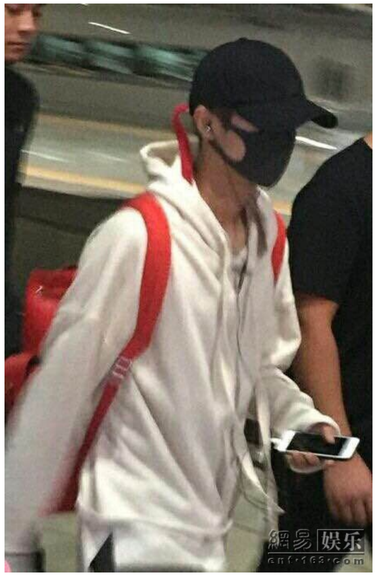
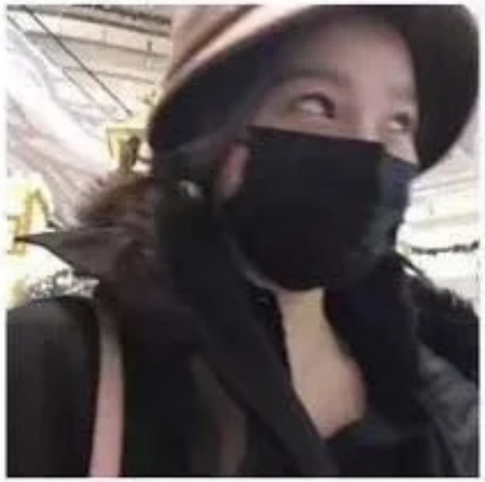
Density Estimation with Gaussian Mixture Models

Liang Zheng
Australian National University
liang.zheng@anu.edu.au

Bayes' theorem

$$\underbrace{p(\boldsymbol{\theta}|\, \boldsymbol{y})}_{\text{posterior}} = \frac{\overbrace{p(\,\boldsymbol{y}|\boldsymbol{\theta})}^{\text{likelihood}}\ \overbrace{p(\boldsymbol{\theta})}^{\text{prior}}}{\underbrace{p(\boldsymbol{y})}_{\text{evidence}}}$$

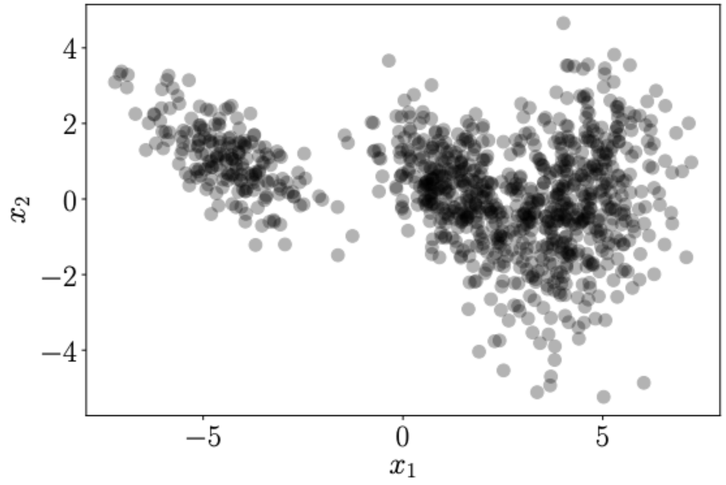




Motivation

建模能力是有限的

- In practice, the Gaussian distribution has limited modeling capabilities.
- Below is a two-dimensional dataset that cannot be meaningfully represented by a single Gaussian



this model can not be generated by a single Gaussian because it has several clusters.
in this case, we can refer to mixture model for the estimation

- We can use mixture models for density estimation.

使用混合模型进行密度估计

- Mixture models can be used to describe a distribution $p(\boldsymbol{x})$ by a convex combination of K simple (base) distributions

mixture model

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k p_k(\boldsymbol{x})$$

pk(x)is teh base distribution

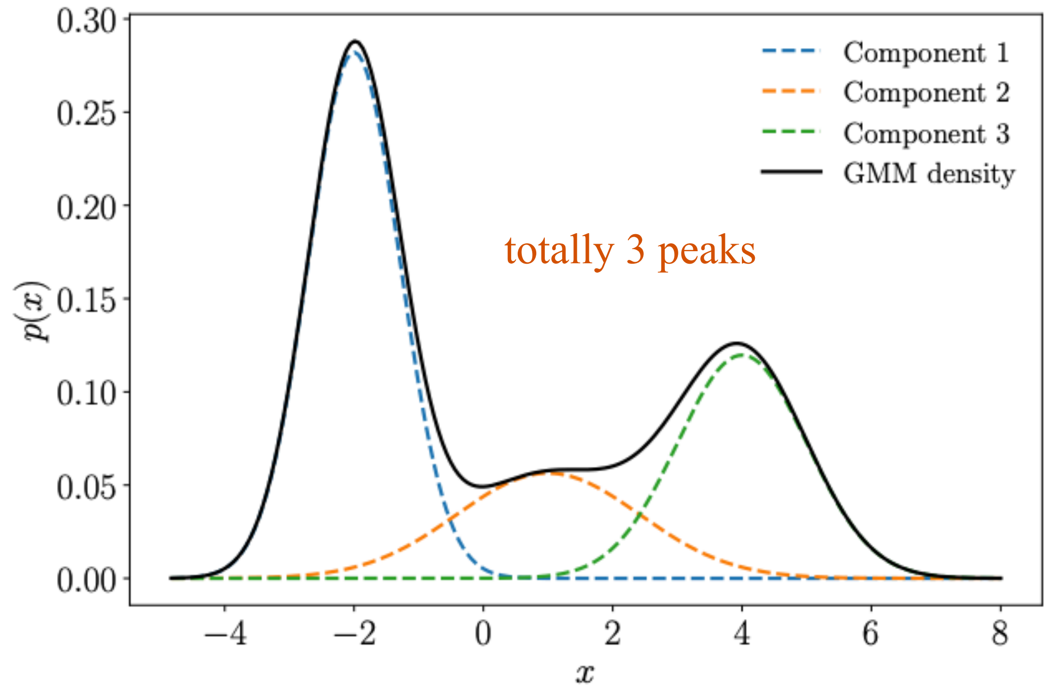
$$0 \leq \pi_k \leq 1, \qquad \sum_{k=1}^{K} \pi_k = 1$$

pi_k is the weight sum

where the components p_k are members of a family of basic distributions, e.g., Gaussians, Bernoullis, or Gammas, and the π_k are mixture weights.

11.1 Gaussian Mixture Model

A GMM is a linear combination of several Gaussian distributions.



totally 3 peaks

based on the black curved, we want to know the parameters of each Gaussian mixture model

The Gaussian mixture distribution (black) is composed of a convex combination of Gaussian distributions and is more expressive than any individual component. Dashed lines represent the weighted Gaussian components.

$$p(x|\boldsymbol{\theta}) = \underbrace{0.5\mathcal{N}\left(x \,\middle|\, -2, \frac{1}{2}\right)}_{0.5 \text{ is pi\_1}} + \underbrace{0.2\mathcal{N}(x|1, 2)}_{0.2 \text{ is pi\_2}} + \underbrace{0.3\mathcal{N}(x|4, 1)}_{0.3 \text{ is pi\_3}}$$

the mixture model is the sum of the three components

choose K base on the individual components numbers (peak numbers)

11.1 Gaussian Mixture Model

it's actually a probability density model

- A Gaussian mixture model (GMM) is a density model where we combine a finite number of K Gaussian distributions $N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ so that

The optimization method based on this equation is called the maximum likelihood estimate

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

K is a hyper parameter

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

and we will have k sets of parameters

where we defined $\boldsymbol{\theta} := \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k : k = 1, \cdots, K\}$ as the collection of all parameters of the GMM.

K means mu, the covariance sigma,mixture weight pi are the parameters in GMM

- GMM gives us significantly more flexibility for modeling complex densities than a simple Gaussian distribution.
- **Parameter Learning via Maximum Likelihood**

theta is continuous parameter.

- Assume we are given a dataset $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$, where $\boldsymbol{x}_n, n = 1, \dots, N$, are drawn i.i.d. from an unknown distribution $p(\boldsymbol{x})$. Our objective is to find a good approximation/representation of this unknown distribution $p(\boldsymbol{x})$ by means of a GMM with K components.

11.2 Parameter Learning via Maximum Likelihood

- Assume we are given a dataset $\boldsymbol{\mathcal{X}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$, where $\boldsymbol{x}_n, n = 1, \dots, N$, are drawn i.i.d. from an unknown distribution $p(\boldsymbol{x})$.
- Our objective is to find a good approximation/representation of this unknown distribution $p(\boldsymbol{x})$ by means of a GMM with K components.

Example

- We consider a one-dimensional dataset $\mathcal{X} = \{-3, -2.5, -1, 0, 2, 4, 5\}$ consisting of 7 data points and wish to find a GMM with $K = 3$ components that models the density of the data.

It's hard to know what is P(x) really looks like, because it's kind of hidden things, but we want to use some GMM to uncover it.

- We initialize the mixture components as

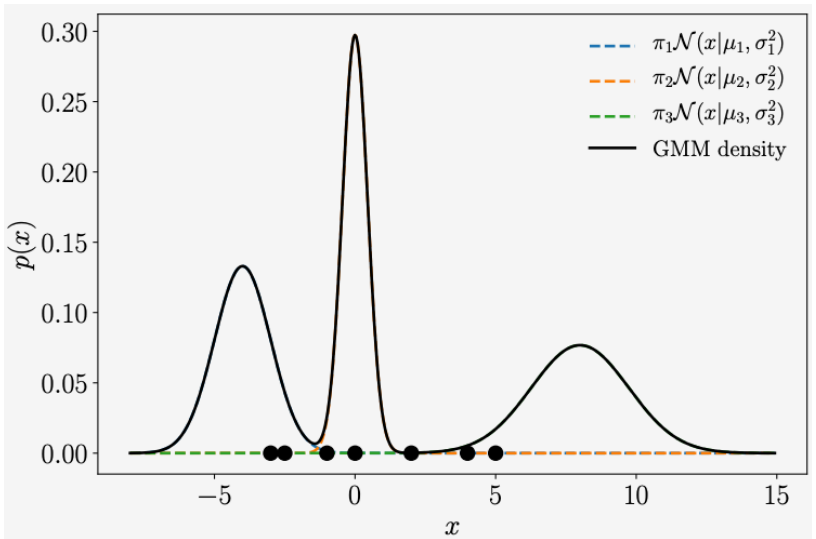
first step: Initialization

$$p_1(x) = \mathcal{N}(x|-4, 1)$$
$$p_2(x) = \mathcal{N}(x|0, 0.2)$$
$$p_3(x) = \mathcal{N}(x|8, 3)$$

这些数据都只是猜测

and assign them equal weights $\pi_1 = \pi_2 = \pi_3 = \frac{1}{3}$.

- We can view the corresponding model and the data points below.



Not a good fit to the data points. The model should predicts that there will be many points under the peak.

Our goal is to estimate the underlying probability function p(x)

- How to obtain a maximum likelihood estimate $\boldsymbol{\theta}_{ML}$ of model parameters $\boldsymbol{\theta}$?
- We start by writing down the likelihood, i.e., the predictive distribution of the training data given the parameters. We exploit our i.i.d. assumption, which leads to the factorized likelihood

likelihood for single data point

we have N data points

$$p(\mathcal{X}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\theta}),$$

the product of each individual probability

$$p(\underbrace{\boxed{\boldsymbol{x}_n}}_{\text{Observed data}}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \underbrace{\boxed{\pi_k}}_{\text{Mixture proportion}} \underbrace{\boxed{\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}}_{\text{Mixture component}}$$

where every individual likelihood term $p(\boldsymbol{x}_n|\boldsymbol{\theta})$ is a Gaussian mixture density.

we assume every every points are identically and independently i.i.d)

- Then we obtain the log-likelihood (loss function) as

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

find the parameter that can gives the max likelihood

- We aim to find parameters $\boldsymbol{\theta}_{ML}^*$ (including $\boldsymbol{\mu}_k^*, \boldsymbol{\Sigma}_k^*, \pi_k^*$) that maximize log-likelihood $\mathcal{L}$ defined above.

however, unlike MLE in linear regression, we can not obtain a closed form solution, but we can exploit an iterative schema to find good model parameter theta. turn out to be EM.

- We obtain the following necessary conditions when we optimize the log-likelihood with respect to the GMM parameters $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_\mathrm{k}, \pi_k$:

The mean mu_k of Gaussian is kind of the centre

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_k} = \mathbf{0}^\mathrm{T} \Leftrightarrow \sum_{n=1}^{N} \frac{\partial \log p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\mu}_k} = \mathbf{0}^\mathrm{T}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}_k} = 0 \Leftrightarrow \sum_{n=1}^{N} \frac{\partial \log p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_k} = \mathbf{0}$$

$$\frac{\partial \mathcal{L}}{\partial \pi_k} = 0 \Leftrightarrow \sum_{n=1}^{N} \frac{\partial \log p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \pi_k} = 0$$

- For all three necessary conditions, by applying the chain rule, we require partial derivatives of the form

$$\frac{\partial \log p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{1}{p(\boldsymbol{x}_n|\boldsymbol{\theta})} \frac{\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$

the orange part is known

- where $\boldsymbol{\theta} = \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_\mathrm{k}, \pi_k \colon k = 1, \cdots, K\}$ are the model parameters and

$$\frac{1}{p(\boldsymbol{x}_n|\boldsymbol{\theta})} = \frac{1}{\sum_{j=1}^{K} \pi_j \mathcal{N}\left(\boldsymbol{x}_n \middle| \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)}$$

11.2.1 Responsibilities

r means the membership

n means the n^th data point, k means the k^th Gaussian component

- We define the quantity

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

r_nk is the probability that the n^th data point belongs to the k^th Gaussian component

as the responsibility of the kth mixture component for the nth data point.

- We can see r_{nk} is proportional to the likelihood

$$p(\boldsymbol{x}_n | \pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

of the kth mixture component given the data point.

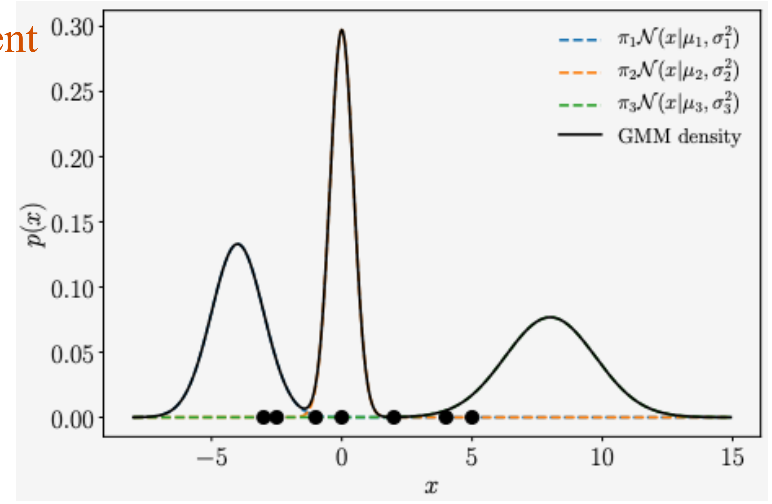
- The responsibility r_{nk} represents the posterior probability that $\boldsymbol{x}_n$ has been generated by the kth mixture component
- Note that $\boldsymbol{r}_n := [r_{n1}, \cdots, r_{nK}]^{\mathrm{T}} \in \mathbb{R}^K$ is a (normalized) probability vector, i.e., $\sum_k r_{nk} = 1$ with $r_{nk} \geq 0$.
- This probability vector distributes probability mass among the K mixture components, and we can think of $\boldsymbol{r}_n$ as a “soft assignment” of $\boldsymbol{x}_n$ to the K mixture components.

$\sum_k r_{nk} = 1$

Example - Responsibilities

- From the figure below, we compute the responsibilities r_{nk}

- $$\begin{array}{ccc|l} \text{first.} & \text{second.} & \text{third.} & \text{Gaussian component} \\ \hline 1.0 & 0.0 & 0.0 & \\ 1.0 & 0.0 & 0.0 & \\ 0.057 & 0.943 & 0.0 & \\ 0.001 & 0.999 & 0.0 & \\ 0.0 & 0.066 & 0.934 & \\ 0.0 & 0.0 & 1.0 & \\ 0.0 & 0.0 & 1.0 & \end{array} \in \mathbb{R}^{N\times K}$$



- The nth row tells us the responsibilities of all mixture components for x_n.
- The sum of all K responsibilities for a data point (sum of every row) is 1.
- The kth column gives us an overview of the responsibility of the kth mixture component.
- The third mixture component (third column) is not responsible for any of the first four data points, but takes much responsibility of the remaining data points.
- The sum of all entries of a column gives us the values N_k, i.e., the total responsibility of the kth mixture component. In our example, we get $N_1 = 2.058$, $N_2 = 2.008$, $N_3 = 2.934$. $N_k = \sum_n r_{nk}$.
- We will determine the updates of the model parameters $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and π_k for given responsibilities

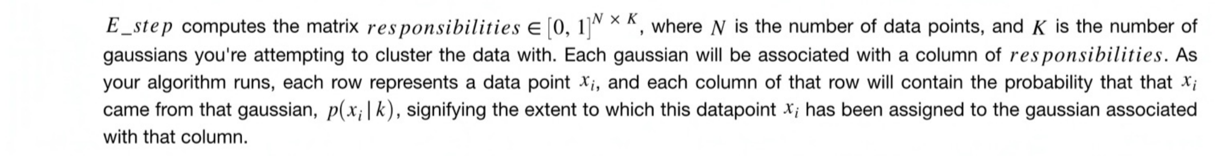
11.3 EM Algorithm

K-means is a special, simple case of the Expectation Maximisation algorithm

also want to determine the hyper parameter k

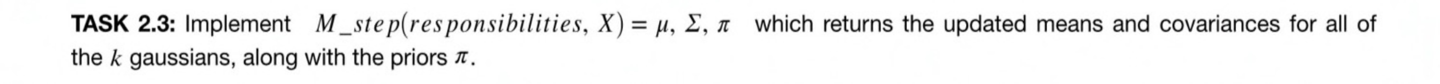
- In GMM, we first initialize the parameters $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and π_k and alternate until convergence between the following two steps

- E-step: Evaluate the responsibilities r_{nk} (probability of data point n belonging to mixture component k)



E_step computes the matrix $responsibilities \in [0, 1]^{N \times K}$, where N is the number of data points, and K is the number of gaussians you're attempting to cluster the data with. Each gaussian will be associated with a column of $responsibilities$. As your algorithm runs, each row represents a data point x_i, and each column of that row will contain the probability that that x_i came from that gaussian, $p(x_i \mid k)$, signifying the extent to which this datapoint x_i has been assigned to the gaussian associated with that column.

- M-step: Use the updated responsibilities to re-estimate the parameters $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and π_k



TASK 2.3: Implement $M\_step(responsibilities, X) = \mu, \Sigma, \pi$ which returns the updated means and covariances for all of the k gaussians, along with the priors π.

Why use EM step

because the direct opyimisation of the likelihood in GMM is not visible. We can't directly optimize that function. Thus we have to optimize the parameter one by one. And we can not get a closed-form, thus we need to generate an iteration to find the best parameters.

Check your understanding

different components have different mixture weight

- F Given a dataset generated by a mixture of 3 Gaussians, when we randomly sample a data point, it has the probability of 1/3 belonging to each Gaussian.
- T A GMM is a linear combination of several Gaussian distributions.
- T In GMM, K (number of Gaussians) is a hyperparameter.
- F If a dataset is not generated by Gaussian distributions, it cannot be modeled by GMM.

we can approximate any data distribution with GMM,
but likelihood might be very low or model doesn't perform well.

11.3 EM Algorithm

- Initialize $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k$. (below is an example)
  - $\pi_k = 1/K$ for all k
  - $\boldsymbol{\mu}_k$: centroids from k-means algorithm or using randomly chosen data points
  - $= \boldsymbol{\Sigma}$ the sample variance, for all k

- E-step: Evaluate responsibilities r_{nk} for every data point $\boldsymbol{x}_n$ using current parameters $\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$:

$$r_{nk} = \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}\left(\boldsymbol{x}_n \middle| \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)}$$

every thing is known, so use it to calculate the r_nk

- M-step: Re-estimate parameters $\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ using the current responsibilities r_{nk} (from E-step):

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} \boldsymbol{x}_n$$

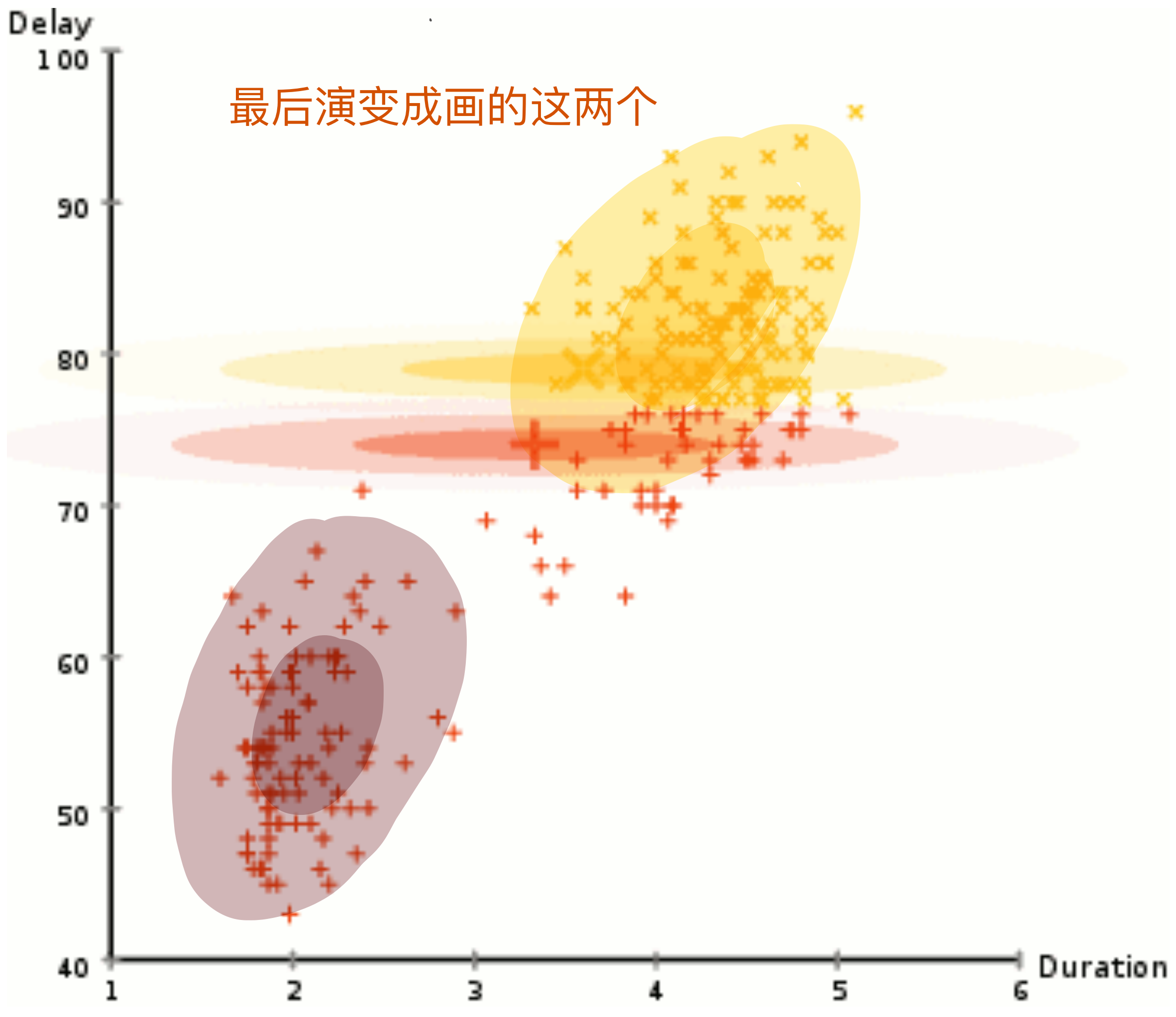
sigma_k is the covariance matrix for k^th Gaussian.

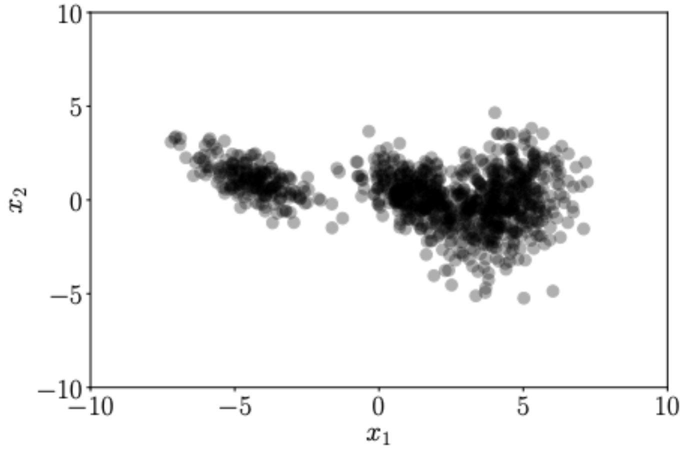
$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

$$\pi_k = \frac{N_k}{N}$$

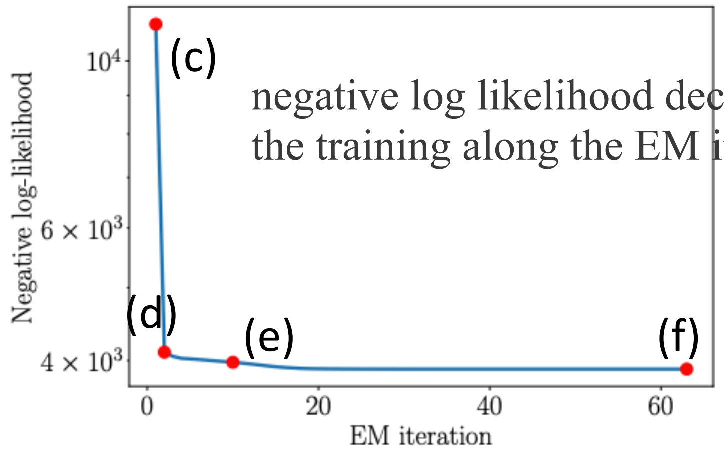
N is the total number of the data points

11.3 EM Algorithm



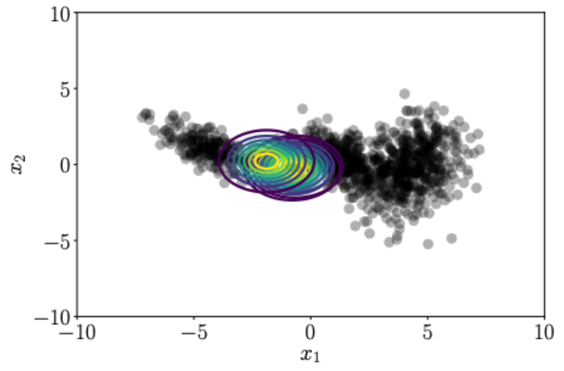


(a) Dataset.

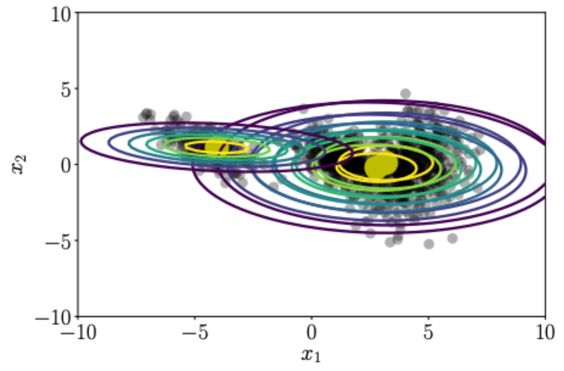


negative log likelihood decrease in the training along the EM iteration

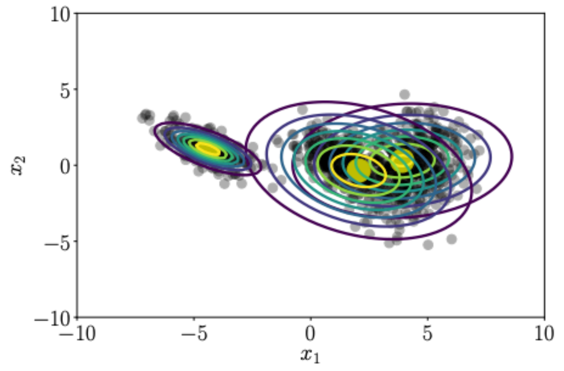
(b) Negative log-likelihood.



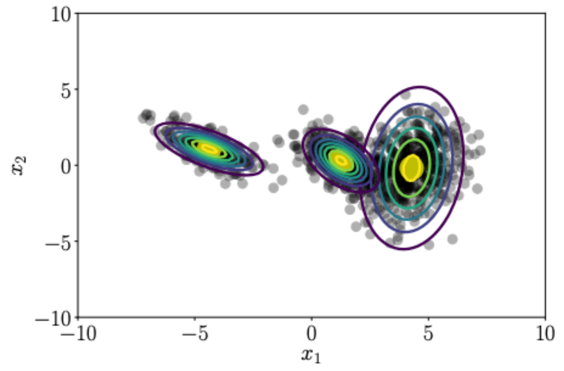
(c) EM initialization.



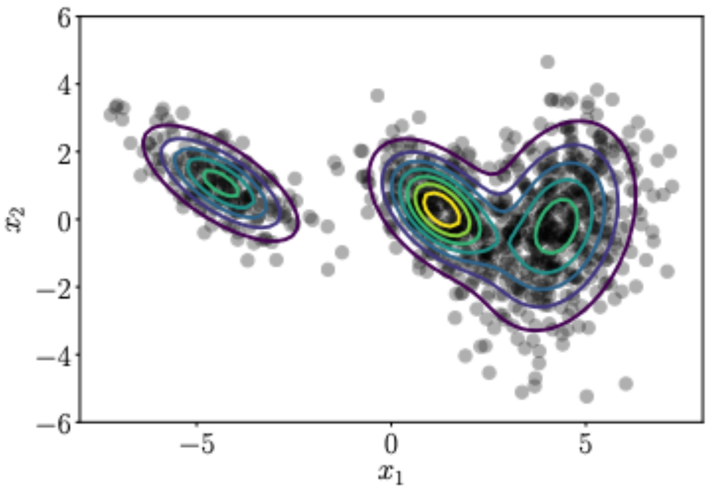
(d) EM after one iteration.

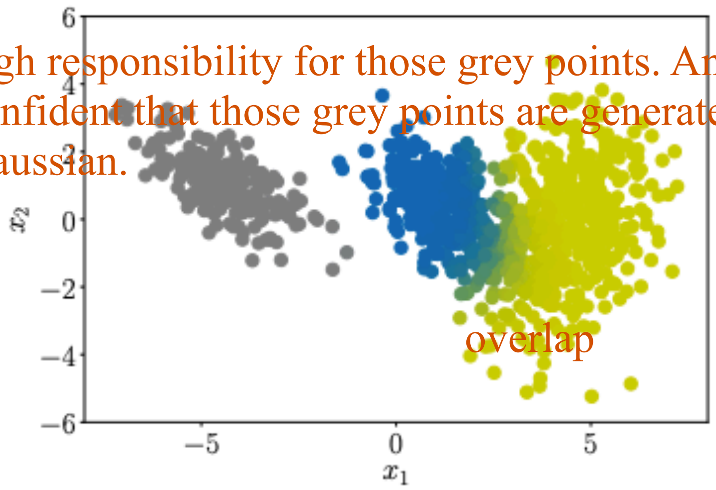


(e) EM after 10 iterations.



(f) EM after 62 iterations.


(a) GMM fit after 62 iterations.



high responsibility for those grey points. And very confident that those grey points are generated by this Gaussian.

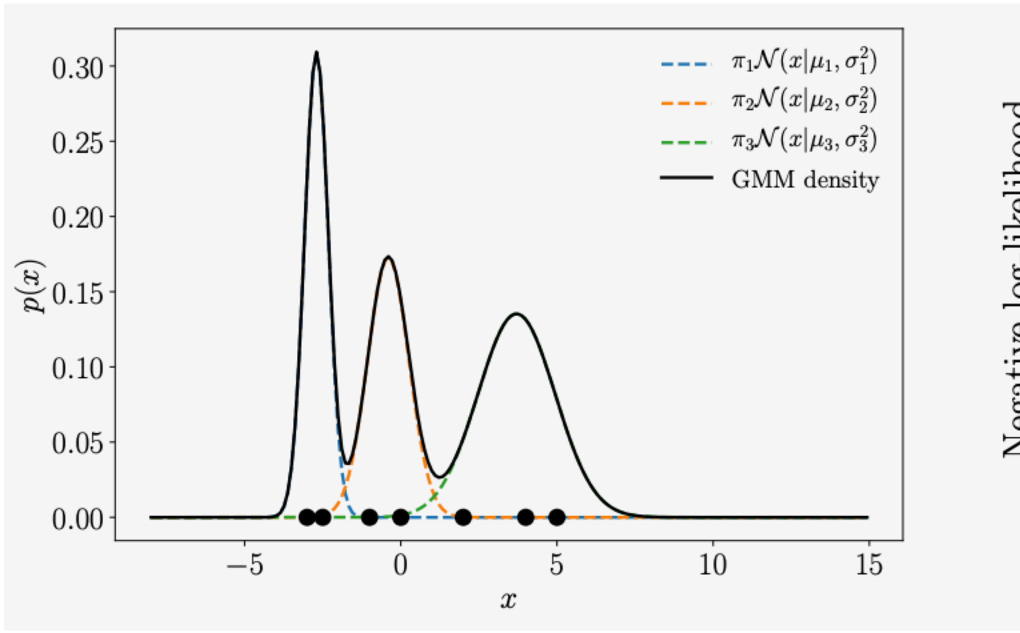
(b) Dataset colored according to the responsibilities of the mixture components.

- The dataset is colored according to the responsibilities of the mixture components when EM converges.
- A single mixture component is highly responsible for the data on the left.
- The overlap of the two data clusters on the right could have been generated by two mixture components.
- It becomes clear that there are data points that cannot be uniquely assigned to a single component (either blue or yellow), such that the responsibilities of these two clusters for those points are around 0.5.

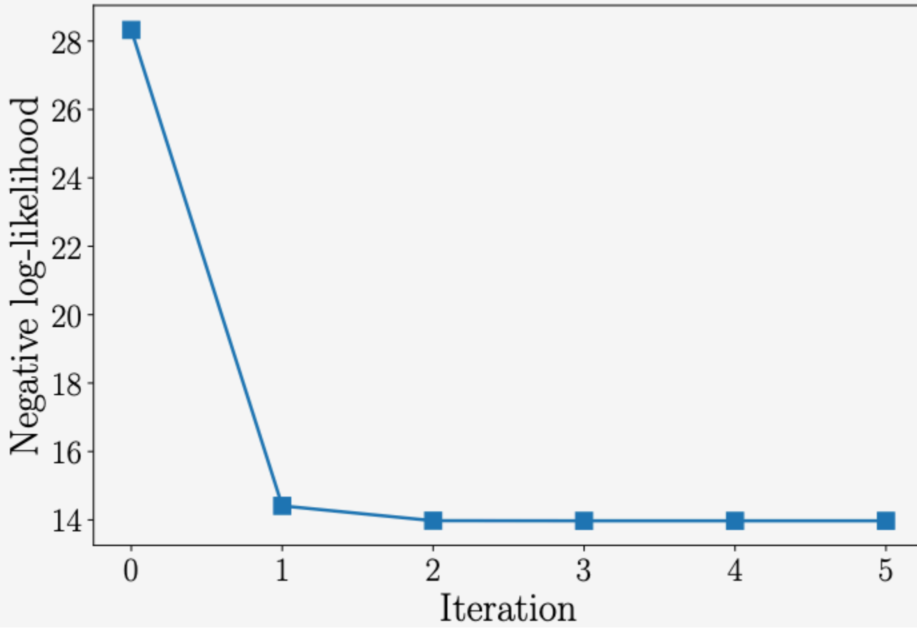
11.3 EM Algorithm

- The final GMM is given as

$$p(x) = 0.29\mathcal{N}(x|-2.75\,,0.06) + 0.28\mathcal{N}(x|-0.50\,,0.25) + 0.43\mathcal{N}(x|3.64\,,1.63)$$



Final GMM fit. After five iterations, the EM algorithm converges and returns this GMM



Negative log-likelihood as a function of the EM iterations.

11.2.2 Updating the Means

- The update of the mean parameters $\boldsymbol{\mu}_k, k = 1, \ldots, K$, of the GMM is given by

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

- Proof: Calculate the gradient of the log-likelihood with respect to $\boldsymbol{\mu}_k$
- Considering

$$\mathcal{L}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k) = \log p(\mathcal{X}|\boldsymbol{\theta}) = \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\boldsymbol{\theta})$$

the gradient of the log-likelihood with respect to the mean parameters requires us to compute partial derivative.

p is the probability density function of the GMM

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

- We have

when j not equal j, the partial derivative is 0. so it can be simplify

$$\frac{\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\mu}_k} = \sum_{j=1}^{K} \pi_j \frac{\partial \mathcal{N}\left(\boldsymbol{x}_n \middle| \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)}{\partial \boldsymbol{\mu}_k} = \pi_k \frac{\partial \mathcal{N}\left(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)}{\partial \boldsymbol{\mu}_k}$$

only the k^th mixture component depends on mu_k

- Recall our knowledge in multivariate Gaussian distribution and vector calculus

$$p(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{D}{2}}|\boldsymbol{\Sigma}|^{-\frac{1}{2}} exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))$$

$$\frac{\partial \boldsymbol{x}^{\mathrm{T}}\boldsymbol{B}\boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{x}^{\mathrm{T}}(\boldsymbol{B}+\boldsymbol{B}^{\mathrm{T}})$$

- We have

$$\frac{\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\mu}_k} = \pi_k(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\mathcal{N}\left(\boldsymbol{x}_n \middle| \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

11.2.2 Updating the Means

- The desired partial derivative of $\mathcal{L}$ with respect to $\boldsymbol{\mu}_k$ is given as

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_k} = \sum_{n=1}^{N} \frac{\partial \log p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\mu}_k} = \sum_{n=1}^{N} \frac{1}{p(\boldsymbol{x}_n|\boldsymbol{\theta})} \frac{\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\mu}_k},$$

$$= \sum_{n=1}^{N} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1} \underbrace{\boxed{\frac{\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}}_{} = r_{nk}$$

$$= \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1}$$

- We now solve the above gradient for $\boldsymbol{\mu}_k^{new}$ so that $\frac{\partial \mathcal{L}(\boldsymbol{\mu}_k^{new})}{\partial \boldsymbol{\mu}_k} = \boldsymbol{0}^{\mathrm{T}}$ and obtain

$$\sum_{n=1}^{N} r_{nk} \boldsymbol{x}_n = \sum_{n=1}^{N} r_{nk} \boldsymbol{\mu}_k^{new} \Leftrightarrow \boldsymbol{\mu}_k^{new} = \frac{\sum_{n=1}^{N} r_{nk} \boldsymbol{x}_n}{\boxed{\sum_{n=1}^{N} r_{nk}}} = \frac{1}{\boxed{N_k}} \sum_{n=1}^{N} r_{nk} \boldsymbol{x}_n$$

where we define

$$N_k := \sum_{n=1}^{N} r_{nk}$$

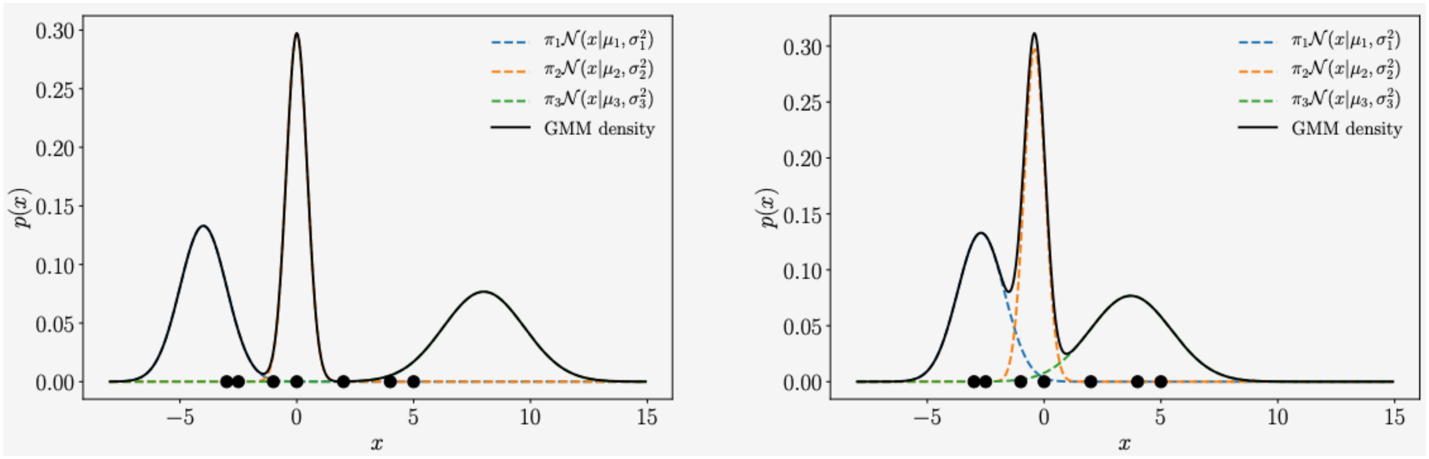
as the total responsibility of the kth mixture component for the entire dataset.

- This concludes the proof.

11.2.2 Updating the Means

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

- This is an importance-weighted Monte Carlo estimate of the mean.
- The importance weights of data point $\boldsymbol{x}_n$ is r_{nk}
- Mean update



Initialization:

$\mathcal{X} = \{-3, -2.5, -1, 0, 2, 4, 5\}$

$$\pi_1 = \pi_2 = \pi_3 = \frac{1}{3}$$
$$p_1(x) = \mathcal{N}(x|-4, 1)$$
$$p_2(x) = \mathcal{N}(x|0, 0.2)$$
$$p_3(x) = \mathcal{N}(x|8, 3)$$

$$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 1.0 & 0.0 & 0.0 \\ 0.057 & 0.943 & 0.0 \\ 0.001 & 0.999 & 0.0 \\ 0.0 & 0.066 & 0.934 \\ 0.0 & 0.0 & 1.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$$

$$\mu_1 : -4 \longrightarrow -2.7$$
$$\mu_2 : 0 \longrightarrow -0.4$$
$$\mu_3 : 8 \longrightarrow 3.7$$

$$-2.7 = \frac{-3\times1 - 2.5\times1 - 1\times0.057 - 0\times0.001}{1 + 1 + 0.057 + 0.001}$$

11.2.3 Updating the Covariances

- The update of the covariance parameters $\boldsymbol{\Sigma}_k, k = 1, \ldots, K$ is given by

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

- *Proof* We compute the partial derivatives of the log-likelihood $\mathcal{L}$ with respect to the covariances $\boldsymbol{\Sigma}_k$, set them to $\mathbf{0}$, and solve for $\boldsymbol{\Sigma}_k$. We start by

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}_k} = \sum_{n=1}^{N} \frac{\partial \log p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_k} = \sum_{n=1}^{N} \frac{1}{p(\boldsymbol{x}_n|\boldsymbol{\theta})} \frac{\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_k}$$

- We already know $1/p(\boldsymbol{x}_n|\boldsymbol{\theta})$. To obtain $\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})/\partial \boldsymbol{\Sigma}_k$, we have,

$$\begin{aligned}
\frac{\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_k} &= \frac{\partial}{\partial \boldsymbol{\Sigma}_k}\left(\pi_k (2\pi)^{-\frac{D}{2}} \det(\boldsymbol{\Sigma}_k)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)\right)\right) \\
&= \pi_k (2\pi)^{-\frac{D}{2}} \left[\frac{\partial}{\partial \boldsymbol{\Sigma}_k} \det(\boldsymbol{\Sigma}_k)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)\right)\right. \\
&\quad \left. + \det(\boldsymbol{\Sigma}_k)^{-\frac{1}{2}} \frac{\partial}{\partial \boldsymbol{\Sigma}_k} \exp\left(-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)\right)\right]
\end{aligned}$$

11.2.3 Updating the Covariances

- From Vector Calculus, we have the following identities

$$\frac{\partial}{\partial \boldsymbol{\Sigma}_k} \det(\boldsymbol{\Sigma}_k)^{-\frac{1}{2}} = -\frac{1}{2}\det(\boldsymbol{\Sigma}_k)^{-\frac{1}{2}}\boldsymbol{\Sigma}_k^{-1}$$

$$\frac{\partial}{\partial \boldsymbol{\Sigma}_k}(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n-\boldsymbol{\mu}_k) = -\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}$$

- We obtain the desired partial derivative

$$\frac{\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_k} = \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \cdot \left[-\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{-1} - \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\right)\right]$$

- Thus, the partial derivative of the log-likelihood with respect to $\boldsymbol{\Sigma}_k$ is given by

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}_k} = \sum_{n=1}^{N}\frac{\partial \log p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_k} = \sum_{n=1}^{N}\frac{1}{p(\boldsymbol{x}_n|\boldsymbol{\theta})}\frac{\partial p(\boldsymbol{x}_n|\boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_k}$$

$$= \sum_{n=1}^{N}\underbrace{\frac{\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K}\pi_j \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}_{= r_{nk}} \cdot \left[-\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{-1} - \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\right)\right]$$

$$= -\frac{1}{2}\sum_{n=1}^{N} r_{nk}\left(\boldsymbol{\Sigma}_k^{-1} - \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}_k^{-1}\right)$$

$$= -\frac{1}{2}\boldsymbol{\Sigma}_k^{-1}\underbrace{\sum_{n=1}^{N} r_{nk}}_{N_k} + \frac{1}{2}\boldsymbol{\Sigma}_k^{-1}\left(\sum_{n=1}^{N} r_{nk}(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^{\mathrm{T}}\right)\boldsymbol{\Sigma}_k^{-1}$$

11.2.3 Updating the Covariances

- Setting this partial derivative to $\mathbf{0}$, we obtain the necessary optimality condition

$$N_k \cancel{\boldsymbol{\Sigma}_k^{-1}} = \cancel{\boldsymbol{\Sigma}_k^{-1}} \left(\sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \right) \boldsymbol{\Sigma}_k^{-1}$$

$$\Longleftrightarrow N_k \boldsymbol{I} = \left(\sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \right) \boldsymbol{\Sigma}_k^{-1}$$

- By solving for $\boldsymbol{\Sigma}_k$, we obtain

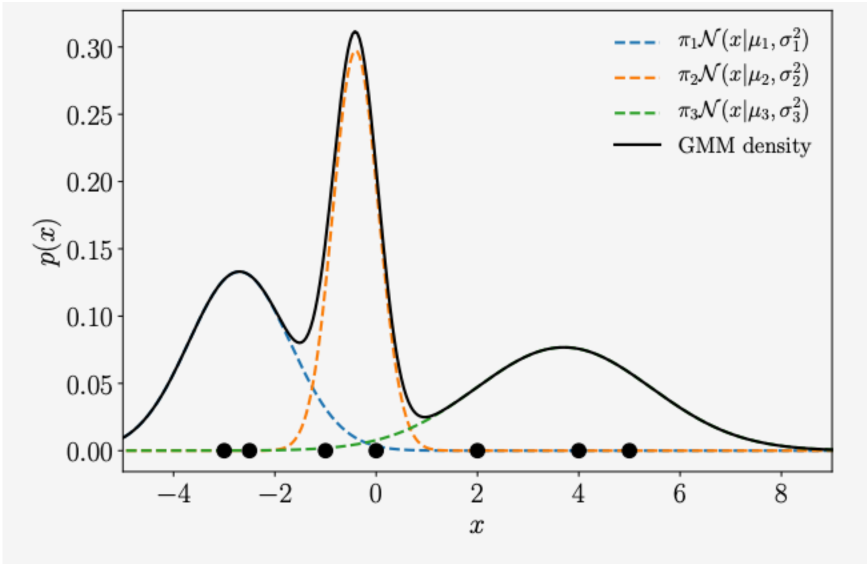
先把muk 算出来

$$\boldsymbol{\Sigma}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

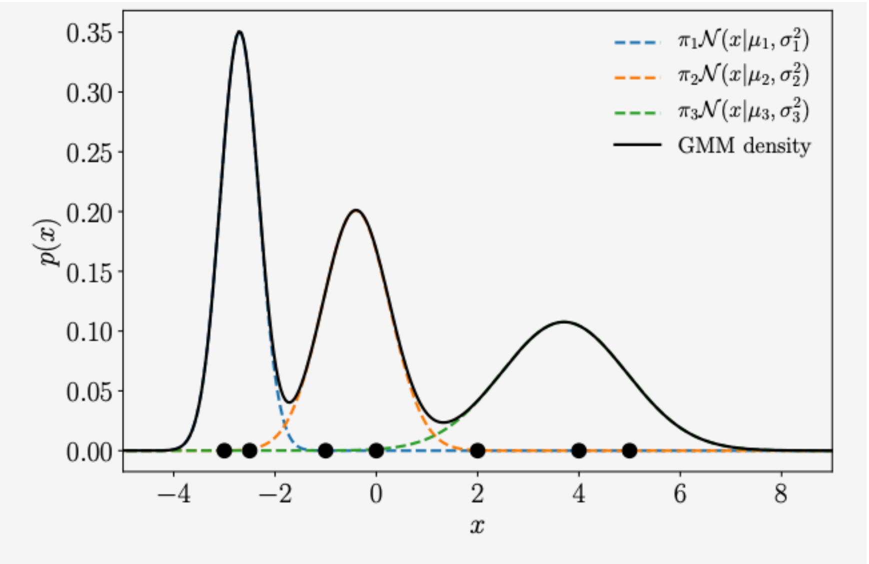
- This gives us a simple update rule for $\boldsymbol{\Sigma}_k$ for $k = 1, \ldots, K$ and proves our theorem.

- This update method is the weighted covariance of data points $\boldsymbol{x}_n$ associated with the kth component.
- The weights are the responsibilities r_{nk}

11.2.3 Updating the Covariances



(a) GMM density and individual components prior to updating the variances.



(b) GMM density and individual components after updating the variances.

$$\sigma_1^2 : 1 \longrightarrow 0.14$$
$$\sigma_2^2 : 0.2 \longrightarrow 0.44$$
$$\sigma_3^2 : 3 \longrightarrow 1.53$$

shrink significantly, whereas the variance of the second component increase slightly.

11.2.4 Updating the Mixture Weights

- The mixture weights of the GMM are updated as

$$\pi_k^{new} = \frac{N_k}{N}, k = 1, \cdots, K$$

where N is the number of data points

- *Proof* We calculate the partial derivative of the log-likelihood with respect to the weight parameters $\pi_k, k = 1, \ldots, K$.
- We have the constraint

still, the sum of the mixture weight should be 1

$$\sum_k \pi_k = 1$$

- Using Lagrange multipliers (will not be covered in this course), we have

拉格朗日乘数

$$\mathfrak{L} = \mathcal{L} + \lambda\left(\sum_{k=1}^{K} \pi_k - \mathbf{1}\right)$$

we want the thing in the bracket as small as possible

$$= \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) + \lambda\left(\sum_{k=1}^{K} \pi_k - \mathbf{1}\right)$$

we want the likelihood L as large as possible

$$\mathfrak{L} = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) + \lambda\left(\sum_{k=1}^{K} \pi_{\underset{j}{k}} - \mathbf{1}\right)$$

- We obtain the partial derivative with respect to π_k as

only when j = k
we will have lambda in the derivative

$$\frac{\partial \mathfrak{L}}{\partial \pi_k} = \sum_{n=1}^{N} \frac{\mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}\left(\boldsymbol{x}_n \middle| \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)} + \lambda$$

$$= \frac{1}{\pi_k} \underbrace{\sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}\left(\boldsymbol{x}_n \middle| \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)}}_{= N_k} + \lambda = \frac{N_k}{\pi_k} + \lambda \quad = 0$$

- The partial derivative with respect to the Lagrange multiplier λ is

$$\frac{\partial \mathfrak{L}}{\partial \lambda} = \sum_{k=1}^{K} \pi_k - \mathbf{1} \quad = 0$$

- Setting both partial derivatives to 0 yields the system of equations

$$\pi_k = -\frac{N_k}{\lambda}$$

$$1 = \sum_{k=1}^{K} \pi_k$$

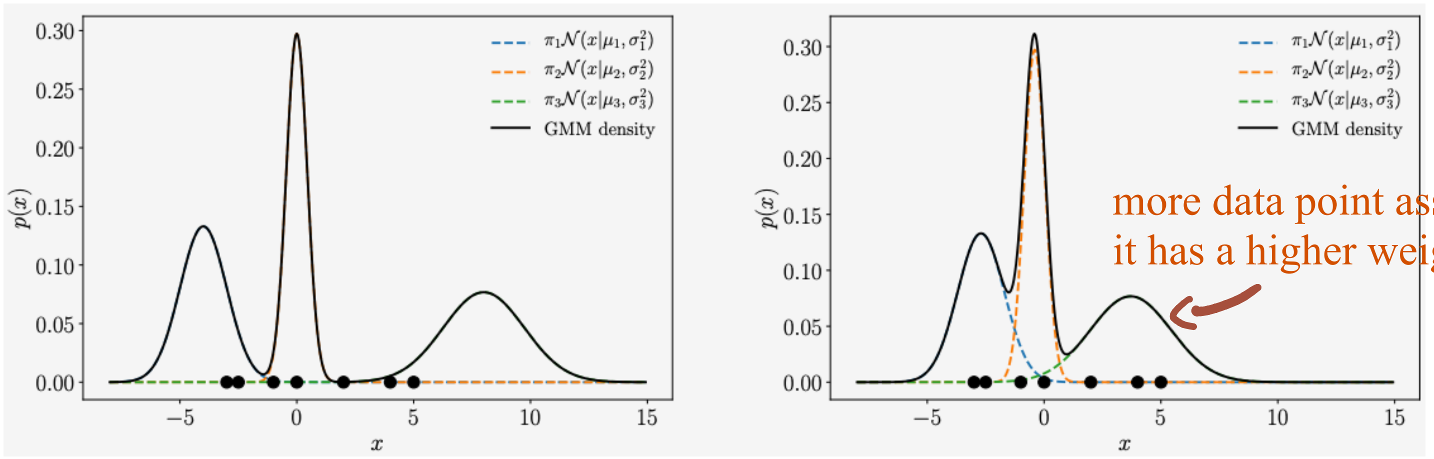
- Using the two equations, we obtain

$$\sum_{k=1}^{K} \pi_k = 1 \Leftrightarrow -\sum_{k=1}^{K} \frac{N_k}{\lambda} = 1 \Leftrightarrow -\frac{N}{\lambda} = 1 \Leftrightarrow \lambda = -N$$

- This allows us to substitute $-N$ for λ in $\pi_k = -\frac{N_k}{\lambda}$ to obtain

$$\pi_k^{new} = \frac{N_k}{N}$$

which gives us the update for the weight parameters π_k and proves the Theorem.



more data point assigned to it, so it has a higher weight

$$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 1.0 & 0.0 & 0.0 \\ 0.057 & 0.943 & 0.0 \\ 0.001 & 0.999 & 0.0 \\ 0.0 & 0.066 & 0.934 \\ 0.0 & 0.0 & 1.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$$

$$\pi_1 : \frac{1}{3} \longrightarrow 0.29 \quad (11.50)$$

$$\pi_2 : \frac{1}{3} \longrightarrow 0.29 \quad (11.51)$$

$$\pi_3 : \frac{1}{3} \longrightarrow 0.42 \quad (11.52)$$

$$0.29 = \frac{1 + 1 + 0.057 + 0.001}{7}$$

- We see that the third component gets more weight/importance, while the other components become slightly less important.

Generating a new dataset with GMM

- For a given GMM with parameters $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k, k = 1, \ldots, K$, we want to generate a dataset with N data points.
- We sample an index k from $\{1, 2, \ldots, K\}$ with probabilities $\pi_1, \ldots, \pi_k$
- We generate a number of $N\pi_k$ data points for the kth component
- In the kth component, every data point is sampled as $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

Overall, after updated the means, the variance, and the weight once, we obtain the new GMM . compare with the previous figure, we can see the parameter updates caused the GMM density to shift some its mass toward the data points. And it is remarkable better than before.

This is also evidenced by the log-likelihood values, which increase after one complete update cycle.

Comparising GMM with K-Means

k-means is hard:

because a point can only belong to 1 cluster.

For example: r_{ik}: $r_{11}.r_{12}.r_{13}$. only one of them can be 1. and the rest 2 are 0.

GMM is soft: r_{ik}: r_{11}, r_{12}, r_{13}. Our three Gaussian Components will always have a value of r_{ik}. Not strictly to be 0.

K-means vs GMM

Algorithms.

1. k-Means
   a. Given hard labels, compute centroids
   b. Given centroids, compute hard labels
2. GMM

   higher computational complexity than k-means

   a. Given soft labels, compute Gaussians
   b. Given Gaussians, compute soft labels

- Like k-means, GMM may get stuck in local minima.
- Unlike k-means, the local minima are more favorable because soft labels allow points to move between clusters slowly.

Check your understanding

after convergence, model will have a smaller loss

- T If K takes a greater value, the likelihood becomes greater after convergence.
- T Assume we have N data points. The maximum likelihood will be achieved if we set $K = N$. however, it will be overfitting
- F In GMM, the EM algorithm gives us global minimum ✓, because we can update π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ through closed-form solutions.
- T GMM has a higher computational complexity than kmeans.
- F When the N data points are close to each other in the feature space, we should set K to a small value.

should look at the relatively position of the points.
when you look at the data set close enough, they could be quite far

what's wrong with 3: we can update pi, mu and sigma through closed-form solutions, but we are actually using the EM to optimise the parameters one by one, not jointly