Math!

Jimmy Kelliher

2022-04-01

2.2 Newton-Raphson Algorithm

The Newton-Raphson method is an efficient algorithm for computing the maximum likelihood estimator (MLE) under certain conditions. However, to even consider computing an MLE, we need to choose a reasonable model for our data so that we can construct our likelihood function.

Toward that aim, let $\mathbf{X} \equiv (x_{ij})$ denote the $n \times p$ matrix corresponding to our $n = 569$ observations and our $p = 21$ features (including an intercept term). Because our outcome $\mathbf{Y} \equiv (y_1, \ldots, y_n)^t$ is binary, it is natural to model $y_i \,|\, x_{i1}, \ldots, x_{ip} \sim \text{Bernoulli}(\pi_i)$, where

$$\begin{aligned}
\pi_i &\equiv P(y_i = 1 \,|\, x_{i1}, \ldots, x_{ip}) \\
&= \left(1 + \exp\left(-\sum_{j=1}^{p} \beta_j x_{ij} \right) \right)^{-1}, \quad \text{such that} \\
&\log\left(\frac{\pi_i}{1 - \pi_i} \right) = \sum_{j=1}^{p} \beta_j x_{ij}
\end{aligned}$$

for a vector of parameters $\boldsymbol{\beta} \in \mathbb{R}^p$. The likelihood function of $\boldsymbol{\beta}$ given our data is

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\beta} \,|\, \mathbf{X}, \mathbf{Y}) &\equiv f(\mathbf{X}, \mathbf{Y} \,|\, \boldsymbol{\beta}) \\
&= \prod_{i=1}^{n} \pi_i^{y_i} (1 - \pi_i)^{1 - y_i} \\
&= \prod_{i=1}^{n} \left(\frac{\pi_i}{1 - \pi_i} \right)^{y_i} (1 - \pi_i).
\end{aligned}$$

To employ the Newton-Raphson procedure, we need to compute the gradient vector and hessian matrix corresponding to $\mathcal{L}$. However, as computing derivatives of the likelihood function is tedious, we instead consider the log-likelihood function

$$\begin{aligned}
l(\boldsymbol{\beta} \,|\, \mathbf{X}, \mathbf{Y}) &\equiv \log \mathcal{L}(\boldsymbol{\beta} \,|\, \mathbf{X}, \mathbf{Y}) \\
&= \log \left(\prod_{i=1}^{n} \left(\frac{\pi_i}{1 - \pi_i} \right)^{y_i} (1 - \pi_i) \right) \\
&= \sum_{i=1}^{n} \left(y_i \log\left(\frac{\pi_i}{1 - \pi_i} \right) - \log\left(\frac{1}{1 - \pi_i} \right) \right) \\
&= \sum_{i=1}^{n} \left(y_i \sum_{j=1}^{p} \beta_j x_{ij} - \log\left(1 + \exp\left(\sum_{j=1}^{p} \beta_j x_{ij} \right) \right) \right).
\end{aligned}$$

Toward computing the derivative of the l, we define

$$\eta_i \equiv \sum_{j=1}^{p} \beta_j x_{ij}$$

for each $i \in \{1, \ldots, n\}$. Observe that for each $k \in \{1, \ldots, p\}$,

$$\frac{\partial \eta_i}{\partial \beta_k} = x_{ik} \quad \text{and} \quad \frac{\partial \pi_i}{\partial \beta_k} = x_{ik}\pi_i(1 - \pi_i).$$

Thus, it follows that

$$\begin{aligned}
\frac{\partial l}{\partial \beta_k} &= \frac{\partial}{\partial \beta_k} \left(\sum_{i=1}^{n} \left(y_i \eta_i - \log\left(1 + e^{\eta_i}\right) \right) \right) \\
&= \sum_{i=1}^{n} \left(y_i \frac{\partial \eta_i}{\partial \beta_k} - \frac{e^{\eta_i}}{1 + e^{\eta_i}} \frac{\partial \eta_i}{\partial \beta_k} \right) \\
&= \sum_{i=1}^{n} x_{ik}(y_i - \pi_i), \quad \text{and}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial l^2}{\partial \beta_k \beta_l} &= \frac{\partial}{\partial \beta_l} \left(\sum_{i=1}^{n} x_{ik}(y_i - \pi_i) \right) \\
&= -\sum_{i=1}^{n} x_{ik} \frac{\partial \pi_i}{\partial \beta_l} \\
&= -\sum_{i=1}^{n} x_{ik} x_{il} \pi_i (1 - \pi_i).
\end{aligned}$$

for $k, l \in \{1, \ldots p\}$. The above expressions completely characterize the gradient vector and hessian matrix corresponding to the log-likelihood l. However, it will be convenient to express these objects compactly via matrix notation. Toward this aim, we define $\boldsymbol{\pi} \equiv (\pi_1, \ldots, \pi_n)^t$ to be a vector of probabilities and

$$\mathbf{W} \equiv \begin{pmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_n \end{pmatrix}, \quad \text{where} \quad w_i \equiv \pi_i(1 - \pi_i)$$

for each $i \in \{1, \ldots, n\}$. We can think of $\mathbf{W}$ as a pseudo-weight matrix, noting that its entries do not generally sum to unity. Given this notation, the gradient vector and hessian matrix corresponding to l are given by

$$\nabla l(\boldsymbol{\beta} \,|\, \mathbf{X}, \mathbf{Y}) = \mathbf{X}^t(\mathbf{Y} - \boldsymbol{\pi}) \quad \text{and} \quad \nabla^2 l(\boldsymbol{\beta} \,|\, \mathbf{X}, \mathbf{Y}) = -\mathbf{X}^t \mathbf{W} \mathbf{X},$$

respectively. Our Newton-Raphson algorithm is then characterized by the procedure

$$\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t + (\mathbf{X}^t \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^t (\mathbf{Y} - \boldsymbol{\pi}).$$

To assess if this updating procedure is well-behaved, we must investigate the properties of the hessian matrix, thereby motivating the following proposition.

Proposition 1. The hessian matrix of l is negative semi-definite. If $\pi_i \in (0, 1)$ for at least one $i \in \{1, \ldots, n\}$, and if $\mathbf{X}$ is of full rank, then the hessian matrix is negative definite.

Proof. Because $\mathbf{W}$ is a diagonal matrix with non-negative elements, it is positive semi-definite. Thus, for any $u \in \mathbb{R}^p \setminus \{\mathbf{0}\}$, it follows that $\mathbf{X}u \in \mathbb{R}^p$, and hence

$$\begin{aligned} u^t \nabla^2 l(\boldsymbol{\beta} \,|\, \mathbf{X}, \mathbf{Y}) u &= -u^t \mathbf{X}^t \mathbf{W} \mathbf{X} u \\ &= -(\mathbf{X}u)^t \mathbf{W} (\mathbf{X}u) && \text{(by rules for transpose of a product)} \\ &\leq 0. && \text{(by positive semi-definiteness of } \mathbf{W}) \end{aligned}$$

That is, the hessian matrix is negative semi-definite. If we further have that $\mathbf{X}$ is of full rank, then $\mathbf{X}u \in \mathbb{R}^p \setminus \{\mathbf{0}\}$. Moreover, if $\pi_i \in (0, 1)$ for at least one $i \in \{1, \ldots, n\}$, then $\mathbf{W}$ is positive definite. Together, these facts make the above inequality strict, such that the hessian matrix is negative definite. □

The above result provides us with two key insights: (1) if our data matrix is not of full rank, the hessian matrix might not be negative definite; and (2) if our fitted probabilities are all either zero or one, then our hessian matrix will not be negative definite. By culling highly correlated covariates during the exploratory data analysis phase, we have precluded scenario (1) (and indeed, we find that the Newton-Raphson algorithm does not converge if all 30 covariates are considered). To prevent scenario (2), we seed our initial guess at $\boldsymbol{\beta} = \mathbf{0}$, which corresponds to $\pi_i = \frac{1}{2}$ for all $i \in \{1, \ldots, n\}$. As added precautions, we further control for ascent direction and allow step-halving, though this is not critical for such a well-behaved optimization problem.