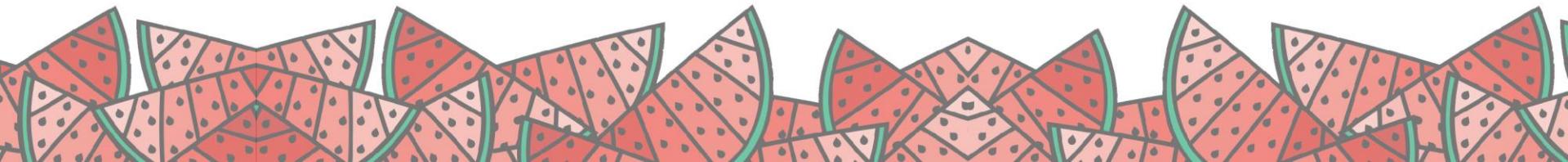


周志华 著

MACHINE
LEARNING

机器学习

清华大学出版社



第二章：模型评估与选择

大纲

□ 经验误差与过拟合

□ 评估方法

□ 性能度量

□ 比较检验

□ 偏差与方差

□ 阅读材料

经验误差与过拟合

□ 错误率&误差：

- 错误率：错分样本的占比： $E = a/m$
- 误差：样本真实输出与预测输出之间的差异
 - 训练(经验)误差：训练集上
 - 测试误差：测试集
 - 泛化误差：除训练集外所有样本

由于事先并不知道新样本的特征，我们只能努力使经验误差最小化；

很多时候虽然能在训练集上做到分类错误率为零，但多数情况下这样的学习器并不好

经验误差与过拟合

□ 过拟合：

学习器把训练样本学习的“太好”，将训练样本本身的特点当做所有样本的一般性质，导致泛化性能下降

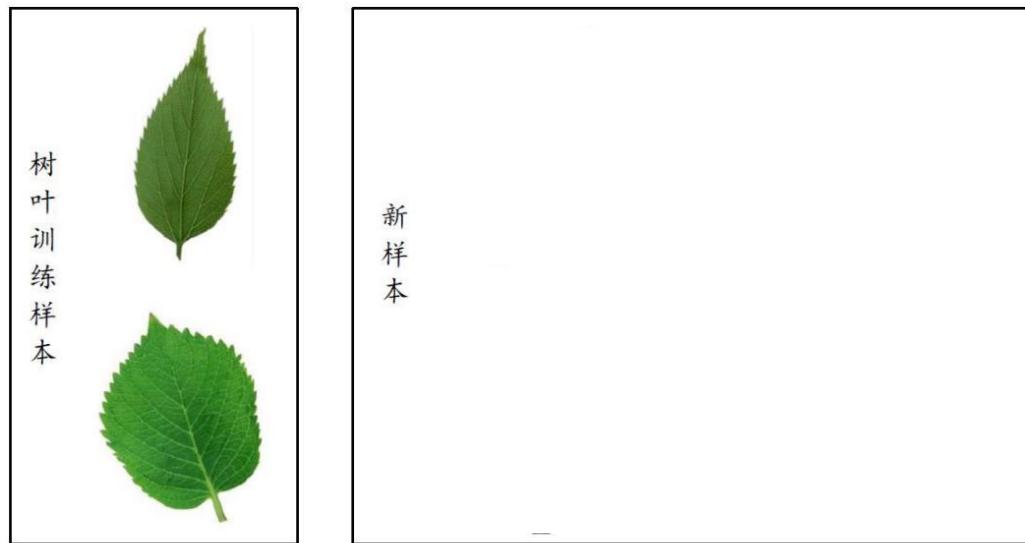
- 优化目标加正则项
- early stop

□ 欠拟合：

对训练样本的一般性质尚未学好

- 决策树：拓展分支
- 神经网络：增加训练轮数

经验误差与过拟合



过拟合、欠拟合的直观类比

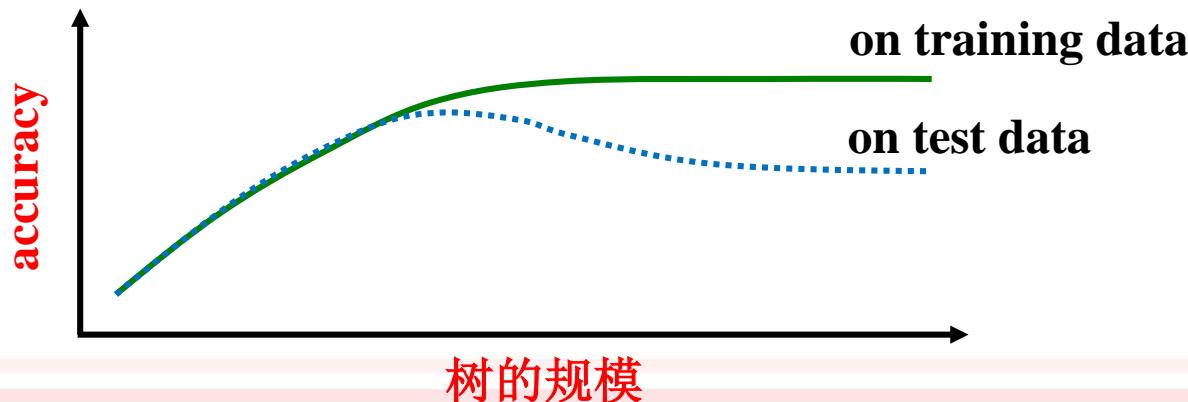
过拟合：学习器把训练样本本身特点当做所有潜在样本都会具有一般性质。

欠拟合：训练样本的一般性质尚未被学习器学好。

避免过度拟合数据

□ 过度拟合

- 对于一个假设，当存在其它的假设对训练样例的拟合比它差，但事实上在实例的整个分布上表现得却更好时，我们说这个假设过度拟合训练样例。
- 定义：给定一个假设空间 H ，一个假设 $h \in H$ ，如果存在其它的假设 $h' \in H$ ，使得在训练样例上 h 的错误率比 h' 小，但在整个实例分布上 h' 的错误率比 h 小，那么就说假设 h 过度拟合训练数据。

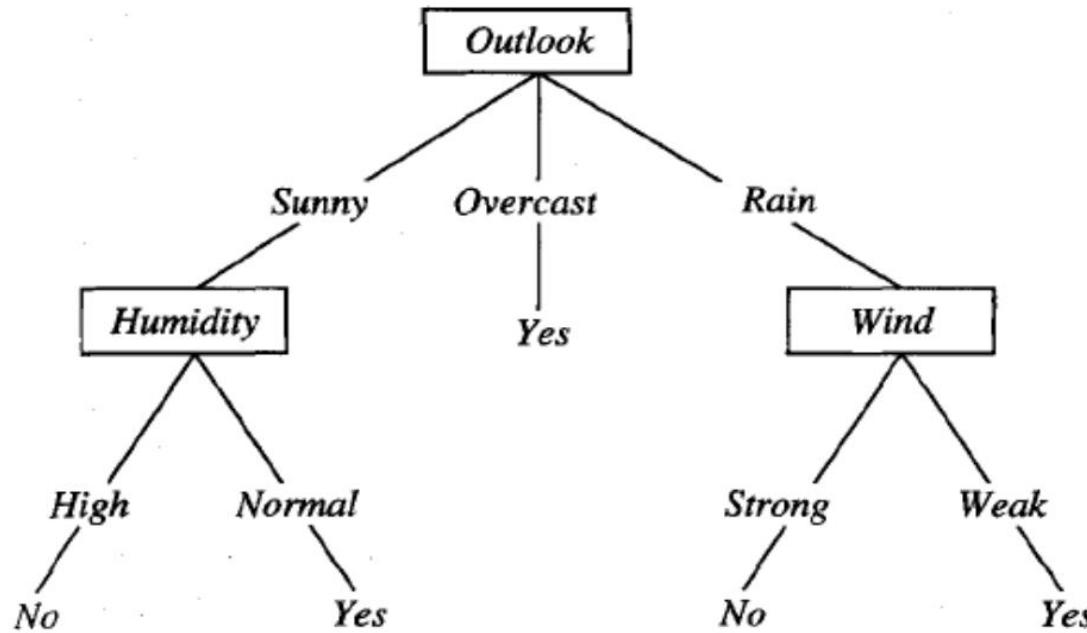


避免过度拟合数据 (2)

□ 导致过度拟合的原因 (1)

- 一种可能原因是训练样例含有随机错误或噪声

<Outlook=Sunny, Temperature=Hot, Humidity=Normal, Wind=Strong, PlayTennis=No>



避免过度拟合数据 (3)

□ 导致过度拟合的原因 (2)

- 当训练数据没有噪声时，过度拟合也有可能发生，特别是当少量的样例被关联到叶子节点时，很可能出现巧合的规律性，使得一些属性恰巧可以很好地分割样例，但却与实际的目标函数并无关系。
- 过度拟合使决策树的精度降低 (10—25) %

避免过度拟合数据 (4)

□ 避免过度拟合的方法

- 及早停止树增长
- 后修剪法

□ 两种方法的特点

- 第一种方法更直观
- 第一种方法中，精确地估计何时停止树增长很困难
- 第二种方法被证明在实践中更成功

避免过度拟合数据 (5)

□ 避免过度拟合的关键

- 使用什么样的准则来确定最终正确树的规模

□ 解决方法

- 使用与训练样例截然不同的一套分离的样例，来评估通过后修剪方法从树上修剪节点的功用。
- 使用所有可用数据进行训练，但进行统计测试来估计扩展（或修剪）一个特定的节点是否有可能改善在训练集合外的实例上的性能。
- 使用一个明确的标准来衡量训练样例和决策树的复杂度，当这个编码的长度最小时停止树增长。

避免过度拟合数据 (6)

□ 方法评述

- 第一种方法是最普通的，常被称为训练和验证集法。
- 可用数据分成两个样例集合：
 - 训练集合，形成学习到的假设
 - 验证集合，评估这个假设在后续数据上的精度
- 方法的动机：即使学习器可能会被训练集合误导，但验证集合不大可能表现出同样的随机波动
- 验证集合应该足够大，以便它本身可提供具有统计意义的实例样本。

大纲

□ 经验误差与过拟合

□ 评估方法

□ 性能度量

□ 比较检验

□ 偏差与方差

□ 阅读材料

评估方法

现实任务中往往会对学习器的泛化性能、时间开销、存储开销、可解释性等方面的因素进行评估并做出选择

我们假设测试集是从样本真实分布中独立采样获得，将测试集上的“测试误差”作为泛化误差的近似，所以测试集要和训练集中的样本尽量互斥。

评估方法

通常将包含 m 个样本的数据集 $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ 拆分成训练集 S 和测试集 T :

□ 留出法:

- 直接将数据集划分为两个互斥集合 (1000: 700/300)
- 训练/测试集划分要尽可能保持数据分布的一致性
 - 保留类比的采样: 分层采样
- 一般若干次随机划分、重复实验取平均值
 - 100次随机划分=100次结果
- 训练/测试样本比例设置 ? S大 ? T大 ?
 - 比例通常为2:1~4:1

评估方法

□ 交叉验证法：

将数据集分层采样划分为 k 个大小相似的互斥子集，每次用 $k-1$ 个子集的并集作为训练集，余下的子集作为测试集，最终返回 k 个测试结果的均值， k 最常用的取值是**10**.

评估方法

与留出法类似，将数据集 D 划分为 k 个子集同样存在多种划分方式，为了减小因样本划分不同而引入的差别（撒豆子）， k 折交叉验证通常随机使用不同的划分重复 p 次，最终的评估结果是这 p 次 k 折交叉验证结果的均值，例如常见的“10次10折交叉验证”

假设数据集 D 包含 m 个样本，若令 $k = m$ ，则得到留一法（LOO）：

- 不受随机样本划分方式的影响
- 结果往往比较准确
- 当数据集比较大时，计算开销难以忍受

评估方法

□ 自助法：

以自助采样法为基础，对数据集 D 有放回采 m 次得到训练集 D' , $D \setminus D'$ 用做测试集。

- 实际模型与预期模型都使用 m 个训练样本
- 约有 $1/3$ 的样本没在训练集中出现 ?
- 从初始数据集中产生多个不同的训练集，对集成学习有很大的好处
- 自助法在数据集较小、难以有效划分训练/测试集时很有用；由于改变了数据集分布可能引入估计偏差，在数据量足够时，留出法和交叉验证法更常用。

作业

- 数据集包含1000个样本，其中500个正例，500个反例，将其划分为包含70%样本的训练集和30%样本的测试集用于留出法评估，试估算共有多少种划分方式。

大纲

□ 经验误差与过拟合

□ 评估方法

□ 性能度量

□ 比较检验

□ 偏差与方差

□ 阅读材料

性能度量

性能度量是衡量模型泛化能力的评价标准，反映了任务需求；使用不同的性能度量往往会导致不同的评判结果

在预测任务中，给定样例集 $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ 评估学习器的性能 f 也即把预测结果 $f(\mathbf{x})$ 和真实标记比较。

回归任务最常用的性能度量是“均方误差”：

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

性能度量

对于分类任务,错误率和精度是最常用的两种性能度量:

- 错误率: 分错样本占样本总数的比例
- 精度: 分对样本占样本总数的比率

分类错误率

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

精度

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

性能度量

信息检索、Web搜索等场景中经常需要衡量正例被预测出来的比率或者预测出来的正例中正确的比率，此时查准率和查全率比错误率和精度更适合。

统计真实标记和预测结果的组合可以得到“混淆矩阵”

分类结果混淆矩阵

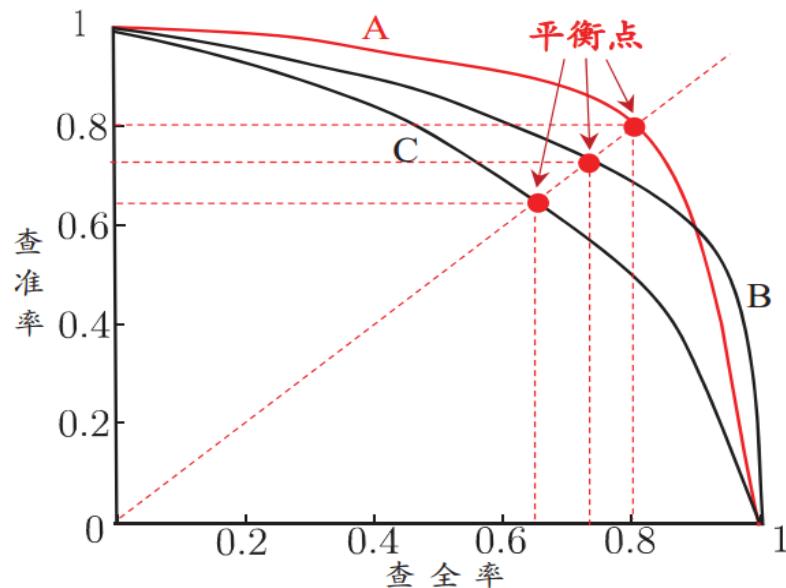
真实情况	预测结果	
	正例	反例
正例	TP (真正例)	FN (假反例)
反例	FP (假正例)	TN (真反例)

查准率 $P = \frac{TP}{TP + FP}$

查全率 $R = \frac{TP}{TP + FN}$

性能度量

根据学习器的预测结果按正例可能性大小对样例进行排序，并逐个把样本作为正例进行预测，则可以得到查准率-查全率曲线，简称“P-R曲线”



P-R曲线与平衡点示意图

平衡点是曲线上“查准率 = 查全率”时的取值，可用来用于度量P-R曲线有交叉的分类器性能高低

性能度量

比P-R曲线平衡点更常用的是F1度量：

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

比F1更一般的形式 F_β ,

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

$\beta = 1$: 标准F1

$\beta > 1$: 偏重查全率(逃犯信息检索)

$\beta < 1$: 偏重查准率(商品推荐系统)

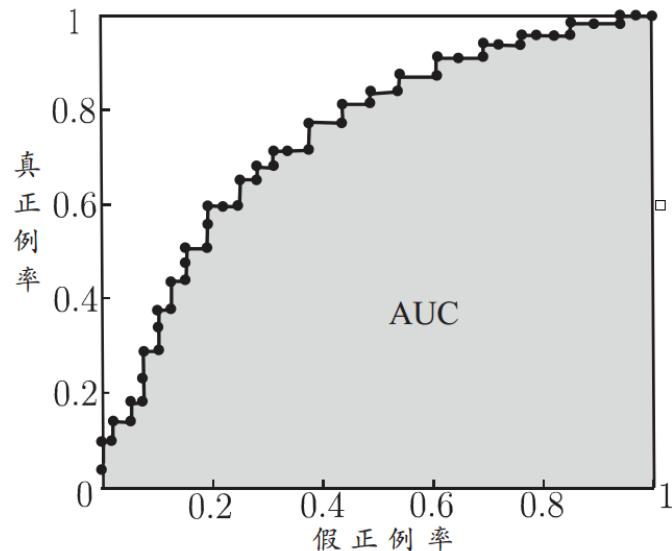
性能度量

类似P-R曲线，根据学习器的预测结果对样例排序，并逐个作为正例进行预测，以“假正例率”为横轴，“真正例率”为纵轴可得到ROC曲线，全称“受试者工作特征”.

ROC图的绘制：给定 m^+ 个正例和 m^- 个负例，根据学习器预测结果对样例进行排序，将分类阈值设为每个样例的预测值，当前标记点坐标为 (x, y) ，当前若为真正例，则对应标记点的坐标为 $(x, y + \frac{1}{m^+})$ ；当前若为假正例，则对应标记点的坐标为 $(x + \frac{1}{m^-}, y)$ ，然后用线段连接相邻点.

性能度量

若某个学习器的ROC曲线被另一个学习器的曲线“包住”，则后者性能优于前者；否则如果曲线交叉，可以根据ROC曲线下面积大小进行比较，也即AUC值。



基于有限样例绘制的 ROC 曲线
与 AUC

假设ROC曲线由 $\{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ 的点按序连接而形成($x_1 = 0, x_m = 1$)，则：
AUC可估算为：

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

AUC衡量了样本预测的排序质量。

代价敏感错误率

现实任务中不同类型的错误所造成的后果很可能不同，为了权衡不同类型错误所造成的不同损失，可为错误赋予“非均等代价”。

以二分类为例，可根据领域知识设定“代价矩阵”，如下表所示，其中 $cost_{ij}$ 表示将第 i 类样本预测为第 j 类样本的代价。损失程度越大， $cost_{01}$ 与 $cost_{10}$ 值的差别越大。

在非均等代价下，不再最小化错误次数，而是最小化“总体代价”，则“代价敏感”错误率相应的为：

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

代价曲线

在非均等代价下，ROC曲线不能直接反映出学习器的期望总体代价，而“代价曲线”可以。

代价曲线的横轴是取值为[0, 1]的正例概率代价

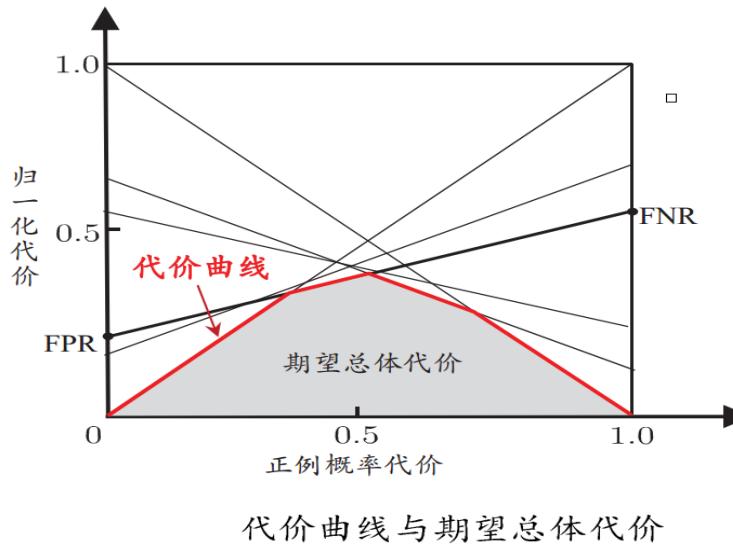
$$P(+)cost = \frac{p \times cost_{01}}{p \times cost_{01} + (1 - p) \times cost_{10}}$$

纵轴是取值为[0, 1]的归一化代价

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

代价曲线

代价曲线图的绘制：ROC曲线上每个点对应了代价曲线上的一条线段，设ROC曲线上点的坐标为(TPR, FPR)，则可相应计算出FNR，然后在代价平面上绘制一条从(0, FPR)到(1, FNR)的线段，线段下的面积即表示了该条件下的期望总体代价；如此将ROC曲线上的每个点转化为代价平面上的一条线段，然后取所有线段的下界，围成的面积即为所有条件下学习器的期望总体代价。



大纲

□ 经验误差与过拟合

□ 评估方法

□ 性能度量

□ 比较检验

□ 偏差与方差

□ 阅读材料

性能评估

□ 关于性能比较：

- 测试性能并不等于泛化性能
- 测试性能随着测试集的变化而变化
- 很多机器学习算法本身有一定的随机性

直接选取相应评估方法在相应度量下比大小的方法不可取！

假设检验为学习器性能比较提供了重要依据，基于其结果我们可以推断出若在测试集上观察到学习器A比B好，则A的泛化性能是否在统计意义上优于B，以及这个结论的把握有多大。

二项检验

记泛化错误率为 ϵ ，测试错误率为 $\hat{\epsilon}$ ，假定测试样本从样本总体分布中独立采样而来，我们可以使用“二项检验”对 $\epsilon \leq \epsilon_0$ 进行假设检验。

假设 $\epsilon \leq \epsilon_0$ ，若测试错误率小于

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

则在 α 的显著度下，假设不能被拒绝，也即能以 $1 - \alpha$ 的置信度认为，模型的泛化错误率不大于 ϵ_0 。

t检验

对应的，面对多次重复留出法或者交叉验证法进行多次训练/测试时可使用“t检验”。

假定得到了 k 个测试错误率， $\hat{\epsilon}_1, \hat{\epsilon}_2, \dots, \hat{\epsilon}_k$ ，假设 $\epsilon = \epsilon_0$ 对于显著度 α ，若 $[t_{-\alpha/2}, t_{\alpha/2}]$ 位于临界范围 $|\mu - \epsilon_0|$ 内，则假设不能被拒绝，即可认为泛化错误率 $\epsilon = \epsilon_0$ ，其置信度为 $1 - \alpha$ 。

交叉验证t检验

现实任务中，更多时候需要对不同学习器的性能进行比较

对两个学习器A和B, 若k折交叉验证得到的测试错误率分别为 $\epsilon_1^A, \dots, \epsilon_k^A$ 和 $\epsilon_1^B, \dots, \epsilon_k^B$ ，可用k折交叉验证“成对t检验”进行比较检验。若两个学习器的性能相同，则他们使用相同的训练/测试集得到的测试错误率应相同，即 $\epsilon_i^A = \epsilon_i^B$.

交叉验证t检验

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, 则差值应该为0, 继而用 $\Delta_1, \dots, \Delta_k$ 来对“学习器A与B性能相同”这个假设做t检验。

假设检验的前提是测试错误率为泛化错误率的独立采样, 然而由于样本有限, 使用交叉验证导致训练集重叠, 测试错误率并不独立, 从而过高估计假设成立的概率, 为缓解这一问题, 可采用“5*2交叉验证”法.

5*2交叉验证法

所谓5*2折交叉验证就是做5次二折交叉验证，每次二折交叉验证之前将数据打乱，使得5次交叉验证中的数据划分不重复。为缓解测试数据错误率的非独立性，仅计算第一次2折交叉验证结果的平均值 $\mu = 0.5(\Delta_1^1 + \Delta_1^2)$ 和每次二折实验计算得到的方差 $\sigma_i^2 = \left(\Delta_i^1 - \frac{\Delta_i^1 + \Delta_i^2}{2}\right)^2 + \left(\Delta_i^2 - \frac{\Delta_i^1 + \Delta_i^2}{2}\right)^2$ ，则变量

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

服从自由度为5的t分布。

McNemar检验

对于二分类问题，留出法不仅可以估计出学习器A和B的测试错误率，还能获得两学习器分类结果的差别，如下表所示

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 $e_{01} = e_{10}$
则 $|e_{01} - e_{10}|$ 应服从正态分布

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

服从自由度为1的 χ^2 分布。

Friedman检验

交叉验证t检验和McNemar检验都是在一个数据集上比较两个算法的性能，可以用Friedman检验在一组数据集上对多个算法进行比较。

假定用 D_1, D_2, D_3, D_4 四个数据集对算法 A, B, C 进行比较。

先使用留出法或者交叉验证法得到每个算法在每个数据集上的测试结果，然后在每个数据集上根据性能好坏排序，并赋序值 $1, 2, \dots$ ；若算法性能相同则平分序值，继而得到每个算法的平均序值 r_i 。

Friedman检验

得到表格如下所示，由平均序值进行Friedman检验来判断这些算法是否性能都相同。

算法比较序值表			
数据集	算法 A	算法 B	算法 C
D_1	1	2	3
D_2	1	2.5	2.5
D_3	1	2	3
D_4	1	2	3
平均序值	1	2.125	2.875

则变量：

$$\tau_{\chi^2} = \frac{12N}{k(k+1)} \left(\sum_{i=1}^k r_i^2 - \frac{k(k+1)^2}{4} \right)$$

服从自由度为 $k-1$ 的 χ^2 分布

其中 N, k 表示数据集和算法数目

Nemenyi后续检验

若“所有算法的性能相同”这个假设被拒绝，说明算法的性能显著不同，此时可用Nemenyi后续检验进一步区分算法。

Nemenyi检验计算平均序值差别的临界阈值

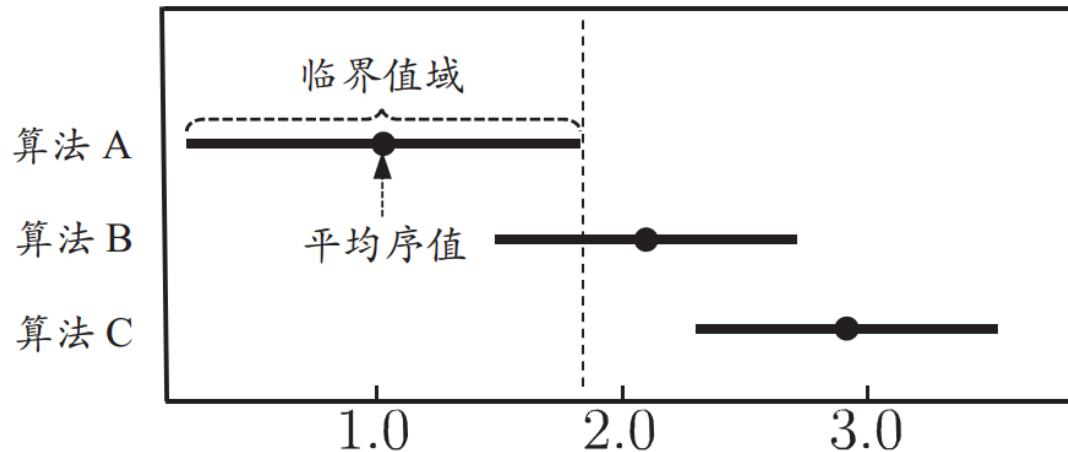
$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}$$

如果两个算法的平均序值之差超出了临界阈值CD，则以相应的置信度拒绝“两个算法性能相同”这一假设。

Friedman检验图

根据上例的序值结果可绘制如下Friedman检验图，横轴为平均序值，每个算法圆点为其平均序值，线段为临界阈值的大小。

若两个算法有交叠(A和B)，则说明没有显著差别；
否则有显著差别(A和C)，算法A明显优于算法C.



大纲

□ 经验误差与过拟合

□ 评估方法

□ 性能度量

□ 比较检验

□ 偏差与方差

□ 阅读材料

偏差与方差

通过实验可以估计学习算法的泛化性能，而“偏差-方差分解”可以用来帮助解释泛化性能。偏差-方差分解试图对学习算法期望的泛化错误率进行拆解。

对测试样本 x ,令 y_D 为 x 在数据集中的标记, y 为 x 的真实标记, $f(\mathbf{x}; D)$ 为训练集 D 上学得模型 f 在 x 上的预测输出。以回归任务为例：学习算法的期望预期为：

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

使用样本数目相同的不同训练集产生的方差为

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

噪声为

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

偏差与方差

期望输出与真实标记的差别称为偏差，即 $bias^2(\mathbf{x}) = (\bar{f}(\mathbf{x}) - y)^2$

为便与讨论，假定噪声期望为0，也即 $\mathbb{E}_D[y_D - y] = 0$ ，对泛化误差分解

$$\begin{aligned} E(f; D) &= \mathbb{E}_D \left[(f(\mathbf{x}; D) - y_D)^2 \right] \\ &= \mathbb{E}_D \left[(f(\mathbf{x}; D) - \bar{f}(\mathbf{x}) + \bar{f}(\mathbf{x}) - y_D)^2 \right] \\ &= \mathbb{E}_D \left[(f(\mathbf{x}; D) - \bar{f}(\mathbf{x}))^2 \right] + \mathbb{E}_D \left[(\bar{f}(\mathbf{x}) - y_D)^2 \right] \\ &\quad + \mathbb{E}_D \left[2(f(\mathbf{x}; D) - \bar{f}(\mathbf{x}))(\bar{f}(\mathbf{x}) - y_D) \right] \\ &= \mathbb{E}_D \left[(f(\mathbf{x}; D) - \bar{f}(\mathbf{x}))^2 \right] + \mathbb{E}_D \left[(\bar{f}(\mathbf{x}) - y_D)^2 \right] \end{aligned}$$

偏差与方差

$$\begin{aligned} &= \mathbb{E}_D \left[(f(\mathbf{x}; D) - \bar{f}(\mathbf{x}))^2 \right] + \mathbb{E}_D \left[(\bar{f}(\mathbf{x}) - y + y - y_D)^2 \right] \\ &= \mathbb{E}_D \left[(f(\mathbf{x}; D) - \bar{f}(\mathbf{x}))^2 \right] + \mathbb{E}_D \left[(\bar{f}(\mathbf{x}) - y)^2 \right] + \mathbb{E}_D \left[(y - y_D)^2 \right] \\ &\quad + 2\mathbb{E}_D \left[(\bar{f}(\mathbf{x}) - y)(y - y_D) \right] \end{aligned}$$

又由假设中噪声期望为0， 可得

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

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

也即泛化误差可分解为偏差、方差与噪声之和。

偏差与方差

$$E(f; D) = \textcolor{red}{bias^2(x)} + \textcolor{yellow}{var(x)} + \textcolor{blue}{\varepsilon^2}$$

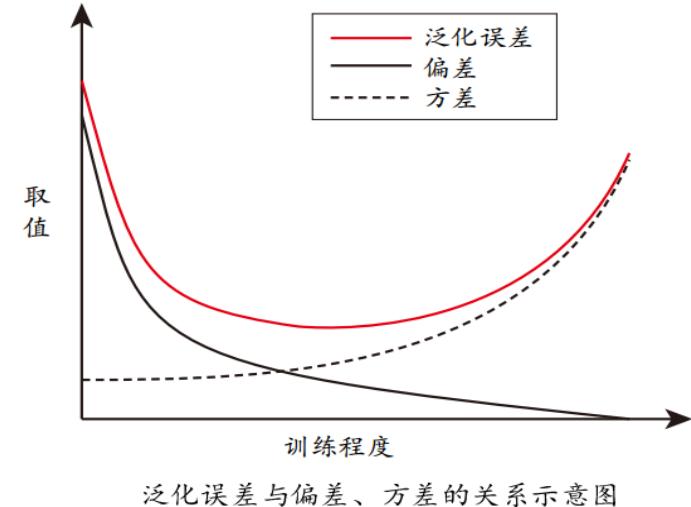
- 偏差度量了学习算法期望预测与真实结果的偏离程度；即刻画了学习算法本身的拟合能力；
- 方差度量了同样大小训练集的变动所导致的学习性能的变化；即刻画了数据扰动所造成的影响；
- 噪声表达了在当前任务上任何学习算法所能达到的期望泛化误差的下界；即刻画了学习问题本身的难度。

泛化性能是由学习算法的能力、数据的充分性以及学习任务本身的难度所共同决定的。给定学习任务为了取得好的泛化性能，需要使偏差小（充分拟合数据）而且方差较小（减少数据扰动产生的影响）。

偏差与方差

一般来说，偏差与方差是有冲突的，称为偏差-方差窘境。
如右图所示，假如我们能控制算法的训练程度：

- 在训练不足时，学习器拟合能力不强，训练数据的扰动不足以使学习器的拟合能力产生显著变化，此时偏差主导泛化错误率；
- 随着训练程度加深，学习器拟合能力逐渐增强，方差逐渐主导泛化错误率；
- 训练充足后，学习器的拟合能力非常强，训练数据的轻微扰动都会导致学习器的显著变化，若训练数据自身非全局特性被学到则会发生过拟合。



大纲

□ 经验误差与过拟合

□ 评估方法

□ 性能度量

□ 比较检验

□ 偏差与方差

□ 阅读材料

阅读材料

- 自助采样法在机器学习中有重要用途, [Efron and Tibshirani, 1993]对此有详细讨论。
- ROC曲线在二十世纪八十年代后期被引入机器学习 [Spackman, 1989], AUC则是从九十年代中期起在机器学习领域广为使用 [Bradley, 1997]. [Hand and Till, 2001]将ROC曲线从二分类任务推广到多分类任务. [Fawcett, 2006]综述了ROC曲线的用途.
- [Drummond and Holte, 2006]发明了代价曲线. 代价敏感学习 [Elkan, 2001; Zhou and Liu, 2006]专门研究非均等代价下的学习。

阅读材料

- [Dietterich, 1998]指出了常规k折交叉验证法存在的风险，并提出了5*2折交叉验证法。[Demsar, 2006]讨论了对多个算法进行比较检验的方法。

- [Geman et al., 1992]针对回归任务给出了偏差-方差-协方差分解，后来被简称为偏差-方差分解。但仅基于均方误差的回归任务中推导，对分类任务，由于0/1损失函数的跳变性，理论上推导出偏差-方差分解很困难。已有多种方法可通过试验队偏差和方差进行估计[Kong and Dietterich, 1995; Kohavi and Wolpert, 1996; Breiman, 1996; Friedman, 1997; Domingos, 2000].