

# 机器学习

By Asteria

## 1. 绪论

### 1.1. 定义与术语

形式化定义：

- P：计算机程序在某任务类T上的性能
- T：计算机程序希望实现的任务类
- E：表示经验，即历史的数据集

若计算机程序通过利用经验E在任务T上获得了性能P的改善，则称该程序对E进行了学习。

输入空间 $\mathcal{X}$

输出空间 $\mathcal{Y}$

训练数据集+label（标记）=>训练集

版本空间：与训练集一致的假设集合

### 1.2. 问题分类

#### 1.2.1. 监督学习

分类：预测离散值

- 二分类：正类/负类（反类）

$$\mathcal{Y} = -1, 1 \text{ 或 } 0, 1$$

- 多分类

$$|\mathcal{Y}| > 2$$

回归：预测连续值

$$\mathcal{Y} = \mathbb{R} \tag{1}$$

#### 1.2.2. 无监督学习

聚类：事先不标记

### 1.3. 假设空间、版本空间

### 1.4. 归纳偏好

No Free Lunch：没有完美的模型

## 2. 模型评估与选择

## 2.1. 经验误差与过拟合

- 错误率=分类错误样本数/样本总数

$$E = \frac{a}{m} \quad (2)$$

- 精度=(1-错误率)×100%

$$acc = (1 - E) \times 100\% \quad (3)$$

- 误差=实际预测输出-样本真实输出

训练误差/经验误差：训练集上的误差

泛化误差：新样本上的误差

## 2.2. 评估方法

### 2.2.1. 留出法

$$D = S \cup T, S \cap T = \emptyset$$



- 数据分布一致性——分层采样
- 进行若干次随机划分、重复实验后取平均值、标准差



- 偏差-方差角度

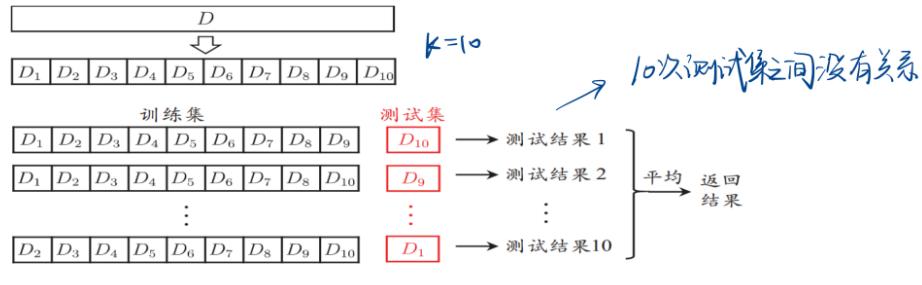
测试集较小时，评估结果方差大；

训练集较小时，评估结果偏差大

### 2.2.2. k折交叉验证法

- 数据集划分为k个子集
  - 大小相似
  - 数据分布一致
  - 互斥
- 每次选1个为测试集，剩下为训练集
- 进行k次训练和测试，求均值

k最常用的取值是10



10 折交叉验证示意图

- p次k折交叉验证  
随机使用不同划分方式（均划分为k个子集）重复p次  
共需进行 $kp$ 次训练和验证

$$|D| = m$$

- $k=m$ , 留1法, 每次测试集只有一个样本  
优点：准确（训练集只比原集合少一个）（留一法估计结果未必最准确）  
缺点：样本多时开销大

### 2.2.3. 自助法/包外估计

有放回随机采样 $|D| = m$ 次，得到 $D'$

样本始终不被采到的概率

$$\lim_{m \rightarrow \infty} \left(1 - \frac{1}{m}\right)^m \mapsto \frac{1}{e} \approx 0.368 , \quad (2.1)$$

初始训练集中约有36.8%样本未出现在训练集 $D'$ 中，作为测试集

- 优点：实际训练模型与原模型都使用了 $m$ 个训练样本
- 缺点：改变数据分布，引入估计偏差

数据量足够时，留出法和交叉验证法更常用

## 2.3. 性能度量

- 均方误差

$$E(f; D) = \frac{1}{m} \sum_{i=1}^m (f(\mathbf{x}_i) - y_i)^2 . \quad (2.2)$$

基于数据分布和概率密度函数的表示为

$$E(f; \mathcal{D}) = \int_{\mathbf{x} \sim \mathcal{D}} (f(\mathbf{x}) - y)^2 p(\mathbf{x}) d\mathbf{x}. \quad (2.3)$$

### 2.3.1. 错误率与精度

- 错误率

$$E(f; D) = \frac{1}{m} \sum_{i=1}^m \mathbb{I}(f(\mathbf{x}_i) \neq y_i). \quad (2.4)$$

$$E(f; \mathcal{D}) = \int_{\mathbf{x} \sim \mathcal{D}} \mathbb{I}(f(\mathbf{x}) \neq y) p(\mathbf{x}) d\mathbf{x}, \quad (2.6)$$

- 精度

$$\begin{aligned} \text{acc}(f; D) &= \frac{1}{m} \sum_{i=1}^m \mathbb{I}(f(\mathbf{x}_i) = y_i) \\ &= 1 - E(f; D). \end{aligned} \quad (2.5)$$

$$\begin{aligned} \text{acc}(f; \mathcal{D}) &= \int_{\mathbf{x} \sim \mathcal{D}} \mathbb{I}(f(\mathbf{x}) = y) p(\mathbf{x}) d\mathbf{x} \\ &= 1 - E(f; \mathcal{D}). \end{aligned} \quad (2.7)$$

### 2.3.2. 查准率、查全率、 $F_1$

表 2.1 分类结果混淆矩阵

真实情况	预测结果	
	正例	反例
正例	$TP$ (真正例)	$FN$ (假反例)
反例	$FP$ (假正例)	$TN$ (真反例)

P、N描述预测结果，对应正反

T、F描述预测正误，对应真假

- 查准率：P中T的比例

$$P = \frac{TP}{TP + FP}, \quad (2.8)$$

- 查全率（召回率）：找到的正例占总正例比例

$$R = \frac{TP}{TP + FN}. \quad (2.9)$$

- 一对矛盾的度量

宏xx——对P、R平均

- 宏查准率 macro-P (多个混淆矩阵)

$$\text{macro-}P = \frac{1}{n} \sum_{i=1}^n P_i, \quad (2.12)$$

- 宏查全率

$$\text{macro-}R = \frac{1}{n} \sum_{i=1}^n R_i , \quad (2.13)$$

- 宏F1

$$\text{macro-}F1 = \frac{2 \times \text{macro-}P \times \text{macro-}R}{\text{macro-}P + \text{macro-}R} . \quad (2.14)$$

微xx——对TP、FP等平均

- 微查准率 micro-P

$$\text{micro-}P = \frac{\overline{TP}}{\overline{TP} + \overline{FP}} , \quad (2.15)$$

- 微查全率

$$\text{micro-}R = \frac{\overline{TP}}{\overline{TP} + \overline{FN}} , \quad (2.16)$$

- 微F1

$$\text{micro-}F1 = \frac{2 \times \text{micro-}P \times \text{micro-}R}{\text{micro-}P + \text{micro-}R} . \quad (2.17)$$

### 2.3.3. P-R曲线

将样本按照“可能为正例的概率”由高到低排序，逐个把样本作为正例预测，每次计算查全率和查准率，得到P-R曲线

- 横坐标：查全率
- 纵坐标：查准率

为绘图方便和美观，示意图显示出单调平滑曲线；但现实任务中的P-R曲线常是非单调、不平滑的，在很多局部有上下波动。

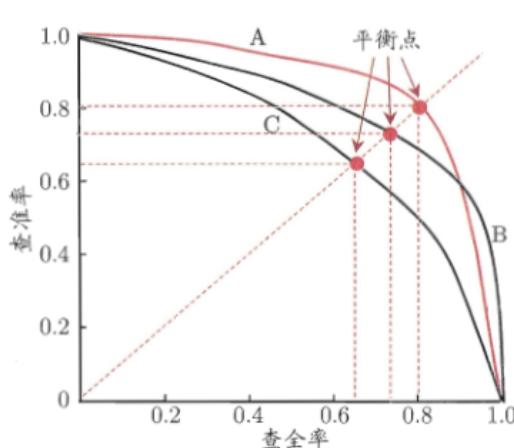


图 2.3 P-R曲线与平衡点示意图

比较学习器：

- 外侧P-R曲线对应的学习器更优——A优于C
- 曲线下面积
- 平衡点 (BEP) :  $P=R$

A优于B

- F1度量：基于P和R的调和平均定义  
与算数平均和几何平均相比，调和平均更重视较小值

$$\frac{1}{F1} = \frac{1}{2} \cdot \left( \frac{1}{P} + \frac{1}{R} \right) \quad (4)$$

$$F1 = \frac{2 \times P \times R}{P + R} = \frac{2 \times TP}{\text{样例总数} + TP - TN} \quad (2.10)$$

- $F_\beta$ : F1的一般形式，加入对P、R的不同偏好（加权调和平均）

$$\frac{1}{F_\beta} = \frac{1}{1 + \beta^2} \left( \frac{1}{P} + \frac{\beta^2}{R} \right) \quad (5)$$

$$F_\beta = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R}, \quad (2.11)$$

$\beta (> 0)$ : 相对重要性

- $\beta > 1$  查全率影响更大
- $\beta < 1$  查准率影响更大

#### 2.3.4. ROC与AUC

##### ROC

- 横轴：假正例率

$$FPR = \frac{FP}{TN + FP}.$$

- 纵轴：真正利率

$$TPR = \frac{TP}{TP + FN},$$

##### 由来

根据学习器预测结果对样例排序

逐个把样本作为正例进行预测

ROC曲线图中的每个坐标点对应一种预测方案（某个预测阈值）

- 对角线(0,0) (1,1)——随机猜测模型（按等可能概率p将正例和反例预测为正，p值即为对应点的横纵坐标）
- 折线(0,0)-(0,1)-(1,1)——理想模型（所有正例排列在反例前面）

##### 基于有限样例绘制——近似ROC曲线

1. 给定 $m^+$ 个正例和 $m^-$ 个反例，根据学习器的预测结果排序（按预测为正例的概率从大到小排列）
2. 分类阈值设置为最大，所有样例均预测为反例
3. TPR=FPR=0, (0,0)点
4. 分类阈值依次设置为每个样例的预测值，即依次将每个样例划分为正例
5. 设前一个标记点为(x,y)
  - 若当前为真正例，则对应 $(x, y + \frac{1}{m^+})$ , 即真正利率增大 $\frac{1}{m^+}$
  - 若当前为假正例，则对应 $(x + \frac{1}{m^-}, y)$ , 即假正例率增大 $\frac{1}{m^-}$
6. 线段连接相邻的点

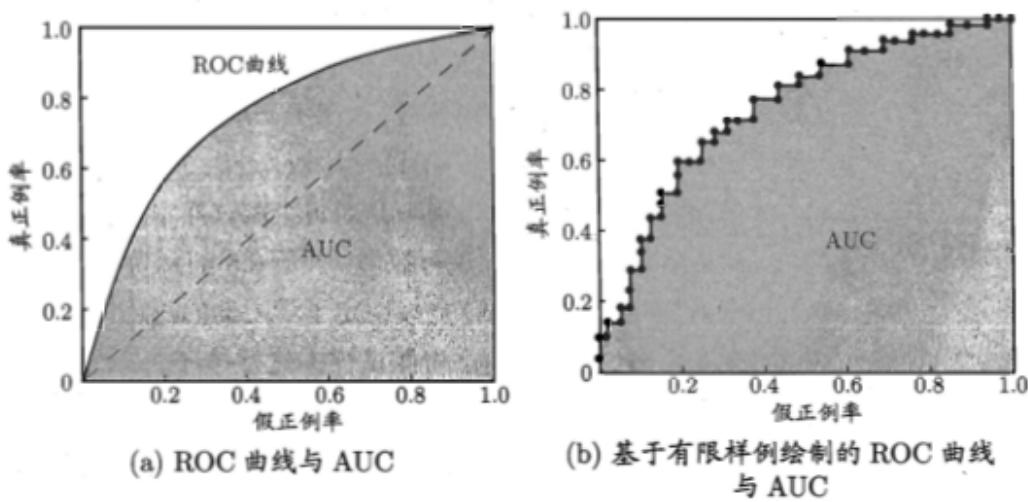


图 2.4 ROC 曲线与 AUC 示意图

## 性能比较

- 外圈的优于内圈的
- 面积 (AUC) 大的优于面积小的，估算方式：

$$AUC = \frac{1}{2} \sum_{i=1}^{m-1} (x_{i+1} - x_i) \cdot (y_i + y_{i+1}) .$$

每个小矩形近似为小直角梯形

## AUC

样本预测的排序质量，而排序质量依赖于学习器的性能

如前所述，面积最大的情况是——所有正例被正确的排列在反例前的理想模型，此时学习器的效果很好

## 排序损失(loss)

$$\ell_{rank} = \frac{1}{m^+ m^-} \sum_{\mathbf{x}^+ \in D^+} \sum_{\mathbf{x}^- \in D^-} \left( \mathbb{I}(f(\mathbf{x}^+) < f(\mathbf{x}^-)) + \frac{1}{2} \mathbb{I}(f(\mathbf{x}^+) = f(\mathbf{x}^-)) \right), \quad (2.21)$$

- $f(+)< f(-)$  1\*罚分
- $f(+)=f(-)$  0.5\*罚分

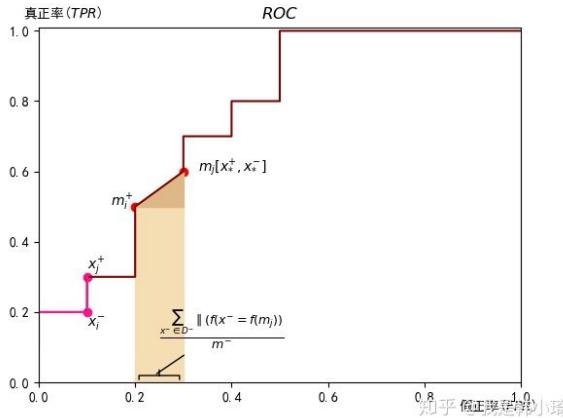
$$AUC = 1 - l_{rank} \quad (6)$$

仿照loss的定义，可以认为AUC积分规则为：

- $f(>)> f(<)$  1\*得分
- $f(>)=f(<)$  0.5\*得分

## 公式证明

显然，在ROC曲线中，一条横线对应一个（或多个）负样本  $x_i^-$ ，一条竖线对应一个（或多个）正样本  $x_j^+$ ，而一条斜线则对应多个正负样本  $x_*^+, x_*^-$ ，且  $f(x_*^+) = f(x_*^-)$ ，即样本的预测值相同。如下图所示：



其中  $m_i^+$  对应一个（或多个预测值相同的）正例，而  $m_j$  对应多个预测值相同的正负例  $x_*^+, x_*^-$

显然阴影部分的宽为  $\frac{\sum_{x^- \in D^-} \| (f(x^-) = f(m_j)) \|}{m^-}$

$m_i^+$  的纵坐标为  $\frac{\sum_{x_+ \in D^+} \| (f(x^+) > f(m_i)) \|}{m^+}$

$m_j$  纵坐标为  $\frac{\sum_{x_+ \in D^+} \| (f(x^+) > f(m_j)) \|}{m^+} + \frac{\sum_{x_+ \in D^+} \| (f(x^+) = f(m_j)) \|}{m^+}$

于是阴影部分面积则为

$$\frac{(\sum_{x^- \in D^-} \| (f(x^-) = f(m_j)) \|) * (\sum_{x^+ \in D^+} (\frac{1}{2} \| (f(x^+) = f(m_j)) \| + \| (f(x^+) > f(m_j)) \|))}{m^- m^+} \quad (7)$$

令  $D_{|f(D^-)|}^-$  表示预测值唯一的负例集合，即原负例集合中以预测值  $f(x)$  去重，那么AUC值为：

$$\sum_{m \in D_{|f(D^-)|}^-} \frac{(\sum_{x^- \in D^-} \| (f(x^-) = f(m)) \|) * (\sum_{x^+ \in D^+} (\frac{1}{2} \| (f(x^+) = f(m)) \| + \| (f(x^+) > f(m)) \|))}{m^- m^+} \quad (8)$$

其中

$$\sum_{m \in D_{|f(D^-)|}^-} \left( \sum_{x^- \in D^-} \| (f(x^-) = f(m)) \| \right) * \sum_{x^+ \in D^+} \| (f(x^+) = f(m)) \| = \sum_{x^+ \in D^+} \sum_{x^- \in D^-} \| (f(x^+) = f(x^-)) \|$$

而

$$\sum_{m \in D_{|f(D^-)|}^-} \left( \sum_{x^- \in D^-} \| (f(x^-) = f(m)) \| \right) * \sum_{x^+ \in D^+} \| (f(x^+) > f(m)) \| = \sum_{x^+ \in D^+} \sum_{x^- \in D^-} \| (f(x^+) > f(x^-)) \| \quad (9)$$

于是

$$AUC = \frac{1}{m^+ m^-} \sum_{x^+ \in D^+} \sum_{x^- \in D^-} (\| (f(x^+) > f(x^-)) \| + \frac{1}{2} \| (f(x^+) = f(x^-)) \|) \quad (10)$$

于是  $AUC + l_{rank} = 1$

### 2.3.5. 代价敏感错误率与代价曲线

**非均等代价**——为权衡不同类型的错误所造成的不同损失

**代价矩阵**

$$cost_{ii} = 0$$

更关注代价比值

表 2.2 二分类代价矩阵

真实类别	预测类别	
	第 0 类	第 1 类
第 0 类	0	$cost_{01}$
第 1 类	$cost_{10}$	0

以上表中第0类为正例，第1类为反例，**代价敏感的错误率**

$$E(f; D; cost) = \frac{1}{m} \left( \sum_{x_i \in D^+} \mathbb{I}(f(x_i) \neq y_i) \times cost_{01} + \sum_{x_i \in D^-} \mathbb{I}(f(x_i) \neq y_i) \times cost_{10} \right).$$

非均等代价下，ROC曲线不能直接反映学习器的期望总体代价

#### 代价曲线(cost curve)

- 横轴：[0,1]（正例概率代价）

$$P(+)cost = \frac{p \times cost_{01}}{p \times cost_{01} + (1-p) \times cost_{10}},$$

p：样例为正例的概率

- 纵轴：归一化代价

$$cost_{norm} = \frac{FNR \times p \times cost_{01} + FPR \times (1-p) \times cost_{10}}{p \times cost_{01} + (1-p) \times cost_{10}},$$

假反例率  $FNR = 1 - TPR$

#### 绘制方法

- ROC上一点对应代价平面上一条线段  
(TPR,FPR)——线段(0,FPR)-(1,FNR=1-TPR)
- 将ROC曲线上的每个点转化为代价平面上的一条线段
- 取所有线段的下界

每条线段下的面积代表该条件下的期望总体代价

所有下界围成的面积为所有条件下学习器的期望总体代价

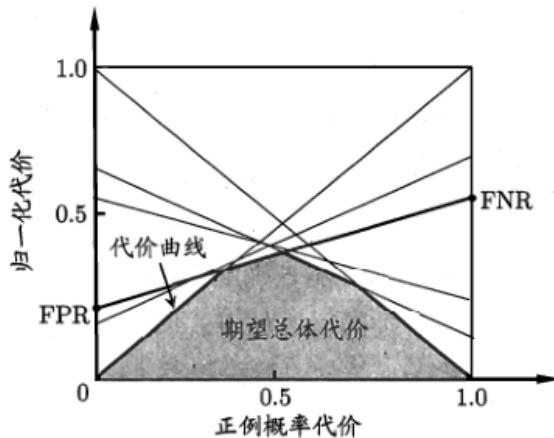


图 2.5 代价曲线与期望总体代价

## 2.4. 比较检验

比较学习器性能存在的问题：

- 希望比较泛化性能，但实验评估的是在某些测试集上体现出来的性能
- 测试集上的性能和测试集的选择有关
- 机器学习算法本身包含随机性，相同参数在同一测试集上多次运行结果未必相同

以错误率 $\epsilon$ 为性能指标

### 2.4.1. 假设检验

假设：对错误率分布的猜想

假定测试样本是从样本总体分布中独立采样得到

设学习器的泛化错误率 $= \epsilon$ ，测试错误率 $\hat{\epsilon}$ ，测试样本 $m$ 个

测试样本中恰有 $\hat{\epsilon} \times m$ 个被误分类

则该学习器将 $m'$ 个样本误分类，其余正确分类的概率 $= \epsilon^{m'} (1 - \epsilon)^{m-m'}$

则其恰将 $\hat{\epsilon} \times m$ 个样本误分类的概率为

$$P(\hat{\epsilon}; \epsilon) = \binom{m}{\hat{\epsilon} \times m} \epsilon^{\hat{\epsilon} \times m} (1 - \epsilon)^{m - \hat{\epsilon} \times m}.$$

也即：该学习器被测得测试错误率为 $\hat{\epsilon}$ 的概率

解  $\partial P(\hat{\epsilon}; \epsilon) / \partial \epsilon = 0$  可知， $P(\hat{\epsilon}; \epsilon)$  在  $\epsilon = \hat{\epsilon}$  时最大

$|\epsilon - \hat{\epsilon}|$  越大， $P(\hat{\epsilon}; \epsilon)$  越小，符合二项分布，如下图

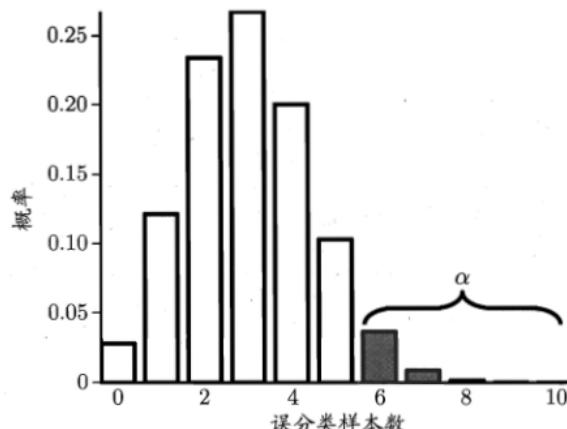


图 2.6 二项分布示意图( $m = 10, \epsilon = 0.3$ )

$m = 10$ , 测得3个被误分类的概率最大

#### 2.4.1.1. 二项检验

用二项检验检验  $\epsilon \leq 0.3$

假设  $\epsilon \leq \epsilon_0$

拒绝域  $\{\epsilon > \bar{\epsilon}\}$

在  $1 - \alpha$  概率内能观测到的最大错误率:

$$\bar{\epsilon} = \max \epsilon \quad \text{s.t.} \quad \sum_{i=\epsilon_0 \times m+1}^m \binom{m}{i} \epsilon^i (1-\epsilon)^{m-i} < \alpha .$$

(控制“在  $\epsilon \leq \epsilon_0$  成立的情况下错误的拒绝该假设/认为  $\epsilon > \epsilon_0$  (第一类错误——错杀好人) ”的概率小于  $\alpha$ , 只是这里“认为  $\epsilon > \epsilon_0$  ”的概率符合二项分布)

满足上式的最大  $\epsilon$  使得第二类错误概率最小

式子左边对应于图2.6阴影部分

下面比较  $\bar{\epsilon}$  和拒绝域

$1 - \alpha$  置信度下.....

#### 2.4.1.2. t检验

$k$  个测试错误率, 平均错误率  $\mu$ , 方差  $\sigma^2$

$$\mu = \frac{1}{k} \sum_{i=1}^k \hat{\epsilon}_i ,$$

$$\sigma^2 = \frac{1}{k-1} \sum_{i=1}^k (\hat{\epsilon}_i - \mu)^2 .$$

考虑到这  $k$  个测试错误率可看作泛化错误率  $\epsilon_0$  的独立采样, 则变量

$$\tau_t = \frac{\sqrt{k}(\mu - \epsilon_0)}{\sigma} \quad (2.30)$$

服从自由度为  $k-1$  的  $t$  分布, 如图 2.7 所示.

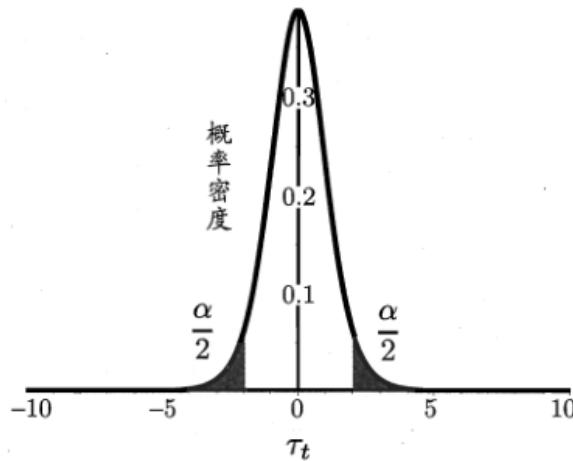


图 2.7  $t$  分布示意图( $k = 10$ )

假设:  $\mu = \epsilon_0$

对假设“ $\mu = \epsilon_0$ ”和显著度  $\alpha$ , 我们可计算出当测试错误率均值为  $\epsilon_0$  时, 在  $1 - \alpha$  概率内能观测到的最大错误率, 即临界值. 这里考虑双边(two-tailed)假设, 如图 2.7 所示, 两边阴影部分各有  $\alpha/2$  的面积; 假定阴影部分范围分别为  $[-\infty, t_{-\alpha/2}]$  和  $[t_{\alpha/2}, \infty]$ . 若平均错误率  $\mu$  与  $\epsilon_0$  之差  $|\mu - \epsilon_0|$  位于临界值范围

$[t_{-\alpha/2}, t_{\alpha/2}]$  内, 则不能拒绝假设“ $\mu = \epsilon_0$ ”, 即可认为泛化错误率为  $\epsilon_0$ , 置信度为  $1 - \alpha$ ; 否则可拒绝该假设, 即在该显著度下可认为泛化错误率与  $\epsilon_0$  有显著不同.  $\alpha$  常用取值有 0.05 和 0.1. 表 2.3 给出了一些常用临界值.

表 2.3 双边  $t$  检验的常用临界值

$\alpha$	$k$				
	2	5	10	20	30
0.05	12.706	2.776	2.262	2.093	2.045
0.10	6.314	2.132	1.833	1.729	1.699

## 2.4.2. 交叉验证 $t$ 检验

### 2.4.2.1. 成对 $t$ 检验

基本思想: 若两个学习器性能相同, 则它们使用相同的训练/测试集 (同一折) 得到的测试错误率应相同

对两个学习器 A 和 B, 若我们使用  $k$  折交叉验证法得到的测试错误率分别为  $\epsilon_1^A, \epsilon_2^A, \dots, \epsilon_k^A$  和  $\epsilon_1^B, \epsilon_2^B, \dots, \epsilon_k^B$ , 其中  $\epsilon_i^A$  和  $\epsilon_i^B$  是在相同的第  $i$  折训练/测试集上得到的结果, 则可用  $k$  折交叉验证 “成对  $t$  检验” (paired  $t$ -tests) 来进行比较检验. 这里的基本思想是若两个学习器的性能相同, 则它们使用相同的训练/测试集得到的测试错误率应相同, 即  $\epsilon_i^A = \epsilon_i^B$ .

具体来说, 对  $k$  折交叉验证产生的  $k$  对测试错误率: 先对每对结果求差,  $\Delta_i = \epsilon_i^A - \epsilon_i^B$ ; 若两个学习器性能相同, 则差值均值应为零. 因此, 可根据差值  $\Delta_1, \Delta_2, \dots, \Delta_k$  来对“学习器 A 与 B 性能相同”这个假设做  $t$  检验, 计算出差值的均值  $\mu$  和方差  $\sigma^2$ , 在显著度  $\alpha$  下, 若变量

$$\tau_t = \left| \frac{\sqrt{k}\mu}{\sigma} \right| \quad (2.31)$$

小于临界值  $t_{\alpha/2, k-1}$ , 则假设不能被拒绝, 即认为两个学习器的性能没有显著差别; 否则可认为两个学习器的性能有显著差别, 且平均错误率较小的那个学习器性能较优. 这里  $t_{\alpha/2, k-1}$  是自由度为  $k - 1$  的  $t$  分布上尾部累积分布为  $\alpha/2$  的临界值.

### 2.4.2.2. 5x2 交叉验证

$5 \times 2$  交叉验证是做 5 次 2 折交叉验证, 在每次 2 折交叉验证之前随机将数据打乱, 使得 5 次交叉验证中的数据划分不重复. 对两个学习器 A 和 B, 第  $i$  次 2 折交叉验证将产生两对测试错误率, 我们对它们分别求差, 得到第 1 折上的差值  $\Delta_i^1$  和第 2 折上的差值  $\Delta_i^2$ . 为缓解测试错误率的非独立性, 我们仅计算第 1 次 2 折交叉验证的两个结果的平均值  $\mu = 0.5(\Delta_1^1 + \Delta_1^2)$ , 但对每次 2 折实验的结果都计算出其方差  $\sigma_i^2 = \left(\Delta_i^1 - \frac{\Delta_1^1 + \Delta_1^2}{2}\right)^2 + \left(\Delta_i^2 - \frac{\Delta_1^1 + \Delta_1^2}{2}\right)^2$ . 变量

$$\tau_t = \frac{\mu}{\sqrt{0.2 \sum_{i=1}^5 \sigma_i^2}} \quad (2.32)$$

服从自由度为 5 的  $t$  分布, 其双边检验的临界值  $t_{\alpha/2, 5}$  当  $\alpha = 0.05$  时为 2.5706,  $\alpha = 0.1$  时为 2.0150.

核心做法: 5 次 2 折实验, 只计算第一次的均值, 分别计算五次的方差, 5 词方差均值作为最终方差

#### 2.4.3. McNemar 检验

表 2.4 两学习器分类差别列联表

算法 B	算法 A	
	正确	错误
正确	$e_{00}$	$e_{01}$
错误	$e_{10}$	$e_{11}$

假设: 两学习器性能相同,  $e_{01} = e_{10}$

$|e_{01} - e_{10}|$  应当服从正态分布, 且均值为 1, 方差为  $e_{01} + e_{10}$ . 因此变量

$$\tau_{\chi^2} = \frac{(|e_{01} - e_{10}| - 1)^2}{e_{01} + e_{10}} \quad (2.33)$$

服从自由度为 1 的  $\chi^2$  分布, 即标准正态分布变量的平方. 给定显著度  $\alpha$ , 当以上变量值小于临界值  $\chi_{\alpha}^2$  时, 不能拒绝假设, 即认为两学习器的性能没有显著差别; 否则拒绝假设, 即认为两者性能有显著差别, 且平均错误率较小的那个学习器性能较优. 自由度为 1 的  $\chi^2$  检验的临界值当  $\alpha = 0.05$  时为 3.8415,  $\alpha = 0.1$  时为 2.7055.

大于临界值时拒绝原假设

#### 2.4.4. Friedman 检验和 Nemenyi 后续检验

课本/ppt

##### 2.4.4.1. Friedman 检验——F 分布

多个算法参与比较

大于临界值时拒绝原假设

#### 2.4.4.2. Nemenyi后续检验——Tukey分布

当“所有算法性能相同”的假设被拒绝时，进行后续检验区分算法

## 2.5. 偏差与方差

偏差-方差分解：解释学习算法泛化性能

- 期望预测

$$\bar{f}(\mathbf{x}) = \mathbb{E}_D[f(\mathbf{x}; D)] ,$$

- 方差

使用样本数相同的不同训练集产生的方差为

$$var(\mathbf{x}) = \mathbb{E}_D \left[ (f(\mathbf{x}; D) - \bar{f}(\mathbf{x}))^2 \right] ,$$

方差度量了同样大小的训练集的变动导致的学习性能的变化，刻画数据扰动所造成的影响

- 噪声

噪声为

$$\varepsilon^2 = \mathbb{E}_D \left[ (y_D - y)^2 \right] .$$

噪声表达当前任务上任何学习算法所能达到的期望泛化误差的下界，刻画学习问题本身的高度

- 偏差

期望输出与真实标记的差别称为偏差(bias)，即

$$bias^2(\mathbf{x}) = (\bar{f}(\mathbf{x}) - y)^2 .$$

偏差度量学习算法的期望预测与真实结果的偏离程度，刻画学习算法本身的拟合能力

为便于讨论，假定噪声期望为零，即  $\mathbb{E}_D[y_D - y] = 0$ .

$$E(f; D) = bias^2(\mathbf{x}) + var(\mathbf{x}) + \varepsilon^2 ,$$

也就是说，泛化误差可分解为偏差、方差与噪声之和。

上式仅在基于均方误差的回归任务中得以推导出。

**偏差-方差窘境**

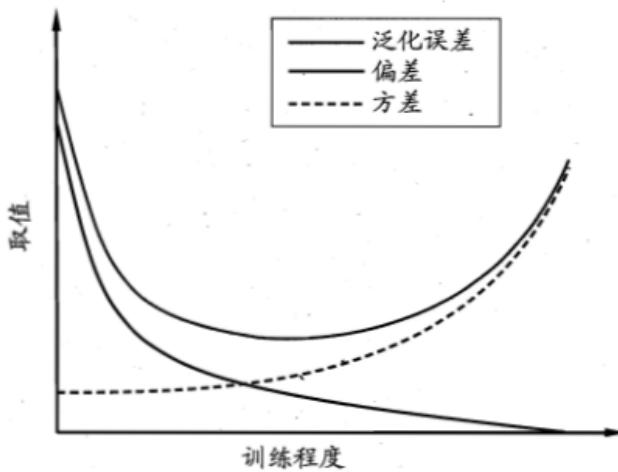


图 2.9 泛化误差与偏差、方差的关系示意图

### 3. 线性模型

#### 3.1. 基本形式

$d$ 个属性描述的示例  $\vec{x} = (x_1; x_2; \dots; x_d)$

$x_i$ 是示例在第 $i$ 个属性上的取值

##### 线性模型

通过属性的线性组合进行预测

$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \dots + w_d x_d + b ,$$

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

$w$ 表达各属性在预测中的重要性

#### 3.2. 线性回归

##### 数据集

$$D = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \dots, (\vec{x}_m, y_m)\} \quad (11)$$

(示例，标记)

##### 离散属性

若属性值间存在“序”的关系，可通过连续化将其转化为连续值（方可计算距离）

如：

{高，低}→{1.0, 0.0}

{高，中，低}→{1.0, 0.5, 0.0}

若属性值间不存在序关系， $k$ 个属性值通常转化为 $k$ 维向量（类似独热码编码）

如：

{黄瓜，西瓜，南瓜}→{(0,0,1), (0,1,0), (1,0,0)}

### 3.2.1. 单属性

#### 3.2.1.1. 模型建立

即  $D = \{(x_i, y_i)\}_{i=1}^m$ , 其中  $x_i \in \mathbb{R}$ .

线性回归试图学得

$$f(x_i) = wx_i + b, \text{ 使得 } f(x_i) \simeq y_i .$$

确定  $w, b$ —均方误差最小化, 即最小二乘法

均方误差

回归任务最常用的性能度量是“均方误差”(mean squared error)

$$E(f; D) = \frac{1}{m} \sum_{i=1}^m (f(x_i) - y_i)^2 . \quad (2.2)$$

更一般的, 对于数据分布  $\mathcal{D}$  和概率密度函数  $p(\cdot)$ , 均方误差可描述为

$$E(f; \mathcal{D}) = \int_{\mathbf{x} \sim \mathcal{D}} (f(\mathbf{x}) - y)^2 p(\mathbf{x}) d\mathbf{x} . \quad (2.3)$$

几何意义: 欧氏距离

$$\begin{aligned} (w^*, b^*) &= \arg \min_{(w, b)} \sum_{i=1}^m (f(x_i) - y_i)^2 \\ &= \arg \min_{(w, b)} \sum_{i=1}^m (y_i - wx_i - b)^2 . \end{aligned}$$

#### 3.2.1.2. 最小二乘参数估计

$$E_{(w,b)} = \sum_{i=1}^m (y_i - wx_i - b)^2$$

是关于  $w$  和  $b$  的凸函数

$$\frac{\partial E_{(w,b)}}{\partial w} = 2 \left( w \sum_{i=1}^m x_i^2 - \sum_{i=1}^m (y_i - b) x_i \right) , \quad (3.5)$$

$$\frac{\partial E_{(w,b)}}{\partial b} = 2 \left( mb - \sum_{i=1}^m (y_i - wx_i) \right) , \quad (3.6)$$

然后令式(3.5)和(3.6)为零可得到  $w$  和  $b$  最优解的闭式(closed-form)解

$$w = \frac{\sum_{i=1}^m y_i (x_i - \bar{x})}{\sum_{i=1}^m x_i^2 - \frac{1}{m} \left( \sum_{i=1}^m x_i \right)^2} , \quad (3.7)$$

$$b = \frac{1}{m} \sum_{i=1}^m (y_i - wx_i) , \quad (3.8)$$

其中  $\bar{x} = \frac{1}{m} \sum_{i=1}^m x_i$  为  $x$  的均值.

### 3.2.2. 多元线性模型

#### 3.2.2.1. 模型建立

参数向量（列向量）

$$\hat{\vec{w}} = (\vec{w}; b) = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ b \end{pmatrix} \quad (12)$$

数据集属性矩阵

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1d} & 1 \\ x_{21} & x_{22} & \dots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \dots & x_{md} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{x}_1^T & 1 \\ \mathbf{x}_2^T & 1 \\ \vdots & \vdots \\ \mathbf{x}_m^T & 1 \end{pmatrix} ,$$

标记向量（列向量）

$$\vec{y} = (y_1; y_2; \dots; y_m) = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \quad (13)$$

$$\mathbf{X} \hat{\vec{w}} = \begin{pmatrix} f(\vec{x}_1) \\ f(\vec{x}_2) \\ \vdots \\ f(\vec{x}_m) \end{pmatrix} \quad (14)$$

#### 3.2.2.2. 参数估计

$$\hat{\vec{w}}^* = \arg \min_{\hat{\vec{w}}} (\mathbf{y} - \mathbf{X} \hat{\vec{w}})^T (\mathbf{y} - \mathbf{X} \hat{\vec{w}}) . \quad (3.9)$$

令  $E_{\hat{\vec{w}}} = (\mathbf{y} - \mathbf{X} \hat{\vec{w}})^T (\mathbf{y} - \mathbf{X} \hat{\vec{w}})$ , 对  $\hat{\vec{w}}$  求导得到

$$\frac{\partial E_{\hat{\vec{w}}}}{\partial \hat{\vec{w}}} = 2 \mathbf{X}^T (\mathbf{X} \hat{\vec{w}} - \mathbf{y}) . \quad (3.10)$$

求导方法：

将  $E_{\hat{w}} = (\mathbf{y} - \mathbf{X}\hat{\mathbf{w}})^T(\mathbf{y} - \mathbf{X}\hat{\mathbf{w}})$  展开可得

$$E_{\hat{w}} = \mathbf{y}^T \mathbf{y} - \mathbf{y}^T \mathbf{X} \hat{\mathbf{w}} - \hat{\mathbf{w}}^T \mathbf{X}^T \mathbf{y} + \hat{\mathbf{w}}^T \mathbf{X}^T \mathbf{X} \hat{\mathbf{w}}$$

对  $\hat{\mathbf{w}}$  求导可得

$$\frac{\partial E_{\hat{w}}}{\partial \hat{\mathbf{w}}} = \frac{\partial \mathbf{y}^T \mathbf{y}}{\partial \hat{\mathbf{w}}} - \frac{\partial \mathbf{y}^T \mathbf{X} \hat{\mathbf{w}}}{\partial \hat{\mathbf{w}}} - \frac{\partial \hat{\mathbf{w}}^T \mathbf{X}^T \mathbf{y}}{\partial \hat{\mathbf{w}}} + \frac{\partial \hat{\mathbf{w}}^T \mathbf{X}^T \mathbf{X} \hat{\mathbf{w}}}{\partial \hat{\mathbf{w}}}$$

由矩阵微分公式  $\frac{\partial \mathbf{a}^T \mathbf{x}}{\partial \mathbf{x}} = \frac{\partial \mathbf{x}^T \mathbf{a}}{\partial \mathbf{x}} = \mathbf{a}$ ,  $\frac{\partial \mathbf{x}^T \mathbf{A} \mathbf{x}}{\partial \mathbf{x}} = (\mathbf{A} + \mathbf{A}^T) \mathbf{x}$  (更多矩阵微分公式可查阅 [2], 矩阵微分原理可查阅 [3]) 可得

$$\begin{aligned} \frac{\partial E_{\hat{w}}}{\partial \hat{\mathbf{w}}} &= 0 - \mathbf{X}^T \mathbf{y} - \mathbf{X}^T \mathbf{y} + (\mathbf{X}^T \mathbf{X} + \mathbf{X}^T \mathbf{X}) \hat{\mathbf{w}} \\ &= 2\mathbf{X}^T (\mathbf{X} \hat{\mathbf{w}} - \mathbf{y}) \end{aligned}$$

讨论

- $\mathbf{X}^T \mathbf{X}$  满秩或正定

1. 正定一定满秩

2.  $\text{rank}\{X^T X\} = \text{rank}\{XX^T\} = \text{rank}\{X^T\} = \text{rank}\{X\} = d + 1$ , 于是  $\mathbf{X}$  是列满秩的

当  $\mathbf{X}^T \mathbf{X}$  为满秩矩阵(full-rank matrix)或正定矩阵(positive definite matrix)时, 令式(3.10)为零可得

$$\hat{\mathbf{w}}^* = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}, \quad (3.11)$$

$$\text{令 } \hat{\mathbf{x}}_i = (\mathbf{x}_i, 1)$$

线性回归模型为

$$f(\hat{\mathbf{x}}_i) = \hat{\mathbf{x}}_i^T (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}. \quad (3.12)$$

- 其他

可解出多个  $\hat{\mathbf{w}}$ , 它们都能使均方误差最小化.

根据学习算法的归纳偏好选择作为输出的解, 常见的做法是引入正则项

详细推导见 [公式详解](#)

### 3.2.3. 变形

#### 3.2.3.1. 对数线性回归

$$\ln y = \mathbf{w}^T \mathbf{x} + b. \quad (3.14)$$

试图让  $\exp\{\vec{w}^T \vec{x} + b\}$  逼近  $y$

实际求的是输入空间到输出空间的非线性函数映射

样例  $\xrightarrow[\text{线性映射}]{f(x)}$  预测值  $\ln y \xrightarrow[\text{非线性关系}]{} \text{与真实标记建立联系}$  (15)

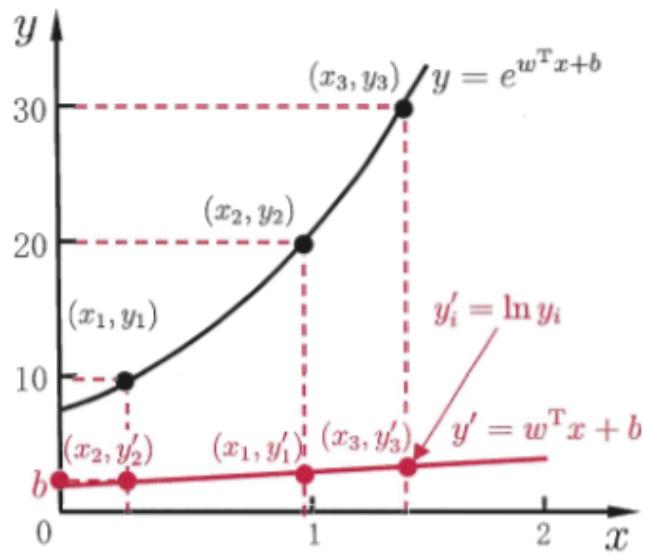


图 3.1 对数线性回归示意图

### 3.2.3.2. 广义线性模型

$g(\cdot)$  连续且充分光滑.

更一般地, 考虑单调可微函数  $g(\cdot)$ , 令

$$y = g^{-1}(w^T x + b), \quad (3.15)$$

这样得到的模型称为“广义线性模型”(generalized linear model), 其中函数  $g(\cdot)$  称为“联系函数”(link function). 显然, 对数线性回归是广义线性模型在  $g(\cdot) = \ln(\cdot)$  时的特例.

## 3.3. 对数几率回归

利用广义线性模型, 找一个单调可微函数将分类任务的真实标记和预测值联系起来

### 3.3.1. 二分类任务

#### 3.3.1.1. 模型建立

**Heaviside函数/单位阶跃函数 (unit-step function)**

$$y = \begin{cases} 0, & z < 0; \\ 0.5, & z = 0; \\ 1, & z > 0, \end{cases} \quad (3.16)$$

即若预测值  $z$  大于零就判为正例, 小于零则判为反例, 预测值为临界值零则可任意判别, 如图 3.2 所示.

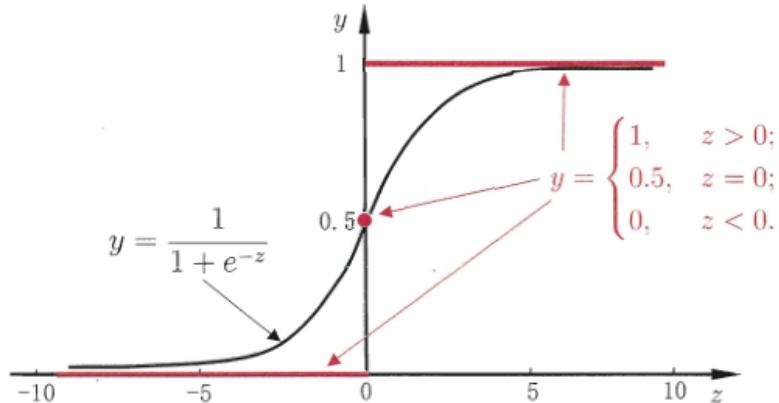


图 3.2 单位阶跃函数与对数几率函数

但不连续, 找近似替代函数

**对数几率函数(logistic function)/对率函数**

$$y = \frac{1}{1 + e^{-z}}. \quad (3.17)$$

- 是一种Sigmoid函数, 输出为接近0或1的值, 且在 $z=0$ 附近很陡
- 任意阶可导
- 凸函数

带入

$$y = g^{-1}(\mathbf{w}^T \mathbf{x} + b), \quad (3.15)$$

得到

$$y = \frac{1}{1 + e^{-(\mathbf{w}^T \mathbf{x} + b)}}. \quad (3.18)$$

$$\ln \frac{y}{1-y} = \mathbf{w}^T \mathbf{x} + b. \quad (3.19)$$

若将  $y$  视为样本  $\mathbf{x}$  作为正例的可能性，则  $1 - y$  是其反例可能性，两者的比值

$$\frac{y}{1 - y} \quad (3.20)$$

称为“几率”(odds)，反映了  $\mathbf{x}$  作为正例的相对可能性。对几率取对数则得到“对数几率”(log odds，亦称 logit)

$$\ln \frac{y}{1 - y}. \quad (3.21)$$

优点

- 对分类问题不需假设数据分布，而是直接对分类的可能性进行建模
- 不仅给出类别预测，而且给出近似概率预测

### 3.3.1.2. 参数估计

书 (没看懂。。)

**先验概率 (prior probability)**：指根据以往经验和分析。在实验或采样前就可以得到的概率。

**后验概率 (posterior probability)**：指某件事已经发生，想要计算这件事发生的原因是由某个因素引起概率。

最小化

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^m \left( -y_i \boldsymbol{\beta}^T \hat{\mathbf{x}}_i + \ln \left( 1 + e^{\boldsymbol{\beta}^T \hat{\mathbf{x}}_i} \right) \right). \quad (3.27)$$

式(3.27)是关于  $\boldsymbol{\beta}$  的高阶可导连续凸函数，根据凸优化理论 [Boyd and Vandenberghe, 2004]，经典的数值优化算法如梯度下降法(gradient descent method)、牛顿法(Newton method)等都可求得其最优解，于是就得到

$$\boldsymbol{\beta}^* = \arg \min \ell(\boldsymbol{\beta}). \quad (3.28)$$

## 3.4. 线性判别分析 (Linear)

基本思想：给定训练样例集，设法将样例投影到一条直线上，使得同类样例的投影点尽可能接近、异类样例投影点尽可能远离

面对新样本时：将其投影到同样的这条直线上，再根据投影点的位置来确定新样本的类别

### 3.4.1. 二分类

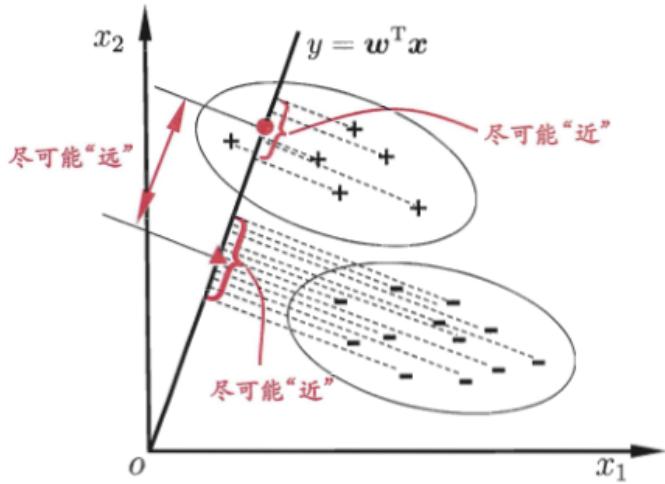


图 3.3 LDA 的二维示意图。‘+’、‘-’ 分别代表正例和反例，椭圆表示数据簇的外轮廓，虚线表示投影，红色实心圆和实心三角形分别表示两类样本投影后的中心点。

数据集  $D = \{(\hat{x}_i, y_i)\}_{i=1}^m, y_i \in \{0, 1\}\}$

第*i*类示例集合  $\mathbf{X}_i$ 、均值向量  $\mu_i$ 、协方差矩阵  $\Sigma_i$

投影直线  $w$

投影中心  $w^T \mu_0, w^T \mu_1$

协方差  $w^T \Sigma_0 w, w^T \Sigma_1 w$

直线是一维空间，故上面四个值都是实数

- 同类样例投影点尽可能近 → 同类样例投影点协方差尽可能小

即  $w^T \Sigma_0 w + w^T \Sigma_1 w$  尽量小

- 异类样例投影点尽可能远 → 类中心之间的距离尽可能大

即  $\|w^T \mu_0 - w^T \mu_1\|_2^2$  尽可能大

- 最大化目标

$$\begin{aligned} J &= \frac{\|w^T \mu_0 - w^T \mu_1\|_2^2}{w^T \Sigma_0 w + w^T \Sigma_1 w} \\ &= \frac{w^T (\mu_0 - \mu_1)(\mu_0 - \mu_1)^T w}{w^T (\Sigma_0 + \Sigma_1) w} . \end{aligned}$$

- 类内散度矩阵

$$\begin{aligned} \mathbf{S}_w &= \Sigma_0 + \Sigma_1 \\ &= \sum_{x \in X_0} (x - \mu_0)(x - \mu_0)^T + \sum_{x \in X_1} (x - \mu_1)(x - \mu_1)^T \end{aligned}$$

对称矩阵

- 类间散度矩阵

$$\mathbf{S}_b = (\mu_0 - \mu_1)(\mu_0 - \mu_1)^T ,$$

- 最大化目标化为  $-S_b$  与  $S_w$  的广义瑞利商

$$J = \frac{w^T \mathbf{S}_b w}{w^T \mathbf{S}_w w} .$$

- 求解  $w$

$w$ 是解，则 $w$ 的任意常数倍也是解

$$\text{令 } \mathbf{w}^T \mathbf{S}_w \mathbf{w} = 1,$$

等价于

$$\begin{aligned} & \min_{\mathbf{w}} -\mathbf{w}^T \mathbf{S}_b \mathbf{w} \\ \text{s.t. } & \mathbf{w}^T \mathbf{S}_w \mathbf{w} = 1. \end{aligned}$$

### 拉格朗日乘子法

由拉格朗日乘子法，上式等价于

$$\mathbf{S}_b \mathbf{w} = \lambda \mathbf{S}_w \mathbf{w}, \quad (3.37)$$

其中  $\lambda$  是拉格朗日乘子。注意到  $\mathbf{S}_b \mathbf{w}$  的方向恒为  $\mu_0 - \mu_1$ ，不妨令

$$\mathbf{S}_b \mathbf{w} = \lambda(\mu_0 - \mu_1), \quad (3.38)$$

代入式(3.37)即得

$$\mathbf{w} = \mathbf{S}_w^{-1}(\mu_0 - \mu_1). \quad (3.39)$$

考虑到数值解的稳定性，在实践中通常是对  $\mathbf{S}_w$  进行奇异值分解，即  $\mathbf{S}_w = \mathbf{U} \Sigma \mathbf{V}^T$ ，这里  $\Sigma$  是一个实对角矩阵，其对角线上的元素是  $\mathbf{S}_w$  的奇异值，然后由  $\mathbf{S}_w^{-1} = \mathbf{V} \Sigma^{-1} \mathbf{U}^T$  得到  $\mathbf{S}_w^{-1}$ 。

值得一提的是，LDA 可从贝叶斯决策理论的角度来阐释，并可证明，当两类数据同先验、满足高斯分布且协方差相等时，LDA 可达到最优分类。

同先验？

### 3.4.2. 多分类

$N$ 个分类，第*i*类示例数为 $m_i$

- 全局散度矩阵

$$\begin{aligned} \mathbf{S}_t &= \mathbf{S}_b + \mathbf{S}_w \\ &= \sum_{i=1}^m (\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^T, \end{aligned}$$

$\hat{\boldsymbol{\mu}}$ 是所有示例的均值向量

- 类内散度矩阵

$$\mathbf{S}_w = \sum_{i=1}^N \mathbf{S}_{w_i},$$

$$\text{其中 } \mathbf{S}_{w_i} = \sum_{\mathbf{x} \in X_i} (\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T.$$

为每个类别的散度矩阵

于是有

- 类间散度矩阵

$$\begin{aligned}\mathbf{S}_b &= \mathbf{S}_t - \mathbf{S}_w \\ &= \sum_{i=1}^N m_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T.\end{aligned}$$

使用上述  $S_b, S_w, S_t$  中的任意两个即可以实现多分类 LDA

常见的，采用优化目标

$$\max_{\mathbf{W}} \frac{\text{tr}(\mathbf{W}^T \mathbf{S}_b \mathbf{W})}{\text{tr}(\mathbf{W}^T \mathbf{S}_w \mathbf{W})}, \quad (3.44)$$

其中  $\mathbf{W} \in \mathbb{R}^{d \times (N-1)}$ ,  $\text{tr}(\cdot)$  表示矩阵的迹(trace). 式(3.44)可通过如下广义特征值问题求解:

$$\mathbf{S}_b \mathbf{W} = \lambda \mathbf{S}_w \mathbf{W}. \quad (3.45)$$

(3.45) 的证明:

$\mathbf{W}$  的闭式解则是  $\mathbf{S}_w^{-1} \mathbf{S}_b$  的  $N - 1$  个最大广义特征值所对应的特征向量组成的矩阵.

若将  $\mathbf{W}$  视为一个投影矩阵，则多分类 LDA 将样本投影到  $N - 1$  维空间， $N - 1$  通常远小于数据原有的属性数. 于是，可通过这个投影来减小样本点的维数，且投影过程中使用了类别信息，因此 LDA 也常被视为一种经典的监督降维技术.

## 3.5. 多分类学习

一般的，多分类拆分成多个二分类，再对二分类的预测结果进行集成获得最终的多分类结果

### 3.5.1. 拆分策略

#### 3.5.1.1. 一对一 OvO

$N$  个类别两两配对，产生  $\frac{N(N-1)}{2}$  个二分类任务

最终预测结果可通过投票产生/根据分类器的预测置信度集成

#### 3.5.1.2. 一对其余 OvR

每次将一个类作为正例，其他作为反例，训练  $N$  个分类器

测试时

- 若仅有一个分类器预测为正类，则该类作为分类结果
- 若多个分类器预测为正类，则考虑各分类器预测置信度最大的作为分类结果

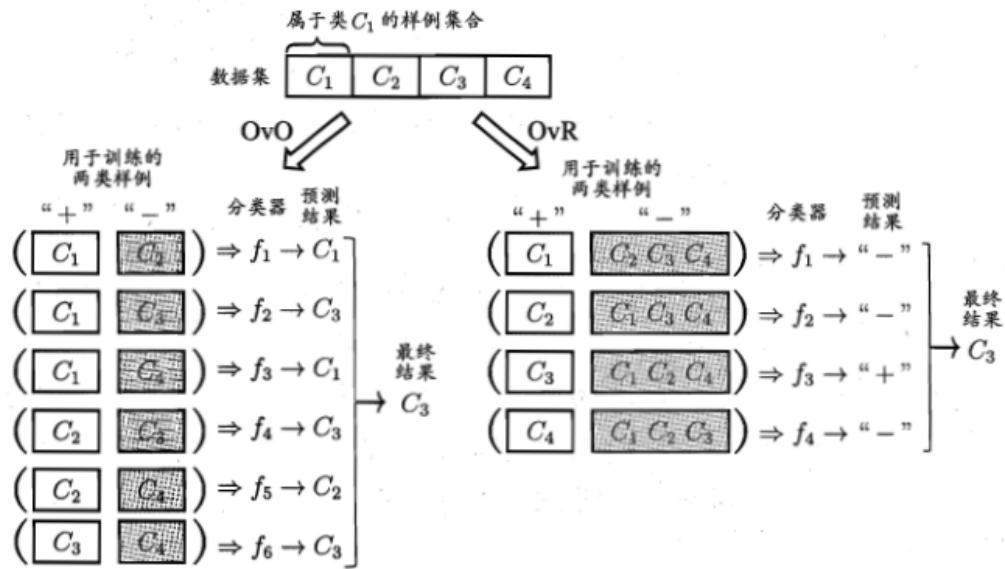


图 3.4 OvO 与 OvR 示意图

比较：

- OvO 存储开销和测试时间开销较 OvR 大
- 但 OvR 的每个分类器训练时均使用全部训练样例，而 OvO 每次只用到两个类别的样例，因此类别很多时，OvO 训练开销通常更小

### 3.5.1.3. 多对多 MvM

每次若干个类作为正类，若干个类作为反类

OvO 和 OvR 都是 MvM 的特例

正反类构造需要特殊设计

常用 MvM 技术——纠错输出码(ECOC)

#### 3.5.1.3.1. ECOC

ECOC [Dietterich and Bakiri, 1995] 是将编码的思想引入类别拆分，并尽可能在解码过程中具有容错性。ECOC 工作过程主要分为两步：

- 编码：对  $N$  个类别做  $M$  次划分，每次划分将一部分类别划为正类，一部分划为反类，从而形成一个二分类训练集；这样一共产生  $M$  个训练集，可训练出  $M$  个分类器。
- 解码： $M$  个分类器分别对测试样本进行预测，这些预测标记组成一个编码。将这个预测编码与每个类别各自的编码进行比较，返回其中距离最小的类别作为最终预测结果。

#### 编码矩阵

用于类别划分

- 二元码：正类/反类
- 三元码：正类/反类/停用类

$C_3$  类的样例作为正例,  $C_2$  和  $C_4$  类的样例作为反例; 在图 3.5(b) 中, 分类器  $f_4$  将  $C_1$  和  $C_4$  类的样例作为正例,  $C_3$  类的样例作为反例. 在解码阶段, 各分类器的预测结果联合起来形成了测试示例的编码, 该编码与各类所对应的编码进行比较, 将距离最小的编码所对应的类别作为预测结果. 例如在图 3.5(a) 中, 若基于欧氏距离, 预测结果将是  $C_3$ .

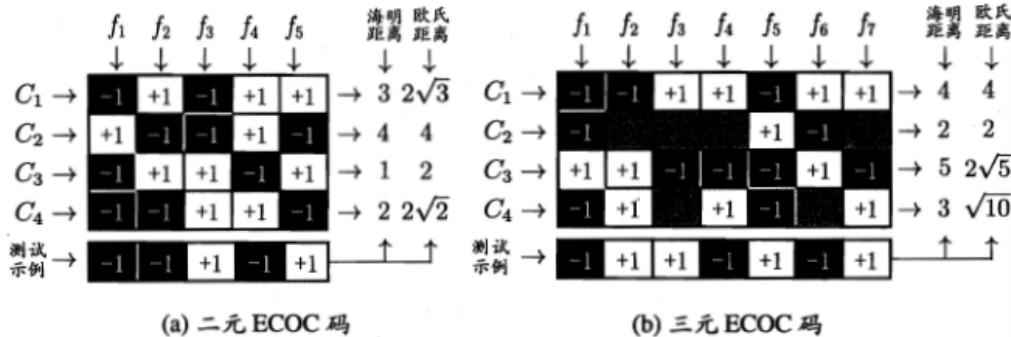


图 3.5 ECOC 编码示意图. “+1”、“-1”分别表示学习器  $f_i$  将该类样本作为正、反例; 三元码中“0”表示  $f_i$  不使用该类样本

海明距离: 两个码对应位置不同的个数; 与停用类不同记为0.5

一般, 对同一个学习任务, ECOC编码越长, 纠错能力越强

假设任意两个类别之间最小的海明距离为d, 那么此纠错输出码最少能矫正  $\frac{d-1}{2}$  位的错误

但编码越长, 需训练的分类器越多, 计算、存储开销越大

有限类别数, 可能的组合数目是有限的, 码长超过一定范围后失去意义

对同等长度的编码, 理论上, 任意两个类别之间的编码距离越短, 则纠错能力越强

### 一个好的ECOC应满足

- 行分离: 任意两个类别之间的codeword足够大
- 列分离: 任意两个分类器  $f_i$ ,  $f_i$  的输出应相互独立, 无关联。这一点可以通过使分类器  $f_i$  编码与其他分类编码的海明距离足够大实现, 且与其他分类编码的反码的海明距离也足够大

如果两个分类器的编码类似或者完全一致, 很多算法 (比如C4.5) 会有相同或者类似的错误分类, 如果这种同时发生的错误过多, 会导致纠错输出码失效。

解释:

若增加两个类似的编码, 那么当误分类时, 就从原来的1变成3, 导致与真实类别的codeword海明距离增长。极端情况, 假设增加两个相同的编码, 此时任意两个类别之间最小的海明距离不会变化依然为d, 而纠错输出码输出的codeword与真实类别的codeword的海明距离激增 (从1变成3)。所以如果有过多同时发出的错误分类, 会导致纠错输出码失效。

两个分类器的编码也不应该互为反码, 因为很多算法 (比如C4.5, 逻辑回归) 对待0-1分类其实是对称的, 即将0-1类互换, 最终训练出的模型是一样的。也就是说两个编码互为补码的分类器是会同时犯错的。同样也会导致纠错输出码失效。

对于  $k$  种类别分类, 去除反码/全0/全1, 剩下  $2^k - 1$  种可行编码

### 构造方法

对  $3 \leq k \leq 7$ , 构造长为  $2^{k-1} - 1$  的编码

第1行: 全1

第2行:  $2^{k-2}$  个0, 后跟  $2^{k-2} - 1$  个1

第3行： $2^{k-3}$ 个0，后跟 $2^{k-3}$ 个1，后跟 $2^{k-3}$ 个0，后跟 $2^{k-3} - 1$ 个1

第*i*行： $2^{k-i}$ 个0和 $2^{k-i}$ 个1交替，最后一组少一个1

### NP问题

最终分类性能受多方面的影响：

- 编码的理论纠错性能
- 二分类子问题难度

## 3.6. 类别不平衡问题

可能导致类别不平衡的原因：

- 拆分多分类时（即使原本没有不平衡），OvR、MvM得到的二分类可能不平衡

对 OvR、MvM 来说，由于对每个类进行了相同的处理，其拆解出的二分类任务中类别不平衡的影响会相互抵消。因此通常不需专门处理。

三类做法

1. 欠采样：对训练集中的反例“欠采样”，即去除一些反例使得正反例数目接近（假设不平衡时反例>正例），**随机丢弃可能丢失重要信息**

代表算法：利用集成学习机制，将反例划分成若干个集合供不同学习器使用，使得全局来看不丢失信息

2. 过采样：对训练集中的正例“过采样”，**不是重复采样**

代表算法

- SMOTE——对训练集中的正例进行插值产生额外的正例
- MIXUP数据增强

3. 阈值移动：基于原始训练集学习，将式(3.48)嵌入决策过程

再放缩策略

“训练集是真实样本总体的无偏采样”并不总成立

未必能有效基于训练集观测几率来推断出真实几率

线性分类器

从线性分类器的角度讨论容易理解，在我们用  $y = \mathbf{w}^T \mathbf{x} + b$  对新样本  $\mathbf{x}$  进行分类时，事实上是在用预测出的  $y$  值与一个阈值进行比较，例如通常在  $y > 0.5$  时判别为正例，否则为反例。 $y$  实际上表达了正例的可能性，几率  $\frac{y}{1-y}$  则反映了正例可能性与反例可能性之比值，阈值设置为 0.5 恰表明分类器认为真实正、反例可能性相同，即分类器决策规则为

$$\text{若 } \frac{y}{1-y} > 1 \text{ 则 预测为正例.} \quad (3.46)$$

无偏采样意味着真实样本总体的类别比例在训练集中得以保持。

然而, 当训练集中正、反例的数目不同时, 令  $m^+$  表示正例数目,  $m^-$  表示反例数目, 则观测几率是  $\frac{m^+}{m^-}$ , 由于我们通常假设训练集是真实样本总体的无偏采样, 因此观测几率就代表了真实几率。于是, 只要分类器的预测几率高于观测几率就应判定为正例, 即

$$\text{若 } \frac{y}{1-y} > \frac{m^+}{m^-} \text{ 则 预测为正例.} \quad (3.47)$$

但是, 我们的分类器是基于式(3.46)进行决策, 因此, 需对其预测值进行调整, 使其在基于式(3.46)决策时, 实际是在执行式(3.47)。要做到这一点很容易, 只需令

$$\frac{y'}{1-y'} = \frac{y}{1-y} \times \frac{m^-}{m^+}. \quad (3.48)$$

实际希望比较

$$\frac{y}{1-y} \times \frac{m^-}{m^+} \text{ 和 } 1 \quad (16)$$

$y'$  是否有用 或者有含义呢

由于算法本身的实现方式不变 (使用的分类器基于类别平衡分类决策规则), 只能对预测值做变换

即将(3.48)得到的  $y' = \frac{m^- y}{m^+(1-y) + m^- y}$  作为最终预测值

特别的, 将(3.48)中的  $\frac{m^-}{m^+}$  换成  $\frac{\text{cost}^+}{\text{cost}^-}$  (正例误分) / (反例误分) 即是代价敏感学习

评价:

- 欠采样的时间开销远小于过采样

## 4. 决策树

### 4.1. 基本流程

叶结点——决策结果

根结点——样本全集

其他结点——属性测试

根结点到叶结点的路径——一个判定测试序列

决策树学习目的——产生一棵泛化能力强的决策树

决策树桩: 只有一层划分的决策树

---

**输入:** 训练集  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ ;  
 属性集  $A = \{a_1, a_2, \dots, a_d\}$ .

**过程:** 函数 TreeGenerate( $D, A$ )

- 1: 生成结点 node;
- 2: if  $D$  中样本全属于同一类别  $C$  then
- 3: 将 node 标记为  $C$  类叶结点; return
- 4: end if
- 5: if  $A = \emptyset$  OR  $D$  中样本在  $A$  上取值相同 then
- 6: 将 node 标记为叶结点, 其类别标记为  $D$  中样本数最多的类; return
- 7: end if
- 8: 从  $A$  中选择最优划分属性  $a_*$ ;
- 9: for  $a_*$  的每一个值  $a_*^v$  do
- 10: 为 node 生成一个分支; 令  $D_v$  表示  $D$  中在  $a_*$  上取值为  $a_*^v$  的样本子集;
- 11: if  $D_v$  为空 then
- 12: 将分支结点标记为叶结点, 其类别标记为  $D$  中样本最多的类; return
- 13: else
- 14: 以 TreeGenerate( $D_v, A \setminus \{a_*\}$ ) 为分支结点
- 15: end if
- 16: end for

**输出:** 以 node 为根结点的一棵决策树

---

图 4.2 决策树学习基本算法

- l2-l3: 第一个return——某个子集只含一类样本
- l14: 对于离散属性, 每个属性不重复使用; 划分次数通常不超过属性个数
- l5: 所有属性均作为划分依据之后,  $A=\emptyset$ , 此时无法再划分剩下样本; 或者剩下的样本不属于同一类, 但在  $A$  中剩下属性上表现相同, 也无法划分剩下样本  
——> 少数服从多数, 不唯一时任选一个 (利用当前结点的后验分布)
- l11-l12: 若某个属性对应子分支为空 (可能因为没有收集到相应的样本), 采用盲猜的做法, 将对应叶结点标记为  $D$  中数量最多的类别 (全体样本的分布作为当前结点的先验分布)

## 4.2. 划分选择

如何选择最优划分属性

——随着划分进行, 希望决策树的分支所包含的样本尽可能属于同一类

### 4.2.1. 信息增益

信息熵: 度量样本集合纯度的常用指标

假定当前样本集合  $D$  中第  $k$  类样本所占的比例为  $p_k$  ( $k = 1, 2, \dots, |\mathcal{Y}|$ ), 则  $D$  的信息熵定义为

计算信息熵时约定: 若  $p = 0$ , 则  $p \log_2 p = 0$ .

$$\text{Ent}(D) = - \sum_{k=1}^{|\mathcal{Y}|} p_k \log_2 p_k . \quad (4.1)$$

$\text{Ent}(D)$  的最小值为 0,  $\text{Ent}(D)$  的值越小, 则  $D$  的纯度越高.  
最大值为  $\log_2 |\mathcal{Y}|$ .

-log: 凸函数

$$\text{若 } f \text{ 为凸函数, 则 } \sum_i \alpha_i f(x_i) \leq f(\sum_i \alpha_i x_i)$$

- 最大值

若 $f$ 为凸函数，则 $\sum_i \alpha_i f(x_i) \leq f(\sum_i \alpha_i x_i)$

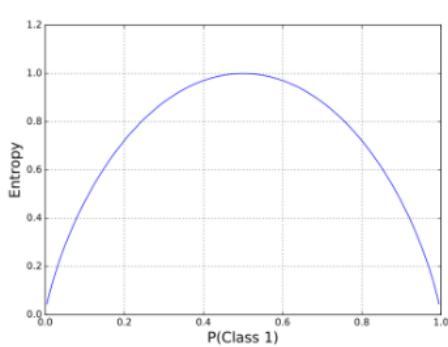
$$\text{易证: } Ent(D) = \sum_k p_k \log_2 \frac{1}{p_k} \leq \log_2 \sum_k p_k \frac{1}{p_k} = \log_2 |\mathcal{Y}|$$

均匀分布的熵最大

计算信息熵时约定：若 $p = 0$ ，则 $p \log p = 0$

$$f\left(\frac{1}{n}, \dots, \frac{1}{n}\right) = -\sum_{k=1}^n \frac{1}{n} \log_2 \frac{1}{n} = -n \cdot \frac{1}{n} \log_2 \frac{1}{n} = \log_2 n$$

各样本均分，样本纯度最低



二分类：

$$Ent(D) = -p_1 \log p_1 - (1-p_1) \log(1-p_1)$$

$p_1 = 0.5$ 取最大

- 最小值：0

$$x_k = 1, x_1 = x_2 = \dots = x_{k-1} = x_{k+1} = \dots = x_n = 0$$

所有样本属于同一类，样本纯度最高

### 信息增益

- 信息增益是变量间相互依赖性的度量，是联合分布和边缘分布乘积的相似程度度量

$$\begin{aligned} I(X, Y) &= \sum_y \sum_x P(X=x, Y=y) \log \frac{P(X=x, Y=y)}{P(X=x)P(Y=y)} \\ &= H(Y) - \sum_x P(X=x)H(Y|X=x) \\ &= H(Y) - H(Y|X) \\ &= H(X) - H(X|Y) \end{aligned}$$

衡量变量之间的互信息 $I(X, Y)$ ， $H(Y) = Ent(Y)$

互信息描述的是：给定变量 $Y$ 情况下，导致变量 $X$ 的信息熵的下降值

- 若 $X$ 、 $Y$ 相互独立， $\log$ 项为0，互信息为0
- 否则互信息>0

由此定义信息增益：

$$\text{Gain}(D, a) = \text{Ent}(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} \text{Ent}(D^v). \quad (4.2)$$

解释：

离散属性  $a$  有  $V$  个可能取值  $\{a^i, i = 1, 2, \dots, V\}$

用  $a$  对样本集  $D$  进行划分，产生  $V$  个分支结点  $\{D^i, i = 1, 2, \dots, V\}$

(4.2) 中  $\sum_{v=1}^V \frac{|D^v|}{|D|} \text{Ent}(D^v)$  部分是各个分结点子集信息熵（条件熵）的加权平均

$\text{Ent}(D^V)$  越小，子结点纯度越高， $\text{Gain}(D^V)$  越大 —> 使用属性  $a$  进行划分获得的“纯度提升”越大

I8 选择最优划分属性  $a_* = \arg \max_{a \in A} \text{Gain}(D, a)$ .

特别的，若给每个样本一个不同的编号，以编号作为划分属性，得到信息增益接近于 1 — 每个编号下只有一个样本，纯度已达最大，但这样的划分不具有泛化能力

—> 信息增益准则对可取值数目较多的属性有所偏好

#### 4.2.2. 增益率

C4.5 决策树算法 减少上述偏好的不利影响

使用“增益率(Gain ratio)”而不是“信息增益”选择最优划分属性

增益率

$$\text{Gain\_ratio}(D, a) = \frac{\text{Gain}(D, a)}{\text{IV}(a)}, \quad (4.3)$$

其中

$$\text{IV}(a) = - \sum_{v=1}^V \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|} \quad (4.4)$$

称为属性  $a$  的“固有值”(intrinsic value) [Quinlan, 1993]. 属性  $a$  的可能取值数目越多(即  $V$  越大)，则  $\text{IV}(a)$  的值通常会越大. 例如，对表 4.1 的西瓜数据集 2.0，有  $\text{IV}(\text{触感}) = 0.874 (V = 2)$ ,  $\text{IV}(\text{色泽}) = 1.580 (V = 3)$ ,  $\text{IV}(\text{编号}) = 4.088 (V = 17)$ .

增益率准则对可能取值数目较少的属性有所偏好

C4.5 的做法 — 启发式：

1. 从候选划分属性中找出信息增益高于平均水平的属性
2. 进一步选出增益率最高的

#### 4.2.3. 基尼指数

CART 决策树 使用基尼指数选择划分属性

##### 4.2.3.1. 数据集的基尼集

基尼值度量数据集纯度

$$\begin{aligned}
 \text{Gini}(D) &= \sum_{k=1}^{|Y|} \sum_{k' \neq k} p_k p_{k'} \\
 &= 1 - \sum_{k=1}^{|Y|} p_k^2. \tag{4.5}
 \end{aligned}$$

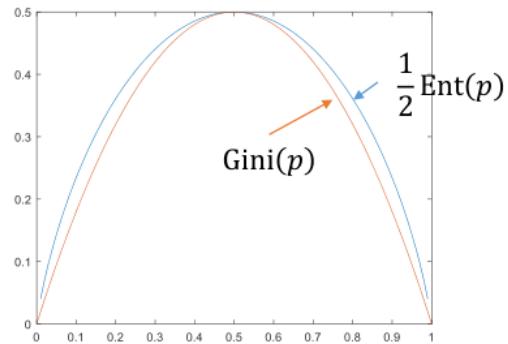
直观来说,  $\text{Gini}(D)$  反映了从数据集  $D$  中随机抽取两个样本, 其类别标记不一致的概率. 因此,  $\text{Gini}(D)$  越小, 则数据集  $D$  的纯度越高.

### 基尼值和Ent的关系

- 数据集  $D$  的纯度可用 “基尼值” 来度量  $\log x \approx x - 1$

$$\text{Gini}(D) = \sum_{k=1}^{|Y|} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{|Y|} p_k^2 = \sum_{k=1}^{|Y|} p_k (1 - p_k) \approx - \sum_{k=1}^{|Y|} p_k \log p_k$$

- 反映了从  $D$  中随机抽取两个样本, 其类别标记不一致的概率
- $\text{Gini}(D)$  越小, 数据集  $D$  的纯度越高



#### 4.2.3.2. 属性 $a$ 的基尼指数

基于上述关系和

$$\text{Gain}(D, a) = \text{Ent}(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} \text{Ent}(D^v). \tag{4.2}$$

( $\text{Ent}(D)$  和数据集的属性划分没有关系, 是为了调整单调性 (?) )

给出基尼指数的定义:

$$\text{Gini\_index}(D, a) = \sum_{v=1}^V \frac{|D^v|}{|D|} \text{Gini}(D^v). \tag{4.6}$$

选择最优化划分属性  $a_* = \arg \min_{a \in A} \text{Gini\_index}(D, a).$

### 4.3. 剪枝处理

-- 应对过拟合的主要手段

### 4.3.1. 判断泛化性能是否提升

见**2.2 性能评估**

本节：留出法

判断依据：划分在验证集上的精度

### 4.3.2. 预剪枝

决策树生成过程中，每个结点划分前判断：

- 若划分不能提高决策树的泛化性能，不再划分，当前结点标记为叶结点，少数服从多数

优点

- 降低过拟合风险
- 减少决策树训练时间开销和测试时间开销

缺点

- 带来欠拟合风险：基于贪心本质禁止泛化性能没有提高的分支展开，但基于这些分支的后续划分可能带来性能提高

### 4.3.3. 后剪枝

生成完整决策树后，自底向上考察非叶结点

- 若将非叶结点的子树替换为叶结点能提高泛化性能，则替换

然后考察结点⑤，若将其领衔的子树替换为叶结点，则替换后的叶结点包含编号为 {6, 7, 15} 的训练样例，叶结点类别标记为“好瓜”，此时决策树验证集精度仍为 57.1%。于是，可以不进行剪枝。

此种情形下验证集精度虽无提高，但根据奥卡姆剃刀准则，剪枝后的模型更好。因此，实际的决策树算法在此种情形下通常要进行剪枝。本书为绘图的方便，采取了不剪枝的保守策略。

后剪枝策略通常比预剪枝保留更多的分支

优点

- 欠拟合风险小
- 泛化性能往往优于预剪枝决策树

缺点

- 训练时间开销大于未剪枝和预剪枝

## 4.4. 连续与缺失值

### 4.4.1. 连续值处理

连续属性离散化

#### C4.5采用二分法

连续属性  $a$ ，样本集  $D$ ， $a$  在  $D$  上只会出现有限个取值，设为  $n$

从小到大排列并选取划分点  $t$ ，将  $D$  分为  $D_t^+$  和  $D_t^-$

## 选取划分点†

包含  $n - 1$  个元素的候选划分点集合

取相邻属性值的中点构成

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} \mid 1 \leq i \leq n - 1 \right\},$$

类比离散情形，有

$$\begin{aligned} \text{Gain}(D, a) &= \max_{t \in T_a} \text{Gain}(D, a, t) \\ &= \max_{t \in T_a} \text{Ent}(D) - \sum_{\lambda \in \{-, +\}} \frac{|D_t^\lambda|}{|D|} \text{Ent}(D_t^\lambda), \end{aligned} \quad (4.8)$$

其中  $\text{Gain}(D, a, t)$  是样本集  $D$  基于划分点  $t$  二分后的信息增益。于是，我们就可选择使  $\text{Gain}(D, a, t)$  最大化的划分点。

实际操作中可将  $D$  中出现的不大于(4.8)选出的最大属性值作为最终的划分值，使得最终决策树使用的划分点在训练集中都出现过

即：相比于离散属性，连续属性再得到信息增益的同时需要求出划分点

离散属性可以重复作为划分依据

### 4.4.2. 缺失值处理

随机缺失/非随机缺失

#### 4.4.2.1. 选择划分属性

样本集  $D$ ,  $\tilde{D}$  为  $D$  在属性  $a$  上没有缺失值的子集

对  $\tilde{D}$  进行两种划分

- 按在属性  $a$  上的取值： $\tilde{D}^v$  为  $\tilde{D}$  在属性  $a$  上取值为  $a^v$  的样本子集

$$\tilde{D} = \bigcup_{v=1}^V \tilde{D}^v$$

- 按样本最终分类： $\tilde{D}^k$ ,  $k = 1, 2, \dots, |\mathcal{Y}|$

$$\tilde{D} = \bigcup_{k=1}^{|\mathcal{Y}|} \tilde{D}_k$$

假定我们为每个样本  $\mathbf{x}$  赋予一个权重  $w_{\mathbf{x}}$ ,

决策树学习开始阶段，根结点各样本权值初始化为1

$$\rho = \frac{\sum_{\mathbf{x} \in \tilde{D}} w_{\mathbf{x}}}{\sum_{\mathbf{x} \in D} w_{\mathbf{x}}}, \quad (4.9)$$

-- 属性  $a$  无缺失值样本所占比例

$$\tilde{p}_k = \frac{\sum_{\mathbf{x} \in \tilde{D}_k} w_{\mathbf{x}}}{\sum_{\mathbf{x} \in \tilde{D}} w_{\mathbf{x}}} \quad (1 \leq k \leq |\mathcal{Y}|), \quad (4.10)$$

$$\sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k = 1$$

-- 无缺失值样本中第  $k$  类所占比例

$$\tilde{r}_v = \frac{\sum_{x \in \tilde{D}^v} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq v \leq V). \quad (4.11)$$

$$\sum_{v=1}^V \tilde{r}_v = 1.$$

——无缺失值中属性 $a$ 取值为 $a^v$ 的样本所占比例

将无缺失值下的信息增益推广为

$$\begin{aligned} \text{Gain}(D, a) &= \rho \times \text{Gain}(\tilde{D}, a) \\ &= \rho \times \left( \text{Ent}(\tilde{D}) - \sum_{v=1}^V \tilde{r}_v \text{Ent}(\tilde{D}^v) \right), \end{aligned} \quad (4.12)$$

其中

$$\text{Ent}(\tilde{D}) = - \sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k \log_2 \tilde{p}_k.$$

$\rho$ 使得无缺失集上的信息增益减小为一定比例——结合下面的划分方式可以理解 划分对属性值缺失的样本没有贡献（存在于各个子类中，仍导致不纯）

#### 4.4.2.2. 划分含缺失值样本

- 样本属性值已知：划分到对应子结点，权值保持
- 样本属性值缺失：将样本划入所有子结点，但权重不同

在属性值为 $a_v$ 的结点中权值变为 $\tilde{r}_v \cdot w_x$

即：让样本以不同概率划入不同子结点，“概率”取各属性值在 $\tilde{D}$ 中的出现频率

### 4.5. 回归树

- 叶子结点标记——均值；中位数
- 优化目标——如：均方误差
  - 划分前误差

求 $c$ , s.t.

$$\min_c \sum_i (y_i - c)^2 \rightarrow c = \text{avg}(\{y | (\mathbf{x}, y) \in D\})$$

◦ 划分后误差

$$\text{error}(D, a) = \sum_{v=1}^V \min_{c^v} \sum_{(\mathbf{x}_i, y_i) \in D^v} (y_i - c^v)^2 \rightarrow c^v = \text{avg}(\{y | (\mathbf{x}, y) \in D^v\})$$

理解：划分前 $D$ 只有一个 $c$ 计算整个均方误差，划分后每个子集 $D^v$ 有自己的 $c^v$ 分别计算均方误差

若平方换成绝对值—— $c$ =中位数；求中位数时间复杂度： $O(n)$

- 最优划分属性

$$a^* = \arg \min_{a \in A} \text{error}(D, a)$$

## 4.6. 多变量决策树

属性——坐标轴

d个属性描述——d维坐标空间中的一个数据点

样本分类——在坐标空间中寻找不同类样本之间的分类边界

决策树形成的分类边界特点：轴平行（每段与坐标轴平行）

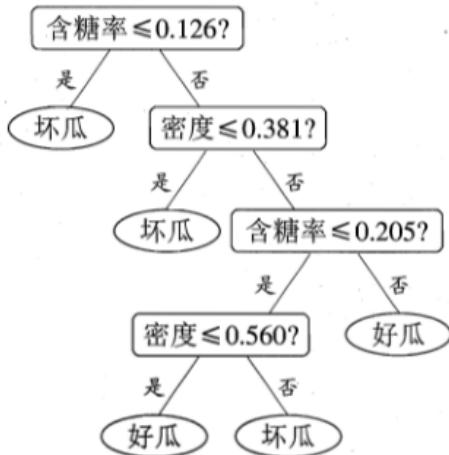


图 4.10 在西瓜数据集 3.0 $\alpha$  上生成的决策树

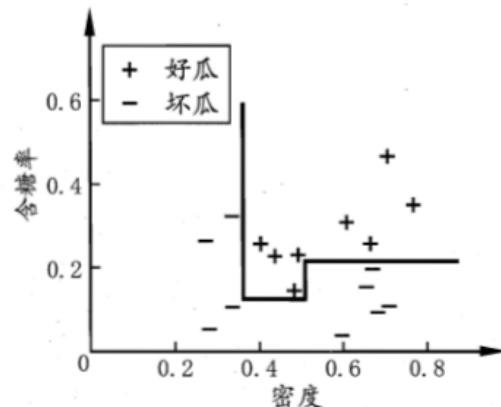


图 4.11 图 4.10 决策树对应的分类边界

边界复杂时预测时间开销大

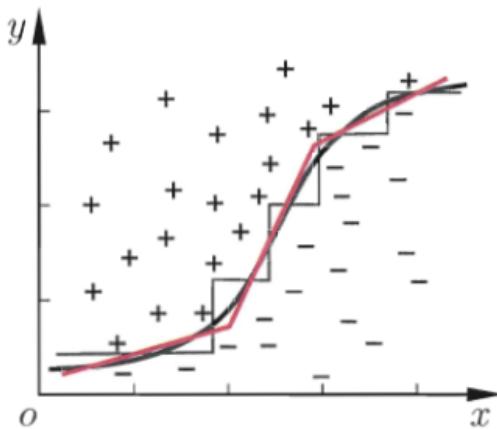


图 4.12 决策树对复杂分类边界的分段近似

多变量决策树——实现斜划分

非叶结点变为对属性的线性组合进行测试

每个非叶结点是一个形如  $\sum_{i=1}^d w_i a_i = t$  的线性分类器

权重  $w_i$  和  $t$  可在该结点所含的样本集和属性集上学习得到

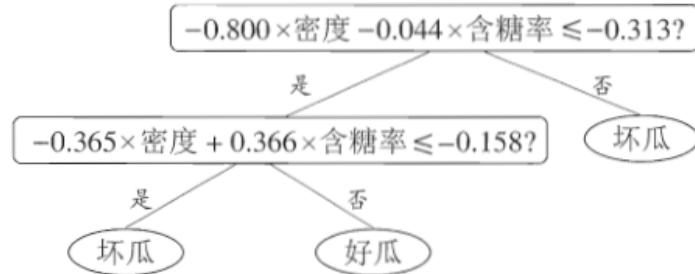


图 4.13 在西瓜数据集 3.0 $\alpha$  上生成的多变量决策树

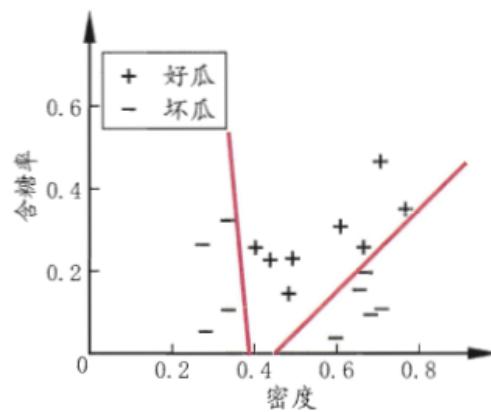


图 4.14 图 4.13 多变量决策树对应的分类边界

离散属性 决策树复杂度

## 5. 神经网络

### 5.1. 神经元 (neuron) 模型

**神经网络：**由具有适应性的简单单元组成的广泛并行互连的网络，它的组织能够模拟生物神经系统对真实世界物体所做出的交互反应

#### 5.1.1. M-P神经元模型

1. 输入：神经元收到n个其他神经元传递的带权重输入信号
2. 处理：将总输入与神经元的阈值比较
3. 输出：通过激活函数处理产生输出

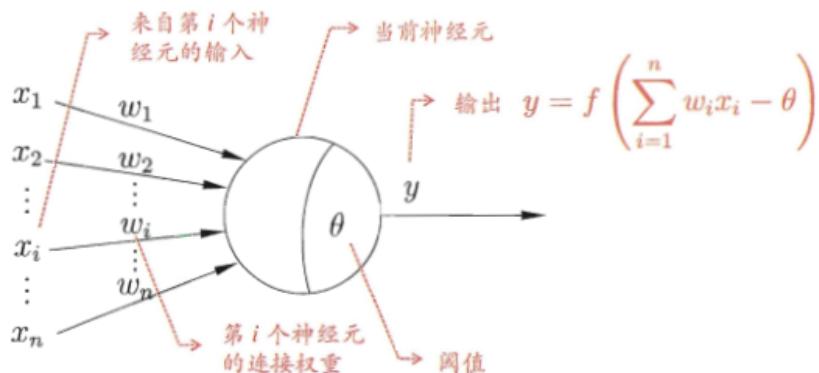


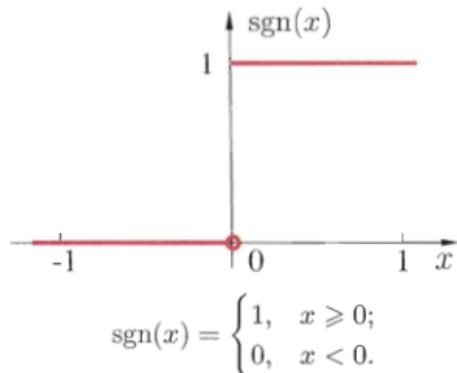
图 5.1 M-P 神经元模型

### 5.1.2. 激活函数

#### 5.1.2.1. 阶跃函数

一种理想的激活函数

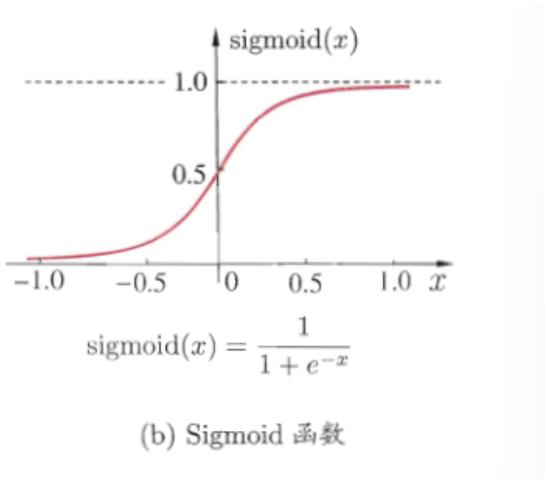
- 将输入值映射为0或1
- 1——神经元兴奋
- 0——神经元抑制



(a) 阶跃函数

缺点：不连续、不光滑， 不适应梯度下降法

#### 5.1.2.2. Sigmoid函数/挤压函数



(b) Sigmoid 函数

许多神经元按一定的层次结构连接起来——神经网络

将神经网络看作一个数学模型：

由若干  $y_j = f(\sum_i w_i x_i - \theta_j)$  函数相互嵌套得到。

### 5.1.2.3. softmax

见BP算法-多分类问题下

深度学习

### 5.1.2.4. tanh

### 5.1.2.5. ReLU

$$f(x) = \max(0, x) \quad (17)$$

支持稀疏表示，只有一部分神经元被激活

### 5.1.2.6. Leaky ReLU

### 5.1.2.7. Maxout

### 5.1.2.8. ELU

## 5.2. 感知机与多层网络

### 5.2.1. 感知机

#### 5.2.1.1. 定义

由两层神经元组成，只有一层功能神经元

- 输入层：接收外界输入信号，传递给输出层
- 输出层：M-P神经元（阈值逻辑单元）

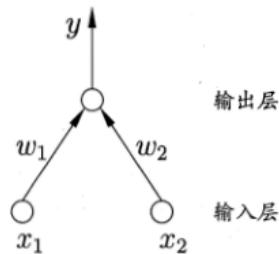


图 5.3 两个输入神经元的感知机网络结构示意图

#### 5.2.1.2. 实现逻辑或、与、非

取  $y = f(\sum_i w_i x_i - \theta)$  中  $f$  为阶跃函数

- 与

$$y = f(1 \cdot x_1 + 1 \cdot x_2 - 2) \quad (18)$$

- 或

$$y = f(1 \cdot x_1 + 1 \cdot x_2 - 0.5) \quad (19)$$

- 非

$$y = f(-0.6 \cdot x_1 + 0 \cdot x_2 + 0.5) \quad (20)$$

### 5.2.1.3. 参数更新

利用经验的方法更新参数（而不是梯度下降）

- 参数统一

将阈值 $\theta$ 看作一个固定输入为 $-1.0$ 的“哑结点”，对应连接权重 $w_{n+1}$ 是真正的阈值

- 学习规则

$$w_i \leftarrow w_i + \Delta w_i, \quad (5.1)$$

$$\Delta w_i = \eta(y - \hat{y})x_i, \quad (5.2)$$

$\eta \in (0, 1)$ : 学习率

- 学习目标：减小 $y$ 和 $\hat{y}$ 差值
- 预测正确时，感知机不变

### 5.2.1.4. 学习能力

只有输出层神经元进行激活函数处理（只有一层功能神经元），学习能力有限。

若两类模式线性可分（可以用线性超平面划分），则感知机学习过程一定收敛；否则将发生震荡， $w$ 难以稳定（e.g. 异或）。

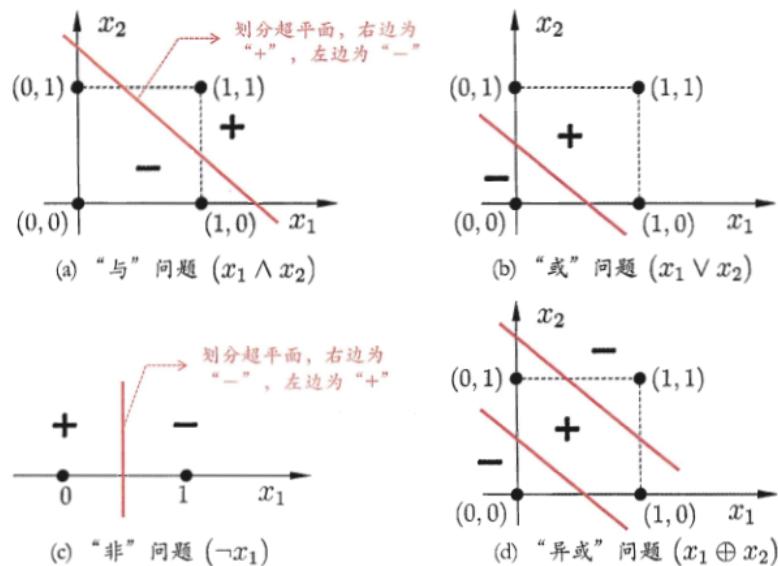


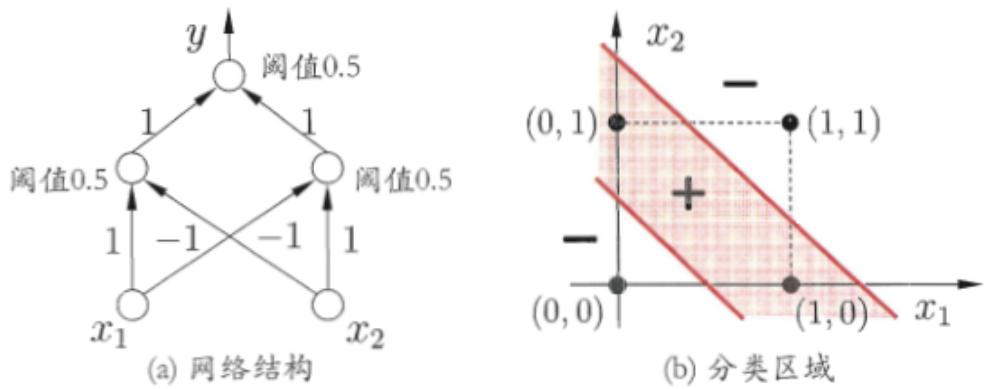
图 5.4 线性可分的“与”“或”“非”问题与非线性可分的“异或”问题

### 5.2.2. 多层功能神经元

—为了解决非线性可分问题

- 输入层
- 隐含层（隐层）
- 输出层

输出层和隐含层神经元都是有激活函数的功能神经元



### 5.2.2.1. 多层前馈神经网络

multi-layer feedforward neural networks

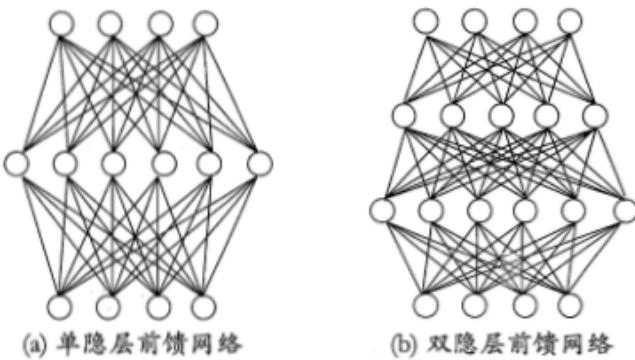


图 5.6 多层前馈神经网络结构示意图

**定义：**每层神经元与下一层神经元全互联；神经元之间不存在同层连接和跨层连接

**前馈：**网络拓扑结构上不存在环或回路

**工作方式：**输入层只接收外界输入，不进行函数处理；隐含层和输出层神经元对信号加工

**学习：**根据训练数据调整神经元之间的**连接权**和每个功能神经元的**阈值**

**表示能力：**

若多层前馈神经网络的隐层有

- 足够多（隐藏）神经元
- 任意“挤压”性质的激活函数

则多层前馈神经网络可以以任意精度逼近**有界闭集**上的任意**连续函数**

## 5.3. 误差逆传播算法（BP算法）

最成功的训练神经网络算法

### 5.3.1. 模型建立

- 离散属性需要先进行处理：若属性值间存在“序”关系，则可以进行连续化；否则转化为 $k$ 维向量， $k$ 为属性值数

训练集  $D = \{(\vec{x}_1, \vec{y}_1), (\vec{x}_2, \vec{y}_2), \dots, (\vec{x}_m, \vec{y}_m)\}$ ,  $\vec{x}_i \in \mathbb{R}^d$  ( $d$ 维特征向量),  $\vec{y}_d \in \mathbb{R}^l$  ( $l$ 维输出向量)

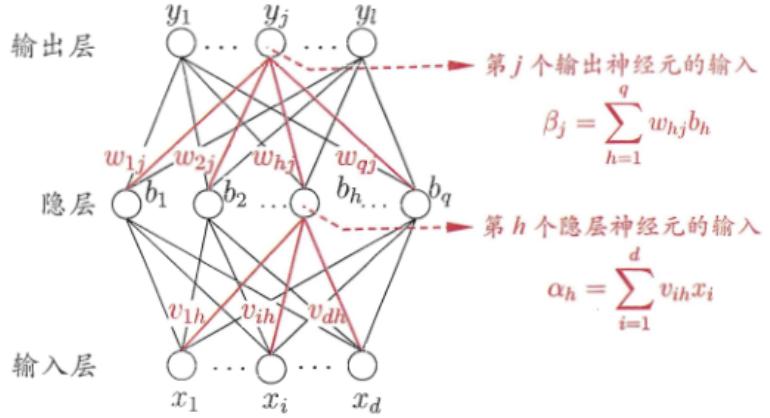


图 5.7 BP 网络及算法中的变量符号

隐层第  $h$  个神经元阈值:  $\gamma_h$

输出层第  $j$  个神经元阈值:  $\theta_j$

$$\begin{aligned}
 \vec{x} &\xrightarrow{\text{加权和}} \text{隐层神经元输入 } \alpha_i && \xrightarrow{\text{隐层激活函数}} \text{隐层输出 } b_i \\
 &\xrightarrow{\text{加权和}} \text{输出层神经元输入 } \beta_j && \xrightarrow{\text{输出层激活函数}} \text{输出 } y_j, \vec{y}
 \end{aligned} \tag{21}$$

共  $(d + l + 1)q + l$  个参数需要确定

### 5.3.2. BP 算法

激活函数均选择 *Sigmoid* 函数, 误差函数选择均方误差

对训练例  $(\mathbf{x}_k, \mathbf{y}_k)$ , 假定神经网络的输出为  $\hat{\mathbf{y}}_k = (\hat{y}_1^k, \hat{y}_2^k, \dots, \hat{y}_l^k)$ , 即

$$\hat{y}_j^k = f(\beta_j - \theta_j), \tag{5.3}$$

则网络在  $(\mathbf{x}_k, \mathbf{y}_k)$  上的均方误差为

$$E_k = \frac{1}{2} \sum_{j=1}^l (\hat{y}_j^k - y_j^k)^2. \tag{5.4}$$

$$\begin{aligned}
 \frac{\partial E_K}{\partial \hat{y}_j^k} &= \hat{y}_j^k - y_j^k \\
 \frac{\partial \hat{y}_j^k}{\partial \beta_j} &= f'(\beta_j - \theta_j)
 \end{aligned} \tag{22}$$

参数更新规则:

$$\mathbf{v} \leftarrow \mathbf{v} + \Delta \mathbf{v}. \tag{5.5}$$

更新方向: 目标的负梯度方向

### 5.3.2.1. $\Delta w_{hj}$

$$\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{hj}} . \quad (5.6)$$

求更新公式：

$\Delta w_{hj}$ 先影响到第  $j$  个输出神经元的输入值  $\beta_j$ , 再影响到其输出值  $\hat{y}_j^k$ , 继而影响  $E_K$

$$\frac{\partial E_k}{\partial w_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial w_{hj}} . \quad (5.7)$$

(链式法则)

根据  $\beta_j$  的定义, 显然有

$$\frac{\partial \beta_j}{\partial w_{hj}} = b_h . \quad (5.8)$$

Sigmoid函数性质：

$$f'(x) = f(x)(1 - f(x)) , \quad (5.9)$$

由公式(18)

$$\begin{aligned} g_j &= -\frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \\ &= -(\hat{y}_j^k - y_j^k)f'(\beta_j - \theta_j) \\ &= \hat{y}_j^k(1 - \hat{y}_j^k)(y_j^k - \hat{y}_j^k) . \end{aligned} \quad (5.10)$$

$\Rightarrow$

$w_{hj}$  的更新公式

$$\Delta w_{hj} = \eta g_j b_h . \quad (5.11)$$

### 5.3.2.2. $\Delta \theta_j$

$$\Delta \theta_j = -\eta \frac{\partial E_k}{\partial \theta_j}$$

又

$$\begin{aligned}
 \frac{\partial E_k}{\partial \theta_j} &= \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \theta_j} \\
 &= \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial [f(\beta_j - \theta_j)]}{\partial \theta_j} \\
 &= \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot f'(\beta_j - \theta_j) \times (-1) \\
 &= \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot f(\beta_j - \theta_j) \times [1 - f(\beta_j - \theta_j)] \times (-1) \\
 &= \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \hat{y}_j^k (1 - \hat{y}_j^k) \times (-1) \\
 &= \frac{\partial}{\partial \hat{y}_j^k} \left[ \frac{1}{2} \sum_{j=1}^l (\hat{y}_j^k - y_j^k)^2 \right] \cdot \hat{y}_j^k (1 - \hat{y}_j^k) \times (-1) \\
 &= \frac{1}{2} \times 2(\hat{y}_j^k - y_j^k) \times 1 \cdot \hat{y}_j^k (1 - \hat{y}_j^k) \times (-1) \\
 &= (y_j^k - \hat{y}_j^k) \hat{y}_j^k (1 - \hat{y}_j^k) \\
 &= g_j
 \end{aligned}$$

所以

$$\Delta \theta_j = -\eta \frac{\partial E_k}{\partial \theta_j} = -\eta g_j$$

### 5.3.2.3. $\Delta v_{ih}$

$$\Delta v_{ih} = -\eta \frac{\partial E_k}{\partial v_{ih}}$$

又



$$\begin{aligned}
 \frac{\partial E_k}{\partial v_{ih}} &= \sum_{j=1}^l \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} \cdot \frac{\partial \alpha_h}{\partial v_{ih}} \\
 &= \sum_{j=1}^l \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} \cdot x_i \\
 &= \sum_{j=1}^l \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} \cdot f'(\alpha_h - \gamma_h) \cdot x_i \\
 &= \sum_{j=1}^l \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot w_{hj} \cdot f'(\alpha_h - \gamma_h) \cdot x_i \\
 &= \sum_{j=1}^l (-g_j) \cdot w_{hj} \cdot f'(\alpha_h - \gamma_h) \cdot x_i \\
 &= -f'(\alpha_h - \gamma_h) \cdot \sum_{j=1}^l g_j \cdot w_{hj} \cdot x_i \\
 &= -b_h(1 - b_h) \cdot \sum_{j=1}^l g_j \cdot w_{hj} \cdot x_i \\
 &= -e_h \cdot x_i
 \end{aligned}$$

所以

$$\Delta v_{ih} = -\eta \frac{\partial E_k}{\partial v_{ih}} = \eta e_h x_i$$

按照箭头方向思考链式法则的式子

**求和符号：**  $b_h$  通过影响每个  $\beta_j$  最终影响  $E_k$

$$\begin{aligned}
e_h &= -\frac{\partial E_k}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} \\
&= -\sum_{j=1}^l \frac{\partial E_k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} f'(\alpha_h - \gamma_h) \\
&= \sum_{j=1}^l w_{hj} g_j f'(\alpha_h - \gamma_h) \\
&= b_h(1 - b_h) \sum_{j=1}^l w_{hj} g_j . \tag{5.15}
\end{aligned}$$

#### 5.3.2.4. $\Delta \gamma_h$

因为

$$\Delta \gamma_h = -\eta \frac{\partial E_k}{\partial \gamma_h}$$

又

$$\begin{aligned}
\frac{\partial E_k}{\partial \gamma_h} &= \sum_{j=1}^l \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} \cdot \frac{\partial b_h}{\partial \gamma_h} \\
&= \sum_{j=1}^l \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} \cdot f'(\alpha_h - \gamma_h) \cdot (-1) \\
&= -\sum_{j=1}^l \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot w_{hj} \cdot f'(\alpha_h - \gamma_h) \\
&= -\sum_{j=1}^l \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot w_{hj} \cdot b_h(1 - b_h) \\
&= \sum_{j=1}^l g_j \cdot w_{hj} \cdot b_h(1 - b_h) \\
&= e_h
\end{aligned}$$

所以

$$\Delta \gamma_h = -\eta \frac{\partial E_k}{\partial \gamma_h} = -\eta e_h$$

#### 5.3.2.5. 参数更新总结

$w_{hj}$  的更新公式

$$\Delta w_{hj} = \eta g_j b_h . \tag{5.11}$$

类似可得

$$\Delta \theta_j = -\eta g_j , \tag{5.12}$$

$$\Delta v_{ih} = \eta e_h x_i , \tag{5.13}$$

$$\Delta \gamma_h = -\eta e_h , \tag{5.14}$$

输出层参数更新用到：

- $g_j = \hat{y}(1 - \hat{y})(y - \hat{y})$ ——只和输出层误差有关

## 输出层神经元梯度项

- $b_h$ ——前一层隐层神经元输出

隐层参数更新用到：

- $e_h = b_h(1 - b_h) \sum_{j=1}^l w_{hj}g_j$ ——当前隐层神经元输出、后一层输出层权重 $w_{hj}$ 与输出层误差的函数

## 隐层神经元的梯度项

- 输入 $x_i$

学习率 $\eta$ ：控制更新步长，常设置为0.1

不同层功能神经元可以使用不同的学习率（5.11、5.12 || 5.13、5.14）

### 5.3.2.6. 标准BP算法

1. 样例输入到输入层神经元
2. 产生输出结果
3. 计算输出层误差
4. 误差逆向传播到隐层神经元
5. 调整参数
6. 迭代循环至误差足够小、防止过拟合

---

输入：训练集  $D = \{(\mathbf{x}_k, \mathbf{y}_k)\}_{k=1}^m$ ;  
学习率  $\eta$ .

过程：

- 1: 在 $(0, 1)$ 范围内随机初始化网络中所有连接权和阈值
- 2: **repeat**
- 3:   **for all**  $(\mathbf{x}_k, \mathbf{y}_k) \in D$  **do**
- 4:     根据当前参数和式(5.3) 计算当前样本的输出  $\hat{\mathbf{y}}_k$ ;
- 5:     根据式(5.10) 计算输出层神经元的梯度项  $g_j$ ;
- 6:     根据式(5.15) 计算隐层神经元的梯度项  $e_h$ ;
- 7:     根据式(5.11)-(5.14) 更新连接权  $w_{hj}, v_{ih}$  与阈值  $\theta_j, \gamma_h$
- 8:   **end for**
- 9: **until** 达到停止条件

输出：连接权与阈值确定的多层前馈神经网络

---

图 5.8 误差逆传播算法

但上图中的算法每次只针对一个训练样例更新参数

### 5.3.2.7. 累积误差逆传播算法

#### 训练目标

最小化训练集上的累计误差

$$E = \frac{1}{m} \sum_{k=1}^m E_k , \quad (5.16)$$

#### 对比

- 标准BP：
  - 参数更新频繁
  - 对不同样例的更新结果可能相互抵消
  - 一般需要更多的迭代次数
- 累积BP：

- 读取整个训练集一遍后才对参数更新
- 累积误差下降到一定程度后，进一步下降非常缓慢

### 5.3.3. 过拟合

缓解方法

- 早停

训练集误差降低，验证集误差开始升高时停止训练

- 正则化

在误差目标函数中增加一个用于描述网络复杂度的部分，如：连接权与阈值的平方和。

$$E = \lambda \frac{1}{m} \sum_{k=1}^m E_k + (1 - \lambda) \sum_i w_i^2 , \quad (5.17)$$

### 5.3.4. 多分类问题

#### 5.3.4.1. 多分类损失-交叉熵

期望输出： $\vec{d}$ , one-hot

实际输出： $\vec{y}$

均方误差： $\frac{1}{2} \sum_j (y_j - d_j)^2$

**Cross Entropy** (交叉熵)

熵： $H = - \sum_{i=0}^n p(x_i) \log(p(x_i))$

$p(x_i)$ :  $x_i$  出现概率

KL散度（可以理解为相对熵）：

$$\begin{aligned} D_{KL}(p, q) &= \sum_{i=0}^n p(x_i) \log\left(\frac{p(x_i)}{q(x_i)}\right) \\ &= -H - \sum_{i=0}^n p(x_i) \log(q(x_i)) \end{aligned} \quad (23)$$

样本真实分布已知，真实分布的熵固定不变，只关注后面求和部分，即“交叉熵”

$$Loss = - \sum_j d_j \log y_j \quad (24)$$

实际情况中， $\vec{d}$  只有一位为1，因此，对一个属于 $j$ 类的样本  $Loss = -\ln a_j$

#### 5.3.4.2. softmax

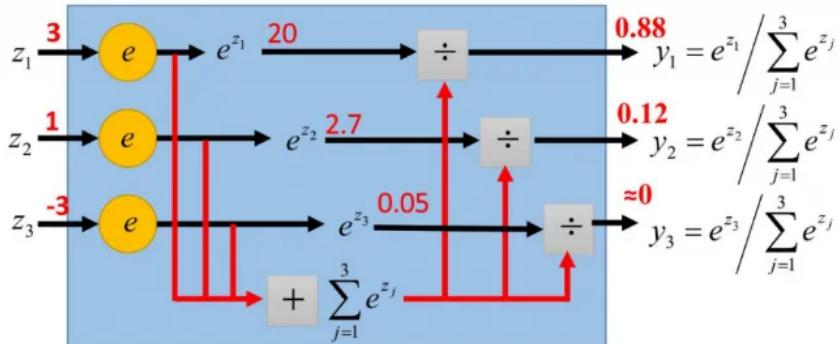
$$f(e_i) = \frac{e^i}{\sum_j e^j} \quad (25)$$

- Softmax layer as the output layer

### Probability:

- $1 > y_i > 0$
- $\sum_i y_i = 1$

### Softmax Layer



softmax得到的输出向量满足：各维度和为1，可以理解为概率

#### 5.3.4.2.1. 溢出问题（作业）

2. 讨论  $\frac{\exp(x_i)}{\sum_{j=1}^c \exp(x_j)}$  和  $\log \frac{c}{\sum_{j=1}^c \exp(x_j)}$  的数值溢出问题.

$$(1) \frac{\exp(x_i)}{\sum_{j=1}^c \exp(x_j)}$$

①  $x_i$  是较大正数时导致分子分母都很大，超出计算机可处理范围，出现数值上溢

② 所有  $x_j$  都是很小的负数时，分母可能产生下溢，出现除数为0的非法情况

解决方法：

$$\begin{aligned} \text{设常数 } a > 0, \frac{\exp(x_i)}{\sum_{j=1}^c \exp(x_j)} &= \frac{a \exp(x_i)}{a \sum_{j=1}^c \exp(x_j)} = \frac{\exp(x_i + \ln a)}{\sum_{j=1}^c \exp(x_j + \ln a)} \\ &= \frac{\exp(x_i + a')}{\sum_{j=1}^c \exp(x_j + a')} \end{aligned}$$

即：一组数同时平移不改变函数结果

取  $a' = -\max_{1 \leq j \leq c}(x_j)$   $\Rightarrow \exp$  指数不超过0，不会出现①③  
且分母中至少有一项为  $e^{\max(x_j) - \max(x_j)} = 1$ ，不会出现②

③ 在①②解决的基础上，若  $x_i - \max(x_j)$  是很小负数，计算结果  $\rightarrow 0$

此时若对结果求对数，也会出错

解决方法：

$$\log \left[ \frac{e^{x_i - M}}{\sum_{j=1}^c e^{x_j - M}} \right] = x_i - M - \log \left[ \sum_{j=1}^c e^{x_j - M} \right], \text{ 其中 } M = \max_{1 \leq j \leq c}(x_j)$$

用右边式子代替直接求  $\log$ ，真数求和部分至少有一项为  $e^{M-M} = 1$ 。

$$(2) \log \sum_{j=1}^c \exp(x_j)$$

① 当  $x_i$  为较大正数时，数值上溢

② 当  $x_i$  均为较小负数时，数值下溢

解决方法：

$$\begin{aligned} \log \left( \sum_{j=1}^c \exp(x_j - M) \right) &= \log \left( \sum_{j=1}^c \frac{\exp(x_j)}{\exp(M)} \right) \\ &= \log \left( \sum_{j=1}^c \exp(x_j) \right) - M \end{aligned}$$

$$\text{故 } \log \sum_{j=1}^c \exp(x_j) = M + \log \left( \sum_{j=1}^c \exp(x_j - M) \right)$$

$$M = \max_{1 \leq j \leq c} \{x_j\}$$

则  $\exp$  的指数部分不超过 0，不会发生 ① 上溢

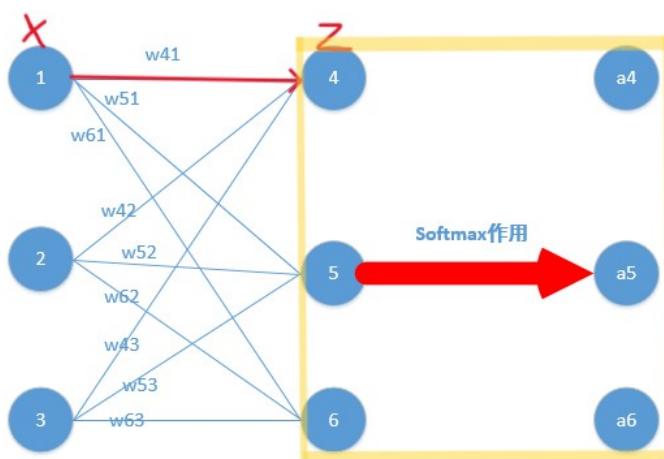
且  $\log$  的真数部分至少有  $-1$   $\exp(M-M)=1$ 。不会发生 ② 下溢。

### 5.3.4.2.2. 求梯度

#### 求导

在梯度下降优化参数时，链式法则第一步需要对 softmax 求导

eg.



注意：这里黄色框内为一层功能神经元

$$a_4 = \frac{e^{z_4}}{z^{z_4} + z^{z_5} + z^{z_6}}$$

$$a_5 = \frac{e^{z_5}}{z^{z_4} + z^{z_5} + z^{z_6}} \quad a_6 = \frac{e^{z_6}}{z^{z_4} + z^{z_5} + z^{z_6}}$$

假设真实输出为  $(1, 0, 0, 0)$ ，则  $Loss = -\ln a_4$

$$\begin{aligned}\frac{\partial Loss}{\partial w_{41}} &= \frac{\partial Loss}{\partial a_4} \frac{\partial a_4}{\partial z_4} \frac{\partial z_4}{\partial w_{41}} \\ &= -\frac{1}{a_4} \frac{\partial a_4}{\partial z_4} x_1\end{aligned}\tag{26}$$

下面求  $\frac{\partial a_j}{\partial z_i}$

①  $j=i$

$$\begin{aligned}\frac{\partial a_j}{\partial z_i} &= \frac{\partial a_i}{\partial z_i} = \frac{\partial}{\partial z_i} \left( \frac{e^{z_i}}{\sum_k e^{z_k}} \right) = \frac{e^{z_i} \sum_k e^{z_k} - (e^{z_i})^2}{(\sum_k e^{z_k})^2} \\ &= \frac{e^{z_i}}{\sum_k e^{z_k}} - \left( \frac{e^{z_i}}{\sum_k e^{z_k}} \right)^2 \\ &= a_i (1 - a_i)\end{aligned}$$

②  $j \neq i$

$$\frac{\partial a_j}{\partial z_i} = \frac{\partial}{\partial z_i} \left( \frac{e^{z_j}}{\sum_k e^{z_k}} \right) = \frac{-e^{z_i} e^{z_j}}{(\sum_k e^{z_k})^2} = -a_i a_j$$

梯度计算（作业）

3.  $\frac{\exp(x_i)}{\sum_{j=1}^c \exp(x_j)}$  和  $\log \frac{\exp(x_i)}{\sum_{j=1}^c \exp(x_j)}$  关于  $\vec{x} = (x_1, \dots, x_c)$  求梯度

$$\begin{array}{ll} f_i & g_i \\ \parallel & \parallel \\ f_i & g_i \end{array}$$

$$\begin{aligned}\frac{\partial f_i}{\partial x_j} : \quad \textcircled{1} \quad i=j \text{ 时}, \quad \frac{\partial f_i}{\partial x_j} &= \frac{\partial f_i}{\partial x_i} = \frac{e^{x_i} \sum_{j=1}^c \exp(x_j) - (e^{x_i})^2}{(\sum_{j=1}^c \exp(x_j))^2} \\ &= \frac{e^{x_i}}{\sum_{j=1}^c \exp(x_j)} - \left[ \frac{e^{x_i}}{\sum_{j=1}^c \exp(x_j)} \right]^2 \\ &= f_i (1 - f_i)\end{aligned}$$

$$\textcircled{2} \quad i \neq j \text{ 时}, \quad \frac{\partial f_i}{\partial x_j} = \frac{-e^{x_j} e^{x_i}}{\left[ \sum_{j=1}^c \exp(x_j) \right]^2} = -f_i f_j$$

$$\Rightarrow \nabla f_i = (-f_{i1}, -f_{i2}, \dots, f_i(1-f_i), \dots, -f_{i(i+1)}, \dots, -f_{ic})$$

↑  
第*i*行

$$\frac{\partial g_i}{\partial x_j} : \begin{aligned} \textcircled{1} \quad i=j &\Rightarrow \frac{\partial g_i}{\partial x_i} = \frac{\partial g_i}{\partial x_i} = \frac{\sum_{j=1}^c \exp(x_j)}{\sum_{j=1}^c \exp(x_j)} \frac{\partial}{\partial x_i} \frac{\exp(x_i)}{\sum_{j=1}^c \exp(x_j)} \\ &= \frac{1}{f_i} \frac{\partial f_i}{\partial x_i} \\ &= 1 - f_i \end{aligned}$$

$$\textcircled{2} \quad i \neq j \Rightarrow \frac{\partial g_i}{\partial x_j} = \frac{\sum_{j=1}^c \exp(x_j)}{\exp(x_i)} \frac{\partial}{\partial x_j} \frac{\exp(x_i)}{\sum_{j=1}^c \exp(x_j)} \\ = \frac{1}{f_i} \frac{\partial f_i}{\partial x_j} = -f_j$$

$$\Rightarrow \nabla g_i = (-f_1, -f_2, \dots, -f_{i-1}, \underset{\uparrow}{1 - f_i}, -f_{i+1}, \dots, -f_c)$$

第*i*行

## 5.4. 全局最小与局部极小

参数寻优方法

- 梯度下降法：沿负梯度方向
- 跳出局部极小：
  - 多组不同参数值初始化多个神经网络，最终取误差最小的
  - 模拟退火技术  
每一步以一定概率接受比当前解更差的结果；每步迭代中，接受“次优解”的概率随着时间的推移而逐渐降低，从而保证算法稳定性
  - 随机梯度下降  
在计算梯度时加入随机因素，使得即便在局部极小点，计算出的梯度仍可能不为0
- 遗传算法

## 5.5. 其他神经网络算法

书

### 5.5.1. RBF网络

### 5.5.2. ART网络

竞争型学习

- 竞争方式：eg. 输入向量与识别层神经元对应模式类的代表向量之间的距离
- 当最大相似度大于识别阈值时，当前输入样本被归为该类；否则新增一个识别层神经元，代表向量为当前输入向量
- 更新网络连接权：使得胜者在面对相似样本时计算出的相似度更大

**性能影响因素：识别阈值**

**优点：**

- 缓解了“可塑性-稳定性窘境”；
- 可以进行增量学习（批量更新）/在线学习（一个样例一更新）  
面对新样例对已训得的模型更新，而不重新训练整个模型

### 5.5.3. SOM网络——自组织映射网络

竞争学习型无监督神经网络

- 高维输入数据映射到低维空间，保持数据在高维空间的拓扑结构——高维中相似的样本点映射到网络输出层中的临近神经元

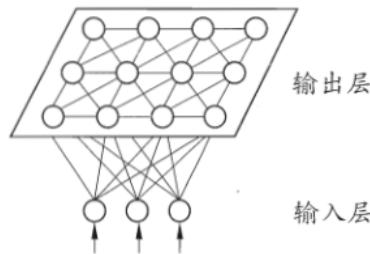


图 5.11 SOM 网络结构

### 5.5.4. 级联相关网络

结构自适应网络

- 级联：建立层次连接的层次结构
- 相关：最大化新神经元的输出与网络误差之间的相关性

训练较快，但数据较小时容易陷入过拟合

#### 5.5.4.1. Elman网络

递归神经网络：允许网络中出现环形结构

可以处理与时间有关的动态变化

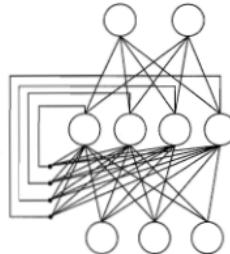


图 5.13 Elman 网络结构

#### 5.5.4.2. Boltzmann机

递归神经网络

基于能量的模型

- 显层：表示数据的输入与输出
- 隐层：数据的内在表达

## 5.6. 深度学习

增加隐层的数目比增加隐层神经元数目更有效，前者增加了激活函数的嵌套层数

特征学习/表示学习

概述

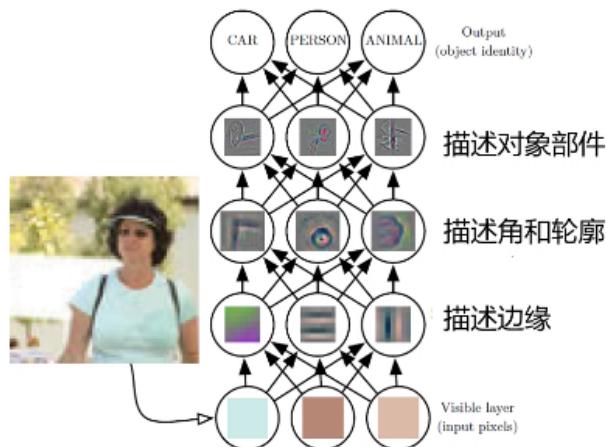
靠近输入的特征 (low level) 中往往没有语义信息，多为线性变换，描述边缘特征

低级特征组合出高级特征

.....

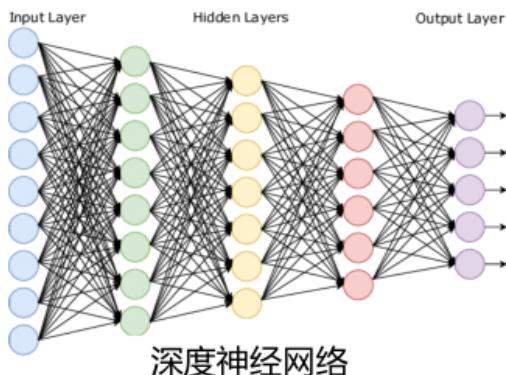
—>样本—坐标空间中的一个点

- 深度学习将大千世界表示为嵌套的层次概念体系
  - 由较简单概念间的联系定义复杂概念
  - 从一般抽象概括到高级抽象表示

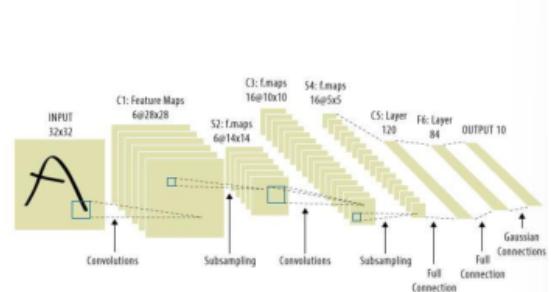


传统cv抽取的特征多为低级特征

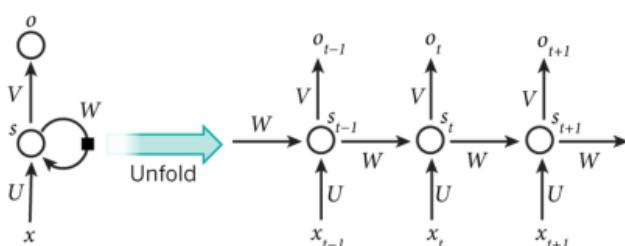
### 5.6.1. 深度神经网络



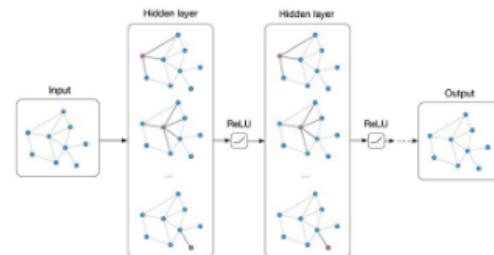
深度神经网络



深度卷积网络

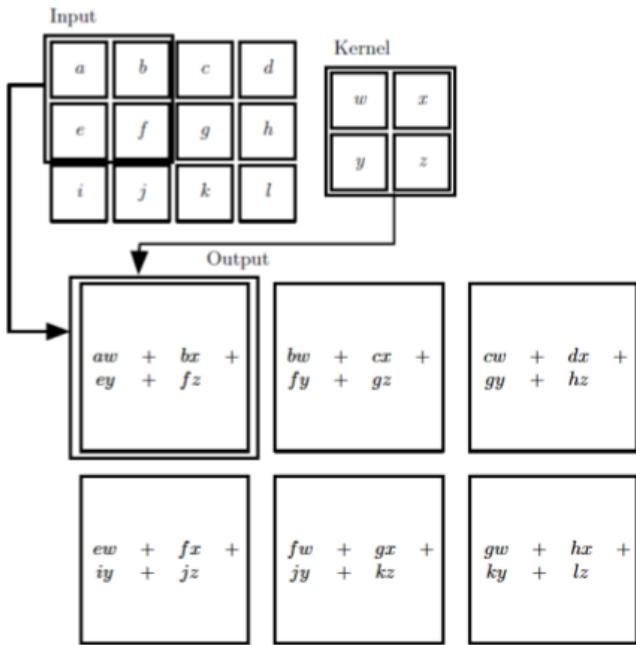


循环神经网络



图卷积网络

- 卷积神经网络：保留矩阵的信息（压扁成向量会丢失信息）



- 参数在卷积核上，卷积核平移；相比于前馈神经网络，参数量急剧下降
- 多个不同卷积核——>多个不同输出

### 5.6.2. 预训练+微调

预训练：无监督逐层训练

### 5.6.3. 权共享

一组神经元使用相同的连接权

在CNN中使用

每组神经元（每个“平面”）用相同的连接权

### 5.6.4. 微调

## 6. 支持向量机

### 6.1. 间隔与支持向量机

问题：分类问题中存在多个划分超平面时，应该去找哪一个？

在样本空间中，划分超平面可通过如下线性方程来描述：

$$\mathbf{w}^T \mathbf{x} + b = 0, \quad (6.1)$$

- $\vec{w} = (w_1; w_2; \dots; w_d)$  法向量，决定超平面的方向
- $b$  位移项，决定超平面和原点之间的距离
- 样本空间中的任意点  $\vec{x}$  到超平面  $(\vec{w}, b)$  的距离

$$r = \frac{|\vec{w}^T \vec{x} + b|}{\|\vec{w}\|} \quad (27)$$

若超平面  $(w', b')$  能将训练样本正确分类, 则总存在缩放变换  $\zeta w \mapsto w'$  和  $\zeta b \mapsto b'$  使式(6.3)成立.

假设超平面  $(w, b)$  能将训练样本正确分类, 即对于  $(x_i, y_i) \in D$ , 若  $y_i = +1$ , 则有  $w^T x_i + b > 0$ ; 若  $y_i = -1$ , 则有  $w^T x_i + b < 0$ . 令

$$\begin{cases} w^T x_i + b \geq +1, & y_i = +1; \\ w^T x_i + b \leq -1, & y_i = -1. \end{cases} \quad (6.3)$$

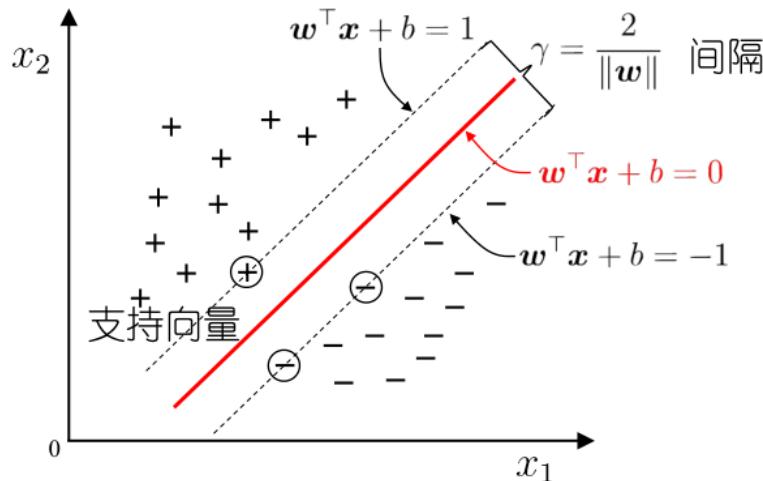
- 支持向量

到超平面最近、使得(6.3)等式成立的几个训练样本点

- 间隔

两个异类支持向量到平面的距离之和 (由距离计算式和(6.3)可得)

$$\gamma = \frac{2}{\|w\|} \quad (28)$$



欲找到具有“最大间隔”(maximum margin)的划分超平面, 也就是要找到能满足式(6.3)中约束的参数  $w$  和  $b$ , 使得  $\gamma$  最大, 即

$$\max_{w,b} \frac{2}{\|w\|} \quad (6.5)$$

$$\text{s.t. } y_i(w^T x_i + b) \geq 1, \quad i = 1, 2, \dots, m.$$

间隔貌似仅与  $w$  有关, 但事实上  $b$  通过约束隐式地影响着  $w$  的取值, 进而对间隔产生影响.

显然, 为了最大化间隔, 仅需最大化  $\|w\|^{-1}$ , 这等价于最小化  $\|w\|^2$ . 于是, 式(6.5)可重写为

$$\min_{w,b} \frac{1}{2} \|w\|^2 \quad (6.6)$$

$$\text{s.t. } y_i(w^T x_i + b) \geq 1, \quad i = 1, 2, \dots, m.$$

这就是支持向量机(Support Vector Machine, 简称 SVM)的基本型.

## 6.2. 对偶问题

解(6.5)(6.6)

## 6.2.1. 前置知识

### 6.2.1.1. 凸优化问题

考虑一般地约束优化问题

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, m \\ & h_j(\mathbf{x}) = 0, \quad j = 1, 2, \dots, n \end{aligned}$$

若目标函数  $f(\mathbf{x})$  是凸函数, 不等式约束  $g_i(\mathbf{x})$  是凸函数, 等式约束  $h_j(\mathbf{x})$  是仿射函数, 则称该优化问题为凸优化问题。

如果函数  $f: R^n \rightarrow R^m$  是仿射函数, 那么对于任意的  $\mathbf{x}, \mathbf{y} \in R^n, \alpha, \beta \in R$ ,

且  $\alpha + \beta = 1$ , 存在

$$f(\alpha\mathbf{x} + \beta\mathbf{y}) = \alpha f(\mathbf{x}) + \beta f(\mathbf{y})$$

- 线性函数是过原点的仿射函数 ( $\alpha = 1 \& \beta = 0$ )

### 6.2.1.2. KKT条件

考虑一般的约束优化问题

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, m \\ & h_j(\mathbf{x}) = 0, \quad j = 1, 2, \dots, n \end{aligned}$$

若  $f(\mathbf{x}), g_i(\mathbf{x}), h_j(\mathbf{x})$  的一阶偏导连续,  $\mathbf{x}^*$  是优化问题的局部解,  $\boldsymbol{\mu} = (\mu_1; \mu_2; \dots; \mu_m)$ ,  $\boldsymbol{\lambda} = (\lambda_1; \lambda_2; \dots; \lambda_n)$  为拉格朗日乘子向量,  $L(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \sum_{i=1}^m \mu_i g_i(\mathbf{x}) + \sum_{j=1}^n \lambda_j h_j(\mathbf{x})$  为拉格朗日函数, 且该优化问题满足任何一个特定的约束限制条件, 则一定存在  $\boldsymbol{\mu}^* = (\mu_1^*; \mu_2^*; \dots; \mu_m^*)$ ,  $\boldsymbol{\lambda}^* = (\lambda_1^*; \lambda_2^*; \dots; \lambda_n^*)$ , 使得:

- (1)  $\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\mu}^*, \boldsymbol{\lambda}^*) = \nabla f(\mathbf{x}^*) + \sum_{i=1}^m \mu_i^* \nabla g_i(\mathbf{x}^*) + \sum_{j=1}^n \lambda_j^* \nabla h_j(\mathbf{x}^*) = 0$ ;
- (2)  $h_j(\mathbf{x}^*) = 0, \quad j = 1, 2, \dots, n$ ;
- (3)  $g_i(\mathbf{x}^*) \leq 0, \quad i = 1, 2, \dots, m$ ;
- (4)  $\mu_i^* \geq 0, \quad i = 1, 2, \dots, m$ ;
- (5)  $\mu_i^* g_i(\mathbf{x}^*) = 0, \quad i = 1, 2, \dots, m$ .

以上 5 条便是 Karush–Kuhn–Tucker Conditions (简称 KKT 条件)。KKT 条件是局部解的必要条件, 也就是说只要该优化问题满足任何一个特定的约束限制条件, 局部解就一定会满足以上 5 个条件。常用的

### 6.2.1.3. 拉格朗日对偶函数

考虑一般地约束优化问题

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, m \\ & h_j(\mathbf{x}) = 0, \quad j = 1, 2, \dots, n \end{aligned}$$

设上述优化问题的定义域为  $D = \text{dom } f \cap \bigcap_{i=1}^m \text{dom } g_i \cap \bigcap_{j=1}^n \text{dom } h_j$ , 可行集为  $\tilde{D} = \{\mathbf{x} | \mathbf{x} \in D, g_i(\mathbf{x}) \leq 0, h_j(\mathbf{x}) = 0\}$  (显然  $\tilde{D}$  是  $D$  的子集), 最优值为  $p^* = \min\{f(\tilde{\mathbf{x}})\}, \tilde{\mathbf{x}} \in \tilde{D}$ 。上述优化问题的拉格朗日函数定义为

$$L(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \sum_{i=1}^m \mu_i g_i(\mathbf{x}) + \sum_{j=1}^n \lambda_j h_j(\mathbf{x})$$

其中  $\boldsymbol{\mu} = (\mu_1; \mu_2; \dots; \mu_m)$ ,  $\boldsymbol{\lambda} = (\lambda_1; \lambda_2; \dots; \lambda_n)$  为拉格朗日乘子向量。相应地拉格朗日对偶函数  $\Gamma(\boldsymbol{\mu}, \boldsymbol{\lambda})$  (简称对偶函数) 定义为  $L(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\lambda})$  关于  $\mathbf{x}$  的下确界, 即

$$\Gamma(\boldsymbol{\mu}, \boldsymbol{\lambda}) = \inf_{\mathbf{x} \in D} L(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\lambda}) = \inf_{\mathbf{x} \in D} \left( f(\mathbf{x}) + \sum_{i=1}^m \mu_i g_i(\mathbf{x}) + \sum_{j=1}^n \lambda_j h_j(\mathbf{x}) \right)$$

## 对偶函数性质

- 无论优化问题是否是凸优化的，对偶函数恒为凹函数
- 当  $\mu \succeq 0$  时 ( $\mu \succeq 0$  表示  $\mu$  的分量均为非负)， $\Gamma(\mu, \lambda)$  构成了上述优化问题最优值  $p^*$  的下界，

$$\Gamma(\mu, \lambda) \leq p^*$$

### 推导

设  $\bar{x} \in \bar{D}$  是优化问题的可行点，则  $g_i(\bar{x}) \leq 0, h_j(\bar{x}) = 0$ ，因此，当  $\mu \succeq 0$  时， $\mu_i g_i(\bar{x}) \leq 0, \lambda_j h_j(\bar{x}) = 0$  恒成立，所以

$$\sum_{i=1}^m \mu_i g_i(\bar{x}) + \sum_{j=1}^n \lambda_j h_j(\bar{x}) \leq 0$$

根据上述不等式可以推得

$$L(\bar{x}, \mu, \lambda) = f(\bar{x}) + \sum_{i=1}^m \mu_i g_i(\bar{x}) + \sum_{j=1}^n \lambda_j h_j(\bar{x}) \leq f(\bar{x})$$

又

$$\Gamma(\mu, \lambda) = \inf_{x \in D} L(x, \mu, \lambda) \leq L(\bar{x}, \mu, \lambda)$$

所以

$$\Gamma(\mu, \lambda) \leq L(\bar{x}, \mu, \lambda) \leq f(\bar{x})$$

进一步地

$$\Gamma(\mu, \lambda) \leq \min\{f(\bar{x})\} = p^*$$

### 6.2.1.4. 拉格朗日对偶问题

希望得到最好下界——

在  $\mu \succeq 0$  的约束下求对偶函数最大值的优化问题称为拉格朗日对偶问题（简称对偶问题）

$$\begin{aligned} & \max \quad \Gamma(\mu, \lambda) \\ & \text{s.t.} \quad \mu \succeq 0 \end{aligned}$$

对偶问题始终可以化成凸优化问题

设对偶问题的最优值为  $d^*$

- $d^* \leq p^*$  弱对偶性
- $d^* = p^*$  强对偶性

$\Rightarrow$  主问题最优下界，即最优解

- Slater 条件

若主问题是凸优化问题，且可行域中至少有一点使得不等式约束严格成立，则强对偶性成立

此时将拉格朗日函数分别对原变量和对偶变量求导，另导数等于 0，即得到原变量和对偶变量的关系

### 6.2.1.5. 二次规划——QP

目标是二次函数

约束条件是线性不等式

非标准二次规划问题中可以包含等式约束。注意到等式约束能用两个不等式约束来代替；不等式约束可通过增加松弛变量的方式转化为等式约束。

假定变量个数为  $d$ ，约束条件的个数为  $m$ ，则标准的二次规划问题形如

$$\begin{aligned} & \min_{\mathbf{x}} \quad \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \\ & \text{s.t.} \quad \mathbf{A} \mathbf{x} \leq \mathbf{b}, \end{aligned} \tag{B.12}$$

其中  $\mathbf{x}$  为  $d$  维向量， $\mathbf{Q} \in \mathbb{R}^{d \times d}$  为实对称矩阵， $\mathbf{A} \in \mathbb{R}^{m \times d}$  为实矩阵， $\mathbf{b} \in \mathbb{R}^m$  和  $\mathbf{c} \in \mathbb{R}^d$  为实向量， $\mathbf{A} \mathbf{x} \leq \mathbf{b}$  的每一行对应一个约束。

若  $\mathbf{Q}$  为半正定矩阵, 则式(B.12)目标函数是凸函数, 相应的二次规划是凸二次优化问题; 此时若约束条件  $\mathbf{A}\mathbf{x} \leq \mathbf{b}$  定义的可行域不为空, 且目标函数在此可行域有下界, 则该问题将有全局最小值. 若  $\mathbf{Q}$  为正定矩阵, 则该问题有唯一的全局最小值. 若  $\mathbf{Q}$  为非正定矩阵, 则式(B.12)是有多个平稳点和局部极小点的 NP 难问题.

### 6.2.2. 求解step1——转化为对偶问题

$$\begin{aligned} & \min_{\mathbf{w}, b} \quad \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t. } & y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, m. \end{aligned} \tag{6.6}$$

拉格朗日函数:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)), \tag{6.8}$$

$$\begin{aligned} L(\mathbf{w}, b, \boldsymbol{\alpha}) &= \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)) \\ &= \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m (\alpha_i - \alpha_i y_i \mathbf{w}^T \mathbf{x}_i - \alpha_i y_i b) \\ &= \frac{1}{2} \mathbf{w}^T \mathbf{w} + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i y_i \mathbf{w}^T \mathbf{x}_i - \sum_{i=1}^m \alpha_i y_i b \end{aligned}$$

其中  $\boldsymbol{\alpha} = (\alpha_1; \alpha_2; \dots; \alpha_m)$ . 令  $L(\mathbf{w}, b, \boldsymbol{\alpha})$  对  $\mathbf{w}$  和  $b$  的偏导为零可得

对  $\mathbf{w}$  和  $b$  分别求偏导数并令其为零

$$\begin{aligned} \boxed{\frac{\partial L}{\partial \mathbf{w}} = \frac{1}{2} \times 2 \times \mathbf{w} + 0 - \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i - 0 = 0 \implies \mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i} \\ \frac{\partial L}{\partial b} = 0 + 0 - 0 - \sum_{i=1}^m \alpha_i y_i = 0 \implies \sum_{i=1}^m \alpha_i y_i = 0 \end{aligned}$$

- $\alpha_i \geq 0$ , 且  $\frac{1}{2} \|\mathbf{w}\|^2$  和  $1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)$  均是关于  $\mathbf{w}$  和  $b$  的凸函数  
 $\Rightarrow$  (6.8) 也是凸函数
- 凸函数极值点就是最值点, 一阶导为0即为最小值点  
 $\Rightarrow$  将上面的等于0时的取值带入(6.8)得到最小值==下确界

因此可以求对偶函数为:

$$\begin{aligned}
\inf_{\mathbf{w}, b} L(\mathbf{w}, b, \boldsymbol{\alpha}) &= \frac{1}{2} \mathbf{w}^T \mathbf{w} + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i y_i \mathbf{w}^T \mathbf{x}_i - \sum_{i=1}^m \alpha_i y_i b \\
&= \frac{1}{2} \mathbf{w}^T \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i - \mathbf{w}^T \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i + \sum_{i=1}^m \alpha_i - b \sum_{i=1}^m \alpha_i y_i \\
&= -\frac{1}{2} \mathbf{w}^T \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i + \sum_{i=1}^m \alpha_i - b \sum_{i=1}^m \alpha_i y_i \\
&= -\frac{1}{2} \mathbf{w}^T \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i + \sum_{i=1}^m \alpha_i \\
&= -\frac{1}{2} (\sum_{i=1}^m \alpha_i y_i \mathbf{x}_i)^T (\sum_{i=1}^m \alpha_i y_i \mathbf{x}_i) + \sum_{i=1}^m \alpha_i \\
&= -\frac{1}{2} \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i^T \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i + \sum_{i=1}^m \alpha_i \\
&= \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j
\end{aligned}$$

对偶问题为(6.11):

$$\max_{\boldsymbol{\alpha}} \inf_{\mathbf{w}, b} L(\mathbf{w}, b, \boldsymbol{\alpha}) = \max_{\boldsymbol{\alpha}} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

约束条件

$$\begin{aligned}
\text{s.t. } \quad & \sum_{i=1}^m \alpha_i y_i = 0, \\
& \alpha_i \geq 0, \quad i = 1, 2, \dots, m.
\end{aligned}$$

作上述转化的原因

- (1) 式 (6.6) 中的未知数是  $\mathbf{w}$  和  $b$ , 式 (6.11) 中的未知数是  $\boldsymbol{\alpha}$ ,  $\mathbf{w}$  的维度  $d$  对应样本特征个数,  $\boldsymbol{\alpha}$  的维度  $m$  对应训练样本个数, 通常  $m \ll d$ , 所以求解式 (6.11) 更高效, 反之求解式 (6.6) 更高效;
- (2) 式 (6.11) 中有样本内积  $\mathbf{x}_i^T \mathbf{x}_j$  这一项, 后续可以很自然地引入核函数, 进而使得支持向量机也能对在原始特征空间线性不可分的数据进行分类。

对偶问题中解出的  $\alpha_i$  是原问题的拉格朗日乘子, 对应训练样本  $(\vec{x}_i, y_i)$

### 6.2.2.1. 梳理总结

原问题:

$$\begin{aligned}
\min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|^2 \\
\text{s.t. } \quad & y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, m.
\end{aligned} \tag{6.6}$$

拉格朗日函数:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^m \alpha_i (1 - y_i (\mathbf{w}^T \mathbf{x}_i + b)), \tag{6.8}$$

其中  $f(\vec{x}) = \vec{w}^T \vec{x} + b$

约束条件改写为

$$y_i f(\mathbf{x}_i) - 1 \geq 0 ;$$

对偶问题：

$$\max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \quad (6.11)$$

$$\text{s.t. } \begin{aligned} & \sum_{i=1}^m \alpha_i y_i = 0 , \\ & \alpha_i \geq 0 , \quad i = 1, 2, \dots, m . \end{aligned}$$

(6.6)满足Slater条件，故最优解满足KKT条件

$$\left\{ \begin{array}{l} \alpha_i \geq 0 ; \\ y_i f(\mathbf{x}_i) - 1 \geq 0 ; \\ \alpha_i (y_i f(\mathbf{x}_i) - 1) = 0 .. \end{array} \right. \quad (6.13)$$

数量关系：

原拉格朗日函数对原变量求导

$$\boxed{\frac{\partial L}{\partial \mathbf{w}} = \frac{1}{2} \times 2 \times \mathbf{w} + 0 - \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i - 0 = 0 \implies \mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i}$$

于是

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w}^T \mathbf{x} + b \\ &= \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b . \end{aligned} \quad (6.12)$$

再考虑(6.13)式的条件：

- 对任意训练样本有  $\alpha_i = 0$  或  $y_i f(\mathbf{x}_i) = 1$
- 前者：样本出现在(6.12)求和中
- 后者：必有  $y_i f(\mathbf{x}_i) = 1$ ，所对应的样本点位于最大间隔边界上，是一个支持向量。

$\Rightarrow$ 训练完成后，大部分样本不需要保留，最终模型只与支持向量有关

### 6.2.3. 求解step2——求解对偶问题

$$\max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \quad (6.11)$$

$$\text{s.t. } \begin{aligned} & \sum_{i=1}^m \alpha_i y_i = 0 , \\ & \alpha_i \geq 0 , \quad i = 1, 2, \dots, m . \end{aligned}$$

**SMO算法**

SMO 的基本思路是先固定  $\alpha_i$  之外的所有参数, 然后求  $\alpha_i$  上的极值. 由于存在约束  $\sum_{i=1}^m \alpha_i y_i = 0$ , 若固定  $\alpha_i$  之外的其他变量, 则  $\alpha_i$  可由其他变量导出. 于是, SMO 每次选择两个变量  $\alpha_i$  和  $\alpha_j$ , 并固定其他参数. 这样, 在参数初始化后, SMO 不断执行如下两个步骤直至收敛:

- 选取一对需更新的变量  $\alpha_i$  和  $\alpha_j$ ;
- 固定  $\alpha_i$  和  $\alpha_j$  以外的参数, 求解式(6.11)获得更新后的  $\alpha_i$  和  $\alpha_j$ .

只要选取的  $\alpha_i$  和  $\alpha_j$  中有一个不满足 KKT 条件, 目标函数就会在迭代后减小

KKT 违背程度越大, 变量更新后可能导致目标函数减幅越大

SMO 先选取违背 KKT 条件程度最大的变量

第二个变量选取使目标函数减小最快的变量

采用**启发式**: 使选取两个变量所对应的样本之间的间隔最大

SMO 算法之所以高效, 恰由于在固定其他参数后, 仅优化两个参数的过程能做到非常高效. 具体来说, 仅考虑  $\alpha_i$  和  $\alpha_j$  时, 式(6.11)中的约束可重写为

$$\alpha_i y_i + \alpha_j y_j = c, \quad \alpha_i \geq 0, \quad \alpha_j \geq 0, \quad (6.14)$$

其中

$$c = - \sum_{k \neq i, j} \alpha_k y_k \quad (6.15)$$

是使  $\sum_{i=1}^m \alpha_i y_i = 0$  成立的常数. 用

$$\alpha_i y_i + \alpha_j y_j = c \quad (6.16)$$

消去式(6.11)中的变量  $\alpha_j$ , 则得到一个关于  $\alpha_i$  的单变量二次规划问题, 仅有的约束是  $\alpha_i \geq 0$ . 不难发现, 这样的二次规划问题具有闭式解, 于是不必调用数值优化算法即可高效地计算出更新后的  $\alpha_i$  和  $\alpha_j$ .

如何确定偏移项  $b$  呢? 注意到对任意支持向量  $(\mathbf{x}_s, y_s)$  都有  $y_s f(\mathbf{x}_s) = 1$ , 即

$$y_s \left( \sum_{i \in S} \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_s + b \right) = 1, \quad (6.17)$$

其中  $S = \{i \mid \alpha_i > 0, i = 1, 2, \dots, m\}$  为所有支持向量的下标集. 理论上, 可选取任意支持向量并通过求解式(6.17)获得  $b$ , 但现实任务中常采用一种更鲁棒的做法: 使用所有支持向量求解的平均值

$$b = \frac{1}{|S|} \sum_{s \in S} \left( y_s - \sum_{i \in S} \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_s \right). \quad (6.18)$$

$$b = \frac{1}{|S|} \sum_{s \in S} \left( \frac{1}{y_s} - \sum_{i \in S} \alpha_i y_i \vec{\mathbf{x}}_i^T \vec{\mathbf{x}}_s \right) \quad (29)$$

## 6.3. 核函数

### 6.3.1. 非线性映射

训练样本非线性可分时，可以将样本从原始空间映射到一个更高维的特征空间，使得样本在这个特征空间内线性可分

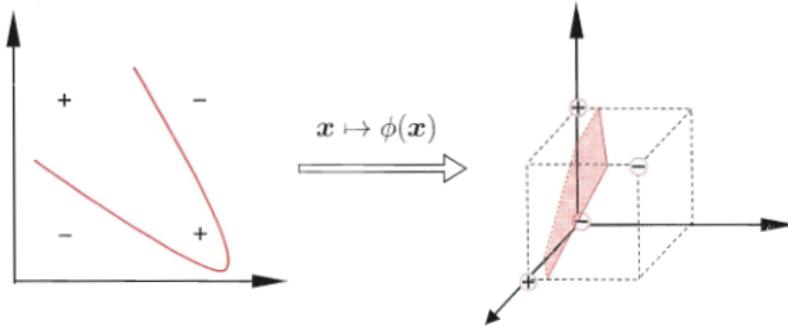


图 6.3 异或问题与非线性映射

若原始空间是有限维的，即属性数有限，则一定存在一个高维特征空间使样本可分

映射后 $\vec{x}$ 的特征向量： $\phi(\vec{x})$

特征空间中划分超平面的模型： $f(\vec{x}) = \vec{w}^T \phi(\vec{x}) + b$

其中  $w$  和  $b$  是模型参数。类似式(6.6)，有

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} \|w\|^2 \\ \text{s.t. } \quad & y_i(w^T \phi(x_i) + b) \geq 1, \quad i = 1, 2, \dots, m. \end{aligned} \tag{6.20}$$

其对偶问题是

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \\ & \alpha_i \geq 0, \quad i = 1, 2, \dots, m. \end{aligned} \tag{6.21}$$

- 特征空间的维数可能很高/无穷维， $\phi(\vec{x}_i)^T \phi(\vec{x}_j)$ 难以计算

### 6.3.2. 核技巧 核函数

#### 核技巧

$$\kappa(\vec{x}_i, \vec{x}_j) = \langle \phi(\vec{x}_i), \phi(\vec{x}_j) \rangle = \phi(\vec{x}_i)^T \phi(\vec{x}_j), \tag{6.22}$$

式(6.21)可重写为

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \\ & \alpha_i \geq 0, \quad i = 1, 2, \dots, m. \end{aligned} \quad (6.23)$$

求解后即可得到

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w}^T \phi(\mathbf{x}) + b \\ &= \sum_{i=1}^m \alpha_i y_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) + b \\ &= \sum_{i=1}^m \alpha_i y_i \kappa(\mathbf{x}, \mathbf{x}_i) + b. \end{aligned} \quad (6.24)$$

$\Rightarrow$ 模型最优解可以通过训练样本的核函数展开——支持向量展式

### 6.3.3. 核函数

**定理 6.1 (核函数)** 令  $\mathcal{X}$  为输入空间,  $\kappa(\cdot, \cdot)$  是定义在  $\mathcal{X} \times \mathcal{X}$  上的对称函数, 则  $\kappa$  是核函数当且仅当对于任意数据  $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ , “核矩阵” (kernel matrix)  $\mathbf{K}$  总是半正定的:

$$\mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_m) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_i, \mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_i, \mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_i, \mathbf{x}_m) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_m, \mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_m, \mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_m, \mathbf{x}_m) \end{bmatrix}.$$

- 对于一个半正定核矩阵, 总能找到一个对应的映射  $\phi$

即: 任何一个核函数都隐式的定义了一个特征空间——“再生核希尔伯特空间” (RKHS)

表 6.1 常用核函数

名称	表达式	参数
线性核 <small><math>d = 1</math> 时退化为线性核.</small>	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$	
多项式核 <small>高斯核亦称 RBF 核.</small>	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j)^d$	$d \geq 1$ 为多项式的次数
高斯核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\ \mathbf{x}_i - \mathbf{x}_j\ ^2}{2\sigma^2}\right)$	$\sigma > 0$ 为高斯核的带宽 (width)
拉普拉斯核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\ \mathbf{x}_i - \mathbf{x}_j\ }{\sigma}\right)$	$\sigma > 0$
Sigmoid 核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta \mathbf{x}_i^T \mathbf{x}_j + \theta)$	$\tanh$ 为双曲正切函数, $\beta > 0, \theta < 0$

- 基本经验: 文本数据通常采用线性核, 情况不明时可先尝试高斯核

也可以通过函数组合得到核函数

- 若  $\kappa_1$  和  $\kappa_2$  为核函数, 则对于任意正数  $\gamma_1, \gamma_2$ , 其线性组合

$$\gamma_1 \kappa_1 + \gamma_2 \kappa_2 \quad (6.25)$$

也是核函数;

- 若  $\kappa_1$  和  $\kappa_2$  为核函数, 则核函数的直积

$$\kappa_1 \otimes \kappa_2(\mathbf{x}, \mathbf{z}) = \kappa_1(\mathbf{x}, \mathbf{z})\kappa_2(\mathbf{x}, \mathbf{z}) \quad (6.26)$$

也是核函数;

- 若  $\kappa_1$  为核函数, 则对于任意函数  $g(\mathbf{x})$ ,

$$\kappa(\mathbf{x}, \mathbf{z}) = g(\mathbf{x})\kappa_1(\mathbf{x}, \mathbf{z})g(\mathbf{z}) \quad (6.27)$$

也是核函数.

## 6.4. 软间隔与正则化

现实任务中往往很难找到合适的核函数使得训练集再特征空间中线性可分;  
且可能线性可分实际是过拟合造成的

### 6.4.1. 软间隔

- 硬间隔: 所有样本必须划分正确

- 软间隔: 允许某些样本不满足约束

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1. \quad (6.28)$$

在最大化间隔的同时, 不满足约束的样本应尽可能少

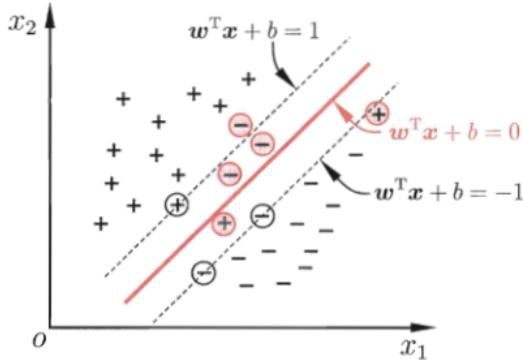


图 6.4 软间隔示意图. 红色圈出了一些不满足约束的样本.

优化目标:

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \ell_{0/1}(y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1), \quad (6.29)$$

其中  $C > 0$  是一个常数,  $\ell_{0/1}$  是“0/1损失函数”

$$\ell_{0/1}(z) = \begin{cases} 1, & \text{if } z < 0; \\ 0, & \text{otherwise.} \end{cases} \quad (6.30)$$

显然, 当  $C$  为无穷大时, 式(6.29)迫使所有样本均满足约束(6.28), 于是式(6.29)等价于(6.6); 当  $C$  取有限值时, 式(6.29)允许一些样本不满足约束.

$\ell_{0/1}$  数学性质不好

- 非凸
- 非连续

替代损失：

通常是凸的连续函数

对率损失是对率函数的变形，对率函数参见 3.3 节。

对率损失函数通常表示为  $\ell_{\log}(\cdot)$ ，因此式(6.33)把式(3.15)中的  $\ln(\cdot)$  改写为  $\log(\cdot)$ 。

$$\text{hinge 损失: } \ell_{\text{hinge}}(z) = \max(0, 1 - z); \quad (6.31)$$

$$\text{指数损失(exponential loss): } \ell_{\exp}(z) = \exp(-z); \quad (6.32)$$

$$\text{对率损失(logistic loss): } \ell_{\log}(z) = \log(1 + \exp(-z)). \quad (6.33)$$

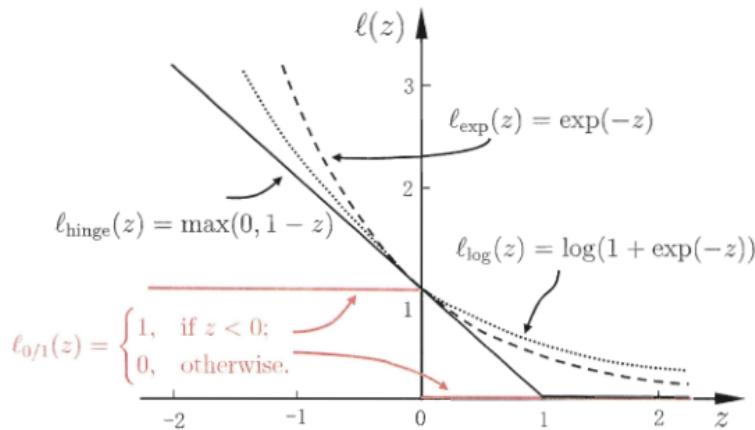


图 6.5 三种常见的替代损失函数：hinge损失、指数损失、对率损失

#### 6.4.1.1. hinge损失

若采用 hinge 损失，则式(6.29)变成

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \max(0, 1 - y_i (\mathbf{w}^\top \mathbf{x}_i + b)). \quad (6.34)$$

引入“松弛变量”(slack variables)  $\xi_i \geq 0$ ，可将式(6.34)重写为

$$\min_{\mathbf{w}, b, \xi_i} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i \quad (6.35)$$

$$\text{s.t. } y_i (\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0, i = 1, 2, \dots, m.$$

#### 软间隔支持向量机

每个松弛变量对应一个样本，表示该样本不满足约束(6.28)的程度

**求解 (6.35)**

二次规划问题

拉格朗日函数：

$$L(\mathbf{w}, b, \alpha, \xi, \mu) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i \\ + \sum_{i=1}^m \alpha_i (1 - \xi_i - y_i (\mathbf{w}^\top \mathbf{x}_i + b)) - \sum_{i=1}^m \mu_i \xi_i , \quad (6.36)$$

其中  $\alpha_i \geq 0, \mu_i \geq 0$  是拉格朗日乘子.

令  $L(\mathbf{w}, b, \alpha, \xi, \mu)$  对  $\mathbf{w}, b, \xi_i$  的偏导为零可得

$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i , \quad (6.37)$$

$$0 = \sum_{i=1}^m \alpha_i y_i , \quad (6.38)$$

$$C = \alpha_i + \mu_i . \quad (6.39)$$

拉格朗日对偶函数

$$L(\vec{\mathbf{w}}, \vec{b}) = \inf \left( \frac{1}{2} \|\vec{\mathbf{w}}\|^2 + C \sum_{i=1}^m \xi_i + \sum_{i=1}^m \alpha_i (1 - \xi_i - y_i (\vec{\mathbf{w}}^\top \vec{\mathbf{x}}_i + b)) - \sum_{i=1}^m \mu_i \xi_i \right) \\ = \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \vec{\mathbf{x}}_i^\top \vec{\mathbf{x}}_j + \underbrace{(\alpha_i + \mu_i) \sum_{i=1}^m \xi_i}_{+ \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i \xi_i - \underbrace{\sum_{i=1}^m \alpha_i y_i (\vec{\mathbf{w}}^\top \vec{\mathbf{x}}_i + b)}_{-\sum_{i=1}^m \mu_i \xi_i} \\ = \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \vec{\mathbf{x}}_i^\top \vec{\mathbf{x}}_j + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i y_i \vec{\mathbf{w}}^\top \vec{\mathbf{x}}_i \\ = \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \vec{\mathbf{x}}_i^\top \vec{\mathbf{x}}_j + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i y_i \left( \sum_{j=1}^m \alpha_j y_j \vec{\mathbf{x}}_j^\top \right) \vec{\mathbf{x}}_i \\ = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \vec{\mathbf{x}}_i^\top \vec{\mathbf{x}}_j$$

约束条件

$$\begin{cases} \alpha_i \geq 0 \\ \mu_i \geq 0 \\ \sum_{i=1}^m \alpha_i y_i = 0 \\ C = \alpha_i + \mu_i \end{cases} \Rightarrow 0 \leq \alpha_i \leq C$$

对偶问题

$$\begin{aligned}
 \max_{\alpha} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\
 \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \\
 & 0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, m.
 \end{aligned} \tag{6.40}$$

对比硬间隔下对偶问题

$$\begin{aligned}
 \max_{\alpha} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\
 \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \\
 & \alpha_i \geq 0, \quad i = 1, 2, \dots, m.
 \end{aligned} \tag{6.11}$$

区别： $\alpha_i$ 有上界

KKT条件

$$\left\{
 \begin{array}{l}
 \alpha_i \geq 0, \quad \mu_i \geq 0, \\
 y_i f(\mathbf{x}_i) - 1 + \xi_i \geq 0, \\
 \alpha_i (y_i f(\mathbf{x}_i) - 1 + \xi_i) = 0, \\
 \xi_i \geq 0, \quad \mu_i \xi_i = 0.
 \end{array}
 \right. \tag{6.41}$$

对任意样本

- $\alpha_i = 0$ , 不对  $f(\vec{x})$  产生影响
- $\alpha_i > 0$ ,  $y_i f(\vec{x}_i) = 1 - \xi_i$ , 支持向量
- 若  $\alpha_i < C$ , 则  $\mu_i > 0$ , 由 KKT 条件,  $\xi_i = 0$ , 样本恰好落在最大间隔边界上
- 若  $\alpha_i = C$ , 则  $\mu_i = 0$ 
  - 若  $\xi_i \leq 1$ , 样本落在最大间隔内部
  - 若  $\xi_i > 1$ , 样本被错误分类

软间隔支持向量机模型最终只与支持向量有关, hinge 损失保持了稀疏性

#### 6.4.1.2. 对率损失

若用对率损失, 几乎得到对率回归模型

$$\ell(\beta) = \sum_{i=1}^m \left( -y_i \beta^T \hat{\mathbf{x}}_i + \ln(1 + e^{\beta^T \hat{\mathbf{x}}_i}) \right). \tag{3.27}$$

说明

上式变形如下：

将  $\beta = (\mathbf{w}; b)$  和  $\hat{\mathbf{x}} = (\mathbf{x}; 1)$  代入式 (3.27) 可得

$$\begin{aligned}\ell(\mathbf{w}, b) &= \sum_{i=1}^m \left( -y_i (\mathbf{w}^\top \mathbf{x}_i + b) + \ln \left( 1 + e^{\mathbf{w}^\top \mathbf{x}_i + b} \right) \right) \\ &= \sum_{i=1}^m \left( \ln \frac{1}{e^{y_i(\mathbf{w}^\top \mathbf{x}_i + b)}} + \ln \left( 1 + e^{\mathbf{w}^\top \mathbf{x}_i + b} \right) \right) \\ &= \sum_{i=1}^m \ln \frac{1 + e^{\mathbf{w}^\top \mathbf{x}_i + b}}{e^{y_i(\mathbf{w}^\top \mathbf{x}_i + b)}} \\ &= \begin{cases} \sum_{i=1}^m \ln \left( 1 + e^{-(\mathbf{w}^\top \mathbf{x}_i + b)} \right), & y_i = 1 \\ \sum_{i=1}^m \ln \left( 1 + e^{\mathbf{w}^\top \mathbf{x}_i + b} \right), & y_i = 0 \end{cases}\end{aligned}$$

对数几率回归常用1和0表示正例和反例，而支持向量机中常用+1和-1表示

若将上式改为+1和-1表示正例和反例，则有

上式可改写为

$$\begin{aligned}\ell(\mathbf{w}, b) &= \begin{cases} \sum_{i=1}^m \ln \left( 1 + e^{-(\mathbf{w}^\top \mathbf{x}_i + b)} \right), & y_i = +1 \\ \sum_{i=1}^m \ln \left( 1 + e^{\mathbf{w}^\top \mathbf{x}_i + b} \right), & y_i = -1 \end{cases} \\ &= \sum_{i=1}^m \ln \left( 1 + e^{-y_i(\mathbf{w}^\top \mathbf{x}_i + b)} \right)\end{aligned}$$

上式的求和项正是式 (6.33) 所表述的对率损失。

对率损失(logistic loss):  $\ell_{log}(z) = \log(1 + \exp(-z))$ . (6.33)

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \ell_{0/1} (y_i (\mathbf{w}^\top \mathbf{x}_i + b) - 1), \quad (6.29)$$

支持向量机与对率回归的优化目标相近，通常性能相当

- 对率回归输出有概率意义，而支持向量机没有
  - 对率回归可以直接用于多分类任务，支持向量机需推广
  - hinge损失保持稀疏性，对率损失是光滑的单调递减函数，不能导出类似支持向量的概念
- 对率回归的解依赖于更多的样本，预测开销更大

#### 6.4.1.3. 不同损失函数

使用不同损失函数替代得到的模型共性：

优化目标的第一项描述划分超平面的“间隔”大小

另一项描述训练集上的误差：  $\sum_{i=1}^m \ell(f(\mathbf{x}_i), y_i)$

一般形式：

$$\min_f \Omega(f) + C \sum_{i=1}^m \ell(f(\mathbf{x}_i), y_i), \quad (6.42)$$

- $\Omega(f)$   
结构风险：描述模型f的某些性质
- $\sum_{i=1}^m \ell(f(\vec{x}_i), \vec{y}_i)$

- C

对两者进行折中

从经验风险最小化的角度看，结构风险描述了希望获得具有何种性质的模型（如：复杂度较小的）

有助于削减假设空间，降低最小化训练误差的过拟合风险

从这个角度，(6.42)是正则化问题——罚函数

- $\Omega(f)$ : 正则化项，可以认为是提供了模型的先验概率

$L_p$ 范数是常用的正则化项

$L_2$ 范数倾向于各分量取值均衡，非零分量尽量稠密

$L_0$ 范数和 $L_1$ 范数倾向于 $w$ 尽量稀疏，非零分量尽量少

- C: 正则化常数

## 6.5. 支持向量回归

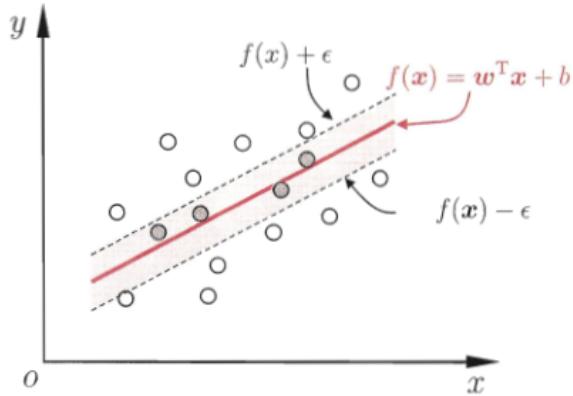


图 6.6 支持向量回归示意图。红色显示出  $\epsilon$ -间隔带，落入其中的样本不计算损失。

SVR问题可形式化为：

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \ell_\epsilon(f(\mathbf{x}_i) - y_i), \quad (6.43)$$

其中  $C$  为正则化常数， $\ell_\epsilon$  是图 6.7 所示的  $\epsilon$ -不敏感损失 ( $\epsilon$ -insensitive loss) 函数

$$\ell_\epsilon(z) = \begin{cases} 0, & \text{if } |z| \leq \epsilon; \\ |z| - \epsilon, & \text{otherwise.} \end{cases} \quad (6.44)$$

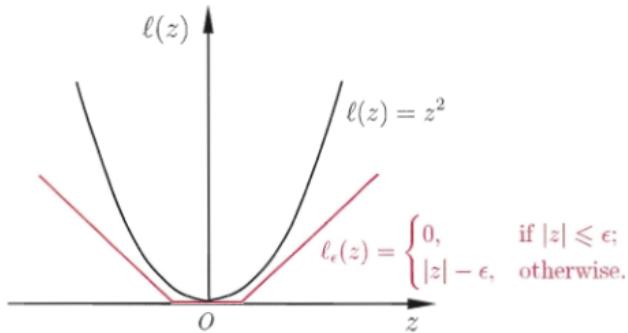


图 6.7  $\epsilon$ -不敏感损失函数

间隔带两侧的松弛程度  
可有所不同.

引入松弛变量  $\xi_i$  和  $\hat{\xi}_i$ , 可将式(6.43)重写为

$$\min_{w, b, \xi_i, \hat{\xi}_i} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m (\xi_i + \hat{\xi}_i) \quad (6.45)$$

$$\text{s.t. } f(\mathbf{x}_i) - y_i \leq \epsilon + \xi_i ,$$

$$y_i - f(\mathbf{x}_i) \leq \epsilon + \hat{\xi}_i ,$$

$$\xi_i \geq 0, \hat{\xi}_i \geq 0, i = 1, 2, \dots, m.$$

### 解释

同软间隔支持向量机 引入松弛变量后:

$$\begin{cases} f(\mathbf{x}_i) - y_i = \xi_i & \text{显然 } \xi_i \geq 0 \\ |f(\mathbf{x}_i) - y_i| < \epsilon \text{ 时, } \xi_i = 0 \\ |f(\mathbf{x}_i) - y_i| \geq \epsilon \text{ 时, } \xi_i = |f(\mathbf{x}_i) - y_i| - \epsilon \end{cases}$$

$$\Rightarrow \min_{w, b, \xi_i} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \xi_i$$

$$\text{s.t. } -\epsilon - \xi_i \leq f(\mathbf{x}_i) - y_i \leq \epsilon + \xi_i$$

$$\xi_i \geq 0, i = 1, 2, \dots, m.$$

$$\Rightarrow \xi_i = \max\{|f(\mathbf{x}_i) - y_i| - \epsilon, 0\}$$

$$\xi_i \geq |f(\mathbf{x}_i) - y_i| - \epsilon$$

$$|f(\mathbf{x}_i) - y_i| \leq \xi_i + \epsilon$$

$$-\epsilon - \xi_i \leq f(\mathbf{x}_i) - y_i \leq \epsilon + \xi_i$$

若两边采用不同松弛程度:

$$f(\mathbf{x}_i) - y_i = \xi_i + \hat{\xi}_i$$

$$\begin{cases} |f(\mathbf{x}_i) - y_i| < \epsilon \text{ 时, } \xi_i = \hat{\xi}_i = 0 \\ |f(\mathbf{x}_i) - y_i| \geq \epsilon \text{ 时, } \xi_i = f(\mathbf{x}_i) - y_i - \epsilon, \hat{\xi}_i = 0 \\ |f(\mathbf{x}_i) - y_i| \leq \epsilon \text{ 时, } \xi_i = 0, \hat{\xi}_i = y_i - f(\mathbf{x}_i) - \epsilon \end{cases}$$

$$\Rightarrow \min_{w, b, \xi_i, \hat{\xi}_i} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m (\xi_i + \hat{\xi}_i)$$

$$\text{s.t. } -\epsilon - \hat{\xi}_i \leq f(\mathbf{x}_i) - y_i \leq \epsilon + \xi_i$$

$$\xi_i \geq 0, \hat{\xi}_i \geq 0, i = 1, \dots, m.$$

$$\Rightarrow \xi_i \geq f(\mathbf{x}_i) - y_i - \epsilon \Rightarrow f(\mathbf{x}_i) - y_i \leq \xi_i + \epsilon$$

$$\hat{\xi}_i \geq y_i - f(\mathbf{x}_i) - \epsilon \Rightarrow f(\mathbf{x}_i) - y_i \geq -\epsilon - \hat{\xi}_i$$

拉格朗日函数:

$$L(w, b, \alpha, \hat{\alpha}, \xi, \hat{\xi}, \mu, \hat{\mu})$$

$$\begin{aligned} &= \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m (\xi_i + \hat{\xi}_i) - \sum_{i=1}^m \mu_i \xi_i - \sum_{i=1}^m \hat{\mu}_i \hat{\xi}_i \\ &+ \sum_{i=1}^m \alpha_i (f(\mathbf{x}_i) - y_i - \epsilon - \xi_i) + \sum_{i=1}^m \hat{\alpha}_i (y_i - f(\mathbf{x}_i) - \epsilon - \hat{\xi}_i). \end{aligned} \quad (6.46)$$

将式(6.7)代入, 再令  $L(w, b, \alpha, \hat{\alpha}, \xi, \hat{\xi}, \mu, \hat{\mu})$  对  $w, b, \xi_i$  和  $\hat{\xi}_i$  的偏导为零可得

$$w = \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) \mathbf{x}_i, \quad (6.47)$$

$$0 = \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i), \quad (6.48)$$

$$C = \alpha_i + \mu_i, \quad (6.49)$$

$$C = \hat{\alpha}_i + \hat{\mu}_i. \quad (6.50)$$

将式(6.47)–(6.50)代入式(6.46), 即可得到 SVR 的对偶问题

$$\begin{aligned} \max_{\alpha, \hat{\alpha}} \quad & \sum_{i=1}^m y_i (\hat{\alpha}_i - \alpha_i) - \epsilon (\hat{\alpha}_i + \alpha_i) \\ & - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m (\hat{\alpha}_i - \alpha_i)(\hat{\alpha}_j - \alpha_j) \mathbf{x}_i^T \mathbf{x}_j \\ \text{s.t.} \quad & \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) = 0, \\ & 0 \leq \alpha_i, \hat{\alpha}_i \leq C. \end{aligned} \quad (6.51)$$

KKT条件

将式 (6.45) 的约束条件全部恒等变形为小于等于 0 的形式可得

$$\begin{cases} f(\mathbf{x}_i) - y_i - \epsilon - \xi_i \leq 0 \\ y_i - f(\mathbf{x}_i) - \epsilon - \hat{\xi}_i \leq 0 \\ -\xi_i \leq 0 \\ -\hat{\xi}_i \leq 0 \end{cases}$$

由于以上四个约束条件的拉格朗日乘子分别为  $\alpha_i, \hat{\alpha}_i, \mu_i, \hat{\mu}_i$ , 所以其对应的 KKT 条件为

$$\begin{cases} \alpha_i (f(\mathbf{x}_i) - y_i - \epsilon - \xi_i) = 0 \\ \hat{\alpha}_i (y_i - f(\mathbf{x}_i) - \epsilon - \hat{\xi}_i) = 0 \\ -\mu_i \xi_i = 0 \Rightarrow \mu_i \xi_i = 0 \\ -\hat{\mu}_i \hat{\xi}_i = 0 \Rightarrow \hat{\mu}_i \hat{\xi}_i = 0 \end{cases}$$

**每个不等式约束对应一个拉格朗日乘子, 且要求乘积均为0**

又由式 (6.49) 和式 (6.50) 有

$$\begin{cases} \mu_i = C - \alpha_i \\ \hat{\mu}_i = C - \hat{\alpha}_i \end{cases}$$

所以上述 KKT 条件可以进一步变形为

$$\begin{cases} \alpha_i (f(\mathbf{x}_i) - y_i - \epsilon - \xi_i) = 0 \\ \hat{\alpha}_i (y_i - f(\mathbf{x}_i) - \epsilon - \hat{\xi}_i) = 0 \\ (C - \alpha_i) \xi_i = 0 \\ (C - \hat{\alpha}_i) \hat{\xi}_i = 0 \end{cases}$$

前两个式子, 括号内=0代表样本落在间隔带外

当样本落在间隔带内时,  $\alpha_i$  和  $\hat{\alpha}_i$  都为0

样本  $(\vec{x}_i, y_i)$  只能处在间隔带的一侧, 即  $f(\mathbf{x}_i) - y_i - \epsilon - \xi_i = 0$  和  $y_i - f(\mathbf{x}_i) - \epsilon - \hat{\xi}_i = 0$  不能同时成立, 因此  $\alpha_i$  和  $\hat{\alpha}_i$  至少有一个为0

进而, 由后两个式子可知,  $\alpha_i = 0$  时,  $\xi_i = 0$ , 即  $\xi_i$  和  $\hat{\xi}_i$  至少有一个为0

$$\begin{cases} \alpha_i (f(\mathbf{x}_i) - y_i - \epsilon - \xi_i) = 0, \\ \hat{\alpha}_i (y_i - f(\mathbf{x}_i) - \epsilon - \hat{\xi}_i) = 0, \\ \alpha_i \hat{\alpha}_i = 0, \xi_i \hat{\xi}_i = 0, \\ (C - \alpha_i) \xi_i = 0, (C - \hat{\alpha}_i) \hat{\xi}_i = 0. \end{cases} \quad (6.52)$$

由(6.47)得SVR的解为

$$f(\mathbf{x}) = \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) \mathbf{x}_i^T \mathbf{x} + b . \quad (6.53)$$

能使式(6.53)中的  $(\hat{\alpha}_i - \alpha_i) \neq 0$  的样本即为 SVR 的支持向量, 此时前两个式子的括号部分有且仅有一个为 0, 支持向量落在间隔带之外

- 若  $0 < \alpha_i < C \Rightarrow \xi_i = 0$

进而有

$$b = y_i + \epsilon - \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) \mathbf{x}_i^T \mathbf{x} . \quad (6.54)$$

因此, 在求解式(6.51)得到  $\alpha_i$  后, 理论上来说, 可任意选取满足  $0 < \alpha_i < C$  的样本通过式(6.54)求得  $b$ . 实践中常采用一种更鲁棒的办法: 选取多个(或所有)满足条件  $0 < \alpha_i < C$  的样本求解  $b$  后取平均值.

### 6.5.1. 非线性特征映射

(6.47)变为

$$\mathbf{w} = \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) \phi(\mathbf{x}_i) . \quad (6.55)$$

SVR表示为

$$f(\mathbf{x}) = \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) \kappa(\mathbf{x}, \mathbf{x}_i) + b , \quad (6.56)$$

其中  $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$  为核函数.

## 6.6. 核方法

**定理 6.2 (表示定理)** 令  $\mathbb{H}$  为核函数  $\kappa$  对应的再生核希尔伯特空间,  $\|h\|_{\mathbb{H}}$  表示  $\mathbb{H}$  空间中关于  $h$  的范数, 对于任意单调递增函数  $\Omega : [0, \infty] \mapsto \mathbb{R}$  和任意非负损失函数  $\ell : \mathbb{R}^m \mapsto [0, \infty]$ , 优化问题

$$\min_{h \in \mathbb{H}} F(h) = \Omega(\|h\|_{\mathbb{H}}) + \ell(h(\mathbf{x}_1), h(\mathbf{x}_2), \dots, h(\mathbf{x}_m)) \quad (6.57)$$

的解总可写为

$$h^*(\mathbf{x}) = \sum_{i=1}^m \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i) . \quad (6.58)$$

### 6.6.1. 核化 非线性拓展

我们先假设可通过某种映射  $\phi: \mathcal{X} \mapsto \mathbb{F}$  将样本映射到一个特征空间  $\mathbb{F}$ , 然后在  $\mathbb{F}$  中执行线性判别分析, 以求得

$$h(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) . \quad (6.59)$$

线性判别分析——LDA:

$$J = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}} . \quad (3.35)$$

这就是 LDA 欲最大化的目标, 即  $\mathbf{S}_b$  与  $\mathbf{S}_w$  的“广义瑞利商”(generalized Rayleigh quotient).

核线性判别分析——KLDA

学习目标:

$$\max_{\mathbf{w}} J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b^\phi \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w^\phi \mathbf{w}} , \quad (6.60)$$

其中  $\mathbf{S}_b^\phi$  和  $\mathbf{S}_w^\phi$  分别为训练样本在特征空间  $\mathbb{F}$  中的类间散度矩阵和类内散度矩阵. 令  $X_i$  表示第  $i \in \{0, 1\}$  类样本的集合, 其样本数为  $m_i$ ; 总样本数  $m = m_0 + m_1$ . 第  $i$  类样本在特征空间  $\mathbb{F}$  中的均值为

$$\boldsymbol{\mu}_i^\phi = \frac{1}{m_i} \sum_{\mathbf{x} \in X_i} \phi(\mathbf{x}) , \quad (6.61)$$

两个散度矩阵分别为

$$\mathbf{S}_b^\phi = (\boldsymbol{\mu}_1^\phi - \boldsymbol{\mu}_0^\phi)(\boldsymbol{\mu}_1^\phi - \boldsymbol{\mu}_0^\phi)^T ; \quad (6.62)$$

$$\mathbf{S}_w^\phi = \sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} (\phi(\mathbf{x}) - \boldsymbol{\mu}_i^\phi)(\phi(\mathbf{x}) - \boldsymbol{\mu}_i^\phi)^T . \quad (6.63)$$

- 用  $\kappa(\vec{x}, \vec{x}_i) = \phi(\vec{x}_i)^T \phi(\vec{x})$  隐式表示映射  $\phi$  和特征空间  $\mathbb{F}$ , 相应的, 后面引入的  $\hat{\boldsymbol{\mu}}_i$  与  $\boldsymbol{\mu}_i^\phi$  相比也发生变化
- $J(\vec{w})$  作为损失函数
- $\Omega = 0$

由表示定理, 该优化问题的解可以表示为:

$$h(\mathbf{x}) = \sum_{i=1}^m \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i) , \quad (6.64)$$

对照(6.59), 有

$$\mathbf{w} = \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i) . \quad (6.65)$$

下面求优化目标

$$\max_{\mathbf{w}} J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b^\phi \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w^\phi \mathbf{w}} , \quad (6.60)$$

**step1.**指示向量

令  $\mathbf{K} \in \mathbb{R}^{m \times m}$  为核函数  $\kappa$  所对应的核矩阵,  $(\mathbf{K})_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ . 令  $\mathbf{1}_i \in \{1, 0\}^{m \times 1}$  为第  $i$  类样本的指示向量, 即  $\mathbf{1}_i$  的第  $j$  个分量为 1 当且仅当  $\mathbf{x}_j \in X_i$ , 否则  $\mathbf{1}_i$  的第  $j$  个分量为 0.

### 说人话

对于二分类问题, 只有两类,  $i$  为 0 或 1

每一类对应一个指示向量,  $m \times 1$  维, 每一行对应一个样本是不是当前类的, 即若第  $j$  行为 1, 表示  $\mathbf{x}_j$  是该类的, 否则不是

e.g.

样本 1, 3 是第 0 类, 样本 2, 4 是第 1 类

$$m = 4$$

$$m_0 = 2, m_1 = 2$$

$$X_0 = \{\mathbf{x}_1, \mathbf{x}_3\}, X_1 = \{\mathbf{x}_2, \mathbf{x}_4\}$$

$$\mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \kappa(\mathbf{x}_1, \mathbf{x}_2) & \kappa(\mathbf{x}_1, \mathbf{x}_3) & \kappa(\mathbf{x}_1, \mathbf{x}_4) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) & \kappa(\mathbf{x}_2, \mathbf{x}_2) & \kappa(\mathbf{x}_2, \mathbf{x}_3) & \kappa(\mathbf{x}_2, \mathbf{x}_4) \\ \kappa(\mathbf{x}_3, \mathbf{x}_1) & \kappa(\mathbf{x}_3, \mathbf{x}_2) & \kappa(\mathbf{x}_3, \mathbf{x}_3) & \kappa(\mathbf{x}_3, \mathbf{x}_4) \\ \kappa(\mathbf{x}_4, \mathbf{x}_1) & \kappa(\mathbf{x}_4, \mathbf{x}_2) & \kappa(\mathbf{x}_4, \mathbf{x}_3) & \kappa(\mathbf{x}_4, \mathbf{x}_4) \end{bmatrix} \in \mathbb{R}^{4 \times 4}$$

$$\mathbf{1}_0 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \in \mathbb{R}^{4 \times 1}$$

$$\mathbf{1}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \in \mathbb{R}^{4 \times 1}$$

**step2.** 引入一种“均值”表示

令

$$\hat{\mu}_0 = \frac{1}{m_0} \mathbf{K} \mathbf{1}_0 , \quad (6.66)$$

$$\hat{\mu}_1 = \frac{1}{m_1} \mathbf{K} \mathbf{1}_1 , \quad (6.67)$$

### 解释

方阵  $\times (0,1)$  列向量: 得到一个列向量, 每一行是方阵中, 指示向量为 1 的行对应的列的和

再看例子

$$\hat{\mu}_0 = \frac{1}{m_0} \mathbf{K} \mathbf{1}_0 = \frac{1}{2} \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) + \kappa(\mathbf{x}_1, \mathbf{x}_3) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) + \kappa(\mathbf{x}_2, \mathbf{x}_3) \\ \kappa(\mathbf{x}_3, \mathbf{x}_1) + \kappa(\mathbf{x}_3, \mathbf{x}_3) \\ \kappa(\mathbf{x}_4, \mathbf{x}_1) + \kappa(\mathbf{x}_4, \mathbf{x}_3) \end{bmatrix} \in \mathbb{R}^{4 \times 1}$$

$$\hat{\mu}_1 = \frac{1}{m_1} \mathbf{K} \mathbf{1}_1 = \frac{1}{2} \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_2) + \kappa(\mathbf{x}_1, \mathbf{x}_4) \\ \kappa(\mathbf{x}_2, \mathbf{x}_2) + \kappa(\mathbf{x}_2, \mathbf{x}_4) \\ \kappa(\mathbf{x}_3, \mathbf{x}_2) + \kappa(\mathbf{x}_3, \mathbf{x}_4) \\ \kappa(\mathbf{x}_4, \mathbf{x}_2) + \kappa(\mathbf{x}_4, \mathbf{x}_4) \end{bmatrix} \in \mathbb{R}^{4 \times 1}$$

由此扩展到一般形式

$$\hat{\mu}_0 = \frac{1}{m_0} \mathbf{K} \mathbf{1}_0 = \frac{1}{m_0} \begin{bmatrix} \sum_{\mathbf{x} \in X_0} \kappa(\mathbf{x}_1, \mathbf{x}) \\ \sum_{\mathbf{x} \in X_0} \kappa(\mathbf{x}_2, \mathbf{x}) \\ \vdots \\ \sum_{\mathbf{x} \in X_0} \kappa(\mathbf{x}_m, \mathbf{x}) \end{bmatrix} \in \mathbb{R}^{m \times 1}$$

$$\hat{\mu}_1 = \frac{1}{m_1} \mathbf{K} \mathbf{1}_1 = \frac{1}{m_1} \begin{bmatrix} \sum_{\mathbf{x} \in X_1} \kappa(\mathbf{x}_1, \mathbf{x}) \\ \sum_{\mathbf{x} \in X_1} \kappa(\mathbf{x}_2, \mathbf{x}) \\ \vdots \\ \sum_{\mathbf{x} \in X_1} \kappa(\mathbf{x}_m, \mathbf{x}) \end{bmatrix} \in \mathbb{R}^{m \times 1}$$

**step3.** 分子推导

$$\vec{w} = \sum_{i=1}^m \alpha_i \phi(\vec{x}_i) \quad (b.65)$$

学习目标:  $\max_{\vec{w}} J(\vec{w}) = \frac{\vec{w}^T S_b^\phi \vec{w}}{\vec{w}^T S_w^\phi \vec{w}}$  (b.66)

$$\text{分子 } \vec{w}^T S_b^\phi \vec{w} = \left( \sum_{i=1}^m \alpha_i \phi(\vec{x}_i) \right)^T S_b^\phi \left( \sum_{i=1}^m \alpha_i \phi(\vec{x}_i) \right)$$

$$= \sum_{i=1}^m \alpha_i \phi(\vec{x}_i)^T S_b^\phi \left( \sum_{j=1}^m \alpha_j \phi(\vec{x}_j) \right)$$

类间协方差矩阵:

$$S_b^\phi = (\mu_1^\phi - \mu_0^\phi)(\mu_1^\phi - \mu_0^\phi)^T$$

$$= \left( \frac{1}{m_1} \sum_{\vec{x} \in X_1} \phi(\vec{x}) - \frac{1}{m_0} \sum_{\vec{x} \in X_0} \phi(\vec{x}) \right) \left( \frac{1}{m_1} \sum_{\vec{x} \in X_1} \phi(\vec{x}) - \frac{1}{m_0} \sum_{\vec{x} \in X_0} \phi(\vec{x}) \right)^T$$

$$= \left( \frac{1}{m_1} \sum_{\vec{x} \in X_1} \phi(\vec{x}) - \frac{1}{m_0} \sum_{\vec{x} \in X_0} \phi(\vec{x}) \right) \left( \frac{1}{m_1} \sum_{\vec{x} \in X_1} \phi(\vec{x})^T - \frac{1}{m_0} \sum_{\vec{x} \in X_0} \phi(\vec{x})^T \right)$$

将其代入上式可得

$$\begin{aligned} \mathbf{w}^T \mathbf{S}_b^\phi \mathbf{w} &= \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i)^T \cdot \left( \frac{1}{m_1} \sum_{\mathbf{x} \in X_1} \phi(\mathbf{x}) - \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \phi(\mathbf{x}) \right) \cdot \left( \frac{1}{m_1} \sum_{\mathbf{x} \in X_1} \phi(\mathbf{x})^T - \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \phi(\mathbf{x})^T \right) \cdot \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i) \\ &= \left( \frac{1}{m_1} \sum_{\mathbf{x} \in X_1} \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) - \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) \right) \\ &\quad \cdot \left( \frac{1}{m_1} \sum_{\mathbf{x} \in X_1} \sum_{i=1}^m \alpha_i \phi(\mathbf{x})^T \phi(\mathbf{x}_i) - \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \sum_{i=1}^m \alpha_i \phi(\mathbf{x})^T \phi(\mathbf{x}_i) \right) \end{aligned}$$

由于  $\kappa(\mathbf{x}_i, \mathbf{x}) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x})$  为标量, 所以其转置等于本身, 即  $\kappa(\mathbf{x}_i, \mathbf{x}) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) = (\phi(\mathbf{x}_i)^T \phi(\mathbf{x}))^T = \phi(\mathbf{x})^T \phi(\mathbf{x}_i) = \kappa(\mathbf{x}_i, \mathbf{x})^T$ , 将其代入上式可得

$$\begin{aligned} \mathbf{w}^T \mathbf{S}_b^\phi \mathbf{w} &= \left( \frac{1}{m_1} \sum_{i=1}^m \sum_{\mathbf{x} \in X_1} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) - \frac{1}{m_0} \sum_{i=1}^m \sum_{\mathbf{x} \in X_0} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) \right) \\ &\quad \cdot \left( \frac{1}{m_1} \sum_{i=1}^m \sum_{\mathbf{x} \in X_1} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) - \frac{1}{m_0} \sum_{i=1}^m \sum_{\mathbf{x} \in X_0} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) \right) \end{aligned}$$

*K[I] 的表示*

$$\Rightarrow \vec{\mathbf{w}}^T \mathbf{S}_b^\phi \vec{\mathbf{w}} = \left( \frac{1}{m_1} \sum_{i=1}^m \sum_{\mathbf{x} \in X_1} \alpha_i \underbrace{\kappa(\vec{\mathbf{x}}_i, \vec{\mathbf{x}})}_{\text{K[I]}} - \frac{1}{m_0} \sum_{i=1}^m \sum_{\mathbf{x} \in X_0} \alpha_i \kappa(\vec{\mathbf{x}}_i, \vec{\mathbf{x}}) \right).$$

$$\begin{aligned} &\left( \frac{1}{m_1} \sum_{i=1}^m \sum_{\mathbf{x} \in X_1} \alpha_i \kappa(\vec{\mathbf{x}}_i, \vec{\mathbf{x}}) - \frac{1}{m_0} \sum_{i=1}^m \sum_{\mathbf{x} \in X_0} \alpha_i \kappa(\vec{\mathbf{x}}_i, \vec{\mathbf{x}}) \right) \\ &= \frac{1}{m_1} \left( \alpha_1 \alpha_2 \cdots \alpha_m \right) \begin{pmatrix} \Sigma K(\vec{\mathbf{x}}_1, \vec{\mathbf{x}}) \\ \Sigma K(\vec{\mathbf{x}}_2, \vec{\mathbf{x}}) \\ \vdots \\ \Sigma K(\vec{\mathbf{x}}_m, \vec{\mathbf{x}}) \end{pmatrix} \\ &= \vec{\alpha}^T \frac{1}{m_1} \hat{K}[I] = \vec{\alpha}^T \hat{M}_1 \end{aligned}$$

由前面部分是  $\vec{\alpha}$  的转置  $\Rightarrow$  转置不变

$$\Rightarrow \frac{1}{m_1} \sum_{i=1}^m \sum_{\mathbf{x} \in X_1} \alpha_i \kappa(\vec{\mathbf{x}}_i, \vec{\mathbf{x}}) = (\vec{\alpha}^T \hat{M}_1^\phi)^T = \hat{\mu}_1^T \vec{\alpha}$$

$$\begin{aligned} \Rightarrow \vec{\mathbf{w}}^T \mathbf{S}_b^\phi \vec{\mathbf{w}} &= (\vec{\alpha}^T \hat{\mu}_1 - \vec{\alpha}^T \hat{\mu}_0) \\ &\quad (\hat{\mu}_1^T \vec{\alpha} - \hat{\mu}_0^T \vec{\alpha}) \\ &= \vec{\alpha}^T (\hat{\mu}_1 - \hat{\mu}_0) (\hat{\mu}_1 - \hat{\mu}_0)^T \vec{\alpha} \\ &\quad \text{设 } \vec{\alpha} \in M. \\ &= \vec{\alpha}^T M \vec{\alpha} \end{aligned}$$

其中

$$\mathbf{M} = (\hat{\mu}_0 - \hat{\mu}_1)(\hat{\mu}_0 - \hat{\mu}_1)^T, \quad (6.68)$$

**step4.** 分母推导

$$\mathbf{w} = \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i). \quad (6.65)$$

$$\begin{aligned}\mathbf{w}^T \mathbf{S}_w^\phi \mathbf{w} &= \left( \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i) \right)^T \cdot \mathbf{S}_w^\phi \cdot \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i) \\ &= \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i)^T \cdot \mathbf{S}_w^\phi \cdot \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i)\end{aligned}$$

类间散度矩阵

其中

$$\begin{aligned}\mathbf{S}_w^\phi &= \sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} (\phi(\mathbf{x}) - \boldsymbol{\mu}_i^\phi) (\phi(\mathbf{x}) - \boldsymbol{\mu}_i^\phi)^T \\ &= \sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} (\phi(\mathbf{x}) - \boldsymbol{\mu}_i^\phi) (\phi(\mathbf{x})^T - (\boldsymbol{\mu}_i^\phi)^T) \\ &= \sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} \left( \phi(\mathbf{x}) \phi(\mathbf{x})^T - \phi(\mathbf{x}) (\boldsymbol{\mu}_i^\phi)^T - \boldsymbol{\mu}_i^\phi \phi(\mathbf{x})^T + \boldsymbol{\mu}_i^\phi (\boldsymbol{\mu}_i^\phi)^T \right) \\ &= \sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} \phi(\mathbf{x}) \phi(\mathbf{x})^T - \sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} \phi(\mathbf{x}) (\boldsymbol{\mu}_i^\phi)^T - \sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} \boldsymbol{\mu}_i^\phi \phi(\mathbf{x})^T + \sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} \boldsymbol{\mu}_i^\phi (\boldsymbol{\mu}_i^\phi)^T\end{aligned}$$

由于

$$\begin{aligned}\sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} \phi(\mathbf{x}) (\boldsymbol{\mu}_i^\phi)^T &= \sum_{\mathbf{x} \in X_0} \phi(\mathbf{x}) (\boldsymbol{\mu}_0^\phi)^T + \sum_{\mathbf{x} \in X_1} \phi(\mathbf{x}) (\boldsymbol{\mu}_1^\phi)^T \\ &= m_0 \boldsymbol{\mu}_0^\phi (\boldsymbol{\mu}_0^\phi)^T + m_1 \boldsymbol{\mu}_1^\phi (\boldsymbol{\mu}_1^\phi)^T\end{aligned}$$

且

$$\begin{aligned}\sum_{i=0}^1 \sum_{\mathbf{x} \in X_i} \boldsymbol{\mu}_i^\phi \phi(\mathbf{x})^T &= \sum_{i=0}^1 \boldsymbol{\mu}_i^\phi \sum_{\mathbf{x} \in X_i} \phi(\mathbf{x})^T \\ &= \underbrace{\boldsymbol{\mu}_0^\phi \sum_{\mathbf{x} \in X_0} \phi(\mathbf{x})^T}_{m_0 \boldsymbol{\mu}_0^\phi (\boldsymbol{\mu}_0^\phi)^T} + \underbrace{\boldsymbol{\mu}_1^\phi \sum_{\mathbf{x} \in X_1} \phi(\mathbf{x})^T}_{m_1 \boldsymbol{\mu}_1^\phi (\boldsymbol{\mu}_1^\phi)^T} \\ &= \underbrace{m_0 \boldsymbol{\mu}_0^\phi (\boldsymbol{\mu}_0^\phi)^T}_{\nwarrow \swarrow} + \underbrace{m_1 \boldsymbol{\mu}_1^\phi (\boldsymbol{\mu}_1^\phi)^T}_{\nwarrow \swarrow}\end{aligned}$$

所以

$$\begin{aligned}\mathbf{S}_w^\phi &= \sum_{\mathbf{x} \in D} \phi(\mathbf{x}) \phi(\mathbf{x})^T - 2 \left[ m_0 \boldsymbol{\mu}_0^\phi (\boldsymbol{\mu}_0^\phi)^T + m_1 \boldsymbol{\mu}_1^\phi (\boldsymbol{\mu}_1^\phi)^T \right] + m_0 \boldsymbol{\mu}_0^\phi (\boldsymbol{\mu}_0^\phi)^T + m_1 \boldsymbol{\mu}_1^\phi (\boldsymbol{\mu}_1^\phi)^T \\ &= \sum_{\mathbf{x} \in D} \phi(\mathbf{x}) \phi(\mathbf{x})^T - m_0 \boldsymbol{\mu}_0^\phi (\boldsymbol{\mu}_0^\phi)^T - m_1 \boldsymbol{\mu}_1^\phi (\boldsymbol{\mu}_1^\phi)^T\end{aligned}$$

再将此式代回  $\mathbf{w}^T \mathbf{S}_w^\phi \mathbf{w}$  可得

$$\begin{aligned}\mathbf{w}^T \mathbf{S}_w^\phi \mathbf{w} &= \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i)^T \cdot \mathbf{S}_w^\phi \cdot \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i) \\ &= \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i)^T \cdot \left( \sum_{\mathbf{x} \in D} \phi(\mathbf{x}) \phi(\mathbf{x})^T - m_0 \boldsymbol{\mu}_0^\phi (\boldsymbol{\mu}_0^\phi)^T - m_1 \boldsymbol{\mu}_1^\phi (\boldsymbol{\mu}_1^\phi)^T \right) \cdot \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i) \\ &= \sum_{i=1}^m \sum_{j=1}^m \sum_{\mathbf{x} \in D} \alpha_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) \phi(\mathbf{x})^T \alpha_j \phi(\mathbf{x}_j) - \sum_{i=1}^m \sum_{j=1}^m \alpha_i \phi(\mathbf{x}_i)^T m_0 \boldsymbol{\mu}_0^\phi (\boldsymbol{\mu}_0^\phi)^T \alpha_j \phi(\mathbf{x}_j) \\ &\quad - \sum_{i=1}^m \sum_{j=1}^m \alpha_i \phi(\mathbf{x}_i)^T m_1 \boldsymbol{\mu}_1^\phi (\boldsymbol{\mu}_1^\phi)^T \alpha_j \phi(\mathbf{x}_j)\end{aligned}$$

- 第一项

$$\begin{aligned}
 & \sum_{i=1}^m \sum_{j=1}^m \sum_{\vec{x} \in D} \alpha_i \underbrace{\phi(\vec{x}_i)^T \phi(\vec{x})}_{K(\vec{x}_i, \vec{x})} \underbrace{\phi(\vec{x})^T \alpha_j}_{\phi(\vec{x}_j) \alpha_j} \\
 & = \sum_{i=1}^m \sum_{j=1}^m \sum_{\vec{x} \in D} \alpha_i \underbrace{K(\vec{x}_i, \vec{x})}_{K_{ij}} \underbrace{K(\vec{x}_j, \vec{x}) \alpha_j}_{\alpha_j} \\
 & = \vec{\alpha}^T K K^T \vec{\alpha}
 \end{aligned}$$

$$\begin{aligned}
 & \left( \begin{array}{cccc} K_{11} & K_{12} & \cdots & K_{1m} \end{array} \right) \left( \begin{array}{c} K_{11} \\ K_{21} \\ \vdots \\ K_{m1} \end{array} \right) \\
 & = \left( \begin{array}{c} \sum_{\vec{x} \in D} K_{1x} K_{1x} \\ \vdots \\ \sum_{\vec{x} \in D} K_{mx} K_{mx} \end{array} \right)
 \end{aligned}$$

- 第二项

$$\begin{aligned}
 & \sum_{i=1}^m \sum_{j=1}^m \alpha_i \phi(\mathbf{x}_i)^T m_0 \boldsymbol{\mu}_0^\phi \left( \boldsymbol{\mu}_0^\phi \right)^T \alpha_j \phi(\mathbf{x}_j) = m_0 \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j \phi(\mathbf{x}_i)^T \boldsymbol{\mu}_0^\phi \left( \boldsymbol{\mu}_0^\phi \right)^T \phi(\mathbf{x}_j) \\
 & = m_0 \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j \phi(\mathbf{x}_i)^T \left[ \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \phi(\mathbf{x}) \right] \left[ \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \phi(\mathbf{x}) \right]^T \phi(\mathbf{x}_j) \\
 & = m_0 \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j \left[ \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) \right] \left[ \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \phi(\mathbf{x})^T \phi(\mathbf{x}_j) \right] \\
 & = m_0 \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j \left[ \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \kappa(\mathbf{x}_i, \mathbf{x}) \right] \left[ \frac{1}{m_0} \sum_{\mathbf{x} \in X_0} \kappa(\mathbf{x}_j, \mathbf{x}) \right] \\
 & = m_0 \boldsymbol{\alpha}^T \hat{\boldsymbol{\mu}}_0 \hat{\boldsymbol{\mu}}_0^T \boldsymbol{\alpha}
 \end{aligned}$$

- 第三项

$$\sum_{i=1}^m \sum_{j=1}^m \alpha_i \phi(\mathbf{x}_i)^T m_1 \boldsymbol{\mu}_1^\phi \left( \boldsymbol{\mu}_1^\phi \right)^T \alpha_j \phi(\mathbf{x}_j) = m_1 \boldsymbol{\alpha}^T \hat{\boldsymbol{\mu}}_1 \hat{\boldsymbol{\mu}}_1^T \boldsymbol{\alpha}$$

将上述三项的化简结果代回再将此式代回  $\mathbf{w}^T \mathbf{S}_b^\phi \mathbf{w}$  可得

$$\begin{aligned}
 \mathbf{w}^T \mathbf{S}_b^\phi \mathbf{w} &= \boldsymbol{\alpha}^T \mathbf{K} \mathbf{K}^T \boldsymbol{\alpha} - m_0 \boldsymbol{\alpha}^T \hat{\boldsymbol{\mu}}_0 \hat{\boldsymbol{\mu}}_0^T \boldsymbol{\alpha} - m_1 \boldsymbol{\alpha}^T \hat{\boldsymbol{\mu}}_1 \hat{\boldsymbol{\mu}}_1^T \boldsymbol{\alpha} \\
 &= \boldsymbol{\alpha}^T \cdot \left( \mathbf{K} \mathbf{K}^T - m_0 \hat{\boldsymbol{\mu}}_0 \hat{\boldsymbol{\mu}}_0^T - m_1 \hat{\boldsymbol{\mu}}_1 \hat{\boldsymbol{\mu}}_1^T \right) \cdot \boldsymbol{\alpha} \\
 &= \boldsymbol{\alpha}^T \cdot \left( \mathbf{K} \mathbf{K}^T - \sum_{i=0}^1 m_i \hat{\boldsymbol{\mu}}_i \hat{\boldsymbol{\mu}}_i^T \right) \cdot \boldsymbol{\alpha} \\
 &= \boldsymbol{\alpha}^T \mathbf{N} \boldsymbol{\alpha}
 \end{aligned}$$

其中， $\mathbf{N} = \mathbf{K} \mathbf{K}^T - \sum_{i=0}^1 m_i \hat{\boldsymbol{\mu}}_i \hat{\boldsymbol{\mu}}_i^T$ . (6.69)

⇒ 目标式化为

$$\max_{\alpha} J(\alpha) = \frac{\alpha^T \mathbf{M} \alpha}{\alpha^T \mathbf{N} \alpha}. \quad (6.70)$$

求解方法参见 3.4 节.

显然, 使用线性判别分析求解方法即可得到  $\alpha$ , 进而可由式(6.64)得到投影函数  $h(x)$ .

。。看看3.4。。

### 6.6.2. 核对数几率回归

对率损失(logistic loss):  $\ell_{log}(z) = \log(1 + \exp(-z))$ . (6.33)

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \ell_{0/1}(y_i (w^T x_i + b) - 1), \quad (6.29)$$

写成如下形式

$$\min_{w,b} \frac{1}{m} \sum_{i=1}^m \log \left( 1 + e^{-y_i (w^T x_i + b)} \right) + \frac{\lambda}{2m} \|w\|^2$$

- $\lambda$  : 调整正则项权重的正则化常数

设  $z_i = \phi(\vec{x}_i)$  是由原始空间经核函数映射到高维空间的特征向量, 则

$$\begin{aligned} & \min_{w,b} \frac{1}{m} \sum_{i=1}^m \log \left( 1 + e^{-y_i (w^T x_i + b)} \right) + \frac{\lambda}{2m} \|w\|^2 \\ & \min_{\vec{w},b} \frac{1}{m} \sum_{i=1}^m \log \left( 1 + e^{-y_i (\vec{w}^T z_i + b)} \right) + \frac{\lambda}{2m} \|\vec{w}\|^2 \end{aligned} \quad (30)$$

??

注意  $\vec{w}$  的维数发生了变化

由表示定理, 上面优化问题的解可以表示为

$$h^*(x) = \sum_{i=1}^m \alpha_i \kappa(x, x_i). \quad (6.58)$$

$$h(x) = w^T \phi(x).$$

$$w = \sum_{j=1}^m \alpha_j z_j$$

将  $w$  代入对数几率回归可得

$$\min_{w,b} \frac{1}{m} \sum_{i=1}^m \log \left( 1 + e^{-y_i (\sum_{j=1}^m \alpha_j z_j^T z_i + b)} \right) + \frac{\lambda}{2m} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j z_i^T z_j$$

用核函数  $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{z}_i^T \mathbf{z}_j = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$  替换上式中的内积运算

$$\min_{w,b} \frac{1}{m} \sum_{i=1}^m \log \left( 1 + e^{-y_i (\sum_{j=1}^m \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j) + b)} \right) + \frac{\lambda}{2m} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

解出  $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_m)$  和  $b$  后，即可得  $f(\mathbf{x}) = \sum_{i=1}^m \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i) + b$ 。

## 7. 集成学习

### 7.1. 个体与集成

集成学习/多分类器系统/基于委员会的学习：通过构建并结合多个学习器完成学习任务

#### 一般结构

产生一组“个体学习器”，用某种策略将他们结合起来

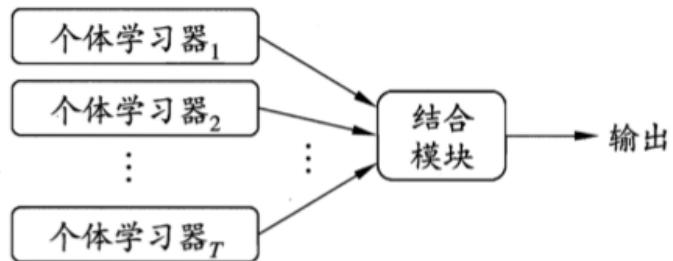


图 8.1 集成学习示意图

- 同质集成：由相同类型的个体学习器构成（基学习器；基学习算法）
- 异质集成：个体学习器（组件学习器）由不同的学习算法生成，没有基学习算法

#### 提升泛化性能

集成学习结果：少数服从多数

个体学习器应至少不差于弱学习器（分类效果略优于随机猜测）

#### 二分类例子

$y \in \{-1, +1\}$ , 真实函数  $f$

设基分类器的错误率为  $\epsilon$ , 即对每个分类器

$$P(h_i(\mathbf{x}) \neq f(\mathbf{x})) = \epsilon. \quad (8.1)$$

若集成通过简单投票法得到,

$$H(\mathbf{x}) = \text{sign} \left( \sum_{i=1}^T h_i(\mathbf{x}) \right). \quad (8.2)$$

假设基分类器的错误率相等, 由Hoeffding不等式推集成的错误率:

8.1 假设抛硬币正面朝上的概率为  $p$ , 反面朝上的概率为  $1 - p$ . 令  $H(n)$  代表抛  $n$  次硬币所得正面朝上的次数, 则最多  $k$  次正面朝上的概率为

$$P(H(n) \leq k) = \sum_{i=0}^k \binom{n}{i} p^i (1-p)^{n-i}. \quad (8.43)$$

对  $\delta > 0$ ,  $k = (p - \delta)n$ , 有 Hoeffding 不等式

$$P(H(n) \leq (p - \delta)n) \leq e^{-2\delta^2 n}. \quad (8.44)$$

试推导出式(8.3).

$T$  个基分类器, 设  $X$  为基分类器分类正确的次数, 则  $X \sim B(T, 1-\epsilon)$

设  $x_i$  为第  $i$  个分类器分类正确的次数, 则  $x_i \sim B(1, 1-\epsilon)$ ,  $i = 1, 2, 3 \dots T$

有

$X \sim B(T, 1-\epsilon)$

$$\begin{aligned} X &= \sum_{i=1}^T x_i \\ \mathbb{E}(X) &= \sum_{i=1}^T \mathbb{E}(x_i) = (1-\epsilon)T \end{aligned}$$

$$\begin{aligned} P(H(x) \neq f(x)) &= P(X < \lfloor \frac{T}{2} \rfloor) \\ &\leq P(X \leq \frac{T}{2}) \\ &= P[X - (1-\epsilon)T \leq \frac{T}{2} - (1-\epsilon)T] \\ &= P[X - (1-\epsilon)T \leq -\frac{T}{2}(1-2\epsilon)] \\ &= P[\frac{1}{T} \sum_{i=1}^T x_i - \frac{1}{T} \sum_{i=1}^T \mathbb{E}(x_i) \leq -\frac{1}{2}(1-2\epsilon)] \end{aligned}$$

由式:

$$\begin{aligned} P(H(n) \leq (p-b)n) &= P(\frac{1}{n}H(n) \leq p-b) \\ &= P(\frac{1}{n}H(n) - p \leq -b) \end{aligned}$$

$$\Rightarrow P[\frac{1}{T} \sum_{i=1}^T x_i - \frac{1}{T} \sum_{i=1}^T \mathbb{E}(x_i) \leq -b] \leq e^{-2b^2 T}$$

$$\text{令 } b = \frac{1}{2}(1-2\epsilon)$$

$$\text{得 } P(H(x) \neq f(x)) \leq \exp\{-2T \frac{1}{4}(1-2\epsilon)^2\}$$

$$\quad \quad \quad = \exp\left\{-\frac{T}{2}(1-2\epsilon)^2\right\}$$

$$\quad \quad \quad \hookrightarrow \sum_{k=0}^{\lfloor T/2 \rfloor} \binom{T}{k} (1-\epsilon)^k e^{T-k}$$

随集成中个体分类器的个数增多 ( $T$ ), 集成的错误率指教率下降, 最终趋于零

上面分析的重要假设: 基学习器的误差相互独立

事实上, 个体学习器的准确率和多样性存在冲突 (即使不同模型)

集成学习方法按照个体学习器的生成方式可以分为

- 序列化方法：个体学习器之间存在强依赖关系，必须串行生成——Boosting
- 并行化方法：个体学习器之间不存在强依赖关系，可以同时生成——Bagging、随机森林

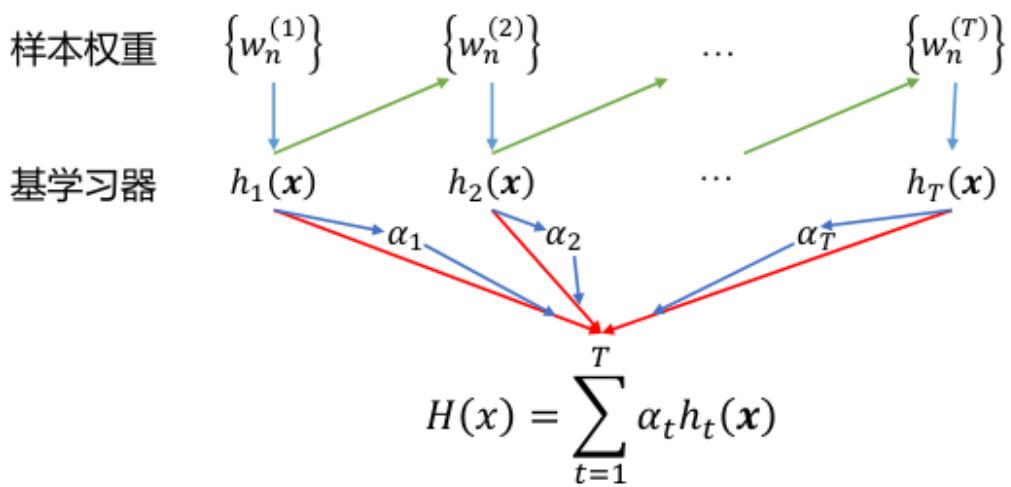
## 7.2. Boosting

Boosting：可以将弱学习器提升为强学习器的方法

### 7.2.1. 工作机制：

- 先从初始训练集训练出来一个基学习器
- 根据基学习器的表现调整训练样本的分布，使得之前基学习器做错的样本在后续受到更多关注
- 新的样本分布下训练下一个基学习器
- 重复上述步骤直到训练T个基学习器
- T个基学习器加权结合

$$D = \{(\mathbf{x}_1, y_1, w_1), (\mathbf{x}_2, y_2, w_2) \dots, (\mathbf{x}_m, y_m, w_m)\}$$



### 7.2.2. 代表算法-AdaBoost

基于加性模型的推导（基学习器的线性组合）：

$$H(\mathbf{x}) = \sum_{t=1}^T \alpha_t h_t(\mathbf{x}) \quad (8.4)$$

$H(x)$ 为连续实值函数

加性模型不采用梯度下降思想

#### 7.2.2.1. 迭代方式

$$H(x) = \sum_{t=1}^{T-1} \alpha_t h_t(x) + \alpha_T h_T(x) \quad (31)$$

共迭代T次

最小化指数损失函数

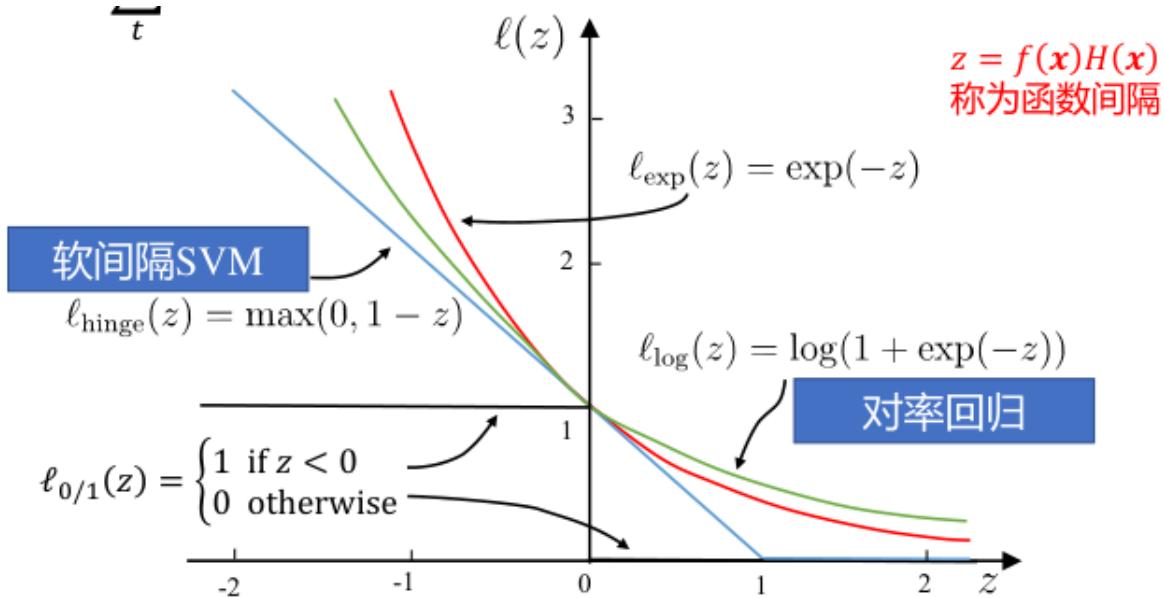
$$\ell_{\text{exp}}(H \mid \mathcal{D}) = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[e^{-f(\mathbf{x})H(\mathbf{x})}] . \quad (8.5)$$

指数损失函数

$$e^{-f(x)H(x)} \quad f \in \{-1, +1\}, H(x) \text{ 是一个实数} \quad (32)$$

指数损失对噪声敏感

$z = f(x)H(x)$  函数间隔



- $H(x)$  与  $f(x)$  同号,  $f(x)H(x) > 0$ ,  $e^{-f(x)H(x)} = e^{-|H(x)|} < 1$ , 且  $|H(x)|$  越大  $e^{-f(x)H(x)}$  越小:
  - $|H(x)|$  大则着分类器本身对预测结果的信心大, 损失小;
  - 若  $|H(x)|$  在零附近, 分类器本身对预测结果信心很小, 损失较大;
- $H(x)$  与  $f(x)$  异号,  $f(x)H(x) < 0$ ,  $e^{-f(x)H(x)} = e^{|H(x)|} > 1$ , 且  $|H(x)|$  越大  $e^{-f(x)H(x)}$  越大:
  - $|H(x)|$  大则着分类器本身对预测结果的信心大, 但结果错误, 损失应大;
  - 若  $|H(x)|$  在零附近, 分类器以较小的信息预测错误, 损失较小;

$\mathbb{E}_{x \sim \mathcal{D}}$

- $\mathcal{D}$  概率分布: 简单理解为在数据集随机抽样, 每个样本被抽到的概率
- 整体表示在概率分布  $\mathcal{D}$  上的期望, 简单理解为数据集  $D$  以概率  $\mathcal{D}$  加权得到的期望

综上所述, 若数据集  $D$  中样本  $x$  的权值分布为  $\mathcal{D}(x)$ , 则式 (8.5) 可写为:

$$\begin{aligned} \ell_{\exp}(H | \mathcal{D}) &= \mathbb{E}_{x \sim \mathcal{D}} [e^{-f(x)H(x)}] \\ &= \sum_{x \in D} \mathcal{D}(x) e^{-f(x)H(x)} \\ &= \sum_{x \in D} \mathcal{D}(x) (e^{-H(x)} \mathbb{I}(f(x) = 1) + e^{H(x)} \mathbb{I}(f(x) = -1)) \end{aligned}$$

特别地, 若针对任意样本  $x$ , 若分布  $\mathcal{D}(x) = \frac{1}{|D|}$ , 其中  $|D|$  为数据集  $D$  样本个数, 则

$$\ell_{\exp}(H | \mathcal{D}) = \mathbb{E}_{x \sim \mathcal{D}} [e^{-f(x)H(x)}] = \frac{1}{|D|} \sum_{x \in D} e^{-f(x)H(x)}$$

而这就是在求传统平均值。

### 7.2.2.2 替代损失一致性

何时  $H(x)$  使得指数损失函数最小化?

若  $H(x)$  能令指数损失函数最小化, 则考虑式(8.5)对  $H(x)$  的偏导

$$\begin{aligned}
\ell_{\exp}(H|\mathcal{D}) &= \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H(\mathbf{x})}] \\
&= \sum_{\mathbf{x} \in D} \mathcal{D}(\mathbf{x}) e^{-f(\mathbf{x})H(\mathbf{x})} \\
&= \sum_{i=1}^{|D|} \mathcal{D}(\mathbf{x}_i) (e^{-H(\mathbf{x}_i)} \mathbb{I}(f(\mathbf{x}_i) = 1) + e^{H(\mathbf{x}_i)} \mathbb{I}(f(\mathbf{x}_i) = -1)) \\
&= \sum_{i=1}^{|D|} (e^{-H(\mathbf{x}_i)} \mathcal{D}(\mathbf{x}_i) \mathbb{I}(f(\mathbf{x}_i) = 1) + e^{H(\mathbf{x}_i)} \mathcal{D}(\mathbf{x}_i) \mathbb{I}(f(\mathbf{x}_i) = -1)) \\
&= \sum_{i=1}^{|D|} (e^{-H(\mathbf{x}_i)} P(f(\mathbf{x}_i) = 1 | \mathbf{x}_i) + e^{H(\mathbf{x}_i)} P(f(\mathbf{x}_i) = -1 | \mathbf{x}_i))
\end{aligned}$$

当对  $H(x_i)$  求导时，求和号中只有含  $x_i$  项不为 0，由求导公式

$$\frac{\partial e^{-H(\mathbf{x})}}{\partial H(\mathbf{x})} = -e^{-H(\mathbf{x})} \quad \frac{\partial e^{H(\mathbf{x})}}{\partial H(\mathbf{x})} = e^{H(\mathbf{x})}$$

于是

$$\frac{\partial \ell_{\exp}(H | \mathcal{D})}{\partial H(\mathbf{x})} = -e^{-H(\mathbf{x})} P(f(\mathbf{x}) = 1 | \mathbf{x}) + e^{H(\mathbf{x})} P(f(\mathbf{x}) = -1 | \mathbf{x}), \quad (8.6)$$

## 连续情形

$$\begin{aligned}
f &= \int_X \left[ P(f(x)=1|x) \underbrace{\exp[-H(x)]}_{P(f(x)=-1|x)} + P(f(x)=-1|x) \underbrace{\exp[H(x)]}_{P(f(x)=1|x)} \right] dx \\
\frac{\partial f}{\partial H(x)} &= \underline{P(x)} \left[ -P(f(x)=1|x) e^{-H(x)} + P(f(x)=-1|x) e^{H(x)} \right]
\end{aligned}$$

令上两式为 0，有

$$H(\mathbf{x}) = \frac{1}{2} \ln \frac{P(f(\mathbf{x}) = 1 | \mathbf{x})}{P(f(\mathbf{x}) = -1 | \mathbf{x})}, \quad (8.7)$$

(1/2 对数几率)  $\ln \frac{y}{1-y}$

因此，

$$\begin{aligned}
\text{sign}(H(\mathbf{x})) &= \text{sign} \left( \frac{1}{2} \ln \frac{P(f(\mathbf{x}) = 1 | \mathbf{x})}{P(f(\mathbf{x}) = -1 | \mathbf{x})} \right) \\
\text{这里忽略了 } P(f(x) = 1 | \mathbf{x}) &= P(f(x) = -1 | \mathbf{x}) \\
\text{的情形.} &= \begin{cases} 1, & P(f(x) = 1 | \mathbf{x}) > P(f(x) = -1 | \mathbf{x}) \\ -1, & P(f(x) = 1 | \mathbf{x}) < P(f(x) = -1 | \mathbf{x}) \end{cases} \\
&= \arg \max_{y \in \{-1, 1\}} P(f(\mathbf{x}) = y | \mathbf{x}), \quad (8.8)
\end{aligned}$$

## 7.1节

## 最小化分类错误率的贝叶斯最优分类器为

$$h^*(\mathbf{x}) = \arg \max_{c \in \mathcal{Y}} P(c | \mathbf{x}), \quad (7.6)$$

(8.8)是(7.6)二分类的特殊形式

$\Rightarrow \text{sign}(H(x))$  达到贝叶斯最优错误率

即：指数损失函数最小化时，分类错误率也最小化

- 替代损失一致性：求解替代损失函数得到的仍是原问题的解

而错误率实际上对应了**0/1损失函数**（非凸，非连续）

$\ell_{0/1}$  是“0/1损失函数”

$$\ell_{0/1}(z) = \begin{cases} 1, & \text{if } z < 0; \\ 0, & \text{otherwise.} \end{cases} \quad (6.30)$$

由此：

指数损失函数是分类任务原本 **0/1 损失函数的一致的替代损失函数**

而指数损失函数有更好的数学性质，例如它是连续可微函数

因此可以以**指数损失函数**作为优化目标

### 7.2.2.3. 算法

---

输入：训练集  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ ;  
基学习算法  $\mathfrak{L}$ ;  
训练轮数  $T$ .

过程：

1:  $\mathcal{D}_1(\mathbf{x}) = 1/m$ .

2: **for**  $t = 1, 2, \dots, T$  **do**

3:    $h_t = \mathfrak{L}(D, \mathcal{D}_t)$ ;

4:    $\epsilon_t = P_{\mathbf{x} \sim \mathcal{D}_t}(h_t(\mathbf{x}) \neq f(\mathbf{x}))$ ;

5:   **if**  $\epsilon_t > 0.5$  **then break**

6:    $\alpha_t = \frac{1}{2} \ln \left( \frac{1-\epsilon_t}{\epsilon_t} \right)$ ;

7:    $\mathcal{D}_{t+1}(\mathbf{x}) = \frac{\mathcal{D}_t(\mathbf{x})}{Z_t} \times \begin{cases} \exp(-\alpha_t), & \text{if } h_t(\mathbf{x}) = f(\mathbf{x}) \\ \exp(\alpha_t), & \text{if } h_t(\mathbf{x}) \neq f(\mathbf{x}) \end{cases}$   
 $= \frac{\mathcal{D}_t(\mathbf{x}) \exp(-\alpha_t f(\mathbf{x}) h_t(\mathbf{x}))}{Z_t}$

8: **end for**

输出： $H(\mathbf{x}) = \text{sign} \left( \sum_{t=1}^T \alpha_t h_t(\mathbf{x}) \right)$

---

图 8.3 AdaBoost 算法

算法：

$$\text{若 } \epsilon_t < 0.5 \Rightarrow \frac{1-\epsilon_t}{\epsilon_t} > 1 \Rightarrow \frac{1}{2} \ln \left( \frac{1-\epsilon_t}{\epsilon_t} \right) > 0$$

$$\Rightarrow \mathcal{D}_{t+1}(\mathbf{x}): \begin{cases} h_t(\mathbf{x}) = f(\mathbf{x}) & \checkmark \Rightarrow \text{权值小} \\ h_t(\mathbf{x}) \neq f(\mathbf{x}) & \times \Rightarrow \text{权值大} \end{cases}$$

#### 7.2.2.4. 算法分析

初始分布得到  $h_1$ , 迭代生成  $h_t, \alpha_t$

$h_t$  在  $\mathcal{D}$  得到

$\alpha_t$  由  $h_t$  得到

##### 7.2.2.4.1. $\alpha_t$

$\alpha_t$  由  $h_t$  得到, 满足:

$\alpha_t h_t$  使得指数损失函数最小化, 直观想到的是最小化  $\ell_{\exp}(H_t | \mathcal{D}) = \ell_{\exp}(H_{t-1} + \alpha_t h_t | \mathcal{D})$

事实上,  $\ell_{\exp}(\alpha_t h_t | \mathcal{D}_t)$  与  $\ell_{\exp}(H_t | \mathcal{D})$  是等价的, 证明在下面

$$\begin{aligned}\ell_{\exp}(\alpha_t h_t | \mathcal{D}_t) &= \mathbb{E}_{x \sim \mathcal{D}_t} [e^{-f(x)\alpha_t h_t(x)}] \\ &= \mathbb{E}_{x \sim \mathcal{D}_t} [e^{-\alpha_t \mathbb{I}(f(x) = h_t(x))} + e^{\alpha_t} \mathbb{I}(f(x) \neq h_t(x))] \\ &= e^{-\alpha_t} P_{x \sim \mathcal{D}_t}(f(x) = h_t(x)) + e^{\alpha_t} P_{x \sim \mathcal{D}_t}(f(x) \neq h_t(x)) \\ &= e^{-\alpha_t} (1 - \epsilon_t) + e^{\alpha_t} \epsilon_t ,\end{aligned}\tag{8.9}$$

其中  $\epsilon_t = P_{x \sim \mathcal{D}_t}(h_t(x) \neq f(x))$ . 考虑指数损失函数的导数

$$\frac{\partial \ell_{\exp}(\alpha_t h_t | \mathcal{D}_t)}{\partial \alpha_t} = -e^{-\alpha_t} (1 - \epsilon_t) + e^{\alpha_t} \epsilon_t ,\tag{8.10}$$

令式(8.10)为零可解得

$$\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right) ,\tag{8.11}$$

也就是算法中权重的更新公式

- 最小化等价性的证明

##### 7.2.2.4.2. $h_t$

- 算法希望:

获得  $H_{t-1}$  之后调整样本分布, 使得  $h_t$  可以纠正  $H_{t-1}$  的一些错误 (理想情况下纠正所有错误)

基于此, 应该最小化

$$\begin{aligned}\ell_{\exp}(H_{t-1} + h_t | \mathcal{D}) &= \mathbb{E}_{x \sim \mathcal{D}} [e^{-f(x)(H_{t-1}(x) + h_t(x))}] \\ &= \mathbb{E}_{x \sim \mathcal{D}} [e^{-f(x)H_{t-1}(x)} e^{-f(x)h_t(x)}] .\end{aligned}\tag{8.12}$$

tips 这里原论文似乎是最大化 (对任意  $\alpha > 0$ )

$$(\alpha_t, h_t(x)) = \arg \min_{\alpha, h} \ell_{\exp}(H_{t-1} + \alpha h | \mathcal{D})$$

课本相当于指定 $\alpha = 1$ , 不影响后续结果

go on

注意到  $f^2(\mathbf{x}) = h_t^2(\mathbf{x}) = 1$ , 式(8.12)可使用  $e^{-f(\mathbf{x})h_t(\mathbf{x})}$  的泰勒展式近似为

$$\begin{aligned}\ell_{\text{exp}}(H_{t-1} + h_t \mid \mathcal{D}) &\simeq \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})} \left( 1 - f(\mathbf{x})h_t(\mathbf{x}) + \frac{f^2(\mathbf{x})h_t^2(\mathbf{x})}{2} \right) \right] \\ &= \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})} \left( 1 - f(\mathbf{x})h_t(\mathbf{x}) + \frac{1}{2} \right) \right]. \quad (8.13)\end{aligned}$$

于是, 理想的基学习器

$$\begin{aligned}h_t(\mathbf{x}) &= \arg \min_h \ell_{\text{exp}}(H_{t-1} + h \mid \mathcal{D}) \\ &= \arg \min_h \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})} \left( 1 - f(\mathbf{x})h(\mathbf{x}) + \frac{1}{2} \right) \right] \\ &= \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ \frac{3}{2} e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})} - e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})} f(\mathbf{x})h(\mathbf{x}) \right] \\ &= \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ \frac{3}{2} e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})} \right] - \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})} f(\mathbf{x})h(\mathbf{x})]\end{aligned}$$

上式自变量为  $h(x)$  ( $H_{t-1}$  是已知的, 因此  $\mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}]$  是一个常数)

于是等价于一个最大化问题

$$\begin{aligned}&= \arg \max_h \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})} f(\mathbf{x})h(\mathbf{x})] \\ &= \arg \max_h \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ \frac{e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}}{\mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}]} f(\mathbf{x})h(\mathbf{x}) \right]\end{aligned}$$

令  $\mathcal{D}_t$  表示一个分布

$$\mathcal{D}_t(\mathbf{x}) = \frac{\mathcal{D}(\mathbf{x})e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}}{\mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}]}, \quad (8.15)$$

又有

$$\mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H(\mathbf{x})}] = \sum_{i=1}^{|\mathcal{D}|} \mathcal{D}(\mathbf{x}_i) e^{-f(\mathbf{x}_i)H(\mathbf{x}_i)}$$

其中由 (8.15)

$$\mathcal{D}_t(\mathbf{x}_i) = \mathcal{D}(\mathbf{x}_i) \frac{e^{-f(\mathbf{x}_i)H_{t-1}(\mathbf{x}_i)}}{\mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}]}$$

于是

$$\begin{aligned}
& \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ \frac{e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}}{\mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}] f(\mathbf{x}) h(\mathbf{x})} \right] \\
&= \sum_{i=1}^{|D|} \mathcal{D}(\mathbf{x}_i) \frac{e^{-f(\mathbf{x}_i)H_{t-1}(\mathbf{x}_i)}}{\mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}] f(x_i) h(x_i)} \\
&= \sum_{i=1}^{|D|} \mathcal{D}_t(\mathbf{x}_i) f(\mathbf{x}_i) h(\mathbf{x}_i) \\
&= \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_t} [f(\mathbf{x}) h(\mathbf{x})]
\end{aligned}$$

即

$$\begin{aligned}
h_t(\mathbf{x}) &= \arg \max_h \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ \frac{e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}}{\mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [e^{-f(\mathbf{x})H_{t-1}(\mathbf{x})}] f(\mathbf{x}) h(\mathbf{x})} \right] \\
&= \arg \max_h \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_t} [f(\mathbf{x}) h(\mathbf{x})] . \tag{8.16}
\end{aligned}$$

当  $f(\mathbf{x}) = h(\mathbf{x})$  时,  $\mathbb{I}(f(\mathbf{x}) \neq h(\mathbf{x})) = 0$ ,  $f(\mathbf{x}) h(\mathbf{x}) = 1$ ,  $1 - 2\mathbb{I}(f(\mathbf{x}) \neq h(\mathbf{x})) = 1$ ;

当  $f(\mathbf{x}) \neq h(\mathbf{x})$  时,  $\mathbb{I}(f(\mathbf{x}) \neq h(\mathbf{x})) = 1$ ,  $f(\mathbf{x}) h(\mathbf{x}) = -1$ ,  $1 - 2\mathbb{I}(f(\mathbf{x}) \neq h(\mathbf{x})) = -1$ 。

于是

$$f(\mathbf{x}) h(\mathbf{x}) = 1 - 2\mathbb{I}(f(\mathbf{x}) \neq h(\mathbf{x})) , \tag{8.17}$$

则理想的基学习器

$$h_t(\mathbf{x}) = \arg \min_h \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_t} [\mathbb{I}(f(\mathbf{x}) \neq h(\mathbf{x}))] . \tag{8.18}$$

$\Rightarrow$ 理想的  $h_t$  应在分布  $\mathcal{D}_t$  下最小化分类错误率

即算法中的  $h_t(\mathbf{x}) = \mathfrak{L}(D, \mathcal{D}_t)$ , 要求分类误差小于  $1/2$

PPT

- 假设  $f(\mathbf{x})$  是确定性的，在第  $t$  步，优化以下损失函数

$$\begin{aligned}
 \ell(H_{t-1} + \alpha_t h_t | D) &= \mathbb{E}_{\mathbf{x}} \left[ \exp \left( -f(\mathbf{x})(H_{t-1}(\mathbf{x}) + \alpha_t h_t(\mathbf{x})) \right) \right] \\
 &= \mathbb{E}_{\mathbf{x}} \left[ \exp(-f(\mathbf{x}) H_{t-1}(\mathbf{x})) \exp(-f(\mathbf{x}) \alpha_t h_t(\mathbf{x})) \right] \\
 D = \{(\mathbf{x}_1, y_1, w_1), \dots, (\mathbf{x}_m, y_m, w_m)\} \\
 \text{样本权重} \quad \{w_n^{(t)}\} &\quad \mathbb{E}_{\mathbf{x}} \left[ \bar{w}_t(\mathbf{x}) \exp(-f(\mathbf{x}) \alpha_t h_t(\mathbf{x})) \right] \\
 \text{基学习器} \quad h_t(\mathbf{x}) &\quad = \mathbb{E}_{\mathbf{x}} \left[ \bar{w}_t(\mathbf{x}) \exp(-\alpha_t) \mathbb{I}(f(\mathbf{x}) = h_t(\mathbf{x})) \right. \\
 &\quad \left. + \bar{w}_t(\mathbf{x}) \exp(\alpha_t) \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x})) \right] \\
 &= \exp(-\alpha_t) \mathbb{E}_{\mathbf{x}} \left[ \bar{w}_t(\mathbf{x}) \mathbb{I}(f(\mathbf{x}) = h_t(\mathbf{x})) \right] \\
 &\quad + \exp(\alpha_t) \mathbb{E}_{\mathbf{x}} \left[ \bar{w}_t(\mathbf{x}) \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x})) \right] \\
 H_t(\mathbf{x}) &= \alpha_t h_t(\mathbf{x}) + H_{t-1}(\mathbf{x})
 \end{aligned}$$

$$\begin{aligned}
 \ell(H_{t-1} + \alpha_t h_t | D) &= \exp(-\alpha_t) \mathbb{E}_{\mathbf{x}} [\bar{w}_t(\mathbf{x}) \mathbb{I}(f(\mathbf{x}) = h_t(\mathbf{x}))] + \exp(\alpha_t) \mathbb{E}_{\mathbf{x}} [\bar{w}_t(\mathbf{x}) \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x}))] \\
 \min_{h_t} \ell(H_{t-1} + \alpha_t h_t | D) &= \exp(-\alpha_t) \mathbb{E}_{\mathbf{x}} \left[ \bar{w}_t(\mathbf{x}) \left( 1 - \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x})) \right) \right] \\
 &\quad + \exp(\alpha_t) \mathbb{E}_{\mathbf{x}} [\bar{w}_t(\mathbf{x}) \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x}))]
 \end{aligned}$$

$$D = \{(\mathbf{x}_1, y_1, w_1), \dots, (\mathbf{x}_m, y_m, w_m)\}$$

$$\begin{aligned}
 \text{样本权重} \quad \{w_n^{(t)}\} &\quad = (\exp(\alpha_t) - \exp(-\alpha_t)) \mathbb{E}_{\mathbf{x}} [\bar{w}_t(\mathbf{x}) \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x}))] \\
 &\quad + \exp(-\alpha_t) \mathbb{E}_{\mathbf{x}} [\bar{w}_t(\mathbf{x})] \\
 \text{基学习器} \quad h_t(\mathbf{x}) &\quad \text{当 } \alpha_t > 0, \exp(\alpha_t) - \exp(-\alpha_t) > 0 \\
 &\quad \min_{h_t} \ell(H_{t-1} + \alpha_t h_t | D) \text{ 等价于 } \min_{h_t} \mathbb{E}_{\mathbf{x}} [\bar{w}_t(\mathbf{x}) \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x}))] \\
 &\quad \bar{w}_t(\mathbf{x}) = \exp(-f(\mathbf{x}) H_{t-1}(\mathbf{x}))
 \end{aligned}$$

$\alpha > 0$ , 最小化  $h_t(\mathbf{x})$  等价于 最小化 加权损失和 ( $\bar{w}_t$  和 不为 1)

希望权重不要显示出现：

$$\begin{aligned}
 h_t^*(\mathbf{x}) &= \operatorname{argmin}_{h_t} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\bar{w}_t(\mathbf{x}) \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x}))] \\
 &= \operatorname{argmin}_{h_t} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[ \frac{\bar{w}_t(\mathbf{x})}{\mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\bar{w}_t(\mathbf{x})]} \mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x})) \right] = \operatorname{argmin}_{h_t} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_t} [\mathbb{I}(f(\mathbf{x}) \neq h_t(\mathbf{x}))]
 \end{aligned}$$

$$\begin{aligned}
D_t(x) &= D(x) \frac{\bar{w}_t(x)}{\mathbb{E}_{x \sim D}[\bar{w}_t(x)]} = D(x) \frac{\exp(-f(x)H_{t-1}(x))}{\mathbb{E}_{x \sim D}[\exp(-f(x)H_{t-1}(x))]} \\
&= D(x) \frac{\exp(-f(x)H_{t-2}(x)) \exp(-\alpha_{t-1}f(x)h_{t-1}(x))}{\mathbb{E}_{x \sim D}[\exp(-f(x)H_{t-1}(x))]} \\
&= D_{t-1}(x) \exp(-\alpha_{t-1}f(x)h_{t-1}(x)) \frac{\mathbb{E}_{x \sim D}[\exp(-f(x)H_{t-2}(x))]}{\mathbb{E}_{x \sim D}[\exp(-f(x)H_{t-1}(x))]} \\
&= \frac{D_{t-1}(x) \exp(-\alpha_{t-1}f(x)h_{t-1}(x))}{Z_{t-1}}
\end{aligned}$$

#### 7.2.2.4.3. 更新分布

- 考虑  $D_t$  和  $D_{t+1}$  的关系

类比(8.15)

(8.19)

$$\begin{aligned}
D_{t+1}(x) &= \frac{D(x) e^{-f(x)H_t(x)}}{\mathbb{E}_{x \sim D}[e^{-f(x)H_t(x)}]} \\
&= \frac{D(x) e^{-f(x)[H_{t-1}(x) + \alpha_t h_t(x)]}}{\mathbb{E}_{x \sim D}[e^{-f(x)H_t(x)}]} \\
&= \frac{D(x) e^{-f(x)H_{t-1}(x)} e^{-f(x)\alpha_t h_t(x)}}{\mathbb{E}_{x \sim D}[e^{-f(x)H_t(x)}]}
\end{aligned}$$

$$= D_t(x) \mathbb{E}_{x \sim D}[e^{-f(x)H_{t-1}(x)}] \frac{e^{-f(x)\alpha_t h_t(x)}}{\mathbb{E}_{x \sim D}[e^{-f(x)H_t(x)}]}$$

$$= D_t(x) \cdot \frac{\mathbb{E}_{x \sim D}[e^{-f(x)H_{t-1}(x)}]}{\mathbb{E}_{x \sim D}[e^{-f(x)H_t(x)}]} \quad \text{括号部分用红圈圈出}$$

$$D_t(x) = \frac{D(x) e^{-f(x)H_{t-1}(x)}}{\mathbb{E}_{x \sim D}[e^{-f(x)H_t(x)}]}$$

$$H_t(x) = H_{t-1}(x) + \alpha_t h_t(x)$$

对  $D_{t+1}(x)$  规范化:  $Z_t$ . 范数化时会乘以  $\alpha_t$

$\Rightarrow$  得到算法规则

$$D_{t+1}(x) = \frac{D_t(x)}{Z_t} e^{-f(x)\alpha_t h_t(x)}$$

#### 7.2.3. 总结

重赋权法/重采样法

Boosting 算法需要在训练的每一轮要检查当前生成的基学习器是否比随即猜测好，不满足则停止学习过程。若基学习器不接受带权重样本-重采样法-根据样本分布重新采样：可能获得重启动机会避免训练过程过早停止；抛弃不满足条件的当前基学习器，根据当前分布重新对样本进行采样，基于新的采样结果重新训练。从偏差-方差分解的角度看，Boosting 主要关注降低偏差，（更好的拟合），可以基于泛化性能差的学习器构建出很强的集成；但方差可能变大（如对不同数据集分布输出不同）。

- 偏差：预测均值与真实值的差

限制：分类问题，指数损失

#### 7.2.4. 梯度提升树 Gradient Boosting GDBT

- 使用CART回归树模型
- 基于残差学习，没有样本权重概念

看PPT吧。。

Adaboost	Gradient Boosting
$H(\mathbf{x}) = \sum_t^T \alpha_t h_t(\mathbf{x})$	$H(\mathbf{x}) = \sum_t^T h_t(\mathbf{x})$
先训练学习器再学习权重	同时学习权重和学习器
$\min_{H(\mathbf{x})} \mathbb{E}_{\mathbf{x}, y} [\exp(-y H(\mathbf{x}))]$	$\min_{H(\mathbf{x})} \mathbb{E}_{\mathbf{x}, y} [\ell(y, H(\mathbf{x}))]$
适用于分类	适用于回归和分类

- 与Adaboost一样，通过前向分布法优化  $\min_{H(\mathbf{x})} \mathbb{E}_{\mathbf{x}, y} [\ell(y, H(\mathbf{x}))]$

将  $H(\mathbf{x})$  看成一个参数（泛函）， $H(\mathbf{x})$  的学习过程看成是一个梯度下降过程

看成梯度

$$H_t(\mathbf{x}) = H_{t-1}(\mathbf{x}) + h_t(\mathbf{x})$$

关于第2项的  
导数在  $H_{t-1}$  的值

负梯度

$$h_t(\mathbf{x}) \approx -\frac{d\ell(y, \hat{y})}{d\hat{y}} \Big|_{\hat{y}=H_{t-1}(\mathbf{x})} = -\ell'(y, H_{t-1}(\mathbf{x}))$$


---

$$\ell(y, H(\mathbf{x})) = (y - H(\mathbf{x}))^2$$

$$-\ell'(y, H_{t-1}(\mathbf{x})) = y - H_{t-1}(\mathbf{x})$$

残差

## • 梯度提升树的算法流程

每个样本关于预测值的导数

梯度

- (1) 初始化  $h_0(\mathbf{x}) = \operatorname{argmin}_{\gamma} \sum_{i=1}^m \ell(y_i, \gamma)$
- (2) 对  $t=1$  to  $T$ 
  - (a) 计算负梯度:  $\tilde{y}_i = -\ell'(y_i, H_{t-1}(\mathbf{x}_i))$ ,  $i \in \{1, 2, \dots, m\}$
  - (b) 通过最小化平方误差, 用基学习器  $h_t(\mathbf{x})$  根据  $\mathbf{x}_i$  拟合  $\tilde{y}_i$
  - (c) 使用线搜索确定步长  $\rho_t$ ,  $\rho_t = \operatorname{argmin}_{\rho_t} \sum_{i=1}^m \ell(y_i, H_{t-1}(\mathbf{x}) + \rho_t h_t(\mathbf{x}))$
  - (d) 更新  $H_t(\mathbf{x}) = H_{t-1}(\mathbf{x}) + \rho_t h_t(\mathbf{x})$
- (3) 输出  $H_T(\mathbf{x})$

容易陷入局部最优

- 回忆: 回归树将空间划分为  $M$  个区域  $R_1, R_2, \dots, R_M$ , 那么回归树的决策函数为

$$h(\mathbf{x}) = \sum_{c=1}^C w_c I(\mathbf{x} \in R_c)$$

- 如果优化平方损失, 在给定区域划分情况下, 最优的  $c_m$  值是对应区域内的平均值

$c_m$  为划分值

任意损失, 如何确定  $w_c$ ?

## 7.3. Bagging与随机森林

为了得到泛化能力强的集成, 集成中的个体学习器应尽可能相互独立; 现实任务中设法使基学习器尽量有较大差异——使用不同的训练样本集, 同时所选样本集应保持训练的有效性

### 7.3.1. Bagging

Bootstrap AGGREGATING

#### 7.3.1.1. 自主采样法

bootstrap sampling

在大小为  $m$  的样本集上进行  $m$  次有放回的取样

由 (2.1) 知, 初始训练集约有 63.2% 的样本出现在采样集中

#### 7.3.1.2. Bagging 算法

得到  $T$  个含  $m$  个训练样本的采样集, 基于每个采样集训练出一个基学习器, 再将基学习器结合

结合方法:

- 分类任务使用简单投票法
- 回归任务使用简单平均法

$\mathcal{D}_{bs}$  是自助采样产生的样本分布.

---

输入: 训练集  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ ;  
基学习算法  $\mathfrak{L}$ ;  
训练轮数  $T$ .  
过程:  
1: **for**  $t = 1, 2, \dots, T$  **do**  
2:    $h_t = \mathfrak{L}(D, \mathcal{D}_{bs})$   
3: **end for**  
输出:  $H(\mathbf{x}) = \arg \max_{y \in \mathcal{Y}} \sum_{t=1}^T \mathbb{I}(h_t(\mathbf{x}) = y)$

---

图 8.5 Bagging 算法

## 时间复杂度

设基学习器的计算复杂度为  $O(m)$ , 采样+投票/平均过程复杂度约为  $O(s)$ , 很小, 训练个数  $T$  是个不太大的常数, 因此  $T(O(m) + O(s))$  与  $O(m)$  同阶, 即训练一个 **Bagging** 集成与直接使用基学习算法训练一个学习器复杂度同阶

### 7.3.1.3. 对比AdaBoost

- 标准AdaBoost只适用二分类任务, 处理指数损失函数
- Bagging适用多分类、回归等任务, 适用任何损失函数

### 7.3.1.4. 评价

#### 7.3.1.4.1. 包外估计

因此Bagging的一个优点是: 每个基学习器只使用了初始训练集中约63.2%的样本, 剩下约36.8%的样本可以用作验证集对泛化性能进行“包外估计”(2.2.3)

为此需要记录每个基学习器使用的训练样本

不妨令  $D_t$  表示  $h_t$  实际使用的训练样本集, 令  $H^{oob}(\mathbf{x})$  表示对样本  $\mathbf{x}$  的包外预测, 即仅考虑那些未使用  $\mathbf{x}$  训练的基学习器在  $\mathbf{x}$  上的预测, 有

$$H^{oob}(\mathbf{x}) = \arg \max_{y \in \mathcal{Y}} \sum_{t=1}^T \mathbb{I}(h_t(\mathbf{x}) = y) \cdot \mathbb{I}(\mathbf{x} \notin D_t), \quad (8.20)$$

上式对包外的数据用“投票法”选择结果(1或-1)

则 Bagging 泛化误差的包外估计为

$$\epsilon^{oob} = \frac{1}{|D|} \sum_{(\mathbf{x}, y) \in D} \mathbb{I}(H^{oob}(\mathbf{x}) \neq y). \quad (8.21)$$

(此处假设  $T$  个基分类器的各自的包外样本的并集一定为训练集  $D$ )

$T$  次独立的随机采样均属于包内的概率为  $0.632^T$

事实上, 包外样本还有许多其他用途. 例如当基学习器是决策树时, 可使用包外样本来辅助剪枝, 或用于估计决策树中各结点的后验概率以辅助对零训练样本结点的处理; 当基学习器是神经网络时, 可使用包外样本来辅助早期停止以减小过拟合风险.

#### 7.3.1.4.2. 偏差-方差

从偏差-方差分解角度看，Bagging主要关注降低方差，在不剪枝决策树、神经网络（**他们容易过拟合**）等易受样本扰动的学习器上效果更明显

**决策桩/朴素贝叶斯分类器作为bagging的基学习器时，效果较差：**

这两个都是高偏差低方差的模型

- 方差大（偏差低）的模型往往是因为**对训练数据拟合得过好，模型比较复杂**，输入数据的一点点变动都会导致输出结果有较大的差异，它描述的是模型输出的预测值相比于真实值的离散程度，方差越大，越离散，所以为什么Bagging适合以不剪枝决策树、神经网络这些**容易过拟合**的模型为基学习器；
- 偏差大（方差低）的模型则相反，往往因为**对训练数据拟合得不够，模型比较简单**，输入数据发生变化并不会导致输出结果有多大改变，它描述的是预测值和真实值直接的差距，偏差越大，越偏离真实值。

#### 7.3.2. 随机森林

是Bagging的一个扩展变体，在以决策树为基学习器构建Bagging集成的基础上，进一步在决策树的训练中引入随机属性选择。

- 传统决策树：在当前结点的属性集合（假设有d个）选择一个最优属性
- RF
  - 1. 对基决策树的每个结点，先从结点的属性集合中随机选择一个包含k个属性的子集
  - 2. 从子集中选择一个最优属性用于划分

**参数k：**控制随机性的引入程度

- 若k=d，即为传统决策树
- 若k=1，即为随机选择属性
- 推荐值  $k = \log_a d$

#### 7.3.2.1. 对比Bagging

- Bagging的多样性仅来自于样本扰动（采样）
- 随机森林中的基学习器的多样性来自**样本扰动和自属性扰动**
- 收敛性类似
  - 随机森林的起始性能往往较差（包含基学习器少，属性扰动降低性能）
  - 随机森林的训练效率常优于Bagging

因为选择属性时不再需要考察结点的**所有属性**，只需要考察**一个属性子集**

### 7.4. 结合策略

#### 7.4.1. 学习器结合优点

- 统计方面：学习任务的假设空间很大，可能存在多个在训练集上达到同等性能的假设，单学习器容易误选导致泛化性能不佳，结合多学习器降低误选风险
- 计算方面：学习算法往往陷入局部极小（可能泛化性能很糟），多次运行结合可以降低陷入糟糕局部极小点的风险
- 表示方面：学习任务的真实假设可能不在当前考虑的假设空间中，单学习器无效；多学习器结合使得相应的假设空间有所扩大，学得更好的近似

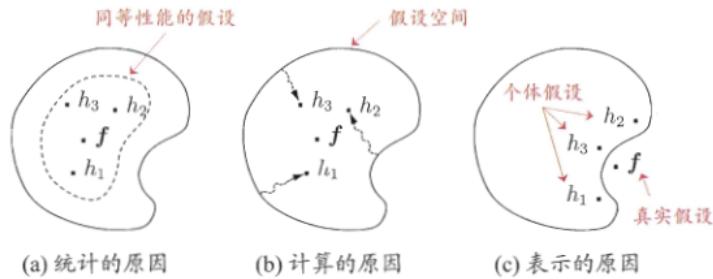


图 8.8 学习器结合可能从三个方面带来好处 [Dietterich, 2000]

### 7.4.2. 平均法

- 简单平均法

$$H(\mathbf{x}) = \frac{1}{T} \sum_{i=1}^T h_i(\mathbf{x}). \quad (8.22)$$

- 加权平均法

$$H(\mathbf{x}) = \sum_{i=1}^T w_i h_i(\mathbf{x}). \quad (8.23)$$

其中  $w_i$  是个体学习器  $h_i$  的权重, 通常要求  $w_i \geq 0$ ,  $\sum_{i=1}^T w_i = 1$ .

集成学习中的各种结合方法都可视为其特例或变体 (变化在于如何确认权重)

#### 加权平均未必优于简单平均

例如估计出个体学习器的误差, 然后令权重大小与误差大小成反比.

加权平均法的权重一般是从训练数据中学习而得, 现实任务中的训练样本通常不充分或存在噪声, 这将使得学出的权重不完全可靠. 尤其是对规模比较大的集成来说, 要学习的权重比较多, 较容易导致过拟合. 因此, 实验和应用均显示出, 加权平均法未必一定优于简单平均法 [Xu et al., 1992; Ho et al., 1994; Kittler et al., 1998]. 一般而言, 在个体学习器性能相差较大时宜使用加权平均法, 而在个体学习器性能相近时宜使用简单平均法.

### 7.4.3. 投票法

对分类任务来说, 学习器  $h_i$  将从类别标记集合  $\{c_1, c_2, \dots, c_N\}$  中预测出一个标记, 最常见的结合策略是使用投票法(voting). 为便于讨论, 我们将  $h_i$  在样本  $\mathbf{x}$  上的预测输出表示为一个  $N$  维向量  $(h_i^1(\mathbf{x}); h_i^2(\mathbf{x}); \dots; h_i^N(\mathbf{x}))$ , 其中  $h_i^j(\mathbf{x})$  是  $h_i$  在类别标记  $c_j$  上的输出.

- 绝对多数投票法(majority voting)

$$H(\mathbf{x}) = \begin{cases} c_j, & \text{if } \sum_{i=1}^T h_i^j(\mathbf{x}) > 0.5 \sum_{k=1}^N \sum_{i=1}^T h_i^k(\mathbf{x}); \\ \text{reject}, & \text{otherwise.} \end{cases} \quad (8.24)$$

即若某标记得票过半数, 则预测为该标记; 否则拒绝预测.

若学习任务要求必须提供结果，则绝对多数投票退化为相对多数投票。

- 相对多数投票法(plurality voting)

$$H(\mathbf{x}) = c_{\arg \max_j \sum_{i=1}^T h_i^j(\mathbf{x})}. \quad (8.25)$$

即预测为得票最多的标记，若同时有多个标记获最高票，则从中随机选取一个。

- 加权投票法(weighted voting)

$$H(\mathbf{x}) = c_{\arg \max_j \sum_{i=1}^T w_i h_i^j(\mathbf{x})}. \quad (8.26)$$

与加权平均法类似， $w_i$  是  $h_i$  的权重，通常  $w_i \geq 0$ ,  $\sum_{i=1}^T w_i = 1$ .

式(8.24)~(8.26)没有限制个体学习器输出值的类型。在现实任务中，不同类型个体学习器可能产生不同类型的  $h_i^j(\mathbf{x})$  值，常见的有：

- 类标记:  $h_i^j(\mathbf{x}) \in \{0, 1\}$ , 若  $h_i$  将样本  $\mathbf{x}$  预测为类别  $c_j$  则取值为 1, 否则为 0. 使用类标记的投票亦称“硬投票”(hard voting).
- 类概率:  $h_i^j(\mathbf{x}) \in [0, 1]$ , 相当于对后验概率  $P(c_j | \mathbf{x})$  的一个估计. 使用类概率的投票亦称“软投票”(soft voting).

虽然分类器估计出的类概率一般不太准，但基于类概率进行结合往往优于直接基于类标记进行比较的性能  
注意事项

- 不同类型的  $h_i^j(\mathbf{x})$  不能混用
- 若预测类别的同时输出分类置信度，分类置信度可以转化为类概率使用  
但需要规范化处理，如：
  - 支持向量机的分类间隔值必须使用如Patt缩放等进行校准才能作为类概率
- 不同类型的学习器类概率值不能直接比较，通常可将类概率输出转化为类标记输出（如类概率最大的类设为1）

#### 7.4.4. 学习法

通过另一个学习器（次级学习器/元学习器）进行结合

个体学习器：初级学习器

典型代表：**Stacking**

#### 7.4.4.1. Stacking

##### 算法过程

1. 从初始训练集训练出初级学习器（同质/异质）

2. 生成一个新数据集用于训练次级学习器

若直接使用初级学习器的训练集来产生次级训练集，过拟合风险较大

一般采用交叉验证或留一法，用初级学习器未使用的样本生成次级学习器的训练样本

**eg.** *k*折交叉验证中生成训练样本的方法

初始训练集随机划分为*k*个大小相似的集合 $D_1 \dots D_k$

$D_j$ 表示第*j*折的测试集

$\tilde{D}_j = D - D_j$ 表示第*j*折的训练集

**训练集.** 给定  $T$  个初级学习算法，初级学习器  $h_t^{(j)}$  通过在  $\tilde{D}_j$  上使用第  $t$  个学习算法而得。对  $D_j$  中每个样本  $\mathbf{x}_i$ ，令  $z_{it} = h_t^{(j)}(\mathbf{x}_i)$ ，则由  $\mathbf{x}_i$  所产生的次级训练样例的示例部分为  $\mathbf{z}_i = (z_{i1}; z_{i2}; \dots; z_{iT})$ ，标记部分为  $y_i$ 。于是，在整个交叉  $h_t^j$  的训练没有用到  $D_j$  中样本，用  $T$  个不同方法得到  $T$  个这样的学习器，他们共同生成一个基于  $D_j$  中样本  $x_i$  的新样本

也就是说：对于  $D$  中的某个样本，考虑所有没有用这个样本训练的  $T$  个学习器（他们属于同一折），这些学习器共同生成该样本对应的新样本，最后得到的新的训练集大小与  $D$  相同

每一折对进行上述操作，

验证过程结束后，从这  $T$  个初级学习器产生的次级训练集是  $D' = \{(\mathbf{z}_i, y_i)\}_{i=1}^m$

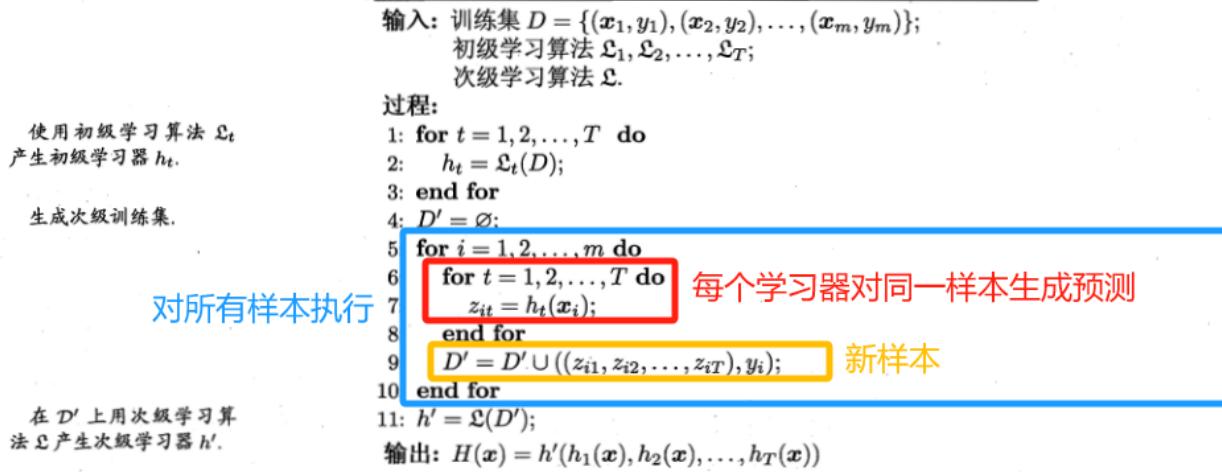


图 8.9 Stacking 算法

? 交叉验证这里有点晕

MLR 是基于线性回归的分类器，它对每个类分别进行线性回归，属于该类的训练样例所对应的输出被置为 1，其他类置为 0；测试示例将被分给输出值最大的类。

WEKA 中的 StackingC 算法就是这样实现的。

次级学习器的输入属性表示和次级学习算法对 Stacking 集成的泛化性能有很大影响。有研究表明，将初级学习器的输出类概率作为次级学习器的输入属性，用多响应线性回归(Multi-response Linear Regression, 简称 MLR) 作为次级学习算法效果较好 [Ting and Witten, 1999]，在 MLR 中使用不同的属性集更佳 [Seewald, 2002]。

#### 7.4.5. 贝叶斯模型平均 (BMA)

贝叶斯模型平均(Bayes Model Averaging, 简称 BMA)基于后验概率来为不同模型赋予权重, 可视为加权平均法的一种特殊实现. [Clarke, 2003] 对 Stacking 和 BMA 进行了比较. 理论上来说, 若数据生成模型恰在当前考虑的模型中, 且数据噪声很少, 则 BMA 不差于 Stacking; 然而, 在现实应用中无法确保数据生成模型一定在当前考虑的模型中, 甚至可能难以用当前考虑的模型来进行近似, 因此, Stacking 通常优于 BMA, 因为其鲁棒性比 BMA 更好, 而且 BMA 对模型近似误差非常敏感.

### 7.5. 多样性

#### 7.5.1. 误差-分歧分解

- 个体学习器 $h_i$ 的分歧

$$A(h_i | \mathbf{x}) = (h_i(\mathbf{x}) - H(\mathbf{x}))^2$$

表示个体学习器结果与预测结果的差值的平方, 即为个体学习器的“分歧”。

描绘多样性

- 集成的分歧

$$\begin{aligned}\overline{A}(h | \mathbf{x}) &= \sum_{i=1}^T w_i A(h_i | \mathbf{x}) \\ &= \sum_{i=1}^T w_i (h_i(\mathbf{x}) - H(\mathbf{x}))^2.\end{aligned}\quad (8.28)$$

各个个体学习器的“分歧”加权平均的结果

- 个体学习器的平方误差

$$E(h_i | \mathbf{x}) = (f(\mathbf{x}) - h_i(\mathbf{x}))^2, \quad (8.29)$$

个体学习器与真实值之间差值的平方, 描绘准确率

- 集成的平方误差

$$E(H | \mathbf{x}) = (f(\mathbf{x}) - H(\mathbf{x}))^2. \quad (8.30)$$

由 (8.28) 知

$$\begin{aligned}
 \bar{A}(h|\mathbf{x}) &= \sum_{i=1}^T w_i (h_i(\mathbf{x}) - H(\mathbf{x}))^2 \\
 &= \sum_{i=1}^T w_i (h_i(\mathbf{x})^2 - 2h_i(\mathbf{x})H(\mathbf{x}) + H(\mathbf{x})^2) \\
 &= \sum_{i=1}^T w_i h_i(\mathbf{x})^2 - H(\mathbf{x})^2
 \end{aligned}$$

又因为

$$\begin{aligned}
 &\sum_{i=1}^T w_i E(h_i|\mathbf{x}) - E(H|\mathbf{x}) \\
 &= \sum_{i=1}^T w_i (f(\mathbf{x}) - h_i(\mathbf{x}))^2 - (f(\mathbf{x}) - H(\mathbf{x}))^2 \\
 &= \sum_{i=1}^T w_i h_i(\mathbf{x})^2 - H(\mathbf{x})^2
 \end{aligned}$$

所以

$$\bar{A}(h|\mathbf{x}) = \sum_{i=1}^T w_i E(h_i|\mathbf{x}) - E(H|\mathbf{x})$$

(8.31)

⇒ 对某个示例  $x$  有：个体学习器分歧的加权均值 = 个体学习器误差的加权均值 - 集成的平方误差

令  $p(\mathbf{x})$  表示样本的概率密度，则在全样本 上有：

$$\sum_{i=1}^T w_i \int A(h_i | \mathbf{x}) p(\mathbf{x}) d\mathbf{x} = \sum_{i=1}^T w_i \int E(h_i | \mathbf{x}) p(\mathbf{x}) d\mathbf{x} - \int E(H | \mathbf{x}) p(\mathbf{x}) d\mathbf{x} . \quad (8.32)$$

- $\int A(h_i | \mathbf{x}) p(\mathbf{x}) d\mathbf{x}$  个体在全样本上的分歧
- $\sum_{i=1}^T w_i \int A(h_i | \mathbf{x}) p(\mathbf{x}) d\mathbf{x}$  集成在全样本上的分歧

即是 (8.31) 结论用到全样本上的结果

- 个体学习器在全样本上的泛化误差（准确率）

$$E_i = \int E(h_i | \mathbf{x}) p(\mathbf{x}) d\mathbf{x} , \quad (8.33)$$

- 个体学习器在全样本上的分歧项（多样性）

$$A_i = \int A(h_i | \mathbf{x}) p(\mathbf{x}) d\mathbf{x} . \quad (8.34)$$

(连续情形)

离散情形： $A_i = \sum_{\mathbf{x} \in D} A(h_i | \mathbf{x}) p_{\mathbf{x}}$

- 集成的泛化误差

$$E = \int E(H | \mathbf{x}) p(\mathbf{x}) d\mathbf{x} . \quad (8.35)$$

(8.33) ~ (8.35) 带入 (3.32)

$\bar{E} = \sum_{i=1}^T w_i E_i$  表示个体学习器泛化误差的加权均值

$\bar{A} = \sum_{i=1}^T w_i A_i$  表示个体学习器加权分歧值

则有

## 误差-分歧分解

$$E = \bar{E} - \bar{A}. \quad (8.36)$$

集成的泛化误差=个体分析器加权泛化误差-个体分析器加权分歧值

——个体准确率（泛化误差）越高、多样性（分歧值）越好，集成越好

上面推导只适用于回归学习，难以直接推广到分类学习任务上

### 7.5.2. 多样性度量

估算个体学习器的多样化程度

典型做法：考虑个体分类器两两相似/不相似性

#### 7.5.2.1. 成对型多样性度量

给定数据集  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ , 对二分类任务,  $y_i \in \{-1, +1\}$ , 分类器  $h_i$  与  $h_j$  的预测结果列联表(contingency table)为

		$h_i = +1$	$h_i = -1$
$h_j = +1$	$a$	$c$	
	$b$	$d$	

其中,  $a$  表示  $h_i$  与  $h_j$  均预测为正类的样本数目;  $b, c, d$  含义由此类推;  
 $a + b + c + d = m$ . 基于这个列联表, 下面给出一些常见的多样性度量.

- 不合度量(disagreement measure)

$$dis_{ij} = \frac{b + c}{m}. \quad (8.37)$$

$dis_{ij}$  的值域为  $[0, 1]$ . 值越大则多样性越大.

- 相关系数(correlation coefficient)

$$\rho_{ij} = \frac{ad - bc}{\sqrt{(a+b)(a+c)(c+d)(b+d)}}. \quad (8.38)$$

$\rho_{ij}$  的值域为  $[-1, 1]$ . 若  $h_i$  与  $h_j$  无关, 则值为 0; 若  $h_i$  与  $h_j$  正相关则值为正, 否则为负.

- $Q$ -统计量( $Q$ -statistic)

$$Q_{ij} = \frac{ad - bc}{ad + bc}. \quad (8.39)$$

$Q_{ij}$  与相关系数  $\rho_{ij}$  的符号相同, 且  $|Q_{ij}| \leq |\rho_{ij}|$ .

- $\kappa$ -统计量( $\kappa$ -statistic)

$$\kappa = \frac{p_1 - p_2}{1 - p_2}. \quad (8.40)$$

其中,  $p_1$  是两个分类器取得一致的概率;  $p_2$  是两个分类器偶然达成一致的概率, 它们可由数据集  $D$  估算:

$$p_1 = \frac{a + d}{m}, \quad (8.41)$$

$$p_2 = \frac{(a + b)(a + c) + (c + d)(b + d)}{m^2}. \quad (8.42)$$

若分类器在  $D$  上完全一致, 则  $\kappa = 1$  (不希望)

若偶然一致, 则  $\kappa = 0$

$\kappa$  通常非负, 仅在达成一致的概率低于偶然性时取负

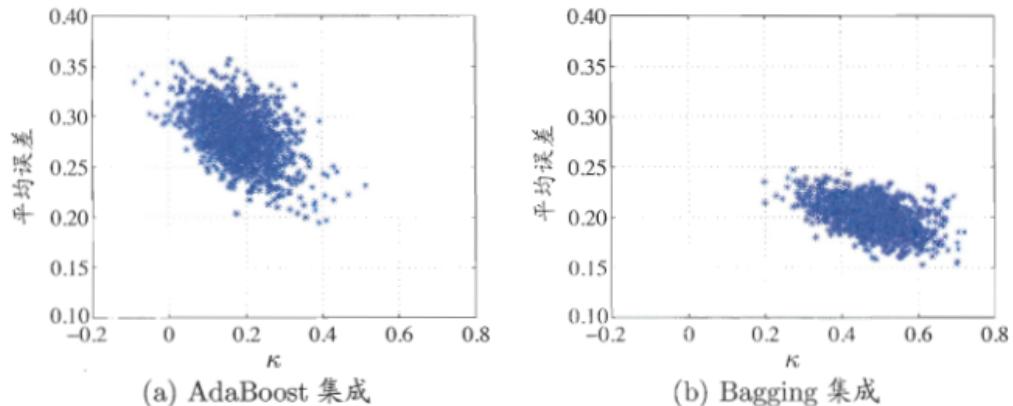
### 7.5.2.2. $\kappa$ -误差图

每一对分类器作为一个点

横坐标:  $\kappa$

纵坐标: 平均误差

- 点云越高, 误差越大, 个体分类器准确性越低
- 点云越靠右, 个体学习器多样性越小



### 7.5.3. 多样性增强

引入随机性——对数据样本、输入属性、输出表示、算法参数进行扰动

#### 7.5.3.1. 数据样本扰动

采样法

e.g.

Bagging: 自助采样

AdaBoost: 序列采样

- 对不稳定基学习器很有效, 如: 决策树、神经网络
- 稳定基学习器对样本扰动不敏感, 如: 线性学习器、支持向量机、朴素贝叶斯、 $k$ 近邻学习器

### 7.5.3.2. 输入属性扰动

样本的属性子集视为**子空间**, 不同属性子集提供观察数据的不同视角

#### 随机子空间算法

<p><math>d'</math> 小于初始属性数 <math>d</math>.</p> <p><math>\mathcal{F}_t</math> 包含 <math>d'</math> 个随机选取的属性, <math>D_t</math> 仅保留 <math>\mathcal{F}_t</math> 中的属性.</p>	<p><b>输入:</b> 训练集 <math>D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}</math>; 基学习算法 <math>\mathcal{L}</math>; 基学习器数 <math>T</math>; 子空间属性数 <math>d'</math>.</p> <p><b>过程:</b></p> <pre>1: for <math>t = 1, 2, \dots, T</math> do 2:   <math>\mathcal{F}_t = \text{RS}(D, d')</math> 3:   <math>D_t = \text{Map}_{\mathcal{F}_t}(D)</math> 4:   <math>h_t = \mathcal{L}(D_t)</math> 5: end for</pre> <p><b>输出:</b> <math>H(\mathbf{x}) = \arg \max_{y \in \mathcal{Y}} \sum_{t=1}^T \mathbb{I}(h_t(\text{Map}_{\mathcal{F}_t}(\mathbf{x})) = y)</math></p>
---	--

图 8.11 随机子空间算法

**适用情况:** 包含大量冗余属性的数据

- 在子空间中训练可以产生多样性大的个体
- 属性减少节省时间
- 减少一些冗余属性不会大幅降低训练效果

### 7.5.3.3. 输出表示扰动

**翻转法:** 随即改变一些样本的标记

**输出调制法:** 将分类输出转化为回归输出再构建个体学习器

**ECOC法:** 将原任务拆解为多个可同时求解的子任务 (利用纠错输出码将多分类拆为一系列二分类任务来训练基学习器)

### 7.5.3.4. 算法参数扰动

- 随即设置不同参数
- 负相关法: 正则化项
- 参数较少的算法: 替换某些环节
  - 单一学习器时用交叉验证等方法确定最终参数

## 8. 聚类

预先标签未知

聚类过程自动生成不相交的簇

### 8.1. 性能度量

聚类目标: 同一族样本尽可能相似, 不同簇样本尽可能不同

- 外部指标: 与参考模型比较

对数据集  $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ , 假定通过聚类给出的簇划分为  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$ , 参考模型给出的簇划分为  $\mathcal{C}^* = \{C_1^*, C_2^*, \dots, C_s^*\}$ . 相应地, 令  $\lambda$  与  $\lambda^*$  分别表示与  $\mathcal{C}$  和  $\mathcal{C}^*$  对应的簇标记向量. 我们将样本两两配对考虑, 定义

$$a = |SS|, \quad SS = \{(\mathbf{x}_i, \mathbf{x}_j) \mid \lambda_i = \lambda_j, \lambda_i^* = \lambda_j^*, i < j\}, \quad (9.1)$$

$$b = |SD|, \quad SD = \{(\mathbf{x}_i, \mathbf{x}_j) \mid \lambda_i = \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\}, \quad (9.2)$$

$$c = |DS|, \quad DS = \{(\mathbf{x}_i, \mathbf{x}_j) \mid \lambda_i \neq \lambda_j, \lambda_i^* = \lambda_j^*, i < j\}, \quad (9.3)$$

$$d = |DD|, \quad DD = \{(\mathbf{x}_i, \mathbf{x}_j) \mid \lambda_i \neq \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\}, \quad (9.4)$$

$SS$ :  $\mathcal{C}$ 中同簇,  $\mathcal{C}^*$ 中同簇

$SD$ :  $\mathcal{C}$ 同簇,  $\mathcal{C}^*$ 不同簇

常用外部指标——[0,1] 越大越好

- Jaccard系数 (JC)

$$JC = \frac{a}{a + b + c}. \quad (9.5)$$

- FM指数 (FMI)

$$FMI = \sqrt{\frac{a}{a+b} \cdot \frac{a}{a+c}}. \quad (9.6)$$

- Rand指数 (RI)

$$RI = \frac{2(a+d)}{m(m-1)}. \quad (9.7)$$

常用内部度量

定义

- $C$ 内两两之间平均距离

$$\text{avg}(C) = \frac{2}{|C|(|C|-1)} \sum_{1 \leq i < j \leq |C|} \text{dist}(\mathbf{x}_i, \mathbf{x}_j), \quad (9.8)$$

- $C$ 内样本间最远距离

$$\text{diam}(C) = \max_{1 \leq i < j \leq |C|} \text{dist}(\mathbf{x}_i, \mathbf{x}_j), \quad (9.9)$$

- 簇i、j间最小样本距离

$$d_{\min}(C_i, C_j) = \min_{\mathbf{x}_i \in C_i, \mathbf{x}_j \in C_j} \text{dist}(\mathbf{x}_i, \mathbf{x}_j), \quad (9.10)$$

- 簇i、j中心距离

$$d_{\text{cen}}(C_i, C_j) = \text{dist}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j), \quad (9.11)$$

内部指标:

- DB指数 (DBI)

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \left( \frac{\text{avg}(C_i) + \text{avg}(C_j)}{d_{\text{cen}}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)} \right). \quad (9.12)$$

越小越好

- Dunn指数 (DI)

$$DI = \min_{1 \leq i \leq k} \left\{ \min_{j \neq i} \left( \frac{d_{\min}(C_i, C_j)}{\max_{1 \leq l \leq k} \text{diam}(C_l)} \right) \right\}. \quad (9.13)$$

越大越好

## 8.2. 距离计算

### 8.2.1. 距离度量的性质

**非负性**:  $dist(x_i, x_j) \geq 0$

**同一性**:  $dist(x_i, x_j) = 0$  当且仅当  $x_i = x_j$

**对称性**:  $dist(x_i, x_j) = dist(x_j, x_i)$

**传递性**:  $dist(x_i, x_j) \leq dist(x_i, x_k) + dist(x_k, x_j)$

(传递性也即三角不等式)

但用于相似度度量的距离未必一定满足上面四个性质

如:

- 余弦距离不满足传递性

### 8.2.2. 常用距离

#### 8.2.2.1. 闵可夫斯基距离

$$dist(x_i, x_j) = \left( \sum_u^n |x_{iu} - x_{ju}|^p \right)^{1/p}$$

$p = 2$ : 欧式距离

主要应用连续属性上

$p = 1$ : 曼哈顿距离, 也称街区距离

- p=2

$$dist_{ed}(x_i, x_j) = ||x_i - x_j||_2 = \sqrt{\sum_{u=1}^n |x_{iu} - x_{ju}|^2}. \quad (9.19)$$

#### 8.2.2.2. 离散属性

- 可比较: 用数值表示相对大小
  - 不可比较: 无法计算距离
- 采用**VDM**处理无序属性

- 令  $m_{u,a}$  表示属性  $u$  上取值为  $a$  的样本数， $m_{u,a,i}$  表示在第  $i$  个样本簇中在属性  $u$  上取值为  $a$  的样本数。则属性  $u$  在两个离散值  $a$  和  $b$  的 VDM 距离为

$$VDM_p(a, b) = \sum_{i=1}^k \left| \frac{m_{u,a,i}}{m_{u,a}} - \frac{m_{u,b,i}}{m_{u,b}} \right|^p$$

#### 8.2.2.3. 混合属性

$$\text{MinkovDM}_p = \left( \sum_{u=1}^{n_c} |x_{iu} - x_{ju}|^p + \sum_{u=n_c+1}^n VDM_p(x_{iu}, x_{ju}) \right)$$

#### 8.2.2.4. 加权距离

- 加权闵可夫斯基距离

$$dist_{wmk}(x_i, x_j) = \left( w_1 |x_{i1} - y_{j1}|^p + \dots + w_n |x_{in} - x_{jn}|^p \right)^{\frac{1}{p}}$$

满足  $w_i \geq 0, \sum_i^n w_i = 1$

### 8.3. 原型聚类

原型：样本空间中具有代表性的点

#### 8.3.1. k均值算法

样本集  $D = \{x_1, x_2, \dots, x_m\}$

簇划分  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$

最小化平方误差

$$E = \sum_{i=1}^k \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2, \quad (9.24)$$

其中  $\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\mathbf{x} \in C_i} \mathbf{x}$  是簇  $C_i$  的均值向量.

上式一定程度上刻画了簇内样本围绕簇均值向量的紧密程度， $E$  越小则簇样本的相似度越高  
采用贪心策略

# 原型聚类—K均值

输入: 样本集  $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ ;

聚类簇数  $k$ .

过程:

初始化每个簇的均值向量

```

1: 从  $D$  中随机选择  $k$  个样本作为初始均值向量  $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_k\}$ 
2: repeat
3:   令  $C_i = \emptyset$  ( $1 \leq i \leq k$ )                                将每个样本分配给最近的簇
4:   for  $j = 1, \dots, m$  do
5:     计算样本  $\mathbf{x}_j$  与各均值向量  $\boldsymbol{\mu}_i$  ( $1 \leq i \leq k$ ) 的距离:  $d_{ji} = \|\mathbf{x}_j - \boldsymbol{\mu}_i\|_2$ ;
6:     根据距离最近的均值向量确定  $\mathbf{x}_j$  的簇标记:  $\lambda_j = \arg \min_{i \in \{1, 2, \dots, k\}} d_{ji}$ ;
7:     将样本  $\mathbf{x}_j$  划入相应的簇:  $C_{\lambda_j} = C_{\lambda_j} \cup \{\mathbf{x}_j\}$ ;
8:   end for
9:   for  $i = 1, \dots, k$  do
10:    计算新均值向量:  $\boldsymbol{\mu}'_i = \frac{1}{|C_i|} \sum_{\mathbf{x} \in C_i} \mathbf{x}$ ;  计算每个簇的均值向量
11:    if  $\boldsymbol{\mu}'_i \neq \boldsymbol{\mu}_i$  then
12:      将当前均值向量  $\boldsymbol{\mu}_i$  更新为  $\boldsymbol{\mu}'_i$ 
13:    else
14:      保持当前均值向量不变
15:    end if
16:  end for
17: until 当前均值向量均未更新
18: return 簇划分结果

```

输出: 簇划分  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$

$$E(\mathbf{T}, \boldsymbol{\mu}) = \sum_{i=1}^m \left\| \mathbf{x}_i - \mathbf{t}_i^\top \boldsymbol{\mu} \right\|_2^2 = \|\mathbf{X} - \mathbf{T}\boldsymbol{\mu}\|_F^2 \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^\top \\ \mathbf{t}_2^\top \\ \vdots \\ \mathbf{t}_m^\top \end{bmatrix}$$

$$\mathbf{t}_i^\top = \begin{bmatrix} 0, \dots, 1, \dots, 0 \\ 1 & c_i & k \end{bmatrix} \quad \boldsymbol{\mu} = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_k] \in \mathbb{R}^{k \times d}$$

优化问题:

$$\min_{T, \mu} E(T, \mu) = \|\mathbf{X} - \mathbf{T}\boldsymbol{\mu}\|_F^2 \quad (33)$$

## 算法流程 (迭代优化) :

初始化每个簇的均值向量

repeat

1. 将每个样本分配给最近的簇;  $\mathbf{T}^{(t)} \leftarrow \min_T E(\mathbf{T}, \boldsymbol{\mu}^{(t-1)})$
2. 计算每个簇的均值向量;  $\boldsymbol{\mu}^{(t)} \leftarrow \min_{\boldsymbol{\mu}} E(\mathbf{T}^{(t)}, \boldsymbol{\mu})$

until 当前均值向量均未更新

**step.1**

- 考虑  $\mathbf{T}^{(t)} \leftarrow \min_{\mathbf{T}} E(\mathbf{T}, \boldsymbol{\mu}^{(t-1)})$

- 由于样本间互不依赖，考虑求解第*i*个样本的  $\mathbf{t}_i$

$$\min_{t_i} \|x_i - \mathbf{t}_i^\top \boldsymbol{\mu}\|_2^2 = \left\| x_i - \sum_c t_{ic} \boldsymbol{\mu}_c \right\|_2^2 \quad \Leftrightarrow \min_c \|x_i - \boldsymbol{\mu}_c\|$$

将样本*i*划分给距离最近的簇

$t_{ic}$  在所有的  $c$  中只能有一个地方取值为 1，其余均为 0，即  $\sum_c t_{ic} = 1$

### step.2

- 进一步考虑  $\boldsymbol{\mu}^{(t)} \leftarrow \min_{\boldsymbol{\mu}} E(\mathbf{T}^{(t)}, \boldsymbol{\mu})$
- 求  $E(\mathbf{T}, \boldsymbol{\mu})$  关于  $\boldsymbol{\mu}$  的梯度，令其等于 0，则

$$\boldsymbol{\mu} = (\mathbf{T}^\top \mathbf{T})^{-1} \mathbf{T}^\top \mathbf{X}$$

$$\mathbf{T}^\top \mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_m] \begin{bmatrix} \mathbf{t}_1^\top \\ \mathbf{t}_2^\top \\ \vdots \\ \mathbf{t}_m^\top \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{t}}_1^\top \\ \tilde{\mathbf{t}}_2^\top \\ \vdots \\ \tilde{\mathbf{t}}_k^\top \end{bmatrix} [\tilde{\mathbf{t}}_1, \tilde{\mathbf{t}}_2, \dots, \tilde{\mathbf{t}}_k] = \begin{bmatrix} \tilde{\mathbf{t}}_1^\top \tilde{\mathbf{t}}_1 & \tilde{\mathbf{t}}_1^\top \tilde{\mathbf{t}}_2 & \cdots & \tilde{\mathbf{t}}_1^\top \tilde{\mathbf{t}}_k \\ \tilde{\mathbf{t}}_2^\top \tilde{\mathbf{t}}_1 & \tilde{\mathbf{t}}_2^\top \tilde{\mathbf{t}}_2 & \cdots & \tilde{\mathbf{t}}_2^\top \tilde{\mathbf{t}}_k \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{\mathbf{t}}_k^\top \tilde{\mathbf{t}}_1 & \tilde{\mathbf{t}}_k^\top \tilde{\mathbf{t}}_2 & \cdots & \tilde{\mathbf{t}}_k^\top \tilde{\mathbf{t}}_k \end{bmatrix}$$

$$\mathbf{T}^\top \mathbf{X} = \begin{bmatrix} \tilde{\mathbf{t}}_1^\top \mathbf{X} \\ \tilde{\mathbf{t}}_2^\top \mathbf{X} \\ \vdots \\ \tilde{\mathbf{t}}_k^\top \mathbf{X} \end{bmatrix} = \begin{bmatrix} \sum_{x \in C_1} x \\ \sum_{x \in C_2} x \\ \vdots \\ \sum_{x \in C_k} x \end{bmatrix} = \begin{bmatrix} |C_1| & 0 & \cdots & 0 \\ 0 & |C_2| & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & |C_k| \end{bmatrix}$$

$$\boldsymbol{\mu} = (\mathbf{T}^\top \mathbf{T})^{-1} \mathbf{T}^\top \mathbf{X} = \begin{bmatrix} \frac{1}{|C_1|} \sum_{x \in C_1} x \\ \frac{1}{|C_2|} \sum_{x \in C_2} x \\ \vdots \\ \frac{1}{|C_k|} \sum_{x \in C_k} x \end{bmatrix}$$

《机器学习概论》 2023/11/26

ppt这里没看懂

### 8.3.2. 学习向量量化(LVQ)

假设数据样本带有类别标记

m个样本，n维特征

学习目标

一组n维原型向量  $\{\vec{p}_1, \vec{p}_2, \dots, \vec{p}_q\}$ ，每个原型向量代表一个聚类簇

---

**输入:** 样本集  $D = \{(x_1, y_1), \dots, (x_m, y_m)\}$ ;  
原型向量个数  $q$ , 各原型向量预设的类别标记  $\{t_1, t_2, \dots, t_q\}$ ;  
学习率  $\eta \in (0, 1)$ .

**过程:**

```

1: 初始化一组原型向量  $\{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_q\}$ 
2: repeat
3:   从样本集  $D$  随机选取样本  $(\mathbf{x}_j, y_j)$ ;
4:   计算样本  $\mathbf{x}_j$  与  $\mathbf{p}_i$  ( $1 \leq i \leq q$ ) 的距离:  $d_{ji} = \|\mathbf{x}_j - \mathbf{p}_i\|_2$ ;
5:   找出与  $\mathbf{x}_j$  距离最近的原型向量;  $i^* = \arg \min_{i \in \{1, 2, \dots, q\}} d_{ji}$ ;
6:   if  $y_j = t_{i^*}$  then
7:      $\mathbf{p}' = \mathbf{p}_{i^*} + \eta \cdot (\mathbf{x}_j - \mathbf{p}_{i^*})$ 
8:   else
9:      $\mathbf{p}' = \mathbf{p}_{i^*} - \eta \cdot (\mathbf{x}_j - \mathbf{p}_{i^*})$ 
10:  end if
11:  将原型向量  $\mathbf{p}_{i^*}$  更新为  $\mathbf{p}'$ 
12: until 满足停止条件
13: return 当前原型向量
输出: 原型向量  $\{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_q\}$ 

```

根据两者的类别标记是否一致来对原型向量进行更新  
对样本  $\mathbf{x}_j$ , 若最近的原型向量  $\mathbf{p}_{i^*}$  和  $\mathbf{x}_j$  的类别标记相同, 则令  $\mathbf{p}_{i^*}$  向  $\mathbf{x}_j$  的方向靠拢

$$\begin{aligned}\|\mathbf{p}' - \mathbf{x}_j\| &= \|\mathbf{p}_{i^*} + \eta(\mathbf{x}_j - \mathbf{p}_{i^*}) - \mathbf{x}_j\| \\ &= (1 - \eta)\|\mathbf{p}_{i^*} - \mathbf{x}_j\| \\ &< \|\mathbf{p}_{i^*} - \mathbf{x}_j\|\end{aligned}$$

《机器学习概论》 2023/11/26

- 初始化: 可以从相应类别标记的样本中随机选取一个作为原型向量

## 训练结果

每个原型向量  $\mathbf{p}_i$  定义了与之相关的一个区域  $R_i$ , 该区域内每个样本满足:  $\mathbf{p}_i$  是距离样本最近的原型向量

即:  $R_i = \{\mathbf{x} \in \mathcal{X} \mid \|\mathbf{x} - \mathbf{p}_i\|_2 \leq \|\mathbf{x} - \mathbf{p}_{i'}\|_2, i' \neq i\}$ . (9.27)

### 8.3.3. 高斯混合聚类

#### 概率模型

我们先简单回顾一下(多元)高斯分布的定义. 对  $n$  维样本空间  $\mathcal{X}$  中的随机向量  $\mathbf{x}$ , 若  $\mathbf{x}$  服从高斯分布, 其概率密度函数为

$\Sigma$ : 对称正定矩阵;  
 $|\Sigma|$ :  $\Sigma$  的行列式;  
 $\Sigma^{-1}$ :  $\Sigma$  的逆矩阵.

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1} (\mathbf{x}-\mu)}, \quad (9.28)$$

其中  $\mu$  是  $n$  维均值向量,  $\Sigma$  是  $n \times n$  的协方差矩阵. 由式(9.28)可看出, 高斯分布完全由均值向量  $\mu$  和协方差矩阵  $\Sigma$  这两个参数确定. 为了明确显示高斯分布

$p_{\mathcal{M}}(\cdot)$  也是概率密度函数,  $\int p_{\mathcal{M}}(\mathbf{x}) d\mathbf{x} = 1$ .

$$p_{\mathcal{M}}(\mathbf{x}) = \sum_{i=1}^k \alpha_i \cdot p(\mathbf{x} \mid \mu_i, \Sigma_i), \quad (9.29)$$

由  $k$  个混合成分组成, 每个混合成分对应一个高斯分布

$\alpha_i > 0$ : 混合系数, 和为 1

- 训练样本生成

根据混合系数定义的先验分布选择高斯混合成分 (以混合系数作为概率)

- 生成训练集  $D = \{x_1, x_2, \dots, x_m\}$ , 对于给定样本  $x_j$ , 它由哪个高斯混合成分生成?

—求后验概率  $p_{\mathcal{M}}(z_j = i \mid x_i)$

由贝叶斯公式:

$$p_{\mathcal{M}}(z_j = i | \mathbf{x}_j) = \frac{P(z_j = i) \cdot p_{\mathcal{M}}(\mathbf{x}_j | z_j = i)}{p_{\mathcal{M}}(\mathbf{x}_j)}$$

分子第 1 项  $P(z_j = i) = \alpha_i$ ; 第 2 项即第  $i$  个高斯混合成分生成样本  $\mathbf{x}_j$  的概率  $p(\mathbf{x}_j | \mu_i, \Sigma_i)$ , 根据式 (9.28) 将  $\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$  替换为  $\mathbf{x}_j, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$  即得; 分母  $p_{\mathcal{M}}(\mathbf{x}_j)$  即为将  $\mathbf{x}_j$  代入式 (9.29) 即得。

注意, 西瓜书中后面将  $p_{\mathcal{M}}(z_j = i | \mathbf{x}_j)$  记为  $\gamma_{ji}$ , 其中  $1 \leq j \leq m, 1 \leq i \leq k$ 。

$$\begin{aligned} p_{\mathcal{M}}(z_j = i | \mathbf{x}_j) &= \frac{P(z_j = i) \cdot p_{\mathcal{M}}(\mathbf{x}_j | z_j = i)}{p_{\mathcal{M}}(\mathbf{x}_j)} \\ &= \frac{\alpha_i \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{l=1}^k \alpha_l \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}. \end{aligned} \quad (9.30)$$

- 确定样本簇标记 (在训练模型之后)

$$x_j \text{ 簇标记 } \lambda_j = \arg \max_{i \in \{1, 2, \dots, k\}} \gamma_{ji}$$

即使得后验概率最大

若将所有  $\gamma_{ji}$  组成一个矩阵  $\Gamma$ , 其中  $\gamma_{ji}$  为第  $j$  行第  $i$  列的元素, 矩阵  $\Gamma$  大小为  $m \times k$ , 即

$$\Gamma = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \cdots & \gamma_{1k} \\ \gamma_{21} & \gamma_{22} & \cdots & \gamma_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{m1} & \gamma_{m2} & \cdots & \gamma_{mk} \end{bmatrix}_{m \times k}$$

其中  $m$  为训练集样本个数,  $k$  为高斯混合模型包含的混合模型个数。可以看出, 式 (9.31) 就是找出矩阵  $\Gamma$  第  $j$  行的所有  $k$  个元素中最大的那个元素的位置。

- 求模型参数

最大化对数似然估计

$$\begin{aligned} LL(D) &= \ln \left( \prod_{j=1}^m p_{\mathcal{M}}(\mathbf{x}_j) \right) \\ &= \sum_{j=1}^m \ln \left( \sum_{i=1}^k \alpha_i \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \right), \end{aligned} \quad (9.32)$$

解释

训练集  $D$  中既然出现了  $x_j$ , 则希望求解的参数使得这种可能性  $p_{\mathcal{M}}(x_j)$  最大;

假设  $m$  个样本相互独立, 则他们恰好一起出现的概率是  $\prod_{j=1}^m p_{\mathcal{M}}(x_j)$ , 即为似然函数

为了防止连乘下溢, 采用对数似然

- EM 算法 (期望最大化算法) 迭代优化**

根据公式 (9.28) 可知:

$$p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i)\right)$$

又根据公式 (9.32), 由

$$\frac{\partial LL(D)}{\partial \boldsymbol{\mu}_i} = \frac{\partial LL(D)}{\partial p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} \cdot \frac{\partial p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\partial \boldsymbol{\mu}_i} = 0$$

其中:

$$\begin{aligned} \frac{\partial LL(D)}{\partial p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} &= \frac{\partial \sum_{j=1}^m \ln \left( \sum_{l=1}^k \alpha_l \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \right)}{\partial p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} \\ &= \sum_{j=1}^m \frac{\partial \ln \left( \sum_{l=1}^k \alpha_l \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \right)}{\partial p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} \\ &= \sum_{j=1}^m \frac{\alpha_i}{\sum_{l=1}^k \alpha_l \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)} \end{aligned}$$

$$\begin{aligned} \frac{\partial p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\partial \boldsymbol{\mu}_i} &= \frac{\partial \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i)\right)}{\partial \boldsymbol{\mu}_i} \\ &= \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \cdot \frac{\partial \exp\left(-\frac{1}{2} (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i)\right)}{\partial \boldsymbol{\mu}_i} \\ &= \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \cdot \exp\left(-\frac{1}{2} (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i)\right) \cdot -\frac{1}{2} \frac{\partial (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i)}{\partial \boldsymbol{\mu}_i} \\ &= \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \cdot \exp\left(-\frac{1}{2} (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i)\right) \cdot \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i) \\ &= p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \cdot \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i) \end{aligned}$$

其中, 由矩阵求导的法则  $\frac{\partial \mathbf{a}^T \mathbf{X} \mathbf{a}}{\partial \mathbf{a}} = 2\mathbf{X}\mathbf{a}$  可得:

$$\begin{aligned} -\frac{1}{2} \frac{\partial (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i)}{\partial \boldsymbol{\mu}_i} &= -\frac{1}{2} \cdot 2\boldsymbol{\Sigma}_i^{-1} (\boldsymbol{\mu}_i - \mathbf{x}_j) \\ &= \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i) \end{aligned}$$

因此有:

$$\frac{\partial LL(D)}{\partial \boldsymbol{\mu}_i} = \sum_{j=1}^m \frac{\alpha_i}{\sum_{l=1}^k \alpha_l \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)} \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \cdot \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i) = 0$$

此前有

$$\begin{aligned} p_{\mathcal{M}}(z_j = i | \mathbf{x}_j) &= \frac{P(z_j = i) \cdot p_{\mathcal{M}}(\mathbf{x}_j | z_j = i)}{p_{\mathcal{M}}(\mathbf{x}_j)} \\ &= \frac{\alpha_i \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{l=1}^k \alpha_l \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)} . \end{aligned} \quad (9.30)$$

因此

$$\sum_{j=1}^m \gamma_{ji} (\mathbf{x}_j - \boldsymbol{\mu}_i) = 0$$

$$\boldsymbol{\mu}_i = \frac{\sum_{j=1}^m \gamma_{ji} \mathbf{x}_j}{\sum_{j=1}^m \gamma_{ji}}$$

⇒ 混成成分均值可以用样本的加权平均估计, 样本权重是样本属于这个成分的后验概率

同理可得

$$\boldsymbol{\Sigma}_i = \frac{\sum_{j=1}^m \gamma_{ji} (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T}{\sum_{j=1}^m \gamma_{ji}}. \quad (9.35)$$

- 拉格朗日求混合系数

增加  $\sum \alpha_i = 1$  约束条件，得到拉格朗日形式

对于混合系数  $\alpha_i$ , 除了要最大化  $LL(D)$ , 还需满足  $\alpha_i \geq 0$ ,  $\sum_{i=1}^k \alpha_i = 1$ . 考虑  $LL(D)$  的拉格朗日形式

$$LL(D) + \lambda \left( \sum_{i=1}^k \alpha_i - 1 \right), \quad (9.36)$$

其中  $\lambda$  为拉格朗日乘子. 由式(9.36)对  $\alpha_i$  的导数为 0, 有

$$\sum_{j=1}^m \frac{p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{l=1}^k \alpha_l \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)} + \lambda = 0, \quad (9.37)$$

两边同乘以  $\alpha_i$ , 对所有样本求和可知  $\lambda = -m$ , 有

$$\alpha_i = \frac{1}{m} \sum_{j=1}^m \gamma_{ji}, \quad (9.38)$$

即每个高斯成分的混合系数由样本属于该成分的平均后验概率确定.

勘误： (9.38) 上一行，改为 对所有混合成分求和

### EM算法总结

1. E：根据当前参数计算每个样本属于每个高斯成分的后验概率
2. M：更新模型参数，最小化似然函数

---

<p>EM 算法的 E 步.</p> <p>EM 算法的 M 步.</p> <p>例如达到最大迭代轮数.</p>	<p><b>输入:</b> 样本集 <math>D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}</math>; 高斯混合成分个数 <math>k</math>.</p> <p><b>过程:</b></p> <ol style="list-style-type: none"> <li>1: 初始化高斯混合分布的模型参数 <math>\{(\alpha_i, \mu_i, \Sigma_i) \mid 1 \leq i \leq k\}</math></li> <li>2: <b>repeat</b></li> <li>3:   <b>for</b> <math>j = 1, 2, \dots, m</math> <b>do</b></li> <li>4:     根据式(9.30)计算 <math>\mathbf{x}_j</math> 由各混合成分生成的后验概率, 即 <math>\gamma_{ji} = p_M(z_j = i \mid \mathbf{x}_j) (1 \leq i \leq k)</math></li> <li>5:   <b>end for</b></li> <li>6:   <b>for</b> <math>i = 1, 2, \dots, k</math> <b>do</b></li> <li>7:     计算新均值向量: <math>\mu'_i = \frac{\sum_{j=1}^m \gamma_{ji} \mathbf{x}_j}{\sum_{j=1}^m \gamma_{ji}}</math>;</li> <li>8:     计算新协方差矩阵: <math>\Sigma'_i = \frac{\sum_{j=1}^m \gamma_{ji} (\mathbf{x}_j - \mu'_i)(\mathbf{x}_j - \mu'_i)^T}{\sum_{j=1}^m \gamma_{ji}}</math>;</li> <li>9:     计算新混合系数: <math>\alpha'_i = \frac{\sum_{j=1}^m \gamma_{ji}}{m}</math>;</li> <li>10:   <b>end for</b></li> <li>11:   将模型参数 <math>\{(\alpha_i, \mu_i, \Sigma_i) \mid 1 \leq i \leq k\}</math> 更新为 <math>\{(\alpha'_i, \mu'_i, \Sigma'_i) \mid 1 \leq i \leq k\}</math></li> <li>12: <b>until</b> 满足停止条件</li> <li>13: <math>C_i = \emptyset (1 \leq i \leq k)</math></li> <li>14: <b>for</b> <math>j = 1, 2, \dots, m</math> <b>do</b></li> <li>15:   根据式(9.31)确定 <math>\mathbf{x}_j</math> 的簇标记 <math>\lambda_j</math>;</li> <li>16:   将 <math>\mathbf{x}_j</math> 划入相应的簇: <math>C_{\lambda_j} = C_{\lambda_j} \cup \{\mathbf{x}_j\}</math></li> <li>17: <b>end for</b></li> </ol> <p><b>输出:</b> 簇划分 <math>\mathcal{C} = \{C_1, C_2, \dots, C_k\}</math></p>
--	---

---

图 9.6 高斯混合聚类算法

## 8.4. 密度聚类

假设聚类结构可以根据样本分布的紧密程度确定

从样本密度角度考虑样本之间的可连接性, 基于可连接样本不断扩展聚类簇以获得最终的聚类结果

### 8.4.1. DBSCAN算法

#### 8.4.1.1. 定义

- **$\epsilon$ 邻域:** 对样本  $\mathbf{x}_i \in D$ , 其  $\epsilon$ 邻域包含样本集  $D$  中与  $\mathbf{x}_i$  的距离不大于  $\epsilon$  的样本;
- **核心对象:** 若样本  $\mathbf{x}_i$  的  $\epsilon$  邻域至少包含  $\text{MinPts}$  个样本, 则该样本点为核心对象;
- **密度直达:** 若样本  $\mathbf{x}_j$  位于样本  $\mathbf{x}_i$  的  $\epsilon$  邻域中, 且  $\mathbf{x}_i$  是一个核心对象, 则称样本  $\mathbf{x}_j$  由  $\mathbf{x}_i$  密度直达;
- **密度可达:** 对样本  $\mathbf{x}_i$  与  $\mathbf{x}_j$ , 若存在样本序列  $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n$ , 其中  $\mathbf{p}_1 = \mathbf{x}_i$ ,  $\mathbf{p}_n = \mathbf{x}_j$  且  $\mathbf{p}_{i+1}$  由  $\mathbf{p}_i$  密度直达, 则该两样本密度可达;
- **密度相连:** 对样本  $\mathbf{x}_i$  与  $\mathbf{x}_j$ , 若存在样本  $\mathbf{x}_k$  使得两样本均由  $\mathbf{x}_k$  密度可达, 则称该两样本密度相连。

**簇:** 由密度可达关系导出的最大密度相连样本集合

即满足连接性、最大性的非空样本子集

$$\text{连接性(connectivity): } \mathbf{x}_i \in C, \mathbf{x}_j \in C \Rightarrow \mathbf{x}_i \text{ 与 } \mathbf{x}_j \text{ 密度相连} \quad (9.39)$$

$$\text{最大性(maximality): } \mathbf{x}_i \in C, \mathbf{x}_j \text{ 由 } \mathbf{x}_i \text{ 密度可达} \Rightarrow \mathbf{x}_j \in C \quad (9.40)$$

$x_i$ 由核心对象密度可达  $\Rightarrow x_j$ 由核心对象密度可达  $\Rightarrow x_i, x_j$ 密度相连

最大性不会违反连接性

#### 8.4.1.2. 算法

实际上，若 $x$ 为核心对象，由 $x$ 密度可达的所有样本组成的集合记为 $X = \{x' \in D \mid x' \text{由 } x \text{ 密度可达}\}$ ，则 $X$ 为满足连接性与最大性的簇。

---

输入：样本集  $D = \{x_1, x_2, \dots, x_m\}$ ;  
邻域参数  $(\epsilon, MinPts)$ .

过程：

```
1: 初始化核心对象集合:  $\Omega = \emptyset$ 
2: for  $j = 1, 2, \dots, m$  do
3:   确定样本  $x_j$  的  $\epsilon$ -邻域  $N_\epsilon(x_j)$ ;
4:   if  $|N_\epsilon(x_j)| \geq MinPts$  then
5:     将样本  $x_j$  加入核心对象集合:  $\Omega = \Omega \cup \{x_j\}$ 
6:   end if
7: end for

8: 初始化聚类簇数:  $k = 0$ 
9: 初始化未访问样本集合:  $\Gamma = D$ 
10: while  $\Omega \neq \emptyset$  do
11:   记录当前未访问样本集合:  $\Gamma_{old} = \Gamma$ ;
12:   随机选取一个核心对象  $o \in \Omega$ , 初始化队列  $Q = < o >$ ;
13:    $\Gamma = \Gamma \setminus \{o\}$ ;
14:   while  $Q \neq \emptyset$  do
15:     取出队列  $Q$  中的首个样本  $q$ ;
16:     if  $|N_\epsilon(q)| \geq MinPts$  then
17:       令  $\Delta = N_\epsilon(q) \cap \Gamma$ ; 未访问过的可达样本
18:       将  $\Delta$  中的样本加入队列  $Q$ ;
19:        $\Gamma = \Gamma \setminus \Delta$ ;
20:     end if
21:   end while

22:    $k = k + 1$ , 生成聚类簇  $C_k = \Gamma_{old} \setminus \Gamma$ ;
23:    $\Omega = \Omega \setminus C_k$ 
24: end while
```

---

输出：簇划分  $C = \{C_1, C_2, \dots, C_k\}$

图 9.9 DBSCAN 算法

内层while循环一轮生成一个聚类簇

- 找出所有核心对象，任选一个核心对象作为种子
- 找核心对象密度可达的样本生成聚类簇

看书上例子

## 8.5. 层次聚类

在不同层次对数据集划分，形成树形的聚类结构

### 8.5.1. AGNES——自底向上

从单个样本为簇开始，每次选择距离最近的簇合并

集合间距离：

$$\text{最小距离: } d_{\min}(C_i, C_j) = \min_{x \in C_i, z \in C_j} \text{dist}(x, z), \quad (9.41)$$

$$\text{最大距离: } d_{\max}(C_i, C_j) = \max_{x \in C_i, z \in C_j} \text{dist}(x, z), \quad (9.42)$$

$$\text{平均距离: } d_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i} \sum_{z \in C_j} \text{dist}(x, z). \quad (9.43)$$

分别对应单链接、全链接、均链接算法

---

输入：样本集  $D = \{x_1, x_2, \dots, x_m\}$ ;  
聚类簇距离度量函数  $d$ ;  
聚类簇数  $k$ .

过程：

```
1: for  $j = 1, 2, \dots, m$  do
2:    $C_j = \{x_j\}$ 
3: end for
4: for  $i = 1, 2, \dots, m$  do
5:   for  $j = 1, 2, \dots, m$  do
6:      $M(i, j) = d(C_i, C_j);$ 
7:      $M(j, i) = M(i, j)$ 
8:   end for
9: end for
10: 设置当前聚类簇个数:  $q = m$ 
11: while  $q > k$  do
12:   找出距离最近的两个聚类簇  $C_{i^*}$  和  $C_{j^*}$ ;
13:   合并  $C_{i^*}$  和  $C_{j^*}$ :  $C_{i^*} = C_{i^*} \cup C_{j^*};$ 
14:   for  $j = j^* + 1, j^* + 2, \dots, q$  do
15:     将聚类簇  $C_j$  重编号为  $C_{j-1}$ 
16:   end for
17:   删除距离矩阵  $M$  的第  $j^*$  行与第  $j^*$  列;
18:   for  $j = 1, 2, \dots, q - 1$  do
19:      $M(i^*, j) = d(C_{i^*}, C_j);$ 
20:      $M(j, i^*) = M(i^*, j)$ 
21:   end for
22:    $q = q - 1$ 
23: end while
```

输出：簇划分  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$

---

图 9.11 AGNES 算法

流程

1. 初始化
2. 合并最近的
3. 更新距离矩阵

## 8.6. 降维与度量学习

### 关于考试

度量没考过，降维PCA一般会考

*chap13 半监督学习不考*

kNN k近邻

MDS 多维缩放

PCA 主成分分析/线性降维

KPCA 核化线性降维

Isomap 等度量映射

LLE 局部线性嵌入

NCA 近邻成分分析

LVW 包裹式特征选择

PGD 近端梯度下降

RIP 限制等距性