Implementing backpropagation

Now we've seen that the error term for the output layer is

$$\delta_k = (y_k - \hat{y}_k) f'(a_k)$$

and the error term for the hidden layer is

$$\delta_j = \sum [w_{jk} \delta_k] f'(h_j)$$

For now we'll only consider a simple network with one hidden layer and one output unit. Here's the general algorithm for updating the weights with backpropagation:

- Set the weight steps for each layer to zero
    - The input to hidden weights $\Delta w_{ij} = 0$
    - The hidden to output weights $\Delta W_j = 0$
- For each record in the training data:
    - Make a forward pass through the network, calculating the output $\hat{y}$
    - Calculate the error gradient in the output unit, $\delta^o = (y - \hat{y}) f'(z)$

      where $z = \sum_j W_j a_j$, the input to the output unit.
    - Propagate the errors to the hidden layer $\delta_j^h = \delta^o W_j f'(h_j)$
    - Update the weight steps:
        - $\Delta W_j = \Delta W_j + \delta^o a_j$
        - $\Delta w_{ij} = \Delta w_{ij} + \delta_j^h a_i$
- Update the weights, where η\eta is the learning rate and mm is the number of records:
    - $W_j = W_j + \eta \Delta W_j / m$
    - $W_{ij} = W_{ij} + \eta \Delta W_{ij} / m$
- Repeat for e epochs.

Backpropagation exercise

Now you're going to implement the backprop algorithm for a network trained on the graduate school admission data. You should have everything you need from the previous exercises to complete this one.

Your goals here:

- Implement the forward pass.

- Implement the backpropagation algorithm.
- Update the weights.

Code in code section