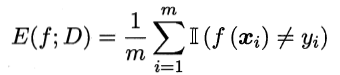
## 第三章 模型评估作业

1. 结合对性能度量部分的阅读，简述错误率、精度、查准率与查全率的含义。
2. **错误率与精度**

**错误率(Error Rate)：**是分类错误的样本数占样本总数的比例。

对样例集D，分类错误率计算公式如下所示。

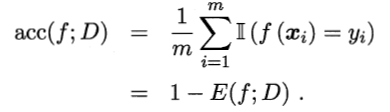




对公式（1.1）解释：统计分类器预测出来的结果与真实结果不相同的个数，然后除以总的样例集D的个数。

**精度(Accuracy)：**是分类正确的样本数占样本总数的比例。

对样例集D，精度计算公式如下所示。

注意：这里的分类正确的样本数指的不仅是正例分类正确的个数还有反例分类正确的个数。



对公式（2）的解释：先统计分类正确的样本数，然后除以总的样例集D的个数。

1. **查准率和查全率**

对于二分类问题，可将样例根据其真实类别与学习器预测类别的组合划分为真正例(true positive)、假正例(false positive)、真反例(true negative)、假反例(false negative)四种情形，令TP、FP、TN、FN分别表示其对应的样例数，则显然有TP+FP+TN+FN=样例总数。分类结果的“混淆矩阵”(confusion matrix)如表1所示。

表1：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(Precision)，又叫准确率**，缩写表示用P。查准率是针对我们预测结果而言的，它表示的是预测为正的样例中有多少是真正的正样例。定义公式如下所示。



**查全率(Recall)，又叫召回率**，缩写表示用R。查全率是针对我们原来的样本而言的，它表示的是样本中的正例有多少被预测正确。定义公式如下所示。

****

1. **F1度量（题干中没有提到，补充一个度量）**



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？

分层抽样：从500个正例中，抽70%做训练集，30%做测试集

从500个反例中，抽70%做训练集，30%做测试集

排列组合计算可得共 种划分方式

1. 已知分类结果混淆矩阵如下，试计算错误率,精度，查准率，查全率，F1。

|  |  |  |
| --- | --- | --- |
| 真实情况 | 预测结果 | |
| 正例 | 反例 |
| 正例 | 100 | 300 |
| 反例 | 200 | 400 |

1. **错误率：**
2. **精度：** 
3. **查准率：**
4. **查全率：**
5. **F1：**
6. 如下所示10个测试样本，’Class’一栏表示每个测试样本的真正标签 （P表示正例；N表示反例），’Score’表示在某分类器中每个测试样本被预测为正样本的概率：

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Class | P | N | N | P | P | N | N | P | P | N |
| Score | 0.93 | 0.85 | 0.80 | 0.7 | 0.55 | 0.50 | 0.40 | 0.3 | 0.2 | 0.1 |

画出ROC 曲线并计算AUC 的值。

**真正例率（）计算公式**：



**假正例率（）计算公式：**



1. ROC曲线的绘制：

* 根据顺序逐个把上述样本作为正例进行预测：

将每一个正例概率预测值设为分类阈值，若大于阈值则分为正类，否则为反类

每次使用公式（4.1）（4.2）计算出真正例率（）与假正例率（）得到ROC曲线上的一个坐标

* 将FPR作为横坐标，TPR作为纵坐标，连接每一个坐标点绘制成曲线图

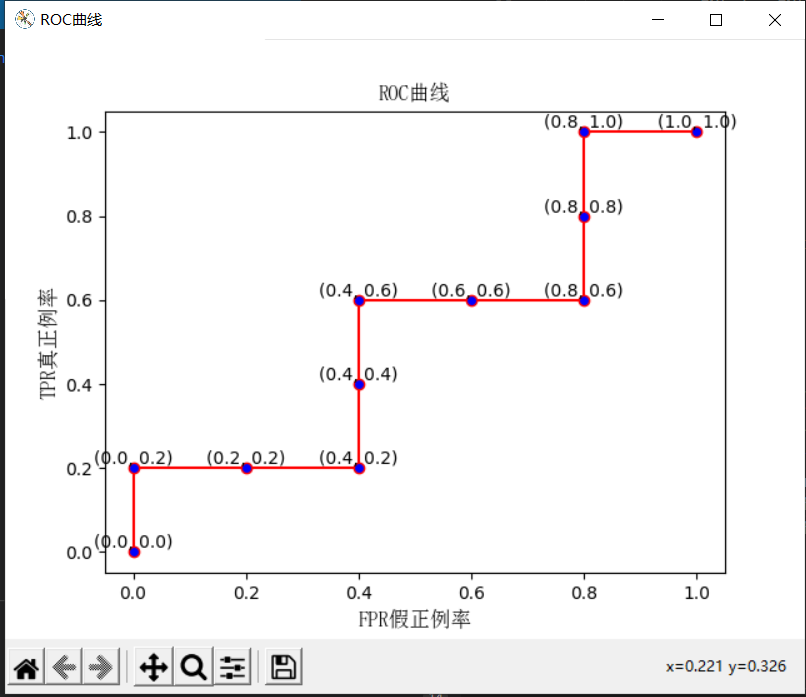
1. AUC值的计算：

从定义可得出，AUC可以通过对ROC曲线下各部分的面积求和可得。假设ROC曲线是由坐标为的点按顺序连接形成的图形，则AUC可以估算为下列公式：

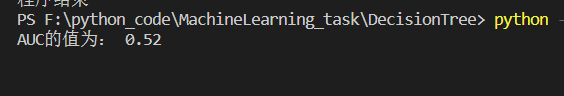


对ROC曲线代入上述公式计算即可得出AUC值

**具体计算过程较为简单重复，先选择编写python程序实现，ROC曲线结果如下：**

****

**AUC的值为0.52**

****

**按照AUC可以通过对ROC曲线下各部分的面积求和验算程序结果步骤如下：，**

,AUC结果正确

**附录代码如下：**

*# -\*- coding: UTF-8 -\*-*

import matplotlib.pyplot as plt

Class = ['p', 'n', 'n', 'p', 'p', 'n', 'n', 'p', 'p', 'n']

*# 真正标签 P表示正例；N表示反例*

Score = [0.93, 0.85, 0.80, 0.7, 0.55, 0.50, 0.40, 0.3, 0.2, 0.1]

*# 预测为正样本的概率*

TP = [0] \* 10

FP = [0] \* 10

FN = [0] \* 10

TN = [0] \* 10

TPR = [0.] \* 10 *# TPR = TP / (TP+FN)*

FPR = [0.] \* 10 *# FPR = FP / (TN+FP)*

AUC = 0.0

for i in range(10):

threshold = Score[i] *# 设置分类阈值为当前score*

for score, clas in zip(Score, Class):

if (score > threshold): *# score > threshold 预测结果为正例*

if (clas == 'p'):

TP[i] = TP[i] + 1 *# 真实标签为正p，为真正例TP*

else:

FP[i] = FP[i] + 1 *# 真是标签为反f，为假正例FP*

else: *# score <= threshold 预测结果为反例*

if (clas == 'p'):

FN[i] = FN[i] + 1 *# 真实标签为正p，为假反例FN*

else:

TN[i] = TN[i] + 1 *# 真是标签为反f，为真反例TN*

for i in range(10):

TPR[i] = TP[i] / (TP[i] + FN[i]) *# TPR = TP / (TP+FN) y坐标*

FPR[i] = FP[i] / (TN[i] + FP[i]) *# FPR = FP / (TN+FP) x坐标*

xy = list(zip(FPR, TPR))

sorted(xy, key=(lambda x: x[0]))

*# xy数组按照FDR的值进行排序，方便计算AUC*

for i in range(9):

AUC = AUC + 0.5 \* (xy[i + 1][0] - xy[i][0]) \* (xy[i + 1][1] + xy[i][1])

*# 计算AUC*

print('AUC的值为：', AUC)

plt.figure("ROC曲线") *# 定义一个图像窗口ROC曲线*

plt.plot(FPR, TPR, color='r', markerfacecolor='blue', marker='o')

*# 绘制ROC曲线*

for fpr, tpr in zip(FPR, TPR): *# 绘制每个点的坐标*

plt.text(fpr, tpr, (fpr, tpr), ha='center', va='bottom', fontsize=10)

plt.xlabel("FPR假正例率")

plt.ylabel("TPR真正例率")

plt.show()

print('程序结束')