PadhAI: Backpropagation - the full version

One Fourth Labs

Computing derivatives w.r.t one weight in any layer

1. Our roadmap for this module
   a. To calculate the desired gradient, we need to compute
   b. Gradient w.r.t output units
   c. Gradient w.r.t hidden units
   d. Gradient w.r.t weights and biases
   e.

$$\frac{\partial L(\theta)}{\partial W_{111}} = \frac{\partial L(\theta)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a_3} \left[\frac{\partial a_3}{\partial h_2} \frac{\partial h_2}{\partial a_2} \quad \frac{\partial a_2}{\partial h_1} \frac{\partial h_1}{\partial a_1} \right] \frac{\partial a_3}{\partial W_{111}}$$

Talk to the weight directly	Talk to the output layer	Talk to the previous hidden layer	Talk to the previous hidden layer	Talk to the weights
		works for any number of output layers		

   f. For the rest of this exercise, our focus is on *Cross Entropy loss* and *Softmax* output.
2. Here, what the sections highlighted in green are what we have covered so far, i.e. the derivative with respect to the last hidden layer and all other subsequent hidden layers
3. The gradients were calculated to be
   a. $\nabla_{h_i} L(\theta) = (W_{i+1})^T \nabla_{a_{i+1}} L(\theta)$
   b. $\nabla_{a_i} L(\theta) = \nabla_{h_i} L(\theta) \odot [..., g'(a_{ik}), ...]$
4. Now, we will be computing the derivative of the loss function w.r.t weights and biases.
5. Recall that $a_k = b_k + W_k h_{k-1}$
6. $\frac{\partial a_{ki}}{\partial W_{kij}} = h_{k-1,j}$
   a. k = layer number
   b. i = current layer neuron number
   c. j = previous/input layer neuron number
7. Now $\frac{\partial L(\theta)}{\partial W_{kij}} = \frac{\partial L(\theta)}{\partial a_{ki}} h_{k-1,j}$
8. We can use this to update the Weight by Gradient Descent
9. $W_{kij} = W_{kij} - \eta \frac{\partial L(\theta)}{\partial W_{kij}}$
10. In the next step, we will look at updating all the weights in a layer simultaneously.