南京大学

NANJING UNIVERSITY

研 究 报 告

通过分析慕测平台数据分析学生编程能力

学　　院：软件学院

创建时间：2020 年 7 月 20 日

学生姓名：戴俊浩，郭增嘉，刁苏阳

学　　号：181250022，181250040，181250026

教　　师：陈振宇

2020 年 07 月 23 日

目录

1 研究问题: 学生编程能力探究

1.1 详细介绍

我们将研究方向设定为学生编程能力的分析，我们通过主成分分析法对平台的数据进行题目难度分析然后结合分析的题目难度，将学生的编程得分等因素综合考量，使用相关性检测之后将高度重复的变量除去，最后使用多元线性回归对高维数据进行分析，产生可视化的三维函数，得出一个学生的编程能力的计算公式，当学生答题产生数据后就能通过相应的算法来得出这个学生的编程能力。

1.2 研究背景

在编程实践越来越热门的今天，网络上已有大量的 OJ 平台供用户练习。这些 OJ 平台之间的竞争是激烈的，为了获取用户，平台必须在一系列重要指标上比别人做得更优秀。而评价 OJ 平台的重要指标之一，就在于其能在有限的时间内帮用户最大化地提升能力。想要满足这个指标，避免让用户陷入盲目的题海战术，这就需要平台能对题目的难度以及用户的编程能力有清楚到足以量化的认识。

1.3 应用场景与价值

我们现阶段的研究成果可以应用于网路上的在线编程平台，通过学生对题目的完成实践、提交次数、得分数量等诸多维度外加对于学生代码的分析，归纳评判一个题目的难度以及一个学生的编程能力指标，尽可能展示一个学生编程能力的真实水平。

在大数据平台的支持下，学生可以明确定位自己的编程能力，同时明晰题目的具体难度水平，从而选择适合自己的题目进行锻炼，方便学生采取合适的学习路径，省却大量筛选题目的实践，有效提升编程能力和代码水平。

2 研究方法

2.1 数据源

使用了慕测平台的数据，分析学生的编程能力，通过慕测提供的 json 文件可以看到学生的 userID 以及一系列的提交记录，使用 python 读取 json 文件获

取所需数据。

2.2 数据分析

2.2.1 主成分分析

在处理题目难度的时候使用了 PCA[1]，首先将题目的数据通过对 json 文件的分析导入到一个矩阵中，这个矩阵是一个 3000*7 的矩阵，7 列是题目的 7 个维度的数据，分别是该题的初始提交分，该题的最终得分，该题的平均分，该题的最终得分和提交次数的比值，该题的前一半的提交的斜率，也就是用户在做题时分数的增长速率还有该题的总耗时以及该题的最终分数和其他题目的最终分数的差值。通过对 json 文件的读取实现了对这些数据的分析。

使用主成分分析法，首先将数据注入到 3000*7 的二维数组[2] 中，从 7 个维度展现这个题目难度 c。将所有的数据减去该列的平均值。做标准化，然后将二维数组转换为矩阵，求矩阵的协方差矩阵即 $C = B^T \cdot B$，然后一次求出特征值和特征向量，对特征值进行排序，选取最大的一个特征值对应的列作为主成分，得到相应的投影矩阵，最后利用投影矩阵得出降维之后的数据（$E = B \cdot D^T$），最后对得到的数据进行标准化，将数据的区间限定为 $[0\ 1]$，这样就使用主成分分析的方法获得了题目的难度系数。

2.2.2 相关性分析与显著性检验

在处理学生编程能力的多维数据时，我们意识到这多个维度之间可能具有高度的相关性。我们对下列维度计算了其之间的 pearson 相关系数，分别是学生的得分（综合了最终得分与难度)、学生的最初得分、学生的分次比、学生的总耗时、学生与该题平均值的差值、学生的规范分。

我们先把数据做标准化处理，然后注入一个 6*271 的矩阵[3]。在对这个数组计算其 pearson 相关系数后，我们得到一个 6*6 的矩阵。这个矩阵的第 i 行第 j 列和第 j 行第 i 列的值均表示原矩阵第 i 行与第 j 行之间的 pearson 相关系数。

$$r_{X,Y} = \frac{n\sum_i X_i Y_i - \sum_i X_i \sum_i Y_i}{\sqrt{n\sum_i X_i^2 - (\sum_i X_i)^2}\sqrt{n\sum_i Y_i^2 - (\sum_i Y_i)^2}}$$

[1]即主成分分析法，下文中全部使用主成分分析法

[2]3000 是题目的数量，虽然实际上题目编号从 200+ 开始，但是使用了全部题号，如果产生更多题目需扩容

[3]271 是用户的总数

同时，我们也得到一个 6*6 的 p-value 矩阵[4]，作为对 pearson 系数的显著性检验。

在最终确认高相关性的维度时，我们使用了相当严格的标准，要求 pearson 相关系数值大于 0.9 或小于-0.9，p-value 小于 0.01。最后返回一个 size 为 2 的二维列表 positive-negative，其中 positive-negative[0] 代表高度正相关的维度对，positive-negative[1] 代表高度负相关的维度对。

2.2.3 多元线性回归

处理学生的编程能力时使用了多元线性回归，我们尝试着将 9 位学生的样本的代码水平进行了评价，最后对他们的编程能力给出了一个分数，为了能够使多元线性回归更加准确这 9 个学生是 2 个较差，5 个一般和 2 个较好，多元线性回归中一共使用了 4 个变量对一个学生的编程能力进行分析，分别是学生提交的所有题目的平均分，学生的提交分数和提交次数的比值，学生做这个题目所用的时间，学生的代码的规范得分。经过了上一步的皮尔森系数的相关性检验这几个数据的相关性并不是特别大，所以将这些数据使用最小二乘法进行多元线性回归，得出了一个学生编程能力的计算公式，当学生产生了答题记录之后就能通过慕测系统 json 数据大致估算出这个学生的编程能力。

首先将数据注入预设好的矩阵当中，每一行的第一项都是 1，作为常数项的系数，将二维的数组转换为矩阵 X，同时将我们对就为同学的评分注入到一个 $9*1$ 的矩阵 Y 中，之后使用

$$\hat{\boldsymbol{B}} = \begin{pmatrix} \hat{b_0} \\ \hat{b_1} \\ \vdots \\ \hat{b_p} \end{pmatrix} = (\boldsymbol{X^T X})^{-1} \boldsymbol{X^T Y}$$

对数据进行计算就可以较为轻松的得出最后的学生的编程能力和几个数据之间的函数关系。

[4]p-value 是一种概率：在原假设为真的前提下，出现该样本或比该样本更极端的结果的概率之和。

2.2.4 数据可视化

在多元线性回归之后，我们获得了各个维度对于结果的影响系数和偏移常量。但高维度的函数更为抽象，我们难以根据单纯的数据判断此次多元线性回归的可信度和可行性。

我们抉择了其中影响最显著的两个维度（用户最终得分，分次比）作为元素绘制回归线进行可视化展示，同时将各个学生的编程能力的相关系数一同绘制入可视化图中展现回归的具体效果和情况。

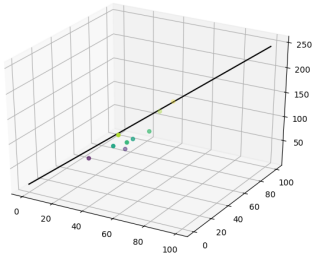

图 1: 可视化

从上图我们可以看到，代表各个学生编程能力的点围绕在回归线附近。由此，初步判断以此多元线性回归所得学生编程能力分析函数的。(进一步确认需要更多信息以及数据的支持)

3 代码解析

3.1 代码逻辑

3.1.1 题目难度分析模块

题目难度部分一共有 9 个文件，其中包括了 7 个模块数据 python 处理文件，1 个标准化函数，1 个 PCA 算法。7 个模块的数据读取和标准化函数都在 component 文件夹中。

(1) 文件夹中的 averageScore.py 读取了的 caseid 对应的所有用户的所有的提交分的平均值，并使用字典的形式进行了存储。

(2) 文件夹中的 CompareTo.py 读取了 caseid 对应的与其他题目的最终得分的平均分的差值，并使用字典的形式进行了存储。

(3) 文件夹中的 FinalScore.py 读取了 caseid 对应的所有用户最终得分的平均分，并使用字典的形式进行了存储。

(4) 文件夹中的 FirstGit.py 读取了 caseid 对应的所有用户的初始提交分的平均分，并使用字典的形式进行了存储。

(5) 文件夹中的 FractionRatio.py 读取了 caseid 对应的所有用户最终得分和用户所有的提交次数的比值，并使用字典的形式进行了存储。

(6) 文件夹中的 HalfSlope.py 读取了 caseid 对应的所有用户的前一半的提交和分数的斜率，并使用字典的形式进行了存储。

(7) 文件夹中的 timeToSolve.py 读取了 caseid 对应的所有用户的做一道题的总耗时的平均值，并使用字典的形式进行了存储。

(8) 最后有一个 normalize.py 文件可以接受一个字典，然后将字典中的所有的值做标准化，转换成 [0,100] 区间的值。

最后在 Usercomponent 文件夹中使用了 difficultMain.py 引用了所有的上述模块，使用主成分分析法分析了数据，算出了最后的题目难度的值

3.1.2 学生编程能力分析模块

学生编程能力部分一共有 10 个文件，其中包括了 6 个模块数据 python 处理文件，1 个标准化函数，1 个 pearson 值与显著检验文件，1 个多元线性回归文件,1 个 pylint.conf 文件。6 个模块的数据读取和 pylint.conf 都在 Usercomponent 文件夹中。

(1) 文件夹中的 userScore.py 读取了的 userid 对应的所有题目的所有的最终得分 * 难度的平均值，并使用字典的形式进行了存储。

(2) 文件夹中的 CompareToAverage.py 读取了 userid 最终得分与本题平均得分的差值的平均值，并使用字典的形式进行了存储。

(3) 文件夹中的 FirstUpload.py 读取了 userid 对应的所有题目的初始提交分的平均分，并使用字典的形式进行了存储。

(4) 文件夹中的 Fraction.py 读取了 userid 对应的所有题目每次提交分之和和题目所有的提交次数的比值，并使用字典的形式进行了存储。

(5) 文件夹中的 CodingStyle.py 读取了 userid 对应的所有题目的规范分，并使用字典的形式进行了存储。

(6) 文件夹中的 Time.py 读取了 userid 对应的所有题目的耗时的平均值，并使用字典的形式进行了存储。

(7) 文件夹中的 pylint.conf 为 CodingStyle 使用的 pylint 的配置文件。

在 Component 文件夹中使用了 normalize.py 接受一个字典，然后将字典中的所有的值做标准化，转换成 [0,100] 区间的值。在 dataSolve 文件夹中使用 Pearson.py 引用上述所有模块计算 pearson 相关系数与 p 值，返回高度正和负相关的列表。在 dataSolve 文件夹中使用 MultiLinear.py 进行多元线性回归，返回学生编程能力的值。

3.1.3 数据可视化

数据可视化技术依托于 python 的 numpy 及 matplotlib.pyplot 包。降序排列、筛选多元线性回归分析所获得的各维度对最终成绩的影响程度，获得两个影响最为显著的维度，绘制多元线性回归线。同时，将对学生的维度分析数据导入图标当中共同展示，从而直观分析多元线性回归线的可行性。

3.2 代码开源地址

代码已由 GitHub 托管，开源项目地址：点击访问[5]

4 案例分析

4.1 分析题目难度

在此处以编号为 2061 的题目为例。

我们从以下 7 个维度衡量题目的难度，分别是初始提交分、最终得分、平均分、分次比、总提交次数前一半的斜率、总耗时、同类型的平均分进行比较。得出的结果依

[5]点击后将跳转到 https://github.com/Analysismen/Student-Coding-Ability

次为 87.23、97.87、97.87、70.77、84.04、1581108484334.31、0.13164424112140694，然后对其作标准化处理，处理后结果依次为 81.59、91.02、91.01、68.3、76.78、14.98、82.17，在对所有题目进行主成分分析得出的主成分为初始提交分，我们认为该题主成分为初始提交分。

4.2 分析学生编程能力

在此处以编号为 3544 的学生为例。

在分析完整题目难度的各个维度并进行主成分分析降维后，我们获得一个用于衡量题目难度的维度数据，每一道在网站上出现的题目都持有一个相应的数据用作该题难度的衡量。

在此基础上，我们以四个维度划分分析学生的编程能力（初始采用六个维度的衡量方式,通过相关性分析后删去两个),分别从该题得分 * 该题难度、分次比、总耗时、规范分方面分别计算打分其中,该生在各方面分别得分为 30.86,39.84,26.36,13.88。在获得了这些数据后，我们便可以套用多元线性回归所得出的公式进行最终编程能力值的计算。

5 项目前瞻: 学生学习路径推荐

该项目在分析完题目难度和学生编程能力之后可以应用于学习学生编程学习路径推荐，在学生使用慕测平台完成一定量的练习之后就可以让后台处理学生的编程数据，通过这些数据，我们能对学生各方面的编程能力有一个量化的认识，并通过多元线性回归最后得出的公式给出一个学生的总的得分。使用学生的编程能力得分结合本项目中对题目难度的算法，可以算出一个题目的难度系数。

在完善项目之后，可以将题目难度和学生的编程能力进行一定的联系，通过聚类算法对题目进行分类，之后联合学生在该类型的题目中的编程能力表现进行适当的题目推荐和类别推荐，再结合分析学习内容的递进顺序的结果，完成学生的学习路径推荐。

6 附录

6.1 图表

由于多元线性回归的维度过高，我们抉择了分析学生编程能力的数据中影响最显著的两个方面（用户最终得分，分次比）和学生的最终编程能力得分作为元素绘制回归线进行三维展示，同时将各个学生的编程能力的相关系数一同绘制入可视化图中展现回归的具体效果和情况。

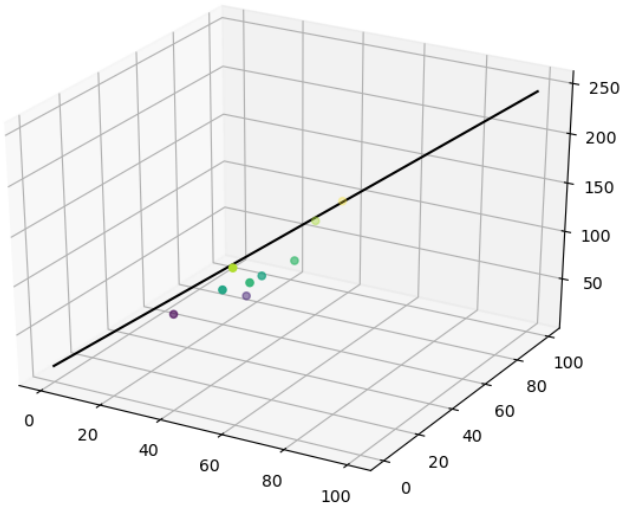

图 2: 可视化图表

6.2 参考文献

CSDN 主成分分析（PCA）原理详解

简书【python】pylint 在项目中的使用

陈振宇，10.3 多元线性回归分析.pdf

陈振宇，10.1 相关性分析.pdf

陈振宇，9.3.1+pca.pdf