1. Threshold-based metrics

(1) Accuracy, Precision, Recall, F1-score

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

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

//模型给出的阳性，有多少是真阳?

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

//阳性有多少被找了出来?

$$F1 = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$$

//二者的调和平均数

TP, TN, FP, FN 构成了混淆矩阵。

Precision 和 Recall是”此消彼长“的关系，所以需要F1-score来综合二者。事实上，Precision-Recall构成了P-R曲线：

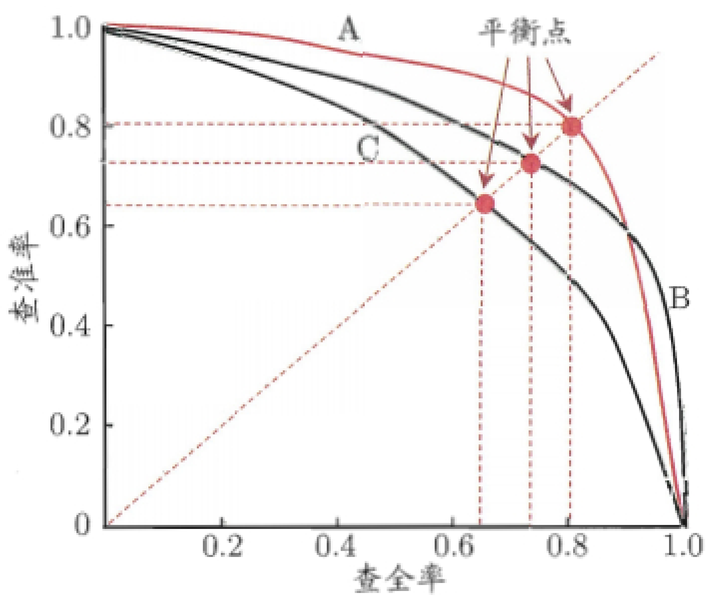


图 2.3 P-R曲线与平衡点示意图

(2) ROC曲线和AUC

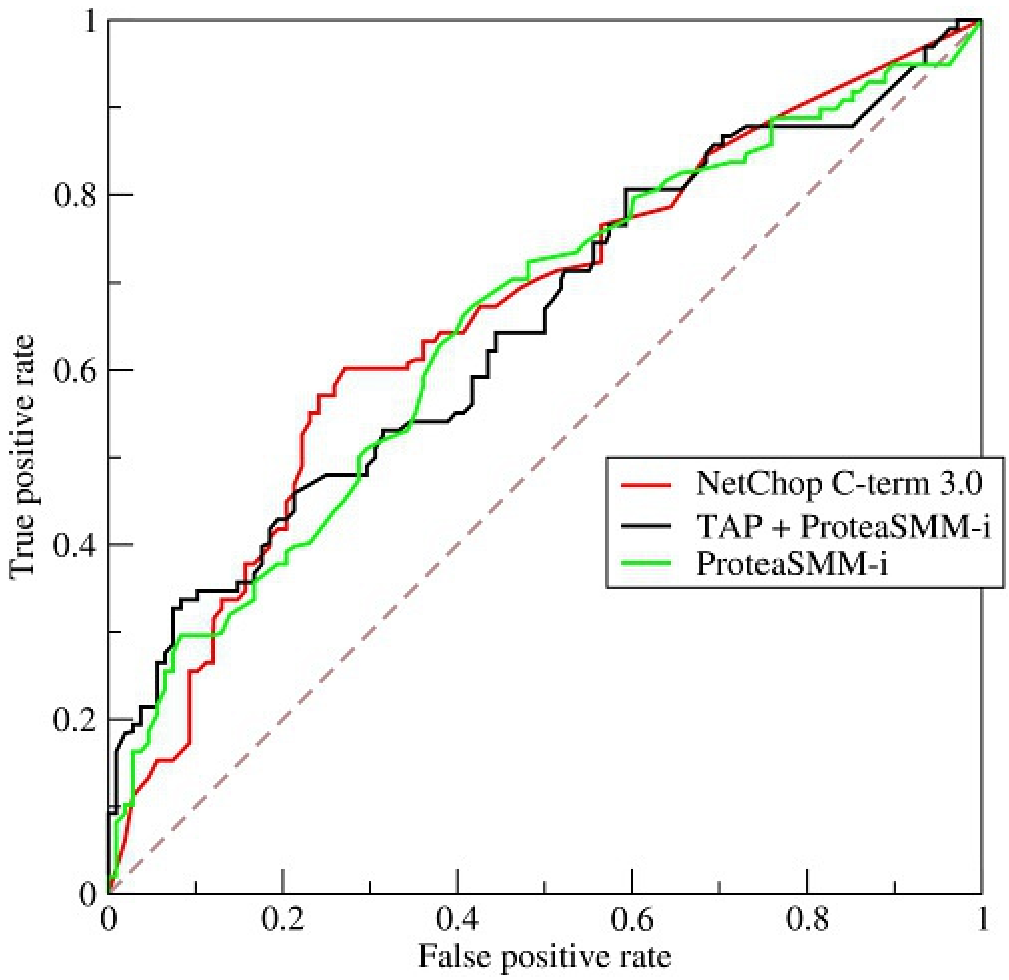
TPR(True Positive Rate):

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

, 就是recall

FPR(False Positive Rate):

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



AUC是工业界评判CTR预估的指标。AUC（Area Under Curve）被定义为ROC曲线下的面积（ROC曲线是通过改变判别器的判别threshold得到的）。又由于ROC曲线一般都处于y=x这条直线的上方，所以AUC的取值范围在0.5和1之间。使用AUC值作为评价标准是因为很多时候ROC曲线并不能清晰的说明哪个分类器的效果更好，而作为一个数值，对应AUC更大的分类器效果更好。

【手撕AUC】

首先要明白AUC的物理含义不仅是ROC曲线下的面积，AUC还有另外一个**物理含义**就是：给定正样本M个，负样本N个，以及他们的预测概率，那么AUC的含义就是所有**穷举所有的正负样本对**，如果正样本的预测概率大于负样本的预测概率，那么就+1；如果如果正样本的预测概率等于负样本的预测概率，那么就＋0.5， 如果正样本的预测概率小于负样本的预测概率，那么就+0；最后把统计处理的个数除以M×N就得到AUC.

```
def AUC(label,pre):
    pos = []  ## 正样本index
    neg = []  ## 负样本index
```

```python
    for i in range(len(label)):
        if(label[i] == 1):
            pos.append(i)
        else:
            neg.append(i)
    cnt = 0
    for i in pos:
        for j in neg:
            if(pre[i]>pre[j]):
                cnt += 1
            if(pre[i] == pre[j]):
                cnt += 0.5
            else:
                cnt += 0
    return cnt / (len(pos)*len(neg))
```

【面试真题：AUC为啥对正负样本比例不敏感？】

$$\mathrm{TPR}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$$
$$\mathrm{FPR}=\frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$$

横轴FPR只关注负样本，与正样本无关；纵轴TPR只关注正样本，与负样本无关。所以横纵轴都不受正负样本比例影响，积分当然也不受其影响。

具体来说，假如我们随机删掉一半个正样本。对于FPR，其分母是所有真的负样本的个数，这个自然是个定值；其分子是被当成正例的负例(false negative),也不受影响，所以FPR不变。对于TPR，其分母是所有真的正样本的个数，此时变成原来的1/2；其分子是所有预测正确的正样本个数，如果我们是随机删除的正样本的话，那么此时也变为原来的1/2，因此TPR也不变。所以积分不受影响。

而PR曲线就没有AUC这种对正负样本比例不敏感的好特性。具体来说，recall就是TPR，所以当我们随机删掉一半正样本的时候，recall不会变化。但是precision就没那么幸运了。

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

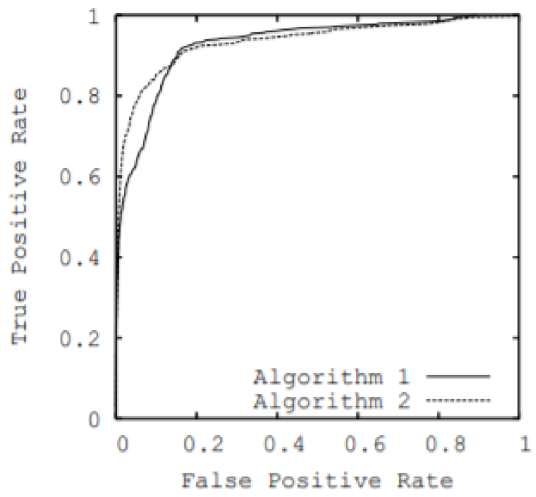
TP还是会减少一半，FP却保持不变，因此总体precision会变小。

【AUC的优势】

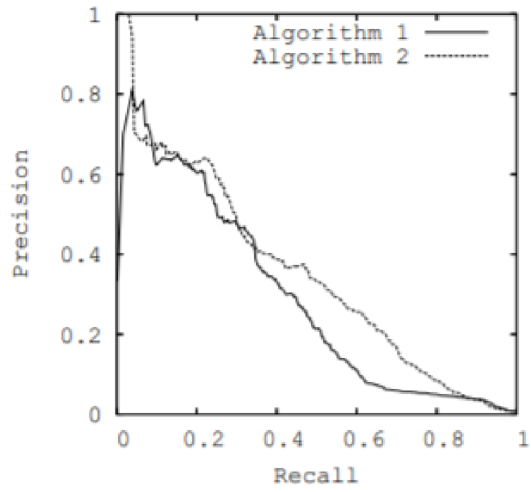
AUC的计算方法**同时考虑了分类器对于正例和负例的分类能力，在样本不平衡的情况下，依然能够对分类器作出合理的评价。** 例如在反欺诈场景，设欺诈类样本为正例，正例占比很少（假设0.1%），如果使用准确率评估，把所有的样本预测为负例，便可以获得**99.9%的准确率**。 但是如果使用AUC，把所有样本预测为负例，TPRate和FPRate同时为0（没有Positive），与(0,0) (1,1)连接，得出**AUC仅为0.5**，成功规避了样本不均匀带来的问题。

【ROC和PR曲线比较】

- ROC可能对类别不平衡的数据集过于"乐观"。例如下图是一个对highly-skewed数据集画出的ROC和PR曲线，可以看出，ROC曲线已经非常接近左上角，似乎已经是一个很好的算法；但是PR曲线却并没有很接近右上角，似乎还有很大的优化空间。另外，ROC中两个算法表现差不多；但是PR曲线中算法2明显优于算法1.



(a) Comparison in ROC space



(b) Comparison in PR space

这是为什么呢？是因为这个数据集里负样本特别的多。FPR的分母就是负样本个数，分子是FP(被分为正例的负样本)。由于分母特别大，所以即使分子很大，这个FPR也是很小的。

- ROC曲线上的点和PR曲线上的点是一一对应的。这是因为在ROC曲线上的每个点都对应唯一的混淆矩阵，这样自然对应PR曲线上唯一的点；对于PR曲线，只要Recall!=0,我们也都可恢复出唯一的混淆矩阵。

【GAUC (grouped AUC)】

AUC在传统的机器学习二分类中还是很能打的，但是在搜推领域如果只是使用AUC会带来问题。这是因为传统的AUC可以评判二分类，但是推荐领域要算的是对于**每个人的二分类**结果。上文说到，AUC要惩罚那些所有得分比正样本还大的那些负样本，对于一个用户自然可以这样去比较；但是不同用户的得分是不可比的。

> auc反映的是整体样本间的一个排序能力，而在计算广告领域，我们实际要衡量的是不同用户对不同广告之间的排序能力， 实际更关注的是同一个用户对不同广告间的排序能力，为此，参考了阿里妈妈团队之前有使用的group auc的评价指标。group auc实际是计算每个用户的auc，然后加权平均，最后得到group auc，这样就能减少不同用户间的排序结果不太好比较这一影响。

$$GAUC = \frac{\sum_{(u,p)} w_{(u,p)} * AUC_{(u,p)}}{\sum_{(u,p)} w_{(u,p)}}$$

实际处理时权重一般可以设为**每个用户view或click的次数**，而且**会过滤掉单个用户全是正样本或负样本的情况。**

但是实际上一般还是主要看auc这个指标，但是当发现auc不能很好的反映模型的好坏(比如auc增加了很多，实际效果却变差了)，这时候可以看一下gauc这个指标。

【灵敏度和特异性】

- **TPR**：true positive rate，真阳性样本在实际阳性样本中的占比，又称为灵敏度。 计算公式为：TPR=TP/(TP+ FN)
- **TNR**：true negative rate，真阴性样本在实际阴性样本中的占比，又称为特异度。

（3）多分类问题

1) Macro F1： 宏平均

Macro 算法在计算 Precision 与 Recall 时是先分别计算每个类别的Precision 与 Recall， 然后再进行平均。

$$Macro_{F1-score} = \frac{1}{N}\sum_{i=0}^{N} F1-score_i$$

其中，N为类别数。

Macro F1 本质上是所有类别的统计指标的算术平均值来求得的，这样单纯的平均忽略了**样本之间分布可能存在极大不平衡的情况。**

2) Micro F1 ： 微平均

Micro 算法在计算 Precision 与 Recall 时会将所有类直接放到一起来计算。

$$\text{Precision}_{micro} = \frac{\sum_{i=1}^{L} TP}{\sum_{i=1}^{L} TP + \sum_{i=1}^{L} FP}$$

$$\text{Recall}micro = \frac{\sum_{i=1}^{L} TP}{\sum_{i=1}^{L} TP + \sum_{i=1}^{L} FN}$$

$$\text{Micro F1} = \frac{2 \cdot \text{Precision}_{micro} \cdot \text{Recall}_{micro}}{\text{Precision}_{micro} + \text{Recall}_{micro}}$$

Macro vs Micro

Macro 相对 Micro 而言，**小类别起到的作用更大。**考虑到实际的环境中，这种指标明显是有问题的，因为小类别起到的作用太大。 而对于 Micro 来说，其考虑到了这种样本不均衡的问题， 因此在这种情况下相对较佳。

总的来说， 如果你的类别比较均衡，则随便； 如果你认为大样本的类别应该占据更重要的位置， 使用Micro； 如果你认为小样本也应该占据重要的位置，则使用 Macro。

为了解决 Macro 无法衡量样本均衡问题，一个很好的方法是求加权的 Macro， 因此 Weighted F1 出现了。

3). Weight F1

Weighted 算法算术 Macro 算法的改良版，是为了解决Macro中没有考虑样本不均衡的原因， 在计算 Precision与 Recall 时候，各个类别的 Precision 与 Recall要乘以**该类在总样本中的占比**来求和。

2. Ranking-based Metrics：加入位置信息

(1) Hit Rate

在top-K推荐中，HR是一种常用的衡量召回率的指标，计算公式为：

$$HR@k = \frac{Hit@k}{|trueset|}$$

例如，三个用户购买的商品个数分别是10,12,8, 模型得到的top-10推荐列表中，分别有6个，5个，4个在测试集中，那么此时HR@10的值是 (6+5+4)/(10+12+8) = 0.5。

（2）MRR(Mean Reciprocal Rank, 平均倒数排名)

把标准答案在被评价系统给出结果中的排序取倒数作为它的准确度，再对所有的问题取平均。例如有3个query如下图所示：

Query	Results	Correct response	Rank	Reciprocal rank
cat	catten, cati, **cats**	cats	3	1/3
torus	torii, **tori**, toruses	tori	2	1/2
virus	**viruses**, virii, viri	viruses	1	1

(黑体为返回结果中最匹配的一项)

这个系统的MRR值为：(1/3 + 1/2 + 1)/3 = 11/18=0.61。

（3）MAP（Mean Average Precision）

MAP（Mean Average Precision）是信息检索/推荐领域用以衡量搜索/推荐引擎的排序性能的评价指标。例如对于【命中，命中，未命中，未命中，未命中】和【未命中，未命中，未命中，命中，命中】这两个top-5的推荐列表，虽然他们的precision都是2/5，但是显然第一个推荐列表的性能要高于第二个推荐列表，因为其在第1、2位就已命中。

MAP可以由它的三个部分来理解：P, AP, MAP

i) P: precision

正确率只是考虑了返回结果中相关文档的个数，没有考虑文档之间的序。对一个搜索引擎或推荐系统而言返回的结果必然是有序的，而且越相关的文档排的越靠前越好，于是有了AP的概念。

ii). AP: average precision

对一个有序的列表，计算AP的时候要先求出每个位置上的precision，然后对所有的位置的precision再做个average。如果该位置的文档是不相关的则该位置 precision=0.

对于推荐列表【命中，命中，未命中，未命中，未命中】(假设用户实际购买了3种商品，而我们的推荐系统在前5个只给出了2个):

ID	Correctness	Score
1	命中，1	1
2	命中，1	1
3	未命中，0	2/3
4	未命中，0	1/2
5	未命中，0	2/5

Score行代表到目前为止的准确率。

$$AP@5=\frac{1}{\min(3,5)}\left(1\cdot 1+1\cdot 1+\frac{2}{3}\cdot 0+\frac{2}{4}\cdot 0+\frac{2}{5}\cdot 0\right)=\frac{2}{3}$$

用户实际点了3个

搜索召回了5个

iii) MAP

MAP(Mean Average Precision)，即为所有query的AP取均值.

(4) Normalized Discounted Cummulative Gain (NDCG)

归一化折损累计增益(NDCG) 通常用来评价搜索/推荐算法的好坏。

i. 累计增益（CG）

CG，cumulative gain，是DCG的前身，只考虑到了相关性的关联程度，没有考虑到位置的因素。它是一个搜素结果相关性分数的总和。指定位置p上的CG为：

$$\mathrm{CG_p}=\sum_{i=1}^{p} rel_i$$

$$rel_i$$

代表i这个位置上的相关度。

假设搜索一个query，最理想的结果是: doc1、doc2、 doc3。而出现的结果是 doc3、doc1、doc2的话，CG的值是没有变化的，因此需要下面的DCG。

ii. 折损累计增益（DCG）

DCG， Discounted 的CG，就是在每一个CG的结果上除以一个折损值，为什么要这么做呢？目的就是为了让排名越靠前的结果越能影响最后的结果。假设排序越往后，价值越低。到第i个位置的时候，它的价值是1/log2(i+1)，所以：

$$\mathrm{DCG_p}=\sum_{i=1}^{p}\frac{rel_i}{\log_2(i+1)}=rel_1+\sum_{i=2}^{p}\frac{rel_i}{\log_2(i+1)}$$

iii) 归一化折损累计增益（NDCG）

NDCG: Normalized 的DCG，由于不同query搜索之后返回的数量是不一致的，而DCG是一个累加的值，没法针对两个不同的搜索结果进行比较（因为返回结果越多，DCG会越大），因此需要归一化处理，这里是除以IDCG。IDCG为理想情况下最大的DCG值。

举例

假设搜索回来的6个结果，其相关性分数分别是 3、2、3、0、1、2

i	reli	$\log_2(i+1)$	$rel_i / \log_2(i+1)$
1	3	1	3
2	2	1.58	1.26
3	3	2	1.5
4	0	2.32	0
5	1	2.58	0.38
6	2	2.8	0.71

所以 DCG = 3+1.26+1.5+0+0.38+0.71 = 6.86

为了求NDCG，首先需要计算IDCG：假如我们实际召回了8个物品，除了上面的6个，还有两个结果，假设第7个相关性为3，第8个相关性为0。那么在理想情况下的相关性分数排序应该是：3、3、3、2、2、1、0、0。计算IDCG@6:

i	reli	$log_2(i+1)$	$rel_i /log_2(i+1)$
1	3	1	3
2	3	1.58	1.89
3	3	2	1.5
4	2	2.32	0.86
5	2	2.58	0.77
6	1	2.8	0.35

所以IDCG = 3+1.89+1.5+0.86+0.77+0.35 = 8.37

因此，最终 NDCG@6 = DCG/IDCG = 6.86/8.37 = 81.96%