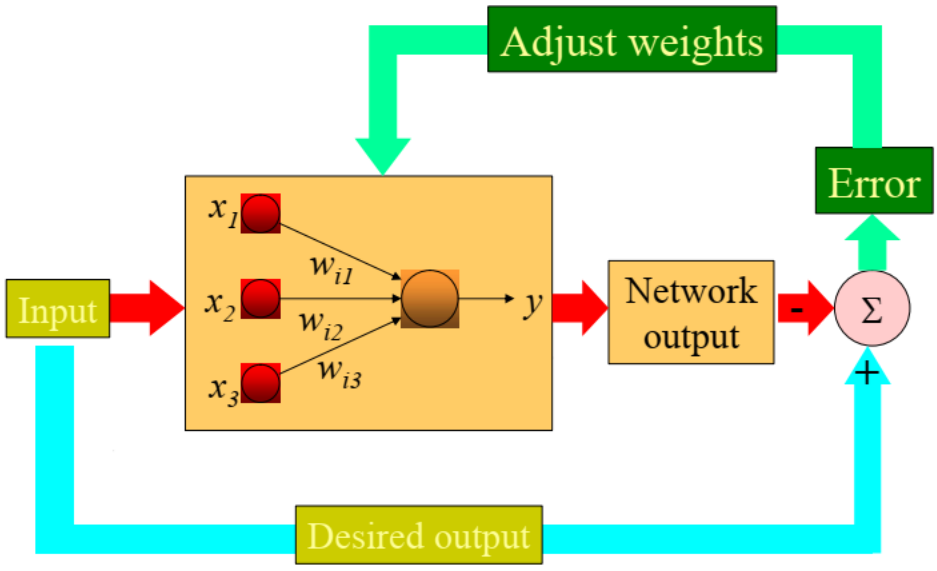
1. Perceptrons

The perceptron is the simplest form of a neural network used for the classification of patterns said to be linearly separable . basically , it consists of a single neuron with adjustable synaptic weights and bais.

The summing node of the neuronal model computes a linear combination of the inputs applied to its synapses, and also incorporates an externally applied bais. The resulting sum, that is applied to a hard limiter.accordingly, the neuron produced an output equal to +1 if the hard limiter input is positive, and -1 if it is negative.

Learning Algorithm: Training Perceptron Networks



Perceptron Algorithm

Step 1 – Initialize the following to start the training:

- Weights
- Bias
- Learning rate α

For easy calculation and simplicity, weights and bias must be set equal to 0 and the learning rate must be set equal to 1.

Step 2 – Continue step 3-8 when the stopping condition is not true.

Step 3 – Continue step 4-6 for every bipolar training pair s : t.

Step 4 – Activate each input unit as follows:

$$x_i = S_i \ (i=1 \text{ to } n)$$

Step 5 – Obtain the net input with the following relation:

$$y_{in} = b + \sum_{i}^{n} x_i \ w_i$$

Here 'b' is bias and 'n' is the total number of input neurons.

Step 6 – Apply the following activation function to obtain the final output:

$$a=hardlim(n)= \begin{cases} 1 \ if \ n \geq 0 \\ 0 \ otherwise. \end{cases}$$

Step 7 – Adjust the weight and bias as follows :

Case 1 – if $y \neq t$ then, $w_i(new) = w_i(old)+\alpha(t-y_{in})x_i$

$$b(new) = b(old)+\alpha(t-y_{in})$$

Case 2 – if $y = t$ then, $w_i(new) = w_i(old)$

$$b(new) = b(old)$$

Here 'y' is the actual output and 't' is the desired/target output. $(t-y_{in})$ is the computed error.

Step 8 – Test for the stopping condition, which will happen when there is no change in weight or the highest weight change occurred during training is smaller than the specified tolerance.

note

$$W_i(new\ ) = W_i\ \ (old\ ) + \eta(d_k - y_k).x_i, 0 \le i \le N$$

$b^{new} = b^{old} + e$. where $\mathbf{e} = \mathbf{d} - \mathbf{y}$

Where η(the learning rate) is a positive gain fraction less than 1 and *dk* is the desired output. Note that the weights remain the same if the network makes the correct decision.

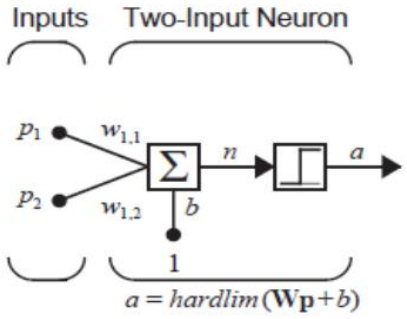
Example:

The output of the network is given by

a = **hardlim(Wp** + **b)**

$$a = hardlim(n) = \begin{cases} 1 \text{ if } n \ge 0 \\ 0 \text{ otherwise.} \end{cases}$$

$= hardlim(w1, 1\ p1 + w1, 2\ p2 + b)$



Unified Learning Rule

The three rules in Eq. (4.31) can be rewritten as a single expression. First we will define a new variable, the perceptron error *e*:

$$e = t - a. \tag{4.32}$$

We can now rewrite the three rules of Eq. (4.31) as:

$$\begin{aligned} &\text{If } e = 1, \text{ then } {}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} + \mathbf{p}. \\ &\text{If } e = -1, \text{ then } {}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old} - \mathbf{p}. \\ &\text{If } e = 0, \text{ then } {}_1\mathbf{w}^{new} = {}_1\mathbf{w}^{old}. \end{aligned} \tag{4.33}$$

weight vector of a single neuron perceptron. We can generalize this rule for the multiple-neuron perceptron of Figure 4.1 as follows. To update the *i*th row of the weight matrix use:

$$ {}_i\mathbf{w}^{new} = {}_i\mathbf{w}^{old} + e_i\mathbf{p} . \tag{4.36} $$

To update the *i*th element of the bias vector use:

$$ b_i^{new} = b_i^{old} + e_i . \tag{4.37} $$

The *perceptron rule* can be written conveniently in matrix notation:

$$ \mathbf{W}^{new} = \mathbf{W}^{old} + \mathbf{e}\mathbf{p}^T , \tag{4.38} $$

and

$$ \mathbf{b}^{new} = \mathbf{b}^{old} + \mathbf{e} . \tag{4.39} $$

Perceptron convergence procedure (1)

Step1: Initialize weights and thresholds: Set the w_{ij} and the b_j to small random values in the range [-1,+1].

Step2: Present new continuous valued input $p_0, p_1, \cdots, p_n$ along with the desired output vector $T = (t_0, \cdots, t_n)$.

Step3: Calculate actual output: $a_j = f(\Sigma w_{ij} p_i + b_j)$ where f(.) = hardlim(.) = sgn(.).

Step 4: When an error occurs adapt the weights with:
$w_{ij}^{new} = w_{ij}^{old} + \eta[t_j - a_j] \times p_i$ where: $0 < \eta \leq 1$ (learning rate)
and: $b^{new} = b^{old} + e$ where: $e = \eta[t_j - a_j]$

Step 5: Repeat by going to step 2 until no error.

مثال \\ قم بتدريب شبكة perceptron للتعلم على مدخلات ومخرجات بوابة OR علماً ان معامل التعلم 1 والاوزان نفرضها عشوائية.

سوف ندرب الصف الاول من البوابة

x1	x2	T
0	**0**	**0**
0	1	1
1	0	1
1	1	1

$$x1=0 , x2 = 0 ,T=0 , w1= 0, w2= 0$$

سيتم تدريب وحدة المدرك الحسي وفقاً لعملية المنطقية او "OR". ولنبدأ باختيار الاوزان عشوائياً ولتكون قيمتها صفراً في البداية وقيمة معدل التعلم 1(وذلك لتسهيل العلميات الحسابية).

ويتم تحديث الاوزان وفقا للقاعدة تحديث الاوزان وعلى النحو الاتي:

$$W_1 \text{ new} = W_1 \text{ old} + 1 (y - \hat{y}) X_1$$

$$W_2 \text{ new} = W_2 \text{ old} + 1 (y - \hat{y}) X_2$$

وتستمر عملية التدريب مع تكرار تحديث الاوزان حتى تستقر الاوزان عند قيم معينة لا تتغير بعدها. ويتم حساب المخرجات وفقاً لصيغة الاتية:

$$Y = \begin{cases} 1 & If \quad \sum_{i=1}^{n} W_i x_i > o \\ 0 & Otherwise \end{cases}$$

$$\hat{y}_1 = f(X_1W_1 + X_2W_2) = f(0*0 + 0*0) = f(0) = 0$$

1- تعليم الخلية على استخراج وحساب القيمة الصحيحة للبوابة وذلك من خلال معادلة المجموع التالية :

s= w1* x1+ w2 * x2
= 0 * 0 + 0 * 0
= 0 == T

2- مقارنة القيمة الناتجة من المعادلة s مع حد العتبة حيث ان حد العتبة يكون اما يساوي 1 او اصغر او يساوي الصفر ومن ثم مقارنة النتيجة مرة ثانية مع النتيجة الاصلية فاذا كانت مطابقة وخالية من الخطا فانه سوف يتم الانتقال الى الصف التالي (x1= 0 , x2 =1 , T =1) .

3- كتابة معادلة المجموع :-

s= w1* x1+ w2 * x2
= 0 * 0 + 0 * 1
= 0 == error (T =1)

4- ظهرت حالة الخطا و في هذه الحالة نغير الاوزان وذلك من خلال المعادلات الاتية

w1new=w1old +a*(o desired(المطلوبة)- o (الناتجة)) * x1
= 0 + 1 * (1 – 0) * 0
= **0**

w2new=w2old +a*(o desired(المطلوبة)- o (الناتجة)) * x2
= 0 + 1 * (1 – 0) * 1
= **1**

5- نطبق المعادلة الخاصة بالمجموع ولكن هذه المرة مع الاوزان الجديدة والصف التالي من المدخلات (x1= 1 , x2= 0 , T= 1) وكما يلي :

$s = w1 * x1 + w2 * x2$
$= 0 * 1 + 1 * 1$
$= 1 \quad == T$

والان نطبقها مع الصف الاخير (X1= 1, X2 =1 , T =1)

$s = w1 * x1 + w2 * x2$
$= 1 * 1 + 1 * 1$
$= 2$ وهذه القيمة مطابقة لشرط حد العتبة و T

ولتدريب الخلية تطبق المعادلة على كل المدخلات مع تكرار العملية حتى تستقر الاوزان عند حد معين ولاتتغير بعدها ، اي انه بعد ان طبقت المعادلة على الصف الاخير فانه ترجع من جديد اي من بداية المدخلات وتصحيح كل الحالات وتغير الاوزان الى ان تستقر .

والجدول الاتي يبين خطوات عملية تدريب الشبكة العصبية الاصطناعية.

x_1	x_2	W_1old	W_2old	y	y^	error	W_1 new	W_2 new
0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	1	0	1
1	0	0	1	1	0	1	1	1
1	1	1	1	1	1	0	1	1
0	0	1	1	0	0	0	1	1
0	1	1	1	1	1	0	1	1

EX 1:- $X_1 = [0\ 0]$; $X_2 = [0\ 1]$; $X_3 = [1\ 0]$; $X_4 = [1\ 1]$

I	T
0	0
0	0
0	0
1	1

$T_1 = 0$; $T_2 = 0$; $T_3 = 0$; $T_4 = 1$

$$f(y) = \begin{cases} 1 & ; X > 0 \\ 0 & ; X \leq 0 \end{cases}$$

$\theta = 0.2$; $\alpha = 0.1$; $W = [0.3 \quad -0.1]$

Solution:

$$Y_1 = X_1 W_1 - \theta = [0\ 0]\begin{bmatrix} 0.3 \\ -0.1 \end{bmatrix} - 0.2 = -0.2$$

$\therefore f(Y_1) = f(-0.2) = 0 \Longrightarrow \boxed{Y_1 = T_1}$

$\therefore W_2 = W_1$

$$Y_2 = X_2 W_2 - \theta = [0\ 1]\begin{bmatrix} 0.3 \\ -0.1 \end{bmatrix} - 0.2 = -0.3$$

$\therefore f(Y_2) = f(-0.3) = 0 \Longrightarrow \boxed{Y_2 = T_2}$

$$Y_3 = X_3 \cdot W_3 - \theta$$

$$[1\ 0]\begin{bmatrix} 0.3 \\ -0.1 \end{bmatrix} - 0.2 = 0.1$$

$\therefore f(Y_3) = 1 \Longrightarrow \boxed{Y_3 \neq T_3}$

$Y_3 \neq T_3$ ∴ إذن نعمل على تعديل الوزن

$$W_{4\,new} = W_{3\,old} + \alpha(T_3 - Y_3)\cdot X_3$$

$$= \begin{bmatrix} 0.3 \\ -0.1 \end{bmatrix} + 0.1(0-1)\begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0.3 \\ -0.1 \end{bmatrix} + \begin{bmatrix} -0.1 \\ 0 \end{bmatrix} \Longrightarrow \quad W_4 = \begin{bmatrix} 0.2 \\ -0.1 \end{bmatrix}$$

$$Y_4 = X_4 \cdot W_4 - \theta = \begin{bmatrix} 1 & 1 \end{bmatrix}\begin{bmatrix} 0.2 \\ -0.1 \end{bmatrix} - 0.2 = -0.1$$

$$\therefore f(-0.1) = 0 \Longrightarrow \boxed{Y_4 \neq T_4}$$

نعمل على تعديل الوزن .

$$W_{5\,new} = W_{4\,old} + \alpha(T_4 - Y_4)\cdot X_4$$

$$= \begin{bmatrix} 0.2 \\ -0.1 \end{bmatrix} + 0.1(1-0)\begin{bmatrix} 1 \\ 1 \end{bmatrix} \Longrightarrow W_5 = \begin{bmatrix} 0.3 \\ 0 \end{bmatrix}$$

$$Y_5 = X_5 \cdot W_5 - \theta = \begin{bmatrix} 0 & 0 \end{bmatrix}\begin{bmatrix} 0.3 \\ 0 \end{bmatrix} - 0.2 = -0.2$$

$$\therefore f(-0.2) = 0 \Longrightarrow \boxed{Y_5 = T_5}$$

$$\Longrightarrow W_5 = W_6$$

ومن ثم نكمل الحل . . . .

شروط التوقف

1* اذا حققت الأوزان جميع الإدخالات target الوزن المثالي بمعنى

$Y_1 = T_1$, $Y_2 = T_2$, $Y_3 = T_3$, $Y_4 = T_4$ ...

2* في حالة الأوزان لم تحقق أحد الإدخالات target نعمل على تعديل الوزن ومن ثم ننتقل الى الخطوة التالية وحساب الوزن الجديد المعدّل الى ان نصل الى الوزن المثالي في جميع الحالات .

3* اذا تم تحديد عدد التكرارات في السؤال iterations

X_1	X_2	X_3	Y = target
5	4	9	1
0	2	1	0
5	3	7	1
1	3	1	0

$W = [1 \quad 0 \quad 1]$

$$y = \begin{cases} 1 & , X > 0 \\ 0 & , X \leq 0 \end{cases}$$

b عند معطاة :-

$\therefore b = 0$

$\alpha = 0.1$

Solution :- لاحظ // اذا α لم تعطى بالسؤال معناها = 1 ، اذن

$$Y = \sum_{i=1}^{n} X_i * W_i + b$$

$Y_1 = X_1 \cdot W_1 + X_2 \cdot W_2 + X_3 \cdot W_3$

$= 5(1) + 4(0) + 9(1) = 14 \Rightarrow f(14) = 1$

$\Rightarrow Y_1 = T_1 \Rightarrow \therefore W_1 = W_2$

$Y_2 = 0(1) + 2(0) + 1(1) = 1 \Rightarrow f(1) = 1$

$\Rightarrow Y_2 \neq t_2$

:- يجب أن نعدّل الأوزان .

$W_3 = W_2 + \alpha (T_2 - Y_2) X_2$

$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + 0.1 (0-1) * \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + (-0.1) \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}$

$= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ -0.2 \\ -0.1 \end{bmatrix} \Rightarrow W_3 = \begin{bmatrix} 1 \\ -0.2 \\ 0.9 \end{bmatrix}$

$Y_3 = 5(1) + 3(-0.2) + 7(0.9) = 10.7$ (5)

$\Rightarrow f(10.7) = 1$

$\Rightarrow Y_3 = T_3 \Rightarrow \boxed{W_3 = W_4}$

$Y_4 = 1(1) + 3(-0.2) + 1(0.9) = 1.3$

$\Rightarrow f(1.3) = 1$

$\Rightarrow Y_4 \neq T_4$

∴ نعمل على تعديل الوزن

$W_5 = W_4 + \alpha(T_4 - Y_4) * X_4$

$= \begin{bmatrix} 1 \\ -0.2 \\ 0.9 \end{bmatrix} + 0.1(0-1)\begin{bmatrix} 1 \\ 3 \\ 1 \end{bmatrix} \Rightarrow W_5 = \begin{bmatrix} 0.9 \\ -0.5 \\ 0.8 \end{bmatrix}$

وهكذا ستستمر حتى نصل

$Y_1 = T_1$, $Y_2 = T_2$, $Y_3 = T_3$, $Y_4 = T_4$

عندها نتوقف وينتهي الحل .