Cost Function

Let's first define a few variables that we will need to use:

- L = total number of layers in the network

- s_l = number of units (not counting bias unit) in layer l

- K = number of output units/classes

Recall that in neural networks, we may have many output nodes. We denote $h_\Theta (x)_k$ as being a hypothesis that results in the kth output. Our cost function for neural networks is going to be a generalization of the one we used for logistic regression. Recall that the cost function for regularized logistic regression was:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \ \log(h_\theta (x^{(i)})) + (1 - y^{(i)}) \ \log(1 - h_\theta (x^{(i)})) \right] + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2$$

For neural networks, it is going to be slightly more complicated:

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

We have added a few nested summations to account for our multiple output nodes. In the first part of the equation, before the square brackets, we have an additional nested summation that loops through the number of output nodes.

In the regularization part, after the square brackets, we must account for multiple theta matrices. The number of columns in our current theta matrix is equal to the number of nodes in our current layer (including the bias unit). The number of rows in our current theta matrix is equal to the number of nodes in the next layer (excluding the bias unit). As before with logistic regression, we square every term.

Note:

- the double sum simply adds up the logistic regression costs calculated for each cell in the output layer

- the triple sum simply adds up the squares of all the individual Θs in the entire network.

- the i in the triple sum does **not** refer to training example i

✓ 完成