**Backpropagation Intuition**

**Note:** [4:39, the last term for the calculation for *z*13 (three-color handwritten formula) should be *a*22 instead of *a*12. 6:08 - the equation for cost(*i*) is incorrect. The first term is missing parentheses for the log() function, and the second term should be (1 − *y*(*i*))log(1 − h*θ*(*x*(*i*))). 8:50 - *δ*(4) = *y* − *a*(4) is incorrect and should be *δ*(4) = *a*(4) − *y*.]

Recall that the cost function for a neural network is:



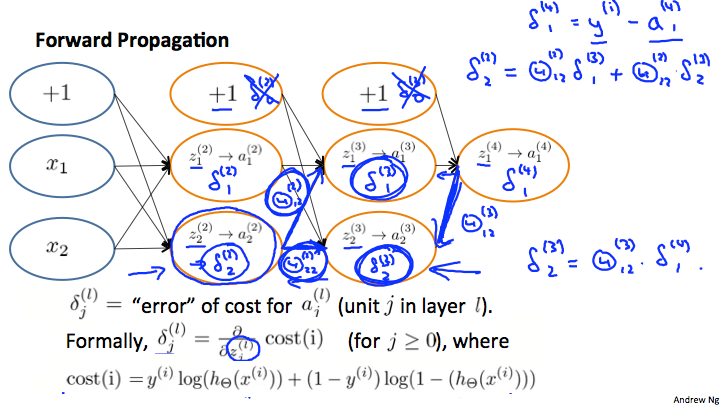
If we consider simple non-multiclass classification (k = 1) and disregard regularization, the cost is computed with:

|  |
| --- |
| cost(t)=y(t) log⁡(hΘ(x(t)))+(1−y(t)) log⁡(1−hΘ(x(t))) |

Intuitively, δj(l) is the "error" for aj(l) (unit j in layer l). More formally, the delta values are actually the derivative of the cost function:

|  |
| --- |
| δj(l)=∂∂zj(l)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 δj(l):



In the image above, to calculate δ2(2), we multiply the weights Θ12(2) and Θ22(2) by their respective δ values found to the right of each edge. So we get δ2(2)= Θ12(2)\*δ1(3)+Θ22(2)\*δ2(3). To calculate every single possible δj(l), we could start from the right of our diagram. We can think of our edges as our Θij. Going from right to left, to calculate the value of δj(l), you can just take the over all sum of each weight times the δ it is coming from. Hence, another example would be δ2(3)=Θ12(3)\*δ1(4).