Gaussian Mixture Models

Chiara Cangelosi, Gianluca Notarangelo

January 31, 2025

Contents

Introduction

Density estimation is the problem of reconstructing the probability density function from a given sample. It is a technique applied in different contexts, like clustering (unsupervised learning), feature engineering, and data modeling. Many methods have been developed with this aim in mind, like Histogram and Kernel Density estimation or Gaussian Mixture Models, on which we will focus.

Gaussian Mixture Models

Introduction

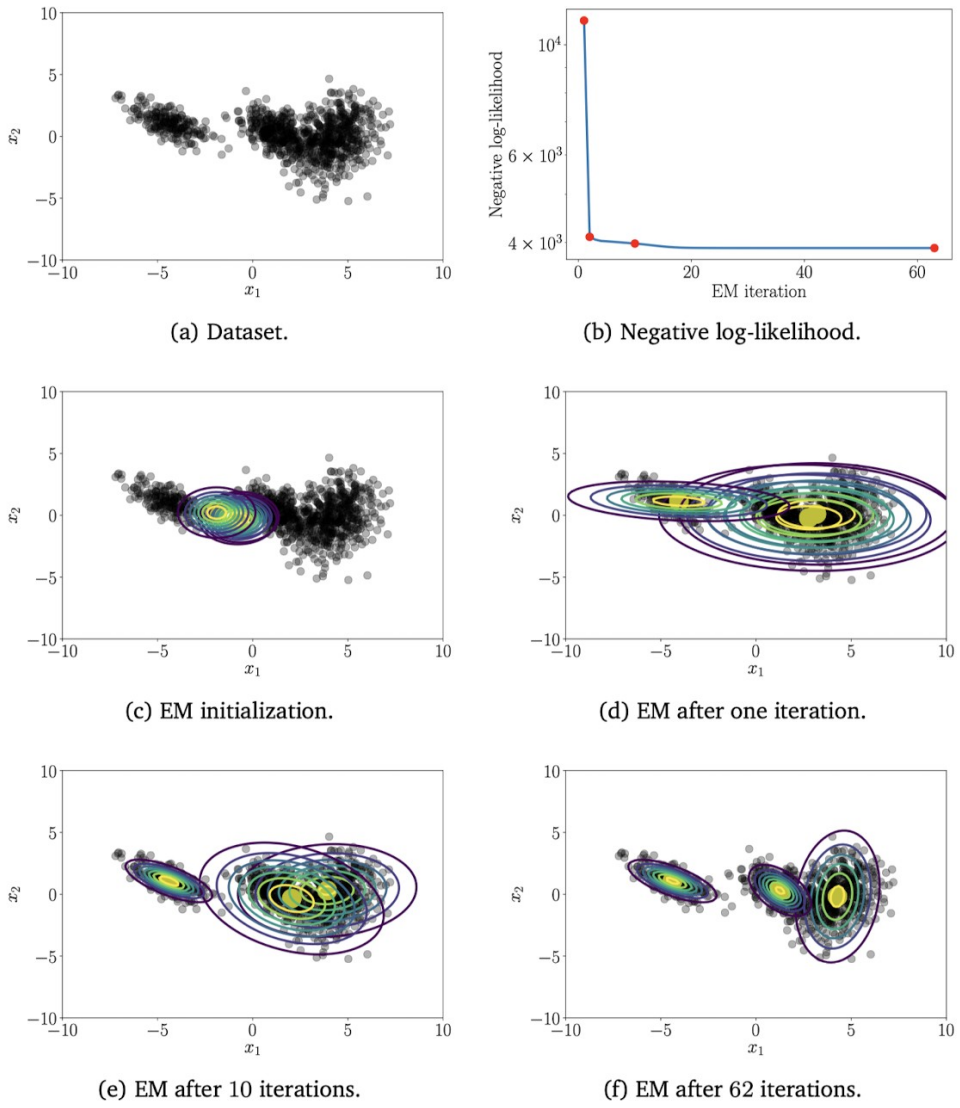
One of the many goals of machine learning is to represent data. Suppose you have a data set X, how can this data be represented? The obvious way is to just use the whole data set as is. However this isn't always an efficient way, for example when the data set is too large it may be difficult to manipulate the data or do calculations over them. A useful technique is to use the underlying data distribution instead of the data themselves. In general, the real underlying distribution is unknown, therefore we need to come up with a model of the data that works well enough. Single distributions such as the Gaussian or Beta distribution have limited modeling capabilities. A more expressive family of distributions can be obtained by combining more than one distribution of the same kind into what is knows as a "mixture model". Using normal distributions, we therefore obtain a Gaussian Mixture Model.

Let's consider the two-dimensional data set plotted in 1.a. We could try to model it with a single gaussian distribution, but it is evident that the data are not normally distributed and more than 1 center characterizes the data set. A mixture of gaussians however, that is a linear combination of gaussian distributions each with it's own coefficient π_k, is much more flexible and can offer a more precise approximation of the real distribution.

$$p(x) = \sum_{i=1}^{K} \pi_k p_k(x) \quad \rightarrow \quad p(x|\theta) = \sum_{i=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{1}$$

$$0 \leq \pi_k \leq 1, \quad \sum_{i=1}^{K} \pi_k = 1 \tag{2}$$

The expectation-maximization algorithm (EM) is an approach for performing maximum likelihood estimation of the coefficients of the gaussians and the of the linear combination coefficients. Through this iterative algorithm, as shown in figures 1.c, 1.d, 1.e and 1.f, one can refine the parameters of the model until is fits the data satisfactorily. Once the process is completed, we are left with a mixture model of the data set than can be used to identify clusters, classify new data or just understand better the dynamic of the process being studied.



(a) Dataset. (b) Negative log-likelihood.

(c) EM initialization. (d) EM after one iteration.

(e) EM after 10 iterations. (f) EM after 62 iterations.

Figure 1: EM algorithm for fitting a GMM to a 2D dataset

Model fitting on one-dimensional data

A Gaussian mixture model is a combination of a finite number of K Gaussian distributions $\mathcal{N}(x \mid \mu_k, \Sigma_k)$:

$$p(x \mid \theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k) \tag{3}$$

$$0 \leq \pi_k \leq 1, \quad \sum_{k=1}^{K} \pi_k = 1 \tag{4}$$

in which $\boldsymbol{\theta} = \{\mu_k, \Sigma_k, \pi_k : k = 1, ..., K\}$ is the collection of all the parameters of the model, where π_k are the mixture weights.

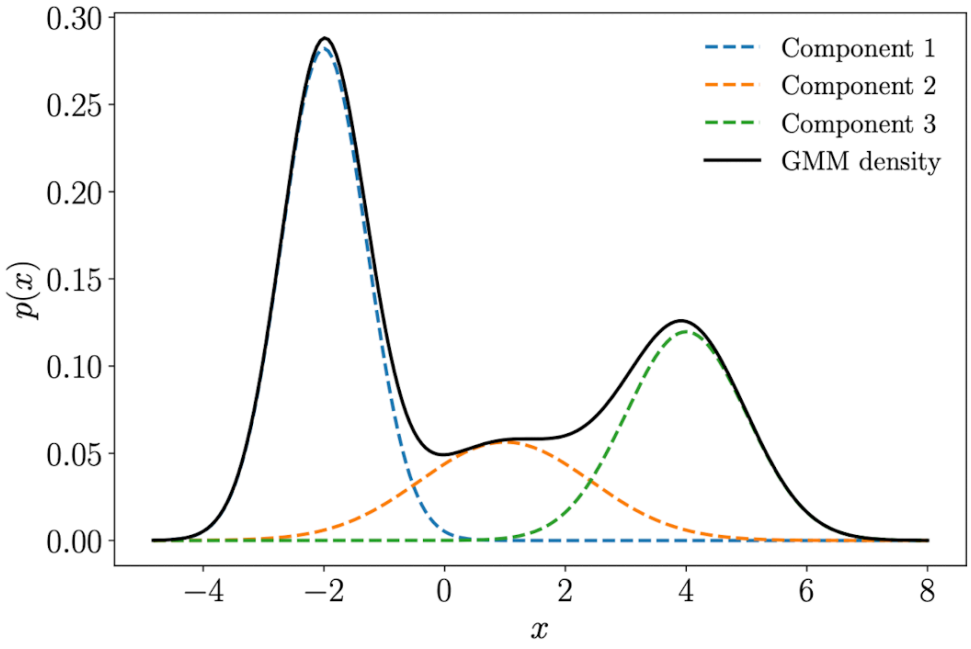


Figure 2: The Gaussian mixture distribution (black) is made up of a combination of Gaussian distributions

In Figure 2 we see an example of how three distinct gaussians combine to form a gaussian mixture. The figure corresponds to the following linear combination:

$$p(x \mid \boldsymbol{\theta}) = 0.5\mathcal{N}\left(x \mid -2, \frac{1}{2}\right) + 0.2\mathcal{N}(x \mid 1, 2) + 0.3\mathcal{N}(x \mid 4, 1)$$

Let us now delve into how this kind of model is actually adapted to the data at hand through the EM algorithm. To do so, we will follow the whole procedure on a simple 1D data set, so that the formulas

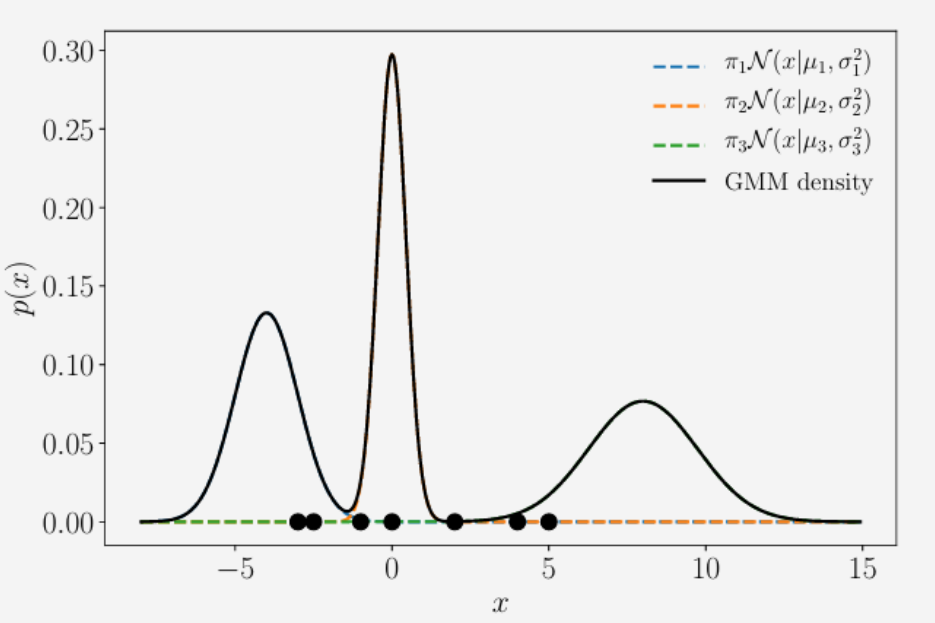


Figure 3: Initialized GMM for our χ data set

are as simple as possible. We assume that the data points are independent and identically distributed from an underlying unknown distribution and are given by $\chi = \{-3, -2.5, -1, 0, 2, 4, 5\}$.

We need to decide how to initialize our mixture, that is we need to choose K (the number of distributions) and the many coefficients. This choice has an influence on how fast the convergence will be, but this is a simple dataset and the parameters have been chosen manually without much care. Our mixture is made up of 3 components and the weights of the combination have been chosen to be equal, that is $\pi_k = \frac{1}{3}, k = 1, 2, 3$. The means and standard deviations of the gaussians are as follows:

$$p_1(x) = \mathcal{N}(x \mid -4, 1)$$

$$p_2(x) = \mathcal{N}(x \mid 0, 0.2)$$

$$p_3(x) = \mathcal{N}(x \mid 8, 3)$$

In Figure 3 we can see the points on the line and our initial Gaussian mixture. The three peaks are very distinct and not really coherent with the actual distribution of the data. This is where the EM algorithm comes in our help. We will start by showing a practical application of this optimization

method and than offer a more general definition.

As we said, the EM algorithm is based on maximum likelihood maximization, so we need to introduce our likelihood function. Since the points are assumed to be independent, this is just the product of the probabilities of each point given the set of parameters $\boldsymbol{\theta}$. The probability of obtaining each point is given by the mixture valued at that specific point, so as to obtain:

$$p(\mathcal{X} \mid \boldsymbol{\theta}) = \prod_{n=1}^{N} p\left(\boldsymbol{x}_n \mid \boldsymbol{\theta}\right), \quad p\left(\boldsymbol{x}_n \mid \boldsymbol{\theta}\right) = \sum_{k=1}^{K} \pi_k \mathcal{N}\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right),$$

The log-likelihood will then be:

$$\log p(\mathcal{X} \mid \boldsymbol{\theta}) = \sum_{n=1}^{N} \log p\left(\boldsymbol{x}_n \mid \boldsymbol{\theta}\right) = \underbrace{\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)}_{=:\mathcal{L}}.$$

To maximize this function, one could think of setting it's gradient to zero. However, this procedure produces an equation that cannot be solved analytically but only numerically. The EM algorithm offers an alternative in the form of an iterative process which will, hopefully, converge to a good enough choice of parameters. This could be only a local maxima but repeating the steps with various initializations should, in most cases, offer solutions that are close to the global optimum.

The main idea is to update each set of parameters, the means, the covariances and the weights, one at a time while keeping the others blocked. At the end we re-evaluate the model with the new parameters. If the precision we would like to obtain has been attained, than we can stop, otherwise we repeat the steps again.

Responsabilities

The quantity r_{nk}

$$r_{nk} := \frac{\pi_k \mathcal{N}\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)}{\sum_{j=1}^{K} \pi_j \mathcal{N}\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)} \tag{5}$$

represents the responsibility of the kth mixture component for the nth data point. It is propor-

tional to the likelihood of the mixture component, as a consequence mixture components have a higher responsibility if a data point might be a possible sample from that mixture component.

We compute the responsibilities r_{nk} of our example:

$$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 1.0 & 0.0 & 0.0 \\ 0.057 & 0.943 & 0.0 \\ 0.001 & 0.999 & 0.0 \\ 0.0 & 0.066 & 0.934 \\ 0.0 & 0.0 & 1.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix} \in R^{N \times K}$$

Each row of the matrix represents the responsibilities of all mixture components for a specific data point, with the row summing to 1. The columns indicate the responsibilities of each mixture component across all data points. For instance, the third component (column) has no responsibility for the first four points, but dominates the remaining ones. The sum of each column gives N_k, which represents the total responsibility of the k-th component: $N_1 = 2.058$, $N_2 = 2.008$, $N_3 = 2.934$.

We are going to calculate the parameters $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, $\boldsymbol{\pi}_k$ (respectively mean, covariance, and weight of each component in the Gaussian Mixture Model (GMM)) iteratively because their updates depend on the responsibilities $\boldsymbol{r}_{nk}$, which in turn rely on these parameters.

Updating the Means

The update of the mean parameters $\mu_k, k = 1, ..., K$, of the GMM is given by

$$\boldsymbol{\mu}_k^{\text{new}} = \frac{\sum_{n=1}^{N} r_{nk} \boldsymbol{x}_n}{\sum_{n=1}^{N} r_{nk}} \tag{6}$$

Since the update of the means μ_k for each mixture component relies on the current values of all means, covariance matrices Σ_k and mixture weights ω_k through the responsibilities r_{nk}, it is not

possible to derive a direct closed-form solution for all μ_k simultaneously.

Since r_{nk} represents the responsibility or the probability that the k-th mixture component generated x_n , the mean μ_k is "pulled" towards the data points for which the correspondent component has more responsability.

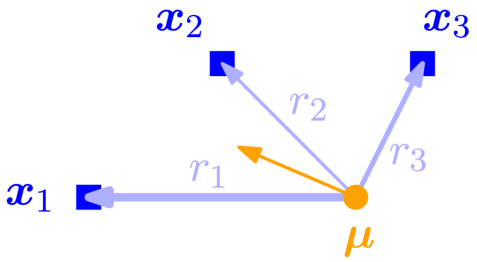


Figure 4: Update of the mean parameter of mixture component in a GMM

The updated mean can also be interpreted as the expected value of all data points in the distribution given by $r_k = [r_{1k}, ..., r_{nk}]/N_k$, which is a normalized vector and therefore interpretable as a probability.

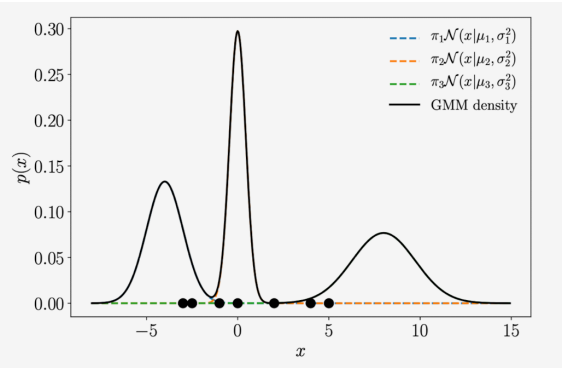


Figure 5: GMM density and individual components prior to updating the mean values

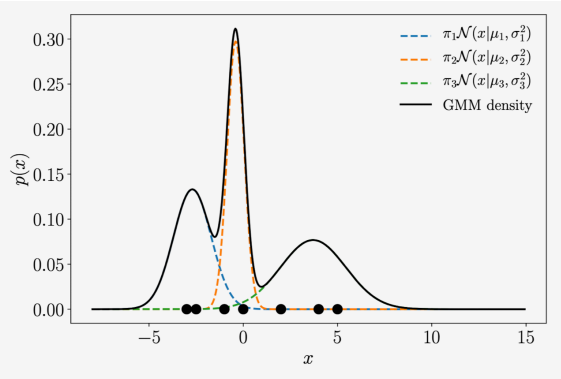


Figure 6: GMM density and individual components after updating the mean values

In the example illiustrated in 5 and 6, the mean values have been updated:

$$\mu_1 : -4 \rightarrow 2.7$$

$$\mu_2 : 0 \rightarrow -0.4$$

$$\mu_3 : 8 \rightarrow 3.7$$

The means of the first and third mixture components adjust significantly to align more closely with the data distribution, while the mean of the second component changes less noticeably.

Updating the Covariances

The update of the covariance parameters $\Sigma_k, k = 1, ..., K$ of the GMM is given by

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} \left(\boldsymbol{x}_n - \boldsymbol{\mu}_k\right) \left(\boldsymbol{x}_n - \boldsymbol{\mu}_k\right)^\top \tag{7}$$

Similarly to the update of $_k$, we can view the update of the covariance in the equation as the expected value weighted by the importance of the squared difference between the data points and their center $\tilde{X}_k := \{x_1 \to \mu_k, \ldots, x_N \to \mu_k\}$.

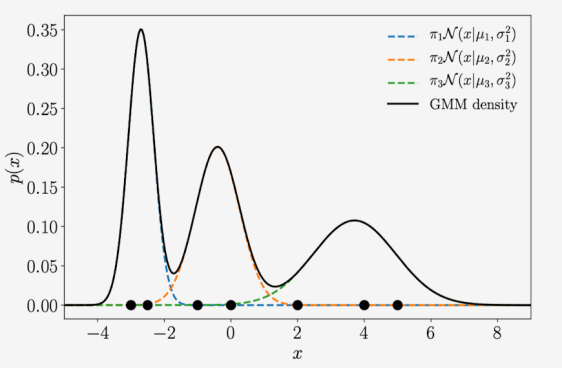


Figure 7: GMM density and individual components prior to updating the variances.

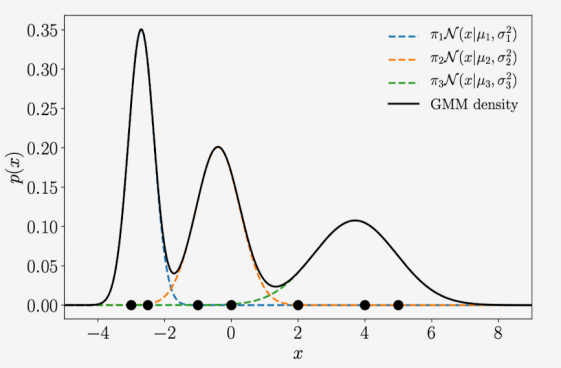


Figure 8: GMM density and individual components after updating the variances.

In the example illiustrated in 7 and 8, the variance values have been updated:

$$\sigma_1^2 : 1 \to 0.14$$

$$\sigma_2^2 : 0.2 \to 0.44$$

$$\sigma_3^2 : 3 \to 1.53$$

In this case, we observe that the variances of the first and third components decrease notably, while the variance of the second component increases slightly.

Updating the Mixture Weights

The mixture weights of the GMM are updated as

$$\boldsymbol{\pi}_k^{\text{new}} = \frac{N_k}{N}, k = 1, ..., K, \tag{8}$$

where N is the number of data points .

We can interpret the mixture weight in equation (11.42) as the ratio between the total responsibility of the kth cluster and the total number of data points. Since $N_k = \sum_{i=1}^{N} r_{nk}$, the total number of data points can also be seen as the sum of the responsibilities of all the mixture components, meaning that ω_k represents the relative importance of the kth mixture component for the dataset.

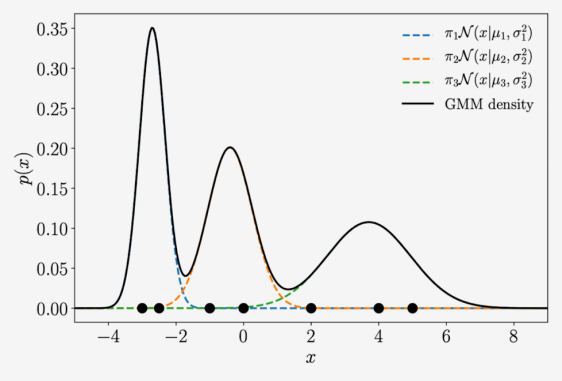


Figure 9: GMM density and individual components prior to updating the mixture weights.

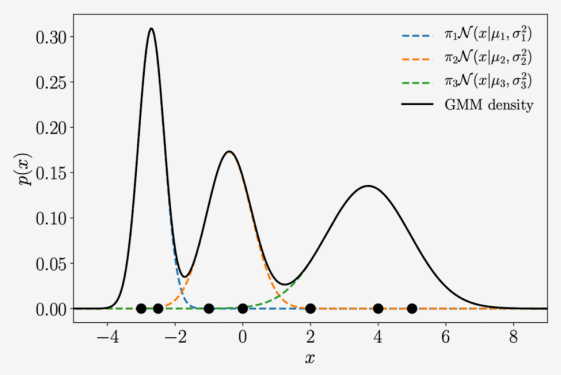


Figure 10: GMM density and individual components after updating the mixture weights.

The mixture weights are updated as it follows:

$$\pi_1 : 1/3 \rightarrow 0.29$$

$$\pi_2 : 1/3 \rightarrow 0.29$$

$$\pi_3 : 1/3 \rightarrow 0.42$$

After updating the mixture weights, the third component gains more importance, while the other components become slightly less influential.

By updating the means, variances, and weights, the GMM in 10 is notably improved compared to its initial state in 4. The parameter updates have shifted the GMM density toward the data points. This improvement is further reflected in the log-likelihood, which increased from -28.3 (initialization) to -14.4 after the full update cycle.

The algorithm in general

What we did above is a specific instance of a more general algorithm described in 1977 by Dempster et al. and known as expectation maximization (EM) algorithm. It is applicable in many situations where some parameters may be estimated by maximum likelihood. The algorithm can be described as follows:

- Initialization: select a set of the model parameters to start with.
- E-step: compute the responsabilities with the current parameters.
- M-step: update the parameters given the new responsabilities values.

We can than outline the process we followed in our example as:

- Initialization: Select heuristically the set of parameters $\boldsymbol{\theta} = \{\mu_k, \Sigma_k, \pi_k : k = 1, ..., K\}$ to begin with.
- E-step: compute the data set responsabilities with

$$r_{nk} = \frac{\pi_k \mathcal{N}\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)}{\sum_{j=1}^{K} \pi_j \mathcal{N}\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)}$$

- M-step: update the parameters through the following formulas

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} \boldsymbol{x}_n,$$
$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} \left(\boldsymbol{x}_n - \boldsymbol{\mu}_k\right) \left(\boldsymbol{x}_n - \boldsymbol{\mu}_k\right)^\top,$$
$$\pi_k = \frac{N_k}{N}.$$

This iterative process continues until some termination condition is satisfied. This condition can be defined as a threshold value for either the negative log-likelihood function or the parameter values directly.