

## How to evaluate clusters produced by $k$ -means or hierarchical clustering?

Hard to evaluate if we don't already know the correct clusters

### Types of evaluation

- Unsupervised:
  - cluster cohesion (compactness)
  - cluster separation (isolation)
- Supervised
  - How well clusters match externally supplied class labels

## Unsupervised evaluation

$$cohesion(C_i) = \frac{1}{\sum_{\mathbf{x}, \mathbf{y} \in C_i} (distance(\mathbf{x}, \mathbf{y}))}$$

$$separation(C_i, C_j) = \sum_{\mathbf{x} \in C_i, \mathbf{y} \in C_j} (distance(\mathbf{x}, \mathbf{y}))$$

## Alternative cohesion and separation measures

$$\text{cohesion}(C_i) = \sum_{\mathbf{x} \in C_i} \text{distance}(\mathbf{x}, \boldsymbol{\mu}_{C_i})$$

$$\text{SSE}(C_i) = \frac{1}{N} \sum_{\mathbf{x} \in C_i} \text{distance}(\mathbf{x}, \boldsymbol{\mu}_{C_i})^2, \text{ where } N \text{ is number of data points in } C_i$$

$$\text{separation}(C_i, C_j) = \text{distance}(\boldsymbol{\mu}_{C_i}, \boldsymbol{\mu}_{C_j})$$

## Supervised evaluation measures

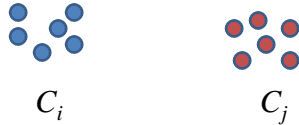
- **Entropy:** Degree to which each cluster consists of objects of a single class.

Let  $p_{ij} = m_{ij}/m_i$ , where  $m_i$  is the number of objects in cluster  $C_i$  and  $m_{ij}$  is the number of objects of class  $j$  in cluster  $C_i$ . Thus  $p_{ij}$  is the probability that a member of cluster  $C_i$  is in class  $j$ .

$$\text{entropy}(C_i) = \sum_{j=1}^L p_{ij} \log_2 p_{ij}$$

**Example:** Assume two clusters,  $C_i$  and  $C_j$ .

What is entropy of  $C_i$  if  $p_{ij} = 0$ ?



What is entropy of  $C_i$  if  $p_{ij} = 1$ ?



What is entropy of  $C_i$  if one half the instances in cluster  $i$  are in class  $j$ ?



- **Precision:** Fraction of a cluster  $C_i$  that consists of objects of a specified class  $j$ .

$$precision(C_i, j) = p_{ij}$$

- **Recall:** Fraction of objects of class  $j$  that are contained in cluster  $C_i$ .

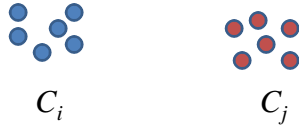
$$recall(C_i, j) = \frac{m_{ij}}{m_j}$$

- **F-measure:** Extent to which a cluster  $C_i$  contains *only* objects of a particular class  $j$  and *all* objects of that class.

$$f(C_i, j) = \frac{2 \times precision(C_i, j) \times recall(C_i, j)}{precision(C_i, j) + recall(C_i, j)}$$

**Example:** Assume two clusters,  $C_i$  and  $C_j$ .

What is  $f(C_i, j)$  if  $p_{ij} = 0$ ?



What is  $f(C_i, j)$  if  $p_{ij} = 1$ ?



What is  $f(C_i, j)$  if one half the instances in cluster  $i$  are in class  $j$ ?



## Discriminative vs. Generative Clustering

- Discriminative:
  - Cluster data points by similarities
  - Example:  $k$ -means and hierarchical clustering
- Generative
  - Generate models to describe different clusters, and use these models to classify points
  - Example: Clustering via finite Gaussian mixture models

## Univariate and Multivariate Gaussian Distribution

- Univariate:

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

- Multivariate ( $D$ -dimensional)

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

where  $\boldsymbol{\mu}$  is a  $D$ -dimensional mean vector,  $\boldsymbol{\Sigma}$  is a  $D \times D$  covariance matrix, and  $|\boldsymbol{\Sigma}|$  is the determinant of  $\boldsymbol{\Sigma}$ .

## Likelihood Functions

- Given normally distributed one-dimensional data points  $\mathcal{X} = \{x_1, \dots, x_m\}$ , likelihood function is:

$$p(\mathcal{X} | \mu, \sigma^2) = \prod_{n=1}^N \mathcal{N}(x_n | \mu, \sigma^2)$$

- Likelihood: probability of data given model (or parameters of model)

- How to estimate parameters  $\mu$  and  $\sigma$  from data?
- Maximize the likelihood function.

$$p(\mathcal{X} | \mu, \sigma^2) = \prod_{n=1}^N \mathcal{N}(x_n | \mu, \sigma^2)$$

- Problem: individual values of  $\mathcal{N}(x_n | \mu, \sigma^2)$  are typically very small. (Can underflow numerical precision of computer.)
- Solution: Work with log likelihood instead of likelihood.

$$\ln p(\mathcal{X} | \mu, \sigma^2) = -\frac{1}{2\sigma^2} \sum_{n=1}^N (x_n - \mu)^2 - \frac{N}{2} \ln \sigma^2 - \frac{N}{2} \ln(2\pi)$$

- Maximize

$$\ln p(\mathcal{X} | \mu, \sigma^2) = -\frac{1}{2\sigma^2} \sum_{n=1}^N (x_n - \mu)^2 - \frac{N}{2} \ln \sigma^2 - \frac{N}{2} \ln(2\pi)$$

with respect to  $\mu$ .

We get

$$\mu_{ML} = \frac{1}{N} \sum_{n=1}^N x_n$$

Now maximize with respect to  $\sigma^2$ :

$$\sigma^2_{ML} = \frac{1}{N} \sum_{n=1}^N (x_n - \mu_{ML})^2$$

## Finite Gaussian Mixture Models (following Chris Bishop's book)

- Clustering by Gaussian mixture models: A “soft” version of  $k$ -means
- **Assumption:** Data is generated from mixture of  $K$  Gaussians. Each cluster will correspond to a single Gaussian. Each point  $\mathbf{x}$  will have some probability distribution over the  $K$  clusters
- **Goal:** Find parameters of these Gaussians such that these probability distributions are highly peaked

- Let  $\mathbf{X}$  be a set of multivariate data points (vectors):  
 $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ .
- General expression for finite Gaussian mixture model:

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

- Let  $\mathbf{z}$  be a  $K$ -dimensional binary vector in which one particular element  $z_k = 1$  iff all other elements = 0, where

$$p(z_k = 1) = \pi_k \text{ (mixing coefficient).}$$

[This will simplify notation and math.]

- We can write

$$p(\mathbf{z}) = \prod_{k=1}^K \pi_k^{z_k} .$$

Since

$$p(\mathbf{x} | z_k = 1) = N(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k),$$

we can also write

$$p(\mathbf{x} | \mathbf{z}) = \prod_{k=1}^K \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k} .$$

Thus, 
$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z}) p(\mathbf{x} | \mathbf{z})$$

$$= \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (\text{from (1)})$$

Define:

$$\gamma(z_k) \equiv p(z_k = 1 | \mathbf{x})$$

This is the quantity we want to determine!  $\gamma(z_k)$  is the degree of confidence that Gaussian  $k$  “explains” data point  $\mathbf{x}$ .

By Bayes rule:

$$\gamma(z_k) = \frac{p(z_k = 1) p(\mathbf{x} | z_k = 1)}{\sum_{j=1}^K p(z_j = 1) p(\mathbf{x} | z_j = 1)}$$

$$= \frac{\pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

## Maximum Likelihood

- Let  $\mathbf{X}$  be an  $N \times D$  matrix whose rows are the components of the data vectors  $\mathbf{x}_j$ . Let  $\mathbf{Z}$  be an  $N \times K$  matrix whose rows are the components of the vectors  $\mathbf{z}_n$

- Likelihood function:

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

- Log likelihood function:

$$\ln p(\mathbf{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^N \ln \left( \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

- Given  $\mathbf{X}$ , we can maximize this function to find  $\{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}_{ML}$

## Maximizing the log-likelihood parameters

Using

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

we can take the derivative of the log likelihood function with respect to  $\boldsymbol{\mu}$ :

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\mu}} (\ln p(\mathbf{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})) &= \frac{\partial}{\partial \boldsymbol{\mu}} \left( \sum_{n=1}^N \ln \left( \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) \right) \\ &= - \sum_{n=1}^N \left( \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \right) \boldsymbol{\Sigma}_k (\mathbf{x}_n - \boldsymbol{\mu}_k) \\ &= - \sum_{n=1}^N \gamma(z_{n,k}) \boldsymbol{\Sigma}_k (\mathbf{x}_n - \boldsymbol{\mu}_k) \end{aligned}$$

- Now we can set this equal to zero,

$$\sum_{n=1}^N \gamma(z_{n,k}) \Sigma_k (\mathbf{x}_n - \boldsymbol{\mu}_k) = 0$$

and multiply both sides by  $\Sigma_k^{-1}$  (assume this inverse exists):

$$\sum_{n=1}^N \gamma(z_{n,k}) (\mathbf{x}_n - \boldsymbol{\mu}_k) = 0$$

$$\sum_{n=1}^N \gamma(z_{n,k}) \mathbf{x}_n - \sum_{n=1}^N \gamma(z_{n,k}) \boldsymbol{\mu}_k = 0$$

$$\boldsymbol{\mu}_k = \frac{1}{\sum_{n=1}^N \gamma(z_{n,k})} \sum_{n=1}^N \gamma(z_{n,k}) \mathbf{x}_n$$

**Interpretation:** The mean for the  $k$ th Gaussian component is the weighted mean of all points in the data set, in which the weighting factor of data point  $\mathbf{x}_n$  is given by the posterior probability ( $\gamma(z_{nk})$ ) that component  $k$  was responsible for generating  $\mathbf{x}_n$ .

**Compare with univariate Gaussian:**

$$\mu_{ML} = \frac{1}{N} \sum_{n=1}^N x_n$$

- Similarly, if we take the derivative of the log likelihood function with respect to  $\Sigma_k$  and set it to zero, we obtain

$$\Sigma_k = \frac{1}{\sum_{n=1}^N \gamma(z_{n,k})} \sum_{n=1}^N \gamma(z_{n,k}) (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^T$$

**Compare with univariate case:**

$$\sigma_{ML}^2 = \frac{1}{N} \sum_{n=1}^N (x_n - \mu_{ML})^2$$

- Finally, if we take the derivative of the log likelihood function with respect to  $\pi_k$  and set it to zero, we obtain (after doing some tricky math stuff):

$$\pi_k = \frac{\sum_{n=1}^N \gamma(z_{n,k})}{N}$$

which is the average “responsibility” that component  $k$  takes for explaining the data points.

However, these do not give closed form solutions for  $\{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}_{ML}$

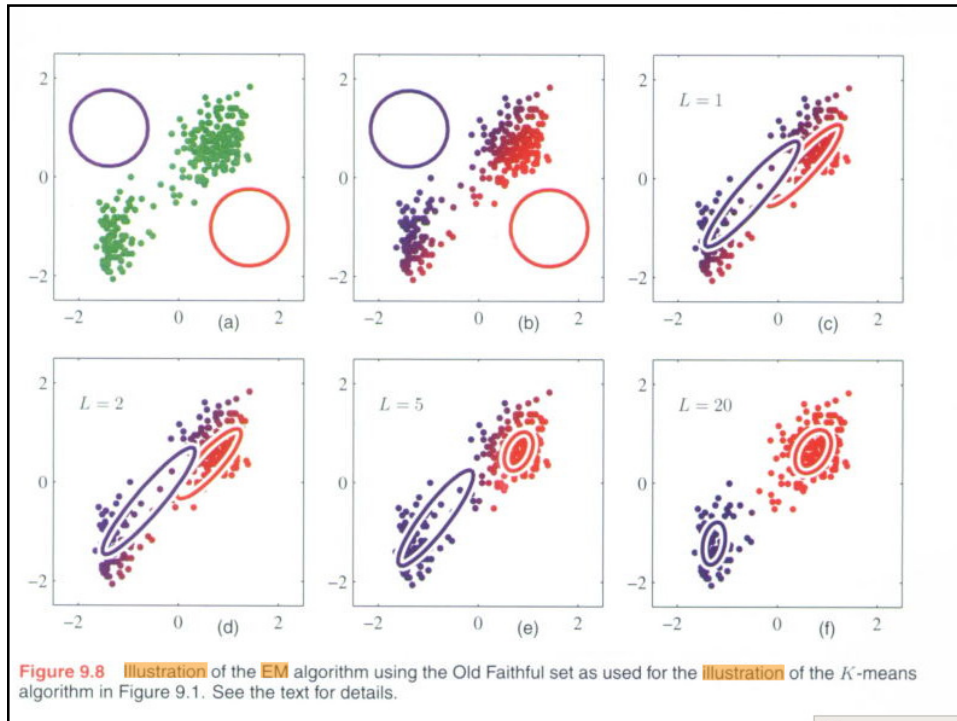
But help motivate the expectation-maximization algorithm for iteratively finding a solution.

## Expectation-Maximization (EM) algorithm

- General idea:
  - Choose random initial values for means (and thus covariances) and mixing coefficients.
  - Alternate between E (expectation) and M (maximization) step:
    - E step: use current values for parameters to evaluate posterior probabilities, or responsibilities, for each data point.
    - M step: Use these probabilities to re-estimate means, covariances, and mixing coefficients.

Repeat until log-likelihood function does not change significantly or the parameters do not change significantly.

- How is this a “soft” version of the  $k$ -means algorithm?



## More detailed version of EM algorithm

1. Initialize the means  $\boldsymbol{\mu}_k$ , covariances  $\boldsymbol{\Sigma}_k$ , and mixing coefficients  $\pi_k$ , and evaluate initial value of log likelihood.

$$\ln p(\mathbf{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^N \ln \left( \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

2. **E step.** Evaluate the responsibilities using the current parameter values

$$\gamma(z_{n,k}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

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

$$\boldsymbol{\mu}_k^{new} = \frac{1}{\underbrace{\sum_{n=1}^N \gamma(z_{n,k})}_{\text{called } N_k \text{ in Bishop book}}} \sum_{n=1}^N \gamma(z_{n,k}) \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{\sum_{n=1}^N \gamma(z_{n,k})} \sum_{n=1}^N \gamma(z_{n,k}) (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T$$

$$\boldsymbol{\pi}_k^{new} = \frac{\sum_{n=1}^N \gamma(z_{n,k})}{N}$$

4. Evaluate the log likelihood with the new parameters

$$\ln p(\mathbf{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^N \ln \left( \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

and check for convergence of either the parameters or the log likelihood. If not converged, return to step 2.

- EM much more computationally expensive than *k-means*
- **Common practice:** Use *k-means* to set initial means, then improve with EM.
  - Initial means: means of clusters found by *k-means*
  - Initial covariances: Sample covariances of the clusters found by *k-means* algorithm.
  - Initial mixture coefficients: fractions of data points assigned to the respective clusters.

- Can prove that EM finds local maxima of log-likelihood function.
- EM is very general technique for finding maximum-likelihood solutions for probabilistic models