神经元和感知机

高　阳，李文斌

http://cs.nju.edu.cn/rl

2023年11月07日

大纲

脑和神经元

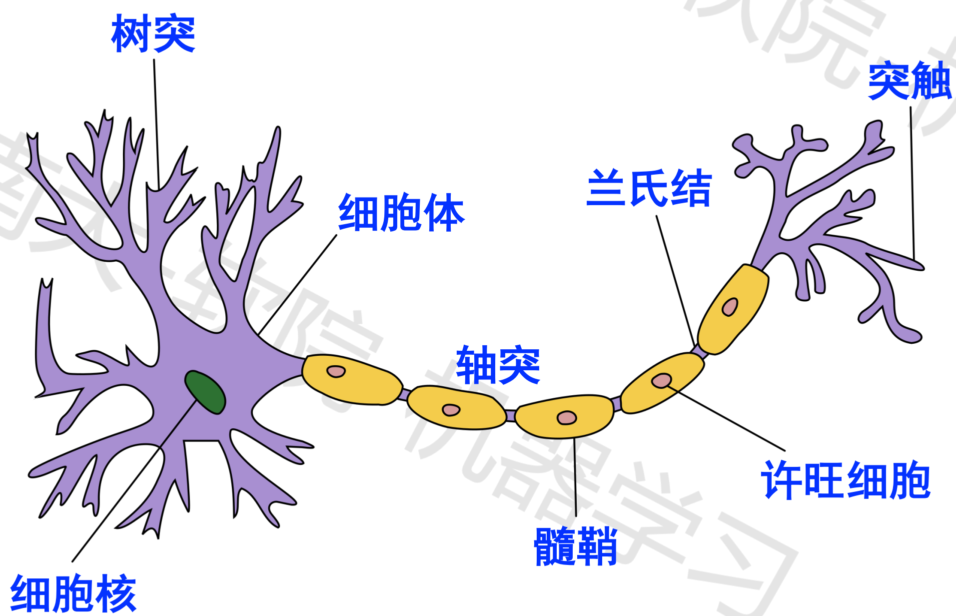
感知机学习

线性可分性

生物学基础

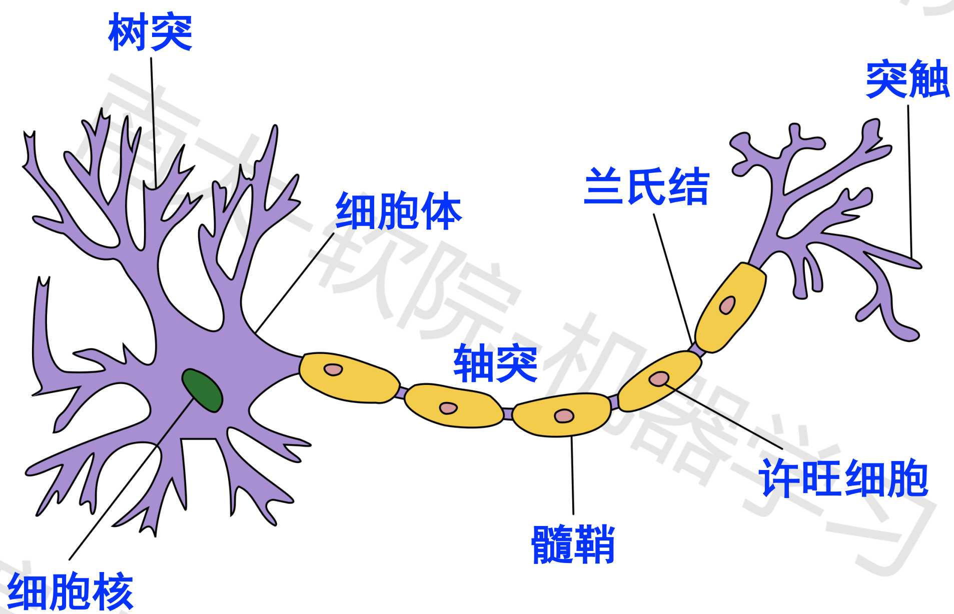
树突： 输入

轴突： 输出



- ✓ 神经元内化学物质调节内部电位
- ✓ 跨膜电位达到一个阈值时，则激活或放电
- ✓ (固定)时间和强度的脉冲传递给轴突
- ✓ 轴突像树枝状，连接到突触

生物学基础



- ✓ 分布/并行计算模型
- ✓ 调整节点间的连接关系

❑ **人脑**

- ✓ 10^{11}个神经元（100 billion）
- ✓ 每个神经元处理速度为10^{-3}s
- ✓ 识别人脸，需要约100个神经元计算

赫布理论Hebbian Theory

- **唐纳德·赫布（1904-1985）于1949年提出**
  - 加拿大生理心理学家
  - 连接强度调整量与输入输出的乘积成正比，显然经常出现的模式将增强神经元之间的连接
  - 与巴普洛夫的‘条件反射’一致
  - 又称为“长程增强机制”(LTP, Long Term Potentiation)

    或“神经可塑”(Neural Plasticity)

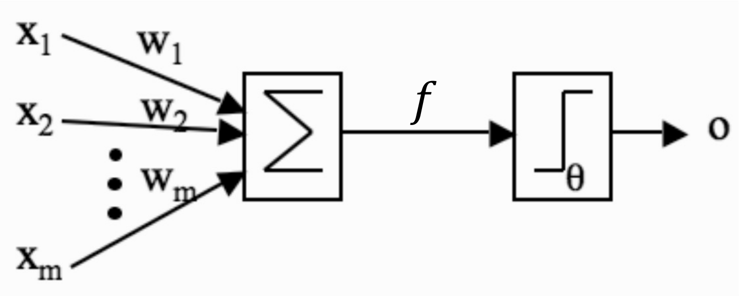
早期发展历史

- 1943年，心理学家W. McCulloch和数理逻辑学家W. Pitts首次提出神经元的数学模型MP模型
- 1948年，冯·诺依曼提出相互再生自动机网络结构
- 1950s，F. Rosenblatt提出感知机模型
- 1960s，Widrow提出非线性多层自适应网络
- 1969年，Minsky出版《感知机》一书
- 1982年和1984年，物理学家Hopfield在美国科学院院刊上发表ANN文章
- 2006年，Hinton发表深度信念网络→深度学习

。。。

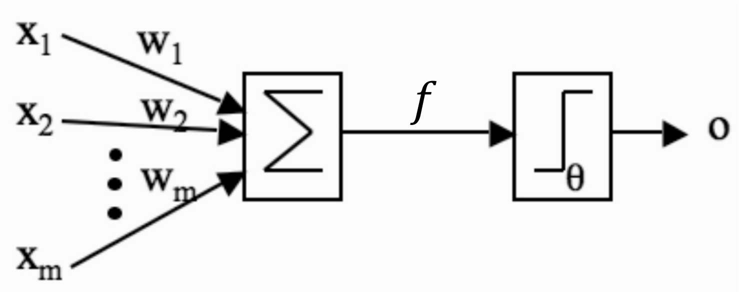
MP神经元基本结构

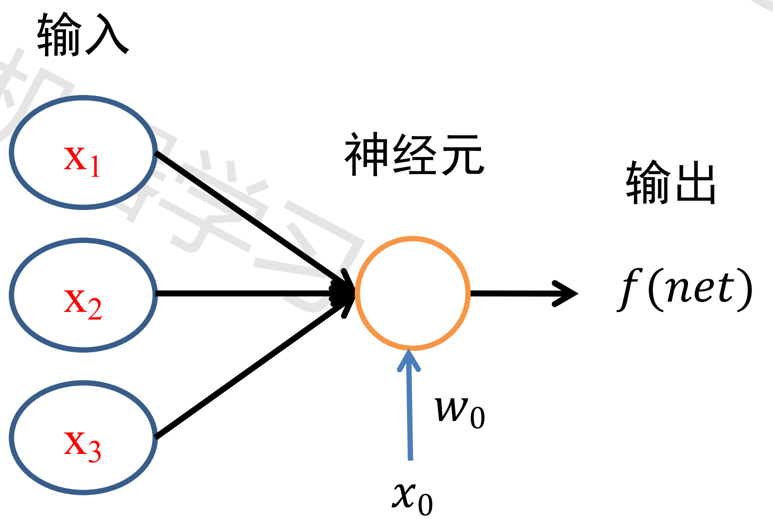
- ❑ 输入$X = [x_1, x_2, x_3, \ldots]$
- ❑ 权值$W = [w_1, w_2, w_3, \ldots]$
- ❑ 激活函数$f(net) = f(\sum(w_i * x_i))$



MP神经元基本结构

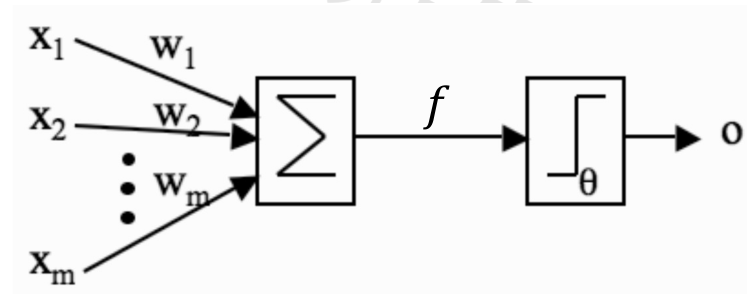
- 输入$X = [x_1, x_2, x_3, \ldots]$
- 权值$W = [w_1, w_2, w_3, \ldots]$
- 激活函数$f(net) = f(\sum(w_i * x_i))$
- 偏置单元（bias unit）x_0，其对应权值为w_0

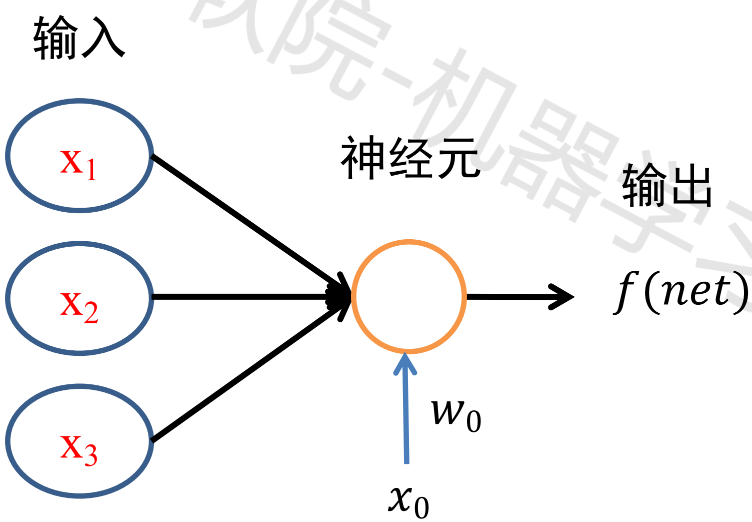




MP神经元基本结构

- ❑ 输入$X = [x_1, x_2, x_3, \dots]$
- ❑ 权值$W = [w_1, w_2, w_3, \dots]$
- ❑ 激活函数$f(net) = f(\sum(w_i * x_i))$
- ❑ 偏置单元（bias unit）x_0，其对应权值为w_0



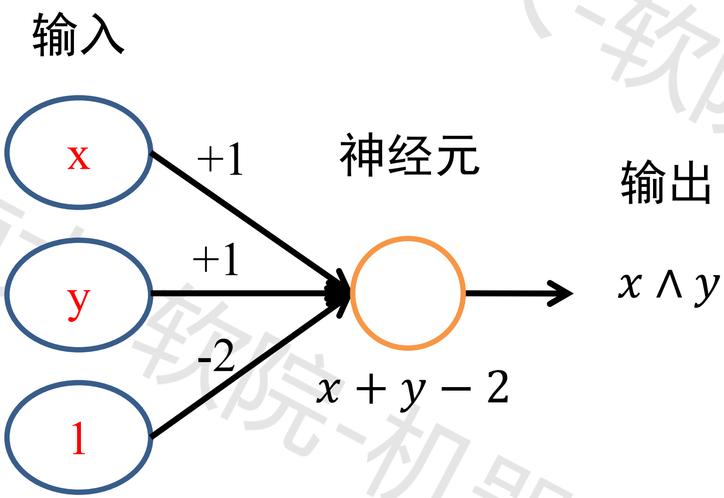


- ✓ 一组输入加权w_i相当于突触
- ✓ 一个加法器把输入信号相加（与收集电荷的细胞膜等价）
- ✓ 一个激活函数（最初是一个阈值函数）决定细胞对于当前的输入是否激活（“放电”）

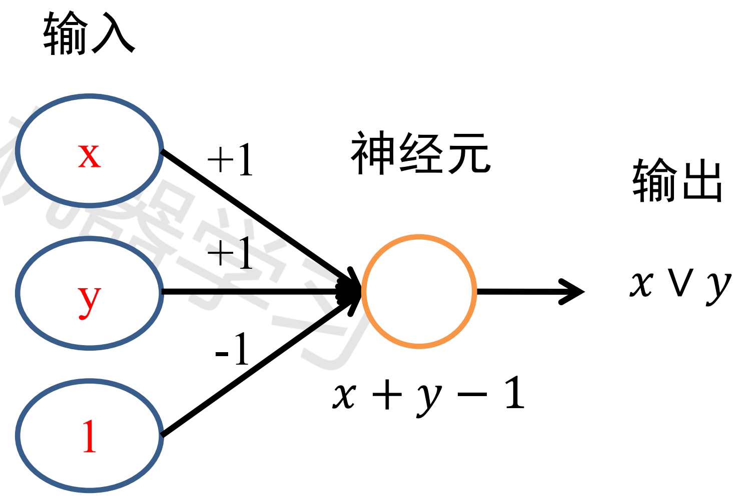
MP神经元模型

$$h = \sum_{i=1}^{m} w_i x_i$$

$$o = g(h) = \begin{cases} 1, & if\ h > 0 \\ 0, & if\ h \leq 0 \end{cases}$$



逻辑与的神经元模型
（阈值为0, 大于等于0输出1）

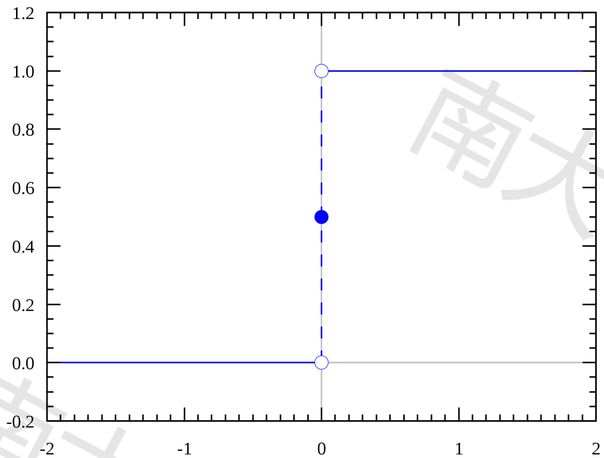


逻辑或的神经元模型
（阈值为0,小于0输出0）

MP神经元的局限

- 输入方面：线性求和 → 非线性求和
- 输出方面：单一输出值 → 脉冲序列
- 更新机制：时钟同步更新 → 随机异步更新
- 权值的物理（生理）意义
  - ✓ 两类神经元连接（兴奋性连接，权值为正；抑制性连接，权值为负）
  - ✓ 但不存在由正到负、或者由负到正的连接

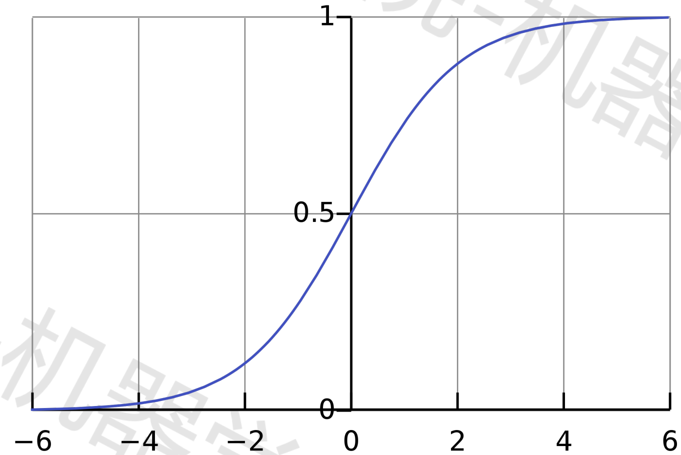
激活函数



单位阶跃函数
Heaviside step function

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

- 不连续函数
- 对变化敏感
- $x = 0$，不可微
- 一般适合单层感知机

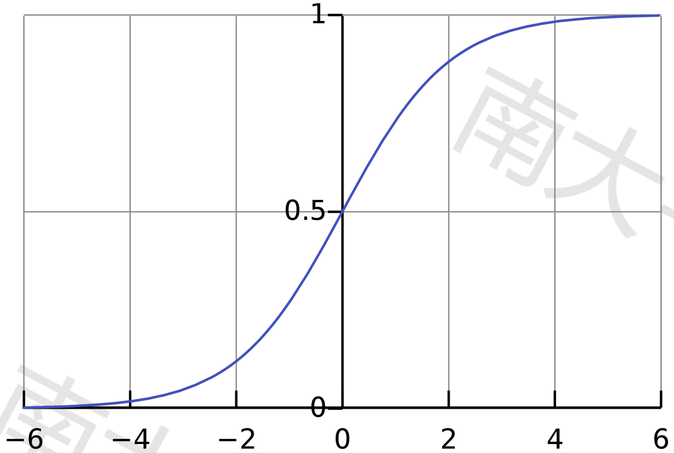


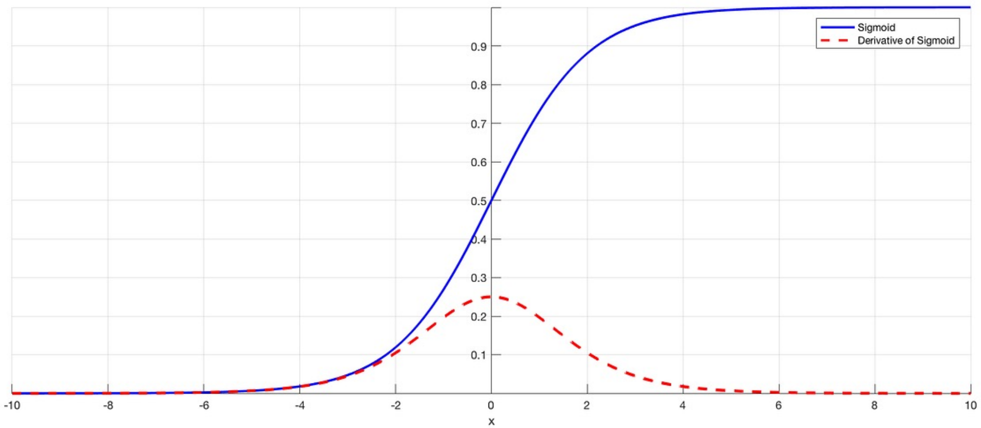
Sigmoid function

$$f(x) := \frac{1}{1 + e^{-x}}$$

- 连续、光滑、严格单调
- 函数范围在(0,1)之间
- S形曲线，非线性函数
- 导数为其本身的函数

激活函数





Sigmoid function

导数：

$$f(x) := \frac{1}{1+e^{-x}}$$

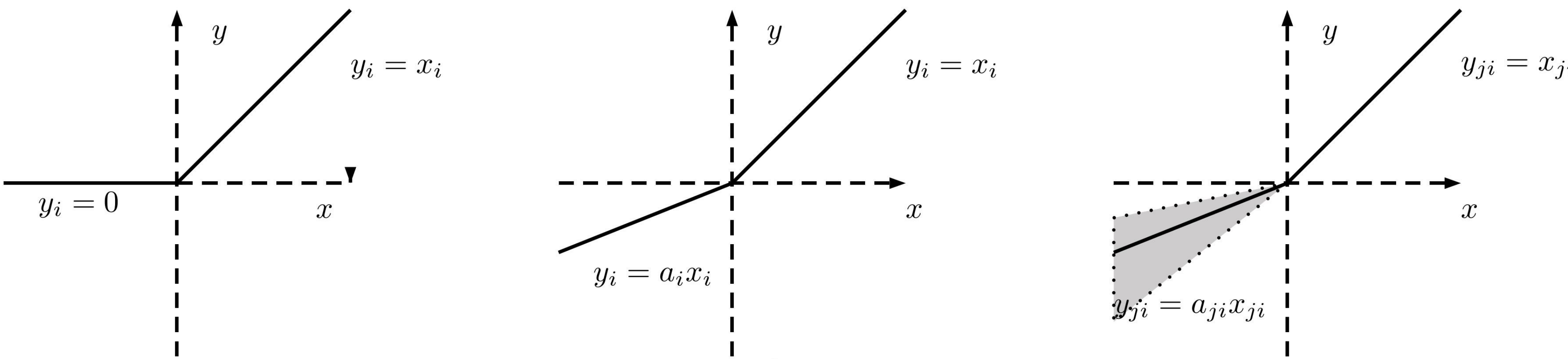
- 连续、光滑、严格单调
- 函数范围在(0,1)之间
- S形曲线，非线性函数
- 导数为其本身的函数

$$f'(x) = \mathrm{f(x)(1-f(x))}$$

- 饱和型激励函数
- 导数始终小于1，在0周围变化
- 容易造成梯度消失问题
- 不以0为对称轴
- 指数计算代价大

激活函数

- **饱和激活函数：** Sigmoid, tanh, …
- **非饱和激活函数：** ReLU, ELU, Leaky ReLU, PReLU, RReLU, …



ReLU　　Leaky ReLU/PReLU　　Randomized Leaky ReLU

$$f(x) \coloneqq \max(0, x)$$

$$f(x) := \begin{cases} x, & if\ x \ge 0 \\ a \cdot x, & if\ x < 0 \end{cases}$$

$$f(a, x) := \begin{cases} x, & if\ x \ge 0 \\ a \cdot x, & if\ x < 0 \end{cases}$$

$$f'(x) := \begin{cases} 1, & if\ x > 0 \\ 0, & if\ x < 0 \end{cases}$$

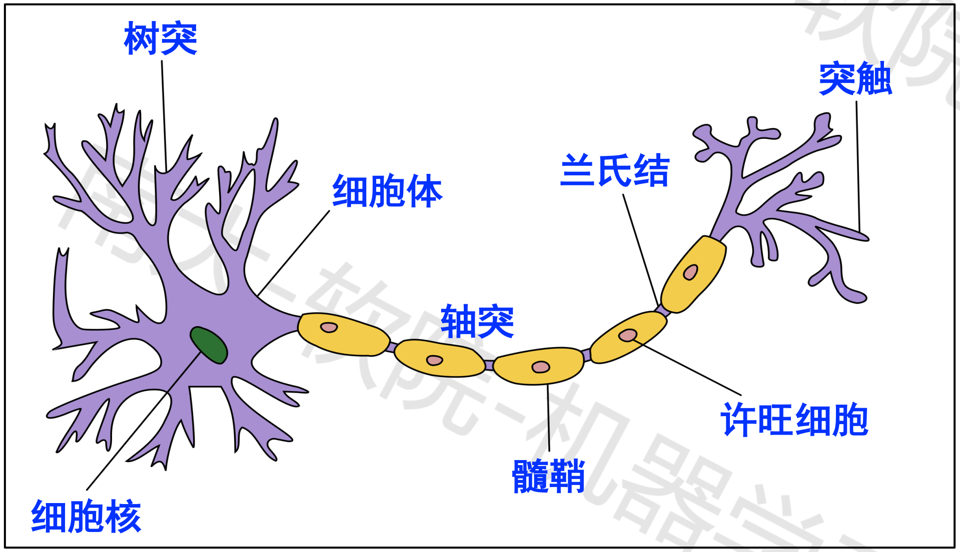
$$f'(x) := \begin{cases} 1, & if\ x \ge 0 \\ a, & if\ x < 0 \end{cases}$$

训练阶段 a从一个均匀分布中随机采样

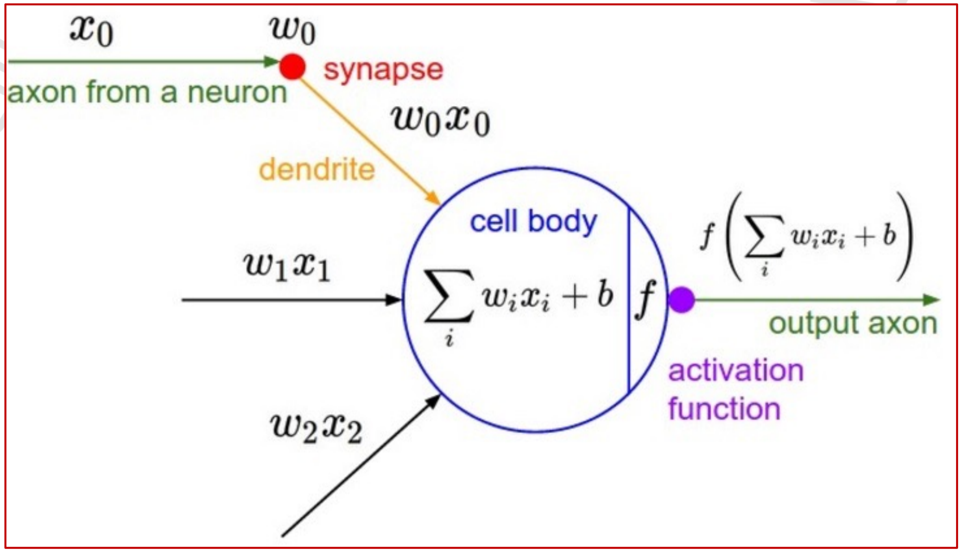
https://en.wikipedia.org/wiki/Activation_function

生物神经元→计算神经元

生物神经元

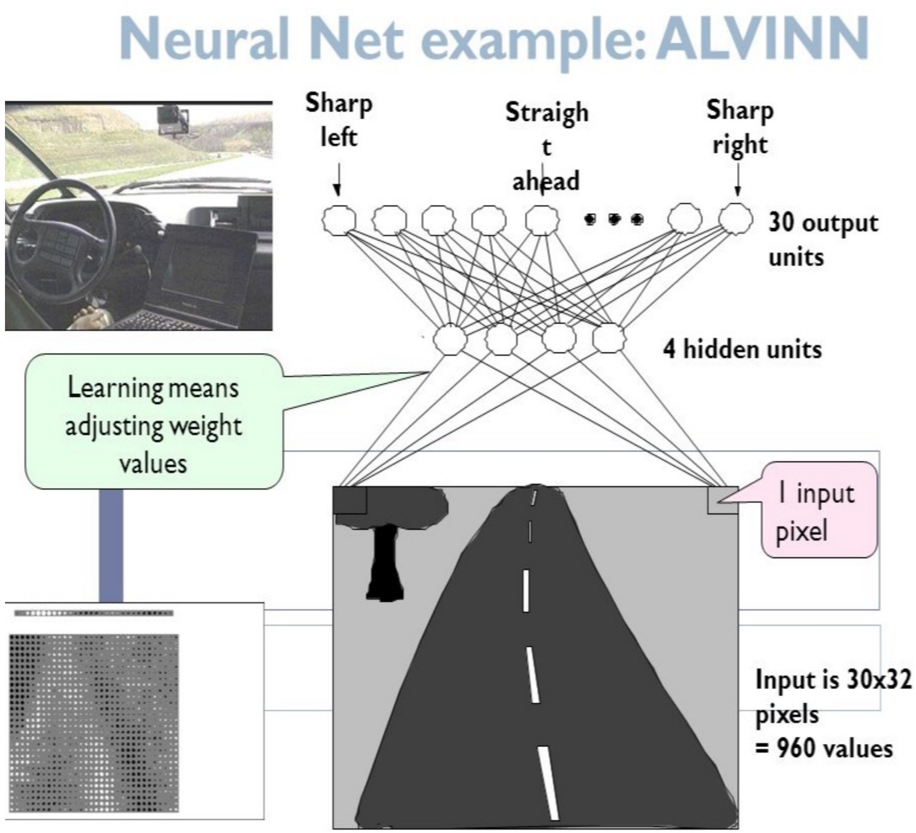


计算神经元



神经元→神经网络表示

- ALVINN : **A**utonomous **L**and **V**ehicle **I**n a **N**eural **N**etwork



- 1989年，CMU研发
- 输入：30*32的像素
- 4个隐藏节点
- 输出：30个驾驶动作(急剧左转、急剧右转，正前方行进……)

有向/无向？有环/无环？结构？

ALVINN: An Autonomous Land Vehicle In a Neural Network. NeurIPS 1989.

大纲

脑和神经元

感知机学习

线性可分性

感知机和感知机学习

- Frank Rosenblatt，1957年，Cornell航空实验室（Cornell Aeronautical Laboratory）
- 最简单形式的前馈式人工神经网络
- 一种二元线性分类器，使用特征向量作为输入，把矩阵上的输入x（实数值向量）映射到输出值$f(x)$上（一个二元的值）

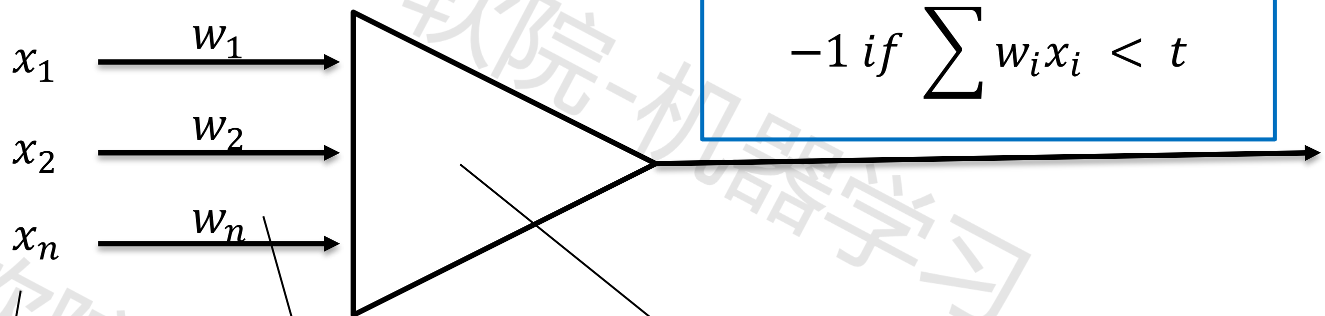
感知机结构

Perceptron Learning
(Rosenblatt, 1957)

Output function

$$1 \ if \sum w_i x_i \ \geq t$$

$$-1 \ if \sum w_i x_i \ < \ t$$



Weights

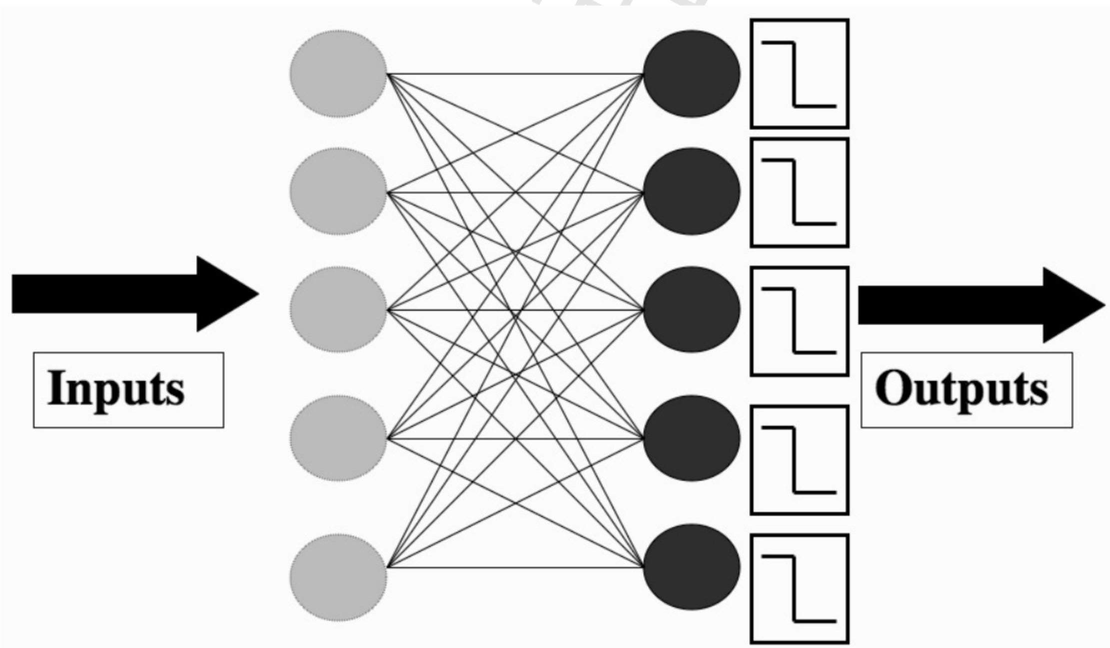
Input Signals
(1 or -1)

Activation
Level $\sum w_i x_i$

Learning Function

$$\Delta W_i \ = \ c(d \ - signal\left(\sum w_i x_i\right))x_i$$

感知机结构



- 非线性前馈网络
  - 同层内无互连
  - 不同层间无反馈
  - 由下层向上层传递
  - 输入输出均为离散值
  - 由阈值函数决定其输出

有监督的学习机制

- c是常数，表示学习率
- d是期望的输出，取值为1或-1
- sign是感知机的输出，取值为1或-1

教材上符号表达

$$w_{ij} \leftarrow w_{ij} - \eta(y_j - t_j) \cdot x_i$$

$$\Delta W_i = c(\mathrm{d} - \mathrm{sign}(\sum \mathrm{w}_i * \mathrm{x}_i))\mathrm{X}_i$$

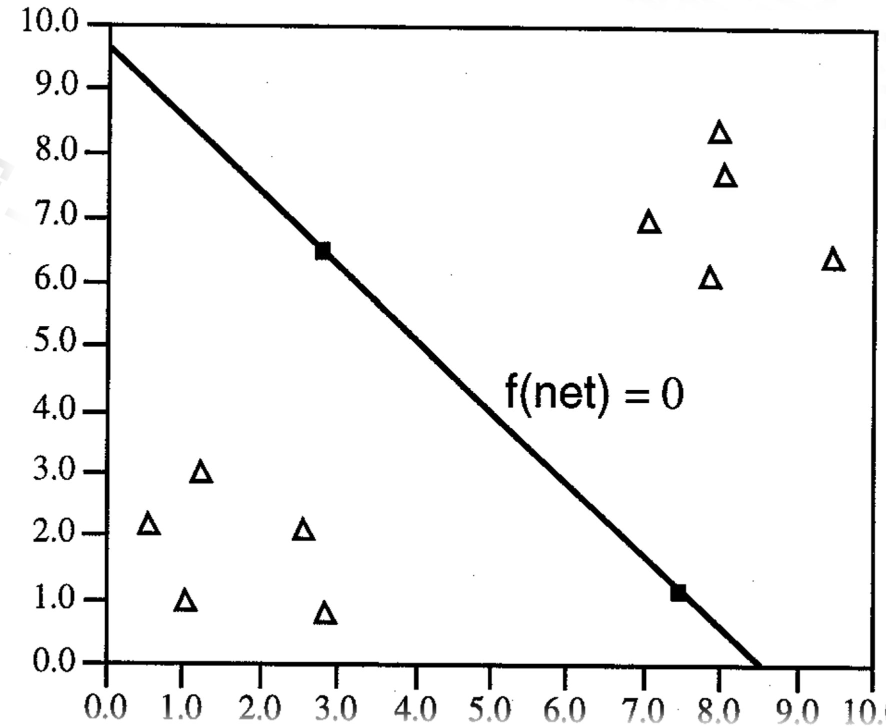
- 期望输出和实际输出相同，不改变权值
- 实际输出为-1，期望输出为+1，则增加$2cX_i$
- 实际输出为+1，期望输出为-1，则减少$2cX_i$

感知机的学习算法

1. 权值初始化
2. 输入样本对
3. 计算输出
4. 根据感知机学习规则调整权重
5. 返回步骤2，输入下一对样本，直至对所有样本的实际输出与期望输出相等

例子

x_1	x_2	Output
1.0	1.0	1
9.4	6.4	–1
2.5	2.1	1
8.0	7.7	–1
0.5	2.2	1
7.9	8.4	–1
7.0	7.0	–1
2.8	0.8	1
1.2	3.0	1
7.8	6.1	–1



例子

x_1	x_2	Output
1.0	1.0	1
9.4	6.4	–1
2.5	2.1	1
8.0	7.7	–1
0.5	2.2	1
7.9	8.4	–1
7.0	7.0	–1
2.8	0.8	1
1.2	3.0	1
7.8	6.1	–1

$$f(net) = f(w_0 * x_0 + w_1 * x_1 + w_2 * x_2)$$

x_0是偏置单元，通常取1

- 假设初始权值为[-0.6,0.75,0.5]
- 对第一行数据来说

  $f_1 = f(-0.6 * 1 + 0.75 * 1 + 0.5 * 1)$

  $= f(0.65) = 1$
- 与期望值一样，所以W向量不变，$W_1 = W_0$

x_1	x_2	Output
1.0	1.0	1
9.4	6.4	–1
2.5	2.1	1
8.0	7.7	–1
0.5	2.2	1
7.9	8.4	–1
7.0	7.0	–1
2.8	0.8	1
1.2	3.0	1
7.8	6.1	–1

$$f(net) = f(w_0 * x_0 + w_1 * x_1 + w_2 * x_2)$$

x_0是偏置单元，通常取1

- 假设初始权值为[-0.6,0.75,0.5]
- 对第一行数据来说

$$f_1 = f(-0.6 * 1 + 0.75 * 1 + 0.5 * 1) = f(0.65) = 1$$

- 与期望值一样，所以W向量不变，$W_1 = W_0$

$$f_2 = f(-0.6 * 1 + 0.75 * 9.4 + 0.5 * 6.4) = f(9.65) = 1$$

- 期望为-1，所以$W_2 = W_1 + 0.2 * (-2)X_2$

$$W_2 = [-0.6,0.75,0.5] - 0.4 * [1,9.4,6.4]$$

$$= [-1.00, -3.01, -2.06]$$

x_1	x_2	Output
1.0	1.0	1
9.4	6.4	–1
2.5	2.1	1
8.0	7.7	–1
0.5	2.2	1
7.9	8.4	–1
7.0	7.0	–1
2.8	0.8	1
1.2	3.0	1
7.8	6.1	–1

$$f(net) = f(w_0 * x_0 + w_1 * x_1 + w_2 * x_2)$$

x_0是偏置单元，通常取1

- 假设初始权值为[-0.6,0.75,0.5]
- 对第一行数据来说

$$f_1 = f(-0.6 * 1 + 0.75 * 1 + 0.5 * 1) = f(0.65) = 1$$

- 与期望值一样，所以W向量不变，$W_1 = W_0$

$$f_2 = f(-0.6 * 1 + 0.75 * 9.4 + 0.5 * 6.4) = f(9.65) = 1$$

- 期望为-1，所以$W_2 = W_1 + 0.2 * (-2)X_2$

$$W_2 = [-0.6,0.75,0.5] - 0.4 * [1,9.4,6.4]$$
$$= [-1.00, -3.01, -2.06]$$

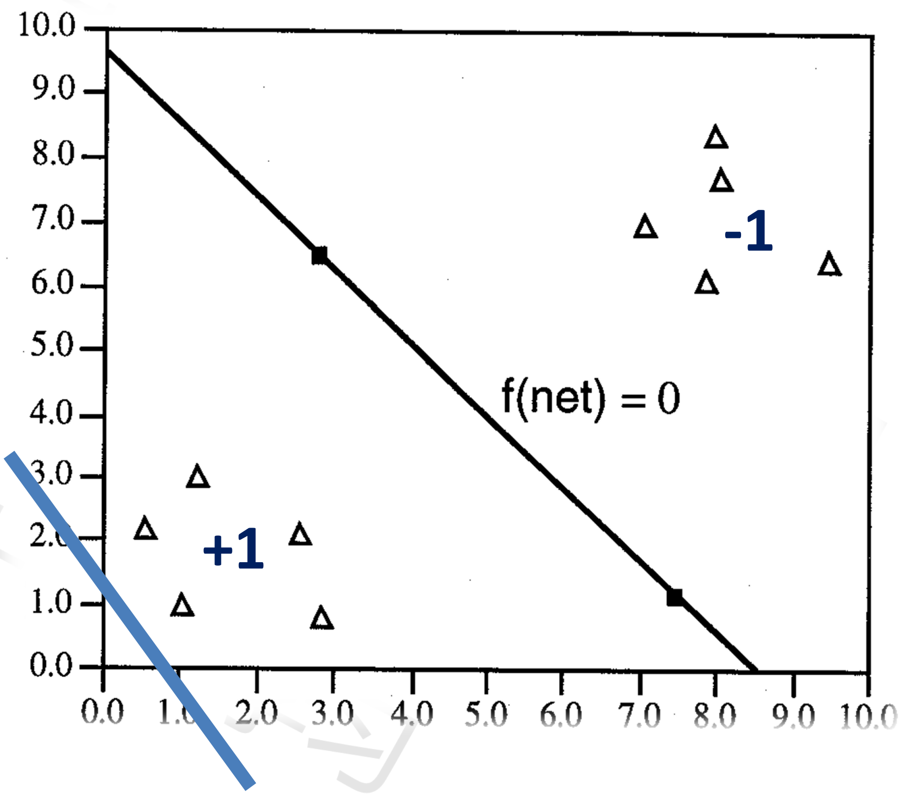
$$f_3 = f(-1 * 1 - 3.01 * 2.5 - 2.06 * 2.1)$$
$$= f(-12.84) = -1$$
$$W_3 = W_2 + 0.2 * 2 * X_3 = [-0.60, -2.01, -1.22]$$

…..

- 最终结果为W=[10.9，-1.3，-1.1]

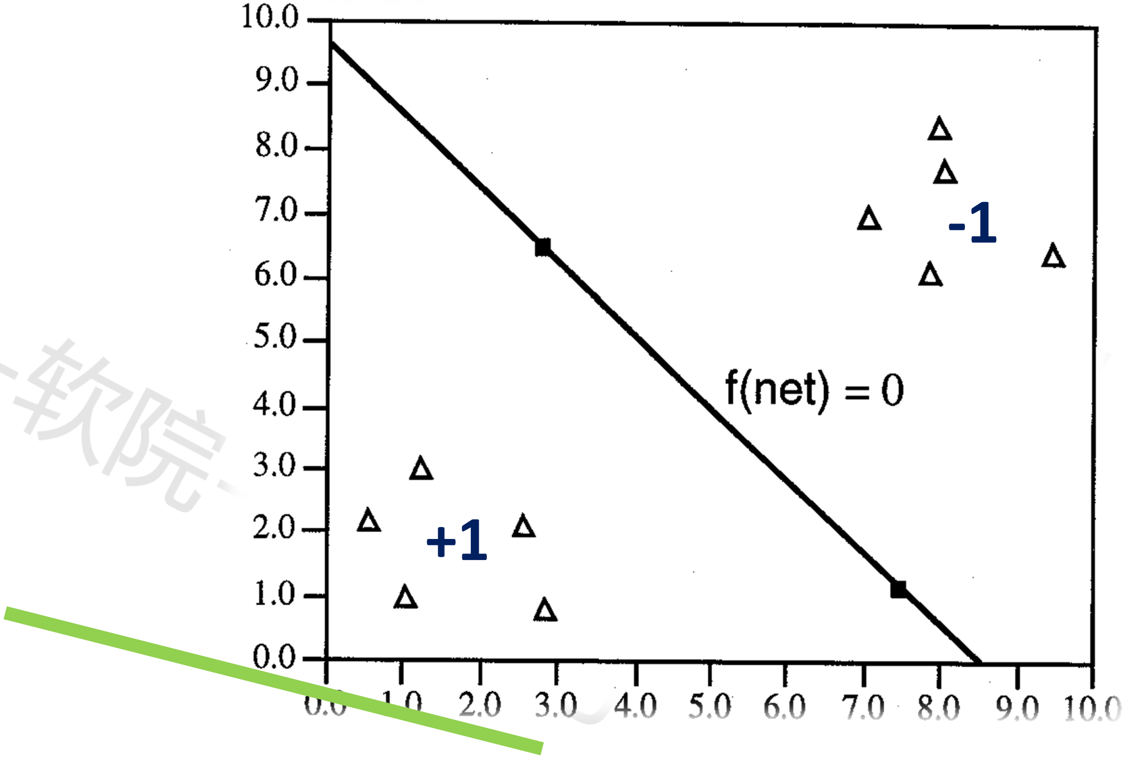
X_0固定为1
a固定为0.2

图示



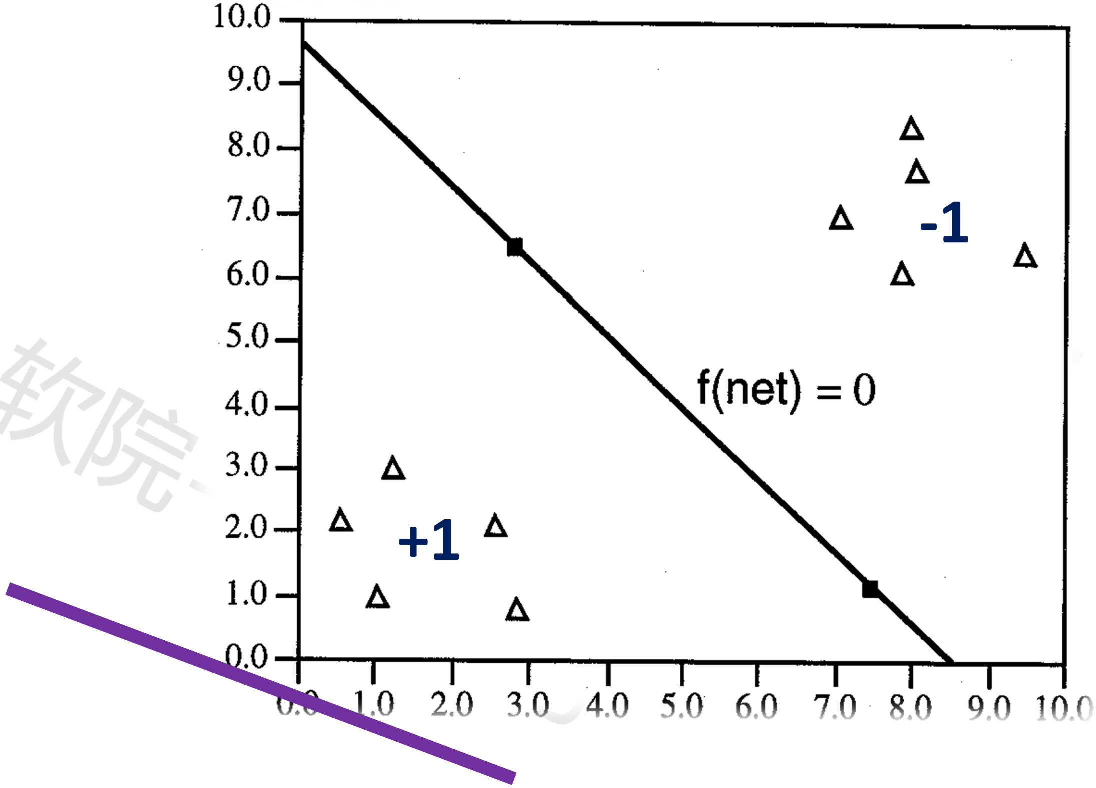
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0
-1
+1
f(net) = 0

图示

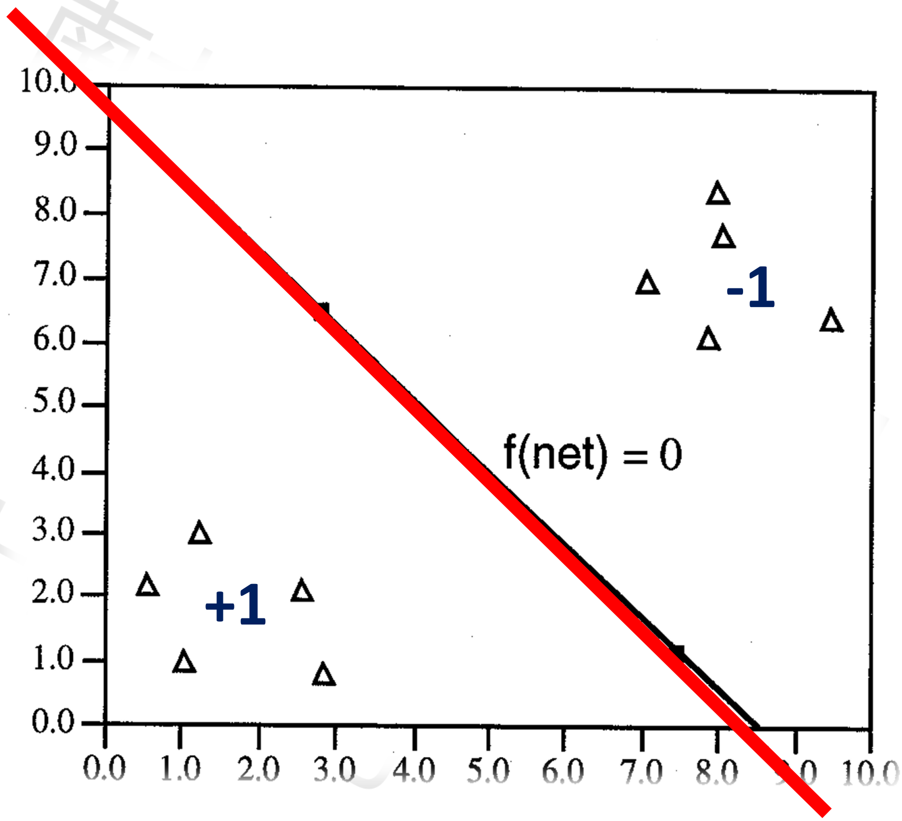


10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
-1
f(net) = 0
+1
0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0

图示


10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
-1
+1
f(net) = 0

图示



10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0
-1
+1
f(net) = 0

大纲

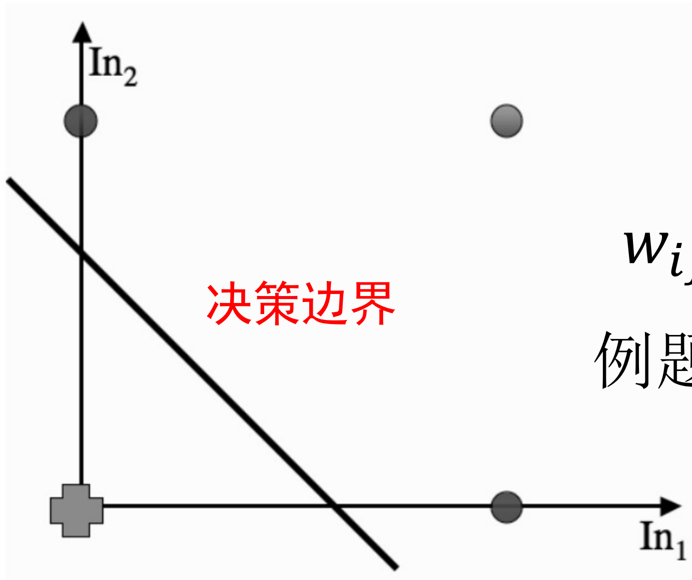
脑和神经元

感知机学习

线性可分性

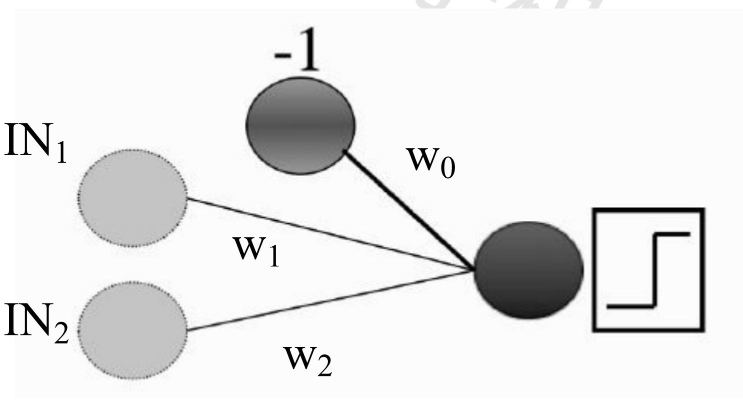
OR函数

In_1	In_2	t
0	0	0
0	1	1
1	0	1
1	1	1



$$w_{ij} \leftarrow w_{ij} - \eta(y_j - t_j) \cdot x_i$$

例题中，令 $\eta = 0.25$



$w_0 = -0.05$，

$w_1 = -0.02$，

$w_2 = \ \ 0.27$，

课堂练习

$w_0 =$?

$w_1 =$?

$w_2 =$?

决策边界

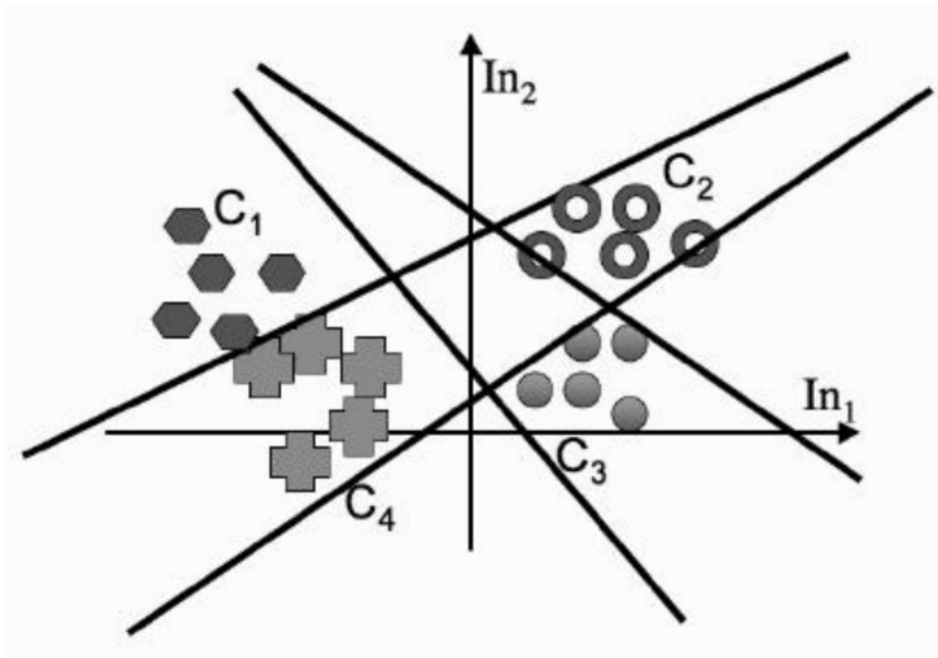
- ❑ **决策边界 Decision boundary**
  - ✓ 鉴别函数 discriminant function $\mathbf{w}^{\mathrm{T}} \cdot \mathbf{x} \geq 0$
  - ✓ 神经元激活(假设激活函数是阈值为0的阶跃函数)
  - ✓ 等价于向量内积 $\mathbf{a} \cdot \mathbf{b} = \|a\|\|b\|\cos\theta$
  - ✓ 假设有两个输入点

$$\left.\begin{aligned} \boldsymbol{w}^T \cdot \mathbf{x}_1 = 1 \\ \boldsymbol{w}^T \cdot \mathbf{x}_2 = 1 \end{aligned}\right\} \boldsymbol{w}^T \cdot (\mathbf{x}_1 - \mathbf{x}_2) = 0$$

❑ **物理解释：**

$\boldsymbol{w}^T$就是找与$\mathbf{x}_1$ $\mathbf{x}_2$连线的垂直线

多分类的决策边界



- 每一个输出神经元定义一条决策边界
- 多个神经元就决定了多分类的决策边界

感知机收敛理论

- [Rosenblatt, 1962] 给定一个线性可分数据集，感知机将在有限次迭代中收敛到一个决策边界
- 定义γ是分离超平面与最接近的数据点之间的距离，则迭代次数的界是$1/\gamma^2$

感知机收敛理论

- [Rosenblatt, 1962] 给定一个线性可分数据集，感知机将在有限次迭代中收敛到一个决策边界
- 定义γ是分离超平面与最接近的数据点之间的距离，则迭代次数的界是$1/\gamma^2$

$$
\begin{aligned}
\mathbf{w}^* \cdot \mathbf{w}^{(\mathbf{t})} &= \mathbf{w}^* \cdot \left(\mathbf{w}^{(\mathbf{t}-1)} + y\mathbf{x}\right) \\
&= \mathbf{w}^* \cdot \mathbf{w}^{(\mathbf{t}-\mathbf{1})} + y\mathbf{w}^* \cdot \mathbf{x} \\
&\geq \mathbf{w}^* \cdot \mathbf{w}^{(\mathbf{t}-\mathbf{1})} + \gamma \geq t\gamma
\end{aligned}
$$

- **证明思路：**

考察权重向量的迭代过程，假设第t轮输出有误差，需要更改t-1轮的权值

结论1：每次权值更新至少比上一轮增加γ

感知机收敛理论

❑ **柯西不等式(Cauchy–Schwartz inequality)**

$$t\gamma \leq \mathbf{w}^* \cdot \mathbf{w}^{(t)} \leq \|\mathbf{w}^*\| \|\mathbf{w}^{(t)}\| \quad \Rightarrow \quad t\gamma \leq \|\mathbf{w}^{(t)}\|$$

假设上一轮输出不正确，此项为0.

思考：神经网络的输入输出

$$\|\mathbf{w}^{(t)}\|^2 = \|\mathbf{w}^{(t-1)} + y\mathbf{x}\|^2$$

$$= \|\mathbf{w}^{(t-1)}\|^2 + \boxed{y^2\|\mathbf{x}\|^2} + \boxed{2y\mathbf{w}^{(t-1)} \cdot \mathbf{x}}$$

$$\leq \|\mathbf{w}^{(t-1)}\|^2 + 1 \leq t$$

根据假设

$$t\gamma \leq \|\mathbf{w}^{(t-1)}\| \leq \sqrt{t} \quad \Rightarrow \quad t \leq 1/\gamma^2$$

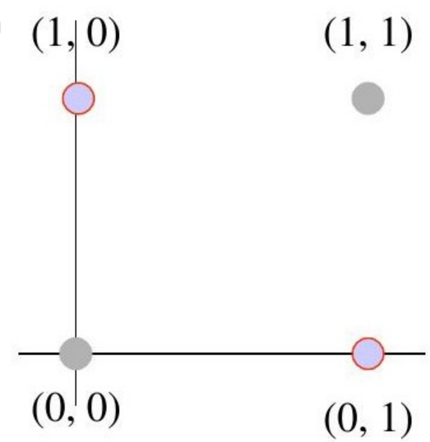
感知机学习缺点

- 感知机属于单层神经网络，不能解决一类**非线性可分**的问题
- 典型的例子就是**异或**

表 1. 异或真值表

输入1	输入2	输出
0	0	0
0	1	1
1	0	1
1	1	0

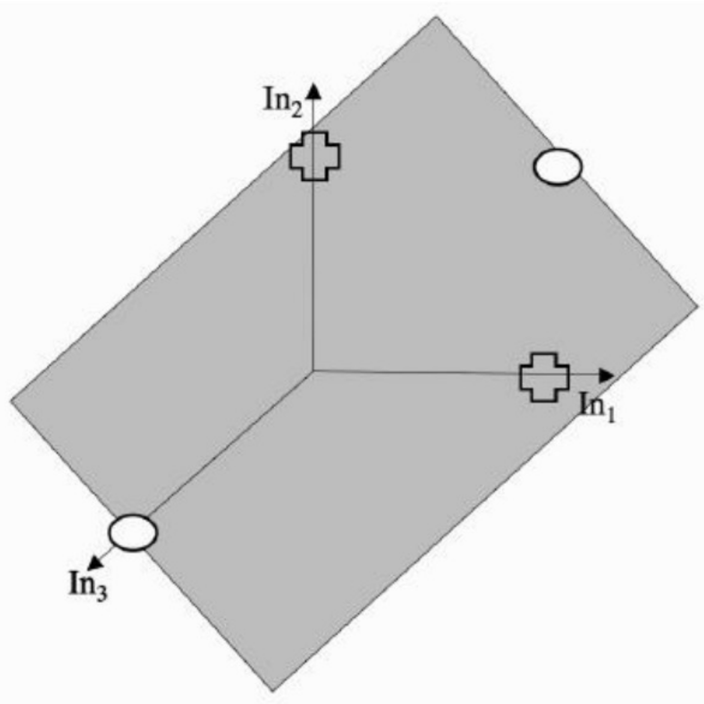
图 1.异或问题



➢ 在二维空间中没有可分离点集{(0,0),(1,1)}和{(0,1),(1,0)}的直线

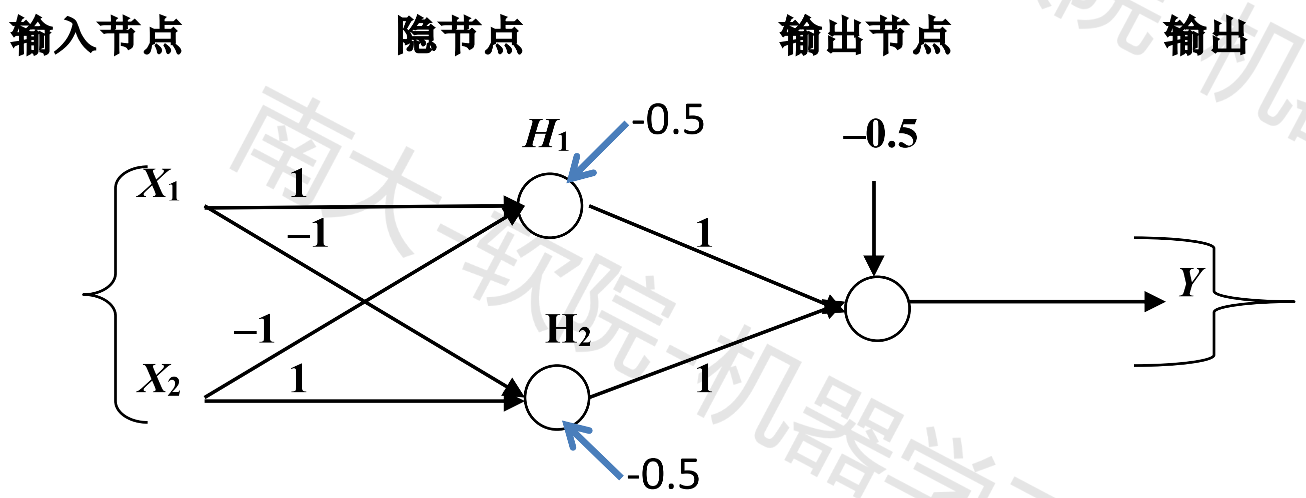
用线性分类器处理异或

In_1	In_2	In_3	Output
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	1



利用核技巧(SVM一讲介绍)，将数据投影到合适的高维空间

用多层网络处理异或



- 权重向量(1,-1)、(-1,1)

输入		隐节点输出		输出节点	X_1 XOR X_2
X_1	X_2	H_1	H_2		
0	0	0	0	–0.5 → 0	0
0	1	–1 → 0	1	0.5 → 1	1
1	0	1	–1 → 0	0.5 → 1	1
1	1	0	0	–0.5 → 0	0

感知机的表达能力

- 感知机是n维实例空间的超平面决策
- 候选假设空间H(决策变量的集合)：所有可能实数权向量的集合

$$H = \{\overrightarrow{W} \mid \overrightarrow{W} \in R^{(n+1)}\}$$

- 广义布尔函数 m-of-n：n个输入值至少有m个为真，则输出为真
- 二层神经网络可以表达所有的布尔函数
  - ✓ 与、或、与非、或非、异或……

感知机学习的不足

- ❑ 感知机学习一定可以收敛吗？
  - ✓ 前提是训练样例必须是线性可分的！！

- ❑ 如果训练样例不是线性可分的，怎么办？
  - ✓ 只能去找一个学习方法，去收敛到目标概念的最佳近似

感知机学习方法只适用在单层网络！

思考和讨论

1. 感知机的表达能力
2. 感知机学习法则
3. 线性可分和非线性可分
4. 感知机收敛定理

谢 谢 ！