CS 188 Introduction to Artificial Intelligence
Fall 2023 Note 22

Author (all other notes): Nikhil Sharma

Author (Bayes' Nets notes): Josh Hug and Jacky Liang, edited by Regina Wang

Author (Logic notes): Henry Zhu, edited by Peyrin Kao

Credit (Machine Learning and Logic notes): Some sections adapted from the textbook *Artificial Intelligence: A Modern Approach*.

Last updated: August 26, 2023

Logistic Regression

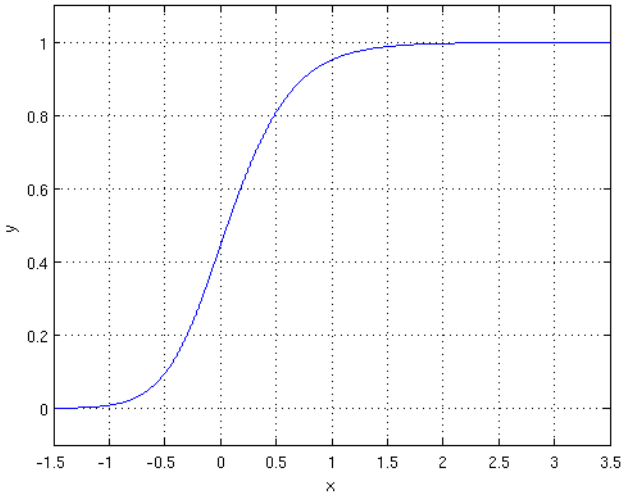
Let's again think about our previous method of linear regression. In it, we assumed that the output is a numeric real-valued number.

What if we instead want to predict a categorical variable? Logistic regression allows us to turn a linear combination of our input features into a probability using the logistic function:

$$h_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}}.$$

It is important to note that though logistic regression is named as regression, this is a misnomer. Logistic regression is used to solve classification problems, not regression problems.

The logistic function $g(z) = \frac{1}{1+e^{-z}}$ is frequently used to model binary outputs. Note that the output of the function is always between 0 and 1, as seen in the following figure:



Intuitively, the logistic function models the probability of a data point belonging in class with label 1. The reason for that is that the output of the logistic function is bounded between 0 and 1, and we want our model

to capture the probability of a feature having a specific label. For instance, after we have trained logistic regression, we obtain the output of the logistic function for a new data point. If the value of the output is greater than 0.5 we classify it with label 1 and we classify it with label 0 otherwise. More specifically, we model the probabilities as follows:

$$P(y=+1|\mathbf{f}(\mathbf{x});\mathbf{w}) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{f}(\mathbf{x})}}$$

and

$$P(y=-1|\mathbf{f}(\mathbf{x});\mathbf{w}) = 1 - \frac{1}{1+e^{-\mathbf{w}^T\mathbf{f}(\mathbf{x})}},$$

where we use $\mathbf{f}(\mathbf{x})$ to denote the function (which often is the identity) of the feature vector $\mathbf{x}$ and the semi-colon ; denotes that the probability is a function of the parameter weights $\mathbf{w}$.

A useful property to note is that the logistic function is that the derivative is $g'(z) = g(z)(1-g(z))$.

How do we train the logistic regression model? The loss function for logistic regression is the $L2$ loss $Loss(\mathbf{w}) = \frac{1}{2}(\mathbf{y} - h_{\mathbf{w}}(\mathbf{x}))^2$. Since a closed form solution is not possible for the logistic regression, we estimate the unknown weights $\mathbf{w}$ via gradient descent. For that we need to compute the gradient of the loss function using the chain rule of differentiation. The gradient of the loss function with respect to the weight of coordinate i is given by:

$$\frac{\partial}{\partial w_i}\frac{1}{2}(y - h_{\mathbf{w}}(\mathbf{x}))^2 = (y - h_{\mathbf{w}}(\mathbf{x}))\frac{\partial}{\partial w_i}(y - h_{\mathbf{w}}(\mathbf{x})) = -(y - h_{\mathbf{w}}(\mathbf{x}))h_{\mathbf{w}}(\mathbf{x})(1 - h_{\mathbf{w}}(\mathbf{x}))x_i,$$

where we used the fact that the gradient of the logistic function $g(z) = \frac{1}{1+e^{-z}}$ satisfies $g'(z) = g(z)(1-g(z))$. We can then estimate the weights using gradient descent and then predict, as detailed above.

Multi-Class Logistic Regression

In multi-class logistic regression, we want to classify data points into K distinct categories, rather than just two. Thus, we want to build a model that output estimates of the probabilities for a new data point to belong to each of the K possible categories. For that reason we use the **softmax function** in place of the logistic function, which models the probability of a new data point with features $\mathbf{x}$ having label i as follows:

$$P(y=i|\mathbf{f}(\mathbf{x});\mathbf{w}) = \frac{e^{\mathbf{w}_i^T\mathbf{f}(\mathbf{x})}}{\sum_{k=1}^{K} e^{\mathbf{w}_k^T\mathbf{f}(\mathbf{x})}}.$$

Note that these probability estimates add up to 1, so they constitute a valid probability distribution. We estimate the parameters $\mathbf{w}$ to maximize the likelihood that we observed the data. Assume that we have observed n labelled data points $(\mathbf{x}_i, y_i)$. The likelihood, which is defined as the joint probability distribution of our samples, is denoted with $\ell(\mathbf{w}_1, \ldots, \mathbf{w}_K)$ and is given by:

$$\ell(\mathbf{w}_1, \ldots, \mathbf{w}_K) = \prod_{i=1}^{n} P(y_i|\mathbf{f}(\mathbf{x}_i);\mathbf{w}).$$

To compute the values of the parameters $\mathbf{w}_i$ that maximize the likelihood, we compute the gradient of the likelihood function with respect to each parameter, set it equal to zero, and solve for the unknown

parameters. If a closed form solution is not possible, we compute the gradient of the likelihood and we use gradient ascent to obtain the optimal values.

A common trick we do to simplify these calculations is to first take the logarithm of the likelihood function which will break the product into summations and simplify the gradient calculations. We can do this because the logarithm is a strictly increasing function and the transformation will not affect the maximizers of the function.

For the likelihood function we need a way to express the probabilities $P(y_i|\mathbf{f}(\mathbf{x}_i);\mathbf{w})$ in which $y \in \{1,\dots,K\}$. So for each data point i, we define K parameters $t_{i,k}$, $k = 1,\dots,K$ such that $t_{i,k} = 1$ if $y_i = k$ and 0 otherwise. Hence, we can now express the likelihood as follows:

$$\ell(\mathbf{w}_1,\dots,\mathbf{w}_K) = \prod_{i=1}^{n}\prod_{k=1}^{K}\left(\frac{e^{\mathbf{w}_k^T\mathbf{f}(\mathbf{x}_i)}}{\sum_{\ell=1}^{K} e^{\mathbf{w}_\ell^T\mathbf{f}(\mathbf{x}_i)}}\right)^{t_{i,k}}$$

and we also obtain for the log-likelihood that:

$$\log\ell(\mathbf{w}_1,\dots,\mathbf{w}_K) = \sum_{i=1}^{n}\sum_{k=1}^{K} t_{i,k}\log\left(\frac{e^{\mathbf{w}_k^T\mathbf{f}(\mathbf{x}_i)}}{\sum_{\ell=1}^{K} e^{\mathbf{w}_\ell^T\mathbf{f}(\mathbf{x}_i)}}\right)$$

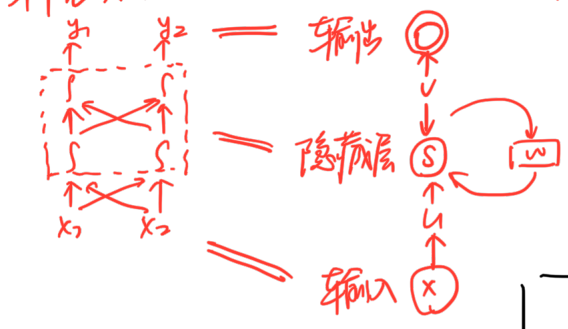
Now that we have an expression for the objective we must estimate the $\mathbf{w}_i$s such that they maximize that objective.

In the example of the multi-class logistic regression the gradient with respect to $\mathbf{w}_j$ is given by:

$$\nabla_{\mathbf{w}_j}\log\ell(\mathbf{w}) = \sum_{i=1}^{n}\nabla_{\mathbf{w}_j}\sum_{k=1}^{K} t_{i,k}\log\left(\frac{e^{\mathbf{w}_k^T\mathbf{f}(\mathbf{x}_i)}}{\sum_{\ell=1}^{K} e^{\mathbf{w}_\ell^T\mathbf{f}(\mathbf{x}_i)}}\right) = \sum_{i=1}^{n}\left(t_{i,j} - \frac{e^{\mathbf{w}_j^T\mathbf{f}(\mathbf{x}_i)}}{\sum_{\ell=1}^{K} e^{\mathbf{w}_\ell^T\mathbf{f}(\mathbf{x}_i)}}\right)\mathbf{f}(\mathbf{x}_i),$$

where we used the fact that $\sum_k t_{i,k} = 1$.

补充知识：RNN（输入2维，隐藏2维，输出2维） input sequence: $\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}\cdots$



$O_t = g(V \cdot S_t)$

$S_t = f(U \cdot X_t + W \cdot S_{t-1})$

S_t 依赖于 X_t 也依赖于 S_{t-1}

(1,4)(4,512) → (1,512) → (1,512) (2,4).