周志华 著

MACHINE LEARNING

机器学习

清华大学出版社

崔磊

QQ：362626744
E-Mail: leicui@nwu.edu.cn
办公室: 信息学院院楼912

本章课件致谢：胡鹏

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
- 相关资料

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

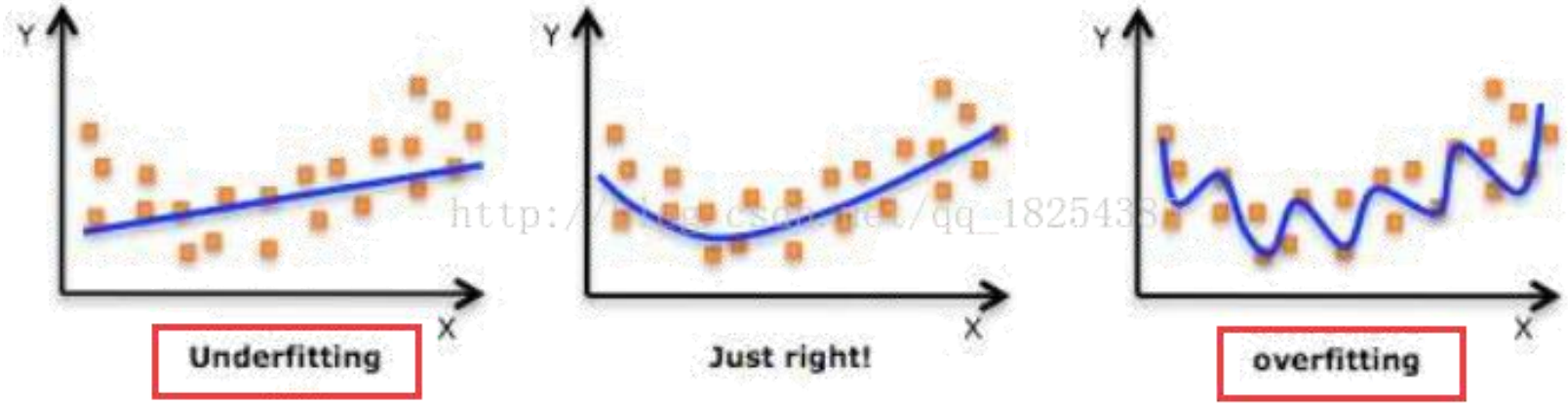
错误率&误差：

- 错误率：错分样本的占比：$E = a/m$
- 误差：样本真实标记与预测标记之间的差异
  - 训练（经验）误差：训练集
  - 测试误差：测试集
  - 泛化误差：除训练集以外所有新样本

期　　望：泛化误差小的学习器

现实情况：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



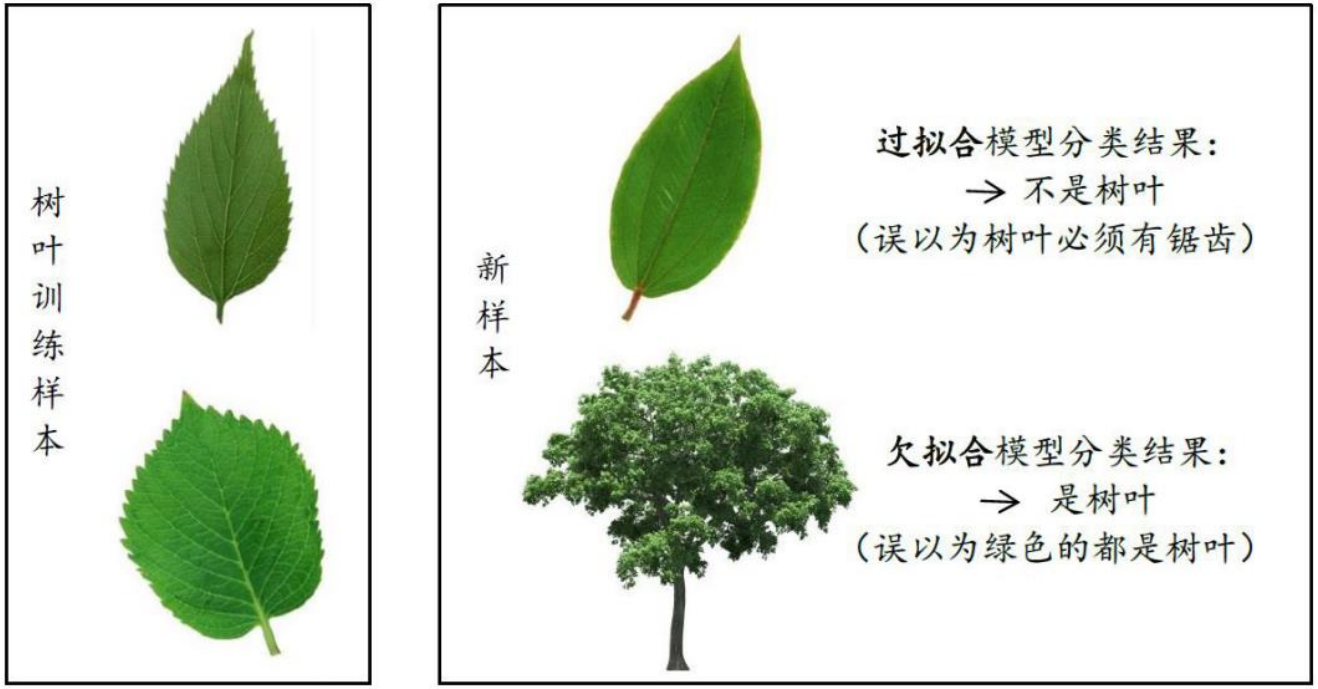
- **欠拟合：**

  对训练样本的一般性质尚未学好，训练样本被提取的特征比较少，导致训练出来的模型不能很好地匹配。

- **过拟合：**

  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，学到了很多没必要的特征**，导致泛化性能下降。

经验误差与过拟合



过拟合、欠拟合的直观类比

过拟合是无法彻底避免的，我们所做的只有‘缓解’，或者是减少其风险。

过拟合和欠拟合

- 过拟合:
  1. 特征选择
  2. 优化目标加正则项
  3. DNN常见方法：early stop/集成学习策略/Dropout策略
  4. 增加训练数据

- 欠拟合:
  1. 增加新特征，可以考虑加入进特征组合、高次特征，来增大假设空间;
  2. 尝试非线性模型，比如核SVM 、决策树、DNN等模型
  3. 增加网络的复杂度
  4. 减少使用正则化数量

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

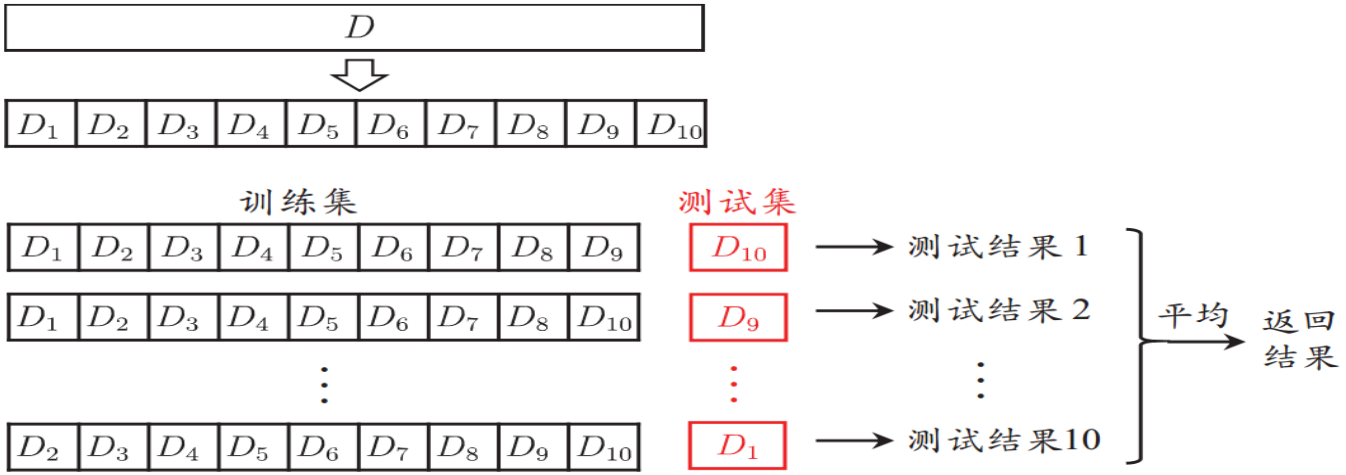
通常将包含m个样本的数据集$D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$拆分成训练集$S$和测试集 T

- 留出法：
  - 直接将数据集划分为两个互斥集合
  - 训练/测试集划分要尽可能保持数据分布的一致性
  - 一般若干次随机划分、重复实验取平均值
  - 训练/测试样本比例通常为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.



10 折交叉验证示意图

评估方法

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，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**”

评估方法

- 自助法：

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

样本在 m 次采样始终不被采样的概率是：$\left(1-\frac{1}{m}\right)^m$

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

评估方法

- 实际模型与预期模型都使用 m 个训练样本
- 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；
- 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。

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

在预测任务中，给定样例集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$

评估学习器的性能 f 也即把预测结果 $f(\boldsymbol{x})$ 和真实标记比较.

回归任务最常用的性能度量是“均方误差”：

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

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

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

精度

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

性能度量

查准率：挑出来的“好瓜”中有多少比例是真正的好瓜

查全率：所有好瓜中有多少比例挑了出来

（推荐系统、医疗辅助诊断等场景）

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

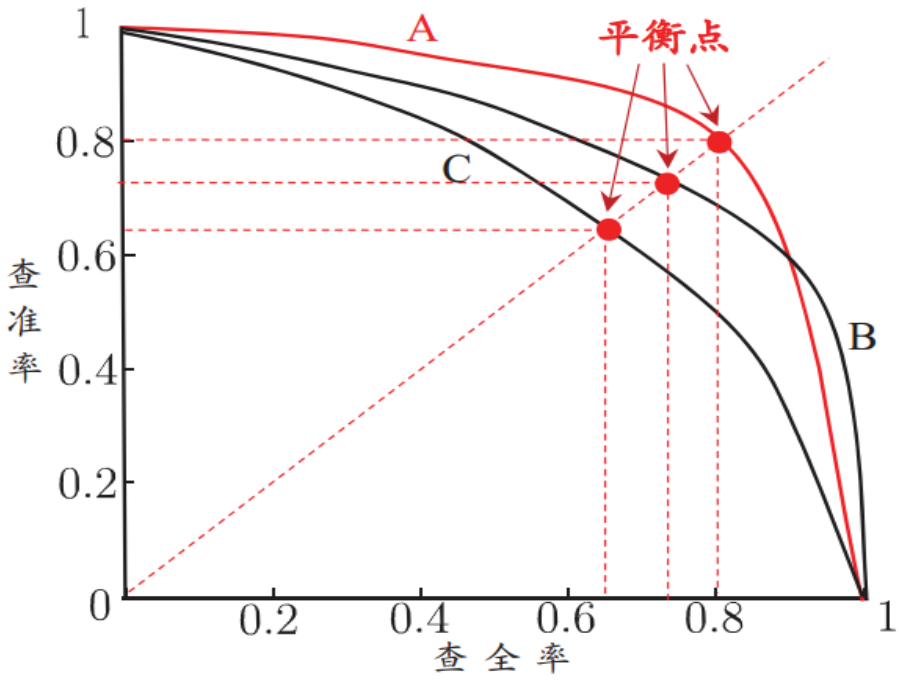
真实情况	预测结果	
	正例	反例
正例	TP（真正例）	FN（假反例）
反例	FP（假正例）	TN（真反例）

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

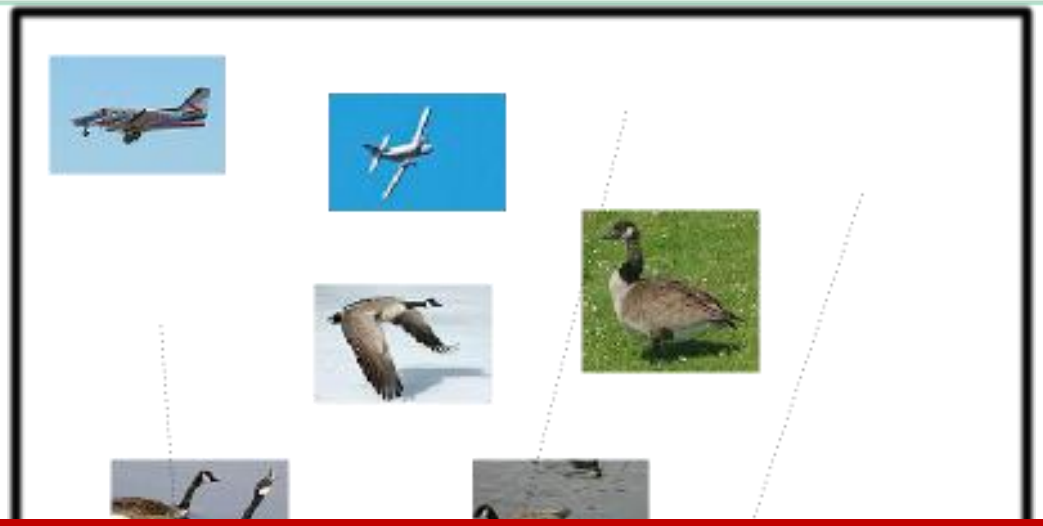
飞机和大雁的故事

假设下面有一个由飞机和大雁组成的图像数据集。

你现在想识别数据集中的所有的飞机。根据机器所识别的结果与图片的实际情况，我们可以得到以下四种识别结果与实际对比的情况：

True Positives（TP）：识别正确，系统认为是飞机，实际是飞机；
True Negatives（TN）：识别正确，系统认为不是飞机，实际不是飞机；
False Positives（FP）：识别错误，系统认为是飞机，实际不是飞机；
False Negatives（FN）：识别错误，系统认为不是飞机，实际是飞机。

性能度量



在上面的识别例子中，有3个TP、1个FP、4个TN、2个FN，也即

查准率= 3 / (3 + 1) = 0.75，

查全率= 3 / (3 + 2) = 0.6。

也就是说，识别的准确率为75%，60%的飞机被识别出来了。

性能度量

调整阈值

一次识别的结果可能说明不了什么，我们可以进行多次识别。识别出的数量可能是一个、两个或者其他个，正确识别的个数也有很多种情况。

根据每次阈值的不同，分割线也会在不同的位置，不同的分割线，对应不同的查全率和查准率。

性能度量

待测样本	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
标记样本	+	+	-	-	+	-	+	-	-	-
$P(+\|x_i)$	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.05

$1++--+-+---$ $P=0;R=0/4$

$+1+--+-+---$ $P=1;R=1/4$

$++1--+-+---$ $P=1;R=2/4$

$++-1-+-+---$ $P=2/3;R=2/4$

$++--1+-+---$ $P=2/4;R=2/4$

$++--+1-+---$ $P=3/5;R=3/4$

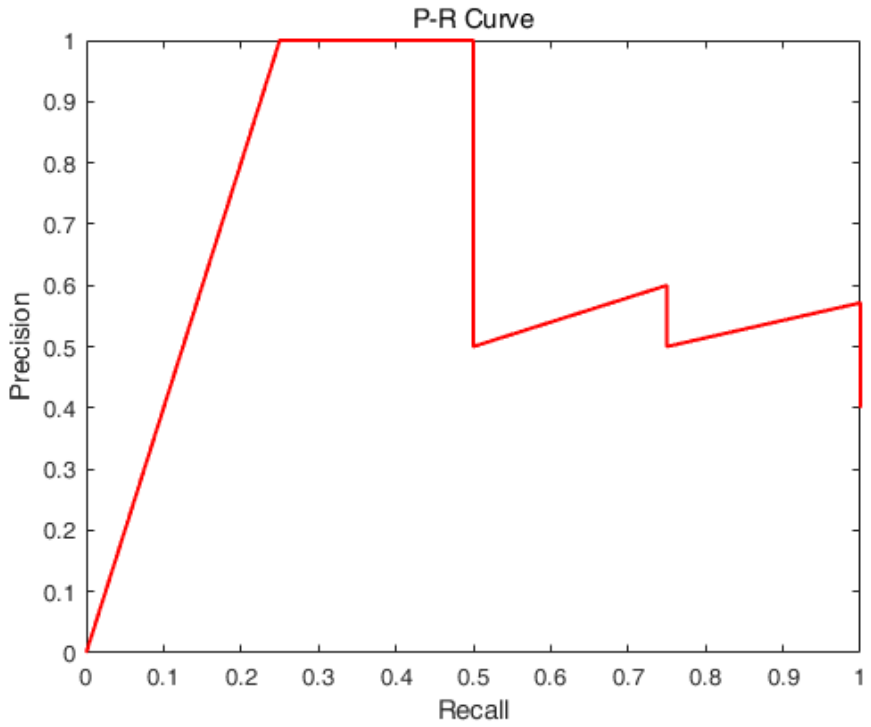
$++--+-1+---$ $P=3/6;R=3/4$

$++--+-+1---$ $P=4/7;R=4/4$

$++--+-+-1--$ $P=4/8;R=4/4$

$++--+-+--1-$ $P=4/9;R=4/4$

$++--+-+---1$ $P=4/10;R=4/4$



性能度量

比P-R曲线平衡点更用常用的是F1度量:

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

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

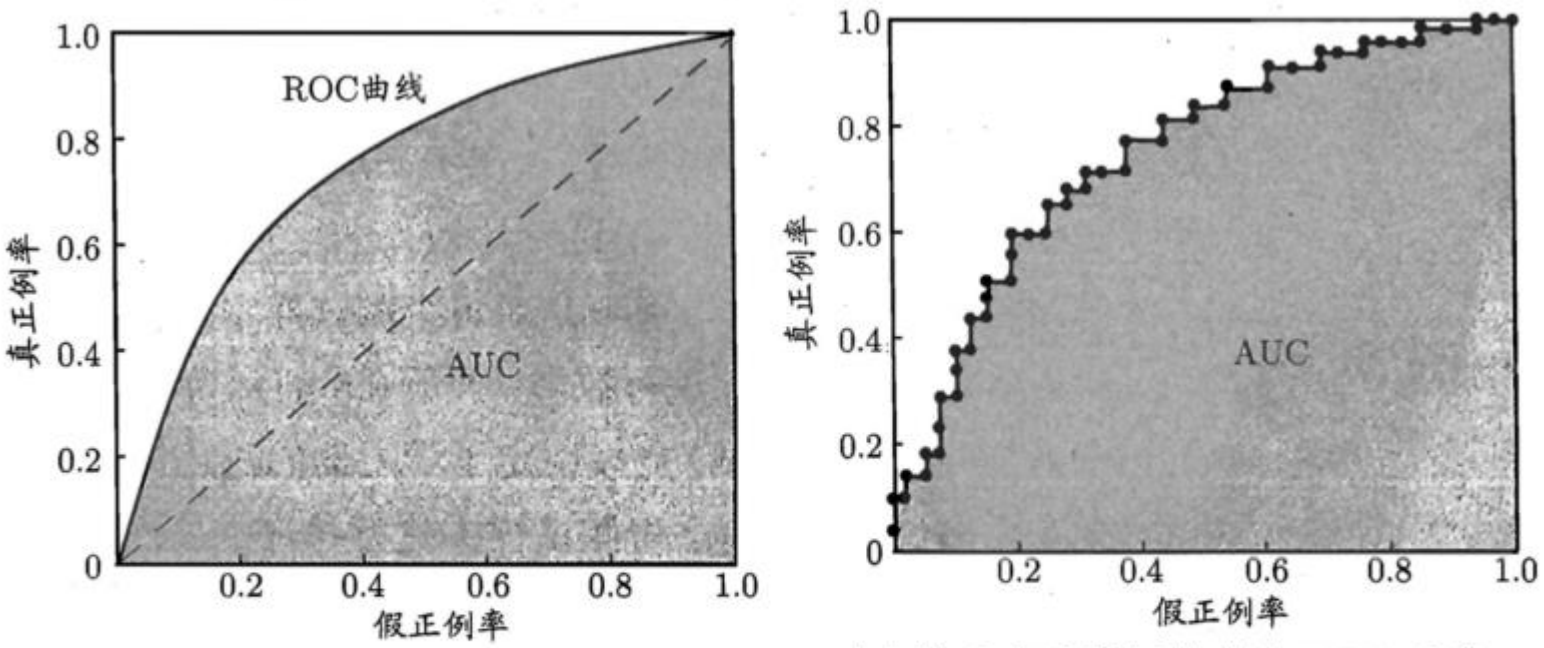
真正率（**TPR**）就是被分为正类的正样本比例：TPR=TP/(TP+FN)

假正率（**FPR**）就是被分为正类的负样本比例：FPR=FP/(FP+TN)

待测样例	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
样本标记	-	+	-	-	+	-	+	+	-
$P(+\|x_i)$	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1

| -+ - - + - + + -　　TPR=0/4; FPR=0/5
- |+ - - + - + + -　　TPR=0/4; FPR=1/5
- +| - - + - + + -　　TPR=1/4; FPR=1/5
- + -| - + - + + -　　TPR=1/4; FPR=2/5
……
- + - - + - + +| -　　TPR=4/4; FPR=4/5
- + - - + - + +- |　　TPR=4/4; FPR=5/5

性能度量



(a) ROC 曲线与 AUC　　(b) 基于有限样例绘制的 ROC 曲线与 AUC

图 2.4 ROC 曲线与 AUC 示意图

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.

性能度量

画ROC曲线方法：

1. 假设已经得出一系列样本被划分为正类的概率Score值，按照由大到小排序。

2. 从高到低，依次将“Score”值作为阈值threshold，当测试样本属于正样本的概率大于或等于这个threshold时，我们认为它为正样本，否则为负样本。举例来说，对于某个样本，其“Score”值为0.6，那么“Score”值大于等于0.6的样本都被认为是正样本，而其他样本则都认为是负样本。

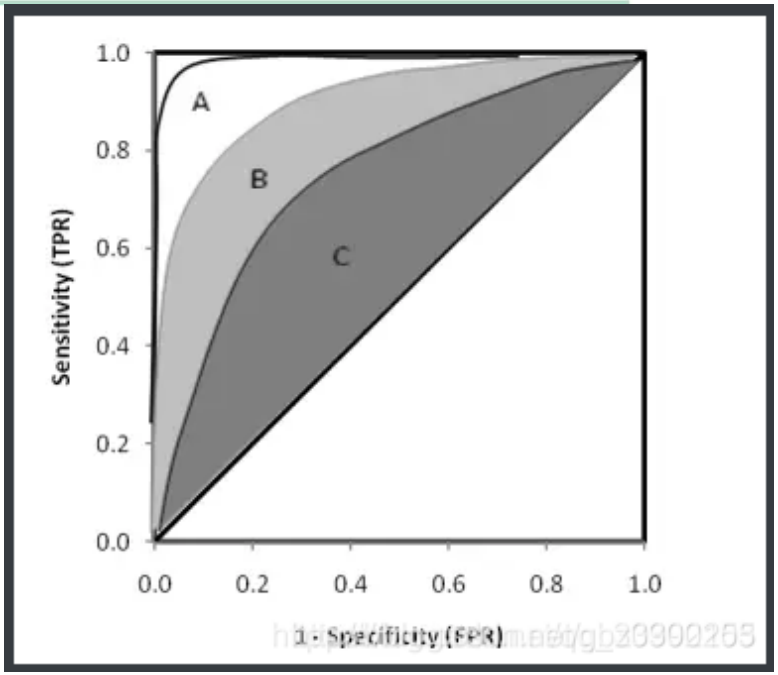
3.每次选取一个不同的threshold，得到一组FPR和TPR，以FPR值为横坐标和TPR值为纵坐标，即ROC曲线上的一点。

4.根据3中的每个坐标点，画图。

性能度量

有4个关键的点：

- 点(0,0)：FPR=TPR=0，分类器预测所有的样本都为负样本。
- (1,1)：FPR=TPR=1，分类器预测所有的样本都为正样本。
- 点(0,1)：FPR=0, TPR=1，此时FN＝0且FP＝0，所有的正样本都正确分类。
- 点(1,0)：FPR=1，TPR=0，此时TP＝0且TN＝0，最差分类器，所有正样本都没识对。



ROC曲线相对于PR曲线有个很好的特性：当测试集中的正负样本的分布变化的时候，ROC曲线能够保持不变，即对正负样本不均衡问题不敏感。

性能度量

AUC值计算：

AUC表示ROC曲线下的面积。ROC曲线本身并不能直观的说明一个分类器性能的好坏，而AUC值作为一个数量值，具有可比较性，可以进行定量的比较。

AUC值对模型性能的判断标准：

- AUC = 1，是完美分类器，采用这个预测模型时，存在至少一个阈值能得出完美预测。绝大多数预测的场合，不存在完美分类器。
- 0.5 < AUC < 1，优于随机猜测。这个分类器（模型）妥善设定阈值的话，能有预测价值。
- AUC = 0.5，跟随机猜测一样（例：丢铜板），模型没有预测价值。
- AUC < 0.5，比随机猜测还差；但只要总是反预测而行，就优于随机猜测

性能度量

假设ROC曲线由 $\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$

的点按序连接而形成 $(x_1 = 0, x_m = 1)$，则：

AUC可估算为：

方式1：计算ROC曲线下的面积

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

AUC衡量了样本预测的排序质量。

方式2：AUC统计意义上去计算

$$AUC = \frac{\sum_{ie\ positiveClass} rank_i - \frac{M(1+M)}{2}}{M \times N}$$

所有的正负样本对中，正样本排在负样本前面占样本对数的比例，即这个概率值。

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。

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

代价敏感错误率

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：

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

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

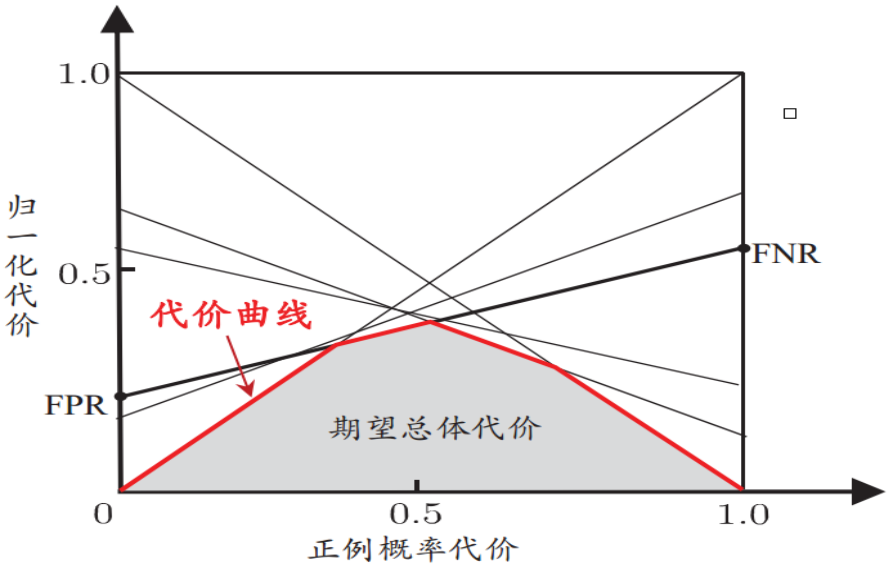
$$P(+)cost = \frac{p \times cost_{01}}{p \times cost_{01} + (1-p) \times cost_{10}}$$

纵轴是取值为[0,1]的归一化代价

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

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。



代价曲线与期望总体代价

总结

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，学到了很多没必要的特征，导致泛化性能下降。

欠拟合：训练样本的一般性质尚未学好。

评估方法
1. 留出法 2. 交叉验证法 3. 自助法

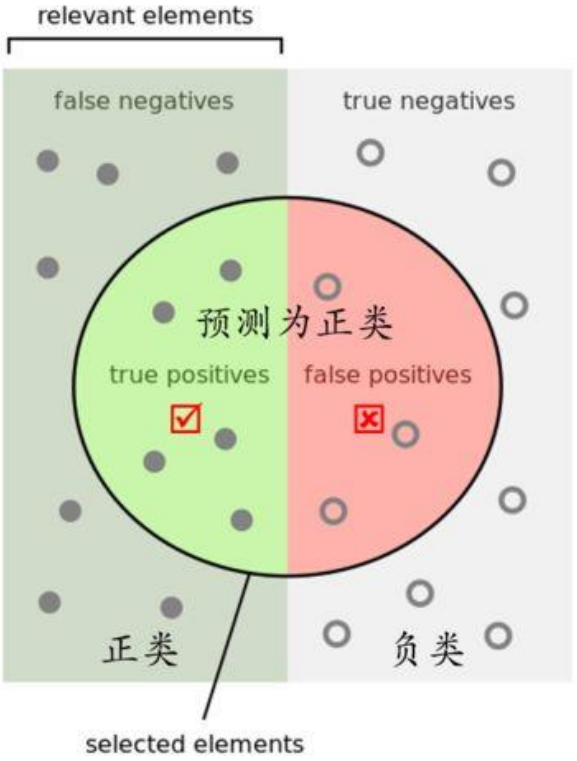
总结

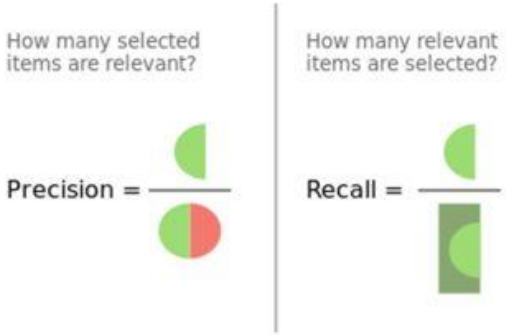
分类结果混淆矩阵

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)

正样本精确率为：
Precision=TP/(TP+FP)，表示的是 正样本识别正确总数 / 所有预测为正样本的样本总数

正样本召回率为：
Recall=TP/(TP+FN)，表示的是正样本识别正确总数 / 实际正样本总数

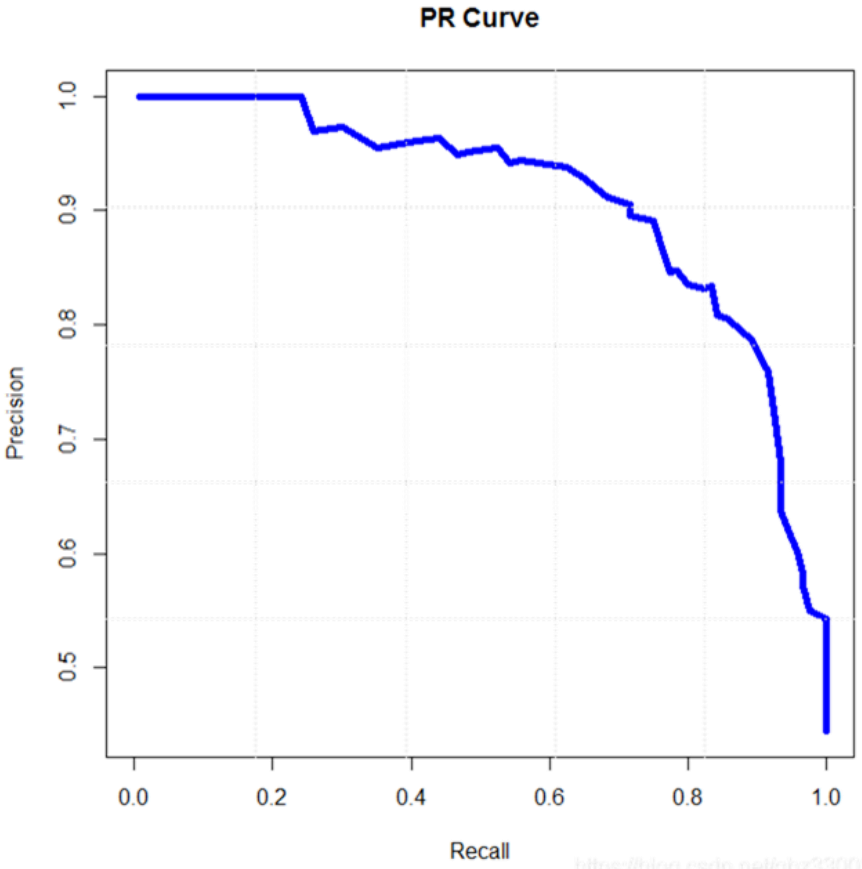




总结

画PR曲线方法：

通过置信度就可以对所有样本进行由高到低排序，再逐个样本的选择阈值，在该样本之前的都属于正例，该样本之后的都属于负例。每一个样本作为划分阈值时，都可以计算对应的precision和recall，那么就可以以此绘制曲线。



总结

画ROC曲线方法：

TPR=TP / (FN + TP） 正样本识别正确总数 / 正负样本识别正确总数
FPR=FP / (TN + FP） 负样本识别错误总数 / 正负样本识别错误总数

1. 假设已经得出一系列样本被划分为正类的概率Score值，按照由大到小排序。

2. 从高到低，依次将“Score”值作为阈值threshold，当测试样本属于正样本的概率大于或等于这个threshold时，我们认为它为正样本，否则为负样本。举例来说，对于某个样本，其“Score”值为0.6，那么“Score”值大于等于0.6的样本都被认为是正样本，而其他样本则都认为是负样本。

3.每次选取一个不同的threshold，得到一组FPR和TPR，以FPR值为横坐标和TPR值为纵坐标，即ROC曲线上的一点。

4.根据每个坐标点，画图。

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
- 相关资料

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华错误率进行拆解。

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，$f(\boldsymbol{x};D)$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：

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

使用样本数目相同的不同训练集产生的方差为

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

噪声为

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

偏差与方差

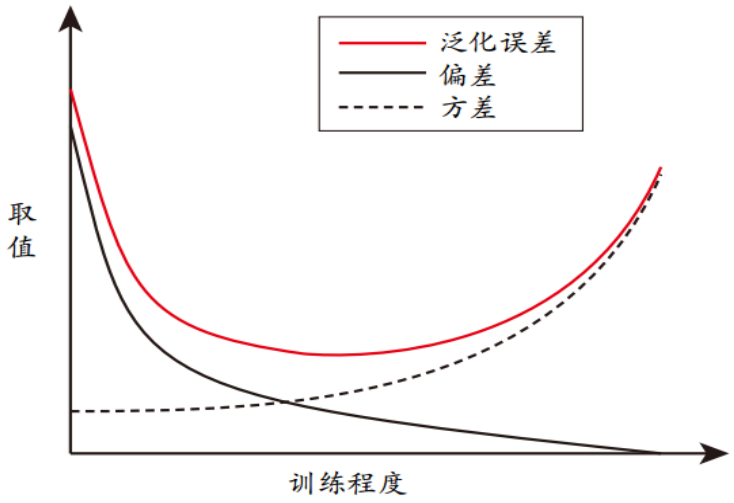
- **偏差**度量了学习算法期望预测与真实结果的偏离程度；即刻画了学习算法本身的拟合能力；
- **方差**度量了同样大小训练集的变动所导致的学习性能的变化；即刻画了数据扰动所造成的影响；
- **噪声**表达了在当前任务上任何学习算法所能达到的期望泛化误差的下界；即刻画了学习问题本身的难度。

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。



泛化误差与偏差、方差的关系示意图

总结

- Bias VS Variance

每种评估器都有是有利有弊。

$$Error = Bias + Variance$$

Error反映的是整个模型的准确度，Bias反映的是模型在样本上的输出与真实值之间的误差，即模型本身的精准度，Variance反映的是模型每一次输出结果与模型输出期望之间的误差，即模型的稳定性。