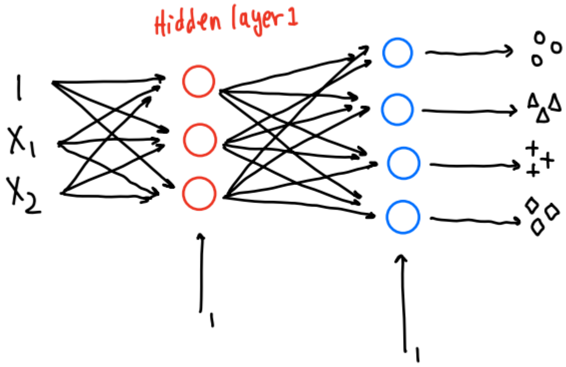
CAP6619 Deep Learning | Assignment 2

Benjamin Luo (6/1/2024)

Problem 1.

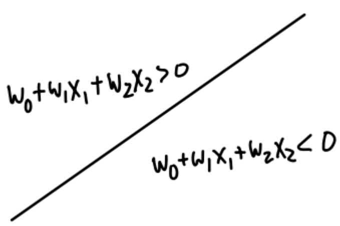


Solution. I chose two nodes (x_1, x_2) in the input layer, as we are dealing with a two-dimensional feature space. Since the classes can be roughly delineated using three lines, I chose to have three nodes in our one hidden layer. As for output nodes, there are 4 unique classes that we are trying to classify for, so each of the four output nodes are responsible for classifying one of the classes (circle, triangle, plus, and diamond). □

Problem 2.

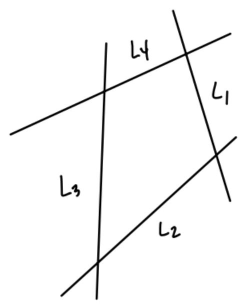
Solution.

a) Single layer neural network



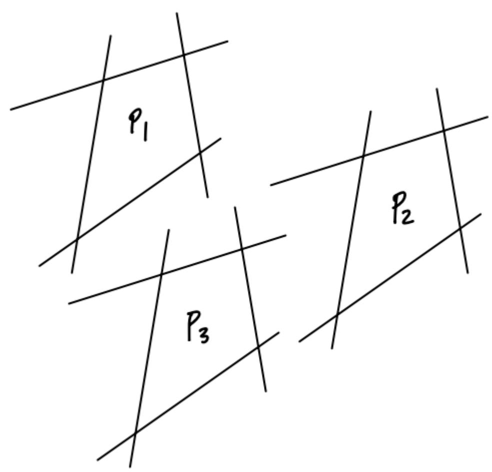
The single layer neural network classifies samples into two possible categories, based on if the output is above or below the line. If the output created by the inputs of the sample is greater than 0, then it is classified as class C_1, and otherwise, it classifies the sample as class C_2.

b) One hidden layer neural network



The decision region of a one hidden layer neural network is represented by the region that is within the lines that each hidden layer node represents. The network, over iterations will adjust weight values based on the lines represented by each hidden layer node.

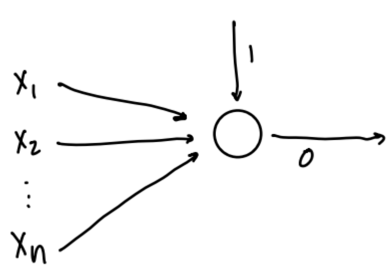
c) Two hidden layer neural network



The decision regions for this two hidden layer look like the three regions like shown in part b), but instead there are three regions instead of only 1, one for each new hidden node of the additional hidden layer.

□

Problem 3.



Solution.
The gradient descent learning objective function of the neuron is as follows:

$$f(w_0, w_1, w_2, ..., w_i) = \frac{1}{2}(d - [w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_i x_i])^2$$

$$\text{Gradient } = \frac{\partial f(w_0, w_1, w_2, \ldots w_i)}{\partial w_i}$$

$$= \frac{1}{2} \cdot 2 \cdot (d - [w_0 + w_1x_1 + w_2x_2 + \ldots + w_ix_i]) \cdot \frac{\partial(d - [w_0 + w_1x_1 + w_2x_2 + \ldots + w_ix_i])}{\partial w_{i+1}}$$

$$= (d - [w_0 + w_1x_1 + w_2x_2 + \ldots + w_ix_i]) \cdot (-1)$$

The gradient descent learning weight update rule for weight w_i can be derived as follows:

$$\Delta w_i = -\eta \frac{\partial f(W)}{\partial w_i}$$

$$\Delta w_i = -\eta[d(n) - w_0 - w_1x_1(n) - w_2x_2(n) - \ldots - w_ix_i(n)] \cdot x_i(n)$$

□

Problem 4.

Solution.

a) To calculate actual output of nodes e and f, we need to first find the outputs of the nodes a, b, c, and d, as those are the nodes whose outputs are the input values for nodes e and f. To do that, we need to first find v_a, v_b, v_c and v_d. This is done by calculating the values of all the input values and their corresponding weights together.

$$v_a = x_0w_{0a} + x_1w_{1a} + x_2w_{2a}$$
$$v_a = 1(0.1) + 0.5(0.1) + 0.5(0.1)$$
$$v_a = 0.1 + 0.05 + 0.05 = 0.2$$

$$v_b = x_0w_{0b} + x_1w_{1b} + x_2w_{2b}$$
$$v_b = 1(0.2) + 0.5(0.2) + 0.5(0.2)$$
$$v_b = 0.2 + 0.1 + 0.1 = 0.4$$

$$v_c = x_0w_{0c} + x_1w_{1c} + x_2w_{2c}$$
$$v_c = 1(0.3) + 0.5(0.3) + 0.5(0.3)$$
$$v_c = 0.3 + 0.15 + 0.15 = 0.6$$

$$v_d = x_0w_{0d} + x_1w_{1d} + x_2w_{2d}$$
$$v_d = 1(0.4) + 0.5(0.4) + 0.5(0.4)$$
$$v_d = 0.4 + 0.2 + 0.2 = 0.8$$

Then, using the sigmoid activation function with $a = 1.0$, we then find o_a, o_b, o_c, and o_d.

$$o_a = \frac{1}{1+e^{-v_a}} = \frac{1}{1+e^{-0.2}} = 0.55$$
$$o_c = \frac{1}{1+e^{-v_c}} = \frac{1}{1+e^{-0.6}} = 0.60$$

$$o_a = \frac{1}{1+e^{-v_b}} = \frac{1}{1+e^{-0.4}} = 0.65$$
$$o_a = \frac{1}{1+e^{-v_d}} = \frac{1}{1+e^{-0.8}} = 0.69$$

After finding o_a, o_b, o_c, o_d, we can calculate v_e and v_f in order to find actual outputs o_e and o_f.

$$v_e = x_0w_{0e} + o_aw_{ae} + o_bw_{be} + o_cw_{ce} + o_dw_{de}$$
$$v_e = 1(0.2) + 0.55(0.5) + 0.60(0.5) + 0.65(0.5) + 0.69(0.5)$$
$$v_e = 1.445$$
$$o_e = \frac{1}{1+e^{-v_e}} = \frac{1}{1+e^{-1.445}}$$

$$o_e = 0.81$$

$v_f = x_0 w_{0f} + o_a w_{af} + o_b w_{bf} + o_c w_{cf} + o_d w_{df}$	$v_f = 1.694$
$v_f = 1(0.2) + 0.55(0.6) + 0.60(0.6) + 0.65(0.6) + 0.69(0.6)$	$o_f = \frac{1}{1+e^{-v_e}} = \frac{1}{1+e^{-1.694}}$
	$o_f = 0.845$

b) We calculate the network error of the instance as follows:
Network error is defined as the sum of the squared errors of the output neurons. To accomplish this, we find the error of output neurons e and f. We know that the instance $[1, 0.5, 0.5$ is labelled as "dog", so $d_e = 1$ and $d_f = 0$.

$e_e = d_e - o_e$	$e_f = d_f - o_f$
$e_e = 1 - 0.81 = 0.19$	$e_f = 0 - 0.845 = -0.845$

$E = \frac{1}{2} \sum_j e_j^2$ for j is the set of output neurons
$E = \frac{1}{2}[(0.19)^2 + (-0.845)^2]$
The network error of instance $[1, 0.5, 0.5]$ is $E = 0.375$.

c) We calculate local gradient for nodes e and f as follows:
Using the local gradient formula for output neurons $\delta_k = o_k(1 - o_k)(d_k - o_k)$,

$\delta_e = o_e(1 - o_e)(d_e - o_e)$	$\delta_f = o_f(1 - o_f)(d_f - o_f)$
$\delta_e = 0.81(1 - 0.81)(1 - 0.81) = 0.029$	$\delta_f = 0.845(1 - 0.845)(0 - 0.845) = -0.111$

d) We calculate local gradient for nodes a and b as follows:
Using the local gradient formula for hidden layer neurons $\delta_h = o_h(1 - o_h) \sum_k = w_{h,k}\delta_k$ for k is a neuron in the output layer:

$\delta_a = o_a(1 - o_a) \sum_k = w_{a,k}\delta_k$	$\delta_b = o_b(1 - o_b) \sum_k = w_{b,k}\delta_k$
$\delta_a = o_a(1 - o_a)[w_{ae}\delta_e + w_{af}\delta_f]$	$\delta_b = o_h(1 - o_h)[w_{be}\delta_e + w_{bf}\delta_f]$
$\delta_a =$ 0.55(1-0.55)[0.5(0.029)+0.6(-0.111)]	$\delta_b =$ 0.6(1-0.6)[0.5(0.029)+0.6(-0.111)]
$\delta_a = -0.0129$	$\delta_b = -0.0125$

□

Problem 5.

Solution.

a) We find the actual outputs of each instance from the network by inputting the values of the instance into the network and going forward and the results of our output neurons are the actual output of the instance. There are three instances I_1, I_2, I_3, their actual outputs are calculated as follows:

i) Instance I_1:

$$
\begin{array}{lll}
v = x_0w_{0a}+x_1w_{1a}+x_2w_{2a} & v_b = x_0w_{0b}+x_1w_{1b}+x_2w_{2b} & v_c = x_0w_{0c}+x_1w_{1c}+x_2w_{2c} \\
v_a = 1(1)+1(1)+0.5(1) & v_b = 1(1)+1(1)+0.5(1) & v_c = 1(1)+1(1)+0.5(1) \\
v_a = 2.5 & v_b = 2.5 & v_c = 2.5 \\
o_a = \frac{1}{1+e^{-2.5}} = 0.924 & o_b = \frac{1}{1+e^{-2.5}} = 0.924 & o_c = \frac{1}{1+e^{-2.5}} = 0.924
\end{array}
$$

ii) Instance I_2:

$$
\begin{array}{lll}
v = x_0w_{0a}+x_1w_{1a}+x_2w_{2a} & v_b = x_0w_{0b}+x_1w_{1b}+x_2w_{2b} & v_c = x_0w_{0c}+x_1w_{1c}+x_2w_{2c} \\
v_a = 1(1)+0(1)+1(1) & v_b = 1(1)+0(1)+1(1) & v_c = 1(1)+0(1)+1(1) \\
v_a = 2 & v_b = 2 & v_c = 2 \\
o_a = \frac{1}{1+e^{-2}} = 0.881 & o_b = \frac{1}{1+e^{-2}} = 0.881 & o_c = \frac{1}{1+e^{-2}} = 0.881
\end{array}
$$

iii) Instance I_3:

$$
\begin{array}{lll}
v = x_0w_{0a}+x_1w_{1a}+x_2w_{2a} & v_b = x_0w_{0b}+x_1w_{1b}+x_2w_{2b} & v_c = x_0w_{0c}+x_1w_{1c}+x_2w_{2c} \\
v_a = 1(1)+0.5(1)+0.5(1) & v_b = 1(1)+0.5(1)+0.5(1) & v_c = 1(1)+0.5(1)+0.5(1) \\
v_a = 2 & v_b = 2 & v_c = 2 \\
o_a = \frac{1}{1+e^{-2}} = 0.881 & o_b = \frac{1}{1+e^{-2}} = 0.881 & o_c = \frac{1}{1+e^{-2}} = 0.881
\end{array}
$$

b) Finding the mean squared error of the network with respect to the three instances means that we will need to find the total squared error for each instance and then divide it by the number of instances. The mean squared error with respect to these instances is calculated as follows:

$$
\begin{array}{lll}
E(I_1) = \frac{1}{2}\sum_j e_j^2(I_1) & E(I_2) = \frac{1}{2}\sum_j e_j^2(I_2) & E(I_3) = \frac{1}{2}\sum_j e_j^2(I_3) \\
E(I_1) = \frac{1}{2}[(d_c(I_1)-o_c(I_1))^2] & E(I_2) = \frac{1}{2}[(d_c(I_2)-o_c(I_2))^2] & E(I_3) = \frac{1}{2}[(d_c(I_3)-o_c(I_3))^2] \\
E(I_1) = \frac{1}{2}[(1-0.924)^2] & E(I_2) = \frac{1}{2}[(1-0.881)^2] & E(I_3) = \frac{1}{2}[(0-0.881)^2] \\
E(I_1) = 0.0029 & E(I_2) = 0.0071 & E(I_3) = 0.3881
\end{array}
$$

$$
E(W) = \frac{1}{3}(0.029 + 0.0071 + 0.3881) = 0.1327
$$

c) Using I_1 to update the weight of the network, the updated weights are calculated as follows:

- Step 1 is to calculate δ_k for output layers, which ends up being δ_c
  $\delta_c = o_c(1-o_c)(d_c-o_c)$
  $\delta_c = 0.924(1-0.924)(1-0.924)$
  $\delta_c = 0.0053$
- Step 2 is to calculate δ_h for hidden layers, which ends up being δ_a and δ_b:

$$
\begin{array}{ll}
\delta_a = o_a(1-o_a)(w_{ac}\delta_c) & \delta_b = o_b(1-o_b)(w_{bc}\delta_c) \\
\delta_a = 0.924(1-0.924)(1\cdot 0.0053) & \delta_b = 0.924(1-0.924)(1\cdot 0.0053) \\
\delta_a = 0.00037 & \delta_b = 0.00037
\end{array}
$$

- Now, final step is to update the weights using the formula $w_{ij} = w_{ij} + \eta\delta_j x_{ij}$:

$$
\begin{array}{lll}
w_{0a} = 1.000037 & w_{0b} = 1.000037 & w_{0c} = 1.00053 \\
w_{1a} = 1.000037 & w_{1b} = 1.000037 & w_{1c} = 1.00053 \\
w_{2a} = 1 & w_{2b} = 1 & w_{2c} = 1
\end{array}
$$

$\square$