Ex4-Gaussian Mixture Models and Expectation-Maximization

Matthis Brocheton

Probleme 1:

<u>two Gaussian random vectors:</u>

$y \sim N(\mu, \Sigma)$
$z \sim N(\mu, \Sigma)$

<u>We want:</u>

$y + z \sim N(\mu + \hat{\mu}, \Sigma + \hat{\Sigma})$

Linearity of Expectation: $\mathbb{E}[y+z] = \mathbb{E}[y] + \mathbb{E}[z] = \mu + \hat{\mu}$

covariance matrix: $\mathbf{cov}(X, Y) = \mathrm{E}\left[(\mathbf{X} - \mathrm{E}[\mathbf{X}])\,(\mathbf{Y} - \mathrm{E}[\mathbf{Y}])^{\mathrm{T}}\right]$

<u>Calculation :</u>

$y + z = \mu + \hat{\mu}$

$Cov(A + B) = Cov(A) + Cov(B) + Cov(A, B)^T + Cov(A, B)$

$Cov(A, B) = 0$

$Cov(y + z) = Cov(y) + Cov(z)$

$Cov(y + z) = \Sigma + \hat{\Sigma}$

<u>So :</u>
$y + z = \mu + \hat{\mu}$

$Cov(y + z) = \Sigma + \hat{\Sigma}$

<u>We obtain:</u>

$y + z \sim N(\mu + \hat{\mu}, \Sigma + \hat{\Sigma})$

Probleme 2:

Multivariate Gaussian Density Function:

$$p(x^{(i)} \mid \mu_k, \Sigma) = \frac{1}{(2\pi)^{m/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x^{(i)} - \mu_k)^T \Sigma^{-1} (x^{(i)} - \mu_k)\right)$$

Log Likelihood in GMM:

$$l(\boldsymbol{\theta}) = \ln p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{\ell=1}^{n} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(i)}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

K-means objective function:

$$\sum_{i=1}^{n} \|x^{(i)} - \mu_{c^{(i)}}\|^2$$

complete-data log-likelihood

$$log\ p(x, z \mid \mu, \epsilon) = \sum [\log \pi_{zi} - \frac{2}{m} log(2\pi\epsilon) - \frac{\mathbf{1}}{2\epsilon} \parallel x^i - \mu_{zi} \parallel^2]$$

We are interested in (because $\epsilon \to 0$):

$-\frac{\mathbf{1}}{2\epsilon} \parallel x^i - \mu_{zi} \parallel^2$ -> $-\frac{\mathbf{1}}{2\epsilon} \sum \parallel x^i - \mu_{zi} \parallel^2$

This is the distortion function of k-means:

$$\sum \parallel x^i - \mu_{zi} \parallel^2$$

If we maximize the log-likelihood the result matches with the function of k-means:

$$\sum \parallel x^i - \mu_{zi} \parallel^2$$

Probleme 3:

a)

$$p(y_i = y|\boldsymbol{\theta}) = p\frac{\lambda_{1i}^y \exp(-\lambda_{1i})}{y!} + (1-p)\frac{\lambda_{2i}^y \exp(-\lambda_{2i})}{y!},$$

$$\lambda_{1i} = \exp\left(\beta_1 x^{(i)}\right), \quad \lambda_{2i} = \exp\left(\beta_2 x^{(i)}\right)$$

if $\beta 1 = \beta 2$ then the model is $\lambda 1^i = \lambda 2^i$ so not identifiable.

So, we need to have $\beta 1 \neq \beta 2$.

If p is equal to 1 or 2 then we use only one model of fish, we need to have $0 < p < 1$to identify the model.

If x^i is same so the values $\lambda 1^i$ and $\lambda 2^i$ will be equal and we won't have enough information to know the relation with x and y.

So, we must have x^i varied.

b)

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$$

The EM algorithm is an iterative method used to estimate parameters θ in probabilistic models involving latent (hidden) variables Z. It alternates between estimating the distribution of the hidden variables (E-step) and updating the model parameters (M-step) by maximizing a function $Q(\theta, \theta^{old})$

General idea

We estimate that:

visible observations X

hidden variables (or "latent") Z

Since Z is not observed, we estimate its distribution in the E-step, and use it to update parameters in the M-step.

E-step

We calculate responsibilities with:

$$w_i^{(0)} = \frac{P(Z_i = C_0, X_i \mid \theta)}{P(Z_i = C_0, X_i \mid \theta) + P(Z_i = C_1, X_i \mid \theta)}$$

$$w_i^{(1)} = 1 - w_i^{(0)} = \frac{P(Z_i = C_1, X_i \mid \theta)}{P(Z_i = C_0, X_i \mid \theta) + P(Z_i = C_1, X_i \mid \theta)}$$

These weights $w_i^{(0)}$ and $w_i^{(1)}$ represent the probability that the data comes from component 0 or 1.

M-step (maximization)

We maximize the function Q (θ, θ Old) to obtain new parameters θ:

$$J(\theta) = \sum_{i=1}^{n} w_i \cdot (\log \lambda + x_i \log p_0 + (1 - x_i) \log(1 - p_0))$$

$$\sum_{i=1}^{n} (1 - w_i) \cdot (\log(1 - \lambda) + x_i \log p_1 + (1 - x_i) \log(1 - p_1))$$

$$\lambda = \frac{1}{n} \sum_{i=1}^{n} w_i$$

$$p_0 = \frac{\sum_{i=1}^{n} w_i \cdot x_i}{\sum_{i=1}^{n} w_i}$$

$$p_1 = \frac{\sum_{i=1}^{n} (1 - w_i) \cdot x_i}{\sum_{i=1}^{n} (1 - w_i)}$$

In this example, the Q-function becomes an explicit weighted log-likelihood function $J(\theta)$, where each data point is weighted by its probability of coming from component 0 or 1.

These formulas correspond to weighted averages, where each point counts more or less according to its probability of belonging to the class.

And after?

We repeat the processus until the parameters converge, E-step, M-step...

c)

In the EM algorithm, the function Q is defined as the expected complete log-likelihood:

$Q(\theta, \theta^{old}) = E_{Z|X,\theta^{old}}\,[log\ p(X, Z \mid \theta)]$

Which can also be written as a sum over the data:

$Q(\theta, \theta^{old}) = \sum \sum p(\ zi \mid xi, \theta^{old}\ )\ log\ p(\ xi, zi \mid \theta\ )$

This function is differentiable compared to θ, and its derivative gives us the direction in which to adjust the parameters to M-step.

<u>*Derivation of the derivative of $Q(\theta, \theta^{Old})$*</u>

Let's call $\gamma i(z)$ responsibility (or conditional probability) calculated in step E:

$\gamma i(z) := p(zi = z \mid xi, \theta^{old})$

So, we have:

$Q(\theta, \theta^{old}) = \sum \sum p(\ zi \mid xi, \theta^{old}\ )\ log\ p(\ xi, zi \mid \theta\ )$

And derivative in relation to θ is:

$\nabla_\theta Q(\theta, \theta^{old}) = \sum\sum \gamma i(z) \nabla_\theta log p(x_i, z \mid \theta)$

This means that the gradient of Q Q is a weighted sum of the gradients of the log-likelihood, with the responsibilities $\gamma i(z)$ as weight.

In the M-STEP, if we cannot resolve:

$\theta^{new} = argmax Q(\theta, \theta^{old})$

So, we take steps in the direction of the gradient, as in classic automatic learning:

$\theta^{t+1} = \theta^t + \eta \cdot \nabla_\theta Q(\theta^t, \theta^{old})$

Where η η is the learning rate.

This allows us to update the parameters in the right direction, even if we cannot resolve the optimization explicitly.

d)

No, the EM algorithm does not always find the global maximum of the likelihood function.
It is guaranteed to converge to a local maximum (or saddle point), but not necessarily the global one.

Why not?

Because the update rule depends on the initial value θ^0, the EM algorithm might:

Converge to the global maximum (if the initialization is lucky)

Get stuck in a local maximum or saddle point (more common)

At each iteration, the likelihood never decreases:

$$log\ \ p(X \mid \theta^{t+1}) \geq \log p(X \mid \theta^t)$$

n a Gaussian Mixture Model, if you initialize two components too close together, the EM algorithm may:

Assign most data to one component

Leave the other component "empty"

Converge to a poor local optimum

e)

The EM algorithm allows you to find a stationary point (a local maximum or a saddle point), but not necessarily the overall maximum of likelihood. Here is how to check if the final parameter actually maximizes the likelihood.

1. Compute the observed-data log-likelihood:

After convergence of EM at parameter θ^*, compute:

$$\log p(X \mid \theta^*)$$

This is the actual likelihood you care about (not Q, which approximates it).

2. Try multiple initializations:

Because EM is sensitive to initialization, it's common to run it multiple times from different starting points:

If all runs converge to the same likelihood, it's a good sign.

If some runs give a higher log-likelihood, then the current solution is not the maximum global.

3. Inspect the second derivative (optional):

To formally check whether θ* is a local maximum, analyze the Hessian matrix of the log-likelihood:

If the Hessian is negative definite, then θ^* is a local maximum.

If not, it might be a saddle point or local minimum.

But this is often hard to compute in complex models.