Machine Learning - Written Assignment 1

khaled tamimy

October 8, 2016

1 Question 3:

1.1 a)

We have:

$$a_1^{(1)} = 0.5 \tag{1}$$

$$a_2^{(1)} = 0.9 \tag{2}$$

Thus:

$$a_1^{(2)} = g(0.2 + 0.5 * 0.5 + 0.1 * 0.9) \tag{3}$$

$$a_1^{(2)} = g(0.54) = 0.63181 \tag{4}$$

And,

$$a_2^{(2)} = g(0.2 + 0.5 * 0.5 + 0.7 * 0.9) \tag{5}$$

$$a_2^{(2)} = g(1.08) = 0.74649 \tag{6}$$

And, finally:

$$a_1^{(3)} = g(0.2 + 1 * 0.63181 + 2 * 0.74649) \tag{7}$$

$$a_1^{(3)} = 0.91091 \tag{8}$$

1.2 b)

We have:

$$\delta^{(l)} = ((\theta^{(l)})^T * \delta^{(l+1)}) * a^{(l)} * (1 - a^{(l)}) \tag{9}$$

And:

$$\delta^{(L)} = \delta^{(3)} = a^{(L)} - y = a^{(3)} - 1 \tag{10}$$

$$\delta^{(L)} = -0.08909 \tag{11}$$

And for the 2nd layer we have:

$$\delta^{(2)} = ((\theta^{(2)})^T * \delta^{(3)}) * a^{(2)} * (1 - a^{(2)}) \tag{12}$$

$$\delta^{(2)} = ((\theta^{(2)})^T * \delta^{(3)}) * a^{(2)} * (1 - a^{(2)}) \tag{13}$$

Thus:

$$\delta_1^{(2)} = -0.020724661378251 \tag{14}$$

$$\delta_2^{(2)} = -0.016859630352291 \tag{15}$$

And for the 1st layer we have:

$$\delta^{(1)} = ((\theta^{(1)})^T * \delta^{(2)}) * a^{(1)} * (1 - a^{(1)}) \tag{16}$$

$$\delta^{(1)} = ((\theta^{(1)})^T * \delta^{(2)}) * a^{(1)} * (1 - a^{(1)}) \tag{17}$$

Thus:

$$\delta^{(1)} = (-0.00638932959419214, -0.004717230510705792) \tag{18}$$

Now we compute the partial derivatives. We have:

$$\theta_{ij}^{(l)} = a_j^{(l)} * \delta_i^{(l+1)} \tag{19}$$

For $\theta_{11}^{(1)}$ we have:

$$\theta_{11}^{(1)} = a_1^{(1)} * \delta_1^{(2)} = 0.5 * -0.020724661378251 = -0.0103623306891255 \tag{20}$$

For $\theta_{12}^{(1)}$ we have:

$$\theta_{12}^{(1)} = a_2^{(1)} * \delta_1^{(2)} = 0.9 * -0.020724661378251 = -0.0186521952404259 \tag{21}$$

For $\theta_{21}^{(1}$ we have:

$$\theta_{21}^{(1)} = a_1^{(1)} * \delta_2^{(2)} = 0.5 * -0.016859630352291 = -0.0084298151761455 \tag{22}$$

For $\theta_{22}^{(1}$ we have:

$$\theta_{22}^{(1)} = a_2^{(1)} * \delta_2^{(2)} = 0.7 * -0.016859630352291 = -0.0118017412466037 \tag{23}$$

For $\theta_1^{(2}$ we have:

$$\theta_1^{(2)} = a_1^{(2)} * \delta_1^{(3)} = 1 * 0.63181 = 0.63181 \tag{24}$$

For $\theta_2^{(2}$ we have:

$$\theta_2^{(2)} = a_2^{(2)} * \delta_2^{(3)} = 2 * 0.74649 = -1.49298 \tag{25}$$

The update for the weights (1 iteration) then is:

$$\theta_{updated}^{(1)} = \theta^{(1)} - \alpha * A \tag{26}$$

Where A is the matrix with $\theta_{ij}^{(1)}$'s partial derivative as the ij-th entry, and where α is the learning rate.

And for $\theta^{(2)}$:

$$\theta_{updated}^{(2)} = \theta^{(2)} - \alpha * A \tag{27}$$

Where A is the matrix with $\theta_{ij}^{(2)}$'s partial derivative as the ij-th entry, and where α is the learning rate.

2 Question 4:

2.1 4.1)

The surface corsses the x1-axis at -1 and the x2-axis at 2, hence, the equation for the line is:

$$(-2) + (-2)x_1 + (1)x_2 = 0 \tag{28}$$

And for the hypothesis we have:

If $w_0 + w_1 * x_1 + w_2 * x_2 > 0$:

$$O(x_0, x_1, x_2) = 1 \tag{29}$$

Otherwise:

$$O(x_0, x_1, x_2) = -1 \tag{30}$$

Thus the weights are: $w_0 = -2$ and $w_1 = -2$ and $w_2 = 1$

2.2 4.2a)

We want the perceptron to evaluate to 1 only when $A = 1$ and $B = 0$. Hence, if we choose $w_1 = 1$ and $w_2 = 1$, we have the following situations, and thus the required Boolean function A AND (NOT B):

if A=1 and B=1, $(1)(1) + (-1)(1) = 0$ and thus $O(x1, x2) = -1$
if A=1 and B=0, $(1)(1) + (-1)(0) = 1 > 0$ and thus $O(x1, x2) = 1$
if A=0 and B=1, $(1)(0) + (-1)(1) < 0$ and thus $O(x1, x2) = -1$
if A=0 and B=0, $(1)(0) + (-1)(0) = 0$ and thus $O(x1, x2) = -1$

2.3 4.2b)

The two-layer network consists of 2 nodes as input, 2 nodes in the hidden layer and 1 output node. The first node in the hidden layer corresponds to an Boolean function A AND (NOT B) and the second node in the hidden layer corresponds to an Boolean function (NOT A) AND B. Then, finally, the node in the output layer corresponds to an Boolean A OR B function.

The respective weights are:

$w_{A,A\bar{B}} = 1$
$w_{A,\bar{A}B} = -1$
$w_{B,A\bar{B}} = -1$

$w_{B,\bar{A}B} = 1$
$w_{A\bar{B},XOR} = 1$
$w_{\bar{A}B,XOR} = 1$

Where the first subscript represents from what node the line is drawn and the second subscript represents TO what node the line is drawn