CS 189: Introduction to Machine Learning - Discussion 11

1. **Initialization of weights for backpropagation**

   Recall that backpropagation is simply a clever method to solve for the gradient of the loss function so we can use it in a numerical optimization method such as gradient descent. Backpropagation uses the chain rule to pass the gradient backwards through the network. Let $\mathcal{L}$ be the final loss. For layer l, let $x^{(l-1)}$ be the input to the layer, and $\delta^{(l)}$ be the gradient with respect to the input. Then we have:

$$\delta^{(l)} = \frac{\partial \mathcal{L}}{\partial x^{(l-1)}} = \left(\frac{\partial x^{(l)}}{\partial x^{(l-1)}} \right) \delta^{(l+1)}$$

   Assume a fully connected 1-hidden layer network with K output nodes and a nonlinearity g. Let $d^{(l)}$ be the number of nodes at layer l. We have:

$$x_j^{(l)} = g\left(\sum_{i=1}^{d^{(l-1)}} w_{ij}^{(l)} x_i^{(l-1)} \right)$$

   (a) Imagine that we initialize the values of our weights to be some constant w. After performing the forward pass, what is the relation between the members of the set $\{x_j^{(1)} : j = 1, \ldots, d^{(1)}\}$ and $\{x_i^{(0)} : i = 1, \ldots, d^{(0)}\}$?

   (b) After the backwards pass of backpropagation, what is the relation between the members of the set $\{\delta_j^{(2)} : j = 1, \ldots, d^{(1)}\}$ and $\{\delta_k^{(3)} : k = 1, \ldots, d^{(2)}\}$?

   (c) After the weights are updated and one iteration of gradient descent has been completed, what can we say about the weights?

   (d) To solve this problem, we randomly initialize our weights. This is called symmetry breaking. Why are we able to set our weights to 0 for logistic regression?

2. Modifying neural networks for fun and profit

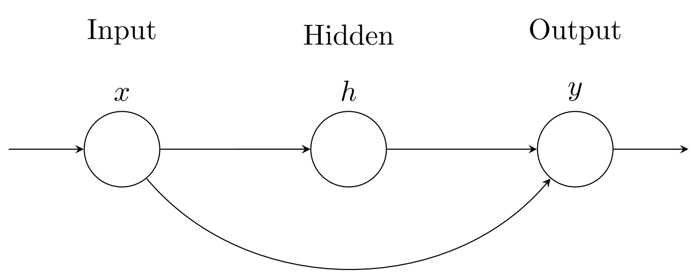
   (a) How could we modify a neural network to perform regression instead of classification?

   Consider a neural network with the addition that the input layer is also fully connected to the output layer. This type of neural network is also called "skip-layer".

   (b) How many weights would this model require? (Let d_0 be the dimensionality of the input vector, and $d_1 \dots d_L$ be the number of nodes in the L following layers. Don't worry about the bias term. Also, you may want to try drawing out the NN.)

   (c) What sort of problems may this sort of neural network introduce? How do we compensate for these problems?

   (d) Consider the simplest skip-layer neural network pictured below. The weights are $w = [w_{xh}, w_{hy}, w_{xy}]^T$.



   Given some non-linear function g, calculate $\nabla_w y$.