人工神经网络

王晓丽

计算机科学与技术学院　副教授

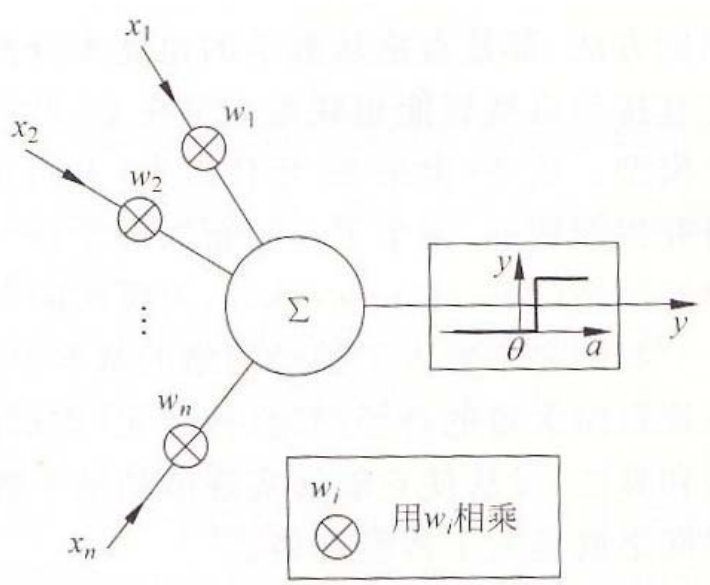
课程目录

- 1. 上次课程回顾
- 2. 监督学习和BP算法
- 3. BP算法的局限性
- 4. BP算法的改进
- 5. 训练与测试

人工神经网络

- 信号累积:

$$y = f\left(\sum_{i=1}^{n} w_i x_i - \theta\right)$$



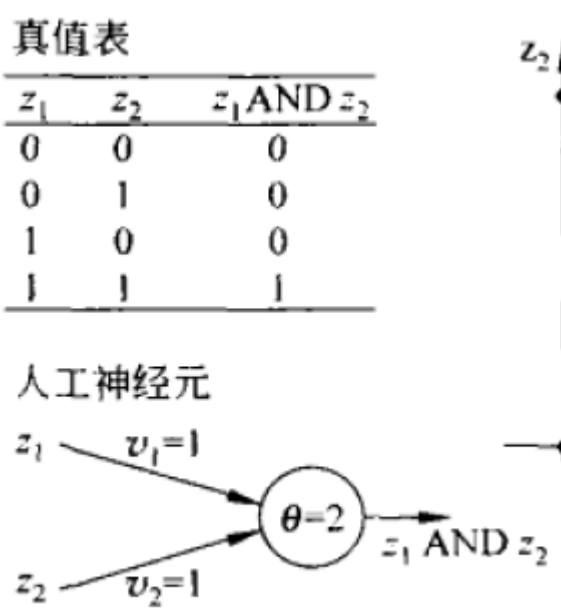
- θ为阈值，f（X）是激活函数

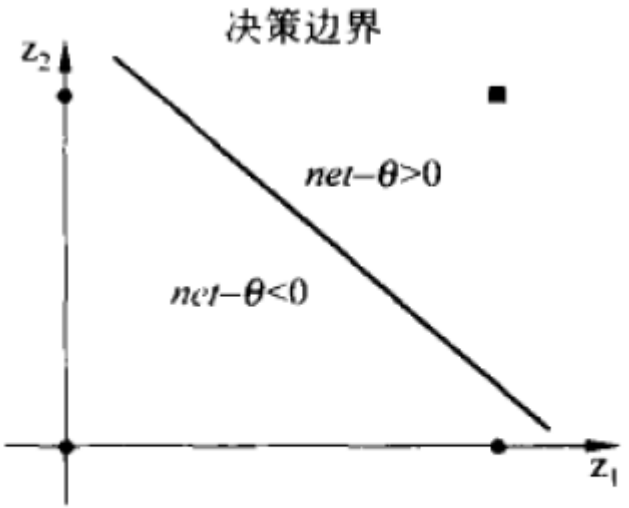
单层感知器

- “与”感知器

真值表

z_1	z_2	z_1 AND z_2
0	0	0
0	1	0
1	0	0
1	1	1





(a) AND 感知器

单层感知器

- “异或”感知器

真值表

z_1	z_2	z_1 XOR z_2
0	0	0
0	1	1
1	0	1
1	1	0

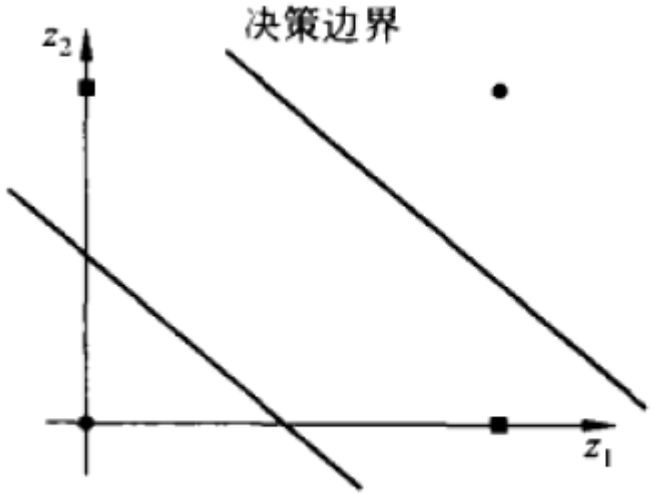
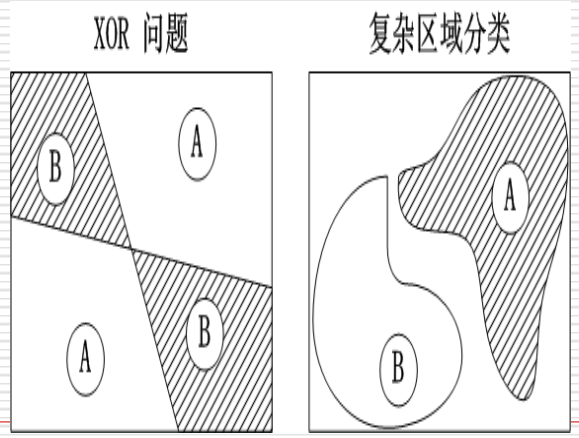


图 2.5　XOR 决策边界

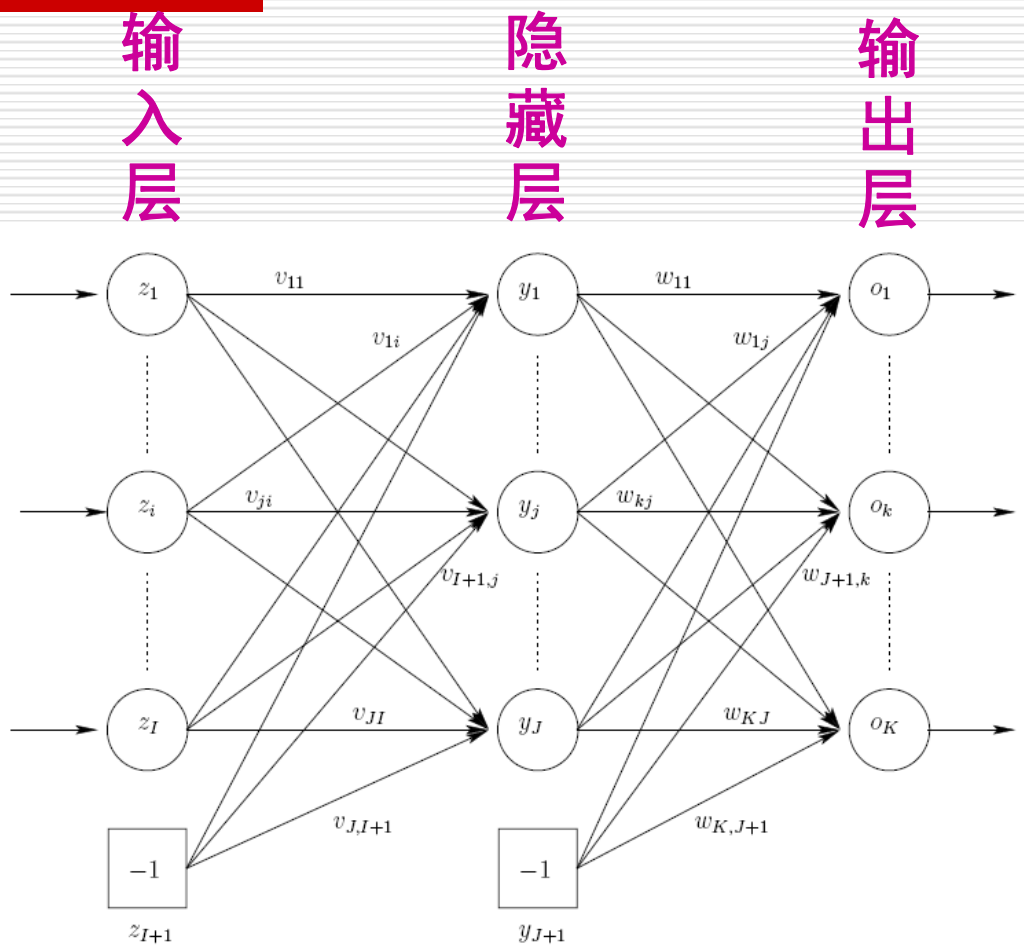
单层感知器

- **局限性：**单层感知器无法解决“异或”问题。
  - 由感知器的几何意义可知，其确定的分类判别方程式是线性方程，因而只能解决线性可分问题，而不能解决线性不可分问题。
  - 这里“线性可分”是指两类样本可以用直线、平面或者超平面分开，否则称为线性不可分。



多层感知器网络

- ✓ 包括输入层、隐藏层和输出层
- ✓ 可以有多个隐层
- ✓ 信息只能从左一层单元传递到相应的右一层单元。



人工神经网络的类型

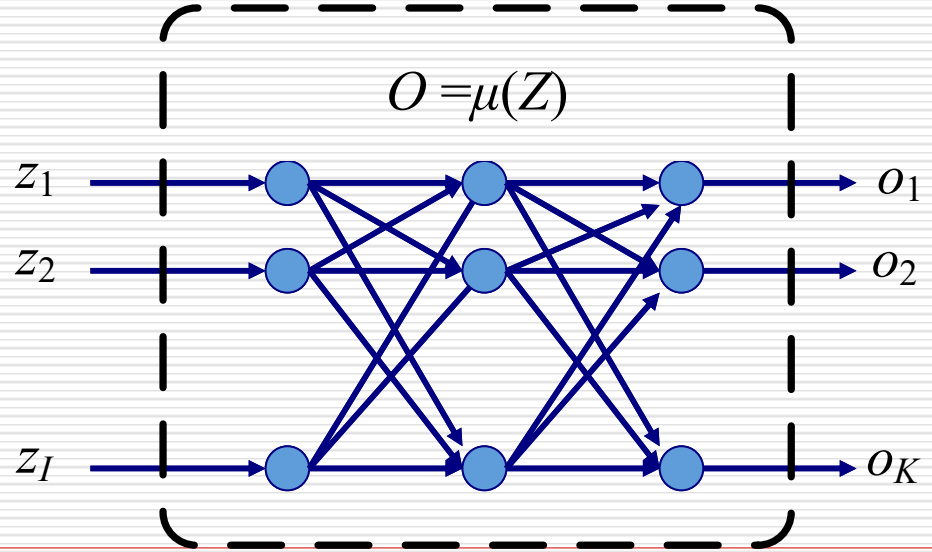
- 神经元之间如何连接？ 拓扑结构
  - 人工神经网络可以看成是以人工神经元为结点，用有向加权弧连接起来的有向图
- 如何确定权重？ 学习方法
  - 监督学习：输入向量的期望输出已知
  - 非监督学习：输入向量的期望输出未知

课程目录

- 1. 上次课程回顾
- **2. 监督学习和BP算法**
- 3. BP算法的局限性
- 4. BP算法的改进
- 5. 训练与测试

2. 监督学习

- ✓ 监督学习需要一个训练集
- ✓ 训练集由输入向量和与每一个输入向量相关联的目标向量组成

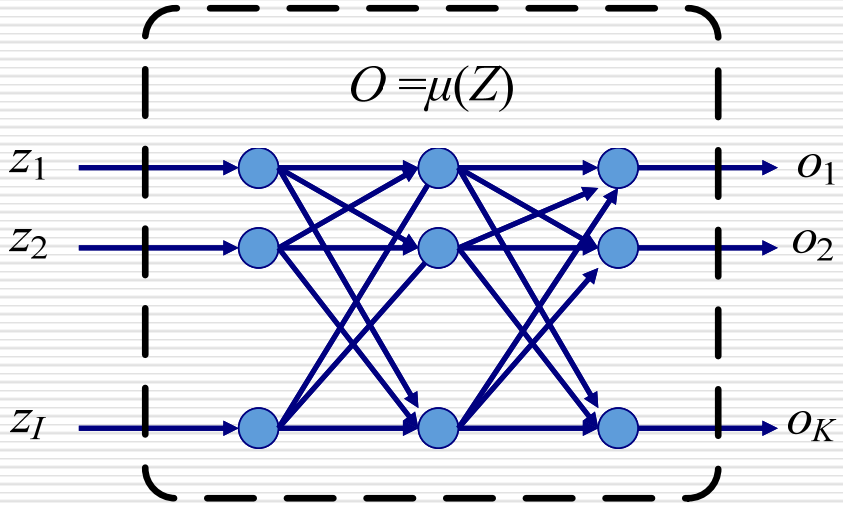


给定一组输入Z和输出O的样本，学习从Z到O的未知函数μ

2. 监督学习

神经网络的监督学习

- 将神经网络学习问题转化为输出误差最小化的优化问题
- 调整权值，使误差函数最小化：



$$\varepsilon=\frac{1}{2}\sum_{p=1}^{N}\left(t_p-o_p\right)^2$$

其中，t_p和o_p分别是第p个样本的目标输出和真实输出；N是训练集中的样本总数。

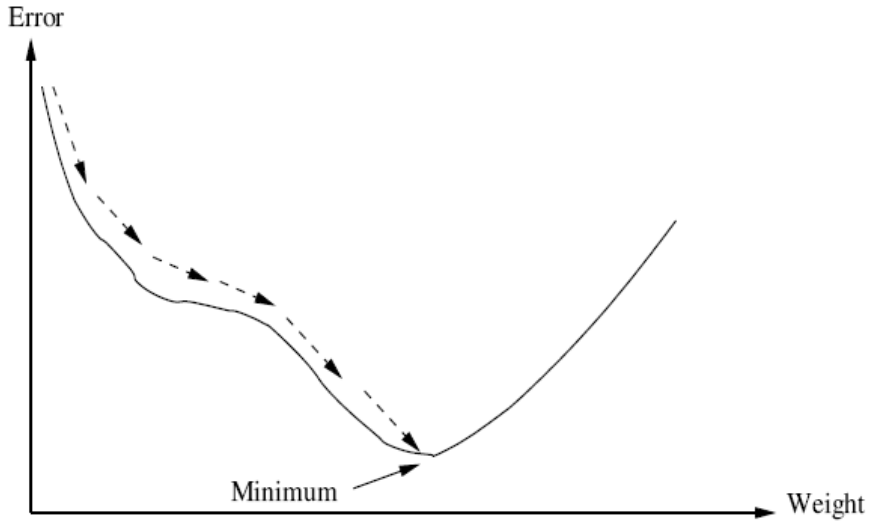
2. BP算法

- 梯度下降优化算法
  - ✓ 目标：找到使误差函数ε最小的权值w
  - ✓ 计算ε的梯度，沿着负梯度方向搜索
  - ✓ 权值更新公式为：

$$\mathrm{w}(t)=w(t-1)+\Delta w(t)$$

其中：$\Delta w(t)=\eta\left(-\frac{\partial \varepsilon}{\partial w}\right)$

步长 η 称为学习率



2. BP算法

- 梯度下降优化算法
  - ✓ 反向传播（BP）算法分为两个阶段
    - 前向传播：对每一个训练样本，从输入层经隐藏层逐层正向计算各结点的输出；
    - 反向传播：由输出误差逐层反向计算隐藏层各结点的误差，并用此误差修正前层的权值。
  - ✓ 反向传播算法采用梯度下降法修正权值，因此要求激活函数可微。采用Sigmoid函数作为激活函数。

2. BP算法

- 求导习题

$$1.\ y = e^x$$

$$2.\ y = \frac{1}{x}$$

$$3.\ y = \frac{1}{2}(x-a)^2$$

$$4.\ y = \frac{1}{1+e^{-x}}$$

2. BP算法

- 求导习题

$$1.\ y = e^{x}$$

$$2.\ y = \frac{1}{x}$$

$$3.\ y = \frac{1}{2}(x-a)^{2}$$

$$4.\ y = \frac{1}{1+e^{-x}}$$

$$5.\ y = \frac{1}{1+e^{-(ax1+bx2-\theta)}}$$

2. BP算法

- 求导习题

$$1.\ y = e^{x}$$

$$2.\ y = \frac{1}{x}$$

$$3.\ y = \frac{1}{2}(x-a)^{2}$$

$$4.\ y = \frac{1}{1+e^{-x}}$$

$$5.\ y = \frac{1}{1+e^{-(ax1+bx2-\theta)}}$$

$$6.\ x1 = \frac{1}{1+e^{-(cx3+dx4-\theta 2)}}$$

2. BP算法

- **BP算法的学习过程如下：**

步骤1：初始化网络权重

每两个神经元之间的网络连接权重w_{ij}被初始化为一个很小的随机数（例如-1.0~1.0或者-0.5~0.5），同时，每个神经元有一个阈值，也被初始化为一个随机数。

步骤2：前向传播

根据训练样本的输入，计算各层神经元节点的输出。每个神经元的计算方法相同，都是由其输入的线性组合得到，公式如下：

$$O_j = \frac{1}{1+e^{-s_j}} = \frac{1}{1+e^{-(\sum_i w_{ij} O_i - \theta_j)}}$$

2. BP算法

- **BP算法的学习过程如下：**

步骤3：计算误差

计算网络的实际输出和期望输出的误差 $\varepsilon = \frac{1}{2}(O-T)^2$

步骤4：反向传播

从输出层反向计算到第一个隐层，并沿着使误差减小方向调整网络中各神经元的连接权值及阈值。 公式如下：

$$\mathrm{w}(t) = w(t-1) + \Delta w(t) \qquad \Delta w(t) = \eta\left(-\frac{\partial \varepsilon}{\partial w}\right)$$

$$\theta(t) = \theta(t-1) + \Delta\theta(t) \qquad \Delta\theta(t) = \eta\left(-\frac{\partial \varepsilon}{\partial \theta}\right)$$

2. BP算法

- **BP算法的学习过程如下：**

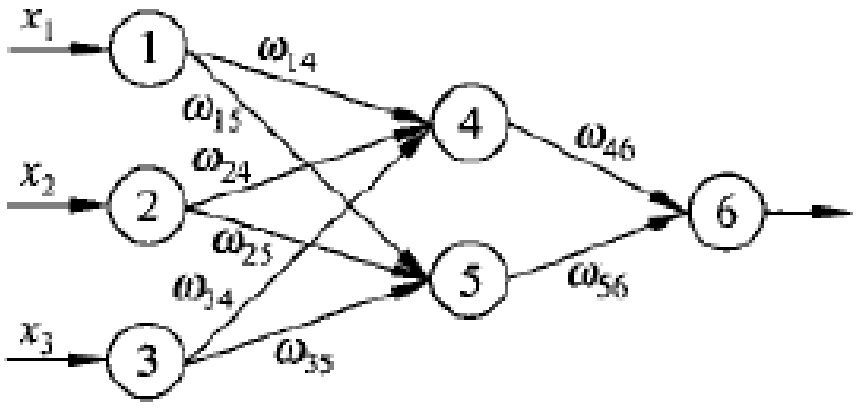
步骤5：判断结束

对于每个样本，如果最终的输出误差小于可接受的范围或迭代次数t达到了一定的阈值，则选取下一个样本，转到步骤2重新继续执行，否则，迭代次数t加1，然后转向步骤2继续使用当前样本进行训练。

2. BP算法

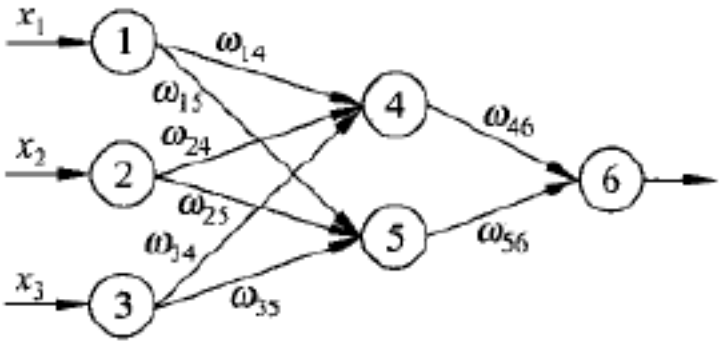
BP算法举例

已知一个前馈神经网络如下图所示，设学习率η=0.9，当前训练样本为 $x=(1,0,1)$，期望输出结果为1。求该网络在当前训练样本下的训练过程。



2. BP算法

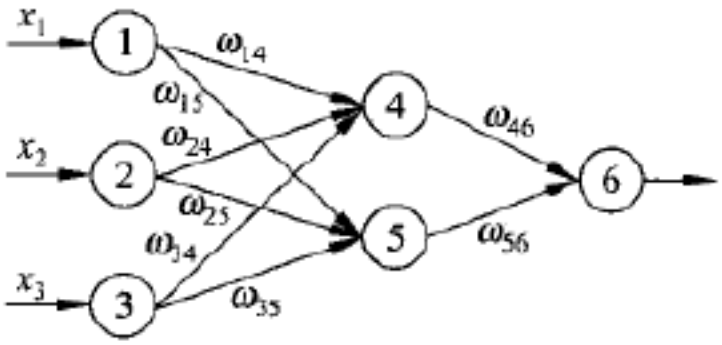
- **BP算法的学习过程如下：**



步骤1：初始化网络权重

w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	0.4	-0.2	-0.1

2. BP算法



- **BP算法的学习过程如下：**

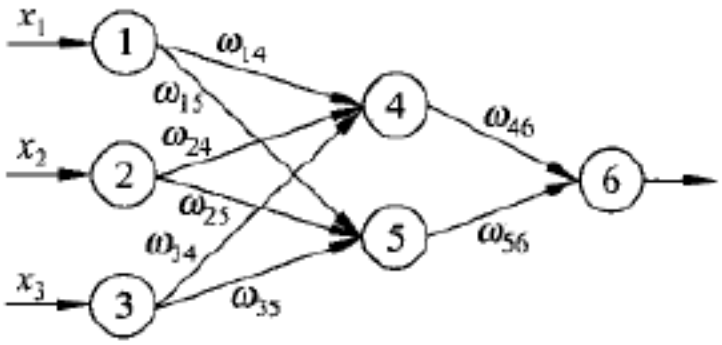
步骤1：初始化网络权重

w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	0.4	-0.2	-0.1

步骤2：前向传播 $x=(1,0,1)$，写出前向传播的公式

神经元 j	总输入S_j	输出O_j
4		
5		
6		

2. BP算法



- **BP算法的学习过程如下：**

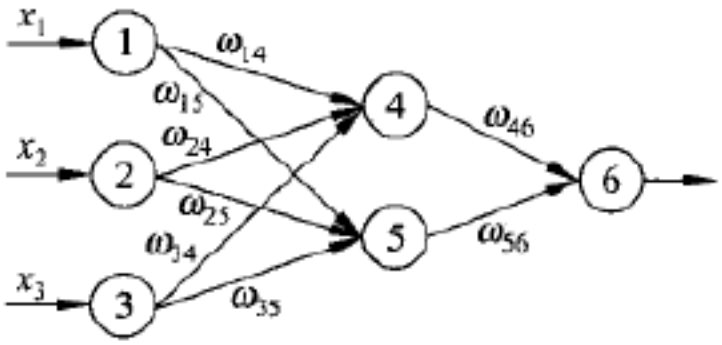
步骤1：初始化网络权重

w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	0.4	-0.2	-0.1

步骤2：前向传播

神经元 j	总输入S_j	输出O_j
4	1*0.2+0*0.4+1*(-0.5)-0.4=-0.7	0.332
5		
6		

2. BP算法



- **BP算法的学习过程如下：**

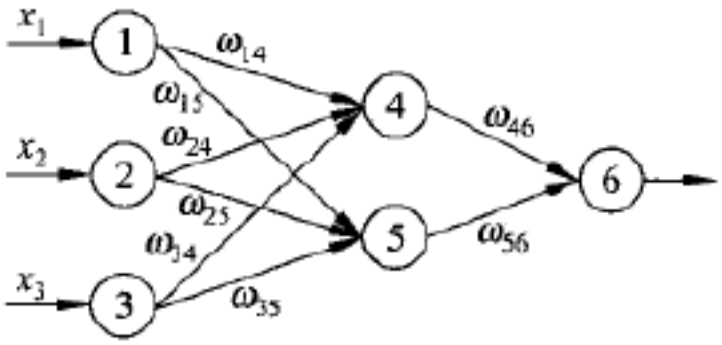
步骤1：初始化网络权重

w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	0.4	-0.2	-0.1

步骤2：前向传播

神经元 j	总输入S_j	输出O_j
4	1*0.2+0*0.4+1*(-0.5)-0.4=-0.7	0.332
5	1*(-0.3)+0*(0.1)+1*0.2+0.2=0.1	0.525
6		

2. BP算法



- **BP算法的学习过程如下：**

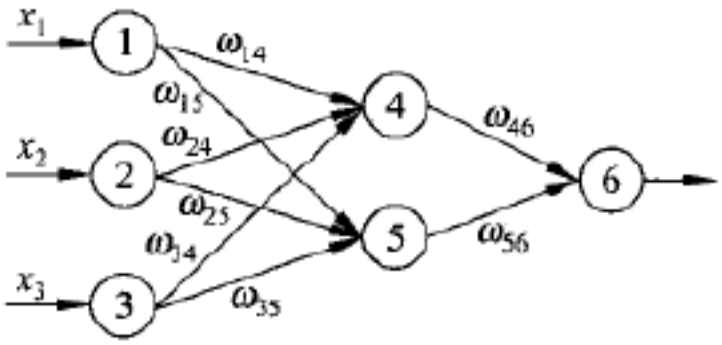
步骤1：初始化网络权重

w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	0.4	-0.2	-0.1

步骤2：前向传播

神经元 j	总输入S_j	输出O_j
4	1*0.2+0*0.4+1*(-0.5)-0.4=-0.7	0.332
5	1*(-0.3)+0*(0.1)+1*0.2+0.2=0.1	0.525
6	(-0.3)*0.332+(-0.2)*0.525+0.1=-0.105	0.474

2. BP算法

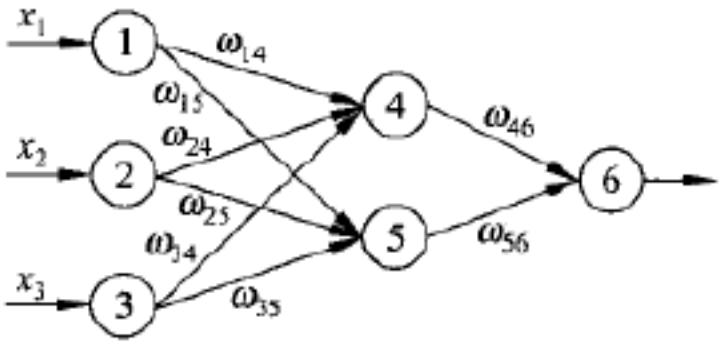


- **BP算法的学习过程如下：**

步骤3：计算误差

$$\varepsilon = \frac{1}{2}(O-T)^2 = \frac{1}{2}(0.474-1)^2 = 0.138$$

2. BP算法



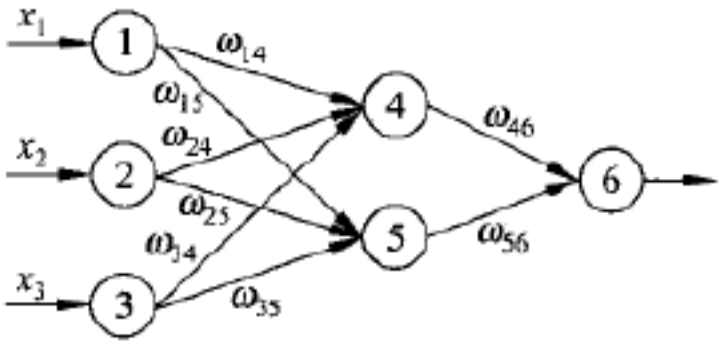
- **BP算法的学习过程如下：**

步骤4：反向传播

步骤4.1：更新最后一层的权重和阈值

$$w'_{46} = w_{46} + \Delta w_{46} \qquad \Delta w_{46} = \eta\left(-\frac{\partial \varepsilon}{\partial w_{46}}\right)$$

2. BP算法



- **BP算法的学习过程如下：**

步骤4：反向传播

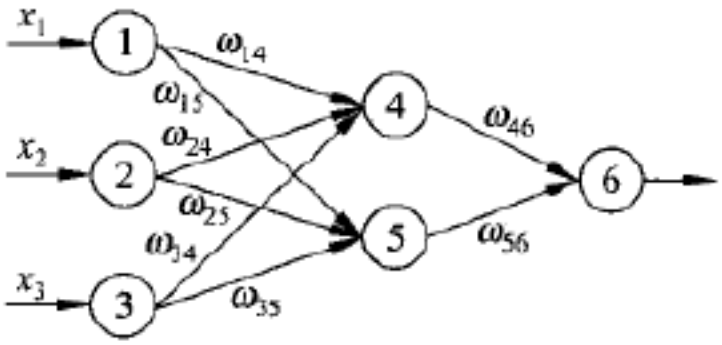
步骤4.1：更新最后一层的权重和阈值

$$w'_{46} = w_{46} + \Delta w_{46} \qquad \Delta w_{46} = \eta\left(-\frac{\partial \varepsilon}{\partial w_{46}}\right)$$

$$\frac{\partial \varepsilon}{\partial w_{46}} = \frac{\partial \varepsilon}{\partial o_6} \bullet \frac{\partial o_6}{\partial s_6} \bullet \frac{\partial s_6}{\partial w_{46}} = (O_6 - T)O_6(1-O_6)O_4$$

$$w'_{46} = w_{46} + \Delta w_{46} = w_{46} + \eta\left(-\frac{\partial \varepsilon}{\partial w_{46}}\right) = w_{46} + \eta(T - O_6)O_6(1-O_6)O_4$$

2. BP算法



- **BP算法的学习过程如下：**

步骤4：反向传播

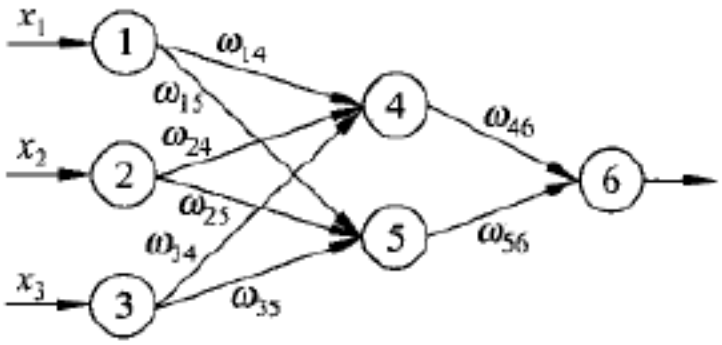
步骤4.1：更新最后一层的权重和阈值

$$w'_{46} = w_{46} + \Delta w_{46} = w_{46} + \eta\left(-\frac{\partial \varepsilon}{\partial w_{46}}\right) = w_{46} + \eta(T - O_6)O_6(1-O_6)O_4$$

$$w'_{56} = w_{56} + \Delta w_{56} = w_{56} + \eta\left(-\frac{\partial \varepsilon}{\partial w_{56}}\right) = w_{56} + \eta(T - O_6)O_6(1-O_6)O_5$$

权重	更新后的值
w_{46}	-0.3+0.9*(1-0.474)*0.474*(1-0.474)*0.332=-0.261
w_{56}	-0.2+0.9*(1-0.474)*0.474*(1-0.474)*0.525=-0.138

2. BP算法



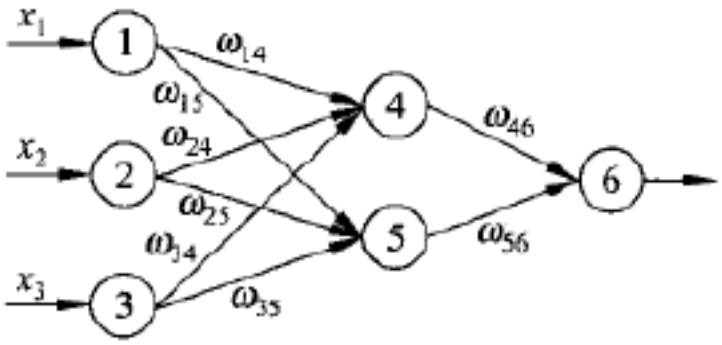
- **BP算法的学习过程如下：**

步骤4：反向传播

步骤4.1：更新最后一层的权重和阈值

$$\theta_6' = \theta_6 + \Delta\theta_6 \qquad \Delta\theta_6 = \eta\left(-\frac{\partial\varepsilon}{\partial\theta_6}\right)$$

2. BP算法



- **BP算法的学习过程如下：**

步骤4：反向传播

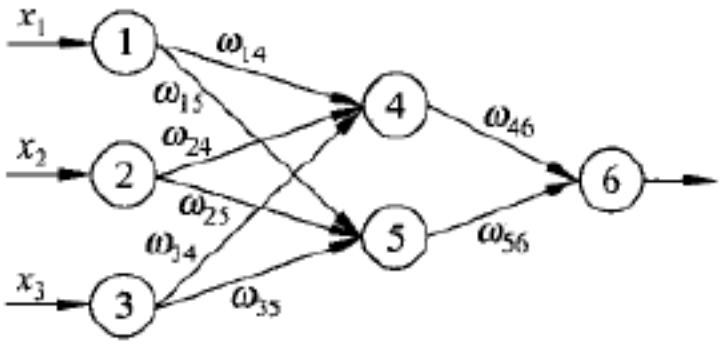
步骤4.1：更新最后一层的权重和阈值

$$\theta_6' = \theta_6 + \Delta\theta_6 \qquad \Delta\theta_6 = \eta\left(-\frac{\partial\varepsilon}{\partial\theta_6}\right)$$

$$\frac{\partial\varepsilon}{\partial\theta_6} = \frac{\partial\varepsilon}{\partial o_6}\bullet\frac{\partial o_6}{\partial s_6}\bullet\frac{\partial s_6}{\partial\theta_6} = (O_6 - T)O_6(1-O_6)(-1)$$

$$\theta_6' = \theta_6 + \Delta\theta_6 = \theta_6 + \eta\left(-\frac{\partial\varepsilon}{\partial\theta_6}\right) = \theta_6 + \eta(T - O_6)O_6(1-O_6)(-1)$$

2. BP算法



- **BP算法的学习过程如下：**

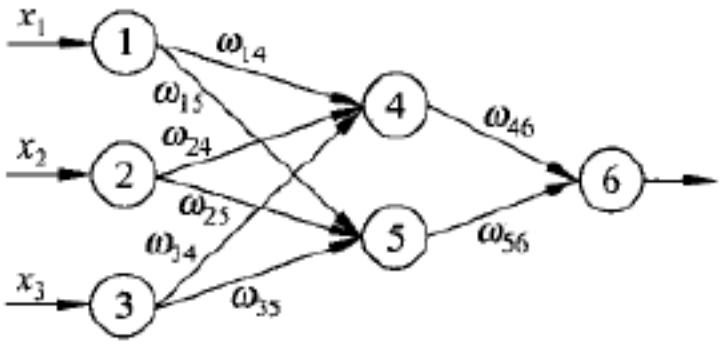
步骤4：反向传播

步骤4.1：更新最后一层的权重和阈值

$$\theta_6' = \theta_6 + \Delta\theta_6 = \theta_6 + \eta\left(-\frac{\partial\varepsilon}{\partial\theta_6}\right) = \theta_6 + \eta(T - O_6)O_6(1-O_6)(-1)$$

权重	更新后的值
θ_6	-0.1+0.9*(1-0.474)*0.474*(1-0.474)*(-1)=-0.218

2. BP算法



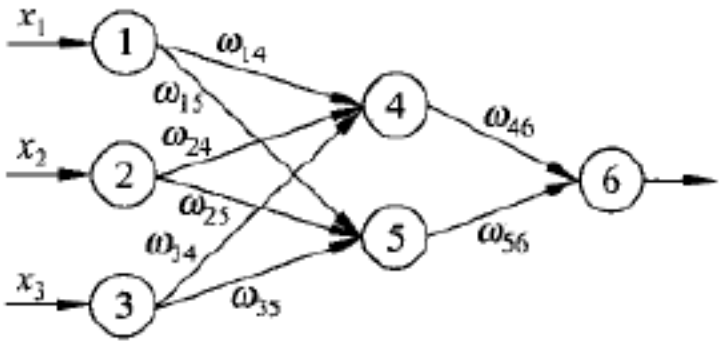
- **BP算法的学习过程如下：**

步骤4：反向传播

步骤4.2：更新倒数第二层的权重和阈值

$$w'_{14} = w_{14} + \Delta w_{14} \qquad \Delta w_{14} = \eta\left(-\frac{\partial \varepsilon}{\partial w_{14}}\right)$$

2. BP算法



- **BP算法的学习过程如下：**

步骤4：反向传播

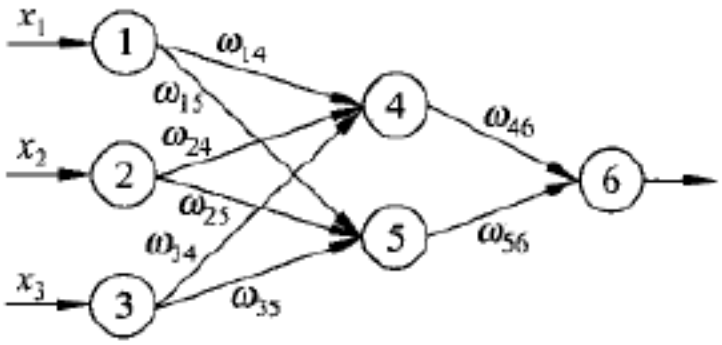
步骤4.2：更新倒数第二层的权重和阈值

$$w'_{14} = w_{14} + \Delta w_{14} \qquad \Delta w_{14} = \eta\left(-\frac{\partial \varepsilon}{\partial w_{14}}\right)$$

$$\frac{\partial \varepsilon}{\partial w_{14}} = \frac{\partial \varepsilon}{\partial o_6} \bullet \frac{\partial o_6}{\partial s_6} \bullet \frac{\partial s_6}{\partial o_4} \bullet \frac{\partial o_4}{\partial s_4} \bullet \frac{\partial s_4}{\partial w_{14}} = (O_6 - T)O_6(1-O_6)w_{46}O_4(1-O_4)x_1$$

$$w'_{14} = w_{14} + \Delta w_{14} = w_{14} + \eta\left(-\frac{\partial \varepsilon}{\partial w_{14}}\right) = w_{14} + \eta(T - O_6)O_6(1-O_6)w_{46}O_4(1-O_4)x_1$$

2. BP算法



- **BP算法的学习过程如下：**

步骤4：反向传播

步骤4.2：更新倒数第二层的权重和阈值

$$w'_{14} = w_{14} + \eta(T - O_6)O_6(1-O_6)w_{46}O_4(1-O_4)x_1$$

$$w'_{15} = w_{15} + \eta(T - O_6)O_6(1-O_6)w_{56}O_5(1-O_5)x_1$$

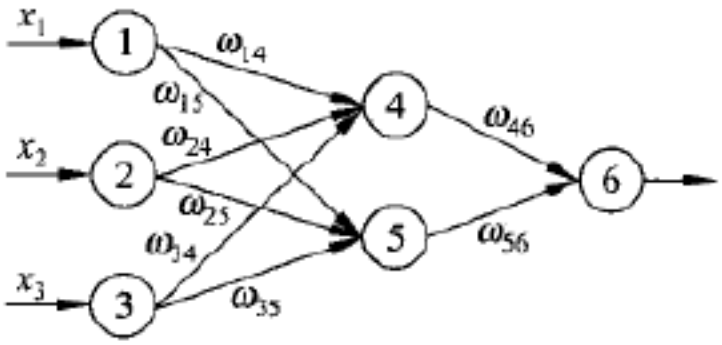
$$w'_{24} = w_{24} + \eta(T - O_6)O_6(1-O_6)w_{46}O_4(1-O_4)x_2$$

$$w'_{25} = w_{25} + \eta(T - O_6)O_6(1-O_6)w_{56}O_5(1-O_5)x_2$$

$$w'_{34} = w_{34} + \eta(T - O_6)O_6(1-O_6)w_{46}O_4(1-O_4)x_3$$

$$w'_{35} = w_{35} + \eta(T - O_6)O_6(1-O_6)w_{56}O_5(1-O_5)x_3$$

2. BP算法



- **BP算法的学习过程如下：**

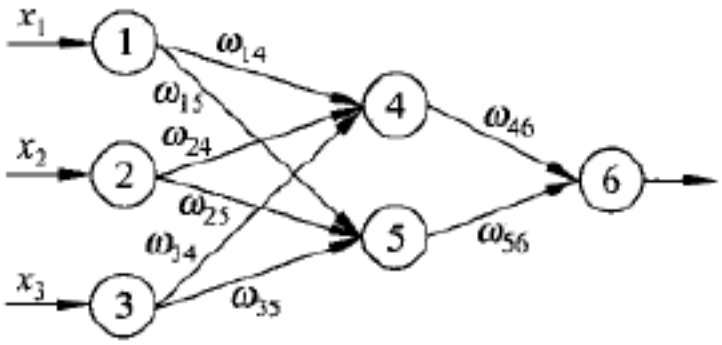
步骤4：反向传播

步骤4.2：更新最后一层的权重和阈值

$$w'_{14} = w_{14} + \eta(T - O_6)O_6(1-O_6)w_{46}O_4(1-O_4)x_1$$

权重	更新后的值
w_{14}	0.2+0.9*(1-0.474)*0.474*(1-0.474)*(-0.3)*0.332* (1-0.332)*1=0.192
w_{15}	-0.306
w_{24}	0.4
w_{25}	0.1
w_{34}	-0.508
w_{35}	0.194

2. BP算法



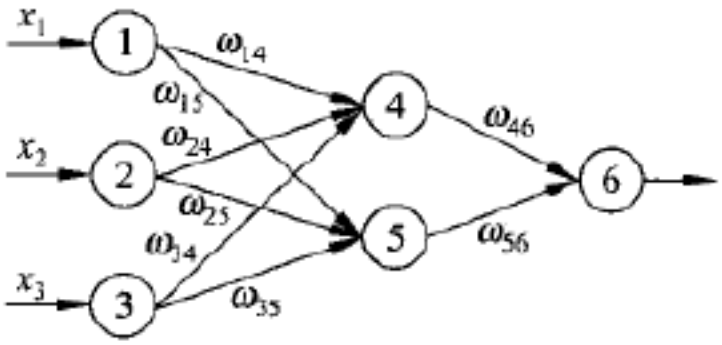
- **BP算法的学习过程如下：**

步骤4：反向传播

步骤4.2：更新倒数第二层的权重和阈值

$$\theta_4' = \theta_4 + \Delta\theta_4 \qquad \Delta\theta_4 = \eta\left(-\frac{\partial\varepsilon}{\partial\theta_4}\right)$$

2. BP算法



- **BP算法的学习过程如下：**

步骤4：反向传播

步骤4.2：更新倒数第二层的权重和阈值

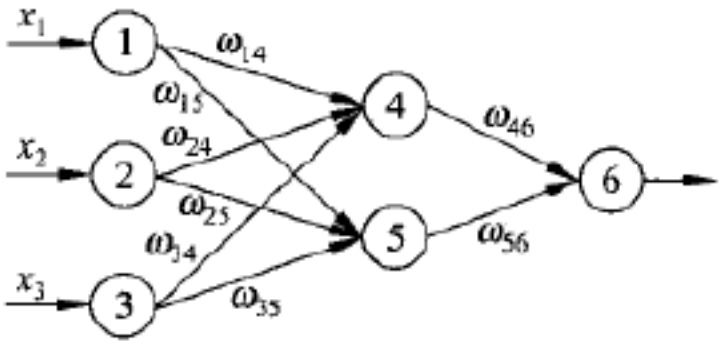
$$\theta_4' = \theta_4 + \Delta\theta_4 = \theta_4 + \eta\left(-\frac{\partial\varepsilon}{\partial\theta_4}\right) = \theta_4 + \eta(T-O_6)O_6(1-O_6)w_{46}O_4(1-O_4)(-1)$$

$$\theta_5' = \theta_5 + \Delta\theta_5 = \theta_5 + \eta\left(-\frac{\partial\varepsilon}{\partial\theta_5}\right) = \theta_5 + \eta(T-O_6)O_6(1-O_6)w_{56}O_5(1-O_5)(-1)$$

权重	更新后的值
θ_4	0.4+0.9*(1-0.474)*0.474*(1-0.474)*(-0.3)*0.332* (1-0.332)*(-1)=0.408
θ_5	-0.2+0.9*(1-0.474)*0.474*(1-0.474)*(-0.2)*0.525* (1-0.525)*(-1)=-0.194

2. BP算法

- **BP算法的学习过程如下：**



步骤5：判断结束

对于每个样本，如果最终的输出误差小于可接受的范围或迭代次数 t 到了设置的上限，则选取下一个样本，转到步骤2重新继续执行，否则，迭代次数 t 加1，然后转向步骤2继续使用当前样本进行训练。

2. BP算法

- 已知两类蠓虫的部分数据如下：
  - 要求对数据进行建模，对触角和翼长分别为(1.24,1.80)，(1.28,1.84)与(1.40,2.04)的3个标本，其类别应该是什么？

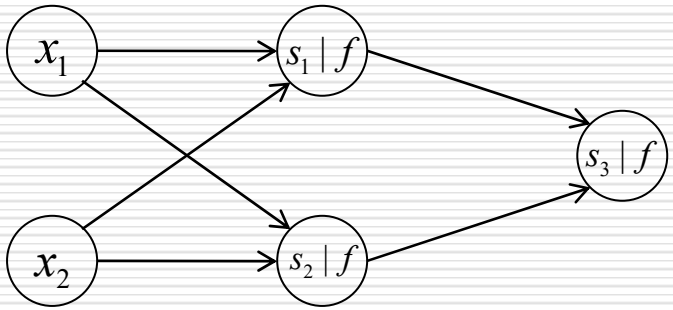
翼长	触角长	目标值
1.78	1.14	0.9
1.96	1.18	0.9
1.86	1.20	0.9
1.72	1.24	0.1
2.00	1.26	0.9
2.00	1.28	0.9
1.96	1.30	0.9
1.74	1.36	0.1

翼长	触角长	目标值
1.64	1.38	0.1
1.82	1.38	0.1
1.90	1.38	0.1
1.70	1.40	0.1
1.82	1.48	0.1
1.82	1.54	0.1
2.08	1.56	0.1

2. BP算法

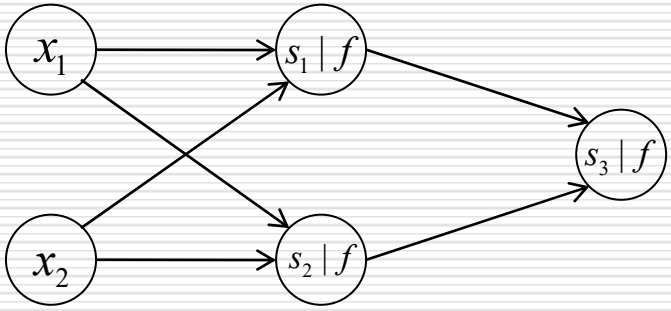
- 已知两类蠓虫的部分数据如下：
  - 要求对数据进行建模，对触角和翼长分别为(1.24,1.80)，(1.28,1.84)与(1.40,2.04)的3个标本，其类别应该是什么？

翼长	触角长	目标值
1.78	1.14	0.9
1.96	1.18	0.9
1.86	1.20	0.9
1.72	1.24	0.1
2.00	1.26	0.9
2.00	1.28	0.9
1.96	1.30	0.9
1.74	1.36	0.1



2. BP算法

- 训练结果



$$s_1 = -5.5921x_1 + 7.5976x_2 - 0.5765$$
$$s_2 = -0.5787x_1 - 0.2875x_2 + 0.2764$$
$$s_3 = -8.4075o_1 + 0.4838o_2 - 3.9829$$

2．BP算法

□ 更新问题

- BP算法采用的是样本更新，即每处理一个样本就更新一次权重和阈值。样本更新的缺陷：样本的顺序对训练结果有较大影响。它会更“偏爱”较后出现的样本。为样本安排一个最优的顺序是非常困难的。

- 解决的办法就是采用周期更新，即处理一遍所有的样本才更新一次权重和阈值。周期更新的好处：可以消除样本顺序对结果的影响。

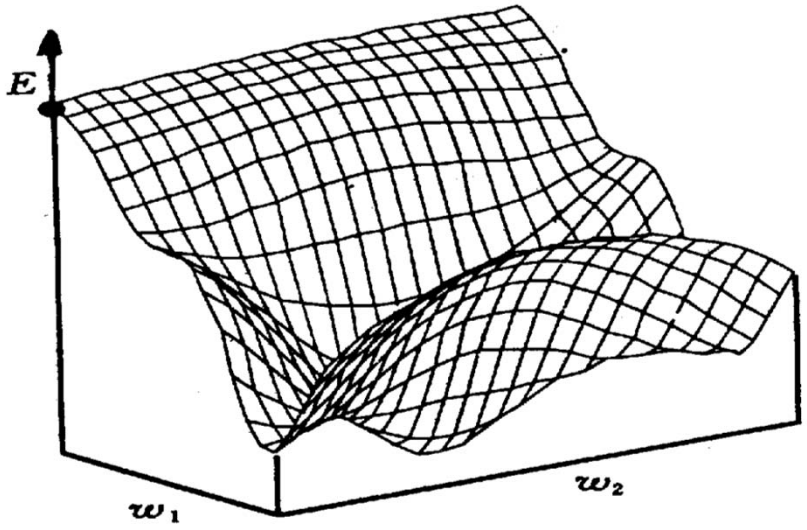
课程目录

- 1. 上次课程回顾
- 2. 监督学习和BP算法
- **3. BP算法的局限性**
- 4. BP算法的改进
- 5. 训练与测试

3. BP算法的局限性

- 误差函数的可调整参数的个数n等于各层权值数加上阈值数。
- 误差E是$n+1$维空间中一个形状极为复杂的曲面，该曲面上的每个点的“高度”对应于一个误差值，每个点的坐标向量对应着n个权值，因此称这样的空间为误差的权空间。

3. BP算法的局限性



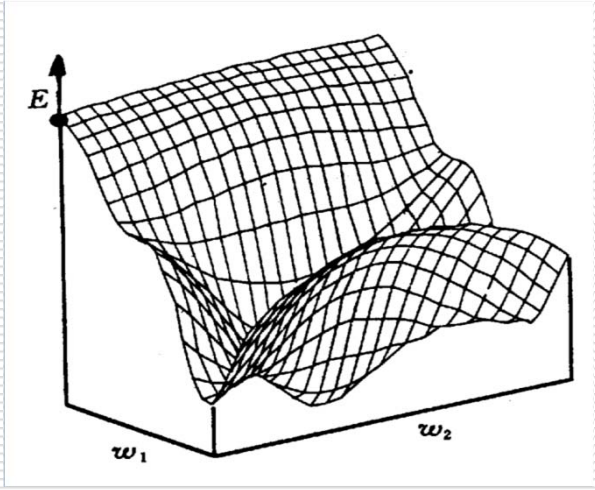
- 曲面的分布特点--------算法的局限性

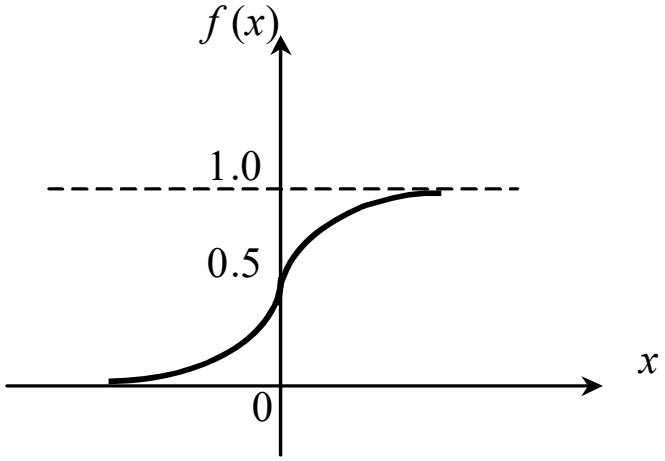
(1)存在平坦区域--------误差下降缓慢，影响收敛速度

(2)存在多个极小点------易陷入局部最小点

3. BP算法的局限性

- 平坦——误差的梯度变化小
- 造成平坦区的原因：各节点的净输入过大





$$\left|\sum_{j=0}^{m} w_{jk} y_j\right| > 3$$

3. BP算法的局限性

- BP算法
  - 以误差梯度下降为权值调整原则
- 导致的结果：
  - 训练经常陷入某个局部极小点而不能自拔，从而使训练无法收敛于给定误差。

4. BP算法的改进

- **BP算法的缺陷总结：**

(1) 易形成局部极小而得不到全局最优；

(2) 训练次数多使得学习效率低，收敛速度慢；

(3) 隐藏层结点的选取缺乏理论指导；

(4) 训练时学习新样本有遗忘旧样本的趋势。

- 针对上述问题，国内外已提出不少有效的改进算法，下面仅介绍其中3种较常用的方法。

4. BP算法的改进

- 改进1 ： 增加动量项
- 改进2： 自适应调节学习率
- 改进3： 引入陡度因子

4. BP算法的改进：增加动量项

- 标准BP算法在修正权值$w(k)$时，只是按k时刻的负梯度方向进行修正，没有考虑积累的经验，即以前的梯度方向，从而使学习过程振荡，收敛缓慢。
- 附加动量法使神经网络在修正权值时不仅考虑误差在梯度上的作用，而且考虑误差变化趋势的影响。

$$W(k+1)=W(k)+\eta[(1-\alpha)\Delta(k)+\alpha\Delta(k-1)]$$

η为学习率，$\eta>0$，α为动量项因子，$0\le\alpha<1$

$\Delta(k)$为k时刻的负梯度，$\Delta(k\text{-}1)$为k-1时刻的负梯度。

- 所加入动量项减小了学习过程的振荡，改善了收敛性。

4. BP算法的改进

- 改进1：增加动量项
- 改进2：自适应调节学习率
- 改进3： 引入陡度因子

4. BP算法的改进：自适应学习率

- 提出的原因：
  - 标准BP算法中，学习率 η 也称为步长，很难确定一个从始至终都合适的最佳学习率。
  - 平坦区域内，η太小会使训练次数增加；
  - 在误差变化剧烈的区域，η太大会因调整量过大而使训练出现振荡，反而使迭代次数增加。

4. BP算法的改进：自适应学习率

- 基本思想：
  - 自适应改变学习率，根据误差变化增大或减小η

- 基本方法：
  - 设一初始学习率，若经过一批次权值调整后使总误差↑，则本次调整无效，令$\eta = \beta\eta\,(\beta<1)$；
  - 若经过一批次权值调整后使总误差↓，则本次调整有效，令$\eta = \theta\eta\,(\theta>1)$。

4. BP算法的改进

- 改进1：增加动量项
- 改进2：自适应调节学习率
- 改进3： 引入陡度因子

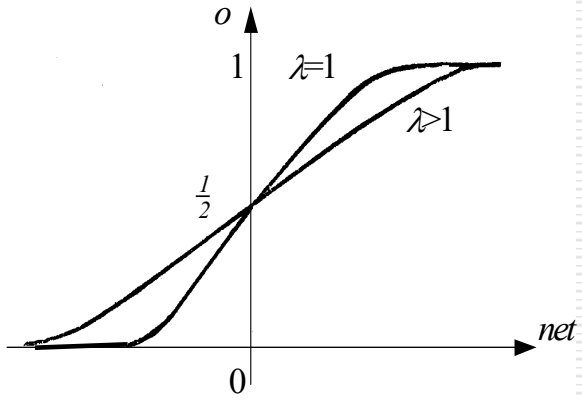
4. BP算法的改进：增加陡度因子

- 提出的原因：
  - 误差曲面上存在着平坦区域。
  - 权值调整进入平坦区的原因是进入了转移函数的饱和区。
- 基本思想：
  - 如果在进入平坦区后，设法压缩神经元的净输入，使其输出退出转移函数的不饱和区，就可以改变误差函数的形状，从而使调整脱离平坦区。

4. BP算法的改进：增加陡度因子

- 基本方法：
  - 在原激活函数中引入一个陡度因子λ

$$o = \frac{1}{1 + e^{-net/\lambda}}$$

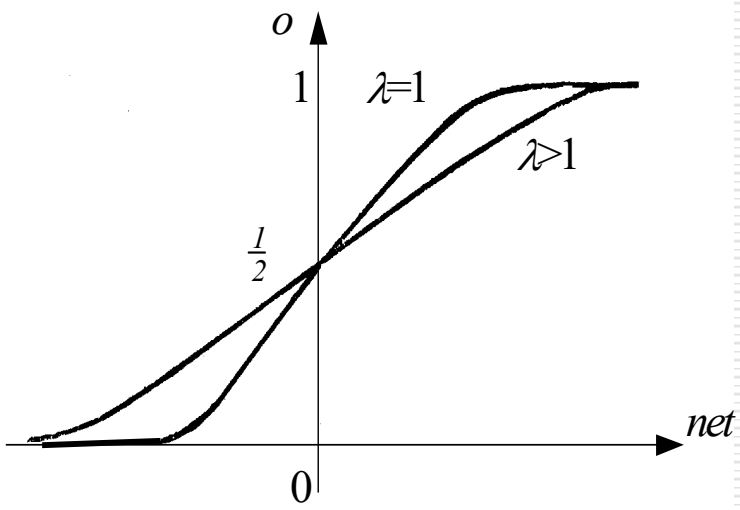


  - 当发现 ΔE 接近零而误差仍较大时，可判断已进入平坦区，此时令 $\lambda > 1$；
  - 当退出平坦区后，再令 $\lambda = 1$。

4. BP算法的改进：增加陡度因子

作用分析：

- $\lambda>1$：神经元的激活函数曲线的敏感区段变长，从而退出饱和值。
- $\lambda=1$：神经元的激活函数恢复原状，对净输入*net*较小的情况有较高的灵敏度。
- 应用结果表明该方法对于提高BP算法的收敛速度十分有效。

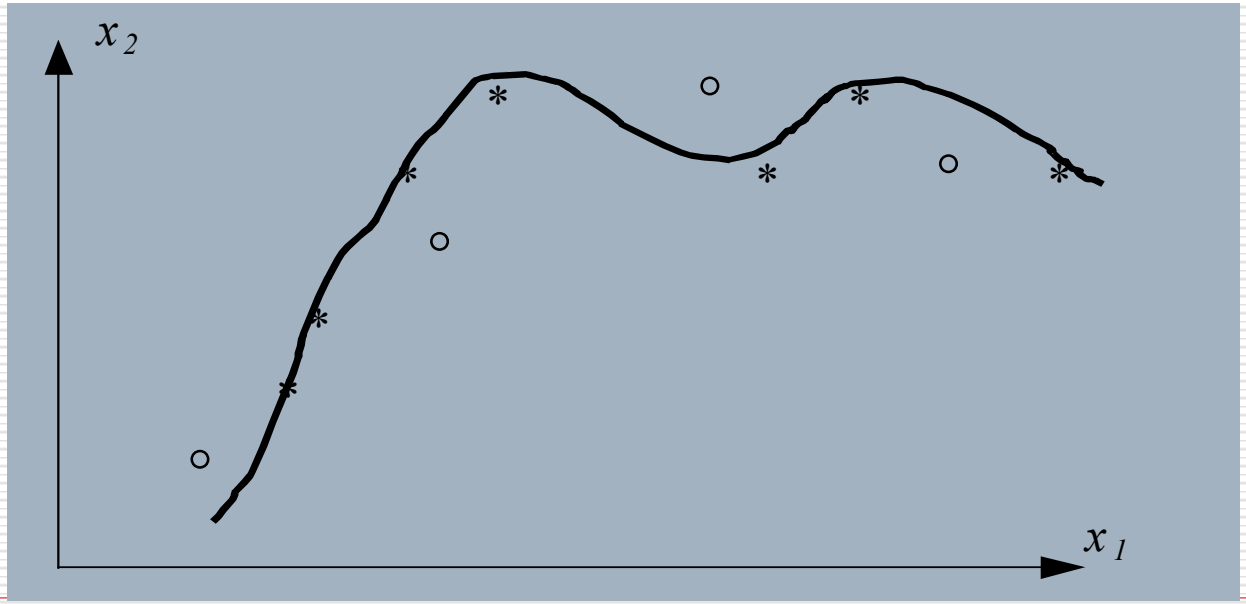


课程目录

- 1. 上次课程回顾
- 2. BP算法介绍
- 3. BP算法的局限性
- 4. BP算法的改进
- **5. 训练与测试**

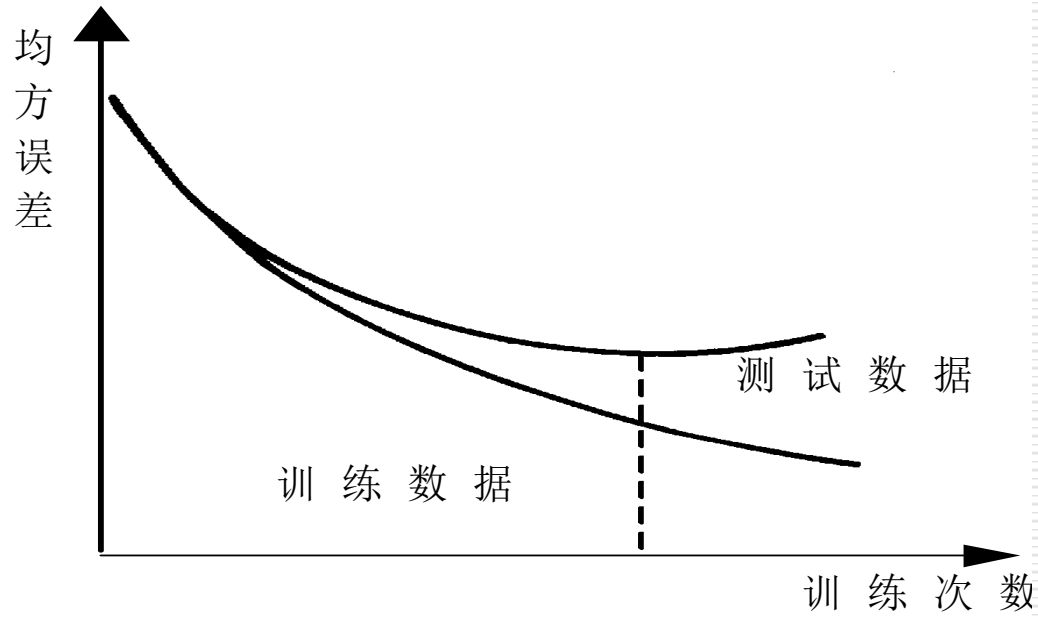
5. 训练与测试

- 神经网络的性能好坏主要看其是否具有很好的泛化能力。
- 泛化能力是指在向网络输入训练时未曾见过的非样本数据，网络也能完成由输入空间向输出空间的正确映射。

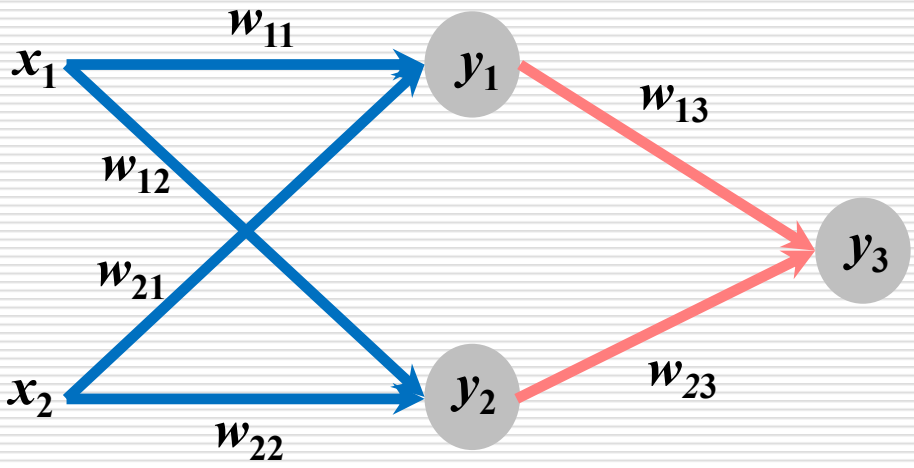


5. 训练与测试

- 在隐节点数一定的情况下，为获得好的泛化能力，存在着一个最佳训练次数。

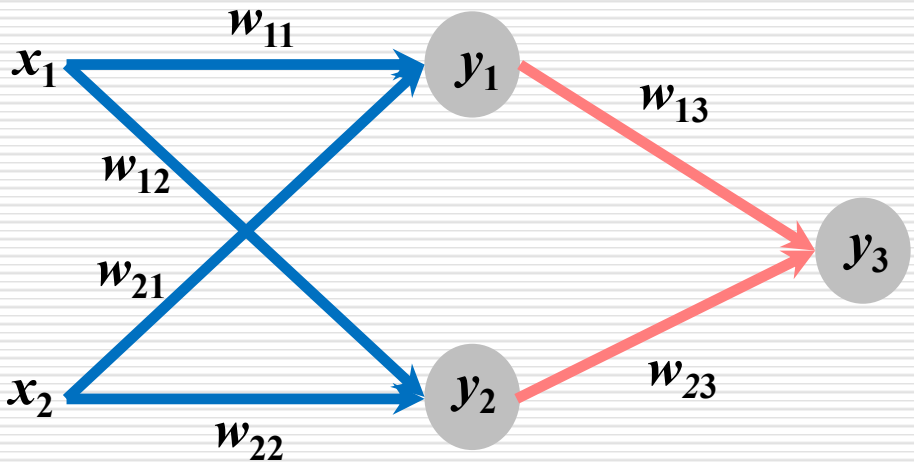


6. 习题



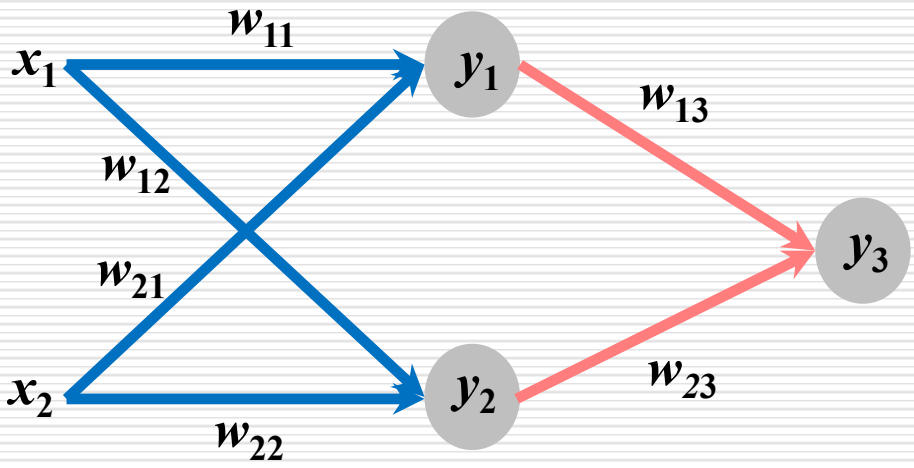
- 学习率$\eta=1$，当前训练样本为 $x_1=1$， $x_2=0$。期望输出结果为1。
- 阈值均为0，激活函数是：$y=\begin{cases} x & x \geq 1 \\ 1 & x<1 \end{cases}$
- 第 k 次学习的权重是：$w_{11}(k)=0$, $w_{12}(k)=2$, $w_{21}(k)=2$, $w_{22}(k)=1$, $w_{13}(k)=1$, $w_{23}(k)=1$。
- 求第 k 次学习的输出 $y_3(k)$ 和第 $k+1$ 次学习的输出 $y_3(k+1)$ 。

6. 习题



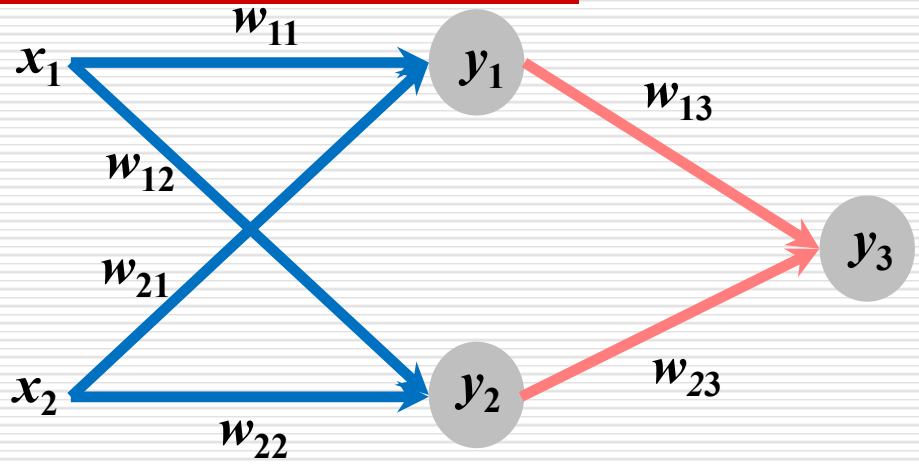
- 第 k 次学习的输出 $y_1(k)=1$，$y_2(k)=2$，$y_3(k)=3$。
- 第 $k+1$ 次学习的权重是：$w_{11}(k)=0$，$w_{12}(k)=0$，$w_{21}(k)=2$，$w_{22}(k)=1$，$w_{13}(k)=-1$，$w_{23}(k)=-3$
- 第 $k+1$ 次学习的输出 $y_1(k)=1$，$y_2(k)=1$，$y_3(k)=1$。

6. 习题



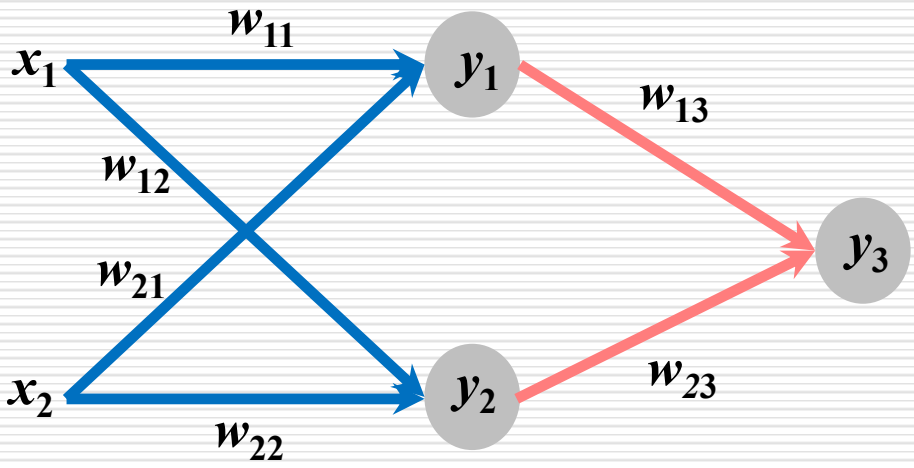
神经元 j	总输入 S_j	输出 y_j
1	1*0 + 0*2 = 0	$S_1 < 1$, $y_1 = 1$
2	1*2 + 0*1 = 2	$S_2 > 1$, $y_2 = 2$
3	1*1 + 2*1 = 3	$S_3 > 1$, $y_3 = 3$

6. 习题



$\triangle w_{ij}$	$w_{ij}(k+1)$
$\triangle w_{13} = -(3-1)*1 = -2$	$w_{13}(k+1) = 1-2 = -1$
$\triangle w_{23} = -(3-1)*2 = -4$	$w_{23}(k+1) = 1-4 = -3$
$\triangle w_{11} = -(3-1)*1*0 = 0$	$w_{11}(k+1) = 0-0 = 0$
$\triangle w_{21} = -(3-1)*1*0 = 0$	$w_{21}(k+1) = 2-0 = 2$
$\triangle w_{12} = -(3-1)*1*1 = -2$	$w_{12}(k+1) = 2-2 = 0$
$\triangle w_{22} = -(3-1)*1*0 = 0$	$w_{22}(k+1) = 1-0 = 1$

6. 习题



神经元 j	总输入 S_j	输出 y_j
1	1*0 + 0*2 = 0	$S_1 < 1$, $y_1 = 1$
2	1*0 + 0*1 = 1	$S_2 >= 1$, $y_2 = 1$
3	1*(-1) + 1*(-3) = -4	$S_3 < 1$, $y_3 = 1$