Logistic Regression

Logistic Regression is a linear classification procedure that coincides with the *maximum entropy* approach. In general maximum entropy approaches are those that consciously eschew any assumptions beyond a set of observations.

Entropy is a quantity that has been studied in detail in several disciplines. You have heard of this in the Second Law of Thermodynamics. You will see this in Information Theory and communication, as well as briefly in many initial probability courses, where entropy is a measure of information that one obtains when observing a random variable. In fact, there is a equivalence between the thermodynamic and the information theoretic views, where information engines convert information into thermodynamic work and vice versa. See.

The view we adopt here is from probability theory. For now, the simplest view suffices to explain logistic regression. If a random variable X takes on values $\{x_1, \ldots, x_k\}$ (recall this set is called the support of X) with probabilities $p_1, \ldots, p_k$ respectively, the entropy of X is defined to be

$$H(X) = \sum_{i=1}^{k} p_i \log \frac{1}{p_i},$$

where the log is to base 2. This quantity $H(X)$ can be interpreted as the amount of average information in bits that we obtain when we observe the realization of X. Information theory deals with how to make measurements of X and what happens to the information in the measurements. Generally speaking, suppose we observe another random variable Y. The amount of information we still get from observing X after the observation of Y is called the conditional entropy $H(X|Y)$. It can be shown that $H(X|Y) \leq H(X)$ with equality if X and Y are independent random variables. Since $H(X)$ would have been the amount of information we would have got if we saw X straight up (without seeing Y first), the amount of information Y provides about X is

$$I(X, Y) = H(X) - H(X|Y).$$

Since $H(X) \geq H(X|Y)$, we have that $I(X, Y) \geq 0$.

Among all possible probability distributions, the uniform distribution $p_1 = p_2 = \ldots = p_k = \frac{1}{k}$ maximizes the entropy $H(X)$ (ie we obtain the maximum amount of information when we observe X). If any element is more likely than another, the entropy is lower.

If we had to formalize the notion that we had no information on X (other than that its support $\{x_1, \ldots, x_k\}$), the uniform distribution would be what

we would intuitively choose. The intuition is that any other distribution q could be interpreted as though we had some assumptions a-priori that made some elements of the support $\{x_1, \ldots, x_k\}$ more likely than others. From the discussion in the previous paragraph, this is equivalent to saying that assumptions on the distribution lower entropy. Therefore, intuitively speaking, we would choose the maximum entropy distribution to capture the notion of no assumptions in a quantitative way.

We can take this further. If we adopt a constraint on X, say $\mathbb{E}X = \mu$. Given this constraint on X, the maximum entropy distribution on X would be the one that adopted no further assumptions other than that $\mathbb{E}X = \mu$, namely the maximum entropy distribution among all distributions with $\mathbb{E}X = \mu$. We can calculate this using variational calculus developed for constrained optimization earlier in the course (and in MATH 243/244). We consider the Lagrangian

$$\sum_{i=1}^{k} p_i \log \frac{1}{p_i} - \lambda \sum p_i x_i - \nu \sum p_i,$$

and set the gradient to $\mathbf{0}$ to get that for all i,

$$\log \frac{1}{p_i} + \log e - \lambda x_i - \nu = 0.$$

The above equation implies that for all i,

$$p_i = \exp(\beta_0 + \beta x_i),$$

where β_0 and $\beta_1 = \lambda \ln 2$ are constants, chosen to ensure that

$$\sum p_i x_i = \mu \text{ and } \sum p_i = 1.$$

The constants β_0 and β_1 are less important, what is more important is the exponential form we get for the probabilities. Indeed maximum entropy distributions always have this form, as the following Theorem attests.

Given m different constraints on a random variable X, $\mathbb{E}r_i(X) = \alpha_i$ where for $1 \leq i \leq m$ r_i are real valued functions and $\alpha_i \in \mathbb{R}$, the distribution on X with maximum entropy is (if X is discrete, f below is a probability mass function, and if X is continuous, f is a probability density function):

$$f(x) = \exp\left(\beta_0 + \sum_{i=1}^{m} \beta_i r_i(x)\right).$$

In the above, $\beta_0, \ldots, \beta_m$ are numbers chosen so that f is a probability mass or density function (*i.e.,* integrates/sums to 1) and $\mathbb{E}r_i(X) = \alpha_i$ for $i = 1, \ldots, m$.

Now if we have a classification problem, logistic regression will assign to each example $\mathbf{x}$ in the example space, a conditional probability distribution $p(y|\mathbf{x})$ that denotes the probability the label of $\mathbf{x}$ is y. So if we have two possible labels, every example $\mathbf{x}$ will be assigned a Bernoulli distribution.

Let $\mathcal{X}$ be the set of all possible instances that we will assign a label to. We call $\mathcal{X}$ the instance space. Suppose the classification problem at hand assigns one of k labels (denoted by y) to each instance $\mathbf{x}$. Without loss of generality, let the set of all labels be $\{1, \ldots, k\}$. The set of all instances in $\mathcal{X}$ that get a label j will be called *class* j, denoted by C_j. Therefore, the classification task is to break the space $\mathcal{X}$ into k regions, that is

$$\mathcal{X} = \bigcup_{j=1}^{k} C_j.$$

We would like C_j to be disjoint for different j as well, but will tolerate putting some instances $\mathbf{x}$ into multiple classes, when we do not have a unique preference for the label. In a slight abuse of notation therefore, we will call $\mathcal{X} = \bigcup_{j=1}^{k} C_j$ a partition despite the fact that C_j are not necessarily disjoint.

Rather than specify the partition directly, we take what is called a *generative* approach. We model the probability distribution $f(X|Y = j)$ (the probability distribution over the instance space given the label, once again f is a pdf if $\mathcal{X}$ is uncountable and a pmf if $\mathcal{X}$ is countable). To get the partition, we use either a maximum likelihood (ML) approach or a maximum a-posteriori (MAP) approach (Bayesian). In the ML approach, we put into C_j every instance $\mathbf{x}$ for which $f(X = \mathbf{x}|Y = j) \geq f(X = \mathbf{x}|Y = l)$ for all $l \neq j$, namely

$$C_j = \left\{ \mathbf{x} \in \mathcal{X} : j = \arg\max_l f(X|Y = l) \right\}.$$

In the MAP approach, we have a prior distribution on the class labels, $\mathbb{P}(Y)$, and we assign

$$C_j = \left\{ \mathbf{x} \in \mathcal{X} : j = \arg\max_l \mathbb{P}(Y = l|X = \mathbf{x}) \right\},$$

where $\mathbb{P}(Y = l|X = \mathbf{x})$ is calculated via Bayes rule using both the prior $\mathbb{P}(Y)$ and the modeled generative distributions $f(X|Y)$ for all values of Y. Namely

$$\mathbb{P}(Y = l|X = \mathbf{x}) = \frac{f(X = \mathbf{x}|Y = l)\mathbb{P}(Y = l)}{f(X = \mathbf{x})} = \frac{f(X = \mathbf{x}|Y = l)\mathbb{P}(Y = l)}{\sum_{i=1}^{k} f(X = \mathbf{x}|Y = i)\mathbb{P}(Y = i)}.$$

Logistic regression models each class, $f(X|Y = j)$, $j = 1, \ldots, k$, using the maximum entropy model subject to constraints on $\mathbb{E}[X|Y = j]$. Note that X is a vector with d components, so $\mathbb{E}[X|Y = j]$ corresponds to a \$d-\$component vector as well. We estimate $\mathbb{E}[X|Y = j]$ from the m training instances $\mathbf{x}_i$ and their labels y_i, \$i=1 ,. . . ,m \$, and impose the constraint:

$$\mathbb{E}[X|Y = j] = \frac{\sum_{\substack{i=1 \\ y_i=j}}^{m} \mathbf{x}_i}{N_j}, \tag{1}$$

where N_j is the number of training examples that have j as their label. Such a maximum entropy model, from the Theorem above, is

$$f(X = \mathbf{x}|Y = j) = \exp\left(\beta_{j0} + \beta_j^T \mathbf{x}\right),$$

where $\beta_{j0} \in \mathbb{R}$, $\beta_j \in \mathbb{R}^d$ has the same number of coordinates as the training instance $\mathbf{x}$, with β_{j0} and β_j chosen to satisfy the fact that $\int f(X|Y = j) = 1$ (or $\sum f(X|Y = j) = 1$ if $\mathcal{X}$ is countable) and the constraint in Equation˜(1) above.

Generally one of the classes, say class 1, is taken as a reference, and we have for $j = 2, \ldots, k$, using Bayes Rule on both the numerator and denominator (and cancelling $f(\mathbf{x})$ in both numerator and denominator):

$$\frac{f(X = \mathbf{x}|Y = j)}{f(X = \mathbf{x}|Y = 1)} = \frac{\mathbb{P}(Y = j|X = \mathbf{x})}{\mathbb{P}(Y = 1|X = \mathbf{x})} \cdot \frac{\mathbb{P}(Y = 1)}{\mathbb{P}(Y = j)}.$$

Rearranging, we get

$$\frac{\mathbb{P}(Y = j|X = \mathbf{x})}{\mathbb{P}(Y = 1|X = \mathbf{x})} = \exp\left(\tilde{\beta}_{j0} + \tilde{\beta}_j^T \mathbf{x}\right). \tag{2}$$

Here $\tilde{\beta}_{j0} = \beta_{j0} - \beta_{10} + \ln \frac{\mathbb{P}(Y=j)}{\mathbb{P}(Y=1)}$, and $\tilde{\beta}_j = \beta_j - \beta_1$, where β_{j0}, β_j (respectively β_{10} and β_1) were the constants we used for maximum entropy modeling for classes j (respectively 1). For the ML case, $\mathbb{P}(Y = j) = \frac{1}{k}$ for all j, therefore $\tilde{\beta}_{j0} = \beta_{j0} - \beta_{10}$.

The above equation˜(2) is the classical formulation for logistic regression. To summarize, we used Maximum Entropy modeling of X given each class label, subject to first moment (expectation) constraints on X given the class label. This modeling is equivalent to˜(2).

While calculating the parameters from data, we do not calculate β_{j0} or β_j from data. Doing so with the ML approach would require us to find β_{j0} and β_j that satisfy

$$\mathbb{E}[X|Y=j] = \frac{\sum_{i:y_i=j} x_i}{n_j},$$

where n_i is the number of examples that belong to class j. However, analytical expressions of $\mathbb{E}[X|Y=j]$ are harder to work with. Instead, implementations fit

$$\sum_{i=1}^{m} \mathbb{P}(Y_i = j|\mathbf{x}_i) = \sum_{i:y_i=j} x_i.$$

The left side is what you would get if you took~(2) and optimized for $\tilde{\beta}$ using the ML procedure.

For those who are interested, here is a brief summary. We set up the log likelihood of the training data. Using~(2) and the fact that $\sum_j \mathbb{P}(Y = j|X = \mathbf{x}) = 1$ for all $\mathbf{x}$, we have for all $2 \leq j \leq k$,

$$\mathbb{P}(Y = j|X = \mathbf{x}) = \frac{\exp\left(\tilde{\beta}_{j0} + \tilde{\beta}_j^T \mathbf{x}\right)}{1 + \sum_{i=2}^{k} \exp\left(\tilde{\beta}_{i0} + \tilde{\beta}_i^T \mathbf{x}\right)}$$

and

$$\mathbb{P}(Y = 1|X = \mathbf{x}) = \frac{1}{1 + \sum_{i=2}^{k} \exp\left(\tilde{\beta}_{i0} + \tilde{\beta}_i^T \mathbf{x}\right)}.$$

Note that there are $k-1$ sets of parameters $\tilde{\beta}_i$, for $i = 2, \ldots, k$.

To simplify exposition, we will consider the case $k = 2$, so there is only one set of parameters $\tilde{\beta}$. In this case, it is convenient to think of the labels as 0 and 1 (instead of 1 and 2), so we can write

$$\mathbb{P}(Y = y|X = \mathbf{x}) = \frac{\exp\left(y\tilde{\beta}_0 + y\tilde{\beta}^T \mathbf{x}\right)}{\sum_{\tilde{y}=0}^{1} \exp\left(\tilde{y}\tilde{\beta}_0 + \tilde{y}\tilde{\beta}^T \mathbf{x}\right)}.$$

Write the likelihood of training examples $(\mathbf{x}_i, y_i)$ given examples $\mathbf{x}_1, \ldots, \mathbf{x}_m$, assuming that the label Y_i given the instance $\mathbf{x}_i$ is independent of all other examples to yield

$$\mathbb{P}(y_1, \ldots, y_n|\mathbf{x}_1, \ldots, \mathbf{x}_n) = \prod_{i=1}^{m} \mathbb{P}(y_i|\mathbf{x}_i) = \frac{\prod_{i=1}^{m} \exp\left(y_i\tilde{\beta}_0 + y_i\tilde{\beta}^T \mathbf{x}\right)}{\left(\sum_{\tilde{y}=0}^{1} \exp\left(\tilde{y}\tilde{\beta}_0 + \tilde{y}\tilde{\beta}^T \mathbf{x}\right)\right)^m}.$$

It is easier to work with log-likelihood,

$$\log \prod \mathbb{P}(y_i|\mathbf{x}_i) = \sum_i^m \log \mathbb{P}(y_i|\mathbf{x}_i).$$

We just find the value of $\tilde{\beta}$ that maximizes the above log likelihood. If you read carefully, you will ask why maximizing the above should give $\tilde{\beta}$ parameters that satisfy~(**??**). Well, it is easy to verify if you note that

$$\frac{d}{dx}\frac{e^x}{1+e^x} = \frac{e^x}{1+e^x}\left(1 - \frac{e^x}{1+e^x}\right),$$

and

$$\frac{d}{dx}\frac{1}{1+e^x} = -\frac{1}{1+e^x}\left(1 - \frac{1}{1+e^x}\right),$$

so the chain rule tells us

$$\begin{aligned}
\nabla_{\tilde{\beta}} \log \prod \mathbb{P}(y_i|\mathbf{x}_i) &= \sum_i^m \nabla_{\tilde{\beta}} \log \mathbb{P}(y_i|\mathbf{x}_i) \\
&= \sum_{i:y_i=1} \frac{1}{\mathbb{P}(y_i|\mathbf{x}_i)} \mathbb{P}(y_i|\mathbf{x}_i)(1 - \mathbb{P}(y_i|\mathbf{x}_i)) \begin{bmatrix} 1 \\ \mathbf{x}_i \end{bmatrix} \\
&\quad - \sum_{i:y_i=0} \frac{1}{\mathbb{P}(y_i|\mathbf{x}_i)} \mathbb{P}(y_i|\mathbf{x}_i)(1 - \mathbb{P}(y_i|\mathbf{x}_i)) \begin{bmatrix} 1 \\ \mathbf{x}_i \end{bmatrix} \\
&= \sum_{i=1}^m (1 - \mathbb{P}(y_i|\mathbf{x}_i)) \begin{bmatrix} 1 \\ \mathbf{x}_i \end{bmatrix}
\end{aligned}$$

When solving for the parameters, it is often better to add in a $\updownarrow_2$ regularization on $\tilde{\beta}$s. To see why, note that if the data is linearly separable, there exists parameter values $\tilde{\beta}$ such that ensures $\mathbb{P}(y_i|\mathbf{x}_i) \geq \mathbb{P}(1 - y_i|\mathbf{x}_i)$ for all i. If this is the case, scaling $\tilde{\beta}$ by any number greater than 1 increases the likelihood, hence pushing the optimal parameters to infinity, a meaningless exercise. In these cases, it is useful to limit the length of $\tilde{\beta}$. As we saw in the Ridge Regression module, we can do this by using ℓ_2 regularization. Implementations (including scikit-learn) use this regularization by default.