COMP 562 – Lecture 12

EM Algorithm for Mixture of Gaussians without Covariance

The model

$$p(h \mid \alpha) = \alpha_h$$
$$p(\mathbf{x} \mid h, \mu) = (2\pi)^{-\frac{d}{2}} \exp\left\{ -\frac{1}{2}(\mathbf{x} - \mu_{h_t})^T(\mathbf{x} - \mu_{h_t}) \right\}$$

is a variant of **Mixture of Gaussians**

α_c is an a-priori probability that a sample comes from class c -- also called **mixing proportion**

The bound

$$\begin{aligned}
\mathcal{B}(\Theta, q) &= \sum_{i=1}^{N} \sum_{h_i} q_i(h_i) \log \frac{p(\mathbf{x}_i, h_i \mid \Theta)}{q_i(h_i)} \\
&= \sum_{i=1}^{T} \sum_{h_i} q_i(h_i) \left[\log \alpha_{h_i} - \frac{d}{2}\log(2\pi) - \frac{1}{2}(\mathbf{x} - \mu_{h_i})^T(\mathbf{x} - \mu_{h_i}) \right] \\
&- \sum_{i=1}^{T} \sum_{h_i} q_i(h_i) \log q_i(h_i)
\end{aligned}$$

In this case $\Theta = (\alpha_1, \ldots, \alpha_K, \mu_1, \ldots, \mu_K)$

Mixture of Gaussians without Covariance -- E-step

In E-step we optimize qs given Θ

$$q_i^{\text{new}} = \underset{q_i}{\operatorname{argmax}} \, \mathcal{B}(\Theta^{\text{old}}, q)$$

In general, we can take derivatives, equate them to zero, and solve:

$$\nabla_{q_i} \mathcal{B}(\Theta^{\text{old}}, q) = 0$$

We can show that in our case, the E-step updates are:

$$\begin{aligned} q_i(h_i = k) = p(h_i = k \mid \mathbf{x}_i, \mu) = \frac{p(\mathbf{x}_i, h_i = k \mid \mu)}{\underbrace{\sum_c p(\mathbf{x}_i, h_i = c \mid \mu)}_{\text{same for all values of } k}} \\ \propto p(\mathbf{x}_i, h_i = k \mid \mu) = p(h_i = k \mid \alpha) p(\mathbf{x} \mid h_i = k, \mu) \\ = \alpha_{h_i} (2\pi)^{-\frac{d}{2}} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \mu_h)^T (\mathbf{x} - \mu_h) \right\} \end{aligned}$$

Mixture of Gaussians without Covariance -- M-step

In M-step we optimize Θ given qs

$$\Theta^{\text{new}} = \underset{\Theta}{\operatorname{argmax}} \, \mathcal{B}(\Theta, q^{\text{new}})$$

In general, we can take derivatives, equate them to zero, and solve:

$$\nabla_{\Theta} \mathcal{B}(\Theta, q^{\text{new}}) = 0$$

We can show that in our case, the M-step updates are:

$$\mu_k^* = \frac{\sum_i q_i(h_i = k) \mathbf{x}_i}{\sum_i q_i(h_i = k)}$$

$$\alpha_k^* = \frac{\sum_i q_i(h_i = k)}{N}$$

EM Algorithm for Mixture of Gaussians with Covariance

The model

$$p(h \mid \alpha) = \alpha_h$$
$$p(\mathbf{x} \mid h, \mu) = (2\pi)^{-\frac{d}{2}} |\Sigma_h|^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2}(\mathbf{x} - \mu_{h_i})^T \Sigma_h^{-1} (\mathbf{x} - \mu_{h_i}) \right\}$$

This is also a variant of **Mixture of Gaussians** Note that we introduced a covariance matrix per cluster

- Hidden variables: h_i -- cluster membership for sample i
- Parameters: $\Theta = (\underbrace{\alpha_1, \ldots, \alpha_K}_{\text{proportions}}, \underbrace{\mu_1, \ldots, \mu_K}_{\text{means}}, \underbrace{\Sigma_1, \ldots, \Sigma_K}_{\text{covariances}})$

EM Algorithm for Mixture of Gaussians with Covariance

We plug-in probabilities $p(\mathbf{x}_i \mid h_i, \Theta)$ and $p(h_i \mid \alpha)$ in the bound

$$\begin{aligned}
\mathcal{B}(\Theta, q) \quad &= \sum_{i=1}^{N} \sum_{h_i} q_i(h_i) \log \frac{p(\mathbf{x}_i, h_i \mid \Theta)}{q_i(h_i)} \\
&= \sum_{i=1}^{N} \sum_{h_i} q_i(h_i) \Big[\log \alpha_{h_i} - \frac{d}{2} \log(2\pi) - \frac{1}{2} \log |\Sigma_{h_i}| \\
&\qquad - \frac{1}{2}(\mathbf{x}_i - \mu_{h_i})^T \Sigma_{h_i}^{-1} (\mathbf{x}_i - \mu_{h_i}) \Big] \\
&- \sum_{i=1}^{N} \sum_{h_i} q_i(h_i) \log q_i(h_i)
\end{aligned}$$

Mixture of Gaussians with Covariance -- E-step

$$q_i(h_i = k) = p(h_i = k \mid \mathbf{x}_i, \mu) = \underbrace{\frac{p(\mathbf{x}_i, h_i = k \mid \mu)}{\sum_c p(\mathbf{x}_i, h_i = c \mid \mu)}}_{\text{same for all values of } k}$$

$$\propto p(\mathbf{x}_i, h_i = k \mid \mu) = p(h_i = k \mid \alpha) p(\mathbf{x} \mid h_i = k, \mu)$$

$$= \alpha_{h_i} (2\pi)^{-\frac{d}{2}} |\Sigma_{h_i}^{-1}| \exp\left\{ -\frac{1}{2} (\mathbf{x} - \mu_{h_i})^T \Sigma_{h_i}^{-1} (\mathbf{x} - \mu_{h_i}) \right\}$$

Implementation:

```
q = numpy.zeros((K,N))           # clusters x samples
q = logjointp(x,Theta)           # compute all joints at once
loglik = numpy.sum(logsumexp(q)) # compute loglikelihood
q = q - logsumexp(q)             # normalizing across clusters
```

Mixture of Gaussians with Covariance -- M-step

Updates for parameters of prior probability $p(h \mid \alpha)$

$$\alpha_k^* = \frac{\sum_i q_i(h_i = k)}{N}$$

Updates for means of clusters

$$\mu_k^* = \frac{\sum_i q_i(h_i = k) \mathbf{x}_i}{\sum_i q_i(h_i = k)}$$

Updates for covariances of clusters

$$\Sigma_k^* = \frac{\sum_i q_i(h_i = k)(\mathbf{x}_i - \mu_k^*)(\mathbf{x}_i - \mu_k^*)^T}{\sum_i q_i(h_i = k)}$$

Debugging EM Algorithm

1. Log-likelihood should always go up!
2. Synthetic data is your friend, if you generate data from your model you get samples and cluster membership
3. E-step computes cluster membership based on parameters. Use this!
    - Synthesize data from ground truth parameters
    - Start your EM from ground truth parameters, not random initialization
    - Does your E step associate samples with correct clusters?
    - Select one sample and look at its posterior probability for the cluster it came from
4. M-step updates parameters based on cluster membership. Use this!
    - Using synthetic data, set `q` to be one-hot according to ground truth
    - Start your M-step with this `q`
    - If you don't get parameters back that are close to the ground truth
    - To isolate a broken update, let M-step update just one parameter (for example mus)
5. Starting your EM with ground truth parameters should not budge too much

Between these tricks you should be able to isolate source of your problem