STAT40810 — Stochastic Models

Brendan Murphy

Week 11

Mixture Models: Miscellaneous

Fitting Mixtures

- In principle, we could fit a mixture model using maximum likelihood and numerical optimization (as seen earlier in the course).
- We have a probability density/mass function and can form a likelihood function.
- We could maximize the log-likelihood to get parameter estimates.
- In practice, this is not a commonly used approach.
- Usually, the EM algorithm is utilized for model fitting.

Mixture Model Fitting

- The mixture model is of the form

$$p(x|\tau, \theta, G) = \sum_{g=1}^{G} \tau_g p(x|\theta_g).$$

- The likelihood function for a mixture model is of the form

$$L = \prod_{i=1}^{n} \sum_{g=1}^{G} \tau_g p(x_i|\theta_g).$$

- The log-likelihood is of the form

$$\ell = \sum_{i=1}^{n} \log \left[\sum_{g=1}^{G} \tau_g p(x_i|\theta_g) \right].$$

Data Augmentation

- Suppose knew the component membership of each observation.
- Let $z_{ig} = 1$ if observation i belongs to component g and $z_{ih} = 0$ for $h \neq g$.
- In this case, the likelihood would be

$$L_c = \prod_{i=1}^{n} \prod_{g=1}^{G} \left[\tau_g p(x_i|\theta_g)\right]^{z_{ig}} .$$

  We call this the *complete-data* likelihood or the *completed* likelihood or the *augmented* likelihood.

- The complete-data log-likelihood would then be

$$\ell_c = \sum_{i=1}^{n} \log \prod_{g=1}^{G} \left[\tau_g p(x_i|\theta_g)\right]^{z_{ig}} = \sum_{i=1}^{n} \sum_{g=1}^{G} z_{ig} \log \left[\tau_g p(x_i|\theta_g)\right] .$$

Data Augmentation

- The complete-data log-likelihood can be written as

$$\begin{aligned}\ell_c &= \sum_{i=1}^{n}\sum_{g=1}^{G} z_{ig} \log\left[\tau_g p(x_i|\theta_g)\right] \\ &= \sum_{i=1}^{n}\sum_{g=1}^{G} z_{ig} \log \tau_g + \sum_{i=1}^{n}\sum_{g=1}^{G} z_{ig} \log p(x_i|\theta_g).\end{aligned}$$

- Maximizing l_c is, in principle, straightforward.
- In practice, it is the same difficulty as fitting a single model.

EM Algorithm

- However, we don't know the z_{ig} values!
- The EM algorithm gives us a way to get around this.
- It works as follows:
  0. Let $t = 0$ and choose initial parameter values $(\tau^{(t)}, \Theta^{(t)})$.
  1. **Expectation step** (E-step): Replace z_{ig} values with $\mathbb{E}(z_{ig}|x_i, \tau^{(t)}, \Theta^{(t)})$.
  2. **Maximization step** (M-step): Maximize l_c with the current z_{ig} values plugged in. Call the new parameter estimates $(\tau^{(t+1)}, \Theta^{(t+1)})$
  3. Check for convergence. If converged, stop. If not, increment t and return to step 1.
- The algorithm is guaranteed to have the following ascent property

$$\ell(\tau^{(t+1)}, \Theta^{(t+1)}) \geq \ell(\tau^{(t)}, \Theta^{(t)})$$

EM Algorithm

- The calculation in the E-step reduces to:

$$\mathbb{E}(z_{ig}|x_i, \tau^{(t)}, \Theta^{(t)}) = \frac{\tau_g p(x_i|\theta_g)}{\sum_{h=1}^{G} \tau_h p(x_i|\theta_h)}.$$

- The calculation in the M-step reduces to:

$$\tau_g^{(t+1)} = \frac{\sum_{i=1}^{n} z_{ig}}{n}$$

and finding a $\Theta^{(t+1)}$ that maximizes

$$\sum_{i=1}^{n} \sum_{g=1}^{G} z_{ig} \log p(x_i|\theta_g).$$

EM Algorithm

- The EM algorithm is used extensively because:
  - Stability
  - Ascent property
  - Relatively easy extension of single model inference
- However:
  - It tends to be slower than numerical optimization methods (eg. Newton Raphson).
  - It doesn't readily give the information required for standard error estimation.