

Backpropagation Intuition

Recall that the cost function for a neural network is:

$$J(\Theta) = -\frac{1}{m} \sum_{t=1}^m \sum_{k=1}^K \left[y_k^{(t)} \log(h_{\Theta}(x^{(t)}))_k + (1 - y_k^{(t)}) \log(1 - h_{\Theta}(x^{(t)}))_k \right] + \frac{\lambda}{2m} \sum_{l=1}^{L-1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} (\Theta_{j,i}^{(l)})^2$$

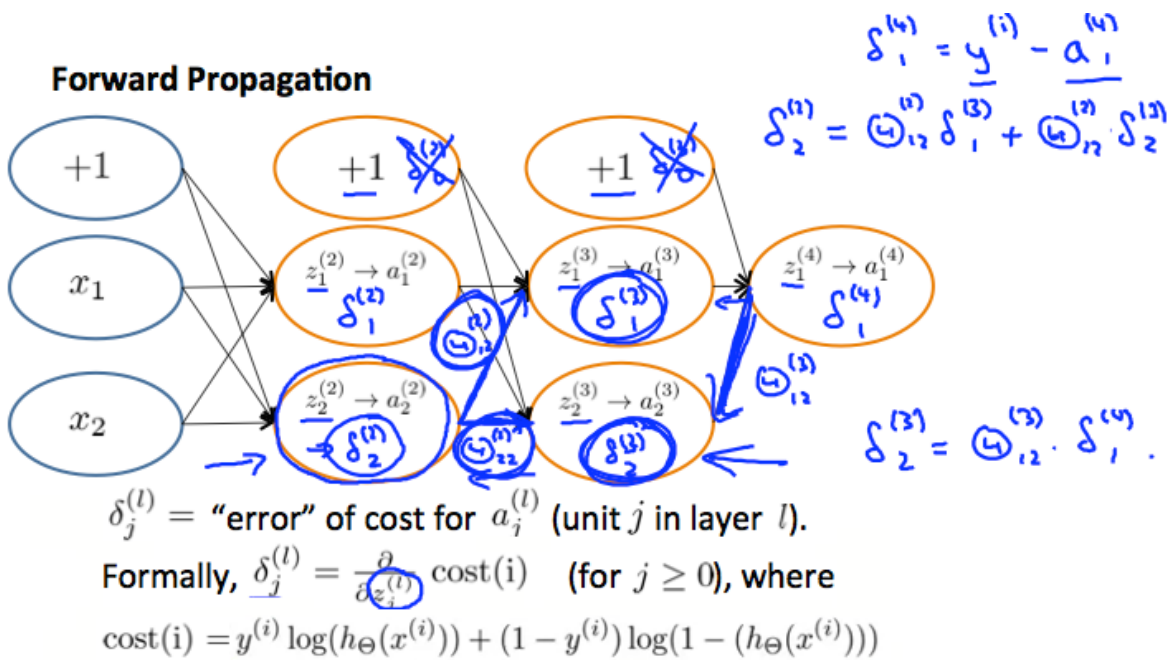
If we consider simple non-multiclass classification ($k = 1$) and disregard regularization, the cost is computed with:

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

Intuitively, $\delta_j^{(l)}$ is the "error" for $a_j^{(l)}$ (unit j in layer l). More formally, the delta values are actually the derivative of the cost function:

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

Recall that our derivative is the slope of a line tangent to the cost function, so the steeper the slope the more incorrect we are. Let us consider the following neural network below and see how we could calculate some $\delta_j^{(l)}$:



In the image above, to calculate $\delta_2^{(2)}$, we multiply the weights $\Theta_{12}^{(2)}$ and $\Theta_{22}^{(2)}$ by their respective δ values found to the right of each edge. So we get $\delta_2^{(2)} = \Theta_{12}^{(2)} * \delta_1^{(3)} + \Theta_{22}^{(2)} * \delta_2^{(3)}$. To calculate every single possible $\delta_j^{(l)}$, we could start from the right of our diagram. We can think of our edges as our Θ_{ij} . Going from right to left, to calculate the value

of $\delta_j^{(l)}$, you can just take the over all sum of each weight times the δ it is coming from. Hence, another example would be $\delta_2^{(3)} = \Theta_{12}^{(3)} * \delta_1^{(4)}$.

✓ Complete

