Chapter 10: Widrow-Hoff Learning

Brandon Morgan

1/18/2021

E10.2

We are wanting to classify two groups of patterns, one with rows the other with columns. Thus for class 1 we will get the following input vectors, 1 for filled in tile, -1 for empty. We are using 1 and -1 because the linear transformation function is being used.

Class 1

$$p_1^T = \begin{bmatrix} 1 & 1 & -1 & -1 \end{bmatrix}, t_1 = 1$$

$$p_2^T = \begin{bmatrix} -1 & -1 & 1 & 1 \end{bmatrix}, t_2 = 1$$

Class 2

$$p_3^T = \begin{bmatrix} 1 & -1 & 1 & -1 \end{bmatrix}, t_3 = -1$$

$$p_4^T = \begin{bmatrix} -1 & 1 & -1 & 1 \end{bmatrix}, t_4 = -1$$

1

Here we will use the LMS algorithm to train an ADALINE network to classify these two groups. An ADALINE network is given by the following transformation $a(k) = purelin(Wp(k) + b$. The LMS algorithm is detailed to be $W(k+1) = W(k) + 2\alpha e(k)p^T(k)$ and $b(k+1) = b(k) + 2\alpha e(k)$, where W denotes the weights, α denotes the learning rate, e denotes the error, and b denotes the bias.

Because neither the learning rate, bias, or initial weights were given, they will all be set to zero, except for the learning rate, where $\alpha = 0.2$ will be used.

$$w_0^T = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, b_0 = 0, \alpha = 0.2$$

Iteration 1

Input p_1

First, we pass in our first input vector, p_1:

$$a(0) = \text{purelin}[\begin{bmatrix}0 & 0 & 0 & 0\end{bmatrix} \begin{bmatrix}1 \\ 1 \\ -1 \\ -1\end{bmatrix} + 0] = 0$$

Next, we calculate the error by subtracting the real classification value by the outcome derived a second ago:

$$e(0) = t(0) - a(0) = 1 - 0 = 1$$

Now, we update the weights:

$$W(1) = W(0) + 2\alpha e(0)p(0)^T = \begin{bmatrix}0 & 0 & 0 & 0\end{bmatrix} + 2(0.2)(1)\begin{bmatrix}1 & 1 & -1 & -1\end{bmatrix} = \begin{bmatrix}.4 & .4 & -.4 & -.4\end{bmatrix}$$

Now, we update the bias:

$$b(1) = b(0) + 2\alpha e(0) = 0 + 2(0.2)(1) = 0.4$$

Input p_2

First, we pass in our second input vector, p_2:

$$a(1) = \text{purelin}[\begin{bmatrix}.4 & .4 & -.4 & -.4\end{bmatrix} \begin{bmatrix}-1 \\ -1 \\ 1 \\ 1\end{bmatrix} + 0.4] = -1.2$$

Next, we calculate the error by subtracting the real classification value by the outcome derived a second ago:

$$e(1) = t(1) - a(1) = 1 - (-1.2) = 2.2$$

Now, we update the weights:

$$W(2) = \begin{bmatrix}.4 & .4 & -.4 & -.4\end{bmatrix} + 2(0.2)(2.2)\begin{bmatrix}-1 & -1 & 1 & 1\end{bmatrix} = \begin{bmatrix}-.48 & -.48 & .48 & .48\end{bmatrix}$$

Now, we update the bias:

$$b(2) = b(1) + 2\alpha e(1) = 0.4 + 2(0.2)(2.2) = 1.28$$

Input p_3

First, we pass in our third input vector, p_3:

$$a(2) = \text{purelin}[\begin{bmatrix}-.48 & -.48 & .48 & .48\end{bmatrix} \begin{bmatrix}1 \\ -1 \\ 1 \\ -1\end{bmatrix} + 1.28] = 1.28$$

Next, we calculate the error by subtracting the real classification value by the outcome derived a second ago:

$$e(2) = t(1) - a(1) = -1 - 1.28 = -2.28$$

Now, we update the weights:

$$W(2) = \begin{bmatrix}-.48 & -.48 & .48 & .48\end{bmatrix} + 2(0.2)(-2.28) \begin{bmatrix}1 & -1 & 1 & -1\end{bmatrix} = \begin{bmatrix}-1.392 & 0.432 & -0.432 & 1.392\end{bmatrix}$$

Now, we update the bias:

$$b(2) = b(1) + 2\alpha e(1) = 1.28 + 2(0.2)(-2.28) = 0.368$$

Input p_4

First, we pass in our fourth input vector, p_4:

$$a(3) = \text{purelin}[\begin{bmatrix}-1.392 & 0.432 & -0.432 & 1.392\end{bmatrix} \begin{bmatrix}-1 \\ 1 \\ -1 \\ 1\end{bmatrix} + 0.368] = 4.016$$

Next, we calculate the error by subtracting the real classification value by the outcome derived a second ago:

$$e(3) = t(2) - a(2) = -1 - 4.016 = -1.6384$$

Now, we update the weights:

$$W(3) = \begin{bmatrix}-1.392 & 0.432 & -0.432 & 1.392\end{bmatrix} + 2(0.2)(-5.016) \begin{bmatrix}-1 & 1 & -1 & 1\end{bmatrix} = \begin{bmatrix}0.614 & -1.574 & 1.574 & -0.614\end{bmatrix}$$

Now, we update the bias:

$$b(3) = b(2) + 2\alpha e(2) = 0.368 + 2(0.2)(-2.28) = -4.648$$

Conclusion

Here we have ran one iteration of the LMS algorithm from scratch; more iterations will be necessary before a convergence has resulted.

2

The only reason the ADALINE network might have difficulty with this problem is if the two categories are not linearly seperable. Because the input vectors p_i are in four dimensions, we are unable to graph these vectors to check if they are linearly seperable.