OSS プロジェクトへのオンボーディング支援のための Good First Issue 自動分類

Automatic Classification of Good First Issues for Onboarding to Open Source Projects

堀口 日向 * 大平 雅雄 †

あらまし 本研究の目的は，OSS プロジェクトの新規開発者向けの Issue を分類する機械学習モデルを構築し，メンテナのラベル付けの負担を軽減することである．結果は，Precision が 0.91，Recall が 0.30 となった（RQ1）．また，重要度が高い特徴量を分析し，GFI の分類には投稿者のプロジェクト内での役割が重要であることが分かった（RQ2）．

1 はじめに

オープンソースソフトウェア（OSS）プロジェクトは，コミュニティからのバグ報告やバグ修正，機能拡張といった貢献によって成り立っている [1]．コミュニティ開発者らの多くはボランティアとして活動しており，自由に開発に参加，離脱することができるため，一般的な OSS プロジェクトでは，プロジェクトを持続可能にするために常に新規開発者を求めている [2]．しかしながら，新規開発者はコミュニティに参加する際様々な障壁に直面し，プロジェクトから離脱する場合がある [3]．

新規開発者の直面する障壁の 1 つに課題選択の障壁がある [4]．多くの OSS プロジェクトでは貢献を行う際，開発者自身が課題票を作成するか，既知の課題の中から選択して取り組む．新規開発者にとっては，不慣れなプロジェクトの課題を理解し，対処するには多くの時間がかかる．

新規開発者がプロジェクトに参加する際の障壁を軽減するために，一部の OSS プロジェクトでは，新規開発者向けの課題に「Good First Issue（GFI）[1]」と呼ばれるラベルを付与している．実際に，GitHub 上で GFI ラベルを使用しているプロジェクトは過去 10 年間で増加しており [5]，GFI の多くは新規開発者によって解決されている [6] [7]．ただし，プロジェクトメンテナは課題を手動で選別しラベル付けを行う必要があり，メンテナにとって負担となる [2]．

本研究の目的は，新規開発者向けの課題を分類する機械学習モデルを構築することである．GitHub 上に存在する GFI ラベルが付与された Issue（以降 GFI と呼ぶ）と通常の Issue を収集し，ランダムフォレストを使って機械学習を行う．構築したモデルを評価するために，以下のリサーチクエスチョンに取り組む．

RQ1: Issue 投稿時に含まれる情報のみを利用して GFI を分類できるか？
メンテナの GFI ラベル付けの負担を軽減するため，分類モデルはメンテナが Issue の詳細を読みラベルを付与する前に，GFI かどうかを判定できる必要がある．そのため，モデルに入力する特徴量として利用できるのは，タイトルと本文，Issue 投稿者の情報のみであり，これらの特徴量を使って学習した際の分類モデルの性能を評価する．

RQ2: 分類に重要な Issue の特徴量は何か？
ランダムフォレストでは，ある条件で決定木のノード分割が行われたときに，不純度がどの程度下がるかを計算することで，ノード分割の条件に使われた特

*Hyuga Horiguchi, 和歌山大学
†Masao Ohira, 和歌山大学

[1] https://docs.github.com/en/issues/using-labels-and-milestones-to-track-work/managing-labels
[2] https://github.blog/2020-01-22-how-we-built-good-first-issues/

徴量が分類結果に与える影響を定量的に測ることができる．重要度が高い特徴量は，GFI と通常 Issue を分ける特徴を理解するうえで役に立つ．

2 分類モデルの構築

2.1 入力する特徴量

本モデルは，メンテナが手動で行っている GFI のラベル付けを自動化することを目的としているため，メンテナが Issue の内容を確認する前の，Issue が投稿された直後の情報を使って GFI かどうかを判定できることが求められる．そこで，Issue 投稿時に必ず含まれる，タイトル，本文，投稿者の情報のみを使って特徴量を考える．すなわち，Issue に対する他の開発者のコメントや，Issue に付与されるラベルの情報は特徴量として利用しない．モデルに入力する特徴量を表 1 に示す．

表 1: モデルに入力する特徴量

特徴量の名前		説明
Title	BoW-n	タイトルの Bag-of-Words（n はタイトルの語彙数）
	Words	タイトルに含まれる単語数
Body	BoW-n	本文の Bag-of-Words（n は本文の語彙数）
	Words	本文に含まれる単語数
AuthorAssociation		Issue 投稿者が持つプロジェクトに対する権限レベル

タイトルと本文 タイトルと本文は，主に英語で記述されているため，1 単語ずつに分割した後 Bag-of-Words としてベクトル化し，特徴量として利用する．また，タイトルと本文の文章の長さは Issue の内容が詳細に説明されているかどうかに関連するため，文章に含まれる単語数も特徴量として利用する．

投稿者 プロジェクトオーナーやメンテナは，新規開発者を惹きつけるために，GFI になりえる Issue を積極的に投稿する可能性がある．そこで，Issue 投稿者がプロジェクトに対して持っている権限レベルをカテゴリ変数として数値化する．

2.2 タイトルと本文の前処理

GitHub 上の Issue のタイトルや本文は，Markdown 形式で記述することができる．また本文には，コード片や URL，バージョン情報などが含まれる．これらの情報は Issue の内容を理解するのに役立つため，正規表現を用いて検出し特定の単語に置換する．またテキストデータに対しては，1 単語ごとのトークン化，小文字化，ストップワードと記号の除去，日付と数字の置換，レマタイズを行った．

2.3 分類モデル

分類モデルの機械学習アルゴリズムには，ランダムフォレストを利用する．モデルは各 Issue の表 1 の特徴量を入力とし，GFI か通常 Issue かの 2 値分類を行う．ランダムフォレストでは，弱学習器として複数の決定木を学習する．個々の決定木は，元のデータセットをブートストラップサンプリングしたデータによって学習し，その際全特徴量のうちランダムサンプリングした特徴量のみを利用する．

決定木の個数と各決定木の特徴量の個数はハイパーパラメータであるため，本研究では [8] で推奨されている値を参考に，決定木の個数を 200，各決定木の特徴量の個数を $\sqrt{F}$ とした．F は全特徴量の個数である．また本モデルの最終的な分類結果は，200 個の決定木のうち過半数が GFI と判定した場合，GFI として分類する．

2.4 不均衡データの学習

3.1 節で示すように，GFI は通常 Issue と比べてサンプル数が非常に少なく不均衡なデータセットになるため，クラスの重みづけを行う．決定木では，ノードを分割することで不純度が減少すると，その分割は分類に効果的であるとみなされる．そこで，不純度を計算する際に各クラスのサンプル数に反比例する重み $w_i = \frac{N}{2N_i}$（i はクラス，N は全サンプル数，N_i はクラス i のサンプル数）を掛けることで，サンプル数の少ない GFI の分類結果が不純度へ大きく影響するようになり，結果として GFI の分類性能が向上することを期待できる．

3 実験方法

3.1 データセット

GFI ラベルを運用しているプロジェクトから GFI と通常 Issue を収集するため，以下の条件を満たす GitHub 上の Issue を全て取得し，それぞれの GFI が属するプロジェクトを特定した．

- 「good first issue」ラベルが付与されている（大文字と小文字の区別はしない）
- 閉じられている（Close されている）

その結果，1 つ以上の GFI を保有している約 7 万件のプロジェクトを特定できた．これらすべてのプロジェクトから全ての Issue を収集するのは非常に時間がかかる上，GFI と通常 Issue のサンプル数の偏りを可能な限り小さくするため，GFI 数が 500 件以上の計 15 プロジェクトのみを対象とした．収集期間は各プロジェクト作成時から 2021 年 6 月 4 日までである．

3.2 評価

3.2.1 RQ1: Issue 投稿時に含まれる情報のみを利用して GFI を分類できるか？

データセットを 9:1 に分割し，9 割を使って 10 分割交差検証を行う．また，交差検証で学習したモデルの中から 1 つを使って残り 1 割のデータセットを分類し，混同行列と後の RQ2 で利用する特徴量の重要度を求める．評価指標には *Precision*，*Recall*，$F1$ を使用する．*Precision* は，モデルが GFI であると分類した Issue のうち，実際に GFI である Issue の割合を表す．*Recall* は，実際に GFI である Issue のうち，モデルが GFI であると分類した Issue の割合を表す．$F1$ は *Precision* と *Recall* の調和平均である．

3.2.2 RQ2: 分類に重要な Issue の特徴量は何か？

特徴量の重要度を用いて，最終的な分類結果に対する特徴量の相対的な重要性の評価を行う．重要度は，決定木のノードをある特徴量を用いて分割した際，その分割によって減少するノードの不純度と分割されるサンプル数のかけ合わせで求められる．ここで不純度は，ノードに異なるクラスのサンプルがどの程度混在しているかを表す．したがって，分割によって不純度を大きく減少させる特徴量であるほど重要度が高くなり，分割に影響するサンプル数が多いほど重要度が高くなる．重要度は決定木によって異なり，ランダムフォレストの場合は決定木が複数あるため，平均値を用いる．

4 結果

4.1 RQ1: Issue 投稿時に含まれる情報のみを利用して GFI を分類できるか？

層化 10 分割交差検証の結果を表 2 に示す．10 個のモデルでそれぞれの評価指標を計算するため，平均値と標準偏差を示す．*Precision* の 0.91 と比べて *Recall* は 0.30 と低く，これはモデルが GFI であると分類したうち 9 割は実際に GFI であるが，7 割の GFI を見逃しているということを示している．また，交差検証で最も F1 の高かったモデルで評価用データを分類した結果，*Precision* は 0.91，*Recall* は 0.31，$F1$ は 0.46 で，混合行列は表 3 となった．

表 2: 10 分割交差検証の結果

Metrics	Mean	Std.
Precision	0.91	0.01
Recall	0.30	0.01
*F*1	0.45	0.01

表 3: 評価用データの分類結果

		分類結果	
		GFI	通常 Issue
実際の	GFI	380	843
クラス	通常 Issue	38	14,867

4.2 RQ2: 分類に重要な Issue の特徴量は何か？

表 4: 重要度上位 10 位までの特徴量

特徴量の名前	重要度		
AuthorAssociation	0.0156	Body BoW-issue	0.0044
Body Words	0.0115	Body BoW-TIMESTAMP	0.0042
Title Words	0.0095	Body BoW-please	0.0039
Body BoW-INLINE_CODE	0.0065	Body BoW-VERSION	0.0037
Body BoW-URL	0.0056	Body BoW-add	0.0037

特徴量の重要度を表 4 に示す．全特徴量の重要度の合計が 1 になるように正規化してある．また，テキストデータの Bag-of-Words の特徴量は，名前の「BoW-」の後に続く単語の重要度となっている．

最も重要度が高かったのは AuthorAssociation で，2 番目の Body Words に比べて 1.4 倍重要度が高かった．AuthorAssociation は，Issue 投稿者がプロジェクトに対してどのような権限を持っているかを示す特徴量で，Issue 投稿者のプロジェクト上での役割を意味する．プロジェクトのオーナーやメンテナは，プロジェクトの保守・運用のため新規開発者を惹きつけることに積極的であると考えられ，GFI になりえる Issue を通常の開発者よりも多く投稿している可能性がある．

重要度が 2，3 番目の本文とタイトルの単語数は，2.1 節で述べた通り，Issue の内容が詳細に説明されているかを表す特徴量である．また，Issue 本文に含まれるインラインコードや URL，バージョン情報の有無も重要度 10 位以内の特徴量に含まれており，これらは投稿された Issue の内容を深く理解するために重要な情報であるといえる．これらの情報が含まれていない Issue は，問題を理解することが難しいか，機能拡張の提案など追加の議論が必要な Issue である可能性があり，新規開発者が取り組むべき Issue としては不向きであると考えられる．

5 議論

5.1 本モデルの有用性

4.1 節で示した通り，本モデルは *Precision* が 0.91 であるのに比べて *Recall* は 0.30 と非常に低い．*Precision* が高いということは，GFI と分類された Issue の中に通常 Issue が紛れ込むことは少なく，確実に新規開発者に適した Issue を分類することができることを意味する．しかしながら，*Recall* が低いということは，GFI になりえる Issue を見逃しやすいため，新規開発者がプロジェクトに貢献する機会が減る可能性がある．Tan ら [5] によれば，GFI ラベルはメンテナがサポート可能な Issue に対してラベル付けされるため，実際には多くの Issue を GFI に分類できたとしても，メンテナの手が回らず新規開発者はサポートが得られない可能性がある．したがって，多少 *Recall* を犠牲にしてでも *Precision* が高いほうが価値がある．

5.2 GFI の特徴

5.2.1 AuthorAssociation の分布

表 5: GFI，通常 Issue 投稿者の AuthorAssociation の分布

AuthorAssociation	GFI[件] (割合)	通常 Issue[件] (割合)
プロジェクトのオーナー	**662 (0.05)**	**15 (0.00)**
プロジェクトを所有する組織のメンバー	3,662 (0.30)	21,873 (0.14)
プロジェクトの共同作業に招待された開発者	772 (0.06)	2,488 (0.02)
過去にこのプロジェクトに コミットしたことがある開発者	4,560 (0.37)	61,392 (0.41)
上記のどれにも当てはまらない開発者	2,727 (0.22)	65,227 (0.43)
合計	12,383	150,995

特徴量の重要度が 1 番高かった AuthorAssociation の分布を表 5 に示す．通常 Issue の合計件数（150,995 件）は GFI の合計件数（12,383 件）より圧倒的に多いにもかかわらず，プロジェクトオーナーが通常 Issue を投稿した件数はわずか 15 件であった．一方，GFI は 662 件投稿されていた．また，GFI の 41%がプロジェクトに対して何らかの権限を持っている開発者（表 5 の上から 3 行分）から投稿されたのに対して，通常 Issue は 16%だけだった．したがって，プロジェクトのコアメンバーは，通常のプロジェクトの課題よりも新規開発者向けの課題に対して多くの労力を割いており，本モデルを利用して Issue の中から新規開発者向けの課題を自動で抽出することで，メンテナの負担を軽減できる．

5.2.2 タイトルと本文の単語数の分布

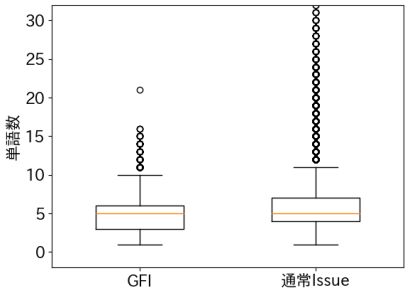

図 1: タイトルの単語数の分布

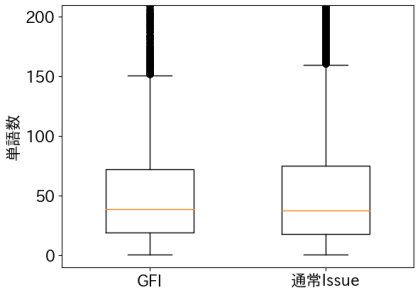

図 2: 本文の単語数の分布

特徴量の重要度が 2，3 番目に高かった，タイトルと本文の単語数（特徴量の名前は Title Words と Body Words）の箱ひげ図をそれぞれ図 1，2 に示す．図 1 の箱ひげ図は，つぶれて見にくくなるのを防ぐために，縦軸の最大値を 30 単語に制限した．同様の理由で，図 2 の縦軸の最大値を 200 単語に制限した．タイトルの単語数の中央値は，GFI は 5 単語で通常 Issue も 5 単語だった．また，本文の単語数の中央値は，GFI は 39 単語で，通常 Issue は 38 単語だった．GFI と通常 Issue で分布に大きな差がないにもかかわらず，特徴量の重要度が高くなった理由については，次の 2 つが考えられる．1 つは，タイトルや本文の単語数に加えて他の特徴量と組み合わせることで，分類性能が向上している場合である．もう 1 つは，外れ値を持つデータを分類する場合，分割時にノードに含まれる他クラスのデータがなく不純度

が下がりやすいため，必然的に重要度は高くなる場合である．図 1，2 を見ると外れ値を含むデータが多いことが分かるため，後者の場合を想定し，四分位範囲の 1.5 倍を超える単語数を持つデータを外れ値としてデータセットから除外し再度学習を行ったところ，*Precision* と *Recall* は変化しなかったが，タイトルと本文の単語数の重要度が上位 10 位から外れた．したがって，タイトルと本文の単語数は外れ値をとるデータを分類するためには有効であるが，GFI を特徴づける要素ではなかった．

5.3 制約

5.3.1 GFI に関連する類似ラベル

本研究では，データ収集時に「beginner friendly」や「easy bug fix」といった「good first issue」に類似するラベルを含めなかった．これらの類似ラベルは「good first issue」と比べて使用頻度が低く [5]，特定のプロジェクトでしか使われないことが多いため，「good first issue」ラベルのみを収集した．

5.3.2 収集した OSS プロジェクトの妥当性

データ収集の対象となった 15 プロジェクトのうち 2 件は，Issue の大半を GFI が占めており，通常 Issue はほとんど存在しなかった．これらのプロジェクトを目視したところ，一般的に利用されることを想定したソフトウェア開発プロジェクトではなかったため，新規開発者が OSS プロジェクトに貢献する際の主要な動機であるスキルアップにはつながらない可能性が高く，多くの新規開発者にとっては GFI として不適当かもしれない．ただし，本研究では最初に 3.1 節で決めた条件にしたがってデータ収集を行い，恣意的にプロジェクトを除外することは避けた．

6 まとめ

本研究では，GFI を分類するランダムフォレストモデルを構築し，*Precision*0.91，*Recall*0.30 を達成した（RQ1）．また，重要度が高い特徴量について追加分析を行い，GFI は通常 Issue に比べてプロジェクトの保守や運用にかかわる開発者らによって投稿されていることが多いということが分かった（RQ2）．つまりプロジェクトメンテナは，通常のプロジェクトの課題よりも新規開発者向けの課題に対して多くの労力を割いており，本モデルを利用して Issue の中から新規開発者向けの課題を自動で分類することは，メンテナの負担を軽減するために意義があるといえる．

現状のモデルは *Recall* が低いため，GFI になりえる Issue を多く見逃している可能性がある．今後は，*Recall* が低い原因を分析し向上させることを目指す．

参考文献

[1] Kevin Crowston, Hala Annabi, and James Howison. Defining open source software project success. In *ICIS '03*, pp. 327–340, 2003.

[2] Andrea Forte and Cliff Lampe. Defining, understanding, and supporting open collaboration: Lessons from the literature. *American Behavioral Scientist*, Vol. 57, No. 5, pp. 535–547, 2013.

[3] Igor Steinmacher, Tayana Conte, Marco Aurélio Gerosa, and David Redmiles. Social barriers faced by newcomers placing their first contribution in open source software projects. In *CSCW '15*, pp. 1379–1392, 2015.

[4] I. Steinmacher, T. U. Conte, and M. A. Gerosa. Understanding and supporting the choice of an appropriate task to start with in open source software communities. In *HICSS '15*, pp. 5299–5308, 2015.

[5] Xin Tan, Minghui Zhou, and Zeyu Sun. A first look at good first issues on github. In *ESEC/FSE '20*, pp. 398–409, 2020.

[6] Adriaan Labuschagne and Reid Holmes. Do onboarding programs work? In *MSR '15*, pp. 381–385, 2015.

[7] Hyuga Horiguchi, Itsuki Omori, and Masao Ohira. Onboarding to open source projects with good first issues: A preliminary analysis. In *SANER '21*, pp. 501–505, 2021.

[8] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, second edition, 2017.