華東師範大學
EAST CHINA NORMAL UNIVERSITY

课程论文
COURSE PAPER

专业选修课程：社会计算

基于人员模式的 GitHub 项目健康度评估

学　　院　数据科学与工程学院

学生姓名　丁正源、贾柏寒

任课教师　钱卫宁

日期：2022 年 05 月 29 日

目录

一． 引言

一个开源项目可供人们参考的指标有很多，包括 star 、fork、PR、MR、贡献者数量等。然而这些明面上的指标很可能是虚荣指标，会受到营销策略的影响（如刷 star、刷 fork 等），导致在评估开源项目的健康度和可持续性时并不准确，甚至失真严重。因而，需要探索，找出真正有价值指标来评估项目健康度。

由于 GitHub 项目与仓库的开发、维护、运营，都需要各类人员作为核心去控制，人员变化对于项目的影响具有至关重要的作用，因此，我们选择由人员进行切入点来探索项目健康度指标。

二． 问题描述

问题一：项目人员分类

开源协作领域，不同的开发者具备不同的角色，也具备不同的专长，在项目发展周期内，每段时期内参与到开源项目的人员组成都有会变化，通过对人员进行分类，可以进一步探究人员组成变化规律，帮助社区核心维护者更好的进行项目管理。

问题二：项目健康度指标提出

一、项目协作度指标

开源项目的发展离不开核心成员的推进，核心成员推进速度相当程度决定了项目的健康水平，我们希望通过分析项目团队内各成员的协作方式，如成员间共同对同一文件的修改，来构建协作关系网络图，探究分析项目的协作程度，从而提出项目健康度的评估指标。

二、项目贡献者比率指标

项目贡献者比率决定了该项目的开发团队人员流失率以及项目对新开发者的吸引力，我们希望能够对项目的人员组成变化进行预测，即预测项目贡献者比率，来对项目健康度的评估提出参考指标。

三． 方法

3.1 项目人员分类

首先对项目参与人员进行初步分类，可分为机器人、核心开发者、外围开发者。

一，依据项目参与人员用户名是否含有 bot 来初步剔除机器人，后续还可以通过分析是否某些用户存在异常的行为数据，如存在极端高频的 IssueCommentEvent 来判断是否为机器人。

二，每个参与项目开发的人员均有其身份属性，我们通过获取能反映开发人员的身份信息的字段如 company、email、bio 等，判断其所属身份是否为企业雇佣身份，将有企业雇佣身份的开发者归类为核心开发者。

图 1. PaddlePaddle/Paddle 仓库用户 dzhwinter 的身份信息

后续工作即对外围开发者群体进行进一步分类

3.1.1 数据采集与处理

数据选取 PaddlePaddle/Paddle 仓库，剔除机器人与核心开发者，统计一定时段内（时间尺度为 3 个月）五种行为事件（openissue、issuecomment、reviewcomment、openPR、mergePR）由每个 actor 触发的发生次数。

事件	描述
IssueCommentEvent	对一个 issue 的评论
PullRequestReviewCommentEvent	对 pullrequest 的代码进行评论的相关操作
IssuesEvent	对 issue 的相关操作
PullRequestEvent	对 pullrequest 的相关操作

表 1. GitHub 行为触发事件表

字段	描述

openissue	待处理的 issue 总数
issuecomment	项目当前时段 issue 的评论总数
openPR	待处理的 pullrequest 总数
reviewcomment	项目当前时段 pullrequest 审核总数
MergePR	合并的 pullrequest

表 2. 数据字段描述表

3.1.2 聚类模型

通过对采集到的数据的初步观察，发现数据较为稀疏，且维度较高，综合考虑下，选择了谱聚类算法。

谱聚类只需要数据之间的相似度矩阵，因此对于处理稀疏数据的聚类很有效，这点传统聚类算法比如 K-Means 很难做到，又由于谱聚类使用了降维，因此在处理高维数据聚类时的复杂度比传统聚类算法好，相似度矩阵构建方式选择使用高斯核函数。

谱聚类主要需要考虑的参数为类别数 n_clusters，此处使用轮廓系数来确定类别数，轮廓系数这一指标无需知道数据集的真实标签。取值范围[-1, 1]，值越大，聚类效果越好。旨在将某个对象与自己的簇的相似程度和与其他簇的相似程度作比较。轮廓系数最高的簇的数量表示簇的数量的最佳选择。

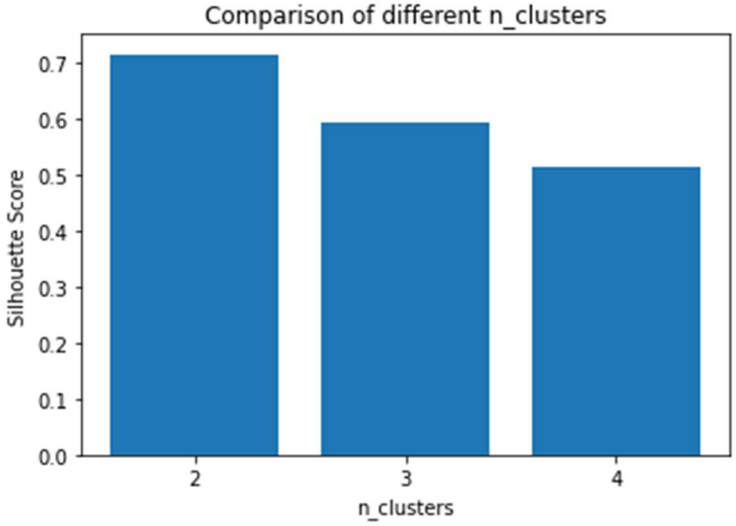

图 2. 不同类别数下轮廓系数比较图

如图 2 所示，最终选择类别数为 2 类，此时聚类效果较为理想。

3.1.3 聚类结果可视化

为了验证聚类结果的合理性，并依据聚类结果为外围开发人员进行打上合适标签，对数据做加权统计，加权规则如下：

$$代码贡献度 = a * openissue + b * issuecomment$$

$$参与讨论度 = c * reviewcomment + d * openPR + e * mergePR$$

$$a = 1, b = 2, c = 3, d = 4, e = 5$$

给出每个外围开发者代码贡献者和参与讨论度两个指标，并依据分类结果做可视化，如图 3 所示：

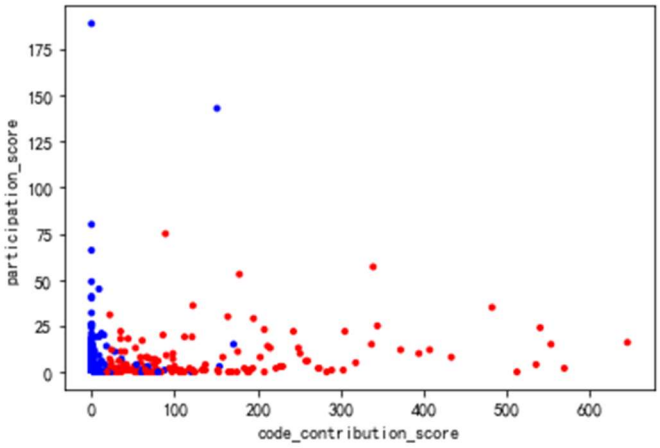

图 3. 聚类结果可视化图

通过上述可视化，验证了聚类结果的有效性，并据此进一步可将人员分类归为代码贡献者和问题报告者两类。

3.2 项目健康度指标

基于上述人员分类模型，将进一步针对项目不同角色进行项目健康度相关性指标的探究，最后针对于核心开发人员以及项目所有贡献者进行探究后，提出了项目协作度指标和项目贡献者比率指标。

3.2.1 项目协作度指标

对于核心开发者群体，开发任务不应该集中在少数几个开发者上，而应该尽可能分散在较多的开发者中，项目协作度越好，反映了项目团队管理和架构更健康成熟。

协作关系建模：

我们所需要构建的协作关系网络图为无向加权图，其中顶点代表项目成员，边代表两成员之间协作关系，协作关系的判断方式选择依据公共修改文件，即若两个成员间存在公共修改文件，即有协作关系，则两个顶点存在边，边的权重即为两个成员公共修改文件数，最后使用邻接矩阵存储图。

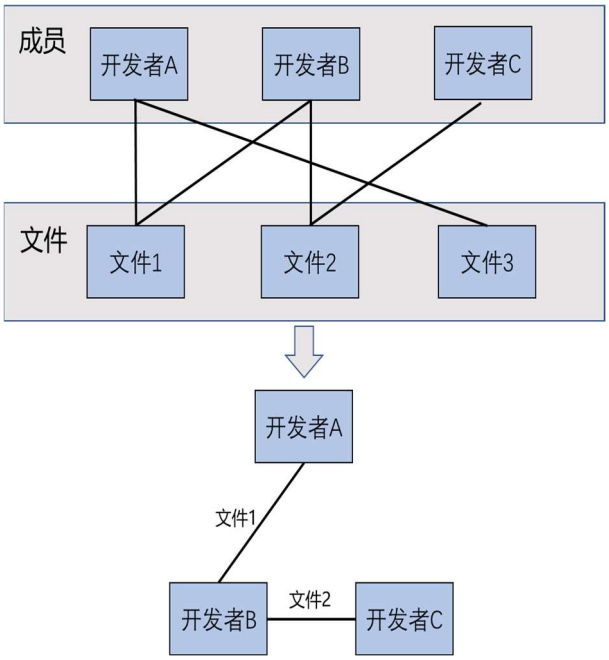

图 4. 协作关系建模示意图

协作关系数据采集与处理：

为了协作网络关系图的可视化展示，数据选取上使用了一个 5 个成员迷你项目仓库 Neon/Neon，还有 4 个大型仓库 PaddlePaddle/Paddle，apache/tvm，apache/echarts，ant-design/ant-design 2021 年以来的数据

通过 git 命令获取到项目成员数据，以及成员修改文件，如下图所示：

```
$ git log --stat --no-merges --pretty=format:"%ae"
19302010065@fudan.edu.cn
 src/views/ExercisePage.vue         |  1 +
 src/views/ListWordsPage.vue        |  3 ++-
 src/views/ReviewPage.vue           | 32 ++++++++++++++++++------------
 src/views/SelectUnit.vue           |  4 ++++
 src/views/TabsPage.vue             | 11 ++++++-----
 tests/unit/random-review.spec.js   |  2 +-
 tests/unit/start-exercise.spec.js  |  2 +-
 7 files changed, 28 insertions(+), 27 deletions(-)

19302010080@fudan.edu.cn
 tests/unit/random-review.spec.js | 98 ++++++++++++++++++++++++++++++++++++++++
 1 file changed, 98 insertions(+)

19302010062@fudan.edu.cn
 package-lock.json          |  2 +-
 package.json               |  2 +-
 src/views/ExercisePage.vue |  1 +
 src/views/ReviewPage.vue   | 25 ++++++++++++++++++++++---
 4 files changed, 25 insertions(+), 5 deletions(-)
```

图 5. git log 获取协作关系原始数据

数据预处理即通过正则表达式匹配字符串来进行数据提取，得到每个成员所修改的文件名集合。

之后需要求解两两成员之间公共修改文件个数，存储于邻接矩阵中。

部分协作关系网络可视化展示：

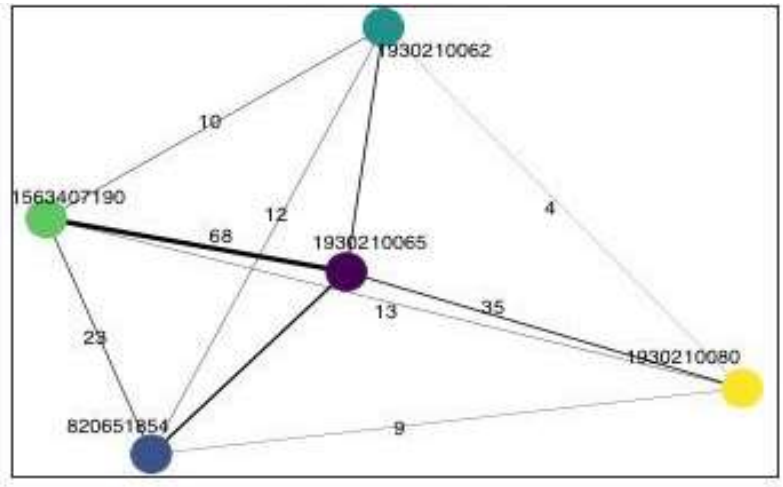

图 6. 迷你项目协作关系网络图

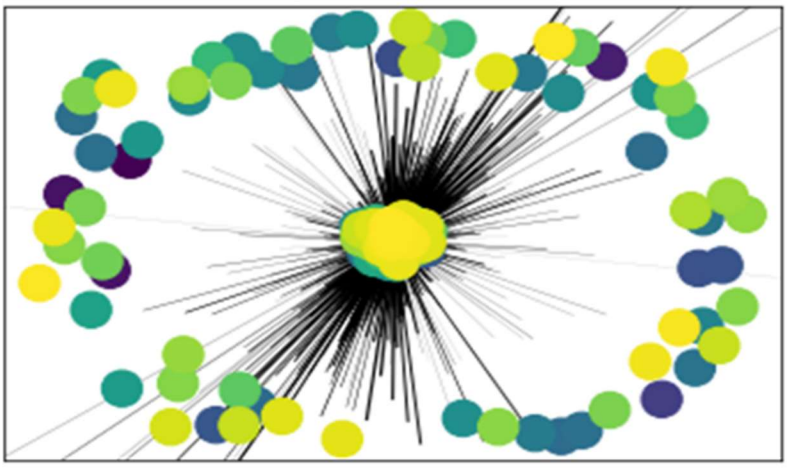

图 7. PaddlePaddle/Paddle 2021 年期间项目成员协作关系网络图

项目协作度指标评估：

根据项目协作关系网络特征，给出项目协作度指标定义如下：

$$CooperativeDegree = \frac{2}{v^2-v}\sum_{i=1}^{n}\mathbb{I}_A(w_i)\,(w_i - Density)$$

$$\text{其中 } Density = \frac{\sum_{i-1}^{n} w_i}{(v^2-v)/2},\quad A = (0,\infty)$$

其中 n 为边的个数，w_i 为第 i 条边的权重，v 为顶点个数。

依据上述所得各项目协作关系网络图与协作度指标定义，得到 5 组项目的项目协作度指数，指数越低，说明项目成员协作关系越密切，任务分配越均衡，项目健康程度越高。

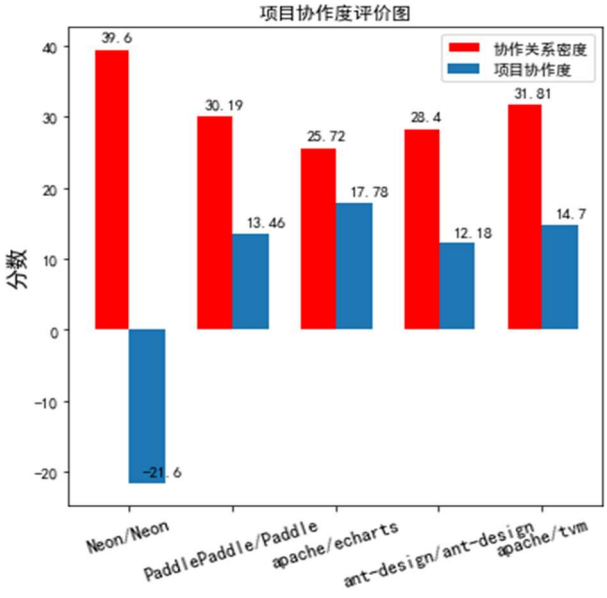

图 8. 各项目协作度评价图

3.2.2 项目贡献者比率指标

本项目首先考虑到了所有项目参与者的影响，即只要对项目有任何的参与度的提升（包括提 issue,fork,star 等）都会被纳入总项目人数，进一步考虑贡献者在总项目人数中的占比，从而得到了项目贡献者比率。

首先对贡献者概念做扩充，默认的贡献者（contributor）的定义只是代码贡献者，但从实际角度出发，参与到社区中的所有开发者，包括提交 bug、参与讨论、参与代码 review 的开发者事实上都对项目是有贡献的。（引用）

项目贡献者比率定义：即项目在一定时段内，有贡献行为的开发者比例，如果持续贡献的开发者比例越高，则说明该项目的开发团队人员流失率较低，项目对新开发者的吸引力强；反之，如果持续贡献的开发者比例越低，则说明该项目的开发团队人员流失率较高，项目对新开发者的吸引力相对就较低。

公式：

项目贡献者比率=

贡献开发者人数/项目总人数 $=(contributor\_num)/(actor\_num)$

（其中，contributor_num 为贡献者人数，actor_num 为项目总人数）

在模型的建立之前首先需要进行探索，包括单独项目数据的可视化分析。在聚合表诸如 ads_open_index__repo_201901_activity 类似的表中选取了 Paddlepaddle/paddlepaddle 项目 2019.1-2021.11 的数据，并对公式中的贡献者人数和项目总人数随时间变化的图表进行了可视化，结果如下：

图 9. 项目总人数与贡献者人数可视化图

注意到项目的贡献者人数随着时间的推移展现了一个总体上升的趋势，而项目总人数相对来说较为稳定，略微有小增长。可以注意到在 2021 年 10 月左右项目贡献者和项目总人数有了一个突变，据了解是该月数据的缺失导致。

3.2.3 项目贡献者比率模型建立

下一步就是建立项目贡献者比率模型。该部分分为三个步骤，首先是数据获取。考虑到一个良好模型的建立需要优质的数据，并且该项目的目的是分析项目健康度与人员模式、人口变化的关系，而低健康度的项目在人员模式和人口变化上不具备成熟性，例如：一些较为冷门的项目人员变化巨大，某个月只有 20%或更少的人员是项目贡献者，但下一个月参与项目的人员大大减少，再下一个月又新增很多人，这样不稳定的突变会对模型的建立造成较大的干扰，因此需要选择人员模式和人口变化较为稳定的项目，选择高健康度的项目一般来说可以达到这样的要求。

由于在模型的探索阶段，先在聚合表中选择了单个项目的数据进行模型的训练评估，这里仍然延续上一阶段选择了 PaddlePaddle/Paddle 项目，但将时间线延长至 2016.8-2021.11，以月为尺度做数据统计，具体说明见表 3。

字段	描述
Actor_num	项目当前时段用户总人数
Contributor_num	项目当前时段贡献者人数
Open_issue	待处理的 issue 总数
Issue_comment	项目当前时段 issue 的评论总数
Open_pull	待处理的 pullrequest 总数
Pr_review	项目当前时段 pullrequest 审核总数
Merge_pull	合并的 pullrequest

表 3. 数据字段描述表

对所有的字段的分布进行可视化如下图：

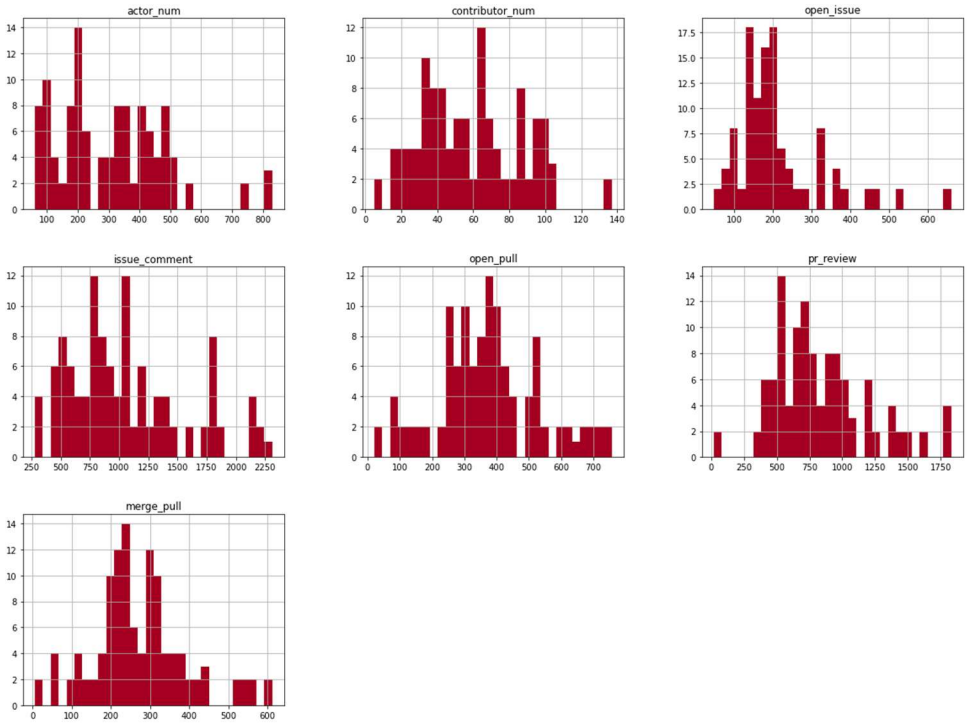

图 10. 相关变量分布图

单独查看与项目贡献者比率公式有关的两个变量：项目贡献者人数、项目总人数

的分布以及他们的联合分布图如下：

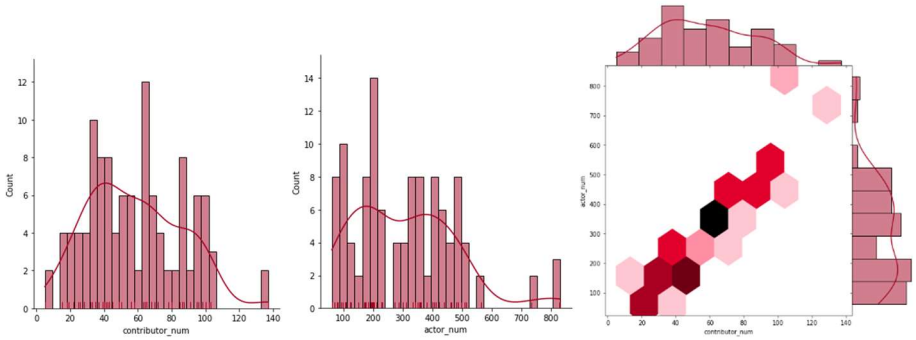

图 11. 项目贡献者人数、项目总人数分布图、联合分布图

变量分布、联合分布总图如下：

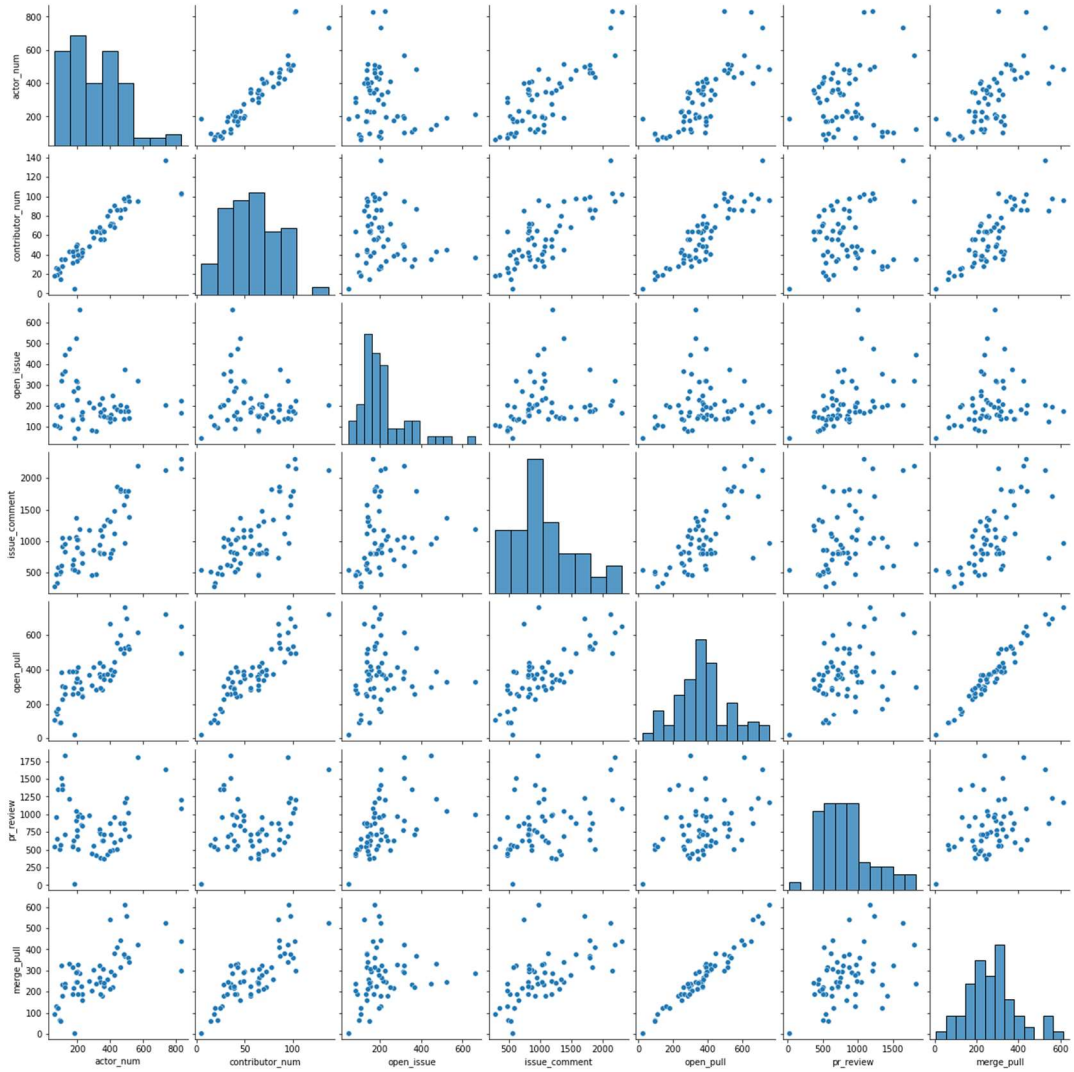

图 12. 变量分布总图

在进行模型的建立前，我们可以先对变量之间的相关性进行分析。相关系数的计算有很多种方法，这里选择的是皮尔逊相关系数[2]：

两个变量之间的皮尔逊相关系数定义为两个变量之间的协方差和标准差的商：

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y},$$

上式定义了总体相关系数，常用希腊小写字母ρ作为代表符号。估算样本的协方差和标准差，可得到皮尔逊相关系数，常用英文小写字母r代表：

$$r = \frac{\sum_{i=1}^{n}(X_i-\overline{X})(Y_i-\overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i-\overline{Y})^2}}.$$

r亦可由(X_i, Y_i)样本点的标准分数均值估计，得到与上式等价的表达式：

$$r = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{X_i - \overline{X}}{\sigma_X}\right)\left(\frac{Y_i - \overline{Y}}{\sigma_Y}\right).$$

其中 $\frac{X_i - \overline{X}}{\sigma_X}$，$\overline{X}$，$\sigma_X$，分别是对$x_i$样本的标准分数、样本平均值和样本标准差。

由此绘制出的变量相关系数图如下：

注意：颜色越偏向蓝色相关性越高，越偏向红色相关性越低

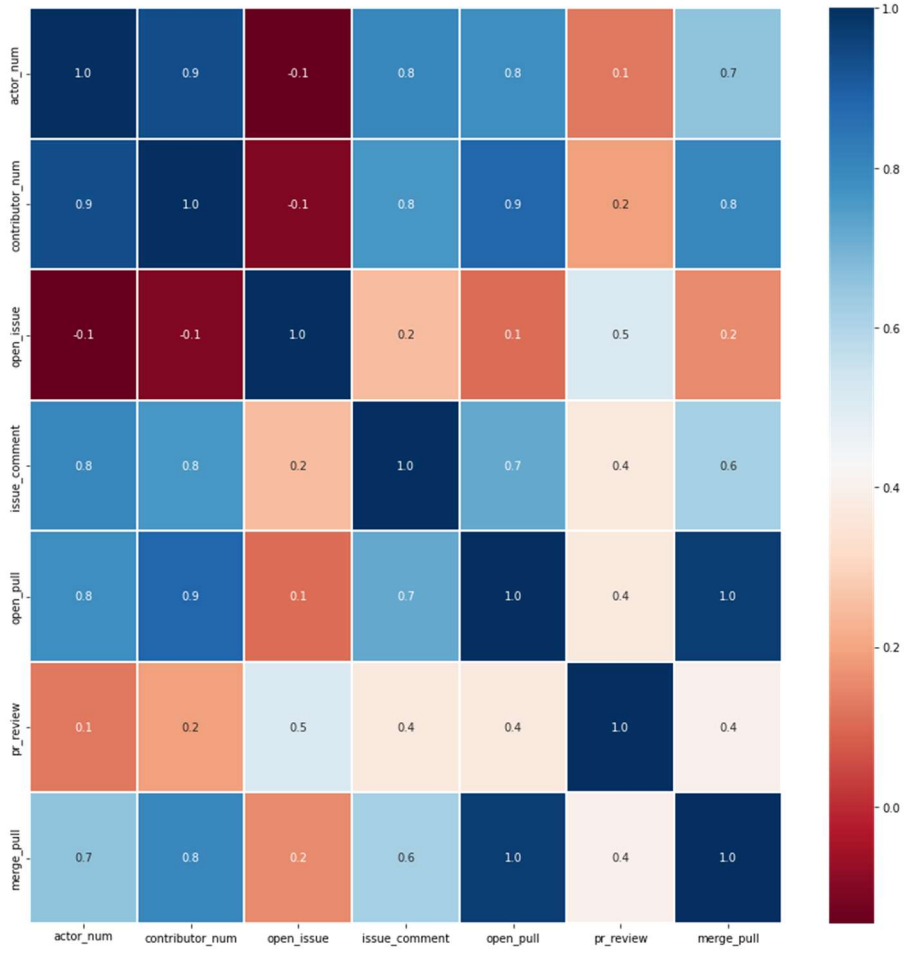

图 13. 变量相关系数图

	actor_num	contributor_num	open_issue	issue_comment	open_pull	pr_review	merge_pull
actor_num	1.000000	0.933156	-0.145859	0.799029	0.781956	0.125314	0.659605
contributor_num	0.933156	1.000000	-0.101499	0.759240	0.881650	0.189896	0.801504
open_issue	-0.145859	-0.101499	1.000000	0.248456	0.105136	0.510614	0.156818
issue_comment	0.799029	0.759240	0.248456	1.000000	0.719206	0.373047	0.615147
open_pull	0.781956	0.881650	0.105136	0.719206	1.000000	0.370713	0.971714
pr_review	0.125314	0.189896	0.510614	0.373047	0.370713	1.000000	0.398076
merge_pull	0.659605	0.801504	0.156818	0.615147	0.971714	0.398076	1.000000

图 14. 相关系数表

从上面的相关性图表中可以发现，项目贡献者与 actor_num, open_pull,merge_pull 相关性较大，而另一方面，项目总人数与 contributor_num, issue_comment,open_pull 相关性较大。但这仅仅是由皮尔逊相关系数得出的结论，后续还需要进行模型的建立、训练与评估。

模型选择工业界常用机器学习模型 XGBoost[3],首先加上一个字段“Centralization”作为项目贡献者比率

actor_num	contributor_num	Centralization
91	22	0.241758
97	15	0.154639
78	19	0.243590
61	18	0.295082
70	26	0.371429
81	25	0.308642
108	28	0.259259
110	28	0.254545
126	35	0.277778
105	35	0.333333

图 15.项目贡献者比率字段

输入的特征空间及真实标注数据如下图：

	open_issue	issue_comment	open_pull	pr_review	merge_pull
0	97	487	93	504	66
1	149	521	93	573	63
2	105	340	141	657	123
3	107	279	107	540	93
4	202	478	160	959	130

```
0      0.241758
1      0.154639
2      0.243590
3      0.295082
4      0.371429
         ...
110    0.204167
111    0.184825
112    0.123947
113    0.186649
114    0.027174
Name: Centralization, Length: 115, dtype: float64
```

图 16.特征空间和真实标注表

3.2.4 项目贡献者比率模型训练、预测与评估

对模型的初始参数设置如下：

booster='gbtree

gamma=0

importance_type='gain’,

learning_rate=0.1

max_depth=15

min_child_weight=1

n_estimators=100

objective='multi:softprob’

random_state=0,

reg_alpha=0,

reg_lambda=1

subsample=0.5

同样的，先选择 PaddlePaddle/Paddle 项目 2016.8-2021.11 的数据（114 条）输入模型，设置训练集测试集划分比为 4:1，进行训练，训练结束后对测试集进行预测

预测结果：

0.18620503, 0.21260186, 0.20847043, 0.28600746, 0.25138202,

0.2073116 , 0.2002323 , 0.16793469, 0.22900482, 0.05378718,

0.30712324, 0.17510937, 0.19454642, 0.16019304, 0.16600634,

0.2002323 , 0.12423189, 0.1938865 , 0.18715791, 0.17703095,

0.21260186, 0.18718275, 0.21249059

真实结果如下：

0.186649,0.184971,0.203980, 0.286667,0.243590,0.204473,

0.197044,0.167845,0.231959,0.027174,0.308642,0.174334,

0.195900,0.154639,0.165829,0.197044,0.123947,0.193333,

0.184825,0.179487,0.184971,0.183406,0.214623

对比大致可以发现预测结果较好，进一步进行评估：选择平均绝对误（MAE）,均方根误差(RMSE),解释回归模型的方差得分(explained_variance_score)三个指标进行评估，结果是平均绝对误差 MAE：0.005584051285284322；均方根误差 RMSE：0.010257772374569311；解释模型

方差得分 （explained_variance_score）：0.9691887530554696。（注：解释回归模型的方差得分，其值取值范围是[0,1]，越接近于 1 说明自变量越能解释因变量的方差变化，值越小则说明效果越差）可以发现用小数据集训练出来的模型效果已经比较好了。然而小数据集可能存在过拟合的现象，即对大量数据的泛化能力不是特别好，因此扩充数据集是必要的。

我们加入了更多高健康度优质项目的数据,覆盖深度学习与前后端等多领域,更具有普适性与一般性。加入项目及时间如下：

项目/仓库名	开始时间
PaddlePaddle/Paddle	2016.8
PaddlePaddle/models	2017.6
apache/tvm	2018.2
apache/echarts	2015.1
ant-design/ant-design	2015.6
NervJS/taro	2018.6
ApolloAuto/apollo	2018.1

注:结束时间均为 2021.11

部分项目预览图：

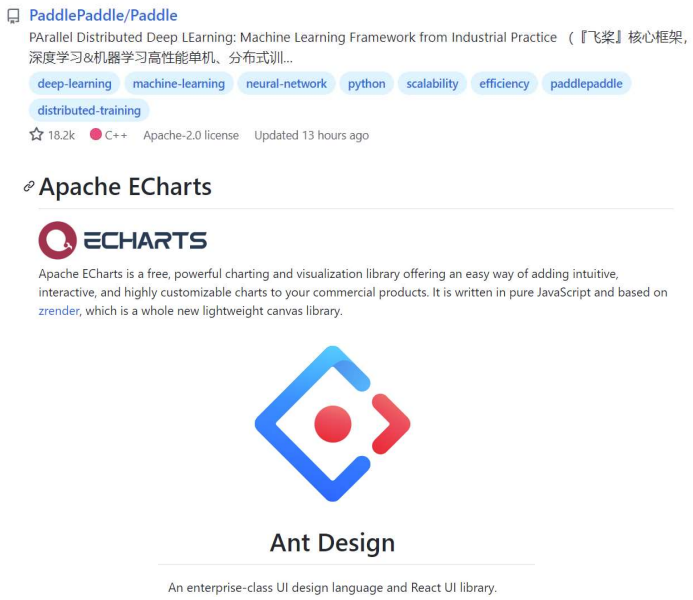

图 17.部分项目预览图

扩充数据集后重新训练,预测,评估， 结果：

平均绝对误差 MAE:0.0015791919422017636,

均方根误差 RMSE：0.039739048078706714

explained_variance_score 解释回归模型方差得分:0.9993514673762636

由这几个评估参数可以发现，扩充数据集后模型效果有了显著增强。模型进一步优化需要进行调参，调参方法使用网格化搜索，即固定其他参数只调整一个参数，将该参数设置在合理范围内并进行遍历,每次训练预测评估产生结果,取最好的结果及对应的参数即可。

$$优化：\theta_k = \underset{\theta_k}{argmin} \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_{pred}^{(i)} - y_{test}^{(i)})^2}$$

$\theta_k \in [\theta_{k_{min}}, \theta_{k_{max}}]$, θ_j不变,$j=1,2,\cdots N, j \neq k$, N为参数个数

例如：首先调整 gamma（XGBoost 在分裂节点时都会看分裂后损失函数的增益，只有增益大于一个阈值，才会对节点进行分裂。该参数指定的就是那个阈值，该参数越大，则表示决策树越难进行分裂，也就意味着算法越保守）将 gamma 设置为[0,1),对每个 gamma 进行训练，训练测试评估结果：

0.9995881019409544,0.9422003893638935,0.8977215349200939,

0.8493242395378537,0.8105942428419968,0.7676911372982713,

0.6966267309779958,0.6966267309779958,0.6150394103732664,

0.6150394103732664

可以发现 gamma 为 0 的时候模型效果最好,并且 gamma 参数对模型性能有极大的影响，因此取 gamma=0，其余参数调整同理，最终结果如下：

Max_depth=6

Min_child_weight=1

Gamma=0

Learning_rate=0.06

Reg_alpha=0

Reg_lambda=1

Subsample=0.9

N_estimators=120

最终优化后回归方差得分为 0.9999380340909436，虽然只有在万分位有一点提升，但是在更大的数据集的表现可能就会更好

四． 评价

本次项目选择从项目参与的人口特征切入，对项目参与人员进行分类，并基于人员的角色分类，进一步探究了项目的核心人员的协作关系，项目的贡献人员增长率，从而提出项目健康度的评估指标，然后运用机器学习的方法进行模型建立，预测，评估与优化，从而探究了项目健康度评估指标与其它特征的关系。该项目的切入角度具与传统的项目健康度评估指标略有不同，人口角度的设置在某种层面上具有创新意义与探索价值。

本次项目的人员分类部分作为从人口角度切入的重要工作与突破口，实现了将 github 项目人员进行分类，从而为后续的人员模式的探究，包括项目健康度指标的建立与评估奠定了基础，具有一定的借鉴意义。

但该分类同样有问题与改进点。首先，人员分类仅仅考虑了简单的用户身份数据与行为数据，没能深入挖掘。如果能做到从多维度对人员进行精准定位，比如做到能判断每个开发者的不同的技术领域专长，据此对这些开发者打上相应标签。这样得出的分类意义更大，分类结果可以用于向开发者推荐项目，帮助开发者发现感兴趣的社区项目，或者也可以用于向社区项目中的开发者是否合适项目的开发任务，是否缺乏相应专长的开发者，从而得以评估项目的健康度，并据此为项目推荐人才，让项目针对性的吸纳领域人才和所缺乏的人才，帮助社区核心维护者更好的进行项目管理。

在项目健康度指标中，项目协作度指标中协作关系网络的建立只选取了简单的公共修改文件作为协作关系，实则有许多特征可以作为协作关系，比如使用成员之间 issue,pr 的协作情况等等，可以对项目整体协作情况有更深的分析。此外，在对大型的开源项目的协作关系网络构建时，由于计算时会产生笛卡尔积，其数据规模会十分庞大，如何对大规模数据计算是一个难点。

项目健康度的另一个评估指标项目贡献者比率的选取有如下优点：一个开源项目传统的贡献者（contributor）主要需要用户的代码贡献，而相对于传统定义，本项目的项目贡献者的定义更加普遍，即参与到社区中的所有开发者，包括提交 bug、参与讨论、参与代码 review 的开发者事实上都对项目是有贡献的,有些意见可能没有被 merge（由于非建设性意见等原因），但是这些意见

的提出者是思考者或讨论者，同样可以被分类到贡献者里去，因此该项目贡献者的定义更加具有一般性和普适性。

但该比率同样有不足：项目贡献者比率越高，在某种意义上代表项目贡献人数很多，讨论度很高，可能该项目是一个成熟的健康项目，但在另一种层面上更可能是项目并没有广泛的受众，即项目贡献者都是维护项目稳定发展的核心开发人员，并且项目的总参与人数很少，导致项目的贡献者比率很高，这样的项目并不能说是一个高健康度的项目。而由于初始就有意识地选择了高健康度的项目进行评估，因此该项目本身的目的就是发掘高健康度项目的人员模式和人口变化情况，在这一方面这样的评价指标是可行的，但是在评估其他非高健康度，不是特别稳定的小项目中，用单一的项目贡献者比率就有待商榷，可能需要更多的聚合指标进行深入评估。但就该项目来说还是可行的。

用项目健康度比率生成的机器学习模型，有如下优点：首先，该模型选用了优化的分布式梯度增强库 XGBoost,改模型具有十分优秀的性能；其次，模型的训练使用了多领域的项目数据，具有一般性与普适性，且数据量在一定程度上减小了过拟合，同时在一定程度上增强了在未知数据集上的泛化能力；再次，模型数据集扩充与网格化搜索的调参让其效果较好，在测试集上达到了 0.9999 的结果。

模型不足和后续的优化：首先模型的不足与上述项目贡献者比率的不足类似，即可能在非高健康度的数据集上效果不是特别好，这首先要优化项目健康度评估指标，使用多重聚合评估指标；其次是数据集仍可以进行扩充，在算力允许的条件下可以尽量扩充数据集，进一步增强模型的泛化能力，在优化项目健康度评估指标的情况下增加不同健康度层次项目数据，使得模型对任何健康度的项目都可以进行评估；最后是在选取模型的过程中，可以多模型进行比较，比如将 LightGBM,XGBoost 等模型分别训练进行比较，甚至可以多种模型训练结果进行集成，取加权平均等。

总之，项目整体上来说符合社会计算这门课将社会科学与计算科学、数据科学等多领域交叉融合的方向，从一个具体角度探究了 github 人员模式和项目健康度等内容，综合运用数据预处理、数据可视化、机器学习等方法，具有一定的探索价值。同时后续也有很多工作可以继续进行探索研究。

五． 相关工作

Linux 基金会有一个开源软件社区健康分析项目组 CHAOSS（The Community Health Analytics Open Source Software），致力于为评估开源社区和项目发展情况提供量化指标，包括社区健康指标体系的制定、开源指标数据采集与开源分析工具，以及全球范围内社区工作组的建设等。表 4 是其 19 年 8 月份发布的一些指标。

指标项	含义
新建 issue	在一个时间段内，有多少新建的 issue？
issue 活跃度	在一个时间段内，有多少 issue 是保持活跃的？
关闭的 issue	在一个时间段内，有多少是已经处理完毕的 issue。
Elephant Factor	Community 的工作分布均匀程度如何？ 公司成员、个人开发者、基金会成员等的分布。
代码变更行数	在特定时间段内对源代码进行的所有更改中的代码行数（添加、删除）的总和是多少？
PR /MR 审核者情况	Community 中谁在积极审查贡献？

表 4. CHAOSS19 年 8 月发布的健康度指标表

六． 结论

本次项目选择从项目参与的人口特征切入，首先对项目参与人员进行分类，基于对人员行为事件数据，实现将项目一定时段内的人口划分为机器人、核心开发者、外围开发者三类，其中外围开发者又可以分类为问题报告者和代码贡献者两类。

此外，基于人员的角色分类，又进一步探究了归纳出了两个项目健康度指标。一是项目协作度，是以核心成员间公共修改文件来建立协作关系，构建项目成员协作网络，并给出项目协作度评估指标来表征项目内成员协作的均衡度。二是项目的贡献人员增长率，并运用机器学习的方法进行模型建立，预

测，评估与优化。最终综合之下，完成了本次基于人员的项目健康度评估指标的建立。

参考文献

[1]开源环境下开发人员行为特征挖掘与分析[J]. 袁霖,王怀民,尹刚,史殿习,李翔. 计算机学报. 2010(10)

[2]软件仓库挖掘领域:贡献者和研究热点[J]. 江贺,陈信,张静宣,韩雪娇,徐秀娟. 计算机研究与发展. 2016(12)

[3]Github 中开发人员的行为特征分析[J].李存燕,洪玫. 计算机科学. 2019(02)

附录：分工

丁正源：项目数据获取与预处理、项目人员分类、核心人员协作度部分

贾柏寒：项目数据获取与预处理、数据初步可视化探索、项目贡献者比率部分

附录：项目进度

4 月 25 号至 5 月 1 号：项目数据获取与预处理、初步可视化探索

5 月 2 号至 5 月 9 号：项目人员分类部分

5 月 10 号至 5 月 19 号：项目健康度评估部分

5 月 20 号至 5 月 27 号：项目收尾工作、ppt 报告制作、论文撰写