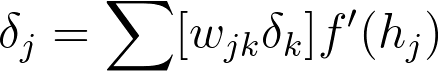
**Implementing backpropagation**

Now we've seen that the error in the output layer is

*δ*​*k*​​=(*y*​*k*​​−​*y*​^​​​*k*​​)*f*​′​​(*a*​*k*​​)

and the error in the hidden layer is



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 Δ*w*​*ij*​​=0
  + The hidden to output weights Δ*W*​*j*​​=0
* For each record in the training data:
  + Make a forward pass through the network, calculating the output ​*y*​^​​
  + Calculate the error gradient in the output unit, *δ*​*o*​​=(*y*−​*y*​^​​)*f*​′​​(*z*) where *z*=∑​*j*​​*W*​*j*​​*a*​*j*​​, the input to the output unit.
  + Propagate the errors to the hidden layer *δ*​*j*​*h*​​=*δ*​*o*​​*W*​*j*​​*f*​′​​(*h*​*j*​​)
  + Update the weight steps,:
    - Δ*W*​*j*​​=Δ*W*​*j*​​+*δ*​*o*​​*a*​*j*​​
    - Δ*w*​*ij*​​=Δ*w*​*ij*​​+*δ*​*j*​*h*​​*a*​*i*​​
* Update the weights, where *η* is the learning rate and *m* is the number of records:
  + *W*​*j*​​=*W*​*j*​​+*η*Δ*W*​*j*​​/*m*
  + *w*​*ij*​​=*w*​*ij*​​+*η*Δ*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.