STAT 383C: Statistical Modeling I **Fall 2016**

Lecture 16 — October 27

Lecturer: Purnamrita Sarkar *Scribe: Giorgio Paulon and Jennifer Starling*

Disclaimer: These scribe notes have been slightly proofread and may have typos etc.

Note: The latex template was borrowed from EECS, U.C. Berkeley.

16.1 Gaussian Mixture Models

Let us suppose that, unlike in LDA, we now observe only the features of each data point $\boldsymbol{x}_i$ and not the corresponding class label, which we will denote as z_i. We assume that the data, given the clustering allocation, follow the model

$$\begin{aligned}
\boldsymbol{X}_i | Z_i = 1 &\sim \mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1) \\
\boldsymbol{X}_i | Z_i = 0 &\sim \mathcal{N}(\boldsymbol{\mu}_2, \Sigma_2) \\
P(Z_i = 1) &= \pi
\end{aligned}$$

The parameters of interest are $\boldsymbol{\theta} = \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \Sigma_1, \Sigma_2, \pi\}$. The **observed** data likelihood function $L(\boldsymbol{x}; \boldsymbol{\theta})$ is given by

$$\begin{aligned}
L(\boldsymbol{x}; \boldsymbol{\theta}) &= \prod_{i=1}^{n} f(\boldsymbol{x}_i; \boldsymbol{\theta}) \\
&= \prod_{i=1}^{n} \sum_{z_i} f(\boldsymbol{x}_i, \boldsymbol{z}_i; \boldsymbol{\theta}) \\
&= \prod_{i=1}^{n} \sum_{z_i} f(\boldsymbol{x}_i | \boldsymbol{z}_i; \boldsymbol{\theta}) P(Z_i = z_i) \\
&= \prod_{i=1}^{n} \{\phi_1(\boldsymbol{x}_1)\pi + \phi_2(\boldsymbol{x}_2)(1-\pi)\} \\
l(\boldsymbol{x}; \boldsymbol{\theta}) &= \sum_{i=1}^{n} \log[\phi_1(\boldsymbol{x}_1)\pi + \phi_2(\boldsymbol{x}_2)(1-\pi)]
\end{aligned}$$

Note that:

- ϕ_1 = density of $N(\mu_1, \Sigma_1)$
- ϕ_2 = density of $N(\mu_2, \Sigma_2)$

This is a hard and non-convex problem, which leads to many local optima. It is easier to think of the **augmented** data likelihood.

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta}) &= \prod_{i=1}^{n} \{f(\boldsymbol{x}_i|z_i; \boldsymbol{\theta}) p(z_i; \boldsymbol{\theta})\} \\ &= \prod_{i=1}^{n} \left\{\phi_1(\boldsymbol{x}_i)^{z_i} \phi_2(\boldsymbol{x}_i)^{1-z_i} \pi^{z_i} (1-\pi)^{1-z_i}\right\} \\ l(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta}) &= \sum_{i=1}^{n} \{z_i \log \phi_1(\boldsymbol{x}_i) + (1-z_i) \log \phi_2(\boldsymbol{x}_i) + z_i \log(\pi) + (1-z_i) \log(1-\pi)\} \end{aligned}$$

If we knew the latent variables z_i we would be in the exact framework of LDA. We can in this case replace the z_i's with their expected values, say γ_i. This iterative approach is divided in two steps, which are collectively called the **E-M Algorithm**.

- **E-Step** (Expectation Step):
  This step calculates the γ_i's, which are the expected value of the z_i's given the data and the current iteration's θ estimates.

$$\begin{aligned} \gamma_i^{(m+1)} &= E[z_i = 1|\boldsymbol{x}_i; \theta^{(m)}] = P(z_i = 1|\boldsymbol{x}_i; \theta^{(m)}) \\ &= \frac{f(\boldsymbol{x}_i|z_i = 1; \theta^{(m)}) P(z_i = 1; \theta^{(m)})}{f(\boldsymbol{x}_i; \theta^{(m)})} \\ &= \frac{\hat{\phi}_1^{(t)}(\boldsymbol{x}_i) \hat{\pi}^{(t)}}{\hat{\phi}_1^{(t)}(\boldsymbol{x}_i) \hat{\pi}^{(t)} + \hat{\phi}_2^{(t)}(\boldsymbol{x}_i)(1 - \hat{\pi}^{(t)})} \end{aligned}$$

- **M-Step** (Maximization Step):
  We can now compute the estimates of all of the parameters using γ_i instead of z_i by solving the following maximization problem

$$\text{argmax}_{\theta} E_{\boldsymbol{Z} \sim P(\boldsymbol{Z}|\boldsymbol{x}; \boldsymbol{\theta})}[l(\boldsymbol{x}, \boldsymbol{z}, ; \theta)].$$

  This is the maximization of an averaged version of the log-likelihood when $Z\ simP(\boldsymbol{Z}|\boldsymbol{x}; \boldsymbol{\theta})$. The resulting estimates are:

$$\hat{\pi} = \frac{\sum_{i=1}^{n} \gamma_i}{n}$$

$$\hat{\boldsymbol{\mu}}_1 = \frac{\sum_{i=1}^{n} \gamma_i \boldsymbol{x}_i}{\sum_{i=1}^{n} \gamma_i}; \qquad \hat{\boldsymbol{\mu}}_2 = \frac{\sum_{i=1}^{n} (1-\gamma_i) \boldsymbol{x}_i}{\sum_{i=1}^{n} (1-\gamma_i)}$$

$$\hat{\Sigma}_1 = \frac{\sum_{i=1}^{n} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_1)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_1)^T \gamma_i}{\sum_{i=1}^{n} \gamma_i}; \qquad \hat{\Sigma}_2 = \frac{\sum_{i=1}^{n} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_2)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_2)^T (1-\gamma_i)}{\sum_{i=1}^{n} (1-\gamma_i)}.$$

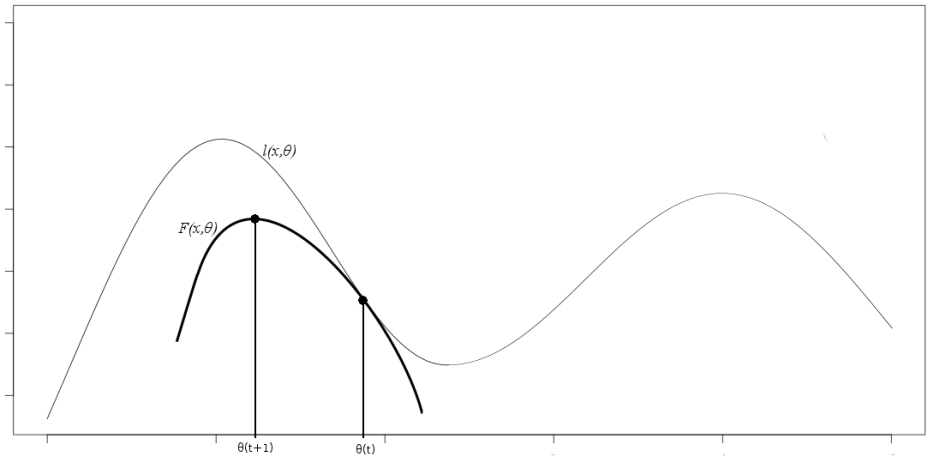

Figure 16.1. Illustration of the Expectation Maximization algorithm.

16.2 Explaining the inner math of the E-M algorithm

16.2.1 Optimization Strategy

It is very difficult to maximize the likelihood $l(\boldsymbol{x};\boldsymbol{\theta})$ when the model is a mixture. So the strategy here (see Figure 16.1) is to find a function $F(\boldsymbol{x};\boldsymbol{\theta}^{(t)})$ that is a lower bound for the observed data likelihood, and that assumes the same value in $\theta^{(t)}$ but that can be easily maximized.

Its maximum will be the next $\theta^{(t+1)}$. To see that the iteration process really works, consider

$$l(\boldsymbol{x};\boldsymbol{\theta}^{(t+1)}) > F(\boldsymbol{x};\boldsymbol{\theta}^{(t+1)}) > F(\boldsymbol{x};\boldsymbol{\theta}^{(t)}) = l(\boldsymbol{x};\boldsymbol{\theta}^{(t)}).$$

This proves that at each step we are increasing the log-likelihood (but we may be moving toward a local maximum).

16.2.2 How to find the lower bound

First recall that by Jensen's inequality $\log[\mathrm{E}[\cdot]] \geq \mathrm{E}[\log(\cdot)]$. Then a candidate function which is bounded by the likelihood can be:

$$\begin{aligned} l(\boldsymbol{x};\boldsymbol{\theta}^{(t)}) &= \log(f(\boldsymbol{x};\boldsymbol{\theta}^{(t)})) = \log\sum_{Z} f(\boldsymbol{x},Z;\boldsymbol{\theta}^{(t)}) = \log\sum_{Z} P(Z)\frac{f(\boldsymbol{x},Z;\boldsymbol{\theta}^{(t)})}{P(Z)} \\ &\geq \sum_{Z} P(Z)\log\frac{f(\boldsymbol{x},Z;\boldsymbol{\theta}^{(t)})}{P(Z)} \end{aligned}$$

As we see now, we can get many different lower bounds for different choices of $P(Z)$. All we need is for $P(Z)$ to be a valid distribution on the latent variable Z. We will now show that if we pick $P(Z) = P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)})$ the lower bound is optimal:

$$\begin{aligned}
F(\boldsymbol{x}; \boldsymbol{\theta}^{(t)}) &= \sum_Z P(Z) \log \frac{f(\boldsymbol{x}, Z; \boldsymbol{\theta}^{(t)})}{P(Z)} \\
&= \sum_Z P(Z) \log \frac{P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)}) f(\boldsymbol{x}; \boldsymbol{\theta}^{(t)})}{P(Z)} \\
&= \sum_Z P(Z) \log f(\boldsymbol{x}; \boldsymbol{\theta}^{(t)}) - \sum_Z P(Z) \log \frac{P(Z)}{P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)})} \\
&= \log f(\boldsymbol{x}, \boldsymbol{\theta}^{(t)}) - D_{KL}(P(Z), P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)})) \\
&= \log f(\boldsymbol{x}, \boldsymbol{\theta}^{(t)}) \qquad \text{if } P(Z) = P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)}).
\end{aligned}$$

Here $D_{KL}(P(Z), P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)}))$ is the Kullback-Leibler divergence between the distributions $P(Z)$ and $P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)})$. You can think of this as a asymmetric distance measure between two distributions which is minimized at 0 when the distributions are identical.

Therefore, the choice of $P(Z) = P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)}$ is optimal and the value of the lower bound is the log-likelihood evaluated at the current point $\boldsymbol{\theta}^{(t)}$.

To sum up, the EM algorithm can be summarized by the following iterative procedure:

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \text{argmax}_{\boldsymbol{\theta}} E_{Z \sim P(Z|\boldsymbol{x}; \boldsymbol{\theta}^{(t)})}[\log f(\boldsymbol{x}, Z; \boldsymbol{\theta})].$$

16.3 How to Choose Starting Parameter Values

The E-M algorithm can get stuck in local maxima, and is a bit sensitive to the choice of starting guesses for $\boldsymbol{\theta} = \{\boldsymbol{\mu_1}, \boldsymbol{\mu_2}, \Sigma_1, \Sigma_2, \pi\}$.

Hastie and Tibshirani (Elements of Statistical Learning, pg. 293) recommend constructing initial guesses as follows:

- For $\hat{\mu_1}$ and $\hat{\mu_2}$, randomly select two y_i values.
- For $\hat{\Sigma_1^2}$ and $\hat{\Sigma_2^2}$, set both equal to the overall sample variance $\sum_{i=1}^n (y_i - \bar{y})(y_i - \bar{y})^T / n$.
- For $\hat{\pi}$, begin at 0.50.

In practice, the E-M algorithm is often run using several different combinations of starting parameter estimates. This prevents relying on one set of starting parameters that may get stuck in a local max.