

人工智能学习笔记

西瓜书学习笔记

烂石

2025 年 6 月 17 日



第一章 绪论

1.1 引言

略

1.2 基本术语

- (i) 样本/示例-sample/instance
- (ii) 训练集-training data
- (iii) 测试集-testing data
- (iv) 标记-label
- (v) 样例-example: 拥有 label 的 instance
- (vi) 泛化-generalization

1.3 假设空间

1.3.1 科学推理

- 归纳-induction 从具体的事实归结出一般性规律
- 演绎-deduction 从基础原理推演出具体状况

1.3.2 归纳学习-inductive learning

- 广义归纳学习
- 狭义归纳学习-概念学习 eg: 布尔概念学习

版本空间-version space

即存在着一个与训练集一致的”假设集合”

1.4 归纳偏好

有多个与训练集一致的假设, 但测试新样本时有不同的输出结果, 那么采用哪种模型(假设)?

1.4.1 ”奥卡姆剃刀”原则

若有多个假设与观察一致, 则选最简单的那个. 利用什么原则, 取决于算法能否获得更好的性能, 泛化能力是否更强

1.4.2 NFL(No Free Lunch Theorem) 定理-”没有免费的午餐”定理

定理 1.4.1 (No Free Lunch 定理). 对于所有学习算法 \mathcal{L}_a 和 \mathcal{L}_b , 在均匀分布的目标函数空间下, 它们的训练外误差满足:

$$\sum_f E_{ote}(\mathcal{L}_a|X, f) = \sum_f E_{ote}(\mathcal{L}_b|X, f)$$

证明.

步骤 1.4.1.1 (定义与假设). 假设样本空间 X 和假设空间 \mathcal{H} 是离散的. 定义:

$$E_{ote}(\mathcal{L}_a|X, f) = \sum_{h \in \mathcal{H}} \sum_{x \in X-X} P(x) \cdot \mathbb{I}(h(x) \neq f(x)) \cdot P(h|X, \mathcal{L}_a)$$

其中 $\mathbb{I}(\cdot)$ 为指示函数。

步骤 1.4.1.2 (总误差求和). 对所有目标函数求和:

$$\sum_f E_{ote}(\mathcal{L}_a|X, f) = \sum_f \sum_h \sum_{x \in X-X} P(x) \mathbb{I}(h(x) \neq f(x)) P(h|X, \mathcal{L}_a) \quad (1.1)$$

步骤 1.4.1.3 (交换求和顺序). 将 \sum_f 移至内部:

$$\begin{aligned} &= \sum_{x \in X-X} P(x) \sum_h P(h|X, \mathcal{L}_a) \\ &\quad \times \underbrace{\sum_f \mathbb{I}(h(x) \neq f(x))}_{\text{关键项}} \end{aligned} \quad (1.2)$$

步骤 1.4.1.4 (计算关键项). 对于二分类问题, 每个 x 处的 $f(x)$ 有等概率取 0 或 1:

$$\sum_f \mathbb{I}(h(x) \neq f(x)) = \frac{1}{2} \cdot 2^{|X|} = 2^{|X|-1}$$

步骤 1.4.1.5 (最终化简). 代入关键项并利用 $\sum_h P(h|X, \mathcal{L}_a) = 1$:

$$\text{原式} = 2^{|X|-1} \sum_{x \in X-X} P(x)$$

该结果与算法 \mathcal{L}_a 无关, 故对任意 $\mathcal{L}_a, \mathcal{L}_b$:

$$\sum_f E_{ote}(\mathcal{L}_a|X, f) = \sum_f E_{ote}(\mathcal{L}_b|X, f)$$

□

由??可知, 脱离具体问题, 空谈” 什么学习算法最好” 是毫无意义的

1.5 发展历程

推理期:1950s-1970s—符号知识, 演绎推理

知识期:1970s 中期—符号知识, 领域知识

学习期:1980s—机器学习, 归纳逻辑程序设计 (Inductive Logic Programming)

统计学习:1990s 中期—向量机 (Support Vector Machine), 核方法 (Kernel Methods)

深度学习:2000s—神经网络

1.6 应用现状

信息科学, 自然科学...

1.7 阅读材料

1.7.1 机器学习

国际会议:ICML,NIPS, COLT,ECML(Europe),ACML(Asia)

国际期刊:JMLR,ML

国内会议:CCML,MLA

人工智能

国际会议:IJCAI,AAAI

国际期刊:AI,JAIR

数据挖掘

国际会议:KDD,ICDM

国际期刊:ACM-TKDD,DMKD

1.7.2 计算机视觉

CVPR(会议),IEEE-TPAMI(期刊)

1.7.3 神经网络

期刊:NC,IEEE-TNNLS

统计学

期刊:AS

1.8 习题

1.8.1 表 1.1 中若只包含编号为 1 和 4 的两个样例，试给出相应的版本空间。

表 1.1 西瓜数据集

编号	色泽	根蒂	敲声	好瓜
1	青绿	蜷缩	浊响	是
2	乌黑	蜷缩	浊响	是
3	青绿	硬挺	清脆	否
4	乌黑	稍蜷	沉闷	否

图 1.1: 表 1.1 西瓜数据集

解:

1.8.2 估算共有多少种可能的假设

与使用单个合取式来进行假设表示相比，使用“析合范式”将使得假设空间具有更强的表示能力。例如

好瓜 \Leftrightarrow (色泽 = ★) \wedge (根蒂 = 蜷缩 \wedge 敲声 = ★)

\vee (色泽 = 乌黑 \wedge 根蒂 = ★) \wedge 敲声 = 沉闷),

表 1.1: 版本空间表

色泽	根蒂	敲声	逻辑表达式
青绿	瓣缩	浊响	$(\text{色泽} = \text{青绿}) \wedge (\text{根蒂} = \text{瓣缩}) \wedge (\text{敲声} = \text{浊响})$
青绿	瓣缩	*	$(\text{色泽} = \text{青绿}) \wedge (\text{根蒂} = \text{瓣缩})$
青绿	*	浊响	$(\text{色泽} = \text{青绿}) \wedge (\text{敲声} = \text{浊响})$
*	瓣缩	浊响	$(\text{根蒂} = \text{瓣缩}) \wedge (\text{敲声} = \text{浊响})$
青绿	*	*	$(\text{色泽} = \text{青绿})$

会把“(色泽 = 青绿) (根蒂 = 蜷缩) (敲声 = 清脆)”以及“(色泽 = 乌黑) (根蒂 = 硬挺) (敲声 = 沉闷)”都分类为“好瓜”。若使用最多包含 k 个合取式的析合范式来表达表 1.1 西瓜分类问题的假设空间，试估算共有多少种可能的假设。

解: 色泽包含两种情况 (青绿, 乌黑), 三种选择 (青绿, 乌黑, *);
根蒂 (蜷缩, 稍蜷, 硬挺) 共 4 种选择;
敲声 (浊响/沉闷/清脆) 共 4 种选择;

除了不能存在 (*, *, *) 的组合, 共包含 $3 \times 4 \times 4 - 1 = 47$ 种组合。
题目求 k 个合取式的所有可能的组合之和, 则有

$$\sum_i^k \binom{47}{i} \quad (1.3)$$

1.8.3 若数据包含噪声, 则假设空间中有可能不存在与所有训练样本都一致的假设。在此情形下, 试设计一种归纳偏好用于假设选择。

解: 刚入门, 可能无法正确回答此问题, 但存在的方法应该有权重噪声注入/梯度稳定性惩罚/稀疏性偏好/集成鲁棒性。

1.8.4 试证明“没有免费的午餐定理”仍成立。

本章 1.4 节在论述“没有免费的午餐”定理时, 默认使用了“分类错误率”作为性能度量来对分类器进行评估。若换用其他性能度量 ℓ , 则式 (1.1) 将改为

$$E_{ote}(\mathcal{L}_a | X, f) = \sum_h \sum_{x \in \mathcal{X}-X} P(x) \ell(h(x), f(x)) P(h | X, \mathcal{L}_a)$$

试证明“没有免费的午餐定理”仍成立。

证明.

步骤 1.8.0.1. 性能度量虽发生了改变, 目标函数仍然均匀分布, 总误差表达式为:

$$\begin{aligned} E_{ote}(\mathcal{L}_a|X, f) &= \sum_h \sum_{x \in \mathcal{X}-X} P(x) \ell(h(x), f(x)) P(h|X, \mathcal{L}_a) \\ &= \sum_{x \in \mathcal{X}-X} P(x) \sum_h P(h|X, \mathcal{L}_a) \sum_f \ell(h(x), f(x)) \end{aligned}$$

步骤 1.8.0.2. 由??证明过程可知: $\sum_f f(x) = 2^{|\mathcal{X}|-1}$

$$\therefore \sum_f \ell(h(x), f(x)) = 2^{|\mathcal{X}|-1} [\ell(h(x), 0) + \ell(h(x), 1)]. \quad (1.4)$$

步骤 1.8.0.3.

$$\because P(h(x) = 0|X, \mathcal{L}_a) = P(h(x) = 1|X, \mathcal{L}_a) = \frac{1}{2}. \therefore \sum_h P(h|X, \mathcal{L}_a) [\ell(h(x), 0) + \ell(h(x), 1)] = \frac{1}{2} [\ell(0, 0) + \ell(0, 1) + \ell(1, 0) + \ell(1, 1)] \quad (1.5)$$

此结果与算法 \mathcal{L}_a 无关。

$$\sum_f E_{ote}(\mathcal{L}_a|X, f) = 2^{|\mathcal{X}|-1} \cdot \frac{1}{2} \sum_{x \in \mathcal{X}-X} P(x) [\ell(0, 0) + \ell(0, 1) + \ell(1, 0) + \ell(1, 1)]. \quad (1.6)$$

□

无论选择何种性能度量 ℓ , 只要真实目标函数 f 在所有可能的函数上均匀分布, 无免费午餐定理仍然成立。算法的平均性能仅由数据分布和度量 ℓ 的对称性决定, 而与算法本身的设计无关。

1.8.5 试述机器学习能在互联网搜索的哪些环节起什么作用。

机器学习贯穿搜索的全流程, 从理解用户意图到动态优化结果, 其核心价值在于通过数据驱动的方式提升搜索效率、准确性和个性化程度。知识库中提到的超参数优化 (如随机搜索)、分布式表示 (如协同过滤) 等技术均为此提供了方法论支持。机器学习贯穿搜索的全流程, 从理解用户意图到动态优化结果, 其核心价值在于通过数据驱动的方式提升搜索效率、准确性和个性化程度。知识库中提到的超参数优化 (如随机搜索)、分布式表示 (如协同过滤) 等技术均为此提供了方法论支持。

第二章 模型评估与选择

2.1 经验误差与拟合

1. 错误率: m 个样本中有 a 个样本分类错误, 则错误率 $E = a/m$.
2. 精度: $1 - a/m$
3. 误差: 学习器的实际预测输出与样本的真实输出之间的差异; 训练集上的叫训练误差/经验误差, 新样本上的叫泛化误差;
4. 过拟合 (更容易遇到的情况): 将个例的特殊性错误地视为样本的普遍性; 欠拟合: 没有学好.
过拟合无法避免 ($N=NP$ 问题尚未证明), 只能缓解或减小其风险
5. 模型选择: 在多种学习算法中, 选择合适的算法中的合适的参数配置.

2.2 评估方法

1. 测试集测试学习器对新样本的判断力, 将测试误差近似于泛化误差. 测试集尽可能的不出现在训练集 (老师更希望学生有举一反三的能力)
2. 如果只有一个数据集, 既要训练又要测试, 如何解决?

2.2.1 留出法

直接将数据集 D 划分为两个互斥的集合, 其中一个集合作为训练集 S , 另一个作为测试集 T . 在 S 上训练出模型后用 T 来评估其测试误差.

1. 训练/测试集的划分要尽可能保持数据分布的一致性 (分类任务时, 至少要保持样本的类别比例相似) 采样角度上看, 则保留类别比例的采样方式通常称为分层采样

例如: 通过对 D 进行分层采样而获得含 70% 样本的训练集 S 和含 30% 样本的测试集 T , 若 D 包含 500 个正例、500 个反例, 则分层采样得到的 S 应包含 350 个正例、350 个反例, 而 T 则包含 150 个正例和 150 个反例; 若 S 、 T 中样本类别比例差别很大, 则误差估计将由于训练/测试数据分布的差异而产生偏差.

2. 单次使用留出法得到的估计结果往往不够稳定可靠, 在使用留出法时, 一般要采用若干次随机划分、重复进行实验评估后取平均值作为留出法的评估结果.

例如进行 100 次随机划分, 每次产生一个训练/测试集用于实验评估, 100 次后就得到 100 个结果, 而留出法返回的则是这 100 个结果的平均.

3. 局限性: 若令训练集 S 包含绝大多数样本, 则训练出的模型可能更接近于用 D 训练出的模型, 但由于 T 比较小, 评估结果可能不够稳定准确; 若令测试集 T 多包含一些样本, 则训练集 S 与 D 差别更大了, 被评估的模型与用 D 训练出的模型相比可能有较大差别, 从而降低了评估结果的保真性 (fidelity). 这个问题没有完美的解决方案, 常见做法是将大约 $2/3 \sim 4/5$ 的样本用于训练, 剩余样本用于测试.

2.2.2 交叉验证法

将数据集分为 k 个相似的子集, 每一折交叉验证时, $k-1$ 为训练集, 剩余的一个为测试集, 共 k 次不同的测试集, 为 k 折交叉验证法. 交叉验证法同样需要进行随机划分, 例如 10 次 10 折交叉验证法和 100 次留出法都需要进行 100 次训练/测试.

留一法 (LOO), 数据集共 m 个, 当 $k=m$ 时, 为留一法. 训练出来的模型会更准确, 但计算开销较大.

2.2.3 自助法 (Bootstrapping)

有放回的重重复随机抽样, 执行 m 次得到 m 个样本的数据集 D' . 其中, 某些样本始终不会被抽中的概率为 $\lim_{m \rightarrow \infty} (1 - \frac{1}{m})^m \mapsto \frac{1}{e} \approx 0.368$ 自助法适用于小样本

2.2.4 小结

方法	数据划分方式	数据利用率	计算成本	适用场景
留出法	单次划分训练集/测试集	低	低	大数据快速验证
交叉验证	多次划分 (k 个子集轮流验证)	高	中	中等数据模型调优
留一法	每次留 1 个样本验证	最高	高	小样本高精度评估
自助法	有放回重采样生成多样本集	灵活	中	小样本统计推断、不确定性估

选择建议

- 数据量大且需快速验证: 留出法。
- 中等数据调参: k -折交叉验证 (如 5 折或 10 折)。
- 小样本高精度需求: 留一法 (但需权衡计算成本)。
- 统计量分布估计或小样本: 自助法。

2.2.5 调参与最终模型

调参对最终模型有关键性影响, 通过验证集来评估模型的好坏。

2.3 性能度量

使用不同的性能度量会导致不同的评判结果。比如，回归任务最常用的性能度量是“均方误差”：

$$E(f; D) = \frac{1}{m} \sum_{i=1}^m m(f(x_i) - y_i)^2 \quad (2.1)$$

更一般的，对于数据分布 D 和概率密度函数 $p(\cdot)$ ，均方误差可表示为：

$$E(f; D) = \int_{x \sim D} (f(x) - y)^2 p(x) dx \quad (2.2)$$

2.3.1 错误率与精度

对于样例集 D ，分类错误率定义为：

$$E(f; D) = \frac{1}{m} \sum_{i=1}^m \mathbb{I}(f(x_i) \neq y_i) \quad (2.3)$$

精度定义为：

$$\begin{aligned} \text{acc}(f; D) &= 1 - E(f; D) \\ &= \frac{1}{m} \sum_{i=1}^m \mathbb{I}(f(x_i) = y_i) \end{aligned} \quad (2.4)$$

更一般的，对于数据分布 D 和概率密度函数 $p(\cdot)$ ，错误率可表示为：

$$E(f; D) = \int_{x \sim D} \mathbb{I}(f(x) \neq y) p(x) dx \quad (2.5)$$

精度可表示为：

$$\begin{aligned} \text{acc}(f; D) &= 1 - E(f; D) \\ &= \int_{x \sim D} \mathbb{I}(f(x) = y) p(x) dx \end{aligned} \quad (2.6)$$

2.3.2 查准率、查全率与 F1 值

查准率（Precision）和查全率（Recall）是分类任务中常用的性能度量，尤其在处理不平衡数据集时非常重要。查准率亦称准确率，表示预测为正例的样本中实际为正例的比例：

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2.7)$$

其中, TP 表示真正例（True Positives），即被正确预测为正例的样本数；FP 表示假正例（False Positives），即被错误预测为正例的样本数。查全率亦称召回率，表示实际为正例的样本中被正确预测为正例的比例：

$$\text{Recall} = \frac{TP}{TP + FN} \quad (2.8)$$

其中,TP 表示真正例, FN 表示假负例 (False Negatives), 即被错误预测为负例的正例样本数。查准率和查全率之间通常存在权衡关系: 提高查准率可能会降低查全率, 反之亦然。P-R 曲线 (Precision-Recall Curve) 可以帮助可视化这种权衡关系。如图??所示, P-R 曲线展示了不同阈值下查准率和查全率的变化情况。

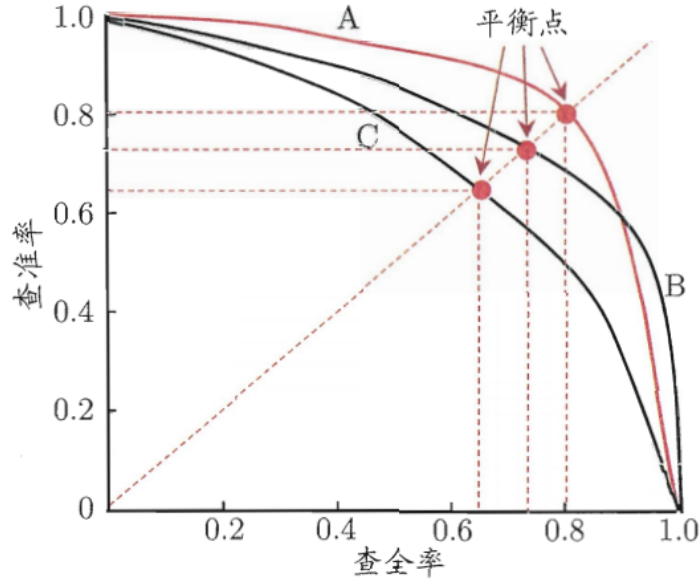


图 2.1: P-R 曲线示例

虽然 BEP 是查准率 = 查全率时的点, 但在实际应用中, 查准率和查全率通常不会相等, 因此需要综合考虑两者的平衡。F1 值是查准率和查全率的调和平均数, 常用来综合评估分类器的性能:

$$F1 = \frac{2 \times P \times R}{P + R} = \frac{2 \times TP}{\text{样例总数} + TP - TN} \quad (2.9)$$

但在实际应用中, 查准率和查全率的权衡关系通常需要根据具体任务和数据集来调整。所以 F1 的一般形式— F_β , 加权调和平均数能够根据任务需求调整查准率和查全率的权

术平均数和重:
平均数相比,
平均数更重
小值

$$F_\beta = \frac{(1 + \beta^2) \times P \times R}{\beta^2 \times P + R} \quad (2.10)$$

对于有多个二分类混淆矩阵的多分类任务, 可以使用宏平均 (Macro-Averaging) 和微平均 (Micro-Averaging) 来计算整体的查准率、查全率和 F1 值。宏平均是对每个类别单独计算查准率和查全率, 然后取平均值:

$$\begin{aligned} P_{macro} &= \frac{1}{n} \sum_{i=1}^n P_i \\ R_{macro} &= \frac{1}{n} \sum_{i=1}^n R_i \\ F1_{macro} &= \frac{2 \times P_{macro} \times R_{macro}}{P_{macro} + R_{macro}} \end{aligned} \quad (2.11)$$

微平均则是将所有类别的 TP、FP 和 FN 加总后计算查准率和查全率：

$$\begin{aligned} P_{micro} &= \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n TP_i + \sum_{i=1}^n FP_i} \\ R_{micro} &= \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n TP_i + \sum_{i=1}^n FN_i} \\ F1_{micro} &= \frac{2 \times P_{micro} \times R_{micro}}{P_{micro} + R_{micro}} \end{aligned} \quad (2.12)$$

2.3.3 ROC 曲线与 AUC

ROC 曲线（Receiver Operating Characteristic Curve）是评估二分类模型性能的常用工具，通过绘制真正率（TPR）与假正率（FPR）的关系来展示分类器在不同阈值下的表现。真正率（TPR）也称为查全率（Recall），表示实际正例中被正确预测为正例的比例：

$$TPR = \frac{TP}{TP + FN} \quad (2.13)$$

假正率（FPR）表示实际负例中被错误预测为正例的比例：

$$FPR = \frac{FP}{FP + TN} \quad (2.14)$$

ROC 曲线通过改变分类阈值，计算不同阈值下的 TPR 和 FPR，从而绘制出一条曲线。理想情况下，ROC 曲线应尽可能接近左上角（TPR=1, FPR=0），表示模型能够正确识别所有正例且不误判负例。

AUC（Area Under the Curve）是 ROC 曲线下的面积，表示模型的整体性能。AUC 值介于 0 和 1 之间，值越大表示模型性能越好。AUC=0.5 表示模型没有区分能力，相当于随机猜测；AUC=1 表示完美分类器。

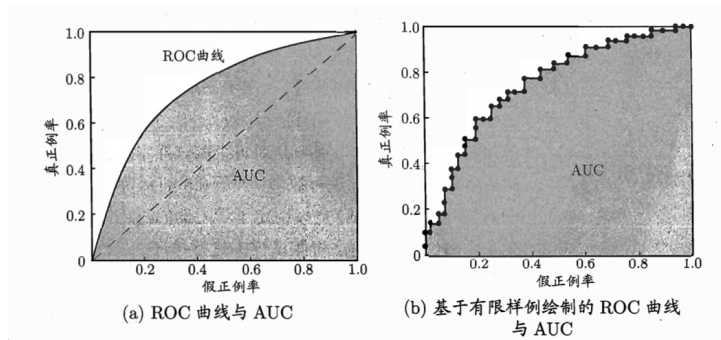


图 2.2: ROC 曲线示例

AUC 可以通过计算 ROC 曲线下的积分来获得，常用的计算方法包括梯形法则和蒙特卡洛积分等。估算公式为：

$$AUC = \frac{1}{2} \sum_{i=1}^{n-1} (FPR_{i+1} - FPR_i)(TPR_{i+1} + TPR_i) \quad (2.15)$$

形式上看,AUC 考虑的是样本预测的排序质量,因此它与排序误差紧密相连,给定 m^+ 个正例和 m^- 个反例,令 D^+ 和 D^- 分别为正例和反例的样本集

则 loss 可以表示为:

$$\ell_{rank} = \frac{1}{m^+m^-} \sum_{x^+ \in D^+} \sum_{x^- \in D^-} (\mathbb{I}(f(x^+) < f(x^-)) + \frac{1}{2}\mathbb{I}(f(x^+) = f(x^-))) \quad (2.16)$$

容易看出, ℓ_{rank} 对应的是 ROC 曲线上的面积,故 AUC 可以表示为:

$$AUC = 1 - \ell_{rank} \quad (2.17)$$

2.3.4 代价敏感错误率与代价曲线

非均等代价: 在某些应用中,不同类型的错误可能具有不同的代价,例如在医疗诊断中,漏诊(假阴性)可能比误诊(假阳性)更严重. 代价敏感错误率: 在这种情况下,需要引入代价敏感错误率来衡量模型的性能,定义为:

$$E_{f;D;cost} = \frac{1}{m} \left(\sum_{x_i \in D^+} \mathbb{I}(f(x_i) \neq y_i) \cdot cost_{FP} + \sum_{x_i \in D^-} \mathbb{I}(f(x_i) \neq y_i) \cdot cost_{FN} \right) \quad (2.18)$$

其中, $cost_{FP}$ 和 $cost_{FN}$ 分别表示假正例和假负例的代价. 代价曲线: 代价曲线是代价敏感错误率随分类阈值变化的图形表示,可以帮助选择最优的分类阈值以最小化总代价. 其中,横轴是取值为 $[0,1]$ 的正例概率代价:

$$P(+)cost = \frac{p \cdot cost_{FP}}{p \cdot cost_{FP} + (1 - p) \cdot cost_{FN}} \quad (2.19)$$

其中, p 为正例的概率,纵轴是取值为 $[0,1]$ 的归一化代价:

$$cost_{norm} = \frac{FNR \cdot p \cdot cost_{FP} + FPR \cdot (1 - p) \cdot cost_{FN}}{p \cdot cost_{FP} + (1 - p) \cdot cost_{FN}} \quad (2.20)$$

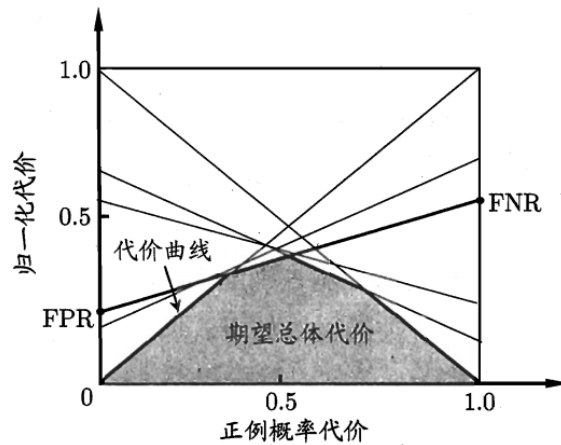


图 2.3: 代价曲线示例

2.4 比较检验

2.4.1 假设检验

假设检验是一种统计方法，用于判断样本数据是否支持某个假设。常见的假设检验包括 t 检验、卡方检验等。在模型评估中，假设检验可以用于比较不同模型的性能差异。泛化错误率是指模型在未见过的数据上的预测错误率。为了评估模型的泛化能力，可以使用假设检验的方法。测试错误率是指模型在测试集上的预测错误率。通过比较测试错误率和泛化错误率，可以判断模型是否存在过拟合现象。在包含 m 个样本的测试集上，泛化错误率为 ϵ 的学习器被测得测试率为 $\hat{\epsilon}$ 的概率为：

$$P(\hat{\epsilon}; \epsilon) = \binom{m}{\hat{\epsilon} \times m} \times \epsilon^{\hat{\epsilon} \times m} \times (1 - \epsilon)^{(1 - \hat{\epsilon}) \times m} \quad (2.21)$$

假设检验的基本步骤如下：

1. 提出原假设 (H_0) 和备择假设 (H_1)。例如， H_0 : 两个模型的性能相同； H_1 : 两个模型的性能不同。
2. 选择显著性水平 (α)，通常为 0.05 或 0.01。
3. 计算检验统计量，如 t 统计量或卡方统计量。
4. 根据检验统计量和自由度查找临界值或计算 p 值。
5. 如果 p 值小于显著性水平，则拒绝原假设，认为两个模型的性能存在显著差异；否则，接受原假设。

2.4.2 交叉验证 t 检验

交叉验证 t 检验是用于比较两个模型在交叉验证中的性能差异的统计方法。假设有两个模型 A 和 B，分别在 k 折交叉验证中得到 k 个性能指标（如准确率、F1 值等），可以使用 t 检验来判断这两个模型的性能是否存在显著差异。t 检验的基本步骤如下：

1. 计算两个模型在 k 折交叉验证中的性能指标的均值和标准差。
2. 计算 t 统计量：

$$t = \frac{\bar{x}_A - \bar{x}_B}{\sqrt{\frac{s_A^2}{k} + \frac{s_B^2}{k}}} \quad (2.22)$$

其中， \bar{x}_A 和 \bar{x}_B 分别为模型 A 和 B 的均值， s_A^2 和 s_B^2 分别为模型 A 和 B 的方差。

3. 根据 t 统计量和自由度 ($df = 2k - 2$) 查找 t 分布表，得到 p 值。
4. 如果 p 值小于显著性水平（如 0.05），则认为两个模型的性能存在显著差异。

5. 通过比较两个模型的性能指标（如准确率、F1 值等），选择性能更优的模型。
6. 如果 p 值大于显著性水平，则认为两个模型的性能差异不显著，可以选择任意一个模型。

2.4.3 McNemar 检验

McNemar 检验是一种用于比较两个分类模型在同一数据集上的性能差异的非参数统计方法。它主要用于二分类问题，适用于评估两个模型在相同样本上的预测结果是否存在显著差异。McNemar 检验的基本步骤如下：

1. 准备一个 2×2 的混淆矩阵，记录两个模型在同一数据集上的预测结果。矩阵的四个元素分别表示：
 - a : 模型 A 正确预测为正例，模型 B 也正确预测为正例。
 - b : 模型 A 正确预测为正例，模型 B 错误预测为负例。
 - c : 模型 A 错误预测为负例，模型 B 正确预测为正例。
 - d : 模型 A 错误预测为负例，模型 B 也错误预测为负例。

2. 计算检验统计量：

$$X^2 = \frac{(b - c)^2}{b + c} \quad (2.23)$$

其中， b 和 c 分别是模型 A 和 B 在同一样本上的错误分类数。

3. 根据自由度（ $df=1$ ）查找卡方分布表，得到 p 值。
4. 如果 p 值小于显著性水平（如 0.05），则认为两个模型的性能存在显著差异；否则，接受原假设，认为两个模型的性能差异不显著。

2.4.4 Freedman 检验与 Nemenyi 后续检验

Freedman 检验是一种非参数统计方法，用于比较多个分类模型在同一数据集上的性能差异。它适用于多分类问题，特别是在多个模型之间进行比较时。

Nemenyi 后续检验是一种用于比较多个模型的成对差异的统计方法。它通常在 Freedman 检验之后进行，以确定哪些模型之间存在显著差异。

1. 准备一个 $k \times n$ 的矩阵，其中 k 为模型数量， n 为样本数量。矩阵的每个元素表示模型在对应样本上的预测结果。
2. 计算每个模型的平均排名，并计算每个模型的排名差异。

3. 计算 Freedman 检验的统计量:

$$Q = \frac{12}{nk(k+1)} \sum_{j=1}^k R_j^2 - 3n(k+1) \quad (2.24)$$

其中, R_j 为第 j 个模型的排名和。

4. 根据自由度 ($df=k-1$) 查找卡方分布表, 得到 p 值。
5. 如果 p 值小于显著性水平 (如 0.05), 则认为至少有两个模型之间存在显著差异。
6. 进行 Nemenyi 后续检验, 计算每对模型之间的成对差异, 并根据差异大小进行排序。
7. 根据 Nemenyi 检验的结果, 确定哪些模型之间存在显著差异, 并进行成对比较。

2.4.5 各种检验的比较

检验方法	适用场景	优点	缺点
假设检验	比较两个模型的性能差异	简单易懂	需要满足一定的假设
交叉验证 t 检验	比较两个模型在交叉验证中的性能差异	考虑了数据的划分	需要多次训练模型
McNemar 检验	比较两个分类模型在同一数据集上的性能差异	适用于二分类问题	仅适用于同一数据集
Freedman 检验	比较多个分类模型在同一数据集上的性能差异	适用于多分类问题	需要满足一定的假设
Nemenyi 后续检验	比较多个模型的成对差异	可以发现具体差异	需要在 Freedman 检验之后进行

表 2.1: 各种检验方法的比较

2.5 偏差与方差

偏差: 模型预测值与真实值之间的差异, 反映模型对数据的拟合程度. 方差: 模型预测值的波动性, 反映模型对数据的敏感程度. 泛化误差: 模型在新样本上的预测误差, 由偏差和方差共同决定.

$$E(f; D) = \text{Bias}^2 + \text{Var} + \text{Noise} \quad (2.25)$$

偏差-方差分解:

$$E(f; D) = \mathbb{E}[(f(x) - y)^2] = \mathbb{E}[(f(x) - \mathbb{E}[f(x)])^2] + \mathbb{E}[(\mathbb{E}[f(x)] - y)^2] + \text{Var}(f(x)) \quad (2.26)$$

偏差-方差权衡:

- 高偏差: 模型过于简单, 无法捕捉数据的复杂性, 导致欠拟合.
- 高方差: 模型过于复杂, 对训练数据的噪声敏感, 导致过拟合.
- 理想模型: 低偏差和低方差, 能够很好地泛化到新样本.

偏差-方差分解的图示:

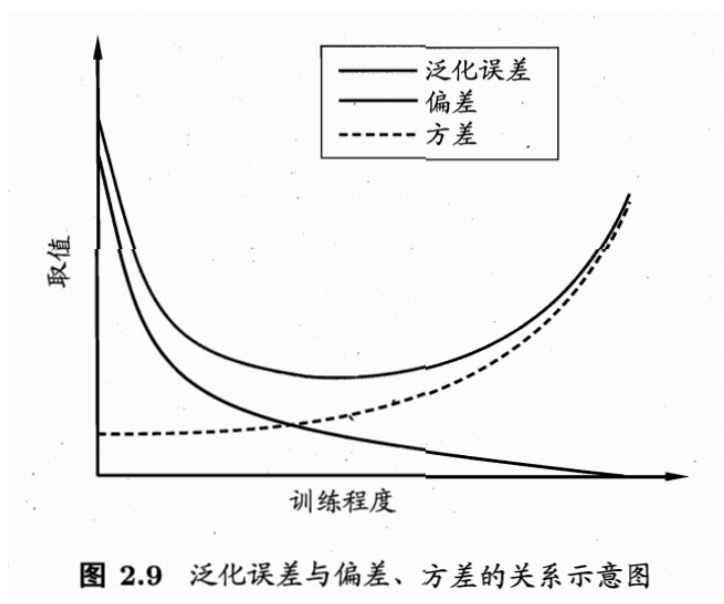


图 2.4: 偏差-方差分解示意图

2.6 阅读材料

略

2.7 习题

2.1

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.

答: 正例的训练集的组合数 $C(500, 350)$, 反例的训练集的组合数 $C(500, 350)$, 所以总的划分方式为 $C(500, 350) * C(500, 350)$.

2.2

数据集包含 100 个样本, 其中正、反例各一半, 假定学习算法所产生的模型是将新样本预测为训练样本数较多的类别 (训练样本数相同时进行随机猜测), 试给出用 10 折交叉验证法和留一法分别对错误率进行评估所得的结果.

答: 10 折交叉验证法将数据集分为 10 个子集, 每次使用 9 个子集作为训练集, 1 个子集作为测试集, 因此每次评估的错误率是基于 9/10 的样本进行的. 留一法则是每次留一个样本作为测试集, 其他 99 个样本作为训练集, 因此每次评估的错误率是基于 99/100 的样本进行的.

2.3

若学习器 A 的 F1 值比学习器 B 高, 试析 A 的 BEP 值是否也比 B 高.

答: 不一定. F1 值是查准率和查全率的调和平均数, 而 BEP 值是查准率和查全率相等时的点. 因此, 即使 A 的 F1 值高于 B, 也不能直接推断 A 的 BEP 值高于 B. 需要具体分析 A 和 B 的查准率和查全率的关系才能得出结论.

2.4

试述真正例率 (TPR)、假正例率 (FPR) 与查准率 (P)、查全率 (R) 之间的联系.

答: 真正例率 (TPR) 也称为查全率 (R), 表示实际正例中被正确预测为正例的比例; 假正例率 (FPR) 表示实际负例中被错误预测为正例的比例; 查准率 (P) 表示预测为正例的样本中实际为正例的比例. 它们之间的关系可以通过混淆矩阵来理解:

$$\begin{aligned} TPR &= \frac{TP}{TP + FN} = R \\ FPR &= \frac{FP}{FP + TN} \\ P &= \frac{TP}{TP + FP} \end{aligned}$$

2.5

试证明式

$$AUC = 1 - \ell_{\text{rank}}. \quad (2.27)$$

证明: AUC (Area Under Curve) 是 ROC 曲线下的面积, 而 ℓ_{rank} 是基于排名的损失函数. 通过构造合适的样本对, 可以证明 AUC 与 ℓ_{rank} 之间的关系, 从而得出上述等式.

2.6

试述错误率与 ROC 曲线的联系.

答: 错误率是分类模型在测试集上的预测错误比例, 而 ROC 曲线是通过绘制真正例率 (TPR) 与假正例率 (FPR) 的关系来评估分类模型性能的图形表示. ROC 曲线的形状和

位置可以反映模型的错误率,例如,如果 ROC 曲线接近左上角,则表示模型具有较低的错误率;如果 ROC 曲线接近对角线,则表示模型的错误率较高。

2.7

试证明任意一条 ROC 曲线都有一条代价曲线与之对应,反之亦然。

证明: 任意一条 ROC 曲线可以通过改变分类阈值来得到不同的 TPR 和 FPR 值,从而形成一条代价曲线。反之,给定一条代价曲线,也可以通过调整分类阈值来得到对应的 TPR 和 FPR 值,从而形成一条 ROC 曲线。因此,任意一条 ROC 曲线都有一条代价曲线与之对应,反之亦然。

2.8

Min-max 规范化和 z-score 规范化是两种常用的规范化方法。令 x 和 x' 分别表示变量在规范化前后的取值,相应的,令 x_{\min} 和 x_{\max} 表示规范化前的最小值和最大值, x'_{\min} 和 x'_{\max} 表示规范化后的最小值和最大值, \bar{x} 和 σ_x 分别表示规范化前的均值和标准差,则 min-max 规范化、z-score 规范化分别如式 (2.43) 和 (2.44) 所示。试析二者的优缺点。

$$x' = x'_{\min} + \frac{x - x_{\min}}{x_{\max} - x_{\min}} \times (x'_{\max} - x'_{\min}),$$

$$x' = \frac{x - \bar{x}}{\sigma_x}.$$

答: Min-max 规范化的优点是能够保持原始数据的分布形状,并且对异常值不敏感;缺点是对数据的最小值和最大值敏感,可能导致信息丢失。Z-score 规范化的优点是能够消除不同特征之间的量纲影响,并且对异常值敏感;缺点是可能导致数据分布形状改变。

2.9

试述 χ^2 检验过程。

答: χ^2 检验过程包括以下步骤:

1. 提出原假设和备择假设。原假设通常是样本数据符合某种分布或两个样本之间没有显著差异。
2. 计算 χ^2 统计量, 公式为:

$$\chi^2 = \sum_{i=1}^k \frac{(O_i - E_i)^2}{E_i}$$

其中, O_i 是观察频数, E_i 是期望频数, k 是类别数。

3. 根据自由度 (df) 查找 χ^2 分布表, 得到临界值或计算 p 值。
4. 如果 p 值小于显著性水平 (如 0.05), 则拒绝原假设, 认为样本数据存在显著差异; 否则, 接受原假设。

5. 结果解释：根据检验结果，分析样本数据的特征或类别之间的关系。

2.10

试述在 Friedman 检验中使用式的区别。

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

答: Friedman 检验是一种非参数统计方法, 用于比较多个算法在多次实验中的表现。在 Friedman 检验中, 两种常用的统计量分别是 τ_{χ^2} 和 τ_F , 它们分别用于评估算法的整体表现和排名差异。

τ_{χ^2} 是基于卡方分布的统计量, 用于衡量算法在不同实验中的表现差异。它可以通过以下公式计算:

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

或者等价地表示为:

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

其中, k 是算法的数量, N 是实验的次数, r_i 是第 i 个算法的平均排名。

τ_F 是基于 F 分布的统计量, 用于进一步评估排名差异的显著性。它可以通过以下公式计算:

$$\tau_F = \frac{(N-1)\tau_{\chi^2}}{N(k-1) - \tau_{\chi^2}}$$

τ_{χ^2} 主要用于初步评估算法的整体表现, 而 τ_F 则用于进一步分析排名差异的显著性。两者的主要区别在于, τ_{χ^2} 提供了一个整体的性能评估, 而 τ_F 则更关注于排名差异的具体程度。

第三章 线性模型

3.1 基本形式

线性模型是统计学中一种重要的模型形式，通常用于描述因变量与一个或多个自变量之间的线性关系。其向量形式可以表示为：

$$\mathbf{y} = \boldsymbol{\beta}^\top \mathbf{X} + \epsilon \quad (3.1)$$

其中， \mathbf{y} 是因变量的观测值向量， \mathbf{X} 是自变量的设计矩阵， $\boldsymbol{\beta}$ 是待估计的参数向量， ϵ 是误差项。

3.2 线性回归

最小二乘法是线性模型中最常用的参数估计方法，其目标是 minimized 观测值与模型预测值之间的平方差：

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \quad (3.2)$$

其中， $\hat{\boldsymbol{\beta}}$ 是参数的估计值。 $\arg \min$ 表示取使目标函数最小化的参数值。

当 $\mathbf{X}^\top \mathbf{X}$ 不是满秩时，最小二乘法的解可能不唯一，此时可以使用岭回归等方法进行正则化，以获得更稳定的参数估计。

广义线性模型（GLM）是线性模型的推广，允许因变量服从非正态分布，并通过链接函数将期望值与线性预测器联系起来。GLM 的形式为：

$$g(\mathbb{E}[Y|\mathbf{X}]) = \boldsymbol{\beta}^\top \mathbf{X} \quad (3.3)$$

其中， $g(\cdot)$ 是链接函数， $\mathbb{E}[Y|\mathbf{X}]$ 是因变量的条件期望。例如，当 $g(\cdot) = \log(\cdot)$ 时，GLM 为对数线性模型，适用于计数数据；当 $g(\cdot) = \text{logit}(\cdot)$ 时，GLM 为逻辑回归模型，适用于二分类数据。

```
1 % 线性回归的Matlab代码示例
2 clc,clear;
3 % 生成模拟数据
4 n = 100; % 样本数量
5 X = [ones(n, 1), randn(n, 2)]; % 自变量矩阵，第一列为常数项（截距）
```

```
6 beta_true = [1; 2; 3]; % 真实参数
7 y = X * beta_true + randn(n, 1) * 0.5; % 添加噪声生成因变量
8
9 % 拟合线性回归模型
10 mdl = fitlm(X, y);
11
12 % 输出回归结果
13 disp('参数估计:');
14 disp(mdl.Coefficients.Estimate);
15 disp(['R方: ', num2str(mdl.Rsquared.Ordinary)]);
16 disp(['调整后的R方: ', num2str(mdl.Rsquared.Adjusted)]);
17 disp(['均方误差: ', num2str(mean(mdl.Residuals.Raw.^2))]);
18
19 % 绘制回归结果及诊断图
20 figure;
21 plot(mdl);
22 title('线性回归拟合图');
23
24 % 输出结果
25 %{
26 参数估计:
27     0
28    1.0325
29    2.0137
30    3.0050
31 R方: 0.98584
32 调整后的R方: 0.98555
33 均方误差: 0.22526
34 %}
```

Listing 3.1: 线性回归的 Matlab 代码

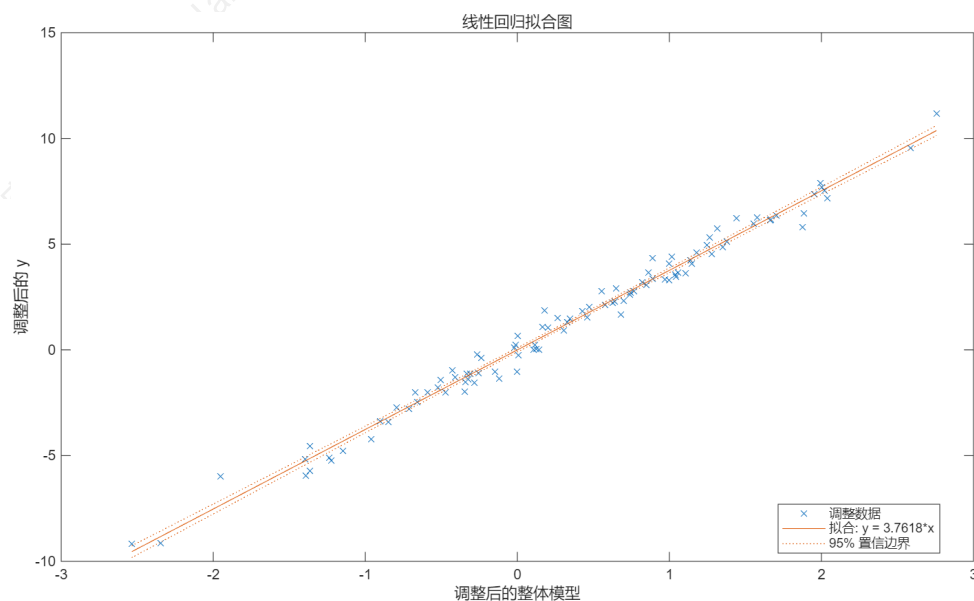


图 3.1: 线性回归拟合结果

3.3 对数几率回归

对数几率回归（Logistic Regression）是一种广义线性模型，主要用于二分类问题。其模型形式为：

$$P(Y = 1|\mathbf{X}) = g(\beta^T \mathbf{X}) = \frac{1}{1 + e^{-\beta^T \mathbf{X}}} \quad (3.4)$$

其中， $P(Y = 1|\mathbf{X})$ 是在给定自变量 \mathbf{X} 的条件下，因变量 Y 取值为 1 的概率， $g(\cdot)$ 是 sigmoid 函数。