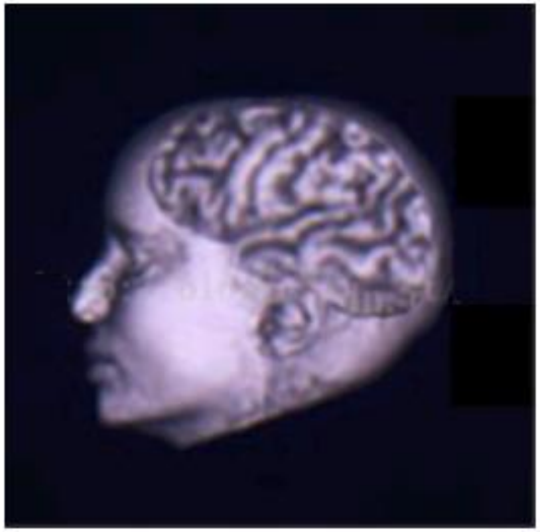
神经网络

陈飞宇

fchen@cqu.edu.cn

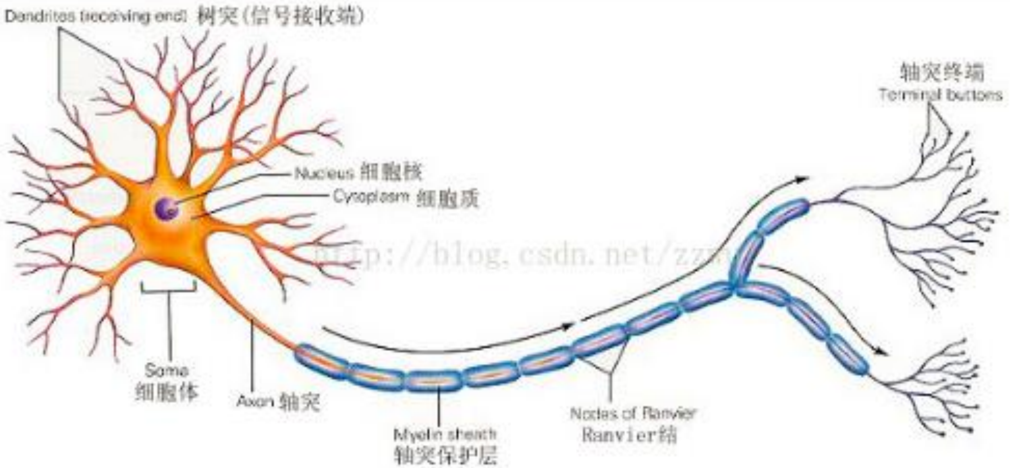
办公室：软件学院529

How our brain works?

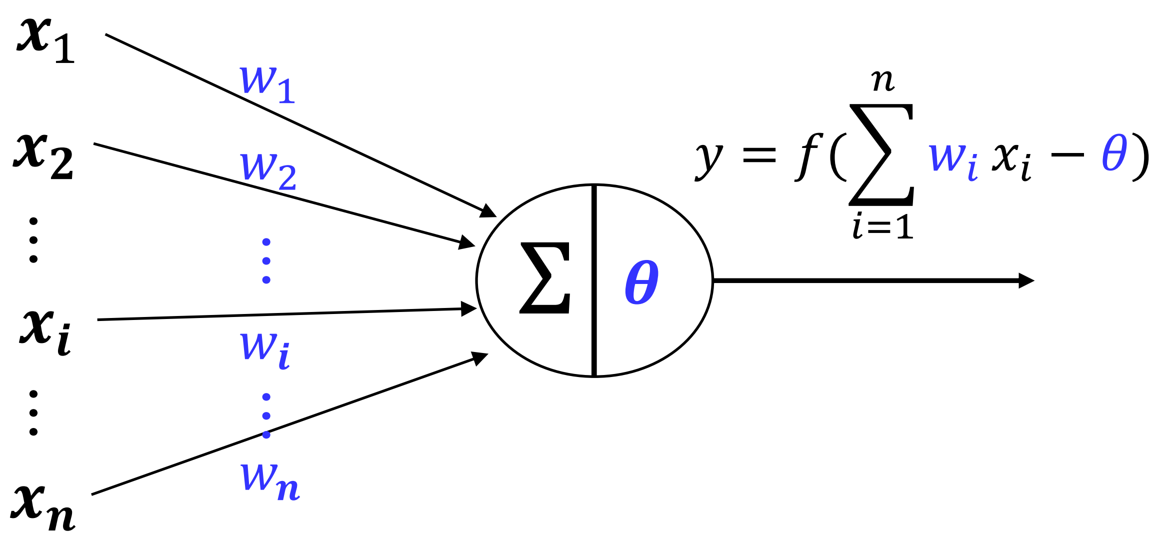


1 大脑半球像半个核桃

2 大脑皮层由灰质白质组成



神经元模型



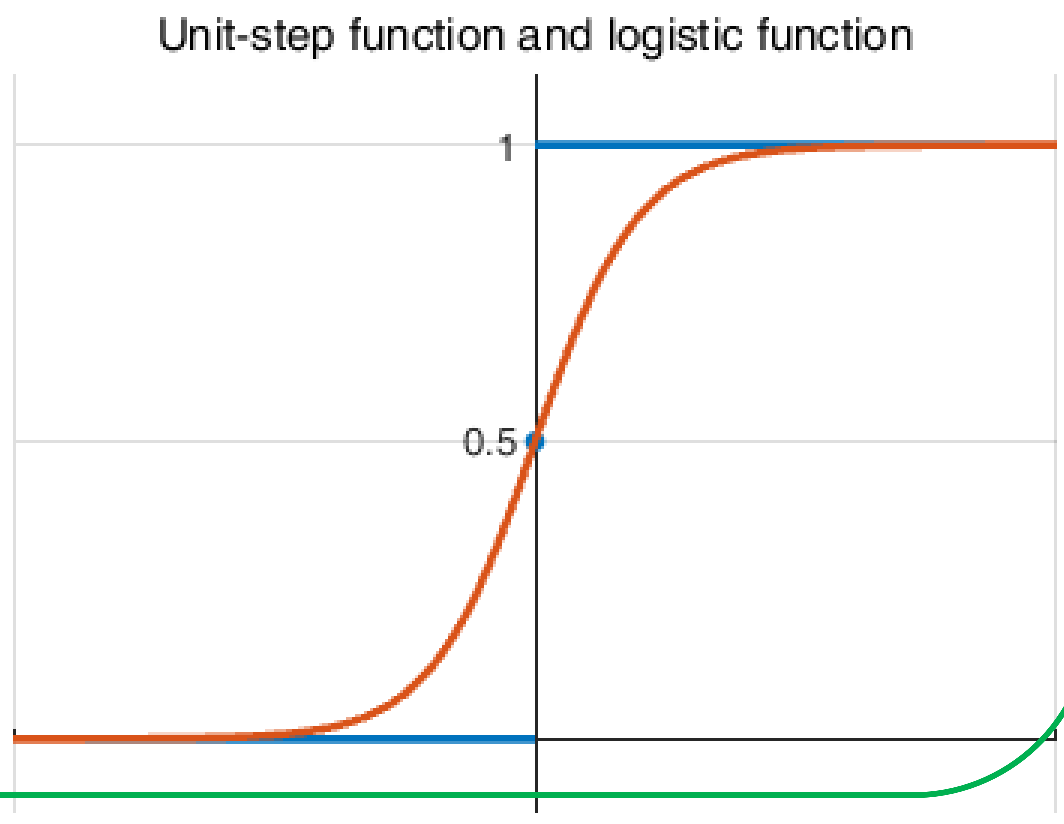
- x_i ： 第 i 个神经元的输入
- w_i ： 第 i 个神经元的权重
- θ ： 阈值(threshold)或称为偏置（bias）
- y ： 神经元状态（兴奋或抑制）

激活函数（ Activation function ）

由于单位阶跃函数不是一个连续函数，我们通常选择一些性质好的函数作为替代函数 。

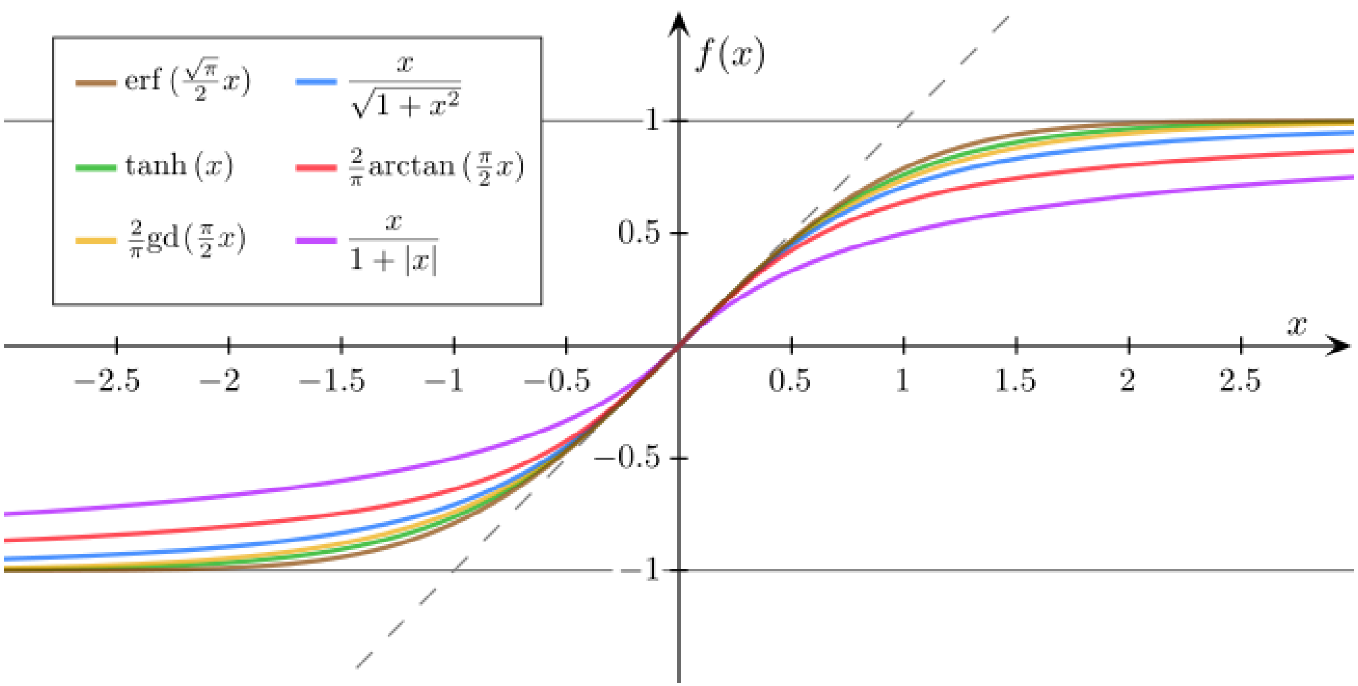
Logistic function:

$$y = \frac{1}{1 + e^{-z}}$$



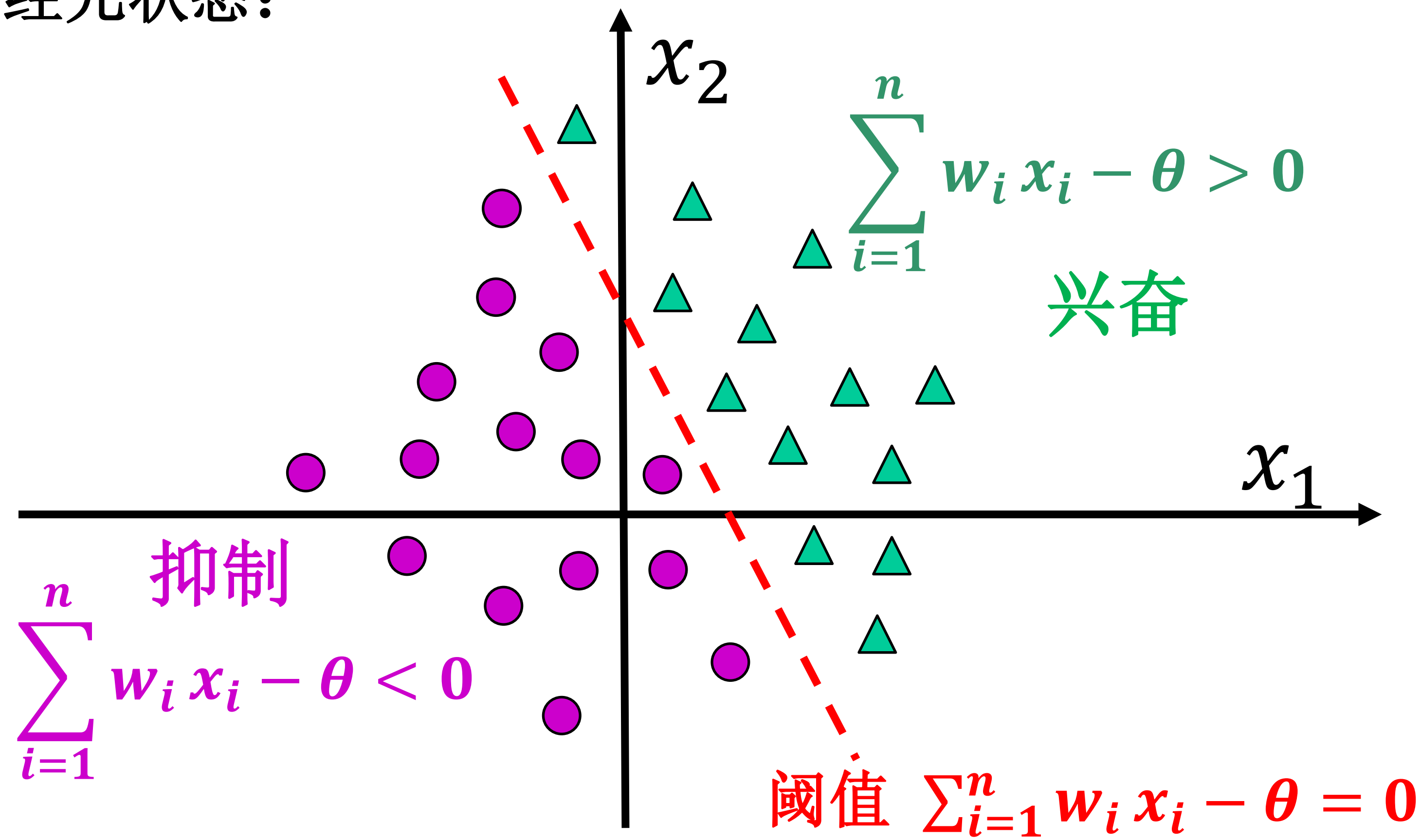
激活函数（ Activation function ）

$f(\cdot)$函数称为激活函数(activation function)或挤压函数（Squashing function）.



二分类问题

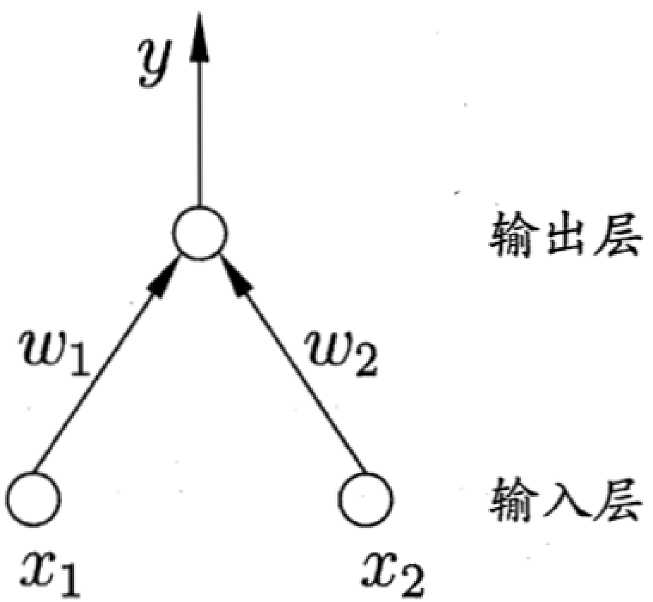
神经元状态：



x_2
x_1
$\sum_{i=1}^{n} w_i x_i - \theta > 0$
兴奋
抑制
$\sum_{i=1}^{n} w_i x_i - \theta < 0$
阈值 $\sum_{i=1}^{n} w_i x_i - \theta = 0$

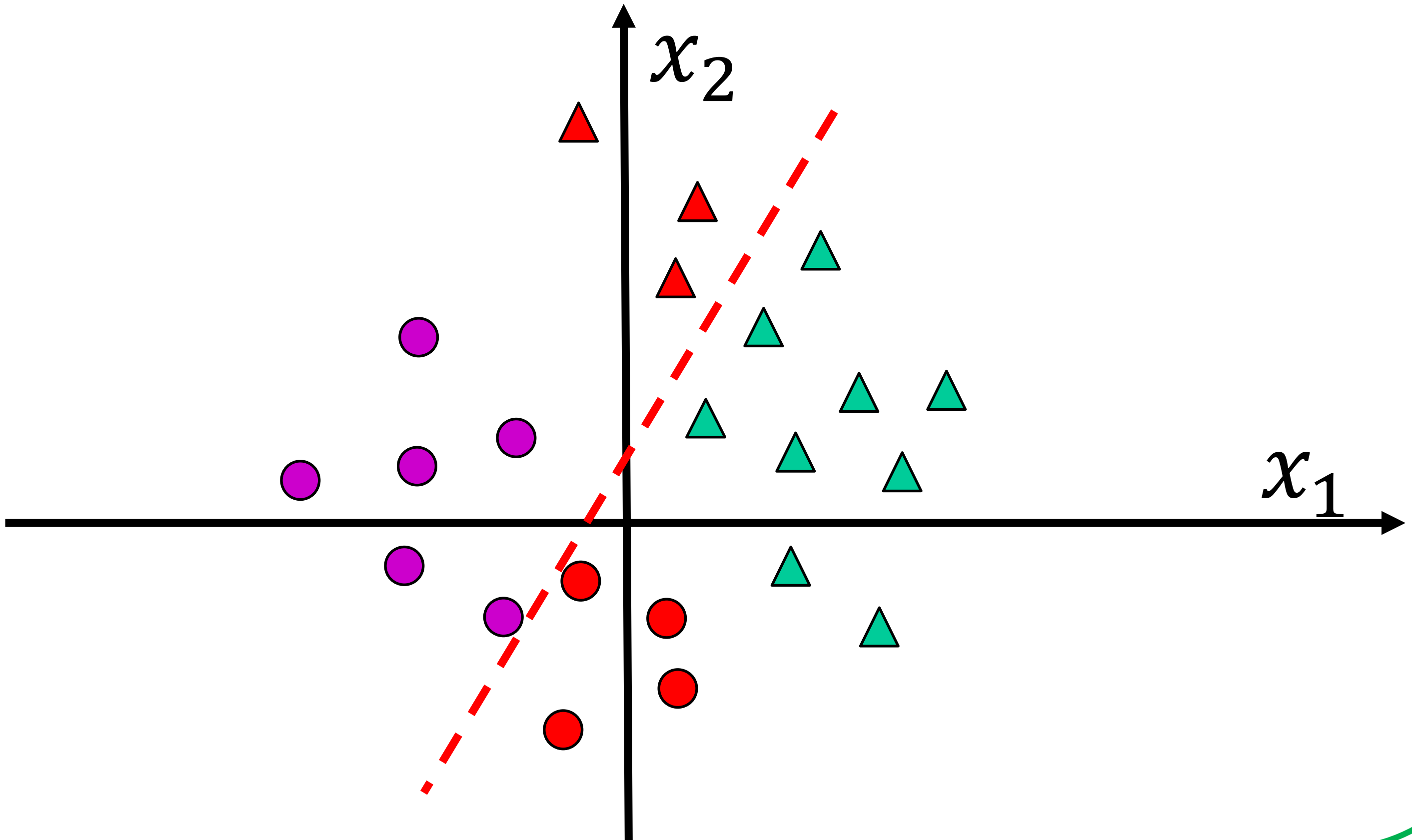
多层感知机（Multilayer Perceptron）

感知机（Perceptron）由两层神经元组成，是最简单的神经网络。



感知机原理

考虑二分类问题，红色点代表着误分类点。



误分类点

考虑输出变量 $y_i \in \{0,1\}$，定义

$$\gamma_i = (\hat{y}_i - y_i)\frac{w^{\mathrm{T}}x_i - \theta}{\|w\|} \geq 0$$

其中 $\hat{y}_i = f(w^{\mathrm{T}}x_i - \theta)$。

若 $\hat{y}_i = y_i$，则 $\gamma_i = 0$；若 $\hat{y}_i \neq y_i$，则 $\gamma_i = \frac{|w^{\mathrm{T}}x_i - \theta|}{\|w\|}$

因此$(\hat{y}_i - y_i)(w^{\mathrm{T}}x_i - \theta)$衡量着误分类点$x_i$到分类超平面的距离。

感知机模型

令 $\boldsymbol{w_0}=\boldsymbol{\theta}, \boldsymbol{x_0}=-\mathbf{1}$, 即将阈值 $\boldsymbol{\theta}$ 看作是“哑结点” $\boldsymbol{x_0}$ 所对应的权重, 则感知机最小化误分类点到分类平面的距离和：

$$\min_{\hat{\boldsymbol{w}}} \sum_{i=0}^{n} (\hat{y}_i - y_i)\hat{\boldsymbol{w}}^{\mathrm{T}}\hat{\boldsymbol{x}}_i$$

$\hat{\boldsymbol{w}} = [\boldsymbol{w_0}; \boldsymbol{w_1}; \cdots; \boldsymbol{w_n}]$， $\hat{\boldsymbol{x}} = [\boldsymbol{x_0}; \boldsymbol{x_1}; \cdots; \boldsymbol{x_n}]$。

注意，上述求和仅当 $\hat{y}_i \neq y_i$ 时起作用！

感知机模型

感知机模型：

$$\min_{\hat{\boldsymbol{w}}} \sum_{i=0}^{n} (\hat{y}_i - y_i) \hat{\boldsymbol{w}}^{\mathrm{T}} \hat{\boldsymbol{x}}_i$$

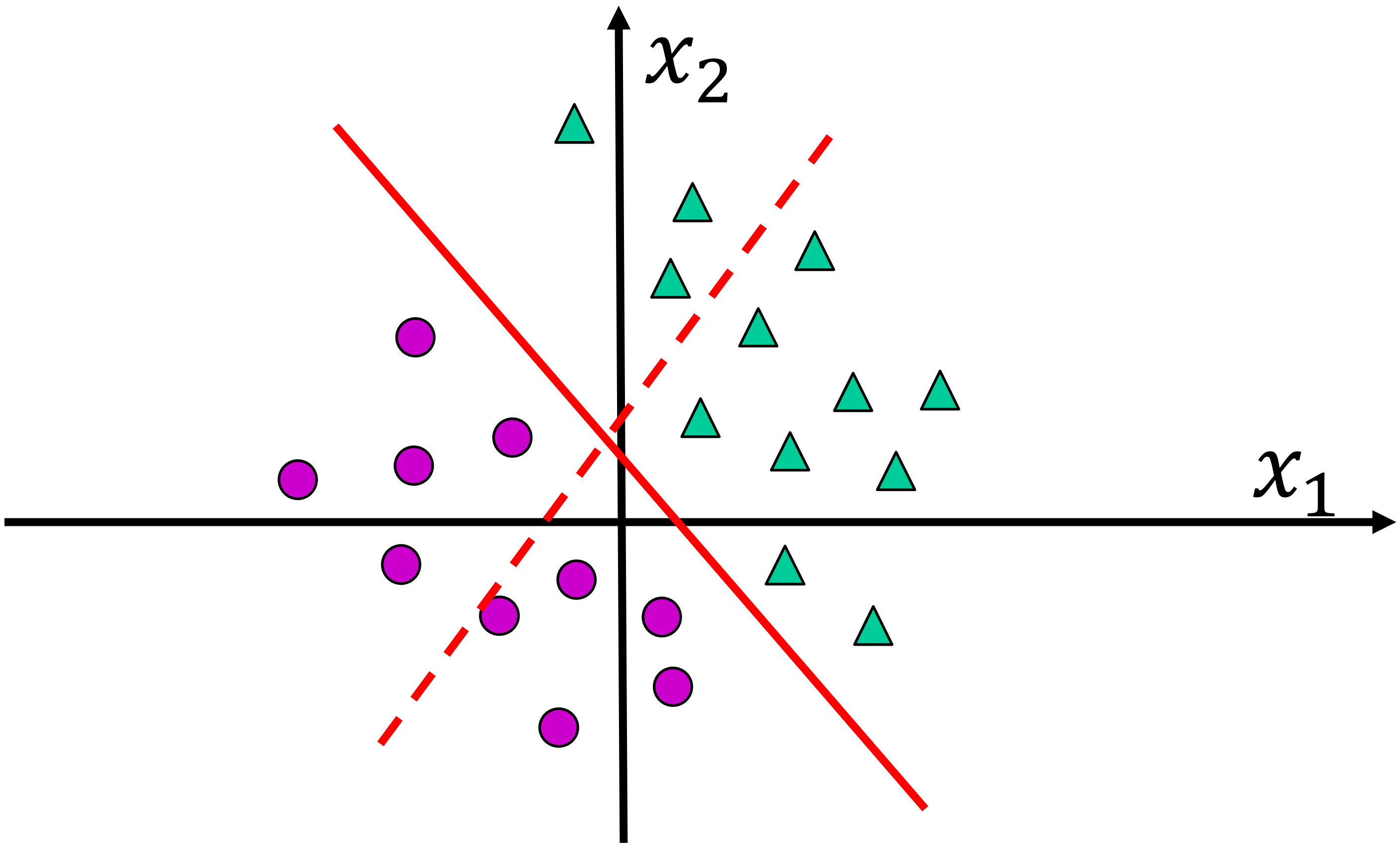
梯度下降法：

$$\hat{\boldsymbol{w}} \leftarrow \hat{\boldsymbol{w}} - \eta \Delta \hat{\boldsymbol{w}}$$

其中

$$\Delta \hat{\boldsymbol{w}} = \frac{\partial \left(\sum_{i=0}^{n} (\hat{y}_i - y_i) \hat{\boldsymbol{w}}^{\mathrm{T}} \hat{\boldsymbol{x}}_i \right)}{\partial \hat{\boldsymbol{w}}} = \sum_{i=0}^{n} (\hat{y}_i - y_i) \hat{\boldsymbol{x}}_i$$

感知机原理



逻辑运算 I

"与" 运算$(\boldsymbol{x_1} \wedge \boldsymbol{x_2} \wedge \cdots \wedge \boldsymbol{x_n})$，其中$\boldsymbol{x_i} \in \{\mathbf{0}, \mathbf{1}\}$。

令权重 $w_1 = w_2 = \cdots = w_n = 1$, 阈值 $\theta = n$.

则输出 $y = f(\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{w_i}\, \boldsymbol{x_i} - \boldsymbol{\theta}) = f(\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{x_i} - n)$

当且仅当 $\boldsymbol{x}_1 = \boldsymbol{x}_2 = \cdots = \boldsymbol{x}_n = 1$ 时， $y = \mathbf{1}$。

逻辑运算 II

"或" 运算$(\boldsymbol{x_1} \vee \boldsymbol{x_2} \vee \cdots \vee \boldsymbol{x_n})$，其中$\boldsymbol{x_i} \in \{\mathbf{0}, \mathbf{1}\}$。

令权重 $w_1 = w_2 = \cdots = w_n = 1$, 阈值 $\theta = \mathbf{1/2}$.

输出 $y = f\left(\sum_{i=1}^{n} \boldsymbol{w_i}\, \boldsymbol{x_i} - \boldsymbol{\theta}\right) = f\left(\sum_{i=1}^{n} \boldsymbol{x_i} - \mathbf{1/2}\right)$

当且仅当至少有一个 $\boldsymbol{x_i} = 1$ 时，$y = \mathbf{1}$。

逻辑运算 III

"非"运算$(\sim \boldsymbol{x_i})$，其中$\boldsymbol{x_i} \in \{\mathbf{0}, \mathbf{1}\}$。

令 $w_i = \mathbf{-2}$, 其他 $w_j = \mathbf{0}$，阈值 $\theta = \mathbf{-1}$.

输出 $y = f(\sum_{i=1}^{n} \boldsymbol{w_i x_i} - \boldsymbol{\theta}) = f(\mathbf{-2}\boldsymbol{x_i} + \mathbf{1})$

当 $\boldsymbol{x_i} = \mathbf{1}$ 时，$y = f(\mathbf{-1}) = \mathbf{0}$;
当 $\boldsymbol{x_i} = \mathbf{0}$ 时，$y = f(\mathbf{1}) = \mathbf{1}$。

逻辑运算 IV

"异或" 运算($x_i \oplus x_j$)，其中$x_i \in \{0, 1\}$。

若感知机可以解决异或运算，则

当 $x_i = x_j$ 时，$y = f(\sum_{i=1}^{n} w_i x_i - \theta) = 0$

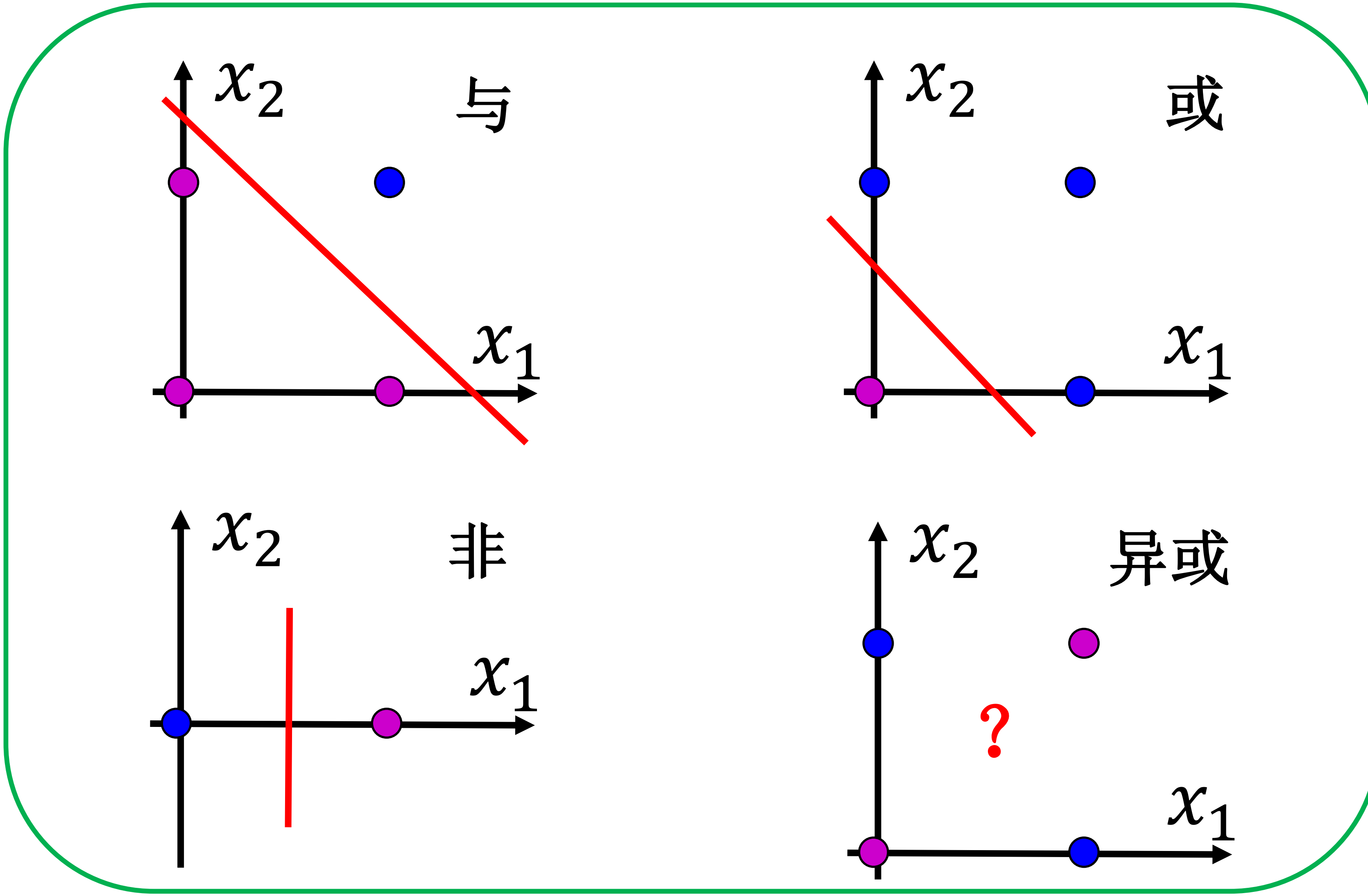
$$w_i + w_j - \theta < 0, -\theta < 0$$

当 $x_i \neq x_j$ 时，$y = f(\sum_{i=1}^{n} w_i x_i - \theta) = 1$

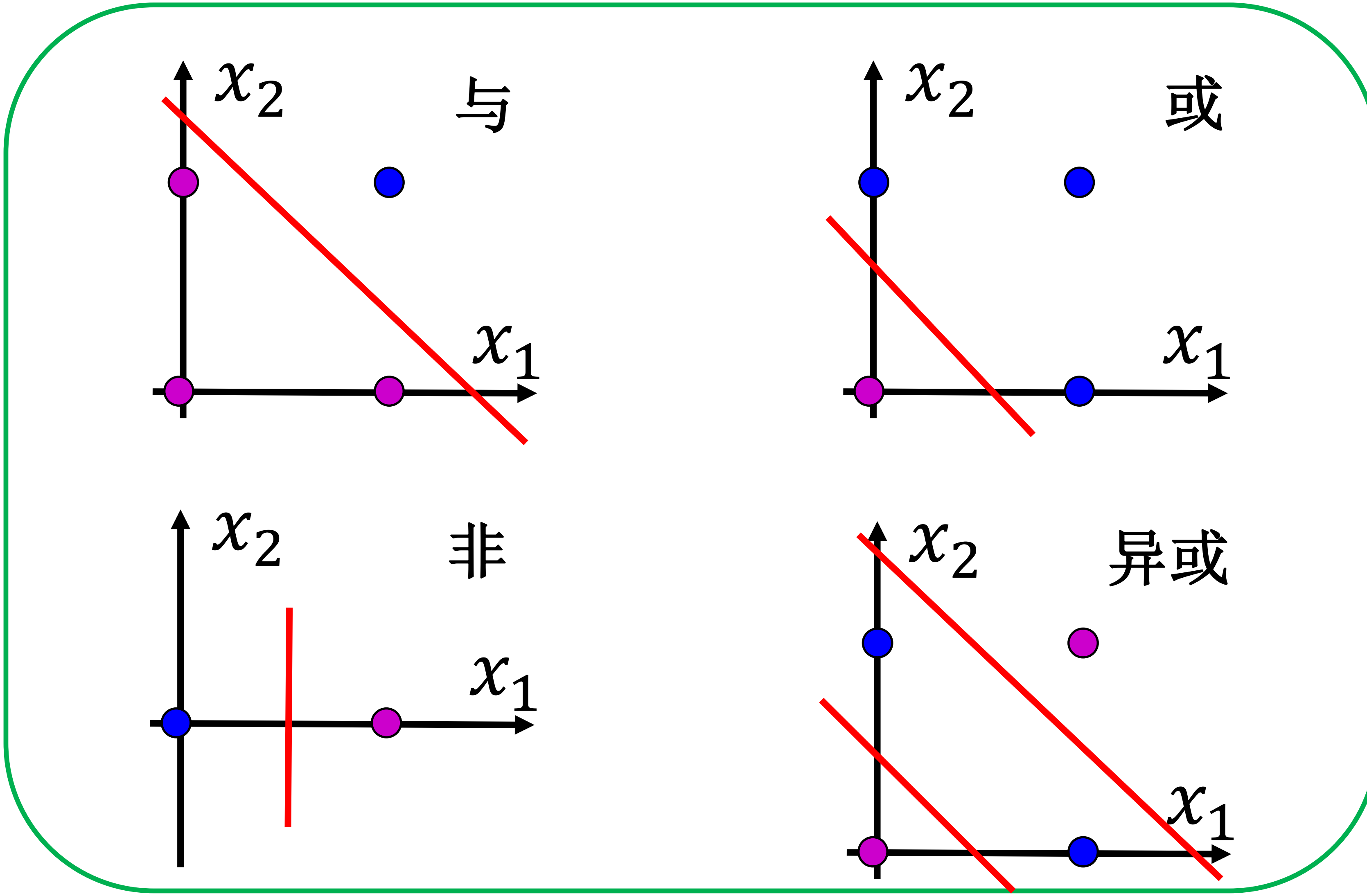
$$w_i - \theta \geq 0, w_j - \theta \geq 0$$

不存在这样的 w_i和w_j！！！

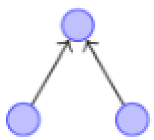
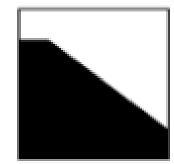
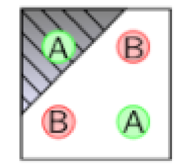
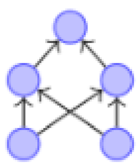
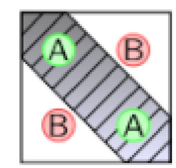
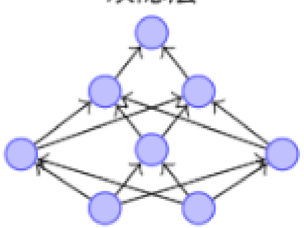
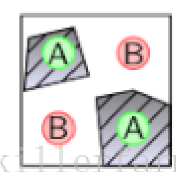
逻辑运算



逻辑运算

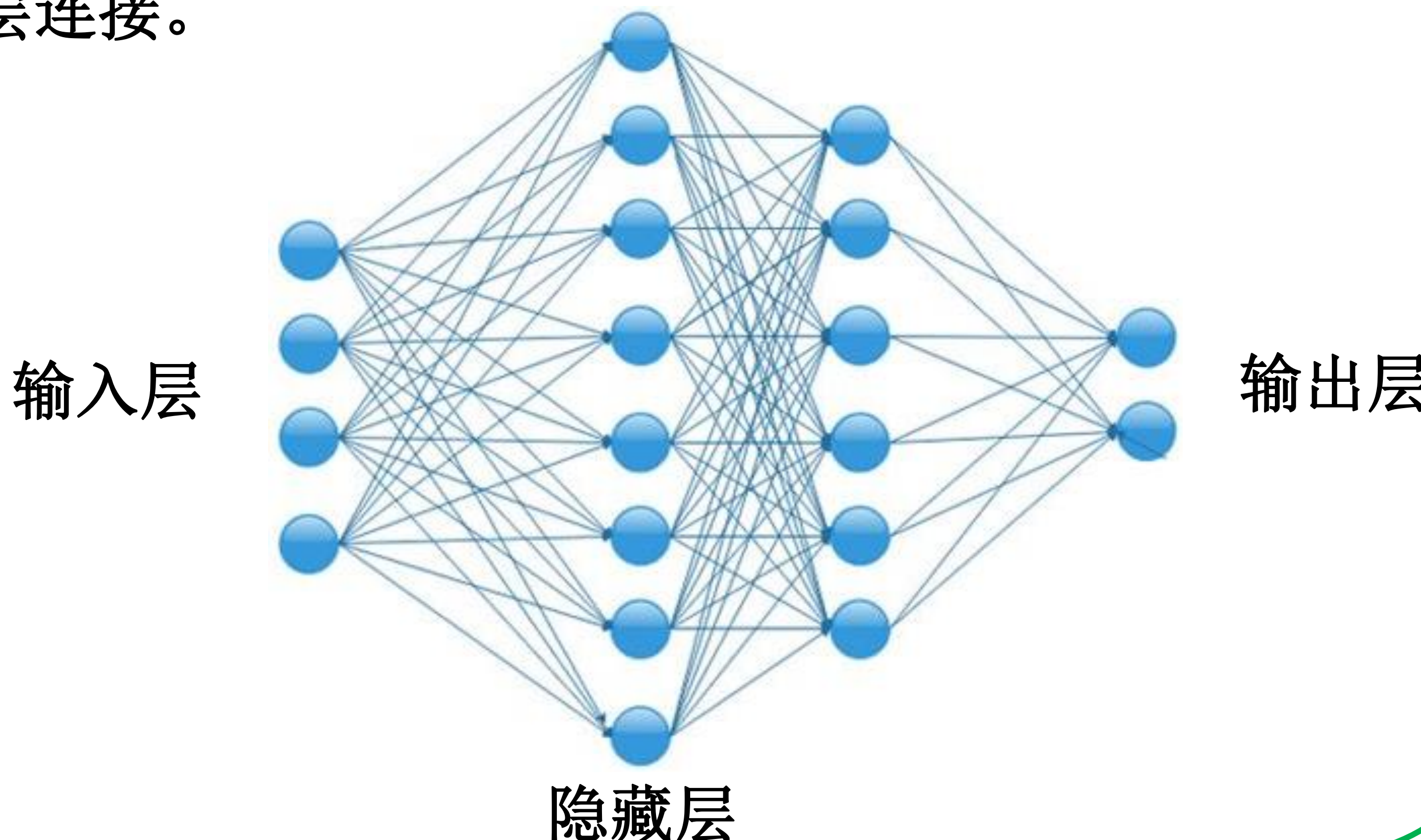


多层神经网络

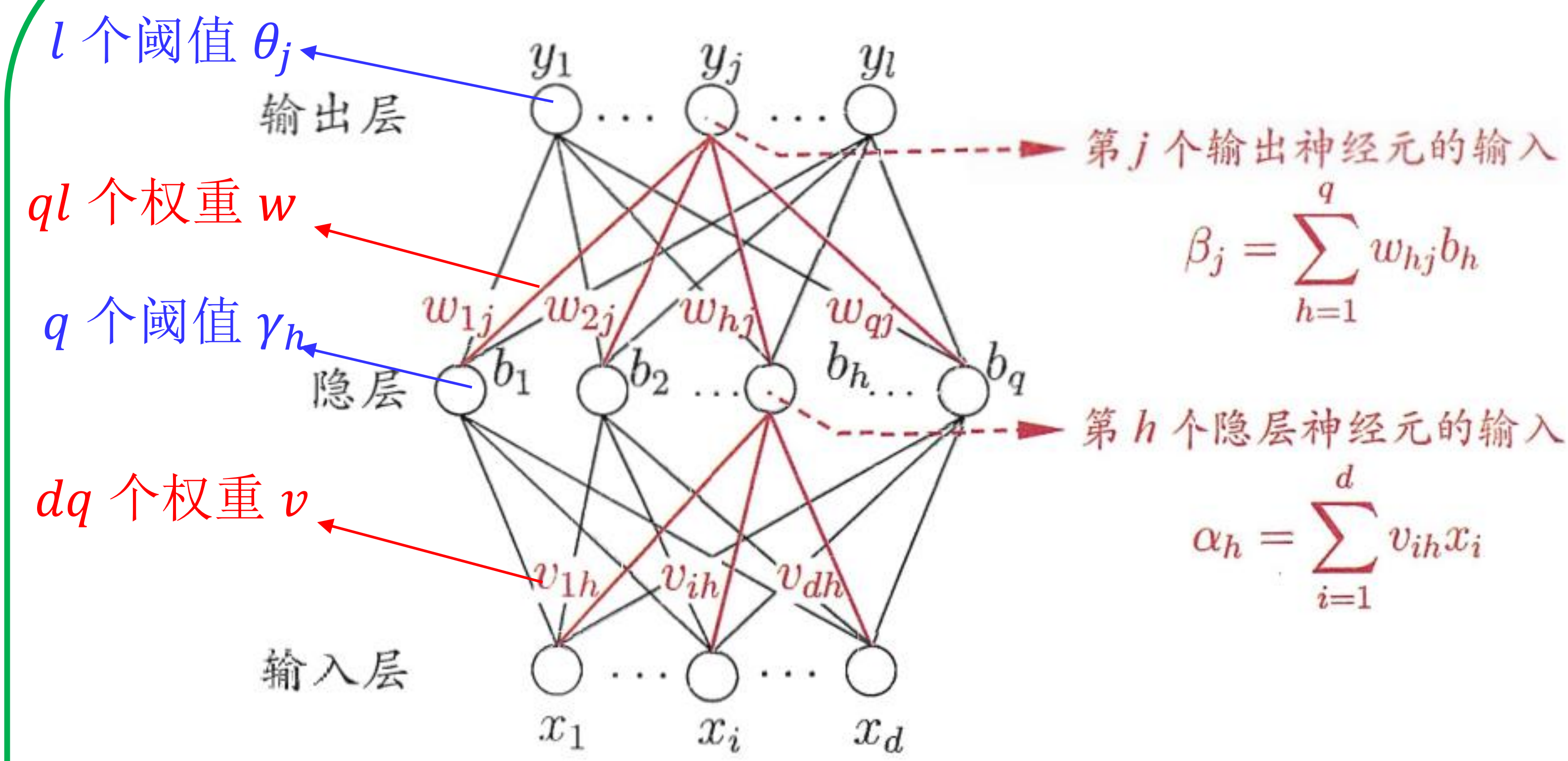
结构	决策区域类型	区域形状	异或问题
无隐层 	由一超平面分成两个		
单隐层 	开凸区域或闭凸区域		
双隐层 	任意形状（其复杂度由单元数目确定）		

多层前馈神经网络

多层前馈神经网络: 每层神经元与下一层神经元完全相连，神经元之间不存在同层连接，也不存在跨层连接。



误差逆传播算法



总参数：$l + ql + q + dq$

图 5.7 BP 网络及算法中的变量符号

误差逆传播算法

BP算法优化准则：

$$\min_{\theta,\omega,\gamma,\nu} \quad E_k = \frac{1}{2}\sum_{j=1}(\hat{y}_j^k - y_j^k)^2$$

其中预测值$\hat{y}_j^k$与权重$\theta, \omega, \gamma, \nu$相关。

梯度下降法：

$$\frac{\partial E_k}{\partial \theta}, \quad \frac{\partial E_k}{\partial \omega}, \quad \frac{\partial E_k}{\partial \gamma}, \quad \frac{\partial E_k}{\partial \nu}$$

更新ω

$E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$；

$\hat{y}_j^k = f(\beta_j - \theta_j)$；　$\beta_j = \sum_{h=1}^{q} \omega_{hj} b_h$

链式法则：$\frac{\partial E_k}{\partial \omega_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \frac{\partial \hat{y}_j^k}{\partial \beta_j} \frac{\partial \beta_j}{\partial \omega_{hj}}$

$$\frac{\partial E_k}{\partial \hat{y}_j^k} = \frac{\partial(\frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2)}{\partial \hat{y}_j^k} = \hat{y}_j^k - y_j^k$$

因为 $f'(x) = f(x)(1 - f(x))$

$$\frac{\partial \hat{y}_j^k}{\partial \beta_j} = f(\beta_j - \theta_j)(1 - f(\beta_j - \theta_j)) = \hat{y}_j^k(1 - \hat{y}_j^k)$$

更新ω

$$E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2;$$

$$\hat{y}_j^k = f(\beta_j - \theta_j);\quad \beta_j = \sum_{h=1}^{q} \omega_{hj} b_h$$

链式法则：$\frac{\partial E_k}{\partial \omega_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k}\frac{\partial \hat{y}_j^k}{\partial \beta_j}\frac{\partial \beta_j}{\partial \omega_{hj}}$

令 $g_j = -\frac{\partial E_k}{\partial \hat{y}_j^k}\frac{\partial \hat{y}_j^k}{\partial \beta_j} = (y_j^k - \hat{y}_j^k)\hat{y}_j^k(1-\hat{y}_j^k)$

$$\frac{\partial \beta_j}{\partial \omega_{hj}} = \frac{\partial(\sum_{h=1}^{q} \omega_{hj} b_h)}{\partial \omega_{hj}} = b_h$$

更新ω

$E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$；

$\hat{y}_j^k = f(\beta_j - \theta_j)$； $\beta_j = \sum_{h=1}^{q} \omega_{hj} b_h$

链式法则： $\frac{\partial E_k}{\partial \omega_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \frac{\partial \hat{y}_j^k}{\partial \beta_j} \frac{\partial \beta_j}{\partial \omega_{hj}} = -g_j b_h$

$$\omega_{hj} \leftarrow \omega_{hj} + \eta g_j b_h$$

更新 θ

$E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$； $\hat{y}_j^k = f(\beta_j - \theta_j)$；

链式法则： $\frac{\partial E_k}{\partial \theta_j} = \frac{\partial E_k}{\partial \hat{y}_j^k} \frac{\partial \hat{y}_j^k}{\partial \theta_j} = -\frac{\partial E_k}{\partial \hat{y}_j^k} \frac{\partial \hat{y}_j^k}{\partial \beta_j} = \boldsymbol{g_j}$

$$\boldsymbol{\theta_j} \leftarrow \boldsymbol{\theta_j} - \boldsymbol{\eta g_j}$$

同理可得

$$\frac{\partial E_k}{\partial \gamma}, \quad \frac{\partial E_k}{\partial \nu}$$

BP算法小结

核心思想：利用前向传播，计算第 n 层输出值

优化目标：输出值和实际值的残差。

计算方法：将残差按影响逐步传递回第 $n-1, n-2, \cdots, 2$层，以修正各层参数。（即所谓的误差逆传播）

主要工具：链式法则（复合函数求偏导）。

BP算法局限性

- 容易过拟合！

  早停、正则化

- 容易陷入局部最优！

  选取多次初值、随机梯度下降法

- 难以设置隐层个数！

  试错法