Abhishek Jhunjhunwala Follow
Feb 26, 2019 · 4 min read · Listen

Is Logistic Regression a good multi-class classifier ?

Overview

All of us might have heard about or used the famous Logistic Regression algorithm at some point especially the data enthusiasts. And I am sure that a lot of you might have used it for multi classification without pondering over the fact whether it was meant to do that in the first place. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. Well, there are a few jargon here but the important phrase is 'dependent binary variable' which means that LR was by definition meant to classify or predict binary classes. But we know that it works for multi class problems as well, so lets discuss how we can do that and how good is logistic regression for multi class compared to some other machine learning algorithms that do the same task.

Multinomial Logistic Regression

Multinomial logistic regression is a form of logistic regression used to predict a target variable have more than 2 classes. It is a modification of logistic regression using the softmax function instead of the sigmoid function the cross entropy loss function. The softmax function squashes all values to the range [0,1] and the sum of the elements is 1.

$$\text{softmax}(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

$$H(p,q) = -\sum_x p(x) \log q(x).$$

This function has a range of [0,inf] and is equal to 0 when p=q and infinity when p is very small compared to q or vice versa. For an example x, the class **scores** are given by vector z=Wx+b, where W is a C×M matrix and b is a length C vector of biases. We define the label y as a one-hot vector equal to 1 for the correct class c and 0 everywhere else. The loss for a training example x with predicted class distribution y and correct class c will be :

$$\hat{y} = \text{softmax}(z)$$

$$\begin{aligned} \text{loss} &= H(y, \hat{y}) \\ &= -\sum_i y_i \log \hat{y}_i \\ &= -\log \hat{y}_c \end{aligned}$$

As in the binary case, the loss value is exactly the negative log probability of a single example x having true class label c. Thus, minimizing the sum of the loss over our training examples is equivalent to maximizing the log likelihood. We can learn the model parameters W and b by performing gradient descent on the loss function with respect to these parameters.

Now, there are two common methods to perform multi-class classification using the binary classification logistic regression algorithm: one-vs-all and one-vs-one. In one-vs-all, we train C separate binary classifier for each class and run all those classifiers on any new example x we want to predict and take the class with the maximum score. In one-vs-one, we train C choose 2 classifiers = C(C-1)/2 one for each possible pair of class and choose the class with maximum votes while predicting for a new example

input values from all classes between 0 and 1 and returns the probabilities of each class. But softmax is not a linear function and non uniform which causes problems when you are trying to find the most optimum set of weights for your classifier. Lets take a case with 3 classes corresponding to y1, y2 and y3 and two examples for which the y values are [1,3,2] and [1,4,2]. The output of the sigmoid function for the two examples will be [0.09, 0.67, 0.24] and [0.04, 0.84, 0.11]. You can see the problem here, the difference between the y2 values 3 and 4 causes the sigmoid function to assign a probability of 67% and 84% while the remaining classes are constant. The cross entropy loss function considering that y3 was the correct class will be -log(0.24) and -log(0.11) that is 0.62 and 0.96 which is a considerable difference for not such a significant change in the input vectors.

Conclusion

So, the issue with softmax function is that it blows small differences out of proportion which makes our classifier biased towards a particular class which is not desired. Now, same be said of the sigmoid function but in case of binary classification we will ultimately reach the optimum point. So, the one-vs-one or one-vs-all is better approach towards multi-class classification using logistic regression. In the multiclass case, the sklearn package in python uses the one-vs-rest (OvR) scheme if the ‘multi_class’ option is set to ‘ovr’ and uses the cross-entropy loss if the ‘multi_class’ option is set to ‘multinomial’. (Currently, the ‘multinomial’ option is supported only by the ‘lbfgs’, ‘sag’ and ‘newton-cg’ solvers.)