# **Backpropagation (Light Math)**

## **Setting the context**

Can we use the same learning algorithm as before?

1. Here is the learning algorithm as discussed in the previous chapter, the no-math version
2. Consider the Neural Network with the following configuration
3. The algorithm
   1. **Initialise:** w111, w112, … w313, b1, b2, b3 randomly
   2. **Iterate over data**
      1. Compute ŷ
      2. Compute L(w,b) Cross-entropy loss function
      3. w111 = w111 - η𝚫w111
      4. w112 = w112 - η𝚫w112

…

* + 1. w313 = w111 - η𝚫w313
    2. bi = bi + η𝚫bi
    3. Pytorch/Tensorflow have functions to compute
  1. **Till satisfied**
     1. Number of epochs is reached ( ie 1000 passes/epochs)
     2. Continue till Loss < ε (some defined value)

1. In this section, we will be looking at the light-math version, where we will be computing the derivatives
2. Derivatives for all layers from 1 to L
3. Once we know the gradients, we can use them in the Gradient Descent algorithm to compute the weights of the network