Machine Learning - Michaelmas Term 2016
Lecture 8 : Classification: Logistic Regression

Lecturer: Varun Kanade

In the previous lecture, we studied two different generative models for classification—Naïve Bayes and Gaussian Discriminant Analysis. Today, we'll study a discriminative model called Logistic Regression.[1]

1 Logistic Regression

In its most basic form, logistic regression is a method for *binary classification*, *i.e.,* when there are only two classes. In such a setting it is mathematically convenient to label these classes as 0 and 1 (as we'll do in this lecture), or -1 and 1 (as we'll do in the next lecture). However, it is important to bear in mind that this is purely a mathematical convenience.

Logistic Regression is a discriminative model, *i.e.,*, we only model the conditional distribution over the output y, given the inputs $\mathbf{x}$ and model parameters $\mathbf{w}$,

$$p(y \mid \mathbf{w}, \mathbf{x}) \tag{1}$$

The specific form of this model is the following. Let us suppose that the inputs are $\mathbf{x} \in \mathbb{R}^D$. Furthermore, we'll assume that an extra column has been added, say $x_0 = 1$, for each datapoint so that we do not need to handle the constant term explicitly. Then the logistic regression model for the conditional distribution over y, given $\mathbf{x}$ and $\mathbf{w}$ is:

$$p(y \mid \mathbf{w}, \mathbf{x}) = \text{Bernoulli}(\sigma(\mathbf{w} \cdot \mathbf{x})), \tag{2}$$

where $\sigma : \mathbb{R} \to (0, 1)$ is the sigmoid function given by $\sigma(t) = \frac{1}{1+e^{-t}}$. (Note that as $t \to -\infty$, $\sigma(t) \to 0$ and as $t \to \infty$, $\sigma(t) \to 1$.) We encountered this function in the previous lecture; the shape of the function is shown in Figure 1. Recall that σ maps $\mathbb{R} \to (0, 1)$, so $\sigma(t)$ can be interpreted as a probability. Thus in (2), y is modelled as Bernoulli random variable with expectation $\sigma(\mathbf{w} \cdot \mathbf{x})$. Recall, that a Bernoulli random variable with mean (parameter) θ, takes the value 1 with probability θ and the value 0 with probability $1 - \theta$.

As a result, the specific functional form of the model, $\sigma(\mathbf{w} \cdot \mathbf{x})$ can be interpreted as estimating the probability that the class label is 1.[2]

1.1 Prediction Using Logistic Regression

Let us suppose that we have estimated the model parameters and now wish to predict the class for a new input $\mathbf{x}_{\text{new}}$. The model specifies the probability that the class label is 1,

$$p(y_{\text{new}} = 1 \mid \mathbf{x}_{\text{new}}, \mathbf{w}) = \sigma(\mathbf{w} \cdot \mathbf{x}_{\text{new}}) = \frac{1}{1 + \exp(-\mathbf{w} \cdot \mathbf{x}_{\text{new}})} \tag{3}$$

[1] As if it weren't bad enough, that a generative model has the word discriminant in its name, as in the case of Gaussian Discriminant Analysis (*i.e.,* QDA and LDA), despite being a method for classification, logistic regression has 'regression' in its name. The reason why this is will soon be clear.

[2] In fact this functional form is one of a family of models referred to as *generalised linear models*. These are models where the expected output is modelled as a linear function composed with a univariate function, *i.e.,* $\mathbb{E}[y \mid \mathbf{x}, \mathbf{w}, f] = f(\mathbf{w} \cdot \mathbf{x})$ for $f : \mathbb{R} \to \mathbb{R}$. These models can be used to capture (limited) non-linearities without resorting to basis function expansion and are also used for regression problems; logistic regression may be viewed as one of these models, although it is almost exclusively used for classification. As an aside, to further confuse matters, there is a thing called *general linear models* (not generali*s*ed) that are different from generalised linear models!

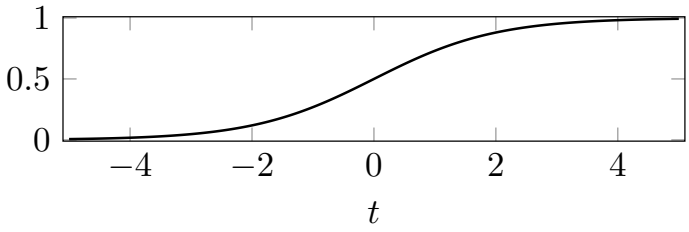
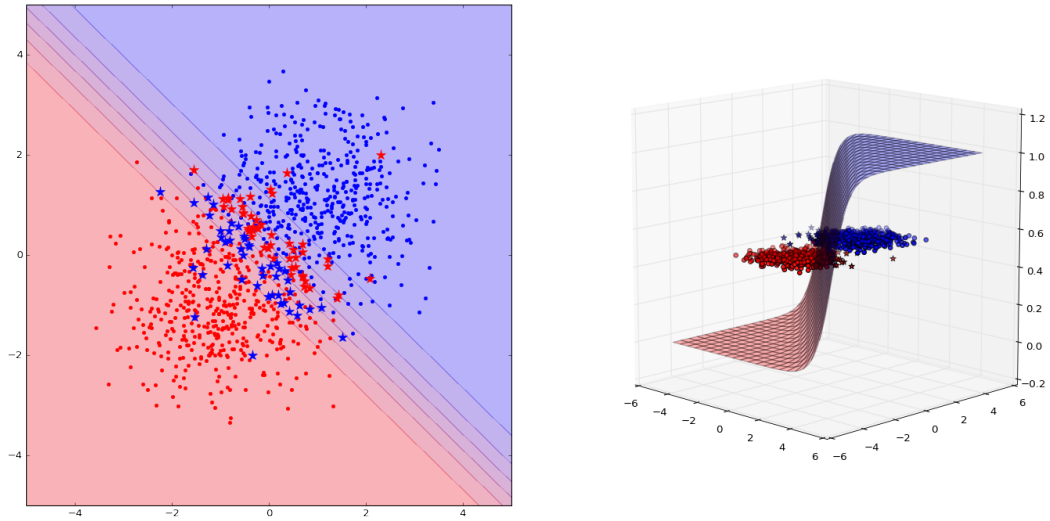


Figure 1: The sigmoid function.



(a) (b)

Figure 2: (a) Scatter plot of the data and the contour of the the class labels. The data marked by '*' markers represent mistakes made by the logistic regression classifier. (b) The same data but projected in three dimensions (the z values of datapoints are irrelevant and chosen to make the errors more visible); the plot also shows the shape of the function $\sigma(\mathbf{w} \cdot \mathbf{x})$

Notice the similarity of this prediction rule with the one we used in the case of LDA with two classes. The prediction rule has exactly the same functional form, however, the method used to obtain model parameters are very different. In order to make an actual class prediction, we can simply threshold at $\frac{1}{2}$, thus we have:

$$\widehat{y}_{\text{new}} = \mathbb{1}(\sigma(\mathbf{w} \cdot \mathbf{x}_{\text{new}}) \geq \frac{1}{2}) = \mathbb{1}(\mathbf{w} \cdot \mathbf{x}_{\text{new}} \geq 0) \tag{4}$$

From the functional form above, it is clear that the separating boundary is linear (a hyperplane in high dimensions). Figure 2 shows the separating boundary as well as the shape of the function $\sigma(\mathbf{w} \cdot \mathbf{x})$ for a logistic regression model trained on a simple synthetic dataset.

1.2 Likelihood of Logistic Regression

Let us now write the likelihood of observing the data $\mathcal{D} = \langle (\mathbf{x}_i, y_i) \rangle_{i=1}^{N}$ in terms of the parameters $\mathbf{w}$. Since this is a discriminative model, we are not concerned with modelling the distribution over the inputs $\mathbf{x}_i$, but can in fact think of them as fixed. The only randomness is in the observed values of y_i. (Also, we've assumed that there is a constant 1 feature in the input, so we will not model the bias/constant term separately.)

We can write the likelihood of observing the outputs $\mathbf{y}$ given the model parameters $\mathbf{w}$ and

the inputs $\mathbf{X}$ as:

$$p(\mathbf{y} \mid \mathbf{X}, \mathbf{w}) = \prod_{i=1}^{N} \sigma(\mathbf{w}^\mathsf{T}\mathbf{x}_i)^{y_i} \cdot (1 - \sigma(\mathbf{w}^\mathsf{T}\mathbf{x}_i)^{1-y_i} \tag{5}$$

Recall that the matrix $\mathbf{X}$ is constructed by choosing its i^{th} row to be $\mathbf{x}_i^\mathsf{T}$. To keep notation tidy, we'll use $\mu_i = \sigma(\mathbf{w}^\mathsf{T}\mathbf{x}_i)$. As always, it'll be more convenient to deal with the negative log-likelihood than the likelihood itself. The negative log-likelihood can be expressed as:

$$\mathrm{NLL}(\mathbf{y} \mid \mathbf{X}, \mathbf{w}) = -\sum_{i=1}^{N} \big(y_i \ln \mu_i + (1 - y_i) \ln(1 - \mu_i)\big) \tag{6}$$

Let us first look at the contribution made by a single datapoint $(\mathbf{x}_i, y_i)$ to the negative log-likelihood. Since $\mu_i = \sigma(\mathbf{x}_i, y_i)$ this quantity is given by:

$$\mathrm{NLL}(y_i \mid \mathbf{x}_i, \mathbf{w}) = -(y_i \log \mu_i + (1 - y_i) \log(1 - \mu_i))$$

The form of this expression is reminiscent of the cross-entropy (discussed in Lecture 3). In fact it is exactly the cross entropy, where the observation y_i is deterministically either 0 or 1, and μ_i represents the probability that model estimates the outcome as 1. Let us consider the case when $y_i = 1$; since $\mu_i \in (0, 1)$, $\mathrm{NLL}(y_i \mid \mathbf{x}_i, \mathbf{w}) = -y_i \log \mu_i$ in this case. Thus as $\mu_i \to 1$, we have $\mathrm{NLL}(y_i \mid \mathbf{w}, \mathbf{x}_i) \to 0$ and as $\mu_i \to 0$, $\mathrm{NLL}(y_i \mid \mathbf{w}, \mathbf{x}_i) \to \infty$. Thus, there is a hefty penalty for being overconfident about a wrong prediction!

1.2.1 Iteratively Reweighted Least Squares

Let us now return to the question of estimating the parameters $\mathbf{w}$ by minimising the negative log-likelihood given in (6). We will be a bit short on details for computing the gradient and the Hessian; this is left as an exercise on Problem Sheet 3. The gradient and the Hessian of the NLL are given below:

$$\nabla_{\mathbf{w}} \mathrm{NLL}(\mathbf{y} \mid \mathbf{X}, \mathbf{w}) = \sum_{i=1}^{N} \mathbf{x}_i(\mu_i - y_i) = \mathbf{X}^\mathsf{T}(\boldsymbol{\mu} - \mathbf{y}) \tag{7}$$

$$\mathbf{H}_{\mathbf{w}} = \mathbf{X}^\mathsf{T}\mathbf{S}\mathbf{X} \tag{8}$$

where $\mathbf{S}$ is a diagonal matrix, with $S_{ii} = \mu_i(1 - \mu_i)$.

Let us verify that the Hessian is positive semi-definite. Recall that a $D \times D$ symmetric matrix $\mathbf{A}$ is positive semi-definite, if for every $\mathbf{z} \in \mathbb{R}^D$, $\mathbf{z}^\mathsf{T}\mathbf{A}\mathbf{z} \geq 0$. In the case of the Hessian defined in (8), let $\mathbf{z}' = \mathbf{X}\mathbf{z}$, so that $\mathbf{z}^\mathsf{T}\mathbf{H}_{\mathbf{w}}\mathbf{z} = \sum_{i=1}^{D} S_{ii}(z_i')^2$. Since $S_{ii} = \mu_i(1 - \mu_i)$ for $\mu_i \in (0, 1)$, this term must be non-negative. (If $N > D$ and $\mathbf{X}$ has rank D, then in fact, in this case $\mathbf{H}_{\mathbf{w}}$ is postive definite, *i.e.,* $\mathbf{z}\mathbf{H}_{\mathbf{w}}\mathbf{z} \geq 0$ and equality holds if and only if $\mathbf{z} = \mathbf{0}$. Clearly $\sum_{i=1}^{D} S_{ii}(z_i')^2 = 0$ if and only if $\mathbf{z}' = \mathbf{0}$; since $\mathbf{X}$ has rank D and $N > D$, $\mathbf{z}' = \mathbf{0}$ if and only if $\mathbf{z} = \mathbf{0}$.) Since, the Hessian is positive semi-definite everywhere, we know that the negative log-likelihood NLL is a convex function of $\mathbf{w}$. Thus, we can estimate $\mathbf{w}$ using standard convex optimisation methods (although if $\mathbf{X}$ does not have rank D, we may be in a degenerate case).

If the dimension D is modest, then we can apply Newton's method to estimate $\mathbf{w}$. Let $\mathbf{w}_t$ be the estimated parameters after t Newton steps. Let us denote the gradient and the Hessian at this point by $\mathbf{g}_t$ and $\mathbf{H}_t$, where

$$\mathbf{g}_t = \mathbf{X}^\mathsf{T}(\boldsymbol{\mu}_t - \mathbf{y}) = -\mathbf{X}^\mathsf{T}(\mathbf{y} - \boldsymbol{\mu}_t)$$
$$\mathbf{H}_t = \mathbf{X}^\mathsf{T}\mathbf{S}_t\mathbf{X}$$

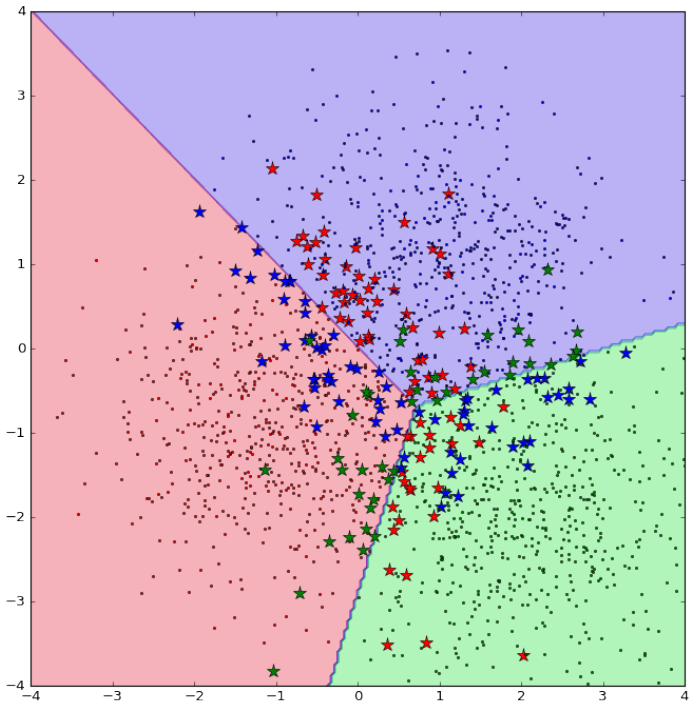


Figure 3: Multiclass Logistic Regression

As per the Newton update rule, we have:

$$\begin{aligned}
\mathbf{w}_{t+1} &= \mathbf{w}_t - \mathbf{H}_t^{-1}\mathbf{g}_t \\
&= \mathbf{w}_t + (\mathbf{X}^\mathsf{T}\mathbf{S}_t\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}(\mathbf{y} - \boldsymbol{\mu}_t) \\
&= (\mathbf{X}^\mathsf{T}\mathbf{S}_t\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{S}_t(\mathbf{X}\mathbf{w}_t + \mathbf{S}_t^{-1}(\mathbf{y} - \boldsymbol{\mu}_t)) \\
&= (\mathbf{X}^\mathsf{T}\mathbf{S}_t\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{S}_t\mathbf{z}_t
\end{aligned}$$

Where $\mathbf{z}_t = \mathbf{X}\mathbf{w}_t + \mathbf{S}_t^{-1}(\mathbf{y} - \boldsymbol{\mu}_t)$. Then $\mathbf{w}_{t+1}$ is a solution of the following problem:

$$\underset{\mathbf{w}}{\text{minimise}} \quad \sum_{i=1}^{N} S_{t,ii}(z_{t,i} - \mathbf{w}^\mathsf{T}\mathbf{x}_i)^2 \tag{9}$$

It is for this reason that this method is called the iteratively reweighted least squares method.

2 Multiclass Logistic Regression

Let us now consider a 'logistic regression'-like model when there are more than two classes. We'll consider some alternative approaches in the next lecture that use binary classifiers generically to obtain multi-class classifiers. However, in the case of logistic regression, it is relatively easy to modify the model to handle more than two classes.

Let us suppose that we have C classes denoted by $\{1, \ldots, C\}$. We'll have a set of parameter $\mathbf{w}_c \in \mathbb{R}^D$ for every $c \in C$. We can express these as a $D \times C$ matrix $\mathbf{W}$, where the c^{th} column of $\mathbf{W}$ is $\mathbf{w}_c$. Then the discriminative model is defined by the conditional distribution over the

input y, given $\mathbf{W}$ and $\mathbf{x}$ as,

$$p(y = c \mid \mathbf{x}, \mathbf{W}) = \frac{\exp(\mathbf{w}_c \cdot \mathbf{x})}{\sum_{c'=1}^{C} \exp(\mathbf{w}_{c'} \cdot \mathbf{x})} \tag{10}$$

Note that the RHS of the above equation is simply a *softmax*. We can view the softmax as a function that maps a vector (with positive or negative entries) to a probability distribution as follows: Let $\mathbf{a} \in \mathbb{R}^D$ be a some vector then,

$$\text{softmax}([a_1, \ldots, a_D]^\mathsf{T}) = \left[\frac{e^{a_1}}{Z}, \ldots, \frac{e^{a_D}}{Z}\right]^\mathsf{T}, \tag{11}$$

where $Z = \sum_{i=1}^{D} e^{a_i}$. Thus, we can simply rewrite (10) as

$$p(y \mid \mathbf{x}, \mathbf{W}) = \text{softmax}\left([\mathbf{w}_1 \cdot \mathbf{x}, \ldots, \mathbf{w}_C \cdot \mathbf{x}]^\mathsf{T}\right) \tag{12}$$

Note that the decision boundaries between different classes are still linear (see Fig. 3).

As in the case of (binary) logistic regression, we can write out the negative log-likelihood, show that it is convex and use a convex optimisation approach to estimate the parameters $\mathbf{W}$. The details are given in Murphy (2012, Chap. 8.3.7). However, we'll omit the details here. We'll return to much more general models that use the *softmax* and the *sigmoid* in the context of neural networks.

3 Discussion

In these two lectures, we've seen generative and discriminative models for logistic regression. In general there is no clear way of deciding which type of model is preferable; there are advantages and disadvantages to both approaches. Refer to Murphy (2012, Chap. 8.6) for a detailed comparison of the two approaches.

It is worth pointing out that may ideas in machine learning are can be applied in different contexts. For example, it is possible to use basis function expansion and regularization methods for logistic regression, as we did in the case of linear regression. (In fact regularisation may be necessary if the data itself is linearly separable. Why?) So if we are faced with a classification problem, where we believe that the clasification boundaries should be non-linear, we could perform polynomial (or kernel-based) basis expansion and use ℓ_1 or ℓ_2 regularisation if we believe that there is a risk of overfitting.

References

Kevin P. Murphy. *Machine Learning : A Probabilistic Perspective*. MIT Press, 2012.