机器学习（西瓜书）

注　解

（第 2 章　模型评估与选择）

https://blog.csdn.net/jbb0523

前言

经常听人说南大周老师所著的《机器学习》（以下统称为西瓜书）是一本入门教材，是一本科普性质的教科书。在该书第十次印刷之际，周老师在“如何使用本书”中也提到“这是一本入门级教科书”。然而，本人读起来却感觉该书远不止“科普”“入门”那么简单，书中的很多公式需要思考良久方能推导，很多概念需要反复咀嚼才能消化。边读边想着是不是应该将自己学习时遇到的一些知识难点的解析分享出来，以帮助更多的人入门。自己的确也随手做过一些笔记，但由于怀疑这仅是自己的个别现象，毕竟读书期间，思考更多的是如何使用单片机、DSP、ARM、FGPA 等，而这些基本是不需要推导任何公式的，因此作罢。偶然间在周老师的新浪微博看到如下对话：

此时方知，可能“读不懂”并不是个别现象。因此决定写一本“西瓜书注解”或者称为“西瓜书读书笔记”，对自己研读西瓜书时遇到的“台阶”进行解释和推导，以帮助更多的人能够更快地进入到这个领域。另外，近期越来越强地意识到，扎扎实实地推导一些基础算法的公式，无论是对于理解算法本身机理还是进行学术研究，都是非常有必要的。

自己会根据个人学习进度和研究需要按章发布，不知道能不能坚持写完，加油！

毕竟自己也是一名初学者，所以可能一些概念解释并不完整、一些公式推导并不优美，甚至会存在错误，这是不可避免的，不接受漫骂，但欢迎将问题反馈给我，共同学习进步！

（网盘链接：https://pan.baidu.com/s/1QtEiNnk8jMzmbs0KPBN-_w）

第 2 章目录

第 2 章 模型评估与选择

如西瓜书前言所述，本章属于第 1 部分机器学习基础知识；如果说第 1 章介绍了什么是机器学习及机器学习的相关数学符号，那么本章则进一步介绍机器学习的相关概念，具体来说介绍内容正如本章名称“模型评估与选择”所述的相关概念，为后续章节介绍算法做好准备，因为研究机器学习最常做的事情就是比较自己的算法与别人算法的优劣。

2.1 经验误差与过拟合

梳理本节的几个概念：

错误率(error rate)：$E=\frac{a}{m}$，其中m为样本个数，a为分类错误样本个数

精度(accuracy)：精度=1−错误率

误差(error)：学习器的实际预测输出与样本的真实输出之间的差异

训练误差(training error)：学习器在训练集上的误差，又称为**经验误差(empirical error)**

泛化误差(generalization)：学习器在新样本上的误差

泛化误差和经验误差用于分类问题的定义式可参见第 267 页的式(12.1)和式(12.2)。

先补充一个概念叫“**代价**”**(cost)**，指将某种类别样本预测为另一种类别时付出的代价；进一步还有**代价函数(cost function)**（cost 也可以翻译为损失，但损失一般对应的单词是 loss，损失和代价概念基本一致，详见接下来的分析，另外第 131 页图 6.5 给出了四种损失函数）。

再补充一个概念叫“**风险**”**(risk)**，在第 7 章（第 147 页）边注中提到“决策论中将‘**期望损失**’称‘风险’(risk)”。

进一步地，学习器在训练集上的风险称为**经验风险**。在实际训练学习器时，常见的做法是**经验风险最小化(Empirical Risk Minimization, ERM)**，参见第 278 页的式(12.30)。

接下来辨析一下以上几个概念。

错误率和精度最容易理解，而且很明显是针对分类问题的，并且是数据集层面的概念。

误差的概念更适用于回归问题，当然分类问题也可以由回归问题求解，比如 3.3 节的对数几率回归，虽然名字里有“回归”，但实际解决的是二分类问题；但是，根据第 267 页的式(12.1)和式(12.2)的定义可以看出，在分类问题中也会使用误差的概念，此时的“差异”指的是学习器的实际预测输出的类别与样本真实的类别是否一致，若一致则“差异”为 0，若不一致则“差异”为 1，训练误差是在训练集上差异的平均值，而泛化误差则是在新样本上差异的平均值，即在数据集层面考虑“误差”的概念时，误差与错误率相同。

代价和损失在很多场合概念一致，学习器预测出错时就要付出代价或产生损失，但表 2.2 则只习惯称为代价矩阵(cost matrix)。对于分类问题，将某种类别样本预测为另一种类别时要付出代价；例如，对于前面谈到的分类问题中的误差（若一致则“差异”为 0，若不一致则“差异”为 1），若此时代价矩阵为预测正确则代价为 0、预测错误则代价为 1（即代价不敏感），你也可以说预测正确产生的损失为 0，预测错误产生的损失为 1，因此从数据集层面考虑时，这里的错误率、误差、代价、损失都是一个概念；代价函数或损失函数是以学习器参数（如线性二分类器$h(\boldsymbol{x})=\mathrm{sign}(\boldsymbol{w}^{\top}\boldsymbol{x}+b)$中的$\boldsymbol{w},b$）为自变量的函数，最小化代价函数就可以得到学习器的参数（即经验风险最小化）。

在刚刚讨论的错误率、误差、代价、损失的场景中，其实风险的概念也是一样的；针对某个数据集，错误率可以理解为分类错误样本个数的平均值，误差、代价、损失也都是平均值层面的概念，即“期望损失”，即风险（即风险与错误率一样都是数据集层面的概念）。

另外，以上讨论不区分“期望”、“均值”、“平均值”三个概念，具体参见博客《均值与期望：傻傻分不清？》（https://blog.csdn.net/wangyaninglm/article/details/80197579）。

过拟合(overfitting)是由于模型的学习能力相对于数据来说过于强大；反过来说，**欠拟合(underfitting)**是因为模型的学习能力相对于数据来说过于低下。暂且抛开“没有免费的午餐”定理不谈，例如对于第 8 页图 1.4 中的训练样本（黑点）来说，用类似于抛物线的曲线 A 去拟合则较为合理，而比较崎岖的曲线 B 相对于训练样本来说学习能力过于强大，但若仅用一条直线去训练则相对于训练样本来说直线的学习能力过于低下。

其实这里面的道理可以用第 12 章的计算学习理论去解释：不严谨地说，**泛化误差**的**上限**是**经验误差**加上**学习器学习能力**。当**学习器学习能力**很强大时，虽然**经验误差**会很小（甚至等于 0），但由于上限中的另一部分**学习器学习能力**很大，也会使**泛化误差**很大。

2.2 评估方法

本节介绍了三种模型评估方法：留出法、交叉验证法、自助法（还是直接称为 bootstrap 比较好）。三种方法在看论文时均可见到，只是对比不同算法的性能时多见留出法和交叉验证法，而集成学习（详见第 8 章的 8.2 节和 8.3 节）产生基分类器时多见自助法。

1、留出法的 Matlab 实现

在 Matlab 中可用如下几行代码实现留出法采样的过程（以训练集占 67%的样本为例）：

```
n_rand = randperm(m);%m is the number of training examples
idx_train = n_rand(1:round(m*0.67));
idx_test = setdiff(1:m, idx_train);
```

以上得到了训练集的索引 idx_train，只需按此索引从数据集中得到训练集即可，剩余样本索引存在 idx_test 中，即测试集；若要得到T次留出法划分，只需重复以上过程T次即可。

实际上，Matlab 自带有 randsample 函数也可以实现以上过程：

```
idx_train = randsample(1:m,round(m*0.67));
idx_test = setdiff(1:m,idx_train);
```

第 1 行表示从整数集合 1~m（第 1 个参数 1:m）中以均匀分布，用无放回采样方式（无第 3 个参数，即默认为'false'）随机选取 round(m*0.67)个样本（第 2 个参数 round(m*0.67)）。

至于本节中提到的“分层采样”问题，一般来说，只要对数据集样本顺序进行随机重排，一般划分后得到的训练集和测试集各类别样本比例基本一致。随机重排在第 1 种实现方式中由 n_rand = randperm(m);实现，第 2 种实现方式的 randsample 函数自带此功能。

2、交叉验证法的 Matlab 实现

以下是本人参考 WEKA 写的一个基于 Matlab 的交叉验证划分函数：

```
function [ index_train,index_test ] = CV_data_partition( numInstances,numFolds,numFold )
% Author: https://blog.csdn.net/jbb0523
% Version: 1.0@2018-05-04 14:20
% Description: data set partition for n-fold Cross Validation
% Reference: weka.core.Instances (trainCV and testCV)
% Inputs:
%           numInstances: number of instances in dataset
%           numFolds      : n-folds
%           numFold       : n-th fold
```

```
% Outputs:
%            index_train : training set index (n-1 folds)
%            index_test    : testing set index (1 fold)
    if numFolds < 2
        error('Number of folds must be at least 2!');
    end
    if numFolds > numInstances
        error('Can not have more folds than instances!');
    end
    index_all = 1:numInstances;
    numInstForFold = floor(numInstances/numFolds);
    if numFold < mod(numInstances,numFolds)+1
        numInstForFold = numInstForFold + 1;
        offset = numFold;
    else
        offset = mod(numInstances,numFolds)+1;
    end
    first = (numFold-1) * floor(numInstances/numFolds) + offset;
    last = first+numInstForFold-1;
    index_test = index_all(first:last);
    index_train = setdiff(index_all,index_test);
%% The end!
end
```

有了以上数据集划分函数，在使用时可以按如下方式调用：

```
rand('state', 1); %fix random seed
idx_rand = randperm(numInstances);
for numFold=1:numFolds
    %split dataset into training set & testing set
    [ idx_train,idx_test ] = CV_data_partition( numInstances,numFolds,numFold );
    X_train = X_load(idx_rand(idx_train),:);
    y_train = y_load(idx_rand(idx_train),:);
    X_test = X_load(idx_rand(idx_test),:);
    y_test = y_load(idx_rand(idx_test),:);

    %your code here...

end
```

其中 numInstances 为样本个数，numFolds 为交叉验证的折数，最常见的是十折交叉验证。X_load 为数据集示例集合，y_load 表示数据集标记集合，均为每行表示一个样本；特别地，y_load 一般来说是一个向量，但对于某些特殊的问题来说，例如 3.7 节（第 68 页）最后一段提到的多标记学习，每个示例将对应多个标记，此时 y_load 将是一个矩阵。

在 Matlab 中，built-in 函数 crossvalind 也可实现上述功能，但这个函数属于 Bioinformatics Toolbox，可能并不是所有人都会安装此工具箱；在更常用的 Statistics Toolbox 中包含一个专门为交叉验证设计的、名为 cvpartition 的函数，也可琢磨一下其使用方法。

特别要注意的是，在对比不同算法的性能时，不同算法要使用相同的数据集划分，在留出法中也应该保证这一点。在上述调用中，通过 rand('state', 1);控制随机种子来实现；这里使用 randperm 进行随机排序，是为了打乱原先样本存储顺序，即常听到的 shuffle 操作。

另外，若个人的计算机配置足够，可将 for 改为 parfor，以实现交叉验证的并行化。

3、自助法的 Matlab 实现

在 Matlab 中可用如下几行代码实现 bootstrap sampling 的过程：

```
idx_bag = zeros(m,1); %m is the number of training examples
for ii=1:m
    idx_bag(ii)=randi(m);%bootstrap sampling
end
idx_oob = setdiff(1:m,idx_bag);
```

以上得到了训练样本的索引 idx_bag，只需按此索引从数据集中得到与数据集样本数量相同的样本集合即可，剩余样本索引存在 idx_oob 中，可以发现约有 0.368*m 个样本；

实际上，Matlab 自带有 randsample 函数也可以实现以上过程：

```
w = ones(1,m);%weights
idx_bag = randsample(1:m,m,'true',w);
idx_oob = setdiff(1:m,idx_bag);
```

第 2 行表示从整数集合 1~m（第 1 个参数 1:m）中以对应权重 w（第 4 个参数 w），用有放回采样方式（第 3 个参数'true'）随机选取 m 个样本（第 2 个参数 m）；此处权重 w 不用规范化，Matlab 在使用时自己会除以 sum(w)从而将权重变为一个概率分布。此处 w 元素均为 1，即标准的 bootstrap 采样，但是 randsample 可以实现针对每个样本以不同的权重进行采样，这一点使用起来很方便。

自助法在第八章 8.3 节还会用到，如这里 bag 和 oob (i.e., out of bag)名称就来自第 8 章。

4、算法参数（超参数）与模型参数

本条分析主要针对书中第 28 页的边注。

首先，区分一下“算法”和“模型”两个概念。网上也有一些辨析，但并不容易理解。个人感觉，在机器学习背景下，在不引起混淆的前提下，认为二者等价即可。当然二者的确是不等价的，个人感觉在西瓜书第 28 页边注中提到的“算法”和“模型”，应该是“算法”的概念大于“模型”的概念；具体来说 k 近邻（参见第 10.1 节）、支持向量机（参见第 6 章）都是一种算法，每个算法按某个参数配置训练之后都会得到一个模型。

其次，算法本身的一些参数，例如 k 近邻的近邻个数 k、支持向量机的参数 C（详见第六章第 130 页式(6.29)）就是这里提到的算法参数，亦称超参数。

最后，算法训练后会得到一个模型，例如支持向量机最终会得到$\boldsymbol{w}$和b具体的值（此处不考虑核函数），这就是模型参数；用模型参数就可以对未见样本做预测了。

这里比较让人困惑的是最后括号中的“例如神经网络在不同轮数停止训练”这句话的含义。个人认为，神经网络（参见第五章）的训练轮数与其隐含层数量及隐含层包含的神经元个数一样，都是算法超参数（当然，训练轮数与其隐含层数量及隐含层包含的神经元个数确实有些不同）。作者在这里的意思到底是训练轮数本身是模型参数，还是不同训练轮数得到的多个神经网络模型（连接权和域值等）是模型参数？

关于模型参数和超参数的区别还可以参见《你知道机器学习中的模型参数和超参数之间的区别吗？》（https://mp.weixin.qq.com/s/Nwd0Dm2_D1eY3n4z_Fw1FA），在知乎中搜索“模型参数和超参数”也可以得到很多结果。

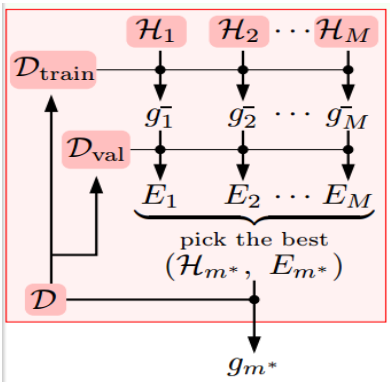
5、训练集/测试集/验证集

本节涉及到训练集(training set)、测试集(testing set)和验证集(validation set)三个概念。

训练集和测试集很容易理解，本节介绍的留出法、交叉验证法、自助法均可以将数据集划分为两部分，其中一部分用于训练，即训练集，剩余部分用于测试，即测试集。

然而，有时算法需要从候选超参数中选择使用其一，例如支持向量机的参数 C（详见第六章第 130 页式(6.29)）；这时需要先将训练集继续划分为两部分，其中一部分（暂称为小训练集）用于训练使用了不同候选超参数的模型，然后将剩余部分（即验证集）用于测试，进而基于验证集测试结果选出较好的超参数；然后，再以此超参数为准，在整个训练集上重新训练，再用测试集进行测试其性能。

可以发现，验证集在选择超参数过程中的角色与算法对比时测试集的角色相同，因为选择超参数过程实际也是在对比多个模型的性能。因此，在工程实际中（可以理解为做比赛），若客户给你提供了数据集（比赛主办方公开的数据集），在研发阶段你可以将数据集划分为训练集和测试集来选择不同的算法，也可以进一步地将训练集继续划分为小训练集和验证集对算法的超参数进行选择，但最后提交给客户的模型（即提交到比赛网站的模型）一定是基于整个数据集重新训练的模型。以下是林轩田《机器学习基石》课程第 15 讲 PPT 中的截图：



研究与工程实际是不一样的。研究是要将自己的算法与其它算法在现有的数据集上作对比，因此肯定是将数据集划分为训练集和测试集，若需要确定算法超参数则需要进一步将训练集划分为小训练集和验证集；而工程实际是在以上基础之上再提交一个学习器给客户，所以最后要基于整个数据集重新训练一遍后，将得到的学习器交给客户。

特别注意，测试集在整个过程中只能用于测试，不能参与到参数的选择过程中（这里指的是算法研究，若对应到工程实际，真实的测试集标记信息本身是不可见的，例如做比赛时最终使用的测试数据）；但在有的论文中，会在最后基于测试集做参数敏感度分析（与其它算法对比时固定某个超参数）；例如，对于支持向量机，在与其它算法对比时使用 C=1 作为默认参数，在实验分析时将 C={0.01, 0.1,1,10,10}几个参数都训练一遍并得到测试结果，来说明算法性能是否会随着 C 的变化而变化，若基本不变那当然可喜可贺，我们希望算法对超参数是不敏感的，这样工程实际中就可以任意取某个参数即可；一般算法给出的默认值是参数敏感度分析中结果最好的那个，但其实这有点使用测试集调参的味道。

2.3 性能度量

其实对于本节内容，重点了解错误率和精度一般情况下就足够了。

1、式(2.2)到式(2.7)的解释（数据分布）

这几个式子本身通俗易懂，几乎不需要什么注解。需要说明的是式(2.2)、式(2.4)和式(2.5)

假设了数据分布$\mathcal{D}$为均匀分布，即每个样本出现的概率相同，而式(2.3)、式(2.6)和式(2.7)则为更一般的表达式。第八章 8.2 节的 AdaBoost 算法主要就是靠改变数据分布增加基分类器的多样性，具体参见相关内容。

数据分布$\mathcal{D}$可理解为从样本空间中以独立同分布（independent and identically distribution, *i.i.d.*）方式采样时样本被采到的概率。例如，样本空间包含 5 个样本$\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_5$（当然，实际的样本空间不可能这么小，很多时候样本空间是无穷大的），数据分布$\mathcal{D} = [0.1, 0.2, 0.3, 0.2, 0.2]$，若以独立同分布方式从样本空间中取 10 个样本（可理解为有放回式抽样），则理论上采样集$D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_3, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_4, \boldsymbol{x}_5, \boldsymbol{x}_5\}$。

2、查准率和查全率(召回率)

这里仅将书中几处介绍查准率(Precision)和查全率(Recall)意义的描述摘录出来。

(1)瓜农拉来一车西瓜，我们用训练好的模型对这些西瓜进行判别，查准率衡量的是“挑出的西瓜中有多少比例是好瓜”，查全率衡量的是“所有好瓜中有多少比例被挑出来了”。

(2)信息检索中，查准率衡量的是“检索出的信息中有多少比例是用户感兴趣的”，查全率衡量的是“用户感兴趣的信息中有多少被检索出来了”。

商品推荐系统中，为了尽可能少的打扰用户，更希望推荐内容确是用户感兴趣的，此时查准率更重要；在逃犯信息检索系统中，更希望尽可能少漏掉逃犯，此时查全率更加重要。

另外，F1 是查准率和查全率的合成指标，可以理解为综合考虑了查准率和查全率的性能度量指标，这个指标更加面向类别不平衡问题（参见第 66~67 页第 3.6 节），例如某数据集 99%为正例，剩余 1%为反例，此时分类器只须一直预测正例即可获得 1%的错误率，然后这显然并没有什么意义。

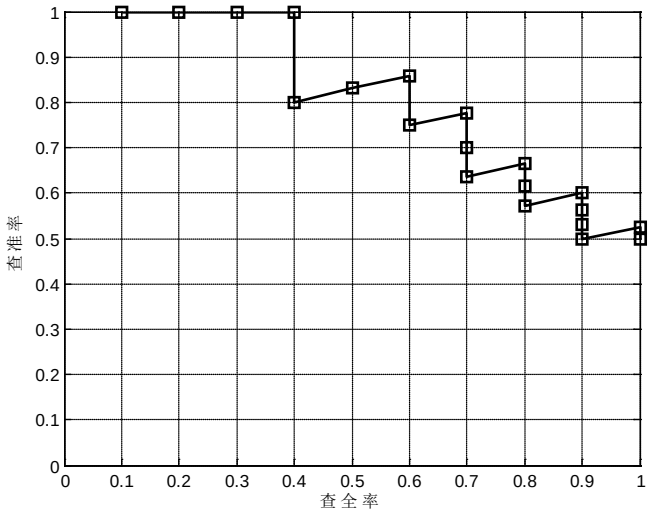
3、图 2.3 的 Matlab 绘制

设有 20 个样本，如下（+表示正例，−表示反例）：

＋＋＋＋－＋＋－＋－－＋－－＋－－－＋－

上面根据预测结果对样例进行了排序，排在最左侧的是学习器认为“最可能”是正例的样本，排在最右侧的则是学习器认为“最不可能”是正例的样本。现从左往右逐个把样本作为正例进行预测，则依次计算出查全率和查准率，分别存于 x 和 y 当中，画出 P-R 曲线：

```
x=[0.1 0.2 0.3 0.4 0.4 0.5 0.6 0.6 0.7 0.7 0.7 0.8 0.8 0.8 0.9 0.9 0.9 0.9 1 1];
y=[1/1 2/2 3/3 4/4 4/5 5/6 6/7 6/8 7/9 7/10 7/11 8/12 8/13 8/14 9/15 9/16 9/17 9/18 10/19 10/20];
plot(x,y,'ks-','linewidth',2);axis([0 1 0 1]);xlabel('查全率');ylabel('查准率');grid on;
```



从式(2.8)可知，查准率的极限为正例所占比例，因为当把所有样本预测为正例时，TP+FP为样本总数，TP 为正例样本数，**因此 P-R 曲线不会像图 2.3 那样，当到最右侧（查全率等于 1 时）查准率趋于零，除非正例样本在整个样本集中所占比例相当小，接近于零**。

从式(2.9)可知，当把所有样本预测为正例时，查全率 R 等于 1，因为此时 FN=0。

4、式(2.10)的解释

将式(2.8)和式(2.9)代入式(2.10)，得

$$\begin{aligned} F1 =& \frac{2 \times P \times R}{P + R} = \frac{2 \times \frac{TP}{TP+FP} \times \frac{TP}{TP+FN}}{\frac{TP}{TP+FP} + \frac{TP}{TP+FN}} \\ =& \frac{2 \times TP \times TP}{TP(TP + FN) + TP(TP + FP)} \\ =& \frac{2 \times TP}{(TP + FN) + (TP + FP)} \end{aligned}$$

已知数据集$D = \{(\boldsymbol{x}_i, y_i) \mid 1 \leq i \leq m\}$，其中标记$y_i \in \{0, 1\}$（1 表示正例，0 表示反例）；设$\boldsymbol{x}_i$的二值化预测结果为$h_i \in \{0, 1\}$，则

$$F1 = \frac{2 \sum_{i=1}^{m} y_i h_i}{\sum_{i=1}^{m} h_i + \sum_{i=1}^{m} y_i}$$

5、式(2.12)到式(2.17)的解释

式(2.12)的macro-P和式(2.13)的macro-R是基于各个二分类问题的P和R计算而得的；式(2.15)的micro-P和式(2.16)的micro-R是基于各个二分类问题的TP、FP、TN、FN计算而得的；“宏”可以认为是只关注宏观而不看具体细节，而“微”可以认为是要从具体细节做起（对于各个二分类问题而言，相比于其P和R指标来说，TP、FP、TN、FN更微观，因为P和R是基于TP、FP、TN、FN计算而得）。

式(2.14)的macro-$F1$是为了使macro-$F1$、macro-P和macro-R三者之间满足类似于式(2.10)的关系；式(2.17)的micro-$F1$亦是为了使micro-$F1$、micro-P和micro-R三者之间满足类似于式(2.10)的关系。

这里想说的是，这仅是一种定义而已，在作者的论文[X.-Z. Wu and Z.-H. Zhou. A unified view of multi-label performance measures. In: ICML'17, Sydney, Australia, 2017, pp.3780-3788.]中对macro-$F1$和micro-$F1$的定义就与式(2.14)和式(2.17)不同，其中macro-$F1$定义如下：

$$\text{macro-}F1 = \frac{1}{n} \sum_{j=1}^{n} (F1)_j = \frac{1}{n} \sum_{j=1}^{n} \frac{2 \sum_{i=1}^{m_j} y_{ij} h_{ij}}{\sum_{i=1}^{m_j} h_{ij} + \sum_{i=1}^{m_j} y_{ij}}$$

其中$(F1)_j$表示第j个二分类问题的$F1$值，y_{ij}和h_{ij}分别表示第j个二分类问题第i个样本的标记和二值化预测结果，m_j表示第j个二分类问题的样本个数；micro-$F1$定义如下：

$$\begin{aligned} \text{macro-}F1 =& \frac{2 \sum_{j=1}^{n} \sum_{i=1}^{m_j} y_{ij} h_{ij}}{\sum_{j=1}^{n} \sum_{i=1}^{m_j} h_{ij} + \sum_{j=1}^{n} \sum_{i=1}^{m_j} y_{ij}} \\ =& \frac{2 \times \sum_{j=1}^{n} TP_j}{\sum_{j=1}^{n} (TP_j + FN_j) + \sum_{j=1}^{n} (TP_j + FP_j)} \end{aligned}$$

这里定义的macro-$F1$与macro-P和macro-R三者之间不再满足类似于式(2.10)的关系；micro-$F1$与micro-P和micro-R三者之间也不再满足类似于式(2.10)的关系。

6、式(2.18)和式(2.19)的解释

式(2.18)定义了**真正例率** TPR。先解释式子中出现的真正例和假反例，真正例即实际为正例预测结果也为正例，假反例即实际为正例但预测结果为反例；式(2.18)分子为真正例，分母为真正例和假反例之和（即实际的正例个数），因此式(2.18)的含义是**所有正例当中有多大比例被预测为正例**（即查全率 Recall，参见式(2.9)）。

式(2.19)定义了**假正例率** FPR。先解释式子中出现的假正例和真反例，假正例即实际为反例但预测结果为正例，真反例即实际为反例预测结果也为反例；式(2.19)分子为假正例，分母为真反例和假正例之和（即实际的反例个数），因此式(2.19)的含义是**所有反例当中有多大比例被预测为正例**。

总结一下，**真正例率**是所有正例中预测为正例的比例（即查全率 Recall），**假正例率**是所有反例中预测为正例的比例。除了真正例率 TPR 和假正例率 FPR，还有真反例率 TNR 和假反例率 FNR，有点儿迷糊没关系，更多可参见后面的“ROC 与 AUC 的再解释”。

7、图 2.4 中的 ROC 曲线解释

先解释两种特殊情形，即**“对角线对应于‘随机猜测’模型，而点$(0,1)$则对应于将所有正例排在所有反例之前的‘理想模型’”**。

为了理解这句话，先来看一下 ROC 绘图过程：**“给定m^+个正例和m^-个反例，根据学习器预测结果对样例进行排序，然后把分类阈值设为最大，即把所有样例均预测为反例，此时真正例率和假正例率均为 0**（无样例被预测为正例，因此真正例 TP 和假正例 FP 均为 0，根据式(2.18)和式(2.19)可知真正例率 TPR 和假正例率 FPR 均为 0），**在坐标$(0,0)$处标记一个点；然后，将分类阈值依次设为每个样例的预测值，即依次将每个样例划分为正例；设前一个标记点坐标为(x,y)，当前若为真正例，则对应标记点的坐标为$(x,y+1/m^+)$；当前若为假正例，则对应标记点的坐标为$(x+1/m^-,y)$，然后用线段连接相邻点即得**（阈值依次减小，最终所有样例均被预测为正例，每有一个正例随阈值减小被预测为正例，则真正例率 TPR 增加$1/m^+$，每有一个反例随阈值减小被预测为正例，则反正例率 FPR 增加$1/m^-$，整个纵坐标轴 TPR 被分为m^+份，整个横坐标轴被分为m^-份，整个绘图过程除原点外需要$m^+ + m^-$步）” 。

对于“理想模型”，即将所有正例排在所有反例之前，在画 ROC 曲线过程中，由于前m^+个样例均为真正例，即从$(0,0)$一直沿纵轴方向增加m^+次，每次增加$1/m^+$，到达坐标$(0,1)$；接下来m^-个样例均为假正例，即一直沿着横坐标走，轨迹为从$(0,1)$至$(1,1)$。

对于“随机猜测”模型，即根据预测结果对样例进行排序基本是正例和反例均匀分布（假如正例个数等于反例个数，则应该是一个正例一个反例），在画 ROC 曲线过程中，向纵方向走一步，然后向横轴方向走一步，因此轨迹为从$(0,0)$至$(1,1)$。

例如，包含 20 个样本的理想模型和随机猜测模型样本排序如下：

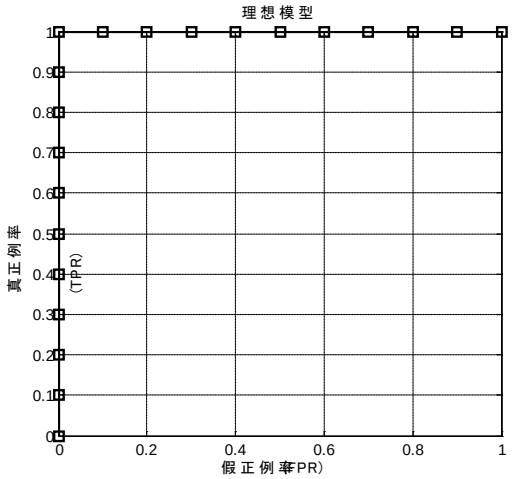
理想模型：＋＋＋＋＋＋＋＋＋＋－－－－－－－－－－

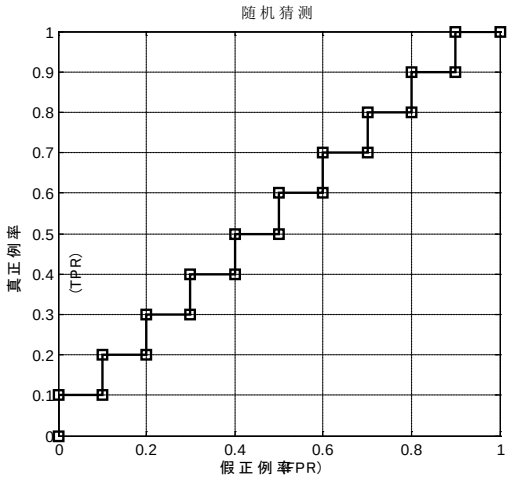
随机猜测：＋－＋－＋－＋－＋－＋－＋－＋－＋－＋－

执行如下 Matlab 代码，可分别绘出理想模型和随机猜测模型的 ROC 曲线：

```
TPRideal=[0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0];
FPRideal=[0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0];
TPRrand=[0 0.1 0.1 0.2 0.2 0.3 0.3 0.4 0.4 0.5 0.5 0.6 0.6 0.7 0.7 0.8 0.8 0.9 0.9 1.0 1.0];
FPRrand=[0 0.0 0.1 0.1 0.2 0.2 0.3 0.3 0.4 0.4 0.5 0.5 0.6 0.6 0.7 0.7 0.8 0.8 0.9 0.9 1.0];
figure;
subplot(121);plot(FPRideal,TPRideal,'ks-','linewidth',2);
axis([0 1 0 1]);xlabel('假正例率(FPR)');ylabel('真正例率(TPR)');title('理想模型');grid on;
```

```
subplot(122);plot(FPRrand,TPRrand,'ks-','linewidth',2);
axis([0 1 0 1]);xlabel('假正例率(FPR)');ylabel('真正例率(TPR)');title('随机猜测');grid on;
```



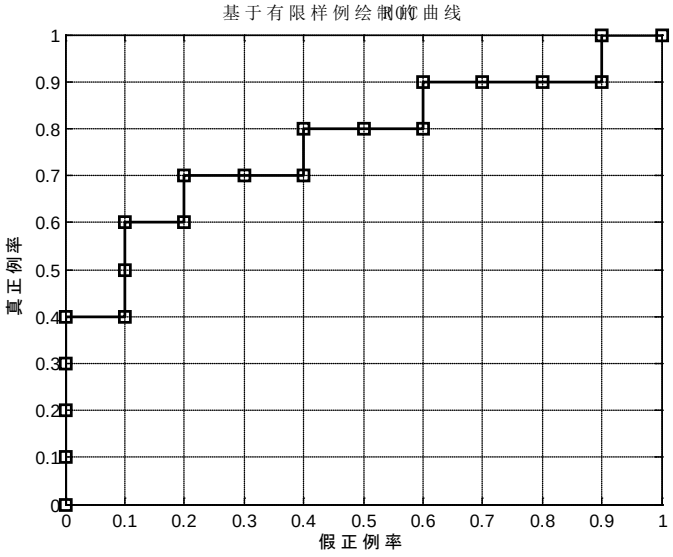


下面还给出了一个包含 20 个样本的一般模型样本排序如下

＋＋＋＋－＋＋－＋－－＋－－＋－－－＋－

执行如下 Matlab 代码，可绘出其对应的 ROC 曲线：

```
TPR=[0 0.1 0.2 0.3 0.4 0.4 0.5 0.6 0.6 0.7 0.7 0.7 0.8 0.8 0.8 0.9 0.9 0.9 0.9 1.0 1.0];
FPR=[0 0.0 0.0 0.0 0.0 0.1 0.1 0.1 0.2 0.2 0.3 0.4 0.4 0.5 0.6 0.6 0.7 0.8 0.9 0.9 1.0];
plot(FPR,TPR,'ks-','linewidth',2);axis([0 1 0 1]);
xlabel('假正例率');ylabel('真正例率');title('基于有限样例绘制的 ROC 曲线');grid on;
```



8、式(2.20)的推导

实际上，只要把这个式子变形一下就容易理解了：

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

其中$(x_{i+1}-x_i)$为矩形的底，$\frac{(y_{i+1}+y_i)}{2}$为矩形的高（取y_{i+1}和y_i的平均值）。由前面 ROC 曲线绘制过程可以看出，ROC 曲线要么沿纵轴方向走，要么沿横轴方向走；当曲线沿纵轴往

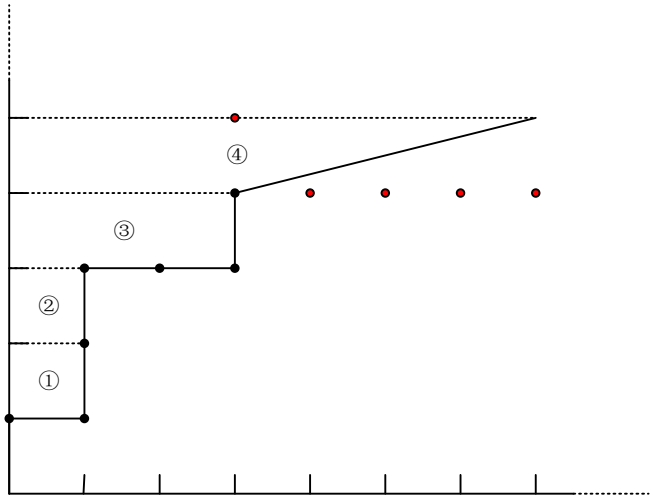
上走时有$x_{i+1}-x_i=0$，而当曲线沿横轴往右走时有$y_{i+1}=y_i$（暂不考虑存在正例和反例预测值相等的情况）。

其实，换个思维去理解也容易：每一小块面积都是一个梯形（矩形也是梯形），梯形的上底和下底是y_i和y_{i+1}，高为$(x_{i+1}-x_i)$，梯形的面积公式为：面积=(上底+下底)*高/2

9、式(2.21)和式(2.22)的推导

式(2.21)是一个定义，这里主要难点是书中式(2.21)下方第 2 行的“容易看出，ℓ_{rank}对应的是 ROC 曲线之上的面积”这句话；若理解此点后，式(2.22)就很简单了，AUC 是 ROC 曲线之下的面积，ℓ_{rank}是 ROC 曲线之上的面积，曲线之上和曲线之下的面积之和等于 1。

其实在图 2.4(b)中，有一种特殊情况没有显示，即当学习器的预测结果对于某些正例和某些反倒相同时该如何绘制曲线？若仍然依次将每个样例划分为正例，则曲线仍旧是折线（要么沿纵轴方向走，要么沿横轴方向走），但这个折线将不唯一，这与这些预测结果相同的正例和反例的顺序有关（因为相同，所以排序随机）；若反例在前，则先沿横轴方向走再沿纵轴方向走；若正例在前，则先沿纵轴方向走再沿横轴方向走，当然最终的终点是一样的；但折线不同，则折线之上和折线之下的面积就不同，而平均的效果是走斜线。下面用一个例子来说明求解 ROC 曲线之上面积的过程：



如图所示，横轴每一格为$1/m^-$，纵轴每一格为$1/m^+$，前七个点的走势是“上-横-上-上-横-横-上”，即前七个点分别为“正-反-正-正-反-反-正”例，与纵轴之间共形成三块面积①②③，分别计算：

(1)面积①大小为：$\frac{1}{m^+}\times\left(1\times\frac{1}{m^-}\right)$

其中 1 表示第三个点（正例）前面有一个反例（第二个点）；

(2)面积②大小为：$\frac{1}{m^+}\times\left(1\times\frac{1}{m^-}\right)$

其中 1 表示第四个点（正例）前面有一个反例（第二个点）；

(3)面积③大小为：$\frac{1}{m^+}\times\left(3\times\frac{1}{m^-}\right)$

其中 3 表示第七个点（正例）前面有三个反例（第二个点、第五个点、第六个点）；

实际上第一个点（正例）也形成一块面积，只是面积大小为零：$\frac{1}{m^+}\times\left(0\times\frac{1}{m^-}\right)$，其中 0 表示第一个点（正例）前面有零个反例。

从以上三块面积的计算可以看出，**若没有正例和反例预测结果相同的情况**，每个正例对应一块面积，面积大小为$\frac{1}{m^+} \times \frac{n(\boldsymbol{x}^+)}{m^-}$，其中$n(\boldsymbol{x}^+) = \sum_{\boldsymbol{x}^- \in D^-} \mathbb{I}(f(\boldsymbol{x}^+) < f(\boldsymbol{x}^-))$，即排序在该正例前面的反例个数（$f(\boldsymbol{x})$越大越排前面）。将所有面积合在一起，即得

$$\begin{aligned}\ell_{\text{rank}} &= \sum_{\boldsymbol{x}^+ \in D^+} \left(\frac{1}{m^+} \times \frac{n(\boldsymbol{x}^+)}{m^-} \right) \\ &= \sum_{\boldsymbol{x}^+ \in D^+} \left(\frac{1}{m^+} \times \frac{1}{m^-} \times \sum_{\boldsymbol{x}^- \in D^-} \mathbb{I}\left(f(\boldsymbol{x}^+) < f(\boldsymbol{x}^-)\right) \right) \\ &= \frac{1}{m^+} \times \frac{1}{m^-} \sum_{\boldsymbol{x}^+ \in D^+} \sum_{\boldsymbol{x}^- \in D^-} \mathbb{I}\left(f(\boldsymbol{x}^+) < f(\boldsymbol{x}^-)\right)\end{aligned}$$

其中第二个等号就是将$n(\boldsymbol{x}^+)$代入。

若出现正例和反例预测结果相同的情况，如第八个点至第十二个点，共有一个正例和四个反例预测结果相同，此时折线走势有好多种情况；但前面也分析了，平均情况是如图斜线所示，此时形成面积④；面积④是一个梯形，其中：

(1)梯形上底（短底）为$3 \times \frac{1}{m^-}$，其中 3 表示预测结果相同的样例前面有三个反例（第二个点、第五个点、第六个点）；

(2)梯形下底（长底）为$3 \times \frac{1}{m^-} + 4 \times \frac{1}{m^-}$，其中 4 表示当前预测结果相同的五个样例共有四个反例；

(3)梯形的高为$1 \times \frac{1}{m^+}$，其中 1 表示当前预测结果相同的五个样例共有一个正例；

根据梯形面积公式：面积=(上底+下底)*高/2，当前梯形面积为

$$\left(3 \times \frac{1}{m^-} + 3 \times \frac{1}{m^-} + 4 \times \frac{1}{m^-}\right) \times \left(1 \times \frac{1}{m^+}\right) \times \frac{1}{2}$$

用$n(\boldsymbol{x}^*) = \sum_{\boldsymbol{x}^- \in D^-} \mathbb{I}(f(\boldsymbol{x}^*) < f(\boldsymbol{x}^-))$表示预测结果相同的样例前面反例个数（此处等于 3），用$s^-(\boldsymbol{x}^*)$表示当前预测结果相同的样例当中反例的个数（此处等于 4），$s^+(\boldsymbol{x}^*)$表示当前预测结果相同的样例当中正例的个数（此处等于 1），则上式可写为

$$\left(n(\boldsymbol{x}^*) \times \frac{1}{m^-} + n(\boldsymbol{x}^*) \times \frac{1}{m^-} + s^-(\boldsymbol{x}^*) \times \frac{1}{m^-}\right) \times \left(s^+(\boldsymbol{x}^*) \times \frac{1}{m^+}\right) \times \frac{1}{2}$$

整理一下，将$1/2$放入(上底+下底)内部，并将公因式$1/m^-$和$1/m^+$提出，得

$$\frac{1}{m^+} \times \frac{1}{m^-} \times \left(n(\boldsymbol{x}^*) + s^-(\boldsymbol{x}^*) \times \frac{1}{2}\right) \times s^+(\boldsymbol{x}^*)$$

注意这个结果与当前预测结果相同的样例当中包含正例的个数$s^+(\boldsymbol{x}^*)$成正比，即可以看成是由$s^+(\boldsymbol{x}^*)$个

$$\frac{1}{m^+} \times \frac{1}{m^-} \times \left(n(\boldsymbol{x}^*) + s^-(\boldsymbol{x}^*) \times \frac{1}{2}\right)$$

相加而得（每个正例均对应 1 块上述面积），针对$s^+(\boldsymbol{x}^*)$个正例当中任意一个$\boldsymbol{x}^+$，

$$n(\boldsymbol{x}^*) = n(\boldsymbol{x}^+) = \sum_{\boldsymbol{x}^- \in D^-} \mathbb{I}(f(\boldsymbol{x}^+) < f(\boldsymbol{x}^-))$$

$$s^-(\boldsymbol{x}^*) = \sum_{\boldsymbol{x}^- \in D^-} \mathbb{I}(f(\boldsymbol{x}^+) = f(\boldsymbol{x}^-))$$

易知，对于第一种情况“若没有正例和反例预测结果相同的情况”是当前情况的一种特例，即$s^-(\boldsymbol{x}^*)=0$时，因此可得ℓ_{rank}通用的公式(2.21)

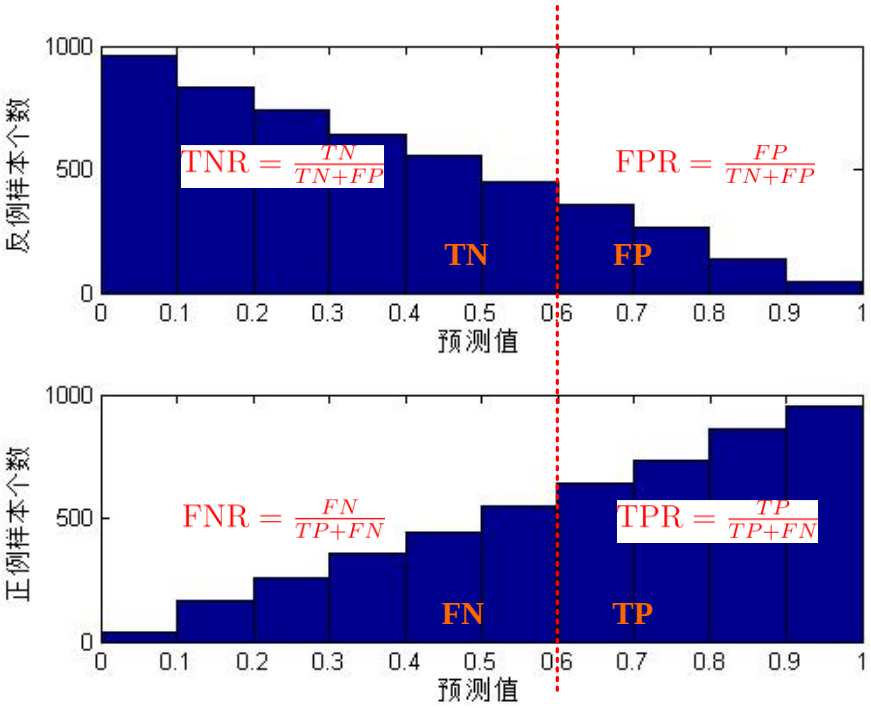
$$\begin{aligned}\ell_{\text{rank}} &= \sum_{\boldsymbol{x}^+\in D^+}\left(\frac{1}{m^+}\times\frac{1}{m^-}\times\left(n(\boldsymbol{x}^*)+s^-(\boldsymbol{x}^*)\times\frac{1}{2}\right)\right)\\ &= \sum_{\boldsymbol{x}^+\in D^+}\left(\frac{1}{m^+}\times\frac{1}{m^-}\times\left(\sum_{\boldsymbol{x}^-\in D^-}\mathbb{I}\left(f(\boldsymbol{x}^+)<f(\boldsymbol{x}^-)\right)+\sum_{\boldsymbol{x}^-\in D^-}\mathbb{I}\left(f(\boldsymbol{x}^+)=f(\boldsymbol{x}^-)\right)\times\frac{1}{2}\right)\right)\\ &= \frac{1}{m^+}\times\frac{1}{m^-}\times\sum_{\boldsymbol{x}^+\in D^+}\sum_{\boldsymbol{x}^-\in D^-}\left(\mathbb{I}\left(f(\boldsymbol{x}^+)<f(\boldsymbol{x}^-)\right)+\frac{1}{2}\mathbb{I}\left(f(\boldsymbol{x}^+)=f(\boldsymbol{x}^-)\right)\right)\end{aligned}$$

即式(2.21)的ℓ_{rank}为 ROC 曲线之上的面积。在 ROC 曲线图中，横轴假正例率和纵轴真正例率均为从 0 至 1，因此整块面积为 1；而 AUC 为 ROC 曲线之下的面积，所以有式(2.22)。

10、ROC 与 AUC 的再解释

看书到此，可能感觉也知道了什么是 ROC、什么是 AUC，却又感觉没有理解其含义。

先梳理四个概念：真正例率 TPR、假正例率 FPR、真反例率 TNR、假反例率 FNR，各概念的定义如下图所示：



注：上图由如下步骤绘出，首先随机生成了 10000 个样本预测值（位于 0~1 之间）；然后将每个样本随机指派为正例或反例，样本预测值越大被指派为正例的概率越大，反之被指派为反例的概念越大，指派结果作为对应的 10000 个样本真实标记；最后将 0~1 划分为 10 个区间，分别统计每个区间内包含的正例个数和反例个数，并用直方图表示。

由上图可以看出，无论分类域值设置为多少，肯定都会有样本被分类错误，这也是实际当中最常见的情形。设当前分类域值等于 0.6，则可以分别得到真正例、假正例、真反例、假反例的个数 TP、FP、TN、FN，进而可依次计算出真正例率 TPR、假正例率 FPR、真反例率 TNR、假反例率 FNR。

实际上，TPR、FPR、TNR、FNR 就是把表 2.1 中的四个值均除以对应的行元素之和，而行元素之和分别表示正例样本个数和反例样本个数，因此还可以使用条件概率做如下解释：

$$\begin{aligned}\mathrm{TPR} &= P\left(h(\boldsymbol{x})=1 \mid f(\boldsymbol{x})=1\right)\\ \mathrm{FPR} &= P\left(h(\boldsymbol{x})=1 \mid f(\boldsymbol{x})=0\right)\\ \mathrm{TNR} &= P\left(h(\boldsymbol{x})=0 \mid f(\boldsymbol{x})=0\right)\\ \mathrm{FNR} &= P\left(h(\boldsymbol{x})=0 \mid f(\boldsymbol{x})=1\right)\end{aligned}$$

其中$f(\boldsymbol{x})$为样本$\boldsymbol{x}$真实标记，$h(\boldsymbol{x})$为样本$\boldsymbol{x}$预测标记，1表示正例，0表示反例；即 TPR 为当样本为正例时被预测为正例的概率（**正例**被**预测正确**的比例）、FPR 为当样本为反例时被预测为正例的概率（**反例**被**预测错误**的比例）、TNR 为当样本为反例时被预测为反例的概率（**反例**被**预测正确**的比例）、FNR 为当样本为正例时被预测为反例的概率（**正例**被**预测错误**的比例）。例如测试集包含$m=m^{+}+m^{-}$个样本，其中m^{+}表示正例样本个数（即$\mathrm{TP}+\mathrm{FN}$），m^{-}表示反例样本个数（即$\mathrm{FP}+\mathrm{TN}$），则总的预测错误的样本个数为$m^{+}\times\mathrm{FNR}+m^{-}\times\mathrm{FPR}$，预测正确的样本个数为$m^{+}\times\mathrm{TPR}+m^{-}\times\mathrm{TNR}$。

接下来对 ROC 和 AUC 做几条解释：

(1)由上图还可以看出，绘制 ROC 曲线时使用的 TPR 和 FPR，都是基于分类域值右侧（大于分类域值）的样本。

(2)由式(2.21)可知，ℓ_{rank}表示**任取一对正例和反例，正例预测值小于反例预测值的概率**（简单起见可以暂不考虑预测值相等的情况）；这是因为分母$m^{+}m^{-}$表示所有正例和反例组合对的个数，分子的求和项表示正例和反例组合对中，正例预测值小于反例预测值的组合对个数。显然，这个概率越小越好（正例预测值应该大于反例预测值）。

(3)由式(2.22)，$\mathrm{AUC}+\ell_{\mathrm{rank}}=1$，因此 AUC 表示**任取一对正例和反例，正例预测值大于反例预测值的概率**。显然，这个概率越大越好。

(4)按西瓜书所述，ROC 曲线绘制的过程，其实就是将上图中分类域值从右往左（即从 1 到 0）滑动一遍，依次记录每个域值时的 TPR 和 FPR，描点画线即可。

(5)ROC 曲线的绘制所使用的样本集应该是测试集，即 ROC 曲线上的每个点表示了当取不同分类域值时，分类器泛化性能的度量（正例样本的精度 TPR 和反例样本的错误率 FPR）。

(6)进一步深入讨论一下 ROC 曲线的绘制，在 2.3.4 节即将提到的代价曲线原文献[Drummond et al., MLJ06]中，也提到了 ROC 曲线的绘制，但与西瓜书中直观的讲法不同：

> The method used to generate the set of ROC points for a given classifier (or learning algorithm) depends on the classifier. Some classifiers have parameters for which different settings produce different ROC points. For example, with naive Bayes (Duda & Hart, 1973; Clark & Niblett, 1989) an ROC curve is produced by varying its threshold parameter. If such a parameter does not exist, algorithms such as decision trees can be modified to include costs producing different trees corresponding to different points (Breiman et al., 1984). The counts at the leaves may also be modified, thus changing the leaf's classification, allowing a single tree to produce multiple points (Bradford et al., 1998; Ferri et al., 2002). Alternatively the class frequencies in the training set can be changed by under- or oversampling to simulate a change in class priors or misclassification costs (Domingos, 1999).
>
> Figure 6 shows a set of ROC points for C4.5 on the sonar data set from the UCI collection. Each point corresponds to a classifier trained using a different class distribution produced by undersampling the training set. Even though ROC analysis does not commit to any particular

对于 ROC 曲线中的各个点(TPR, FPR)，西瓜书中针对某 predictor（可理解为概率预测函数）是由遍历不同分类阈值而得；而以上文献也讲了一些方法（没看懂），反正就是得到一堆不同条件下的(TPR, FPR)，描点连线即可。这里想说的是，ROC 曲线上每一个点都表示该 predictor 在特定条件下得到的一个二分类器，我们当然希望该二分类器的假正例率（反例被预测错误的概率，横轴）越小越好，真正例率（正例被预测正确的概率，纵轴）越大越好，

所以这个点越靠左上角（即点(0,1)）越好。

以上观点参考了以下两篇博客：

ROC 曲线与 AUC 值：(https://www.cnblogs.com/gatherstars/p/6084696.html)

AUC，ROC 我看到的最透彻的讲解：(https://blog.csdn.net/u013385925/article/details/80385873)

11、式(2.23)的解释

本式很容易理解，只是注意该式上方第三行提到“若将表 2.2 中的**第 0 类作为正类、第 1 类作为反类**”，若不注意这句话可能按习惯（0 为反类、1 为正类）会产生误解。**为避免这种尴尬的情况，在接下来的解释中将**$cost_{01}$**记为**$cost_{+-}$**，**$cost_{10}$**记为**$cost_{-+}$。

本式可以进一步扩展。首先，对本式变形如下：

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

其中m^+和m^-分别表示正例子集D^+和反例子集D^-的样本个数。由本式的形式可知，本式已经假定了样例集D中样本的类别信息已知（训练集或测试集）。其中：

$\frac{1}{m^+}\sum_{\boldsymbol{x}_i \in D^+}\mathbb{I}(f(\boldsymbol{x}_i)\neq y_i))$表示正例子集$D^+$预测错误样本所占比例，即**假反例率** FNR；

$\frac{1}{m^-}\sum_{\boldsymbol{x}_i \in D^-}\mathbb{I}(f(\boldsymbol{x}_i)\neq y_i))$表示反例子集$D^-$预测错误样本所占比例，即**假正例率** FPR；

$\frac{m^+}{m}$表示样例集D中**正**例所占比例，或理解为随机从D中取一个样例取到**正**例的概率；

$\frac{m^-}{m}$表示样例集D中**反**例所占比例，或理解为随机从D中取一个样例取到**反**例的概率；

因此，对于$E(f;D;cost)$，若用p表示样例为正例的概率，则样例为反例的概率为$1-p$，上式可进一步写为

$$E(f;D;cost) = p \times \mathrm{FNR} \times cost_{01} + (1-p) \times \mathrm{FPR} \times cost_{10}$$

这实际上就是期望代价的表达式。

12、式(2.24)的解释

(1)当$cost_{+-} = cost_{-+}$时，将式(2.24)分子分母约分，得：

$$P(+)cost = \frac{p}{p+(1-p)} = p$$

其中p是样例为正例的概率（或者近似认为是正例在样本集中所在的比例）。也就是说，当代价不敏时($cost_{+-} = cost_{-+}$)，$P(+)cost$就是样例为正例的概率；那么，当代价敏感时($cost_{+-} \neq cost_{-+}$)，可称$P(+)cost$为正例所占的加权比例。具体来说，对于样本集合

$$D = \{\boldsymbol{x}_1^+, \boldsymbol{x}_2^+, \boldsymbol{x}_3^-, \boldsymbol{x}_4^-, \boldsymbol{x}_5^-, \boldsymbol{x}_6^-, \boldsymbol{x}_7^-, \boldsymbol{x}_8^-, \boldsymbol{x}_9^-, \boldsymbol{x}_{10}^-\}$$

其中$\boldsymbol{x}^+$表示正例，$\boldsymbol{x}^-$表示反例；可以看出$p=0.2$；若想让正例得到更多重视，考虑敏感代价$cost_{+-}=4$和$cost_{-+}=1$（即正例预测出错时，承受的代价是 4），这实际等价于在以下数据集上进行代价不敏感的训练：

$$D' = \{\boldsymbol{x}_1^+, \boldsymbol{x}_1^+, \boldsymbol{x}_1^+, \boldsymbol{x}_1^+, \boldsymbol{x}_2^+, \boldsymbol{x}_2^+, \boldsymbol{x}_2^+, \boldsymbol{x}_2^+, \boldsymbol{x}_3^-, \boldsymbol{x}_4^-, \boldsymbol{x}_5^-, \boldsymbol{x}_6^-, \boldsymbol{x}_7^-, \boldsymbol{x}_8^-, \boldsymbol{x}_9^-, \boldsymbol{x}_{10}^-\}$$

即将每个正例样本复制 4 份（若$\boldsymbol{x}_1^+$出错，则有四个$\boldsymbol{x}_1^+$一起出错，相当于进行代价敏感的训练时$\boldsymbol{x}_1^+$的代价为 4，参见式(2.23)），此时可计算出$P(+)cost=0.5$：

$$P(+)cost = \frac{p \times cost_{+-}}{p \times cost_{+-} + (1-p) \times cost_{-+}} = \frac{0.2 \times 4}{0.2 \times 4 + (1-0.2) \times 1} = 0.5$$

也就是正例样本在等价的数据集D'中所占比例。

(2)对于不同的$cost_{+-}$和$cost_{-+}$取值对，若二者的比值保持相同，则$P(+)cost$不变。例如，对于上面的例子，若设$cost_{+-} = 40$和$cost_{-+} = 10$，所得$P(+)cost$仍为 0.5。

(3)根据此式，还可以相应地有反例概率代价$P(-)cost$：

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

13、式(2.25)的解释

(1)补充一：对于空间中的两个点A和B（当是一维空间时即两个值）和$p \in [0,1]$，则$pA + (1-p)B$一定在线段AB上；当p从 0 变到 1 时，点$pA + (1-p)B$的轨迹为从B到A。特别地，当A和B为两个值且假设$A > B$时，有

$$B \leq pA + (1-p)B \leq A$$

易知，当$p = 1$时，$pA + (1-p)B = A$；当$p = 0$时，$pA + (1-p)B = B$。

实际上，这个结论很容易理解，只要将表达式换种形式即可

$$pA + (1-p)B = p(A-B) + B$$

这里B为线段起点，$A - B$为线段长度；例如对于$p \in [0,1]$，有$6 \leq 6 + p(8-6) \leq 8$。

(2)补充二：对于包含m个样本的测试集，分类器$h(\boldsymbol{x})$总的代价（或损失）是

$$\begin{aligned} cost_{se} =& m \times p \times \mathrm{FNR} \times cost_{+-} + m \times (1-p) \times \mathrm{FPR} \times cost_{+-} \\ &+ m \times p \times \mathrm{TPR} \times cost_{++} + m \times (1-p) \times \mathrm{TNR} \times cost_{--} \end{aligned}$$

其中p是正例在样本集中所在的比例（或严格地称为测试集中样例为正例的概率），则$m \times p$表示测试集中正例样本个数，$m \times (1-p)$表示测试集中反例样本个数；下标中的“se”表示 sensitive，即对代价敏感；进一步地，根据 TPR、FPR、TNR、FNR 的含义（参见前面的“ROC 与 AUC 的再解释”），则有

$m \times p \times \mathrm{FNR}$表示正例预测为反例（正例预测错误）的样本个数

$m \times (1-p) \times \mathrm{FPR}$表示反例预测为正例（反例预测错误）的样本个数

$m \times p \times \mathrm{TPR}$表示正例预测为正例（正例预测正确）的样本个数

$m \times (1-p) \times \mathrm{TNR}$表示反例预测为反例（反例预测正确）的样本个数

以上各种样本个数乘以相应的代价则得到总的代价$cost$。但是，这个代价与测试集样本个数有关（例如，若依上式方法计算，同一个分类器在 100 个样本的测试集上和 1000 个样本的测试集上代价相差 10 倍），因此一般除以测试集样本个数得到期望代价（类似式(2.23)），即

$$\begin{aligned} \mathbb{E}[cost_{se}] =& p \times \mathrm{FNR} \times cost_{+-} + (1-p) \times \mathrm{FPR} \times cost_{+-} \\ &+ p \times \mathrm{TPR} \times cost_{++} + (1-p) \times \mathrm{TNR} \times cost_{--} \end{aligned}$$

一般来说，预测出错才会产生代价，预测正确则没有代价，即$cost_{++} = cost_{--} = 0$，即

$$\mathbb{E}[cost_{se}] = p \times \mathrm{FNR} \times cost_{+-} + (1-p) \times \mathrm{FPR} \times cost_{+-}$$

此即式(2.23)的解释中的扩展内容；在代价不敏感场景下，只要预测出错，一般就记 1 个损失，即$cost_{+-} = 1$，$cost_{-+} = 1$；因此上式变为

$$\mathbb{E}[cost_{un}] = p \times \mathrm{FNR} + (1-p) \times \mathrm{FPR}$$

其中下标“un”表示 unsensitive。

由补充一的解释可知，$\mathbb{E}[cost_{se}]$位于$\mathrm{FNR} \times cost_{+-}$与$\mathrm{FPR} \times cost_{+-}$之间，$\mathbb{E}[cost_{un}]$位于$\mathrm{FNR}$与$\mathrm{FPR}$之间；进一步地，由 FNR、FPR 和$cost_{+-}$、$cost_{-+}$的含义可知，

$\mathbb{E}[cost_{un}]$一定是 0 到 1 之间的值，而$\mathbb{E}[cost_{se}]$取值与$cost_{+-}$、$cost_{-+}$有关。

(3)由式(2.24)中$P(+)cost$的定义，可将式(2.25)重写为：

$$cost_{norm} = \mathrm{FNR} \times P(+)cost + \mathrm{FPR} \times (1 - P(+)cost)$$

在式(2.24)中解释到$P(+)cost$表示在代价敏感时正例所占的加权比例，那么$1 - P(+)cost$就表示在代价敏感时反例所占的加权比例；再结合补充二有关代价不敏感时期望代价$\mathbb{E}[cost_{un}]$的表达式，可知$cost_{norm}$表示在代价敏感时的期望代价。

实际上，$cost_{norm}$就是对补充二中的$\mathbb{E}[cost_{se}]$进行了归一化。具体来说，$\mathbb{E}[cost_{se}]$的最大值当$\mathrm{FNR} = \mathrm{FPR} = 1$时取得，即所有样本全部预测错误时期望代价最大（废话），即

$$\max\left(\mathbb{E}[cost_{se}]\right) = p \times cost_{+-} + (1 - p) \times cost_{+-}$$

对$\mathbb{E}[cost_{se}]$归一化，即

$$\frac{\mathbb{E}[cost_{se}]}{\max\left(\mathbb{E}[cost_{se}]\right)} = \frac{p \times \mathrm{FNR} \times cost_{+-} + (1 - p) \times \mathrm{FPR} \times cost_{+-}}{p \times cost_{+-} + (1 - p) \times cost_{+-}}$$

(4)结合补充一的解释，若FNR用代价平面上的点$(1, \mathrm{FNR})$表示，FPR用代价平面上的点$(0, \mathrm{FPR})$表示，则$cost_{norm}$表示两点之间连线上的任意某个点；当$P(+)cost$从 0 变到 1 时，$cost_{norm}$轨迹从$(0, \mathrm{FPR})$到$(1, \mathrm{FNR})$，即图 2.5 中的各条线段。

14、图 2.5 的解释

首先，图 2.5 的纵轴由“归一化代价”改为“归一化期望代价”更容易理解，阴影部分由“期望总体代价”改为“总体期望代价”更容易理解。

这里关键在于如何理解“然后取所有线段的下界，围成的面积（即图 2.5 中的阴影部分，个人注）即为在所有条件下学习器的期望总体代价”这句话？

要想理解这个问题，就要知道“所有条件下”到底指的哪些条件？原话中“所有条件下学习器的期望总体代价”的名词部分为“期望总体代价”，“所有条件下学习器的”作为定语修饰名词。图 2.5 中纵轴指归一化代价，所以这里的“所有条件”应该指图 2.5 的横轴，进一步地“期望总体代价”就是代价曲线对横轴变量的积分。从图 2.5 中可以看到横坐标轴为正例概率代价$P(+)cost$，即式(2.24)；前面已经分析过，当代价不敏感时，$P(+)cost$将退化为样例为正例的概率p（即当不考虑代价敏感的情况时横轴取为p）。

那么问题又来了，“样例为正例的概率p”是针对哪个数据集的？训练集 or 测试集？难道训练集和测试集的“样例为正例的概率p”不是固定的么？我们使用训练集得到分类器，然后基于测试集画出图 2.4 的 ROC 曲线和图 2.5 的代价曲线，这里使用的训练集和测试集已知，即“样例为正例的概率p”（或正例所占的比例）已知，那么代价曲线再以p为自变量还有什么意义么？

实际上，代价曲线的原始文献[Drummond et al., MLJ06]专门对此做了强调（原文献中$p(+)$就是此处的p）。原文中提到了三个$p(+)$，即 “$p_{train}(+)$ is the percentage of positive examples in the dataset used to learn the classifier. $p_{test}(+)$ is the percentage of positive examples in the dataset used to build the classifier’s confusion matrix. $p_{deploy}(+)$ is the percentage of positive examples when the classifier is deployed (put to use).”，而$p(+)$指的是$p_{deploy}(+)$，也就是未来真正使用该学习器时遇到的数据集（记为 deployed set）样例为正例的比例，而这个比例是会变化的，原因参见原文献“One reason is that the class distribution and costs can change with time, or with the location where the classifier is deployed. The second reason is that the distribution in the datasets used for training and evaluating the classifier may not reflect reality.”，在下一条解释中也会提及。

It is important to be perfectly clear about the exact meaning of $p(+)$, because in the lifetime of a classifier learned from training data there are at least three distinct sets of examples that might each have a different proportion of positive examples. $p_{train}(+)$ is the percentage of positive examples in the dataset used to learn the classifier. $p_{test}(+)$ is the percentage of positive examples in the dataset used to build the classifier's confusion matrix. $p_{deploy}(+)$ is the percentage of positive examples when the classifier is deployed (put to use). It is $p_{deploy}(+)$ that should be used for $p(+)$, because it is the performance during deployment that we wish to estimate of a classifier. However, because we do not necessarily know $p_{deploy}(+)$ at the time we are learning or evaluating the classifier we would like to visualize the classifier's performance across all possible values of $p_{deploy}(+)$. Cost curves do precisely that.

知道了p的含义之后，可以知道针对 deployed set，它的p是不确定的（疑问：如果训练集和测试集都是经独立同分布从样本空间中采样而得，那么 deployed set 的p难道不应该近似与训练集和测试集相等么？）。假设当前 deployed set 的p等于p_0，那么 ROC 曲线上的所有点（或图 2.5 中的每条从(0,FPR)到(1,FNR)的线段）所对应的二分类器集合（ROC 曲线上的每个点都是一个二分类器，详见前面“ROC 与 AUC 的再解释”中的阐述）之中，能够取得最小的归一化代价就是图 2.5 中代价曲线（红线）当$p = p_0$时的值。遍历所有p的取值，则得图 2.5 中的代价曲线（红线）。也就是说，对于 deployed set，当实际p为不同值时，将由二分类器集合中不同的二分类器获得最小的归一化期望代价。

特别地，当$p = 0$时（即只有反例样本没有正例样本），当 FPR=0 时（即反例样本预测错误率为 0）获得最小期望代价，此时其实并不关心 FNR 的值（即正例样本预测错误的概率），因为没有正例样本，例如对于前面“图 2.4 中的 ROC 曲线解释”中的第 3 幅 ROC 曲线图，将有五对(FPR, FNR)值可取得最小代价，即(0, 1), (0, 0.9), (0, 0.8), (0, 0.7), (0, 0.6)，注意 FNR=1-TPR。当$p = 1$时（即只有正例样本没有反例样本），当 FNR=0 时（即正例样本预测错误率为 0）获得最小期望代价，此时其实并不关心 FPR 的值（即反例样本预测错误的概率），因为没有反例样本，例如对于前面“图 2.4 中的 ROC 曲线解释”中的第 3 幅 ROC 曲线图，将有两对(FPR, FNR)值可取得最小代价，即(0.9, 0), (1, 0)，注意 FNR=1-TPR。由此两点分析可以理解，为何图 2.5 中的代价曲线（红线）在$p = 0$和$p = 1$一定等于 0。

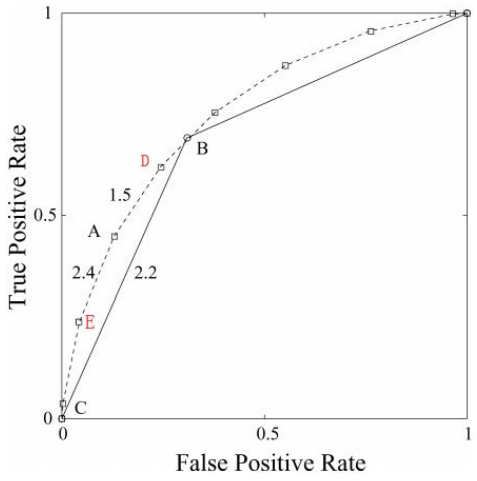
因此，图 2.5 的代价曲线（红线）是针对 deployed set 所有p的取值，由 ROC 曲线上的所有点对应的二分类器集合计算而得的最小的归一化期望代价；代价曲线对变量p求定积分，即代价曲线与横轴的面积，也就是这里定义的期望总体代价。

15、ROC 曲线与代价曲线的比较

这里关键在于如何理解式(2.24)上方一段话提到“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’则可达此目的”的含义，即代价曲线相对于 ROC 曲线究竟多展示了什么信息？

代价曲线可以直观的看出，在 deployed set 取某个正例概率代价（横坐标）的条件下，哪个分类器(TPR, FPR)的期望代价比较小；针对横坐标的每个值，找出所有分类器(TPR, FPR)中最小的期望代价，所有这些最小期望代价构成的曲线就是代价曲线。那么，ROC 曲线是不是也可以看出期望代价呢？或者说，在样例为正例的概率为p、敏感代价权重为$cost_{+-}$和$cost_{-+}$时，ROC 是不是可以比较两个不同的分类器$(\mathrm{TPR}_1, \mathrm{FPR}_1)$和$(\mathrm{TPR}_2, \mathrm{FPR}_2)$的期望代价呢？

答案是肯定的。具体来说，如下图所示（代价曲线原文献[Drummond et al., MLJ06]的 Fig.2，图中线段上的数字表示该线段的斜率）：



其中 B 点分类器记为$(\mathrm{TPR}_1, \mathrm{FPR}_1)$，C 点分类器记为$(\mathrm{TPR}_2, \mathrm{FPR}_2)$，则两点连线的斜率为$\frac{\mathrm{TPR}_1-\mathrm{TPR}_2}{\mathrm{FPR}_1-\mathrm{FPR}_2}$（顺序必须是右上方的点 B 减左下方的点 C），若有

(1) $\frac{\mathrm{TPR}_1-\mathrm{TPR}_2}{\mathrm{FPR}_1-\mathrm{FPR}_2} > \frac{(1-p)\times cost_{-+}}{p\times cost_{+-}}$，则 B 点分类器的期望代价**小于** C 点分类器的期望代价；

(2) $\frac{\mathrm{TPR}_1-\mathrm{TPR}_2}{\mathrm{FPR}_1-\mathrm{FPR}_2} = \frac{(1-p)\times cost_{-+}}{p\times cost_{+-}}$，则 B 点分类器的期望代价**等于** C 点分类器的期望代价；

(3) $\frac{\mathrm{TPR}_1-\mathrm{TPR}_2}{\mathrm{FPR}_1-\mathrm{FPR}_2} < \frac{(1-p)\times cost_{-+}}{p\times cost_{+-}}$，则 B 点分类器的期望代价**大于** C 点分类器的期望代价；

简单证明第(1)个结论，其它两个类同。对于(1)中的条件，变形如下

$$(\mathrm{TPR}_1 - \mathrm{TPR}_2) \times p \times cost_{+-} > (\mathrm{FPR}_1 - \mathrm{FPR}_2) \times (1-p) \times cost_{-+}$$

移项，得

$$-\mathrm{TPR}_2 \times p \times cost_{+-} + \mathrm{FPR}_2 \times (1-p) \times cost_{-+} > -\mathrm{TPR}_2 \times p \times cost_{+-} + \mathrm{FPR}_2 \times (1-p) \times cost_{-+}$$

由于$\mathrm{FNR} = 1 - \mathrm{TPR}$，对上面的不等式两边同时加上$p \times cost_{+-}$，得

$$\mathrm{FNR}_2 \times p \times cost_{+-} + \mathrm{FPR}_2 \times (1-p) \times cost_{-+} > \mathrm{FNR}_2 \times p \times cost_{+-} + \mathrm{FPR}_2 \times (1-p) \times cost_{-+}$$

根据式(2.23)的解释中的扩展内容或式(2.25)的解释中的补充二对期望损失的推导可知，上面不等式的左边就是 C 点分类器的期望损失，右边就是 B 点分类器的期望损失，即 B 点分类器的期望代价**小于** C 点分类器的期望代价，证毕！

另外，期望代价越小（即代价敏感版的错误率，参见式(2.23)的解释），分类器性能越好！

已知p、$cost_{+-}$和$cost_{-+}$，则可计算出$OP = \frac{(1-p)\times cost_{-+}}{p\times cost_{+-}}$；若$2.2 > OP$，则 B 点分类器的期望代价**小于** C 点分类器的期望代价，即 B 点分类器的性能好于 C 点分类器；反之若$2.2 < OP$，则 B 点分类器的性能次于 C 点分类器。

进一步地，若$2.4 > OP > 1.5$，说明 D 点分类器性能次于 A 点分类器（$OP > 1.5$）且 A 点分类器好于 E 点分类器（$2.4 > OP$），也就是说 A 点分类器是最好的；若$1.5 > OP$，则 A 点分类器好于 E 点分类器，且 D 点分类器性能好于 A 点分类器，也就是说 ROC 曲线上的分类器集合中即有比 A 点分类器好的，也有比 A 点分类器差的；若$2.4 < OP$，则 A 点分类器次于 E 点分类器，且 D 点分类器性能次于 A 点分类器，也可以说明 ROC 曲线上的分类器集合中即有比 A 点分类器好的，也有比 A 点分类器差的；即代价曲线原文献中的：

incident to A are the numbers, 1.5 and 2.4, shown in the figure. If the right-hand side of Eq. (1) evaluates to a value between 1.5 and 2.4 classifier A will give the best performance of any of the classifiers on this ROC curve. But if it evaluates to a value outside this range, classifier A will perform worse than one or more other classifiers on this curve.

可以看出，在p、$cost_{+-}$和$cost_{-+}$分别取某值时，从 ROC 曲线观察两个点对应的分类器的性能（期望代价）比较烦琐，很不直观；而代价曲线完全克服了这一点，只要将p、$cost_{+-}$和$cost_{-+}$代入式(2.24)计算出$P(+)cost$，也就是图 2.5 的横轴，就可以看出哪个分类器的期望代价较小。

到此，可以理解代价曲线相比于 ROC 曲线增加了哪些信息。另外，由于图 2.4 的 ROC 曲线中的每个点对应图 2.5 中的一条从(0, FPR)到(1, FNR)的线段，因此图 2.5 中的阴影部分面积和形状，与横轴到底是$P(+)cost$还是p并没有关系，ROC 定了，这块阴影也就固定了。但是书中式(2.24)上方一段话提到“**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’则可达此目的”，为什么这里一定要强调“在非均等代价下”这个条件呢？我们从代价曲线原文献中找找线索……

代价曲线原文献 Introduction 部分提到，机器学习领域一般用错误率（或精度）作为性能指标，现在偶尔也用 AUC，但在代价敏感学习(cost-sensitive learning)中，期望损失更常用：

a research setting, the appropriate measure of performance is not so clear. The machine learning community has traditionally used error rate (or accuracy) as its default performance measure. Recently, however, area under the ROC curve (AUC) has been used in some studies (Bradley, 1997; Karwath & King, 2002; Weiss and Provost, 2003; Yan et al., 2003). In cost-sensitive learning, expected cost under a range of cost matrices has been the preferred measure (Bradford et al., 1998; Domingos, 1999; Margineantu & Dietterich, 2000).

由式(2.23)的解释可知，期望损失相当于代价敏感版本的错误率，即二者在本质意义上相同，所以在代价敏感学习中本来就应该使用期望损失作为性能指标。在代价曲线原文献中，接下来开始说标量度量(scalar measure)的缺点（例如错误率、AUC 等，计算出来就是一个数，称为标量度量）：

An important example of this failing occurs when the cost of misclassifying examples in one class is much different than the cost of misclassifying examples in the other class, or when one class is much rarer than the other (Japkowicz et al., 1995; Kubat et al., 1998; Ling & Li, 1998). A scalar measure can give the expected performance given a probability distribution over costs and class ratios, but it will not indicate for which costs and class ratios one classifier outperforms the other. Adams and Hand (1999) emphasize this point, and specifically mention

接下来就说 ROC 曲线是一个可选的方案，因为它是一个二维图形：

An alternative to scalar measures which overcomes all these difficulties when there are two classes has been known for decades: ROC plots. An ROC plot is a two-dimensional plot, with

夸了一阵 ROC 之后，话峰一转，开始说其实 ROC 有些事情也不能做：

In particular, ROC plots do not allow any of the following important experimental questions to be answered by visual inspection:

- what is classifier C's performance (expected cost) given specific misclassification costs and class probabilities?
- for what misclassification costs and class probabilities does classifier C outperform the trivial classifiers that assign all examples to the same class?
- for what misclassification costs and class probabilities does classifier C1 outperform classifier C2?

- what is the difference in performance between classifier C1 and classifier C2?
- what is the average of performance results from several independent evaluations of classifier C (e.g. from 10-fold cross-validation)?
- what is the 90% confidence interval for classifier C's performance?
- for what misclassification costs and class probabilities is the difference in performance between classifier C1 and classifier C2 statistically significant?

意思就是不能根据 ROC 目测(visual inspection)出以上问题的答案，因此作者提出了代价曲线。

可以看出，上面提到的问题主要是 ROC 不能根据 misclassification costs（即$cost_{+-}$和$cost_{-+}$）和 class probabilities（即p）做一些事情，而在代价曲线中作者直接将这三个参数合成为式(2.24)的正例概率代价作为横坐标，即将 ROC 中的一个点扩展为代价曲线中的一条线，这当然会带来更多的可视化信息。

代价曲线的作者之所以一直强调 misclassification costs 和 class probabilities，作者自己给出了两个原因："One reason is that the class distribution and costs can change with time, or with the location where the classifier is deployed. The second reason is that the distribution in the datasets used for training and evaluating the classifier may not reflect reality."，即p、$cost_{+-}$和$cost_{-+}$可能会随着时间或分类器使用场景而发生变化，还有就是训练和评估分类器的数据集可能并不能反应真实的p。对此，作者还专门举了两个例子，一是同为 UCI 中的 credit application datasets 数据集，Japanese credit dataset 和 German credit dataset 的正例所占比例不一样；二是 UCI 中的 Splice dataset 包含的正例和反例相同，但 actual DNA sequences the ratio 一般是 1:20，即 Splice dataset 与真实场景有很大的差别。

那么，所有这些与"非均等代价"又有什么特别的关系么？难道是非均等代价场景中，数据集的p、$cost_{+-}$和$cost_{-+}$这些参数更容易发生变化？而在均等代价场景中，p不怎么会发生变化（此时$cost_{+-}$和$cost_{-+}$本就相等）？若如此，均等价代价场景下，直接计算错误率就好；但在非均等代价场景下，利用式(2.23)计算出的代价敏感错误率（期望损失）会有较大误差，因为这里面使用的假反例率 FNR、假正例率 FPR、样例为正例的概率p（参见式(2.23)的解释）是根据测试集(test set)计算而得，但在真实使用时，真正的数据(deployed set)与此有较大差异，所以会有较大误差，其中 deployed set 参见代价曲线的原始文献[Drummond et al., MLJ06]对$p(+)$（就是此处的p）的解释时的含义，在图 2.5 的解释中亦有相应截图。基于此，图 2.5 中代价曲线对横轴的正例概率代价积分，则可得所有可能正例概率代价时归一化代价的和，即期望总体代价（阴影部分面积）。

注：以上四条解释参考了代价曲线原文献[Drummond, C., & Holte, R. C. (2006). Cost curves: An improved method for visualizing classifier performance. Machine Learning, 65(1), 95-130.]和知乎 https://www.zhihu.com/question/63492375。

2.4 比较检验

为什么要做比较检验？这在本节开篇的两段话已经交代的很清楚了。简单来说，就是通过 2.2 节的评估方法，得到每个算法/模型在 2.3 节各个性能度量上的表现，但这些值实际上是一个随机变量，因此并不能简单用比较大小来说明多个算法的优劣。

如果不做算法研究，不需要写论文对比多种算法的性能，本节可以暂时跳过；另外，本节有关检测变量的公式，例如式(2.30)、式(2.32)~式(2.36)，并不需要清楚是怎么来的（这是统计学家要做的事情），我们只需知道如何由测试结果计算出这些检测变量，并根据检测变量服从的分布类型得到临界值，然后将检测变量与临界值对比即可。

1、式(2.26)的解释

这个公式很容易理解，就是概率论中典型的有放回抽样问题。为了便于说明问题，这里将符号表达改一下：泛化错误率为$\varepsilon = \epsilon$的学习器，被测得测试错误率为$\hat{\varepsilon} = \hat{\epsilon}$的概率为：

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

即$\hat{\varepsilon}, \varepsilon$表示变量，$\hat{\epsilon}, \epsilon$表示变量值，其中

$$\binom{m}{\hat{\epsilon} \times m} = \frac{m!}{(\hat{\epsilon} \times m)!(m - \hat{\epsilon} \times m)!}$$

即中学时学的组合数，表示从包含m个元素的集合中不重复地抽取$\hat{\epsilon} \times m$个元素的可能取法，中学课本中记为$C_m^{\hat{\epsilon} \times m}$，有的资料中也记为$C(m, \hat{\epsilon} \times m)$。

以上公式中，若已知$\varepsilon = \epsilon$，求$\hat{\varepsilon}$为任意值时的条件概率，即后验概率$P(\hat{\varepsilon} \mid \varepsilon = \epsilon)$；反之，若已知$\hat{\varepsilon} = \hat{\epsilon}$，求其由$\varepsilon$为任意值时导致$\hat{\varepsilon} = \hat{\epsilon}$的概率，即似然概率$P(\hat{\varepsilon} = \hat{\epsilon} \mid \varepsilon)$。后验概率和似然概率区别在于：对于 P(果|因)，若已知“因”的取值则 P(果|因)为后验概率，即在知道“因”的取值后“果”发生的概率；反之若已知“果”的取值则 P(果|因)为似然概率，即当前的“果”更像是由哪个“因”所导致的。特别地，

$$\hat{\epsilon} = \arg\max_{\hat{\varepsilon}} P(\hat{\varepsilon} \mid \varepsilon = \epsilon)$$

表示最大后验概率（maximum a posteriori, MAP）估计，而

$$\epsilon = \arg\max_{\varepsilon} P(\hat{\varepsilon} = \hat{\epsilon} \mid \varepsilon)$$

表示最大似然估计（maximum likelihood estimation, MLE）。

若欲求最大似然估计，只需令$\partial P(\hat{\varepsilon} = \hat{\epsilon} \mid \varepsilon)/\partial\varepsilon = 0$，解得$\varepsilon = \hat{\epsilon}$，即泛化错误率$\varepsilon$等于测试错误率$\hat{\epsilon}$，其中

$$\begin{aligned}
\frac{\partial P(\hat{\varepsilon} = \hat{\epsilon} \mid \varepsilon)}{\partial \varepsilon} &= \binom{m}{\hat{\epsilon} \times m} \frac{\partial \varepsilon^{\hat{\epsilon} \times m} (1-\varepsilon)^{m - \hat{\epsilon} \times m}}{\partial \varepsilon} \\
&= \binom{m}{\hat{\epsilon} \times m} (\hat{\epsilon} \times m \times \varepsilon^{\hat{\epsilon} \times m - 1} (1-\varepsilon)^{m - \hat{\epsilon} \times m} + \varepsilon^{\hat{\epsilon} \times m} \times (m - \hat{\epsilon} \times m) \times (1-\varepsilon)^{m - \hat{\epsilon} \times m - 1} \times (-1)) \\
&= \binom{m}{\hat{\epsilon} \times m} \varepsilon^{\hat{\epsilon} \times m - 1} (1-\varepsilon)^{m - \hat{\epsilon} \times m - 1} (\hat{\epsilon} \times m \times (1-\varepsilon) - \varepsilon \times (m - \hat{\epsilon} \times m)) \\
&= \binom{m}{\hat{\epsilon} \times m} \varepsilon^{\hat{\epsilon} \times m - 1} (1-\varepsilon)^{m - \hat{\epsilon} \times m - 1} (\hat{\epsilon} \times m - \varepsilon \times m)
\end{aligned}$$

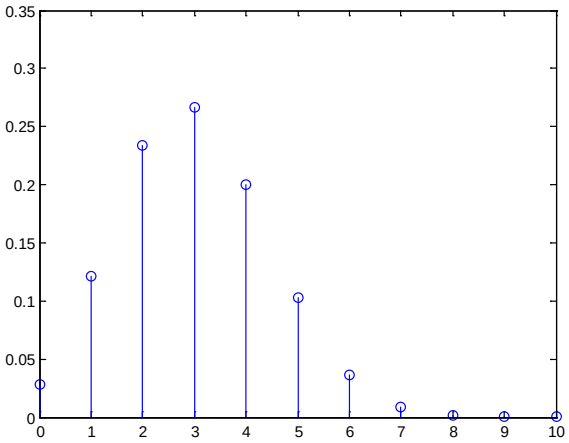
其中$C_m^{\hat{\epsilon} \times m}$为常数，$\varepsilon^{\hat{\epsilon} \times m - 1}$只有当$\varepsilon = 0$时等于 0，$(1-\varepsilon)^{m - \hat{\epsilon} \times m - 1}$只有当$\varepsilon = 1$时等于 0，所以只能令$(\hat{\epsilon} \times m - \varepsilon \times m) = 0$，解得$\varepsilon = \hat{\epsilon}$。

2、图 2.6 的 Matlab 绘制

图 2.6 画的是后验概率$P(\hat{\varepsilon} \mid \varepsilon = 0.3)$，可用 Matlab 如下代码绘出：

```
clear all;close all;
pp=zeros(11,1);
%画出泛化错误率为 epsilon_real 的学习器，被测得测试错误率为[0:m]/m 各自的概率
m = 10;%样本总数
epsilon_real = 0.3;
for k=0:m
    pp(k+1)=    nchoosek(m,k)*(epsilon_real^k) * ((1-epsilon_real)^(m-k));
end
```

figure;stem(0:m,pp)



列表如下：

$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$	$k=10$
0.028248	0.121061	0.233474	0.266828	0.200121	0.102919	0.036757	0.009002	0.0014470	0.000138	0.000006

各列分别表示当$\varepsilon = 0.3$时，10 个样本中测得 k 个样本被错误分类的概率。

3、式(2.27)的解释

首先，本式应该做如下勘误（即 max 改为 min，ϵ_0与ϵ互换，此条解释不再使用式(2.26)解释中的符号，勘误版次截止到 2018 年 12 月第 30 次印刷）

$$\bar{\epsilon} = \min \epsilon' \quad \text{s.t.} \quad \sum_{i=\epsilon' \times m+1}^{m} \binom{m}{i} \epsilon_0^i (1-\epsilon_0)^{m-i} < \alpha$$

因为符号ϵ接下来表示学习器的泛化错误率，因此这里换为了ϵ'；约束条件表示当学习器的泛化错误率$\epsilon = \epsilon_0$时，测得测试错误率$\hat{\epsilon} > \epsilon'$的概率需要小于$\alpha$，即$P(\hat{\epsilon} > \epsilon' \mid \epsilon = \epsilon_0) < \alpha$；式(2.27)求出满足该约束条件最小的$\epsilon'$，即为临界值$\bar{\epsilon}$；该临界值的意义在于，当$\epsilon = \epsilon_0$时，事件$\hat{\epsilon} > \bar{\epsilon}$出现的概率小于$\alpha$，也就是事件$\hat{\epsilon} \leq \bar{\epsilon}$出现的概率不小于$1-\alpha$。因为一般情况下设置$\alpha$的值很小(0.1 或 0.05)，因此可以认为若假设$\epsilon = \epsilon_0$成立，则事件$\hat{\epsilon} > \bar{\epsilon}$为小概率事件；若小概率事件发生了，则有理由怀疑最初的假设$\epsilon = \epsilon_0$是不正确的。补充一点，若临界值$\bar{\epsilon}$使得$P(\hat{\epsilon} > \bar{\epsilon} \mid \epsilon = \epsilon_0) < \alpha$成立，则对于任意$\epsilon_0' < \epsilon_0$也有$P(\hat{\epsilon} > \bar{\epsilon} \mid \epsilon = \epsilon_0') < \alpha$成立（简单理解，$\epsilon_0$越小，则图 2.6 面积越往左挤），进而可得$P(\hat{\epsilon} > \bar{\epsilon} \mid \epsilon \leq \epsilon_0) < \alpha$。临界值$\bar{\epsilon}$亦即图 2.6 中阴影部分面积小于$\alpha$时所对应的最小的横轴取值减 1（再除以样本个数规范化），也就是在 Matlab 中用$\text{icdf}(\text{'Binomial'}, 1-\alpha, m, \epsilon_0)$计算出的值。

其次，以此问题为背景专门解释一下什么是假设检验。考虑假设“$\epsilon \leq \epsilon_0$”（即要检验的假设），若测得测试错误率$\hat{\epsilon}$小于临界值$\bar{\epsilon}$（即观测到$\hat{\epsilon} < \bar{\epsilon}$，实际不大于即可，也就是$\hat{\epsilon} \leq \bar{\epsilon}$），则根据二项检验可得出结论：在$\alpha$的显著度下，假设“$\epsilon \leq \epsilon_0$”不能被拒绝，即能以$1-\alpha$的置信度认为，学习器的泛化错误率不大于$\epsilon_0$（即假设“$\epsilon \leq \epsilon_0$”基本上是猜对了）。

这里可能会有一个思维误区，对于第 39 页最上方的一段话：

此时若测试错误率 $\hat{\epsilon}$ 小于临界值 $\bar{\epsilon}$, 则根据二项检验可得出结论: 在 α 的显著度下, 假设“$\epsilon \leqslant \epsilon_0$”不能被拒绝, 即能以 $1-\alpha$ 的置信度认为, 学习器的泛化错误率不大于 ϵ_0; 否则该假设可被拒绝, 即在 α 的显著度下可认为学习器的泛化错误率大于 ϵ_0.

很容易将红线划出的部分理解为“只要观测到测试错误率$\hat{\epsilon}$小于临界值$\bar{\epsilon}$，则学习器泛化错误率不大于ϵ_0的概率至少为$1-\alpha$”，这是完全将假设检验本身的概念理解错了，混淆了当测得测试错误率$\hat{\epsilon}$小于临界值$\bar{\epsilon}$时，【事件$\epsilon \leq \epsilon_0$以不小于$1-\alpha$的概率成立】与【能以$1-\alpha$的置信度认为假设$\epsilon \leq \epsilon_0$成立】这两件事情；其中

(a)当测得测试错误率$\hat{\epsilon}$小于临界值$\bar{\epsilon}$时，【事件$\epsilon \leq \epsilon_0$以不小于$1-\alpha$的概率成立】翻译为数学符号为$P(\epsilon \leq \epsilon_0 \mid \hat{\epsilon} \leq \bar{\epsilon}) \geq 1-\alpha$；

(b)当测得测试错误率$\hat{\epsilon}$小于临界值$\bar{\epsilon}$时，【能以$1-\alpha$的置信度认为假设$\epsilon \leq \epsilon_0$成立】翻译为数学符号为$P(\hat{\epsilon} \leq \bar{\epsilon} \mid \epsilon \leq \epsilon_0) \geq 1-\alpha$；

可以看出这完全是两个问题，即$P(A \mid B)$与$P(B \mid A)$的关系。

最后，重复一遍此处的假设检验问题：

假设$\epsilon \leq \epsilon_0$成立（先假设好，是不是接受该假设后面再说），若$\hat{\epsilon} < \bar{\epsilon}$的概率不小于$1-\alpha$，则接受假设$\epsilon \leq \epsilon_0$，即若$P(\hat{\epsilon} \leq \bar{\epsilon} \mid \epsilon \leq \epsilon_0) \geq 1-\alpha$成立，则认为假设$\epsilon \leq \epsilon_0$猜对了！

有关假设检验的更多内容，请参考[盛骤, 谢式千, 潘承毅. 概率论与数理统计[M]. 4 版. 北京: 高等教育出版社, 2008.]第八章或其它相关教材。

本解释参考了知乎 https://www.zhihu.com/question/287895170。

3、式(2.28)到式(2.28)及图 2.7 的解释

先说一下式(2.27)有关方差的公式，有人认为应该除以样本数量k，有人认为应该除以样本数量减 1 即$k-1$。简单来说，根据总体样本集求方差时就除以总体样本数量，而根据抽样样本集求方差就除以抽样样本集数量减 1；总体样本集是真正想调查的对象集合，而抽样样本集是从总体样本集中被选出来的部分样本组成的集合，用来估计总体样本集的方差；一般来说，总体样本集是不可得的，我们拿到的都是抽样样本集。CSDN 博主 hearthougan 有篇博客《彻底理解样本方差为何除以 n-1》通过严格的数学推导证明了样本方差应该除以 n-1 才会得到总体样本的无偏估计，若除以 n 则得到的是有偏估计。

言归正传，这里要做的事情是：根据k个测试错误率$\hat{\epsilon}_1, \hat{\epsilon}_2, \ldots, \hat{\epsilon}_k$的平均值$\mu$和方差$\sigma^2$，判断在$\alpha$的显著度下，假设“$\mu = \epsilon_0$”是否成立。根据假设检验的含义，即求两个临界值，使得如下式子成立

$$P(\mu_\vee \leq \mu \leq \mu^\wedge \mid \mu = \epsilon_0) \geq 1-\alpha$$

也就是说如果观察到$\mu_\vee \leq \mu \leq \mu^\wedge$，则能以$1-\alpha$的置信度认为假设“$\mu = \epsilon_0$”成立，其中区间$[\mu_\vee, \mu^\wedge]$应该是包含$\epsilon_0$的取值范围。这是因为若$\mu = \epsilon_0$，则观测到的平均值$\mu$应该与$\epsilon_0$大小差不多；倘若观测到的$\mu$比$\epsilon_0$小很多（例如$\mu < \mu_\vee$）或比$\epsilon_0$大很多（例如$\mu > \mu^\wedge$），则认为假设“$\mu = \epsilon_0$”不成立（即当初假设错了）。式(2.30)对观测值μ进行了变换，其中由分子中的$\mu - \epsilon_0$项可以看出最终的检测变量τ_t位于 0 值附近。

最后说一个勘误，第 39 页最后一行开始写到：“若平均错误率μ与ϵ_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‘$\mu = \epsilon_0$’”，这里是式(2.30)的$\tau_t$服从于图 2.7 的 t 分布，而不是$\mu - \epsilon_0$更不是$|\mu - \epsilon_0|$，所以个人感觉这里应该改为“若$\tau_t$位于临界值范围$[t_{-\alpha/2}, t_{\alpha/2}]$内……”。

4、式(2.31)的解释

第一，该式就是对式(2.28)到式(2.28)的一个具体应用。

第二，注意 2.4.2 节标题下第二行“其中ϵ_i^A和ϵ_i^B是在相同的第i折训练/测试集上得到的

结果”，这一点在 2.2 节注解“交叉验证法的 Matlab 实现”中也特别强调了：“特别要注意的是，在对比不同算法的性能时，不同算法要使用相同的数据集划分，在留出法中也应该保证这一点”。

5、列联表(contingency table)的解释

第 41 页表 2.4 称为“列联表”，看论文时偶尔能遇到该术语（第 187 页第 2 行还会出现）。列联表是统计学中双变量相关分析时常见的概念。例如，现在要调查某医院各职业（医生、护士、行政人员）和性别（男、女）的相关关系，则需要先列出列联表：

		职业		
		医生	护士	行政人员
性别	男	a	b	c
	女	d	e	f

其中男医生人数为a，男护士人数为b，男行政人员人数为c，女医生人数为d，女护士人数为e，女行政人员人数为f，该医院总人数（或参与调查的人数）为$a+b+c+d+e+f$。有了列联表，就可以计算一些统计量，如克莱姆相关系数。

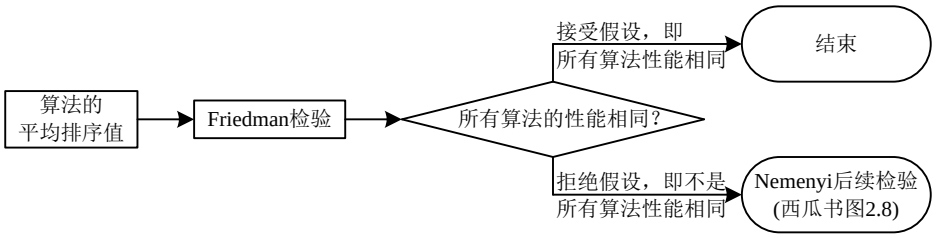
列联表又称交叉资料表、交互分类表等，可参见百度百科词条列联表。

6、Friedman 检验与 Nemenyi 后续检验的解释

很多时候基于诸如交叉验证 t 检验，在每个数据集上依次比较自己的算法与对比算法，发现自己的算法与对比算法在统计意义上并无显著差别，但是自己又会感觉自己的算法就是要好于对比算法，因为在每个数据集上的指标都略好于对比算法（虽然达不到统计意义上优于对方）。Friedman 检验基于算法排序比较各算法的性能，一定程度上可以克服这种情况。

Friedman 检验的假设是在所有数据集上“所有算法性能相同”，若假设被拒绝则说明算法的性能显著不同，此时就需要使用 Nemenyi 后续检验进一步比较哪些算法之间性能有显著不同。Nemenyi 后续检验结果一般会画成图 2.8 的形式，可以直观地看出各算法性能之间的关系。

检验流程如下：



其实算法两两之间也可以基于算法排序进行统计检验，这就是威尔科克森符号秩检验(Wilcoxon Signed-Ranks Test)，详见[Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1), 1-30.]的第 3.1.3 节。

实际上，paired t-test 等直接基于指标数值的检验称为 parametric tests，而 Friedman 检验以及 Wilcoxon 检验等基于排序的检验称为 non-parametric tests。

注意到书中介绍的 Nemenyi 后续检验针对的是把所有算法两两对比(when all classifiers are compared to each other)，而在实际中我们需要的是将自己提出的算法与对比算法比较即可(when all classifiers are compared with a control classifier)，这时就要使用 Bonferroni-Dunn 检验而不是 Nemenyi 检验了，具体流程详见文章[Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1), 1-30.]的第 3.2.2 节，其实就是将式(2.36)中的q_α改变一下，即由论文中的 Table 5(a)换为 Table 5(b)。

7、Friedman 检验 Matlab 实现

主要分两步，第一是根据性能结果得到类似表 2.5 的算法比较序值表，第二是根据这个表的数据结合式(2.34)和式(2.35)计算检验变量τ_F。

第一步由如下函数实现(Friedman_mtx_rank.m)：

```
function [ r ] = Friedman_mtx_rank( s,order )
% Author: https://blog.csdn.net/jbb0523
% Version1.0@2019-05-06
%s: k x N matrix,
%     where k is the number of classifiers, N is the number of data sets
%order: sort in descending/ascending order, specifically
%            'descend', the greater the better, i.e. greater value rank 1st (such as accuracy)
%            'ascend', the smallest the better, i.e. smallest value rank 1st (such as error rate)
    [k,N] = size(s);
    [S,r] = sort(s,order);
    idx = k*repmat(0:N-1, k, 1)' + r';
    R = repmat(1:k, N, 1);
    S = S';

    for i=1:N
        for j=1:k
            index = (S(i,j) == S(i,:));
            R(i,index) = mean(R(i,index));
        end
    end
    r(idx) = R;
end
```

第二步由如下函数实现(Friedman_test.m)：

```
function [ tao_F, critical_val ] = Friedman_test( Ranks,alpha,verbose )
% Author: https://blog.csdn.net/jbb0523
% Version1.0@2019-05-06
% Ranks: k x N matrix, ranking matrix,
%     where k is the number of classifiers, N is the number of data sets
    if nargin<2
        alpha = 0.05;%set significance level as 0.05 in default
    end
    if nargin<3
        verbose = 0;%no outputs in default
    end
    Ranks = Ranks';
    r = mean(Ranks);
    [N,k] = size(Ranks);
    tao_gamma_2 = 12*N/k/(k+1)*(sum(r.^2)-k*(k+1)^2/4);%Eq.(2.34) in watermelon book
    tao_F = (N-1)*tao_gamma_2/(N*(k-1)-tao_gamma_2);%Eq.(2.35) in watermelon book
    critical_val = icdf('F',1-alpha,k-1,(k-1)*(N-1));
```

```
    if verbose
        disp(['tao_F = ',num2str(tao_F),' while critical_val = ',num2str(critical_val)]);
    end
end
```

有了以上两个函数之后，设包含了k个分类器在N个数据集上的分类精度结果存在大小为$k \times N$矩阵 accuracy_mtx 中（即存储形式为表 2.5 的转置），则以下三行可以实现 Friedman 检验过程（因为精度越大越好，即最大的精度应该排序为 1，所以第 1 行参数设为'descend'）：

```
Ranks = Friedman_mtx_rank(accuracy_mtx,'descend');
[tao_F,critical_val] = Friedman_test(Ranks);
disp(['Friedman test for Acc: tao_F = ', num2str(tao_F), ' while critical_val = ', num2str(critical_val)]);
```

若发现检测结果比临界值大，则拒绝假设“所有算法性能相同”，继续进行后续检验。

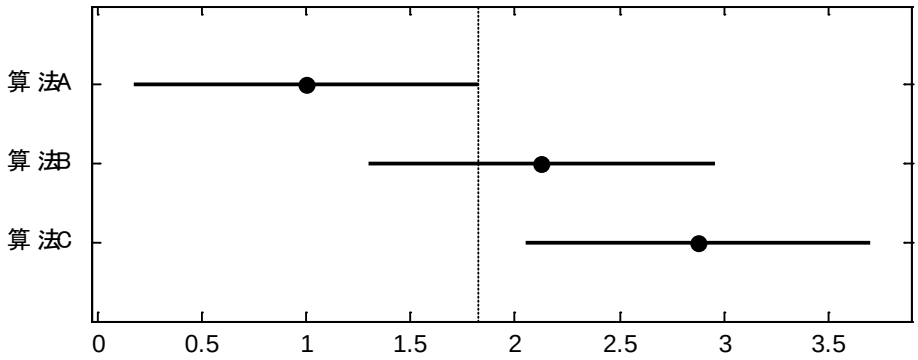
8、图 2.8 的 Matlab 绘制

听说有些同学还会赤膊上阵，在 Visio 等软件中手绘类似图 2.8 的检验图（很惊讶），且不说费时费力，准确性也无法保证。下面给出一段参考代码，可以在 Matlab 中画出图 2.8：

```
%  绘制西瓜书 P44 的图 2.8Friedman 检验图
% Author: https://blog.csdn.net/jbb0523
clear all;close all;clc;
%%  输入参数
ranks = [1 2.125 2.875];%平均序值
N = 4;%数据集个数
k = 3;%算法个数
q_a = 2.344;%当 k=3,alpha=0.05 时表 2.7 中的 q_a 值
algo_name = cell(1,k);
algo_name{1} = '算法 C';algo_name{2} = '算法 B';algo_name{3} = '算法 A';%逆序
%%  绘图
CD = q_a*sqrt(k*(k+1)/6/N);%根据式(2.36)计算临界值域
y_algo = (k:-1:1);%各算法的纵坐标

%绘出各算法的圆点
plot(ranks,y_algo,'k.','MarkerSize',20);hold on;

%画出各算法的横线段及算法 A 右侧的虚线
for i=1:k
    tmp_x_line = [ranks(i)-CD/2,ranks(i)+CD/2];%横坐标
    tmp_y_line = [y_algo(i),y_algo(i)];%纵坐标
    plot(tmp_x_line,tmp_y_line,'k-','linewidth',2);hold on;
    %画算法 A 右侧的虚线
    if i==1%ranks 中的第 1 个值对应算法 A
        x_dotted_line = [ranks(i)+CD/2,ranks(i)+CD/2];%横坐标
        y_dotted_line = [0,k+1];%纵坐标
        plot(x_dotted_line,y_dotted_line,'k:');hold on;
```

```
    end
end

%设置坐轴标显示
xlim([ranks(1)-CD/2-0.2,ranks(k)+CD/2+0.2]);%横坐标轴显示范围
ylim([0,k+1]);%纵坐标轴显示范围
%set(gca,'xtick',1:k);
set(gca,'ytick',1:k);
set(gca,'yticklabel',algo_name);
hold off;
```

其中开头的**平均序值向量** ranks 可基于上一条解释中 Friedman_mtx_rank 函数直接由性能结果矩阵而得，即 ranks=mean(Ranks')，其中 Ranks 是 Friedman_mtx_rank 输出的**序值矩阵**，这里的转置是因为 Friedman_mtx_rank 函数的输出是$k \times N$矩阵（k为分类器个数，N为数据集个数），而 mean 对矩阵求均值时按每列求平均，而我们要得到某个算法在N个数据集上的平均排名，所以要转置，若感觉烦琐可以修改 Friedman_mtx_rank 设置。程序运行结果如下：



文章[Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1), 1-30.]的第 3.2.4 节提出了另一种展示检验结果的可视化方法，即文中的 Figure 2，具体参见原论文，也可以进一步思考如何编程绘制。

2.5 偏差与方差

不考虑噪声，偏差很大可以认为是由模型欠拟合引起的，方差很大可以认为是由模型过拟合引起的。

1、数据集中的标记与真实标记

边注中已经提到，有可能出现噪声使得$y_D \neq y$。例如由众包（crowdsourcing）得到的标记就有可能是错的，详见作者的综述文章[Zhou, Z. H. . (2018). A brief introduction to weakly supervised learning. National Science Review, 5(01), 48-57.]的最后一部分内容“Inaccurate Supervision”。

2、式(2.37)到式(2.42)的推导

首先，梳理一下书中的符号：对测试样本$\boldsymbol{x}$，令y_D为$\boldsymbol{x}$在数据集中的标记，y为$\boldsymbol{x}$的真实标记，$f(\boldsymbol{x};D)$为训练集D上学得模型f在$\boldsymbol{x}$上的预测输出。

估计作者写这些符号时也有些矛盾：$\boldsymbol{x}$为测试样本，所以应该不在训练集D中，但刚刚提到“令y_D为$\boldsymbol{x}$在数据集中的标记”，那么y_D为$\boldsymbol{x}$在哪个数据集中的标记呢？所以，个人感觉这里就简单认为$\boldsymbol{x}$属于训练集D。另外，设有n个训练集$D_1,\ldots,D_n$，这n个训练集都是以独立同分布方式从样本空间中采样而得，并且恰好都包含测试样本$\boldsymbol{x}$，该样本在这n个训练集的标记分别为$y_{D_1},\ldots,y_{D_n}$。书中已明确，此处以回归任务为例，即标记为实值。

针对式(2.37)，可简单理解如下：

$$\bar{f}(\boldsymbol{x})=\mathbb{E}_D[f(\boldsymbol{x};D)]=\frac{1}{n}(f(\boldsymbol{x};D_1)+\ldots+f(\boldsymbol{x};D_n))$$

针对式(2.38) ，可简单理解如下：

$$\begin{aligned}var(\boldsymbol{x})=&\mathbb{E}_D[\left(f(\boldsymbol{x};D)-\bar{f}(\boldsymbol{x})\right)^2]\\=&\frac{1}{n}\left(\left(f(\boldsymbol{x};D_1)-\bar{f}(\boldsymbol{x})\right)^2+\ldots+\left(f(\boldsymbol{x};D_n)-\bar{f}(\boldsymbol{x})\right)^2\right)\end{aligned}$$

针对式(2.39) ，可简单理解如下：

$$\varepsilon^2=\mathbb{E}_D[(y_D-y)^2]=\frac{1}{n}\left((y_{D_1}-y)^2+\ldots+(y_{D_n}-y)^2\right)$$

有了以上基础，接下来推导最复杂的式(2.41)：

第 1 个等号：这是期望泛化误差的定义式，类似地，可直观地写为

$$\mathbb{E}_D[(f(\boldsymbol{x};D)-y_D)^2]=\frac{1}{n}\left((f(\boldsymbol{x};D_1)-y_{D_1})^2+\ldots+(f(\boldsymbol{x};D_n)-y_{D_n})^2\right)$$

我们只能使用数据集中的标记y_D评估学习器泛化性能，因为真实标记y未知；

第 2 个等号：常用的配项技巧，减去$\bar{f}(\boldsymbol{x})$再加上$\bar{f}(\boldsymbol{x})$，相当于没作任何变化；

第 3 个等号：令$a=f(\boldsymbol{x};D)-\bar{f}(\boldsymbol{x}),b=\bar{f}(\boldsymbol{x})-y_D$，则$(a+b)^2=a^2+b^2+2ab$，再结合数学期望的性质$\mathbb{E}[X+Y]=\mathbb{E}[X]+\mathbb{E}[Y]$即可；

第 4 个等号只须证明第 3 个等号最后一项等于 0；首先

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

这就是使用了数学期望的性质$\mathbb{E}[X+Y]=\mathbb{E}[X]+\mathbb{E}[Y]$；对于第 1 项

$$\begin{aligned}\mathbb{E}_D\left[2\left(f(\boldsymbol{x};D)-\bar{f}(\boldsymbol{x})\right)\bar{f}(\boldsymbol{x})\right]=&\mathbb{E}_D\left[2f(\boldsymbol{x};D)\bar{f}(\boldsymbol{x})-2\bar{f}(\boldsymbol{x})\bar{f}(\boldsymbol{x})\right]\\=&2\mathbb{E}_D\left[f(\boldsymbol{x};D)\right]\bar{f}(\boldsymbol{x})-2\bar{f}(\boldsymbol{x})\bar{f}(\boldsymbol{x})\\=&2\bar{f}(\boldsymbol{x})\bar{f}(\boldsymbol{x})-2\bar{f}(\boldsymbol{x})\bar{f}(\boldsymbol{x})\\=&0\end{aligned}$$

其中第 1 个等号就是乘开，把式子的扩号去掉；第 2 个等号是因为期望$\bar{f}(\boldsymbol{x})$为常量（式(2.37)），因此$\mathbb{E}_D[\cdot]$运算不起作用；第 3 个等号使用了$\bar{f}(\boldsymbol{x})$的定义（式(2.37)）；对于第 2 项

$$\begin{aligned}\mathbb{E}_D\left[2\left(f(\boldsymbol{x};D)-\bar{f}(\boldsymbol{x})\right)y_D\right]=&\mathbb{E}_D\left[2f(\boldsymbol{x};D)y_D-2\bar{f}(\boldsymbol{x})y_D\right]\\=&\mathbb{E}_D\left[2f(\boldsymbol{x};D)y_D\right]-\mathbb{E}_D\left[2\bar{f}(\boldsymbol{x})y_D\right]\\=&2\mathbb{E}_D\left[f(\boldsymbol{x};D)\right]\mathbb{E}_D\left[y_D\right]-2\bar{f}(\boldsymbol{x})\mathbb{E}_D\left[y_D\right]\\=&2\bar{f}(\boldsymbol{x})\mathbb{E}_D\left[y_D\right]-2\bar{f}(\boldsymbol{x})\mathbb{E}_D\left[y_D\right]\\=&0\end{aligned}$$

其中第 1 个等号就是乘开，把式子的扩号去掉；第 2 个等号根据$\mathbb{E}[X+Y]=\mathbb{E}[X]+\mathbb{E}[Y]$；第 3 个等号中第 1 项$\mathbb{E}_D\left[2f(\boldsymbol{x};D)y_D\right]=2\mathbb{E}_D\left[f(\boldsymbol{x};D)\right]\mathbb{E}_D\left[y_D\right]$根据数学期望的性质“当随机变量$X$和$Y$相互独立时，$\mathbb{E}[XY]=\mathbb{E}[X]\mathbb{E}[Y]$”，而此处正如边注中所说“考虑到噪声不依赖于$f$”，即$f(\boldsymbol{x};D)$与$y_D$相互独立（$y_D$等于$y$加上噪声，$y$为常量）；第 3 个等号中第 2

项是因为期望$\bar{f}(\boldsymbol{x})$为常量；第 4 个等号使用了$\bar{f}(\boldsymbol{x})$的定义（式(2.37)）；

综上所述：式(2.41)中第 3 个等号最后一项等于 0，代入即得第 4 个等号；

第 5 个等号类似于第 2 个等号，对第 2 项使用了配项技巧；

第 6 个等号类似于第 3 个等号，对第 2 项使用了性质$\mathbb{E}[X+Y]=\mathbb{E}[X]+\mathbb{E}[Y]$；

第 7 个等号只须证明第 6 个等号最后一项等于 0；由于$\bar{f}(\boldsymbol{x})$和y均为常量，因此

$$\begin{aligned}2\mathbb{E}_D\left[\left(\bar{f}(\boldsymbol{x})-y\right)\left(y-y_D\right)\right] &=2\left(\bar{f}(\boldsymbol{x})-y\right)\mathbb{E}_D\left[\left(y-y_D\right)\right]\\ &=2\left(\bar{f}(\boldsymbol{x})-y\right)\left(y-\mathbb{E}_D\left[y_D\right]\right)\\ &=2\left(\bar{f}(\boldsymbol{x})-y\right)\left(y-y\right)\\ &=0\end{aligned}$$

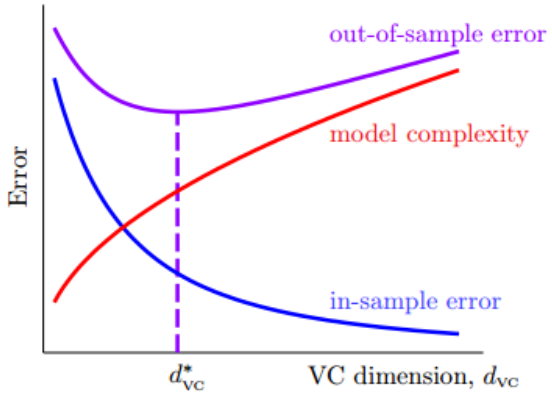
上式第 3 个等号使用了等式$\mathbb{E}_D[y_D]=y$，这是由于在式(2.40)下方提到：为便于讨论，假定噪声期望为零，即$\mathbb{E}_D[y_D-y]=\mathbb{E}_D[y_D]-y=0$，其中$y$为$\boldsymbol{x}$的真实标记，是一个常量；因此有$\mathbb{E}_D[y_D]=y$。到此为止，式(2.41)推导结束。

针对式(2.42)，就是将式(2.38)、式(2.39)和式(2.40)代入到式(2.41)最后的表达式即可。

接下来摘抄书中第 45~46 页有关偏差、方差、噪声的含义：偏差(2.40)度量了学习算法的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(2.38)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(2.39)则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。

3、图 2.9 的解释

图 2.9 中横坐标表示的训练程度可以理解为模型的复杂程度。模型越简单（比如直线）其表达能力越弱，反之模型越复杂（比如 n 阶多项式）其表达能力越强。类似于图 2.9 的关系还有下图所示的训练误差(i.e., in-sample error)、模型复杂度(i.e., model complexity)和泛化误差(i.e., out-of-sample error)三者随着 VC 维（参见 12.4 节）的变化趋势关系（摘自林轩田《机器学习基石》第 7 讲）：



实际上，上图与图 2.9 表示的问题本来也就是一回事……

2.6 本章小节

对于本章内容来说，一般需要懂的很容易懂，一般看不懂的也不需要懂。

需要懂且很容易懂的包括错误率（和精度）、过拟合、欠拟合的概念，以及 2.2 节介绍的评估方法。

接下来的内容，比如对于做信息检索的人来说，需要懂的查准率和查全率其实也很容易

懂，只是当把查准率、查全率、真正例率、假正例率、真反例率、假反例率等放到一起时可能就有点懵了，其实理解这些概念的含义就容易区分了。

查准率就是预测为正例的样本中有多大比例预测正确了，查全率就是数据集中的正例有多少比例被预测为正例（即正例中预测正确的比例），真正例率就是查全率，假正例率就是数据集中的反例有多少比例被预测为正例（即反例中预测错误的比例），真反例率就是数据集中的反例有多少比例被预测为反例（即反例中预测正确的比例），假反例率就是数据集中的正例有多少比例被预测为反例（即正例中预测错误的比例）。

不同于以上分类指标（必须得到 0/1 这样的二值预测结果才能计算），AUC 是排序指标（预测结果需要为类似于概率预测值时才能计算，概率预测值再配以一个门限，就得到了 0/1 二值预测），表示从数据集中任取一对正例和反例，正例预测值大于反例预测值的概率。

F1 与错误率、AUC 等指标的侧重有所不同，例如对于类别不平衡问题（第 3.6 节）F1 指标相比于错误率、AUC 可能更有参考意义。有时会发现自己的算法在错误率（第 2.3.1 节）等分类指标和 AUC（第 2.3.3 节）等排序指标上都很好，唯独在 F1 上的结果不尽如人意，原因可能与此有关。

代价曲线与 ROC 曲线紧密相关，代价曲线的作者在论文中也反复与 ROC 做对比，若真想理解代价曲线，还是把原论文仔细读一读吧。代价曲线 2006 年发表在 MLJ 上，相对于教材来说应该属于新内容了吧。从这一点可以看出学术大牛写的教材和普通人写的教材还是有本质区别的：一般人写教材通常就是找几本经典教材，然后根据大纲要求的授课内容合理选择保留哪些知识点、按自己讲课思路调整一下内容顺序，然后对内容换一种表达方法，增删一些习题，好一些的还会加一点自己的理解，当然很多时候添加的理解还可能是对知识点的过拟合；而学术大牛写教材可以把这个学科前沿研究中沉淀下来的经典内容逐步放到教材中；所以，真正的教学与科研相结合是需要真才实料的。另外，同样是写了一本书，编、编著、著那是真的有本质区别的。

比较检验基于概率论与数理统计课程（其实是数理统计部分）中的假设检验，2.4.1 节通过一个具体的学习器泛化错误率的例子说明什么是假设检验；式(2.27)有笔误（截止到 2018 年 12 月第 30 次印刷），若读者对假设检验概念理解很深的话倒也很容易看出来，但若本就对假设检验理解不透彻，很容易陷入思维误区；其实核心问题就是要知道，假设检验中的置信度 1-α 是指，在待检验的假设成立的条件下，观测变量在临界值范围内的后验概率（一定注意这里是后验概率，而不是似然概率）。然后接下来介绍的几种检验在论文中都可以见到；本节内容会用就好，不要对每个公式都刨根问底儿，有时不求甚解也并没有什么坏处。

偏差方差分解重在理解其含义，实际中你又不能真地把偏差和方差求出来，能够大概判断什么时候模型过拟合了（可能需要继续收集更多的数据），什么时候模型欠拟合（应该使用更复杂的模型），也就够了。