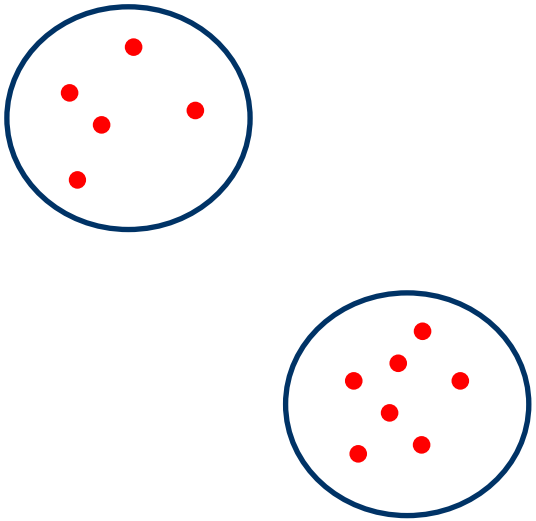
So Far…

- We have discussed
  - Supervised learning
    - Goal: learn a mapping from inputs $\boldsymbol{x}$ to outputs y
    - Training data: a labeled set of input-output pairs
  - Various methods to learn this mapping functions

- It's time for
  - Unsupervised learning
    - We are only given inputs
    - Goal: find "interesting patterns"

Discovering clusters: Clustering



What is Clustering and Why?

- Cluster: A collection of data objects
  - similar (or related) to one another within the same group
  - dissimilar (or unrelated) to the objects in other groups
- Clustering
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes

- Typical applications
  - As a stand-alone tool.
    - Image segmentation, News clustering
  - As a preprocessing step for other algorithms

The K-Means Algorithm (MacQueen'67)

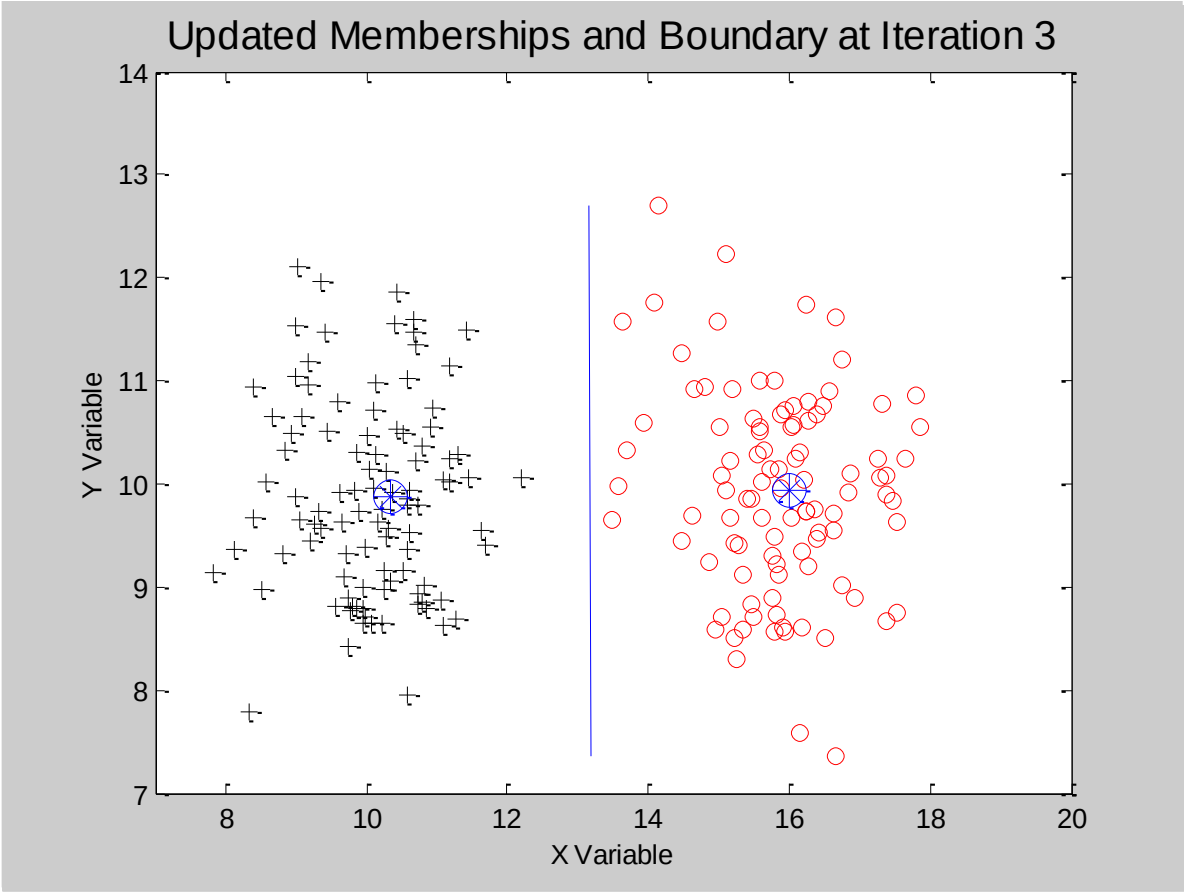
- Given K, the k-means algorithm is performed:
  1. Randomly pick K data points as the seed points $\boldsymbol{\mu}_k$
  2. Recursively run the following steps until converge
     1. Assign each point to the cluster with the nearest seed point

$$z_i = \arg\min_k \|\boldsymbol{x}_i - \boldsymbol{\mu}_k\|^2$$

     2. Update the seed point for each cluster

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \boldsymbol{x}_i^{(k)}$$

K-Means Example



The K-Means Algorithm (MacQueen'67)

- Given K, the k-means algorithm is performed:
  1. Randomly pick K data points as the seed points $\boldsymbol{\mu}_k$
  2. Recursively run the following steps until converge

t

  1. Assign each point to the cluster with the nearest seed point

NKd

$$z_i = \arg\min_k \|\boldsymbol{x}_i - \boldsymbol{\mu}_k\|^2$$

  2. Update the seed point for each cluster

$$\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{i=1}^{N_k} \boldsymbol{x}_i^{(k)}$$

$$\mathcal{L}_{TSD}(\boldsymbol{z}, \boldsymbol{\mu}) = \sum_{i=1}^{N} \left\|\boldsymbol{x}_i - \boldsymbol{\mu}_{z_i}\right\|^2$$

- The initial $\boldsymbol{\mu}_k$ don't need to be a real data point
- It is possible that empty clusters appear
- Converge?
- Complexity $O(tNKd)$

The K-Means Algorithm (MacQueen'67)

- Given K, the k-means algorithm is performed:
  1. Randomly pick K data points as the seed points $\boldsymbol{\mu}_k$
  2. Recursively run the following steps until converge
     1. Assign each object to the cluster with the nearest seed point

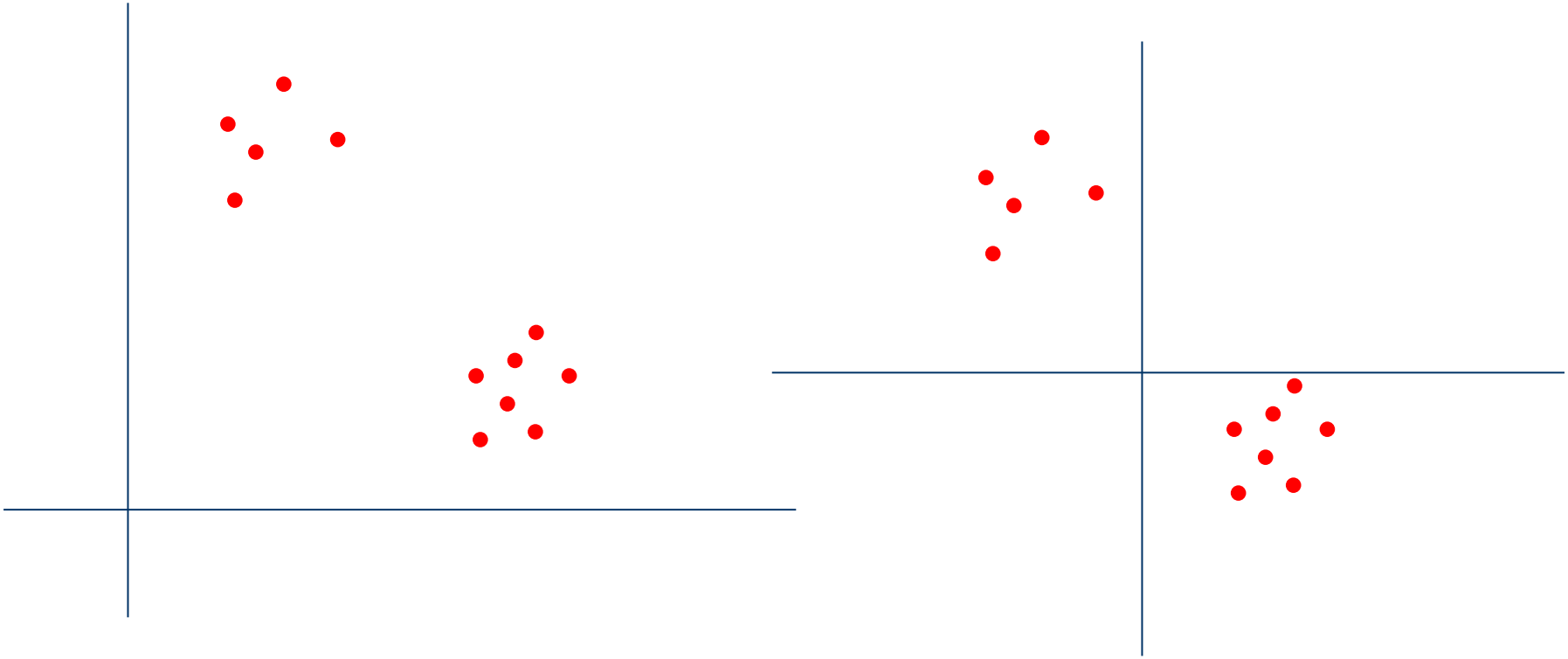
$$z_i = \arg\min_k \|\boldsymbol{x}_i - \boldsymbol{\mu}_k\|^2$$

     2. Update the seed point for each cluster

$$\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{i=1}^{N_k} \boldsymbol{x}_i^{(k)} \qquad \mathcal{L}_{TSD}(\boldsymbol{z}, \boldsymbol{\mu}) = \sum_{i=1}^{N} \|\boldsymbol{x}_i - \boldsymbol{\mu}_{z_i}\|^2$$

- Greedy algorithm leads to local optimum
  - In practice, we can run it from multiple starting points and pick the solution with the lowest $\mathcal{L}_{TSD}$
- Implicit assumptions about the "shapes" of clusters
  - Spherical

Clustering with a input of distance matrix



- Input: a pair-wise distance matrix
  - Kmeans cannot be applied, why?
  - The seed nodes of each cluster are virtual points

$$\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{i=1}^{N_k} \boldsymbol{x}_i^{(k)}$$

Partitioning Around Medoids (PAM) Kaufman & Rousseeuw'87

- The seeds in Kmeans:
  - Centroid: the "middle" of a cluster

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \boldsymbol{x}_i^{(k)}$$

  - Not necessarily a real point in a cluster

- The seeds in Kmedoids
  - Medoid: the centrally located point in the cluster.

K-Medoids with D input

- Input: the pairwise distances matrix D
  - Compute the row (or, column) sum of D, d
  - Finds the smallest entry in d

$O(N^2)$

- Given K and D, the k-medoids algorithm is performed:
  1. Randomly pick K data points as the seed points
  2. Recursively run the following steps until converge
     1. Assign each point to the cluster with the nearest seed point
     2. Update the seed point for each cluster using the above algorithm

How to find the medoid

- Inputs: N data points
  - Compute the centroid $\boldsymbol{\mu}$
  - Finds the point which is closest to $\boldsymbol{\mu}$

$$\arg\min_{x_k} \sum_j ||\boldsymbol{x}_j - \boldsymbol{x}_k||^2 \qquad \boldsymbol{\mu} = \frac{1}{N}\sum_j \boldsymbol{x}_j$$

$$\sum_j ||\boldsymbol{x}_j - \boldsymbol{x}_k||^2 = \sum_j ||\boldsymbol{x}_j - \boldsymbol{\mu} + \boldsymbol{\mu} - \boldsymbol{x}_k||^2$$

$$= \sum_j ||\boldsymbol{x}_j - \boldsymbol{\mu}||^2 + \sum_j 2(\boldsymbol{x}_j - \boldsymbol{\mu})^T(\boldsymbol{\mu} - \boldsymbol{x}_k) + \sum_j ||\boldsymbol{\mu} - \boldsymbol{x}_k||^2$$

$$= \sum_j ||\boldsymbol{x}_j - \boldsymbol{\mu}||^2 + N||\boldsymbol{\mu} - \boldsymbol{x}_k||^2$$

  - Choose $\boldsymbol{x}_k$ be the point closest to $\boldsymbol{\mu}$

K-Medoids with vectors input

- Inputs: N data points
  - Compute the centroid $\boldsymbol{\mu}$
  - Finds the point which is closest to $\boldsymbol{\mu}$

- Given K, the k-medoids algorithm is performed:

1. Randomly pick K data points as the seed points

t 2. Recursively run the following steps until converge

   1. Assign each point to the cluster with the nearest seed point

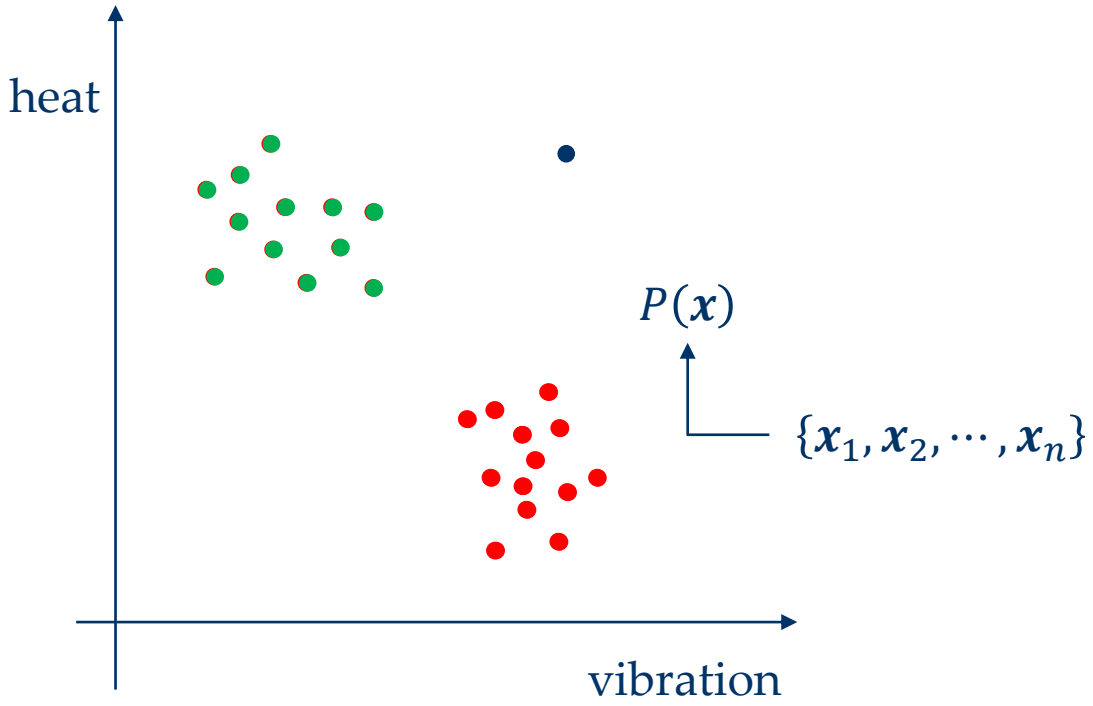
NKd

   2. Update the seed point for each cluster using the above algorithm

Nd

$$O(tN(K+1)d)$$

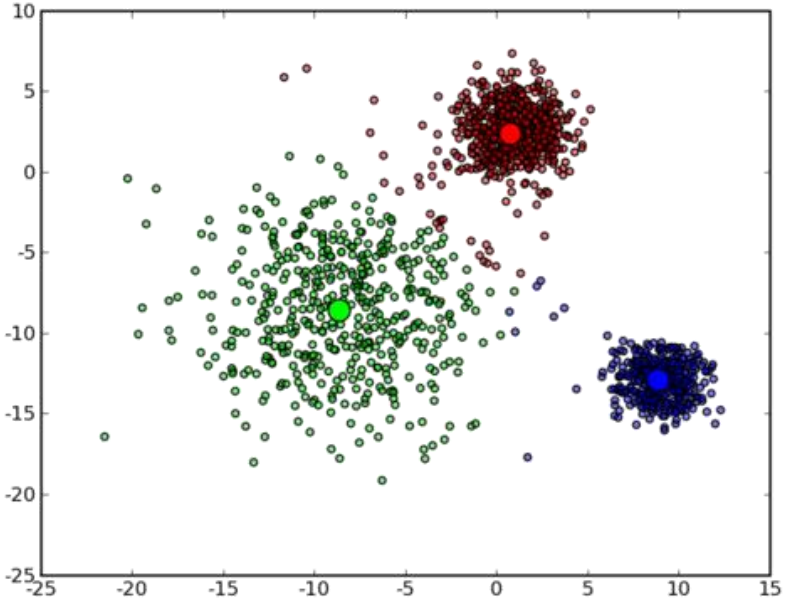
Density Estimation

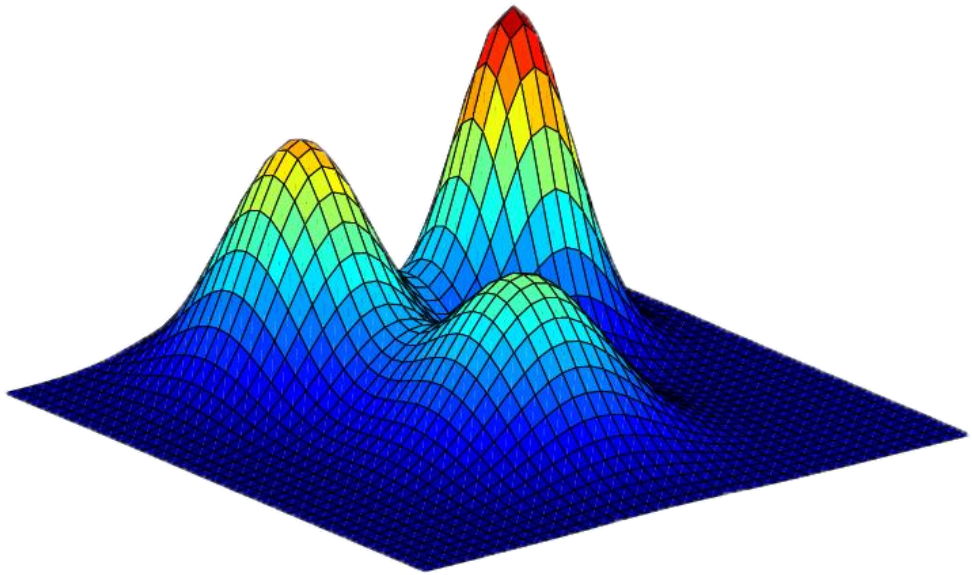


- Anomaly detection

Gaussian Mixture Model

- Gaussian Mixture Model (GMM) is one of the most popular clustering methods which can be viewed as a linear combination of different Gaussian components.





Gaussian Mixture Model

- Linear combination of Gaussians
  - Assumption: K Gaussians, each has a contribution of π_k to the data points

$$\begin{cases} p(\boldsymbol{x};\boldsymbol{\Theta}) = \sum_{k=1}^{K} \pi_k p_k(\boldsymbol{x};\boldsymbol{\theta}_k) \\ \boldsymbol{\Theta} = \{\pi_1, \cdots, \pi_K, \boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_K\}, \sum_{k=1}^{K} \pi_k = 1, \pi_k \in [0,1] \\ p_k(\boldsymbol{x};\boldsymbol{\theta}_k) = \mathcal{N}(\boldsymbol{x};\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) \end{cases}$$

  - Parameters to be estimated: $\pi_k, \boldsymbol{\mu}_k, \Sigma_k$

Parameters Estimation for GMM

- The log-likelihood function:

$$\log \prod_{i=1}^{N} p\left(\boldsymbol{x}^{(i)}; \boldsymbol{\Theta}\right) = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}\left(\boldsymbol{x}^{(i)}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) \right)$$

- How to find the parameters?

Gaussian Mixture Model

- There is a latent (hidden/unobserved) random variable z

$$P(\boldsymbol{x}_i, z_i) = P(\boldsymbol{x}_i|z_i)P(z_i)$$

- z_i follows the multinomial distribution $P(z_i = k) = \pi_k$
- $P(\boldsymbol{x}_i|z_i = k)$, Gaussian $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$
- If we know the variables z_i

$$\mathcal{L}(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{n} \log P(\boldsymbol{x}_i, z_i; \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

$$\widehat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x_i}$$

$$\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{x_i} - \widehat{\boldsymbol{\mu}})(\boldsymbol{x_i} - \widehat{\boldsymbol{\mu}})^{\boldsymbol{T}}$$

$$\pi_k = \frac{\sum_{i=1}^{n} I\{z_i = k\}}{n}$$

$$\widehat{\boldsymbol{\mu}_k} = \frac{\sum_{i=1}^{n} I\{z_i = k\}\boldsymbol{x_i}}{\sum_{i=1}^{n} I\{z_i = k\}}$$

$$\widehat{\boldsymbol{\Sigma}_k} = \frac{\sum_{i=1}^{n} I\{z_i = k\}(\boldsymbol{x_i} - \widehat{\boldsymbol{\mu}})(\boldsymbol{x_i} - \widehat{\boldsymbol{\mu}})^{\boldsymbol{T}}}{\sum_{i=1}^{n} I\{z_i = k\}}$$

Parameters Estimation for GMM

$$Q_k^{(i)} \triangleq P\left(z^{(i)} = k | \boldsymbol{x}^{(i)}\right) = \frac{p\left(\boldsymbol{x}^{(i)} | z^{(i)} = k\right) P\left(z^{(i)} = k\right)}{P\left(\boldsymbol{x}^{(i)}\right)}$$

E-Step

$$= \frac{p\left(\boldsymbol{x}^{(i)} | z^{(i)} = k\right) P\left(z^{(i)} = k\right)}{\sum_{k=1}^{K} p\left(\boldsymbol{x}^{(i)} | z^{(i)} = k\right) P\left(z^{(i)} = k\right)}$$

$\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ π_k

$$P\left(z^{(i)} = k | \boldsymbol{x}^{(i)}\right) = Q_k^{(i)}$$

M-Step

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{n} Q_k^{(i)} \boldsymbol{x}^{(i)}}{\sum_{i=1}^{n} Q_k^{(i)}}$$

$$\widehat{\boldsymbol{\mu}_k} = \frac{\sum_{i=1}^{n} I\{z_i = k\} \boldsymbol{x_i}}{\sum_{i=1}^{n} I\{z_i = k\}}$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{n} Q_k^{(i)} \left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k\right)\left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k\right)^{\boldsymbol{T}}}{\sum_{i=1}^{n} Q_k^{(i)}}$$

$$\widehat{\boldsymbol{\Sigma}_k} = \frac{\sum_{i=1}^{n} I\{z_i = k\} (\boldsymbol{x_i} - \widehat{\boldsymbol{\mu}})(\boldsymbol{x_i} - \widehat{\boldsymbol{\mu}})^{\boldsymbol{T}}}{\sum_{i=1}^{n} I\{z_i = k\}}$$

$$\pi_k = \frac{\sum_{i=1}^{n} Q_k^{(i)}}{n}$$

$$\pi_k = \frac{\sum_{i=1}^{n} I\{z_i = k\}}{n}$$

Expectation Maximization

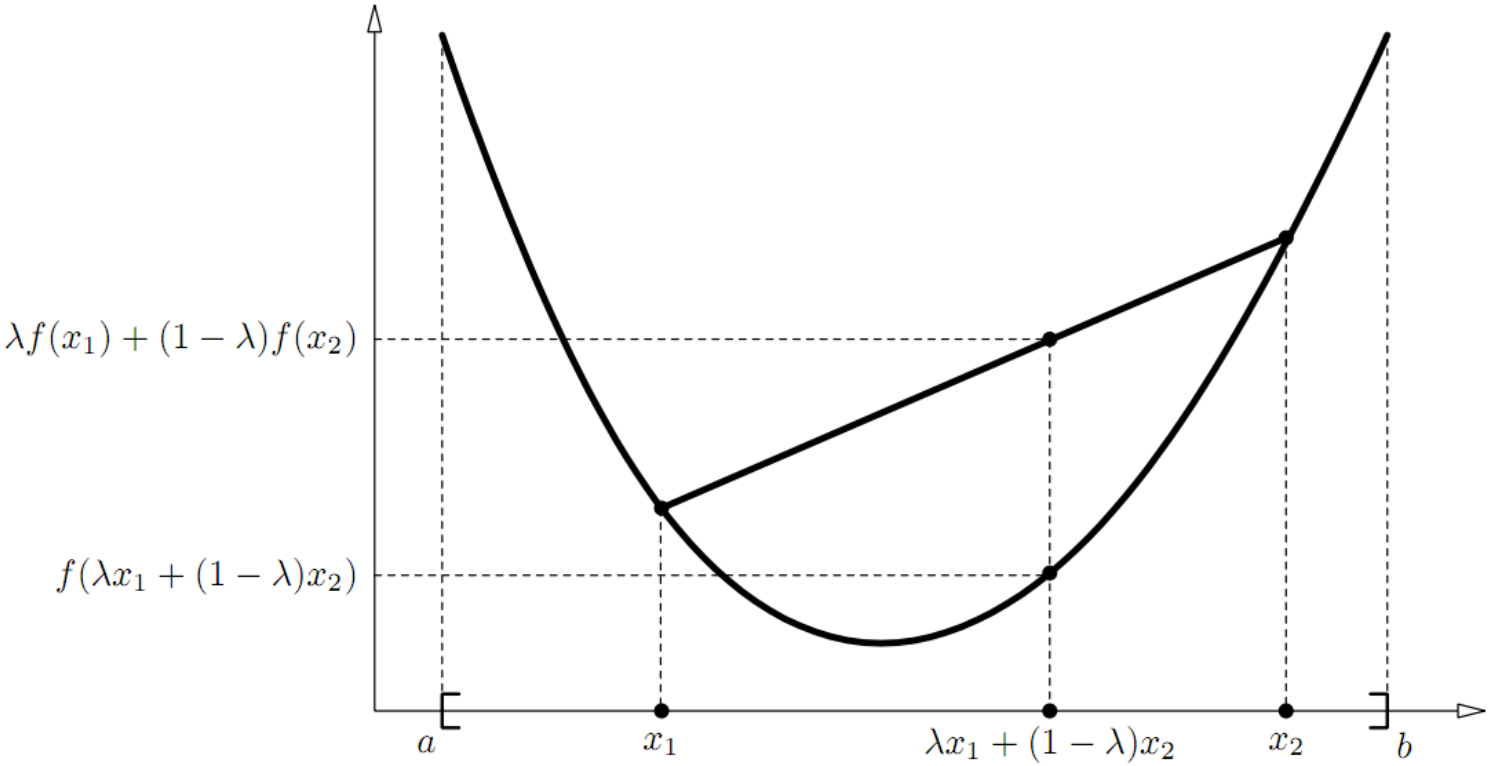
Convex sets

- A set C is convex if the line segment between any two points in C lies in C
- For any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in C$ and θ with $0 \leq \theta \leq 1$, we have

$$\theta \boldsymbol{x}_1 + (1 - \theta)\boldsymbol{x}_2 \in C$$

Convex functions

- $f: \mathbb{R}^n \to \mathbb{R}$ is convex if $\mathrm{dom} f$ is a convex set and
$$f(\theta \boldsymbol{x} + (1-\theta)\boldsymbol{y}) \leq \theta f(\boldsymbol{x}) + (1-\theta) f(\boldsymbol{y})$$
for all $\boldsymbol{x}, \boldsymbol{y} \in \mathrm{dom} f, 0 \leq \theta \leq 1$



- f is concave if $-f$ is convex
- f is strictly convex if $\mathrm{dom} f$ is a convex set and
$$f(\theta \boldsymbol{x} + (1-\theta)\boldsymbol{y}) < \theta f(\boldsymbol{x}) + (1-\theta) f(\boldsymbol{y})$$
for all $\boldsymbol{x}, \boldsymbol{y} \in \mathrm{dom} f, \boldsymbol{x} \neq \boldsymbol{y}, 0 < \theta < 1$

- Example:
  - $\log x$, $\ln x$ on $\mathbb{R}^+$ is strictly concave
  - $ax + b$ on $\mathbb{R}$ is both concave and convex
  - e^{ax}, for any $a \in \mathbb{R}$ is convex

Jensen's inequality

- if f is convex, then

$$f\left(\sum_{i=1}^{n}\lambda_i x_i\right) \le \sum_{i=1}^{n}\lambda_i f(x_i) \qquad \lambda_i \ge 0, \sum_i \lambda_i = 1$$

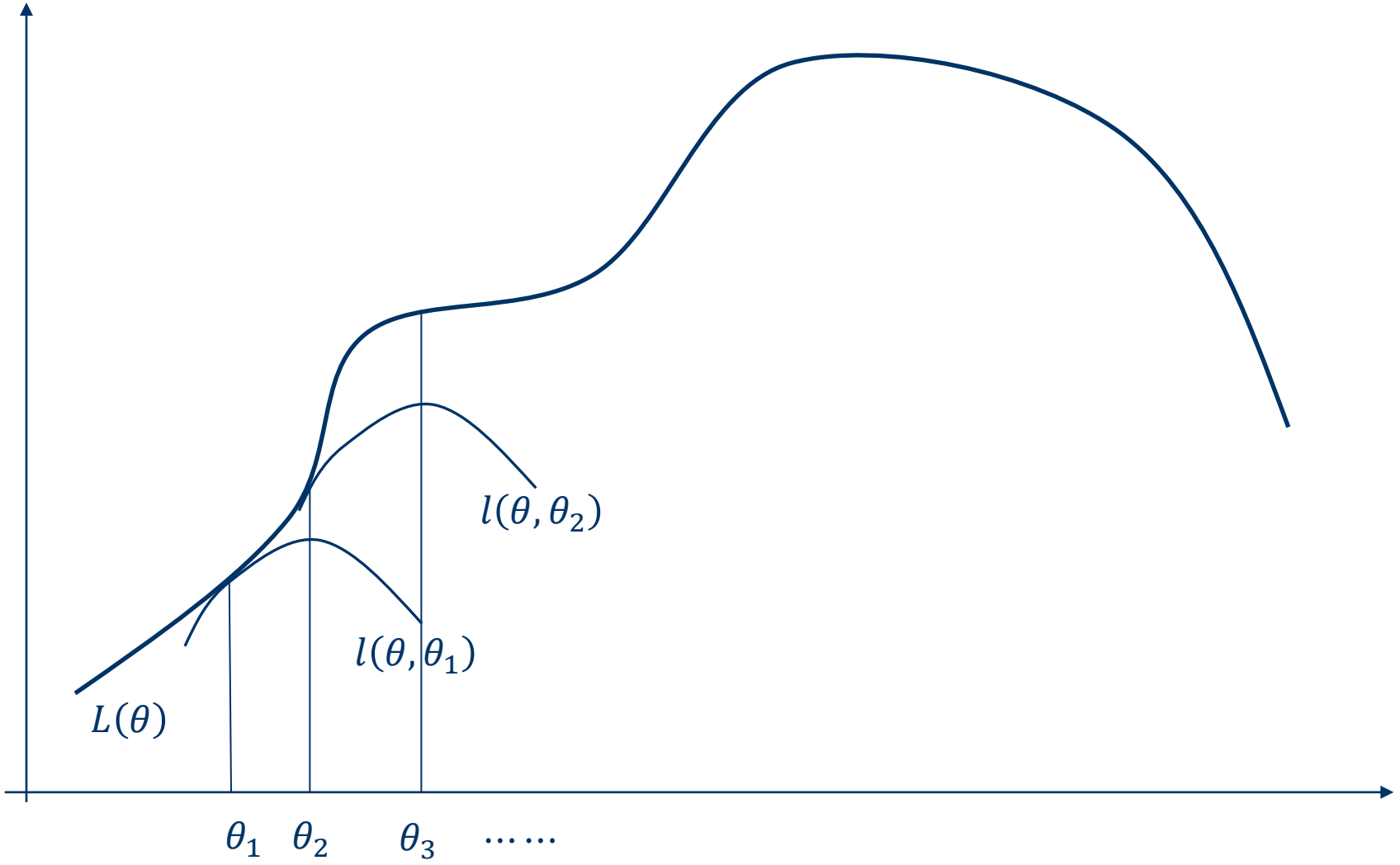
$$f(E[X]) \le E[f(X)]$$

- for any random variable X
- if f is strictly convex, then $E[f(X)] = f(E[X])$ holds true if and only if X=$E[X]$ with probability 1 (i.e., if X is a constant)

- $\ln x$: strictly concave

$$\ln\left(\sum_{i=1}^{n}\lambda_i x_i\right) \ge \sum_{i=1}^{n}\lambda_i \ln(x_i)$$

How EM works: an illustrative example



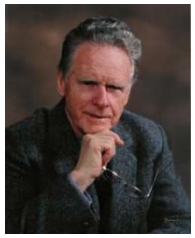
Problem Description

- We have a model $P(\boldsymbol{x}, z; \boldsymbol{\theta})$ and we only observe $\boldsymbol{x}$
- Suppose we have a training set $\{\boldsymbol{x}_1 \cdots \boldsymbol{x}_n\}$ consisting of n independent examples. We wish to fit the parameters to the data, where the log-likelihood is given by

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log P(\boldsymbol{x}_i; \boldsymbol{\theta}) = \sum_{i=1}^{n} \log \sum_{z} P(\boldsymbol{x}_i, z; \boldsymbol{\theta})$$

Non-concave function!

- EM algorithm gives an efficient iterative procedure for maximum likelihood estimation with latent variables z
- The EM algorithm was explained and given its name in a classic 1977 paper by Arthur Dempster, Nan Laird, and Donald Rubin.



Arthur Dempster, Nan Laird, and Donald Rubin

EM algorithm

- Each iteration of the EM algorithm consists of two processes: The Expectation-step, and the Maximization-step.

  Maximize a non-concave function

- In the expectation, or E-step, the missing data are estimated given the observed data and current estimate of the model parameters.

  Find a tight lower bound concave function

- In the M-step, the likelihood function is maximized under the assumption that the missing data are known. The estimate of the missing data from the E-step are used as the actual missing data.

  Maximize the concave function

- Convergence is assured since the algorithm is guaranteed to increase the likelihood at each iteration.

Details

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \sum_{z} P(\boldsymbol{x}_i, z; \boldsymbol{\theta})$$

Let Q be some distribution over the z's .

Note that $\sum_z Q(z) = 1$, $Q(z) \geq 0$, we get:

$$E_{z\sim Q}\left[\frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{Q(z)}\right]$$

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \sum_{z} P(\boldsymbol{x}_i, z; \boldsymbol{\theta}) = \sum_{i=1}^{n} \log \sum_{z} Q(z) \frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{Q(z)}$$

$$\geq \sum_{i=1}^{n} \sum_{z} Q(z) \log \frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{Q(z)} = l(\boldsymbol{\theta})$$

log is a strictly concave function

$$E_{z\sim Q}\left[\log \frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{Q(z)}\right]$$

If f is strictly concave, then $E[f(X)] = f(E[X])$ holds true if and only if X=$E[X]$ with probability 1

$\frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{Q(z)}$ is a constant $\quad Q(z) \propto P(\boldsymbol{x}_i, z; \boldsymbol{\theta}) \quad \sum_z Q(z) = 1 \quad Q(z) = \frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{\sum_z P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}$

Details

$$Q(z) = \frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{\sum_z P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}$$

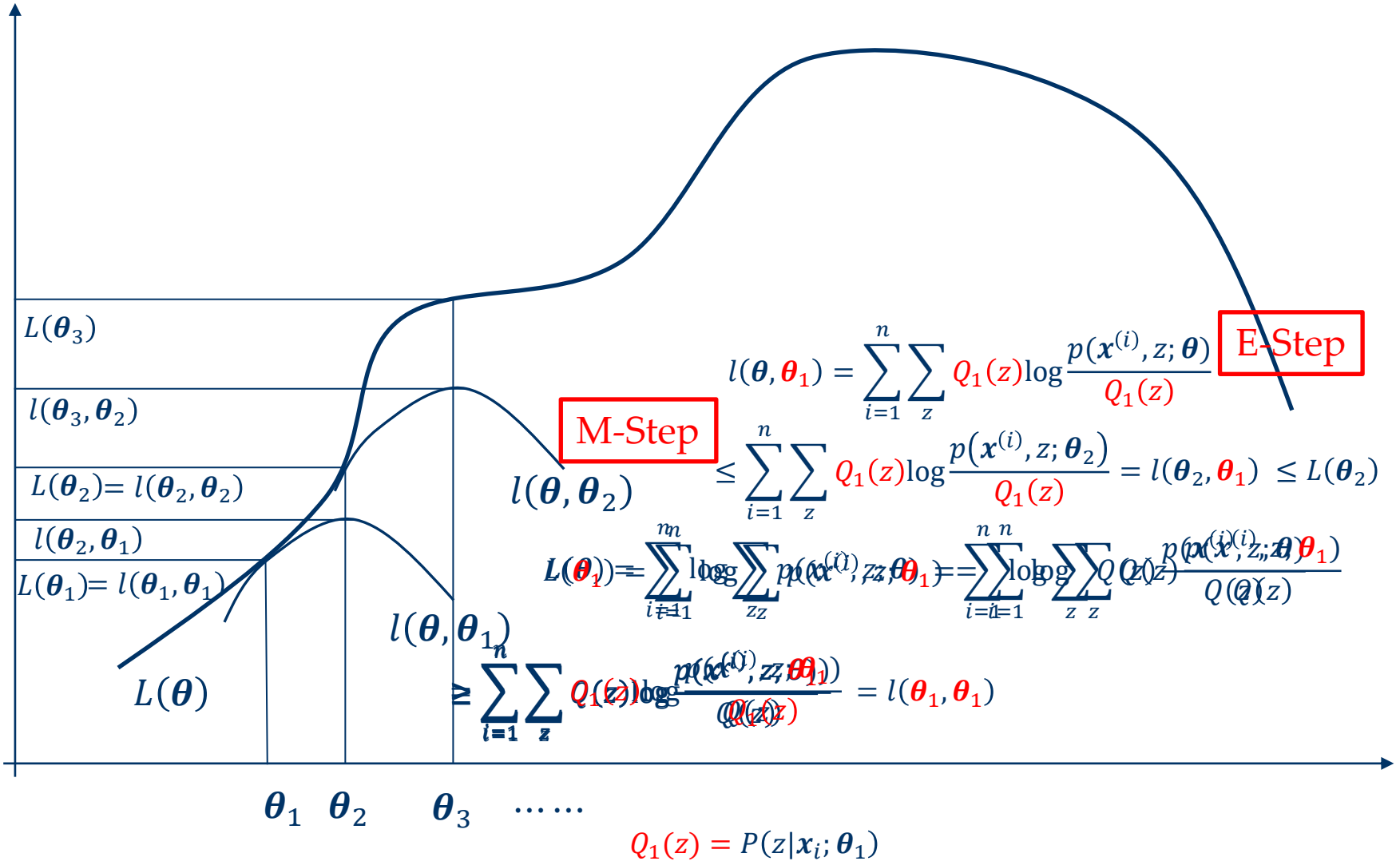
$$= \frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{P(\boldsymbol{x}_i; \boldsymbol{\theta})}$$

$$= P(z|\boldsymbol{x}_i; \boldsymbol{\theta})$$

Satisfying the requirement:

$\frac{P(\boldsymbol{x}_i, z; \boldsymbol{\theta})}{Q(z)}$ = constant that does not depend on z

Proof of convergence



Algorithm Summary

Repeat until convergence {

(E-step) for each i :

$$Q(z) = P\left(z|\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right)$$

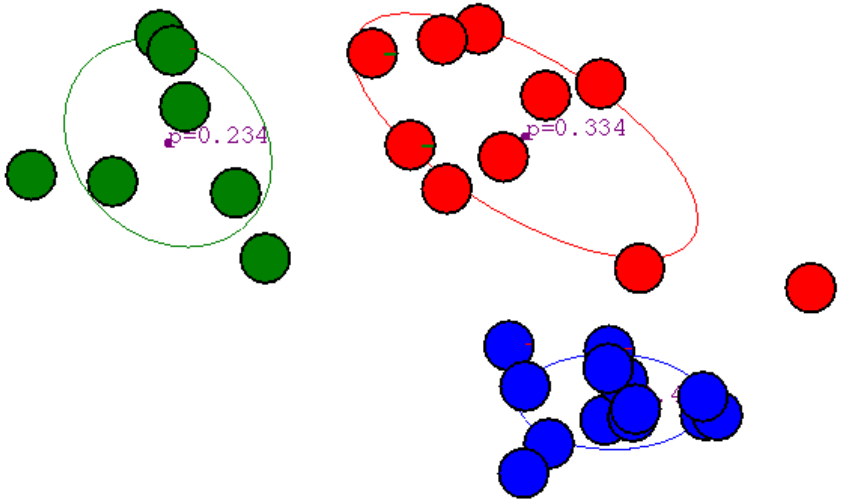
(M-step) Set:

$$\boldsymbol{\theta} := \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{m} \sum_{z} Q(z) \log \frac{p\left(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)};\boldsymbol{\theta}\right)}{Q(z)}$$

}

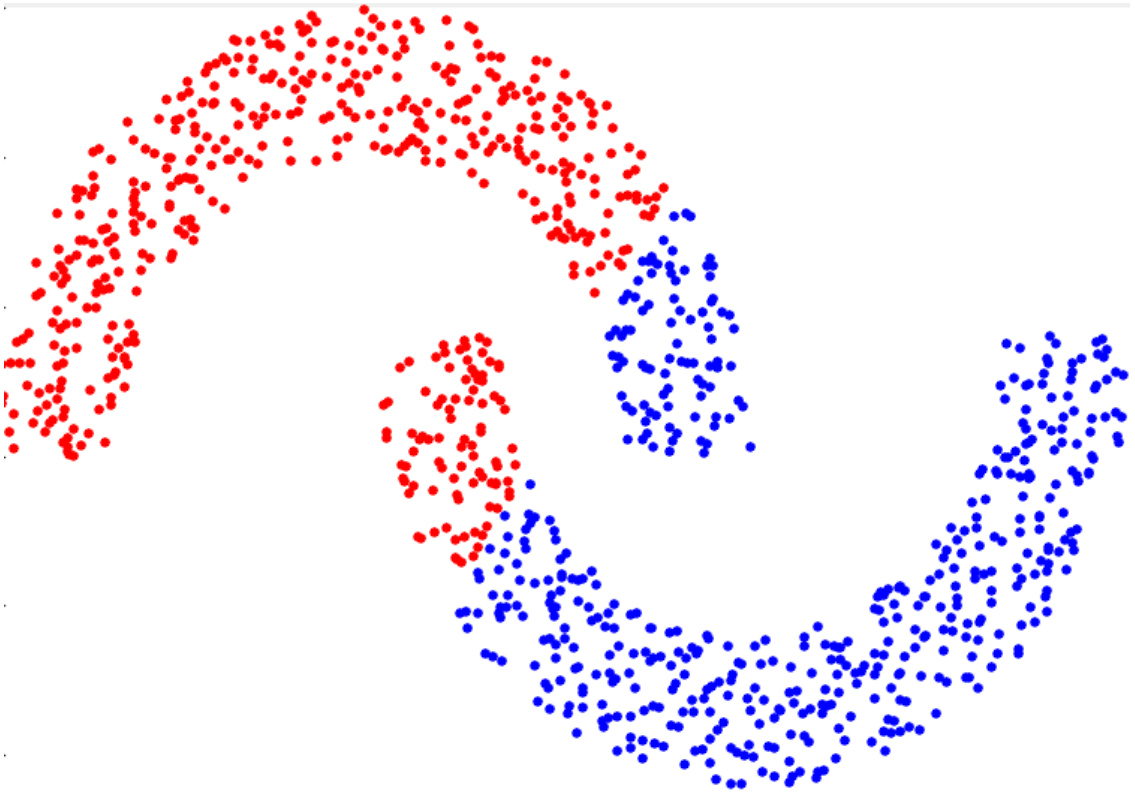
Since the function is not convex, it is possible for the algorithm to converge to local minima or saddle points in unusual cases

Gaussian Mixture Model: An example



Distance versus connectivity based on density

Kmeans Output



Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as density-connected points

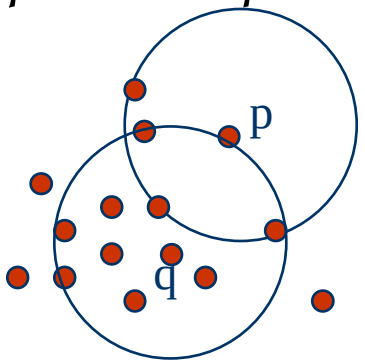
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition

- Several interesting studies:
  - DBSCAN: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - DENCLUE: Hinneburg & D. Keim (KDD'98)
  - CLIQUE: Agrawal, et al. (SIGMOD'98)

Density-Based Clustering: Basic Concepts

- Two parameters:
  - *Eps*: Maximum radius of the neighborhood
  - *MinPts*: Minimum number of points in an Eps-neighborhood of that point
- $N_{Eps}(p)$: {q belongs to D | 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:

$$|N_{Eps}(q)| \geq MinPts$$

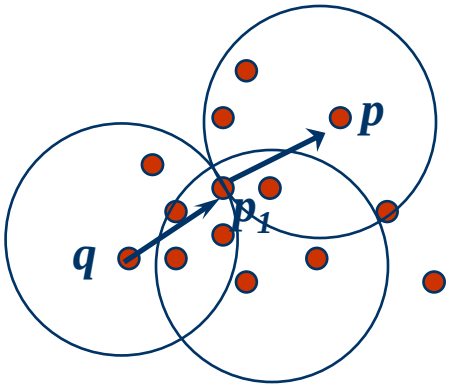


MinPts = 5

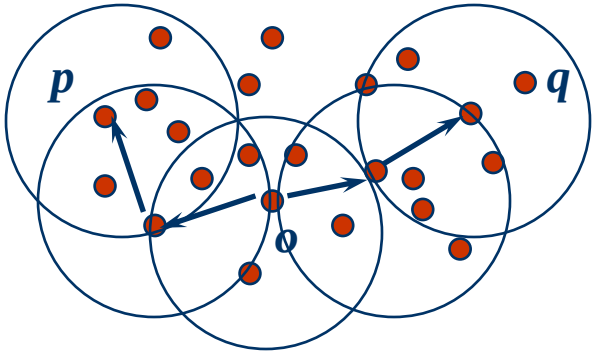
Eps = 1 cm

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

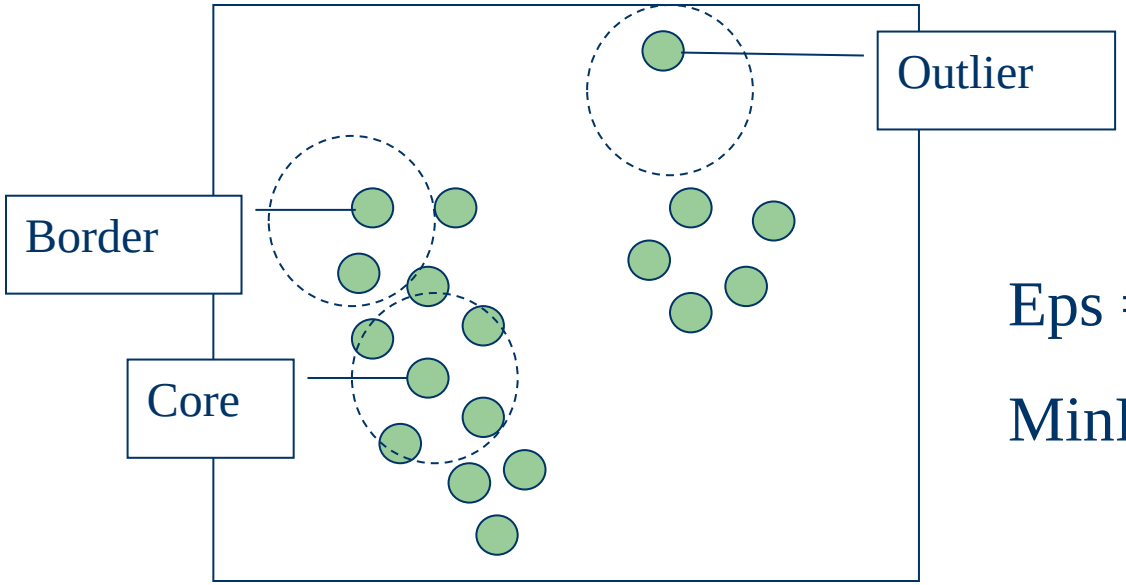


- Density-connected
  - A point p is density-connected to a point q w.r.t. *Eps*, *MinPts* if there is a point o such that both, p and q are density-reachable from o w.r.t. *Eps* and *MinPts*



DBSCAN: Density-Based Spatial Clustering of Applications with Noise

- Relies on a *density-based* notion of cluster: A *cluster* is defined as a maximal set of density-connected points
- Discovers clusters of arbitrary shape in spatial databases with noise



DBSCAN: The Algorithm

1. Arbitrary select a point *p*
2. Retrieve all points density-reachable from *p* w.r.t. *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: Sensitive to Parameters

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

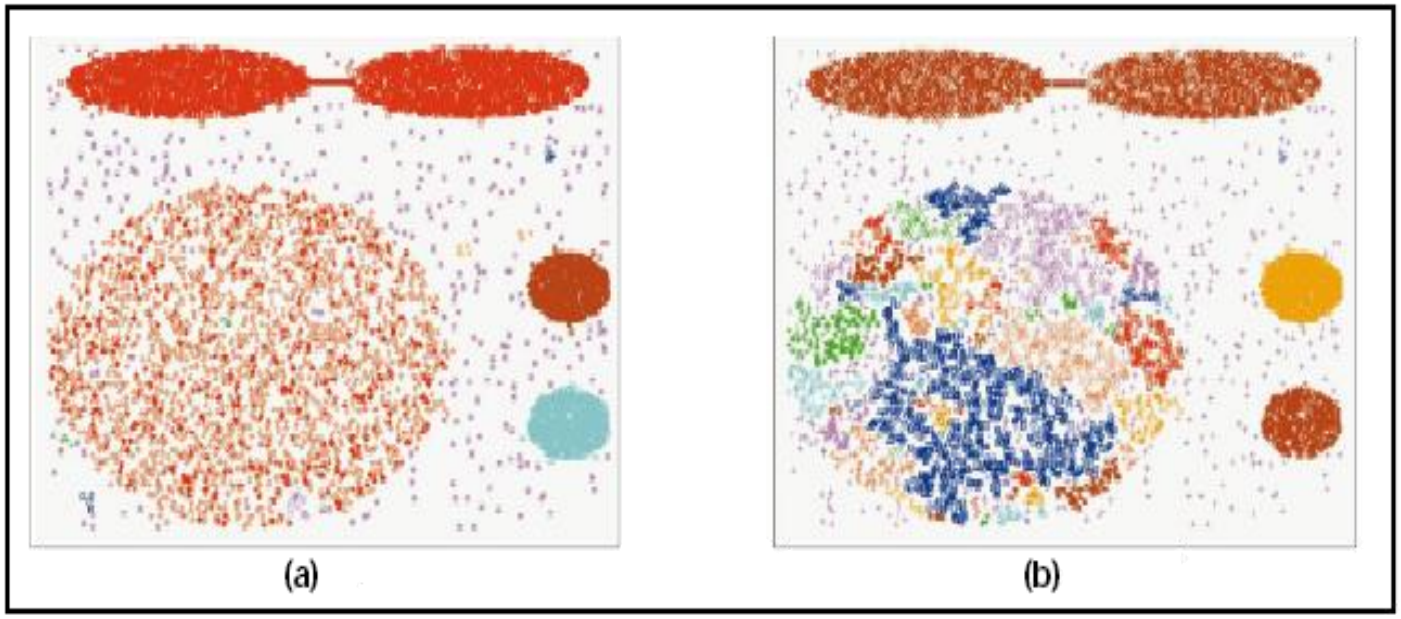


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.