Logistic Regression and Classification

Stephen Vardeman

Analytics Iowa LLC

ISU Statistics and IMSE

Linear log conditional probability ratios

The common covariance MVN_p model behind LDA makes the ratios

$$\ln\left(\frac{P[y=k|\mathbf{x}]}{P[y=l|\mathbf{x}]}\right)$$

linear in $\mathbf{x}$. An alternative to those model assumptions is to simply assume that for all $k < K$

$$\ln\left(\frac{P[y=k|\mathbf{x}]}{P[y=K|\mathbf{x}]}\right) = \beta_{k0} + \sum_{j=1}^{p} \beta_{kj} x_j \qquad (1)$$

There are then $K-1$ constants β_{k0} and $K-1$ (p-dimensional) vectors $\boldsymbol{\beta}_k = (\beta_{k1}, \beta_{k2}, \ldots, \beta_{kp})$ to be specified (not necessarily tied to class probabilities or mean vectors or a common within-class covariance matrix for $\mathbf{x}$).

Conditional distributions, not joint

The set of relationships (1) do not fully specify a joint distribution for $(\mathbf{x}, y)$. They specify only the nature of the conditional distributions of $y|\mathbf{x}$.

This situation is exactly analogous to that in ordinary simple linear regression. A bivariate normal distribution for (x, y) has normal conditional distributions for y with a constant variance and mean linear in x. But one may make those assumptions conditionally on x, without assuming anything about the marginal distribution of x (that in the bivariate normal model is univariate normal).

Implied conditionals and classification

Using $\boldsymbol{\theta}$ to stand for a vector containing all the constants β_{k0} and the vectors $\boldsymbol{\beta}_k$, the linear log probability ratio assumptions produce the forms

$$p_k(\mathbf{x}, \boldsymbol{\theta}) = P[y = k|\mathbf{x}] = \frac{\exp\left(\beta_{k0} + \sum_{j=1}^{p} \beta_{kj} x_j\right)}{1 + \sum_{k=1}^{K-1} \exp\left(\beta_{k0} + \sum_{j=1}^{p} \beta_{kj} x_j\right)}$$

for $k < K$, and

$$p_K(\mathbf{x}, \boldsymbol{\theta}) = P[y = K|\mathbf{x}] = \frac{1}{1 + \sum_{k=1}^{K-1} \exp\left(\beta_{k0} + \sum_{j=1}^{p} \beta_{kj} x_j\right)}$$

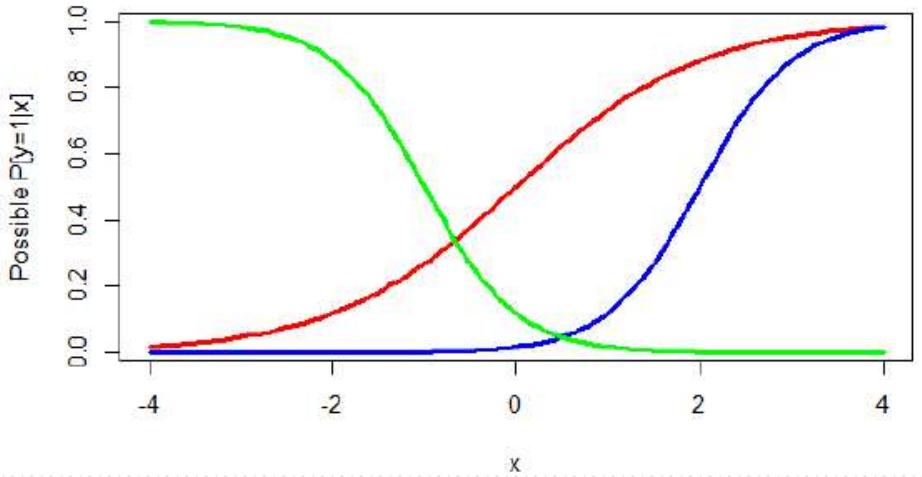
Optimal 0-1 loss classification is then based on maximizing $p_k(\mathbf{x}, \boldsymbol{\theta})$ over $k = 1, 2, \ldots, K$.

Examples for *p=1* and *K=2*

Below is a plot of several different $p = 1$ forms for $p_1(x, \beta_0, \beta_1)$ in a $K = 2$ model. The parameter sets are

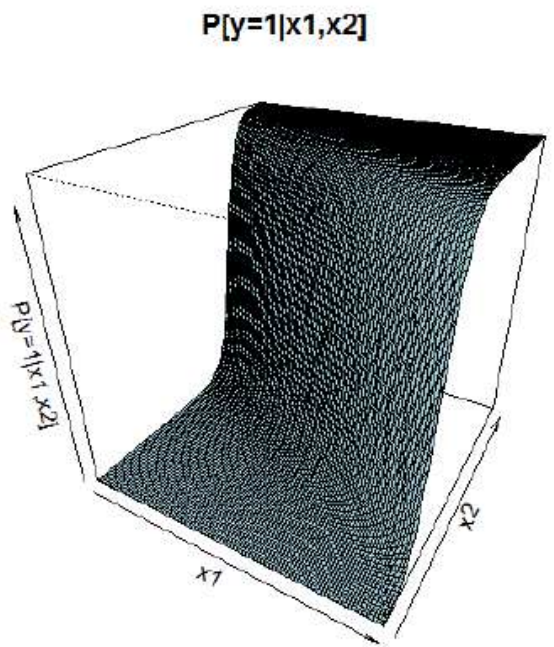
Red	$\beta_0 = 0, \beta_1 = 1$
Blue	$\beta_0 = -4, \beta_1 = 2$
Green	$\beta_0 = -2, \beta_1 = -2$

In each case $p_1(x, \beta_0, \beta_1) = .5$ where $x = -\beta_0/\beta_1$, the function increases in x exactly when $\beta_1 > 0$, and curve steepness increases with $|\beta_1|$.



An example for *p=2* and *K=2*

In a $K = 2$ case with $p = 2$, (for $\{1, 2\}$ coding of y) the kind of relationship pictured below holds. $p_1(\mathbf{x}, \beta_0, \beta_1, \beta_2)$ defines an "s-shaped surface" that is "steep" when coefficients β_1, β_2 have large absolute values, is constant on lines $\beta_0 + \beta_1 x_1 + \beta_2 x_2 = c$, and takes the value .5 on the line $\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$.



Maximum likelihood fitting

Assumption (1) generalizes the model producing LDA, and methods of fitting based on training data are necessarily fundamentally different. Using maximum likelihood in LDA, the K probabilities π_k, the K means $\boldsymbol{\mu}_k$, and the covariance matrix $\boldsymbol{\Sigma}$ are chosen to maximize the likelihood

$$\prod_{i=1}^{N} \pi_{y_i} g\left(\mathbf{x}_i | \boldsymbol{\mu}_{y_i}, \boldsymbol{\Sigma}\right)$$

This is a mixture model and the *complete* likelihood is involved, i.e. a joint density for the N *pairs* $(\mathbf{x}_i, y_i)$.

On the other hand, standard logistic regression methodology maximizes

$$\prod_{i=1}^{N} p_{y_i}\left(\mathbf{x}_i, \boldsymbol{\theta}\right)$$

over choices of $\boldsymbol{\theta}$. This is a *conditional* (on the $\mathbf{x}_i$ observed) likelihood.

K=2 logistic regression and classification

In a $K = 2$ case with $\{-1, 1\}$ coding for y, $1/N$ times the negative log-likelihood is

$$\frac{1}{N}\sum_{i=1}^{N} \ln\left[1 + \exp\left(y_i\left(\beta_0 + \sum_{j=1}^{p}\beta_j x_{ij}\right)\right)\right] \tag{2}$$

For $(\mathbf{x}, y) \sim P$ this is an empirical version of

$$\mathrm{E}\ln\left[1 + \exp\left(-y\left(-\beta_0 - \sum_{j=1}^{p}\beta_j x_j\right)\right)\right]$$

This is $\ln 2$ times an upper bound on the error rate of a classifier with voting function $-\beta_0 - \sum_{j=1}^{p}\beta_j x_j$. So coefficient vectors giving small values of (2) (i.e. large likelihood) can be expected to produce classifiers with small (0-1 loss) Err.

Penalized fitting

Optimization of (2) ignores the potential for overfitting. Penalization (for standardized inputs) of the logistic regression coefficients is a means of investigating a natural spectrum of fitted logistic regressions. For example, `glmnet` will optimize the elastic net penalized negative loglikelihood

$$\frac{1}{N}\sum_{i=1}^{N}\ln\left[1+\exp\left(y_i\left(\beta_0+\boldsymbol{\beta}'\mathbf{x}_i\right)\right)\right]+\lambda\left(\alpha\sum_{j=1}^{p}|\beta_j|+\frac{(1-\alpha)}{2}\sum_{j=1}^{p}\beta_j^2\right)$$

(where $\boldsymbol{\beta}\in\Re^p$). Comparison of **cross-validation** classification error rates across a grid of coefficient vectors (λ,α) affords appropriate choice of **complexity**.

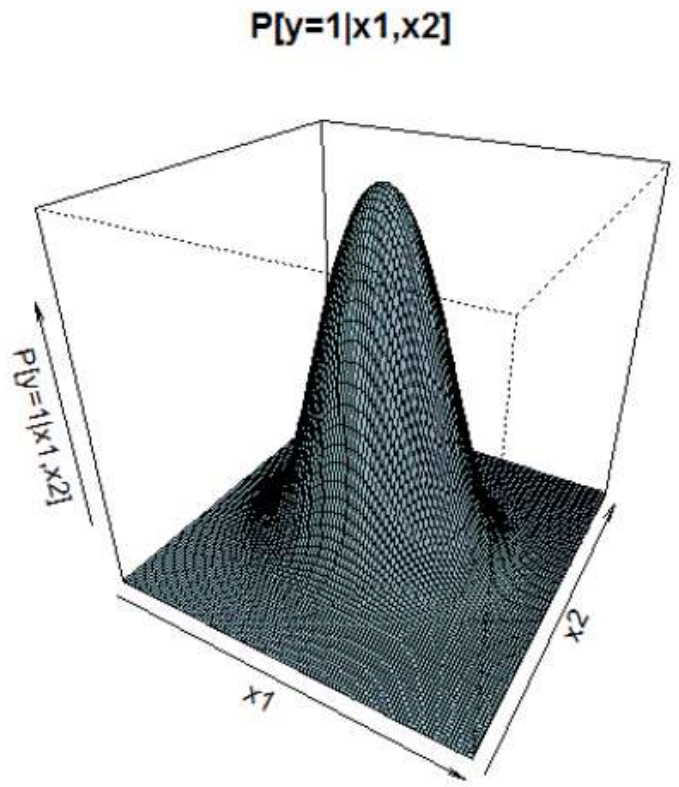
Basis functions and transforms

Good logistic regression models produce good classifiers when one classifies according to the largest predicted probability. And just as the usefulness of LDA can be extended by consideration of transforms/features made from an original p-dimensional $\mathbf{x}$, the same is true for logistic regression.

For example, beginning with x_1 and x_2 and creating additional predictors x_1^2, x_2^2, and $x_1 x_2$, one can use logistic regression technology based on the 5-dimensional input $\left(x_1, x_2, x_1^2, x_2^2, x_1 x_2\right)$ to create classification boundaries that are **quadratic** in the original x_1 and x_2. An example of the kind of functional form for the conditional probability that $y = k$ given a bivariate input $\mathbf{x}$ that can result is on the next slide.

Hypothetical conditional probability *y=1*

The plot below results when one uses the quadratic form $-.2x_1^2 - .3x_2^2$ to make logistic probabilities that $y = 1$ (for 1-2 coding). Constant-probability contours of such a surface are ellipses in (x_1, x_2)-space.



Case-control studies and fitting

It is common to encounter situations where (say in a $K = 2$ context with 0-1 coding) π_0 is quite small. Rather than trying to do analysis on a random sample of $(\mathbf{x}, y)$ pairs where there would be relatively few $y = 0$ cases, there are a number of potentially important practical reasons for doing analysis of a dataset consisting of random sample of N_0 instances ("cases") with $y = 0$ and a random sample of N_1 instances ("controls") with $y = 1$, where $N_0 / (N_0 + N_1)$ is nowhere nearly as small as π_0. (In fact, N_1 on the order of 5 or 6 times N_0 is often recommended.)

For $K = 2$

$$\ln\left(\frac{P[y=0|\mathbf{x}]}{P[y=1|\mathbf{x}]}\right) = \ln\left(\frac{\pi_0 g_0(\mathbf{x})}{\pi_1 g_1(\mathbf{x})}\right) = \ln\left(\frac{\pi_0}{\pi_1}\right) + \ln\left(\frac{g_0(\mathbf{x})}{g_1(\mathbf{x})}\right)$$

Case-control studies and fitting cont.

So under the logistic regression assumption that

$$\ln\left(\frac{P\left[y=0|\mathbf{x}\right]}{P\left[y=1|\mathbf{x}\right]}\right)=\beta_0+\mathbf{x}'\boldsymbol{\beta}$$

fitting to a case-control data set should produce

$$\begin{aligned}\hat{\beta}_0^{cc}+\mathbf{x}'\hat{\boldsymbol{\beta}}^{cc} &\approx \ln\left(\frac{N_0}{N_1}\right)+\ln\left(\frac{g_0\left(\mathbf{x}\right)}{g_1\left(\mathbf{x}\right)}\right)\\ &=\ln\left(\frac{P\left[y=0|\mathbf{x}\right]}{P\left[y=1|\mathbf{x}\right]}\right)+\ln\left(\frac{N_0}{N_1}\right)-\ln\left(\frac{\pi_0}{\pi_1}\right)\end{aligned}$$

So estimated coefficients appropriate for the original context are (a specialized instance of the general formula for shifting conditional probabilities for $y|\mathbf{x}$ based on class frequencies differing from the π_ks)

$$\hat{\beta}_0\equiv\hat{\beta}_0^{cc}-\ln\left(\frac{N_0}{N_1}\right)+\ln\left(\frac{\hat{\pi}_0}{1-\hat{\pi}_0}\right)\quad\text{and}\quad\hat{\boldsymbol{\beta}}=\hat{\boldsymbol{\beta}}^{cc}$$

Separating hyperplanes

In the $K = 2$ group case use again -1-1 coding. If there is a $\boldsymbol{\beta} \in \Re^p$ and real number β_0 such that in the training data

$$y = 1 \text{ exactly when } \mathbf{x}'\boldsymbol{\beta} + \beta_0 > 0$$

a "separating hyperplane"

$$\left\{\mathbf{x} \in \Re^p \,|\, \mathbf{x}'\boldsymbol{\beta} + \beta_0 = 0\right\}$$

can be found via logistic regression. The (conditional) likelihood will not have a maximum, but if one follows a search path far enough toward limiting value of 0 for the loglikelihood or 1 for the likelihood, satisfactory $\boldsymbol{\beta} \in \Re^p$ and β_0 from an iteration of the search algorithm will produce separation.

Perceptron Algorithm

A famous older algorithm for finding a separating hyperplane is the so-called "perceptron" algorithm. It can be defined as follows. From some starting points $\boldsymbol{\beta}^0$ and β_0^0 cycle through the training data cases in order (repeatedly as needed). At any iteration l, take

$$\left\{\boldsymbol{\beta}^l = \boldsymbol{\beta}^{l-1} \text{ and } \beta_0^l = \beta_0^{l-1}\right\} \text{ if } \left\{\begin{array}{l} y_i = 1 \text{ and } \mathbf{x}_i'\boldsymbol{\beta} + \beta_0 > 0, \text{ or} \\ y_i = -1 \text{ and } \mathbf{x}_i'\boldsymbol{\beta} + \beta_0 \leq 0 \end{array}\right\}$$

$$\left\{\begin{array}{l} \boldsymbol{\beta}^l = \boldsymbol{\beta}^{l-1} + y_i\mathbf{x}_i \\ \text{and } \beta_0^l = \beta_0^{l-1} + y_i \end{array}\right\} \qquad \text{otherwise}$$

This will eventually identify a separating hyperplane when a series of N iterations fails to change the values of $\boldsymbol{\beta}$ and β_0.

Non-uniqueness of separating hyperplanes

If there is a separating hyperplane, it will typically not be unique. One can attempt to define and search for "optimal" such hyperplanes that, e.g., maximize distance from the plane to the closest training vector. The material that follows on support vector machines is exactly in this direction.