

# Feedforward Neural Networks in Depth, Part 3: Cost Functions

Dec 22, 2021

This post is the last of a three-part series in which we set out to derive the mathematics behind feedforward neural networks. In short, we covered forward and backward propagations in [the first post](#), and we worked on activation functions in [the second post](#). Moreover, we have not yet addressed cost functions and the backpropagation seed  $\partial J / \partial \mathbf{A}^{[L]} = \partial J / \partial \hat{\mathbf{Y}}$ . It is time we do that.

## Binary Classification

In binary classification, the cost function is given by

$$\begin{aligned} J &= f(\hat{\mathbf{Y}}, \mathbf{Y}) = f(\mathbf{A}^{[L]}, \mathbf{Y}) \\ &= -\frac{1}{m} \sum_i (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)) \\ &= -\frac{1}{m} \sum_i (y_i \log(a_i^{[L]}) + (1 - y_i) \log(1 - a_i^{[L]})), \end{aligned}$$

which we can write as

$$J = -\frac{1}{m} \underbrace{\sum_{\text{axis}=1} (\mathbf{Y} \odot \log(\mathbf{A}^{[L]}) + (1 - \mathbf{Y}) \odot \log(1 - \mathbf{A}^{[L]}))}_{\text{scalar}}. \quad (1)$$

Next, we construct a computation graph:

$$\begin{aligned} u_{0,i} &= a_i^{[L]}, \\ u_{1,i} &= 1 - u_{0,i}, \\ u_{2,i} &= \log(u_{0,i}), \\ u_{3,i} &= \log(u_{1,i}), \\ u_{4,i} &= y_i u_{2,i} + (1 - y_i) u_{3,i}, \\ u_5 &= -\frac{1}{m} \sum_i u_{4,i} = J. \end{aligned}$$

Derivative computations are now as simple as they get:

$$\begin{aligned}
\frac{\partial J}{\partial u_5} &= 1, \\
\frac{\partial J}{\partial u_{4,i}} &= \frac{\partial J}{\partial u_5} \frac{\partial u_5}{\partial u_{4,i}} = -\frac{1}{m}, \\
\frac{\partial J}{\partial u_{3,i}} &= \frac{\partial J}{\partial u_{4,i}} \frac{\partial u_{4,i}}{\partial u_{3,i}} = -\frac{1}{m}(1 - y_i), \\
\frac{\partial J}{\partial u_{2,i}} &= \frac{\partial J}{\partial u_{4,i}} \frac{\partial u_{4,i}}{\partial u_{2,i}} = -\frac{1}{m}y_i, \\
\frac{\partial J}{\partial u_{1,i}} &= \frac{\partial J}{\partial u_{3,i}} \frac{\partial u_{3,i}}{\partial u_{1,i}} = -\frac{1}{m}(1 - y_i) \frac{1}{u_{1,i}} = -\frac{1}{m} \frac{1 - y_i}{1 - a_i^{[L]}}, \\
\frac{\partial J}{\partial u_{0,i}} &= \frac{\partial J}{\partial u_{1,i}} \frac{\partial u_{1,i}}{\partial u_{0,i}} + \frac{\partial J}{\partial u_{2,i}} \frac{\partial u_{2,i}}{\partial u_{0,i}} \\
&= \frac{1}{m}(1 - y_i) \frac{1}{u_{1,i}} - \frac{1}{m}y_i \frac{1}{u_{0,i}} \\
&= \frac{1}{m} \left( \frac{1 - y_i}{1 - a_i^{[L]}} - \frac{y_i}{a_i^{[L]}} \right).
\end{aligned}$$

Thus,

$$\frac{\partial J}{\partial a_i^{[L]}} = \frac{1}{m} \left( \frac{1 - y_i}{1 - a_i^{[L]}} - \frac{y_i}{a_i^{[L]}} \right),$$

which implies that

$$\frac{\partial J}{\partial \mathbf{A}^{[L]}} = \frac{1}{m} \left( \frac{1}{1 - \mathbf{A}^{[L]}} \odot (1 - \mathbf{Y}) - \frac{1}{\mathbf{A}^{[L]}} \odot \mathbf{Y} \right). \quad (2)$$

In addition, since the sigmoid activation function is used in the output layer, we get

$$\begin{aligned}
\frac{\partial J}{\partial z_i^{[L]}} &= \frac{\partial J}{\partial a_i^{[L]}} a_i^{[L]} (1 - a_i^{[L]}) \\
&= \frac{1}{m} \left( \frac{1 - y_i}{1 - a_i^{[L]}} - \frac{y_i}{a_i^{[L]}} \right) a_i^{[L]} (1 - a_i^{[L]}) \\
&= \frac{1}{m} ((1 - y_i) a_i^{[L]} - y_i (1 - a_i^{[L]})) \\
&= \frac{1}{m} (a_i^{[L]} - y_i).
\end{aligned}$$

In other words,

$$\frac{\partial J}{\partial \mathbf{Z}^{[L]}} = \frac{1}{m} (\mathbf{A}^{[L]} - \mathbf{Y}). \quad (3)$$

Note that both  $\partial J / \partial \mathbf{Z}^{[L]} \in \mathbb{R}^{1 \times m}$  and  $\partial J / \partial \mathbf{A}^{[L]} \in \mathbb{R}^{1 \times m}$ , because  $\mathbf{n}^{[L]} = \mathbf{1}$  in this case.

# Multiclass Classification

In multiclass classification, the cost function is instead given by

$$\begin{aligned} J &= f(\hat{\mathbf{Y}}, \mathbf{Y}) = f(\mathbf{A}^{[L]}, \mathbf{Y}) \\ &= -\frac{1}{m} \sum_i \sum_j y_{j,i} \log(\hat{y}_{j,i}) \\ &= -\frac{1}{m} \sum_i \sum_j y_{j,i} \log(a_{j,i}^{[L]}), \end{aligned}$$

where  $j = 1, \dots, n^{[L]}$ .

We can vectorize the cost expression:

$$J = -\frac{1}{m} \underbrace{\sum_{\substack{\text{axis}=0 \\ \text{axis}=1}} \mathbf{Y} \odot \log(\mathbf{A}^{[L]})}_{\text{scalar}}. \quad (4)$$

Next, let us introduce intermediate variables:

$$\begin{aligned} u_{0,j,i} &= a_{j,i}^{[L]}, \\ u_{1,j,i} &= \log(u_{0,j,i}), \\ u_{2,j,i} &= y_{j,i} u_{1,j,i}, \\ u_{3,i} &= \sum_j u_{2,j,i}, \\ u_4 &= -\frac{1}{m} \sum_i u_{3,i} = J. \end{aligned}$$

With the computation graph in place, we can perform backward propagation:

$$\begin{aligned} \frac{\partial J}{\partial u_4} &= 1, \\ \frac{\partial J}{\partial u_{3,i}} &= \frac{\partial J}{\partial u_4} \frac{\partial u_4}{\partial u_{3,i}} = -\frac{1}{m}, \\ \frac{\partial J}{\partial u_{2,j,i}} &= \frac{\partial J}{\partial u_{3,i}} \frac{\partial u_{3,i}}{\partial u_{2,j,i}} = -\frac{1}{m}, \\ \frac{\partial J}{\partial u_{1,j,i}} &= \frac{\partial J}{\partial u_{2,j,i}} \frac{\partial u_{2,j,i}}{\partial u_{1,j,i}} = -\frac{1}{m} y_{j,i}, \\ \frac{\partial J}{\partial u_{0,j,i}} &= \frac{\partial J}{\partial u_{1,j,i}} \frac{\partial u_{1,j,i}}{\partial u_{0,j,i}} = -\frac{1}{m} y_{j,i} \frac{1}{u_{0,j,i}} = -\frac{1}{m} \frac{y_{j,i}}{a_{j,i}^{[L]}}. \end{aligned}$$

Hence,

$$\frac{\partial J}{\partial a_{j,i}^{[L]}} = -\frac{1}{m} \frac{y_{j,i}}{a_{j,i}^{[L]}}.$$

Vectorization is trivial:

$$\frac{\partial J}{\partial \mathbf{A}^{[L]}} = -\frac{1}{m} \frac{1}{\mathbf{A}^{[L]}} \odot \mathbf{Y}. \quad (5)$$

Furthermore, since the output layer uses the softmax activation function, we get

$$\begin{aligned} \frac{\partial J}{\partial z_{j,i}^{[L]}} &= a_{j,i}^{[L]} \left( \frac{\partial J}{\partial a_{j,i}^{[L]}} - \sum_p \frac{\partial J}{\partial a_{p,i}^{[L]}} a_{p,i}^{[L]} \right) \\ &= a_{j,i}^{[L]} \left( -\frac{1}{m} \frac{y_{j,i}}{a_{j,i}^{[L]}} + \sum_p \frac{1}{m} \frac{y_{p,i}}{a_{p,i}^{[L]}} a_{p,i}^{[L]} \right) \\ &= \frac{1}{m} \left( -y_{j,i} + a_{j,i}^{[L]} \underbrace{\sum_p y_{p,i}}_{\sum \text{probabilities}=1} \right) \\ &= \frac{1}{m} (a_{j,i}^{[L]} - y_{j,i}). \end{aligned}$$

Note that  $p = 1, \dots, n^{[L]}$ .

To conclude,

$$\frac{\partial J}{\partial \mathbf{Z}^{[L]}} = \frac{1}{m} (\mathbf{A}^{[L]} - \mathbf{Y}). \quad (6)$$

## Multi-Label Classification

We can view multi-label classification as  $j$  binary classification problems:

$$\begin{aligned} J &= f(\hat{\mathbf{Y}}, \mathbf{Y}) = f(\mathbf{A}^{[L]}, \mathbf{Y}) \\ &= \sum_j \left( -\frac{1}{m} \sum_i (y_{j,i} \log(\hat{y}_{j,i}) + (1 - y_{j,i}) \log(1 - \hat{y}_{j,i})) \right) \\ &= \sum_j \left( -\frac{1}{m} \sum_i (y_{j,i} \log(a_{j,i}^{[L]}) + (1 - y_{j,i}) \log(1 - a_{j,i}^{[L]})) \right), \end{aligned}$$

where once again  $j = 1, \dots, n^{[L]}$ .

Vectorization gives

$$J = -\frac{1}{m} \underbrace{\sum_{\substack{\text{axis}=1 \\ \text{axis}=0}} (\mathbf{Y} \odot \log(\mathbf{A}^{[L]}) + (1 - \mathbf{Y}) \odot \log(1 - \mathbf{A}^{[L]}))}_{\text{scalar}}. \quad (7)$$

It is no coincidence that the following computation graph resembles the one we constructed for binary classification:

$$\begin{aligned}
 u_{0,j,i} &= a_{j,i}^{[L]}, \\
 u_{1,j,i} &= 1 - u_{0,j,i}, \\
 u_{2,j,i} &= \log(u_{0,j,i}), \\
 u_{3,j,i} &= \log(u_{1,j,i}), \\
 u_{4,j,i} &= y_{j,i} u_{2,j,i} + (1 - y_{j,i}) u_{3,j,i}, \\
 u_{5,j} &= -\frac{1}{m} \sum_i u_{4,j,i}, \\
 u_6 &= \sum_j u_{5,j} = J.
 \end{aligned}$$

Next, we compute the partial derivatives:

$$\begin{aligned}
 \frac{\partial J}{\partial u_6} &= 1, \\
 \frac{\partial J}{\partial u_{5,j}} &= \frac{\partial J}{\partial u_6} \frac{\partial u_6}{\partial u_{5,j}} = 1, \\
 \frac{\partial J}{\partial u_{4,j,i}} &= \frac{\partial J}{\partial u_{5,j}} \frac{\partial u_{5,j}}{\partial u_{4,j,i}} = -\frac{1}{m}, \\
 \frac{\partial J}{\partial u_{3,j,i}} &= \frac{\partial J}{\partial u_{4,j,i}} \frac{\partial u_{4,j,i}}{\partial u_{3,j,i}} = -\frac{1}{m} (1 - y_{j,i}), \\
 \frac{\partial J}{\partial u_{2,j,i}} &= \frac{\partial J}{\partial u_{4,j,i}} \frac{\partial u_{4,j,i}}{\partial u_{2,j,i}} = -\frac{1}{m} y_{j,i}, \\
 \frac{\partial J}{\partial u_{1,j,i}} &= \frac{\partial J}{\partial u_{3,j,i}} \frac{\partial u_{3,j,i}}{\partial u_{1,j,i}} = -\frac{1}{m} (1 - y_{j,i}) \frac{1}{u_{1,j,i}} = -\frac{1}{m} \frac{1 - y_{j,i}}{1 - a_{j,i}^{[L]}}, \\
 \frac{\partial J}{\partial u_{0,j,i}} &= \frac{\partial J}{\partial u_{1,j,i}} \frac{\partial u_{1,j,i}}{\partial u_{0,j,i}} + \frac{\partial J}{\partial u_{2,j,i}} \frac{\partial u_{2,j,i}}{\partial u_{0,j,i}} \\
 &= \frac{1}{m} (1 - y_{j,i}) \frac{1}{u_{1,j,i}} - \frac{1}{m} y_{j,i} \frac{1}{u_{0,j,i}} \\
 &= \frac{1}{m} \left( \frac{1 - y_{j,i}}{1 - a_{j,i}^{[L]}} - \frac{y_{j,i}}{a_{j,i}^{[L]}} \right).
 \end{aligned}$$

Simply put, we have

$$\frac{\partial J}{\partial a_{j,i}^{[L]}} = \frac{1}{m} \left( \frac{1 - y_{j,i}}{1 - a_{j,i}^{[L]}} - \frac{y_{j,i}}{a_{j,i}^{[L]}} \right),$$

and

$$\frac{\partial J}{\partial \mathbf{A}^{[L]}} = \frac{1}{m} \left( \frac{1}{1 - \mathbf{A}^{[L]}} \odot (1 - \mathbf{Y}) - \frac{1}{\mathbf{A}^{[L]}} \odot \mathbf{Y} \right). \quad (8)$$

Bearing in mind that we view multi-label classification as  $j$  binary classification problems, we also know that the output layer uses the sigmoid activation function. As a result,

$$\begin{aligned}
 \frac{\partial J}{\partial z_{j,i}^{[L]}} &= \frac{\partial J}{\partial a_{j,i}^{[L]}} a_{j,i}^{[L]} (1 - a_{j,i}^{[L]}) \\
 &= \frac{1}{m} \left( \frac{1 - y_{j,i}}{1 - a_{j,i}^{[L]}} - \frac{y_{j,i}}{a_{j,i}^{[L]}} \right) a_{j,i}^{[L]} (1 - a_{j,i}^{[L]}) \\
 &= \frac{1}{m} ((1 - y_{j,i}) a_{j,i}^{[L]} - y_{j,i} (1 - a_{j,i}^{[L]})) \\
 &= \frac{1}{m} (a_{j,i}^{[L]} - y_{j,i}),
 \end{aligned}$$

which we can vectorize as

$$\frac{\partial J}{\partial \mathbf{Z}^{[L]}} = \frac{1}{m} (\mathbf{A}^{[L]} - \mathbf{Y}). \tag{9}$$



Subscribe

Jonas Lalin

Yet another blog about deep learning.

