Parameter transformations – LDA

Fixing the last group K as a reference group we have for $k = 1, \ldots, K-1$ that

$$\log \frac{\Pr(Y = k|X = x)}{\Pr(Y = K|X = x)} = \underbrace{\log \frac{\pi_k}{\pi_K} + \frac{1}{2}\mu_K^T \Sigma^{-1} \mu_K - \frac{1}{2}\mu_k^T \Sigma^{-1} \mu_k}_{\beta_{k0}} + x^T \underbrace{\Sigma^{-1}(\mu_k - \mu_K)}_{\beta_k}$$

Thus

$$\Pr(Y = k|X = x) = \frac{\exp(\beta_{k0} + x^T \beta_k)}{1 + \sum_{l=1}^{K-1} \exp(\beta_{l0} + x^T \beta_l)}$$

for $k = 1, \ldots, K-1$. The conditional distribution depends upon $\pi_1, \ldots, \pi_{K-1}, \mu_1, \ldots, \mu_k, \Sigma$ through the parameter transformation

$$(\pi_1, \ldots, \pi_{K-1}, \mu_1, \ldots, \mu_K, \Sigma) \mapsto (\beta_{10}, \ldots, \beta_{(K-1)0}, \beta_1, \ldots, \beta_{K-1}).$$

Logistic Regression

We consider $K = 2$ and encode the y-variable as 0 or 1. The logistic regression model is given by

$$\Pr(Y = 1 \mid X = x) = \frac{\exp((1, x^T)\beta)}{1 + \exp((1, x^T)\beta)}$$

Hence

$$\Pr(Y = 0 \mid X = x) = 1 - \frac{\exp((1, x^T)\beta)}{1 + \exp((1, x^T)\beta)} = \frac{1}{1 + \exp((1, x^T)\beta)}.$$

We saw that the conditional distribution of Y given X in the LDA setup is a logistic regression model.

Figure 4.12 – South African Heart Disease Data

A typical use of logistic regression. The response variable is Myocardial Infarction. The two cases (0/1) are color coded in the plot.

The plot reveals pair-wise – and marginal – effects of the 7 observed variables on MI.

And clear correlations between `obesity` and `sbp` (systolic blood pressure), say.

Logistic Regression – Notation

Given a dataset $(y_1, x_1), \ldots, (y_N, x_N)$ write

$$\mathbf{p}(\beta) = (p_i(\beta))_{i=1}^N, \quad p_i(\beta) = \frac{\exp((1, x_i^T)\beta)}{1 + \exp((1, x_i^T)\beta)}.$$

With $h : \mathbb{R}^N \to \mathbb{R}^N$

$$h_i(z) = -\log(1 + \exp(z_i))$$

and taking coordinatewise logarithm

$$\log \mathbf{p}(\beta) = \mathbf{X}\beta + h(\mathbf{X}\beta)$$

and

$$\log(\mathbf{1} - \mathbf{p}(\beta)) = h(\mathbf{X}\beta)$$

Logistic Regression – The Minus-Log-Likelihood Function

The (conditional) likelihood function of observing $y_1, \ldots, y_N$ given $x_1, \ldots, x_N$ is

$$\mathcal{L}(\beta) = \prod_{i=1}^{N} p_i(\beta)^{y_i}(1 - p_i(\beta))^{1-y_i}$$

and the minus-log-likelihood function is

$$\begin{aligned} l(\beta) &= -\mathbf{y}^T(\mathbf{X}\beta + h(\mathbf{X}\beta)) - (\mathbf{1} - \mathbf{y})^T h(\mathbf{X}\beta) \\ &= -\mathbf{y}^T\mathbf{X}\beta - \mathbf{1}^T h(\mathbf{X}\beta) \end{aligned}$$

Observe that $D_z h(z)$ is diagonal with

$$D_z h(z)_{ii} = -\frac{\exp(z_i)}{1 + \exp(z_i)}$$

Logistic Regression – The MLE

By differentiation

$$\begin{aligned} D_\beta l(\beta) &= -\mathbf{y}^T\mathbf{X} - \mathbf{1}^T D_z h(\mathbf{X}\beta)\mathbf{X} \\ &= -\mathbf{y}^T\mathbf{X} + \mathbf{p}(\beta)^T\mathbf{X} \\ &= (\mathbf{p}(\beta)^T - \mathbf{y}^T)\mathbf{X} \end{aligned}$$

and

$$D_\beta^2 l(\beta) = D_\beta \mathbf{p}(\beta)^T\mathbf{X} = \mathbf{X}^T\mathbf{W}(\beta)\mathbf{X}$$

with

$$\begin{aligned} \mathbf{W}(\beta) &= \mathrm{diag}(\mathbf{p}(\beta))\mathrm{diag}(1 - \mathbf{p}(\beta)) \\ &= \left\{ \begin{array}{ccc} p_1(\beta)(1 - p_1(\beta)) & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & p_N(\beta)(1 - p_N(\beta)) \end{array} \right\} \end{aligned}$$

Likelihood Equation

The non-linear likelihood estimation equation reads (after transposition)

$$\mathbf{X}^T \mathbf{p}(\beta) = \mathbf{X}^T \mathbf{y}$$

Since $D^2_\beta l(\beta) = \mathbf{X}^T \mathbf{W}(\beta) \mathbf{X}$ is positive definite whenever $\mathbf{X}$ has full rank $p + 1$, the minus-log-likelihood function is strictly convex and a minimum is unique.

There is no solution if the x-values for the two groups can be separated completely by a hyperplane.

Logistic Regression – Algorithm

A first order Taylor expansion

$$\mathbf{p}(\beta) \simeq \mathbf{p}(\beta^0) + \mathbf{W}(\beta^0)\mathbf{X}(\beta - \beta^0)$$

around β^0 yields the approximating equation

$$\mathbf{X}^T\mathbf{W}(\beta^0)\mathbf{X}\beta = \mathbf{X}^T\mathbf{W}(\beta^0)(\underbrace{\mathbf{X}\beta^0 + \mathbf{W}(\beta^0)^{-1}(\mathbf{y} - \mathbf{p}(\beta^0))}_{\text{adjusted response}=\mathbf{z}_0}).$$

The solution is precisely the solution of the weighted least squares problem

$$\operatorname*{argmin}_{\beta} \, (\mathbf{z}_0 - \mathbf{X}\beta)^T\mathbf{W}(\beta^0)(\mathbf{z}_0 - \mathbf{X}\beta)$$

Iteration yielding a sequence β^n, $n \geq 0$, is known as the iterative reweighted least squares algorithm – or IRLS – using the adjusted response

$$\mathbf{z}_n = \mathbf{X}\beta^n + \mathbf{W}(\beta^n)^{-1}(\mathbf{y} - \mathbf{p}(\beta^n))$$

in the $(n+1)$'th iteration. The algorithm is equivalent to the Newton-Raphson algorithm.

Multinomial Regression and LDA

It is possible to formulate a multinomial version of the binary logistic regression model.

The algorithm for estimation becomes more complicated.

LDA relies on MLE for the full parameter in the full distribution of (X, Y). Logistic/multinomial regression relies on MLE in the conditional distribution of $Y \mid X$.

Logistic regression makes fewer distributional assumptions. Deviations from normality could affect LDA in the negative direction.

If the distributional assumptions for LDA are fulfilled, LDA is a little more efficient.

Penalized logistic regression

With $J : \mathbb{R}^{p+1} \to [0, \infty)$ we can consider the penalized minus-log-likelihood

$$l(\beta) + \lambda J(\beta).$$

If $J(\beta) = \sum_{i=1}^{p} \beta_i^2$ or $J(\beta) = \sum_{i=1}^{p} |\beta_i|$ there is always a minimizer.

Efficient algorithms (especially for lasso in the R package `glmnet`) are based on iterations that solve a penalized weighted least squares problem.

Large p Small N Problems

When $p > N$ and in particular when $p \gg N$ new issues arise.

- We are never able to estimate all parameters without regularization. E.g. in a regression there are p parameters but the $\mathbf{X}$-matrix only has rank N.
- Signals can drown in noise.
- Big matrices, computational challenges.

As a rule of thumb; choose simple methods over complicated methods when $p \gg N$, regularize and bet on "sparsity".

Figure 18.1

Simulation study with $Y = \sum_{i=1}^{p} \beta_j X_j + \sigma\epsilon$.

Diagonal or Independence LDA

Recall that the estimated LDA classifier can be determined by

$$\delta_k(x) = \log \pi_k - \frac{1}{2}(x - \hat{\mu}_k)^T \hat{\Sigma}^{-1}(x - \hat{\mu}_k)$$

and we classify to $\text{argmax}_k\{\delta_k(x)\}$.

If

$$\hat{\Sigma} = \text{diag}(s_1^2, \ldots, s_p^2)$$

this simplifies to

$$\delta_k(x) = -\sum_{j=1}^{p} \frac{(x_j - \bar{x}_{kj})^2}{s_j^2} + 2\log \pi_k$$

where $x = (x_1, \ldots, x_p)^T$ and

$$\bar{x}_{kj} = \frac{1}{N_k} \sum_{i:y_i=k} x_{ij}$$

is the average of the j'th coordinate in the k'th group.

Shrunken Centroids

Note that the variance of $\bar{x}_{kj} - \bar{x}_j$ is

$$m_k^2 \sigma^2 \quad \text{with} \quad m_k^2 = \frac{1}{N_k} - \frac{1}{N}.$$

Introduce the general shrunken centroids

$$\bar{x}'_{kj} = \bar{x}_j + m_k(s_j + s_0) g \left(\frac{\bar{x}_{kj} - \bar{x}_j}{m_k(s_j + s_0)} \right)$$

with s_0 a small, positive constant.

$$g_\Delta(d) = \text{sign}(d)(|d| - \Delta)_+$$

is known as soft thresholding.

$$g_\Delta(d) = d1(|d| \geq \Delta)_+$$

as hard thresholding.

Figure 18.4 – Train and Test Error

The parameter Δ is a tuning parameter for shrunken centroids. With 43 genes, $\Delta = 4.3$, we get a training error of 0 – but also a test error of 0.

Figure 18.4 – Centroid Profiles and Shrunken Centroids

Figure 18.3 – Heat Map

Elastic Net

The penalization function

$$\sum_{j=1}^{p} \alpha|\beta_j| + (1-\alpha)\beta_j^2$$

is known as the elastic net penalty.

For multinomial regression the penalized minus-log-likelihood function is

$$-\sum_{i=1}^{N} \log \Pr(Y = y_i | X = x_i) + \lambda \sum_{k=1}^{K} \sum_{j=1}^{p} \alpha|\beta_{kj}| + (1-\alpha)\beta_{kj}^2$$

There is an efficient implementation in the `glmnet` package for R.

Note that intercepts are not penalized and subject to the constraint that they sum to 0. All other redundancies in the parameterization are dealt with by the penalization.

Regularized Discriminant Analysis

Choosing the estimator

$$\hat{\Sigma}(\alpha) = \alpha\hat{\Sigma} + (1-\alpha)\mathrm{diag}(\hat{\Sigma})$$

for $\alpha \in [0, 1]$ we get a regularized covariance estimator usable for LDA.

The `rda` function in the `rda` library does this in combination with nearest shrunken centroids with `regularization="R"`. With `regularization="S"` one gets

$$\hat{\Sigma}(\alpha) = \alpha\hat{\Sigma} + (1-\alpha)I_p$$

It is a little unclear which of three suggested centroid shrinkage methods from the paper Guo et al. (2006), see book, that is implemented in the R package `rda`.

Support Vector Classifiers

Support vector machines are popular two class classifiers and have a reputation for being among the best performing.

With $y_i \in \{-1, 1\}$, $x_i \in \mathbb{R}^p$ and $f : \mathbb{R}^p \to \mathbb{R}$ we compute the predictor of y_i as $\mathrm{sign}(f(x_i))$. With f in a reproducing kernel Hilbert space $\mathcal{H}$ estimation is done by minimization of

$$\sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \lambda ||f||_{\mathcal{H}}^2$$

Thus the loss function $L : \{-1, 1\} \times \mathbb{R} \to \mathbb{R}$ is special and given as

$$L(y, z) = [1 - yz]_+ = \max\{1 - yz, 0\}$$

Support Vector Classifiers – example

Simplest example: $\mathcal{H} = \mathbb{R}^{p+1}$ and $f(x) = x^T\beta + \beta_0$ for $\beta \in \mathbb{R}^p$. Hence the objective is minimization of

$$\sum_{i=1}^{N}[1 - y_i(x_i^T\beta + \beta_0)]_+ + \lambda \sum_{i=0}^{p} \beta_i^2.$$

This problem can be equivalently formulated as a constraint optimization problem; minimize

$$\sum_{i=0}^{p} \beta_i^2 + C \sum_{i=1}^{N} \xi_i$$

subject to $\xi_i \geq 0$ and $y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i$ for $i = 1, \ldots, N$.

The kernel trick

A reproducing kernel Hilbert space is characterized by a kernel, $K(x, x')$, which is reproducing:

$$\langle K(\cdot, x), K(\cdot, x') \rangle = K(x, x').$$

Solutions to the optimization problem above take the form

$$f(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i).$$

The problem reduces to optimization of

$$\sum_{i=1}^{N} [1 - y_i(\mathbf{K}\alpha)_i]_+ + \lambda \alpha^T \mathbf{K} \alpha$$

where $\mathbf{K}_{ij} = K(x_i, x_j)$.

This is the kernel trick, which can reduce a high-dimensional problem (dimension $p \gg N$) to a reasonable sized problem of dimension N.

Kernel examples

- $K(x, x') = x^T x'$ – the linear kernel.
- $K(x, x') = (1 + \gamma x^T x')^d$ – the polynomial kernel.
- $K(x, x') = \exp(-\gamma ||x - x'||^2)$ – the radial basis kernel.
- $K(x, x') = \tanh(\kappa_1 x^T x' + \kappa_2)$ – the sigmoid or neural network kernel.

Except for the linear kernel the kernels have, in addition to the penalization parameter, one or two other tuning parameters.