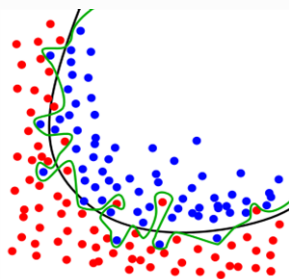


媒体与认知

Media and Cognition



V 1.0 2022.3.3

清华大学电子工程系

王生进 彭良瑞 李亚利

Email: {wgsgj, penglr, liyali13}@tsinghua.edu.cn

第 2 讲 机器学习基础

一、引言

二、线性回归

三、逻辑回归与分类

四、模型评估

问题

- 机器学习的回归和分类任务有何区别与联系？
- 回归和分类问题的目标函数分别如何定义？

一、引言

➤ 机器学习

- ◆ 根据训练样本对系统输入输出之间依赖关系进行建模，以便在测试阶段对输入做出预测

➤ 基本任务

- ◆ 监督学习(supervised learning)：训练样本有真值标签

- 回归

- 例如：文本定位，拟合外接框坐标

- 分类

- 例如：文字与非文字区域判别

- ◆ 非监督学习(unsupervised learning)：训练样本无标签

- 聚类

ICDAR 2013 Focused Scene Text Challenge

(<https://rrc.cvc.uab.es/?ch=2>)

◆ 文本定位 Text Localization 或 文本检测

- Obtain a **rough estimation of the text areas in the image**
- Output bounding boxes of words or text lines

◆ 文本图像分割 Text Segmentation

- **pixel level separation of text from the background**

◆ 单词识别 Word Recognition,

- Given the bounding boxes of words Output the corresponding **text transcriptions**

◆ 端到端 End-to-End 文本检测与识别

- **Localize and recognize all words in a single step**



样本图像示例



作业1 中的字符分类 (区分字符与背景) 任务样本示例

字符图像分割真值



30	114	234	167	354	152	334	192	375	"K"
113	141	130	213	354	198	335	233	375	"E"
184	140	168	256	355	241	335	276	376	"E"
78	217	94	299	352	284	335	321	376	"P"

字符K图像分割真值中的RGB 数值

字符K中心的坐标 字符K外接框

样本真值标注示例

机器学习的三要素

➤ 模型定义

- ◆ 给定训练集数据, $D = \{x^{(i)}, y^{(i)}\}, i \in \{1, 2, \dots, N\}$,
- ◆ 模型包括函数映射形式 f 及参数 θ

$$y = f(x, \theta)$$

➤ 目标函数

- ◆ 损失函数或代价函数

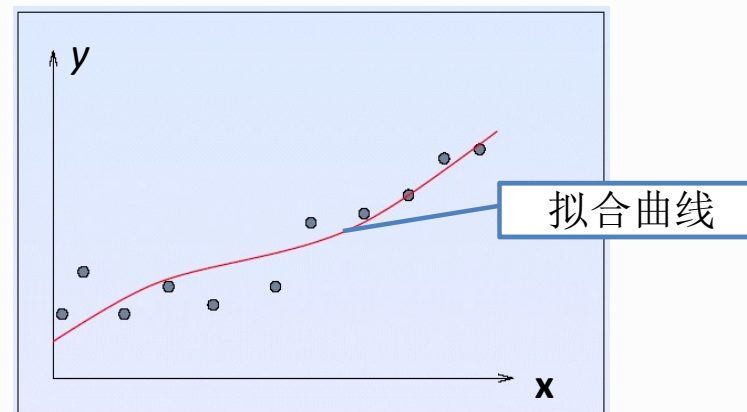
$$L(f(x, \theta), y)$$

➤ 求解方法

- ◆ 优化问题求解 $\theta^* = \arg \min_{\theta} L(f(x, \theta), y)$

回归与分类

► 回归问题：输入标量或向量 \mathbf{x} ，输出随 \mathbf{x} 变化的连续变量 y



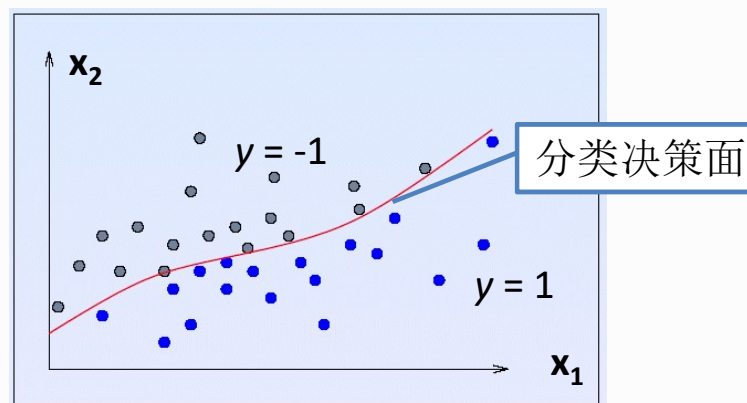
► 分类问题：输入特征向量 \mathbf{x} ，输出类别标签 y

◆ 两类情况下

- $y \in \{-1, 1\}$ 或 $y \in \{0, 1\}$

◆ 多类情况下

- 类别数为 C , $y \in \{1, 2, \dots, C\}$



模式识别(Pattern Recognition)基本概念

➤ 模式(pattern)

- ◆ 模式是人们根据需要及特定的环境条件，对自然事物形成的抽象分类概念
- ◆ 模式集合记为：

$$\Omega = \{\omega_i\}, \quad i = 1, 2, \dots, C$$

➤ 样本(sample, object)

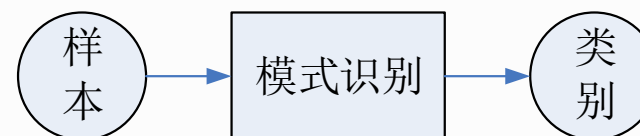
- ◆ 样本是自然界的具體事物，具有一定的类别特性，是抽象模式的具体体现
- ◆ 某一样本观测数据记为：

$$\vec{x} = (x_1, x_2, \dots, x_m)^T$$

➤ 模式分类/模式识别：

- ◆ 寻求样本观测量与类别属性的联系：

$$y = g(\vec{x}) = \omega_i$$



特征提取与分类器

➤ 特征提取

- ◆ 特征是某一事物表现出来的特点与表征，是区别于其他事物的关键
- ◆ 特征提取：对输入的样本观测数据进行处理或变换，得到有利于分类的特征向量

$$z = (z_1, z_2, \dots, z_d)^T = h(x)$$

➤ 分类器

- ◆ 对输入特征向量，利用分类器模型参数，给出类别判决结果

生成式模型 与 鉴别式模型

➤ 生成式模型 Generative models:

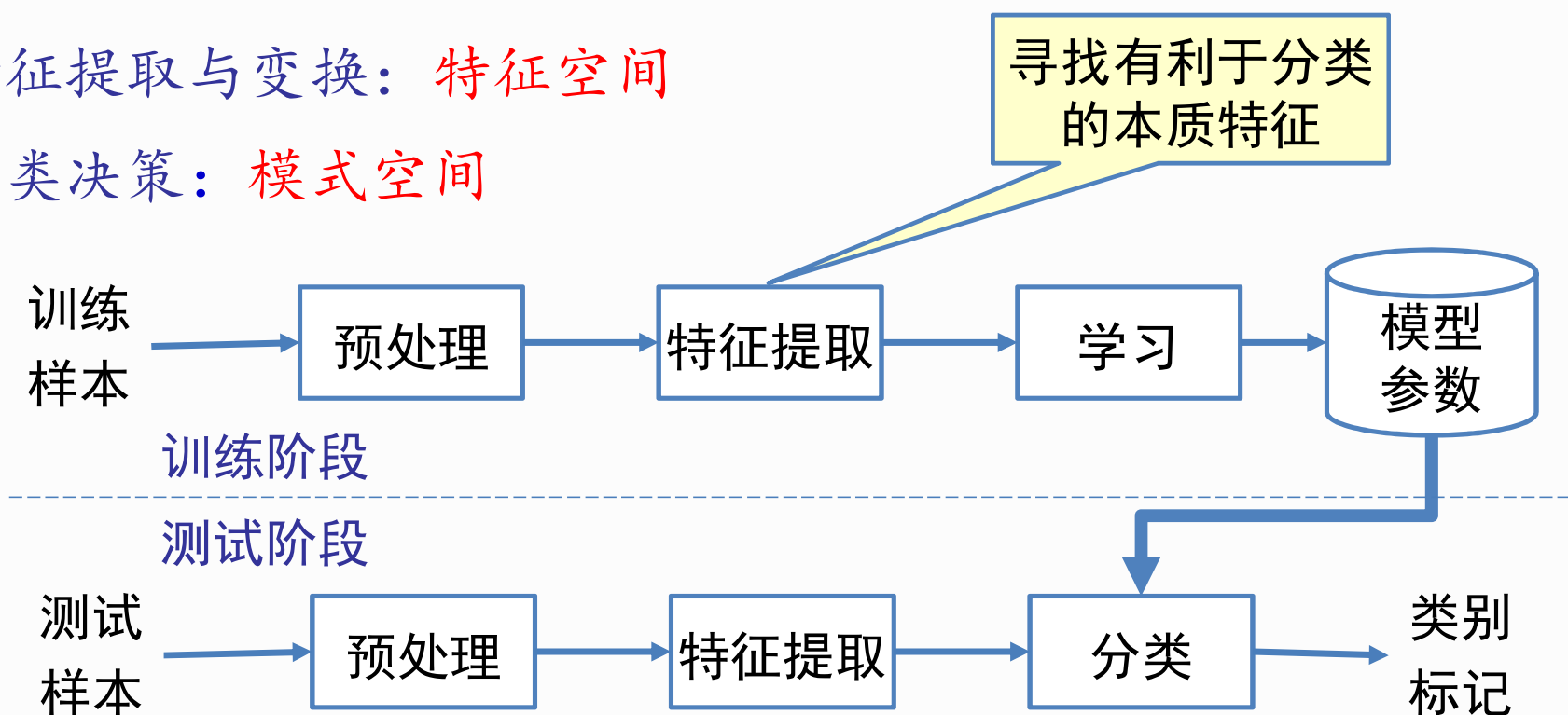
- ◆ 对特征向量和类别的联合概率分布建模
- ◆ 典型方法：
 - 最小距离分类器：计算样本到各类别中心的距离，选择距离最小的类别作为输出

➤ 鉴别式模型 Discriminative models:

- ◆ 直接用函数对分类决策面进行建模
- ◆ 典型方法：如神经网络

模式识别系统

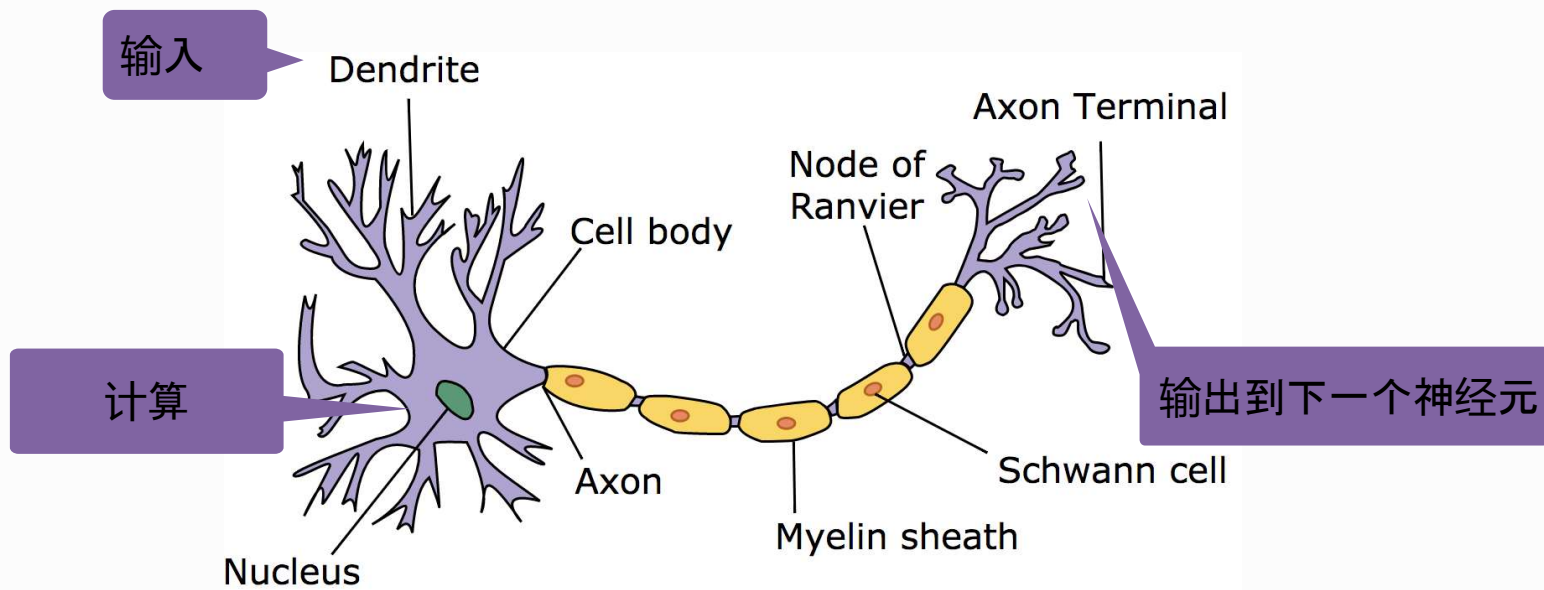
- 数据获取：信号空间
- 预处理
- 特征提取与变换：特征空间
- 分类决策：模式空间



神经生理学的启发

- 人脑是由大量(100亿个以上)的神经元(Neuron)和神经胶质细胞(1000亿个以上)等组成
 - ◆ 神经元的功能是接受、处理和传送信息
 - ◆ 神经元之间通过突触建立联系，构成复杂的神经回路（nerve circuit）
 - 神经回路是脑内信息处理的基本单位
- 神经元内的神经冲动主要通过电传导
- 神经元之间的传导主要通过化学传导

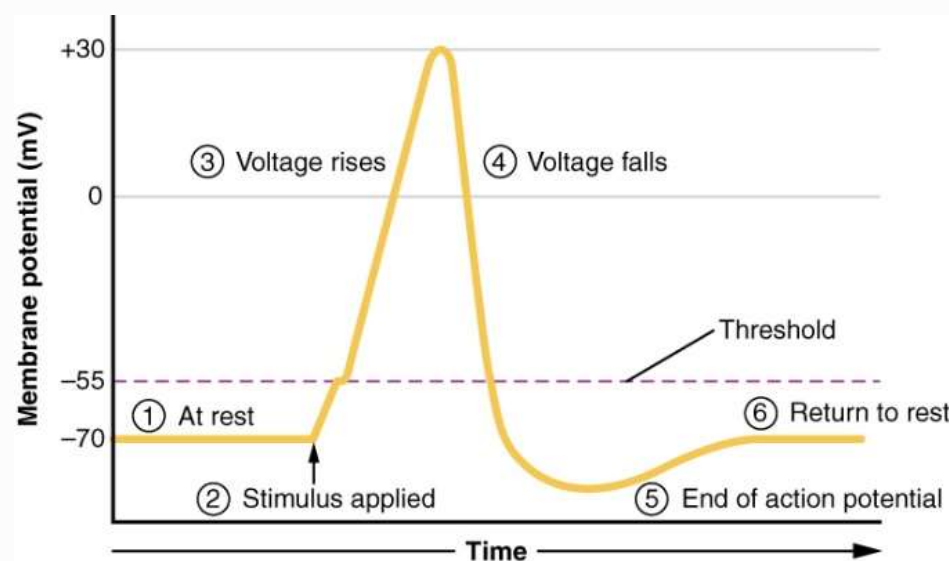
神经元的结构



神经冲动电传导的法则

➤ 全或无的法则:

- ◆ “全或无”法则是指每个神经元都有一个刺激阈值，对阈值以下的刺激不发生反应；对阈值以上的刺激，不论其强弱均给出同样高度（幅值）的神经脉冲发放

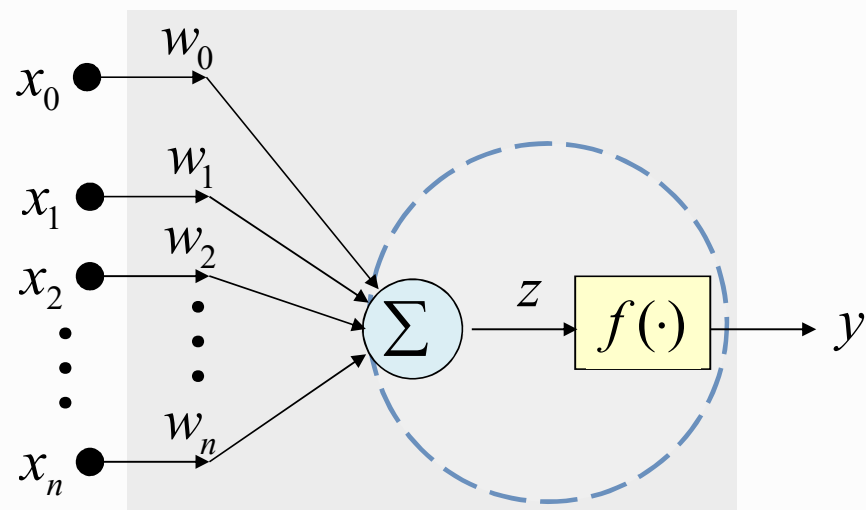


人工神经元

$$z = \sum_{i=0}^n w_i x_i$$

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

$$y = f(z)$$

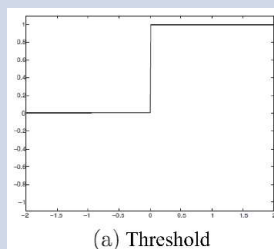


激活函数示例

输出范围为[0,1]

(a) 阈值函数:

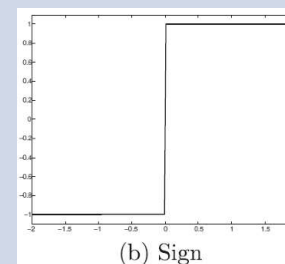
$$f(z) = \begin{cases} 1 & z > 0 \\ 0 & z \leq 0 \end{cases}$$



输出范围为[-1,1]

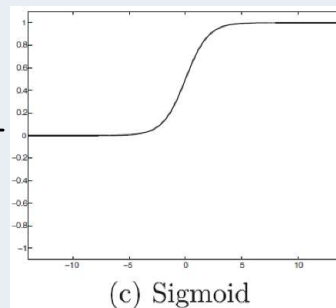
(b) 符号函数 sign:

$$f(z) = \begin{cases} 1 & z > 0 \\ -1 & z \leq 0 \end{cases}$$



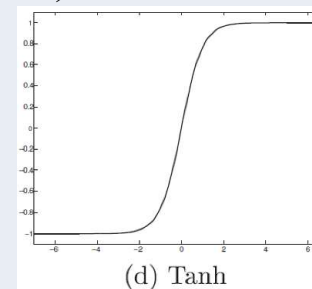
(c) sigmoid函数:

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



(d) 双曲正切函数 (Tanh):

$$f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$



线性回归、感知机、逻辑回归

	线性回归	感知机	逻辑回归
机器学习任务	回归	分类	分类
模型	$f(x) = w^T x + b$ $\mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}$	$f(x) = \text{sign}(w^T x + \mathbf{b})$ $\mathbf{W} \in \mathbb{R}^{k \times n}, \mathbf{b} \in \mathbb{R}^k$	$f(x) = \text{sigmoid}(w^T x + \mathbf{b})$ $\mathbf{W} \in \mathbb{R}^{k \times n}, \mathbf{b} \in \mathbb{R}^k$
目标函数	均方误差	样本到分类界面的距离	交叉熵

此题未设置答案，请点击右侧设置按钮

Sigmoid函数 $f(x) = \frac{1}{1+e^{-x}}$ 的导数是：

A $\frac{e^{-x}}{(1+e^{-x})^2}$

B $\frac{-e^{-x}}{(1+e^{-x})^2}$

提交

理解分类器：以数字识别为例

0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1	0	0

数字8的图像

1	1	1	1	1	1	1	1	1	1	1	1	1	-1	-1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1
1	1	1	1	1	1	1	1	1	1	1	1	1	-1	-1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1
1	1	1	1	1	1	1	1	1	1	1	1	1	-1	-1

数字3的模式

模式匹配

0	1	1	1	0
1	0	0	0	1
0	1	1	1	0
1	0	0	0	1
0	1	1	1	0

•

1	1	1	1	-1
-1	-1	-1	-1	1
1	1	1	1	-1
-1	-1	-1	-1	1
1	1	1	1	-1

=

$$\begin{aligned} & 0 \times 1 + 1 \times 1 + 1 \times 1 + 1 \times 1 + 0 \times (-1) + \\ & 1 \times (-1) + 0 \times (-1) + 0 \times (-1) + 0 \times (-1) + 1 \times 1 + \\ & 0 \times 1 + 1 \times 1 + 1 \times 1 + 1 \times 1 + 0 \times (-1) + \\ & 1 \times (-1) + 0 \times (-1) + 0 \times (-1) + 0 \times (-1) + 1 \times 1 + \\ & 0 \times 1 + 1 \times 1 + 1 \times 1 + 1 \times 1 + 0 \times (-1) \\ & = 9 \end{aligned}$$

$$net = w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n$$

其中： $w_i (i = 1, 2, \dots, n)$: 表示数字模式
 $x_i (i = 1, 2, \dots, n)$: 表示数字图像

数字3、数字8与模式3的匹配

[illegible]

●

[illegible]
$$= 143$$
[illegible][illegible]

= 115

存在的问题与对策

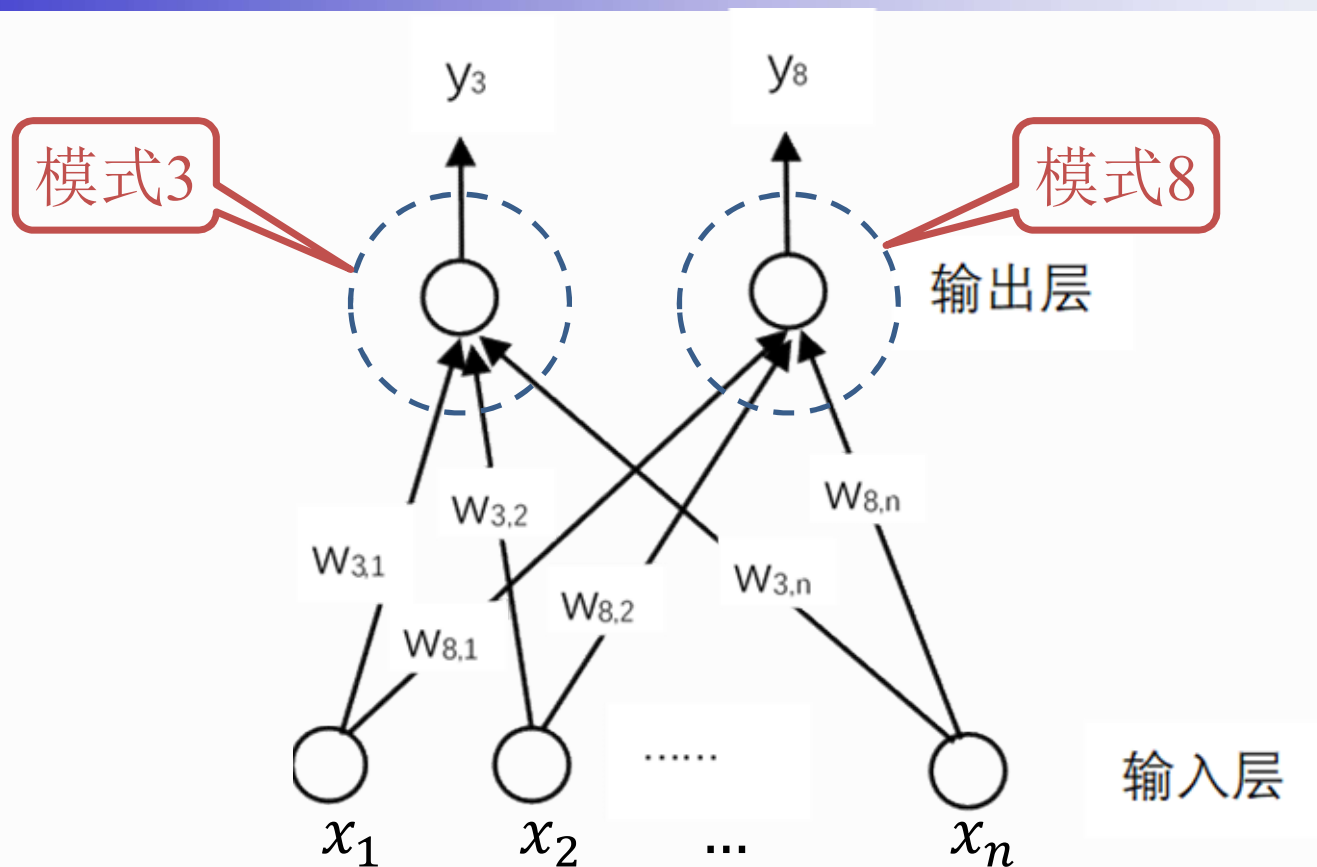
➤ 笔画多少带来的问题

- ◆ 1的笔画少
- ◆ 8的笔画多

➤ 如何评判匹配的程度？

- ◆ 引入偏置量
- ◆ 引入激活函数

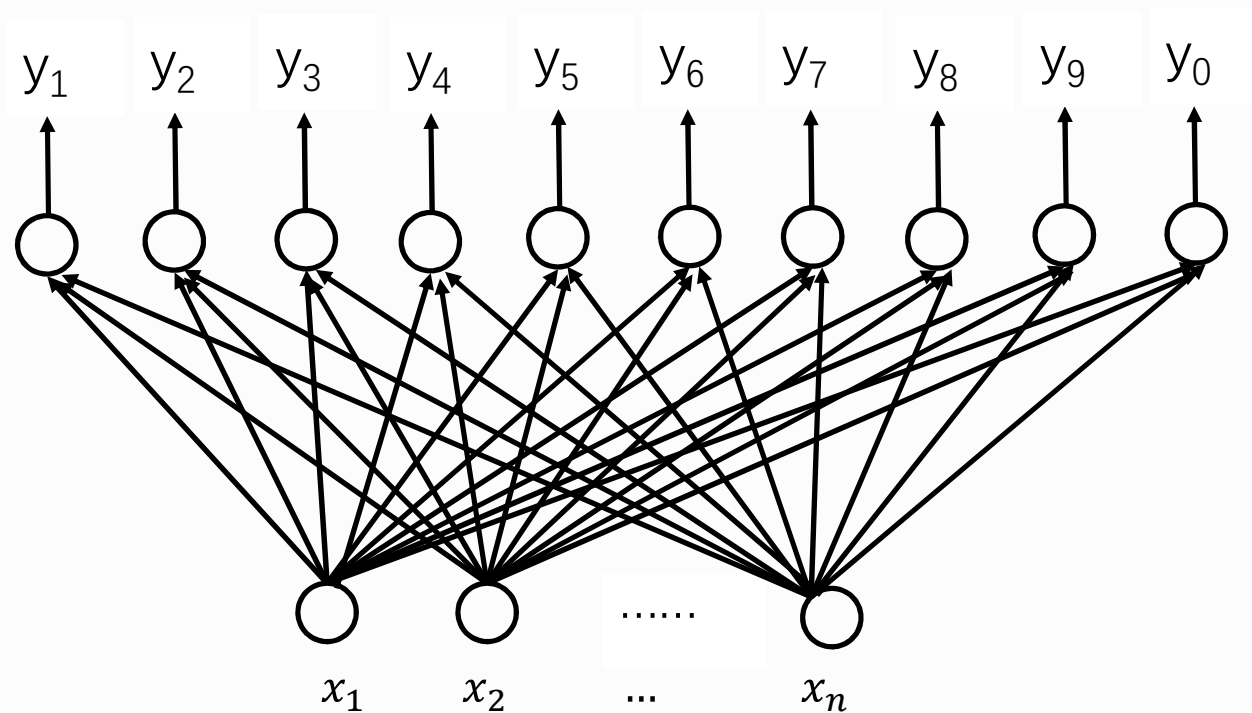
简单的神经网络



$$y_3 = \text{sigmoid}(w_{3,1} \cdot x_1 + w_{3,2} \cdot x_2 + \dots + w_{3,n} \cdot x_n + b_3)$$

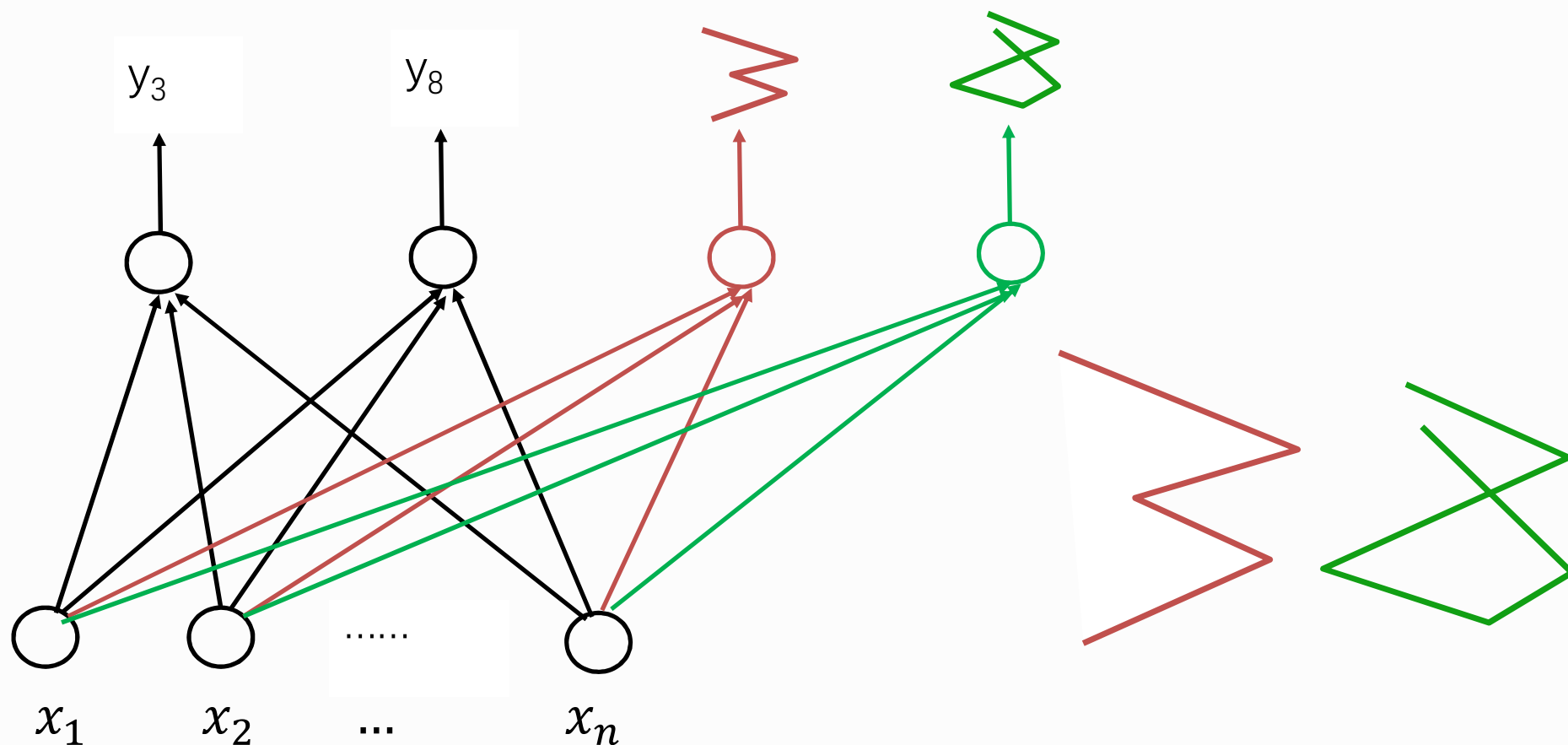
$$y_8 = \text{sigmoid}(w_{8,1} \cdot x_1 + w_{8,2} \cdot x_2 + \dots + w_{8,n} \cdot x_n + b_8)$$

数字识别神经网络

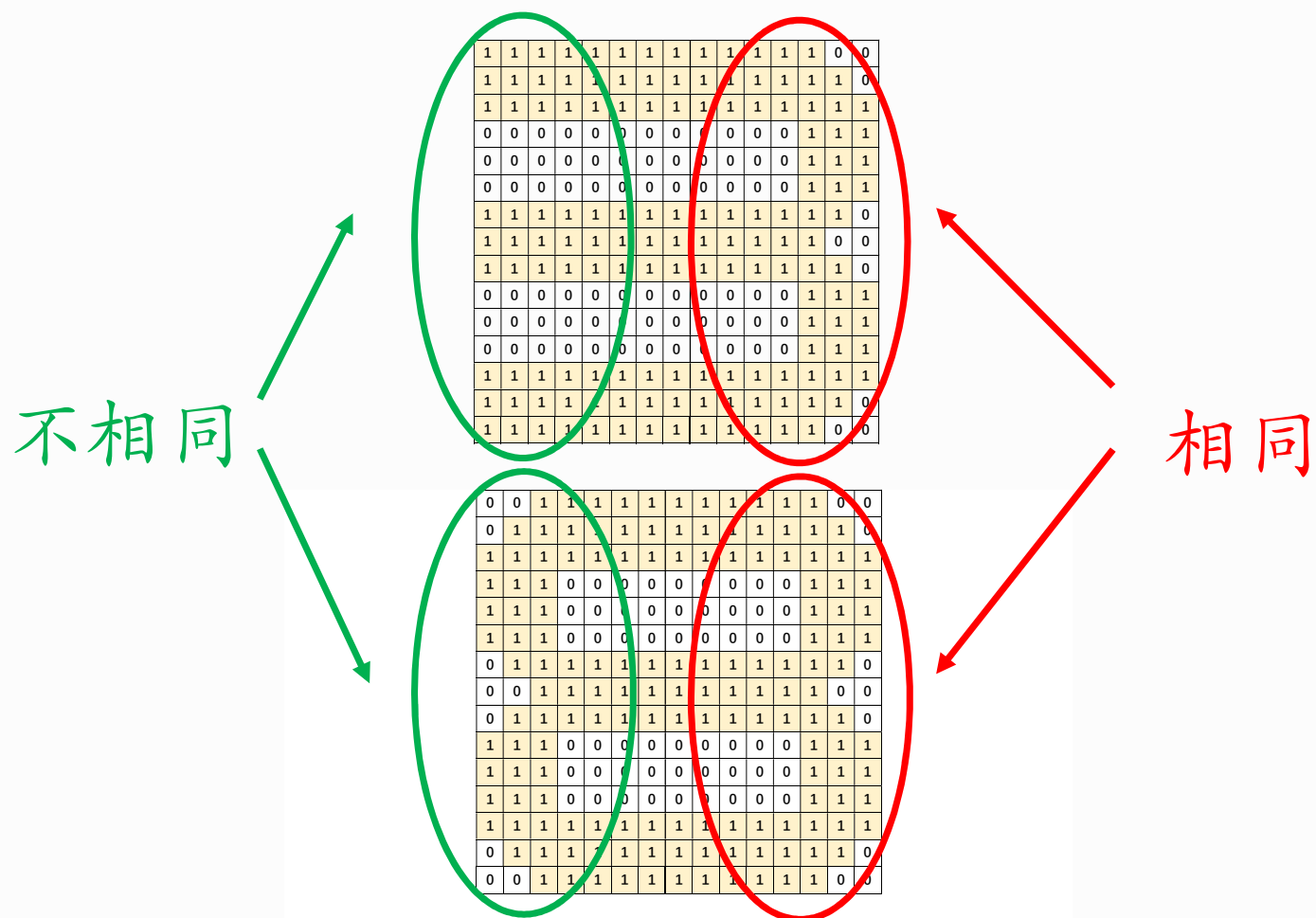


$$y_i = \text{sigmoid}(w_{i,1} \cdot x_1 + w_{i,2} \cdot x_2 + \dots + w_{i,n} \cdot x_n + b_i)$$

神经网络的横向扩展——增加模式



神经网络的纵向扩展——局部模式



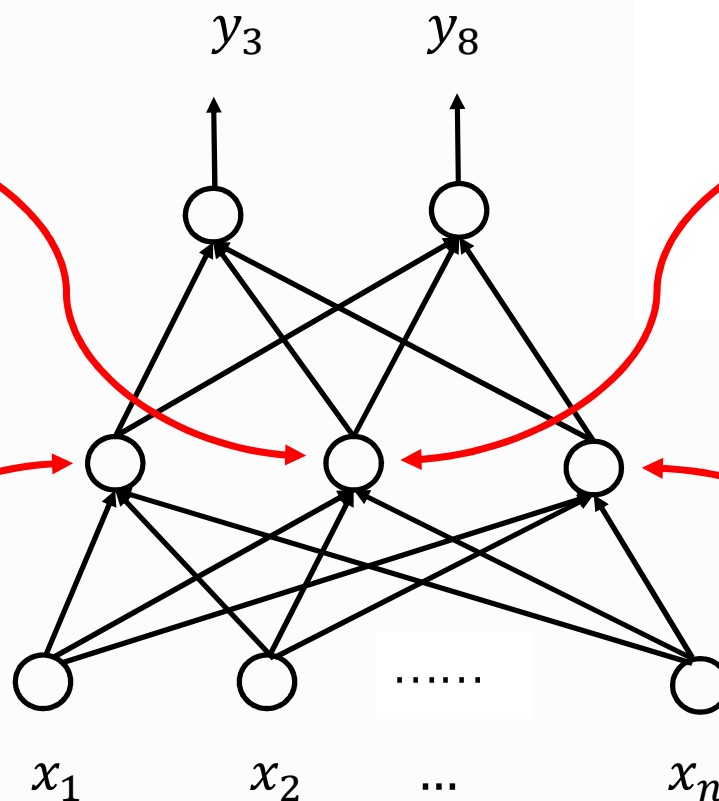
让神经网络更深——模式组合

3
的
右
部

1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0

3
的
左
部

1	1	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	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	1	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	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1



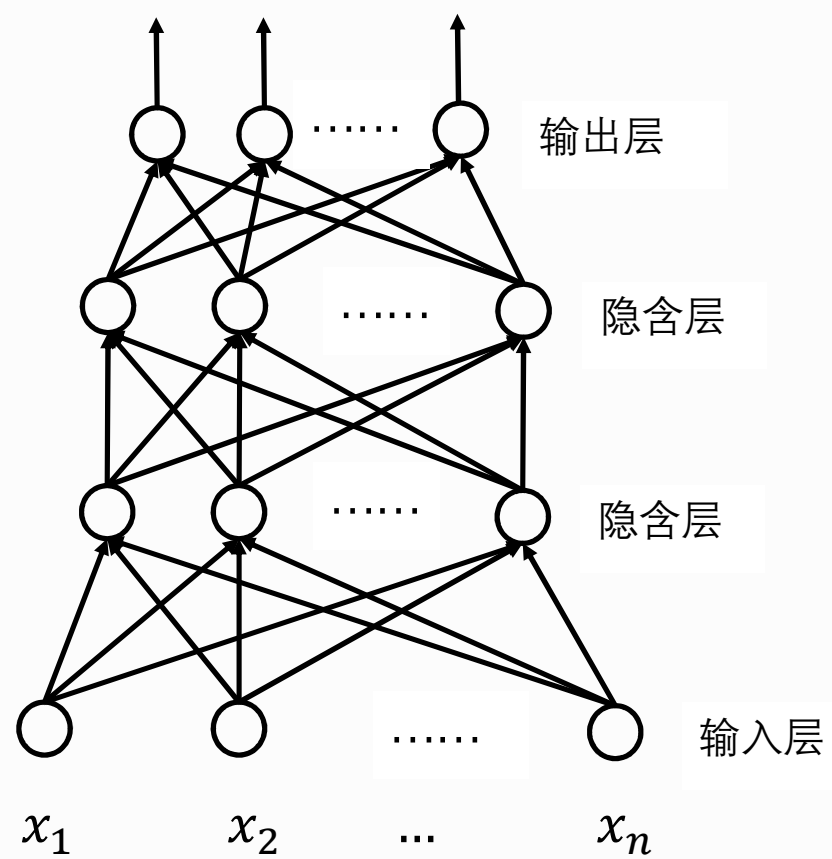
8
的
右
部

1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0

8
的
左
部

0	0	1	1	1	1	1	1
0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	1	1	1	1	1	1	1
1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1

多层神经网络



二、线性回归方法

➤ 输入: $\mathbf{x} = [x_1, x_2, \dots, x_n]^T$

➤ 模型有 n 个权重系数和1个偏置量:

$$\mathbf{w} = [w_1, w_2, \dots, w_n]^T, b$$

➤ 输出:

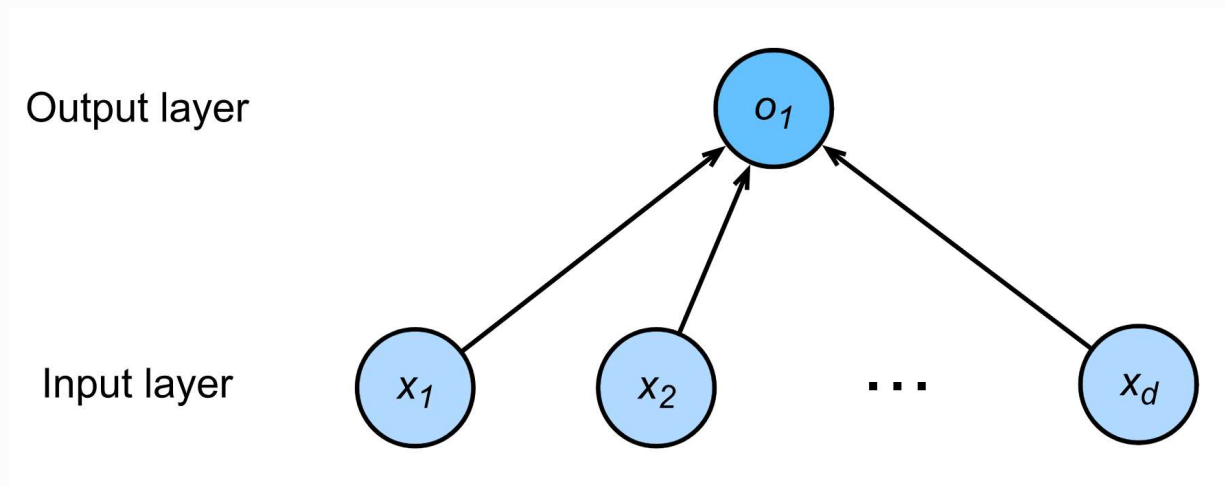
$$y = w_1x_1 + w_2x_2 + \dots + w_nx_n + b$$

向量化形式:

$$y = \mathbf{w}^T \mathbf{x} + b$$

线性回归方法

➤ 线性回归方法是一个单层单节点的网络



数据集

➤ 数据集的采集与加工

- ◆ 输入数据 x 与输出数据 y

$$D = \{x^{(i)}, y^{(i)}\}, i \in \{1, 2, \dots, N\}$$

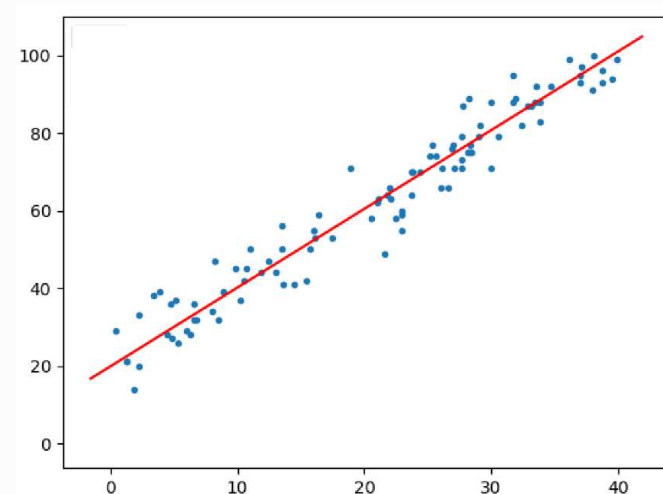
- ◆ 单变量线性回归

- 如 学习成绩与投入时间的关系，输入为标量

- ◆ 多变量线性回归

- 如 房价与工资、贷款利率等的关系，输入为向量

- ◆ 数据集一般可划分为训练集、验证集、测试集



模型求解

➤ 定义目标函数

- ◆ **最小二乘法**: 找到一条直线, 使所有样本点到直线上预测点的均方误差(mean-square error, MSE)最小

$$L(\mathbf{X}, \mathbf{y}, \mathbf{w}, b) = \frac{1}{N} \sum_{i=1}^N (y^{(i)} - \mathbf{w}^T \mathbf{x}^{(i)} - b)^2$$

➤ 求解

$$\mathbf{w}^*, \mathbf{b}^* = \arg \min_{\mathbf{w}, b} L(\mathbf{X}, \mathbf{y}, \mathbf{w}, b)$$

待求解的参数也可记为 $\theta = \{\mathbf{w}, b\}$

利用优化方法求解：误差反向传播

➤ 选择 w, b 的初值

➤ 训练迭代过程：

- ◆ 根据输入数据进行前向计算
- ◆ 利用误差反向传播，计算 $\partial L / \partial w, \partial L / \partial b$
- ◆ 梯度下降法调整模型参数

$$w \leftarrow w - \eta \partial L / \partial w$$

$$b \leftarrow b - \eta \partial L / \partial b$$

- 梯度：用于寻找更新权重的方向
- 学习率 η ：
 - 一个在训练模型前设定的超参数（不是待求解的模型参数）
 - 用于指定更新参数的强度

训练方法：梯度下降法调整参数



选取样本批量大小 Batch Size

➤ 批量梯度下降法 Batch Gradient Descent (BGD)

- ◆ 利用所有N个训练样本计算平均梯度

$$\theta \leftarrow \theta - \eta \left[\frac{1}{N} \sum_{i=1}^N \nabla_{\theta} L(\theta, x^{(i)}, y^{(i)}) \right]$$

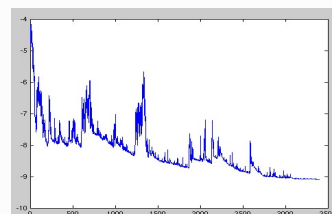
速度慢，数据量大时内存不足

➤ 随机梯度下降法 Stochastic Gradient Descent (SGD)

- ◆ 随机选取单个样本计算梯度

$$\theta \leftarrow \theta - \eta \nabla_{\theta} L(\theta, x^{(i)}, y^{(i)})$$

方差大，损失函数震荡严重



➤ 小批量随机梯度下降法

Mini-batch Stochastic Gradient Descent

- ◆ 每次迭代(iteration)选取小批量样本计算梯度, Batch Size = M
- ◆ N个样本训练一轮(Epoch)的迭代次数为 $(N+M-1)/M$

$$\theta \leftarrow \theta - \eta \left[\frac{1}{M} \sum_{i=1}^M \nabla_{\theta} L(\theta, x^{(i)}, y^{(i)}) \right]$$

对当前批次每个样本的损失取平均

随机梯度下降法(Stochastic Gradient Descent, SGD)

输入：训练集 $D = \{x^{(i)}, y^{(i)}\}, i \in \{1, 2, \dots, N\}$ ，验证集，学习率 η

算法：

初始化模型参数 θ ;

repeat

对训练集 D 中的样本随机重排序;

for $i = 1, \dots, N$ do

从训练集 D 选取样本 $(x^{(i)}, y^{(i)})$;

前向计算;

误差反向传播, 计算 $\frac{\partial L(\theta; x^{(i)}, y^{(i)})}{\partial \theta}$;

更新参数 $\theta \leftarrow \theta - \eta \frac{\partial L(\theta; x^{(i)}, y^{(i)})}{\partial \theta}$;

end

until 模型 $f(x, \theta)$ 在验证集上的误差不再下降;

输出: θ

一次迭代
(iteration)

一轮训练
(epoch)

线性回归方法小结

➤ 问题：拟合线性变化的数据

➤ 模型： $y = \mathbf{w}^T \mathbf{x} + b$

➤ 目标函数：均方误差

➤ 迭代求解方法

- ◆ 参数初始化

- ◆ 迭代

- 前向计算
- 误差反向传播，计算梯度
- 利用梯度下降法更新模型参数

讨论

► 试分析什么情况下不用考虑下式中的偏置量 b ?

$$y = \mathbf{w}^T \mathbf{x} + b$$

(可在雨课堂中发送弹幕回答)

三、逻辑回归与分类

➤ 神经网络的早期研究：感知机(Perceptron)

- ◆ 感知机由Rosenblatt在1957年提出，是神经网络的基础
- ◆ 输入样本的特征向量，输出为样本类别标签（或称类别真值），记为 1和-1
- ◆ 感知机对应于样本空间中的分类超平面，属于鉴别式模型
- ◆ 利用梯度下降法将损失函数极小化
 - 损失函数为误分类样本点到分类界面距离

二分类的线性分类模型：感知机

➤ 感知机

- ◆ 给定训练集数据 $D = \{\mathbf{x}^{(i)}, y^{(i)}\}, i \in \{1, 2, \dots, N\}$ ，样本的特征向量 $\mathbf{x} \subseteq R^d$ ，样本类别标号 $y \in \{1, -1\}$ ，感知机是将特征向量映射为类别标签的函数：

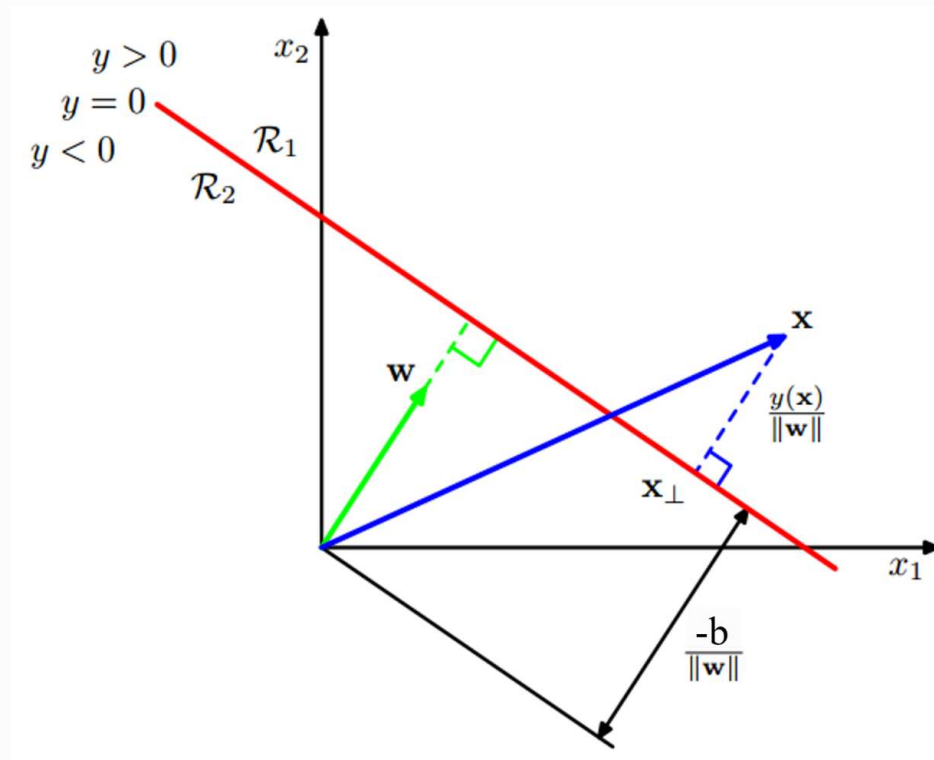
$$y = \text{sign}(\mathbf{w}^T \mathbf{x} + b)$$

- ◆ 模型参数：权值向量 \mathbf{w} ，偏置量 b
- ◆ 激活函数采用符号函数：

$$\text{sign}(z) = \begin{cases} 1, & z > 0 \\ -1, & z \leq 0 \end{cases}$$

感知机几何解释

- 分类决策面对应于线性方程 $\mathbf{w}^T \mathbf{x} + b = 0$
- \mathbf{w} 为法向量， b 为偏置量



感知机学习策略

► 如何定义损失函数？

◆ 如果采用误分类点的数目，存在的问题是不可对参数求导

◆ 可计算所有误分类点到分类决策面的等效距离之和

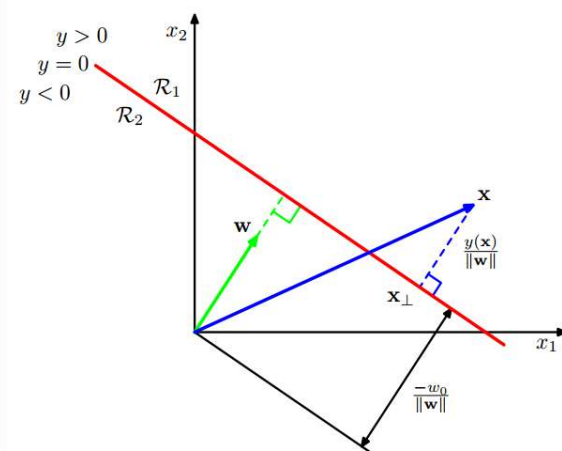
误分类点：当 $y = 1$ 时， $\hat{y} = w^T x + b < 0$

当 $y = -1$ 时， $\hat{y} = w^T x + b > 0$

误分类点到分类决策面的等效距离： $-y(w^T x + b)$

所有误分类点到分类决策面的等效距离之和：

$$\sum_{x \in D} \max(0, -y(w^T x + b))$$



感知机学习算法

➤ 目标函数: $L(w, b) = \sum_{x \in D} \max(0, -y(w^T x + b))$

➤ 优化求解问题

$$\min_{w, b} L(w, b)$$

➤ 对 w , b 初始化, 采用随机梯度下降法进行求解

◆ 若 x 为误分类点, $y(w^T x + b) < 0$:

计算梯度: $\frac{\partial L(w, b)}{\partial w} = -y x$ $\frac{\partial L(w, b)}{\partial b} = -y$

更新参数:

$$w \leftarrow w + \eta y x \quad b \leftarrow b + \eta y$$

线性回归与逻辑回归

➤ 线性回归

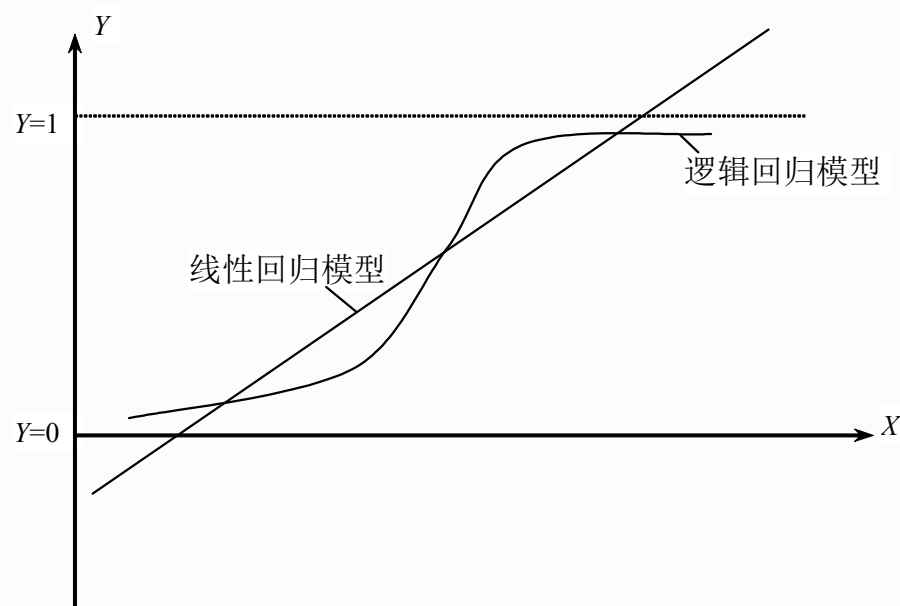
$$y = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

➤ 逻辑回归

◆ 分两步计算

- 线性加权 $z = \mathbf{w}^T \mathbf{x} + b$
- Sigmoid激活函数

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



逻辑回归的目标函数

➤ 交叉熵通常用于度量概率分布的差异

$$H(P, Q) = - \int_{-\infty}^{\infty} p(x) \log q(x) dx$$

◆ 类别真值看作一种特殊的概率分布

$$p = \begin{cases} 1, & \text{样本类别真值为 1} \\ 0, & \text{样本类别真值为 0} \end{cases}$$

◆ 模型输出分类结果及对应的概率估计

- 属于正类的概率为 q
- 属于负类的概率为 $1 - q$

◆ 二分类交叉熵(Binary Cross Entropy loss)目标函数


$$\text{loss}(x, \theta) = -p \log q - (1 - p) \log(1 - q)$$

PyTorch 中二分类交叉熵目标函数示例

```
>>> import torch.nn.functional as F
```

```
>>> m = nn.Sigmoid()
```

```
>>> loss = nn.BCELoss()
```

 二分类交叉熵 Binary Cross Entropy Loss

```
>>> input = torch.randn(3, requires_grad=True)
```

```
>>> target = torch.empty(3).random_(2)
```

```
>>> output = loss(m(input), target)
```

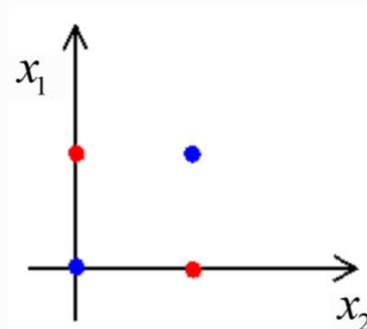
```
>>> output.backward()
```

异或问题

► 异或（exclusive OR, XOR）问题

- ◆ 二维平面中不存在将所有样本分为两类的一条直线，即为线性不可分情形
- ◆ 单个人工神经元（感知机、逻辑回归模型）为线性分类模型，不能解决异或问题

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



思考：如何对人工神经元进行拓展，使之可以解决异或问题？

四、模型评估

➤ 模型的容量(capacity)

- ◆ 模型复杂度：模型参数量大小等

➤ 误差(error):

- ◆ 模型的实际输出与样本的对应真值之间的差异
- ◆ 训练误差/经验误差(empirical error)
- ◆ 测试误差/泛化误差(generalization error)

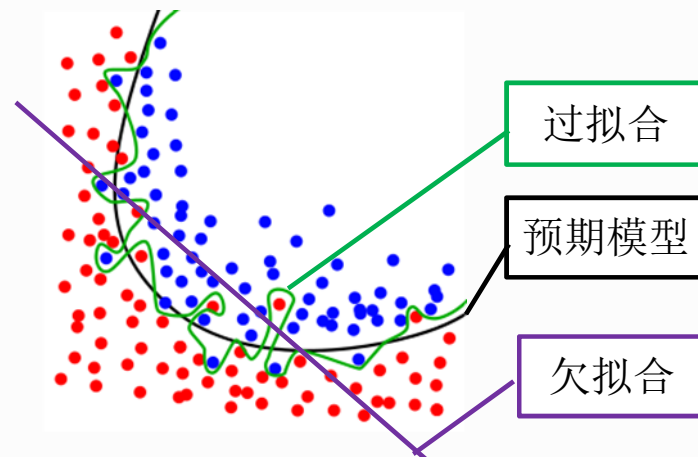
➤ 模型的选择

- ◆ 拟合能力强的模型一般复杂度会比较高，容易过拟合
- ◆ 如果限制模型复杂度，降低拟合能力，可能会欠拟合

模型的过拟合与欠拟合

➤ 理想模型

- ◆ 模型通过训练集学到了适用于所有潜在样本的普遍规律



➤ 过拟合(overfitting)

- ◆ 模型在训练集上过度学习，把训练样本部分自身特点当作所有潜在样本分类的普遍规律，缺乏对问题本质的理解，表现为训练误差小，但测试误差大，出现模型泛化性能下降

➤ 欠拟合(underfitting)

- ◆ 模型没有得到训练集样本上的分类规律，表现为训练误差大

评价指标

➤ 错误率(error rate)与识别率(recognition rate或recognition accuracy)

◆ 错误率：分类错误的样本占样本总数的比例

• 假设 n 个样本中有 e 个样本分类错误，错误率为 e/n

◆ 识别率：

$$(n - e)/n = 1 - e/n$$

评价指标 (续)

➤ 两类模式分类的混淆矩阵

真实类别	预测为正类的数目	预测为负类的数目
正类	TP (True Positive, 真正类)	FN (False Negative, 假负类)
负类	FP (False Positive, 假正类)	TN (True Negative, 真负类)

➤ 召回率 Recall、准确率 Precision、 F_1 分数

$$\text{Recall} = \frac{TP}{TP + FN} \quad \text{Precision} = \frac{TP}{TP + FP} \quad F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

➤ 真阳性率(True Positive Rate)、假阳性率(False Positive Rate)

$$\text{TPR} = \frac{TP}{TP + FN} \quad \text{FPR} = \frac{FP}{TN + FP}$$

例题：计算Recall 和Precision

➤ 混淆矩阵示例

真实类别	预测为正类的数目	预测为负类的数目
正类	7 (TP)	3 (FN)
负类	1 (FP)	9 (TN)

➤ 召回率 Recall、准确率 Precision的计算

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{7}{7 + 3} = 70\%$$

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{7}{7 + 1} = 87.5\%$$

交叉验证方法

➤ 交叉验证方法(Cross Validation)

- ◆ CV是用来验证分类器的性能一种统计分析方法，将原始数据(dataset)进行分组,一部分做为训练集(training set),另一部分做为验证集(validation set)
- ◆ K-折交叉验证 (K-fold Cross Validation)
 - 将原始数据分成K组(一般是均分，例如K为10)，将每个子集数据分别做一次验证集,其余的K-1组子集数据作为训练集，这样得到K个模型
 - K个模型的分类准确率平均数作为分类器的性能指标
- ◆ 留一法 (Leave-One-Out)
 - 每个样本单独作为验证集，其余的N-1个样本作为训练集

小结

➤ 机器学习的基本任务

- ◆ 回归：线性回归方法
- ◆ 分类：
 - 两类分类（二分类）问题：感知机、逻辑回归
- ◆ 模型的评估

➤ 优化问题求解方法

- ◆ 目标函数
- ◆ 误差反向传播，利用梯度下降法调整参数

本周需要掌握的内容

➤ 概念

- ◆ 机器学习中的回归与分类任务
- ◆ 用于优化问题求解的目标函数、梯度下降法
- ◆ 人工神经元（线性加权、激活函数）

➤ 数学基础

- ◆ 函数求导
- ◆ 均方误差
- ◆ 二分类交叉熵

➤ 编程实践

- ◆ 网络学堂“课程文件”→“编程实践”栏目中的“第二周-Pytorch编程实践教程”
 - [tutorial-2-1-pytorch.ipynb](#) PyTorch基础
 - [tutorial-2-2-tensor.ipynb](#) PyTorch 张量操作

参考书

➤ 周志华，机器学习，清华大学出版社，2016年

◆ 第三章第3.1-3.3节



➤ 李航，统计学习方法，清华大学出版社，2012年

◆ 第二章



➤ 《动手学深度学习》(PyTorch版)

◆ 第二章“预备知识”

◆ 第三章“深度学习基础”第3.1-3.3节

<https://github.com/ShusenTang/Dive-into-DL-PyTorch>



谢谢大家！