An implementation of the EM algorithm for a mixture of Gaussian densities

Wagner L. Truppel

January 2001

Suppose we have n d-dimensional data points, $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, and c clusters. Let $\hat{M}_{ij}^{(s)}$ be the estimated membership weight associated with the data point $\boldsymbol{x}_i$ and the j-th cluster (*i.e.*, the estimated probability that the data point $\boldsymbol{x}_i$ belongs to the j-th cluster), obtained after s iterations of the algorithm, where $1 \leq i \leq n$ and $1 \leq j \leq c$. From an initial estimate of $\hat{p}_j$, $\hat{\boldsymbol{\mu}}_j$, and $\hat{\boldsymbol{\Sigma}}_j$, we compute the following quantities, in an iterative manner and in the order shown below:

$$
\begin{aligned}
\hat{\boldsymbol{z}}_{ij}^{(s)} &= \boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j^{(s)} && (s \geq 0, \quad 1 \leq i \leq n, \quad 1 \leq j \leq c)\,, \\
\hat{r}_{ij}^{(s)} &= \exp\Big[-\tfrac{1}{2}\, \hat{\boldsymbol{z}}_{ij}^{(s)} \cdot (\hat{\boldsymbol{\Sigma}}_j^{(s)})^{-1} \cdot \hat{\boldsymbol{z}}_{ij}^{(s)} \Big] && (s \geq 0, \quad 1 \leq i \leq n, \quad 1 \leq j \leq c)\,, \\
\hat{m}_{ij}^{(s)} &= |\hat{\boldsymbol{\Sigma}}_j^{(s)}|^{-1/2}\, \hat{r}_{ij}^{(s)}\, \hat{p}_j^{(s)} && (s \geq 0, \quad 1 \leq i \leq n, \quad 1 \leq j \leq c)\,, \\
\hat{q}_i^{(s)} &= \sum_{j=1}^{c} \hat{m}_{ij}^{(s)} && (s \geq 0, \quad 1 \leq i \leq n)\,, \\
\hat{\ell}^{(s)} &= \frac{1}{n} \sum_{i=1}^{n} \ln \hat{q}_i^{(s)} - \frac{1}{2}\, d \ln(2\pi) && (s \geq 0)\,, \\
\hat{M}_{ij}^{(s+1)} &= \hat{m}_{ij}^{(s)} \,/ \hat{q}_i^{(s)} && (s \geq 0, \quad 1 \leq i \leq n, \quad 1 \leq j \leq c)\,, \\
\hat{p}_j^{(s)} &= \frac{1}{n} \sum_{i=1}^{n} \hat{M}_{ij}^{(s)} && (s \geq 1, \quad 1 \leq j \leq c)\,, \\
\hat{\boldsymbol{\mu}}_j^{(s)} &= \frac{1}{n\, \hat{p}_j^{(s)}} \sum_{i=1}^{n} \boldsymbol{x}_i\, \hat{M}_{ij}^{(s)} && (s \geq 1, \quad 1 \leq j \leq c)\,, \\
\hat{\boldsymbol{\Sigma}}_j^{(s)} &= \frac{1}{n\, \hat{p}_j^{(s)}} \sum_{i=1}^{n} (\hat{\boldsymbol{z}}_{ij}^{(s)} \otimes \hat{\boldsymbol{z}}_{ij}^{(s)})\, \hat{M}_{ij}^{(s)} && (s \geq 1, \quad 1 \leq j \leq c)\,.
\end{aligned}
$$

Note that the computation of $\hat{\boldsymbol{\Sigma}}_j^{(s)}$ requires a re-computation of the $\hat{\boldsymbol{z}}_{ij}^{(s)}$ vectors.

The quantity $\hat{\ell}^{(s)}$ is the estimated log-likelihood per data point after the s-th iteration. Initialization of the algorithm requires an initial estimate of the cluster membership weights, the cluster mean vectors, and the cluster covariance matrices.

- Initialization of the cluster mean vectors is accomplished in one of two ways:
  - c distinct data points randomly chosen from the dataset, but not too close to each other (how close is a decision based on the overall *width* of the dataset);
  - the mean vectors returned by one run of the K-means algorithm (with $K = c$);
- Initialization of the cluster membership weights is then performed by partitioning the dataset according to the initial mean vectors;
- Finally, initialization of the cluster covariance matrices is done by computing the covariance matrix of the dataset within each cluster.

Two other requirements are:

- when to stop iterating:
  - Termination is enforced by two mechanisms: execution stops when (1) the log-likelihood increases by an amount less than δ times the value obtained in the previous iteration, that is, when $[\,\hat{\ell}^{(s+1)} - \hat{\ell}^{(s)}\,] < \delta\,|\hat{\ell}^{(s)}|$, for some user-defined value of δ, or when (2) the number of iterations exceeds a user-defined threshold, whichever happens first.
- how to deal with singular solutions:
  - Singular solutions are avoided by monitoring the diagonal components of the estimated covariance matrix at each iteration, forcing them to have a minimum value $\varepsilon\,\sigma^2$, where σ^2 is the variance (in the dataset) associated with the dimension for which the diagonal element of the covariance matrix is too small, and ε is a user-defined small positive number.

Another feature of the implementation is the option of specifying the form of the covariance matrices. The covariance matrices can be forced to be (1) proportional to the identity matrix, (2) diagonal, (3) any other, user-defined, form.

A random-restart version of the basic algorithm is also implemented, returning the solution with highest log-likelihood.

■