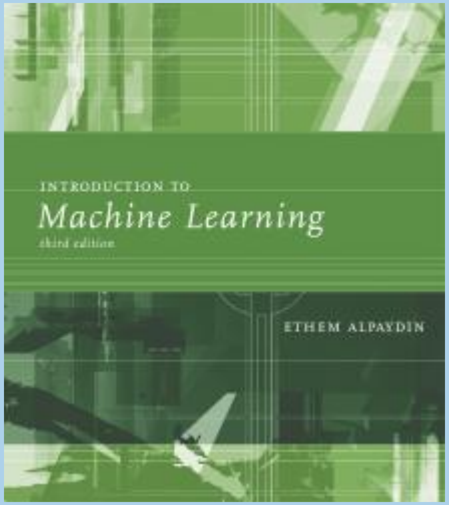


Lecture Slides for

INTRODUCTION TO MACHINE LEARNING

3RD EDITION

ETHEM ALPAYDIN

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml3e

CHAPTER 7:

CLUSTERING

Semiparametric Density Estimation

- □ Parametric: Assume a single model for $p\,(\mathbf{x} \mid C_i)$ (Chapters 4 and 5).
- □ Semiparametric: $p\,(\mathbf{x}|C_i)$ is a mixture of densities

  Multiple possible explanations/prototypes:

  Different handwriting styles, accents in speech.
- □ Nonparametric: No model; data speaks for itself (Chapter 8).

Mixture Densities

$$p(\mathbf{x}) = \sum_{i=1}^{k} p(\mathbf{x}|G_i)P(G_i)$$

where G_i the components/groups/clusters,

$P(G_i)$ mixture proportions (priors),

$p(\boldsymbol{x}|G_i)$ component densities

Gaussian mixture where $p(\boldsymbol{x}|G_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$

parameters $\Phi = \{P(G_i), \boldsymbol{\mu}_i, \Sigma_i\}_{i=1}^{k}$

unlabeled sample $X=\{\boldsymbol{x}^t\}_t$ (unsupervised learning).

Example (2-D Dataset):

$p(\boldsymbol{x}|G_1) \sim N((2, 3), \boldsymbol{\Sigma}_1)$, $P(G_1)=0.5$

$p(\boldsymbol{x}|G_2) \sim N((-4, 1), \boldsymbol{\Sigma}_2)$, $P(G_2)=0.4$

$p(\boldsymbol{x}|G_3) \sim N((0, -9), \boldsymbol{\Sigma}_3)$, $P(G_3)=0.1$

Classes vs. Clusters

- Supervised: $X = \{\boldsymbol{x}^t, \boldsymbol{r}^t\}_t$
- Classes C_i $i=1,\ldots,K$

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x}|\mathrm{C}_i)\,P(\mathrm{C}_i)$$

where $p(\boldsymbol{x}|C_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$

- $\Phi = \{P(C_i), \boldsymbol{\mu}_i, \Sigma_i\}^K_{i=1}$

$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N} \qquad \mathbf{m}_i = \frac{\sum_t r_i^t \mathbf{x}^t}{\sum_t r_i^t}$$

$$\mathbf{S}_i = \frac{\sum_t r_i^t (\mathbf{x}^t - \mathbf{m}_i)(\mathbf{x}^t - \mathbf{m}_i)^T}{\sum_t r_i^t}$$

- Unsupervised : $X = \{\boldsymbol{x}^t\}_t$
- Clusters G_i, $i=1,\ldots,k$

$$p(\mathbf{x}) = \sum_{i=1}^{k} p(\mathbf{x}|G_i)\,P(G_i)$$

where $p(\boldsymbol{x}|\mathrm{G}_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$

- $\Phi = \{P(\mathrm{G}_i), \boldsymbol{\mu}_i, \Sigma_i\}^k_{i=1}$

Labels $\boldsymbol{r}^t_i$?

Motivation: Why Clustering?

Problem: Identify (a small number of) groups of similar objects in a given (large) set of object.

Goals:

- □ Find representatives for homogeneous groups →**Data Compression**
- □ Find "natural" clusters and describe their properties →**"natural" Data Types**
- □ Find suitable and useful grouping →**"useful" Data Classes**
- □ Find unusual data object →**Outlier Detection**

k-Means Clustering

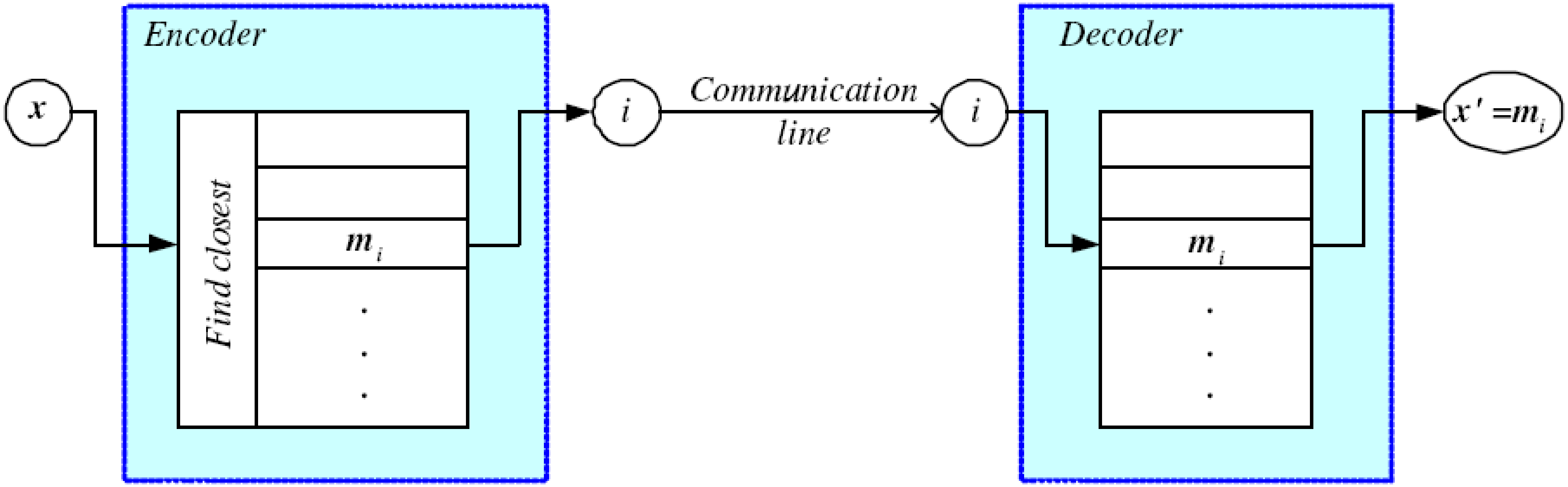
- Find *k* reference vectors (prototypes/codebook vectors/codewords) which best represent data
- Reference vectors, $\mathbf{m}_j$, j =1,…,k
- Use nearest (most similar) reference:

$$\left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_j \left\|\mathbf{x}^t - \mathbf{m}_j\right\|$$

- Reconstruction error $E\left(\{\mathbf{m}_i\}_{i=1}^{k} \middle| \mathbf{X}\right) = \sum_t \sum_i b_i^t \left\|\mathbf{x}^t - \mathbf{m}_i\right\|^2$

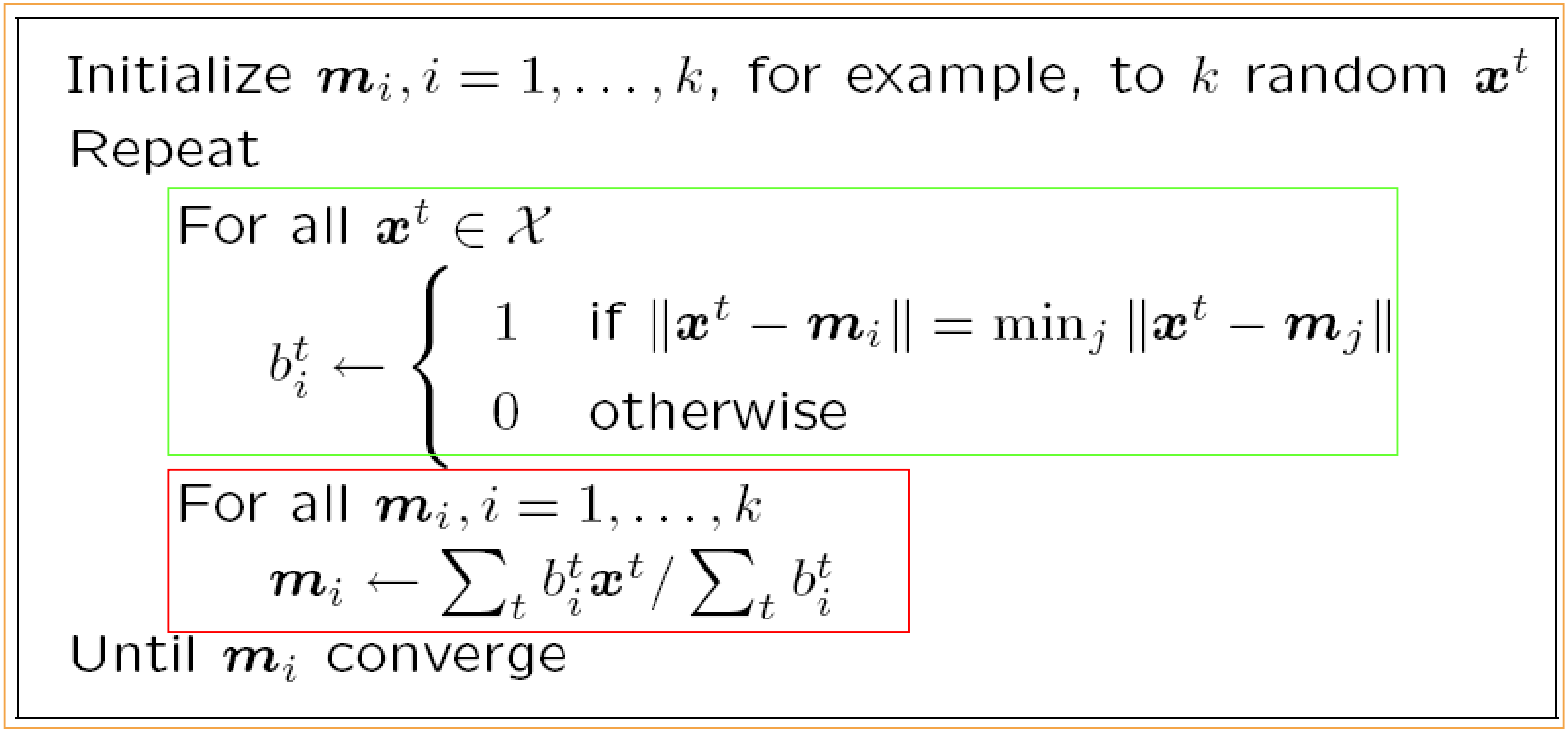
$$b_i^t = \begin{cases} 1 & \text{if } \left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_j \left\|\mathbf{x}^t - \mathbf{m}_j\right\| \\ 0 & \text{otherwise} \end{cases}$$

Encoding/Decoding

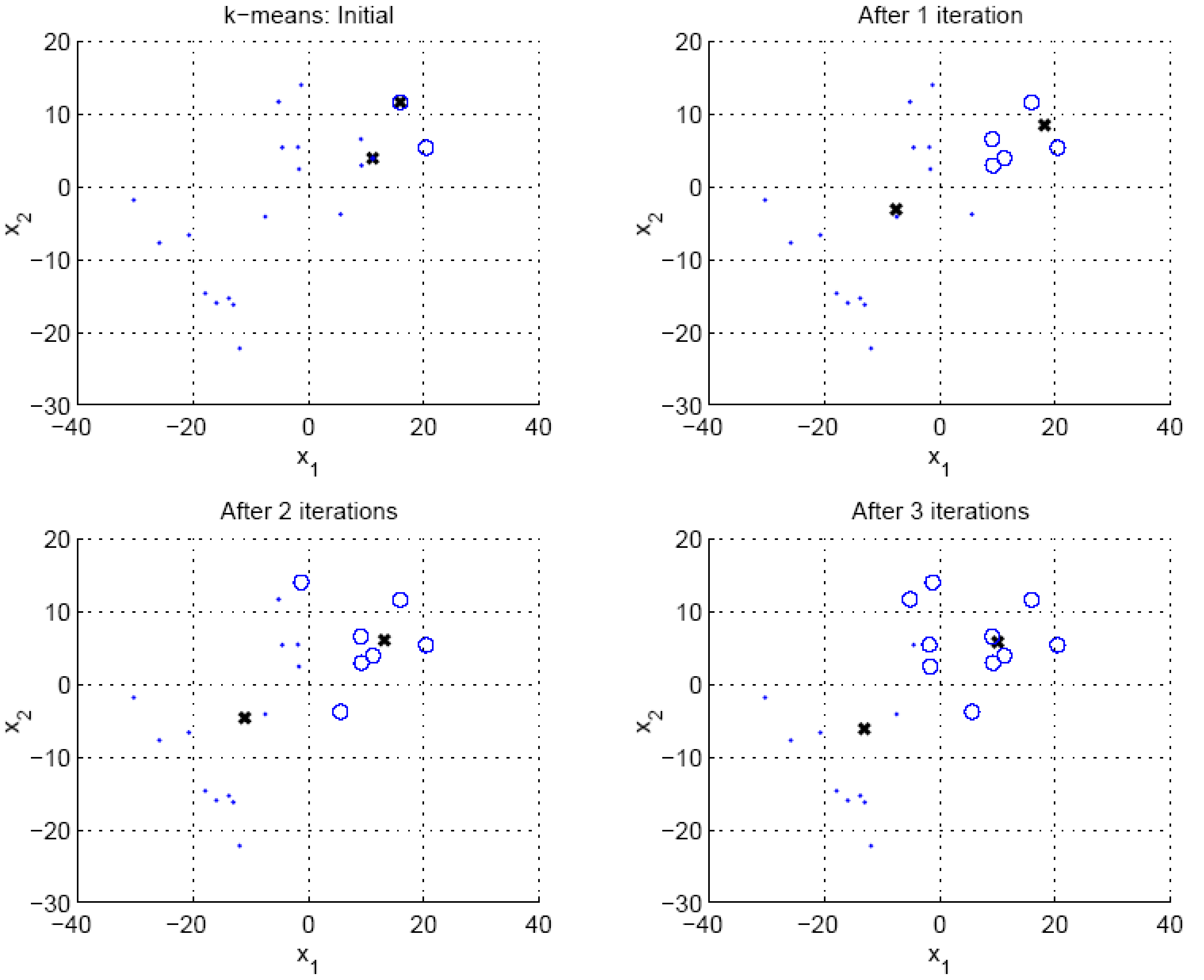


$\mathbf{m}_i$ are also called codebook vectors or code words.

k-means Clustering



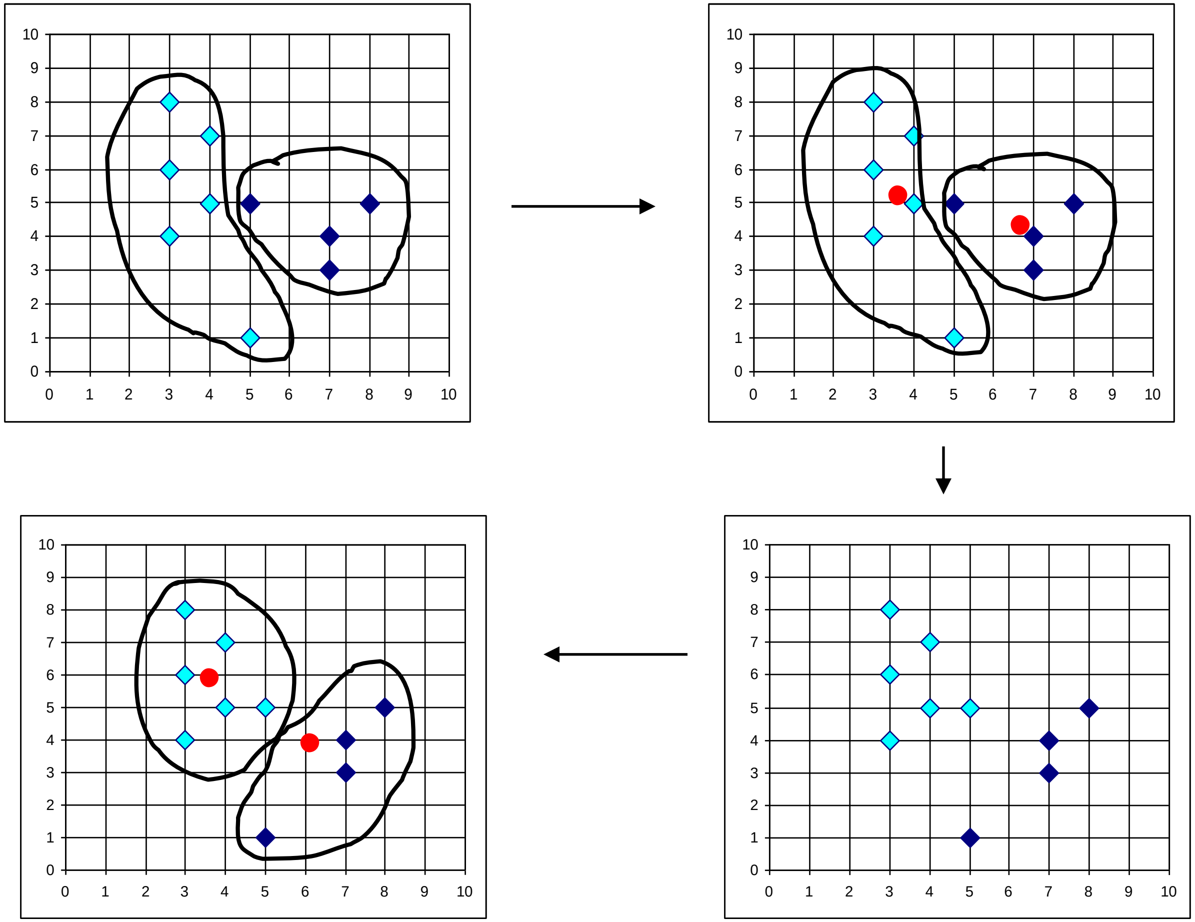
```
Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to $k$ random $\boldsymbol{x}^t$
Repeat
    For all $\boldsymbol{x}^t \in \mathcal{X}$
        $b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$
    For all $\boldsymbol{m}_i, i = 1, \ldots, k$
        $\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$
Until $\boldsymbol{m}_i$ converge
```


k−means: Initial
After 1 iteration
After 2 iterations
After 3 iterations
20
10
0
−10
−20
−30
−40
−20
0
20
40
x_1
x_2

The *K-Means* Clustering Method

- Example

Cluster→"New" Model



Model→"New" Cluster

Methods for Initialization

- Take randomly selected *k* instances as the initial $\mathbf{m}_i$.
- The mean of all data can be calculated and small random vectors may be added to the mean to get the *k* initial $\mathbf{m}_i$.
- We can calculate the principal component, divide its range into *k* equal intervals, partitioning the data into *k* groups, and then take the means of these groups as the initial centers.

Expectation-Maximization (EM)

- Log likelihood with a mixture model

$$\mathcal{L}(\Phi|\mathbf{X}) = \log \prod_t p(\mathbf{x}^t|\Phi)$$

$$= \sum_t \log \sum_{i=1}^{k} p(\mathbf{x}^t|G_i) P(G_i)$$

- Φ includes the priors $P(G_i)$ and also the sufficient statistics of the component densities $p(\mathbf{x}^t|G_i)$.
- Assume hidden variables $\boldsymbol{z}$, which when known, make optimization much simpler.
- Complete likelihood, $\mathcal{L}_c(\Phi\ |X,Z)$, in terms of $\boldsymbol{x}$ and $\boldsymbol{z}$.
- Incomplete likelihood, $\mathcal{L}(\Phi\ |X)$, in terms of $\boldsymbol{x}$.

E- and M-steps

Iterate the two steps

1. E-step: Estimate ***z*** given *X* and current Φ
2. M-step: Find new Φ given **z**, *X*, and old Φ.

$$\text{E-step:} \quad \mathrm{Q}\left(\Phi|\Phi^{l}\right)=E\left[\mathcal{L}_{C}\left(\Phi|\mathbf{X},Z\right)|\mathbf{X},\Phi^{l}\right]$$

$$\text{M-step:} \quad \Phi^{l+1}=\arg\max_{\Phi} Q\left(\Phi|\Phi^{l}\right)$$

An increase in Q increases incomplete likelihood

$$\mathcal{L}\left(\Phi^{l+1}|\mathbf{X}\right)\geq\mathcal{L}\left(\Phi^{l}|\mathbf{X}\right)$$

EM in Gaussian Mixtures

- z^t_i = 1 if $\boldsymbol{x}^t$ belongs to G$_i$, 0 otherwise (labels $\boldsymbol{r}^{\,t}_i$ of supervised learning); assume $p(\boldsymbol{x}|G_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$
- E-step: $$E\left[z_i^t \,|\, \mathbf{X}, \Phi^l\right] = \frac{p\left(\mathbf{x}^t | G_i, \Phi^l\right) P\left(G_i\right)}{\sum_j p\left(\mathbf{x}^t | G_j, \Phi^l\right) P\left(G_j\right)} = P\left(G_i | \mathbf{x}^t, \Phi^l\right) \equiv h_i^t$$

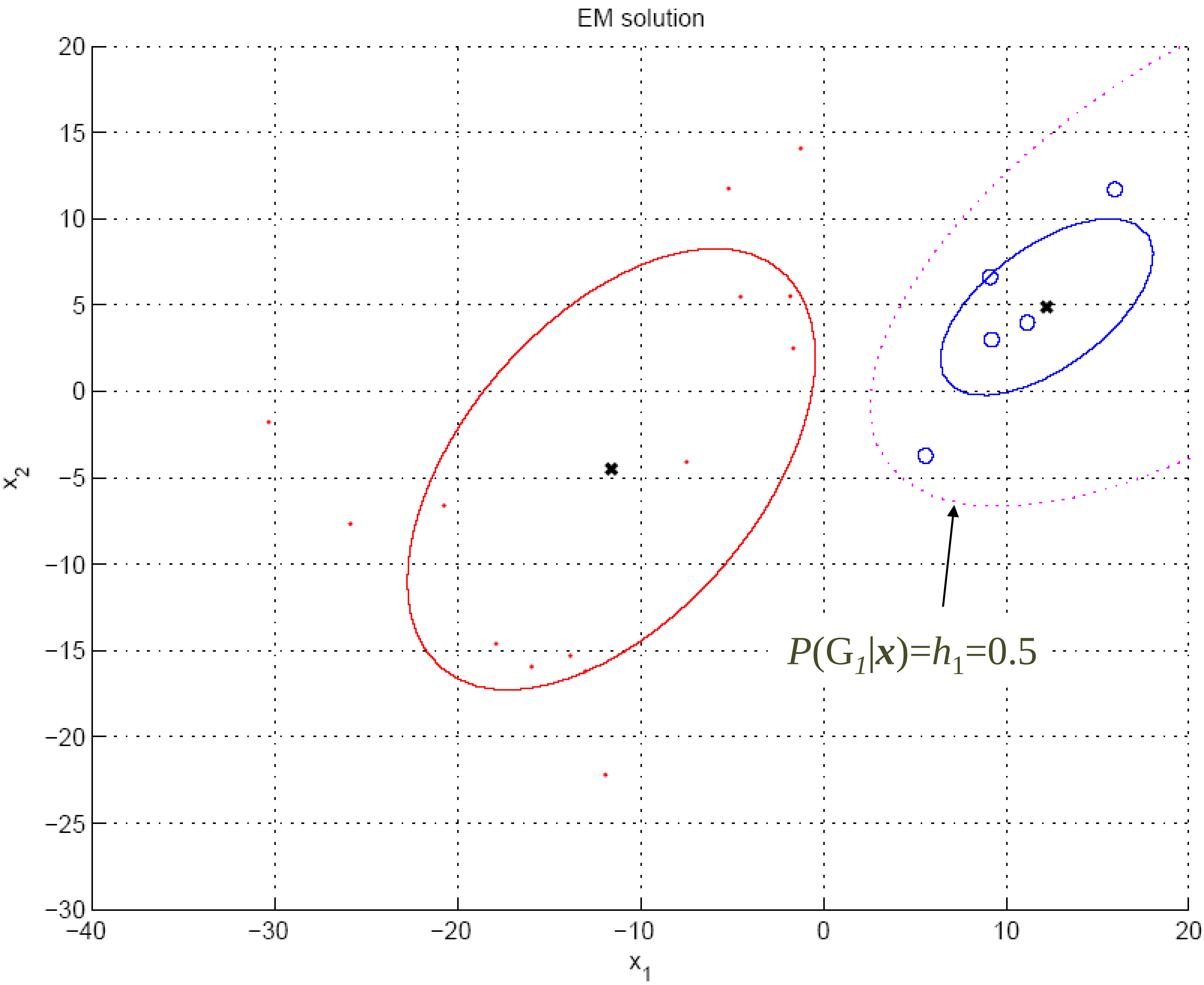
$$h_i^t = \frac{\pi_i |\mathbf{S}_i|^{-1/2} \exp[-(1/2)(\boldsymbol{x}^t - \boldsymbol{m}_i)^T \mathbf{S}_i^{-1} (\boldsymbol{x}^t - \boldsymbol{m}_i)]}{\sum_j \pi_j |\mathbf{S}_j|^{-1/2} \exp[-(1/2)(\boldsymbol{x}^t - \boldsymbol{m}_j)^T \mathbf{S}_j^{-1} (\boldsymbol{x}^t - \boldsymbol{m}_j)]}$$

- M-step: $$P\left(G_i\right) = \frac{\sum_t h_i^t}{N} \qquad \mathbf{m}_i^{l+1} = \frac{\sum_t h_i^t \mathbf{x}^t}{\sum_t h_i^t}$$

$$\mathbf{S}_i^{l+1} = \frac{\sum_t h_i^t \left(\mathbf{x}^t - \mathbf{m}_i^{l+1}\right)\left(\mathbf{x}^t - \mathbf{m}_i^{l+1}\right)^T}{\sum_t h_i^t}$$

Use estimated labels in place of unknown labels

Remarks: h_i^t plays the role of b_i^t in K-Means. h_i^t acts as an estimator for the unknown labels z_i^t.



$P(G_{1}|\boldsymbol{x})=h_1=0.5$

Commonalities between K-Means and EM

1. They start with random clusters and rely on a 2 step-approach to minimize the objective function using the EM-procedure.
2. Use the same optimization procedure of an objective function *$f(a_1,\ldots,a_m,b_1,\ldots,,b_k)$*; we basically, maximize the *a*-values (keeping the *b*-values fixed) and then the *b*-values (keeping the *a*-values fixed) until some convergence is reached. Consequently, both algorithms
   - only find a local minimum of the objective function
   - are sensitive to initialization
3. Both assume the number of clusters *k* is known

Differences between *k*-Means and EM

1. *k*-means is distanced-based and relies on 1-*NN* queries to form clusters. EM is density based/ probabilistic; EM usually works with multivariate Gaussians but can be generalized to work with other probability distributions.
2. *k*-means minimizes the squared distance of on object to its cluster prototype (usually the centroid). EM maximizes the log-likelihood of a sample given a model ($p(X|\boldsymbol{\theta})$); models are assumed to be mixtures of *k* Gaussians and their priors.
3. *k*-means is a hard clustering, EM is a soft clustering algorithm: $h_i^t \in [0,1]$

Differences between K-Means and EM

4. *k*-means cluster models are just *k* centroids; EM models are *k* “priors, means+covariance matrices”.
5. EM directly deals with dependencies between attributes in its density estimation approach: the degree to which an object **x** belongs to a cluster *c* depends on the product of *c*’s prior with the Mahalanobis distance between **x** and the *c*’s mean; therefore, EM clusters do not depend on units of measurements and orientation of attributes in space.
6. The distance metrics can be viewed as an input parameter when using *k*-means, and generalizations of *k*-means have been proposed which use different distance functions. EM implicitly relies on the Mahalanobis distance function which is part of its density estimation approach.

Mixtures of Latent Variable Models

Regularize clusters

1. Assume shared/diagonal covariance matrices
2. Use PCA/FA to decrease dimensionality: Mixtures of PCA/FA

$$p\left(\mathbf{x}_t | G_i\right) = N\left(\mathbf{m}_i, \mathbf{V}_i \mathbf{V}_i^T + \boldsymbol{\psi}_i\right)$$

3. where $\mathbf{V}_i$ and $\boldsymbol{\Psi}_i$ are the factor loadings and specific variances of cluster G_i.
4. Can use EM to learn $\mathbf{V}_i$ and $\boldsymbol{\Psi}_i$ instead of $\mathbf{S}_i$. (Ghahramani and Hinton, 1997; Tipping and Bishop, 1999)

Supervised Learning After Clustering

- Dimensionality reduction methods find correlations between features and group features
- Clustering methods find similarities between instances and group instances
- Allows knowledge extraction through

  number of clusters,

  prior probabilities,

  cluster parameters, i.e., center, range of features.

Example: CRM, customer segmentation

Clustering as Preprocessing

- Estimated group labels h_j (soft) or b_j (hard) may be seen as the dimensions of a new k dimensional space, where we can then learn our discriminant or regressor.
- Local representation (only one b_j is 1, all others are 0; only few h_j are nonzero) vs

  Distributed representation (After PCA; all z_j are nonzero).

Mixture of Mixtures

- In classification, the input comes from a mixture of classes (supervised).
- If each class is also a mixture, e.g., of Gaussians, (unsupervised), we have a mixture of mixtures:

$$p(\mathbf{x}|\mathrm{C}_i) = \sum_{j=1}^{k_i} p(\mathbf{x}|G_{ij}) P(G_{ij})$$

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x}|\mathrm{C}_i) P(\mathrm{C}_i)$$

- where k_i is the number of components making up p($x|C_{\mathrm{i}}$) and G_{ij} is the component j of class i.

Spectral Clustering

- Cluster using predefined pairwise similarities B_{rs} instead of using Euclidean or Mahalanobis distance
- Can be used even if instances not vectorially represented
- Steps:
  1. Use Laplacian Eigenmaps (chapter 6) to map to a new **z** space using B_{rs}
  2. Use *k*-means in this new **z** space for clustering

Hierarchical Clustering

- □ Cluster based on similarities/distances
- □ Distance measure between instances $\boldsymbol{x}^r$ and $\boldsymbol{x}^s$

Minkowski (L_p) (Euclidean for p = 2)

$$d_m\left(\mathbf{x}^r,\mathbf{x}^s\right)=\left[\sum_{j=1}^{d}\left(x_j^r-x_j^s\right)^p\right]^{1/p}$$

City-block distance

$$d_{cb}\left(\mathbf{x}^r,\mathbf{x}^s\right)=\sum_{j=1}^{d}\left|x_j^r-x_j^s\right|$$

Agglomerative Clustering

- Start with N groups each with one instance and merge two closest groups at each iteration
- Distance between two groups G_i and G_j:
  - Single-link:

$$d\left(G_i, G_j\right) = \min_{\mathbf{x}^r \in G_i, \mathbf{x}^s \in G_j} d\left(\mathbf{x}^r, \mathbf{x}^s\right)$$

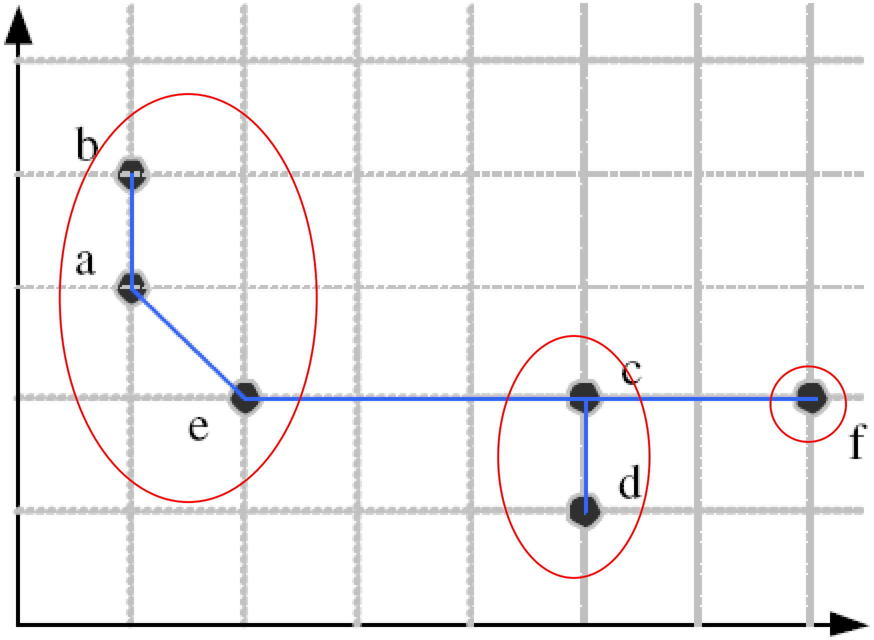
  - Complete-link:

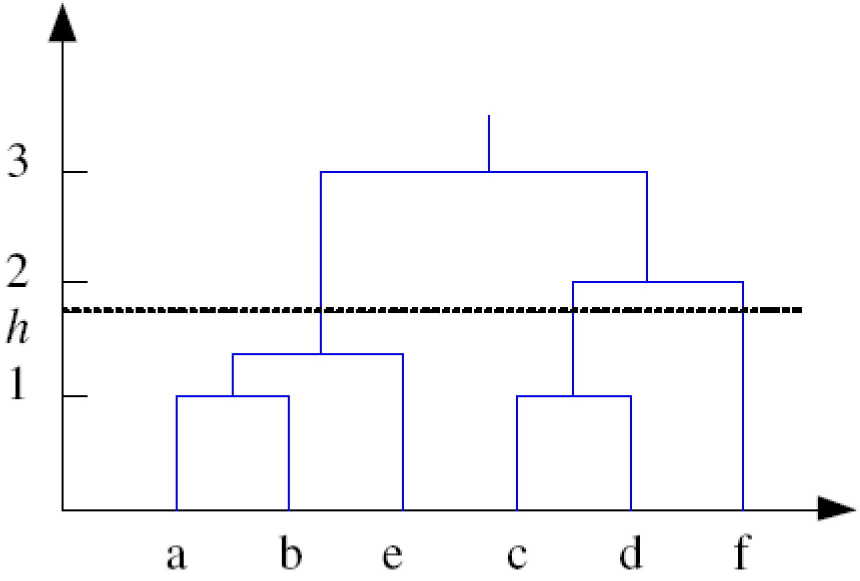
$$d\left(G_i, G_j\right) = \max_{\mathbf{x}^r \in G_i, \mathbf{x}^s \in G_j} d\left(\mathbf{x}^r, \mathbf{x}^s\right)$$

  - Average-link, centroid

$$d\left(G_i, G_j\right) = \operatorname*{ave}_{\mathbf{x}^r \in G_i, \mathbf{x}^s \in G_j} d\left(\mathbf{x}^r, \mathbf{x}^s\right)$$

Example: Single-Link Clustering





Dendrogram

Choosing *k*

- □ Defined by the application, e.g., image quantization
- □ Plot data (after PCA) and check for clusters
- □ Incremental (leader-cluster) algorithm: Add one at a time until “elbow” (reconstruction error/log likelihood/intergroup distances)
- □ Manually check for meaning
- □ Run with multiple *k*-values and compare the results