研究问题：自动定义题目难度

代码开源地址：<https://github.com/caposerenity/Data_Science>

研究方法：

就自定义题目难度这一问题，为我们使用了test\_data这一较全面的数据集，在下载所有测试数据集后，我们首先抽样查看了约一百份不同题型题目、不同同学的提交代码，发现其中存在使用C++等python外的语言编码、面向测试用例编程以刷分等现象。因此我们在评估题目难度之前首先对数据进行了预处理，清洗了面向测试用例的提交的分数记录。

在预处理筛出面向用例提交记录中，我们采取了正则表达式等特征匹配方法。针对先前抽样中出现的面向用例特征进行筛除，主要包含了：连续的if()print（）；else if（）print（）的情况；直接以if（）print（）开头且代码长度短于设置阈值的情况；仅有print（）内容且print中内容未包含运算的情况等若干种面向用例特征。通过对这些行为进行一定的赋分，达到阈值即认为存在面向用例的作弊行为，将这次提交标记为【cheat】并将分数改为0，同时在该同学这道题目的case记录下增加cheat标签。在对挑出的另外20次面向用例提交和30次正常提交记录进行的测试中，我们实现了95%的面向用例检测率和0%的误诊率。基于此，我们对test\_data中的所有提交记录进行了面向用例和是否为python代码的预处理过程。

数据预处理结束后，我们考察了每道题提交的平均分数，20分以下占比，满分占比，平均debug时间和提交次数，综合量化得出每道题目的难度系数。

对于每道题的分数，我们发现在经过预处理后的数据集中，出现的分数有大量的0分和大量的满分，因此仅仅以平均分作为判断难度的依据有失偏颇。因此我们另外综合了每道题的满分率和低分率（20分以下占比），使用公式levelbase=((100-average)/100\*0.6+score20/num\*0.2+(num-score100)/num\*0.2)\*100，得到难度的基础系数（百分制，难度系数越高难度越大）。