قواعد التعلم

د. ياسر خضرا

2/11/2022

تمهيد:
تكلمنا في المحاضرة السابقة عن perceptron ثم قمنا بتوصيف مكوناته من دخل وأوزان وتابع الجمع (بالإضافة إلى الانزياح) وتابع التفعيل الذي يُقرر الخرج، ثم تكلمنا عن قاعدة hebb لل perceptron وأن ال perceptron يستخدم في Supervised Learning حيث الدخل والخرج معروفين والهدف من الشبكة هو إيجاد الأوزان، وسنتكلم في هذه المحاضرة عن كيفية تدريب الشبكة والحصول على هذه الأوزان.

Learning rules in neural network

ذكرنا سابقاً أن الهدف الرئيسي من الشبكة العصبية هو التعلم ولهذا الغرض وُجدت قواعد للتعلّم، حيث تُعرف قواعد التعلم بأنها الطرائق الرياضية أو المنطق الرياضي الذي يُعتمد عليه لمساعدة الشبكة العصبونية على فهم الشروط المحيطة بعملية التعلم والحصول على الأوزان.

1. Hebbian learning rule:

الهدف من كل دورة أثناء التدريب هو تحسين الأوزان:

$$W \leftarrow W + x^{(k)}.y$$

- ✓ W الوزن الجديد
- ✓ W الوزن السابق
- ✓ x شعاع الدخل
- ✓ k ترتيب الدفعة الحالية ضمن العينات
- ✓ y قيمة الخرج

▪ ليس المقصود ب W قيمة احدة وإنما شعاع، فإذا كان لدينا مثلاً 10 مداخل يوافقهم 10 أوزان فيجب تحسينهم جميعاً وليس قيمة واحدة.

- لدينا n دخل، على كل عنصر دخل وزن خاص به:

$$W = [\theta, W_1, W_2, \dots . W_n]^T$$

حيث θ هي الانزياح (العتبة) وهي اصطلاحاً القيمة -1 .

- $x^k = \left[-1 \; x_1^{(k)} \; x_2^{(k)} \; \ldots \ldots \; x_n^{(k)}\right]^T$

y : الخرج معروف كما قلنا سابقاً.

الخوارزمية:

يتم تهيئة الأوزان الابتدائية بأصفار ثم من أجل كل شعاع دخل نُحسّن الأوزان بما فيهم الانزياح (أي شعاع الوزن كاملاً) وفق القاعدة المشروحة سابقاً، مع الانتباه على أنه لا يوجد شرط للتوقف، وإنما سندور بعدد العينات الموجودة، الاوزان التي تنتج من آخر دورة هي الاوزان الصحيحة المطلوبة.

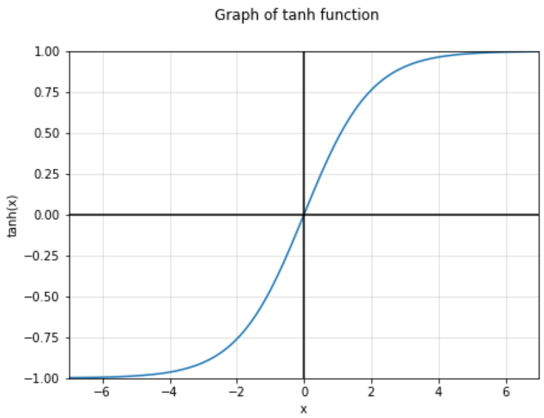
- Set all weights to zero, $w_i = 0 \; for \;\; i = 0 \; to \; n$.
- Foe each input vector, $x^{(k)}: y = d$ (target for desired output):
  - ✓ Update weight and bias by applying Hebb rule for all $i = 0 \; to \; n$.

- في هذه الخوارزمية، الخرج المعروف هو نفسه الخرج المرغوب بجدول الحقيقة،أي $y = d$

Example of Hebbian learning rule: (Boolean AND)

لدينا perceptron يريد تحقيق العلاقة المنطقية AND، حيث تابع التفعيل المستخدم في هذه القاعدة هو:

$$y = g(u) = \frac{1 - e^{-u}}{1 - e^{+u}}$$



حيث نعتمد فيه على:

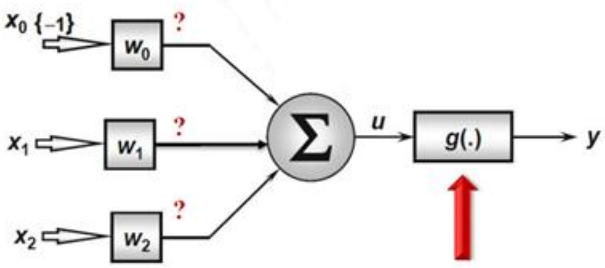
$$g(u) = \begin{cases} +1 \;\; if \;\; u \geq 0 \\ -1 \;\; if \;\; u < 0 \end{cases}$$

إذاً نلاحظ أن خرج هذا التابع لا يعطي 0، وبناءً عليه سيتغير جدول الحقيقة الخاص ب AND لأن AND تعتمد 0 منطقي و 1 منطقي، إذاً ستصبح القيمة -1 بدلاً من 0 (المبدأ نفسه ولكن الاختلاف بالترميز حيث سيصبح ترميز off: -1 و on: $+1$ اعتماداً على هذا التابع).

ومنه يكون جدول الحقيقة الخاص ب AND :

x_0	x_1	x_2	$output$
-1	-1	-1	-1
-1	-1	1	-1
-1	$+1$	-1	-1
-1	$+1$	1	1

بنية الشبكة العصبية لدينا كالتالي:



والهدف هو إيجاد الأوزان

نقسم أشعة الدخل إلى دفعات (أشعة الدخل هي أسطر جدول الحقيقة)

$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$
-1	-1	-1	-1
-1	-1	1	1
-1	1	-1	1

y	-1	-1	-1	1

إذاً أول شعاع دخل $x^{(1)}$، قيمة الخرج الموافقة له -1

ثاني شعاع دخل $x^{(2)}$، قيمة الخرج الموافقة له -1

ثالث شعاع دخل $x^{(3)}$، قيمة الخرج الموافقة له -1

رابع شعاع دخل $x^{(4)}$، قيمة الخرج الموافقة له 1

بما أنه لدينا 4 عينات دخل (أشعة الدخل) فيجب أن نقوم بأربع دورات، ويجب الانتباه إلى ما ذكرناه سابقاً أنه لا يوجد شرط للتوقف بهذه الخوارزمية، وإنما ندور بعدد العينات.

1. Iteration 1:

$$\boxed{W_{(new)} = W_{(old)} + x^{(1)}.\, y}$$

حيث $W_{(new)}$: شعاع مؤلف من 3 أوزان: وزن خاص ب x_0، ووزن خاص ب x_1 ، ووزن خاص ب x_2 .

$W_{(old)}$ الأوزان المهيئة بصفر.

$x^{(1)}$ شعاع العينة الأولى.

y الخرج المعروف سابقاً.

$$W_{(new)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \times (-1) = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

تذكرة: ضرب شعاع بقيمة: نضرب كل عنصر بالشعاع بهذه القيمة.

2. Iteration 2:

$$W_{(new)} = W_{(old)} + x^{(2)}.y$$

$$W_{(new)} = \begin{bmatrix}1\\1\\1\end{bmatrix} + \begin{bmatrix}-1\\-1\\1\end{bmatrix} \times (-1) = \begin{bmatrix}2\\2\\0\end{bmatrix}$$

3. Iteration 3:

$$W_{(new)} = W_{(old)} + x^{(3)}.y$$

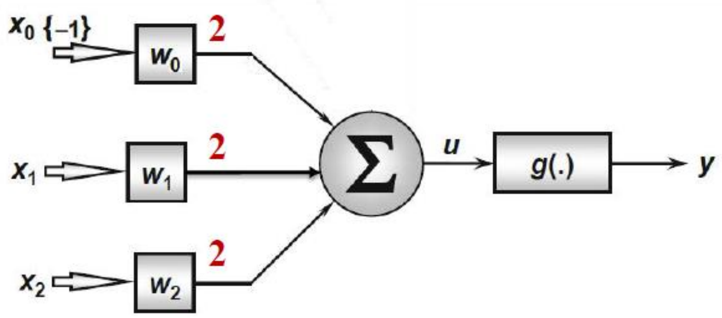
$$W_{(new)} = \begin{bmatrix}2\\2\\0\end{bmatrix} + \begin{bmatrix}-1\\1\\-1\end{bmatrix} \times (-1) = \begin{bmatrix}3\\1\\1\end{bmatrix}$$

4. Iteration 4:

$$W_{(new)} = W_{(old)} + x^{(4)}.y$$

$$W_{(new)} = \begin{bmatrix}3\\1\\1\end{bmatrix} + \begin{bmatrix}-1\\1\\1\end{bmatrix} \times (1) = \begin{bmatrix}2\\2\\2\end{bmatrix}$$

- $W_{(old)}$ في كل دورة هي الأوزان التي تنتج عن الدورة السابقة.



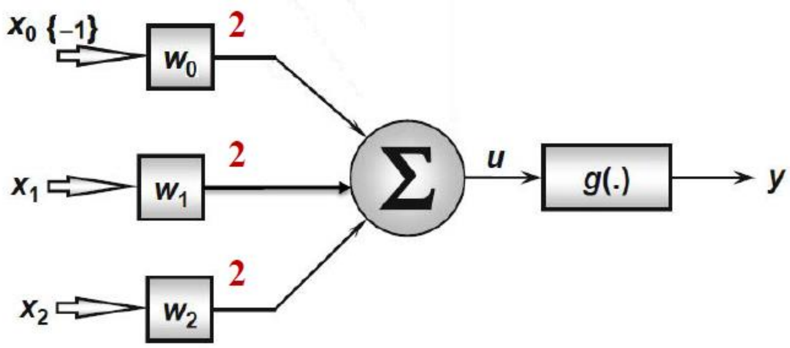
$\begin{bmatrix}2\\2\\2\end{bmatrix}$ هذه هي الأوزان النهائية المطلوبة، وللتحقق من هذه الأوزان:

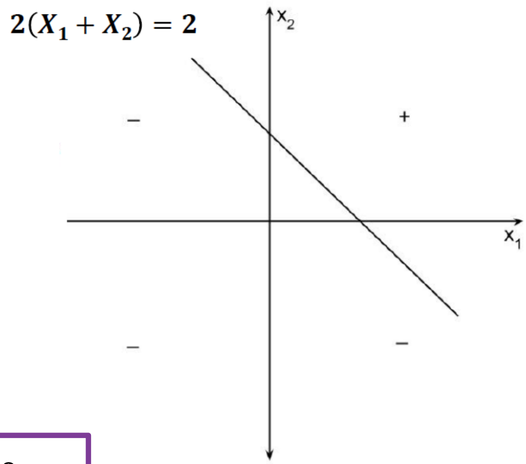
$$W^T.x^{(1)} = u = 2(-1) + 2(-1) + 2(-1) = -6.0 \Rightarrow y = g(u) = -1 \Rightarrow d^{(1)} == y$$
$$W^T.x^{(2)} = u = 2(-1) + 2(-1) + 2(1) = -2.0 \Rightarrow y = g(u) = -1 \Rightarrow d^{(2)} == y$$
$$W^T.x^{(3)} = u = 2(-1) + 2(1) + 2(-1) = -2.0 \Rightarrow y = g(u) = -1 \Rightarrow d^{(3)} == y$$
$$W^T.x^{(4)} = u = 2(-1) + 2(1) + 2(1) = 2.0 \Rightarrow y = g(u) = 1 \Rightarrow d^{(4)} == y$$

إذاً استطاعت هذه الشبكة العصبونية تحقيق بوابة AND بهذه الأوزان.

للتدريب:
يمكن تجريب المثال السابق على بوابات أخرى مثل NAND, OR....
والاختلاف فقط بجدول الحقيقة.

Decision boundary:





$$2x_0 + 2x_1 + 2x_2 = 0$$

وهي معادلة المستقيم الذي يفصل بين الصنفين، وهي عبارة عن جداء الأوزان التي نتجت لدينا بالدخل.

- إذا كانت أكبر من الصفر مع الصنف الأول وأصغر من الصفر مع الصنف الثاني:

$$2(-1) + 2(x_1 + x_2) = 0$$

الانزياح قيمته ثابتة

$$2(x_1 + x_2) = 2$$

2. Perceptron Learning Rule:

$$W \leftarrow W + \eta\left(d^{(k)} - y\right).x^{(k)}$$

✓ W الجديدة
✓ W السابقة
✓ η عامل التعلم (خطوة التعلم)
✓ $d^{(k)}$ القيمة المرغوبة بجدول الحقيقة
✓ y القيمة المحسوبة
✓ $x^{(k)}$ الدخل
✓ $d^{(k)} - y$ الفرق هو خطأ ويجب أن يكون صفر (أي يجب أن تكون القيمة المحسوبة تساوي القيمة المرغوبة)، لذلك نُحسن من خلال الخطأ الناتج.

انتباه: إلى أن y هي ناتج خرج تابع التفعيل.

الخوارزمية:

Begin {PERCEPTRON Algorithm – Training Phase}

<1> Obtain the set of training samples $\{\boldsymbol{x}^{(k)}\}$;

<2> Associate each desired output $\{d^{(k)}\}$ to each sample;

<3> Initialize vector $\boldsymbol{w}$ with small random values;

<4> Specify the learning rate $\{\eta\}$;

<5> Initialize the epoch counter $\{epoch \leftarrow 0\}$;

<6> Repeat the following instructions:

  <6.1> $error \leftarrow$ "none";

  <6.2> For all training samples $\{\boldsymbol{x}^{(k)}, d^{(k)}\}$, do:

    <6.2.1> $u \leftarrow \boldsymbol{w}^T \cdot \boldsymbol{x}^{(k)}$;

    <6.2.2> $y \leftarrow \text{signal}(u)$;

    <6.2.3> If $y \neq d^{(k)}$

      <6.2.3.1> then $\begin{cases} \boldsymbol{w} \leftarrow \boldsymbol{w} + \eta \cdot (d^{(k)} - y) \cdot \boldsymbol{x}^{(k)} \\ error \leftarrow \text{"existent"} \end{cases}$

  <6.3> $epoch \leftarrow epoch + 1$;

Until: $error \leftarrow$ "none"

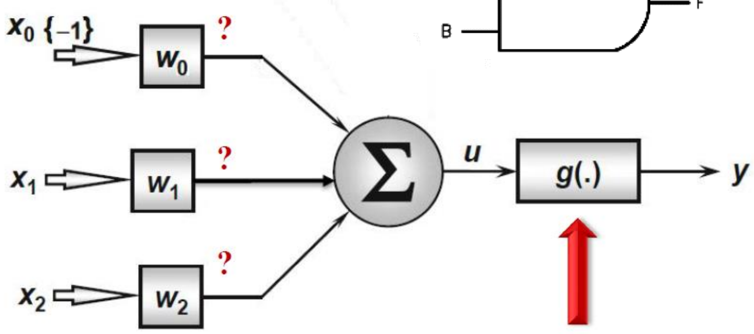
End {PERCEPTRON Algorithm – Training Phase}

بداية نحصل على عناصر التدريب $x^{(k)}$ ثم نرفق مع كل شعاع تدريب القيمة المرغوبة الخاصة فيه ثم نهيئ W بقيم عشوائية (ليس بالضرورة أن تكون متساوية، ولا يجب أن تكون صغيرة جداً ولا كبيرة جداً).

- في هذه الخوارزمية كلما كانت التهيئة قريبة للصحيح كلما كان أفضل، لأنه يوجد شرط للتوقف، وهو الّا يكون هناك خطأ (أي أنه من الممكن أن نقف بعد دورة واحدة بحال تحقق هذا الشرط)
- ثم نحدد معامل التعلم η وهو قيمة محصورة بين $0 < \eta < 1$ وهو قيمة اختيارية والأفضل أن تكون متوسطة ثم نهيئ **epoch** بصفر (للدلالة على أول دورة)، ثم نكرر الدورات إلى أن تَعبر جميع أشعة الدخل دون خطأ بين القيمة المحسوبة والقيمة المرغوبة، ومنه نكون قد حصلنا على الاوزان النهائية.

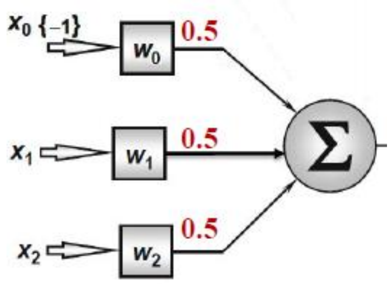
Example of perceptron learning rule (Boolean AND):

x_0	x_1	x_2	$output$
-1	0	0	0
-1	0	1	0
-1	$+1$	0	0
-1	$+1$	1	1



$$g(u) = \begin{cases} 1, & \text{if } u \geq 0 \\ 0, & \text{if } u < 0 \end{cases}$$

- كما نلاحظ تابع التفعيل في هذه القاعدة هو **Step function**
- كل قاعدة تفرض تابع تنشيط معين



$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$
-1	-1	-1	-1
0	0	1	1
0	1	0	1

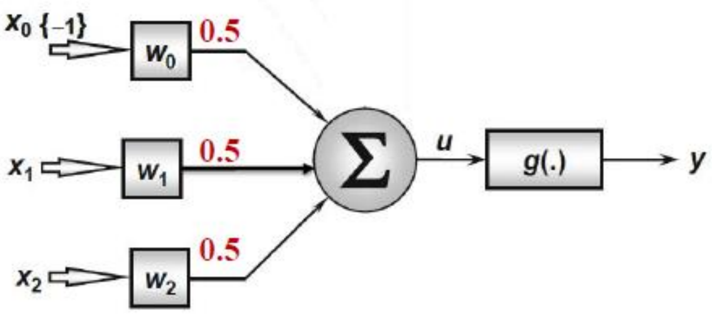
$d^{(1)}$	$d^{(2)}$	$d^{(3)}$	$d^{(4)}$
0	0	0	1

W
0.5
0.5
0.5

$\eta = 0.5$

1. Epoch 1:

$$W^T = [0.5\ \ 0.5\ \ 0.5] \begin{array}{cccc} x^{(1)} & x^{(2)} & x^{(3)} & x^{(4)} \\ \begin{bmatrix}-1\\0\\0\end{bmatrix} & \begin{bmatrix}-1\\0\\1\end{bmatrix} & \begin{bmatrix}-1\\1\\0\end{bmatrix} & \begin{bmatrix}-1\\1\\1\end{bmatrix} \end{array}$$



Error = none

$$W^T.x^{(1)} = u = (0.5)(-1) + (0.5)(0) + (0.5)(0) = -0.5$$

وفقاً لتابع التفعيل المذكور سابقاً، إذا كانت $g(u) = 0 \Leftarrow u < 0$

$$\Rightarrow y = g(u) = 0 \Rightarrow d^{(1)} == y$$

إذاً ليس هناك ***error*** وبالتالي لا يوجد تحديث للأوزان.

$$W^T.x^{(2)} = u = (0.5)(-1) + (0.5)(0) + (0.5)(1) = 0$$

$$y = g(u) = 1 \Rightarrow d^{(2)} \neq y$$

هناك خطأ، إذاً يجب تحديث الأوزان (بما فيهم وزن الانزياح):

$$W = W + \eta\left(d^{(2)} - y\right)x^{(2)}$$

$$W_0 = 0.5 + (0.5)(0-1)(-1) = 1.0$$

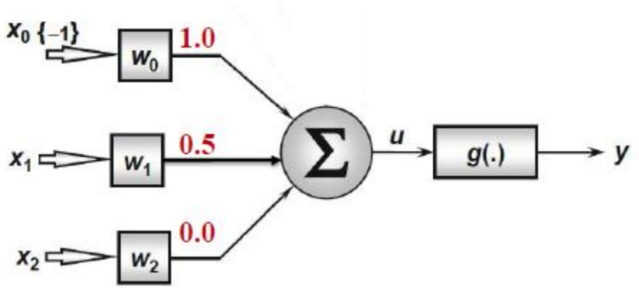
$$W_1 = 0.5 + (0.5)(0-1)(0) = 0.5$$

$$W_2 = 0.5 + (0.5)(0-1)(1) = 0.0$$

$$\Rightarrow W^T = [1.0\ \ 0.5\ \ 0.0]$$
$$error\ \neq none$$

يجب الانتباه في العينات القادمة إلى أنّ شعاع الأوزان تحدث.

$$W^T.x^{(3)} = u = (1.0)(-1) + (0.5)(1) + (0.0)(0) = -0.5$$
$$y = g(u) = 0 \Rightarrow d^{(3)} == y$$



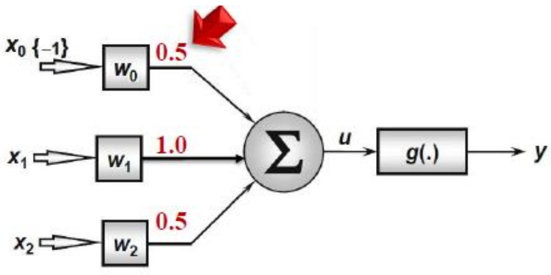
$$W^Tx^{(4)} = u = (1.0)(-1) + (0.5)(1) + (0.0)(1) = -0.5 \Rightarrow y = g(u) = 0 \Rightarrow d^{(4)} \neq y$$
$$W = W + \eta\big(d^{(4)} - y\big)x^{(4)}$$
$$W_0 = 1.0 + (0.5)(1-0)(-1) = 0.5$$
$$W_1 = 0.5 + (0.5)(1-0)(1) = 1.0$$
$$W_2 = 0.0 + (0.5)(1-0)(1) = 0.5$$
$$\Rightarrow W^T = [0.5\ \ 1.0\ \ 0.5]$$
$$error\ \neq none$$

2. Epoch 2:

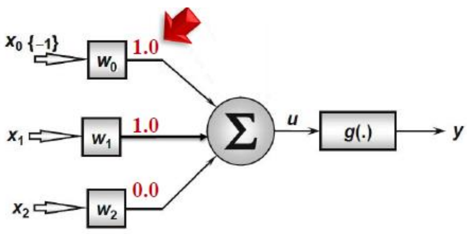
W^T	0.5	1.0	0.5

Error = none

$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$
-1	-1	-1	-1
0	0	1	1
0	1	0	1



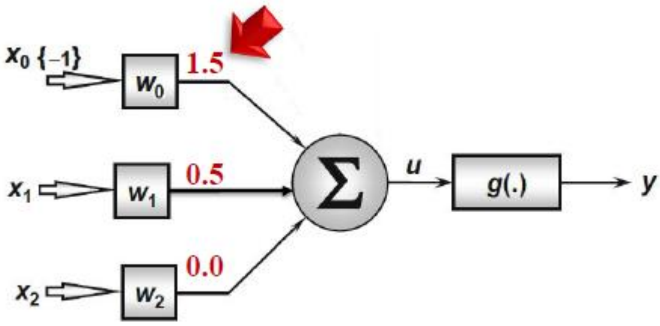
$$W^Tx^{(1)} = u = (0.5)(-1) + (1.0)(0) + (0.5)(0) = -0.5 \Rightarrow y = g(u) = 0 \Rightarrow d^{(1)} = y$$
$$W^Tx^{(2)} = u = (0.5)(-1) + (1.0)(0) + (0.5)(1) = 0 \Rightarrow y = g(u) = 1 \Rightarrow d^{(2)} \neq y$$
$$W = W + \eta\big(d^{(2)} - y\big)x^{(2)}$$
$$W_0 = 0.5 + (0.5)(0-1)(-1) = 1.0$$
$$W_1 = 1.0 + (0.5)(0-1)(0) = 1.0$$
$$W_2 = 0.5 + (0.5)(0-1)(1) = 0.0$$
$$\Rightarrow W^T = [1.0\ \ 1.0\ \ 0.0]$$
$$error\ \neq none$$



W^T	1.0	1.0	0.0

$$W^T x^{(3)} = u = (1.0)(-1) + (1.0)(1) + (0.0)(0) = 0 \Rightarrow y = g(u) = 1 \Rightarrow d^{(3)} \neq y$$
$$W = W + \eta\left(d^{(3)} - y\right)x^{(3)}$$
$$W_0 = 1.0 + (0.5)(0-1)(-1) = 1.5$$
$$W_1 = 1.0 + (0.5)(0-1)(1) = 0.5$$
$$W_2 = 0.0 + (0.5)(0-1)(0) = 0.0$$
$$\Rightarrow W^T = [1.5\ \ 0.5\ \ 0.0]$$
$$error \neq none$$

W^T	1.5	0.5	0.0

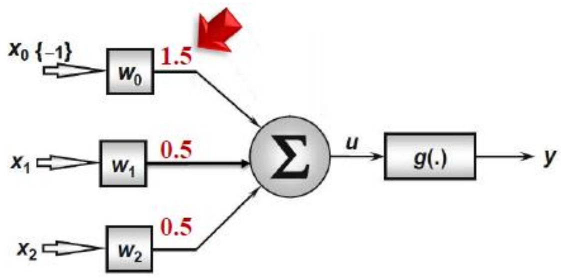


$$W^T x^{(4)} = u = (1.5)(-1) + (0.5)(1) + (0.0)(1) = -1 \Rightarrow y = g(u) = 0 \Rightarrow d^{(4)} \neq y$$
$$W = W + \eta\left(d^{(4)} - y\right)x^{(4)}$$
$$W_0 = 1.5 + (0.5)(1-0)(-1) = 1.0$$
$$W_1 = 0.5 + (0.5)(1-0)(1) = 1.0$$
$$W_2 = 0.0 + (0.5)(1-0)(1) = 0.5$$
$$\Rightarrow W^T = [1.0\ \ 1.0\ \ 0.5]$$
$$error \neq none$$

3. Epoch 3:

W^T	1.5	0.5	0.5

$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$
−1	−1	−1	−1
0	0	1	1
0	1	0	1

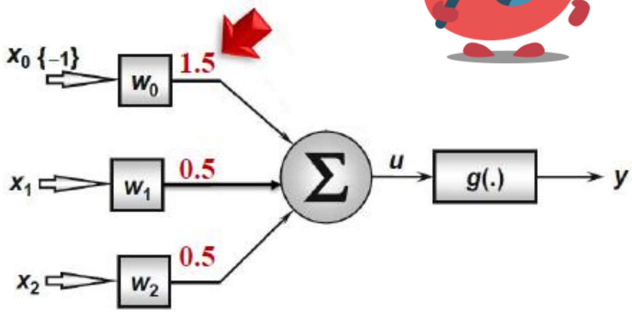


Error = none

$$W^T x^{(1)} = u = (1.0)(-1) + (1.0)(0) + (0.5)(0) = -1.0 \Rightarrow y = g(u) = 0 \Rightarrow d^{(1)} = y$$
$$W^T x^{(2)} = u = (1.0)(-1) + (1.0)(0) + (0.5)(1) = -0.5 \Rightarrow y = g(u) = 0 \Rightarrow d^{(2)} = y$$
$$W^T x^{(3)} = u = (1.0)(-1) + (1.0)(1) + (0.5)(0) = 0 \Rightarrow y = g(u) = 1 \Rightarrow d^{(3)} \neq y$$
$$W = W + \eta\left(d^{(3)} - y\right)x^{(3)}$$
$$W_0 = 1.0 + (0.5)(0-1)(-1) = 1.5$$
$$W_1 = 1.0 + (0.5)(0-1)(1) = 0.5$$
$$W_2 = 0.5 + (0.5)(0-1)(0) = 0.5$$

$$\Rightarrow W^T = [1.5\ \ 0.5\ \ 0.5]$$
$$error\ \neq none$$

W^T	1.5	0.5	0.5



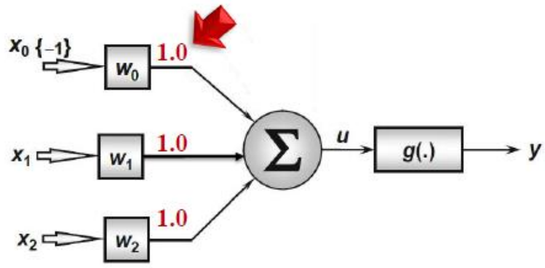
$$W^T x^{(4)} = u = (1.5)(-1) + (0.5)(1) + (0.5)(1) = -0.5 \Rightarrow y = g(u) = 0 \Rightarrow d^{(4)} \neq y$$
$$W = W + \eta\left(d^{(4)} - y\right)x^{(4)}$$
$$W_0 = 1.5 + (0.5)(1-0)(-1) = 1.0$$
$$W_1 = 0.5 + (0.5)(1-0)(1) = 1.0$$
$$W_2 = 0.5 + (0.5)(1-0)(0) = 1.0$$
$$\Rightarrow W^T = [1.0\ \ 1.0\ \ 1.0]$$
$$error\ \neq none$$

4. Epoch 4:

W^T	1.0	1.0	1.0

Error = none

$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$
−1	−1	−1	−1
0	0	1	1
0	1	0	1



$$W^T x^{(1)} = u = (1.0)(-1) + (1.0)(0) + (1.0)(0) = -1.0 \Rightarrow y = g(u) = 0 \Rightarrow d^{(1)} = y$$
$$W^T x^{(2)} = u = (1.0)(-1) + (1.0)(0) + (1.0)(1) = 0 \Rightarrow y = g(u) = 1 \Rightarrow d^{(2)} \neq y$$
$$W = W + \eta\left(d^{(2)} - y\right)x^{(2)}$$
$$W_0 = 1.0 + (0.5)(0-1)(-1) = 1.5$$
$$W_1 = 1.0 + (0.5)(0-1)(0) = 1.0$$
$$W_2 = 1.0 + (0.5)(0-1)(1) = 0.5$$
$$\Rightarrow W^T = [1.5\ \ 1.0\ \ 0.5]$$
$$error\ \neq none$$

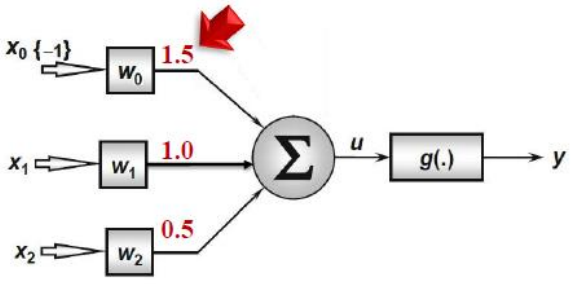
$$W^T x^{(3)} = u = (1.5)(-1) + (1.0)(1) + (0.5)(0) = -0.5 \Rightarrow y = g(u) = 0 \Rightarrow d^{(3)} = y$$
$$W^T x^{(4)} = u = (1.5)(-1) + (1.0)(1) + (0.5)(1) = 0.0 \Rightarrow y = g(u) = 1 \Rightarrow d^{(4)} = y$$

5. Epoch 5:

W^T	1.5	1.0	0.5

$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$
−1	−1	−1	−1
0	0	1	1
0	1	0	1

Error = none

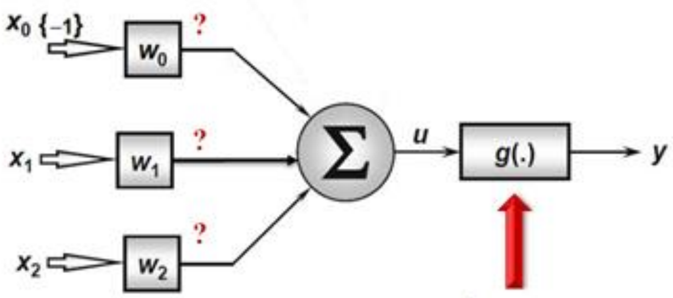


$$W^T x^{(1)} = u = (1.5)(-1) + (1.0)(0) + (0.5)(0) = -1.5 \Rightarrow y = g(u) = 0 \Rightarrow d^{(1)} = y$$
$$W^T x^{(2)} = u = (1.5)(-1) + (1.0)(0) + (0.5)(1) = -1.0 \Rightarrow y = g(u) = 0 \Rightarrow d^{(2)} = y$$
$$W^T x^{(3)} = u = (1.5)(-1) + (1.0)(1) + (0.5)(0) = -0.5 \Rightarrow y = g(u) = 0 \Rightarrow d^{(3)} = y$$
$$W^T x^{(4)} = u = (1.5)(-1) + (1.0)(1) + (0.5)(1) = 0.0 \Rightarrow y = g(u) = 1 \Rightarrow d^{(4)} = y$$

End training

Number of training iterations $= 5 \Rightarrow final\ W = [1.5\ , 1,\ 0.5]$

Boolean OR:



$$g(u) = \begin{cases} 1, & \text{if } u \geq 0 \\ 0, & \text{if } u < 0 \end{cases}$$

OR Gate

x_0	x_1	x_2	$output$
−1	0	0	0
−1	0	1	1
−1	+1	0	1
−1	+1	1	1

Initialization $W = [0, 0, 0]$, Learning rate $= 0.5$

1. Epoch 1:

W_0	W_1	W_2	x_0	x_1	x_2	u	y	d	$d-y$		W_0	W_1	W_2
0	0	0	−1	0	0	0	1	0	−1	⇒	0.5	0.0	0.0
0.5	0	0	−1	0	1	−0.5	0	1	1	⇒	0.0	0.0	0.5
0.0	0.0	0.5	−1	1	0	0.0	1	1	0				
0.0	0.0	0.5	−1	1	1	0.5	1	1	0				

2. Epoch 2:

W_0	W_1	W_2	x_0	x_1	x_2	u	y	d	$d-y$		W_0	W_1	W_2
0	0.0	0.5	−1	0	0	0	1	0	−1	⇒	0.5	0.0	0.5
0.5	0	0.5	−1	0	1	0	1	1	0				
0.5	0.0	0.5	−1	1	0	−0.5	0	1	1	⇒	0.0	0.5	0.5
0.0	0.5	0.5	−1	1	1	1.0	1	1	0				

3. Epoch 3:

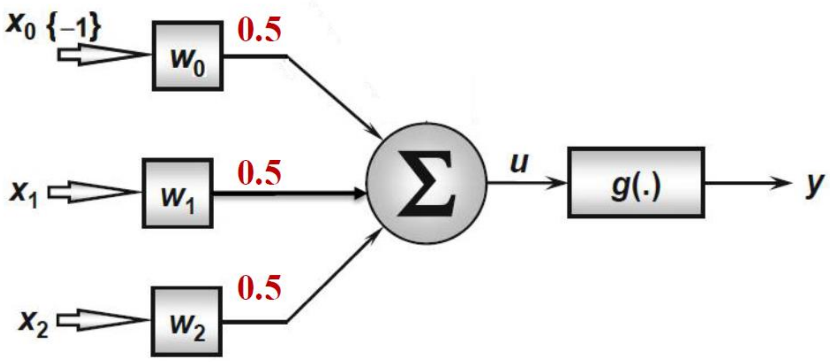
W_0	W_1	W_2	x_0	x_1	x_2	u	y	d	$d-y$		W_0	W_1	W_2
0	0.5	0.5	−1	0	0	0	1	0	−1	⇒	0.5	0.5	0.5
0.5	0.5	0.5	−1	0	1	0	1	1	0				
0.5	0.5	0.5	−1	1	0	0	1	1	0				
0.5	0.5	0.5	−1	1	1	0.5	1	1	0				

4. Epoch 4:

W_0	W_1	W_2	x_0	x_1	x_2	u	y	d	$d-y$		W_0	W_1	W_2
0.5	0.5	0.5	−1	0	0	−0.5	0	0	0				
0.5	0.5	0.5	−1	0	1	0	1	1	0				
0.5	0.5	0.5	−1	1	0	0	1	1	0				
0.5	0.5	0.5	−1	1	1	0	1	1	0				

End training

Number of training iterations $= 4 \Rightarrow W = [0.5 \;\; 0.5 \;\; 0.5]$



The ADALINE (Adaptive Linear Neuron)

- عبارة عن شبكة عصبونية أحادية الطبقة بمداخل متعددة وتعطي ناتجاً، وهي تشبه **Perceptron** إلى حد كبير حيث يعتبر **ADALINE** تطوراً له، فكلاهما أحاديات الطبقة ويعتبران مصنفات للتصنيف الثنائي حيث يؤديان مهمة الفصل الخطي.
- له تطبيقات مختلفة خاصة بمجال معالجة الإشارة التماثلية.
- يعد **ADALINE** الشبكة العصبونية الأولى التي تطبقت بتطبيقات صناعية.

▪ إضاءة: تم تقديم ADALINE في عام 1959 (بعد طرح perceptron بفترة قصيرة) من قِبل Bernard Widrow و Ted Hoff (أحد مخترعي المعالجات الدقيقة) في جامعة ستانفورد.

أجزاء (خصائص) الشبكة:

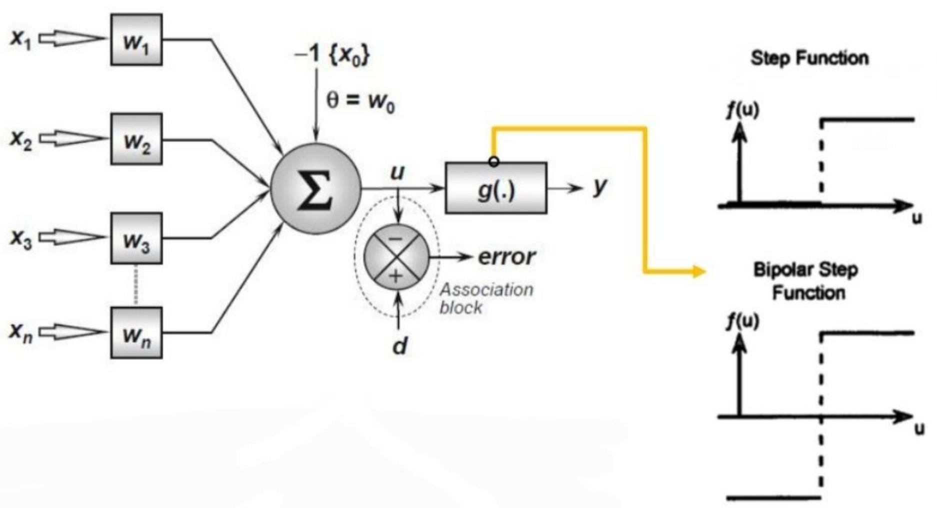
تشبه perceptron في توصيفه، فلها:

1. شعاع الدخل: وهي قيم حقيقية أو ثنائية تؤخذ من المحيط الخارجي.
2. شعاع الأوزان: حيث يُرفق لكل دخل وزن خاص به، وهي قيم حقيقية تهيًّأ بقيم عشوائية.
3. العتبة (الانزياح): ممكن أن تكون سالبة أو موجبة.
4. قيمة الخرج: وهي ثنائية 0 أو 1.
5. تابع التفعيل: هو هنا Step or bipolar step function (والذي يفرق في اختيار أحد التابعين، في AND مثلاً هو:

في Step Function: تكون $0, 1$

في Bipolar Step Function تكون $-1, +1$

6. يُطبق في supervised learning (أي أن الدخل والخرج معروفين).
7. قاعدة التعلم هي دلتا (سَيَرِد شرحها لاحقاً).



ولما كان $(-\theta)$ هو عبارة عن $W_0.x_0$ حيث:

$W_0 = \theta, x_0 = -1$ (وزن الانزياح)

$$u = \sum_{i=1}^{n} W_i.x_i - \theta$$

أمكننا كتابة العبارة السابقة بالشكل:

$$u = \sum_{i=0}^{n} W_i.x_i$$

$$y = g(u)$$

- ال error الذي نبحث عنه والذي يتحكم بدروان خوارزمية ADALINE، حيث تتوقف عندما تصبح أقل من حد معين (وليس بالضرورة أن يصل للصفر)، بينما بقاعدة perceptron كان شرط توقف الخوارزمية هو وصول error للصفر حصراً.

أصبح لدينا الآن ثلاث قواعد للتدريب:

1. قاعدة Hebbian: $W = W + x^{(k)}.y$
2. قاعدة perceptron: $W = W + \eta(d^{(k)} - y).x^{(k)}$
3. قاعدة Delta: $W = W + \eta(d^{(k)} - u).x^{(k)}$

ويجب الانتباه إلى حساب الخطأ، حيث:

- الخطأ في perceptron: $d^{(k)} - y$ (أي ما بعد تابع التفعيل)
- أما في Delta: $d^{(k)} - u$ (أي ما قبل تابع التفعيل)

نتابع في المحاضرة القادمة عن كيفية تدريب ADALINE

The end