شبکه‌های عصبی مصنوعی
۰۱-۷۱۳-۱۱-۱۳
بخش نخست

Hebb's rule
perceptron
LMS, ADALINE
Gradient decent

دانشگاه شهید بهشتی
دانشکده‌ی علوم و مهندسی کامپیوتر
زمستان ۱۳۹۴
احمد محمودی ازناوه

http://faculties.sbu.ac.ir/~a_mahmoudi/

فهرست مطالب

- پیش‌گفتار
  - چرا شبکه‌ی عصبی
  - شبکه‌های عصبی زیستی
- مدل ریاضی تک‌نرون
- الگوریتم یادگیری
- پرسپترون
- قضیه‌ی همگرایی
- Adaline
- نزول گرادیان
- تنظیم نرخ یادگیری

چرا شبکه‌ی عصبی؟

- چگونه می‌توان برنامه‌ای نوشت که هویت یک فرد را از طریق چهره تشخیص دهد یا برنامه‌ای که بتواند اشیاء متفاوت را دسته‌بندی کند؟
- یا برنامه‌ای که با توجه به سابقه‌ی پزشکی و خانوادگی فرد، عمر تقریبی او را حدس بزند!
  - نوشتن چنین برنامه‌هایی بسیار دشوار است، در حالی که مغز انسان ۱۰۰ تا ۲۰۰ میلی‌ثانیه چنین پردازشی را انجام می‌دهد.
  - در این موارد با داده‌های حجیمی روبرو هستیم که ارتباط کاملا دقیق و مشخصی بین آن‌ها برقرار نیست و یا کشف این ارتباط بسیار دشوار است.
  - نمی‌دانیم مغز ما چگونه‌ی چنین کارهایی را انجام می‌دهد.
  - نکته‌ی دیگر این که در مواردی، این ارتباط با مرور زمان تغییر خواهند کرد و این باعث دشواری بیشتر مسأله خواهد شد.

رویکرد یادگیری ماشین

- در الگوریتم‌های «یادگیری ماشین»، تعداد زیادی مثال همراه با پاسخ صحیح دریافت و برنامه‌ای برای حل مسأله تولید می‌کند.
  - در صورت انجام درست کار، برنامه برای نمونه‌های جدید هم درست کار خواهد کرد(**تعمیم‌پذیری**).
  - در صورتی که داده‌ها تغییر کنند، برنامه هم توانایی تغییر خواهد داشت(**وفقی بودن**).
- با توجه به افزایش قدرت محاسبات، انجام حجم عظیمی از محاسبات ارزان‌تر از نوشتن یک الگوریتم خاص می‌باشد.

کاربردها

- بازشناسی الگو(دسته‌بندی و رده‌بندی)
  - تشخیص اشیاء، تشخیص کاراکتر، تشخیص چهره یا تشخیص حالا چهره، تشخیص کلمات
- رگرسیون
- تشخیص ناهنجاری
  - استفاده از کارت اعتباری به صورت نامتعارف
- پیش‌بینی
  - پیش‌بینی قیمت سهام
  - پیش‌بینی لیست فیلم‌های مورد علاقه‌ی یک شخص

مثال

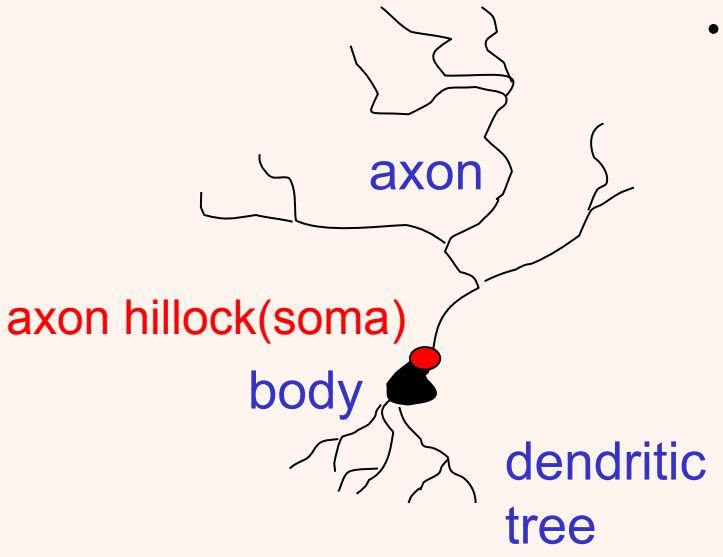
- آموختن یک شیوه‌ی نگارش
  - تشخیص متون Shakespeare
- تشخیص خلوص روغن زیتون
- برچسب‌زدن تصاویر(گوگل)
- تبدیل صوت از یک زبان به زبان دیگر(مایکروسافت)
- یافتن بهترین اهداکننده برای اهدای قلب(سوئد)
- سیستم‌های تحلیل ریسک

شبکه‌های عصبی مصنوعی

- ایده‌ی اصلی این شبکه‌ها مبتنی بر «شبکه‌های عصبی زیستی» است.
- بسیاری از مسائل توسط انسان به سادگی قابل حل می‌باشد.
- مغز به صورت موازی محاسبات را انجام می‌دهد.
- این مدل می‌تواند برای مسائلی که توسط ذهن آدمی به راحتی انجام می‌شود، مفید باشد.
- در واقع شیوه‌ی به کار رفته در ذهن به نوعی الهام بخش ارائه‌ی مدلی برای ایجاد قابلیت‌هایی مشابه با مغز است، هرچند شبکه‌ی عصبی مورد استفاده‌ی ما تفاوت‌های بسیاری با شبکه‌های عصبی زیستی دارد.

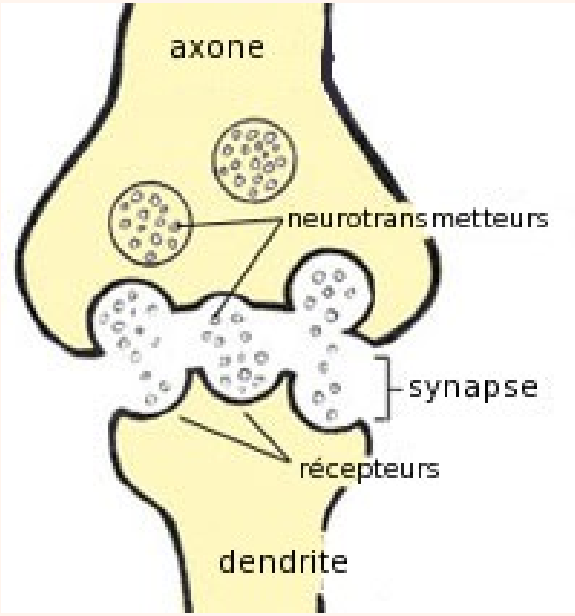
ساختار یک نرون طبیعی

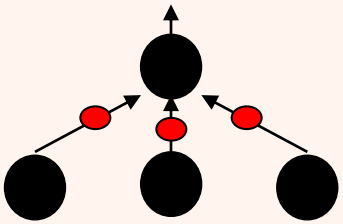
- مغز انسان شامل حدود 10^{11} نرون است که به صورت فوق‌العاده‌ای به هم پیوسته هستند که هر نرون به طور متوسط با 10^{4} نرون دیگر مرتبط است.
- یافته نشان می‌دهند داده‌ها در اتصالات بین نرون‌ها ذخیره می‌شود.
- شامل یک آسه(آکسون) است که شاخه شاخه شده و پیام‌های الکتریکی را به بیرون یاخته هدایت می‌کند.
- یک خوشه از دارینه(دندریت)ها که پیام‌های الکتریکی را از سلول‌های مجاور دریافت می‌کند.



ساختار یک نرون طبیعی(ادامه...)

- همایه(سیناپس) یک ساختار زیستی در پایانه آکسون‌ها است که از راه آن یک سلول عصبی پیام خود را به دندریت یک نورون دیگر یا یاخته ماهیچه‌ای یا یک غده می‌فرستد.
- جسم سلولی مولد این سیگنال‌های ارسالی است. در صورتی که میزان سیگنال دریافتی از طریق دارینه‌ها از یک حد آستانه بیشتر باشد؛ نرون تحریک می‌شود.





مغز چگونه کار می‌کند؟

- هر نرون از نرون‌های دیگری ورودی دریافت می‌کند.
- برخی نرون‌های به سلول‌های گیرنده(receptor) متصل هستند.
- نرون‌ها با ارسال سیگنال‌های الکتریکی با یکدیگر ارتباط برقرار می‌کنند.
- اثر هر ورودی به وزن ارتباط سیناپسی بستگی دارد.
- این وزن‌ها به صورت وفقی تغییر می‌یابند تا کل شبکه محاسبات را به درستی انجام دهد.

مغز چگونه کار می‌کند؟(ادامه...)

- هر بخش قشر مغز وظیفه‌ای خاص دارد.
  - آسیب به هر بخش از مغز یک انسان بالغ، باعث تاثیرات خاصی می‌شود.
  - در صورت انجام فعالیت‌های خاص جریان خون در بخشی از بخش‌ها افزایش می‌یابد.
- بخش‌های مختلف قشر مغز(cortex) بسیار شبیه به هم هستند.
  - در صورتی که در بخشی از آن آسیب ببیند، بخش دیگر می‌تواند عهده‌دار وظیفه‌ی آن بخش شود، در واقع به نظر می‌رسد همه‌ی بخش‌ها از یک شیوه‌ی یادگیری استفاده می‌کنند.

شبکه‌ی عصبی

- شبکه‌ی عصبی پردازشگری با ساختار توزیع شده و قابلیت بالای موازی‌سازی است که از واحدهای پردازشگر ساده‌ای تشکیل شده است و قابلیت ذخیره کردن تجربیات و به کارگیری آن برای استفاده‌های آتی را دارا می‌باشد.
  - از طریق یادگیری از محیط اطراف کسب دانش می‌کند.
  - برای ذخیره‌سازی دانش از وزن‌های سیناپسی استفاده می‌کند.
- عمده مطالب این درس، در مورد نحوه‌ی تنظیم این وزن‌هاست تا بتواند مسائل خاصی را حل کنند.

ویژگی‌های شبکه‌های عصبی

- پردازش موازی (سرعت بالا)
- تحمیل‌پذیری
- محاسبات غیرخطی
- برقراری ارتباط یک‌سری ورودی و یک‌سری خروجی
  – بازیابی اطلاعات
- توانایی تطبیق (adaptivity)
- پاسخ به داده‌های نویزی
- تحمل‌پذیری خطا
- یادگیری

نیازمندی‌های شبکه‌های عصبی

- جمع‌آوری و آنالیز مناسب داده
- طرح، آموزش و تست شبکه‌ی عصبی
- بهنجار کردن(normalize) ورودی‌ها:
  - تغییرات باید به نحوی باشد که قابل برگشت بوده و هیستوگرام ورودی را تغییر ندهد.

تاریخچه‌ی مختصر

- ۱۹۴۳، مفهوم نرون McCulloch&Pitts
- ۱۹۴۹، قانون آموزش Hebb
- ۱۹۵۸، مفهوم پرسپترون Rosenblatt
- ۱۹۶۰، Adaline توسط Widrow&Hoff
- ۱۹۶۹، نقد شبکه‌ی عصبی Minsky&Papert
- ۱۹۷۲، شبکه‌های رقابتی و حافظه‌ی تداعی‌گر
- ۱۹۸۰، الگوریتم یادگیری پس‌انتشار خطا

deep artificial neural networks

مدل نرون

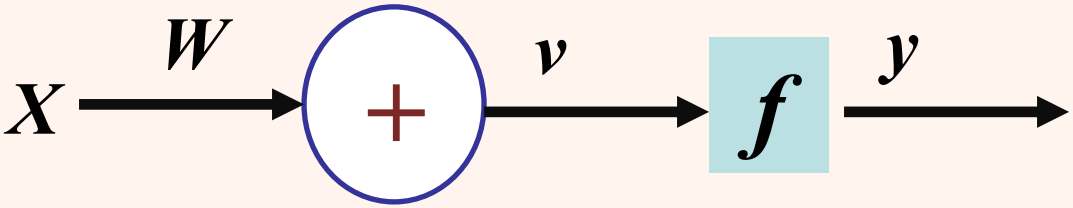
McCulloch and Pitts 1943

A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY

WARREN S. MCCULLOCH and WALTER H. PITTS

- کوچک‌ترین واحد پردازشگر اطلاعات

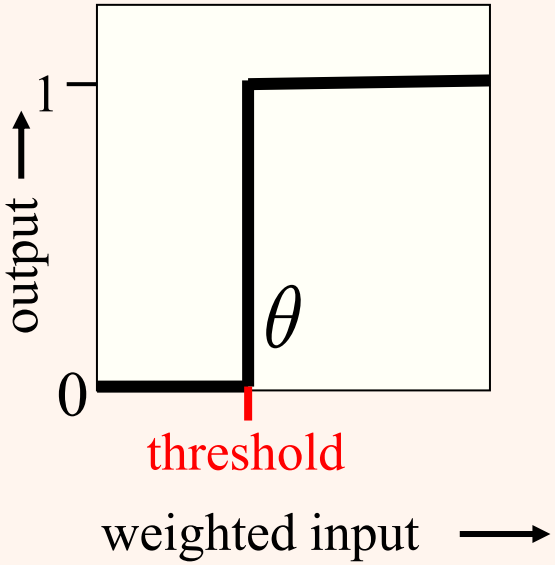
ساختار نرون تک ورودی



$$v = W.X$$

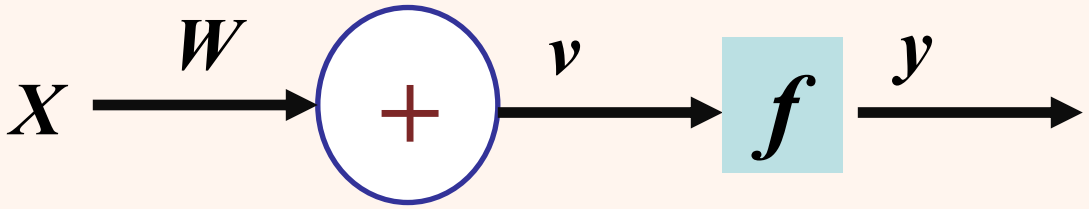
$$y = f(v)$$

$$y = f(W.X)$$



مدل نرون (ادامه...)

- به دو صورت می‌توان چنین نرونی را نمایش داد:



$$v = \sum_i x_i w_i$$

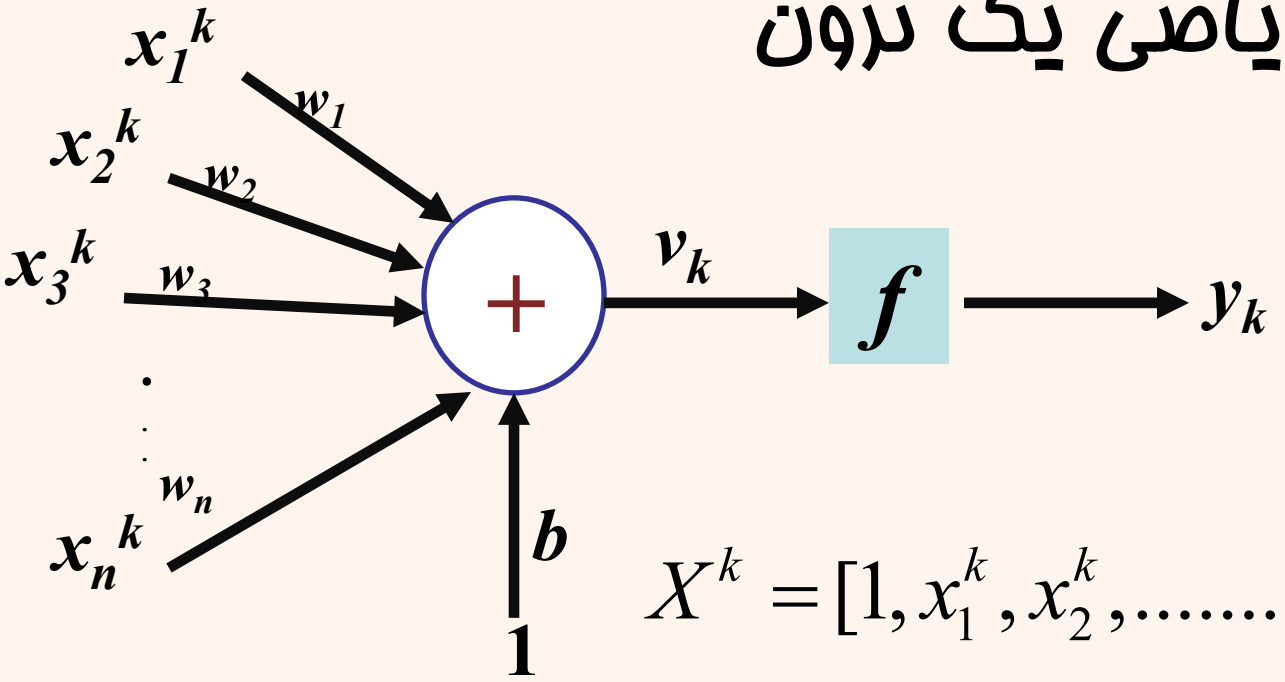
$$v = b + \sum_i x_i w_i$$

$$\theta = -b$$

$$y = \begin{cases} 1 \text{ if } v \geq \theta \\ 0 \text{ otherwise} \end{cases}$$

$$y = \begin{cases} 1 \text{ if } v \geq 0 \\ 0 \text{ otherwise} \end{cases}$$

مدل ریاضی یک نرون



$$X^k = [1, x_1^k, x_2^k, \ldots\ldots, x_n^k]$$

$$W = [w_0 = b, w_1, w_2, \ldots\ldots, w_n]$$

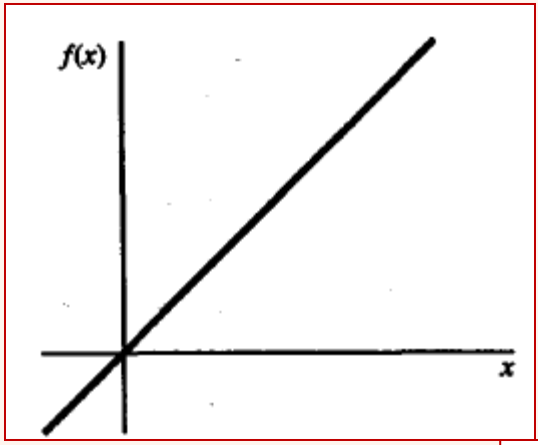
$$u_k = \sum_{i=1}^{n} w_i . x_i^k \quad \Rightarrow \quad v_k = u_k + b$$

$$v_k = W . X^k$$

$$y_k = f\left(\sum_{i=0}^{n} w_i . x_i^k + b \right)$$

تابع انگیزش

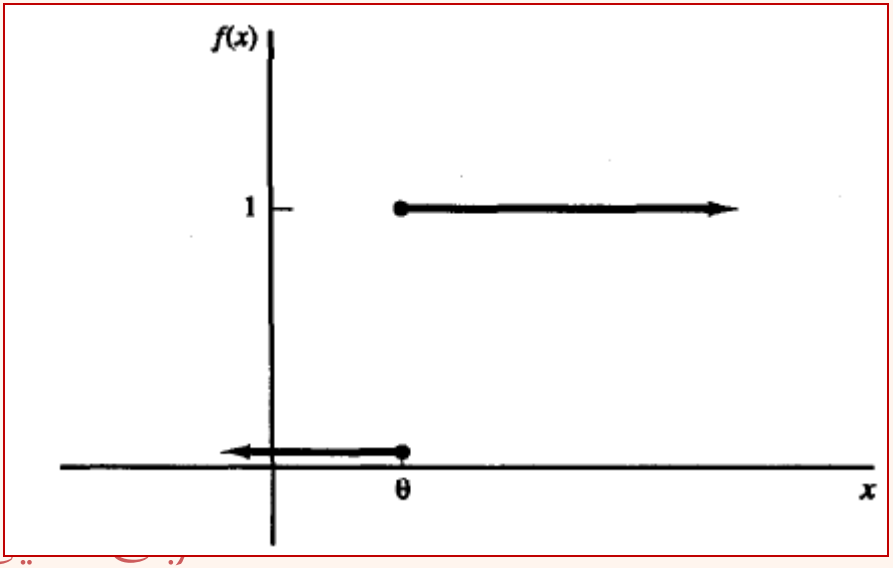
Activation Function



Identity function

$$f(x) = x \qquad \text{for all } x.$$

Binary step function (with threshold θ)

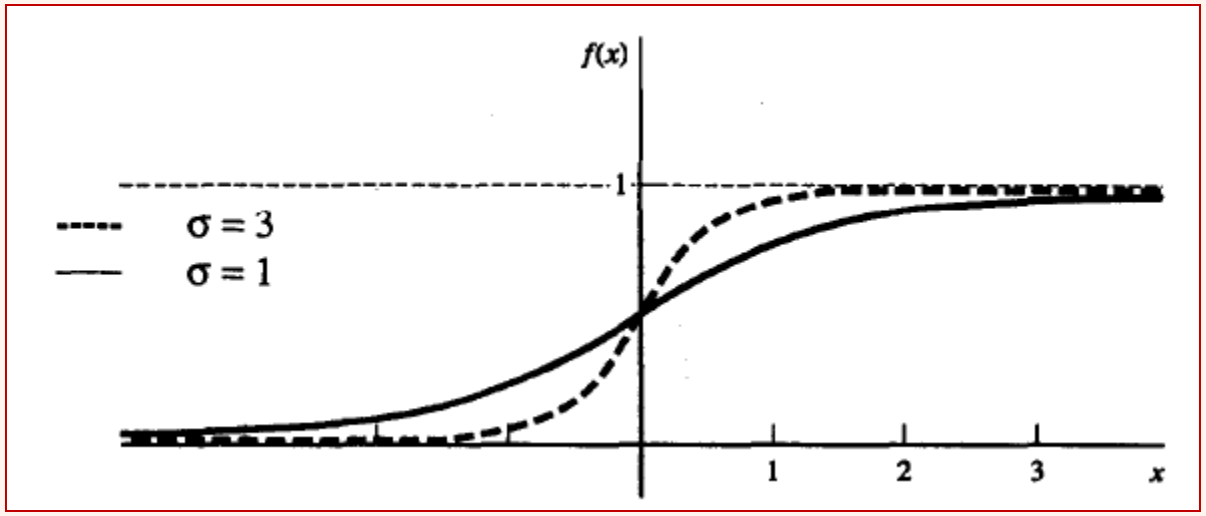


$$f(x) = \begin{cases} 1 & \text{if } x \geq \theta \\ 0 & \text{if } x < \theta \end{cases}$$

تابع انگیزش (ادامه...)

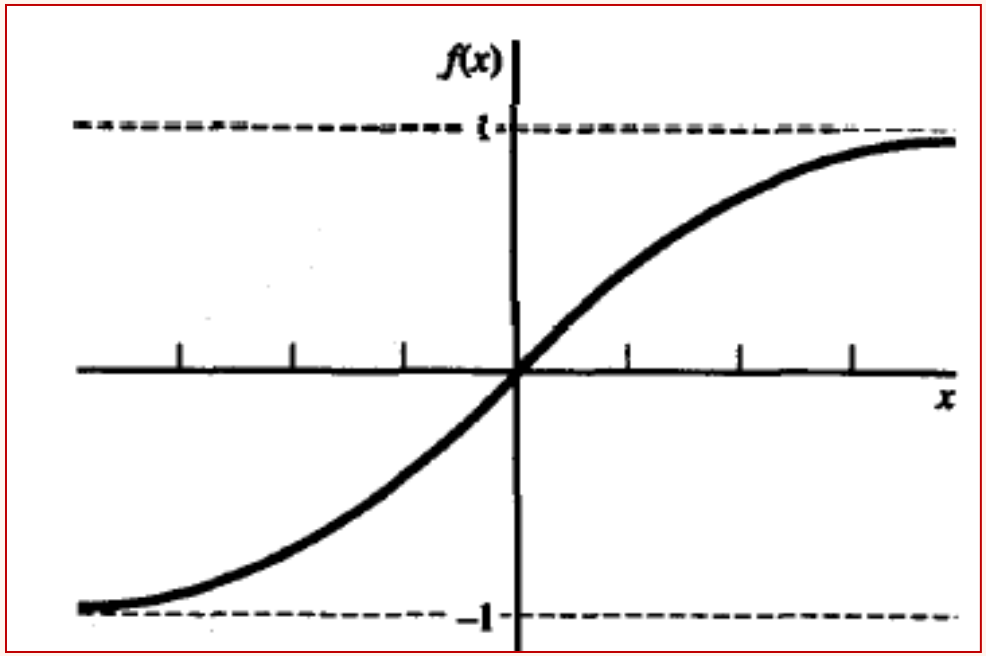
Binary sigmoid

$$f(x) = \frac{1}{1 + \exp(-\sigma x)}$$



تابع انگیزش

Bipolar sigmoid



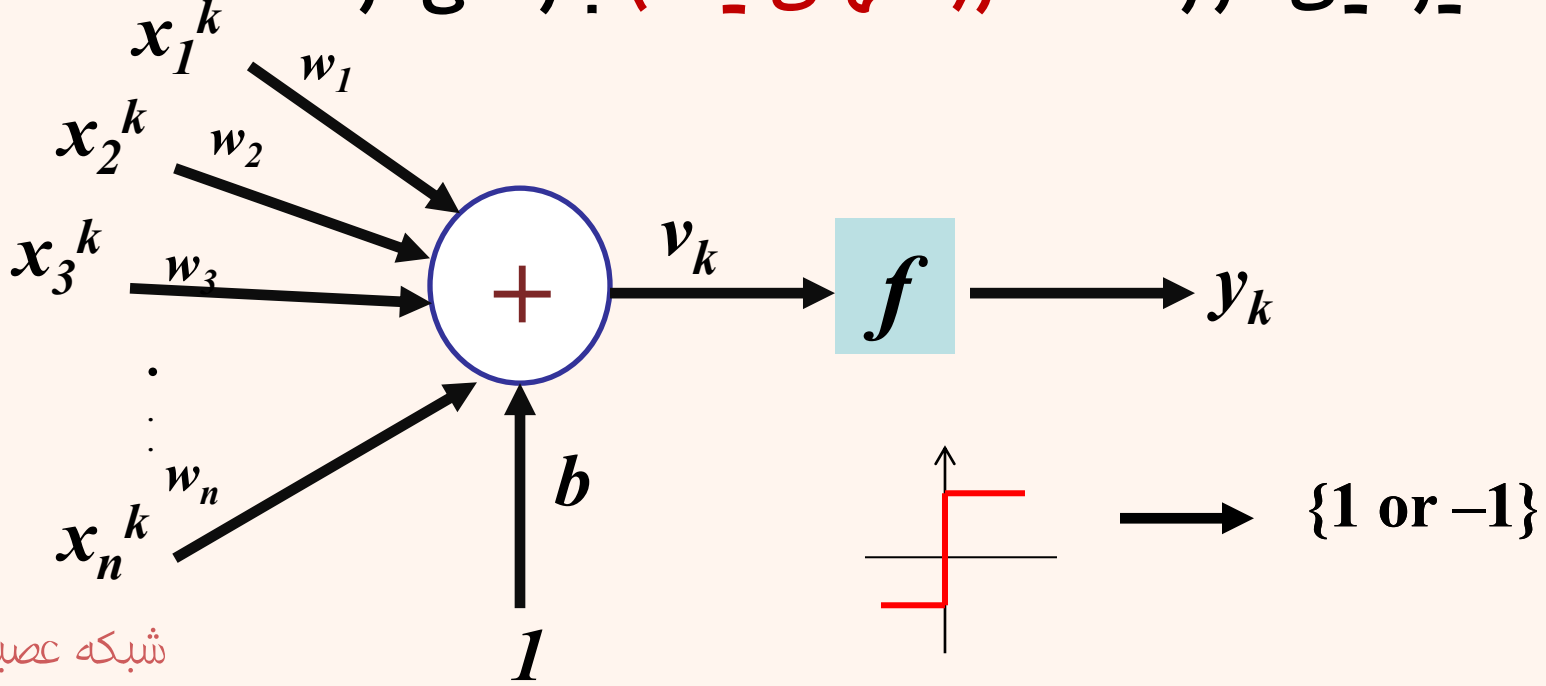
$$g(x) = 2f(x) - 1 = \frac{2}{1 + \exp(-ax)} - 1$$

$$= \frac{1 - \exp(-\sigma x)}{1 + \exp(-\sigma x)}.$$

Rosenblat 1957

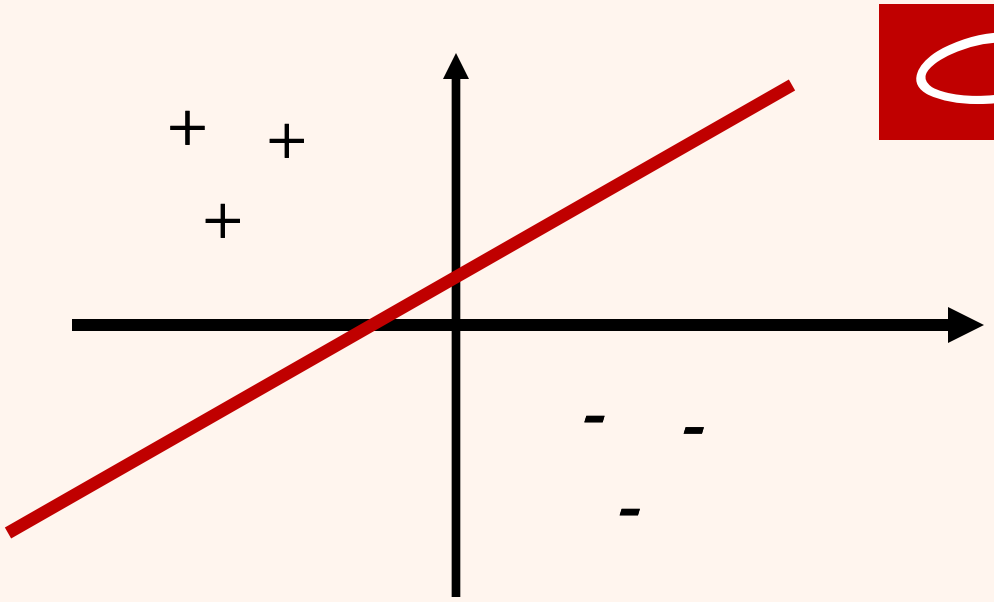
پرسپترون

- یک پرسپترون یک بردار ورودی را گرفته، ترکیبی خطی از آن‌ها را محاسبه نموده، خروجی را فراهم می‌آورد.
- اگر خروجی از میزان آستانه‌ای بالاتر بود یک و در غیر این‌صورت صفر(منهای یک) باز می‌گرداند.



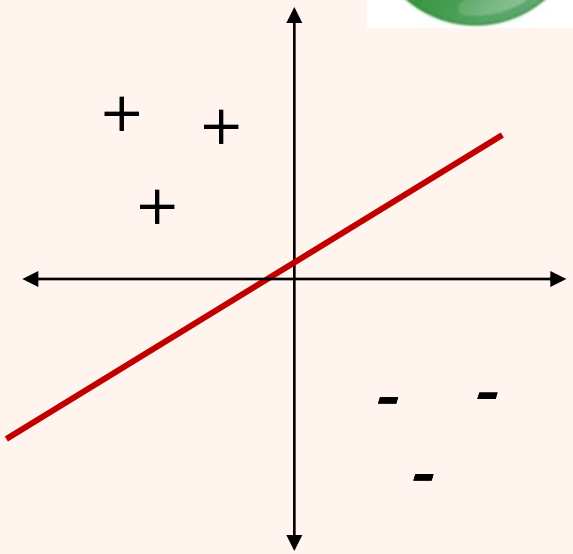
پرسپترون

- پرسپترون توانایی جداسازی داده‌های دوسطحی را داراست.
- می‌توان آن را به صورت یک جداکننده دوسطحی در نظر گرفت.

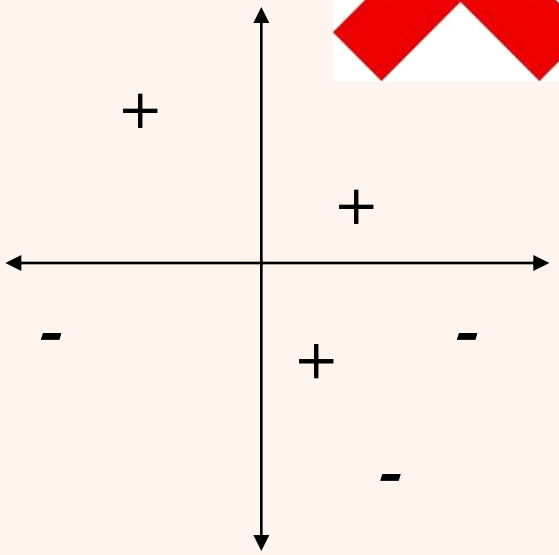


مثال

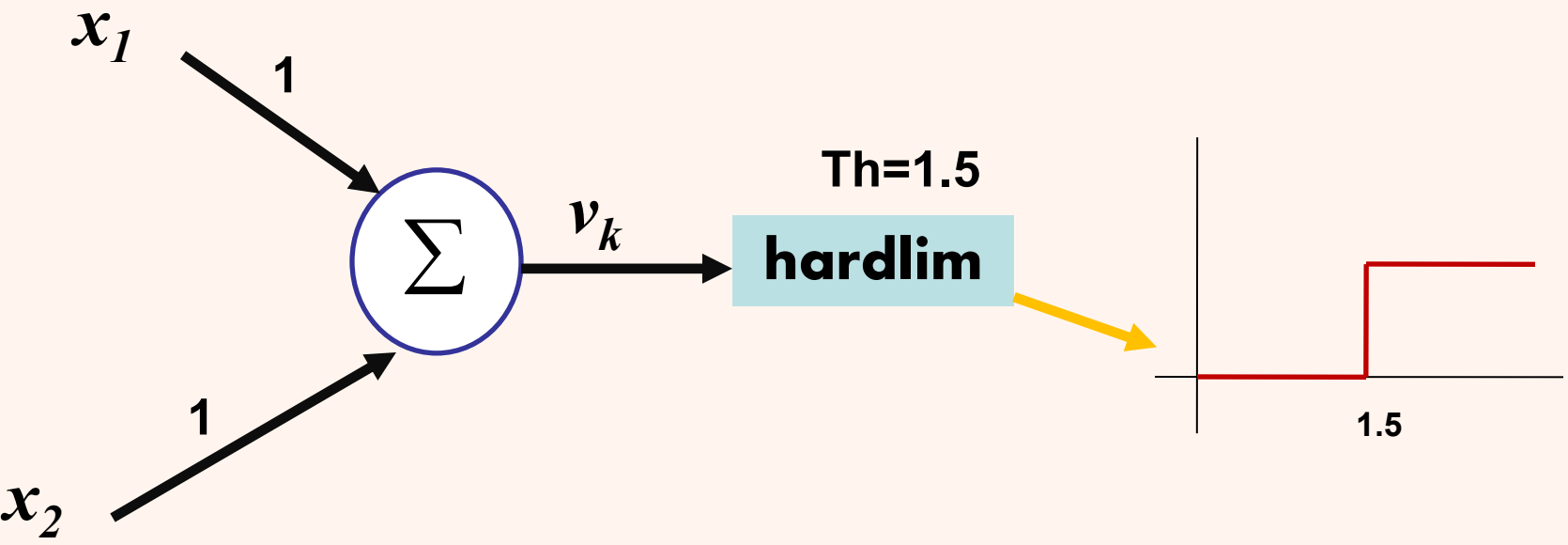
جدایی‌پذیر خطی

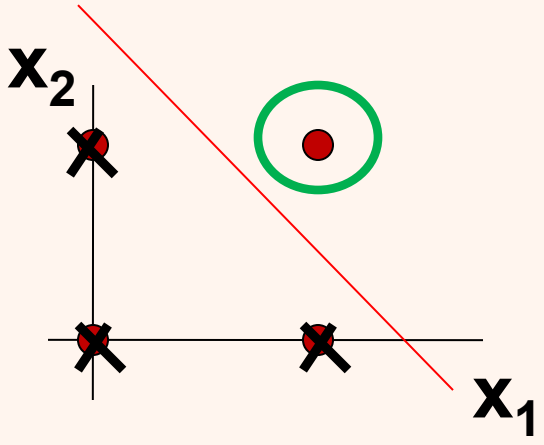


جدایی‌پذیر غیرخطی

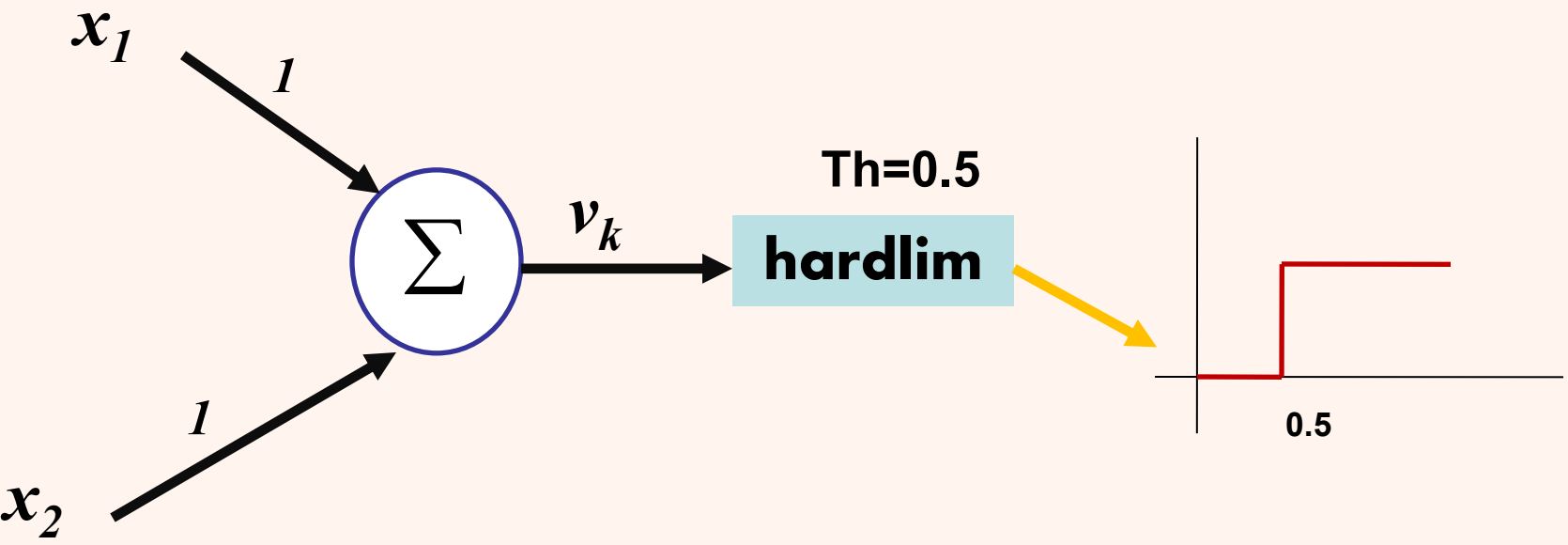


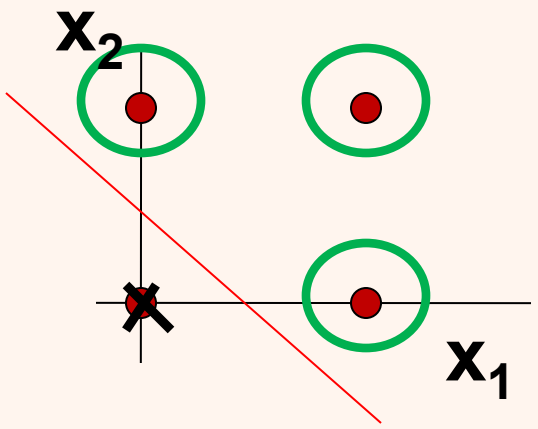
AND Gate



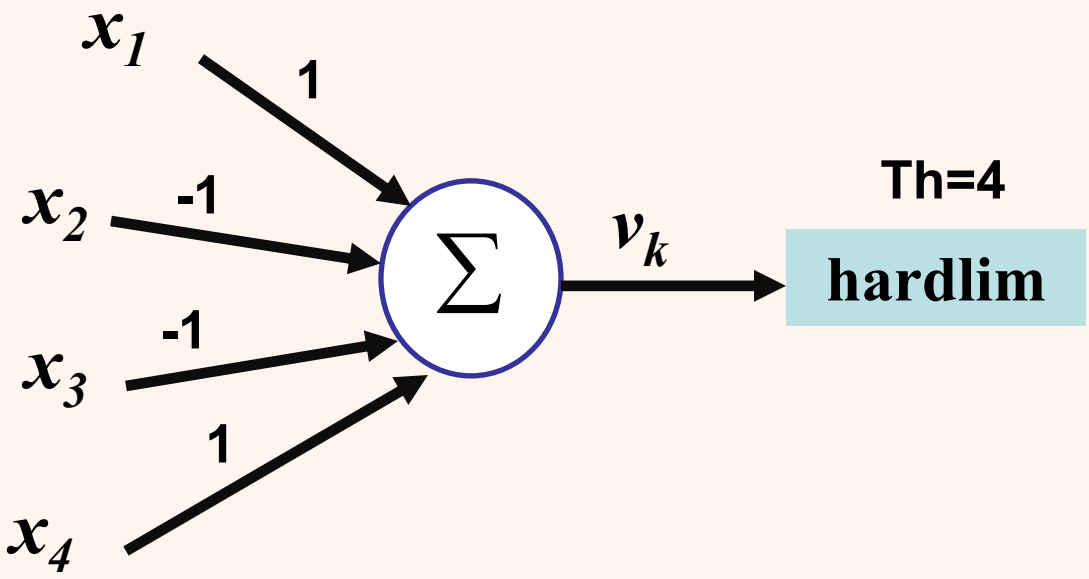


OR Gate



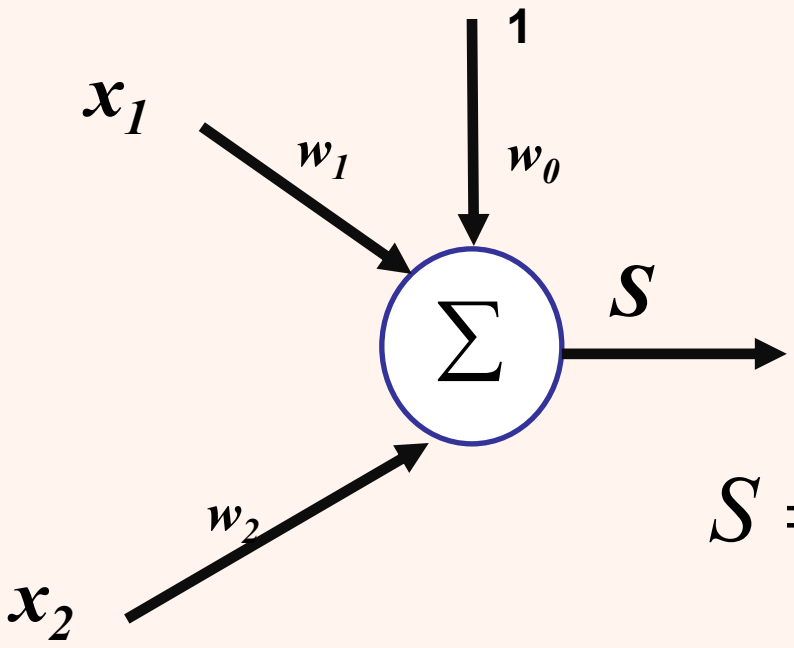


مثال



- به ازای کدام ورودی پاسخ یک است؟

بایاس(سوگیری)



$$S = w_1 x_1 + w_2 x_2 + w_0$$

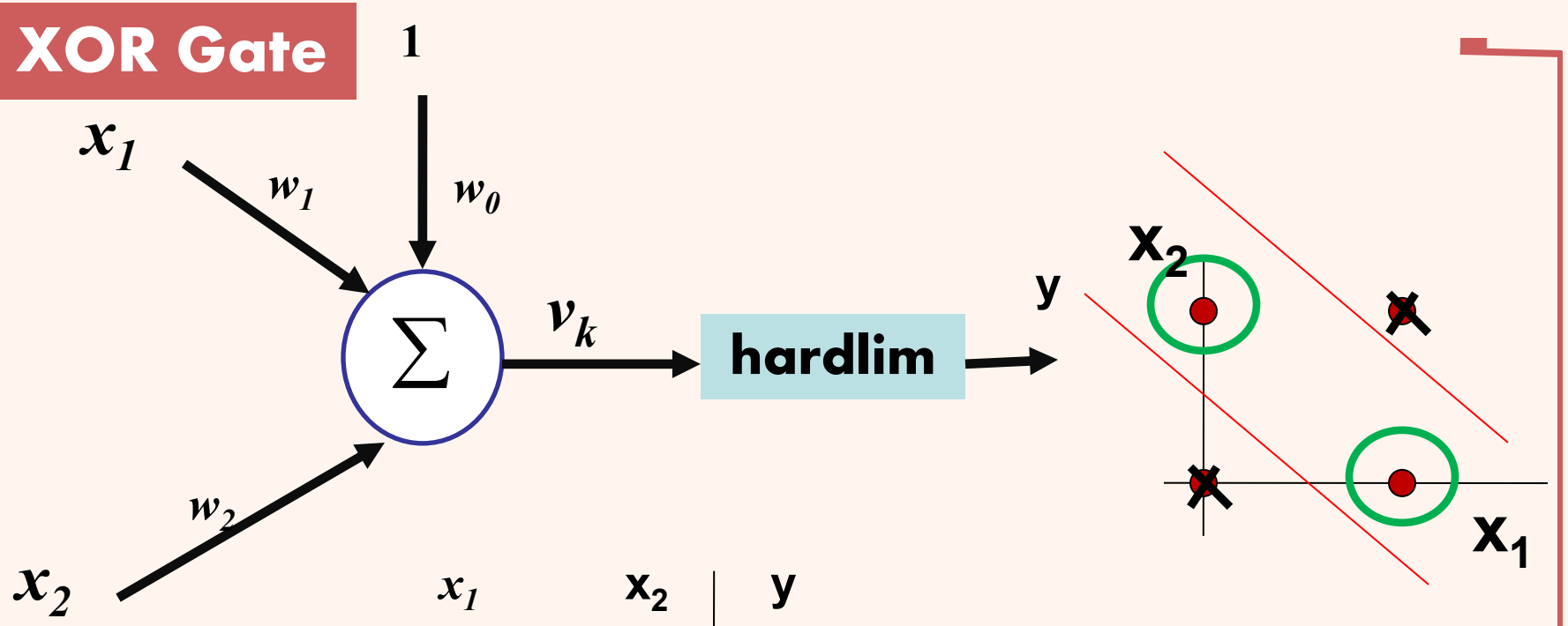
$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{w_0}{w_2}$$

شیب

عرض از مبدا

- به جای تغییر آستانه می‌توان بایاس را تغییر داد.

XOR Gate



x_1	x_2	y
1	1	0
0	0	0
1	0	1
0	1	1

$W_1+ W_2+W_0<0$

$W_0<0$

$W_1+ W_0>0$

$W2+W_0>0$

به وسیله‌ی یک خط جدایی‌پذیر نیستند پس قابلیت جداسازی چنین تابعی را ندارد

هوش مصنوعی و یادگیری

- یادگیری، یکی از مهم‌ترین بخش‌های هوش مصنوعی است. یک سیستم که در محیطی با شرایط متغیر قرار دارد، برای هوشمند بودن باید توانایی آموختن داشته باشد. در چنین حالتی نیازی به پیش‌بینی همه‌ی حالات ممکن نخواهد بود.
- برای حل بسیاری از مسائل در بینایی ماشین، تشخیص صوت و... الگوریتم‌های یادگیری به کار می‌آیند.
  - شناسایی هویت با کمک چهره یکی از این زمینه‌هاست که در «بازشناسی الگو» مطرح می‌شود.

انواع شیوه‌های یادگیری(آموزش)

- یادگیری بانظارت **Supervised learning**
  - یک دسته داده‌ی آموزشی (ورودی و خروجی مطلوب) برای آموزش وجود دارد.
  - کاربردها: درون‌یابی و دسته‌بندی
  - داده‌های آموزشی دارای برچسب هستند.
- یادگیری بی‌نظارت **Unsupervised learning**
  - مجموعه‌ای داده بدون برچسب وجود دارد، هدف یافتن رابطه‌ای بین داده‌هاست.
- یادگیری نیمه‌نظارتی **semisupervised learning**
- یادگیری تقویتی: خروجی مطلوب وجود ندارد، بر اساس یک تابع هزینه یا پاداش شبکه آموزش می‌بینند. **Reinforcement learning**

supervised Learning

یادگیری بانظارت

- در این شیوه همراه با نمونه‌های آموزشی، پاسخ مطلوب هم وجود دارد.
  - پیش‌بینی نمونه‌های جدید
  - استخراج دانش
  - فشرده‌سازی
  - تشخیص نمونه‌های غیرنرمال؛ تشخیص تقلب و سوءاستفاده

Unsupervised Learning

یادگیری بی‌نظارت

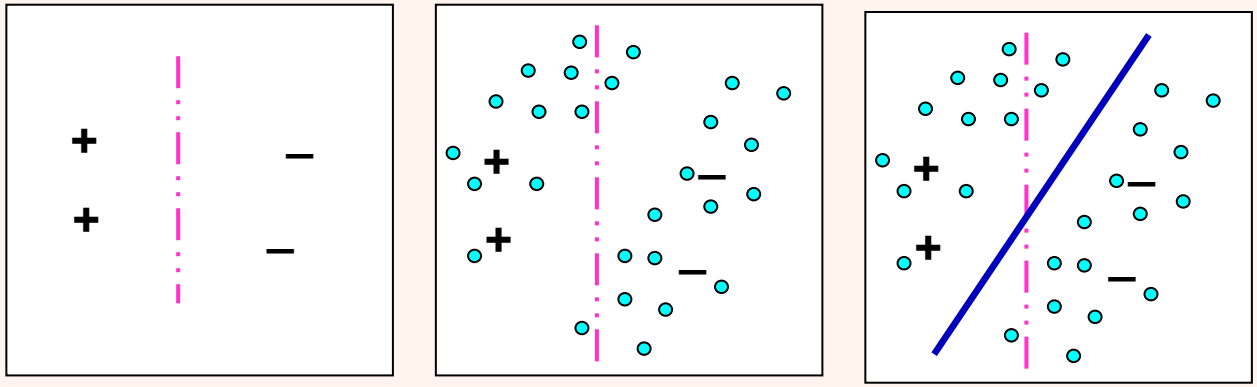
- در این حالت تنها داده‌های ورودی وجود دارند، بدون این که مقدار مطلوب به ازای هر وردی مشخص باشد.
- هدف پیدا کردن «نظم» (regularity) موجود در داده است، آن‌چه معمول و طبیعی است.

Density estimation

- خوشه‌بندی(clustering): گروه‌بندی نمونه‌های مشابه
  - مدیریت ارتباط با مشتری
  - فشرده‌سازی تصویر(چندی‌سازی رنگ)
  - بیوانفورماتیک(Learning motifs)

یادگیری نیمه‌نظارتی

- تنها بخشی از داده‌ها برچسب خورده‌اند، و حجم زیادی از آن بدون برچسب هستند.
- برچسب زدن داده‌ها کار پرهزینه‌ای است.
- از طرفی، داده‌های برچسب نخورده‌ی زیادی در اختیار داریم.



یادگیری تقویتی

- در برخی موارد خروجی یک سیستم، دنباله‌ای از «کنش»هاست. به گونه‌ای که یک حرکت اهمیت ندارد، بلکه سیاستی است که باعث می‌شود مجموع حرکات، به هدف مناسب برسند.
- یک عمل مناسب است در صورتی که در مجموع و در کنار سایر اعمال مناسب باشد. در این حالت الگوریتم یادگیری باید قادر به انتخاب سیاست مناسب باشد.

Game playing
Robot in a maze
Multiple agents, partial observability, ...

ارزیابی الگوریتم‌های یادگیری

- بسته به کاربرد، برای ارزیابی الگوریتم‌های یادگیری، دقت و صحت دسته‌بندی، حجم محاسبات و حافظه‌ی مورد نیاز در نظر گرفته می‌شود.
- شبکه‌های عصبی(الگوریتم‌های یادگیری) متفاوتی وجود دارند؛ بسته به شرایط کاربرد، الگوریتم‌های مختلفی را می‌توان مورد استفاده قرار داد.
- حجم مورد نیاز داده‌های آموزشی، پیچیدگی الگوریتم‌های مورد استفاده و قابلیت تعمیم مسائلی است که باید مورد بررسی قرار گیرند.

- انتخاب وزن‌ها به صورت تصادفی
- اعمال مجموعه‌ی آموزشی (training set)

$$M = \left\{ \left(X^1, d^1 \right), \left(X^2, d^2 \right), \cdots \right.$$

  - اعمال هر ورودی به شبکه و به دست آوردن خروجی
  - مقایسه‌ی خروجی مطلوب و واقعی
  - آموزش شبکه به صورت تغییر وزن‌ها و در جهت نزدیک شدن خروجی مطلوب و واقعی

Hebb 1949

- فرضیه‌ی مطرح شده توسط Hebb در حال حاضر بر روی تحقیقات عصب‌شناسی مؤثر است.
- این فرضیه پیشتر نیز به بیان‌های مختلف مطرح شده بود.

When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased

- در بخش‌های بعدی دوباره با این قانون مواجه خواهیم شد.

قانون آموزش پرسپترون

- مقادیر ورودی ۱ و ۱- هستند.
- تابع فعالیت(انگیزش) پله واحد

$$y(t)=f\left[\sum_i w_i(t)x_i\right]$$

$y(t)$ is correct $\qquad w_i(t+1)=w_i(t)$

$y(t)$ is not correct

$y(t)=-1 \qquad w_i(t+1)=w_i(t)+x_i$

$y(t)=1 \qquad w_i(t+1)=w_i(t)-x_i$

اولین قانون آموزش(ادامه...)

- و بدین شکل در یک رابطه تجمیع شد:

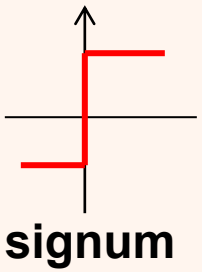
$y(t)$ is correct $\quad w_i(t+1) = w_i(t)$

$y(t)$ is not correct $\quad w_i(t+1) = w_i(t) + d^k x_i^k$

- در صورتی که تابع انگیزش به صورت یکنوا صعودی باشد، بدین ترتیب تغییر وزن‌ها باعث کاهش خطا می‌شود.

مثال ۱

$$X^1=[1 \quad -1 \quad -1 \quad -1], \quad d^1=1$$
$$X^2=[1 \quad 1 \quad -1 \quad -1], \quad d^2=-1$$
$$X^3=[1 \quad 1 \quad 1 \quad 1], \quad d^3=1$$



$t=0, \quad W=[0 \quad 0 \quad 0 \quad 0];$ X^1 $W_{new}=W_{old}+X^1;$

$t=1, \quad W=[1 \quad -1 \quad -1 \quad -1];$ X^2 $W_{new}=W_{old}-X^2;$

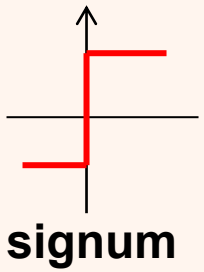
$t=2, \quad W=[0 \quad -2 \quad 0 \quad 0];$ X^3 $W_{new}=W_{old}+X^3;$

$t=3, \quad W=[1 \quad -1 \quad 1 \quad 1];$ X^1 $W_{new}=W_{old}+X^1;$

$t=4, \quad W=[2 \quad -2 \quad 0 \quad 0];$ X^2 $W_{new}=W_{old}-X^2;$

مثال ۲

$$X^1=[1 \quad -1 \quad -1 \quad -1], \quad d^1=1$$
$$X^2=[1 \quad 1 \quad -1 \quad -1], \quad d^2=-1$$
$$X^3=[1 \quad 1 \quad 1 \quad 1], \quad d^3=1$$



$t=5, \quad W=[1 \quad -3 \quad 1 \quad 1]; \quad X^3$  $W_{new}=W_{old}+X^3;$

$t=6, \quad W=[2 \quad -2 \quad -1 \quad 2]; \quad X^1$  $W_{new}=W_{old}+X^1;$

$t=7, \quad W=[3 \quad -3 \quad 1 \quad 1]; \quad X^2$  $W_{new}=W_{old}$

$t=8, \quad W=[3 \quad -3 \quad 1 \quad 1]; \quad X^3$  $W_{new}=W_{old}$

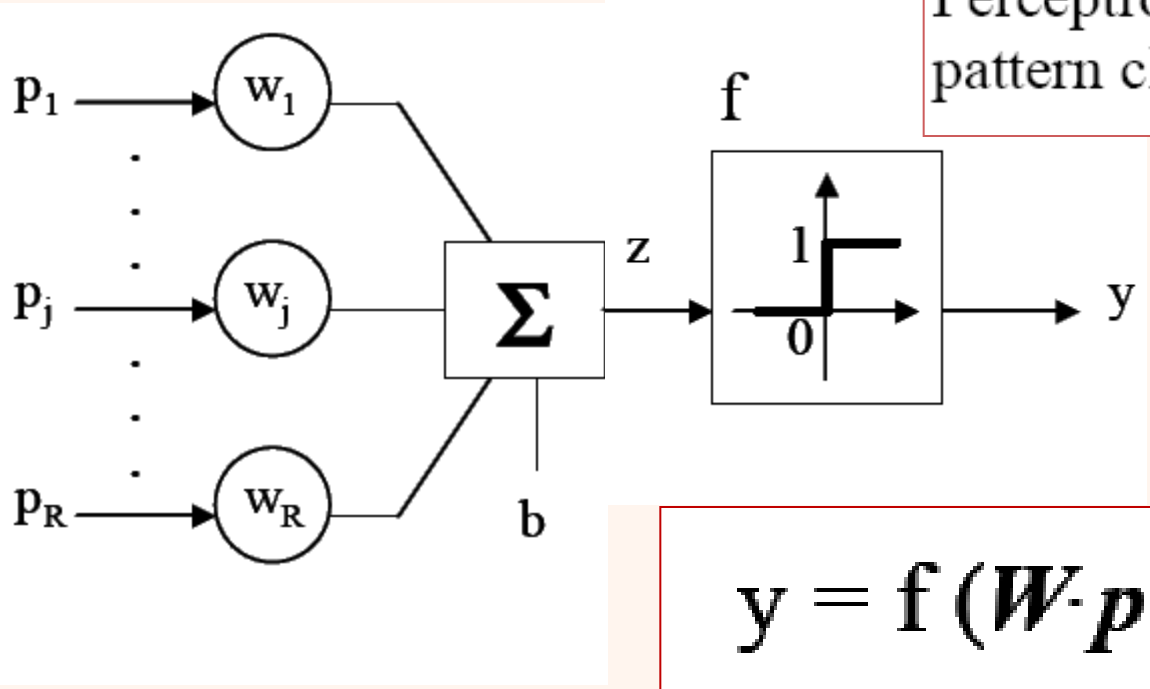
$t=9, \quad W=[3 \quad -3 \quad 1 \quad 1]; \quad X^1$  $W_{new}=W_{old}$

جمع‌بندی

Frank Rosenblatt (1958), Marvin Minski & Seymour Papert (1969)

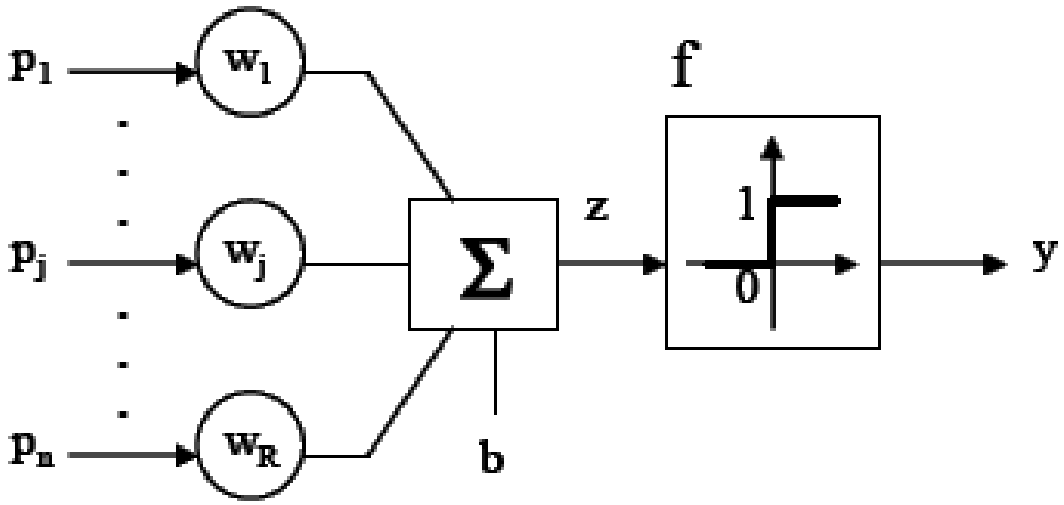
- پرسپترون نرونی است با تابع انگیزش دوسطحی که با توجه به قانون یادگیری (Learning rules) وزن‌ها و بایاس آن به روز می‌شود.

Perceptrons are well suited for pattern classification/recognition.



$$y = f(W \cdot p + b)$$

یادگیری پرسپترون



- بانظارت (ised

$$p = (p_1, \cdots, p_R)^T$$

$$W = (x_1, \cdots, x_R)$$

اگر t خروجی مطلوب باشد برای خطا داریم:

$$e = t - y$$

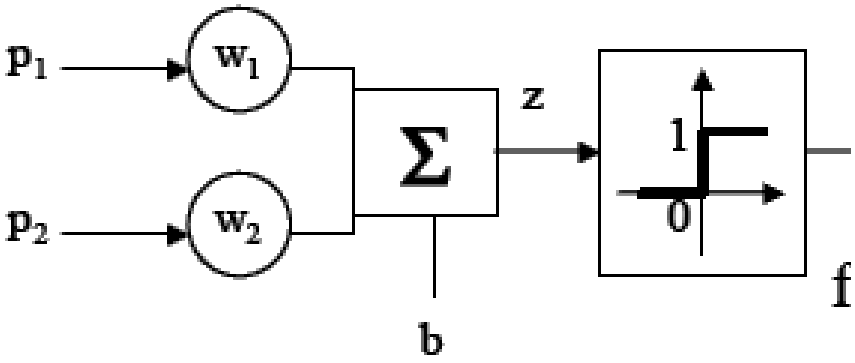
$$\text{if } e = 1, \text{ then } W^{new} = W^{old} + p \,,\; b^{new} = b^{old} + 1;$$

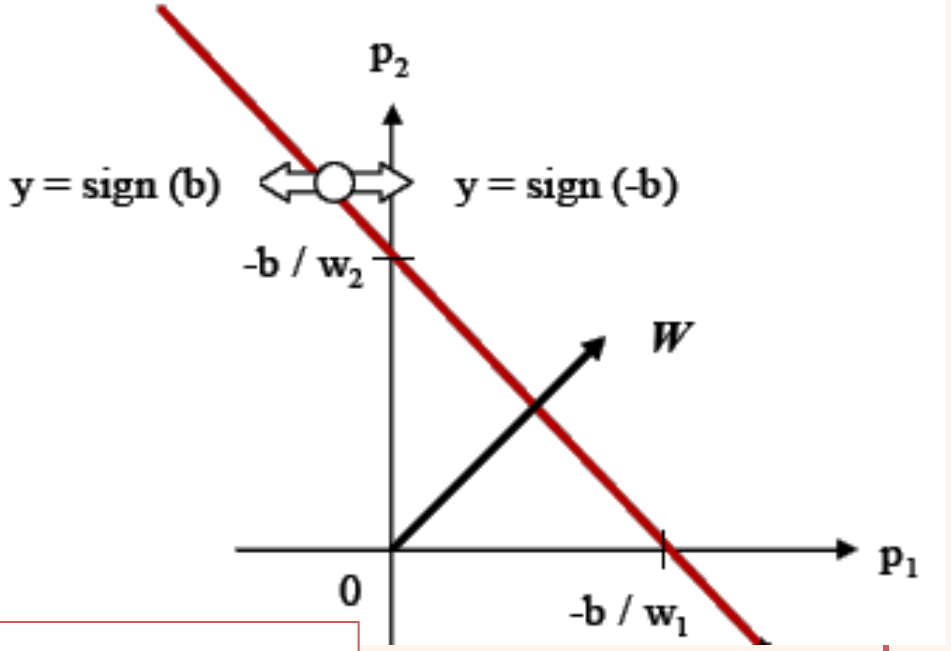
$$\text{if } e = -1, \text{ then } W^{new} = W^{old} - p \,,\; b^{new} = b^{old} - 1;$$

$$\text{if } e = 0, \text{ then } W^{new} = W^{old}.$$

Perceptron learning rule

Two-Input Perceptron





$$y = hardlim\ (z) = hardlim\{ [w_1 , w_2] \cdot [p_1 , p_2]^T + b\}$$

$$w_1 \cdot p_1 + w_2 \cdot p_2 + b = 0$$

مرز همواره بر بردار وزن عمود است

Convergence theorem

قضیه‌ی همگرایی

- در صورت که مجموعه وزن‌های W^* وجود داشته باشد که قابلیت جداسازی یک مجموعه‌ی محدود (جدایی‌پذیر خطی) را داشته باشد، قانون آموزش پرسپترون به یک پاسخ همگرا خواهد شد.
  - این پاسخ الزاما با W^* یکسان نخواهد بود.
  - تمام خروجی‌ها را به گونه‌ای تغییر می‌دهیم که خروجی مطلوب «۱+» شود.
  - وزن اولیه را صفر در نظر می‌گیریم.
  - بردار ورودی n-تایی است.

$$X^k = [1, x_1^k, x_2^k, \ldots\ldots, x_n^k]$$

اثبات قضیه‌ی همگرایی

- هدف محاسبه‌ی حداکثر تعداد مراحلی است که وزن‌ها باید اصلاح شوند. با توجه به مفروضات

$$\forall k,\ \exists \delta \geq 0 \quad W^{*}.X^{k} \geq \delta,$$

- فرض کنید در مرحله‌ی $t+1$ نیاز به اصلاح وزن‌ها وجود دارد:

$$W_{(t+1)} = W_{(t)} + d^{k} X^{k}$$

$$W^{*}.W_{(t+1)} = W^{*}.W_{(t)} + W^{*} X^{k}$$

$$W^{*}W_{(t+1)} \geq W^{*}W_{(t)} + \delta \Rightarrow \boxed{W^{*}W_{(t)} \geq t\,\delta}$$

اثبات قضیه‌ی همگرایی(ادامه ...)

$$\left\|W_{(t+1)}\right\|^2 = W_{(t+1)}W_{(t+1)}^T = \left[W_{(t)} + d^k X^k\right]\left[W_{(t)} + d^k X^k\right]^T$$

این مقدار منفی است

$$\left\|W_{(t+1)}\right\|^2 = \left\|W_{(t)}\right\|^2 + \left\|X^k\right\|^2 + 2W_{(t)}\left[X^k\right]^T$$

$$\left\|W_{(t+1)}\right\|^2 \le \left\|W_{(t)}\right\|^2 + \left\|X^k\right\|^2$$

(n+1)

$$\left\|W_{(t+1)}\right\|^2 \le \left\|W_{(t)}\right\|^2 + (n+1) \Rightarrow \left\|W_{(t)}\right\|^2 \le t(n+1)$$

اثبات قضیه‌ی همگرایی(ادامه ...)

$$W^{*}.W_{(t)} \geq t\delta$$

$$\cos(\theta) = \frac{W^{*}W_{(t)}}{\|W^{*}\|\|W_{(t)}\|} \leq 1$$

$$\|W^{*}\|\|W_{(t)}\| \geq t\,\delta$$

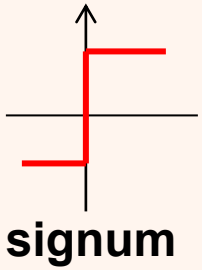
$$\|W_{(t)}\|^{2} \leq t\,(n+1)$$

$$\|W^{*}\|\sqrt{t\,(n+1)} \geq t\,\delta$$

$$t \leq \frac{\|W^{*}\|^{2}\,(n+1)}{\delta^{2}}$$

مثال

$$X^1 = [1 \quad -1 \quad -1 \quad -1], \quad d^1 = 1$$
$$X^2 = [1 \quad 1 \quad -1 \quad -1], \quad d^2 = -1$$
$$X^3 = [1 \quad 1 \quad 1 \quad 1], \quad d^3 = 1$$



$$W^* = [3 \quad -3 \quad 1 \quad 1];$$

$$t \le \frac{\|W^*\|^2 (n+1)}{\delta^2}$$

$$t \le \frac{20 \times 4}{\delta^2}$$

LMS(Least Mean Square)

1960

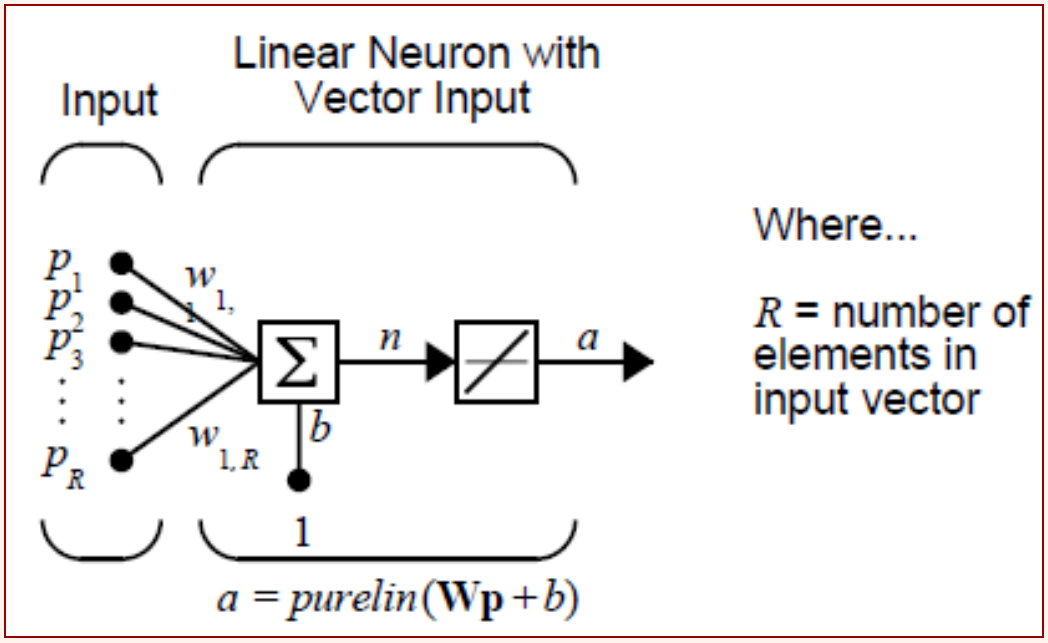
Widrow and his graduate student Hoff introduced ADALINE network and learning rule which they called the LMS(Least Mean Square) Algorithm.

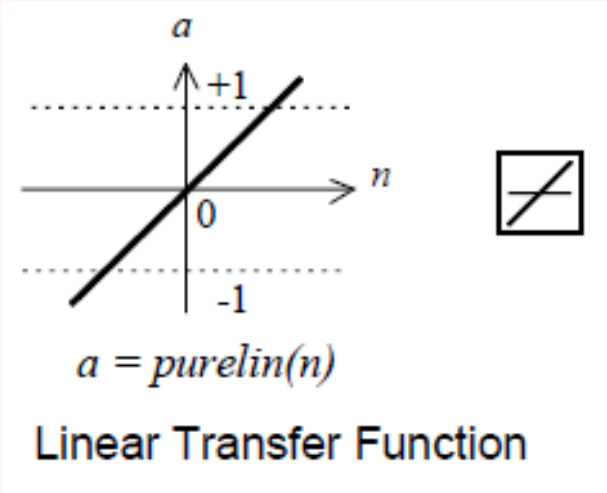
$$w_{new}=w_{old}+\Delta w$$

- برای تولید وزن‌های جدید از تأثیر خطا استفاده می‌شود.
- در این شیوه میزان *به‌روزنمایی* متناسب با میزان خطا خواهد بود و در نتیجه همگرایی سریع‌تر صورت می‌گیرد.

ADALINE

- ADALINE همانند پرسپترون است تنها تابع آن به جای دوسطحی بودن (که مقادیر ۱ و ۱-(0) را به خود اختصاص می‌دهد) تابعی خطی است.
- ADALINE همانند پرسپترون می‌تواند مسائل جدایی‌پذیر خطی را حل کند.





ADALINE

supervised training

$$W_{new}=W_{old}+\Delta W$$

خروجی مطلوب ورودی kام

خروجی واقعی در مرحله‌ی n به ازای ورودی k–ام

$$e_k(n)=d^k-y^k(n)$$

تغییر وزن‌ها در جهت افزایش خروجی ⟸ $e_k(n)>0$

تغییر وزن‌ها در جهت کاهش خروجی ⟸ $e_k(n)<0$

$$w_i(n+1)=w_i(n)+\eta e^k(n)x_i^k$$

ضریب آموزش (یادگیری)

ADALINE

- با فرض این که واحد خروجی دارای تابع انگیزش خطی باشد.
- به ازای N ورودی مسأله را بررسی می‌کنیم.

$$X^1 \xrightarrow{W(1)} y^1 \qquad e_1 \quad \text{will be generated}$$

$$X^2 \xrightarrow{W(1)} y^2 \qquad e_2 \quad "$$

$$X^3 \xrightarrow{W(1)} y^3 \qquad e_3 \quad "$$

$$\vdots \qquad\qquad \vdots$$

$$X^N \xrightarrow{W(1)} y^N \qquad e_N \quad "$$

$$E = \sum_{i=1}^{N} [e_i]^2$$

SSE

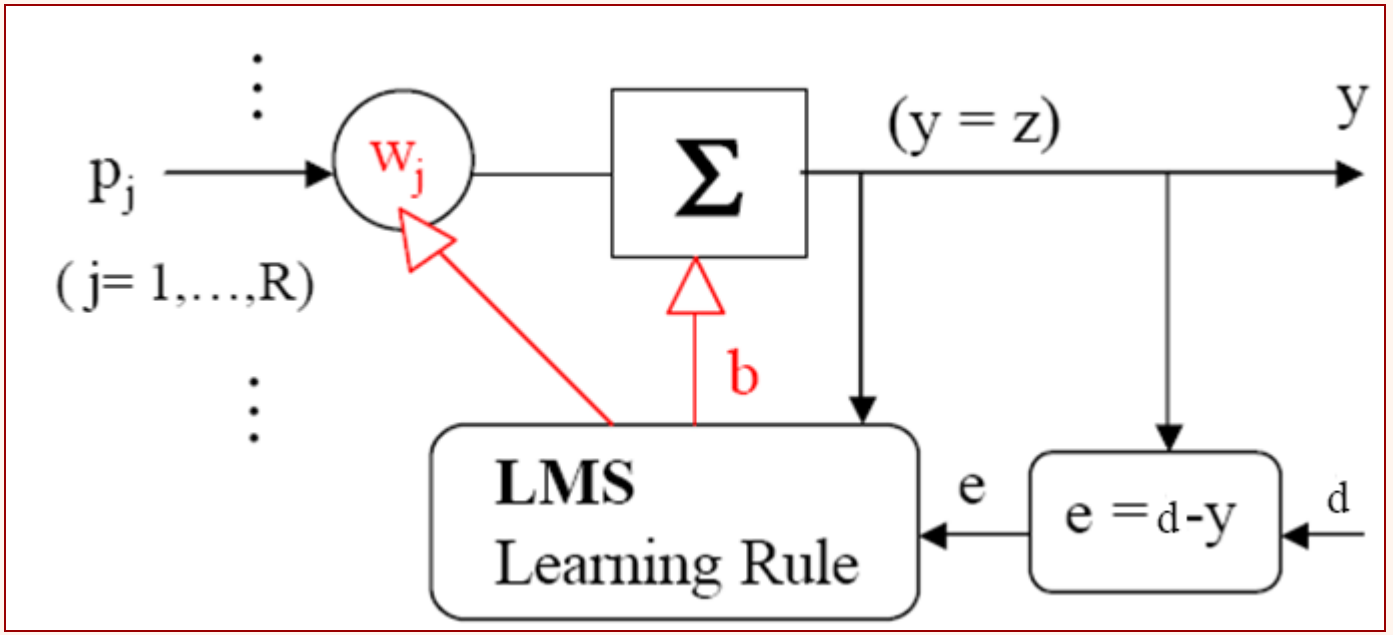
$$E = \frac{\sum_{i=1}^{N} [e_i]^2}{N}$$

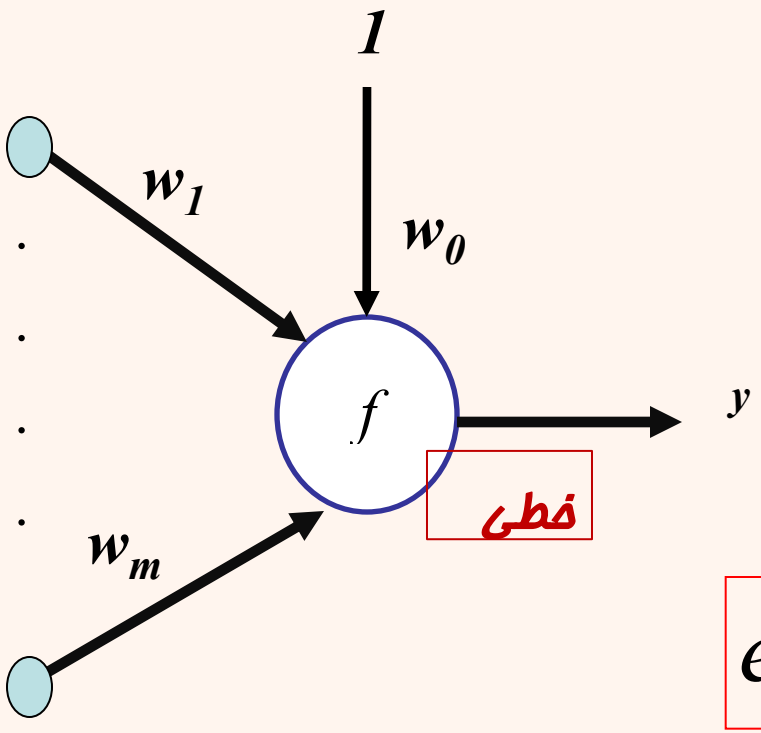
Mean SSE

Widrow-Hoff Learning Rule

LMS(Least Mean Square)

- الگوریتم LMS وزن‌ها و بایاس را به گونه‌ای تغییر می‌دهد که میانگین مربعات خطا (بین خروجی مطلوب و خروجی واقعی) سیستم را به حداقل برساند.





خطای به‌دست آمده به ازای ورودی X^k

$$e_k(n) = d^k - W(n)X^k$$

$$X^k = [1, x_1^k, \ldots\ldots, x_m^k]^T$$

$$\mathbf{X} = [X^1, X^2, \ldots\ldots, X^N]_{(m+1)\times N}$$

N ورودی m تایی

$$D = [d^1, d^2, \ldots\ldots, d^N]_{1\times N}$$

$$W = [w_0, w_1, \ldots\ldots, w_m]_{1\times(m+1)}$$

$$X^k = [1, x_1^k, ..., x_m^k]^T$$

$$\mathbf{X} = [X^1, X^2, ..., X^N]_{(m+1)\times N}$$

$$D = [d^1, d^2, ..., d^N]_{1\times N}$$

$$W = [w_0, w_1, ..., w_m]_{1\times(m+1)}$$

$$e_k(n) = d^k - W(n)X^k$$

Batch Mode

$$SSE = E(n) = \sum_{k=1}^{N} (d^k - W(n)X^k)^2$$

Number of epoch

$$E(n) = \| D - W(n)\mathbf{X} \|^2$$

Y(n)

$E(W(n))$ **پارامتر آزاد برای تابع خطا وزن‌ها هستند.**

$$Y(n) = [y^1(n), y^2(n), ..., y^N(n)]$$

کمینه کردن خطا

- باید به گونه‌ای عمل کرد که تابع خطا طی فرآیند آموزش کم‌تر شود:

$$E(n+1) < E(n) \quad \text{یا} \quad E(W(n)) < E(W(n+1))$$

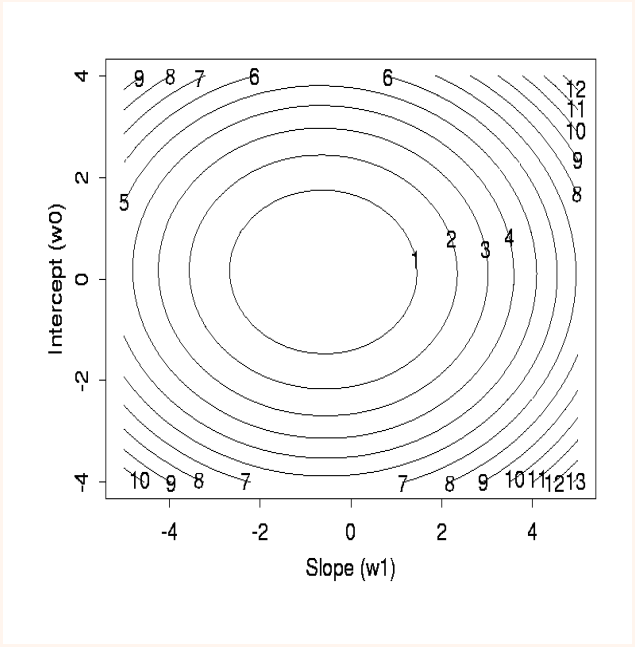
- هدف یافتن وزن بهینه‌ای است که به ازی آن تابع خطا(هزینه) مینیمم شود:

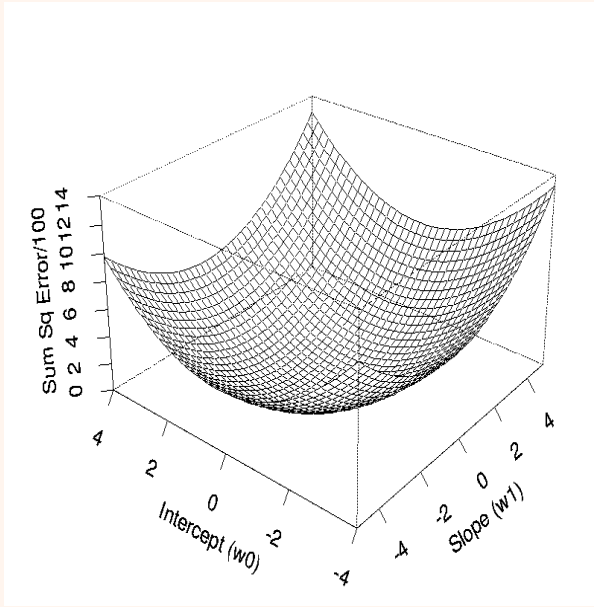
$$E(W^*) \leq E(W)$$

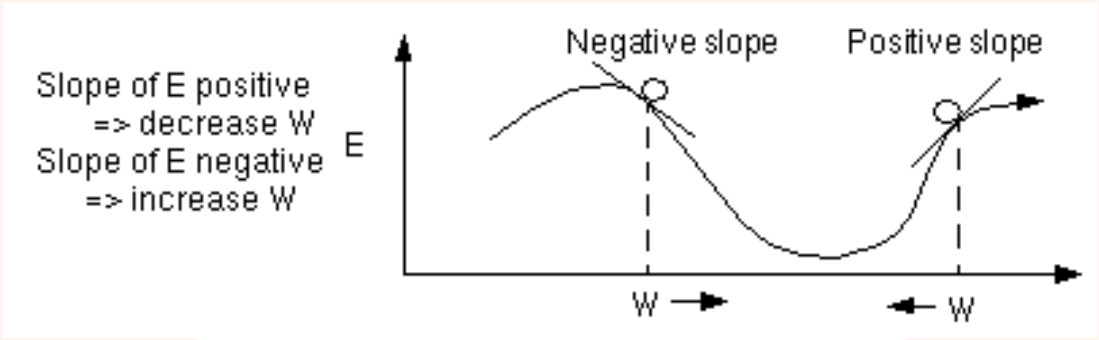
- شرط لازم برای وجود وزن بهینه این است که:

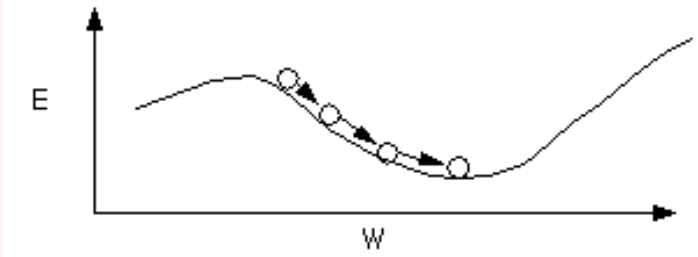
$$\nabla E(W^*) = 0$$

Steepest descent کمینه کردن خطا(ادامه...)









- هدف به حداقل رساندن مقدار E یا S.S.E است:

$$\nabla_w E_{(n)} = [\frac{\partial E(n)}{\partial w_0(n)}, \frac{\partial E(n)}{\partial w_1(n)}, \ldots\ldots\ldots, \frac{\partial E(n)}{\partial w_m(n)}]$$

$$SSE = E(n) = \sum_{k=1}^{N} (d^k - W(n)X^k)^2$$

داشتیم

Batch Mode

$$\frac{\partial E(n)}{\partial w_i(n)} = -2\sum_{k=1}^{N} (d^k - y^k(n)) \frac{\partial y^k(n)}{\partial w_i(n)}$$

$$\frac{\partial E(n)}{\partial w_i(n)} = -2\sum_{k=1}^{N}(d_k - y_k(n))\frac{\partial y_k(n)}{\partial w_i(n)}$$

$$\frac{\partial E(n)}{\partial w_i(n)} = -2\sum_{k=1}^{N}(d_k - y_k(n))x_i^k$$

$$= -2(D - Y(n))[\mathbf{X}_i]^T \qquad \mathbf{X}_i = [x_i^1, x_i^2, \ldots, x_i^N]$$

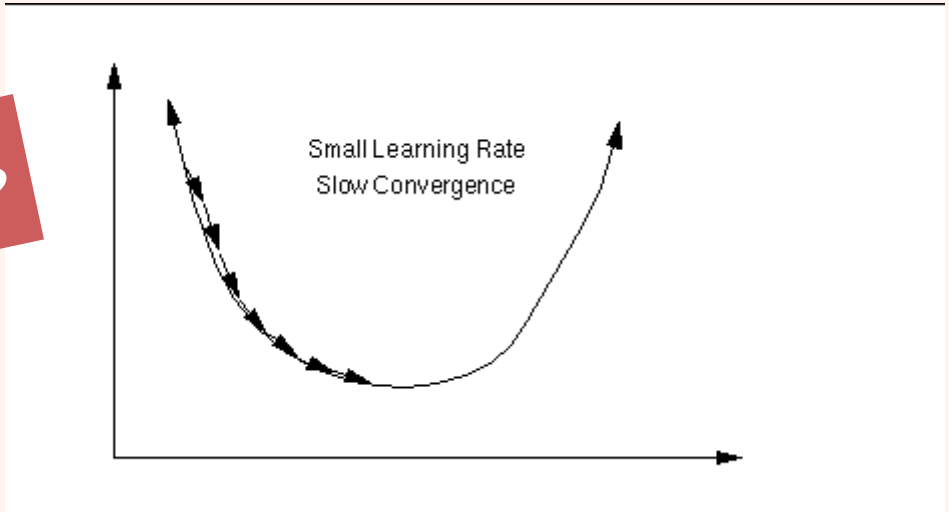
- برای انتخاب w مطلوب

$$w_i(n+1) = w_i(n) - \eta\frac{\partial E(n)}{\partial w_i(n)}$$

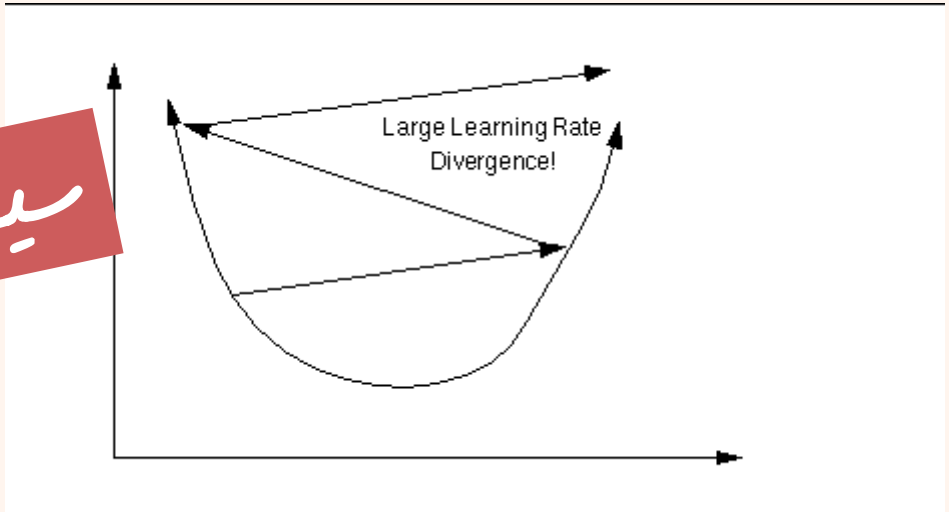
$$w_i(n+1) = w_i(n) + 2\eta(D - y(n))[\mathbf{X}_i]^T$$

تنظیم نرخ یادگیری

همگرایی کند است.



سیستم ناپایدار است.



به دست آوردن محدوده نرخ آموزش

- نرخ آموزش «پایداری» و «سرعت همگرایی» را مشخص می‌کند.

با فرض تک الگو

$$E_{(t+1)} = \left\| D - W_{(t+1)} X \right\|^2$$

$$E_{(t+1)} = \left\| D - \left[W_{(t)} + \eta \left(D - Y_{(t)} \right) X^T \right] X \right\|^2$$

$$E_{(t+1)} = \left\| D - W_{(t)} X - \eta \left(D - Y_{(t)} \right) \| X \|^2 \right\|^2$$

$$E_{(t+1)} = E_{(t)} + \eta^2 \left\| D - Y_{(t)} \right\|^2 \left(\| X \|^2 \right)^2 - 2\eta \left\| \left(D - Y_{(t)} \right) \right\|^2 \| X \|^2$$

به دست آوردن محدوده نرخ آموزش(ادامه...)

$$E_{(t+1)} = E_{(t)} + \eta^2 \left\| D - Y_{(t)} \right\|^2 \left(\| X \|^2 \right)^2 - 2\eta \left\| \left(D - Y_{(t)} \right) \right\|^2 \| X \|^2$$

$$E_{(t+1)} = E_{(t)} \left[1 + \eta^2 \left(\| X \|^2 \right)^2 - 2\eta \| X \|^2 \right]$$

$$E_{(t+1)} = E_{(t)} \left[1 - \eta \| X \|^2 \right]^2$$

$$\frac{E_{(t+1)}}{E_{(t)}} = \left[1 - \eta \| X \|^2 \right]^2 < 1$$

$$-1 < 1 - \eta \| X \|^2 < 1$$

به دست آوردن محدوده نرخ آموزش(ادامه...)

$$-1 < 1 - \eta \|X\|^2 < 1$$

$$0 < \eta \|X\|^2 < 2$$

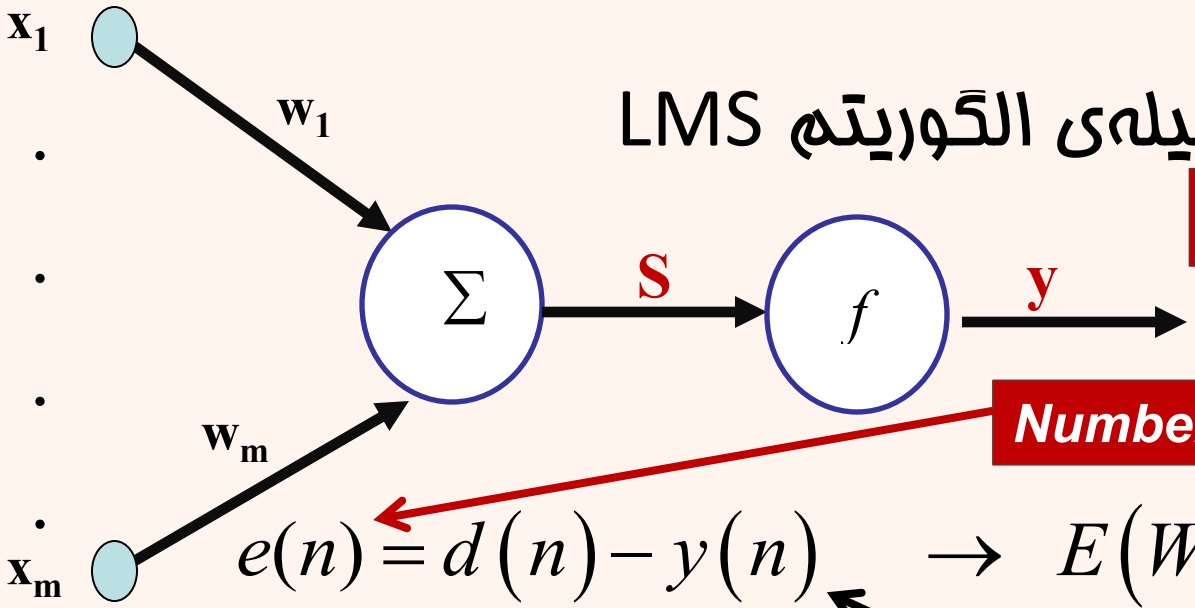
$$0 < \eta < \frac{2}{\|X\|^2}$$

$$0 < \eta < \frac{2}{\max_k \|X^k\|^2}$$

$$\frac{0.1}{\max_k \|X^k\|^2} < \eta < \frac{2}{\max_k \|X^k\|^2}$$

به صورت تجربی

تک‌لایه تک‌واحد با تابع غیر خطی



- حل به وسیله‌ی الگوریتم LMS

Sequential Mode

Number of iteration

$$e(n)=d(n)-y(n) \quad \rightarrow \quad E(W(n))=\frac{1}{2}e^2(n)$$

خروجی به ازای ورودی در تکرار nام

$$w_k(n+1)=w_k(n)-\eta\frac{\partial E}{\partial w_k}$$

$$\frac{\partial E}{\partial w_k}=\frac{1}{2}\frac{\partial e^2}{\partial w_k}=e\frac{\partial e}{\partial w_k}=e\frac{\partial e}{\partial y}\cdot\frac{\partial y}{\partial w_k}$$

$$=e\frac{\partial e}{\partial y}\cdot\frac{\partial y}{\partial s}\cdot\frac{\partial s}{\partial w_k}$$

$$\frac{\partial E}{\partial w_k} = e\frac{\partial e}{\partial y}\frac{\partial y}{\partial s}\frac{\partial s}{\partial w_k}$$

-1

$$y = f(s) \quad \rightarrow \quad \frac{\partial y}{\partial s} = f'(s) \qquad x_k$$

$$= -ef'(s)x_k$$

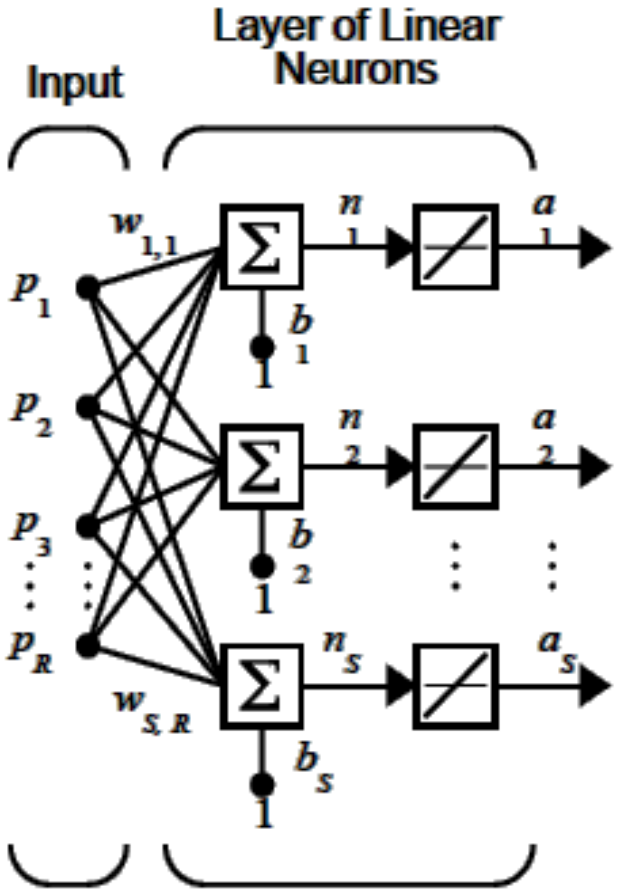
تابع انگیزش باید مشتق پذیر باشد

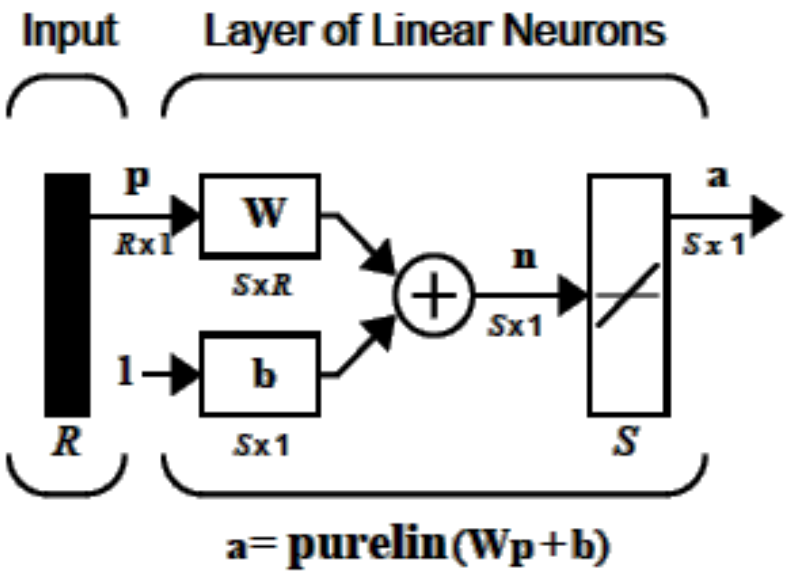
$$w_k(n+1) = w_k(n) + (d - y(n))\, x_k$$

بسته به تابع f متفاوت است

شبکه‌ی تک‌لایه با چند خروجی

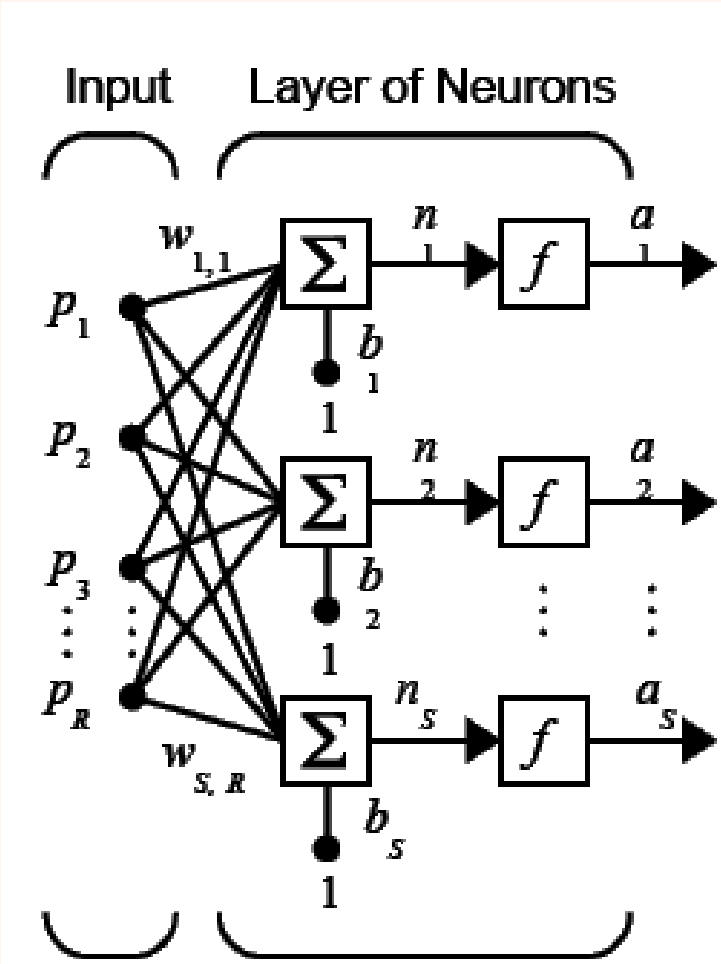
Single-Layer Linear Network





Where... R = numberof elements in input vector

S = numberof neurons in layer



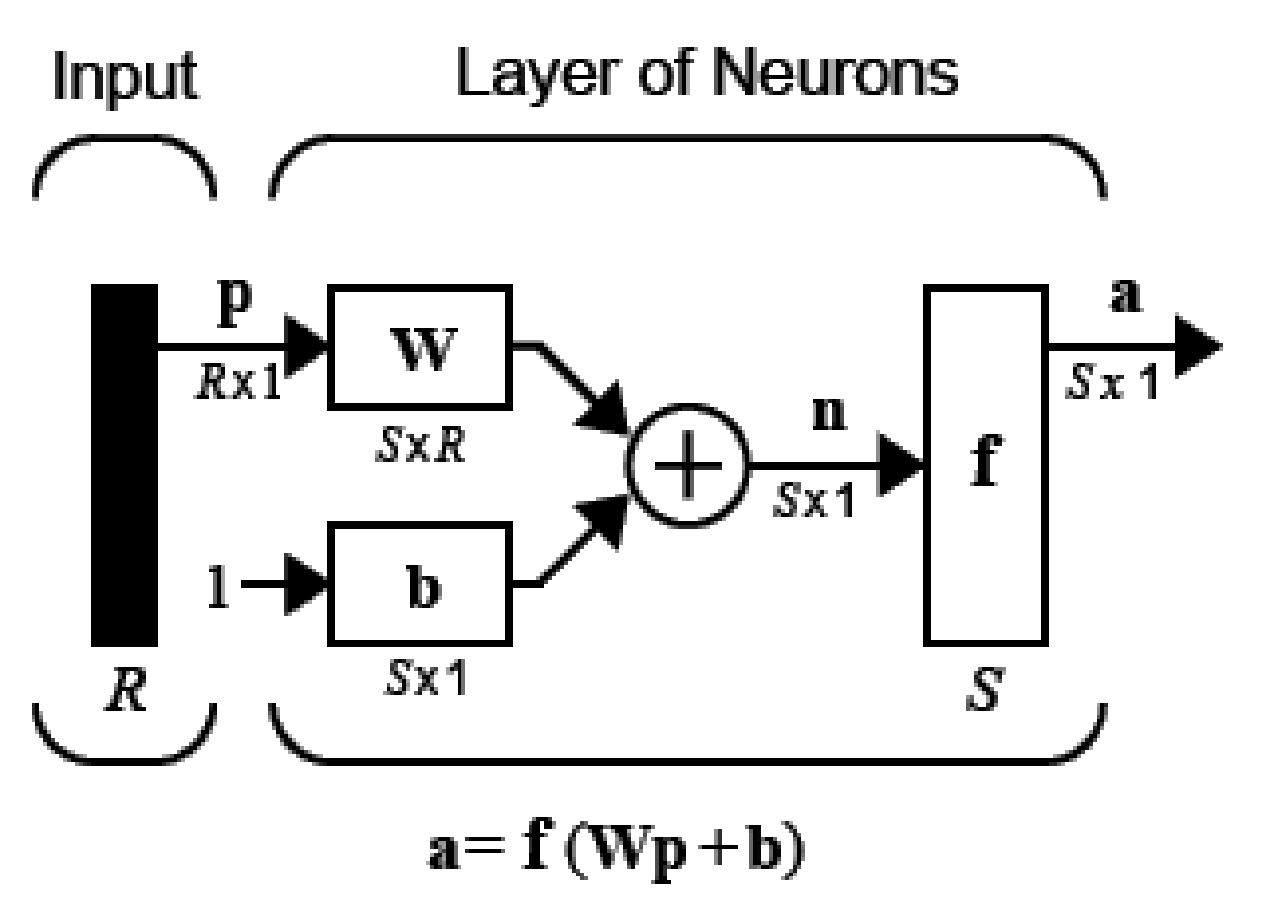
R تعداد المان های ورودی

S تعداد نرون های موجود در یک لایه

$$\mathbf{a} = \mathbf{f}(\mathbf{Wp} + \mathbf{b})$$

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,R} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,R} \\ & & & \\ w_{S,1} & w_{S,2} & \cdots & w_{S,R} \end{bmatrix}$$

شمای شبکه‌ی قبل به اختصار



MATLAB

مثال

- می‌خواهیم پنج داده‌ی زیر را که خروجی‌های مطلوب آن‌ها نیز مشخص است را در دو کلاس طبقه‌بندی کنیم:

```
P1=[0.7,0.2];
T1=[1];
```

```
P2=[-0.1,0.9];
T2=[1];
```

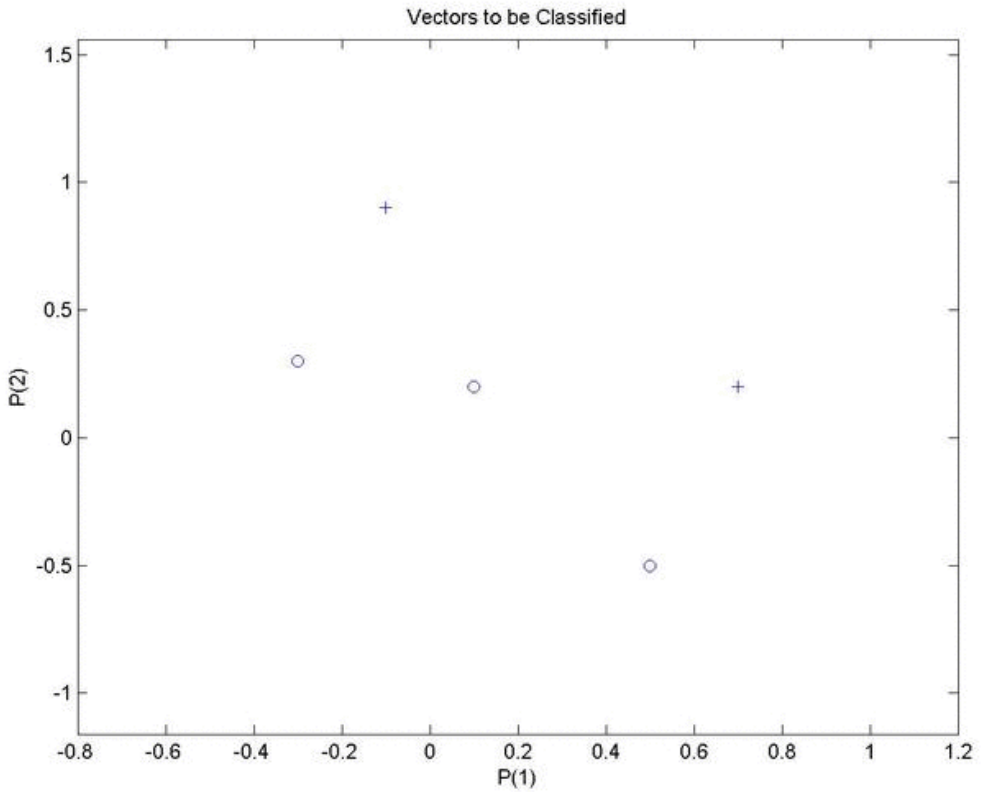
```
P3=[-0.3,0.3];
T3=[0];
```

```
P4=[0.1,0.2];
T4=[0];
```

```
P5=[0.5,-0.5];
T5=[0];
```

```
P=[0.7 -0.1 -0.3 0.1  0.5;
   0.2  0.9  0.3 0.2 -0.5];
T=[1 1 0 0 0];
```

```
P=[0.7 -0.1 -0.3 0.1 0.5;
   0.2 0.9 0.3 0.2 -0.5];
T=[1 1 0 0 0];
W=[0 0];
b=-1;
plotpv(P,T);
plotpc(W,b);
nepoc=0
Y=hardlim(W*P+b);
while any(Y~=T)
    Y=hardlim(W*P+b);
    E=T-Y;
    dW=E*P';
    db=sum(E);
    W=W+dW;
    b=b+db;
    nepoc=nepoc+1;
    disp('epochs='),disp(nepoc),
    disp(W), disp(b);
    plotpv(P,T);
    plotpc(W,b);
end
```

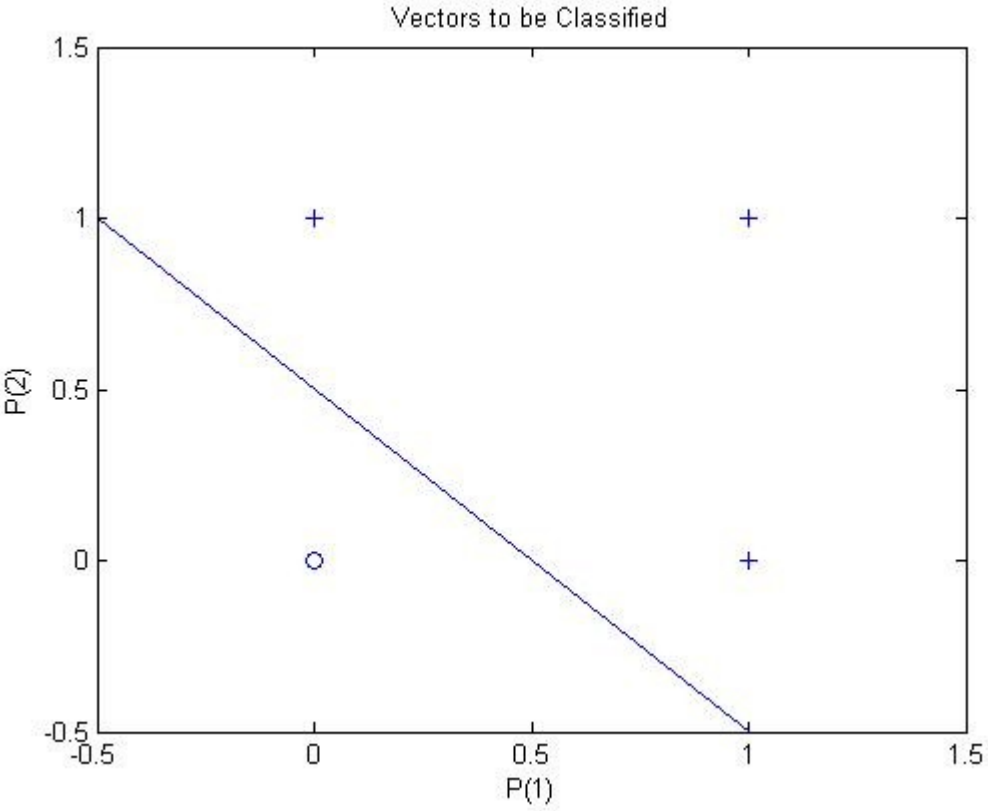



```
[dW,db]= learnp(P,E);
```

Epoch=9

W1=2.7 W2=2.9
B=-2

OR

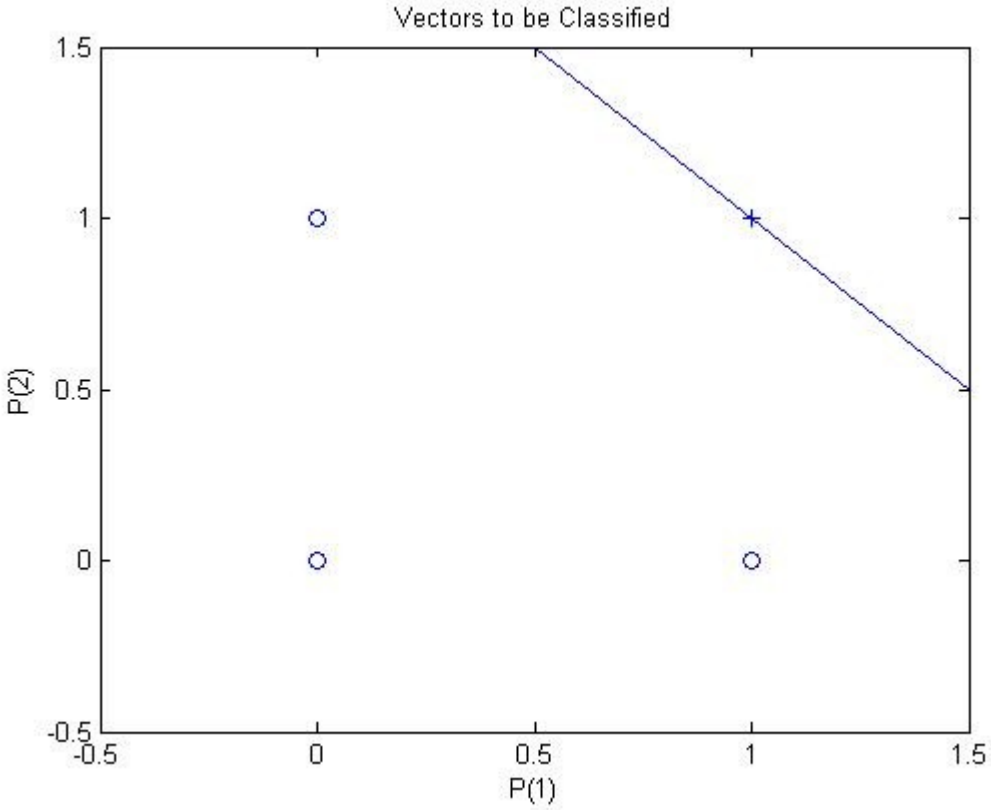


Epochs= 5

W1=2 W2= 2

b= -1

AND

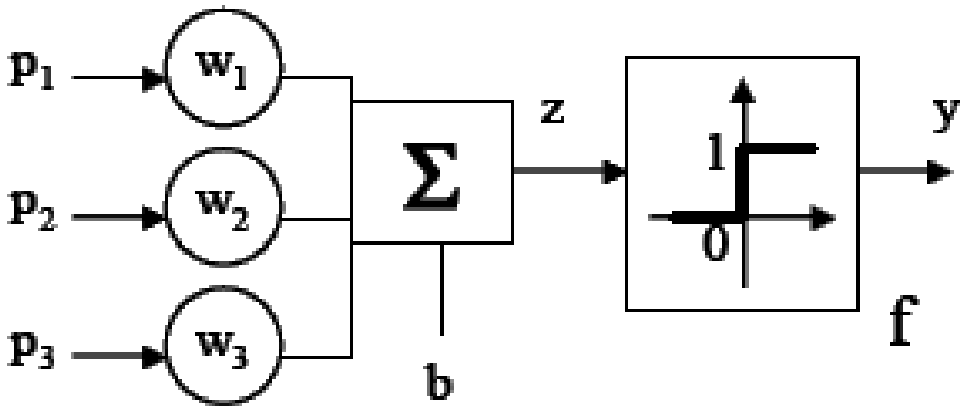


epochs= 4

W1=1 w2= 1

b= -2

پرسپترون سه ورودی



$$y = \text{hardlim}(z) = \text{hardlim}([w_1, w_2, w_3].[p_1, p_2, p_3]^T + b)$$

epochs=
3

w1= 3 w2= -3 w3= 3

b=0