Gradient Descent with Squared Errors

We want to find the weights for our neural networks. Let's start by thinking about the goal. The network needs to make predictions as close as possible to the real values. To measure this, we use a metric of how wrong the predictions are, the **error**. A common metric is the sum of the squared errors (SSE):

$$E = \frac{1}{2}\sum_{\mu}\sum_{j}\left[y_j^{\mu} - \hat{y}_j^{\mu}\right]^2$$

where $\hat{y}$ is the prediction and y is the true value, and you take the sum over all output units j and another sum over all data points μ. This might seem like a really complicated equation at first, but it's fairly simple once you understand the symbols and can say what's going on in words.

First, the inside sum over j. This variable j represents the output units of the network. So this inside sum is saying for each output unit, find the difference between the true value y and the predicted value from the network $\hat{y}$, then square the difference, then sum up all those squares.

Then the other sum over μ is a sum over all the data points. So, for each data point you calculate the inner sum of the squared differences for each output unit. Then you sum up those squared differences for each data point. That gives you the overall error for all the output predictions for all the data points.

The SSE is a good choice for a few reasons. The square ensures the error is always positive and larger errors are penalized more than smaller errors. Also, it makes the math nice, always a plus.

Remember that the output of a neural network, the prediction, depends on the weights

$$\hat{y}_j^{\mu} = f\left(\sum_i w_{ij} x_i^{\mu}\right)$$