Lecture 7-8:

Data clustering

Outline

- Clustering
  - Main concepts
  - Clustering validation measures
- Partitional algorithms
  - kMeans
  - fuzzy cMeans
- Hierarchical algorithms
  - Agglomerative
  - Divisive
- Density based clustering (DBSCAN, DENCLUE)
- Clustering based on probabilistic models (Expectation Maximization)

Aim of clustering (reminder)

What is known?

- A set of data (not necessarily structured)
- A similarity/dissimilarity measure between data (the measure is specific to the problem) based on which is constructed the similarity/dissimilarity matrix

What is desired?

- A model describing the grouping of data in clusters such that data belonging to the same cluster are more similar than data belonging to different clusters

Which is the final aim?

- Check if two data belong to the same cluster
- Find the most appropriate cluster for a new data

Remark: for some clustering methods it is enough to know the matrix of (dis)similarity values

Aim of clustering (reminder)

Examples:

- Customer segmentation = identify groups of customers with similar shopping behaviors
- User profiles extraction = identify groups of users of an e-commerce system or a web service characterized by similar behaviors
- Data summarization / document clustering = identify groups of electronic documents based on their content
- Image segmentation = identify homogeneous regions in an image

Related to

- Communities detection = identify tightly clustered nodes in a network

Clustering allows to:

- Summarize and/or visualize in a different form the data in order to understand them better

Particularities of clustering

It is an unsupervised process:

- The training set contains only the values of the attributes
- The class labels are not known before clustering

The clustering task is ill-defined:

- identifying the clusters is not easy
- It can be a subjective decision

How many clusters?

Six Clusters

Two Clusters

Four Clusters

[images from slides by Kumar, 2004]

Main concepts

- Cluster = group of data which are "similar enough"
- (Di)similarity matrix for a set of n data instances = matrix of n rows and n columns with the (di)similarity between any two data instances
- Clustering

  = set of clusters

  = process of identifying the clusters
- Cluster prototype = "object" which is representative for the data in the cluster
  - Centroid = the mean of the data in the cluster – the centroid is not necessary a data from the cluster
  - Medoid = the data instance from the cluster which is closest to the mean of the cluster – the medoid is one of the data in the cluster
- Cluster radius = average of the distances between the data in the cluster and the cluster prototype
- Cluster diameter = maximum of the distance between two data in the cluster

Types of clustering

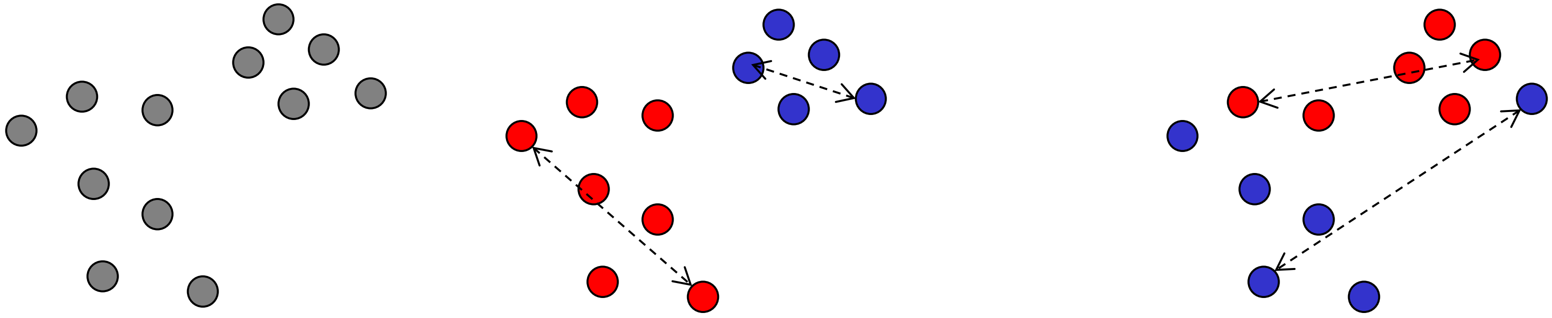
- Crisp vs fuzzy clustering
  - Crisp clustering = each data instance belongs to only one cluster
  - Fuzzy clustering = a data can belong to several clusters and for each cluster there is a membership degree

- Flat vs hierarchical clustering
  - Flat (partitional) clustering = the result is one partition (set of clusters)
  - Hierarchical clustering = the result is a hierarchy of partitions

- Variants of algorithms
  - Partitional algorithms (e.g: kMeans, Fuzzy cMeans)
  - Hierarchical algorithms (e.g. agglomerative algorithm, divisive algorithm)
  - Density based algorithms (e.g. DBSCAN)
  - Probabilistic algorithms (e.g. EM = Expectation Maximization)

Clustering validation measures

- There is no unique indicator which measures the quality of a clustering result
- The most straightforward approach is to estimate:
  - The compactness of data in one cluster (intra-cluster variability – it should be small)
  - The degree of separation between data belonging to different clusters (inter-cluster distance – it should be large)



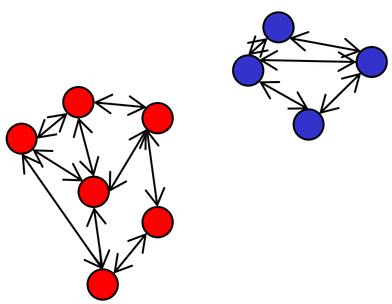
An acceptable clustering   A lower quality clustering

Clustering validation measures

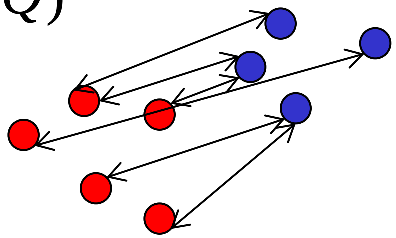
- Intra-cluster to inter-cluster distance ratio = Intra/Inter (smaller values correspond to better clustering)
- Let P be the set of pairs of data instances which belong to the same cluster and Q=DxD-P (the rest of pairs: one data belongs to one cluster and the other data belongs to another cluster)

$$Intra = \sum_{(x_i, x_j) \in P} d(x_i, x_j) / card(P)$$

$$Inter = \sum_{(x_i, x_j) \in Q} d(x_i, x_j) / card(Q)$$



Examples of paired distances involved in the computation of the intra measure



Examples of paired distances involved in the computation of the inter measure

Clustering validation measures

- Silhouette coefficient (it measures the difference between the similarity of an object and its own cluster (cohesion) and its similarity to other clusters (separation))

$$S_i = \frac{D\min_i^{out} - Davg_i^{in}}{\max\{D\min_i^{out}, Davg_i^{in}\}}$$

$$S = \frac{1}{n}\sum_{i=1}^{n} S_i$$

$$Davg_i^{in} = \text{the average of the distances between } x_i \text{ and all other data in the cluster of } x_i$$

$$Davg_i^{j} = \text{the average of the distances between } x_i \text{ and all data in another cluster } j\ (j \neq i)$$

$$D\min_i^{out} = \min_j Davg_i^{j}$$

Remark:

- S takes values in (-1,1)
- Larger values indicate better clustering

kMeans

- Input: data set $D=\{x_1,x_2,\ldots,x_N\}$, K = number of clusters
- Output: a partition $P=\{C_1,C_2,\ldots,C_K\}$ of D

kMeans (D,k)

initialize the centroids c_1, c_2, …, c_K (by random selection from the data set or by using a pre-clustering method)

repeat

- assign each data from D to the cluster corresponding to the closest centroid (with respect to a similarity/distance)
- update each centroid as mean of the data belonging to the corresponding cluster

until <the partition does not change>

Remark: this is the so-called Lloyd variant

kMeans

Characteristics

- kMeans is a center based clustering method which aims to minimize the total sum of squared errors (SSE) – distances between data and their corresponding centroids

$$SSE = \sum_{k=1}^{K} \sum_{x \in C_k} d^2(x, c_k) = \sum_{k=1}^{K} \sum_{x \in C_k} \sum_{j=1}^{n} (x_j - c_{kj})^2$$

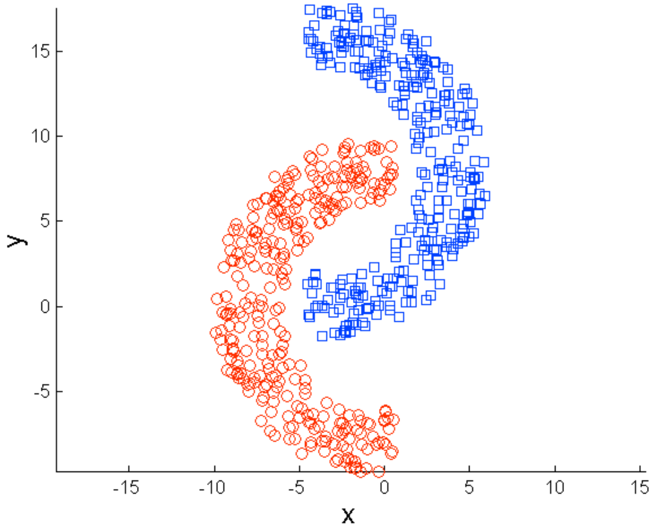
(in the case of Euclidean distance)

- Complexity: O(n*N*K*iterations) (n=number of attributes, N=number of data instances, K=number of clusters)
- Useful pre-processing: normalization
- Useful post-processing:
  - Remove the small clusters
  - Split the loose clusters
  - Merge the close clusters

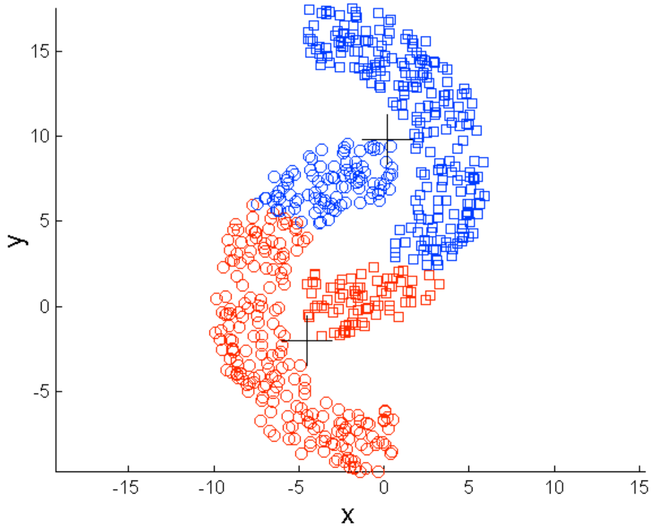
kMeans

Limits:

- It does not work well in the case when the clusters are not "spherical"
  - Solution: use other approaches (e.g. density based clustering)



True clusters



Kmeans result

[from slides Kumar, 2004]

kMeans

Limits: It requires the apriori knowledge of the number of clusters

- Solutions:
  - apply the algorithm for different values of K and select the variant with the best values of the validation criteria
  - Post-process the clustering results by splitting the clusters which are not compact enough and by merging clusters which are close one to each other (e.g. ISODATA algorithm)

ISODATA

Main ideas of ISODATA

- If a cluster size is smaller than Nmin then the cluster should merge with another cluster (the closest one)

- If the distance between two clusters (e.g. the distance between the clusters' prototypes) is smaller than Dmin then the clusters should be merged

- If the variance of a cluster is higher than Vmax and the number of data instances it contains is larger than 2*Nmin then the cluster should be divided in two other clusters:
  - Identify the attribute j for which the variance is maximal
  - From prototype c_k two other prototypes c' and c'' are constructed by replacing the value of attribute j from c_k with $c_k(j)-b$ and $c_k(j)+b$, respectively (b is a user parameter)

Fuzzy cMeans

Main idea of fuzzy (soft) clustering:

- A data instance does not belong only to one cluster but it can belong to several clusters (with a given membership degree for each cluster)

- The output of fuzzy clustering is a membership matrix M of size NxK

  (N= number of data instances, K= number of clusters);

  M(i,j) = a value in [0,1] which corresponds to the degree of membership of data i to cluster j

- To obtain a crisp clustering each data i is assigned to the cluster j characterized by the largest membership value M(i,j), j=1..K

Fuzzy cMeans

Algorithm

- Initialize the membership matrix (M)
- **Repeat**
  - Compute the centroids(c_k, k=1,…K)
  - Update the membership values (m_{ij}, i=1,…,N, j=1,…,K

  until <no significant changes in the membership function>

Computation of centroids

$$c_j = \frac{\sum_{i=1}^{n} M_{ij}^{p} x_i}{\sum_{i=1}^{n} M_{ij}^{p}}, \quad j = \overline{1,K}$$

p>1 is a parameter (e.g. p=2)

Membership values computation

$$M_{ij} = \frac{1}{\left\|x_i - c_j\right\|^{2/(p-1)} \sum_{k=1}^{K} 1/\left\|x_i - c_k\right\|^{2/(p-1)}}$$

$$i = \overline{1,n},\ j = \overline{1,K}$$

Remark: at the end of the clustering process, the data are assigned to the cluster for which the membership value is maximal

Hierarchical algorithms

Remark: one of the main limits of partitional algorithms is the fact that the number of clusters should be known apriori.

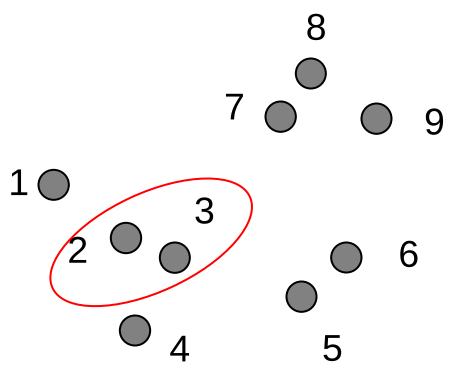
Another approach: construct a hierarchy of partitions

- In a bottom-up manner (agglomerative approach)
  - Start with a partition consisting of one-data clusters (each data belongs to its own cluster)
  - Merge the clusters which are “similar” enough, in an iterative way until all data belong to one cluster

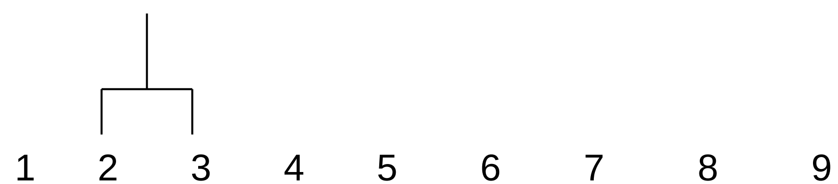
- In a top-down manner (divisive approach)
  - Start with a partition containing one cluster (which contains all data)
  - Divide the “large” clusters by applying a flat clustering (e.g. kMeans) iteratively until the partition consists of singletons (each cluster contains one data instance)

Agglomerative clustering

Idea: identify at each step the most similar clusters and merge them

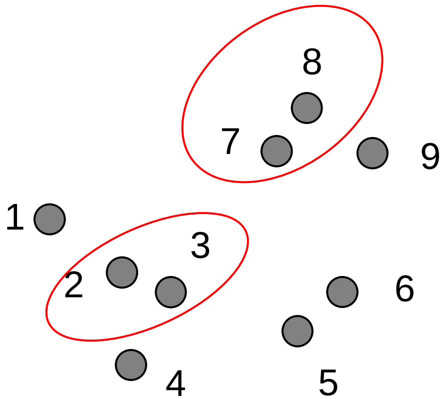


	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0

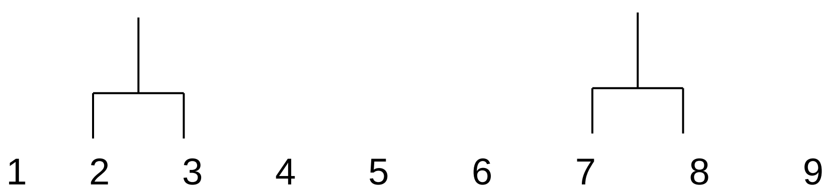


Agglomerative clustering

Idea: identify at each step the most similar clusters and merge them

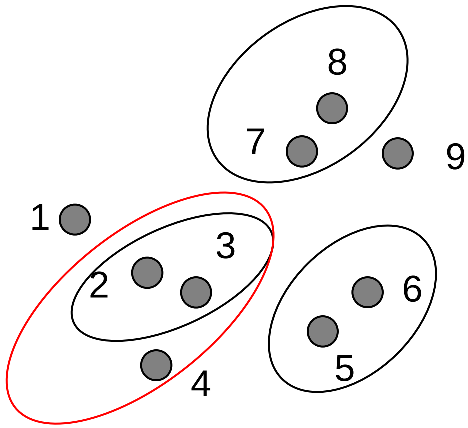


	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0

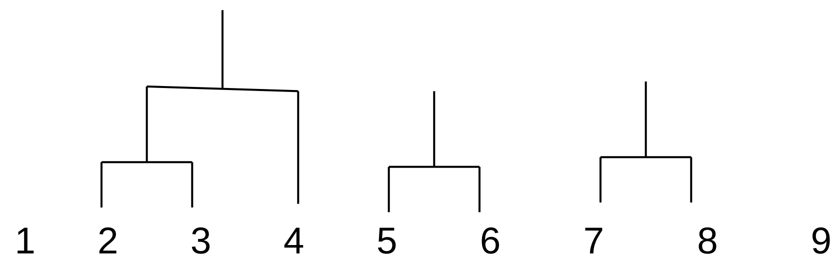


Agglomerative clustering

Idea: identify at each step the most similar clusters and merge them

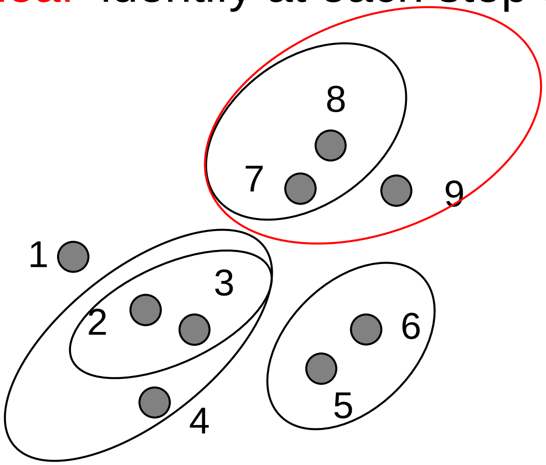


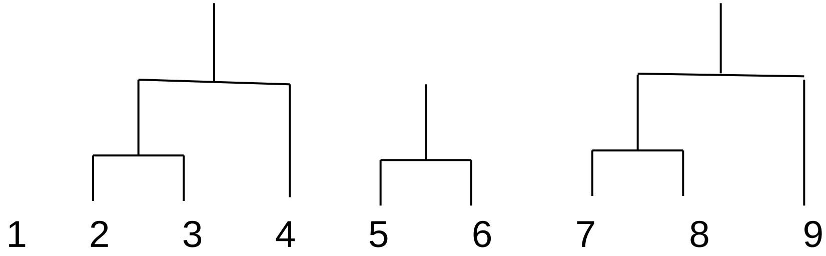
	1	2	3	4	5	6	7	8	9
1	0	2	3	4	7	8	6	8	10
2	2	0	1	2	4	6	7	8	9
3	3	1	0	2	3	5	6	8	9
4	4	2	2	0	3	6	9	10	11
5	7	4	3	3	0	1	4	6	5
6	8	6	5	6	1	0	3	4	3
7	6	7	6	9	4	3	0	1	2
8	8	8	8	10	6	4	1	0	2
9	10	9	9	11	5	3	3	2	0



Agglomerative clustering

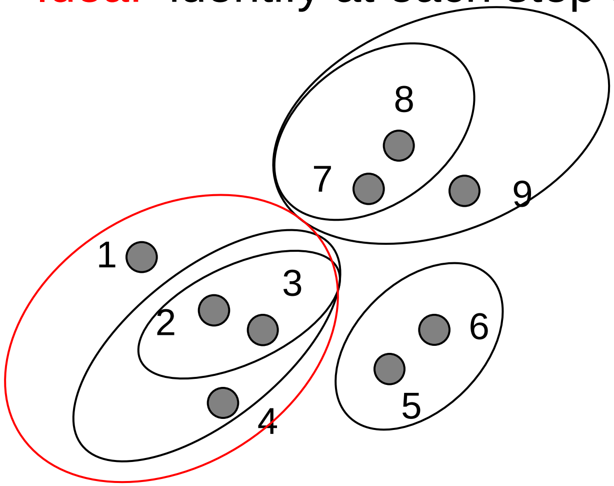
Idea: identify at each step the most similar clusters and merge them

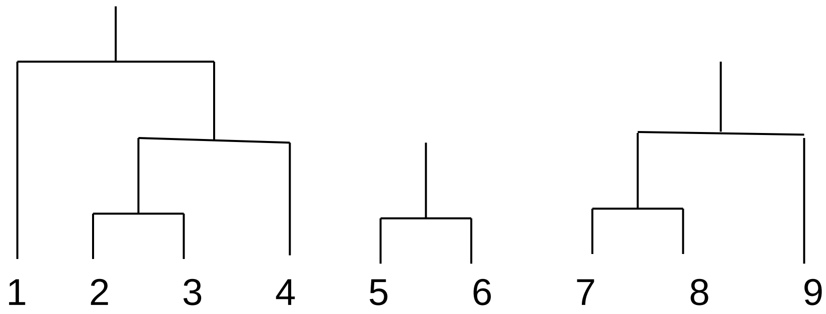




Agglomerative clustering

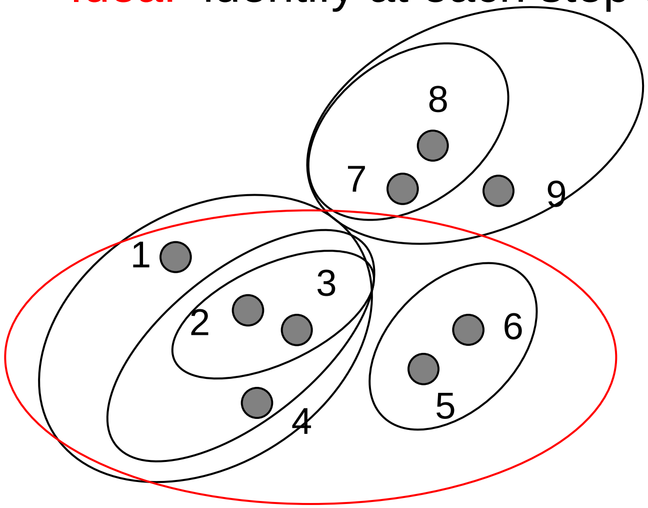
Idea: identify at each step the most similar clusters and merge them

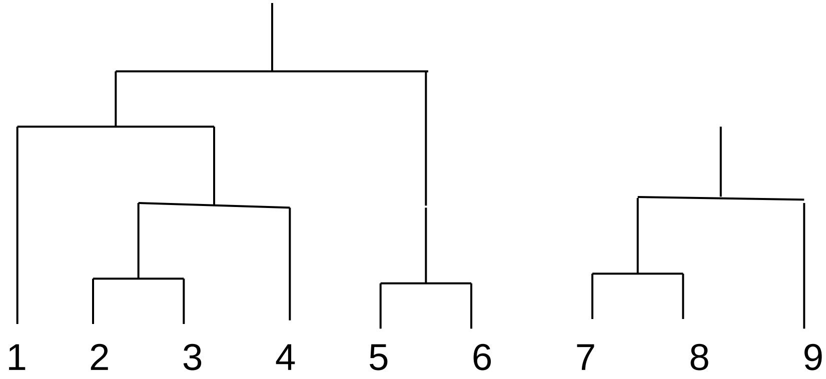




Agglomerative clustering

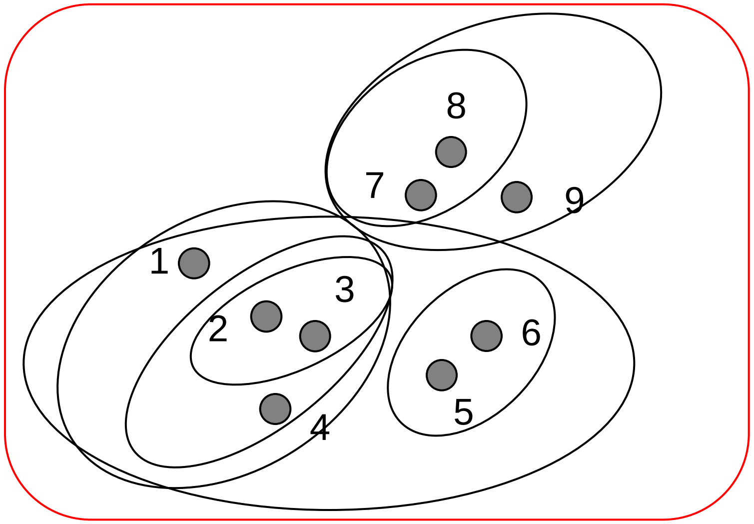
Idea: identify at each step the most similar clusters and merge them



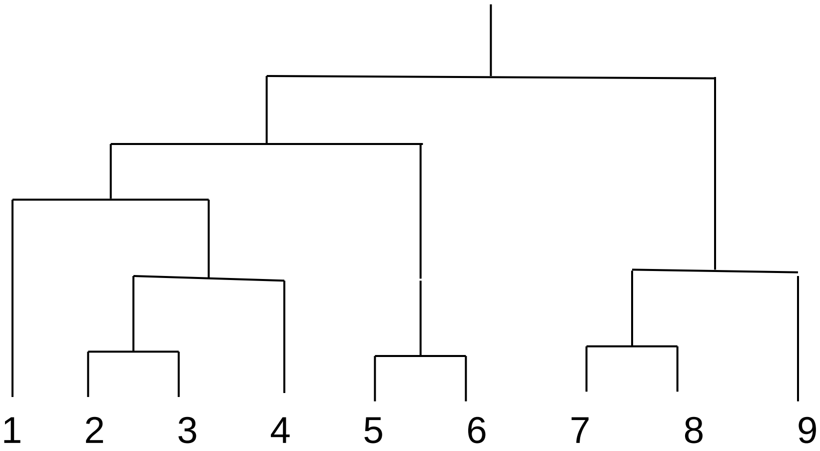


Agglomerative clustering

Idea: identify at each step the most similar clusters and merge them



- The resulting tree is called dendrogram

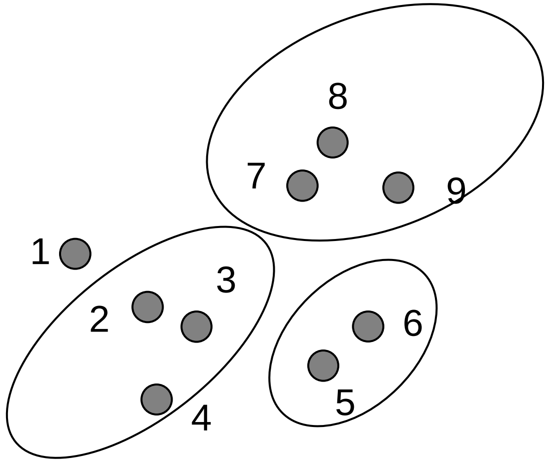


- Representation of the dendrogram: as a set of ordered triples (level, number of clusters, clusters)

{(0,9,{{1},{2},…,{9}}) ,(1,6,{{1},{2,3},{4},{5,6},{7 ,8},{9}}),
(2,4,{{1},{2,3,4},{5,6},{7,8,9}}), (3,3,{{1,2,3,4},{{5,6},{7,8,9}}),
(4,2,{{1,2,3,4,5,6},{7,8,9}),(5,1,{{1,2,3,4,5,6,7,8,9}})}

Agglomerative clustering

Idea: identify at each step the most similar clusters and merge them



- The resulting tree is called dendrogram
- In order to obtain a partition the dendrogram should be cut at a given level

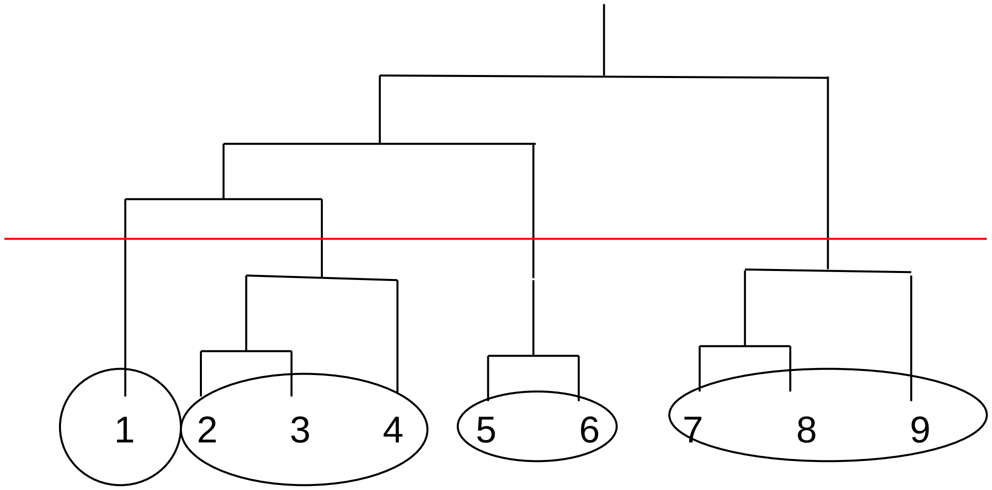
Partition:

C1={1}

C2={2,3,4}

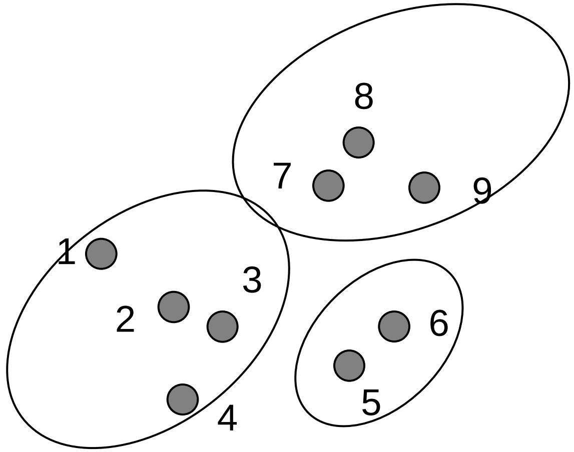
C3={5,6}

C4={7,8,9}

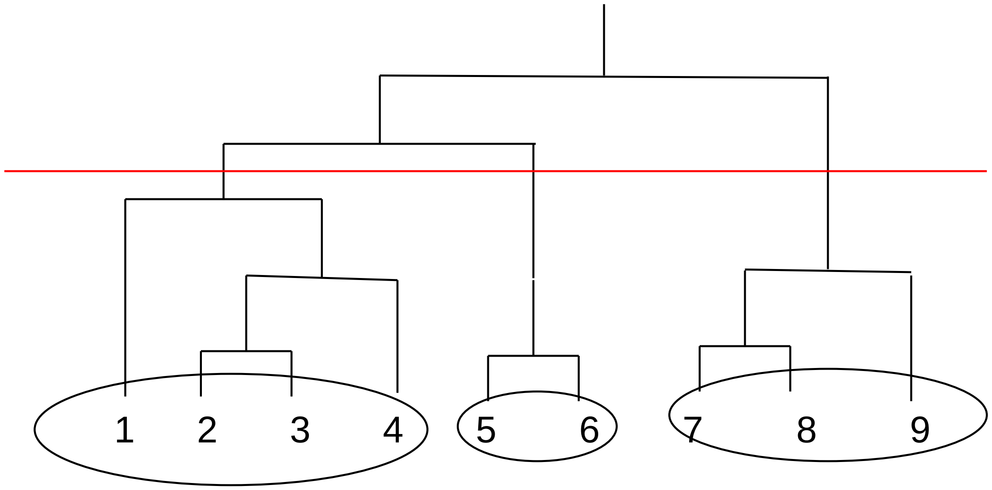


Agglomerative clustering

Idea: identify at each step the most similar clusters and merge them



- By changing the level one obtains a different partition



Partition:

C1={1,2,3,4}

C2={5,6}

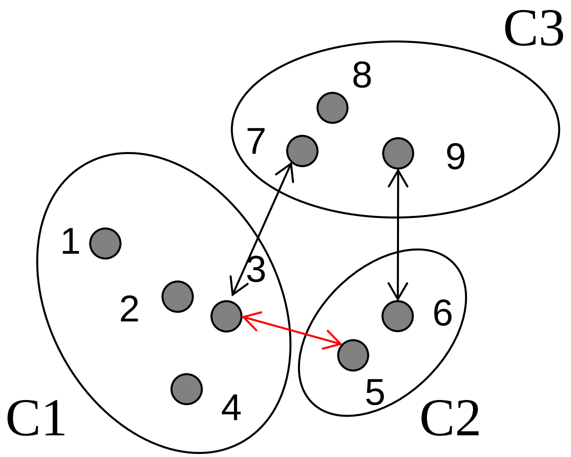
C3={7,8,9}

Agglomerative clustering

Question: how are selected the clusters for merging?

Answer: by using a dissimilarity measure between clusters; there are different ways of computing the dissimilarity measure:

- Single-linkage: the smallest distance between points belonging to different clusters



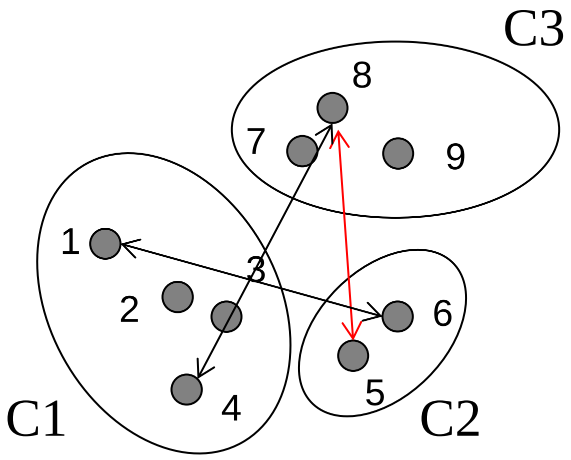
$$D_{SL}(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y)$$

Agglomerative clustering

Question: how are selected the clusters for merging?

Answer: by using a dissimilarity measure between clusters; there are different ways of computing the dissimilarity measure:

- Complete-linkage: the largest distance between points belonging to different clusters



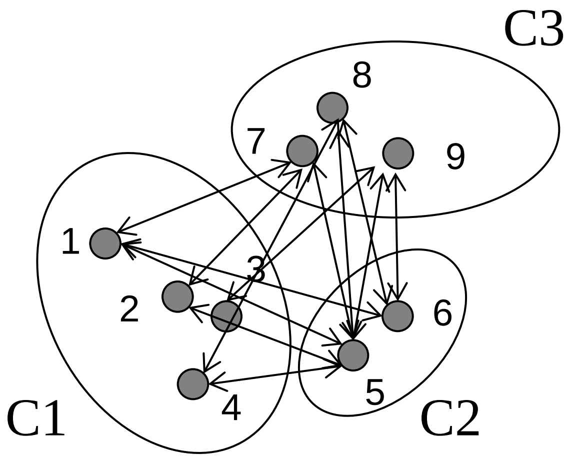
$$D_{CL}(C_i, C_j) = \max_{x \in C_i, y \in C_j} d(x, y)$$

Agglomerative clustering

Question: how are selected the clusters for merging?

Answer: by using a dissimilarity measure between clusters; there are different ways of computing the dissimilarity measure:

- Average-linkage: the average distance between points belonging to different clusters



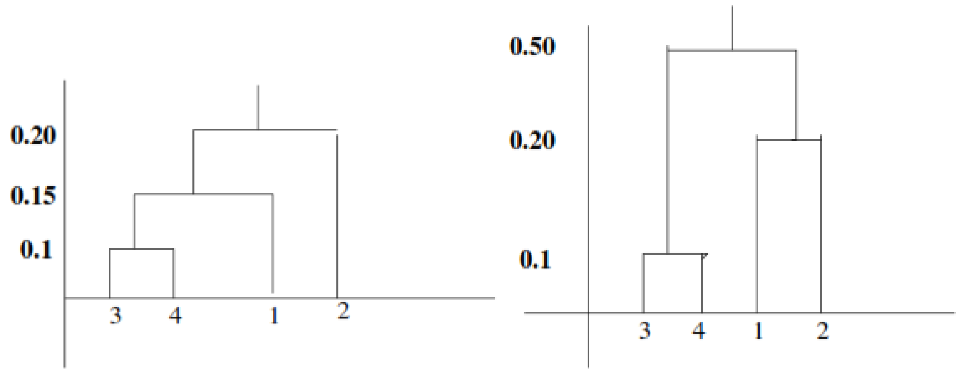
$$D_{AL}(C_i, C_j) = \frac{1}{card(C_i)card(C_j)} \sum_{x \in C_i, y \in C_j} d(x, y)$$

Agglomerative clustering

The dissimilarity between clusters has an influence on the clustering result:

	1	2	3	4
1	0.0	0.20	0.15	0.30
2	0.20	0.0	0.40	0.50
3	0.15	0.40	0.0	0.10
4	0.30	0.50	0.10	0.0

(a) Dissimilarity Matrix



(b) Single Link (c) Complete Link

[Data Clustering: Algorithms and Applications, 2014]

Agglomerative clustering

Algorithm

Input : data set with N instances

$X=\{x_1,x_2,\ldots,x_N\}$ + dissimilarity matrix D

Output: dendrogram (set of triples)

```
agglomerative(X,D)
level=0;  k=N
C={{x1},{x2},...,{xN}};  DE={(level,k,C)}
repeat
  oldk=k
  level=level+1
  (k,C)=mergeClusters(k,C,D)
  D=recompute the dissimilarity  matrix using
      single/complete/average linkage
  DE=union (DE, (level,k,C))
until k=1
```

Remarks:

- The mergeClusters function identifies the closest clusters and merge them
- The algorithm has a quadratic complexity with respect to the number of data instances ($O(N^2)$)
- The agglomerative algorithms are sensitive to the noise in data

Divisive clustering

Generic top-down clustering algorithm

Input : data set with N instances $X=\{x_1,x_2,\ldots,x_N\}$

Output: dendrogram (tree) T

divisive(X,D)

Initialize the tree T with a root node containing the entire data set

Repeat

select a leaf node L from T (based on a specific criterion)

use a flat clustering algorithm to split L into $L_1,L_2,\ldots L_k$

Add $L_1,L_2,\ldots L_k$ as children of L in T

until <a stopping criterion>

Remark: the flat clustering algorithm may be kMeans; a particular case is the bisecting kMeans which is based on splitting each node in two other nodes (by applying kMeans for k=2)

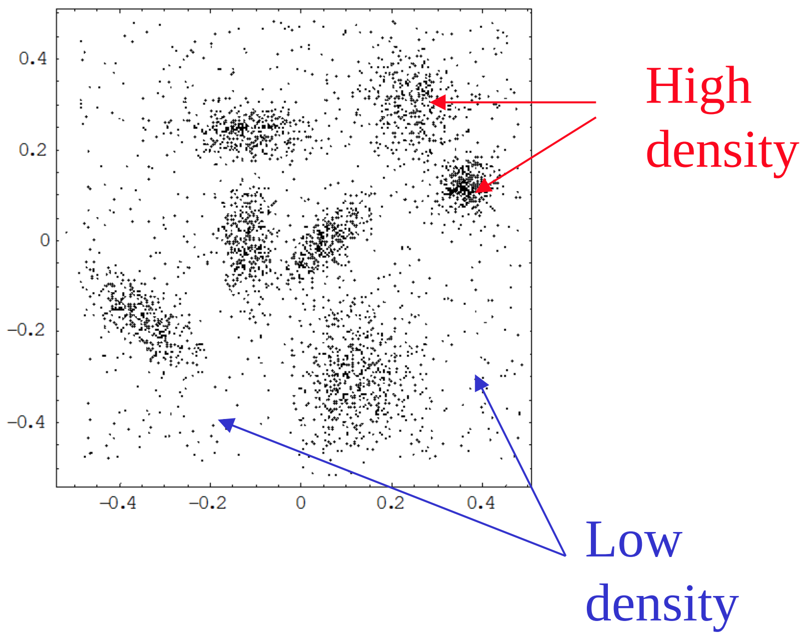
Bisecting Kmeans

- Bisecting K-means algorithm
  - Variant of K-means that can produce a partitional or a hierarchical clustering
  - Based on a binary splitting

```
Initialize the list of clusters to contain the cluster containing all points.
repeat
  Select a cluster from the list of clusters
  for i = 1 to number_of_iterations do
    Bisect the selected cluster using basic K-means
  end for
  Add the two clusters from the bisection with the lowest SSE to the list of clusters.
until Until the list of clusters contains K clusters
```

Density based clustering

- Clusters = dense groups of similar data separated by low density regions
- Basic idea: estimate the local density of data
  - either by determining the number of data in a given neighborhood of the analyzed point (DBSCAN)
  - or by using some influence functions (DENCLUE)
- Main aspects:
  - How is density estimated?
  - How is connectivity defined?
  - What data structures should be used?



DBSCAN

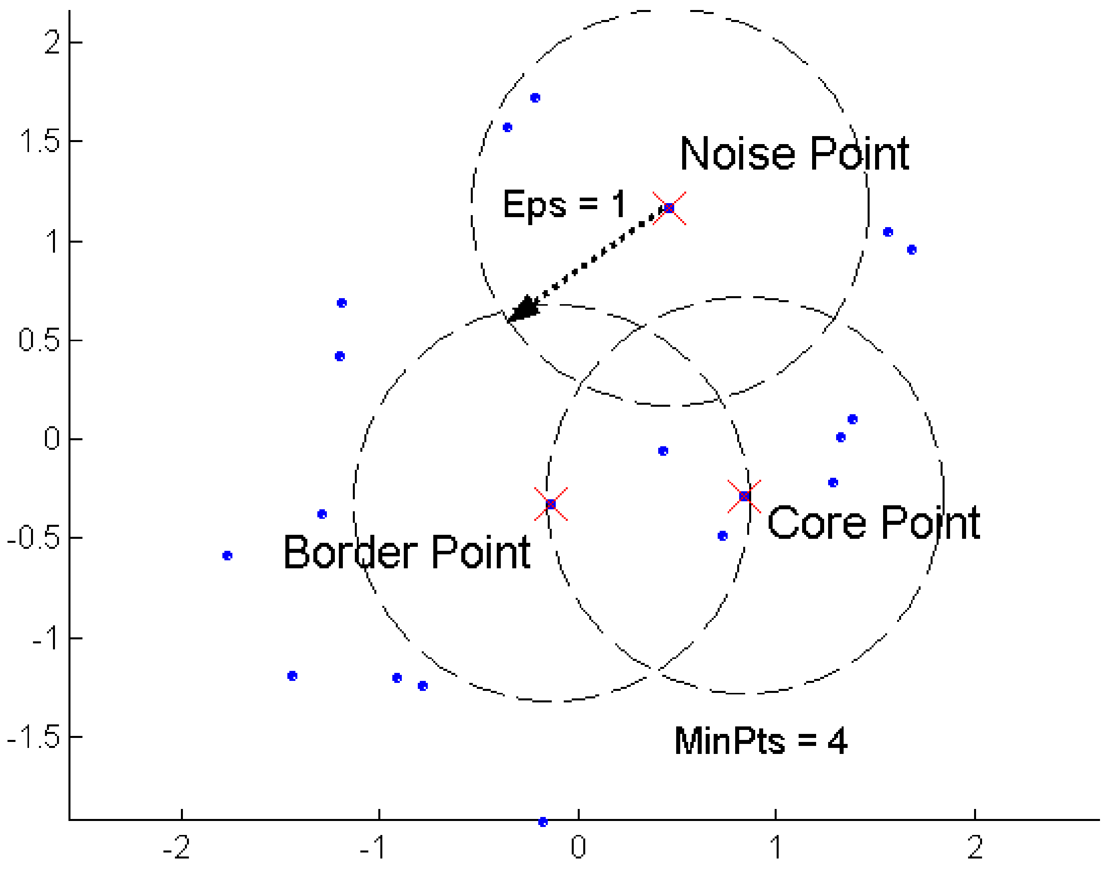
DBSCAN is a density-based algorithm

- Density measured at a point= number of points within a neighborhood of specified radius (Eps)

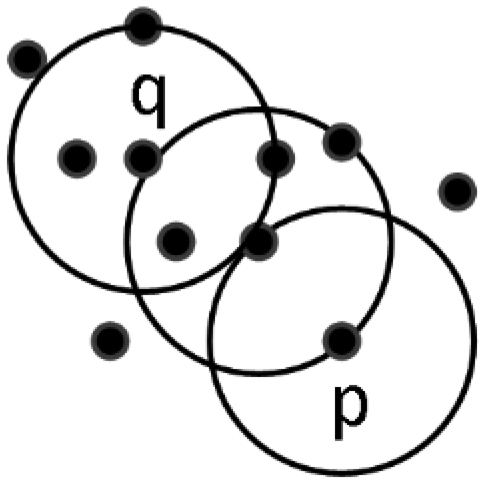
- A point is a core point if it has more than a specified number of points (MinPts) within Eps; these are points that are in the interior of a cluster

- A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point; Two points are connected if they are one in the neighborhood of the other
- A point q is directly density reachable from a core point p if it is in the neighborhood of p; density reachability is defined as transitive closure of direct density reachability (there is a chain of core points s.t. one point is directly reachable from the previous one)
- A noise point is any point that is not a core point or a border point.

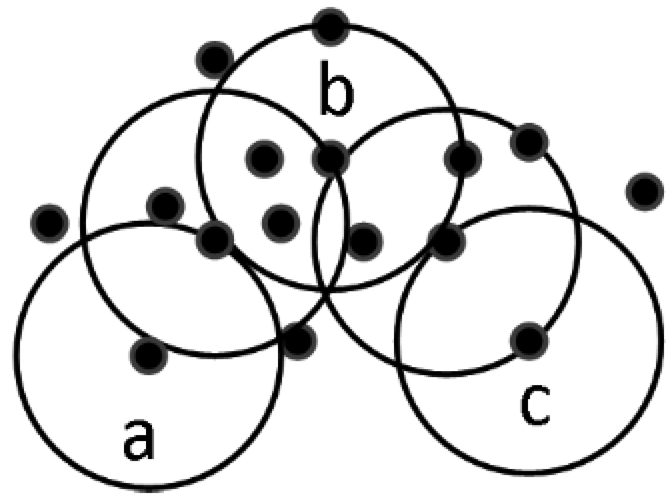
DBSCAN


Noise Point
Eps = 1
Border Point
Core Point
MinPts = 4
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1
0
1
2

DBSCAN



p is density reachable from q



a is density reachable from b
c is density reachable from b
=> a and c are density-connected

Remark:

- Two points, a and b, are density-connected if there exist a third point, c, such that c is reachable both from a and from b
- Two density-connected points belong to the same cluster =>
  a density based cluster is a maximal set of density-connected data

DBSCAN

$current\_cluster\_label \leftarrow 1$

for all core points **do**

  if the core point has no cluster label **then**

    $current\_cluster\_label \leftarrow current\_cluster\_label + 1$

    Label the current core point with cluster label $current\_cluster\_label$

  end if

  for all points in the Eps-neighborhood, except i^{th} the point itself **do**

    if the point does not have a cluster label **then**

      Label the point with cluster label $current\_cluster\_label$

    end if

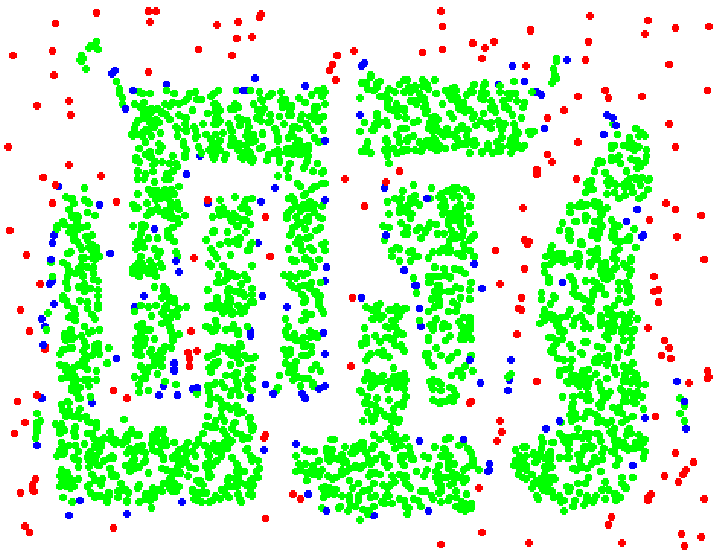
  end for

end for

[images from slides by Kumar, 2004]

DBSCAN

Original Points

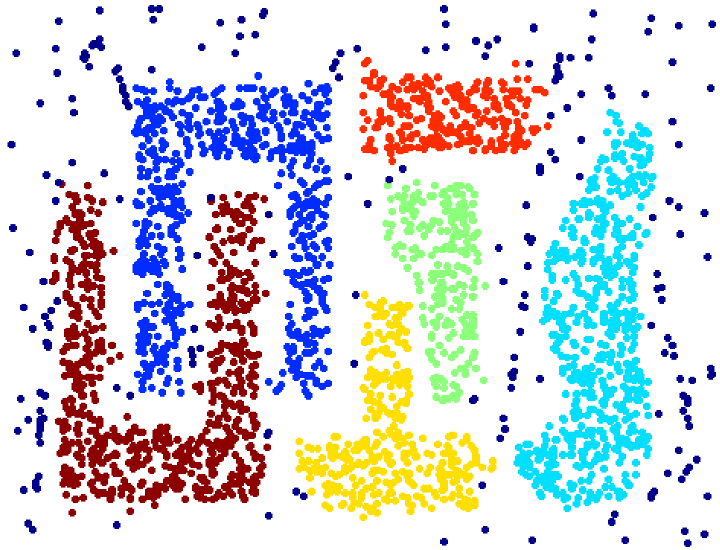


Point types: core, border and noise

Eps = 10, MinPts = 4

DBSCAN

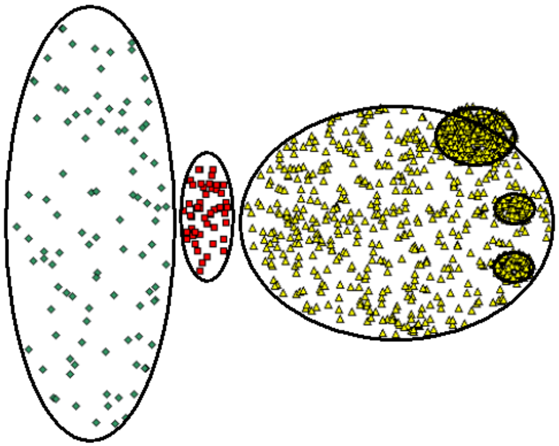
Original Points



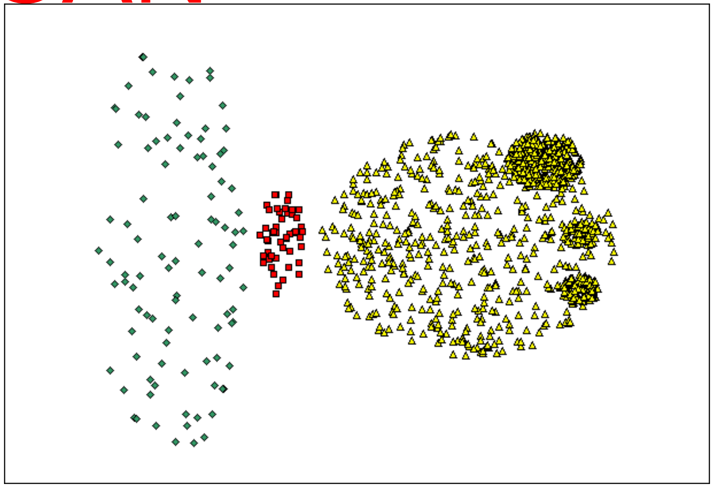
Clusters

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

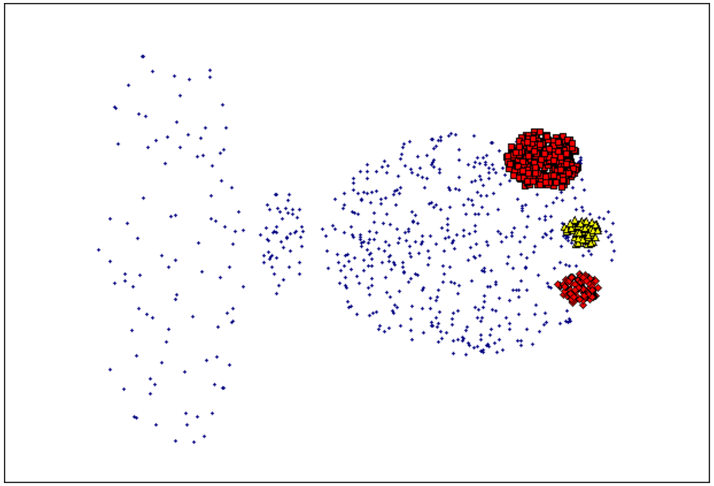
DBSCAN



Original Points



(MinPts=4, Eps=9.75).



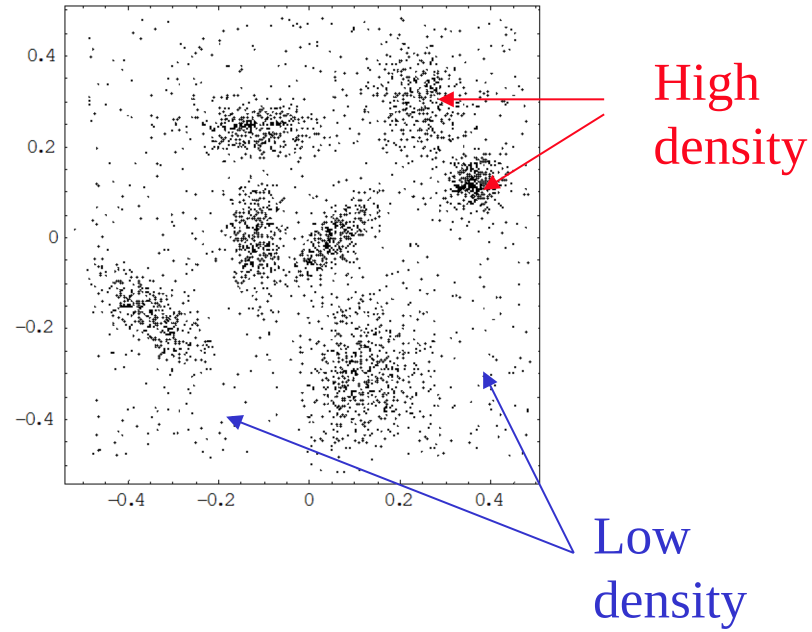
It does not work well:

•Varying densities

• High-dimensional data

DENCLUE

- Clusters = dense groups of similar data separated by low density regions
- Basic idea: estimate the local density of data
  - either by determining the number of data in a given neighborhood of the analyzed point (DBSCAN)
  - or by using some influence functions (DENCLUE)



Influence function

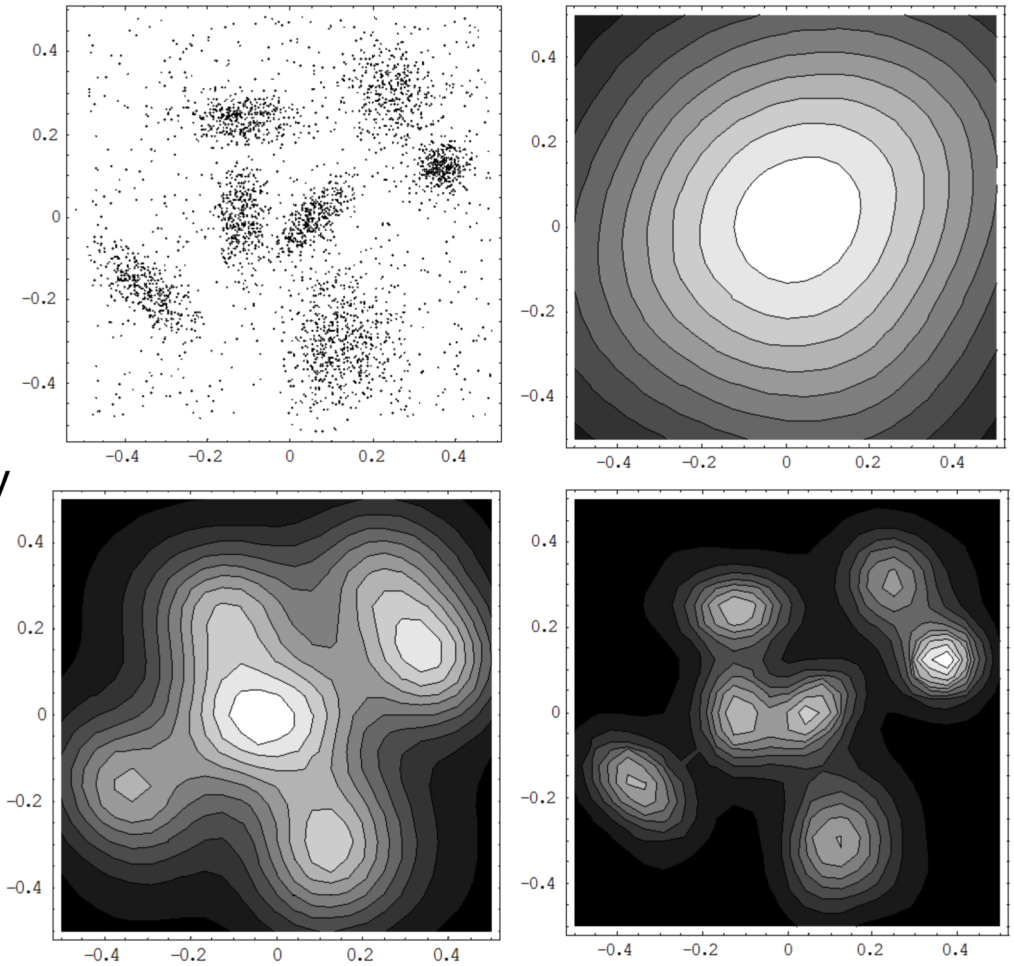
$$I_y(x) = \frac{1}{\sigma^{n/2}} \exp\left(- \frac{\sum_{j=1}^{n} (x_j - y_j)^2}{2\sigma^2} \right)$$

Density function

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} I_{x_i}(x)$$

DENCLUE

- The landscape of the density function is highly dependent on the value of parameter σ
- For appropriate values of σ, the local maxima of the density function correspond to clusters
- For large values of σ the density function landscape has a single maximum
- For small values of σ the local maxima are steep, thus difficult to find



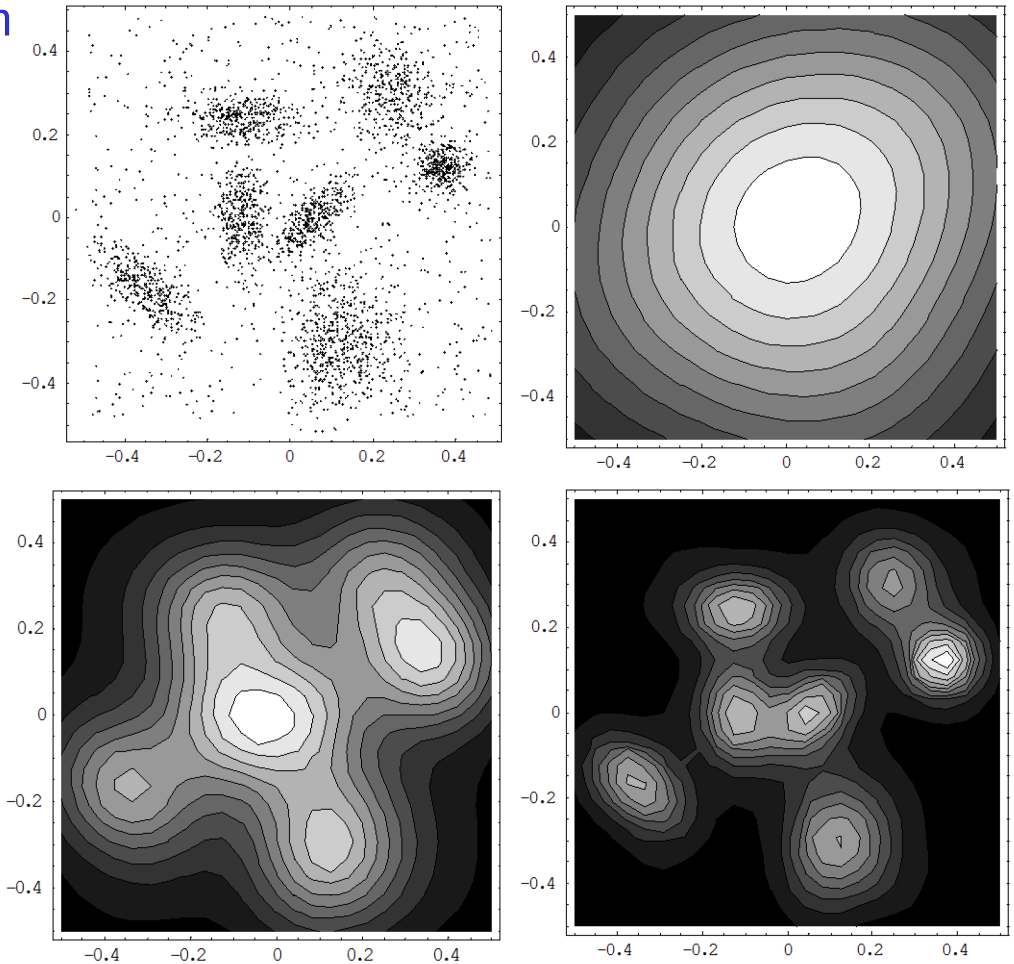
σ=0.1

σ=0.05

DENCLUE

Idea of DENCLUE [Hinneburg, Keim – 1998]: apply a gradient search to find local maxima starting from data points

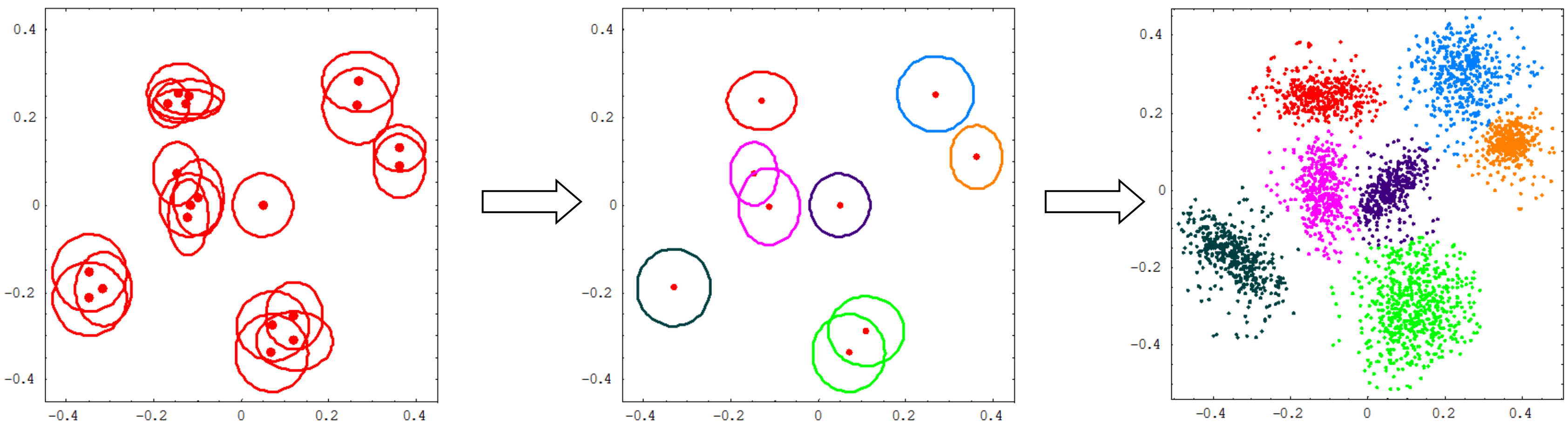
- identify center-defined clusters (one cluster correspond to one local maxima)
- identify arbitrary-shaped clusters (one cluster corresponds to a set of “connected” local maxima)



σ=0.1 σ=0.05

DENCLUE

Example - results



Marked points = maxima of the density function

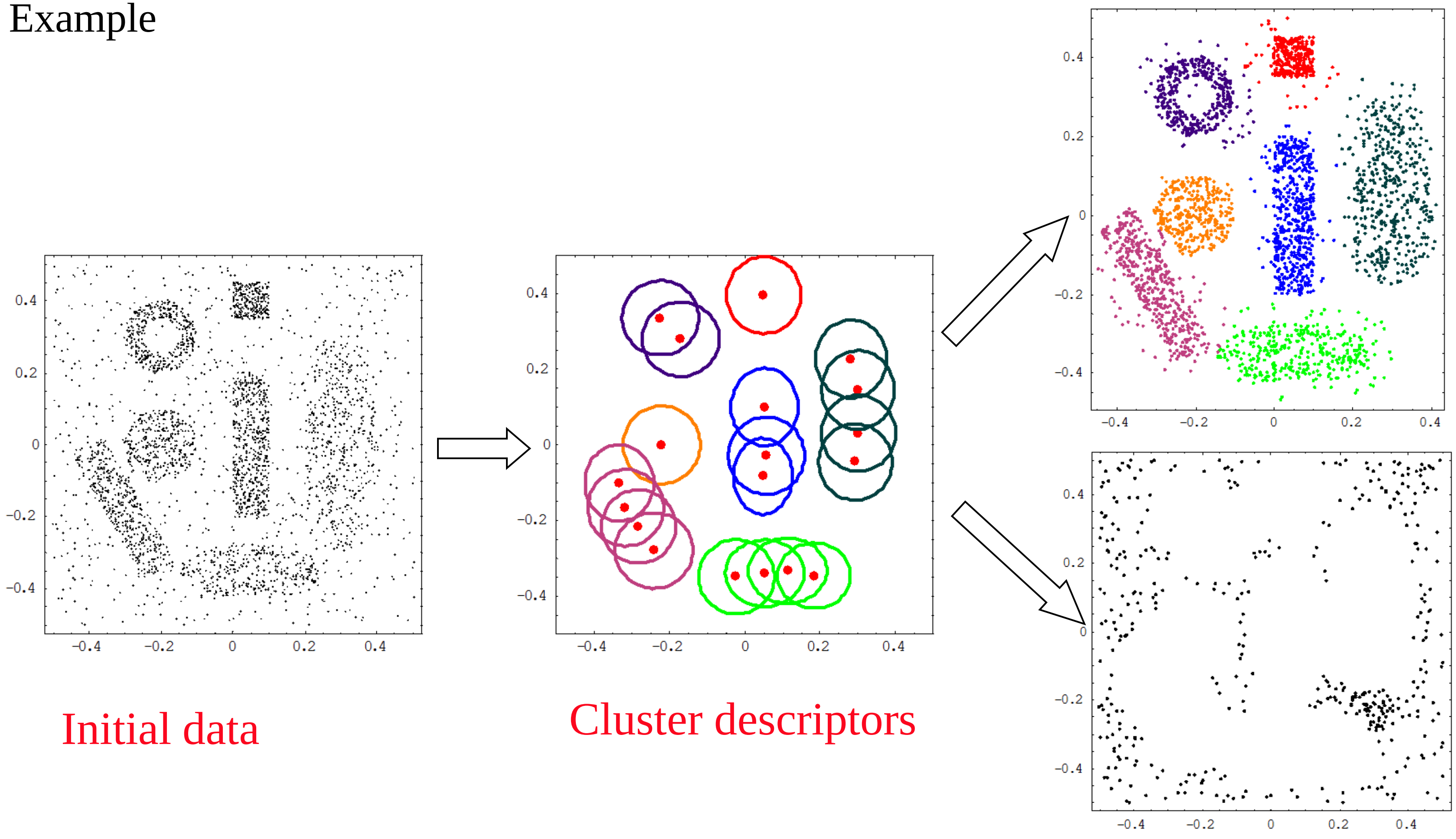
Marked regions = influence area (defined by the parameters of the influence functions)

Groups = the data are distributed in clusters based on the values of the influence functions

DENCLUE

Example

Identified clusters



Initial data

Cluster descriptors

Noise

Probabilistic methods

Main idea:

- The data are generated by a stochastic process (a mixture of probability distributions, each one being in correspondence with a cluster)
- The aim of the clustering algorithm is to discover the probabilistic model, i.e. identify the probability distributions

Example:

- Expectation–Maximization (EM) algorithm; it is based on the following assumptions:
  - each data has been generated by a probability distribution
  - in the generative process, the probability distribution corresponding to each data is selected according to a selection probability

EM algorithm

- Input: data set $D=\{x_1,x_2,\ldots,x_N\}$, K = number of clusters
- Output: a partition $P=\{C_1,C_2,\ldots,C_K\}$ of D

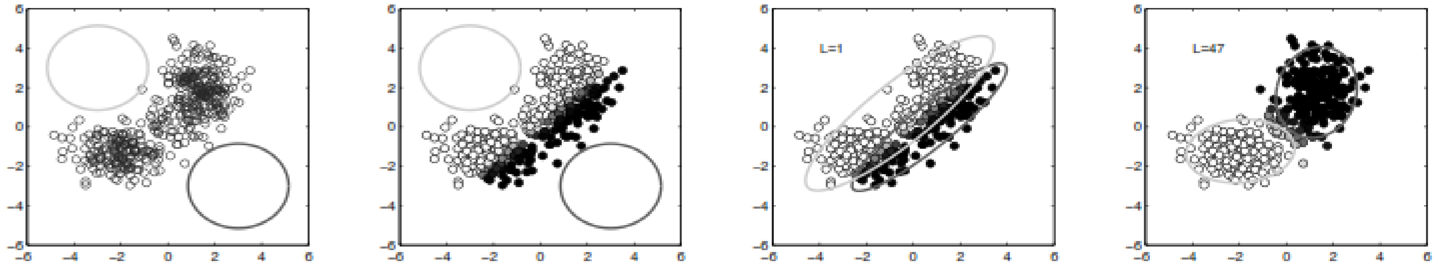
- (E-Step) Determine the expected probability of assignment of data points to clusters with the use of current model parameters.
- (M-Step) Determine the optimum model parameters of each mixture by using the assignment probabilities as weights.

EM algorithm

Algorithm 11 EM for Gaussian Mixtures

Given a set of data points and a Gaussian mixture model, the goal is to maximize the log-likelihood with respect to the parameters.

1: Initialize the means μ_k^0, covariances Σ_k^0, and mixing probabilities π_k^0.
2: **E-step**: Calculate the responsibilities $\gamma(z_{nk})$ using the current parameters based on Equation (3.13).
3: **M-step**: Update the parameters using the current responsibilities. Note that we first update the new means using (3.12), then use these new values to calculate the covariances using (3.14), and finally reestimate the mixing probabilities using (3.15).
4: Compute the log-likelihood using (3.10) and check for convergence of the algorithm. If the convergence criterion is not satisfied, then repeat steps 2–4; otherwise, return the final parameters.



(a) (b) (c) (d)

Data Clustering Algorithms and applications (ed. CC Aggarwal, CK Reddy, 2014)

EM algorithm

Equations involved in the EM algorithm (mixture of normal distributions)

$$\mathcal{N}(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\Sigma|^{1/2}}\exp\{-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\}$$

E-step: Computation of the responsabilities: how "strong" x_n belongs to C_k

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\mu_k,\Sigma_k)}{\sum_{j=1}^{K}\pi_j\mathcal{N}(\mathbf{x}_n|\mu_j,\Sigma_j)}. \quad (3.13)$$

M-step: Estimation of mean, covariance matrix and the mixing probabilities

$$\mu_k = \frac{\sum_{n=1}^{N}\gamma(z_{nk})\mathbf{x}_n}{\sum_{n=1}^{N}\gamma(z_{nk})} \quad (3.12)$$

$$\Sigma_k = \frac{\sum_{n=1}^{N}\gamma(z_{nk})(\mathbf{x}_n-\mu_k)(\mathbf{x}_n-\mu_k)^T}{\sum_{n=1}^{N}\gamma(z_{nk})}. \quad (3.14)$$

$$\pi_k = \frac{\sum_{n=1}^{N}\gamma(z_{nk})}{N} \quad (3.15)$$

Stopping condition: stabilization of the probability model

$$p(\mathbf{x}_n|\Theta) = p(\mathbf{x}_n|\pi,\mu,\Sigma) = \sum_{k=1}^{K}\pi_k\mathcal{N}(\mathbf{x}_n|\mu_k,\Sigma_k). \quad (3.10)$$

Data Clustering Algorithms and applications (ed. CC Aggarwal, CK Reddy, 2014)