算法分析习题讨论

CCF CSP的考试变化

- 难度在逐年加大，前2题也开始出现一些“坑”
- 以往的CCF考题在前面2题中未出现超时问题
- 2020年12月的第二题已经出现了超时问题，以湖大考点为例，大部分学生因超时而只在第二题获得70分。

目录

题目——期末预测之最佳阈值（threshold）

题目来源：CCF CSP 2020年12月第2题

【题目背景】

考虑到安全指数是一个较大范围内的整数、小菜很可能搞不清楚自己是否真的安全，顿顿决定设置一个阈值 θ，以便将安全指数 y 转化为一个具体的预测结果——“会挂科”或“不会挂科”。

因为安全指数越高表明小菜同学挂科的可能性越低，所以当 $y \geqslant \theta$ 时，顿顿会预测小菜这学期很安全、不会挂科；反之若 $y < \theta$，顿顿就会劝诫小菜：“你期末要挂科了，勿谓言之不预也。”

那么这个阈值该如何设定呢？顿顿准备从过往中寻找答案。

题目——期末预测之最佳阈值（threshold）

【题目描述】

具体来说，顿顿评估了 m 位同学上学期的安全指数，其中第 i（$1 \leqslant i \leqslant m$）位同学的安全指数为 y_i，是一个 $[0, 10^8]$ 范围内的整数；同时，该同学上学期的挂科情况记作 $result_i \in \{0, 1\}$，其中 0 表示挂科、1 表示未挂科。

相应地，顿顿用 $predict_\theta(y)$ 表示根据阈值 θ 将安全指数 y 转化为的具体预测结果。

如果 $predict_\theta(y_j)$ 与 $result_j$ 相同，则说明阈值为 θ 时顿顿对第 j 位同学是否挂科预测正确；不同则说明预测错误。

$$\mathrm{predict}_\theta(y) = \begin{cases} 0 & y < \theta \\ 1 & y \geq \theta \end{cases}$$

题目——期末预测之最佳阈值（threshold）

最后，顿顿设计了如下公式来计算最佳阈值 θ^*：

$$\theta^* = \max\left\{\underset{\theta\in\{y_i\}}{\operatorname{argmax}} \sum_{j=1}^{m} \left(\operatorname{predict}_{\theta}(y_j) == \mathit{result}_j\right)\right\}$$

该公式亦可等价地表述为如下规则:

1. 最佳阈值仅在 $\{y_i\}$ 中选取，即与某位同学的安全指数相同;
2. 按照该阈值对这 m 位同学上学期的挂科情况进行预测，预测正确的次数最多（即准确率最高）；
3. 多个阈值均可以达到最高准确率时，选取其中最大的。

题目——期末预测之最佳阈值（threshold）

【输入格式】

从文件 threshold.in 中读入数据。

输入的第一行包含一个正整数 m。

接下来输入 m 行，其中第 i（$1 \leqslant i \leqslant m$）行包括用空格分隔的两个整数 y_i 和 $result_i$，含义如上文所述。

【输出格式】

输出到文件 threshold.out 中。

输出一个整数，表示最佳阈值 θ^* 。

题目——期末预测之最佳阈值（threshold）

【样例 1 输入】

6

0 0

1 0

1 1

3 1

5 1

7 1

【样例 1 输出】

3

【样例 1 解释】

按照规则一，最佳阈值的选取范围为 $\{0,1,3,5,7\}$。

$\theta=0$ 时，预测正确次数为 4；

$\theta=1$ 时，预测正确次数为 5；

$\theta=3$ 时，预测正确次数为 5；

$\theta=5$ 时，预测正确次数为 4；

$\theta=7$ 时，预测正确次数为 3。

阈值选取为 1 或 3 时，预测准确率最高；所以按照规则二，最佳阈值的选取范围缩小为 $\{1,3\}$。

依规则三，$\theta^*=\max\{1,3\}=3$。

题目——期末预测之最佳阈值（threshold）

【样例 2 输入】

8

5 1

5 0

5 0

2 1

3 0

4 0

100000000 1

1 0

【样例 2 输出】

100000000

【子任务】

70% 的测试数据保证 $m \leqslant 200$;

全部的测试数据保证 $2 \leqslant m \leqslant 10^5$

题目解析——期末预测之最佳阈值（threshold）

题目先输入上学期的学生数m，接下来m行分别包含每位学生的安全指数和挂科情况；

要求输出的是最佳阈值，该阈值满足阈值计算公式

$$\theta^* = \max\left\{\underset{\theta\in\{y_i\}}{\operatorname{argmax}} \sum_{j=1}^{m}\left(\operatorname{predict}_\theta\left(y_j\right) == result_j\right)\right\}$$

根据题目中给出的计算规则，可逐一选取每位学生的安全值作为阈值，按照该阈值对根据输入对 m 位同学上学期的挂科情况按照公式

$$\operatorname{predict}_\theta(y) = \begin{cases} 0 & y < \theta \\ 1 & y \geq \theta \end{cases}$$

进行预测，统计预测正确的次数，保存具有最高准确率的最大值，最后输出该值。

学生该题实考情况统计——以湖大考点为例

题目	0	10-60	70	80	90	100
2	41	21	270	1	0	44
比例	10.51%	5.38%	69.23%	0.26%	0.00%	11.28%

实考人数　390

原因分析:

得70的同学基本都是由于超时造成的。

【子任务】

70% 的测试数据保证 $m \leqslant 200$;

全部的测试数据保证 $2 \leqslant m \leqslant 10^5$

常见解题步骤（伪代码描述）——输入输出处理

```
cin>>m;   \\输入学生数m
for(i=0;i<m;i++)
   {
       cin>>stu[i].y>> stu[i].result;//输入安全指数和挂科情况
    }
  ......
 cout<<maxSita<<endl;\\输出最佳阈值
```

常见解题步骤——处理（伪代码描述）

求最佳阈值的处理步骤：

```
maxOK=0;
for(i=0;i<m;i++)
  {  Sita=stu[i].y;  OK=0;        //当前阈值和当前阈值目前判断正确次数，
     for(int j=0;j<m;j++)//计算当前阈值目前判断正确次数
     {
        if(stu[j].y<Sita&&stu[j].result==0){OK++;}
        else if(stu[j].y>=Sita&&stu[j].result==1){OK++;}  }
if(OK>maxOK)
     {
        maxOK=OK;
        maxSita=Sita;
     }
else if(OK=maxOK)  {          maxSita=max(maxSita,Sita);       }   }
```

常见解题步骤（伪代码描述）

```
cin>>m;  \\输入学生数m
for(i=0;i<m;i++)
  {
     cin>>stu[i].y>> stu[i].result;//输入安全指数和挂科情况
   }
maxOK=0;
for(i=0;i<m;i++)
  {  Sita=stu[i].y;  OK=0;
     //当前阈值和当前阈值目前判断正确次数,
     for(int j=0;j<m;j++)//计算当前阈值目前判断正确次数
     {
        if(stu[j].y<Sita&&stu[j].result==0){OK++;}
        else if(stu[j].y>=Sita&&stu[j].result==1){OK++;}  }
    if(OK>maxOK)
        {
          maxOK=OK;
          maxSita=Sita;
        }
    else if(OK=maxOK)
    {
    maxSita=max(maxSita,Sita);
        }
      }
  cout<<maxSita<<endl;
```

根据化简规则，算法复杂度取决于花时间最多的双重循环，为O（m^2）

复杂度相关的问题分析

由于题目

【子任务】

70% 的测试数据保证 $m \leqslant 200$;

全部的测试数据保证 $2 \leqslant m \leqslant 10^5$

测试数据集的输入规模

10组输入文件中:

- 7组m不大于200
- 3组达到10^5

题目名称	期末预测之最佳阈值
题目类型	传统型
目录	threshold
可执行文件名	threshold
输入文件名	threshold.
输出文件名	threshold.
每个测试点时限	1.0 秒
内存限制	512 MiB
子任务数目	10
测试点是否等分	是

前面算法复杂度为O（m^2）

对于70%的测试数据m≤200，而（200）^{2}=40000，实测时可以规定时限内得出结果 ；

对于30%的测试数据m ≤ 10^5， 而（10^5）^{2} =10^{10}， 实测时超过规定时限1秒。

改进解题步骤（伪代码描述）

```
cin>>m;   \\输入学生数m
 for(int i=1;i<=m;i++)
        {cin>>a[i].first>>a[i].secod; \\输入安全指数和挂科情况
         cnt[a[i].secod]++; \\统计通过和挂科的总人数}
 sort(a+1,a+m); \\进行快速排序，数据将按安全指数升序排列，分析：安全指数相同的 \\
学生数据将连续存放
```

改进解题步骤（伪代码描述）

以样例2为例，对a数组快排后的结果

1 0

2 1

3 0

4 0

5 1

5 0

5 0

100000000 1

分析：相同安全指数的学生信息排列将连续，对于相同安全指数，可以不用重复对每位学生挂科情况的预测；

初始时，选择第一个最小值作为预测的阈值，此时每条记录的安全指数均大于等于该阈值，都预测通过的，预测正确数为所有学生挂科情况取值中1的数目，即cnt[1];

选择第i个安全指数比第i-1个安全指数时，预测正确次数增量为0_{i-1}（即第i-1个安全指数0的个数）-1_{i-1}（即第i-1个安全指数1的个数）

即：OK_1=cnt[1],Ok_i=Ok_{i-1}+0_{i-1}-1_{i-1}

可推出结论Ok_i= cnt[1]+now（0）-now（1）其中now（0）和now（1）分别为前i-1个安全指数中0的个数和1的个数

改进解题步骤（伪代码描述）

```
cin>>m;   \\输入学生数m
for(int i=1;i<=m;i++)
        {cin>>a[i].first>>a[i].secod; \\输入安全指数和挂科情况
          cnt[a[i].secod]++; \\统计通过和挂科的总人数}
sort(a+1,a+m); \\进行快速排序，数据将按安全指数升序排列
        \\分析：安全指数相同的学生数据将连续存放
int maxOK=0, maxSita=0; //初始化最高正确次数和最大阈值
for(int i=1;i<=m;i++){
    if(a[i].first==a[i-1].first&&i>0){   \\连续安全指数不重复处理，只累加通过人数和挂科人数
      now[a[i].second]++;continue;}
    int OK=cnt[1] + now[0] -now[1]; \\在某安全指数首次出现时计算对应的预测正确人数
    if(OK >= maxOK) maxOK =OK, maxSita=a[i].first;\\记录当前最大预测正确次数和对应的安全指数
    now[a[i].second]++;        }    cout<< maxSita <<endl;
```

根据化简规则，算法复杂度取决于花时间最多的sort，为O（mlogm）

复杂度相关的问题分析

由于题目

【子任务】

70% 的测试数据保证 $m \leqslant 200$;

全部的测试数据保证 $2 \leqslant m \leqslant 10^5$

测试数据集的输入规模

10组输入文件中:

- 7组m不大于200
- 3组达到10^5

题目名称	期末预测之最佳阈值
题目类型	传统型
目录	threshold
可执行文件名	threshold
输入文件名	threshold.
输出文件名	threshold.
每个测试点时限	1.0 秒
内存限制	512 MiB
子任务数目	10
测试点是否等分	是

改进算法复杂度为O（mlogm）

对于30%的测试数据$m=10^5$， 而$10^5 \log 10^5 = 10^5 * 5\log 10 \approx 6.64*10^6$， 实测时可满足规定时限1秒。