Representative based clustering
Chapter 13

Clustering: what and why?

- Clustering is a method of partitioning a set of data instances into k different groups, such that similar data instances belong to the same group
- For clustering task, the label of data instances are not available, so clustering is also called unsupervised data analysis
- Clustering is important because it provides an insight about the data by finding subgroups of similar data points.
  - After clustering, the properties of each group can be studied independently
- In scenarios, where we are mostly interested in classification, we sometimes perform clustering because the labeling of data instances is costly.
- Clustering methods:
  - Representative Based clustering
  - Density based Clustering
  - Graph or spectral clustering

Representative Based Clustering

- Given
  - a dataset of n points in a d-dimensional space, $D = \{\boldsymbol{x}_i\}_{i=1}^{n}$
  - the number of desired clusters k,
- Goal of representative based clustering is
  - partition the data into k groups $C = \{C_1, C_2, \dots, C_k\}$ so that similar objects are in the same clusters
  - For each cluster, C_i, find a representative data objects (say, $\boldsymbol{\mu}_i$)
  - We typically try to minimize the following objective function, called sum of square errors $SSE(C) = \sum_{i=1}^{k} \sum_{\boldsymbol{x}_j \in C_i} \left\| \boldsymbol{x}_j - \boldsymbol{\mu}_i \right\|^2, \mu_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x}_j \in C_i} \boldsymbol{x}_j$
- This is a combinatorial optimization problem, which is difficult to solve
- Brute-force solution is also not possible, because of exponentially many partitioning.
  - The number of ways n objects can be partitioned into k groups is $S(n,k) = \frac{1}{k!} \sum_{t=0}^{k} (-1)^t \binom{k}{t} (k-t)^n$
  - Brute-force method is not possible

k-means Algorithm

- A greedy iterative algorithm for representative based clustering
- Very efficient, if the dataset are points in R^d
- Does not obtain global optimal solution, but provides a local optimal guaranty
  - Reassigning the cluster membership of exactly one data instance (keeping the membership of the remaining data instances unchanged), does not yield a better solution
- Algorithm
  1. Randomly choose k points in the data space as initial cluster means (also called cluster seeds)
  2. Assign each data points to the closest seed, points assigned to the same seed makes a cluster
  3. Re-compute a new mean for each cluster, as the centroid of the data points in that cluster
  4. Repeat between step 2 and 3 until the convergence is reached

k-means Pseudocode

Algorithm 13.1: K-means Algorithm

K-MEANS $(\mathbf{D}, k, \epsilon)$:

$t = 0$

Randomly initialize k centroids: $\boldsymbol{\mu}_1^t, \boldsymbol{\mu}_2^t, \ldots, \boldsymbol{\mu}_k^t$

repeat

  $t = t + 1$

  `// Cluster Assignment Step`

  foreach $\mathbf{x}_j \in \mathbf{D}$ **do**

    $j^* = \arg\min_i \{\|\mathbf{x}_j - \boldsymbol{\mu}_i^t\|^2\}$ `// Assign` $\mathbf{x}_j$ `to closest centroid`

    $C_{j^*} = C_{j^*} \cup \{\mathbf{x}_j\}$

  `// Centroid Update Step`

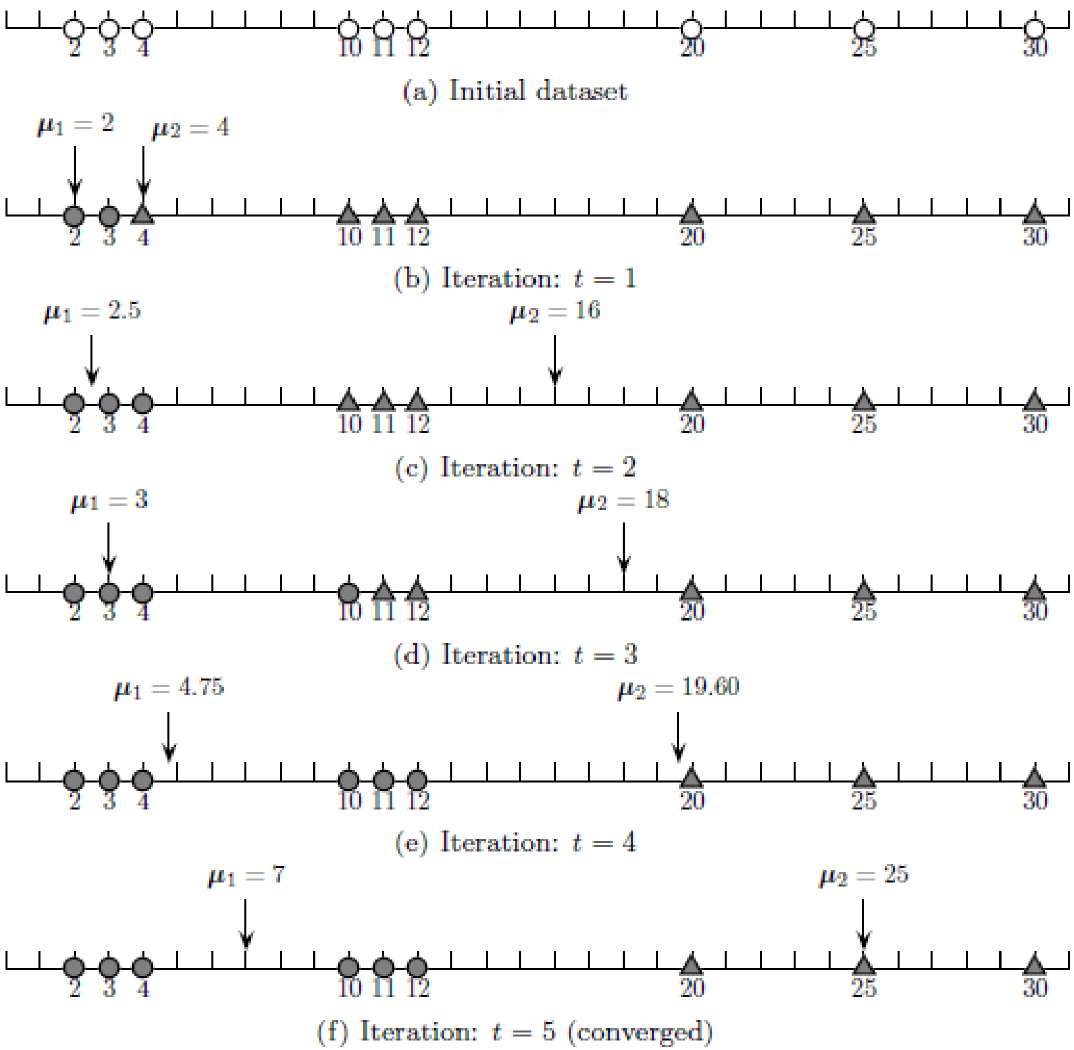
  foreach $i = 1$ *to* k **do**

    $\boldsymbol{\mu}_i^t = \frac{1}{|C_i|} \sum_{\mathbf{x}_j \in C_i} \mathbf{x}_j$

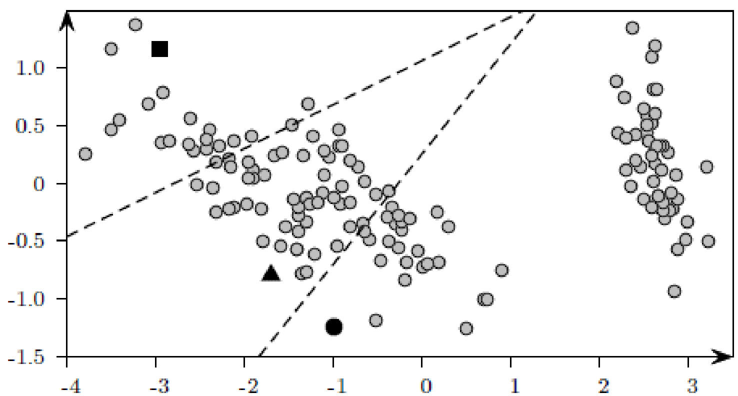
until $\sum_{i=1}^{k} \|\boldsymbol{\mu}_i^t - \boldsymbol{\mu}_i^{t-1}\| \leq \epsilon$

Complexity: is $O(nkdt)$, n : data points, d : dimension size, k : number of clusters, t : number of iterations

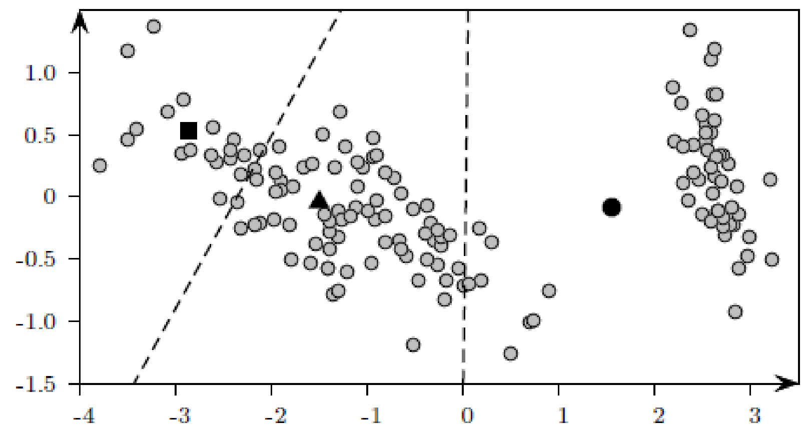
Example (1-D)


2 3 4 10 11 12 20 25 30
(a) Initial dataset
$\mu_1 = 2$ $\mu_2 = 4$
(b) Iteration: $t = 1$
$\mu_1 = 2.5$ $\mu_2 = 16$
(c) Iteration: $t = 2$
$\mu_1 = 3$ $\mu_2 = 18$
(d) Iteration: $t = 3$
$\mu_1 = 4.75$ $\mu_2 = 19.60$
(e) Iteration: $t = 4$
$\mu_1 = 7$ $\mu_2 = 25$
(f) Iteration: $t = 5$ (converged)

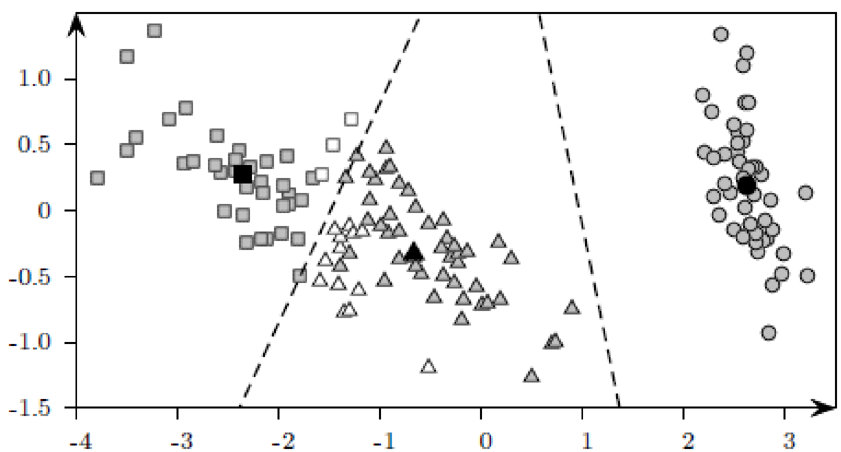
Example (2 –Dimension)



(a) Random Initialization: $t = 0$



(b) Iteration: $t = 1$



(c) Iteration: $t = 8$ (converged)

Expectation Maximization (EM) Clustering

- The k-means approach is an example of hard assignment, where each point can belong to only one cluster
- EM clustering allow soft cluster assignment, where each point has a distinct probability of belonging to each of the clusters
- EM assumes that a cluster is modeled as a multivariate normal distribution with parameters $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$.
- It also assumes that each of the data points is generated by a mixture of k clusters, denoted by $P(C_i)$ such that $\sum_{i=1}^{k} P(C_i) = 1$
- Thus solving the clustering is to find the model parameters $\theta = \{\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, P(C_1), \cdots, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, P(C_k)\}$
- Assuming each column as a random variable X_j, and $X = (X_1, X_2, \cdots, X_d)$ is a vector-random variable. A data point, $\boldsymbol{x}$ can be assumed to be as an instance of the variable X
- Thus the probability density function of the dataset X is given as a mixture model of the k cluster normals, i.e., $f(x) = \sum_{i=1}^{k} f_i(\boldsymbol{x}) P(C_i) = \sum_{i=1}^{k} f(x|\mu_i, \Sigma_i) P(C_i)$

Maximum Likelihood Estimation (MLE)

- For a parametric model, MLE process chooses the parameter θ that maximizes the likelihood $\left(P(D|\theta)\right)$ of the observed data. $\theta^* = \arg\max_{\theta} P(D|\theta)$
- Since each of the n points are assumed to be chosen from an iid distribution, $P(D|\theta) = \prod_{j=1}^{n} f(x_j)$. Instead of maximizing the likelihood, we can also maximize the log-likelihood, $\log P(D|\theta) = \log \prod_{j=1}^{n} f\left(\boldsymbol{x}_j\right) = \sum_{j=1}^{n} \log\left(\sum_{i=1}^{k} f\left(\boldsymbol{x}_j \middle| \boldsymbol{\mu}_i, \Sigma_i\right) P(C_i)\right)$
- Directly maximizing log-likelihood is a difficult task, so we can use the Expectation-Maximization (EM) approach for finding the MLE estimate of θ, which is θ^*.
- EM starts from an initial guess of θ and improve θ iteratively.
  - Expectation step computes the posterior probability $P\left(C_i \middle| x_j\right) = \frac{P\left(\boldsymbol{x}_j \middle| C_i\right) \cdot P(C_i)}{\sum_{a=1}^{k} P\left(\boldsymbol{x}_j \middle| C_a\right) \cdot P(C_a)} = \frac{f\left(\boldsymbol{x}_j \middle| \pi_i, \Sigma_i\right) \cdot P(C_i)}{\sum_{a=1}^{k} f\left(\boldsymbol{x}_j \middle| \pi_a, \Sigma_a\right) \cdot P(C_a)}$
  - Maximization step uses the weights $P\left(C_i \middle| x_j\right)$ to re-estimate θ
  - The process continue until a convergence criteria is met

EM in one Dimension

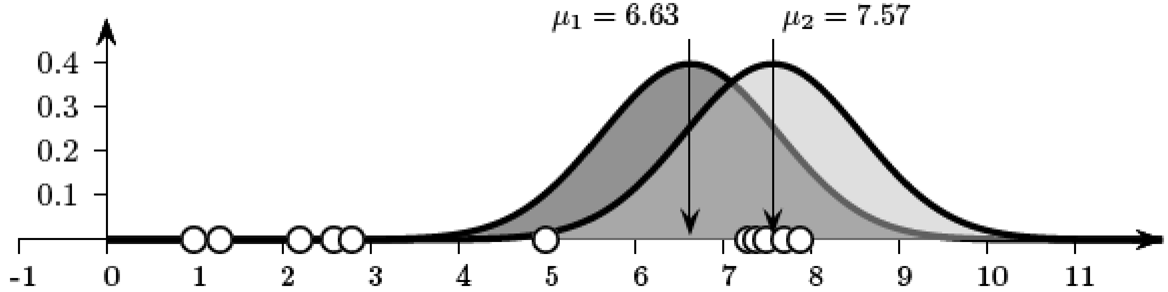
- Consider D consists of points in one dimension, $f_i(x) = f(x|\mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left\{-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right\}$
- Initialization: For each cluster C_i we initialize μ_i, σ^2, and $P(C_i)$ randomly. Mean is selected uniformly at random from the range of possible values, σ^2 is typically assumed to be 1, and $P(C_i) = 1/k$
- Expectation step: This step computes the posterior probabilities: $P(C_i|x_j) = \frac{f(x_j|\mu_i, \sigma_i^2) \cdot P(C_i)}{\sum_{a=1}^{k} f(x_j|\mu_a, \sigma_a^2) \cdot P(C_a)}$
- For convenience, we assume that $P(c_i|x_j) = w_{ij}$. Also assume that $w_i = (w_{i1}, w_{i2}, \cdots, w_{in})^T$
- The new estimate of cluster mean is simply the weighted mean of all the points: $\mu_i = \frac{\sum_{j=1}^{n} w_{ij} \cdot x_j}{\sum_{j=1}^{n} w_{ij}}$
- The new estimate of the cluster variance is the weighted variance across all the points

$$\sigma_i^2 = \frac{\sum_{j=1}^{n} w_{ij}(x_j - \mu_i)^2}{\sum_{j=1}^{n} w_{ij}}$$

- Finally the probability $P(C_i)$ is simply the fraction of total weight belonging to the cluster C_i

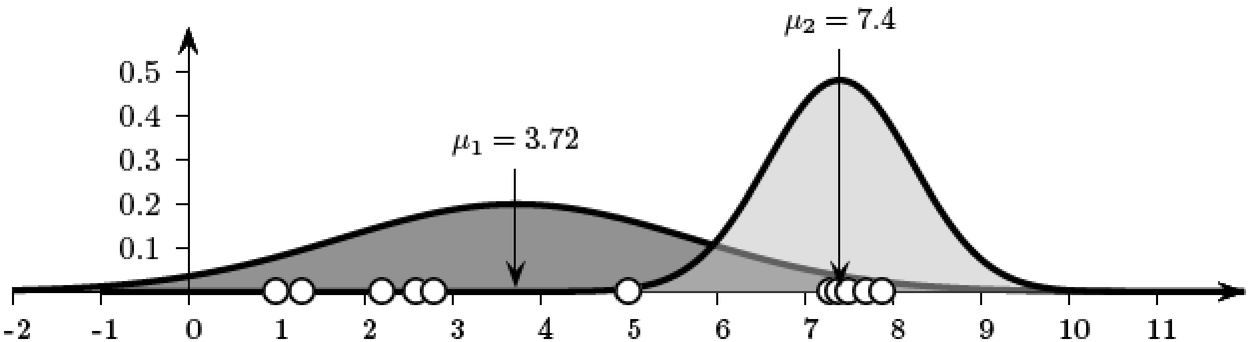
$$P(C_i) = \frac{\sum_{j=1}^{n} w_{ij}}{\sum_{a=1}^{k} \sum_{j=1}^{n} w_{aj}} = \frac{\sum_{j=1}^{n} w_{ij}}{\sum_{j=1}^{n} 1} = \frac{\sum_{j=1}^{n} w_{ij}}{n}$$

Example



(a) Initialization: $t = 0$

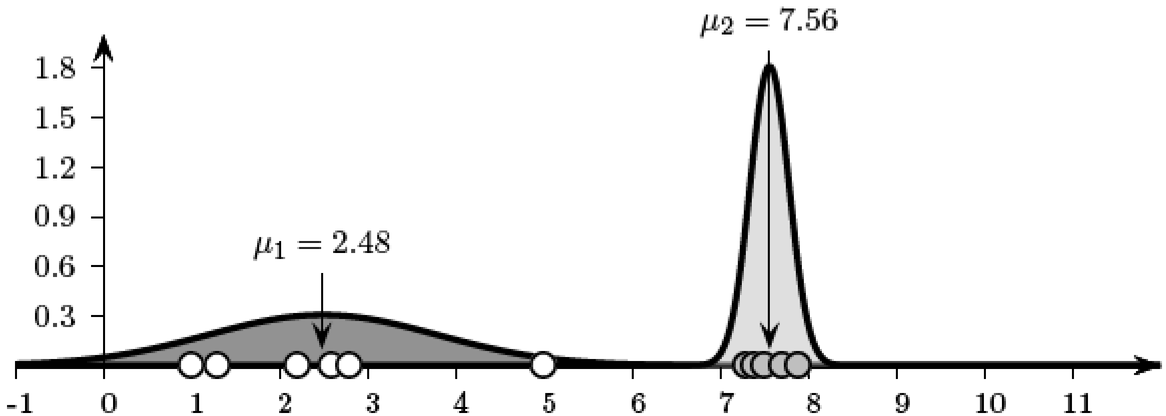
$\mu_1 = 6.63$	$\sigma_1^2 = 1$	$P(C_2) = 0.5$
$\mu_2 = 7.57$	$\sigma_2^2 = 1$	$P(C_2) = 0.5$



(b) Iteration: $t = 1$

$\mu_1 = 3.72$	$\sigma_1^2 = 6.13$	$P(C_1) = 0.71$
$\mu_2 = 7.4$	$\sigma_2^2 = 0.69$	$P(C_2) = 0.29$

Example



(c) Iteration: $t = 5$ (converged)

$\mu_1 = 2.48$	$\sigma_1^2 = 1.69$	$P(C_1) = 0.55$
$\mu_2 = 7.56$	$\sigma_2^2 = 0.05$	$P(C_2) = 0.45$

EM algorithm

Algorithm 13.3: Expectation-Maximization (EM) Algorithm

```
EXPECTATION-MAXIMIZATION (D, k, ε):
t = 0
// Random Initialization
Randomly initialize μ_1^t, ··· , μ_k^t
Σ_i^t = I, ∀i = 1, ..., k
P^t(C_i) = 1/k, ∀i = 1, ..., k
repeat
    t = t + 1
    // Expectation Step
    for i = 1, ..., k and j = 1, ..., n do
        w_ij^t = P^t(C_i|x_j) = f(x_j|μ_i, Σ_i)·P(C_i) / Σ_{a=1}^k f(x_j|μ_a, Σ_a)·P(C_a)
    // Maximization Step
    for i = 1, ..., k do
        μ_i^t = Σ_{j=1}^n w_ij·x_j / Σ_{j=1}^n w_ij
        Σ_i^t = Σ_{j=1}^n w_ij (x_j − μ_i)(x_j − μ_i)^T / Σ_{j=1}^n w_ij
        P^t(C_i) = Σ_{j=1}^n w_ij / n
until Σ_{i=1}^k ‖μ_i^t − μ_i^{t−1}‖ ≤ ε
```

Where:

$$w_{ij}^t = P^t(C_i|\mathbf{x}_j) = \frac{f(\mathbf{x}_j|\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\cdot P(C_i)}{\sum_{a=1}^k f(\mathbf{x}_j|\boldsymbol{\mu}_a,\boldsymbol{\Sigma}_a)\cdot P(C_a)}$$

$$\boldsymbol{\mu}_i^t = \frac{\sum_{j=1}^n w_{ij}\cdot\mathbf{x}_j}{\sum_{j=1}^n w_{ij}} \qquad \boldsymbol{\Sigma}_i^t = \frac{\sum_{j=1}^n w_{ij}(\mathbf{x}_j-\boldsymbol{\mu}_i)(\mathbf{x}_j-\boldsymbol{\mu}_i)^T}{\sum_{j=1}^n w_{ij}} \qquad P^t(C_i) = \frac{\sum_{j=1}^n w_{ij}}{n}$$

EM in d-dimension

- Like 1-D, we need to estimate the parameters, $\boldsymbol{\theta} = \{\boldsymbol{\mu}_i, \Sigma_i, \mathrm{P}(\mathrm{C}_i)\}_{i=1..k}$

$$\boldsymbol{\Sigma}_i = \begin{pmatrix} (\sigma_1^i)^2 & \sigma_{12}^i & \cdots & \sigma_{1d}^i \\ \sigma_{21}^i & (\sigma_2^i)^2 & \cdots & \sigma_{2d}^i \\ \vdots & \vdots & \ddots & \\ \sigma_{d1}^i & \sigma_{d2}^i & \cdots & (\sigma_d^i)^2 \end{pmatrix} \quad \Longrightarrow \quad \boldsymbol{\Sigma}_i = \begin{pmatrix} (\sigma_1^i)^2 & 0 & \cdots & 0 \\ 0 & (\sigma_2^i)^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & (\sigma_d^i)^2 \end{pmatrix}$$

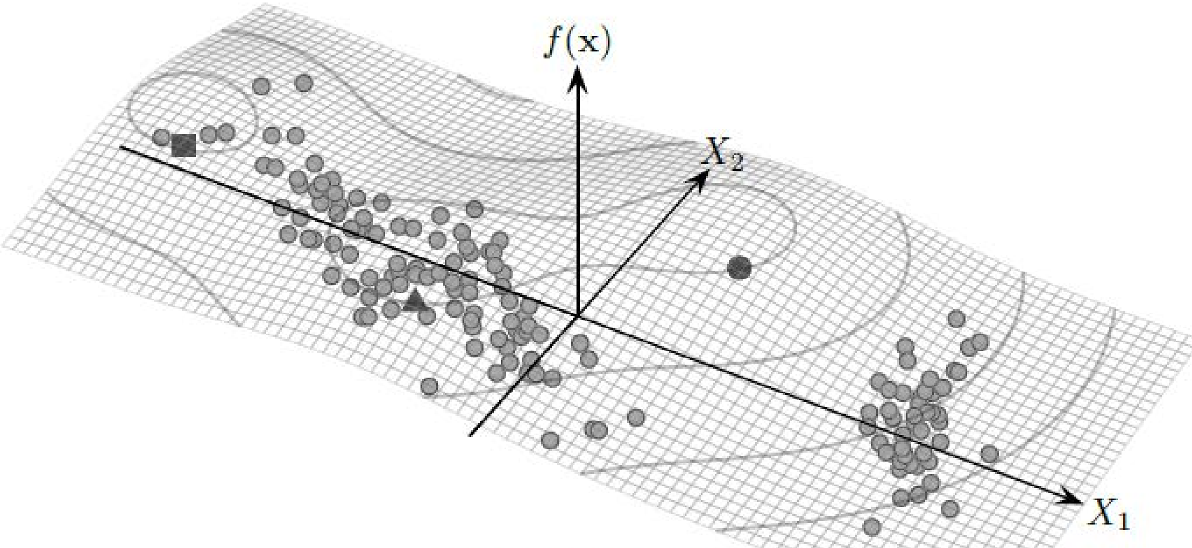
Update rules

$$\boldsymbol{\mu}_i = \frac{\sum_{j=1}^{n} w_{ij} \cdot \mathbf{x}_j}{\sum_{j=1}^{n} w_{ij}} \tag{13.11}$$

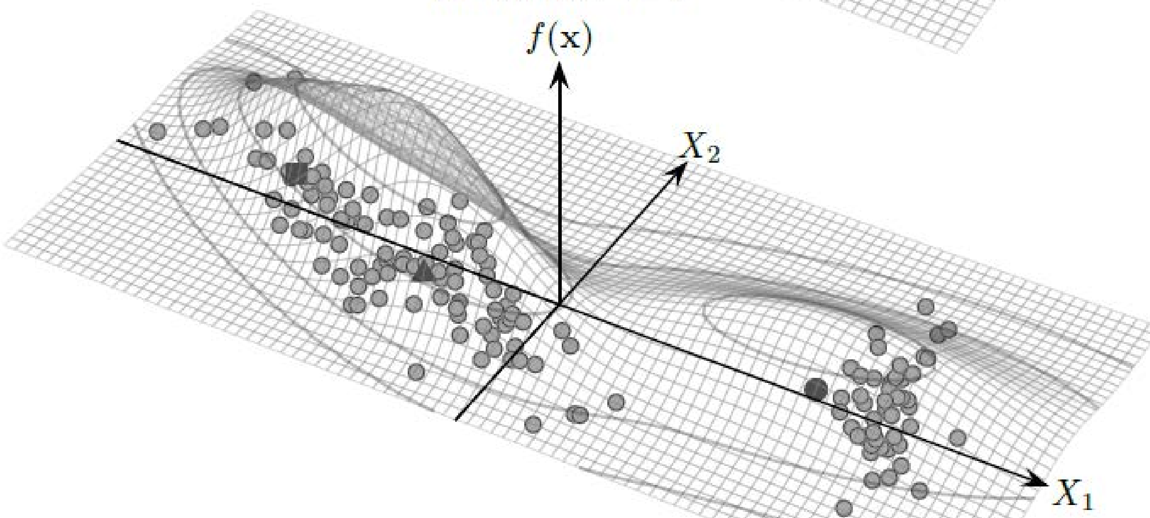
$$\boldsymbol{\Sigma}_i = \frac{\sum_{j=1}^{n} w_{ij} \mathbf{z}_{ji} \mathbf{z}_{ji}^T}{\mathbf{w}_i^T \mathbf{1}} \tag{13.12}$$

$$P(C_i) = \frac{\sum_{j=1}^{n} w_{ij}}{n} = \frac{\mathbf{w}_i^T \mathbf{1}}{n}$$

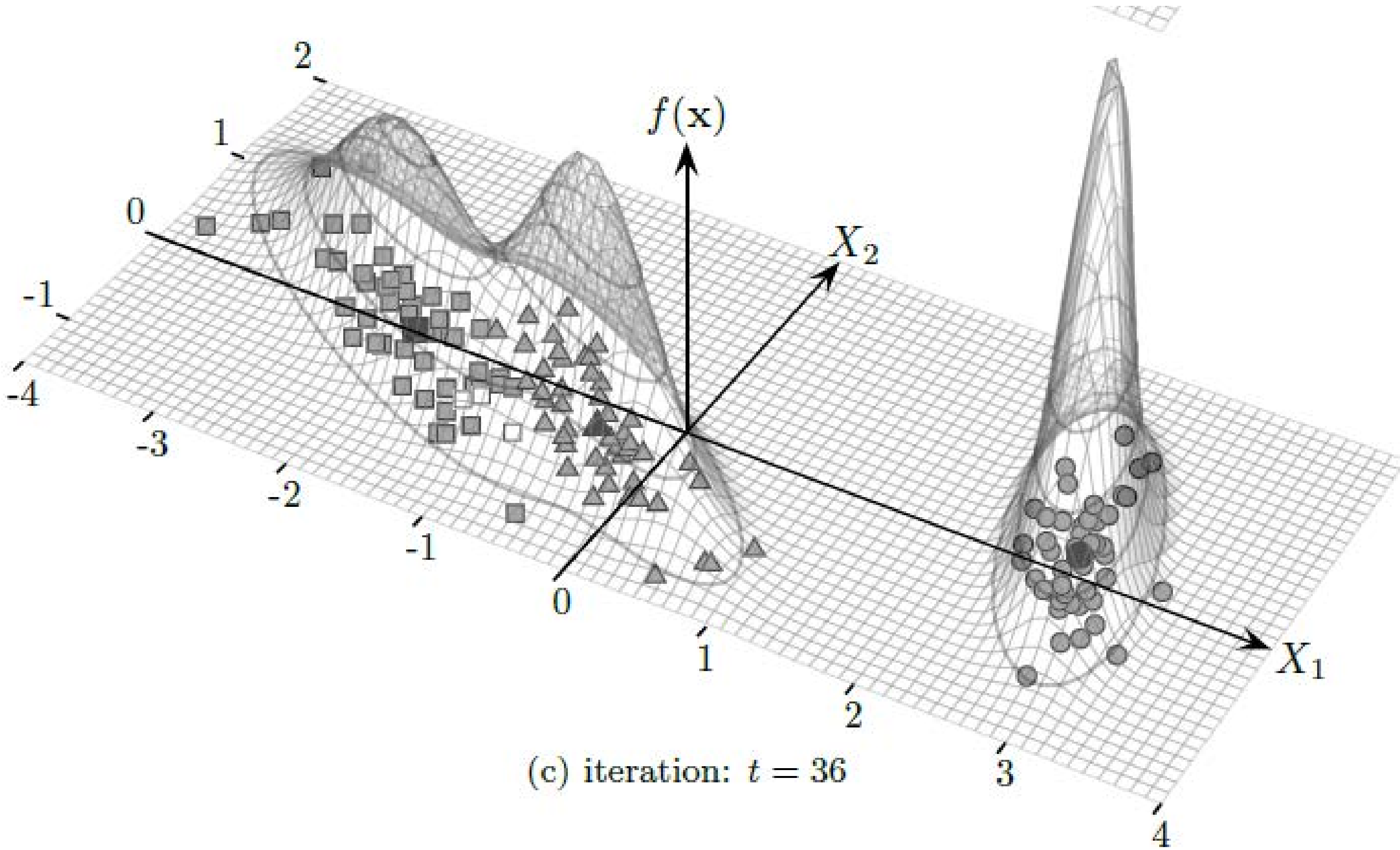
Example



(a) iteration: $t = 0$



(b) iteration: $t = 1$



(c) iteration: $t = 36$

K-Means is a special case of EM

- K-Means is obtained from the EM algorithm as follows:

$$P(\mathbf{x}_j|C_i) = \begin{cases} 1 & \text{if } C_i = \arg\min_a \left\{\|\mathbf{x}_j - \boldsymbol{\mu}_a\|^2\right\} \\ 0 & \text{otherwise} \end{cases}$$

$$P(C_i|\mathbf{x}_j) = \begin{cases} 1 & \text{if } \mathbf{x}_j \in C_i \text{, i.e., if } C_i = \arg\min_{C_a} \left\{\|\mathbf{x}_j - \boldsymbol{\mu}_a\|^2\right\} \\ 0 & \text{otherwise} \end{cases}$$