DATABASE SYSTEMS GROUP

Contents

LMU

Expectation Maximization (EM)

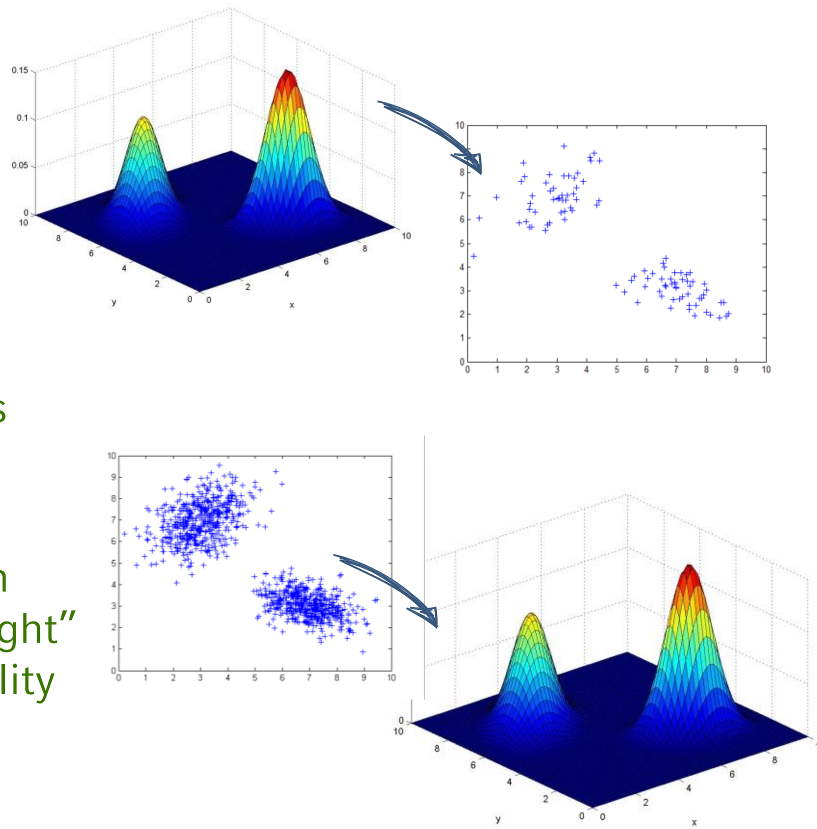
Statistical approach for finding maximum likelihood estimates of parameters in probabilistic models

Here: using EM as clustering algorithm

Approach:
Observations are drawn from one of several components of a mixture distribution.

Main idea:

- Define clusters as probability distributions → each object has a certain probability of belonging to each cluster
- Iteratively improve the parameters of each distribution (e.g. center, “width” and “height” of a Gaussian distribution) until some quality threshold is reached



Additional Literature: C. M. Bishop „Pattern Recognition and Machine Learning“, Springer, 2009

Excursus: Gaussian Mixture Distributions

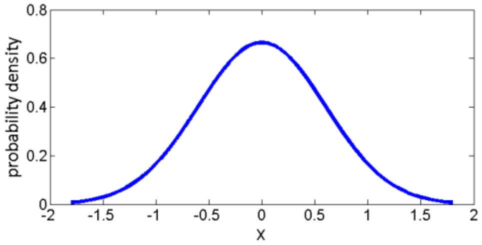
Note: EM is not restricted to Gaussian distributions, but they will serve as example in this lecture.

Gaussian distribution:

- Univariate: single variable $x \in \mathbb{R}$:

$$p(x|\mu, \sigma^2) = \mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{1}{2\sigma^2}\cdot(x-\mu)^2}$$

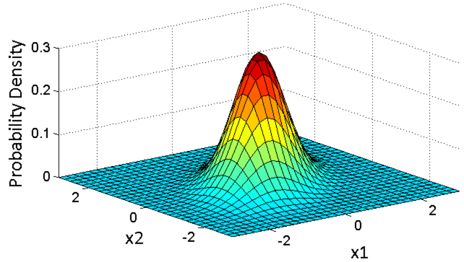
mean $\in \mathbb{R}$ *variance* $\in \mathbb{R}$



- Multivariate: d-dimensional vector $\boldsymbol{x} \in \mathbb{R}^d$:

$$p(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \cdot e^{-\frac{1}{2}\cdot(\boldsymbol{x}-\boldsymbol{\mu})^T\cdot(\boldsymbol{\Sigma})^{-1}\cdot(\boldsymbol{x}-\boldsymbol{\mu})}$$

mean vector $\in \mathbb{R}^d$ *covariance matrix* $\in \mathbb{R}^{d\times d}$

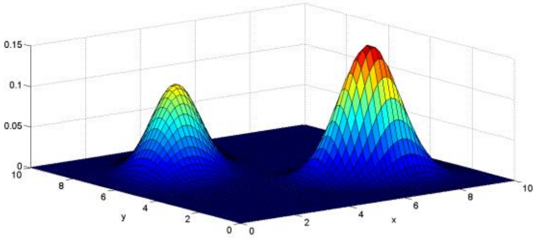


Gaussian mixture distribution with K components:

- d-dimensional vector $\boldsymbol{x} \in \mathbb{R}^d$:

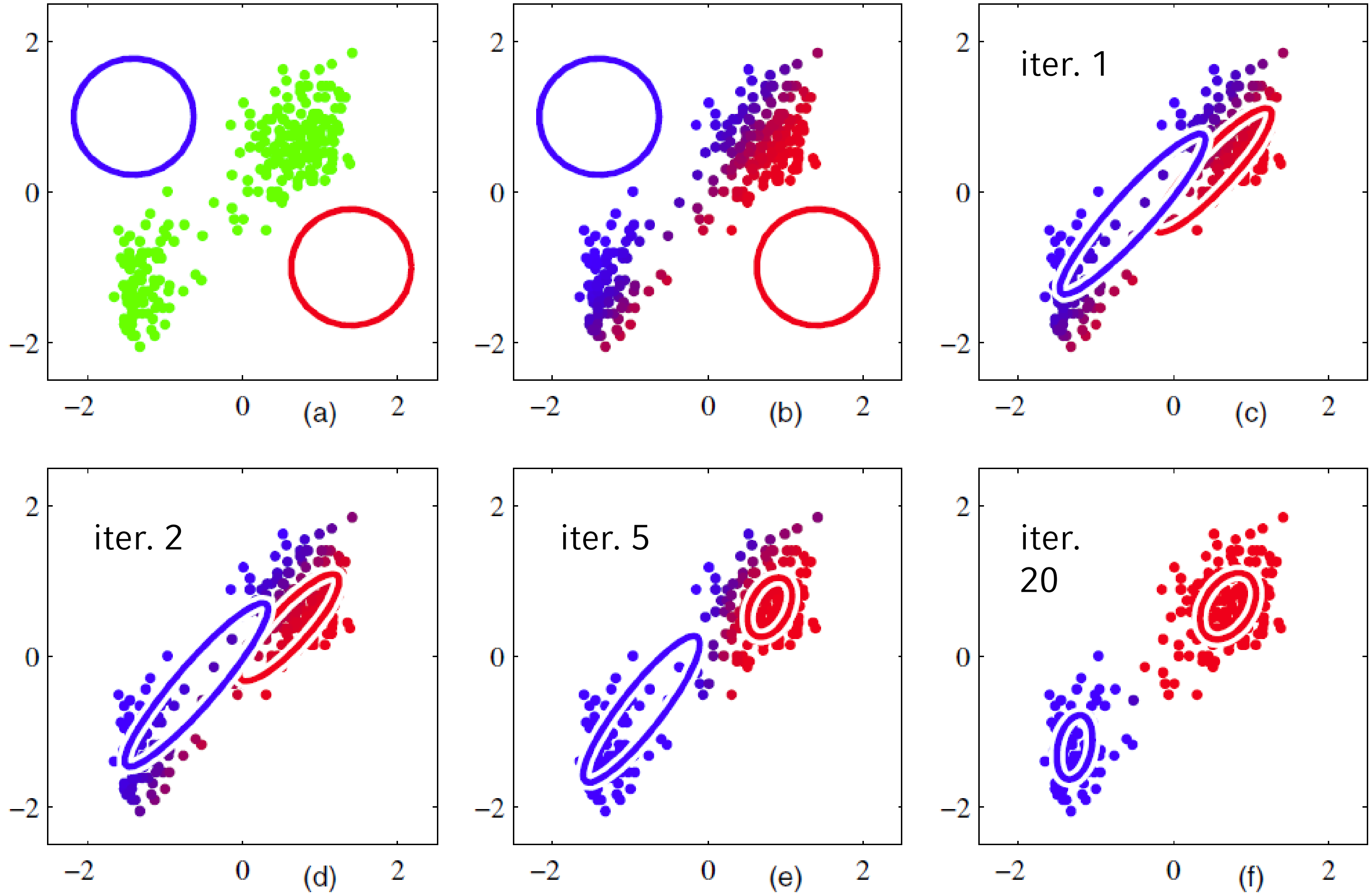
$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

mixing coefficients $\in \mathbb{R}$ *:* $\sum_k \pi_k = 1$ *and* $0 \leq \pi_k \leq 1$



Expectation Maximization (EM): Exemplary Application

Example taken from: C. M. Bischop „Pattern Recognition and Machine Learning“, 2009

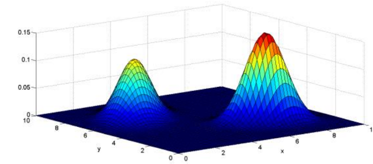


Expectation Maximization (EM)

Note: EM is not restricted to Gaussian distributions, but they will serve as example in this lecture.

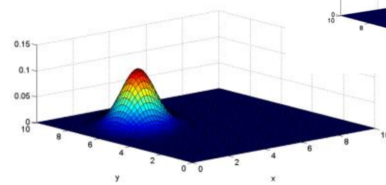
A *clustering* $\mathcal{M} = \{C_1, \ldots, C_K\}$ is represented by a mixture distribution with parameters $\Theta = \{\pi_1, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \ldots, \pi_K, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_K\}$:

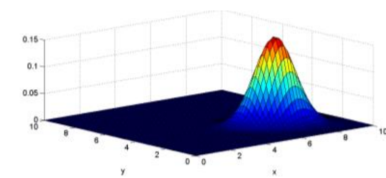
$$p(\boldsymbol{x}|\Theta) = \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$



Each *cluster* is represented by one component of the mixture distribution:

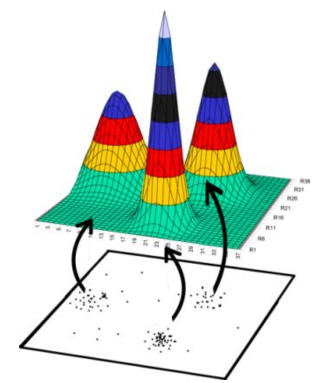
$$p(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$





Given a dataset $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \subseteq \mathbb{R}^d$, we can write the likelihood that all data points $\mathbf{x}_n \in \mathbf{X}$ are generated (independently) by the mixture model with parameters Θ as:

$$\log p(\mathbf{X}|\Theta) = \log \prod_{n=1}^{N} p(x_n|\Theta)$$



Goal: Find the parameters Θ_{ML} with **maximal (log-)likelihood estimation** (MLE)

$$\Theta_{ML} = \arg\max_{\Theta}\{\log p(\mathbf{X}|\Theta)\}$$

Expectation Maximization (EM)

- Goal: Find the parameters Θ_{ML} with the **maximal (log-)likelihood estimation**!

$$\Theta_{ML} = \arg\max_{\Theta}\{\log p(\mathbf{X}|\Theta)\}$$

$$\log p(\mathbf{X}|\Theta) = \log \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \cdot \mathrm{p}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \cdot \mathrm{p}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Maximization with respect to the means:

$$\frac{\partial}{\partial \boldsymbol{\mu}_j} \mathcal{N}(x_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \boldsymbol{\Sigma}_{\mathrm{j}}^{-1}(x_n - \boldsymbol{\mu}_j)\mathcal{N}(x_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$$

$$\frac{\partial \log p(\mathbf{X}|\Theta)}{\partial \boldsymbol{\mu}_j} = \sum_{n=1}^{N} \frac{\partial \log p(\boldsymbol{x}_n|\Theta)}{\partial \boldsymbol{\mu}_j} = \sum_{n=1}^{N} \frac{\frac{\partial p(\boldsymbol{x}_n|\Theta)}{\partial \boldsymbol{\mu}_j}}{p(\boldsymbol{x}_n|\Theta)} = \sum_{n=1}^{N} \frac{\frac{\partial\, \pi_j \cdot p(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\partial \boldsymbol{\mu}_j}}{\sum_{k=1}^{K} p(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} = \sum_{n=1}^{N} \frac{\pi_j \cdot \boldsymbol{\Sigma}_{\mathrm{j}}^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_j)\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{K} p(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

$$\frac{\partial \log p(\mathbf{X}|\Theta)}{\partial \boldsymbol{\mu}_j} = \boldsymbol{\Sigma}_{\mathrm{j}}^{-1} \sum_{n=1}^{N} (\boldsymbol{x}_n - \boldsymbol{\mu}_j) \frac{\pi_j \cdot \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \overset{\text{def}}{=} \mathbf{0}$$

- Define

$$\gamma_j(x_n) \coloneqq \pi_j \cdot \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j).$$

$\gamma_j(x_n)$ is the probability that component j generated the object $\boldsymbol{x}_n$.

Expectation Maximization (EM)

Maximization w.r.t. the means yields:

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma_j(\boldsymbol{x}_n)\, \boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma_j(\boldsymbol{x}_n)}$$

(weighted mean)

Maximization w.r.t. the covariance yields:

$$\boldsymbol{\Sigma}_j = \frac{\sum_{n=1}^{N} \gamma_j(\boldsymbol{x}_n)(\boldsymbol{x}_n - \boldsymbol{\mu}_j)(\boldsymbol{x}_n - \boldsymbol{\mu}_j)^T}{\sum_{n=1}^{N} \gamma_j(\boldsymbol{x}_n)}$$

Maximization w.r.t. the mixing coefficients yields:

$$\pi_j = \frac{\sum_{n=1}^{N} \gamma_j(\boldsymbol{x}_n)}{\sum_{k=1}^{K} \sum_{n=1}^{N} \gamma_k(\boldsymbol{x}_n)}$$

Problem with finding the optimal parameters Θ_{ML}:

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma_j(\boldsymbol{x}_n)\, \boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma_j(\boldsymbol{x}_n)} \quad \text{and} \quad \gamma_j(\boldsymbol{x}_n) = \frac{\pi_j \cdot \mathcal{N}\left(\boldsymbol{x}_n \middle| \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)}{\sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

- Non-linear mutual dependencies.
- Optimizing the Gaussian of cluster j depends on all other Gaussians.

→ There is no closed-form solution!

→ Approximation through iterative optimization procedures

→ Break the mutual dependencies by optimizing $\boldsymbol{\mu}_j$ and $\gamma_j(\boldsymbol{x}_n)$ independently

EM-approach: iterative optimization

1. Initialize means $\boldsymbol{\mu}_j$, covariances $\boldsymbol{\Sigma}_j$, and mixing coefficients π_j and evaluate the initial log likelihood.
2. **E step:** Evaluate the responsibilities using the current parameter values:

$$\gamma_j^{new}(\boldsymbol{x}_n) = \frac{\pi_j \cdot \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

3. **M step:** Re-estimate the parameters using the current responsibilities:

$$\boldsymbol{\mu}_j^{new} = \frac{\sum_{n=1}^{N} \gamma_j^{new}(\boldsymbol{x}_n)\, \boldsymbol{x}_n}{\sum_{n=1}^{N} \gamma_j^{new}(\boldsymbol{x}_n)}$$

$$\boldsymbol{\Sigma}_j^{new} = \frac{\sum_{n=1}^{N} \gamma_j^{new}(\boldsymbol{x}_n)(\boldsymbol{x}_n - \boldsymbol{\mu}_j^{new})(\boldsymbol{x}_n - \boldsymbol{\mu}_j^{new})^T}{\sum_{n=1}^{N} \gamma_j^{new}(\boldsymbol{x}_n)}$$

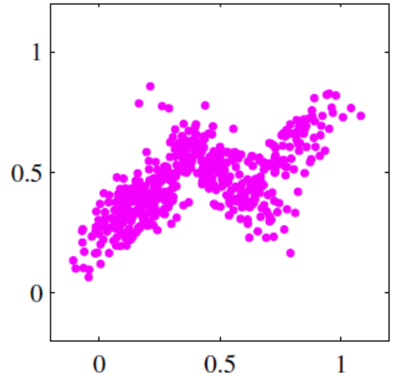
$$\pi_j^{new} = \frac{\sum_{n=1}^{N} \gamma_j^{new}(\boldsymbol{x}_n)}{\sum_{k=1}^{K} \sum_{n=1}^{N} \gamma_k^{new}(\boldsymbol{x}_n)}$$

4. Evaluate the new log likelihood $\log p(\mathbf{X}|\Theta^{\mathrm{new}})$ and check for convergence of parameters or log likelihood ($|\log p(\mathbf{X}|\Theta^{\mathrm{new}})$ - $\log p(\mathbf{X}|\Theta)| \leq \epsilon$).
   If the convergence criterion is not satisfied, set $\Theta = \Theta^{\mathrm{new}}$ and go to step 2.

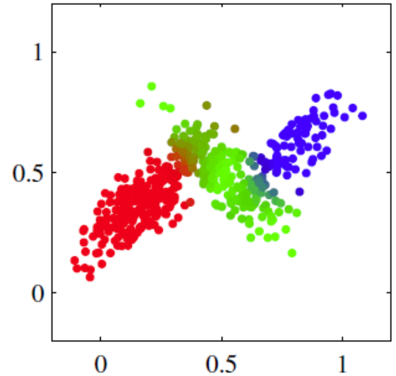
EM: Turning the Soft Clustering into a Partitioning

EM obtains a *soft* clustering (each object belongs to each cluster with a certain probability) reflecting the uncertainty of the most appropriate assignment.

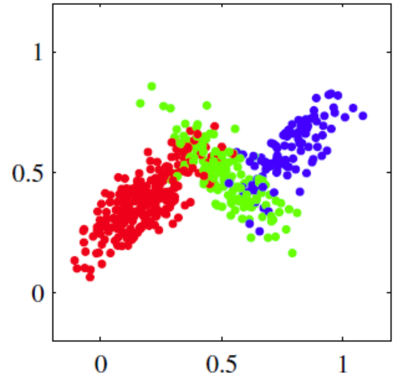
Example taken from: C. M. Bishop „Pattern Recognition and Machine Learning“, 2009



a) input for EM



b) soft clustering result of EM



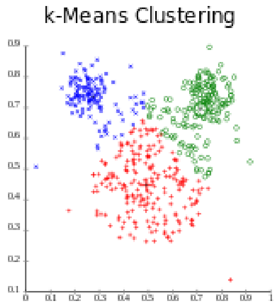
c) original data

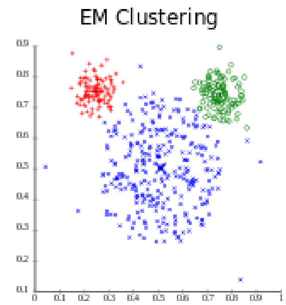
Modification to obtain a *partitioning* variant

– Assign each object to the cluster to which it belongs with the highest probability

$$\text{Cluster}(\text{object}_n) = argmax_{k\in\{1,\dots,K\}}\{\gamma(z_{nk})\}$$

Discussion





- Superior to k-Means for clusters of varying size
  or clusters having differing variances
  → more accurate data representation
- Convergence to (possibly local) maximum
- Computational effort for N objects, K derived clusters, and t iterations:
  - $O(t \cdot N \cdot K)$
  - #iterations is quite high in many cases
- Both - result and runtime - strongly depend on
  - the initial assignment
    - → do multiple random starts and choose the final estimate with highest likelihood
    - → Initialize with clustering algorithms (e.g., K-Means usually converges much faster)
    - → Local maxima and initialization issues have been addressed in various extensions of EM
  - a proper choice of parameter K (= desired number of clusters)
    - → Apply principals of model selection (see next slide)

EM: Model Selection for Determining Parameter K

Classical trade-off problem for selecting the proper number of components K

- If K is too high, the mixture may overfit the data
- If K is too low, the mixture may not be flexible enough to approximate the data

Idea: determine candidate models Θ_K for a range of values of K (from K_{min} to K_{max}) and select the model $\Theta_{K^*} = \max\{\mathrm{qual}(\Theta_K) | K \in \{K_{min}, \dots, K_{max}\}\}$

- Silhouette Coefficient (as for k-Means) only works for partitioning approaches.
- The MLE (Maximum Likelihood Estimation) criterion is nondecreasing in K

Solution: deterministic or stochastic *model selection* methods[MP'00] which try to balance the goodness of fit with simplicity.

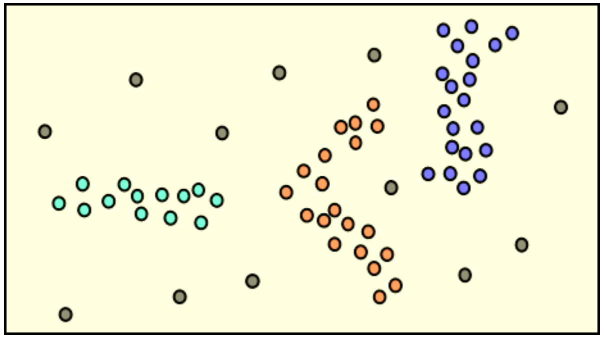
- Deterministic: $qual(\Theta_K) = \log p(\mathbf{X}|\Theta_K) + \mathcal{P}(K)$
  where $\mathcal{P}(K)$ is an increasing function penalizing higher values of K
- Stochastic: based on Markov Chain Monte Carlo (MCMC)

[MP'00] G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley, New York, 2000.

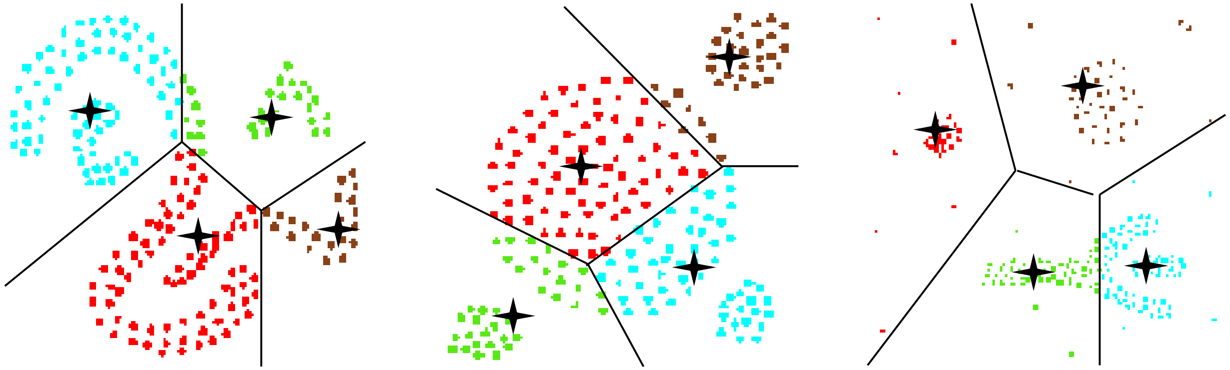
Contents

Density-Based Clustering

- Basic Idea:
  - Clusters are dense regions in the data space, separated by regions of lower object density



- Why Density-Based Clustering?



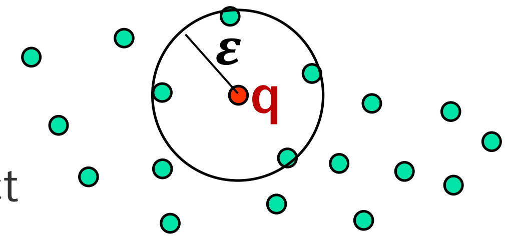
Results of a *k*-medoid algorithm for *k*=4

- Different density-based approaches exist (see Textbook & Papers)
  Here we discuss the ideas underlying the DBSCAN algorithm

Density-Based Clustering: Basic Concept

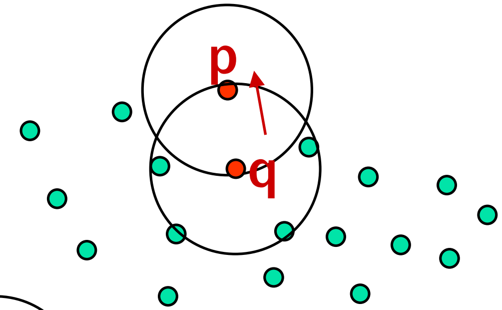
- Intuition for the formalization of the basic idea
  - For any point in a cluster, the local point density around that point has to exceed some threshold
  - The set of points from one cluster is spatially connected
- Local point density at a point *q* defined by two parameters
  - ε–radius for the neighborhood of point *q*:
    $N_\varepsilon(q) := \{p \in D \mid dist(p,q) \leq \varepsilon\}$ *! contains q itself !*
  - ***MinPts*** – minimum number of points in the given neighbourhood $N_\varepsilon(q)$
- *q* is called a ***core object*** (or core point) w.r.t. ε, *MinPts* if $|N_\varepsilon(q)| \geq MinPts$

MinPts = 5 → **q** is a core object

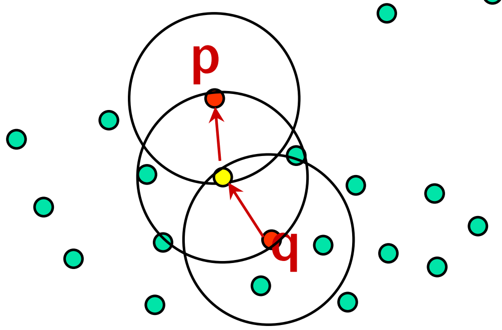


Density-Based Clustering: Basic Definitions

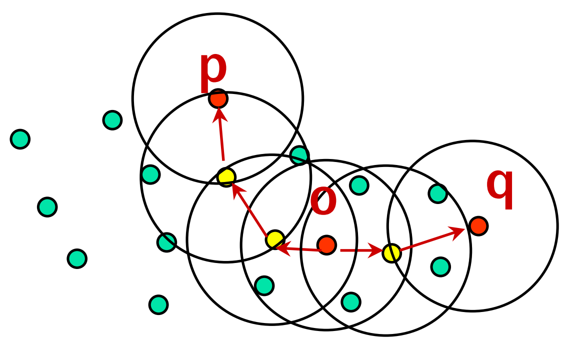
- *p* ***directly density-reachable*** from *q* w.r.t. ε, *MinPts* if
  1) $p \in N_{\varepsilon}(q)$ and
  2) *q* is a core object w.r.t. ε, *MinPts*



- ***density-reachable***: transitive closure of *directly* density-reachable



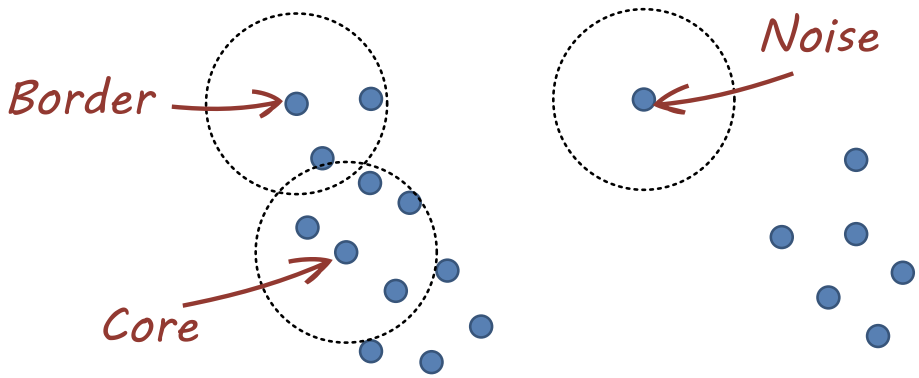
- *p* is ***density-connected*** to a point *q* w.r.t. ε, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. ε, *MinPts*.



Density-Based Clustering: Basic Definitions

- ***Density-Based Cluster***: non-empty subset *S* of database *D* satisfying:
  1) *Maximality*: if *p* is in *S* and *q* is density-reachable from *p* then *q* is in *S*
  2) *Connectivity*: each object in *S* is density-connected to all other objects in *S*

- ***Density-Based Clustering*** of a database *D* : $\{S_1, \ldots, S_n; N\}$ where
  – $S_1, \ldots, S_n$: all density-based clusters in the database *D*
  – $N = D \setminus \{S_1 \cup \ldots \cup S_n\}$ is called the ***noise*** (objects not in any cluster)



$$\varepsilon = 1.0$$
$$MinPts = 5$$

Density-Based Clustering: DBSCAN Algorithm

- Density Based Spatial Clustering of Applications with Noise
- Basic Theorem:
  - Each object in a density-based cluster C is density-reachable from any of its core-objects
  - Nothing else is density-reachable from core objects.

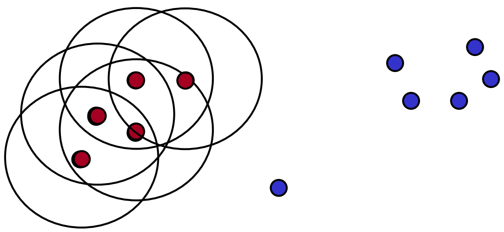
```
for each o ∈ D do
    if o is not yet classified then
        if o is a core-object then
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE
```

  - density-reachable objects are collected by performing successive ε-neighborhood queries.

Ester M., Kriegel H.-P., Sander J., Xu X.: „A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise“, *In KDD 1996*, pp. 226—231.

DBSCAN Algorithm: Example

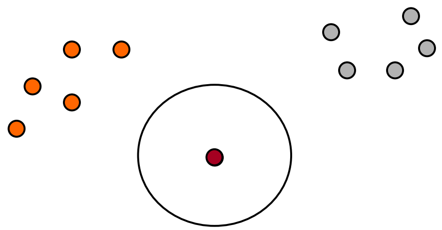
- Parameter
  - $\varepsilon = 2.0$
  - $MinPts = 3$



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

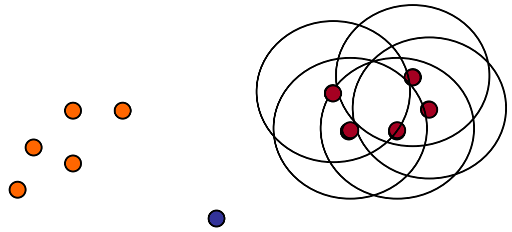
- Parameter
  - $\varepsilon = 2.0$
  - $MinPts = 3$



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
  - $\varepsilon = 2.0$
  - $MinPts = 3$



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