Logistic Regression and Maximum Likelihood Estimation

David Carlson

Decermber 2, 2021

Why Focus on the Logit?

- Logistic regression is one of the most commonly used tools for applied statistics and discrete data analysis. There are basically four reasons for this.

Why Focus on the Logit?

- Logistic regression is one of the most commonly used tools for applied statistics and discrete data analysis. There are basically four reasons for this.
    - Tradition

Why Focus on the Logit?

- Logistic regression is one of the most commonly used tools for applied statistics and discrete data analysis. There are basically four reasons for this.
  - Tradition
  - The quantity $\log \frac{p}{1-p}$ plays an important role in the analysis of contingency tables (the "log odds"). Classification is a bit like having a contingency table with two columns (classes) and infinitely many rows (values of x). With a finite contingency table, we can estimate the log-odds for each row empirically, by just taking counts in the table. With infinitely many rows, we need some sort of interpolation scheme; logistic regression is linear interpolation for the log-odds.

Why Focus on the Logit?

- Logistic regression is one of the most commonly used tools for applied statistics and discrete data analysis. There are basically four reasons for this.
  - Tradition
  - The quantity $\log \frac{p}{1-p}$ plays an important role in the analysis of contingency tables (the "log odds"). Classification is a bit like having a contingency table with two columns (classes) and infinitely many rows (values of x). With a finite contingency table, we can estimate the log-odds for each row empirically, by just taking counts in the table. With infinitely many rows, we need some sort of interpolation scheme; logistic regression is linear interpolation for the log-odds.
  - It's closely related to "exponential family" distributions, where the probability of some vector ν is proportional to $\exp \beta_0 + \sum_{j=1}^{k} f_j(\nu)\beta_j$. If one of the components of ν is binary, and the functions f_j are all the identity function, then we get a logistic regression. Exponential families arise in many contexts in statistical theory, so there are lots of problems which can be turned into logistic regression

Brief Note

Logistic regression often works surprisingly well as a classifier. But, many simple techniques often work surprisingly well as classifiers, and this doesn't really testify to logistic regression getting the probabilities right.

Logistic Regression Intuition

We have a binary output variable y, and we want to model the conditional probability $Pr(y = 1|X = x)$ as a function of x; any unknown parameters in the function are to be estimated by maximum likelihood. By now, it will not surprise you to learn that statisticians have approach this problem by asking themselves "how can we use linear regression to solve this?"

- The most obvious idea is to let $p(y)$ be a linear function of x. Every increment of a component of x would add or subtract so much to the probability. The conceptual problem here is that p must be between 0 and 1, and linear functions are unbounded. Moreover, in many situations we empirically see "diminishing returns" — changing p by the same amount requires a bigger change in x when p is already large (or small) than when p is close to $1/2$. Linear models can't do this.

Logistic Regression Intuition (cont.)

- The next most obvious idea is to let $\log p(y|x)$ be a linear function of x, so that changing an input variable multiplies the probability by a fixed amount. The problem is that logarithms are unbounded in only one direction, and linear functions are not.

Logistic Regression Intuition (cont.)

- The next most obvious idea is to let $\log p(y|x)$ be a linear function of x, so that changing an input variable multiplies the probability by a fixed amount. The problem is that logarithms are unbounded in only one direction, and linear functions are not.
- Finally, the easiest modification of $\log p$ which has an unbounded range is the logistic (or logit) transformation, $\log \frac{p}{1-p}$. We can make this a linear function of x without fear of nonsensical results.

Logistic Regression Formalization

$$\log \frac{p(x)}{1-p(x)} = x\beta,$$

$$p(x|\beta) = \frac{e^{x\beta}}{1+e^{x\beta}},$$

$$= \frac{1}{1+e^{-x\beta}}$$

To minimize the mis-classification rate, we should predict $y = 1$ when $p \geq 0.5$ and $y = 0$ when $p < 0.5$. This means guessing 1 whenever $x\beta$ is non-negative, and 0 otherwise. So logistic regression gives us a linear classifier. The decision boundary separating the two predicted classes is the solution of $\beta_0 + x\beta_1 + \ldots = 0$, which is a point if x is one dimensional, a line if it is two dimensional, etc. One can show (exercise!) that the distance from the decision boundary is $\beta_0/||\beta|| + x\beta/||\beta||$

(Log) Likelihood

- Let $\sigma(z)$ be the sigmoid function, which in logit turns arbitrary "score" z into number between 0 and 1 (i.e., a probability)

(Log) Likelihood

- Let $\sigma(z)$ be the sigmoid function, which in logit turns arbitrary "score" z into number between 0 and 1 (i.e., a probability)
- Recall $\sigma(z) = \frac{1}{1+e^{-z}}$

(Log) Likelihood

- Let $\sigma(z)$ be the sigmoid function, which in logit turns arbitrary "score" z into number between 0 and 1 (i.e., a probability)
- Recall $\sigma(z) = \frac{1}{1+e^{-z}}$
- Interpret each label as Bernoulli random variable: $y \sim Ber(p), p = \sigma(X\beta)$

(Log) Likelihood

- Let $\sigma(z)$ be the sigmoid function, which in logit turns arbitrary "score" z into number between 0 and 1 (i.e., a probability)
- Recall $\sigma(z) = \frac{1}{1+e^{-z}}$
- Interpret each label as Bernoulli random variable: $y \sim Ber(p), p = \sigma(X\beta)$
- Prob. of one data point (pmf of Bernoulli): $p(Y = y|X = x) = \sigma(X\beta)^y \left[1 - \sigma(X\beta)\right]^{(1-y)}$

(Log) Likelihood

- Let $\sigma(z)$ be the sigmoid function, which in logit turns arbitrary "score" z into number between 0 and 1 (i.e., a probability)
- Recall $\sigma(z) = \frac{1}{1+e^{-z}}$
- Interpret each label as Bernoulli random variable: $y \sim Ber(p), p = \sigma(X\beta)$
- Prob. of one data point (pmf of Bernoulli): $p(Y = y|X = x) = \sigma(X\beta)^y \left[1 - \sigma(X\beta)\right]^{(1-y)}$

(Log) Likelihood

- Let $\sigma(z)$ be the sigmoid function, which in logit turns arbitrary "score" z into number between 0 and 1 (i.e., a probability)
- Recall $\sigma(z) = \frac{1}{1+e^{-z}}$
- Interpret each label as Bernoulli random variable: $y \sim Ber(p), p = \sigma(X\beta)$
- Prob. of one data point (pmf of Bernoulli): $p(Y = y|X = x) = \sigma(X\beta)^y \left[1 - \sigma(X\beta)\right]^{(1-y)}$

$$
\begin{aligned}
\mathcal{L}(\theta) &= \prod_{i=1}^{n} p(Y = y_i|X = x_i), \\
&= \prod_{i=1}^{n} \sigma(X\beta)^{y_i} \log \sigma(X_i\beta) + (1 - y_i) \log \left[1 - \sigma(X_i\beta)\right] \\
\ell(\theta) &= \sum_{i=1}^{n} y_i \log \sigma(X_i\beta) + (1 - y_i) \log \left[1 - \sigma(X_i\beta)\right] \\
\frac{\partial \ell}{\partial \beta_j} &= \sum_{i=1}^{n} \left(y_i - p(y_i|X_i, \beta)\right) X_{ij}
\end{aligned}
$$

Optimization

- This is an intuitive, straight-forward formalization

Optimization

- This is an intuitive, straight-forward formalization
- However, as with most GLMs, there is no closed form solution to maximize the derivative of the log-lik

Optimization

- This is an intuitive, straight-forward formalization
- However, as with most GLMs, there is no closed form solution to maximize the derivative of the log-lik
- Start with a guess, then optimize

Optimization

- This is an intuitive, straight-forward formalization
- However, as with most GLMs, there is no closed form solution to maximize the derivative of the log-lik
- Start with a guess, then optimize
- This is where our previous, very generalized MLE optimizer would work well with any pdf

Optimization

- This is an intuitive, straight-forward formalization
- However, as with most GLMs, there is no closed form solution to maximize the derivative of the log-lik
- Start with a guess, then optimize
- This is where our previous, very generalized MLE optimizer would work well with any pdf
- But what exactly do we do to optimize?

Newton's (Newton-Raphson) Method

- Minimize function of one scalar variable, say $f(\beta)$

Newton's (Newton-Raphson) Method

- Minimize function of one scalar variable, say $f(\beta)$
- Find location of (rather than value of) global minimum, β^*

Newton's (Newton-Raphson) Method

- Minimize function of one scalar variable, say $f(\beta)$
- Find location of (rather than value of) global minimum, β^*
- Near minimum we make Taylor expansion (derivative at β^* is zero and second derivative is positive):

Newton's (Newton-Raphson) Method

- Minimize function of one scalar variable, say $f(\beta)$
- Find location of (rather than value of) global minimum, β^*
- Near minimum we make Taylor expansion (derivative at β^* is zero and second derivative is positive):

Newton's (Newton-Raphson) Method

- Minimize function of one scalar variable, say $f(\beta)$
- Find location of (rather than value of) global minimum, β^*
- Near minimum we make Taylor expansion (derivative at β^* is zero and second derivative is positive):

$$f(\beta) \approx f(\beta^*) + \frac{1}{2}(\beta - \beta^*)^2 \left. \frac{\partial^2 f}{\partial \beta^2} \right|_{\beta=\beta^*}$$

- Second derivative being posititve ensures that $f(\beta) > f(\beta^*)$

Newton's Method Algorithm

Replace problem we want to solve with problem we can solve. Second order Taylor expansion, with intial guess $\beta^{(0)}$:

$$f(\beta) \approx f(\beta^{(0)}) + (\beta - \beta^{(0)}) \left. \frac{\partial f}{\partial \theta} \right|_{\beta=\beta^{(0)}} + \frac{1}{2}(\beta - \beta^{(0)})^2 \left. \frac{\partial^2 f}{\partial \theta^2} \right|_{\beta=\beta^{(0)}}$$

Take derivative with respect to β and set to zero at point $\beta^{(1)}$:

$$\begin{aligned} 0 &= f'(\beta^{(0)}) + \frac{1}{2} f''(\beta^{(0)}) 2(\beta^{(1)} - \beta^{(0)}) \\ \beta^{(1)} &= \beta^{(0)} - \frac{f'(\beta^{(0)})}{f''(\beta^{(0)})} \end{aligned}$$

Value $\beta^{(1)}$ better guess at minimum β^* than $\beta^{(0)}$ $\rightarrow$ iterate until convergence

More Than One Dimension Brief Note

- Now, f is $f(\beta_1, \beta_2, \ldots, \beta_k)$

More Than One Dimension Brief Note

- Now, f is $f(\beta_1, \beta_2, \ldots, \beta_k)$
- And $\beta^{(n+1)} = \beta^{(n)} - H^{-1}(\beta^{(n)}) \nabla f(\beta^{(n)})$

More Than One Dimension Brief Note

- Now, f is $f(\beta_1, \beta_2, \ldots, \beta_k)$
- And $\beta^{(n+1)} = \beta^{(n)} - H^{-1}(\beta^{(n)})\nabla f(\beta^{(n)})$
- ∇f is the gradient (recall vector of partials) and H is the Hessian of f (recall matrix of second partials)

More Than One Dimension Brief Note

- Now, f is $f(\beta_1, \beta_2, \ldots, \beta_k)$
- And $\beta^{(n+1)} = \beta^{(n)} - H^{-1}(\beta^{(n)})\nabla f(\beta^{(n)})$
- ∇f is the gradient (recall vector of partials) and H is the Hessian of f (recall matrix of second partials)
- In practice, calculating inverse of Hessian is time-consuming, so there are approximation methods

Iterative Re-Weighted Least Squares (IRLS)

- This is how GLMs are solved in practice

Iteratively Re-Weighted Least Squares (IRLS)

- This is how GLMs are solved in practice
- Newton's Method is very general; let's return to the Logit

Iteratively Re-Weighted Least Squares (IRLS)

- This is how GLMs are solved in practice
- Newton's Method is very general; let's return to the Logit
- For GLMs, linear model for a transformation (of in this case a probability) $g(\cdot)$

Iteratively Re-Weighted Least Squares (IRLS)

- This is how GLMs are solved in practice
- Newton's Method is very general; let's return to the Logit
- For GLMs, linear model for a transformation (of in this case a probability) $g(\cdot)$
- Logit:

$$
\begin{aligned}
g(p) &\equiv \log \frac{p}{1-p} \\
&= X\beta \\
g(y) &\approx g(X\beta) + (y - X\beta)g'(X\beta) \equiv z \\
V(z) &= (g'(X\beta))^2 V(X\beta)
\end{aligned}
$$

Iteratively Re-Weighted Least Squares (IRLS)

- This is how GLMs are solved in practice
- Newton's Method is very general; let's return to the Logit
- For GLMs, linear model for a transformation (of in this case a probability) $g(\cdot)$
- Logit:

$$
\begin{aligned}
g(p) &\equiv \log \frac{p}{1-p} \\
&= X\beta \\
g(y) &\approx g(X\beta) + (y - X\beta)g'(X\beta) \equiv z \\
V(z) &= (g'(X\beta))^2 V(X\beta)
\end{aligned}
$$

- $\rightarrow$ weighted least squares (weighted by inverse of variance)

Iteratively Re-Weighted Least Squares (IRLS)

- This is how GLMs are solved in practice
- Newton's Method is very general; let's return to the Logit
- For GLMs, linear model for a transformation (of in this case a probability) $g(\cdot)$
- Logit:

$$\begin{aligned} g(p) &\equiv \log \frac{p}{1-p} \\ &= X\beta \\ g(y) &\approx g(X\beta) + (y - X\beta)g'(X\beta) \equiv z \\ V(z) &= (g'(X\beta))^2 V(X\beta) \end{aligned}$$

- $\rightarrow$ weighted least squares (weighted by inverse of variance)
- We now have a linear model

Intuitive Understanding of Ordered Logit

- This is useful in its own right and to start understanding “latent” variables

Intuitive Understanding of Ordered Logit

- This is useful in its own right and to start understanding “latent” variables
- Imagine a latent variable $y^* = X\beta + \epsilon$

Intuitive Understanding of Ordered Logit

- This is useful in its own right and to start understanding "latent" variables
- Imagine a latent variable $y^* = X\beta + \epsilon$
- We can reformulate our understanding into a decision rule:

Intuitive Understanding of Ordered Logit

- This is useful in its own right and to start understanding "latent" variables
- Imagine a latent variable $y^* = X\beta + \epsilon$
- We can reformulate our understanding into a decision rule:

Intuitive Understanding of Ordered Logit

- This is useful in its own right and to start understanding "latent" variables
- Imagine a latent variable $y^* = X\beta + \epsilon$
- We can reformulate our understanding into a decision rule:

$$y = \begin{cases} 0 & \text{if} \quad y^* < 0, \\ 1 & \text{if} \quad 0 \leq y^* < \tau_1, \\ 2 & \text{if} \quad \tau_1 \leq y^* < \tau_2, \\ \vdots & \end{cases}$$

- Notice that the first cut-off needs to be set, and either the variance of the distribution (for Probit we set to 1) or the second cut-off needs to be set (why?)