

基于logistic回归模型的Pull Request合并结果预测

陈萍¹, 周礼亮², 张卫丰³

(1. 南京医科大学图书馆, 江苏南京 211166; 2. 中国电子科技集团公司第十研究所, 四川成都 610000;
3. 南京邮电大学计算机学院, 江苏南京 210023)

摘要: Pull Request是GitHub中代码贡献的一种重要方法, 当开发人员希望将其代码更改从本地机器合并到存储项目中所有源代码的主存储库时, 需要提交Pull Request。基于logistic回归模型对Pull Request被拒绝情况进行预测实验, 预测实验中考虑的输入特征为通过数据挖掘关联规则找出的影响Pull Request被拒绝的相关因素, 主要包括修改变更的代码特性、Pull Request描述的文本特性、开发人员以前行为的贡献者特性以及Pull Request过程的交互等。实验评估了包含12个开源项目中140 155个Pull Request的有效性。结果表明, 基于logistic回归模型的预测结果准确率为0.84, 召回率为0.99, F1分数为0.91, 相较基线方法均有一定提高。分析预测实验结果表明, 通过数据挖掘中关联规则找出的影响因素对Pull Request合并结果具有足够的影响力, 可以帮助开发人员将精力集中在主要因素上, 或分配更多资源来克服关键问题, 有利于避免开发人员提交的Pull Request被拒绝, 减少项目成本和时间。

关键词: Pull Request; 影响因素; 数据挖掘; 回归模型; GitHub

DOI: 10.11907/rjdk.241096

中图分类号: TP311

文献标识码: A

开放科学(资源服务)标识码(OSID):

文章编号: 1672-7800(2025)004-0089-04



Prediction of Pull Request Combined Results Based on Logistic Regression Model

CHEN Ping¹, ZHOU Liliang², ZHANG Weifeng³

(1. Library, Nanjing Medical University, Nanjing 211166, China;
2. China Electronics Technology Group Corporation Tenth Research Institute, Chengdu 610000, China;
3. School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China)

Abstract: Pull Request is an important method of code contribution in GitHub. When developers want to merge their code changes from the local machine into the main repository that stores all source code in the project, they need to submit a Pull Request. Based on the logistic regression model, a prediction experiment was conducted on the rejection of Pull Requests. The input features considered in the prediction experiment were the relevant factors that affect the rejection of Pull Requests identified through data mining association rules, mainly including the code characteristics of modified changes, the text characteristics of Pull Request descriptions, the contributor characteristics of developers' previous behaviors, and the interaction of the Pull Request process. The experiment evaluated the effectiveness of 140 and 155 Pull Requests from 12 open source projects. The results showed that the prediction accuracy based on the logistic regression model was 0.84, the recall rate was 0.99, and the F1 score was 0.91, which showed some improvement compared to the baseline method. The analysis and prediction experiment results show that the influencing factors identified through association rules in data mining have sufficient influence on the merge results of Pull Requests, which can help developers focus their energy on the main factors or allocate more resources to overcome key problems, and is conducive to avoiding the rejection of Pull Requests submitted by developers, reducing project costs and time.

Key Words: Pull Request; influencing factors; data mining; regression model; GitHub

收稿日期: 2024-02-01

扫描二维码阅读全文:



基金项目: 国家自然科学基金面上项目(62272214); 南京市国际合作项目(202201010, 202401006)

作者简介: 陈萍(1979-), 女, 硕士, 南京医科大学图书馆副研究馆员, 研究方向为数据挖掘、智慧图书馆; 周礼亮(1982-), 男, 硕士, 中国电子科技集团公司第十研究所高级工程师, 研究方向为航空电子信息系统和传感器管理; 张卫丰(1974-), 男, 博士, CCF高级会员, 南京邮电大学计算机学院教授, 研究方向为代码管理、持续集成。本文通讯作者: 张卫丰。

0 引言

版本控制系统 (Version Control System, VCS) 是配置管理的主要策略之一, 允许开发人员通过开发和维护来控制对软件工件的更改^[1]。分布式版本控制系统 (Distributed Version Control System, DVCS) 的出现增强了支持协作软件开发的模式, 允许开发人员管理多个远程存储库, 实现多个团队在同一项目上协同工作, Git 即为一个典型案例, 其详细记录了软件的整个开发历史^[2]。DVCS 的一项重要功能是系统化规范外部团队成员与核心团队成员之间的协作方式^[3]。外部成员可以克隆存储库、更改代码, 然后请求将更改合并回主存储库, 这一过程是通过 Pull Request 完成的。Pull Request 中包含的贡献将由负责其分析的核心团队中任何成员选择接受或拒绝^[4-7]。随着开源项目的流行, 传入的 Pull Request 数量过多, 给负责接受或拒绝 Pull Request 的核心项目成员带来了沉重负担。核心项目成员需要花费较多精力逐个分析提交的 Pull Request 然后选择接受或拒绝。对于开发人员而言, 提交的 Pull Request 大多数会被拒绝, 且被拒绝后需要花费更多精力来修改并重新提交 Pull Request, 因此, 一种有效的 Pull Request 预测方法可以为核心项目人员在选择接受或拒绝提交的 Pull Request 时提供参考意见。

1 相关研究

Gousios 等^[8]、Rahman 等^[9]和 Tsay 等^[10]对基于 Pull Request 的软件开发模型进行了探索性研究, 试图了解进行合并的原因, 并在某些情况下预测 Pull Request 的合并。然而, 无论是针对项目、开发人员, 还是贡献本身, 以上研究均未探讨 Pull Request 固有的特性。Gousios 等^[8]尝试理解 Pull Request 的生命周期, 并分析导致其接受的因素以及完成该请求所需的时间。该项研究表明, GitHub 上的存储库中有 14% 使用了 Pull Request 策略, 且在 90% 的贡献中, 处理的文件不到 20 个, 修改的代码少于 500 行。此外, 关于 Pull Request 的生命周期, 研究显示 80% 的贡献在 3~7 天内被接受或拒绝。然而, 这项研究并没有说明如何针对 Pull Request 提高合并率。为了更好地使用开源软件环境中包含的信息评估 Pull Request, Tsay 等^[10]分析了与合并 Pull Request 相关的社会技术信号。结果表明, 包含测试代码的贡献更有可能被合并, 请求者、项目和负责分析贡献的开发人员之间的社会关系也会对合并决策产生积极影响, 修改文件数量和评论数量则有助于降低接受概率。尽管该项研究考虑了请求者、项目和请求本身固有的属性, 但并没有探索这些特性如何影响 Pull Request 的合并。Jiang 等^[11]提出一种 CTCPPre 方法来预测 GitHub 中接受的 Pull Request, 提取 17 个特征, 将其分为代码、文本、贡献者和项

目 4 个维度。其中: 代码特征衡量的是修改代码的特征; 文本维度从 Pull Request 的自然语言描述中提取特征; 贡献者维度提取 Pull Request 贡献者之前行为的特征; 项目特征主要分析项目最近的开发环境。所有特征均可以在提交 Pull Request 时立即自动提取。基于这 17 个特征, 利用 XGBoost 分类器预测接受的 Pull Request, 准确率为 0.82, AUC 为 0.76, F1-score 为 0.88; Mohamed 等^[12]提出一种跨项目方法, 将外部项目作为源项目来构建决策树训练模型, 从而预测另一个项目中重新打开的 Pull Request。该方法首先从源项目中提取修改后的代码特性、评估期间的审查特性以及贡献者的开发人员特性; 然后使用决策树分类器对目标项目中的 Pull Request 进行预测, 对包含 100 622 个 Pull Request 的 7 个开源项目进行了跨项目预测有效性评估。实验结果表明, 跨项目预测准确率在 78.76%~96.52% 之间, F1-score 在 53.34%~90.58% 之间。该研究还采用决策树预测器检查特征重要性, 发现 Pull Request 请求数量是大多数项目中最重要特征^[13-14]。

Pull Request 合并结果为二分类事件。相较于决策树模型和其他分类器模型, logistic 回归模型更擅长分析线性关系。基于以上研究, 本文采用 logistic 回归模型对 Pull Request 合并进行预测实验, 充分考虑通过关联规则找出的影响 Pull Request 合并的相关因素, 主要包括修改变更的代码特性、请求描述的文本特性、开发人员以前行为的贡献者特性以及开发环境的项目特性, 以期帮助开发人员将精力集中在主要问题上, 使其提交的 Pull Request 更容易被合并。

2 预测模型构建

基于 logistic 回归模型对 Pull Request 结果进行预测, 预测考虑修改变更的代码特性、Pull Request 描述的文本特性、开发人员以前行为的贡献者特性以及 Pull Request 过程的交互等因素。具体流程见图 1。

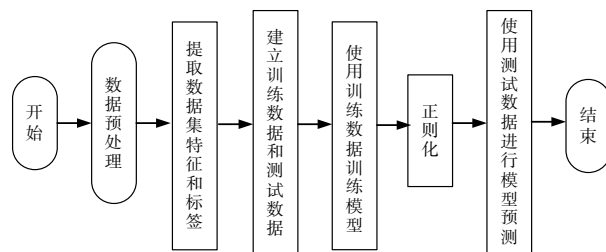


Fig. 1 Prediction model construction process

图 1 预测模型构建流程

3 实验方法

在实验时提出两个 RQ: ①RQ1。使用逻辑回归预测模型 Pull Request 合并结果的准确性如何以及模型性能如何?; ②RQ2。通过正则化优化模型的效果如何?

3.1 数据集

获取多个星星(Star)指数高的受欢迎的 GitHub 开源项目数据集,包括项目和仓库名称、项目中具体的 Pull Request 数量、项目贡献者数量以及星星数量。具体信息如表 1 所示。本文选择 Pull Request 数量多的项目进行实验。实验使用收集的相关因素数据,即每条 pull request 的项目名称、pull request Ids、pull request 是否合并、贡献者经验度、贡献者是否是第一次 pull request、贡献者的社会关系、pull request 描述相关信息、pull request 过程中贡献者评论、pull request 过程持续时间、pull request 过程中评审者评论、文件修改计数、代码行增加计数、代码行删除计数、信息提交数量等。将这些影响因素作为本文实验的输入数据。

Table 1 Pull Request project dataset information
表 1 Pull Request 项目数据集信息

序号	Repository	Pull request #	Contributors #	Stars/k
1	elastic/elasticsearch	52 569	428	57.0
2	tensorflow/models	3 286	384	71.7
3	webpack/webpack	5 426	409	59.7
4	nodejs/node	25 362	543	83.2
5	ohmyzsh/ohmyzsh	5 869	436	61.6
6	bitcoin/bitcoin	15 875	379	59.3
7	netdata/netdata	5 563	407	56.5
8	opencv/opencv	10 854	341	57.6
9	webpack/webpack	5 347	408	59.7
10	redis/redis	4 519	392	51.9
11	golang/go	1 917	474	92.5
12	gohugoio/hugo	3 568	428	55.0

3.2 数据处理

在得到项目中所有 pull request 的相关因素数据后,需要将其处理为数值格式。对于本身即为数值类的数据不作处理,直接作为输入数据,如 pull request Ids、贡献者经验度、pull request 描述相关信息长度、pull request 过程中贡献者评论、pull request 过程持续时间、pull request 过程中评审者评论、文件修改计数、代码行增加计数、代码行删除计数、信息提交数量等^[15-17]。因素合并结果分为两种:merged=true 将数据标记为 1,merged=false 则将数据标记为 0。是否为第一次 pull request 也分为 1 和 0 两种,1 表示是第一次 pull request,0 则表示不是第一次 pull request。贡献者社会关系分为 0、1、2 三种情况,0 表示为无关系,1 表示为项目贡献者,2 表示为项目核心开发人员。

在完成所有 pull request 相关因素数据的收集和处理之后,需要将它们按照 pull request 的相关因素数据排列为一行文本文件。本文将 14 万多条 pull request 相关因素数据处理输出为 14 多万行相关因素数值数据文本,作为后续实验的输入数据。部分相关因素数据如图 2 所示。

3.3 模型评价

Pull Request 合并结果分为成功合并和拒绝合并两种。logistic 回归模型将所有特征输入集合转化为 0 和 1 两类概率。逻辑回归的本质为线性回归,只是在特征到结果的映射中加入一层 Sigmoid 函数映射,即首先对特征线形求和,

1,41306,1,1,1,1,2,0,3,2.1259027777777777,41,0,291
1,41305,3,43,52,1,2,0,17,6.0674652777777778,42,26,
1,41299,1,31,39,3,1,0,2,7.4834027777777778,52,9,32
0,41298,13,5,281,1,2,0,4,0.11166666666666666,48,0
1,41295,1,3,3,1,2,0,4,0.6901041666666666,34,0,291
0,41291,2,13,13,2,2,0,8,2.1839351851851854,51,0,2

Fig. 2 Partial related factor data
图 2 部分相关因素数据

然后使用 Sigmoid 函数进行概率计算,再进行分类。
如表 2 所示,如果预测出 Pull Request 合并结果为成功,实际情况也是成功合并,则称为真正例 TP;如果预测出 Pull Request 合并结果为拒绝,而实际情况为被成功,则称为假反例 FN;如果预测出 Pull Request 合并结果为拒绝,而实际情况为被拒绝,则称为真反例 TN;如果预测出 Pull Request 合并结果为成功,而实际情况为拒绝,则称为假反例 FP。

Table 2 Prognostic and real label
表 2 预测和真实标签

标签	预测正例	预测反例
事实正例	TP(真正例)	FN(假反例)
事实反例	FP(假正例)	TN(真反例)

使用精确率(Precision,P)评价预测结果的准确程度,其为正例样本中真实为正例的比例。计算公式为:

$$P=TP/(TP+FP)$$
 (1)

使用召回率(Recall,R)评价数据预测的全面程度以及模型对正样本的区分能力,其为真实正例样本中预测结果为正例的比例。计算公式为:

$$R=TP/(TP+FN)$$
 (2)

F1 分数(F1-Score)是衡量二分类模型精确度的一种指标,同时兼顾了分类模型的精确率和召回率,可以看作是精确率与召回率的调和平均数。计算公式为:

$$F1-Score=(2*P*R)/(P+R)$$
 (3)

4 实验结果分析

首先提取数据集集中的第一列,即所有 Pull Request 合并结果数据作为 target y,合并结果 y 的值为 1 或 0 两种,1 表明合并成功,0 表明合并被拒绝;然后提取其他列的影响因素数据作为 data x,将数据拆分为 70:30 的标准格式,以便在建模过程中对其进行训练与测试,其中训练数据占总数据的 70%,测试数据占 30%。将控制随机化参数的 random_state 设置为 420。模型评价指标如表 3 所示。可以看出,本文模型的精确率达到 0.84,召回率达到 0.99,F1-score 达到 0.91。表明输入的各项特征对 Pull Request 合并被拒绝这一结果具有足够的影响力。

过拟合现象是指对训练数据预测效果很好但对未知数据预测不理想的现象。正则化是一种常用的防止机器学习模型过拟合的技术。本文分别采用 L1 范式和 L2 范式

Table 3 Model evaluation index data

表3 模型评价指标数据

指标	精确率	召回率	F1-score
merge=0	0.84	0.99	0.91

进行正则化,得到两种正则化在参数0~1.5之间测试集和训练集的预测准确性分数。通过可视化绘图matplotlib库进行显示,结果如图3所示。可以看出,通过L1正则化的模型波动较小,这是由于其对模型数据的压缩程度较高;而L2正则化对模型数据的压缩程度较低,因此模型波动较大。

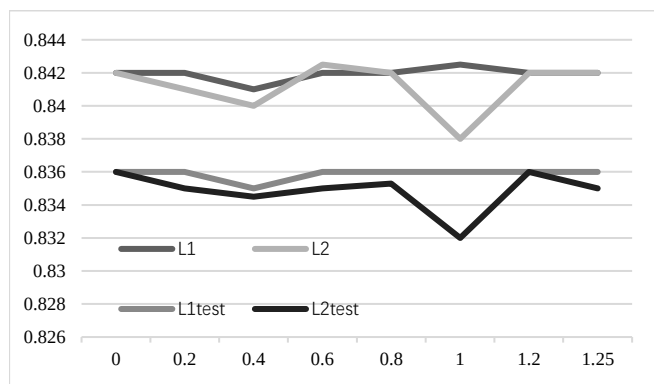


Fig. 3 Regularization result

图3 正则化结果

5 结语

本文通过构建基于 logistic 回归模型的预测方法,系统分析了影响 Pull Request 合并结果的相关因素,包括代码特性、文本描述特性、贡献者行为特性及交互特性等。实验结果表明,模型在准确率、召回率及 F1-score 方面均表现出色,说明以上影响因素对 Pull Request 合并结果具有显著影响力,能够为开发者提供重要参考,帮助其优化提交质量、降低拒绝率、提升协作效率。然而,本文研究仍存在一些不足之处:首先,特征选取主要基于现有文献与数据挖掘规则,可能遗漏了一些潜在的重要影响因素;其次,实验数据集中只包含部分热门开源项目,其结果可能无法完全适用于其他领域或小型项目;最后,本研究仅采用 logistic 回归模型进行预测分析,而未对其他机器学习模型的适用性进行深入探讨。未来研究可在以下方向进一步拓展:一是挖掘更多潜在的影响特征,尤其是复杂项目中的社会技术因素;二是引入更丰富的数据集,覆盖更多项目规模和领域,以提高研究的通用性;三是尝试结合深度学习或集成学习等先进模型进一步提升预测效果;四是探索动态特征建模方法,以适应开源项目中快速变化的开发环境。

参考文献:

- [1] ESTUBLIER J. Software configuration management: a roadmap [C]// Proceedings of the Conference on the Future of Software Engineering, 2000: 279-289.
- [2] SOARES D M, DE LIMA JUNIOR M L, MURTA L, et al. Acceptance factors of pull requests in open-source projects [C]// Proceedings of The 30th Annual ACM Symposium on Applied Computing, 2015: 1541-1546.
- [3] ZANDSTRA M. Version control with git[M]. Berkeley: Apress, 2021.
- [4] CHEN P, CHENG Y F. Analysis of constraints in the development of open source projects for university libraries based on GitHub [J]. Jiangsu Science and Technology Information, 2022, 39 (30): 75-80.
陈萍,程逸凡. 基于GitHub的高校图书馆开源项目开发制约因素分析[J]. 江苏科技信息, 2022, 39(30): 75-80.
- [5] CELINSKA D. Coding together in a social network: collaboration among Github users [C]// Proceedings of the 9th International Conference on Social Media and Society, 2018: 31-40.
- [6] YU Y, WANG H, YIN G, et al. Reviewer recommender of Pull-Requests in GitHub [C]// IEEE International Conference on Software Maintenance and Evolution, 2014: 609-612.
- [7] JIANG J, LO D, MA X, et al. Understanding inactive yet available assignees in GitHub [J]. Information and Software Technology, 2017 (91): 44-45.
- [8] GOUSIOS G, PINZGER M, DEURSEN A V. An exploratory study of the pull-based software development model [C]// Proceedings of the 36th International Conference on Software Engineering, 2014: 345-355.
- [9] RAHMAN M M, ROY C K. An insight into the Pull Requests of GitHub [C]// Proceedings of the 11th Working Conference on Mining Software Repositories, 2014: 364-367.
- [10] TSAY J, DABBISH L, HERBSLEB J. Influence of social and technical factors for evaluating contribution in GitHub [C]// Proceedings of the 36th International Conference on Software Engineering, 2014: 356-366.
- [11] JIANG J, ZHENG J T, YANG Y, et al. CTCPPre: a prediction method for accepted Pull Requests in GitHub [J]. Journal of Central South University, 2020, 27(2): 449-468.
- [12] MOHAMED A, ZHANG L, JIANG J, et al. Predicting which Pull Requests will get reopened in GitHub [C]// 25th Asia-Pacific Software Engineering Conference, 2018: 375-385.
- [13] HOFMANN T. Probabilistic latent semantic indexing [C]// Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1999: 50-57.
- [14] LAI S, JIN L, YANG W. Toward high-performance online HCCR: A CNN approach with DropDistortion, path signature and spatial stochastic max-pooling [J]. Pattern Recognition Letters, 2017 (89): 60-66.
- [15] HAO X, ZHANG G, MA S. Deep learning [J]. International Journal of Semantic Computing, 2017, 10(3): 417-439.
- [16] KOTSIANTIS S, KANELLOPOULOS D. Association rules mining: a recent overview [J]. International Transactions on Computer Science & Engineering, 2008, 32(32): 71-82.
- [17] PEDREGOSA F, VAROQUAUX G, GRAMFORT A, et al. Scikit-learn: machine learning in Python [J]. The Journal of Machine Learning Research, 2011 (12): 2825-2830.

(责任编辑:尹晨茹)