Bayesian Learning - Lecture 7

Mattias Villani

Division of Statistics
Department of Computer and Information Science
Linköping University

Lecture overview

- Exchangeability
- Hierarchical models

Exchangeability

- Let $\mathcal{K} = \{k_1, k_2, ..., k_n\}$ be a permutation of $\mathcal{L} = \{1, 2, ..., n\}$.
- Example: $\mathcal{L} =\{1, 2, 3\}$. Permutations: $\mathcal{K} = \{2, 1, 3\}$, $\mathcal{K} = \{2, 3, 1\}$, $\mathcal{K} = \{3, 2, 1\}$ ...
- $X_1, X_2, ..., X_n$ are *exchangeable* stochastic variables if $X_{k_1}, X_{k_2}, ..., X_{k_n}$ has the same joint distributions for all $n!$ permutations of $\{k_1, k_2, ..., k_n\}$.
- Example: $n = 2$. X_1, X_2 are exchangeable if $p(X_1 = a, X_2 = b) = p(X_2 = a, X_1 = b)$.

Exchangeability, cont.

- *iid* $\Rightarrow$ Exchangeability, but the opposite does not always hold. Exchangeability is less restrictive than *iid*.
- Example: Urn with m marbles: r white and $m-r$ black. Draw $n \leqslant m$ marbles without replacement

$$X_i = \begin{cases} 1 \text{ if } i\text{th draw gives black marble} \\ 0 \text{ if } i\text{th draw gives white marble} \end{cases}, \; i = 1, ..., n$$

$X_1, ..., X_n$ are exchangeable, but not *iid*.

AN HIERARCHICAL BINOMIAL MODEL

- Example:
$$y_j|\theta_j \sim Bin(n_j, \theta_j),\ j = 1, ..., J.$$
- We could do inference on each θ_j separately. Problem: n_j may be small for some j. Not much info then about θ_j.
- If you knew θ_j, would that give information about θ_i, $i \neq j$? If so, then inference about the parameters θ_j, $j = 1, ..., J$, may 'borrow strength' from each other.
- Extreme case: assume $\theta_j = \theta$ for all j. Define $y = \sum_{j=1}^{J} y_j$ and $n = \sum_{j=1}^{J} n_j$. Straightforward to analyze θ with the usual Beta-Binomial approach.
- Intermediate case: tie the θ's together by assuming a superpopulation/prior
$$\theta_j \overset{iid}{\sim} Beta(\alpha, \beta).$$

An hierarchical binomial model, cont.

- Model summary

$$y_j|\theta_j \sim Bin(n_j, \theta_j),\ j = 1, ..., J.$$
$$\theta_j \overset{iid}{\sim} Beta(\alpha, \beta).$$
$$\alpha \sim Gamma(a_1, a_2).$$
$$\beta \sim Gamma(b_1, b_2).$$

- Sample from the joint posterior of $p(\theta, \alpha, \beta|y) = p(\theta|\alpha, \beta, y)p(\alpha, \beta|y)$ by sampling from:
  - $\theta_j|\alpha, \beta, y$, $j = 1, ..., J$, which are independent *Beta* distributions.
  - $p(\alpha, \beta|y)$ can be derived in closed form (similar to eq. 5.8), but cannot be sampled directly. Evaluate on grid and sample.

WinBugs code for the hierarchical binomial model

WinBugs code for the hierarchical binomial model

```
model{
        for( j in 1 : J ) {
                y[j]~dbin(theta[j],n[j])
                theta[j]~dbeta(alpha,beta)
        }
        alpha~dgamma(a1,a2)
        beta~dgamma(b1,b2)
}
```

The one-way normal random effects model

- Consider the data model:

$$y_j|\theta_j \sim N(\theta_j, \sigma_j^2), \ \ \sigma_j^2 \text{ known}$$

- At one extreme we may: estimate each θ_j using the mean $\bar{y}_j$ of observations in the jth group.
- At the other extreme we may: assume $\theta_j = \theta$, for all j. Estimate θ with a pooling of group means $\bar{y}_j$.

The one-way normal random effects model, cont.

- Intermediate: a hierarchical model

$$y_j|\theta_j \sim N(\theta_j, \sigma_j^2), \;\; \sigma_j^2 \text{ known}$$

$$\theta_j|\mu, \tau \sim N(\mu, \tau^2)$$

$$p(\mu, \tau) = p(\mu|\tau)p(\tau) \propto p(\tau)$$

- Here we do not assume equal group mean, yet the estimates of each θ_j borrow strength from each other.

WinBugs code for normal hierarchical model

Normal hierarchical model with known data variances from Gelman et al. (2004). Note: WinBugs and Gelman et al. (2004) uses different notation for variances, hence the mismatch between mathematical and graphical model.

WinBugs code for the normal hierarchical model

```
model {
      for (j in 1:J){
            y[j]~dnorm (theta[j], tau.y[j])
            tau.y[j] <- pow(sigma.y[j], -2)    # tau = 1/sigma^2
            theta[j]~dnorm (mu.theta, tau.theta)
      }
 tau.theta <- pow(sigma.theta, -2)
 mu.theta~dnorm (0.0, 1.0E-6)
 sigma.theta~dunif (0, 1000)
}
```