



Cost Function and Backpropagation

- ✓ **Video:** Cost Function
6 min
- ✓ **Reading:** Cost Function
4 min
- ✓ **Video:** Backpropagation Algorithm
11 min
- ✓ **Reading:** Backpropagation Algorithm
10 min
- ✓ **Video:** Backpropagation Intuition
12 min
- ✓ **Reading:** Backpropagation Intuition
4 min

Backpropagation in Practice

Application of Neural Networks

Review



Backpropagation Intuition

Note: [4:39, the last term for the calculation for z_1^3 (tl a_2^2 instead of a_1^2 . 6:08 - the equation for cost(i) is incorrect for the log() function, and the second term should be $\delta^{(4)} = y - a^{(4)}$ is incorrect and should be $\delta^{(4)} = a^{(4)}$

Recall that the cost function for a neural network is:

$$J(\Theta) = -\frac{1}{m} \sum_{t=1}^m \sum_{k=1}^K y_k^{(t)} \log(h_{\Theta}(x^{(t)}))_k + (1 - y_k^{(t)}) \log(1 - h_{\Theta}(x^{(t)}))_k$$

If we consider simple non-multiclass classification ($k = 1$), the cost function can be computed with:

$$cost(t) = y^{(t)} \log(h_{\Theta}(x^{(t)})) + (1 - y^{(t)}) \log(1 - h_{\Theta}(x^{(t)}))$$

Intuitively, $\delta_j^{(l)}$ is the "error" for $a_j^{(l)}$ (unit j in layer l). It is the derivative of the cost function:

$$\delta_j^{(l)} = \frac{\partial}{\partial z_j^{(l)}} cost(t)$$

Recall that our derivative is the slope of a line tangent to the cost function. The more incorrect we are, the steeper the slope. Let us consider the error for unit j in layer l . We could calculate some $\delta_j^{(l)}$:

Forward Propagation

