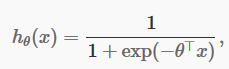
Softmax Regression

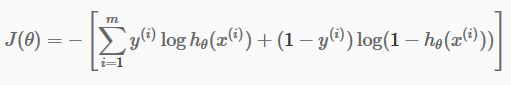
**Introduction**

Softmax regression (or multinomial logistic regression) is a generalization of logistic regression to the case where we want to handle multiple classes. In logistic regression we assumed that the labels were binary: . We used such a classifier to distinguish between two kinds of hand-written digits. Softmax regression allows us to handle where K is the number of classes.

Recall that in logistic regression, we had a training set  of m labeled examples, where the input features are . With logistic regression, we were in the binary classification setting, so the labels were .Our hypothesis took the form:

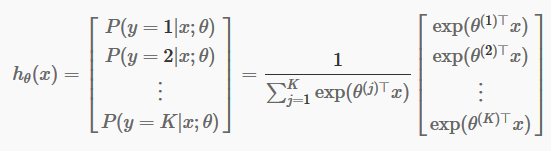


and the model parameters were trained to minimize the cost function



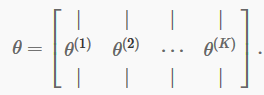
In the softmax regression setting, we are interested in multi-class classification (as opposed to only binary classification), and so the label y can take on K different values, rather than only two. Thus, in our training set ,we now have that .(Note that our convention will be to index the classes starting from 1, rather than from 0.) For example , in the MNIST digist recognition task, we would have K = 10 different classes.

Given a test input x, we want our hypothesis to estimate the probability that  for each value of k = 1,…,K. I.e. we want to estimate the probability of the class label taking on each of the K different possible values. Thus, our hypothesis will output a K – dimensional vector (whose elements sum to 1) giving us our K estimated probabilities. Concretely, our hypothesis takes the form:

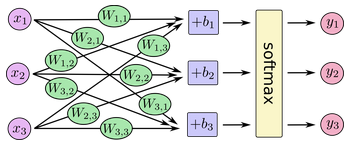


Here  are the parameters of our model. Notice that the term  normalizes the distribution, so that is sums to one.

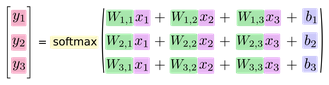
For convenience , we will also write  to denote all the parameters of our model. When you implement softmax regression, it is usually convenient to represent  as n-by-K matrix obtained by concatenating  into columns, so that



You can picture our softmax regression as looking something like the following, although with a lot more xs. For each output, we compute a weighted sum of the xs, add a bias, and the apply softmax.



If we write that out as equations, we get:



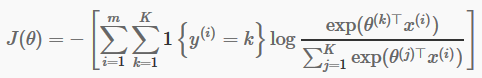
We can ‘vectorize’ this procedure, turning it into a matrix multiplication and vector addition. This is helpful computational efficiency.

More compactly, we can just write:

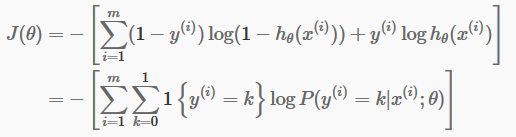


**Cost Function**

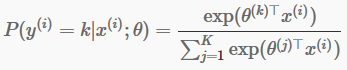
The cost function that used for softmax regression will be:



Notice that this generalizes the logistic regression cost function, which would also have been written:



The softmax cost function is similar, except that we now sum over the K different possible values of the class label. Note also that in softmax regression, we have that



We cannot solve for the minimum of  analytically, and thus as usual we’ll resort to an iterative optimization algorithm. Taking derivatives, one can show that the gradient is:

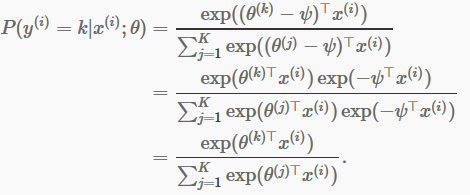


Recall the meaning of the notation. In particular,  is itself a vector, so that its j-th element is the partial derivative of with respect to the j-th element of .

Armed with this formula for the derivative , one can then plug it into a standard optimization package and have it minimize .

**Properties of softmax regression parameterization**

Softmax regression has an **unusual property** that it has a “redundant” set of parameters. To explain what this means, suppose we take each of our parameter vectors , and subtract some fixed vector  from it, so that every is now replaced with (for every j = 1,…,k).Our hypothesis now estimates the class label probabilities as

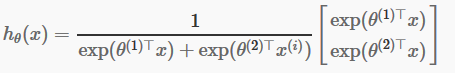


In other words, subtracting  from every  does not affect our hypothesis’ predictions at all! This shows that softmax regression’s parameters are “redundant.” More formally, we say that our softmax model is “overparameterized,” meaning that for any hypothesis we might fit to the data, there are multiple parameter settings that give rise to exactly the same hypothesis function mapping from inputs x to the predictions.

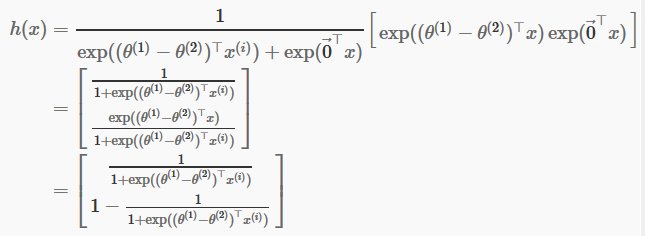
Further , if the cost function is minimized by some setting of the parameters , then it is also minimized by  for any value of . Thus, the minimizer of  is not unique.(Interestingly,  is still convex, and thus gradient descent will not run into local optima problems. But the Hessian is singular / non-invertible, which causes a straightforward implementation of Newton’s method to run into numerical problems.)

**Relationship to Logistic Regression**

In the special case where K = 2, one can show that softmax regression reduces to logistic regression. This shows that softmax regression is a generalization of logistic regression. Concretely, when K = 2, the softmax regression hypothesis outputs



Taking advantage of the fact that this hypothesis is overparameterized and setting , we can subtract  from each of the two parameters, giving us



Thus replacing with a single parameter vector , we find that softmax regression predicts the probability of one of the classes as ,and that of the other class as ,same as logistic regression.

Reference:

1. <http://ufldl.stanford.edu/tutorial/supervised/SoftmaxRegression/>
2. <https://www.tensorflow.org/versions/r0.11/tutorials/mnist/beginners/index.html>