

模型评估方法

1. 留出法 (hold-out)

方法：直接将数据集D划分为两个互斥的集合，训练集合S和测试集合T，在S上训练模型，用T来评估其测试误差

注意：训练 / 测试集的划分要尽可能保持数据分布的一致性，避免因为数据划分过程引入额外的偏差而对最终结果产生影响

缺点与改进：单次使用留出法得到的估计往往不够稳定可靠，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

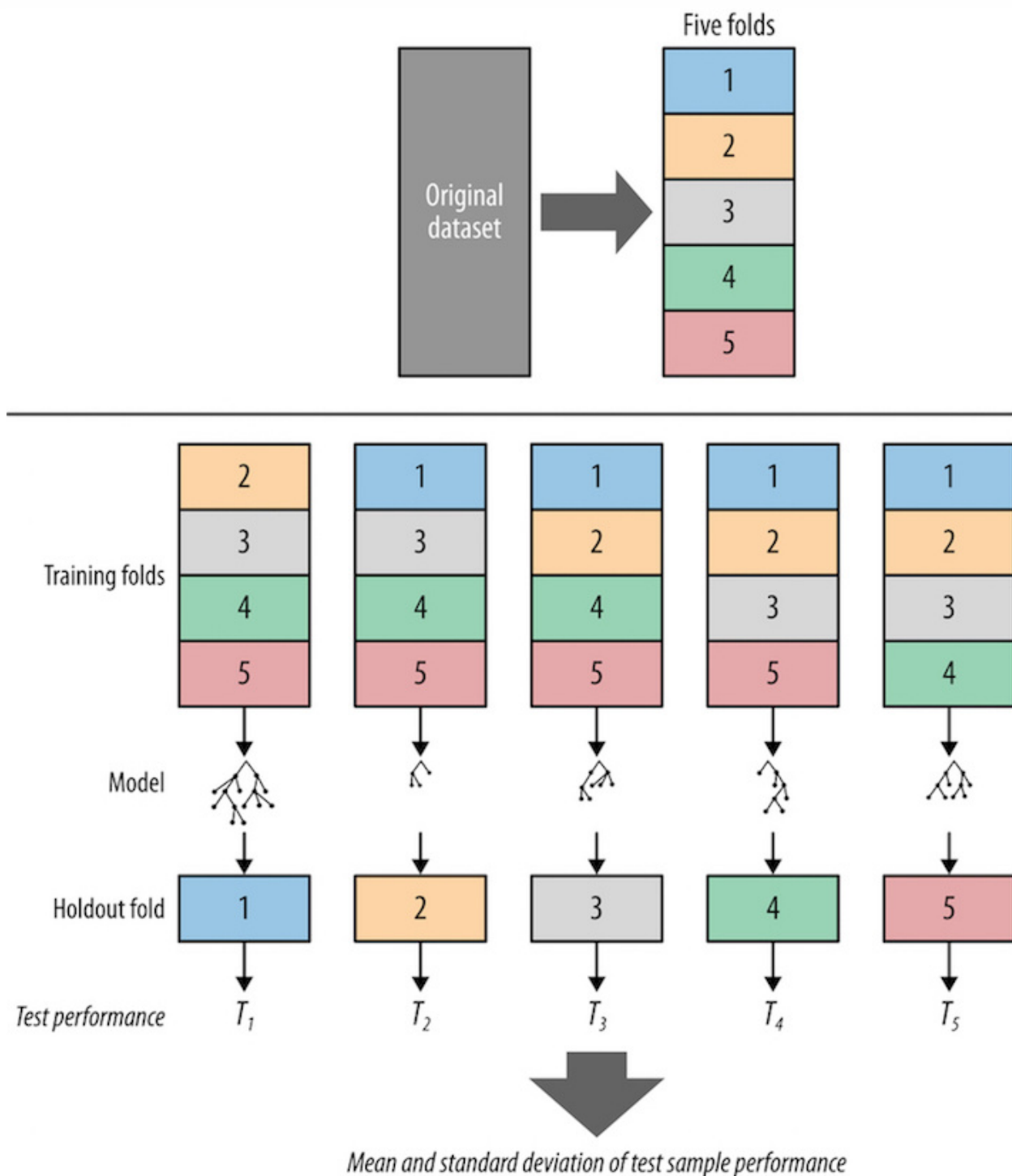
实际运用：实际中一般将大约2/3~4/5的样本用于训练，剩余样本用于测试

2. 交叉验证法 (cross validation)

方法：先将数据集D划分为k个大小相似的互斥子集.每个子集 D_i 都尽可能保持数据分布的一致性，即从D中通过分层采样得到.然后每次用k-1个子集的并集作为训练集，余下的那个子集作为测试集，这样就可以获得k组训练 / 测试集，从而可以进行k次训练和测试，最终返回的是这k个测试结果的均值

实际运用：一般而言k的取值为10，常用的还有5、20等

原理图：



3. 自助法 (bootstrapping)

问题引出：我们希望评估的是用D训练出来的模型，但是留出法和交叉验证法中，由于保留了一部分样本用于测试，因此实际评估的模型所使用的训练集比D小，这必然会引入一些因训练样本规模不同而导致的估计偏差，为此提出自助法。

方法：它以自助采样(bootstrap sampling)为基础.给定包含m个样本的数据集D，我们对它进行采样产生数据集 D' ：每次随机从D中挑选出一个样本，将其拷贝放入 D' ，然后再将该样本放回初始数据集D中，使得该样本在下次采样时仍有可能被采样到；这个过程重复执行m次后，我们就得到可包含m个样本数据的数据集 D' ，这就是自助采样的结果.样本在m次采样中始终不被采到到概率为

$$\lim_{m \rightarrow \infty} \left(1 - \frac{1}{m}\right)^m \rightarrow \frac{1}{e}$$

由此可知通过自助采样，初始数据集D中约有36.8%的样本未出现在采样数据集 D' 中.于是我们可将 D' 用作训练集， $D \setminus D'$ 用作测试集.

优缺点：自助法在数据集较小，难以有效划分训练 / 测试集时很有用，但是，自助法改变了初始数据集的分布，这会引入估计偏差，所以在数据量足够时，一般采用留出法和交叉验证法.

性能度量

1. 错误率 (error rate) 与精度 (accuracy)

错误率定义：

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

精度定义：

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

2. 准确率 (precision) 和召回率 (recall)

问题引出：在现实生活，比如信息检索中，我们往往会关心“检索出的信息中有多少比例是用户感兴趣的”，“用户感兴趣的信息中有多少被检索出来了”，准确率和召回率则比较适用于此类需求的性能度量

绘制如下混淆矩阵 (confusion matrix) 来进一步定义相关概念

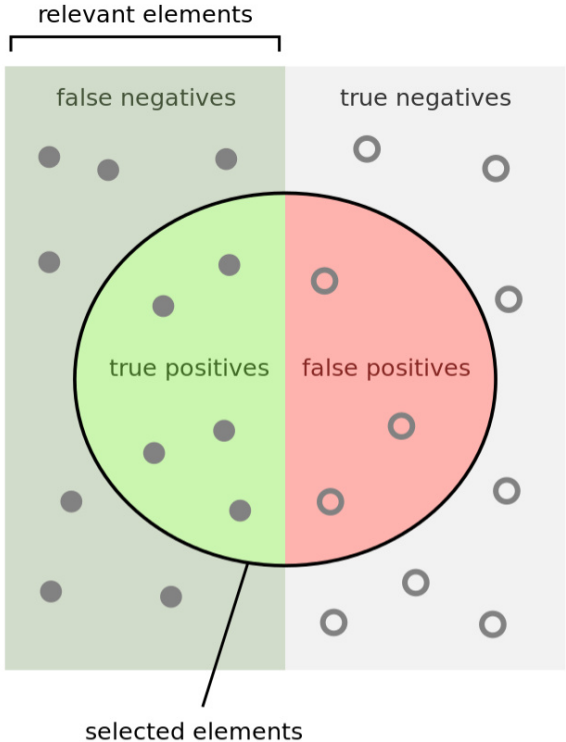


		Predicted condition			
		Total population	Predicted Condition positive	Predicted Condition negative	Prevalence $= \frac{\Sigma \text{Condition positive}}{\Sigma \text{Total population}}$
True condition	condition positive	True positive	False Negative (Type II error)	True positive rate (TPR), Sensitivity, Recall $= \frac{\Sigma \text{True positive}}{\Sigma \text{Condition positive}}$	False negative rate (FNR), Miss rate $= \frac{\Sigma \text{False negative}}{\Sigma \text{Condition positive}}$
	condition negative	False Positive (Type I error)	True negative	False positive rate (FPR), Fall-out $= \frac{\Sigma \text{False positive}}{\Sigma \text{Condition negative}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{True negative}}{\Sigma \text{Condition negative}}$
Accuracy (ACC) = $\frac{\Sigma \text{True positive} + \Sigma \text{True negative}}{\Sigma \text{Total population}}$		Positive predictive value (PPV), Precision $= \frac{\Sigma \text{True positive}}{\Sigma \text{Test outcome positive}}$	False omission rate (FOR) $= \frac{\Sigma \text{False negative}}{\Sigma \text{Test outcome negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR}^+}{\text{LR}^-}$
		False discovery rate (FDR) $= \frac{\Sigma \text{False positive}}{\Sigma \text{Test outcome positive}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{True negative}}{\Sigma \text{Test outcome negative}}$	Negative likelihood ratio (LR-) = $\frac{\text{FNR}}{\text{TNR}}$	

准确率定义：

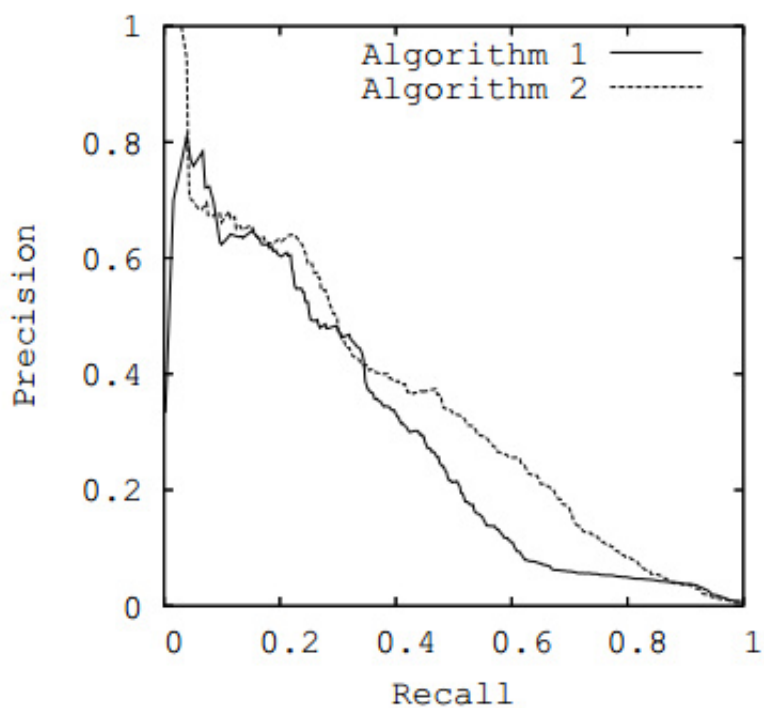
$$Precision = \frac{TP}{TP + FP}$$

召回率定义：

$$Recall = \frac{TP}{TP + FN}$$

集合表示	形象化解释
	<div><p>How many selected items are relevant?</p><p>Precision = </p></div> <div><p>How many relevant items are selected?</p><p>Recall = </p></div>

准确率与召回率之间的权衡可以使用PR曲线来衡量



将准确率和召回率相结合可以得到一些更实用的度量方式

F1度量定义：

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

F1是基于准确率和召回率的调和平均定义的

在一些应用中，对准确率和召回率的重视程度不同，例如在商品推销系统中，为了尽可能少打扰用户，更希望推荐内容是用户感兴趣的，此时准确率更重要.而在逃犯信息检索系统中，更希望尽可能少漏掉逃犯，此时召回率比较重要.

将F1一般化可得到 F_β 的定义：

$$F_\beta = \frac{(1 + \beta^2) \times Precision \times Recall}{(\beta^2 \times Precision) + Recall}$$

其中 $\beta = 1$ 时退化为标准的F1， $\beta > 1$ 时对准确率有更大影响， $\beta < 1$ 时对召回率有更大影响

3. ROC 与AUC

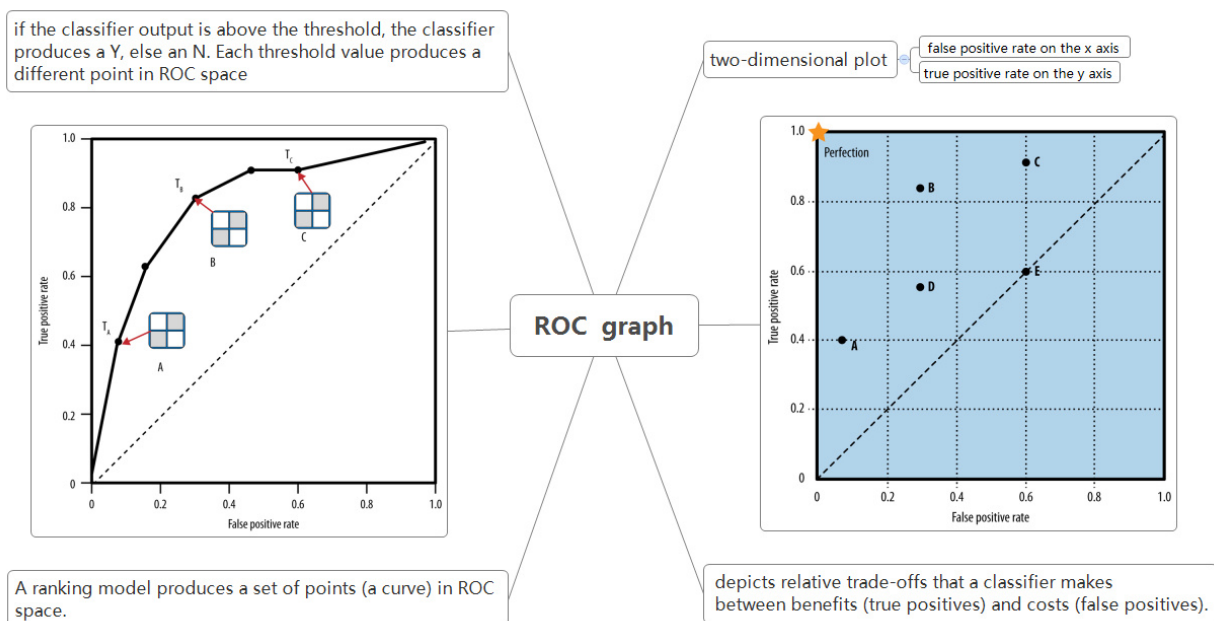
ROC Curve: 在统计中,"受试者工作特征" (receiver operating characteristic) ，是一幅用于描述在不同的阈值下二分类系统的性能表现的图。该图的横坐标为false positive rate (FPR) ，纵坐标为true positive rate (TPR) ,随着阈值的变化可得到一系列坐标点.

TPR的定义：

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

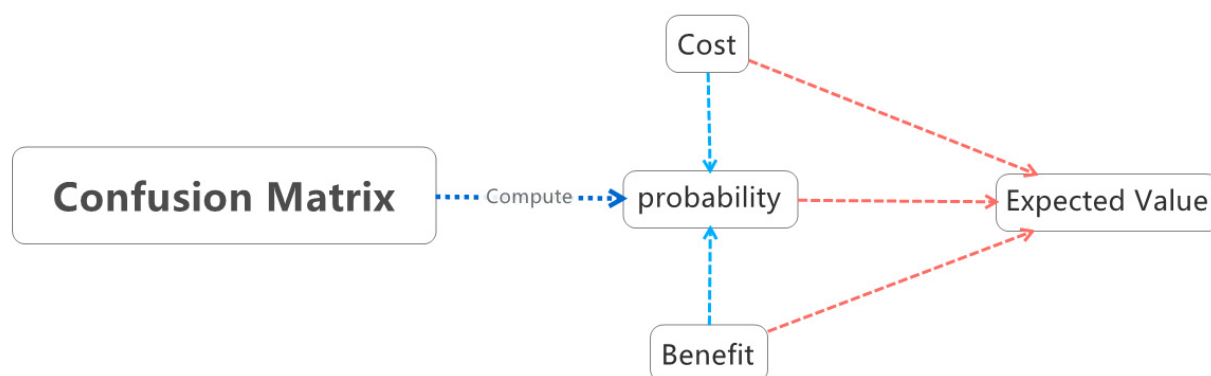
FPR的定义：

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



AUC: Area under the curve 可通过对ROC曲线下各部分的面积求和而得

4. 期望值分析框架



- Confusion Matrix

$$\begin{bmatrix} TruePositive & FalseNegative \\ FalsePositive & TrueNegative \end{bmatrix}$$

- Cost Matrix

$$\begin{bmatrix} Benefit_{TP} & Cost_{FN} \\ Cost_{FP} & Benefit_{TN} \end{bmatrix}$$

$$Expected_Profit = TruePositive \times Benefit_{TP} + FalseNegative \times Cost_{FN} + FalsePositive \times Cost_{FP} + TrueNegative \times Benefit_{TN}$$

5. 偏差与方差

$$Error = Bias + Variance + Noise$$

训练集中包含了一系列的点 $(x_1, y_1), (x_2, y_2), \dots, (x_i, y_i), \dots, (x_n, y_n)$, 我们假定存在一个函数

$y = f(x) + \epsilon$, 其中的 ϵ 的均值为0方差为 σ^2 . 我们需要找到一个函数 $\hat{f}(x)$ 使它尽可能地逼近函数 $f(x)$.

为此我们需要最小化 $(y - \hat{f}(x))^2$, 对这个误差的期望进行分解可得:

$$E[y - \hat{f}(x)]^2 = Bias[\hat{f}(x)]^2 + Var[\hat{f}(x)] + \sigma^2$$

其中:

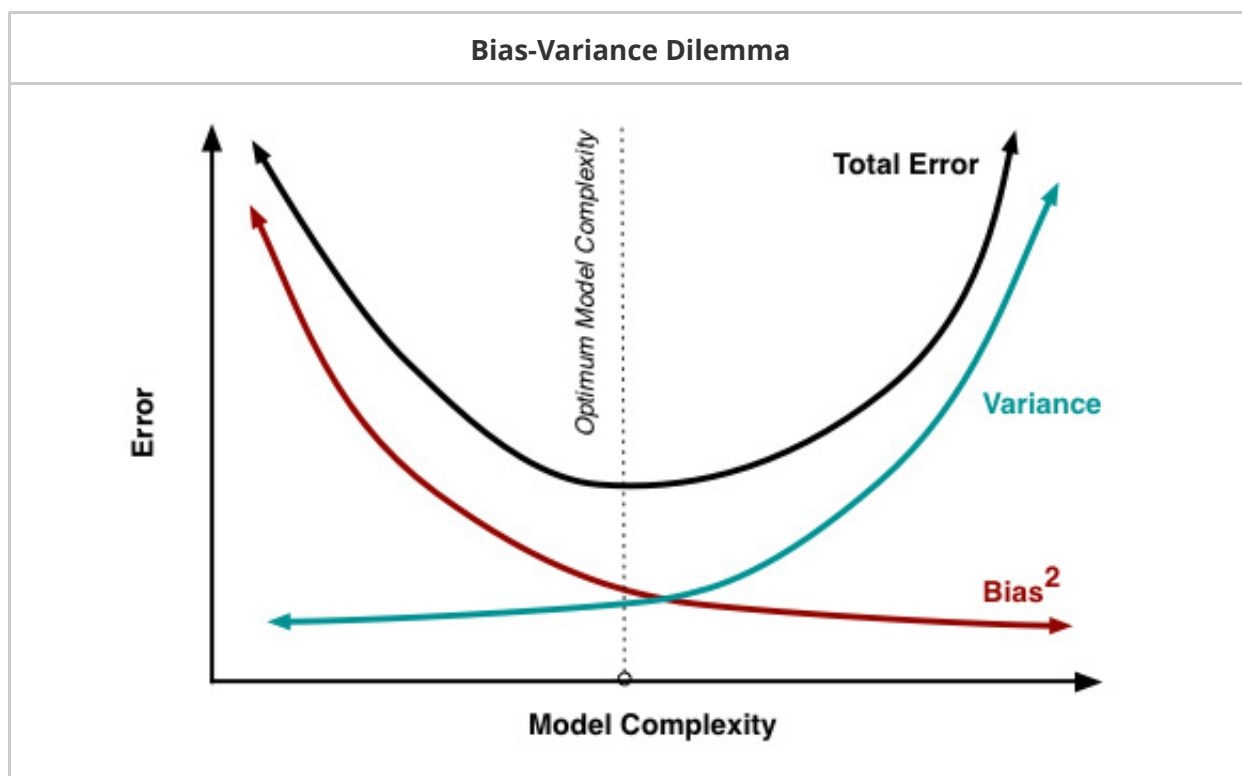
$$Bias[\hat{f}(x)] = E[\hat{f}(x) - f(x)]$$

$$Var[\hat{f}(x)] = E[\hat{f}(x) - E[\hat{f}(x)]]^2$$

偏差度量了学习算法的期望预测与实际结果的偏离程度, 即刻画了学习算法本身的拟合能力

方差度量了同样大小的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

噪声则表达了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



给定学习任务，在训练不足时，学习器的拟合能力不够强，训练数据的扰动不足以使学习器产生显著变化，此时偏差主导了泛化误差率；随着训练程度的加深，学习器的拟合能力逐渐增强，训练数据发生的扰动逐渐能被学习器学到，方差逐渐主导了泛化误差率；在训练程度足够后，学习器的拟合能力已非常强，训练数据发生的轻微扰动都会导致学习器发生显著变化，若训练数据自身的、非全局的特性被学习器学到了，则将发生过拟合。