1 Logistic regression

Logistic regression for two classes simplifies to a likelihood:

(Using $\pi_i(\beta) = \mathbb{P}(Y = 1|X = X_i, \beta)$)

$$\begin{aligned}\ell(\beta) &= \sum_{i=1}^{n} \left(y_i \log(\pi_i(\beta)) + (1 - y_i) \log(1 - \pi_i(\beta))\right) \\ &= \sum_{i=1}^{n} \left(y_i \log(e^{\beta^\top X_i}/(1 + e^{\beta^\top X_i})) - (1 - y_i) \log(1 + e^{\beta^\top X_i})\right) \\ &= \sum_{i=1}^{n} \left(y_i \beta^\top X_i - \log(1 + e^{\beta^\top X_i})\right)\end{aligned}$$

This gets optimized via Newton-Raphson updates and iteratively reweighted least squares

2 Sparse logistic regression

This procedure suffers from all the same problems as least squares.

Now, since we have identified the likelihood as a function we'd like to maximize, we can treat the negative of this quantity as a loss function to minimize and can apply penalized likelihood techniques in the same way as we did before.

This means maximizing (over β_0, β):

$$\sum_{i=1}^{n} \left(y_i(\beta_0 + \beta^\top X_i) - \log(1 + e^{\beta_0 + \beta^\top X_i})\right) - \lambda(\alpha||\beta||_1 + (1 - \alpha)||\beta||_2^2)$$

(Don't penalize the intercept and do standardize the covariates)

This is the logistic elastic net

2.1 logistic elastic net

Recall the Lasso and ridge problems:

$$\min \hat{P}_{\ell\beta} + \lambda\|\beta||_1$$
$$\min \hat{P}_{\ell\beta} + \lambda\|\beta||_2^2$$

The motivation for the logistic elastic net (notice that this penalty is simply a convex combination of the ridge and lasso penalties)

$$\min \hat{P}_{\ell\beta} + \lambda(\alpha||\beta||_1 + (1-\alpha)||\beta||_2^2)$$

comes from the lasso problem not being strictly convex, and therefore potentially having multiple solutions. The logistic elastic net, on the other hand, is strictly convex, yet can still set parameter estimates to zero, and so retains lasso's ability to do model selection.

In `glmnet`, the alpha parameter dictates which of lasso, ridge, or the logistic elastic net is to be used.

3 Sparse logistic regression: Software

Using the `R` package `glmnet` finds the minimum CV solution over a grid of λ values

Unfortunately, the computations are more difficult for path algorithms (such as the `lars` package) due to the coefficient profiles being only piecewise smooth

`glmpath` is an `R` package that does quadratic approximations to the profiles, while still computing the exact points at which the active set changes

(Park, Hastie (2007). It is necessary to set a 'step' size argument for the approximation.)

4 Logistic versus LDA

The log posterior odds via the Gaussian likelihood (LDA) for class g versus G are

$$\begin{aligned}\log\frac{\mathbb{P}(Y=g|X=x)}{\mathbb{P}(Y=G|X=x)} &= \log\frac{\pi_g}{\pi_G} - (\mu_g+\mu_G)^\top\Sigma^{-1}(\mu_g-\mu_G)/2 \\ &\quad + x^\top\Sigma^{-1}(\mu_g-\mu_G) \\ &= \alpha_{g,0} + \alpha_g^\top x\end{aligned}$$

Likewise, multi class logistic follows (for $g = 1, \ldots, G-1$):

$$\log\frac{\mathbb{P}(Y=g|X=x)}{\mathbb{P}(Y=G|X=x)} = \beta_{g,0} + \beta_g^\top x$$

(The choice of base class G is arbitrary)

(They both specify the log-odds as linear models!)

We can write the joint distribution of Y and X as

$$\mathbb{P}(X,Y) = \mathbb{P}(Y|X)\mathbb{P}(X)$$

The previous slide shows that $\mathbb{P}(Y|X)$ is the same for both methods:

$$\mathbb{P}(Y=g|X=x) = \frac{e^{\alpha_{g,0}+\alpha_g^\top x}}{1+\sum_{k=1}^{G-1} e^{\alpha_{k,0}+\alpha_k^\top x}}$$

- Logistic regression leaves $\mathbb{P}(X)$ arbitrary, and implicitly estimates it with the empirical measure
  (This could be interpreted as a frequentist approach, where we are maximizing the likelihood only and using the improper uniform prior)
- LDA models make stronger modeling assumptions:

$$\mathbb{P}(X, Y = g) = \mathbb{P}(X|Y = g)\mathbb{P}(Y = g) = N(x; \mu_g, \Sigma)\pi_g$$

Some remarks:

- Forming logistic requires fewer assumptions
- The MLEs under logistic will be undefined if the classes are perfectly separable
- If some entries in X are qualitative, then the modeling assumptions behind LDA are suspect
- In practice, the two methods tend to give very similar results

5 Basic Linear Geometry

A hyperplane in $\mathbb{R}^p$ is given by

$$\mathcal{H} = \{x \in \mathbb{R}^p : f(x) = \beta_0 + \beta^\top x = 0\}$$

1. The vector β is normal to $\mathcal{H}$.
   Note: Since the surface is a hyperlane, normal means being orthogonal to any vector connecting two points in $\mathcal{H}$.
   To see this, let $\underline{x_1}$ and $\underline{x_2}$ be two points in $\mathcal{H}$
   Then:

$$\begin{aligned} < x_1 - x_2, \beta > &= < x_1, \beta > - < x_2, \beta > \\ &= (-\beta_0) - (-\beta_0) \\ &= 0 \end{aligned}$$

2. For any point $x \in \mathbb{R}^p$, the (signed) length of its orthogonal complement to $\mathcal{H}$ is $f(x)$
   pf: The orthogonal complement onto $\mathcal{H}$, for some point x_p, is the vector which connects x_p to the closest vector in $\mathcal{H}$. This may be found by finding a vector which connects $\mathcal{H}_2$,(the plane parallel to $\mathcal{H}$, which contains x_p) to $\mathcal{H}$, and which is orthogonal to both $\mathcal{H}$ and $\mathcal{H}_2$
   First, note that:

$$\mathcal{H}_2 = \{x \in \mathbb{R}^p : f(x) = \beta^\top x - x_p^T \beta = 0\}$$

   Now, we find the vector $x^* \in \mathcal{H}_2$ orthogonal to both $\mathcal{H}$ and $\mathcal{H}_2$, and $x_0 \in \mathcal{H}$, also orthogonal to the two spaces.
   $x^* \in \mathcal{H}_2$, thus

$$\begin{aligned} {x_*}^T \beta &= x_p^T \beta \\ \rightarrow a\beta^T \beta &= x_p^T \beta && \text{(orthogonal)} \\ \rightarrow a &= x_p^T \beta && (\beta \text{ unit length}) \end{aligned}$$

and so

$$x^* = (x_p^T \beta)\beta$$

$x_0 \in H$, thus

$$\begin{aligned} {x_0}^T \beta &= -\beta_0 \\ \rightarrow b\beta^T \beta &= -\beta_0 && \text{(orthogonal)} \\ \rightarrow b &= -\beta_0 && (\beta \text{ unit length}) \end{aligned}$$

and so

$$x_0 = (-\beta_0)\beta$$

Thus,

$$x^* - x_0 = (x_p^T \beta + \beta_0)\beta$$

whose signed length is $(x_p^T \beta + \beta_0)$

6 Support vector machines (SVM)

Let $Y_i \in \{-1, 1\}$

(w.l.o.g let $||\beta||_2 = 1$)

A classification rule induced by this hyperplane is

$$\hat{Y}(x) = \text{sgn}(x^\top \beta + \beta_0)$$

7 Separating hyperplanes

As our classification rule is based on a hyperplane $\mathcal{H}$

$$\hat{Y}(x) = \text{sgn}(x^\top \beta + \beta_0)$$

we know the signed distance to $\mathcal{H}$ is $f(x) = x^\top \beta + \beta_0$

Under classical separability, we can find a function such that $Y_i f(X_i) > 0$

That is, makes perfect classifications via $\hat{Y}$

The larger the quantity $Y_i f(X_i)$, the more separated the classes

8 Optimal separating hyperplane

This idea can be encoded in the following convex program

$$\max_{\beta_0, \beta} M \quad \text{subject to}$$

$$Y_i f(X_i) \geq M \text{ for each } i$$

Dropping the norm constraint on β, we have the equivalent program

$$min_{\beta_0,\beta} \|\beta\|_2 \text{ subject to}$$

$$Y_i f(X_i) \geq 1 \text{ for each } i$$

(Convex optimization program: quadratic criterion, linear inequality constraints)

Of course, we can't realistically assume that the data are linearly separated (even in a transformed space)

In this case, the previous program has no feasible solution

We need to introduce slack variables that allow for overlap among the classes

$$\min_{\beta_0,\beta} \|\beta\|_2 \text{ subject to}$$

$$Y_i f(X_i) \geq 1 \ -\xi_i, \xi_i \geq 0, \sum \xi_i \leq c, \text{ for each } i$$

(Convex optimization program: quadratic criterion, linear inequality constraints)

This can be rewritten as

$$\min_{\beta_0,\beta} \|\beta\|_2 /2 + C \sum \xi_i \text{ subject to}$$

$$Y_i f(X_i) \geq 1 - \xi_i, \xi_i \geq 0, , \text{ for each } i$$

Note that

- C is the cost parameter
- The separable case corresponds to $C = \infty$

8.1 lagrange function

Recall that, for the general optimization problem

$$\min_x f(x) \text{ subject to}$$

$$g_i(x) \leq 0 \text{ for i} = 1, .., n$$

The corresponding lagrange function is

$$\ell(x, \lambda) = f(x) + \sum \lambda_i f_i(x)$$

Thus, the corresponding Lagrange function to the SVM optimization problem is

$$\ell_{SVM}(\beta, \beta_0, \xi) = \|\beta|_2 /2 + C \sum \xi_i$$
$$- \sum_{i=1}^{n} \gamma_i [Y_i f(X_i) - (1 - \xi_i)] - \sum_{i=1}^{n} \lambda_i \xi_i$$

(Note: we've split the λ_i from the general form into λ_i and γ_i)

We minimize with respect to β_0, β, ξ_i via partial derivatives:

$$\begin{aligned} \beta &= \sum_{i=1}^{n} \gamma_i y_i x_i \\ 0 &= \sum_{i=1}^{n} \gamma_i y_i \\ \gamma_i &= C - \lambda_i \\ \gamma_i, \lambda_i, \xi_i &\geq 0 \end{aligned}$$

Outline of optimization steps

1. Formulate constrained form of problem
2. Convert to (primal) Langragian form
3. Take all relevant (sub)-derivatives and set to zero
4. Substitute these conditions back into primal Langrangian $\longrightarrow$ dual Lagrangian
   (This forms a lower bound on the solution to the constrained primal form of objective)
5. Form Karush-Kuhn-Tucker (KKT) conditions for inequality constraints
6. Examine the conditions for strong duality which implies that the primal and dual forms have the same solution
   (The usual method is via Slater's condition, which says strong duality holds if it is a convex program with non-empty constraint region)

Once minimizers for $\hat{\gamma}$ are found, we can plug-in and get

$$\hat{\beta} = \sum_{i=1}^{n} \hat{\gamma}_i Y_i X_i$$

8.2 KKT conditions

For the following minimization problem

$$\begin{aligned} &\min f(x) \text{ subject to} \\ &g_i(x) \leq 0 \text{ for all } i \end{aligned}$$

the KKT conditions are:

$$g_i(x) \leq 0 \tag{1}$$
$$\lambda_i \geq 0 \tag{2}$$
$$\lambda_i g_i(x) = 0 \tag{3}$$
$$\Delta f(x) + \Sigma \lambda_i \Delta g_i(x) = 0 \tag{4}$$

Where the λ_i are those seen in the problem's lagrange function.

Due to the KKT conditions

$$\begin{aligned} \gamma_i[Y_i(X_i^\top \beta + \beta_0) - (1 - \xi_i)] &= 0 \\ \lambda_i \xi_i &= 0 \\ Y_i(X_i^\top \beta + \beta_0) - (1 - \xi_i) &\leq 0 \end{aligned}$$

(The first two of these equalities are a result of KKT condition (3), and the third is a result of KKT condition (1).)

The $\hat{\gamma}_i$ are nonzero only for i such that the third inequality is a strict inequality. These observations are called the support vectors

By the previous conditions, either $\hat{\xi}_i = 0$ or $\hat{\gamma}_i = C$

Using the condition: $\gamma_i[Y_i(X_i^\top \beta + \beta_0) - (1 - \xi_i)] = 0$

For support vector i, if $\hat{\xi}_i = 0$:

$$Y_i(X_i^\top \beta + \beta_0) - (1 - \xi_i) = 0 \Leftrightarrow \hat{\beta}_0 = 1/Y_i - X_i^\top \hat{\beta}$$

(These estimates are usually averaged to get a final estimate of β_0)

Now, the final classification is given by

$$\hat{Y}(x) = \text{sgn}(\hat{f}(x)) = \text{sgn}(x^\top \hat{\beta} + \hat{\beta}_0)$$

The tuning parameter is given by the cost C

If we return to step (4) from the SVM outline, that is, substitituing these conditions back into the primal lagrangian, to get the dual lagrangian, we get that this dual Lagrangian is:

$$\ell_D(\gamma) = \sum_i \gamma_i - \frac{1}{2} \sum_i \sum_{i'} \gamma_i \gamma_{i'} Y_i Y_{i'} X_i^\top X_{i'}$$

with side conditions: $\gamma_i \in [0, C]$ and $\gamma^\top Y = 0$

The term $X_i^\top X_{i'} = \langle X_i, X_{i'} \rangle$ is an inner product

SVMs therefore depend on the covariates via an inner product only

This leaves them ripe for a kernel method