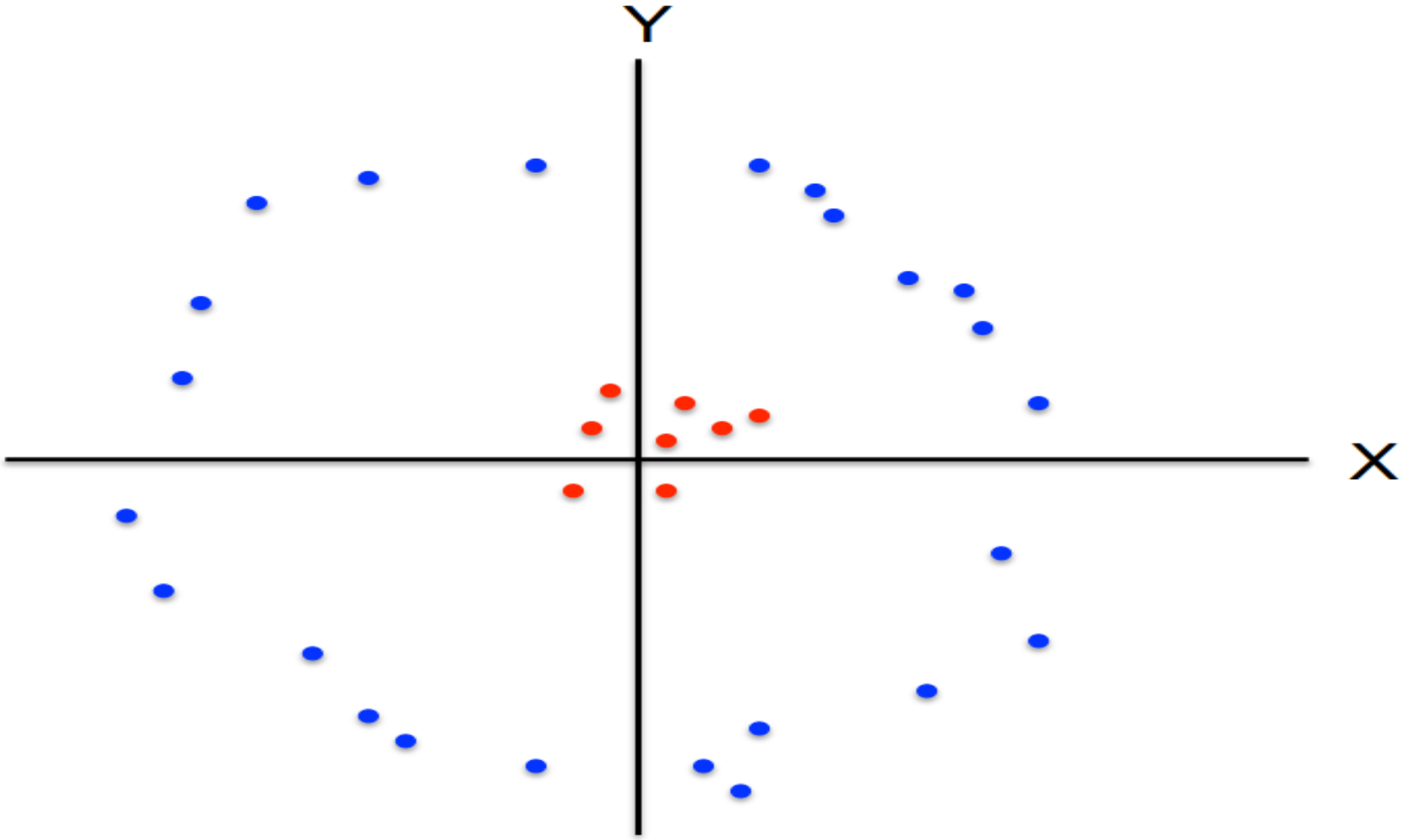


Would be better to have
one cluster here

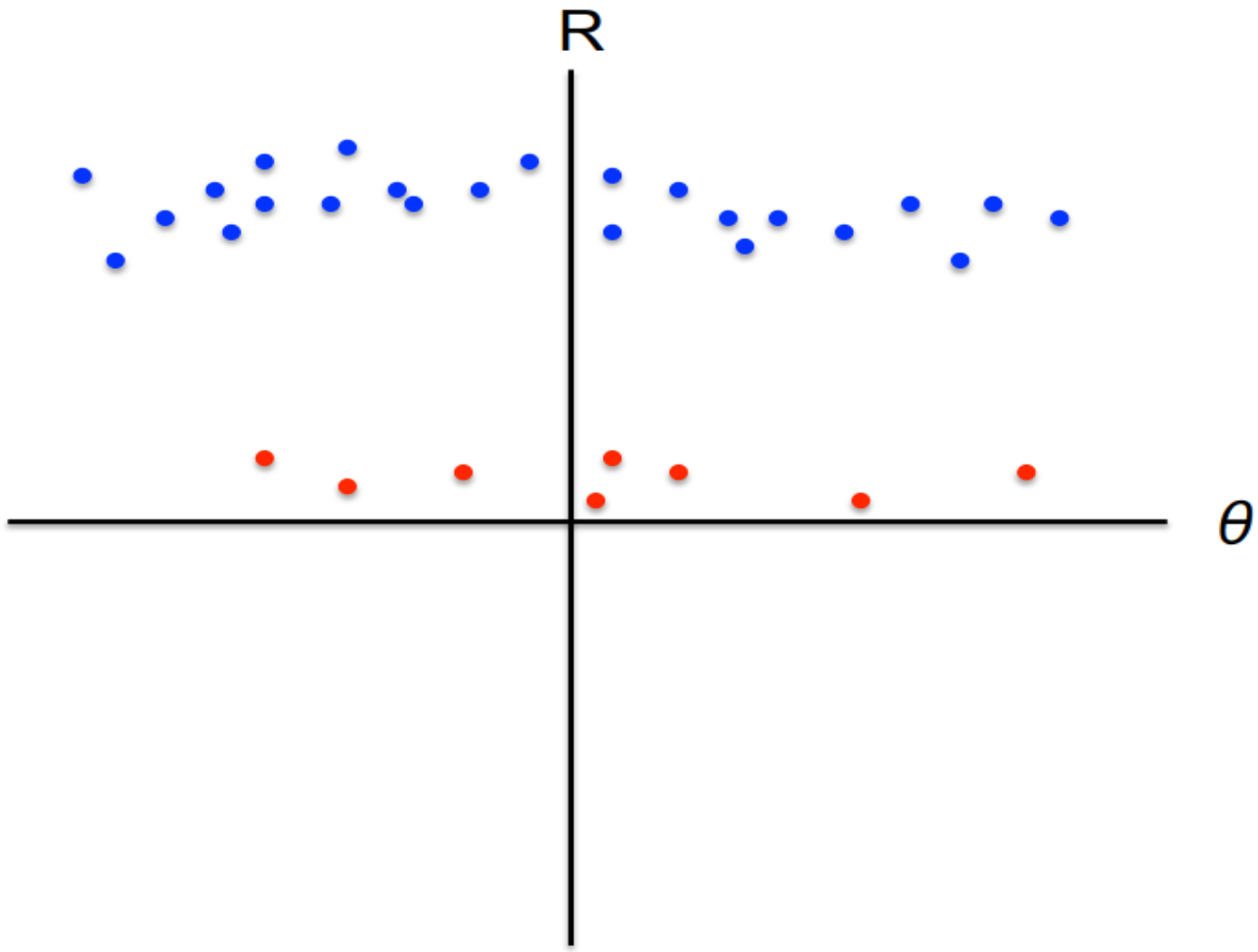


… and two clusters here

K-means not able to properly cluster



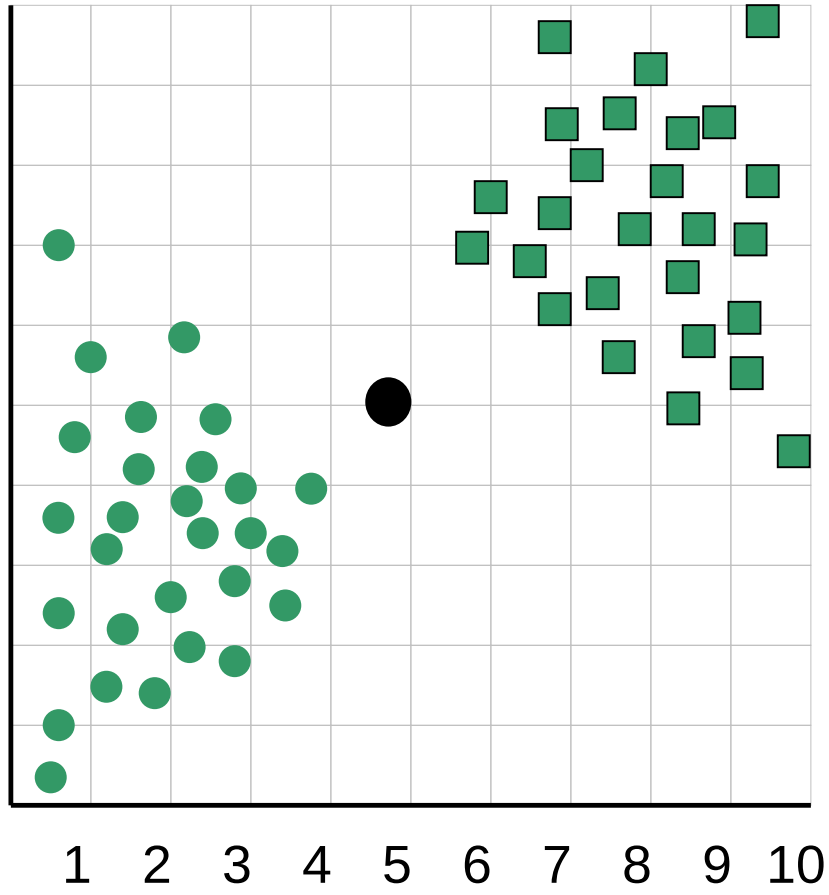
Changing the features (distance function) can help



Um, what about k?

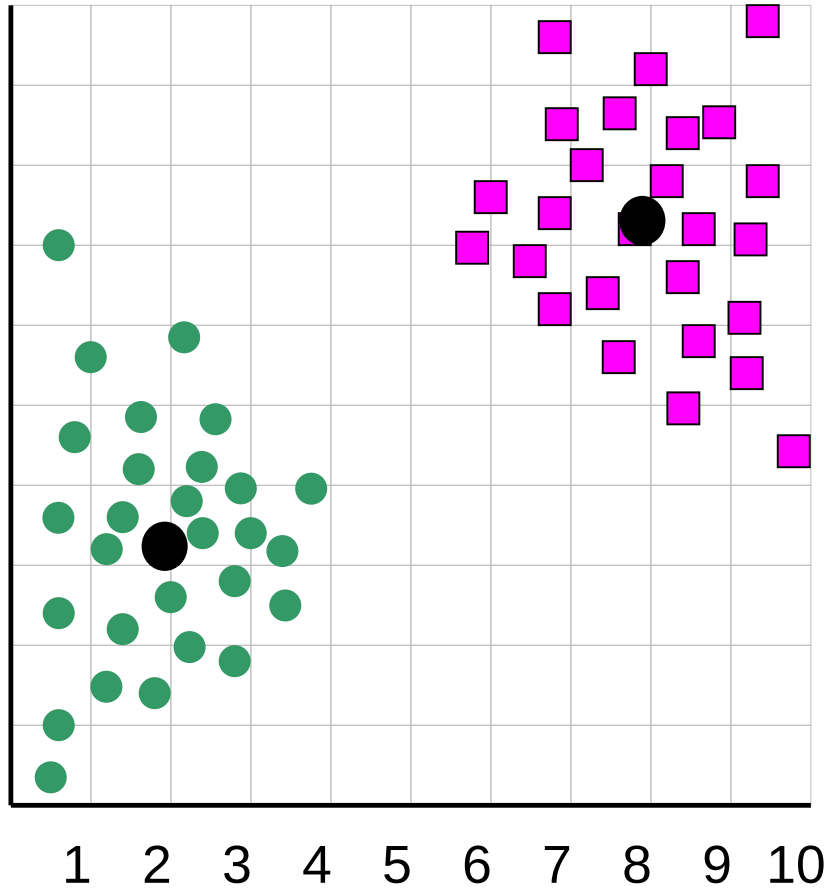
- Idea 1: Use our new trick of cross validation to select k
  - What should we optimize? SSE? Trace?
  - Problem?
- Idea 2: Let our domain expert look at the clustering and decide if they like it
  - How should we show this to them?
  - Problem?
- Idea 3: The “knee” solution

When k = 1, the objective function is 873.0



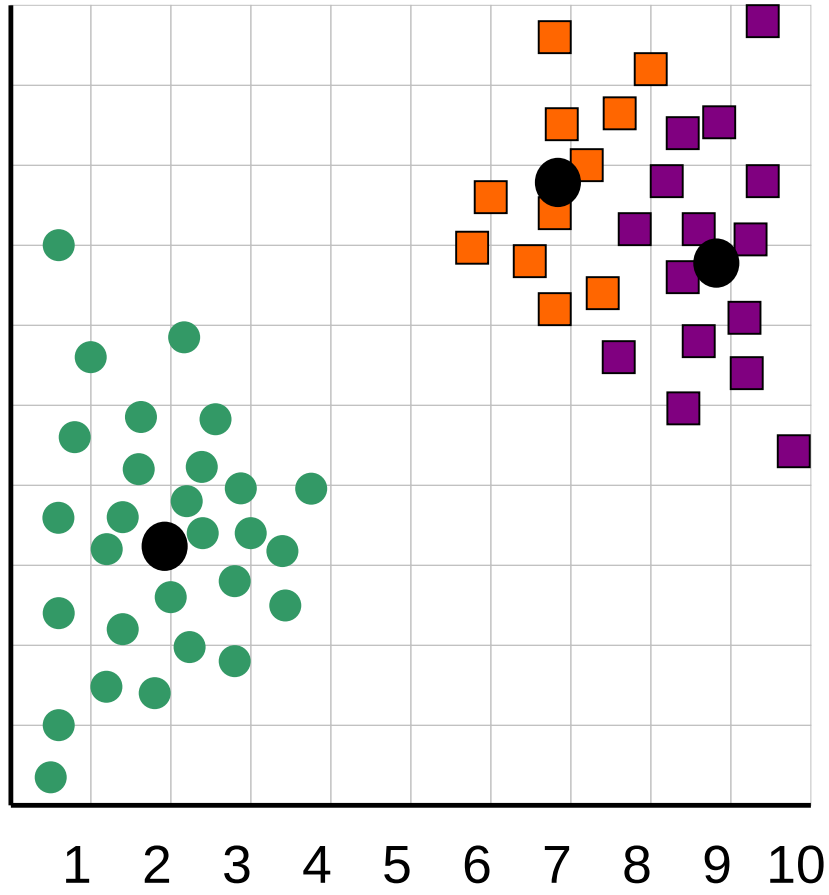
Slide based on one by Eamonn Keogh

When k = 2, the objective function is 173.1



Slide based on one by Eamonn Keogh

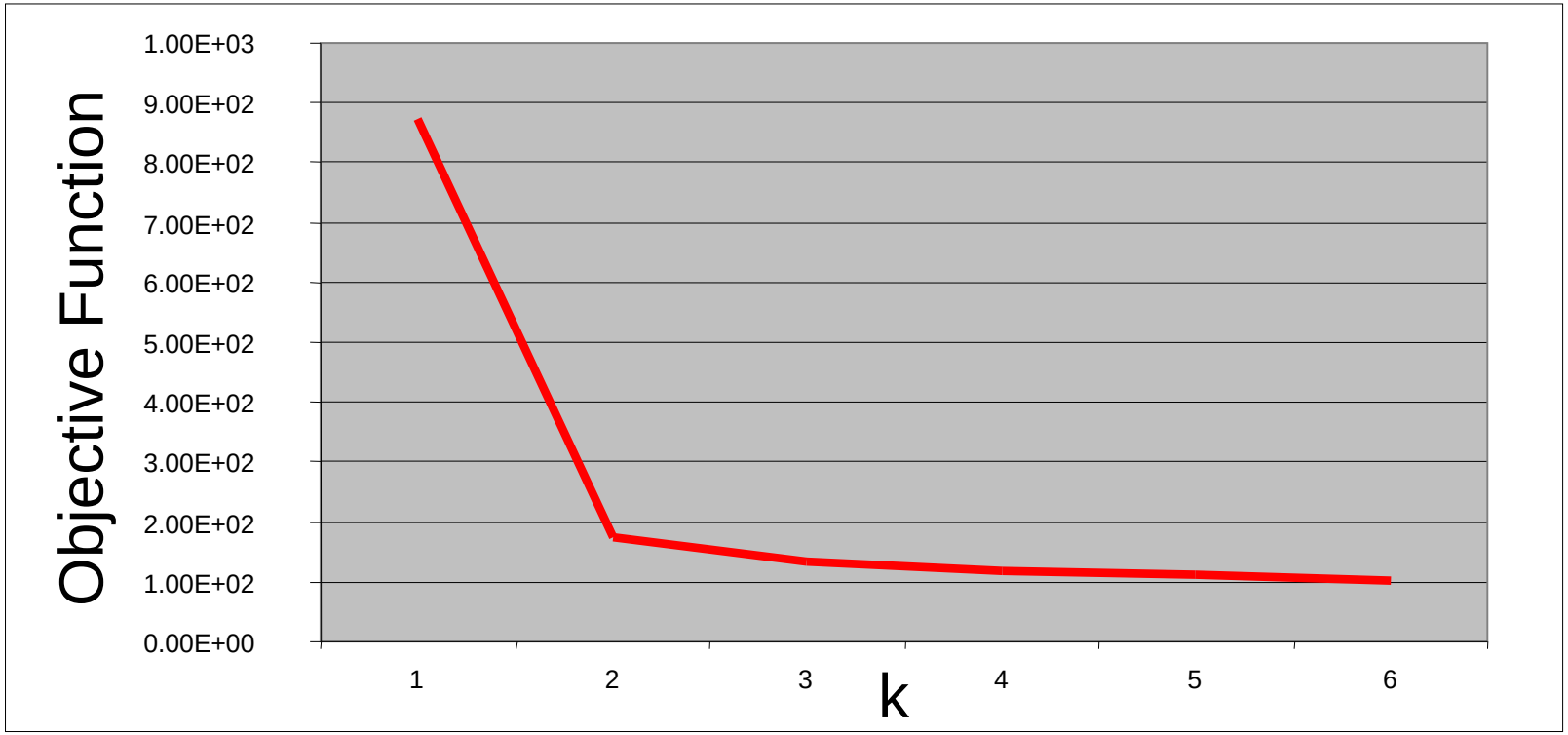
When k = 3, the objective function is 133.6



Slide based on one by Eamonn Keogh

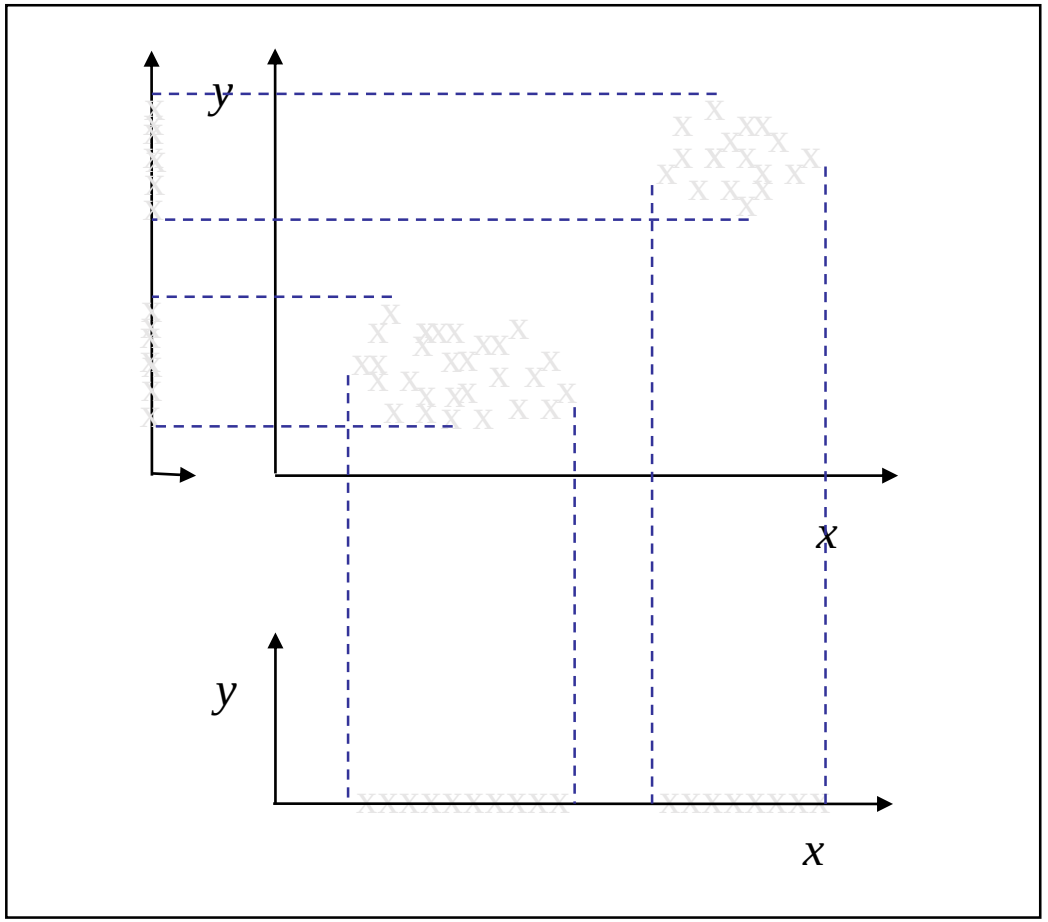
We can plot the objective function values for k equals 1 to 6…

The abrupt change at k = 2, is highly suggestive of two clusters in the data. This technique for determining the number of clusters is known as “knee finding” or “elbow finding”.

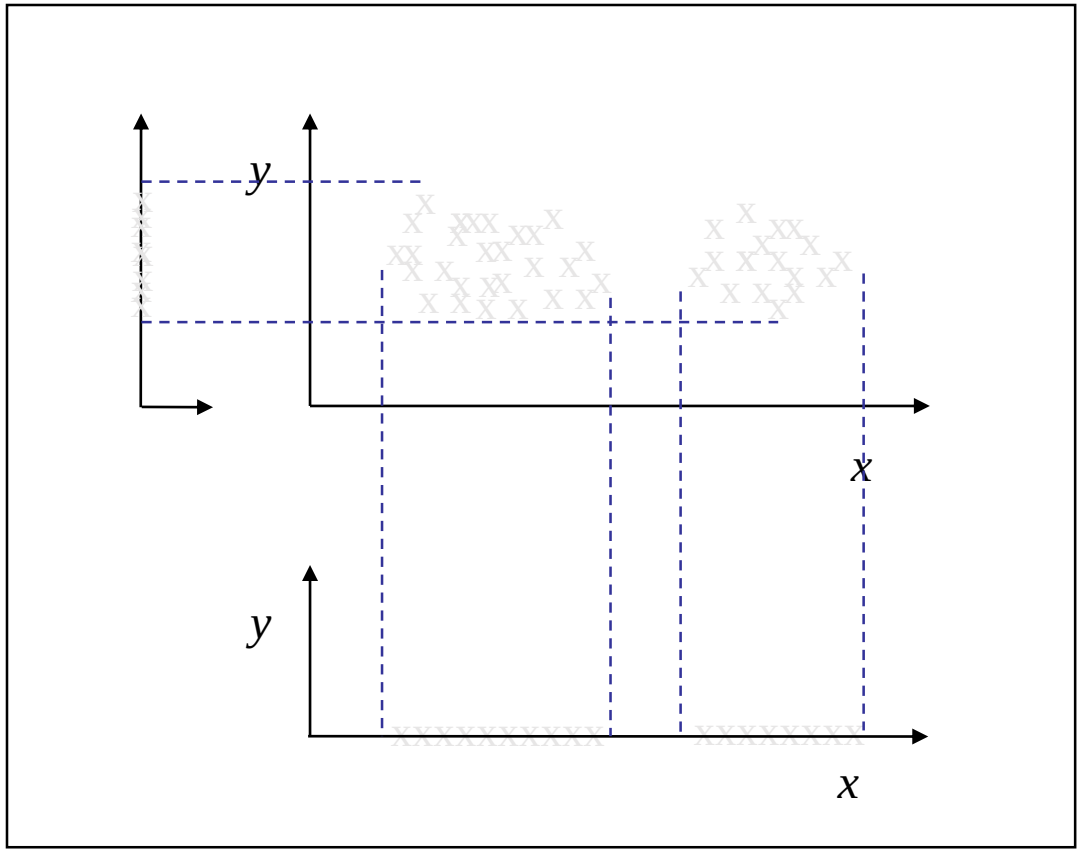


Slide based on one by Eamonn Keogh

Redundant



Irrelevant



Curse of Dimensionality

100 observations cover the 1-D unit interval [0,1] well

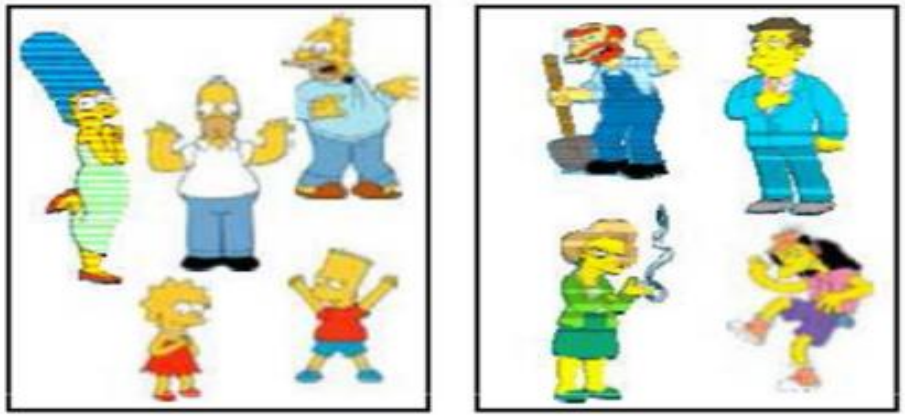
Consider the 10-D unit hypersquare, 100 observations are now isolated points in a vast empty space.

Consequence of the Curse

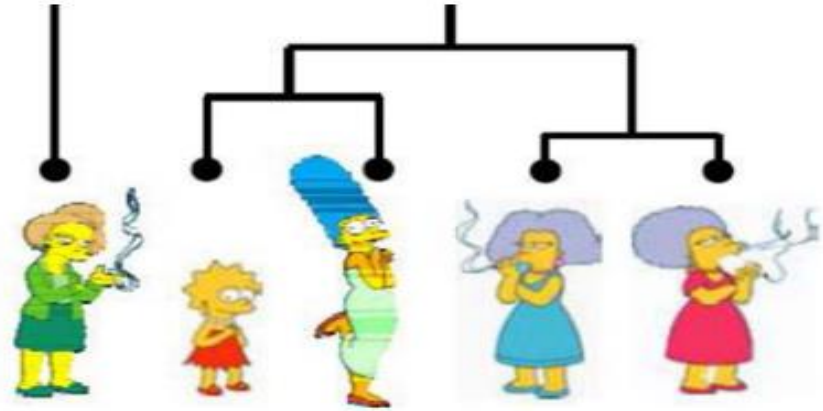
- Suppose the number of samples given to us in the total sample space is fixed
- Let the dimension increase
- Then the distance of the k nearest neighbors of any point increases

Clustering algorithms

- Partition algorithms (Flat)
  - K-means
  - Mixture of Gaussian
  - Spectral Clustering



- Hierarchical algorithms
  - Bottom up – agglomerative
  - Top down – divisive

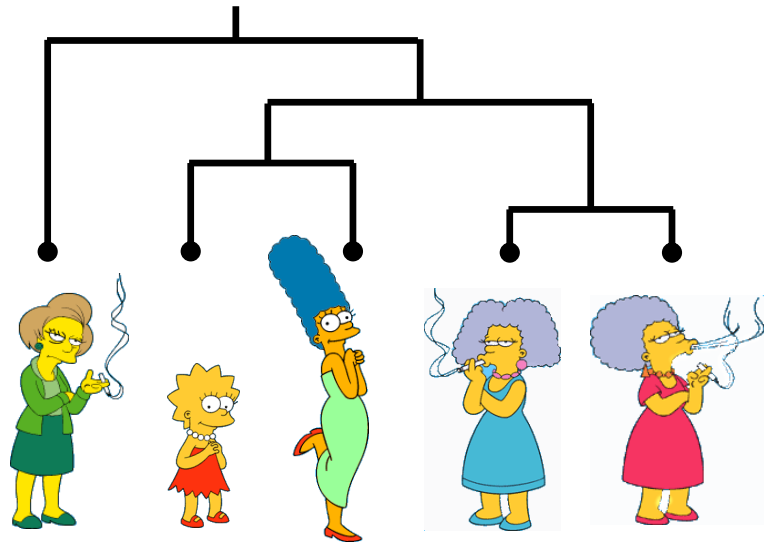


Slide based on one by Eamonn Keogh

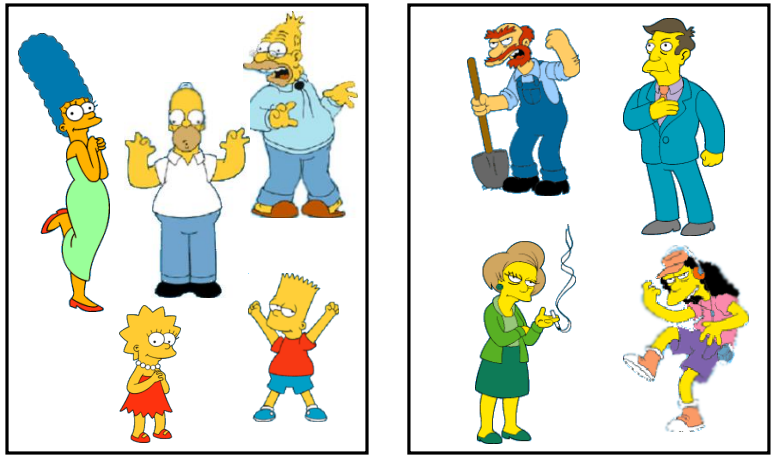
Two Types of Clustering

- **Partitional algorithms:** Construct various partitions and then evaluate them by some criterion
- **Hierarchical algorithms:** Create a hierarchical decomposition of the set of objects using some criterion

Hierarchical

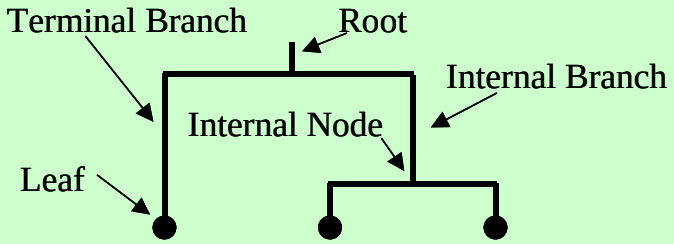


Partitional

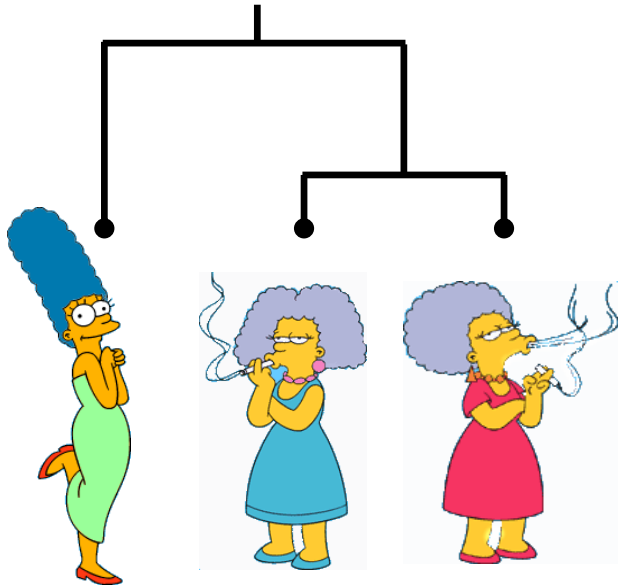


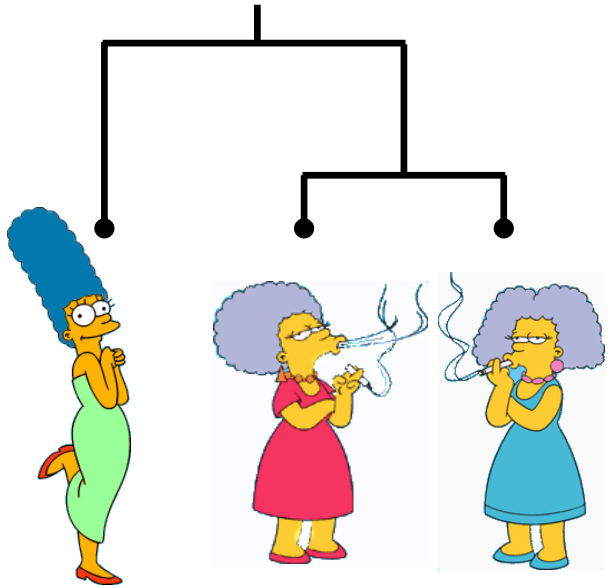
Slide based on one by Eamonn Keogh

Dendogram: A Useful Tool for Summarizing Similarity Measurements



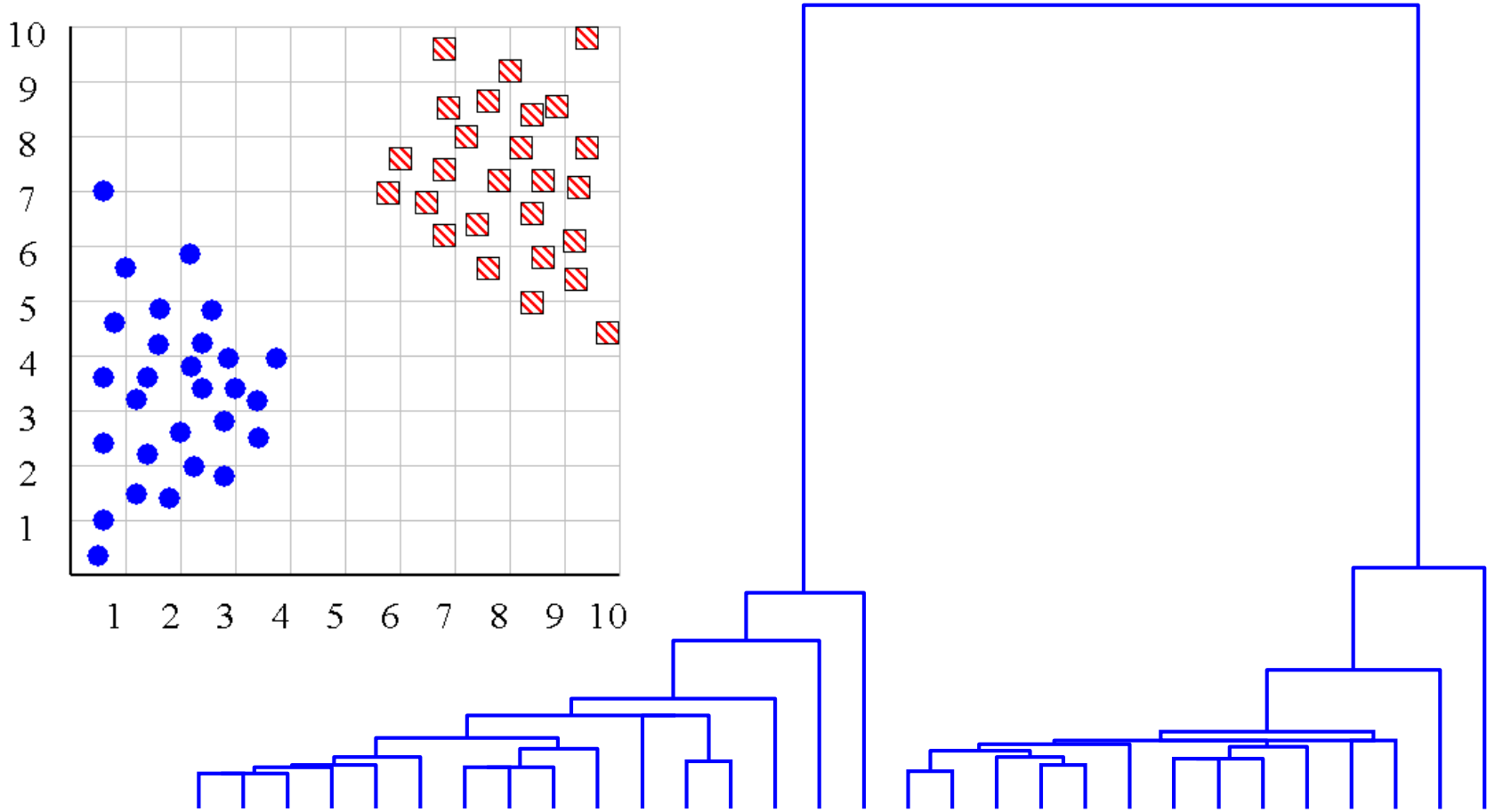
The similarity between two objects in a dendrogram is represented as the height of the lowest internal node they share.





Slide based on one by Eamonn Keogh

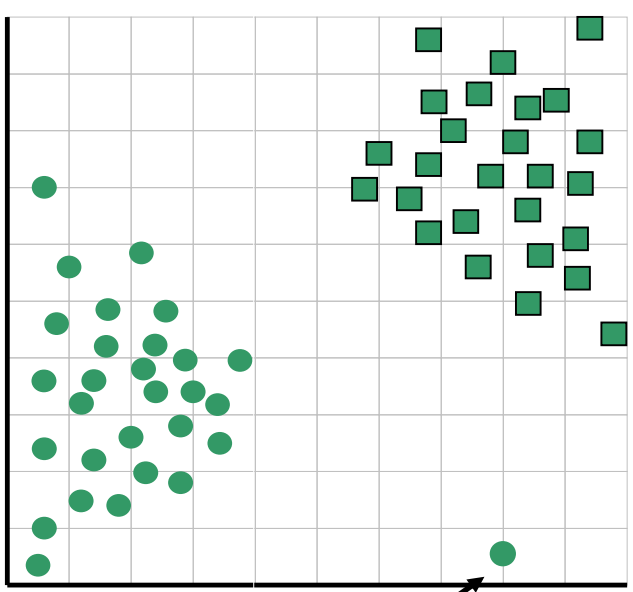
We can look at the dendrogram to determine the “correct” number of clusters.



Slide based on one by Eamonn Keogh

One potential use of a dendrogram: detecting outliers

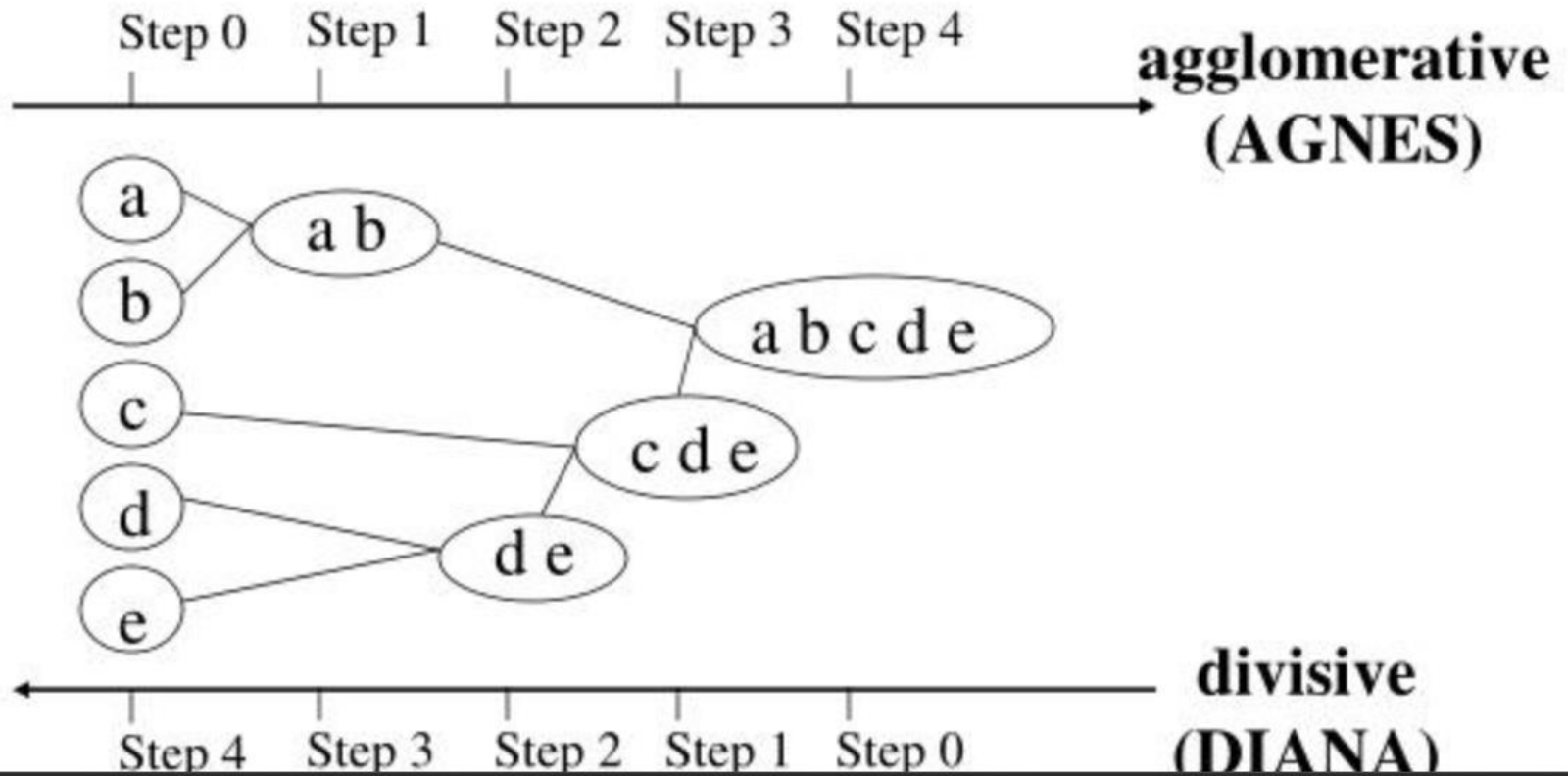
The single isolated branch is suggestive of a data point that is very different to all others



Outlier

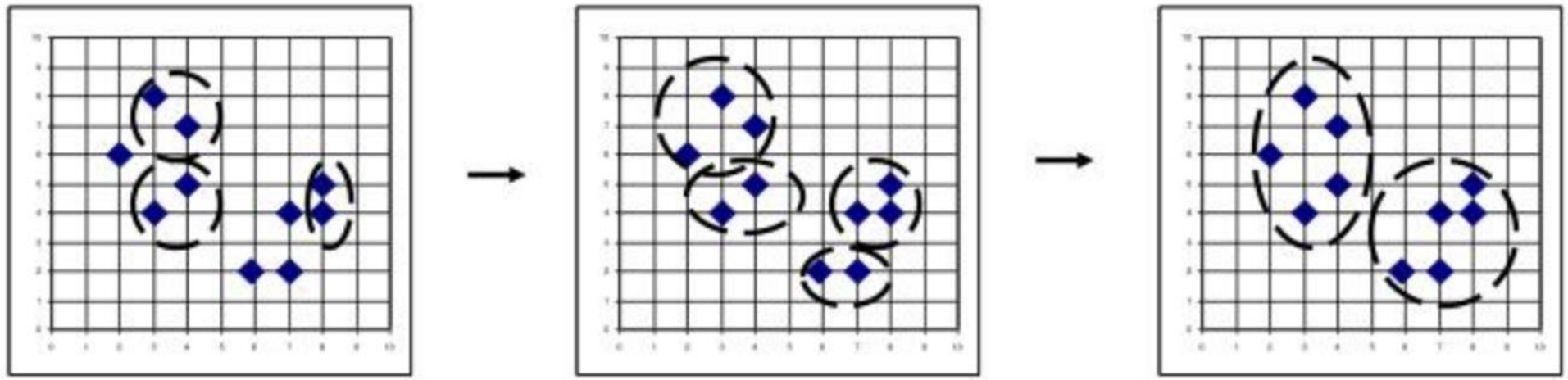
Hierarchical Clustering

- Use distance matrix as clustering criteria. This method does not require the number of clusters ***k*** as an input, but needs a termination condition



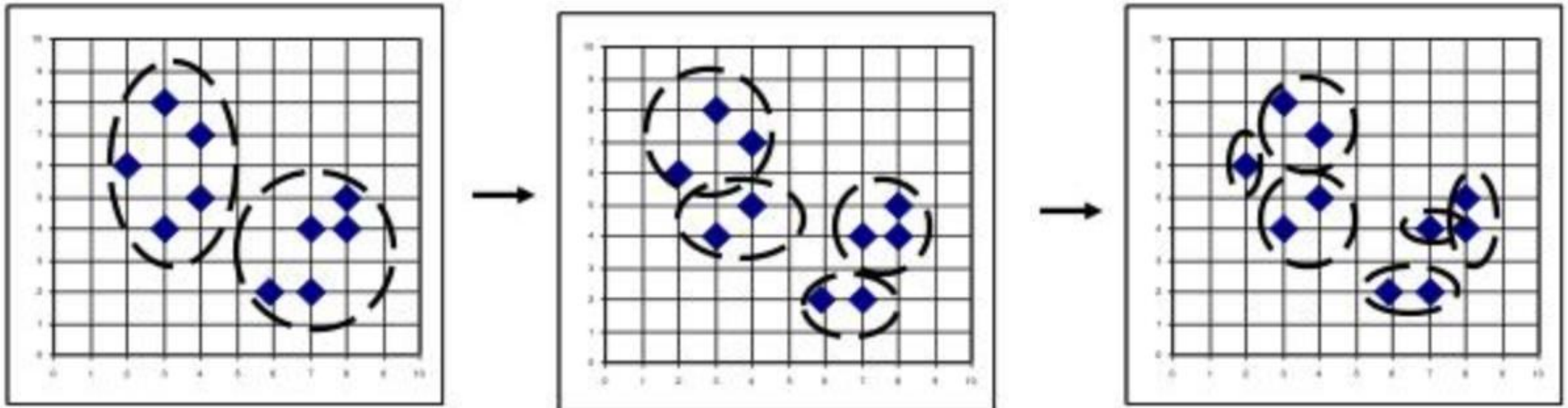
AGNES (Agglomerative Nesting)

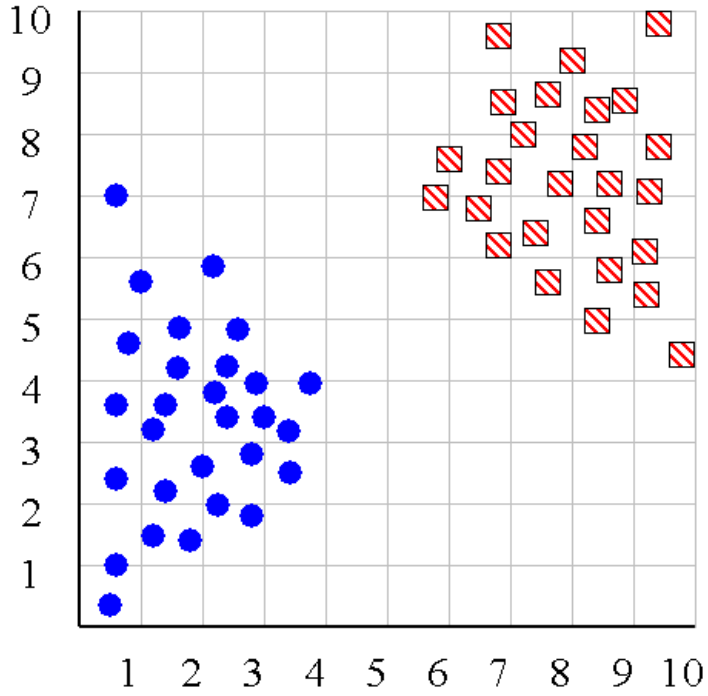
- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Use the Single-Link method and the dissimilarity matrix.
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster



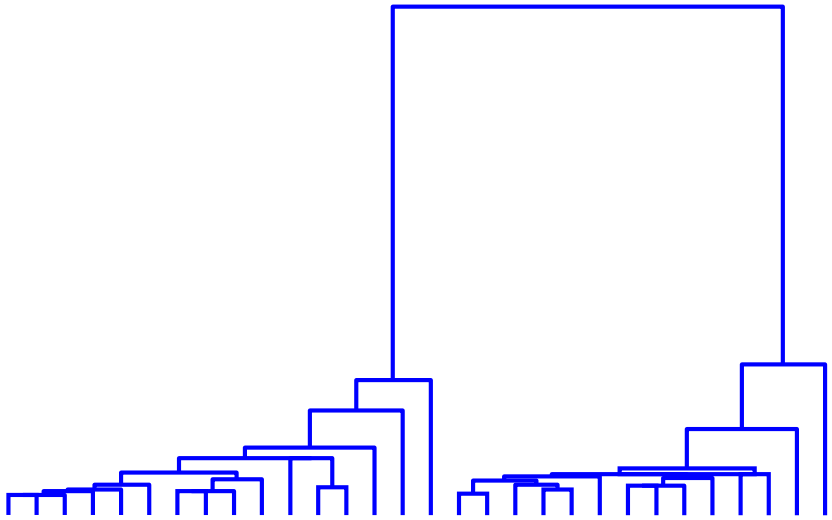
DIANA (Divisive Analysis)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Inverse order of AGNES
- Eventually each node forms a cluster on its own

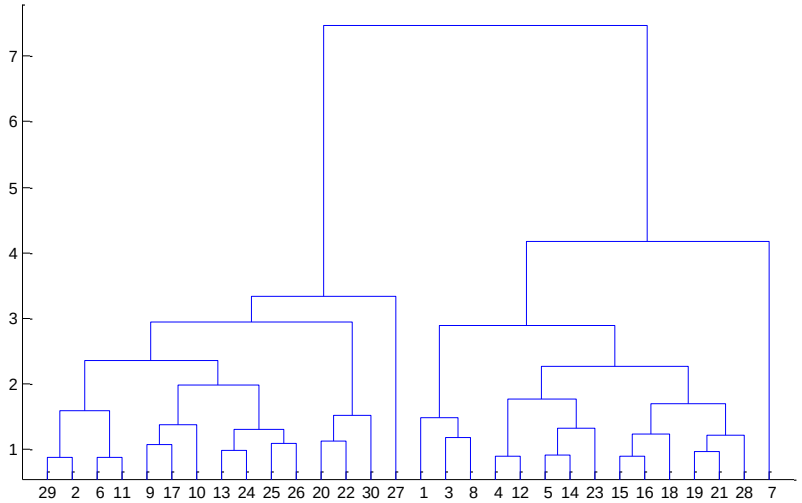




Slide based on one by Eamonn Keogh



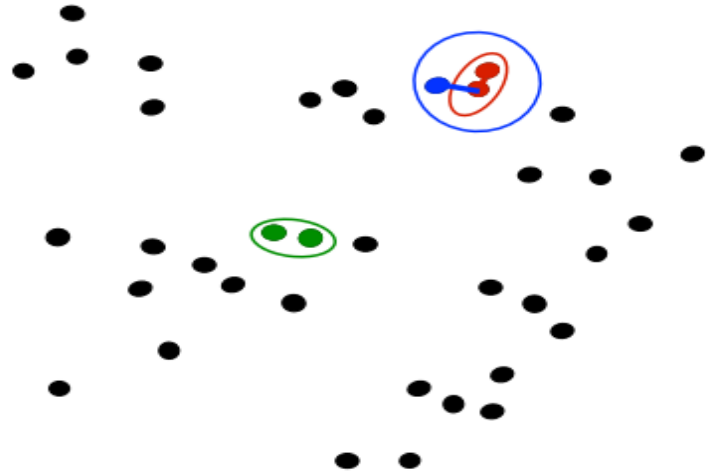
Single linkage



Average linkage

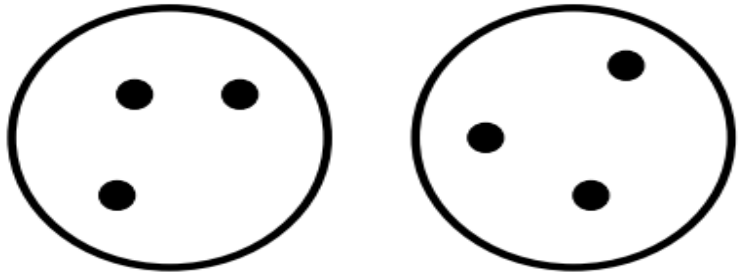
Agglomerative Clustering

- Agglomerative clustering:
  - First merge very similar instances
  - Incrementally build larger clusters out of smaller clusters
- Algorithm:
  - Maintain a set of clusters
  - Initially, each instance in its own cluster
  - Repeat:
    - Pick the two closest clusters
    - Merge them into a new cluster
    - Stop when there's only one cluster left
- Produces not one clustering, but a family of clusterings represented by a dendrogram



Agglomerative Clustering

- How should we define “closest” for clusters with multiple elements?

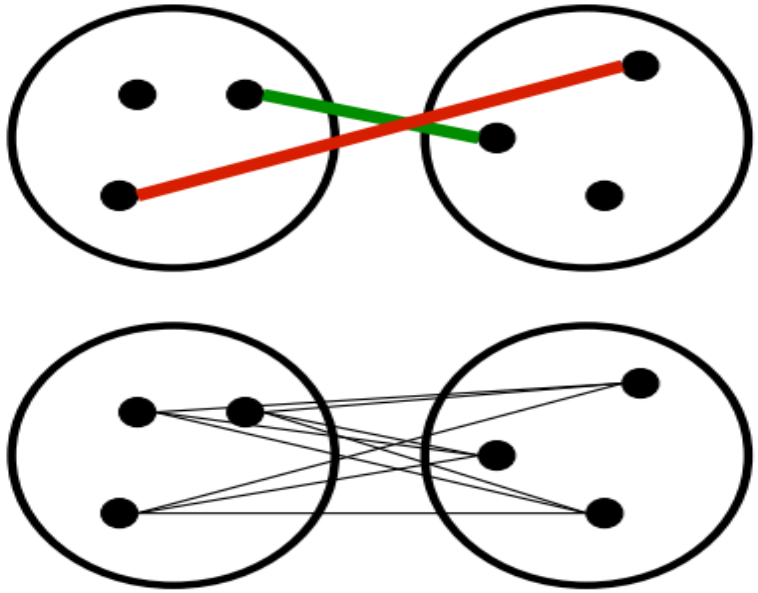


Agglomerative Clustering

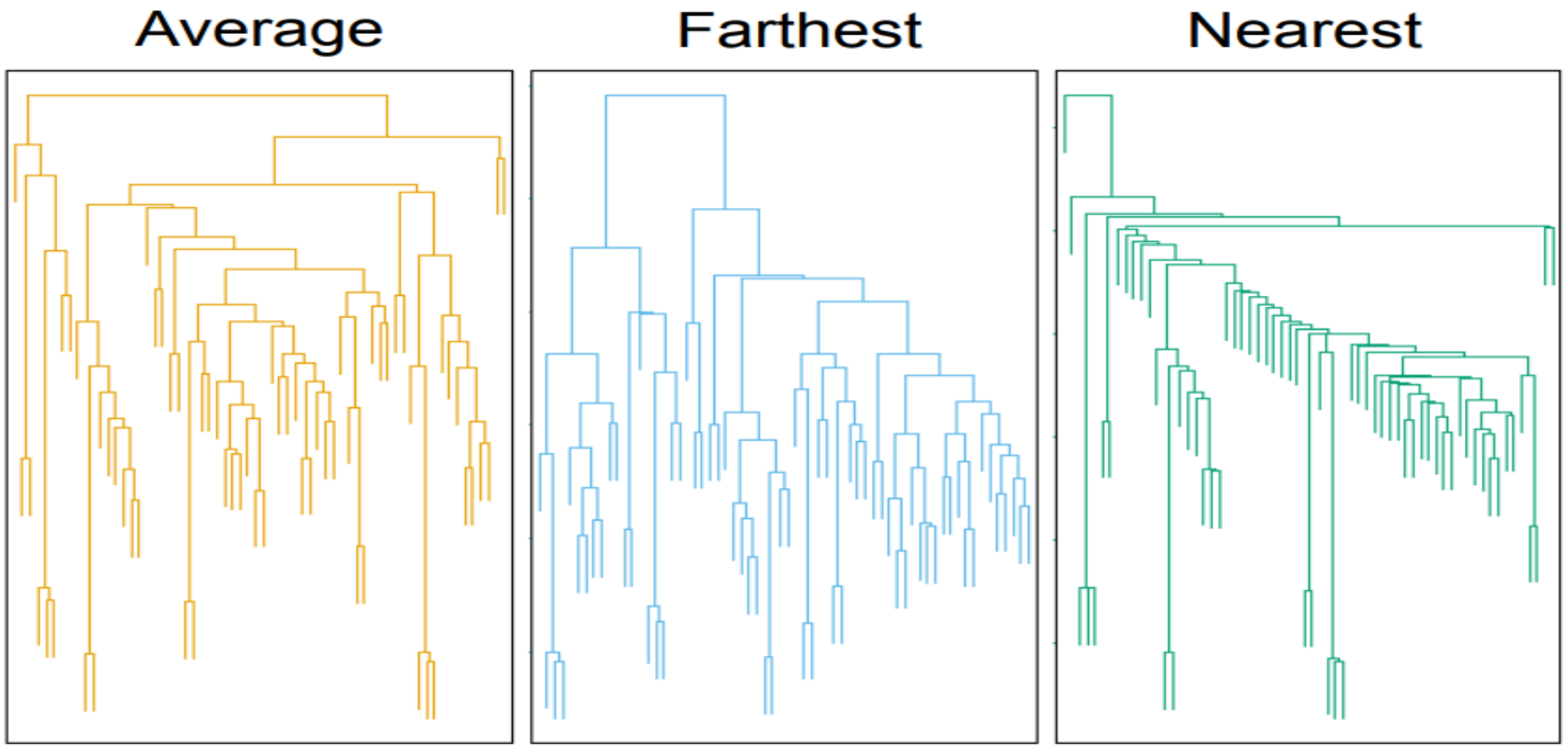
- How should we define "closest" for clusters with multiple elements?

- Many options:
  - Closest pair (single-link clustering)
  - Farthest pair (complete-link clustering)
  - Average of all pairs

- Different choices create different clustering behaviors



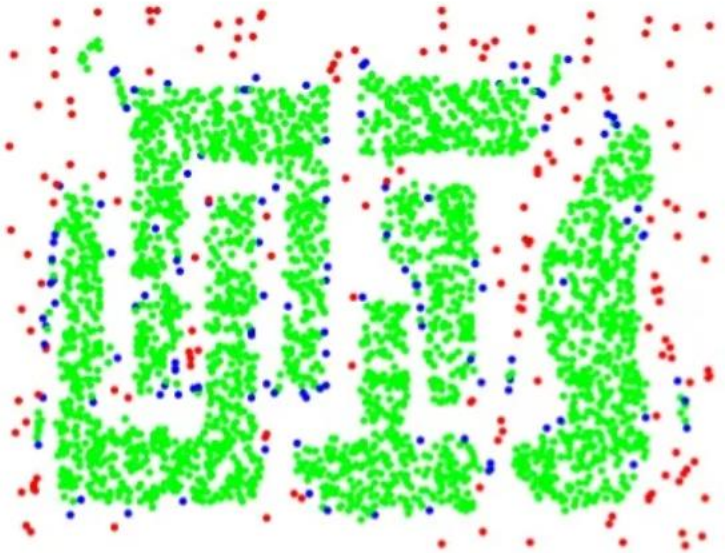
Clustering Behavior



Mouse tumor data from [Hastie *et al.*]

Concepts: Preliminary

Original Points



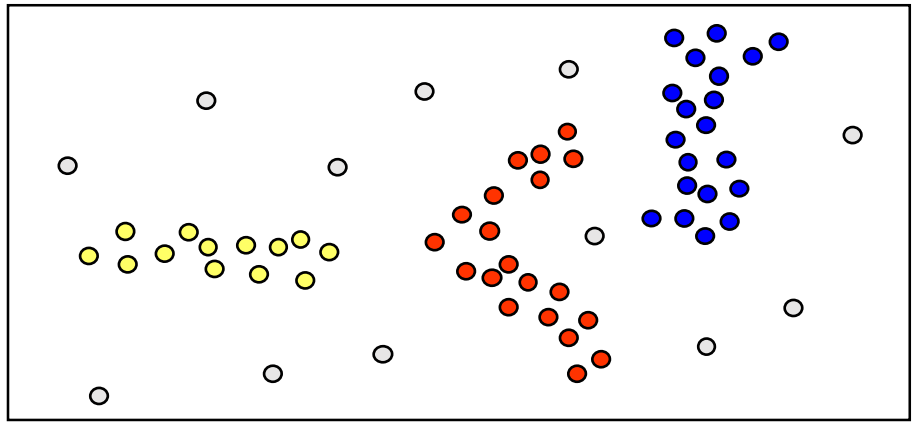
Point types: core, border and noise

Eps = 10, MinPts = 4

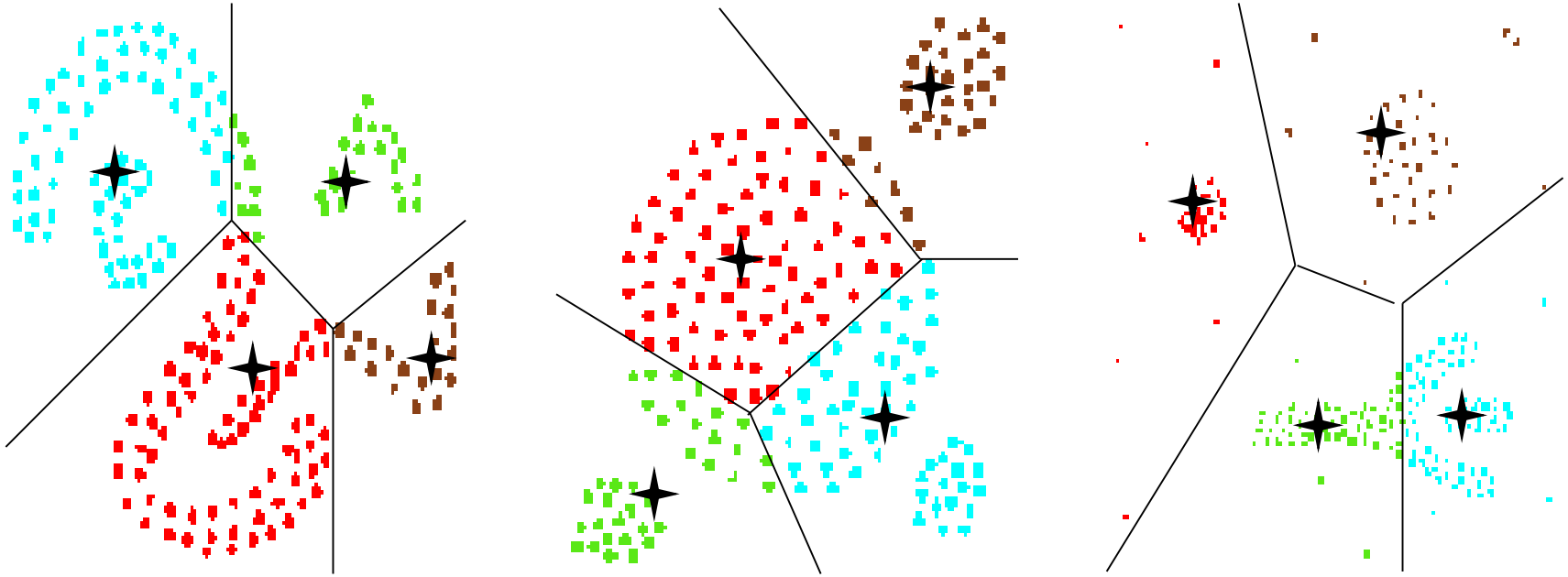
Density-Based Clustering

- *Basic Idea*:

  Clusters are dense regions in the data space, separated by regions of lower object density

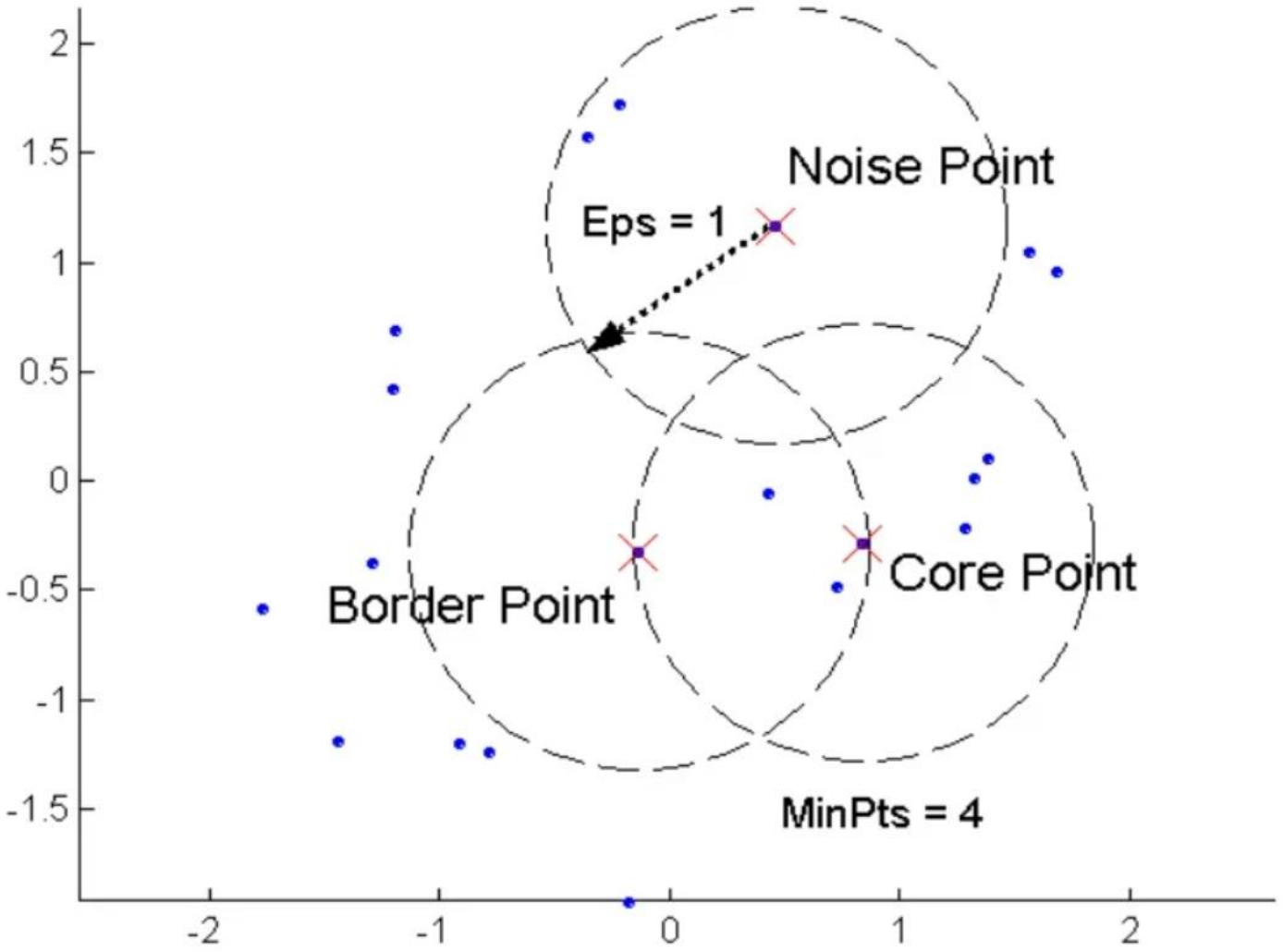


- **Why Density-Based Clustering?**



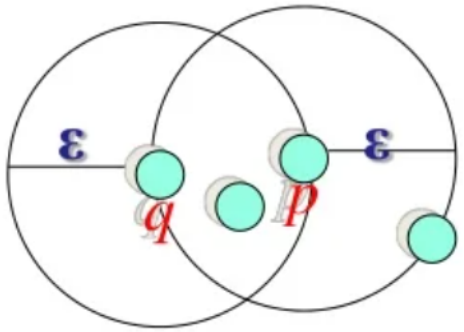
Results of a *k*-medoid algorithm for *k*=4

Concepts: Core, Border, Noise



Concepts: ε-Neighborhood

- ε-Neighborhood - Objects within a radius of ε from an object. (epsilon-neighborhood)
- Core objects - ε-Neighborhood of an object contains at least MinPts of objects



ε-Neighborhood of *p*
ε-Neighborhood of *q*

p is a core object (MinPts = 4)

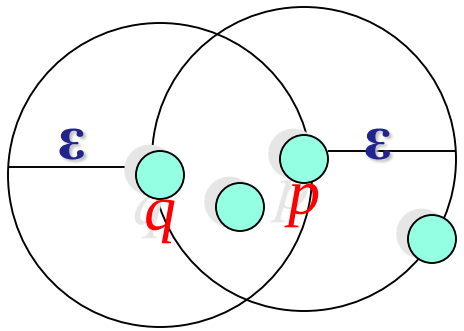
q is not a core object

ε-Neighborhood

- ε-Neighborhood – Objects within a radius of ε from an object.

$$N_\varepsilon(p) : \{q \mid d(p,q) \leq \varepsilon\}$$

- "High density" - ε-Neighborhood of an object contains at least *MinPts* of objects.



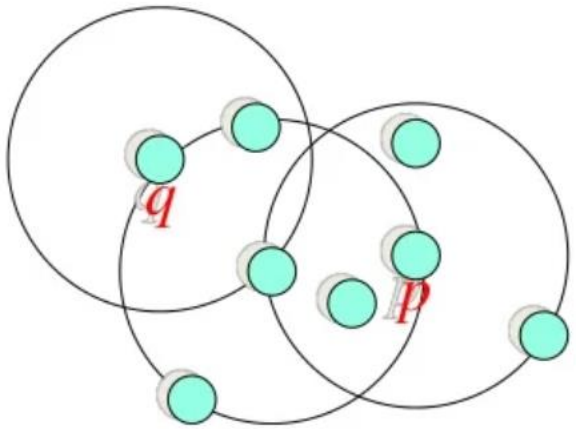
ε-Neighborhood of p

ε-Neighborhood of q

Density of p is "high" (MinPts = 4)

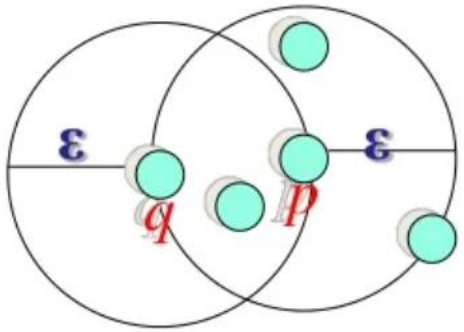
Density of q is "low" (MinPts = 4)

DBScan : Reachability



DBScan : Reachability

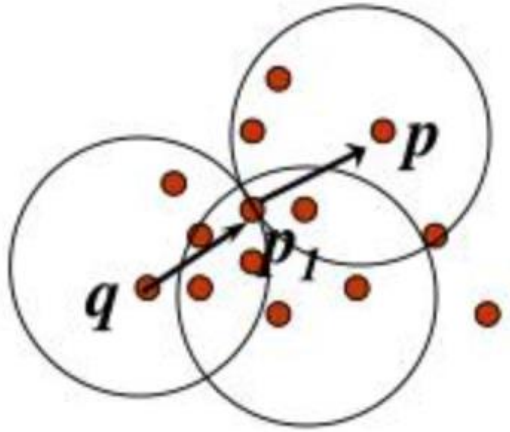
- **Directly density-reachable**
  - An object q is directly density-reachable from object p if q is within the ε-Neighborhood of p and p is a core object.



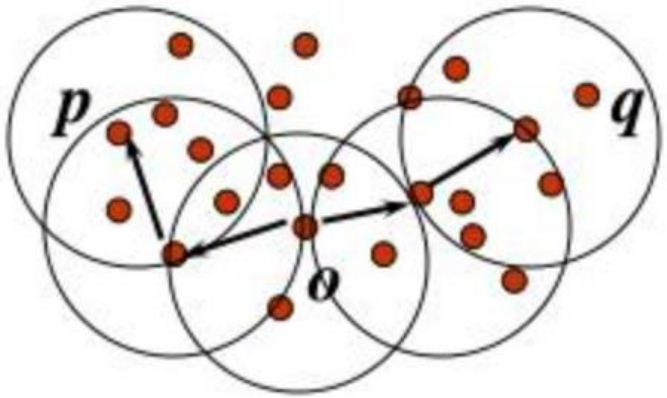
- q is directly density-reachable from p
- p is not directly density-reachable from q.

Density-Reachable and Density-Connected

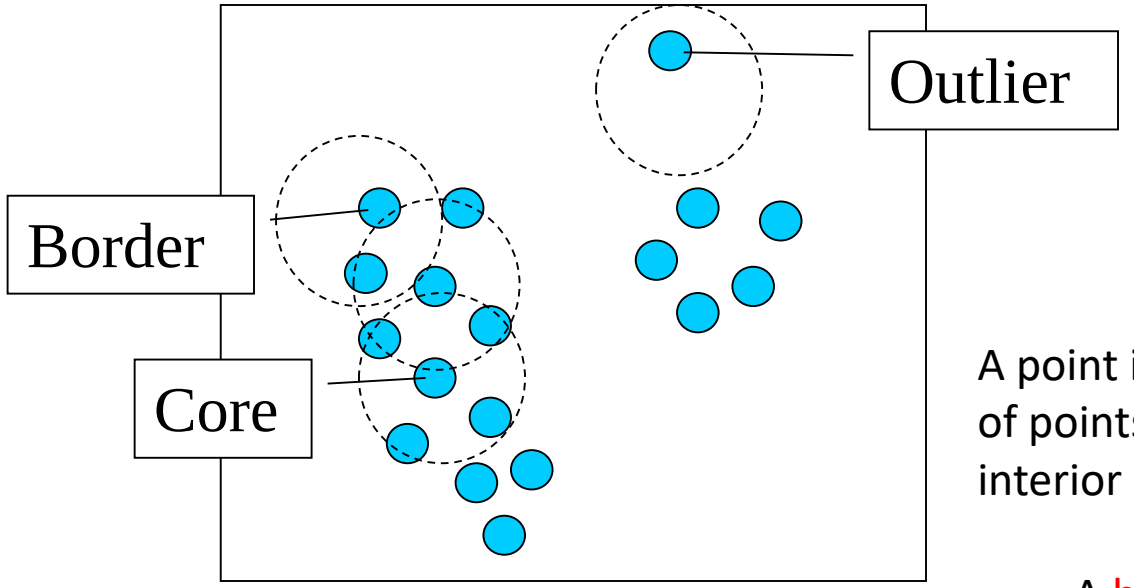
- Density-reachable:
  - A point *p* is density-reachable from a point *q* w.r.t. *Eps*, *MinPts* if there is a chain of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i



- Density-connected
  - A point *p* is density-connected to a point *q* w.r.t. *Eps*, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. *Eps* and *MinPts*



Core, Border & Outlier



Given ε and *MinPts*, categorize the objects into three exclusive groups.

ε = 1unit, MinPts = 5

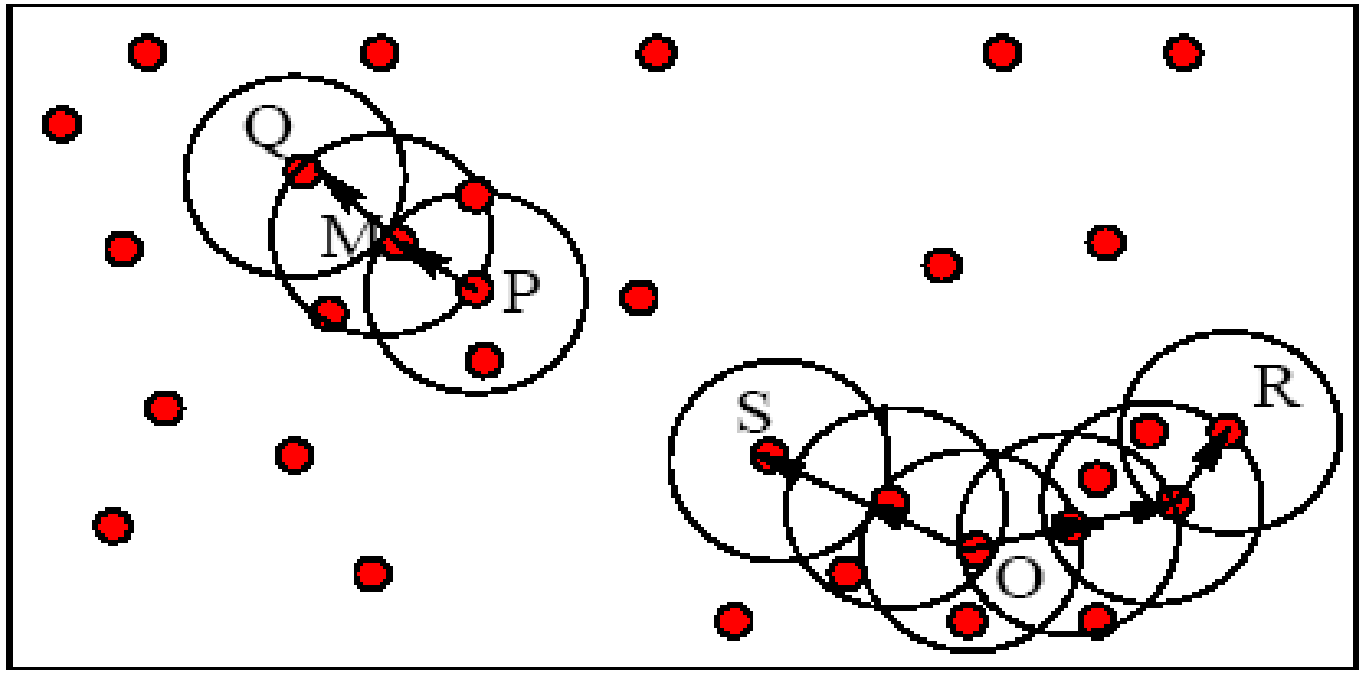
A point is a core point if it has more than a specified number of points (MinPts) within Eps These are points that are at the interior of a cluster.

A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.

A noise point is any point that is not a core point nor a border point.

Example

- M, P, O, and R are core objects since each is in an Eps neighborhood containing at least 3 points



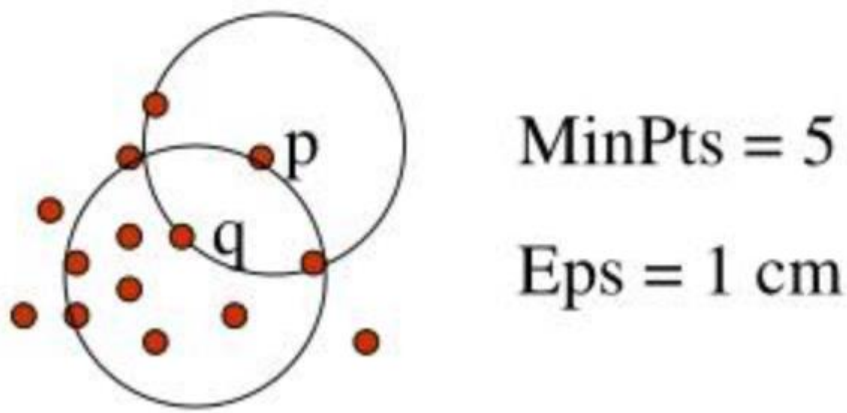
Minpts = 3

Eps=radius
of the circles

Density-Based Clustering: Basic Concepts

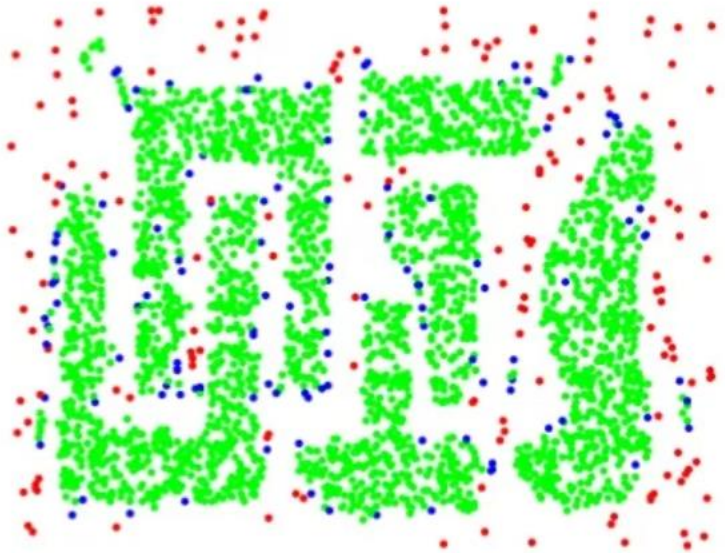
- Two parameters:
  - *Eps*: Maximum radius of the neighbourhood
  - *MinPts*: Minimum number of points in an Eps-neighbourhood of that point
- $N_{Eps}(p)$: $\{q \text{ belongs to } D \mid dist(p,q) <= Eps\}$
- Directly density-reachable: A point *p* is directly density-reachable from a point *q* w.r.t. *Eps*, *MinPts* if
  - *p* belongs to $N_{Eps}(q)$
  - core point condition:

$$|N_{Eps}(q)| >= MinPts$$



Core, Border, Noise points representation

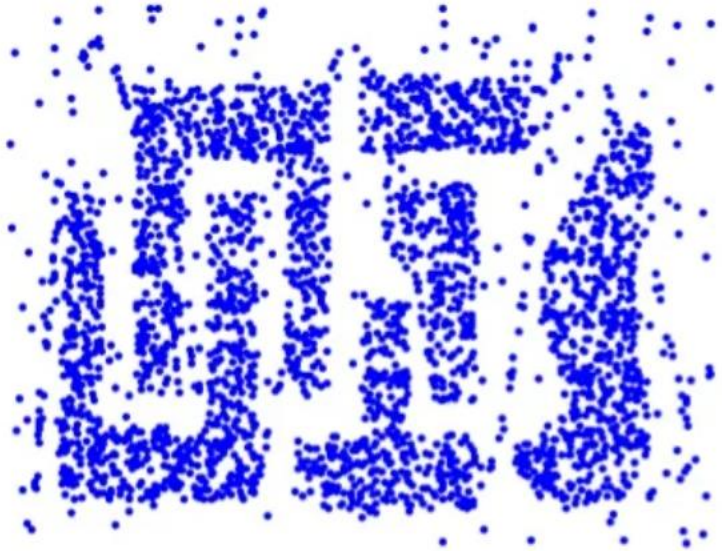
Original Points



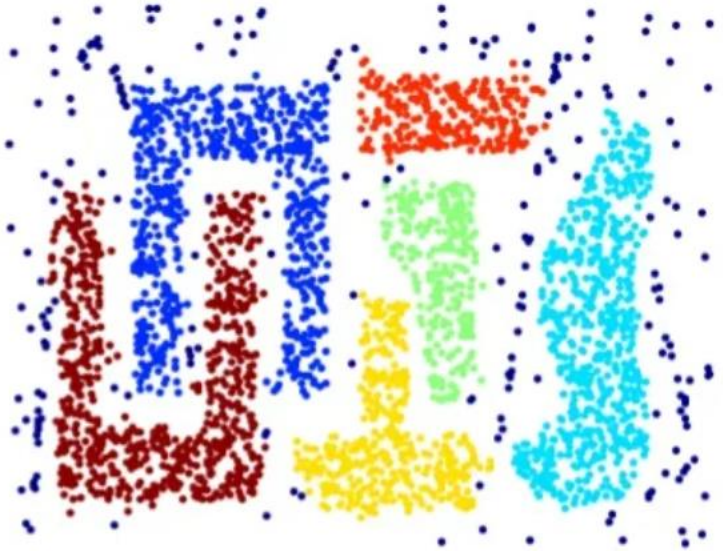
Point types: core, border and noise

Eps = 10, MinPts = 4

Clustering



Original Points



Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

DBScan Algorithm

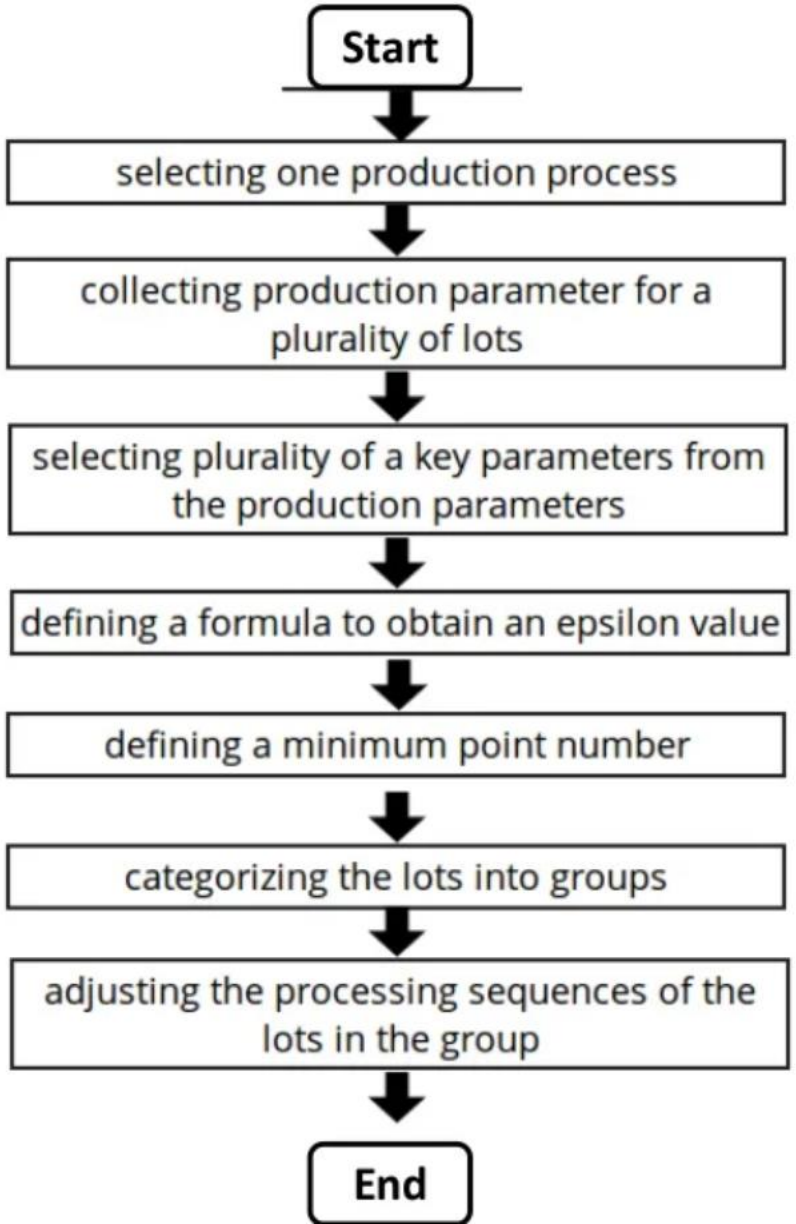
Input: N objects to be clustered and global parameters *Eps*, *MinPts*.

Output: Clusters of objects.

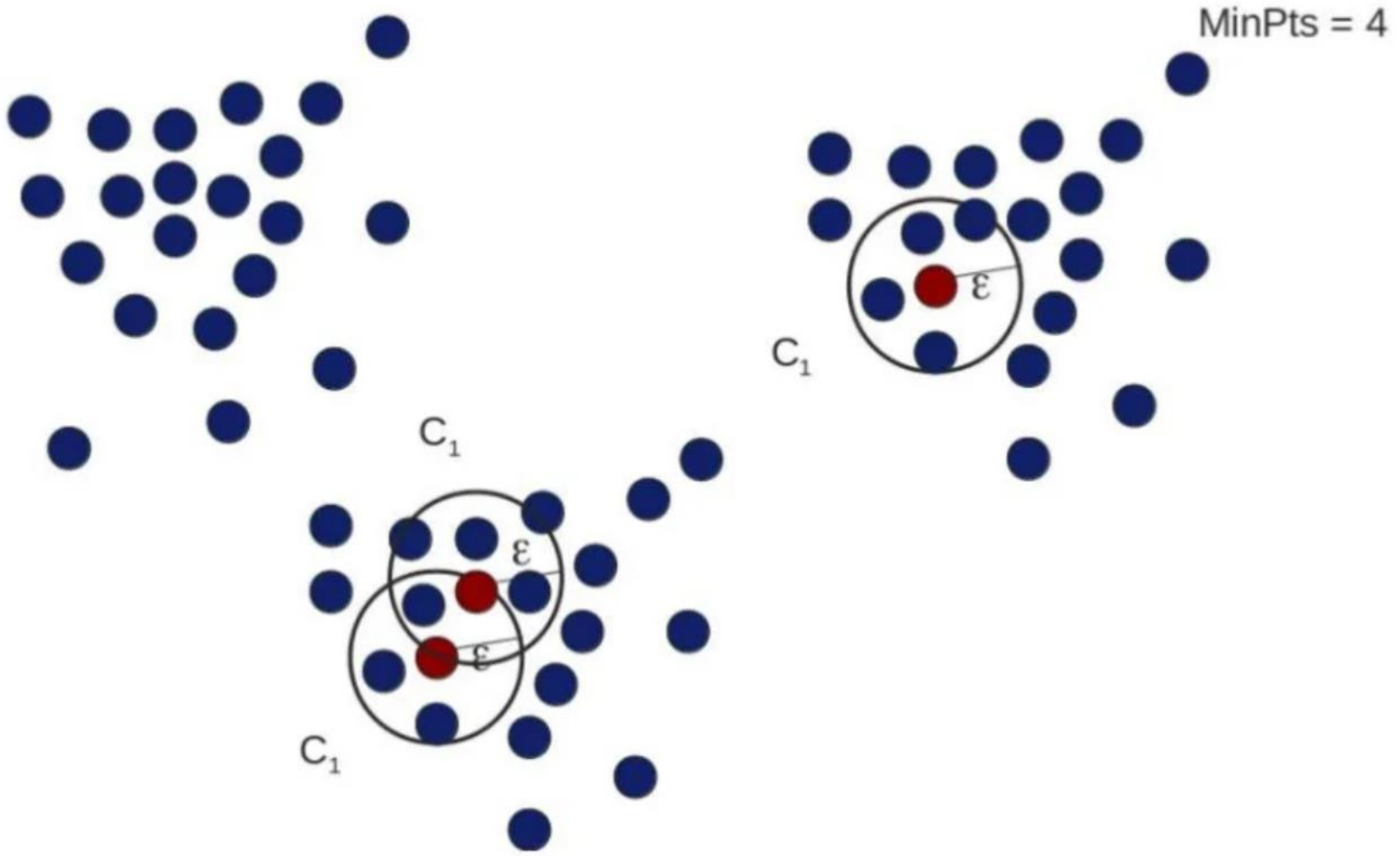
Algorithm:

1) Arbitrary select a point *P*.
2) Retrieve all points density-reachable from *P* wrt ***Eps*** and ***MinPts***.
3) If *P* is a core point, a cluster is formed.
4) If *P* is a border point, no points are density-reachable from *P* and **DBSCAN** visits the next point of the database.
5) Continue the process until all of the points have been processed.

DBScan :Flowchart

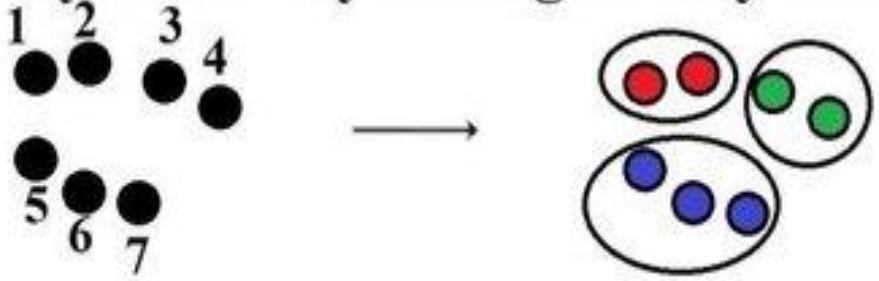


DBScan : Example



A Hard Clustering

- Every node may belong to only one cluster



Community Affiliation

	Nodes						
	1	2	3	4	5	6	7
Cluster 1	1	1	0	0	0	0	0
Cluster 2	0	0	1	1	0	0	0
Cluster 3	0	0	0	0	1	1	1

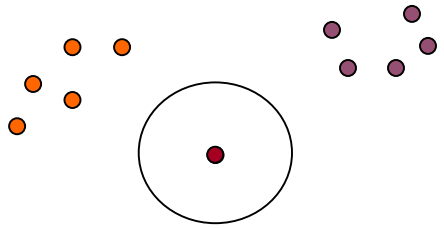
B Soft Clustering

- Every node may belong to several clusters with a fractional degree of membership in each

	Nodes						
	1	2	3	4	5	6	7
Cluster 1	1	1	0	0	1	0	0
Cluster 2	0	0	1	1	0	0	1
Cluster 3	0	0	0	0	1	1	1

DBSCAN Algorithm: Example

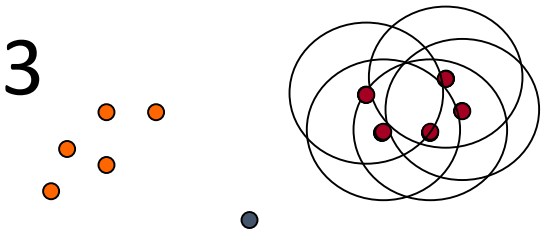
- Parameter
  - ε = 2 cm
  - *MinPts* = 3



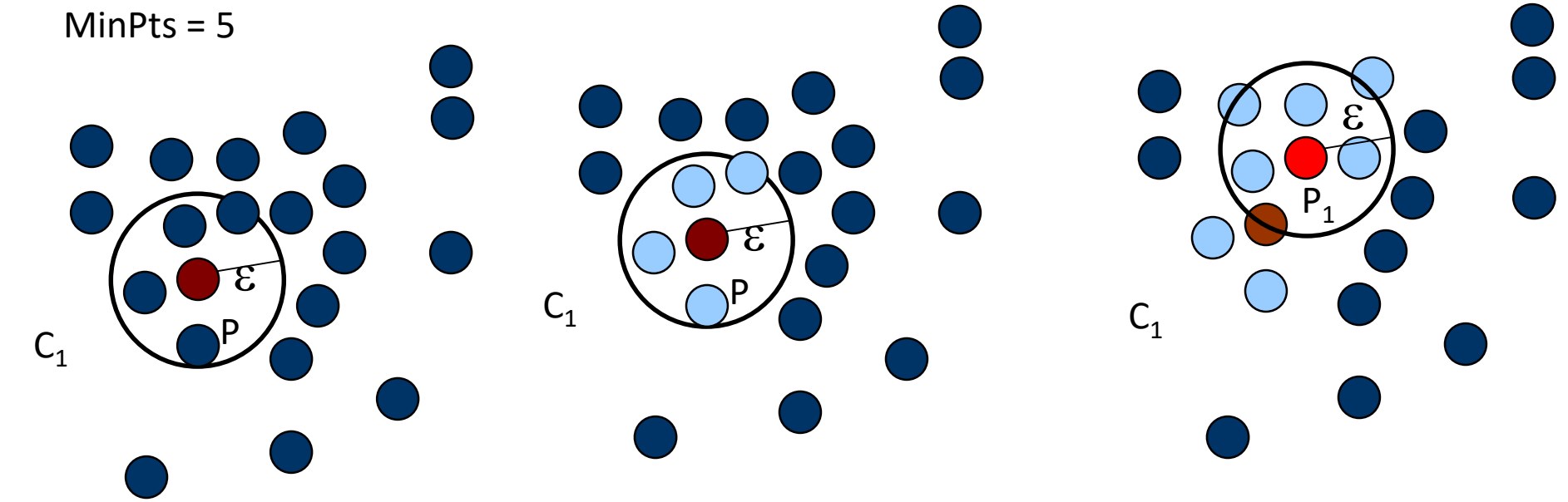
for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

- Parameter
  - ε = 2 cm
  - *MinPts* = 3

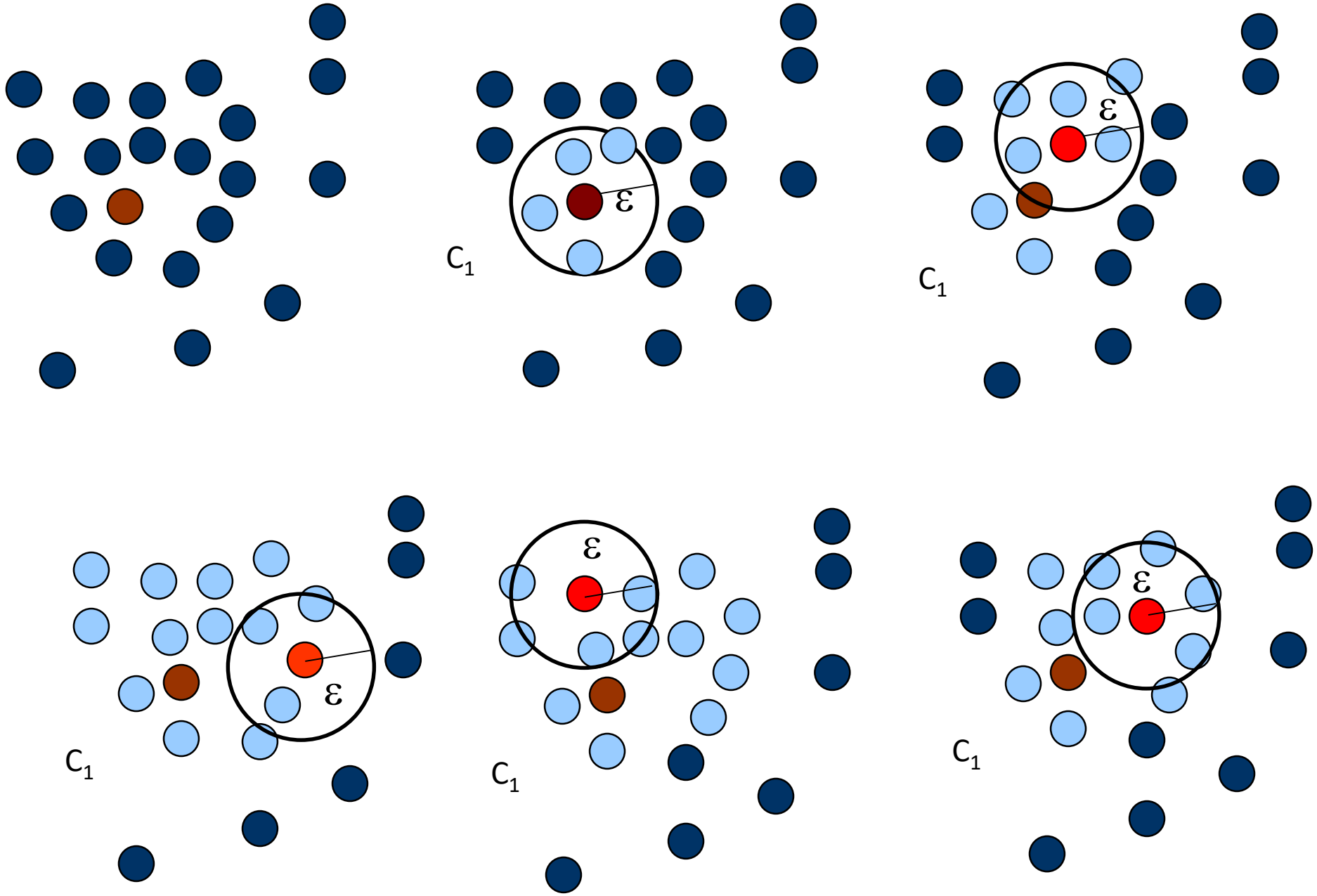


for each $o \in D$ **do**
  if o is not yet classified **then**
    if o is a core-object **then**
      collect all objects density-reachable from o
      and assign them to a new cluster.
    else
      assign o to NOISE



1. Check the ε-neighborhood of p;
2. If p has less than MinPts neighbors then mark p as outlier and continue with the next object
3. Otherwise mark p as processed and put all the neighbors in cluster C

1. Check the unprocessed objects in C
2. If no core object, return C
3. Otherwise, randomly pick up one core object p_1, mark p_1 as processed, and put all unprocessed neighbors of p_1 in cluster C


ε
C1
ε
C1
ε
C1
ε
C1
ε
C1
ε
C1

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

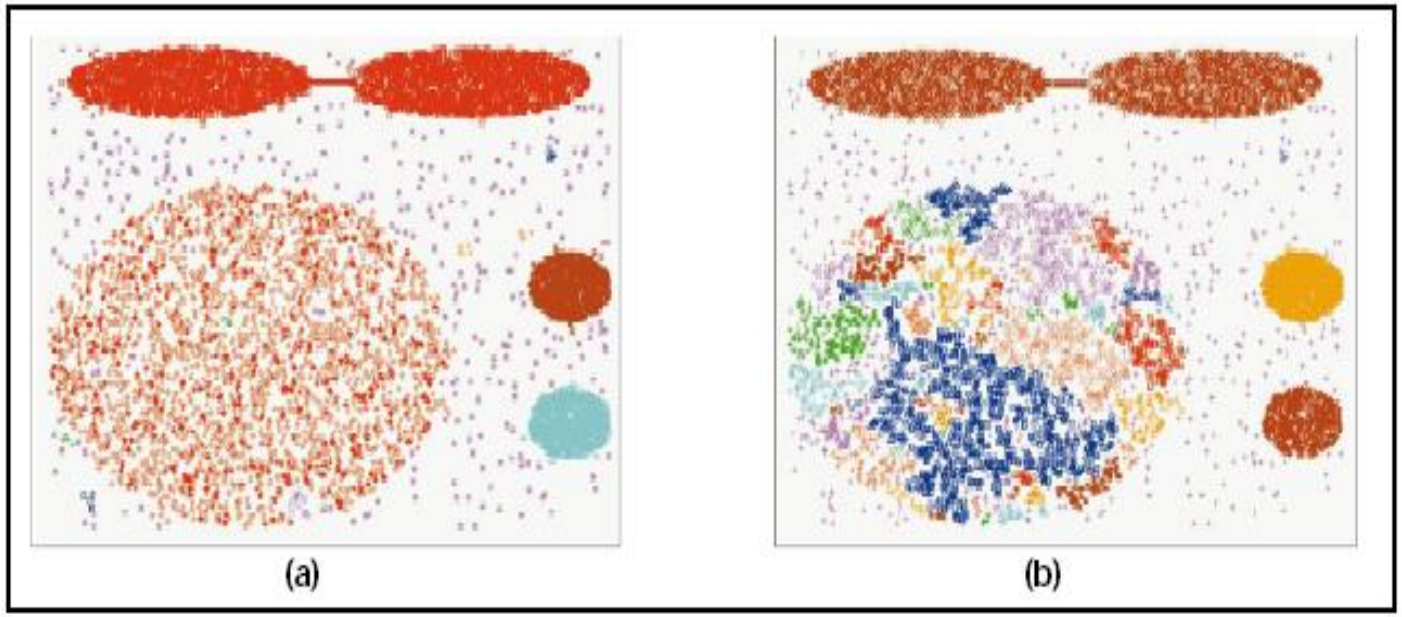


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.