统计中的计算方法 · 课后作业（3）

梁子龙（15300180026）

2018 年 4 月 29 日

作业 1 假设 HMM 的隐状态为 (F, B)，显示状态为 (H, T)，状态转移概率矩阵为

$$\begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix},$$

状态显示概率矩阵为

$$\begin{array}{c} \\ F \\ B \end{array} \begin{array}{c} \begin{array}{cc} H & T \end{array} \\ \begin{pmatrix} 0.5 & 0.5 \\ 0.8 & 0.2 \end{pmatrix} \end{array}$$

观察数据 $X = (x_1, x_2, x_3, x_4) = (H, T, T, T)$，用向前方法和向后方法计算观察数据出现的概率.

答. 首先利用向前方法计算数据出现的概率. 我们这里约定记号 $\alpha_t(j)$ 表示 t 时刻时在隐状态 j 下得到前 t 个观察的概率，a_{ij} 表示状态 i 转移到状态 j 的转移概率，$b_j(o_t)$ 表示状态 j 下得到观察 o_t 的发射概率. 依照向前方法，观察数据出现的概率可以递归地写为

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t).$$

对这一问题，具体计算得

$$\mathrm{P}(X) = \alpha_4(F) + \alpha_4(B) = 0.029.$$

之后，利用向后方法计算该概率. 令约定记号 $\beta_t(j)$ 表示 t 时刻时在隐状态 j 下得到 t 之后的观察的概率. 依照向后方法，观察数据出现的概率同样可以递归地写为

$$\beta_{t-1}(i) = \sum_{j=1}^{N} a_{ij} b_j(o_t) \beta_t(j).$$

具体计算得

$$\mathrm{P}(X) = \frac{1}{2} b_F(H) \beta_1(F) + \frac{1}{2} b_B(H) \beta_1(B) = 0.029.$$

作业 2 假设同题目 *1*，计算对 $X_3 = T$ 对应的隐状态为 B 的概率.

答. 利用向前-向后方法，计算得 $X_3 = T$ 对应得隐状态为 B 的概率 $\gamma_3(B)$ 为

$$\begin{aligned}\gamma_3(B) &= \frac{\alpha_3(B)\beta_3(B)}{\mathrm{P}(X)} \\ &= \frac{0.021 \times 0.32}{0.029} \\ &= 0.232.\end{aligned}$$

作业 3 假设同题目 *1*，计算最优的隐状态路径.

答. 脚本 `assignment03-script.R` 实现了隐 Markov 模型的基本算法．将题目 1 的条件输入，利用 Viterbi 算法计算得最优隐状态路径为

$$Y = (B, F, F, F).$$

作业 4 假设 *HMM* 隐状态为 (A, B)，显示状态为 (L, R)，对附件数据 *assignment03-data.csv*，估计 *HMM* 的参数，并估计最优隐状态路径.

答. 这里由于没有提供 A 或 B 的任何信息，因此两者的顺序在这里不加以区分．利用程序脚本中的 Baum-Welch 算法迭代若干次后，得到所估计出的参数如下：

$$\text{状态转移矩阵}\quad P = \begin{array}{c} \\ A \\ B \end{array}\!\!\begin{array}{c} \begin{array}{cc} \;A\; & \;\;B\; \end{array} \\ \begin{pmatrix} 0.909 & 0.091 \\ 0.268 & 0.732 \end{pmatrix} \end{array}$$

$$\text{发射矩阵}\quad E = \begin{array}{c} \\ A \\ B \end{array}\!\!\begin{array}{c} \begin{array}{cc} \;L\; & \;\;R\; \end{array} \\ \begin{pmatrix} 0.298 & 0.702 \\ 0.762 & 0.238 \end{pmatrix} \end{array}$$

$$\text{初始概率}\quad \pi = \begin{array}{c} \begin{array}{cc} \;A\; & \;\;B\; \end{array} \\ \begin{pmatrix} 1.000 & 0.000 \end{pmatrix} \end{array}$$

再利用 Viterbi 算法得到隐状态路径．在该路径当中，A 出现 885 次，而 B 出现 115 次． □

附录：程序实现中的取对数技术　当观察数据较多时，由向前、向后方法所得到的概率有下溢的可能．于是，在程序实现当中应采用取对数技术．这里参考了 CRAN 中 HMM 库的做法．比如在向前算法中，假设已得到 $\alpha_{t-1}(i)$，那么下一步可以写为

$$\begin{aligned}\log \alpha_t(j) &= \log\left(\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)\right) \\ &= \log b_j(o_t) + \log\left(\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right).\end{aligned}$$

现要求右边一项，取

$$
\begin{aligned}
& s_1 = \log \alpha_{t-1}(1) + a_{1j}, \\
& s_2 = \log \alpha_{t-1}(2) + \log a_{2j} + \log\left(1 + \exp\left(s_1 - \log \alpha_{t-1}(2) - \log a_{2j}\right)\right), \\
& s_3 = \log \alpha_{t-1}(3) + \log a_{3j} + \log\left(1 + \exp\left(s_2 - \log \alpha_{t-1}(3) - \log a_{3j}\right)\right), \\
& \dots
\end{aligned}
$$

即可归纳地得到上式的右边一项．这一技术本质上是在每一次计算时将指数部分提出取对数，将乘法运算转换为加法，以避免出现指数部分过大的情况．该技术同样可以用在向后方法的计算中，从而规避下溢问题．