CS 5785 - Modern Analytics - Lec. 3

Prof. Serge Belongie, Cornell Tech
Scribe: Wilson P., Inna S., Haiwei S., Sonia S.

August 1, 2015

1 Modeling Posterior Probabilities

Continuing discussion on supervised classification methods. Recall that when we studied linear regression on the 0/1 model it seemed unnatural; think of the 2D case where we tried to fit a binary output using a linear function (ramp). Another way of saying this is that we weren't doing a good job of fitting the *posterior probabilities* of the classes with a linear regression.

In linear regression (purple), faraway observations pull on the fit and have "undue influence" on the decision boundary; see Figure 1.[1] Such data points shouldn't matter since they're not anywhere close to the decision boundary. In logistic regression (green), the decision boundary is minimally affected by these points. Also, with a linear model, we don't get anything resembling a posterior probability that lies within $[0, 1]$. Let's look at Bayes' Theorem to see what we can do to fix this.

1.1 Bayes' Theorem

What is the posterior probability (the probability after looking at the evidence)? The key relationship to remember is

$$\text{Posterior Probability} \propto \text{Prior Probability} \cdot \text{Likelihood}$$

Recall that Bayes' theorem tells us

$$\Pr(\theta|X) = \frac{\Pr(\theta)\Pr(X|\theta)}{\Pr(X)}$$

In this expression,

- $\Pr(\theta)$ is the prior probability of the parameter θ (before looking at evidence),
- $\Pr(X|\theta)$ is the likelihood or probability of the evidence given the parameters θ and
- $\Pr(X)$ is the *marginal likelihood*, obtained by summing or integrating out θ. This is the normalization factor.

[1] The figure caption refers to the faraway points as outliers; technically for them to be outliers in this case, they'd need to be red. It's better to refer to that clump of blue points simply as "data far from the margin."

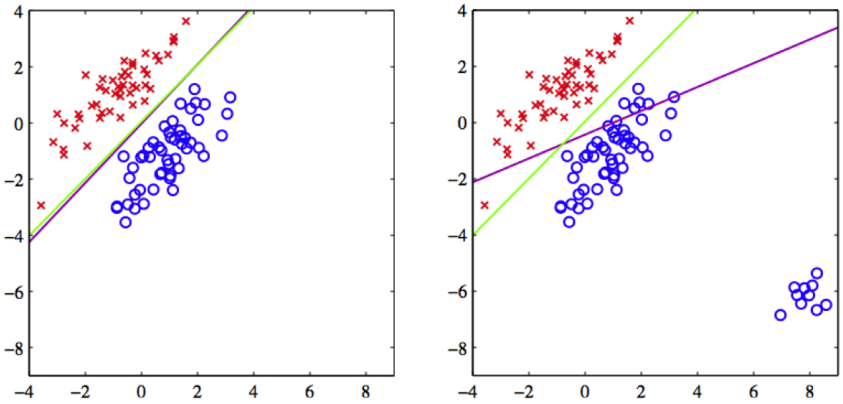


Figure 4.4 The left plot shows data from two classes, denoted by red crosses and blue circles, together with the decision boundary found by least squares (magenta curve) and also by the logistic regression model (green curve), which is discussed later in Section 4.3.2. The right-hand plot shows the corresponding results obtained when extra data points are added at the bottom left of the diagram, showing that least squares is highly sensitive to outliers, unlike logistic regression.

So far we're not specifying a form (e.g., Gaussian) for $\Pr(\cdot)$ for purposes of generality. $\Pr(X|\theta)$ is easy to collect and $\Pr(\theta|X)$ is what we want. For a spam filter, X would capture features of an email message and θ would contain parameters that are indicative of good vs. spam messages, for example. Finding $\Pr(\theta|X)$ is an inference problem: we infer from some measurements what classes the measurements come from. It is what allows us to create a classifier.

In *logistic regression*[2] we want the simplicity of a linear model with the feature of ensuring the posterior probabilities remain in the interval $[0, 1]$.

1.2 Log Odds

A key step in the development of logistic regression is to use the *log odds* or *logit transformation*. In the binary case ($K = 2$) we have two possible classes, $G = 1$ and $G = 2$, for which the regular (i.e., non-log) odds are given by

$$\frac{\Pr(G = 1|X = x)}{\Pr(G = 2|X = x)}$$

Since we only have two possible outcomes, and we know $\sum_{l=1}^{K} \Pr(G = k|X = x) = 1$, we have

$$\Pr(G = 2|X = x) = 1 - \Pr(G = 1|X = x) \tag{1}$$

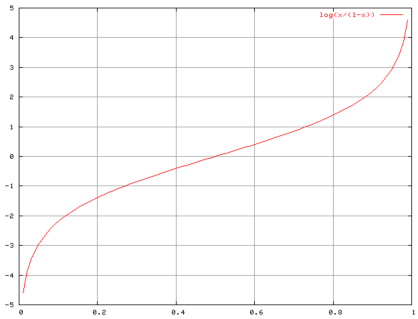
We can think of this case as a weighted coin or equivalently a *Bernoulli* random variable.

In logistic regression we assume that the log odds can be expressed as a linear combination of the input features $x \in \mathbb{R}^p$:

$$\log \frac{\Pr(G = 1|X = x)}{\Pr(G = 2|X = x)} = \beta_0 + \beta^\top x$$

[2]Despite its name, logistic regression is actually a classification method.

Logit or "Log Odds"



$$\operatorname{logit}(p) = \log\left(\frac{p}{1-p}\right) = \log(p) - \log(1-p). \qquad \operatorname{logit}^{-1}(\alpha) = \frac{1}{1+\exp(-\alpha)} = \frac{\exp(\alpha)}{1+\exp(\alpha)}$$

http://en.wikipedia.org/wiki/Logit

Above is an instance of a *generalized linear model*. Compare this with the ordinary linear model from last lecture in which we assumed we could express the output *itself* as a linear combination of the input features. If we exponentiate both sides and use Eqn. 1, we get

$$\Pr(G = 1|X = x) = \frac{\exp(\beta_0 + \beta^\top x)}{1 + \exp(\beta_0 + \beta^\top x)} \tag{2}$$

$$= \frac{1}{1 + \exp\left(-(\beta_0 + \beta^\top x)\right)} \tag{3}$$

This is a sigmoid function, with the form

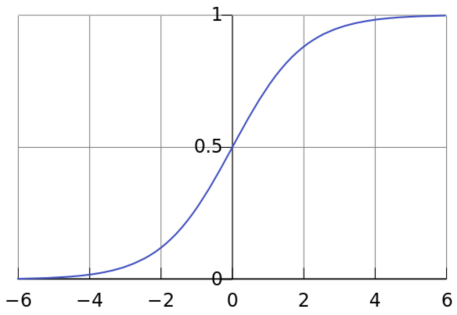
$$\sigma(a) = \frac{1}{1 + e^{-a}}$$

Sigmoid functions have the desirable property of transitioning smoothly from 0 to 1, like a soft step function, which suits it well for representing posterior probabilities. The sigmoid is also differentiable, which makes it more desirable than step functions. The sigmoid is an example of a *saturating nonlinearity*: going left, the values approach and ultimately go to 0, and going right, the values similarly approach and ultimately go to 1. Via this saturating behavior, points far away will not impact the decision boundary as much.

2 Logistic Regression

Now that we have defined a model for the posterior probabilities, let's see how to fit it. This process is called *logistic regression*. We'll continue our focus on the 2-class case with the 0/1 response $y_i = 1$ when $g_i = 1$ and $y_i = 0$ when $g_i = 2$. To simplify notation, we'll let $p_1(x; \theta) = p(x; \theta)$ and $p_2(x; \theta) = 1 - p(x; \theta)$.

Standard logistic sigmoid function



$$P(t) = \frac{1}{1 + e^{-t}}$$

$$\frac{d}{dt}P(t) = P(t) \cdot (1 - P(t)).$$

$$1 - P(t) = P(-t).$$

http://en.wikipedia.org/wiki/Logistic_regression

2.1 Maximum Likelihood Formulation

We are going to use *maximum likelihood* to fit the model in terms of cost. Unlike the case for linear regression, we won't have a closed form solution for this. This will lead us to an instance of an *optimization* method.

The likelihood for N observations is

$$\prod_{i=1}^{N} p_{g_i}(x_i; \theta)$$

where $p_k(x_i; \theta) = \Pr(G = k | X = x_i; \theta)$, assuming the data is *independent and identically distributed* (iid). It's preferable, however, to work with the log of the likelihood instead and turn the products into sums:

$$\ell(\theta) = \sum_{i=1}^{N} \log p_{g_i}(x_i; \theta)$$

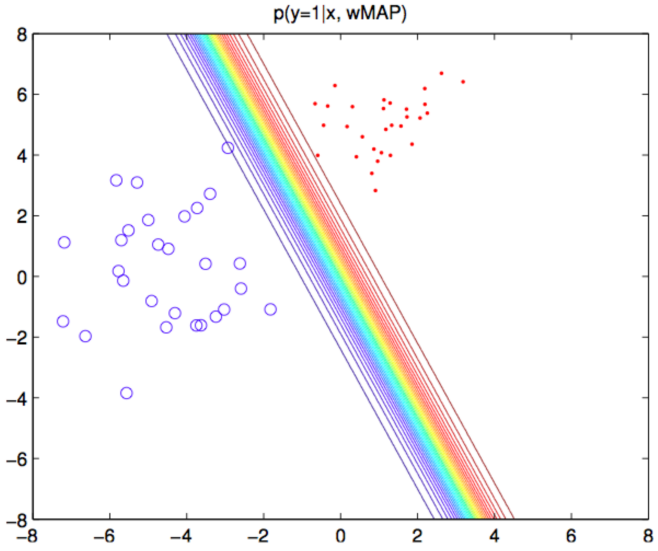
In the simple 2-class case, using $\beta = \{\beta_0, \beta\}$ and tacking on a 1 at the front of x_i to accomodate the intercept, this becomes

$$\ell(\beta) = \sum_{i=1}^{N} \{y_i \log p(x_i; \beta) + (1 - y_i) \log (1 - p(x_i; \beta))\}$$

This is in the form of a *cross entropy* function. The y_i and $1 - y_i$ terms "gate" the two terms in the sum: if y_i is 0 or 1 only one or the other term survives. If we plug in our expression for $\Pr(G = 1 | X = x)$ from Eqn. 2 we get

$$\ell(\beta) = \sum_{i=1}^{N} \left\{ y_i \beta^\top x_i - \log \left(1 + e^{\beta^\top x_i} \right) \right\}$$

This is the quantity we'd like to maximize to fit the model. Performing the fit means determining the optimal $p + 1$ values for β.



[Murphy Fig. 8.6]

In linear regression, in the 2D case, we fit a linear function (a ramp) and thresholded it to get our decision boundary (a line). Again visualizing the 2D case, with logistic regression we will obtain our decision boundary by fitting a sigmoid-shaped function to the data (shaped like a soft cliff) and thresholding it at 0.5. β controls the slope and orientation of the sigmoid to separate the two classes.

Besides doing a better job of modeling the posterior, the saturating effect of the sigmoid means that data points far from the decision boundary do not have undue influence on the resulting fit. This is in contrast to linear regression via least squares as we saw earlier in Bishop Fig. 4.4.

2.2 Maximizing the Log Likelihood

To maximize the log likelihood, we set its derivatives w.r.t. β to zero to obtain the *score equations*:

$$\frac{\partial \ell(\beta)}{\partial \beta} = \sum_{i=1}^{N} x_i(y_i - p(x_i;\beta)) = 0$$

with feature vectors (or data points) $x_i \in \mathbb{R}^{p+1}$, ground truth labels (or *outcome values*) y_i equal to 0 or 1, posterior probabilities $p(x_i;\beta) \in [0,1]$ and a column of $p+1$ zeroes on the RHS. The gradient represented here is a weighted sum of data points with the weight given by the ground truth label minus the thing we're trying to estimate. In this respect, the weight is akin to an error or discrepancy term.

Recall that the likelihood is just a scalar that is a function of $p+1$ values. You can imagine setting up a high dimensional space for β and finding the

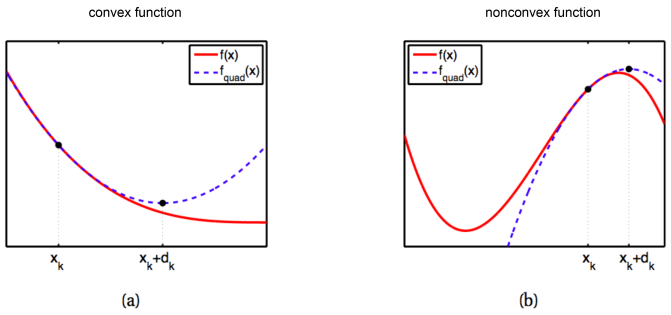


Illustration of Newton's method for minimizing a 1d function

[Murphy Fig. 8.4]

point with the highest likelihood in this space. The gradient can tell us in what direction to take a step to increase the likelihood starting from some (possibly random) initialization.

We have $p + 1$ equations that are nonlinear in β. Recall that $x_i^1 = 1$ since, by convention, the first entry for any feature vector is 1. This tells us that

$$\sum_{i=1}^{N} y_i = \sum_{i=1}^{N} p(x_i; \beta)$$

In other words, the expected number of class 1s (the RHS) matches the observed number of class 1s (the LHS).

To solve the score equations, we will apply the Newton-Raphson algorithm, which uses the second derivative vector counterpart or *Hessian* of the function we wish to optimize; see Murphy Fig. 8.4. In essence we are finding a 2nd order Taylor series approximation of a function around a given point. Given this quadratic function, we can solve for the value that maximizes (or minimizes) it in closed form, pick off the next function value at that point and repeat until convergence. When performing the method, we must perform it many times with different random initializations to ensure we do not get stuck in a local extremum. We'll see how to apply Newton-Raphson to our log likelihood function in the next lecture.