

Assignment 2: Expectation-Maximization Algorithm for Gaussian Mixture Model

1305026

Sajid Hasan Apon

1. Why should you use a Gaussian mixture model (GMM) in the above scenario?

Gaussian Mixture Model(GMM) is an **unsupervised clustering** method. Unsupervised clustering means finding out different categories in a collection of objects and assigning each object to a category.

Clustering presumes that the data are generated from a **mixture distribution**. Thus, the scenario described here is in fact a clustering problem. We know the number of clusters and the set of all points. But we do not know the actual parameters of the distributions that generated the data. What we are trying to do is cluster these points based on **maximizing the probability** and then find out the **centers** of these clusters

A Gaussian mixture model assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters and finds those parameters in a probabilistic way.

In this scenario, learning the actual position of the ships actually implies learning these parameters that generated the noisy sonar data (i.e., mean and covariance).

Hence, it is only logical that we use GMM in this scenario.

2. How will you model your data for GMM?

Data Generation:

We need the actual position of the ships to generate sample data points first. For this, we “fix” the positions of the ships first. These positions are the “means” of the components of the GMM which are then used to generate N data points. After generating these data, we pretend that we never knew the exact positions. In other words, we simply delete (forget) the means.

The procedure is as follow-

k = number of ships

N = number of total data points

W_i = proportion of data points corresponding to the i^{th} ship (component)

for $i = 1$ to k

μ_i = a 2×1 vector of random numbers

Σ_i = a 2×2 symmetric matrix of random numbers

$N_i = N \times W_i$

Generate N_i data points (2×1 vector) from normal(μ_i, Σ_i)

Include these data points in the dataset

Parameter Estimation:

We use the Expectation-Maximization(EM) algorithm to estimate the actual means used for generating the data we are working on. The EM algorithm has two steps – the “Expectation” step guesses the probability of datapoint \mathbf{x}_j being generated from component \mathbf{z}_j , and the “Maximization” step re-estimates the parameters.

The EM algorithm is iteratively performed for a fixed number of passes or until the log-likelihood converges.

3. What are the intuitive meaning of the update equations in **M step**?

The equations in the M step are

$$\mu_i = \frac{\sum_{j=1}^N p_{ij} \mathbf{x}_j}{\sum_{j=1}^N p_{ij}}$$

$$\Sigma_i = \frac{\sum_{j=1}^N p_{ij} (\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^T}{\sum_{j=1}^N p_{ij}}$$

$$w_i = \frac{\sum_{j=1}^N p_{ij}}{N}$$

i) μ_i is the expected mean of the i^{th} component.




The first equation is saying that the value of the mean is the weighted average of all the points, which is a straightforward idea. The weight, p_{ij} , is the probability that data point \mathbf{x}_j is generated from component \mathbf{z}_j .

ii) $(\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^T$ in the second equation is the scaler equivalent of $(x - \mu)^2$.

The $(m, n)^{\text{th}}$ element of Σ_i is equal to the covariance of the m^{th} and n^{th} features. The covariance matrix needs to be symmetric,

and $(\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^T$ is *indeed* symmetric

iii)

| | \mathbf{x}_1 | \mathbf{x}_2 | ... | \mathbf{x}_N |
|----------------|---|---|-----|---|
| \mathbf{z}_1 |  |  | |  |
| \mathbf{z}_2 | | | | |
| ... | | | | |
| \mathbf{z}_k | | | | |
| | 1 | 1 | | 1 |

In the above matrix the $(i, j)^{\text{th}}$ cell is the probability of x_j being generated from z_i . Summing over i along a column results in 1. Hence, all the cells here add up to N .

Hence, summing over j along a row thus gives the proportion of the i^{th} component among the N sample points.

The normalized weight is thus obtained by dividing this sum by N , since $\sum_{j=1}^N w_i = 1$.

4. Derive the log-likelihood function in step 4.

We want to maximize,

$$l = p(X) = \prod_{j=1}^N p(x_j)$$

Taking log on both sides, we get the log likelihood,

$$\begin{aligned} L &= \sum_{j=1}^N \log(p(x_j)) \\ &= \sum_{j=1}^N \log(N(x_j | \mu, \Sigma)) \end{aligned}$$

In case of multiple distributions, $p(x_j) = \sum_{i=1}^k w_i * p(x_j | \mu_i, \Sigma_i)$ where w_i is the overall weight of the i^{th} component.

Thus,

$$L = \sum_{j=1}^N \log \left(\sum_{i=1}^k w_i * N(x_j | \mu_i, \Sigma_i) \right)$$