

# ColumbiaX: Machine Learning

## Lecture 16

Prof. John Paisley

Department of Electrical Engineering  
& Data Science Institute

Columbia University

# SOFT CLUSTERING VS HARD CLUSTERING MODELS

# HARD CLUSTERING MODELS

## Review: K-means clustering algorithm

---

**Given:** Data  $x_1, \dots, x_n$ , where  $x \in \mathbb{R}^d$

**Goal:** Minimize  $\mathcal{L} = \sum_{i=1}^n \sum_{k=1}^K \mathbb{1}\{c_i = k\} \|x_i - \mu_k\|^2$ .

► Iterate until values no longer changing

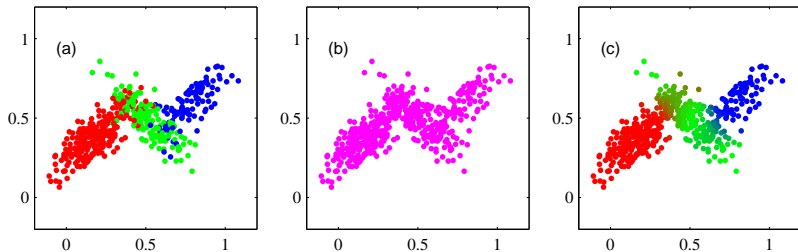
1. Update  $\mathbf{c}$ : For each  $i$ , set  $c_i = \arg \min_k \|x_i - \mu_k\|^2$
  2. Update  $\boldsymbol{\mu}$ : For each  $k$ , set  $\mu_k = (\sum_i x_i \mathbb{1}\{c_i = k\}) / (\sum_i \mathbb{1}\{c_i = k\})$
- 

K-means is an example of a *hard clustering* algorithm because it assigns each observation to only one cluster.

In other words,  $c_i = k$  for some  $k \in \{1, \dots, K\}$ . There is no accounting for the “boundary cases” by hedging on the corresponding  $c_i$ .

# SOFT CLUSTERING MODELS

*A soft clustering algorithm breaks the data across clusters intelligently.*



(left) True cluster assignments of data from three Gaussians.

(middle) The data as we see it.

(right) A soft-clustering of the data accounting for borderline cases.

# WEIGHTED K-MEANS (SOFT CLUSTERING EXAMPLE)

---

## Weighted K-means clustering algorithm

---

**Given:** Data  $x_1, \dots, x_n$ , where  $x \in \mathbb{R}^d$

**Goal:** Minimize  $\mathcal{L} = \sum_{i=1}^n \sum_{k=1}^K \phi_i(k) \frac{\|x_i - \mu_k\|^2}{\beta}$  over  $\phi_i$  and  $\mu_k$

**Conditions:**  $\phi_i(k) > 0$  and  $\sum_{k=1}^K \phi_i(k) = 1$ . Set parameter  $\beta > 0$ .

► Iterate the following

1. Update  $\phi$ : For each  $i$ , update the word allocation weights

$$\phi_i(k) = \frac{\exp\{-\frac{1}{\beta}\|x_i - \mu_k\|^2\}}{\sum_j \exp\{-\frac{1}{\beta}\|x_i - \mu_j\|^2\}}, \text{ for } k = 1, \dots, K$$

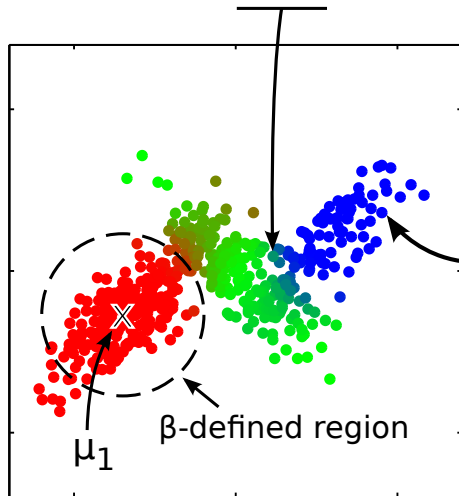
2. Update  $\mu$ : For each  $k$ , update  $\mu_k$  with the *weighted* average

$$\mu_k = \frac{\sum_i x_i \phi_i(k)}{\sum_i \phi_i(k)}$$

---

# SOFT CLUSTERING WITH WEIGHTED K-MEANS

$\phi_i = 0.75$  on green cluster &  $0.25$  blue cluster



$\phi_i = 1$  on blue cluster

When  $\phi_i$  is binary,  
we get back the hard  
clustering model

# MIXTURE MODELS

# PROBABILISTIC SOFT CLUSTERING MODELS

## Probabilistic vs non-probabilistic soft clustering

The weight vector  $\phi_i$  is *like* a probability of  $x_i$  being assigned to each cluster.

A **mixture model** is a probabilistic model where  $\phi_i$  actually *is* a probability distribution according to the model.

Mixture models work by defining:

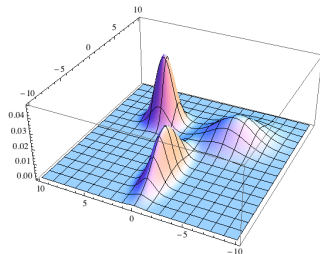
- ▶ A prior distribution on the cluster assignment indicator  $c_i$
- ▶ A likelihood distribution on observation  $x_i$  given the assignment  $c_i$

Intuitively we can connect a mixture model to the Bayes classifier:

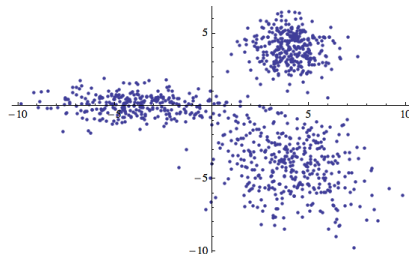
- ▶ Class prior  $\rightarrow$  cluster prior. This time, we *don't* know the “label”
- ▶ Class-conditional likelihood  $\rightarrow$  cluster-conditional likelihood



# MIXTURE MODELS



(a) A probability distribution on  $\mathbb{R}^2$ .



(b) Data sampled from this distribution.

Before introducing math, some key features of a mixture model are:

1. It is a generative model (defines a probability distribution on the data)
2. It is a weighted combination of simpler distributions.
  - ▶ Each simple distribution is in the same distribution family (i.e., a Gaussian).
  - ▶ The “weighting” is defined by a discrete probability distribution.

# MIXTURE MODELS

## Generating data from a mixture model

**Data:**  $x_1, \dots, x_n$ , where each  $x_i \in \mathcal{X}$  (can be complicated, but think  $\mathcal{X} = \mathbb{R}^d$ )

**Model parameters:** A  $K$ -dim distribution  $\pi$  and parameters  $\theta_1, \dots, \theta_K$ .

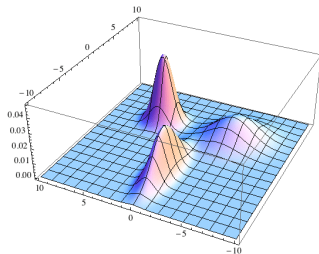
**Generative process:** For observation number  $i = 1, \dots, n$ ,

1. Generate cluster assignment:  $c_i \stackrel{iid}{\sim} \text{Discrete}(\pi) \Rightarrow \text{Prob}(c_i = k | \pi) = \pi_k$ .
2. Generate observation:  $x_i \sim p(x | \theta_{c_i})$ .

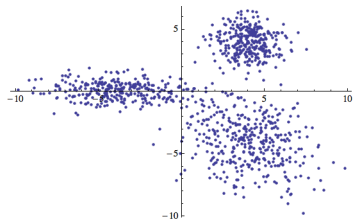
Some observations about this procedure:

- ▶ First, each  $x_i$  is randomly assigned to a cluster using distribution  $\pi$ .
- ▶  $c_i$  indexes the cluster assignment for  $x_i$ 
  - ▶ This picks out the index of the parameter  $\theta$  used to generate  $x_i$ .
  - ▶ If two  $x$ 's share a parameter, they are clustered together.

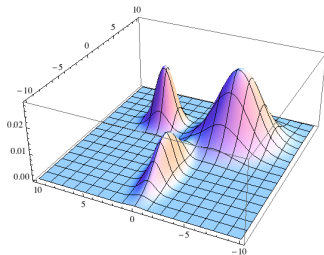
# MIXTURE MODELS



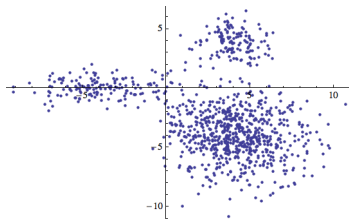
(a) Uniform mixing weights



(b) Data sampled from this distribution.



(c) Uneven mixing weights



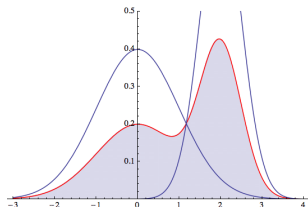
(d) Data sampled from this distribution.

# GAUSSIAN MIXTURE MODELS

# ILLUSTRATION

*Gaussian mixture models are mixture models where  $p(x|\theta)$  is Gaussian.*

## Mixture of two Gaussians



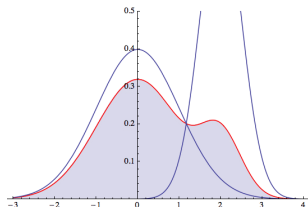
The red line is the density function.

$$\pi = [0.5, 0.5]$$

$$(\mu_1, \sigma_1^2) = (0, 1)$$

$$(\mu_2, \sigma_2^2) = (2, 0.5)$$

## Influence of mixing weights



The red line is the density function.

$$\pi = [0.8, 0.2]$$

$$(\mu_1, \sigma_1^2) = (0, 1)$$

$$(\mu_2, \sigma_2^2) = (2, 0.5)$$

# GAUSSIAN MIXTURE MODELS (GMM)

## The model

**Parameters:** Let  $\pi$  be a  $K$ -dimensional probability distribution and  $(\mu_k, \Sigma_k)$  be the mean and covariance of the  $k$ th Gaussian in  $\mathbb{R}^d$ .

**Generate data:** For the  $i$ th observation,

1. Assign the  $i$ th observation to a cluster,  $c_i \sim \text{Discrete}(\pi)$
2. Generate the value of the observation,  $x_i \sim N(\mu_{c_i}, \Sigma_{c_i})$

**Definitions:**  $\mu = \{\mu_1, \dots, \mu_K\}$  and  $\Sigma = \{\Sigma_1, \dots, \Sigma_K\}$ .

**Goal:** We want to learn  $\pi$ ,  $\mu$  and  $\Sigma$ .

# GAUSSIAN MIXTURE MODELS (GMM)

## Maximum likelihood

Objective: Maximize the likelihood over model parameters  $\pi$ ,  $\boldsymbol{\mu}$  and  $\boldsymbol{\Sigma}$  by treating the  $c_i$  as auxiliary data using the EM algorithm.

$$p(x_1, \dots, x_n | \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^n p(x_i | \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^n \sum_{k=1}^K p(x_i, c_i = k | \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

The summation over values of each  $c_i$  “integrates out” this variable.

We can't simply take derivatives with respect to  $\pi$ ,  $\mu_k$  and  $\Sigma_k$  and set to zero to maximize this because there's no closed form solution.

We could use gradient methods, but EM is cleaner.

# EM ALGORITHM

**Q:** Why not instead just include each  $c_i$  and maximize  $\prod_{i=1}^n p(x_i, c_i | \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$  since (we can show) this is easy to do using coordinate ascent?

**A:** We would end up with a hard-clustering model where  $c_i \in \{1, \dots, K\}$ .  
Our goal here is to have soft clustering, which the sum effectively does.

## EM and the GMM

We will not derive everything from scratch. However, we can treat  $c_1, \dots, c_n$  as the auxiliary data that we integrate out.

Therefore, we use EM to

$$\text{maximize } \sum_{i=1}^n \ln p(x_i | \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad \text{by using} \quad \sum_{i=1}^n \ln p(x_i, c_i | \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Let's look at the outlines of how to derive this.



# THE EM ALGORITHM AND THE GMM

From the last lecture, the generic EM objective is

$$\ln p(x|\theta_1) = \int q(\theta_2) \ln \frac{p(x, \theta_2|\theta_1)}{q(\theta_2)} d\theta_2 + \int q(\theta_2) \ln \frac{q(\theta_2)}{p(\theta_2|x, \theta_1)} d\theta_2$$

The EM objective for the Gaussian mixture model is

$$\begin{aligned} \sum_{i=1}^n \ln p(x_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \sum_{i=1}^n \sum_{k=1}^K q(c_i = k) \ln \frac{p(x_i, c_i = k|\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})}{q(c_i = k)} + \\ &\quad \sum_{i=1}^n \sum_{k=1}^K q(c_i = k) \ln \frac{q(c_i = k)}{p(c_i|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})} \end{aligned}$$

Because  $c_i$  is discrete, the integral becomes a sum.

# EM SETUP (ONE ITERATION)

**First:** Set  $q(c_i = k) \leftarrow p(c_i = k|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$  using Bayes rule:

$$p(c_i = k|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \propto p(c_i = k|\pi)p(x_i|c_i = k, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

We can solve the posterior of  $c_i$  given  $\pi, \boldsymbol{\mu}$  and  $\boldsymbol{\Sigma}$ :

$$q(c_i = k) = \frac{\pi_k N(x_i|\mu_k, \Sigma_k)}{\sum_j \pi_j N(x_i|\mu_j, \Sigma_j)} \implies \phi_i(k)$$

**E-step:** Take the expectation using the updated  $q$ 's

$$Q = \sum_{i=1}^n \sum_{k=1}^K \phi_i(k) \ln p(x_i, c_i = k|\pi, \mu_k, \Sigma_k) + \text{constant w.r.t. } \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}$$

**M-step:** Maximize  $Q$  with respect to  $\pi$  and each  $\mu_k, \Sigma_k$ .

# M-STEP CLOSE UP

**Aside:** How has EM made this easier?

Original objective function:

$$\mathcal{L} = \sum_{i=1}^n \ln \sum_{k=1}^K p(x_i, c_i = k | \pi, \mu_k, \Sigma_k) = \sum_{i=1}^n \ln \sum_{k=1}^K \pi_k N(x_i | \mu_k, \Sigma_k).$$

The log-sum form makes optimizing  $\pi$ , and each  $\mu_k$  and  $\Sigma_k$  difficult.

Using EM here, we have the M-Step:

$$Q = \sum_{i=1}^n \sum_{k=1}^K \phi_i(k) \underbrace{\{\ln \pi_k + \ln N(x_i | \mu_k, \Sigma_k)\}}_{\ln p(x_i, c_i=k | \pi, \mu_k, \Sigma_k)} + \text{constant w.r.t. } \pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}$$

The sum-log form is easier to optimize. We can take derivatives and solve.

# EM FOR THE GMM

## Algorithm: Maximum likelihood EM for the GMM

---

**Given:**  $x_1, \dots, x_n$  where  $x \in \mathbb{R}^d$

**Goal:** Maximize  $\mathcal{L} = \sum_{i=1}^n \ln p(x_i | \pi, \mu, \Sigma)$ .

► Iterate until incremental improvement to  $\mathcal{L}$  is “small”

1. **E-step:** For  $i = 1, \dots, n$ , set

$$\phi_i(k) = \frac{\pi_k N(x_i | \mu_k, \Sigma_k)}{\sum_j \pi_j N(x_i | \mu_j, \Sigma_j)}, \quad \text{for } k = 1, \dots, K$$

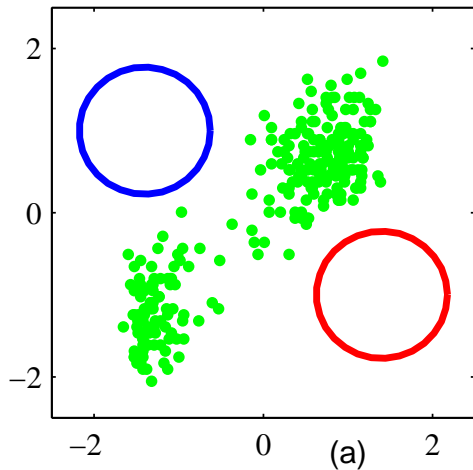
2. **M-step:** For  $k = 1, \dots, K$ , define  $n_k = \sum_{i=1}^n \phi_i(k)$  and update the values

$$\pi_k = \frac{n_k}{n}, \quad \mu_k = \frac{1}{n_k} \sum_{i=1}^n \phi_i(k) x_i \quad \Sigma_k = \frac{1}{n_k} \sum_{i=1}^n \phi_i(k) (x_i - \mu_k)(x_i - \mu_k)^T$$

Comment: The updated value for  $\mu_k$  is used when updating  $\Sigma_k$ .

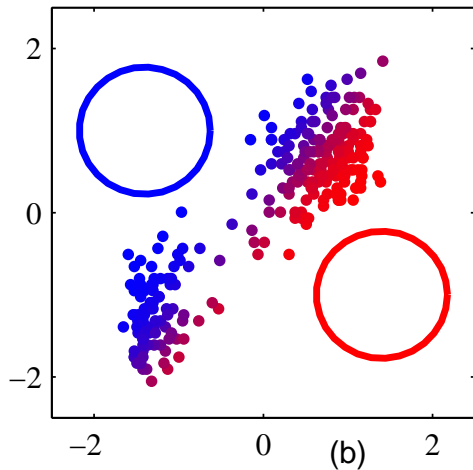
---

# GAUSSIAN MIXTURE MODEL: EXAMPLE RUN



A random initialization

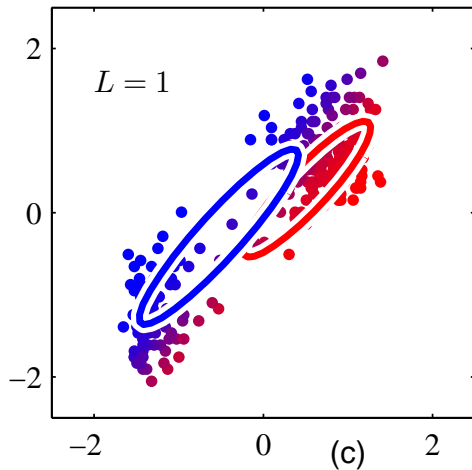
# GAUSSIAN MIXTURE MODEL: EXAMPLE RUN



**Iteration 1 (E-step)**

Assign data to clusters

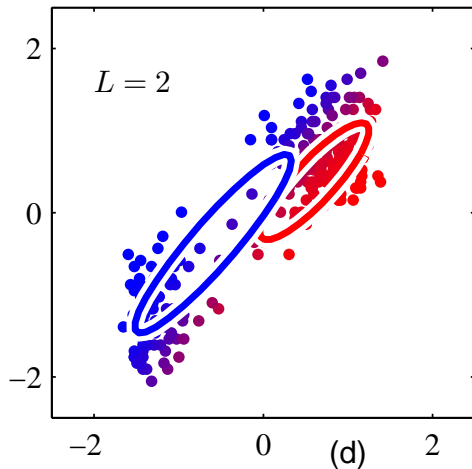
# GAUSSIAN MIXTURE MODEL: EXAMPLE RUN



**Iteration 1 (M-step)**

Update the Gaussians

# GAUSSIAN MIXTURE MODEL: EXAMPLE RUN

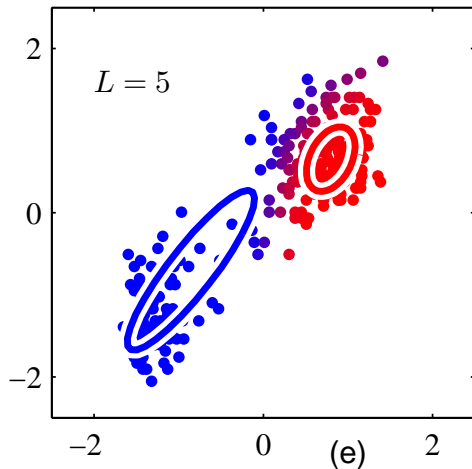


## Iteration 2

Assign data to clusters  
and update the Gaussians



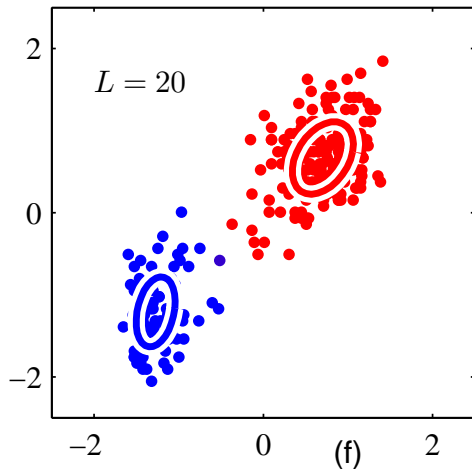
# GAUSSIAN MIXTURE MODEL: EXAMPLE RUN



**Iteration 5 (skipping ahead)**

Assign data to clusters  
and update the Gaussians

# GAUSSIAN MIXTURE MODEL: EXAMPLE RUN



**Iteration 20 (convergence)**

Assign data to clusters  
and update the Gaussians

# GMM AND THE BAYES CLASSIFIER

The GMM feels a lot like a  $K$ -class Bayes classifier, where the label of  $x_i$  is

$$\text{label}(x_i) = \arg \max_k \pi_k N(x_i | \mu_k, \Sigma_k).$$

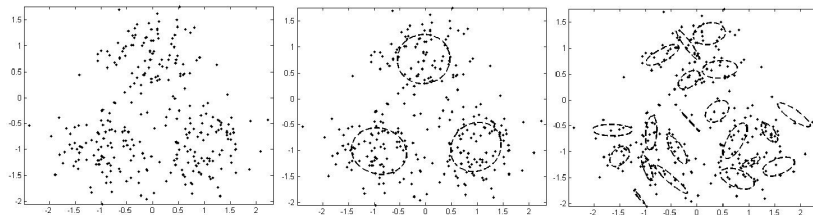
- ▶  $\pi_k$  = class prior, and  $N(\mu_k, \Sigma_k)$  = class-conditional density function.
- ▶ We learned  $\pi$ ,  $\mu$  and  $\Sigma$  using maximum likelihood here too.

For the Bayes classifier, we could find  $\pi$ ,  $\mu$  and  $\Sigma$  with a single equation because the class label was *known*. Compare with the GMM update:

$$\pi_k = \frac{n_k}{n}, \quad \mu_k = \frac{1}{n_k} \sum_{i=1}^n \phi_i(k) x_i \quad \Sigma_k = \frac{1}{n_k} \sum_{i=1}^n \phi_i(k) (x_i - \mu_k)(x_i - \mu_k)^T$$

They're almost identical. But since  $\phi_i(k)$  is changing we have to update these values. With the Bayes classifier, " $\phi_i$ " encodes the label, so it was known.

# CHOOSING THE NUMBER OF CLUSTERS



Maximum likelihood for the Gaussian mixture model can overfit the data. It will learn as many Gaussians as it's given.

There are a set of techniques for this based on the Dirichlet distribution.

A Dirichlet prior is used on  $\pi$  which encourages many Gaussians to disappear (i.e., not have any data assigned to them).

# EM FOR A GENERIC MIXTURE MODEL

## Algorithm: Maximum likelihood EM for mixture models

---

**Given:** Data  $x_1, \dots, x_n$  where  $x \in \mathcal{X}$

**Goal:** Maximize  $\mathcal{L} = \sum_{i=1}^n \ln p(x_i | \pi, \theta)$ , where  $p(x | \theta_k)$  is problem-specific.

► Iterate until incremental improvement to  $\mathcal{L}$  is “small”

1. **E-step:** For  $i = 1, \dots, n$ , set

$$\phi_i(k) = \frac{\pi_k p(x_i | \theta_k)}{\sum_j \pi_j p(x_i | \theta_j)}, \quad \text{for } k = 1, \dots, K$$

2. **M-step:** For  $k = 1, \dots, K$ , define  $n_k = \sum_{i=1}^n \phi_i(k)$  and set

$$\pi_k = \frac{n_k}{n}, \quad \theta_k = \arg \max_{\theta} \sum_{i=1}^n \phi_i(k) \ln p(x_i | \theta)$$

**Comment:** Similar to generalization of the Bayes classifier for any  $p(x | \theta_k)$ .

---