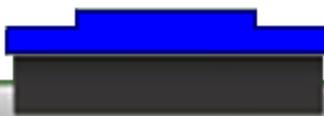


مبانی هوش محاسباتی

فصل یک - شبکه‌های عصبی مصنوعی تک‌لایه



فهرست مطالب

- نرون بیولوژیکی و مدل ریاضی آن
- مفهوم دسته‌بندی
- سلول مک‌کلوج-پیتس
- مفهوم آموزش در شبکه عصبی
- شبکه هب
- شبکه پرسپترون
- شبکه آدالاین



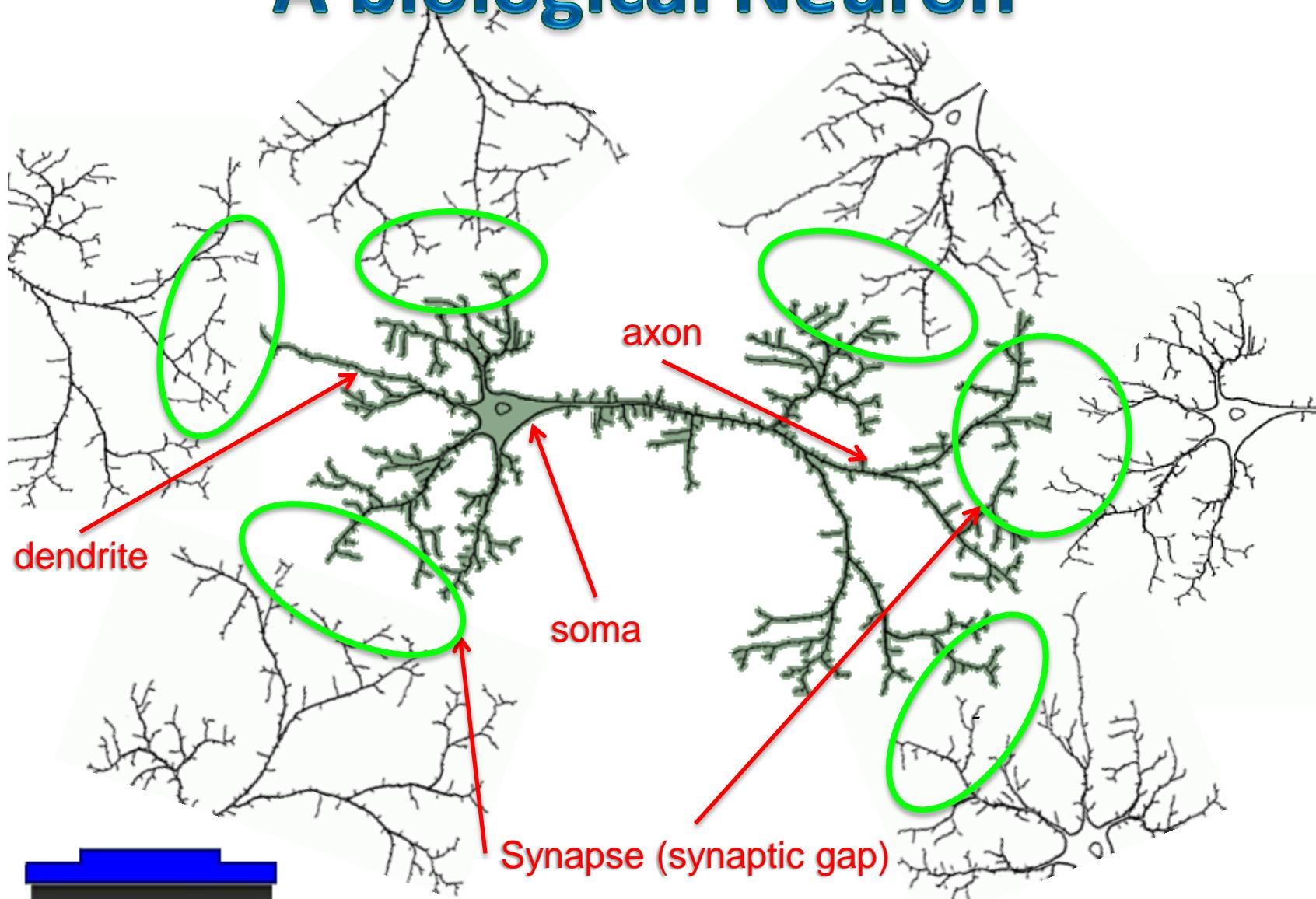
مبانی هوش محاسباتی

فصل یک- شبکه‌های عصبی مصنوعی تک‌لایه

نرون بیولوژیکی و مدل ریاضی آن

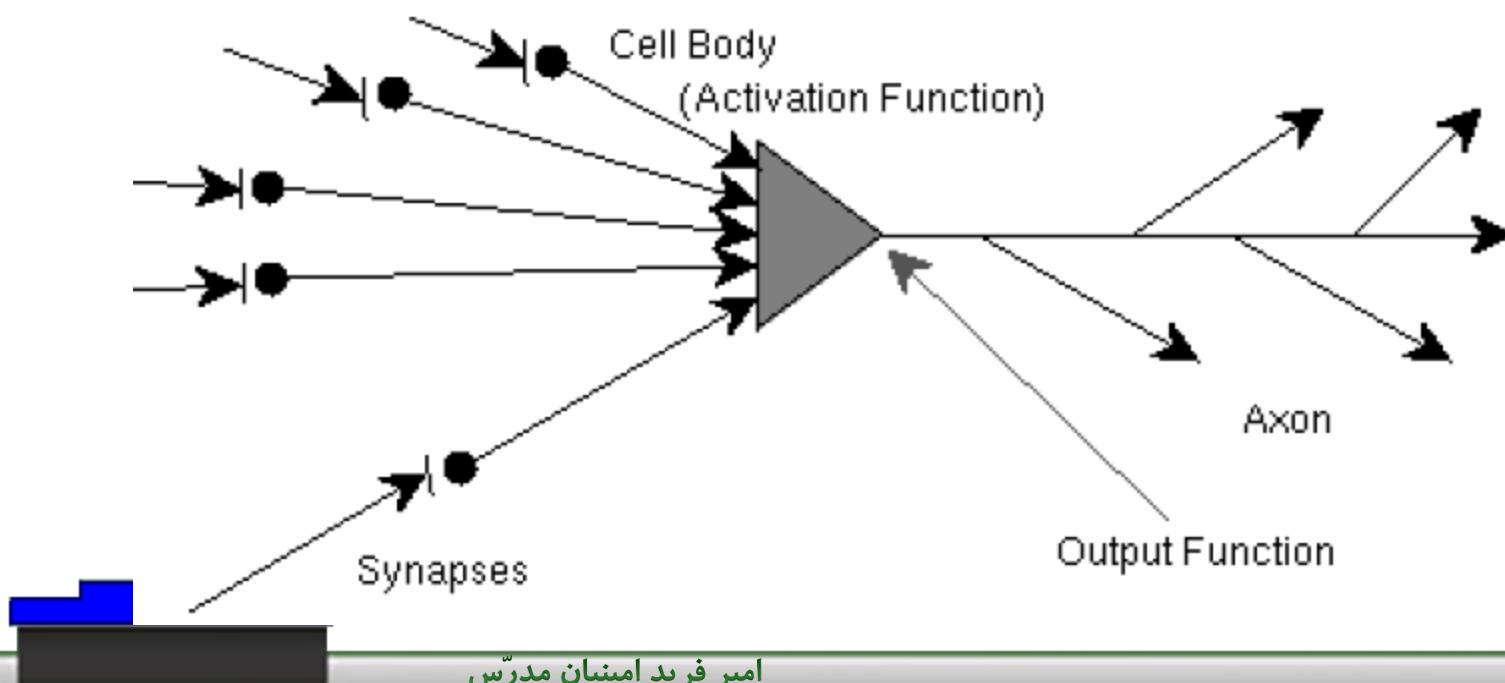


A biological Neuron



مدل سازی یک نرون بیولوژیکی

- مدل ریاضی یک نرون بیولوژیکی
 - توسط McCulloch & Pitts (۱۹۴۳)
 - یون‌های ورودی توسط سیناپس‌ها دریافت و جمع می‌شوند.
 - خروج یونها از آکسون دو وضعیتی است. اگر میزان تمام یون‌های ورودی از حد آستانه مشخصی فراتر رود، آکسون یونها را به خارج منتقل می‌کند.

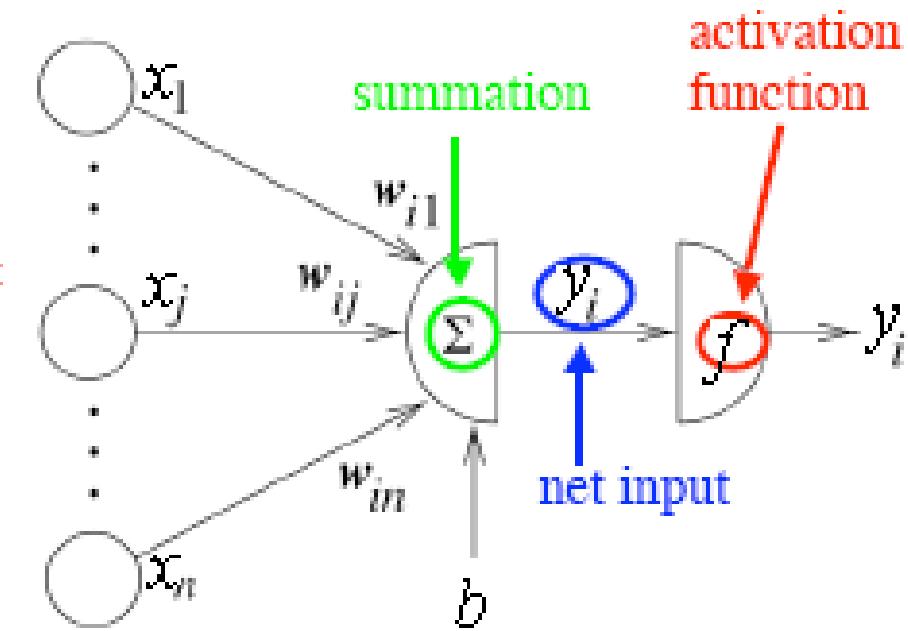
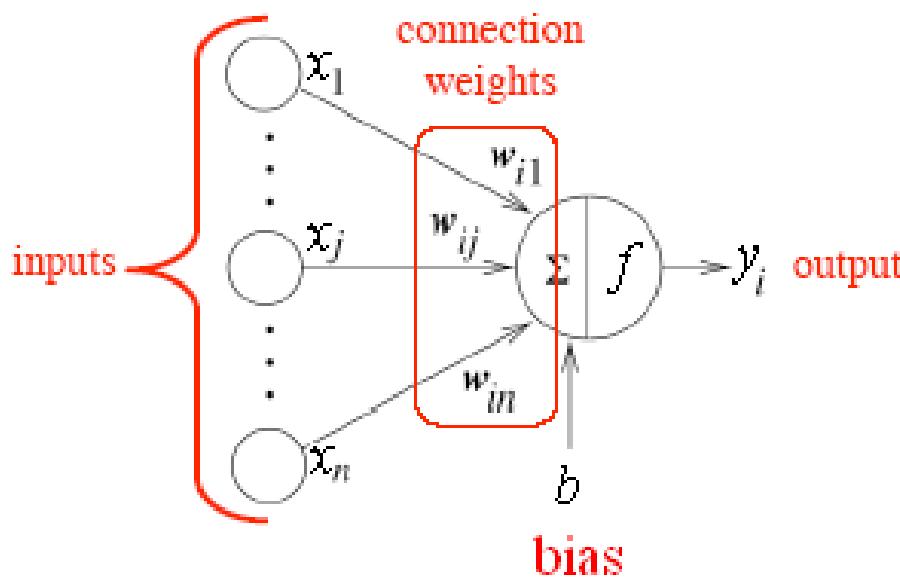


مدل سازی یک نرون بیولوژیکی

- مدل ریاضی نرون

- جمع وزن دار ورودی ها

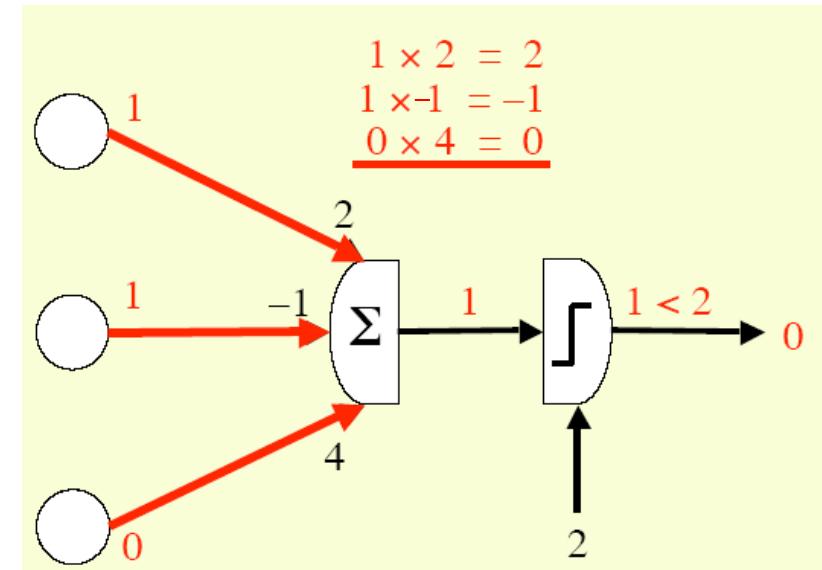
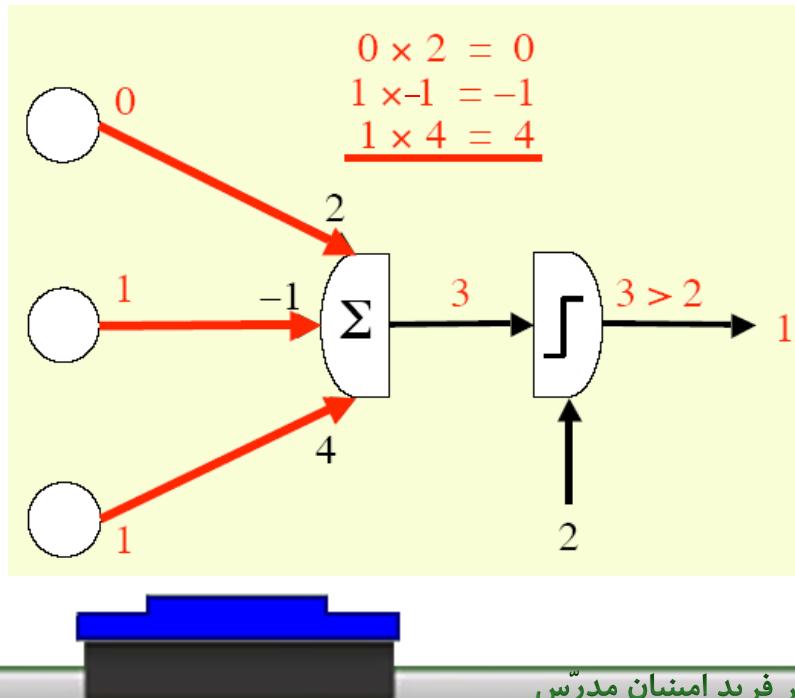
- تعریف تابع فعالیت روی خروجی





مثال عملکرد نرون

- یک مثال ساده از عملکرد نرون ریاضی مک‌کلوج-پیتس
 - تابع فعالیت: تابع پلّه باینری (آستانه‌گذاری)
 - مقدار آستانه: ۲





مبانی هوش محاسباتی

فصل یک- شبکه‌های عصبی مصنوعی تک‌لایه

سلول عصبی مک‌کلوج-پیتس و مفهوم دسته بندی توسط یک سلول عصبی

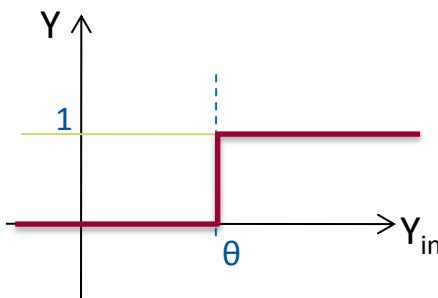
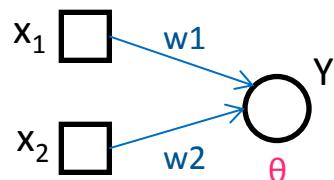


شبکه عصبی McCulloch-Pitts

- ابداع شبکه‌های عصبی - سال ۱۹۴۳
 - استفاده از مدل‌های شبیه سیستم‌های متفکر (مغز انسان) به منظور ایجاد ماشین‌های با قدرت فکر، یادگیری، بـهـیـادـآـورـیـ، استنتاج، و نظیر آن.
- ایده اولیه: به کارگیری مدل ریاضی نرون به عنوان یک دروازه (gate) منطقی ساده.
 - استفاده از وزنها به عنوان پارامترهای سیستم
 - تعیین وزنها به قسمی که مدل ریاضی نرون، خواسته مورد نظر را برآورده نماید.
- نکته بسیار مهم: وزنهای شبکه توسط طراح تعیین شده و همیشه ثابت هستند.



تحلیل فعالیت سلول عصبی مک‌کلوج-پیتس



$$y_{in} = \sum_i w_i x_i = w_1 x_1 + w_2 x_2$$

$$Y = \begin{cases} 0 & y_{in} < \theta \\ 1 & y_{in} \geq \theta \end{cases}$$

- فرض اولیه: یک سلول با دو ورودی، تابع فعالیت پله‌ای با مقدار آستانه معلوم

نقطه بحرانی:

$$y_{in} = \theta$$

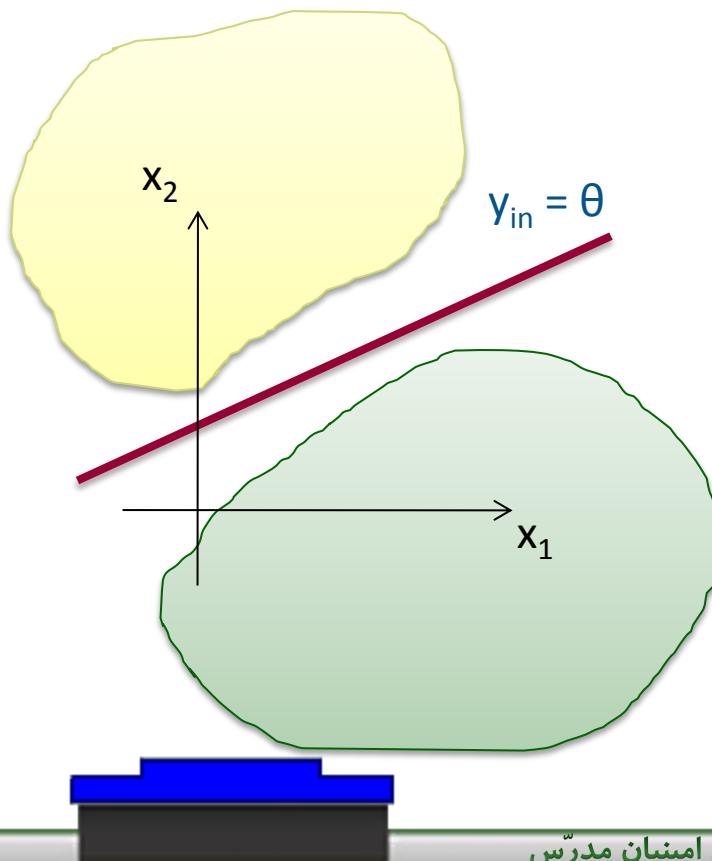
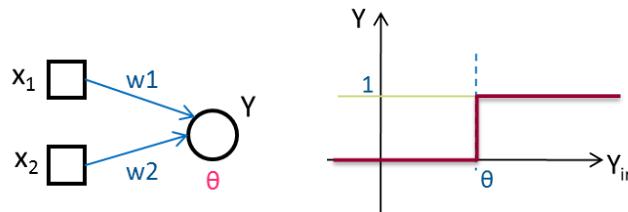
$$w_1 x_1 + w_2 x_2 = \theta$$

- معادله خط در فضای $x_1 x_2$!
- شیب و عرض از مبدأ:

$$m = -\frac{w_1}{w_2}, b = \frac{\theta}{w_2}$$

- w_1 و w_2 و θ پارامترهای مدل و معلوم هستند.
- بنابراین خط در فضای دوبعدی $x_1 x_2$ قابل ترسیم است.

سلول عصبی به عنوان جداکننده خطی عمل می کند!



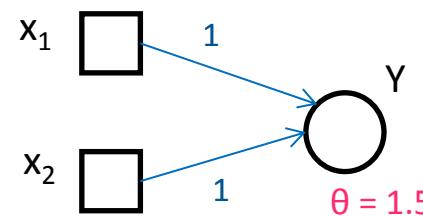
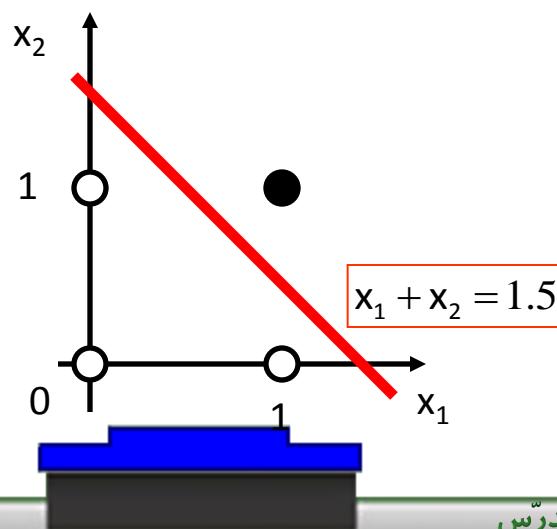
- نقطه بحرانی تابع فعالیت سلول عصبی، در فضای دوبُعدی به شکل یک خط قابل ترسیم است.
 - ورودیهای سلول، مختصات نقطه ورودی را به سلول داده و سلول بسته به اینکه آن نقطه در کدام قسمت صفحه قرار می‌گیرد، خروجی مناسب (فعال/غیرفعال) را تولید می‌کند.
 - سلول عصبی به عنوان جداکننده خطی (Linear Discriminator)
 - یعنی قابلیت جداسازی الگوهایی که به صورت خطی، تفکیکپذیر هستند را دارد.
- تعداد ورودیهای سلول چه نقشی ایفا می‌کند؟
 - هر چه تعداد ورودیها بیشتر باشد، ابعاد فضا بیشتر می‌شود.
 - یعنی نقطه بحرانی تبدیل به یک آبرصفحه $n-1$ -بعدی در فضای n بعدی می‌شود.

مثالی از مک‌کلوچ-پیتس

x_1	x_2	$x_1 \wedge x_2$
1	1	1
1	0	0
0	1	0
0	0	0

- شبیه‌سازی AND منطقی
 - جدول ارزش

– مدل طراحی شده مک‌کلوچ-پیتس



$$y_{in} = \sum_i w_i x_i = x_1 + x_2$$

$$Y = \begin{cases} 0 & y_{in} < 1.5 \\ 1 & y_{in} \geq 1.5 \end{cases}$$

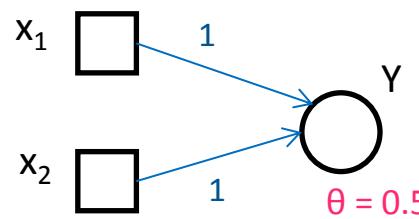
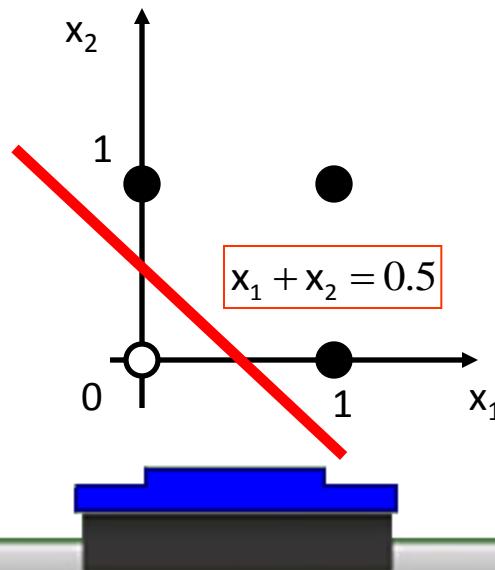
– معادله مرز تصمیم و ترسیم آن

مثالی از مک‌کلوچ-پیتس

x_1	x_2	$x_1 \vee x_2$
1	1	1
1	0	1
0	1	1
0	0	0

- شبیه‌سازی OR منطقی
 - جدول ارزش

– مدل طراحی شده مک‌کلوچ-پیتس

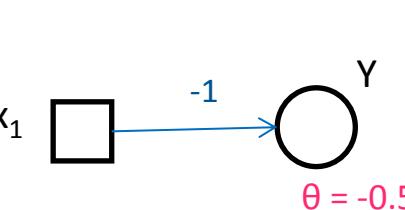
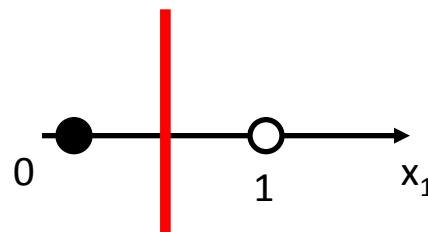


$$y_{in} = \sum_i w_i x_i = x_1 + x_2$$

$$Y = \begin{cases} 0 & y_{in} < 0.5 \\ 1 & y_{in} \geq 0.5 \end{cases}$$

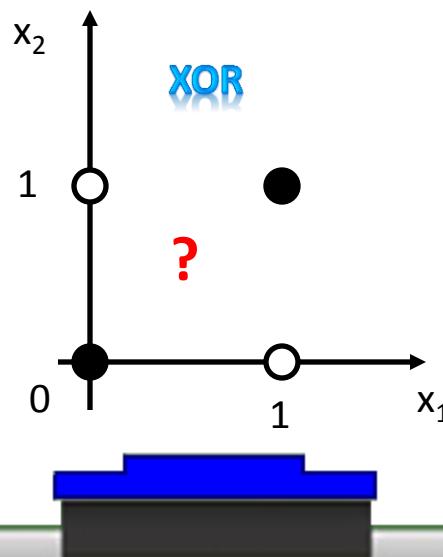
– معادله مرز تصمیم و ترسیم آن

مثالی از مک‌کلوچ-پیتس



$$y_{in} = \sum_i w_i x_i = -x_1$$

$$Y = \begin{cases} 0 & y_{in} < -0.5 \\ 1 & y_{in} \geq -0.5 \end{cases}$$



• آیا می‌توان برای XOR نیز مدل مک‌کلوچ-پیتس یافت؟

- چون نمونه‌های دو گروه قابل تفکیک توسط یک خط راست نیستند، توسط یک سلول مک‌کلوچ-پیتس به تنها‌یی، نمی‌توانیم آنها را تفکیک کنیم.

جمع‌بندی شبکه مک‌کلوج-پیتس

- شروع تاریخچه شبکه‌های عصبی
- استفاده از سلولهای عصبی به عنوان عملگرهای منطقی
 - قابلیت شبیه‌سازی عملگرهای پایه منطقی
 - قابلیت شبیه‌سازی ساده و آسان عملگرهایی که معادل منطقی ندارند.
 - توانایی شبیه‌سازی مدارهای منطقی ترکیبی و ترتیبی.
- اشکال عمدۀ: ثابت بودن وزنها و دیگر پارامترهای مدل
 - همیشه برای ارایه یک شبکه عصبی، باید مساله را به صورت دستی (یا با کمک روش دیگر) حل کرده و سپس نتایج را در قالب وزن به شبکه بدهیم.
 - چون عملی و مفید نیست، هیچکس تمایلی به انجام چنین کاری ندارد!
- چگونه می‌توانیم وزن‌های شبکه را به صورت خودکار محاسبه کنیم؟
 - مفهوم «آموزش یا یادگیری شبکه عصبی»



مبانی هوش محاسباتی

فصل یک- شبکه‌های عصبی مصنوعی تک‌لایه

مفهوم آموزش در شبکه عصبی مصنوعی و شبکه عصبی هپ



آموزش شبکه‌های عصبی

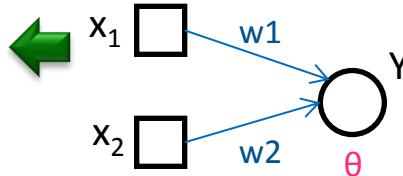
- شبکه McCulloch-Pitts، شبکه‌ای با وزنهای ثابت و قادر به یادگیری است!
- انتخاب دستی وزنهای شبکه‌های عصبی، طاقت‌فرسا، غیربهینه و گاهی غیرممکن است.
- چگونه می‌توان با داشتن مجموعه ورودیها و خروجیها، وزنهای محاسبه نمود؟
- آموزش (Training) یا یادگیری (Learning)، فرآیند محاسبه خودکار وزنهای شبکه است، به گونه‌ای که شبکه با وزنهای محاسبه شده، بتواند در قبال ورودی، خروجی مناسب و درست را تولید نماید.



مفهوم مقدار اولیه (bias)

$$y_{in} = \sum_i w_i x_i = w_1 x_1 + w_2 x_2$$

$$Y = \begin{cases} 0 & y_{in} = w_1 x_1 + w_2 x_2 < \theta \\ 1 & y_{in} = w_1 x_1 + w_2 x_2 \geq \theta \end{cases}$$



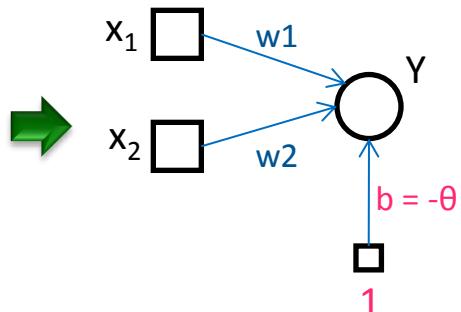
$$Y = \begin{cases} 0 & w_1 x_1 + w_2 x_2 - \theta < 0 \\ 1 & w_1 x_1 + w_2 x_2 - \theta \geq 0 \end{cases}$$

$$y_{in} = w_1 x_1 + w_2 x_2 - \theta$$



$$y_{in} = w_1 x_1 + w_2 x_2 + b$$

$$b = -\theta$$



- مقدار آستانه (θ)

- تعبیر بیولوژیکی:

- ویژگی هسته سلول.

- در طول زمان تغییر نمی کند.

- در فرآیند آموزش، مایلیم بتوانیم این پارامتر را نیز به گونه ای تغییر دهیم که شبکه بتواند الگوها را یاد بگیرد.

- با کمک ریاضیات، و دستکاری معادلات، میتوانیم مفهوم دیگری به نام بایاس ایجاد کنیم که قابل تغییر است!

- بایاس (مقدار اولیه، فعالیت قبلی)

- وزن یک ورودی اضافی برای سلول، که همیشه «فعال» است.

شبکه هب (Hebb Net)

- اولین شبکه عصبی با مفهوم یادگیری
 - توسط روانشناس Donald Hebb به سال ۱۹۴۹.
 - قانون (ایده) آموزش هب (**Hebb Rule**): اگر ورودی و خروجی یک سلول همزمان فعال باشند، وزن بین آنها باید تقویت شود.
 - اگر همزمان غیرفعال باشند؟ اگر یکی فعال و دیگری غیرفعال باشد؟
 - قانون تعمیم یافته هب: اگر ورودی و خروجی همزمان فعال یا غیرفعال باشند، وزن بین آنها تقویت و در غیر اینصورت، تضعیف شود.



الگوریتم آموزش هب

Step 0.

Initialize all weights:

$$w_i = 0 \quad (i = 1 \text{ to } n).$$

Step 1.

For each input training vector and target output pair, $s : t$, do steps 2–4.

Step 2. Set activations for input units:

$$x_i = s_i \quad (i = 1 \text{ to } n).$$

Step 3. Set activation for output unit:

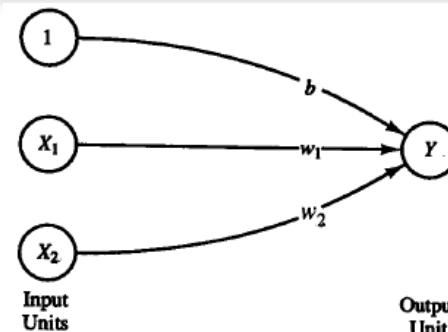
$$y = t.$$

Step 4. Adjust the weights for

$$w_i(\text{new}) = w_i(\text{old}) + x_i \cdot y \quad (i = 1 \text{ to } n).$$

Adjust the bias:

$$b(\text{new}) = b(\text{old}) + y$$



مثال آموزش هب

x_1	x_2	$x_1 \wedge x_2$
1	1	1
1	0	0
0	1	0
0	0	0

- آموزش تابع بولی AND
 - ورودی و خروجی باینری
 - یادآوری:

$w_{i(\text{new})} = x_i \cdot y + w_{i(\text{old})}$: قانون آموزش هب:

- مراحل آموزش هر سطر جدول ارزش به عنوان یک الگو:

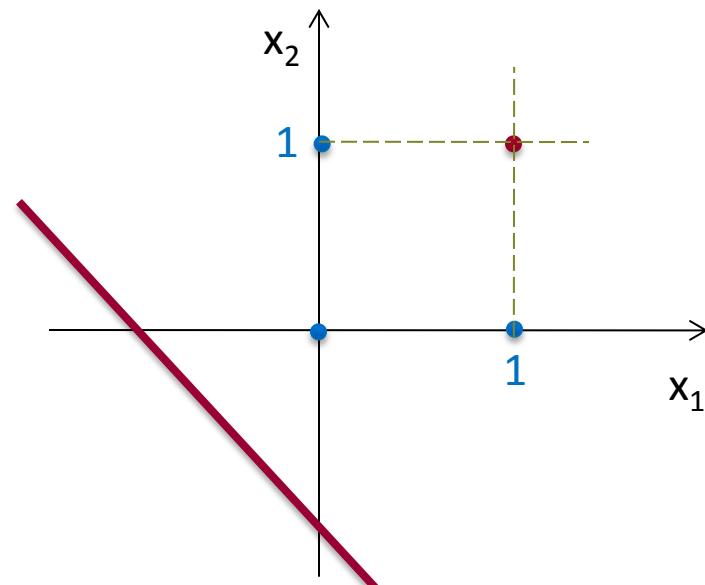
وزنهای اولیه	$w_1(\text{old})$	$w_2(\text{old})$	$b_{(\text{old})}$	x_1	x_2	'1'	y	$x_1 \cdot y$	$x_2 \cdot y$	'1'·y	$w_1(\text{new})$	$w_2(\text{new})$	$b_{(\text{new})}$
	0	0	0	1	1	1	1	1	1	1	1	1	1
	1	1	1	1	0	1	0	0	0	0	1	1	1
	1	1	1	0	1	1	0	0	0	0	1	1	1
	1	1	1	0	0	1	0	0	0	0	1	1	1

وزنهای نهایی

نتیجه آموزش

• آیا شبکه چیزی یاد گرفت؟

- بررسی عملکرد سلول با توجه به وزنهای به دست آمده از الگوریتم آموزش



به نظر می‌رسد اتفاق خوبی نیفتاده است!
چرا؟



- معادله مرز (خط) تصمیم:

$$x_1 + x_2 + 1 = 0$$



مثال آموزش هب

x_1	x_2	$x_1 \wedge x_2$
1	1	+1
1	0	-1
0	1	-1
0	0	-1

- آموزش مجدد تابع بولی AND
 - ورودی باینری؛ خروجی باپولار

وزنهای اولیه	w_1 (old)	w_2 (old)	b (old)	x_1	x_2	'1'	y	$x_1 \cdot y$	$x_2 \cdot y$	'1' $\cdot y$	w_1 (new)	w_2 (new)	b (new)
	0	0	0	1	1	1	+1	1	1	1	1	1	1
	1	1	1	1	0	1	-1	-1	0	-1	0	1	0
	0	1	0	0	1	1	-1	0	-1	-1	0	0	-1
	0	0	-1	0	0	1	-1	0	0	-1	0	0	-2

وزنهای نهایی

- وزنهای نهایی نشان‌دهنده هیچ معادله خطی نیست.

• باز هم یادگیری صورت نپذیرفته است. چرا؟



مثال آموزش هب

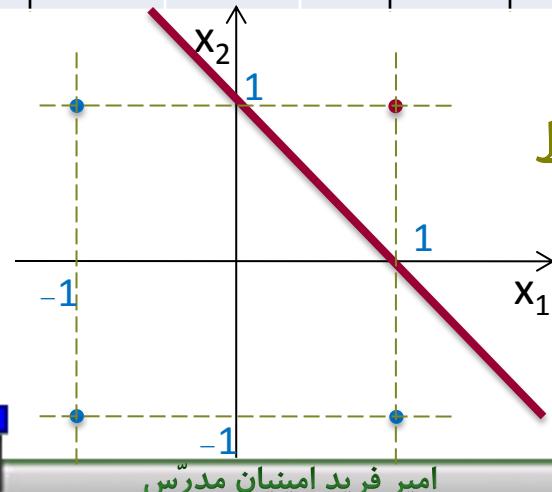
x_1	x_2	$x_1 \wedge x_2$
+1	+1	+1
+1	-1	-1
-1	+1	-1
-1	-1	-1

- آموزش مجدد تابع بولی AND
- ورودی و خروجی با پولار

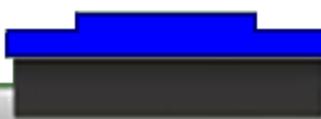
وزنهای اولیه

$w_1(\text{old})$	$w_2(\text{old})$	$b(\text{old})$	x_1	x_2	'1'	y	$x_1 \cdot y$	$x_2 \cdot y$	'1' $\cdot y$	$w_1(\text{new})$	$w_2(\text{new})$	$b(\text{new})$
0	0	0	+1	+1	1	+1	1	1	1	1	1	1
1	1	1	+1	-1	1	-1	-1	1	-1	0	2	0
0	2	0	-1	+1	1	-1	1	-1	-1	1	1	-1
1	1	-1	-1	-1	1	-1	1	1	-1	2	2	-2

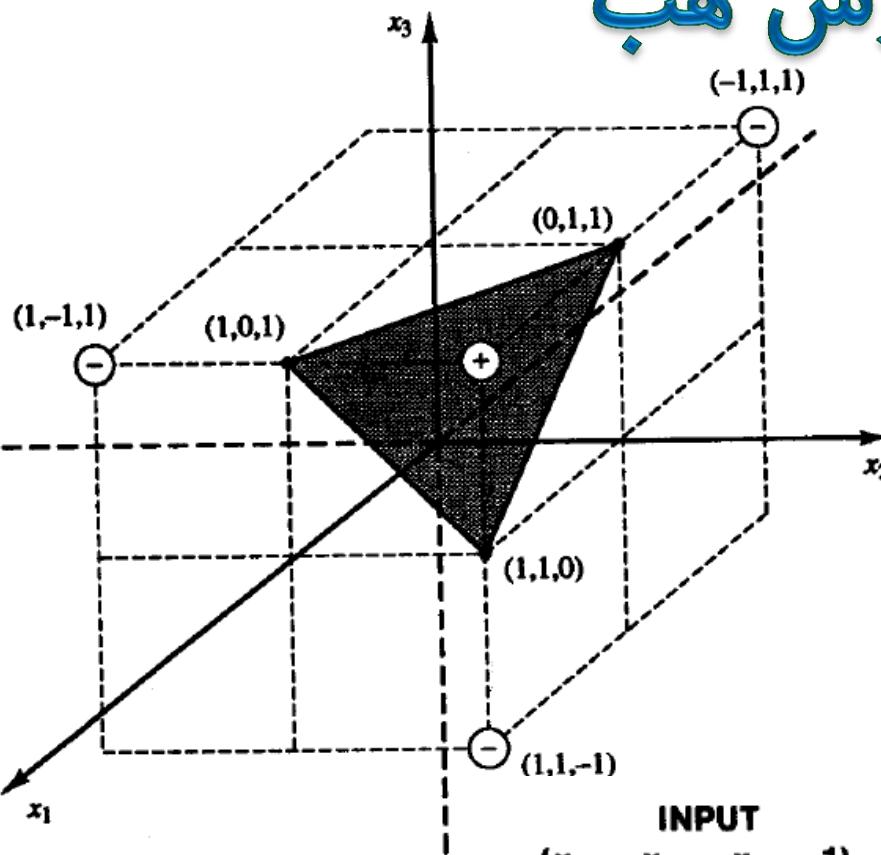
وزنهای نهایی



وزنهای نهایی نشان دهنده خط
 $2x_1 + 2x_2 - 2 = 0$ است.
 موفقیت!



مثال آموزش هب



• آیا شبکه هب با داده‌های باپولار، همیشه قادر به حل مساله هست؟

— در این مثال، با اینکه مساله جواب دارد، شبکه قادر به یادگیری آن نیست.

INPUT (x_1 x_2 x_3 1)	TARGET	WEIGHT CHANGE (Δw_1 Δw_2 Δw_3 Δb)	WEIGHT (w_1 w_2 w_3 b)
(-1 1 1 1)	1	(-1 1 1 1)	(0 0 0 0)
(-1 1 -1 1)	-1	(-1 -1 1 -1)	(0 0 2 0)
(-1 -1 1 1)	-1	(-1 1 -1 -1)	(-1 1 1 -1)
(-1 1 1 1)	-1	(1 -1 -1 -1)	(0 0 0 -2)

جمع‌بندی شبکه هب

- قانون هب: اگر ورودی و خروجی همزمان فعال یا غیرفعال باشند، وزنها تقویت و در غیراینصورت تضعیف می‌شوند.

— مزیت

- سادگی مفهوم؛ سادگی پیاده‌سازی.

— عیب

- مختص شبکه‌های تک‌لایه.
- داده‌های باینری، شبکه را دچار مشکل می‌کند.
 - «صفر»، نتیجه حاصلضرب و آموزش را از بین می‌برد.
- الگوهای آموزشی، باید تفکیک‌پذیر خطی (Linear Separable) باشند.
- حتی با داده‌های بایپولار و الگوهای تفکیک‌پذیر خطی، باز هم ممکن است شبکه به درستی آموزش نبیند.
 - هیچ روال بررسی-ارزیابی برای عملکرد شبکه وجود ندارد. از کجا معلوم روند آموزش به درستی طی می‌شود؟



مبانی هوش محاسباتی

فصل یک- شبکه‌های عصبی مصنوعی تک‌لایه

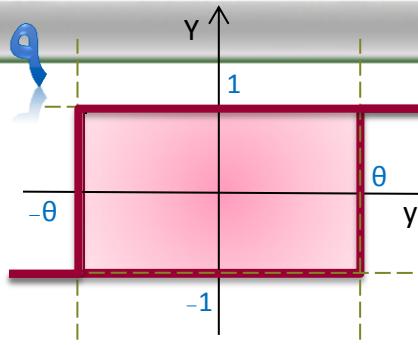
پادگیری تکراری در شبکه‌های عصبی و شبکه عصبی پرسپترون



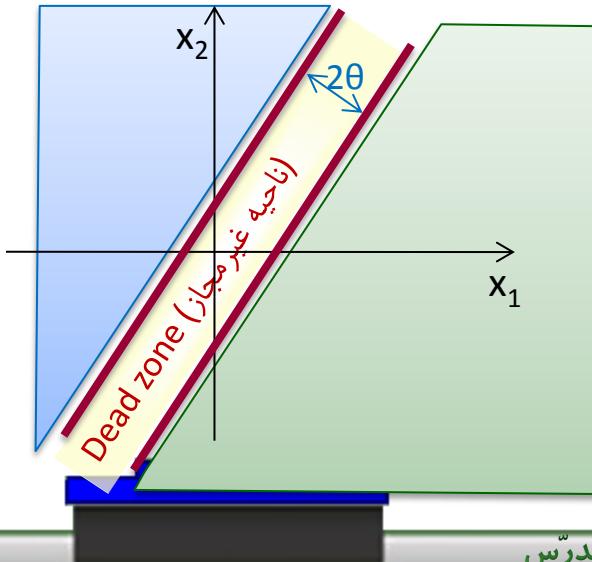
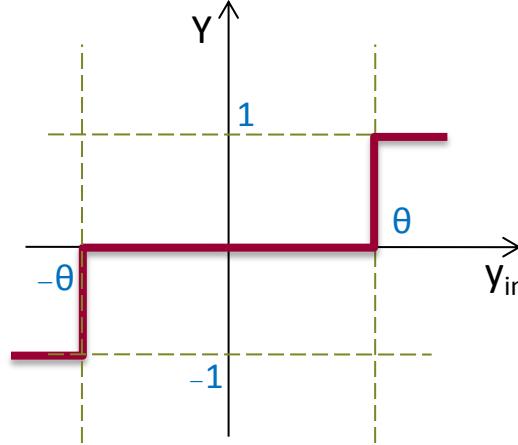
شبکه پرسپترون (Perceptron)

- معرفی شده توسط Rosenblatt به سال ۱۹۶۲.
 - ایده اولیه از عملکرد شبکیه چشم گرفته شده است.
- تفاوت‌ها با شبکه هب:
 - قانون آموزش پرسپترون؛ قوی‌تر از قانون هب است.
 - آموزش، پروسه‌ای تکراری (iterative) است و تا زمان وجود داشتن خطا، آموزش ادامه می‌یابد.
 - هر الگو ممکن است بارها و بارها به شبکه آموزش داده شود.
 - شبکه پرسپترون، تضمین می‌کند که در صورت وجود داشتن جواب، آن را پیدا کند.
 - تابع فعالیت سلول دو آستانه‌ای است.
- ویرایش‌های مختلفی از پرسپترون وجود دارد. ایده اولیه بررسی می‌شود.





تابع فعالیت پرسپترون



- تابع فعالیت دو آستانه‌ای، (هیسترزیس) (Hysteresis)

$$y = \begin{cases} 1 & \theta < y_{in} \\ 0 & -\theta \leq y_{in} \leq \theta \\ -1 & y_{in} < -\theta \end{cases}$$

- با فرض سلول با دو ورودی، داریم:

$$y = \begin{cases} 1 & \theta < w_1x_1 + w_2x_2 + b \\ 0 & -\theta \leq w_1x_1 + w_2x_2 + b \leq \theta \\ -1 & w_1x_1 + w_2x_2 + b < -\theta \end{cases}$$

دو خط بحرانی وجود دارد:

$$w_1x_1 + w_2x_2 + b = \theta$$

$$w_1x_1 + w_2x_2 + b = -\theta$$

$$w_1x_1 + w_2x_2 + (b - \theta) = 0$$

$$w_1x_1 + w_2x_2 + (b + \theta) = 0$$

دو خط موازی، با فاصله 2θ



Step 0.

Initialize weights and bias.

(For simplicity, set weights and bias to zero.)

Set learning rate α ($0 < \alpha \leq 1$).

(For simplicity, α can be set to 1.)

Step 1.

While stopping condition is false, do Steps 2–6.

Step 2. For each training pair $s:t$, do Steps 3–5

Step 3. Set activations of input units:

$$x_i = s_i$$

Step 4. Compute response of output unit:

$$y_in = b + \sum_i x_i w_i$$

$$\rightarrow y = \begin{cases} 1 & \text{if } y_in > \theta \\ 0 & \text{if } -\theta \leq y_in \leq \theta \\ -1 & \text{if } y_in < -\theta \end{cases}$$

تابع فعالیت

Step 5. Update weights and bias if an error occurred for this pattern.

If $y \neq t$,

$$w_i(\text{new}) = w_i(\text{old}) + \alpha t x_i$$

$$b(\text{new}) = b(\text{old}) + \alpha t$$

else

$$w_i(\text{new}) = w_i(\text{old})$$

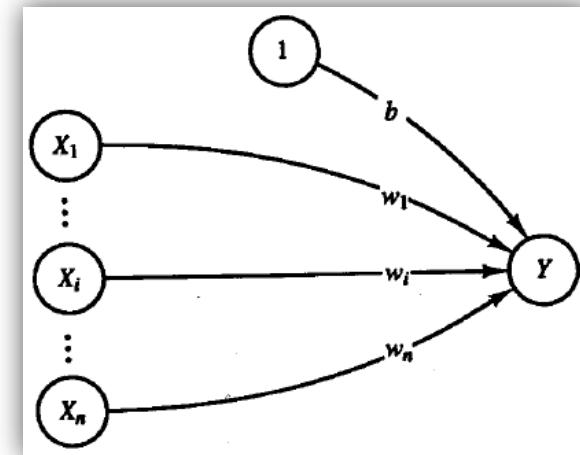
$$b(\text{new}) = b(\text{old})$$

Step 6.

Test stopping condition:

If no weights changed in Step 2, stop; else, continue.

الگوریتم آموزش پر سپترون



مثال آموزش پرسپترون

x_1	x_2	$x_1 \wedge x_2$
1	1	+1
1	0	-1
0	1	-1
0	0	-1

$$y = \begin{cases} 1 & \theta < w_1x_1 + w_2x_2 + b \\ 0 & -\theta \leq w_1x_1 + w_2x_2 + b \leq \theta \\ -1 & w_1x_1 + w_2x_2 + b < -\theta \end{cases}$$

و وزنهای اولیه همگی صفر.

- آموزش تابع بولی AND

– ورودی باینری و خروجی بایپolar.

- انتخاب پارامترها

$\alpha = 1$ و $\theta = 0/2$

- یادآوری

– قانون آموزش پرسپترون:

$$\Delta w_i = \alpha \cdot t \cdot x_i, \quad \Delta b = \alpha \cdot t \quad : (y \neq t)$$

$$\Delta w_i = 0, \quad \Delta b = 0 \quad : \text{در غیر اینصورت:}$$

- دوره ۱:

w ₁	w ₂	b	x ₁	x ₂	'1'	y	t	Δw ₁	Δw ₂	Δb
0	0	0	1	1	1	0	+1	1	1	1
1	1	1	1	0	1	1	-1	-1	0	-1
0	1	0	0	1	1	1	-1	0	-1	-1
0	0	-1	0	0	1	-1	-1	0	0	0

مثال آموزش پرسپترون

- چون در دوره یکم، حداقل یکبار تغییر وزن داشتیم، آموزش پایان نپذیرفته و وارد مرحله دوم می‌شویم.
- دوره ۲ :

وزنهای حاصل از دوره ۱

w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
0	0	-1	1	1	1	-1	+1	1	1	1
1	1	0	1	0	1	1	-1	-1	0	-1
0	1	-1	0	1	1	0	-1	0	-1	-1
0	0	-2	0	0	1	-1	-1	0	0	0

— دوره ۳ :

w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
0	0	-2	1	1	1	-1	+1	1	1	1
1	1	-1	1	0	1	0	-1	-1	0	-1
0	1	-2	0	1	1	-1	-1	0	0	0
0	1	-2	0	0	1	-1	-1	0	0	0

مثال آموزش پرسپترون

- آموزش آنقدر ادامه می‌یابد تا دیگر هیچ تغییر وزنی اتفاق نیافتد.
- دوره ۴:

w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
0	1	-2	1	1	1	-1	+1	1	1	1
1	2	-1	1	0	1	0	-1	-1	0	-1
0	2	-2	0	1	1	0	-1	0	-1	-1
0	1	-3	0	0	1	-1	-1	0	0	0

— دوره ۵:

w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
0	1	-3	1	1	1	-1	+1	1	1	1
1	2	-2	1	0	1	-1	-1	0	0	0
1	2	-2	0	1	1	0	-1	0	-1	-1
1	1	-3	0	0	1	-1	-1	0	0	0

مثال آموزش پرسپترون

دوره ۶ —

w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
1	1	-3	1	1	1	-1	+1	1	1	1
2	2	-2	1	0	1	0	-1	-1	0	-1
1	2	-3	0	1	1	-1	-1	0	0	0
1	2	-3	0	0	1	-1	-1	0	0	0

دوره ۷ —

w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
1	2	-3	1	1	1	0	+1	1	1	1
2	3	-2	1	0	1	0	-1	-1	0	-1
1	3	-3	0	1	1	0	-1	0	-1	-1
1	2	-4	0	0	1	-1	-1	0	0	0

مثال آموزش پرسپترون

دوره ۸ —

w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
1	2	-4	1	1	1	-1	+1	1	1	1
2	3	-3	1	0	1	-1	-1	0	0	0
2	3	-3	0	1	1	0	-1	0	-1	-1
2	2	-4	0	0	1	-1	-1	0	0	0

دوره ۹ —

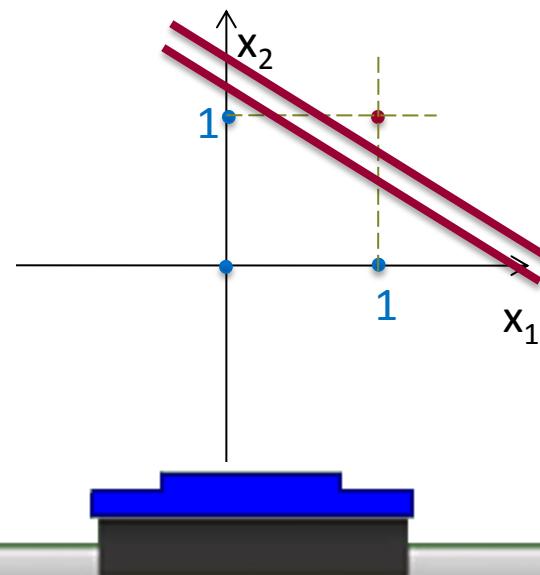
w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
2	2	-4	1	1	1	0	+1	1	1	1
3	3	-3	1	0	1	0	-1	-1	0	-1
2	3	-4	0	1	1	-1	-1	0	0	0
2	3	-4	0	0	1	-1	-1	0	0	0

مثال آموزش پرسپترون

دوره ۱۰ -

w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb
2	3	-4	1	1	1	+1	+1	0	0	0
2	3	-4	1	0	1	-1	-1	0	0	0
2	3	-4	0	1	1	-1	-1	0	0	0
2	3	-4	0	0	1	-1	-1	0	0	0

توجه: هیچ تغییر وزنی صورت نگرفته است! پس روال آموزش پایان می‌پذیرد.



- آموزش تا هنگامی که تغییر وزنی صورت نگیرد، ادامه می‌یابد.

- آیا شبکه به درستی آموزش دیده است؟

- معادله‌های خط، با توجه به وزنهای محاسبه شده:

$$2x_1 + 3x_2 + (-4 - 0.2) = 0$$

$$2x_1 + 3x_2 + (-4 + 0.2) = 0$$

مثال آموزش پرسپترون

x_1	x_2	$x_1 \wedge x_2$
+1	+1	+1
+1	-1	-1
-1	+1	-1
-1	-1	-1

- آموزش تابع بولی AND
 - ورودی و خروجی با ایپولار.
- انتخاب پارامترها

$\alpha = 1$ و $\theta = 0.5$ و وزنهای اولیه همگی صفر.

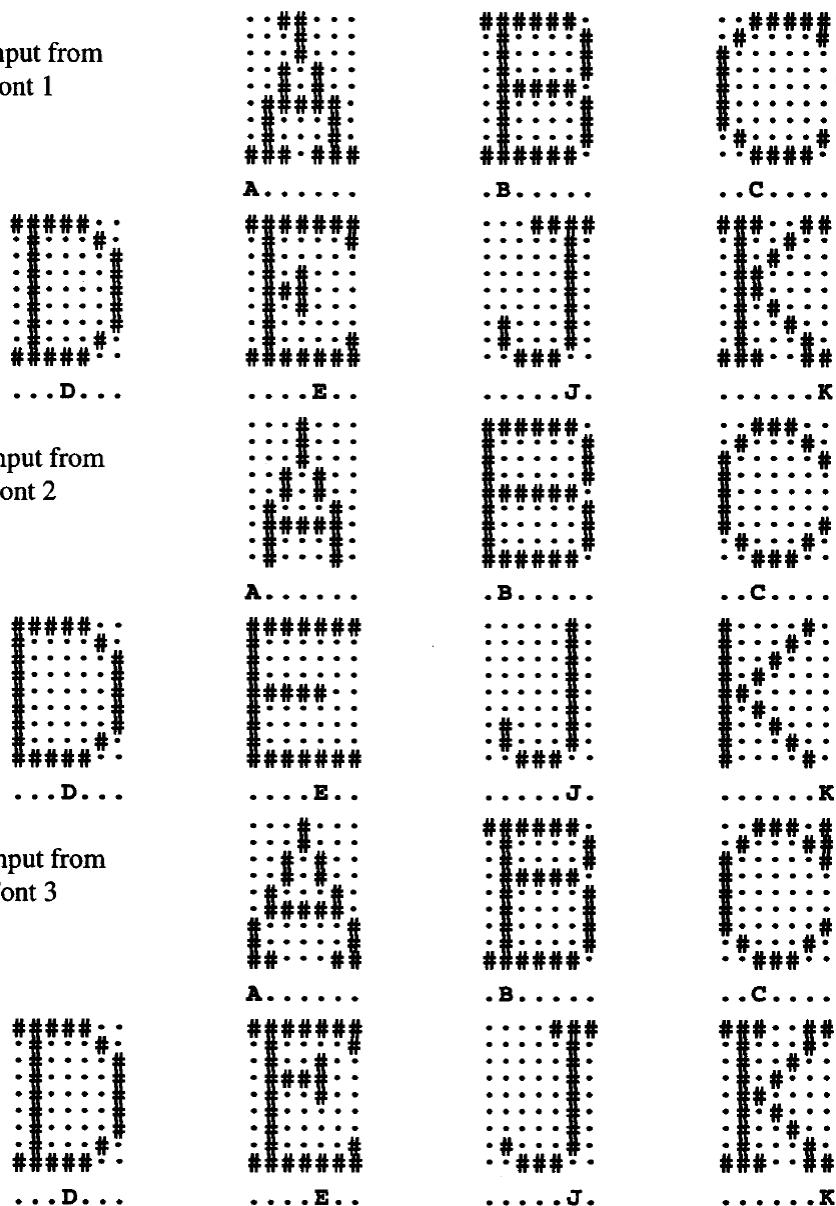
وزنهاي اولیه	w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb	دوره ۱ :
	0	0	0	1	1	1	0	+1	1	1	1	
	1	1	1	1	-1	1	1	-1	-1	1	-1	
	0	2	0	-1	1	1	1	-1	1	-1	-1	
	1	1	-1	-1	-1	1	-1	-1	0	0	0	

وزنهاي اولیه	w_1	w_2	b	x_1	x_2	'1'	y	t	Δw_1	Δw_2	Δb	دوره ۲ :
	1	1	-1	1	1	1	1	+1	0	0	0	
	1	1	-1	1	-1	1	-1	-1	0	0	0	
	1	1	-1	-1	1	1	-1	-1	0	0	0	
	1	1	-1	-1	-1	1	-1	-1	0	0	0	

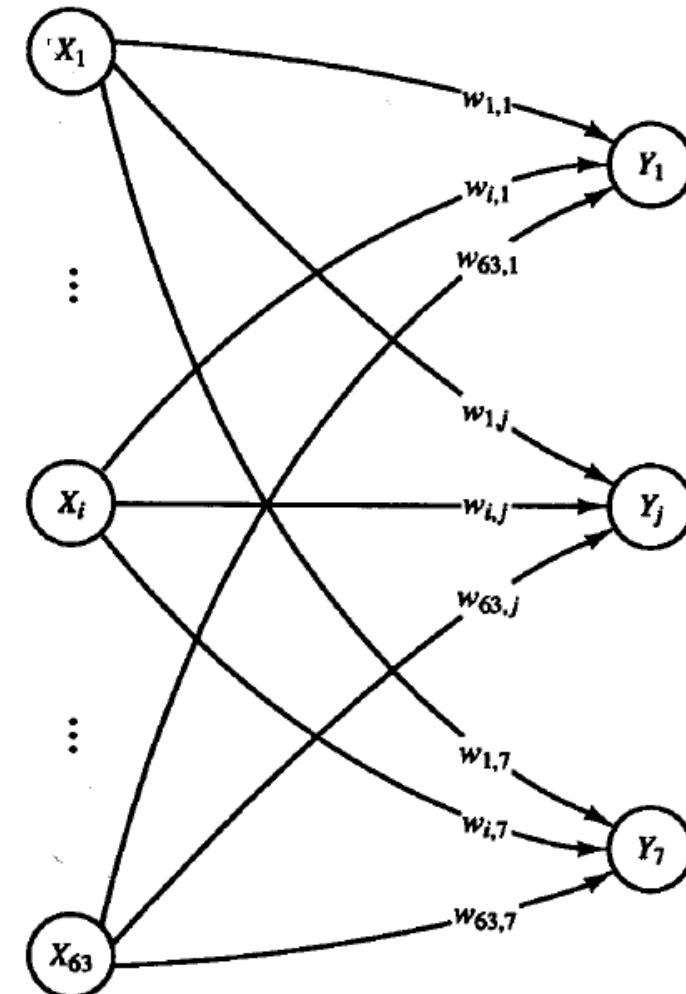
توجه: هیچ تغییر وزنی صورت نگرفته است!
پس روال آموزش پایان می‌پذیرد.

مثال عملی: تشخیص کاراکتر

Input from
Font 1



Input from
Font 2



Input from
Font 3

الگوریتم آموزش شبکه تک لایه پر سپترون

- Step 0.** Initialize weights and biases
(0 or small random values).
- Step 1.** While stopping condition is false, do Steps 1–6.
- Step 2.** For each bipolar training pair $s : t$, do Steps 3–5.
- Step 3.** Set activation of each input unit, $i = 1, \dots, n$:

$$x_i = s_i.$$

- Step 4.** Compute activation of each output unit,
 $j = 1, \dots, m$:

$$y_in_j = b_j + \sum_i x_i w_{ij}.$$

$$y_j = \begin{cases} 1 & \text{if } y_in_j > \theta \\ 0 & \text{if } -\theta \leq y_in_j \leq \theta \\ -1 & \text{if } y_in_j < -\theta \end{cases}$$

- Step 5.** Update biases and weights, $j = 1, \dots, m$;
 $i = 1, \dots, n$:
If $t_j \neq y_j$, then

$$b_j(\text{new}) = b_j(\text{old}) + t_j;$$

$$w_{ij}(\text{new}) = w_{ij}(\text{old}) + t_j x_i.$$

Else, biases and weights remain unchanged.

- Step 6.** Test for stopping condition:
If no weight changes occurred in Step 2, stop; otherwise, continue.

جمع‌بندی شبکه پرسپترون

- قانون آموزش پرسپترون:
- وزنها هنگامی تغییر می‌کند که خطای وجود داشته باشد.
- در مقایسه با هب که به وجود یا عدم وجود خطا کاری نداشتیم.
- چون بر اساس خطای موجود عمل می‌کند، انتظار می‌رود در هنگام پایان الگوریتم، هیچ خطای وجود نداشته باشد!
- اگر یک مساله قابل حل باشد (تفکیک پذیر خطی)، آنگاه اثبات می‌شود که شبکه پرسپترون در جواب مساله همگراست (یعنی حتماً وزنهای مناسب را پیدا می‌کند).
- به شرط انتخاب پارامترهای مناسب برای شبکه (مقادیر θ و α)
- در شبکه هب، هر الگو فقط یکبار به شبکه آموزش داده می‌شد، اما در پرسپترون ممکن است تمام الگوها چندین بار به شبکه آموزش داده شوند. (تا زمانی که خطای مشاهده نشود.)
- ماکزیمم تعداد دفعات تکرار در آموزش پرسپترون قابل محاسبه است.



نکات تکمیلی پرسپترون

- خروجی شبکه پرسپترون همیشه با ایپولار است (به خاطر تابع فعالیت خاص این شبکه)
- ورودی شبکه پرسپترون می‌تواند باینری یا با ایپولار باشد.
- ورودی باینری به خاطر صفر کردن جمله تصحیح اوزان، سرعت الگوریتم را کاهش می‌دهد. (تقریباً تا یک دهم ورودیهای با ایپولار)
- پارامتر α نرخ آموزش است و سرعت آن را کنترل می‌کند.
 - α کوچک، باعث کندی و دقت بیشتر.
 - α بزرگ، باعث سرعت بیشتر ولی نوسان حول جواب.
- پارامتر θ میزان تفکیک دسته‌ها را کنترل می‌کند.
 - θ کوچک: کاهش دقت دسته‌بندی و افزایش تعداد دسته‌ها.
 - θ بزرگ: افزایش دقت دسته‌بندی و کاهش تعداد دسته‌ها.
- انتخاب نامناسب پارامترهای شبکه یعنی α و θ ممکن است باعث عدم همگرایی شبکه شود.



مبانی هوش محاسباتی

فصل یک- شبکه‌های عصبی مصنوعی تک‌لایه

نقش ریاضیات در شبکه‌های عصبی مصنوعی و شبکه عصبی آدالاین



شبکه آدلاین (ADALINE)

- معرفی شده توسط Widrow & Hoff به سال ۱۹۶۰.

— مخفف ADAptive LInear NEuron

— ایده اولیه: استفاده از منطق و اصول ریاضیات شناخته شده در حل معادلات دیفرانسیل و تبدیل آن به روش تغییر وزنهای نرون.

- تفاوت با پرسپترون

— قانون آموزش دقیق‌تر.

- قانون دلتا؛ قانون کمترین مربعات خط (LMS)؛ قانون ویدرو-هاف.

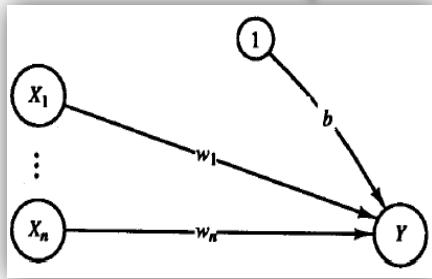
— آموزش، بر اساس میزان خطای صورت می‌پذیرد؛ نه فقط وجود آن.

— تابع فعالیت در فاز آموزش و فاز استفاده متفاوت است.

- شکل ابتدایی که منجر به آموزش شبکه‌های چندلایه شد.



مفهوم خروجی شبکه و خطای شبکه



$$E = \frac{1}{2} \sum_{p=1}^P (y - t_p)^2$$

$$y = y_{in} = \sum_{i=0}^n w_i \cdot x_i^p$$

- شبکه آدالاین:
 - خروجی مطلوب داده شده و خروجی شبکه قابل محاسبه است.

- خطای شبکه، به معنای اختلاف بین این دو است.

- P تعداد الگوها، x_i^p ها الگوی ورودی w_i^p و t_p خروجی مورد انتظار سلول (در هنگام ورود الگوی x_i^p) است.
- تنها پارامترهای سیستم یا معادله فوق، وزن‌ها هستند.

$$x_0 = 1 \quad w_0 = b$$

- هدف، کاهش خطای خروجی شبکه است. این کار به کمک محاسبه مقادیر مناسب برای وزنها (پارامترها) صورت می‌گیرد.



نقطه اکسٹرمم تابع خطای شبکه

$$E = \frac{1}{2} \sum_{p=1}^P (y - t_p)^2$$

$$y = y_{in} = \sum_{i=0}^n w_i \cdot x_i^p$$

$$\frac{\partial E}{\partial w_i} = 0$$

- توجه: تابع خطای (E) به ازای هر متغیر مستقل w_i یک تابع درجه دو است
 - فقط یک مقدار اکسٹرمم وجود دارد که همان نقطه مینیمم است.
 - روش یافتن نقطه اکسٹرمم: مشتق تابع را مساوی صفر قرار می‌دهیم:
- بنابراین، برای رسیدن به نقطه مینیمم در تابع خطای شبکه، باید به ازای هر پارامتر w_i (وزن‌ها)، مشتق نسبی را محاسبه کرده و مساوی صفر قرار دهیم.
 - تعداد معادلات و مشتقات نسبی به تعداد وزن‌ها است.
 - یک دستگاه n معادله، n مجھول!



حل دستگاه معادلات خطی

- برای یک الگوی خاص، خطاب به صورت زیر خواهد بود:

$$E = \frac{1}{2} (t - y)^2$$

- به ازای یک وزن خاص، مثل w_I داریم:

$$\frac{\partial E}{\partial w_I} = \frac{\partial}{\partial w_I} \left(\frac{1}{2} (t - y_{in})^2 \right) = (t - y_{in}) \frac{\partial}{\partial w_I} (t - y_{in}) = (t - y_{in}) \cdot -\frac{\partial}{\partial w_I} \left(\sum_{i=0}^n w_i x_i \right)$$

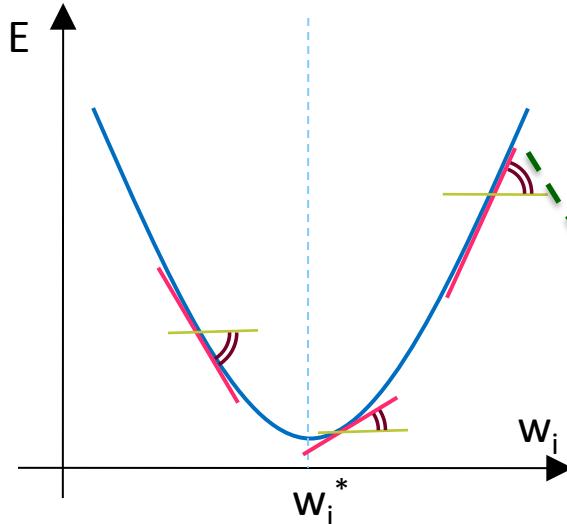
$$\frac{\partial E}{\partial w_I} = -(t - y_{in}) \cdot x_I$$

- هدف، یافتن نقطه مینیمم خطاب به کمک «صفر» کردن جمله (جملات) فوق است.
 - حل دستگاه معادلات فوق، جواب تحلیلی را برای ما حاصل می‌کند.
- اما در عمل، چه کار کنیم اگر جمله بالا (مقدار مشتق) صفر نبود؟
 - راه حل: پارامتر مستقل (وزن) را به گونه‌ای تغییر دهیم که مقدار مشتق به صفر نزدیک شود؛



دید هندسی برای رسیدن به نقطه اکسترمم

$$E = \frac{1}{2} \sum_{p=1}^P (t - y)^2$$



- هدف، یافتن نقطه مینیمم خطابه کمک «صفر» کردن جملات مقدار مشتق است.
- چه کار کنیم اگر مقدار مشتق صفر نبود؟

- راه حل: پارامتر مستقل (وزن) را به گونه‌ای تغییر دهیم که مقدار مشتق به صفر نزدیک شود:

$$\Delta w_i = \alpha(t - y_{in}) \cdot x_i$$

$$\frac{\partial E}{\partial w_i} = -(t - y_{in}) \cdot x_i$$

- « علامت (جهت) تغییر وزن: مخالف مشتق خطابه.
« قدر مطلق (اندازه) تغییر وزن: متناسب با مشتق خطابه.

آموزش آدالین

Step 0.

Initialize weights.

(Small random values are usually used.)

Set learning rate α .

(See comments following algorithm.)

Step 1.

While stopping condition is false, do Steps 2–6.

Step 2. For each bipolar training pair $s:t$, do Steps 3–5.

Step 3. Set activations of input units, $i = 1, \dots, n$:

$$x_i = s_i.$$

Step 4. Compute net input to output unit:

$$y_in = b + \sum_i x_i w_i.$$

Step 5. Update bias and weights, $i = 1, \dots, n$:

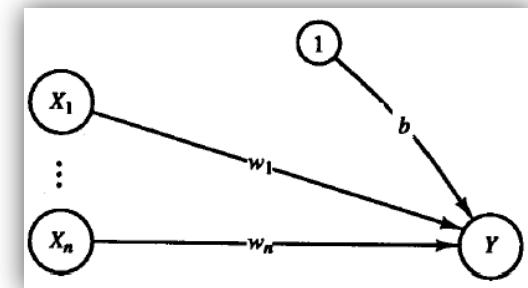
$$b(\text{new}) = b(\text{old}) + \alpha(t - y_in).$$

$$w_i(\text{new}) = w_i(\text{old}) + \alpha(t - y_in)x_i.$$

Step 6.

Test for stopping condition:

If the largest weight change that occurred in Step 2 is smaller than a specified tolerance, then stop; otherwise continue.



استفاده از آدالاین

- Step 0.* Initialize weights
 (from ADALINE training algorithm given in Section 2.4.2).
- Step 1.* For each bipolar input vector x , do Steps 2–4.
- Step 2.* Set activations of the input units to x .
- Step 3.* Compute net input to output unit:

$$y_in = b + \sum_i x_i w_i.$$

- Step 4.* Apply the activation function:

$$y = \begin{cases} 1 & \text{if } y_in \geq 0; \\ -1 & \text{if } y_in < 0. \end{cases}$$



نقش تابع فعالیت آدالاین

- مشاهده می شود که تابع فعالیت در زمان اجرا، متفاوت با زمان آموزش است.
 - در زمان آموزش: تابع همانی.
 - در زمان اجرا: تابع پله (می تواند با توجه به استفاده، بایپولار یا باینری باشد)
- تاثیر تابع فعالیت
 - تابع فعالیت همانی در بخش آموزش، امکان نزدیک کردن هرچه بیشتر ورودی سلول (y_{in}) به خروجی مورد انتظار (t) را فراهم می کند.
 - تابع فعالیت پله در بخش استفاده، امکان تطبیق خروجی سلول (y) بر خروجی مورد انتظار (t) را فراهم می کند.



نرخ آموزش (α)

- نقش نرخ آموزش (α) در شبکه آدلاین
 - α خیلی کوچک: سرعت همگرایی بسیار کم.
 - α خیلی بزرگ: ممکن است باعث عدم همگرایی شود.
 - روش ساده برای انتخاب α برای شبکه آدلاین، فقط با **یک** سلوی
 - با فرض این که n تعداد ورودی‌ها باشد:
 - نکته بسیار مهم: تعیین پارامتر α کاملاً وابسته به خطای مورد انتظار شبکه است. برای خطای کمتر، ممکن است مجبور شویم α را کاهش دهیم.



شرط توقف الگوریتم آموزش

- شرط خاتمه الگوریتم می‌تواند یکی از موارد زیر باشد:
 - کاهش خطای رسانیدن به حد قابل قبول از پیش معلوم (ع)
- این خطای می‌تواند خطای یک الگو، یا مجموع خطای تمام الگوها در یک دوره باشد.
- همگرا شدن وزن‌ها
- اگر بیشترین تغییرات وزن در دوره گذشته، از حد مشخصی کمتر شد (ایده‌آل: تغییر وزن متوقف شود).
- با توجه به قانون دلتا، تغییرات وزن و خطای شبکه رابطه مستقیمی با یکدیگر دارند؛ بنابراین این دو شرط توقف، مستقل از یکدیگر نیستند.
- گذشتن تعداد تکرارها از حد مجاز
- در این صورت شبکه با پارامترهای فعلی، قادر به همگرایی نیست. (یا اصولاً مساله تفکیک پذیرخطی نیست).



میزان خطای قابل قبول (۴)

- نکته‌های مهم:
 - با کاهش ϵ ، تعداد دوره‌های الگوریتم افزایش و وزن‌ها به وزن‌های ایده‌آل نزدیک‌تر (همگرا) می‌شوند.
 - اگر ϵ خیلی کم در نظر گرفته شود، ممکن است شبکه همگرا نشود. در این حالت ممکن است با کاهش نرخ آموزش (α) بتوان همچنان شبکه را با موقیت آموزش داد.
 - حد پایین (کمترین مقدار ممکن) ϵ قابل محاسبه است. اگر ϵ از این مقدار کمتر انتخاب شود، در هیچ صورتی شبکه همگرا نخواهد شد.



مثال: محاسبه دستی وزنها و حد پایین خطای ممکن

$$E = \frac{1}{2} \sum_{p=1}^4 (y - t_p)^2$$

توجه:
 $w_0 = b$
و
 $x_0 = 1$

$$y = y_{in} = \sum_{i=0}^2 w_i \cdot x_i$$

x	t
1	+1
1	-1
0	-1
0	-1

$$E = \frac{1}{2} \sum_{p=1}^4 (w_0 + w_1 x_{1(p)} + w_2 x_{2(p)} - t_{(p)})^2$$

$$E = \frac{1}{2} [(w_0 + w_1 + w_2 - 1)^2 + (w_0 + w_1 + 1)^2 + (w_0 + w_2 + 1)^2 + (w_0 + 1)^2]$$

$$E = w_1^2 + w_2^2 + 2w_0^2 + w_1 w_2 + 2w_1 w_0 + 2w_2 w_0 + 2w_0 + 2$$

$$\begin{cases} \frac{\partial E}{\partial w_1} = 2w_1 + w_2 + 2w_0 = 0 \\ \frac{\partial E}{\partial w_2} = w_1 + 2w_2 + 2w_0 = 0 \\ \frac{\partial E}{\partial w_0} = 2w_1 + 2w_2 + 4w_0 + 2 = 0 \end{cases}$$



$$\begin{cases} w_1 = 1 \\ w_2 = 1 \\ w_0 = -\frac{3}{2} \end{cases}$$



$$E = \frac{1}{2} [(-0.5)^2 + (0.5)^2 + (0.5)^2 + (-0.5)^2] = 0.5$$

وزنهای ایده‌آل شبکه. که توسط آنها،
کمترین خطای ممکن حاصل می‌شود.



کمترین
خطای
ممکن

با مراجعه به اسلاید مربوط به آموزش آدالین،
مشاهده می‌شود که الگوریتم آموزش به وزنها و
خطای ایده‌آل، همگرا شده است!

مثال: آموزش شبکه آدلاین

$\alpha = 0.01$	w_1	w_2	b	Epochs
$\epsilon = 1$	0.4153	0.3975	-0.7821	101
$\epsilon = 0.9$	0.5178	0.5415	-0.9311	130
$\epsilon = 0.7$	0.7791	0.7656	-1.2226	259
$\epsilon = 0.6$	0.8953	0.8900	-1.3671	364
$\epsilon = 0.55$	0.9561	0.9502	-1.4396	477
$\epsilon = 0.52$	0.9920	0.9869	-1.4833	647
$\epsilon = 0.51$	1.0041	0.9991	-1.4979	832
$\epsilon = 0.505$	-	-	-	∞

$\alpha = 0.001$	w_1	w_2	b	Epochs
$\epsilon = 0.505$	0.9955	0.9950	-1.4939	8446
$\epsilon = 0.502$	0.9992	0.9987	-1.4983	10370
$\epsilon = 0.501$	1.0004	0.9999	-1.4998	12025
$\epsilon = 0.5008$	1.0006	1.0001	-1.5001	12951
$\epsilon = 0.5005$	-	-	-	∞

- مساله AND با ورودی باینری و خروجی بایپولار
 - الگوهای ورودی:

s	t
1	1
1	0
0	1
0	0

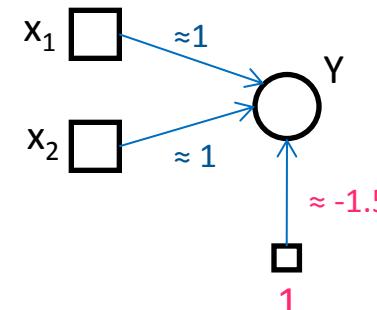
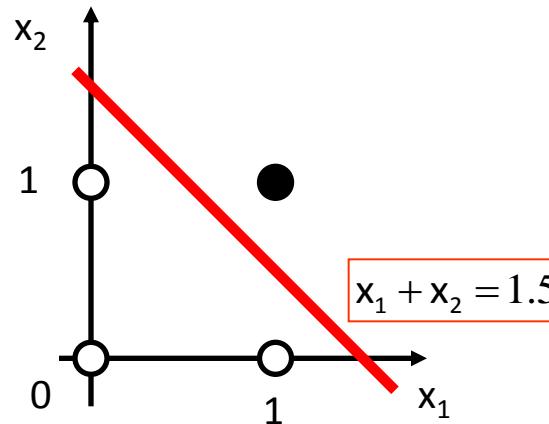
- روش انتخاب وزنهای:
- مقادیر تصادفی، در بازه: [-0.5 / 0.5]

- خروجی برنامه (وزنهای نهایی) و تعداد دفعات تکرار با پارامترهای متفاوت:



مثال: استفاده از شبکه آدلاین

- با فرض تقریب وزنهای به دست آمده از مرحله آموزش، شبکه به صورت زیر حاصل می‌شود:



- حتی اگر فرض کنیم وزنهای سطر اول ($\alpha = 0.01$ و $\epsilon = 1$) را در نظر بگیریم، با توجه به الگوریتم استفاده، داریم:

w_1	w_2	b
0.4153	0.3975	-0.7821

$$y_{in} = 0.4153 * x_1 + 0.3975 * x_2 - 0.7821$$

$$Y = \begin{cases} 1 & y_{in} \geq 0 \\ -1 & y_{in} < 0 \end{cases}$$

x_1	x_2	y_{in}	Y
0	0	-0.7821	-1
0	1	-0.3846	-1
1	0	-0.3668	-1
1	1	0.0307	1

قانون دلتا برای شبکه‌تک‌لایه آدالاین

- استخراج قانون آموزش دلتا، در هنگامی که بیش از یک سلول در لایه خروجی قرار دارد.

$$E = \sum_{j=1}^m \frac{1}{2} (t_j - y_j)^2$$

$$\begin{aligned} \frac{\partial E}{\partial w_{IJ}} &= \frac{\partial}{\partial w_{IJ}} \sum_{j=1}^m \left(\frac{1}{2} (t_j - y_{inj})^2 \right) = \frac{\partial}{\partial w_{IJ}} \left(\frac{1}{2} (t_J - y_{inJ})^2 \right) = (t_J - y_{inJ}) \frac{\partial}{\partial w_{IJ}} (t_J - y_{inJ}) \\ &= (t_J - y_{inJ}) \cdot -\frac{\partial}{\partial w_{IJ}} \left(\sum_{i=1}^n w_{iJ} x_i \right) = -(t_J - y_{inJ}) x_I \end{aligned}$$

- بنابراین، تغییرات وزن طبق قانون دلتا، برای هر وزن در شبکه عصبی‌تک‌لایه آدالاین، به صورت زیر خواهد بود:

$$\Delta w_{IJ} = \alpha (t_J - y_{inJ}) \cdot x_I$$

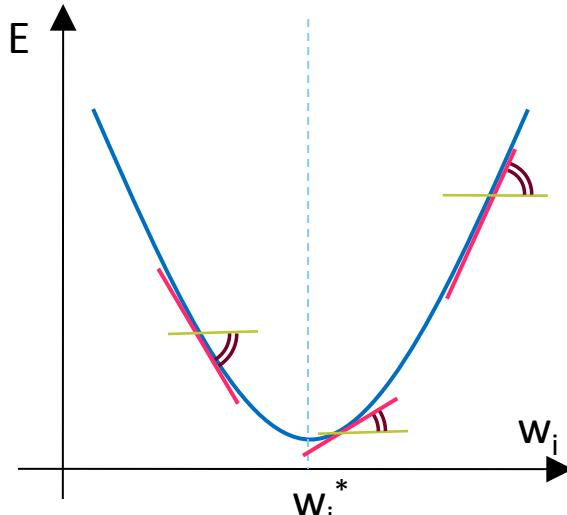


$$\frac{\partial E}{\partial w_i} = -(t - y_{in}) \cdot x_i$$

$$\Delta w_i = \alpha(t - y_{in}) \cdot x_i$$

تغییر قانون دلتا

$$E = \frac{1}{2} \sum_{p=1}^P (t - y)^2$$



- با توجه به تعبیر هندسی مشتق (زاویه خط مماس بر منحنی با محور افقی)
 - جهت تغییرات w_i باید عکس مشتق در نقطه فعلی باشد.
 - « هرگاه مشتق (زاویه) مثبت باشد، w_i باید کاهش یابد.
 - « هرگاه مشتق (زاویه) منفی باشد، w_i باید افزایش یابد.
 - اندازه تغییرات w_i باید متناسب با مقدار مشتق در نقطه فعلی باشد.
 - « اگر مقدار مشتق بزرگ بود (از جواب دور بودیم)، گام بعدی (در تغییر w_i) بزرگ باشد.
 - « اگر مقدار مشتق کوچک بود (به جواب نزدیک بودیم)، گام بعدی (در تغییر w_i) کوچک باشد.

نکاتی درباره قانون دلتا

- علامت (جهت) تغییر وزن: مخالف مشتق خطا.
- قدر مطلق (اندازه) تغییر وزن: متناسب با مشتق خطا.

تغییر هندسی قانون دلتا

- تابع خط: مجموع مربعات خطای الگوها
- تابع درجه دو: پارامترها: وزنهای شبکه
- منحنی تغییرات E بر حسب w_i

- با توجه به تعبیر هندسی مشتق (زاویه خط مماس بر منحنی با محور افقی)

« هرگاه مشتق (زاویه) مثبت باشد، w_i باید کاهش یابد.

« هرگاه مشتق (زاویه) منفی باشد، w_i باید افزایش یابد.

- اندازه تغییرات w_i باید متناسب با مقدار مشتق در نقطه فعلی باشد.

« اگر مقدار مشتق بزرگ بود (از جواب دور بودیم)، گام بعدی (در تغییر w_i) بزرگ باشد.

« اگر مقدار مشتق کوچک بود (به جواب نزدیک بودیم)، گام بعدی (در تغییر w_i) کوچک باشد.

جمع‌بندی شبکه آدلاین

- آموزش به روش کمترین میانگین مربعات (Least-Mean-Squares-LMS)
 - تعریف «خطا» به عنوان تابعی که باید کمینه (بهینه شود) به صورت تابع درجه ۲.
 - به روزآوری و تغییر پارامترهای تابع، با توجه به علامت و اندازه مشتق تابع.
- کار شاخص ویدرو-هاف این بود که توانستند این محاسبات ریاضی و معادلات عددی را در قالب یک قانون آموزش برای سلولهای عصبی، ارایه بدهند.
 - ایده اولیه بهینه‌سازی به روش گرادیان نزولی (Gradient Descent)
 - بهینه‌سازی تابع هدف (که به صورت مجموع توابع مشتق پذیر نوشته شده) به کمک محاسبه گرادیان و کاهش آن.
 - راه حل آموزش شبکه‌های چندلایه.
- در صورت انتخاب مناسب پارامترهای الگوریتم، وزنهای شبکه به سمت مقادیر بهینه، همگرا می‌شوند.
 - مقادیر بهینه برای وزنهای مقادیری است که باعث کمترین مجدور خطای خطا بشوند.
 - مقادیر بهینه برای وزنهای را می‌توان به کمک دستگاه n -معادله n -جهول و به روش تحلیلی نیز به دست آورد.
- مزیت راه حل شبکه عصبی در این حالت، فرار از محاسبات دشوار در هنگامی است که n بزرگ است.
 - راه حل شبکه عصبی، به عنوان یک روش عددی برای حل دستگاه معادلات

محدودیت قانون دلتا

- قانون دلتا، فقط می‌تواند یک شبکه تک‌لایه را آموزش دهد.
 - مشکل آموزش شبکه‌های چندلایه: سلول‌های لایه‌های مخفی
- برای محاسبه تغییر وزن‌ها، محاسبه خطای هر سلول لازم است. از کجا بفهمیم هر سلول لایه میانی، چه مقدار خطا دارد؟
 - برای محاسبه خطای سلول‌های لایه میانی، باید بدانیم خروجی ایده‌آل، یا صحیح، برای هر سلول میانی چه مقدار است؟
- « جواب: نمی‌دانیم! در اکثر مواقع، نمی‌توان با دقت وظیفه و خروجی سلول‌های لایه میانی را تعیین نمود.
- پس چگونه یک شبکه چندلایه بسازیم و آن را آموزش دهیم؟

