Assignment 1
Backpropagation and Multilayer Perceptron

Cláudia Afonso nº 36273 e Rita Rodrigues nº 54859

February 27, 2023

Problem 1

a)

To determine the separation planes, it is first necessary to write the expression for the function signal appearing at the output of neuron 2.

The general expression for the function signal $y_j(n)$ appearing at the output of neuron j at iteration n is:

$$y_j(n) = \varphi(v_j(n))$$

where $v_j(n)$ represents the induced local field produced at the input of the activation function associated with neuron j. The induced local field, $v_j(n)$, is calculated by the following expression:

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n)x_i(n)$$

where m represents the total number of inputs applied to neuron j, $w_{ji}(n)$ represents the synaptic weights associated with each input and $x_i(n)$ represents the inputs applied to neuron j at iteration n.

The synaptic weight w_{j0} (corresponding to the fixed input $x_0 = +1$) equals the bias b_j applied to neuron j.

Therefore, the expression for the function signal appearing at the output of neuron 2 at the first iteration, $y_2(1)$, is:

$$y_2(1) = \varphi(w_{20}x_0 + w_{21}x_1 + w_{22}x_2 + w_{23} \cdot \varphi(w_{10}x_0 + w_{11}x_1 + w_{12}x_2))$$

Since $w_{10} = b_1 = -1.5$, $w_{20} = b_2 = -0.5$, $x_0 = +1$, $w_{11} = w_{12} = w_{21} = w_{22} = +1$, and $w_{23} = -2$, the previous expression is simplified to:

$$y_2(1) = \varphi(x_1 + x_2 - 0.5 - 2 \cdot \varphi(x_1 + x_2 - 1.5))$$

To finally determine the separation planes, the two possible outputs of 0 and 1 for the function signal $y_2(1)$ were calculated for two possible input situations, namely $(1, 0)$ and $(0, 1)$.

Let's begin with point $(0, 1)$.

$$x_1 = 0 \Rightarrow y_2(1) = \varphi(x_2 - 0.5 - 2 \cdot \varphi(x_2 - 1.5))$$

For $y_2(1) = 1$, then it follows that:

$$x_2 - 0.5 - 2 \cdot \varphi(x_2 - 1.5) > 0 \Leftrightarrow x_2 - 0.5 > 2 \cdot \varphi(x_2 - 1.5)$$

The latter inequality can only be valid when $x_2 \,\epsilon\,]0.5, 1.5[\,,]2.5, 3.5[\,,]4.5, 5.5[$, etc.

For $y_2(1) = 0$, then it follows that:

$$x_2 - 0.5 - 2 \cdot \varphi(x_2 - 1.5) \leqslant 0 \Leftrightarrow x_2 - 0.5 \leqslant 2 \cdot \varphi(x_2 - 1.5)$$

Since $\varphi(x_2 - 1.5)$ can only take the values of 0 and 1, then it follows that:

$$2 \cdot \varphi(x_2 - 1.5) = 0 \vee 2 \cdot \varphi(x_2 - 1.5) = 2$$

Thus, the inequality $x_2 - 0.5 \leqslant 2 \cdot \varphi(x_2 - 1.5)$ becomes:

$$x_2 - 0.5 \leqslant 0 \;\vee\; x_2 - 0.5 \leqslant 2$$

from which we obtain:

$$x_2 \leqslant 0.5 \vee x_2 \leqslant 2.5$$

If these results are now combined with the previous ones showing that $y_2(1) = 1$ can only be valid when $x_2 \,\epsilon\,]0.5, 1.5[\,,]2.5, 3.5[$, etc., then $y_2(1) = 0$ can only valid when $x_2 \,\epsilon\, [-0.5, 0.5]\,, [1.5, 2.5]$, etc.

These results will be the same for point $(1, 0)$ and thus $y_2(1) = 1$ can only be valid when $x_1 \,\epsilon\,]0.5, 1.5[\,,]2.5, 3.5[$, etc., while $y_2(1) = 0$ can only valid when $x_1 \,\epsilon\, [-0.5, 0.5]\,, [1.5, 2.5]$, etc.

By now drawing the input and output values, we can arrive at the equations of the separation planes.

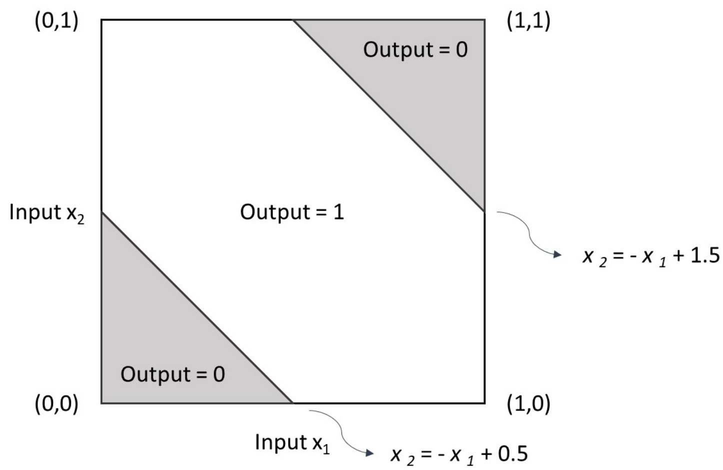

Figure 1: Graphical representation of the separation planes.

b)

As mentioned previously, the expression for the function signal appearing at the output of neuron 2 at the first iteration, $y_2(1)$, is:

$$y_2(1) = \varphi(x_1 + x_2 - 0.5 - 2 \cdot \varphi(x_1 + x_2 - 1.5))$$

By substituting one at a time each of the input values for 0 and 1 to obtain the input patters (0,0), (0,1), (1,0), and (1,1), we can obtain a truth table for this neural network.

If $x_1 = 0 \wedge x_2 = 0$, then:

$$\mathrm{y}_2(1) = \varphi(0 + 0 - 0.5 - 2 \cdot \varphi(0 + 0 - 1.5)) = \varphi(-0.5) = 0$$

If $x_1 = 0 \wedge x_2 = 1$, then:

$$\mathrm{y}_2(1) = \varphi(0+1-0.5-2\cdot\varphi(0+1-1.5)) = \varphi(0.5-2\cdot\varphi(-0.5)) = \varphi(0.5) = 1$$

If $x_1 = 1 \wedge x_2 = 0$, then:

$$\mathrm{y}_2(1) = \varphi(1+0-0.5-2\cdot\varphi(1+0-1.5)) = \varphi(0.5-2\cdot\varphi(-0.5)) = \varphi(0.5) = 1$$

If $x_1 = 1 \wedge x_2 = 1$, then:

$$\mathrm{y}_2(1) = \varphi(1+1-0.5-2\cdot\varphi(1+1-1.5)) = \varphi(1.5-2\cdot\varphi(0.5)) = \varphi(1.5-2) =$$

$\varphi(-0.5) = 0$

Thus, the truth table is as follows:

Table 1: Truth table

x_1	x_2	Output
0	0	0
0	1	1
1	0	1
1	1	0

This is an example of a XOR problem.

c)

In the application of the back-propagation algorithm, two different passes of computation are distinguished. The first pass is referred to as the forward pass, and the second is referred to as the backward pass.

In the forward pass, the synaptic weights remain unaltered throughout the network, and the function signals of the network are computed on a neuron-by-neuron basis.

To calculate 1 iteration of the backpropagation algorithm, the input will be initialized with $x_1 = 1$ and $x_2 = 0$. Therefore, according to the truth table above, the desired output is 1.

Let's start by calculating the induced local field of neuron 1 (hidden layer) during the forward pass. The expression of the induced local field for neuron 1 is:

$$v_1(1) = w_{10}x_0 + w_{11}x_1 + w_{12}x_2$$

Since $w_{10} = b_1 = -1.5$, $x_0 = +1$, $w_{11} = w_{12} = +1$, $x_1 = 1$, and $x_2 = 0$, the previous expression returns:

$$v_1(1) = -1.5 + 1 = -0.5$$

We are considering the activation function as a sigmoid, which as the following general expression:

$$y_j(n) = \varphi(v_j(n)) = \frac{1}{1 + e^{-av_j(n)}}$$

Since we are fixing parameter $a = 1$, then the previous expression becomes:

$$y_j(n) = \varphi(v_j(n)) = \frac{1}{1 + e^{-v_j(n)}}$$

Therefore, the output for neuron 1 (hidden layer) during the forward pass is:

$$y_1(1) = \varphi(v_1(1)) = \frac{1}{1 + e^{-v_1}} = \frac{1}{1 + e^{0.5}} = 0.378$$

Analogous calculations can be done for neuron 2 (output layer). The expression to calculate the local induced field for neuron 2 is:

$$v_2(n) = w_{20}x_0 + w_{21}x_1 + w_{22}x_2 + w_{23} \cdot \varphi(v_1)$$

Since $w_{20} = b_2 = -0.5$, $x_0 = +1$, $w_{21} = w_{22} = +1$, $w_{23} = -2$, $x_1 = 1$, $x_2 = 0$, and $\varphi(v_1) = 0.378$, the previous expression returns:

$$v_2(1) = -0.5 + 1 - 0.756 = -0.256$$

Again, since we are considering the activation function as a sigmoid with parameter $a = 1$, then the output for neuron 2 (output layer) during the forward pass is:

$$y_2(1) = \varphi(v_2(1)) = \frac{1}{1 + e^{-v_2}} = \frac{1}{1 + e^{0.256}} = 0.436$$

The expression to compute the error signal produced at the output of neuron j is:

$$e_j(n) = d_j(n) - o_j(n)$$

where $d_j(n)$ is the desired output of neuron j at iteration n and $o_j(n)$ represents the output of neuron j at iteration n.

Therefore, the error signal produced at the output of neuron 2 is:

$$e_2(1) = 1 - 0.436 = 0.564$$

Since the desired output is different from the actual output for neuron 2, it is necessary to apply a correction factor $\Delta w_{ji}(n)$ to the synaptic weights connecting neuron 1 and neuron 2, as well as those connecting the inputs x_1 and x_2 to neuron 2.

The backward pass starts at the output layer by passing the error signals leftward through the network, layer by layer, and recursively computing the local gradient δ for each neuron. This recursive process permits the synaptic weights of the network to be corrected.

The correction $\Delta w_{ji}(n)$ applied to the synaptic weight connecting input x_i to neuron k is defined by the delta rule, which has the following expression:

$$\Delta w_{ki}(n) = \eta \delta_k(n) x_i(n)$$

where δ_k is the local gradient for neuron k at iteration n.

For an output neuron, the expression for the local gradient is:

$$\delta_k(n) = e_j(n)\varphi'(v_j(n)) = a[d_j(n) - o_j(n)]o_j(n)[1 - o_j(n)]$$

Since $a = 1$, $d_2(1) = 1$ and $o_2(1) = 0.436$, then the previous expression returns:

$$\delta_2(1) = [1 - 0.436] * 0.436 * [1 - 0.436] = 0.139$$

Considering a learning rate parameter η equal to 0.2, it is now possible to calculate $\Delta w_{ki}(n)$ between neuron 2 and inputs x_1 and x_2, as well as the Δw_{20} corresponding to the bias associated with this neuron.

$$\Delta w_{20}(1) = \eta\delta_2(1)x_0(1) = 0.2 * 0.139 * 1 = 0.0278$$

$$\Delta w_{21}(1) = \eta\delta_2(1)x_1(1) = 0.2 * 0.139 * 1 = 0.0278$$

$$\Delta w_{22}(1) = \eta\delta_2(1)x_2(1) = 0.2 * 0.139 * 0 = 0$$

The correction $\Delta w_{kj}(n)$ applied to the synaptic weight connecting neuron j (hidden layer) to neuron k (output layer) is defined by the delta rule, which has the following expression:

$$\Delta w_{kj}(n) = \eta\delta_k(n)y_j(n)$$

where $y_j(n)$ represents the output of neuron j at iteration n.

Thus, the previous formula can be used to calculate $\Delta w_{kj}(n)$ between neuron 2 and neuron 1.

$$\Delta w_{23}(1) = \eta\delta_2(n)y_1(n) = 0.2 * 0.139 * 0.378 = 0.0105$$

We can also calculate the correction $\Delta w_{ji}(n)$ applied to the synaptic weight connecting input x_i to neuron j (hidden layer). To do so, it is necessary to first calculate the local gradient δ_j for neuron 1 (hidden layer), which is given by the following expression:

$$\delta_j(n) = \varphi'(v_j(n))\sum_k \delta_k(n)w_{kj}(n) = ay_j(n)[1 - y_j(n)]\sum_k \delta_k(n)w_{kj}(n)$$

Since $a = 1$, $y_j(n) = y_1(1) = 0.378$, $\delta_k(n) = \delta_2(1) = 0.139$, $w_{kj} = w_{23} = -2$, then the previous expression returns:

$$\delta_1(1) = y_1(1)[1 - y_1(1)] * \delta_2(1) * w_{23} = 0.378 * [1 - 0.378] * 0.139 = 0.033$$

After determining the local gradient for neuron 1 (hidden layer), it is possible to calculate the correction $\Delta w_{ji}(n)$ applied to the synaptic weight connecting this neuron and input x_i. To do so, we use the following general formula:

$$\Delta w_{ji}(n) = \eta\delta_j(n)x_i(n)$$

Thus, the previous formula can be used to calculate $\Delta w_{ji}(n)$ between neuron 1 and inputs x_1 and x_2, as well as the Δw_{10} corresponding to the bias associated with this neuron.

$$\Delta w_{10}(1) = \eta\delta_1(1)x_0(1) = 0.2 * 0.033 * 1 = 0.0066$$

$$\Delta w_{11}(1) = \eta\delta_1(1)x_1(1) = 0.2 * 0.033 * 1 = 0.0066$$

$$\Delta w_{12}(1) = \eta\delta_1(1)x_2(1) = 0.2 * 0.033 * 0 = 0$$

Now it is possible to adjust the synaptic weights of the network according to the following generalized formula:

$$w_{ji}(n+1) = w_{ji}(n) + \Delta w_{ji}(n)$$

Thus, to adjust the synaptic weights between neuron 2 and inputs x_1 and x_2, as well as the bias (w_{20}) associated with this neuron, we have the following calculations:

$$w_{20}(2) = w_{20}(1) + \Delta w_{20}(1) = -0.5 + 0.0278 = -0.4722$$

$$w_{21}(2) = w_{21}(1) + \Delta w_{21}(1) = 1 + 0.0278 = 1.0278$$

$$w_{22}(2) = w_{22}(1) + \Delta w_{22}(1) = 1 + 0 = 1$$

To adjust the synaptic weights between neuron 1 and inputs x_1 and x_2, as well as the bias (w_{10}) associated with this neuron, we have the following calculations:

$$w_{10}(2) = w_{10}(1) + \Delta w_{10}(1) = -1.5 + 0.0066 = -1.4934$$

$$w_{11}(2) = w_{11}(1) + \Delta w_{11}(1) = 1 + 0.0066 = 1.0066$$

$$w_{12}(2) = w_{12}(1) + \Delta w_{12}(1) = 1 + 0 = 1$$

Finally, to adjust the synaptic weights between neuron 1 and neuron 2, we have the following calculation:

$$w_{23}(2) = w_{23}(1) + \Delta w_{23}(1) = -2 + 0.0105 = -1.9895$$

Bibliography

Haykin S. Neural networks and learning machines, 3rd Edition. Pearson Education, Prentice Hall; 2009.