Maximum Likelihood Estimators

(Raschke)

The MLE for logistic regression is:

$$\mathcal{L}(\boldsymbol{w}, b|\boldsymbol{x}) = p(y|\boldsymbol{x}; \boldsymbol{w}, b) = \prod_{i=1}^{n} p\left(y^{(i)} \middle| \boldsymbol{x}^{(i)}; \boldsymbol{w}, b\right) = \prod_{i=1}^{n} \left(\sigma\left(z^{(i)}\right)\right)^{y^{(i)}} \left(1 - \sigma\left(z^{(i)}\right)\right)^{1-y^{(i)}}$$

In practice, it is easier to maximize the (natural) log of this equation, which is called the **log-likelihood** function:

$$l(\boldsymbol{w}, b|\boldsymbol{x}) = \log \mathcal{L}(\boldsymbol{w}, b|\boldsymbol{x}) = \sum_{i=1} \left[y^{(i)} \log\left(\sigma\left(z^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - \sigma\left(z^{(i)}\right)\right)\right]$$

The formula for the gradient in $w = w - \eta \nabla L(w, b)$:

$$\frac{\partial L}{\partial w_j} = \underbrace{\frac{\partial L}{\partial a} \frac{da}{dz} \frac{\partial z}{\partial w_j}}_{\text{Apply chain rule}} \qquad \text{where} \quad a = \sigma(z) = \frac{1}{1 + e^{-z}}$$

1) Derive terms separately:

$$\frac{\partial L}{\partial a} = \frac{a - y}{a - a^2}$$

$$\frac{da}{dz} = \frac{e^{-z}}{(1 + e^{-z})^2} = a \cdot (1 - a)$$

$$\frac{\partial z}{w_j} = x_j$$

2) Combine via chain rule and simplify:

$$\frac{\partial L}{\partial z} = a - y$$

$$\frac{\partial L}{\partial w_j} = (a - y)x_j = -(y - a)x_j$$

The Logistic Binary Classifier

Suppose X is a function that gives outputs $0, 1$. The **logistic function** is the function

$$p(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}. \tag{4.2}$$

We can rewrite this into an **odds formula**:

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0 + \beta_1 X}. \tag{4.3}$$

and take the logarithm to get the **logit** function:

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X. \tag{4.4}$$

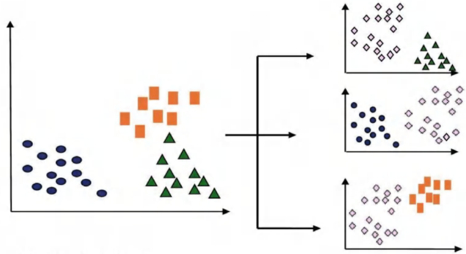
We can extend this using multivariate regression as well ...

One vs. the Rest

Question: How to extend this to K (more than two) classes?

There are a couple of ways . . .

In a **one-vs-rest classifier** (or **OvR** or **OvA**), we train a single binary classifiers several times, each time comparing one of the K categories to the remainder–thus generating K classifiers. Then choose the category K which has the highest probability.



Classifier 1:- [Green]

Classifier 2:- [Blue]

Classifier 3:- [Orange]

Multinomial Logistic Regression

Idea: That approach works for *all* binary classifiers. There is another approach for logistic regression.

Choose a **baseline** class (say the Kth class). We want a function that computes

$$\Pr(Y = k|X = x) = \frac{e^{\beta_{k0}+\beta_{k1}x_1+\cdots+\beta_{kp}x_p}}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0}+\beta_{l1}x_1+\cdots+\beta_{lp}x_p}} \tag{4.10}$$

for $k = 1, \ldots, K-1$, and

$$\Pr(Y = K|X = x) = \frac{1}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0}+\beta_{l1}x_1+\cdots+\beta_{lp}x_p}}. \tag{4.11}$$

It is not hard to show that for $k = 1, \ldots, K-1$,

$$\log\left(\frac{\Pr(Y = k|X = x)}{\Pr(Y = K|X = x)}\right) = \beta_{k0} + \beta_{k1}x_1 + \cdots + \beta_{kp}x_p. \tag{4.12}$$

As in OvR, the idea is to come up with a formula for $P(Y = k|X = x)$ and then take the maximum over these values.

Finding $P(Y = k|X = x)$

We have:

$$\ln \frac{\Pr(Y_i = k)}{\Pr(Y_i = K)} = \boldsymbol{\beta}_k \cdot \mathbf{X}_i, \qquad 1 \leq k < K.$$

If we exponentiate both sides and solve for the probabilities, we get:

$$\Pr(Y_i = k) = \Pr(Y_i = K)\, e^{\boldsymbol{\beta}_k \cdot \mathbf{X}_i}, \qquad 1 \leq k < K$$

Using the fact that all K of the probabilities must sum to one, we find:

$$\begin{aligned}
\Pr(Y_i = K) &= 1 - \sum_{j=1}^{K-1} \Pr(Y_i = j) \\
&= 1 - \sum_{j=1}^{K-1} \Pr(Y_i = K)\, e^{\boldsymbol{\beta}_j \cdot \mathbf{X}_i} \quad \Rightarrow \quad \Pr(Y_i = K) \\
&= \frac{1}{1 + \sum_{j=1}^{K-1} e^{\boldsymbol{\beta}_j \cdot \mathbf{X}_i}}.
\end{aligned}$$

We can use this to find the other probabilities:

$$\Pr(Y_i = k) = \frac{e^{\boldsymbol{\beta}_k \cdot \mathbf{X}_i}}{1 + \sum_{j=1}^{K-1} e^{\boldsymbol{\beta}_j \cdot \mathbf{X}_i}}, \qquad 1 \leq k < K.$$

SKLearn Logistic Regression

LogisticRegression

class **sklearn.linear_model.LogisticRegression**(*penalty='deprecated'*, ***, *C=1.0*, *l1_ratio=0.0*, *dual=False*, *tol=0.0001*, *fit_intercept=True*, *intercept_scaling=1*, *class_weight=None*, *random_state=None*, *solver='lbfgs'*, *max_iter=100*, *verbose=0*, *warm_start=False*, *n_jobs=None*) [source]

The solvers 'lbfgs', 'newton-cg', 'newton-cholesky' and 'sag' support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization (but not both, i.e. elastic-net), with a dual formulation only for the L2 penalty. The Elastic-Net (combination of L1 and L2) regularization is only supported by the 'saga' solver.

For multiclass problems (whenever `n_classes >= 3`), all solvers except 'liblinear' optimize the (penalized) multinomial loss. 'liblinear' only handles binary classification but can be extended to handle multiclass by using `OneVsRestClassifier`.