相関ルール分析とロジスティック回帰分析を用いた fault-prone モジュール予測手法の提案

A Hybrid Faulty Module Detection Using Association Rule Mining and Logistic Regression Analysis

亀井 靖高*　森崎 修司†　門田 暁人‡　松本 健一§

あらまし
Fault-prone モジュールの要因分析と判別精度向上を目的として，モジュールのソースコードメトリクスを入力とした相関ルール分析とロジスティック回帰分析を用いたモデルの組み合わせ手法を提案する．提案手法では，相関ルール分析の特徴である fault の発生要因の理解が容易な点と，ロジスティック回帰分析の網羅的な予測が可能な点とを組み合わせる．評価実験として，NASA/WVU の公開している 2 つのデータセットから fault-prone モジュールを予測したところ，全てのモジュールの fault-prone が予測できること，予測精度の評価値である F1 値が最大 0.16 向上すること，を確認した．

1　はじめに

ソフトウェアテストおよび保守工程において，fault-prone モジュール（fault を含む確率の高いモジュール）[7] を特定しテスト工数を重点的に割り当てることで，テストの効率化，および，信頼性の向上が見込まれる [4]．そのために，モジュールから計測されたメトリクス（プログラム行数，サイクロマティック数，変更行数など）を説明変数とし，モジュールの fault の有無を目的変数とする fault-prone モジュール判別モデルが多数提案されている [3] [5] [7]．代表的なモデルとして，ロジスティック回帰分析がある [5] [9]．ロジスティック回帰分析は，与えられたデータセットから「説明変数の変動によって目的変数が 2 値のどちらを取る確率が高いか」を出力する回帰式を求める手法であり，得られた回帰式によって任意のモジュールの fault あり／なしを判別できる．

一方，非モデルベースの方法として，相関ルール分析を用いた fault-prone モジュール判別方法も提案されている [8]．相関ルール分析は，与えられたデータセットから「ある事象 X が発生した場合に（高確率で）別の事象 Y が発生する」という事実を相関ルールとして抽出する手法であり，データ間の隠された関係を発見できる．相関ルールを用いた fault-prone モジュール判別の特長は，fault の発生（事象 Y）につながる前提条件（事象 X）がルール中に明示されるため，判別の根拠を知ることが容易な点にある．さらに，ルールの信頼度，支持度などの指標値を利用することで，判別精度の高いルールのみを抽出できることも特長である [8]．ただし，相関ルール分析を利用した fault-prone モジュール判別では，全てのモジュールを判別できるとは限らない．モジュールによっては，前提条件が一致する相関ルールが存在せず，判別できない場合があるためである．

本稿では，原因分析が容易である相関ルール分析による予測と，全てのモジュールを網羅的に予測できるロジスティック回帰分析を組み合わせる手法を提案する．評価実験では，相関ルールの出現頻度や前提条件と結果の結びつきを示す支持度，信頼度，リフト値を用いて，予測に用いる相関ルールが満たすべき条件を実験的に調

*Yasutaka Kamei, 奈良先端科学技術大学院大学
†Shuji Morisaki, 奈良先端科学技術大学院大学
‡Akito Monden, 奈良先端科学技術大学院大学
§Ken-ichi Matsumoto, 奈良先端科学技術大学院大学

べる．

2　提案手法

本稿では，相関ルール分析とロジスティック回帰分析を組み合わせる上で，信頼度，支持度などの相関ルール抽出の指標値に着目する．信頼度，支持度などの指標値が一定の値以上の相関ルールを用いることで，前提部と結論部の結びつきが強い相関ルールを予測に用いることができる．そのため，各指標値が一定以上で，かつ前提部の一致する相関ルールが存在する場合は，相関ルールを用いた判別をロジスティック回帰分析より優先的に行う．

相関ルールは「X（前提部）$\Rightarrow Y$（結論部）」と表す．例えば,「$(4 <$ サイクロマティク複雑度 $< 5) \wedge (3 <$ fan-in $< 5) \Rightarrow$ fault」のように表す．このルールでは，モジュールのサイクロマティク複雑度が 4 より大きく 5 以下で，かつ，fan-in が 3 より大きく 5 以下のモジュールの fault を予測することができる．対象データ全体を D，D に含まれる 1 モジュールを T とする．

2.1　相関ルール抽出の指標値

相関ルール抽出の指標値 [1] として以下がある．

- **支持度**　対象データにおける相関ルールの出現頻度であり，$support(X \Rightarrow Y)$ と表記し，$support(X \Rightarrow Y) = s/n$ である．ただし，$s = \big|\{T \in D | X \subset T \cap Y \subset T\}\big|$, $n = \big|\{T \in D\}\big|$.
- **信頼度**　前提部 X が満たされたときに同時に結論部 Y も満たされる割合であり，$confidence(X \Rightarrow Y)$ と表記し，$confidence(X \Rightarrow Y) = s/y$ である．ただし，$y = \big|\{T \in D | X \subset T\}\big|$.
- **リフト値**　前提部 X によりどのくらい結論部 Y が満たされやすくなっているかを示しており，$lift\ (X \Rightarrow Y)$, と表記され，$lift(X \Rightarrow Y) = \frac{confidence(X \Rightarrow Y)}{z/n}$ である．ただし，$z = \big|\{T \in D | Y \subset T\}\big|$.

2.2　モデルの構築と判別手順

提案する組み合わせ手法による，モデルの構築からそのモデルによる判別の流れを図 1 に示す．まず，図 1(a) に示すようにモデル構築用のデータセット（フィットデータ）を用いて，ロジスティック回帰分析によりロジスティック回帰式と，相関ルール分析により相関ルールの集合 R とを抽出する．そして，相関ルールの集合 R に対して，条件に一致する（閾値以上の支持度，信頼度，もしくはリフト値である）ルールのみを選択する（相関ルール集合 R'）．

次に，図 1(b) に示すように，予測対象のモジュールのメトリクス値が相関ルールの前提部に一致するかどうかを判別する．モジュールの予測に相関ルールを用いる場合，前提部が一致するルールの数が 1 つ，もしくは複数以上が一致することが考えられる．前提部が一致する相関ルールが 1 つの場合は，その相関ルールの結論部を判別結果とする．複数以上ある場合は，多数派群（相関ルールの結論部 Y は 0 か 1 となるが，相関ルール集合 R' に多く含まれる側）の結論部を判別結果とする．前提部が一致する相関ルールが存在しない場合は，すべてのモジュールが判別可能なロジスティック回帰分析により判別する．

3　評価実験

3.1　概要

実験の目的は，相関ルール抽出の指標値と予測精度の関係を明らかにすることである．具体的には以下を求める．

- 各指標（支持度，信頼度，リフト値）の閾値の変化による，提案手法の予測精度の変化

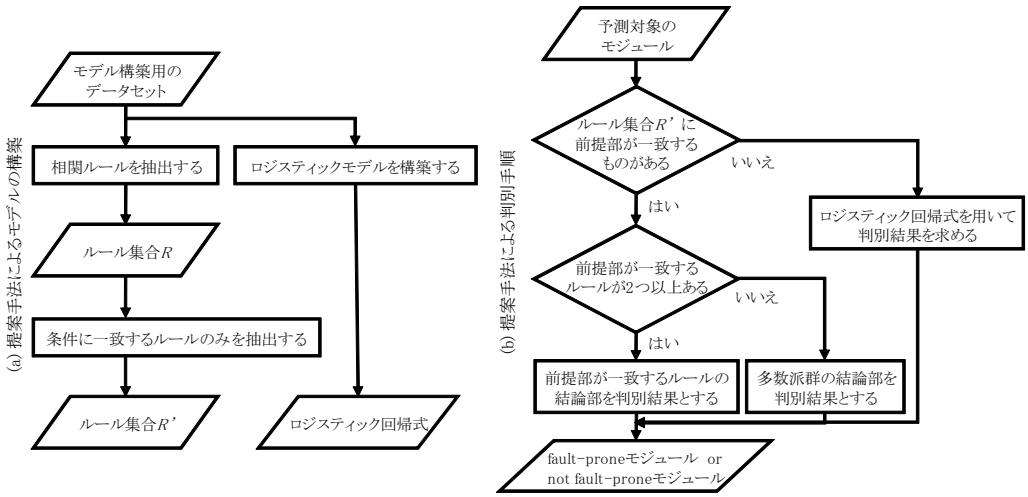

図 1　モデルの構築と判別手順

- 各指標の閾値の変化による，相関ルール分析によって判別できるモジュール数の変化
- ロジスティック回帰分析単体で予測した場合の予測精度

3.2　データセット

実験には，NASA/WVU IV&V facility Metrics Data Program（MDP） [6] が公開しているデータセットのうち，KC1 と KC3 で収集されたデータセットを用いた．KC1 では，C++で開発されたソフトウェアのモジュールデータが収集されている．ソースコードの規模は 43k ステップであり，モジュール数は 2,107 個である．2,107 個のうち，fault を含まないモジュールは 1,782 個，fault を 1 つ以上含むモジュールは 325 個である．KC3 では，Java で開発されたソフトウェアのソースコードメトリクスと fault の情報が収集されている．ソースコードの規模は 18k ステップであり，モジュール数は 458 個である．458 個のうち，fault を含まないモジュールは 415 個，fault を 1 つ以上含むモジュールは 43 個である．

実験では，fault の有無を目的変数，KC1 では 21 種類の，KC3 では 40 種類のソースコードメトリクスを説明変数として用いた．Fault-prone モジュール判別モデルを構築する際には，データセットをランダムに二等分し，一方をモデルを構築するためのフィットデータとし，もう一方をモデルの予測精度を評価するためのテストデータとする．

3.3　実験手順

実験では，以下の手順で相関ルール抽出の指標値と予測精度を求めた．

1. データセットをランダムに二等分し，一方をフィットデータセット fit，もう一方をテストデータセット $test$ とする．
2. 相関ルール分析によって fit から相関ルールを抽出し，閾値を変化させたときそれぞれの $test$ に対する予測精度を評価する．
3. ロジスティック回帰分析によって fit からロジスティック回帰式を求め，$test$ に対する予測精度を評価する．
4. 手順 2 と 3 の相関ルールとロジスティック回帰式を基にモデルを構築し，閾値を変化させたときそれぞれの $test$ に対する提案手法の予測精度を評価する．

手順 1 から 4 をそれぞれのデータセットに対して行った．

3.4 評価指標

評価基準として，F1 値を用いた．F1 値は，fault モジュールと予測したうち実際に fault モジュールである割合を示す適合率と，すべての fault モジュールのうち fault モジュールと予測した割合を示す再現率の調和平均で与えられる [2].

3.5 結果

実験の結果として，各指標の変化による提案手法の予測精度の変化，ロジスティック回帰分析の予測精度を図 2 に示す．図 2(a)，(b)，…，(f) に，データセットと指標値それぞれの組み合わせの結果を示す．グラフの横軸は各指標の閾値，左側の縦軸は F1 値，右側の縦軸は相関ルールによって予測されたモジュールの割合を示す．信頼度，リフト値を変化させる際に設定した最小支持度は 0.01 である．

図 2 に示すように，提案手法を用いることで，相関ルール分析では前提部の一致する相関ルールがなく予測できなかったモジュールをすべて予測することができ，かつ，ロジスティック回帰分析より F1 値が最大 0.16 向上した．2 つのデータセットに共通して提案手法で用いた基準ごとに以下の特徴が見られた．

- **支持度**　閾値の変化に関わらず提案手法の F1 値はロジスティック回帰分析と比較して小さい．閾値を上げていった場合でも，相関ルールにより 80%以上のモジュールを予測している．
- **信頼度**　閾値の変化に関わらず提案手法の F1 値は，ロジスティック回帰分析

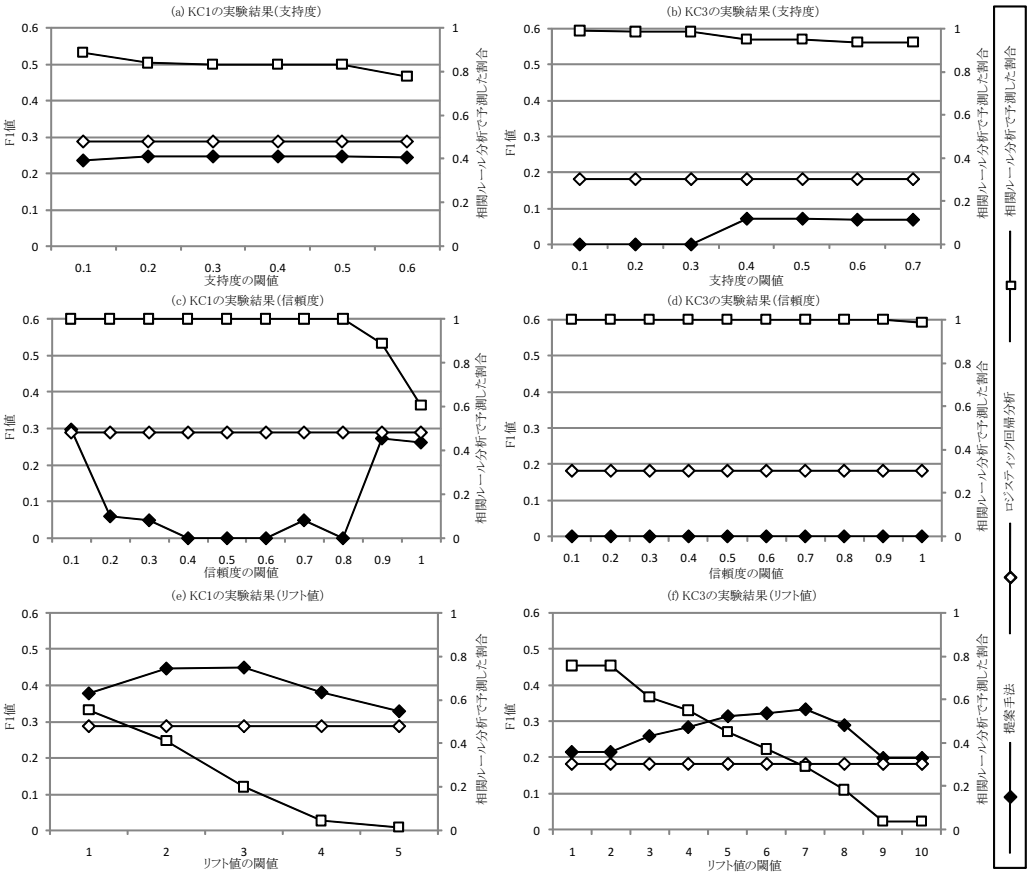

図 2　相関ルール抽出の外的基準による予測可能モジュール割合と予測精度の変化

表 1　KC1 から抽出した相関ルールの一部

番号	前提	結論	支持度	信頼度	リフト値
R_1	$(4.00 \leq m_2 \leq 45.00_{max}) \wedge (0.00_{min} \leq m_4 < 0.06)$ $\wedge (17.00 \leq m_5 \leq 75.00_{max} \wedge (13.00 \leq m_6 \leq 37.00_{max})$	FP	0.05	0.56	3.47
R_2	$(7.00 \leq m_1 \leq 89.00_{max}) \wedge (0.00_{min} \leq m_4 < 0.06)$ $\wedge (17.00 \leq m_5 \leq 75.00_{max}) \wedge (13.00 \leq m_6 \leq 37.00_{max})$	FP	0.05	0.56	3.47
R_3	$(7.00 \leq m_1 \leq 89.00_{max}) \wedge (4.00 \leq m_2 \leq 45.00_{max})$ $\wedge (24.00 \leq m_3 \leq 262.00_{max}) \wedge (0.00_{min} \leq m_4 < 0.06)$	FP	0.05	0.53	3.26

m_1:分岐の数　　m_2:サイクロマティック数　　m_3:コードの実行行数（空行とコメント行以外）
m_4:Halstead 尺度の Level　　m_5:ユニークなオペランドの数　　m_6:ユニークなオペレータの数

と比較して小さい．

- **リフト値**　閾値の変化に関わらず提案手法の F1 値は，ロジスティック回帰分析と比較して大きい．リフト値を一定の高さまで上げることで，F1 値は大きくなり，一定の高さを超えると F1 値は小さくなる．

提案手法の性能向上に寄与していたリフト値，特に最も F1 値が高くなったリフト値周辺の相関ルールの一部を表 1 に示す（紙面の都合上，KC1 のみ）．表 1 の各行は 1 つの相関ルールを示し，各ルールに含まれる前提部の条件範囲が変数「m_i」の上限に達している場合には「$_{max}$」，下限に達している場合には「$_{min}$」が示されており,「FP」は結論部が fault-prone モジュールであることを示す．

相関ルールは，複雑さを表すソースコードメトリクスが大きなもの (分割した区間のうち最大) が，fault-prone となることを示している．例えば，R_1，R_2 では，分岐の数 (m_1)，サイクロマティック数 (m_2)，ユニークなオペランド数 (m_5)，ユニークなオペレータ数 (m_6) が大きくなった場合，fault-prone になることを示すルールである．R_3 も同様の傾向を示しており，R_3 ではソースコード行数が大きいことが fault-prone になる場合があることを示している．また，これらの相関ルールは，ロジスティック回帰分析で予測が外れていたモジュールに対して，正しく予測することができたルールであった．

3.6　考察

実験では，支持度と信頼度を指標として予測に用いる相関ルールを選択するよりも，リフト値を用いてルールを選択するほうが高い予測精度が得られた．支持度と信頼度を指標とした場合には，予測できたモジュール数が大きい．支持度と信頼度を指標とした場合には，網羅性の高いルールが多く選択されるものの予測精度の向上には寄与しないことが考えられる．一方，リフト値を指標とした場合には，予測できたモジュール数が小さかった．リフト値は前提部が結論部に寄与する度合いを測る指標であることからも，リフト値を選択基準とすることで，fault-prone の予測精度に寄与しない相関ルールを選択できたことが推測される．また，リフト値の高い相関ルールはロジスティック回帰分析では予測が外れていること，及び，支持度と信頼度は大きくない，ことを確認した．

表 1 に示す相関ルールのように，複雑さ，及び，規模を表すメトリクスが大きい場合に，fault-prone となる場合が多いことを示す結果が得られた．また，R_1, R_2, R_3 の前提部に Halstead 尺度の Level（m_4）が大きいという条件が含まれた．この結果は Halstead 尺度の Level が fault モジュールの予測に最も寄与するメトリクスであるという報告 [6] と一致するといえる．

相関ルール分析では，前提部が fault-prone，not-fault-prone に寄与していることが推測できる．今回の実験では，リフト値を指標とし，相関ルールを選択することで高い予測精度が得られた．長期にわたって保守が必要なソフトウェアでは，リフト値を指標として，選択した相関ルールに含まれる前提部のソースコードメトリクスを監視することにより，ソースコードメトリクスを意識したリファクタリングが

可能となる．例えば，保守が原因で，分岐の数 (m_1) が特に大きなモジュールに対して，分岐の数が小さくなるようなリファクタリングを実施することで，fault を未然に防ぐことに寄与することが期待される．

4 おわりに

本稿では，fault-prone モジュール判別の精度向上を目的として，相関ルール分析とロジスティック回帰分析を用いたモデルの組み合わせる手法を提案し，その性能を実験的に評価した．実験により得られた主な結果および知見は以下の通りである．

- 提案手法を用いることで，相関ルール分析では予測できなかったモジュールをすべて予測することができ，かつ，ロジスティック回帰分析と比較して F1 値が最大 0.16 向上した．
- 今回用いたデーセットでは，相関ルール分析とロジスティック回帰分析を組み合わせる際に用いる相関ルールの選択には，リフト値が適していることがわかった．一方，支持度と信頼度は適していないことがわかった．
- リフト値を提案手法に用いた場合には，サイクロマティック数や分岐の数などの複雑さを表すメトリクス値が一定の値を超えると，fault を含む確率が高いことを示す相関ルールが含まれていた．

ただし本結論は 2 つのデータセットから結論付けられるものであり，結果の信頼性を向上させるためには他のデータセットを用いる必要がある．相関ルール分析で予測できなかったモジュールの傾向，ロジスティック回帰分析で予測できなかったモジュールの傾向について分析することは今後の課題の 1 つである．

謝辞 本研究の一部は，文部科学省「e-Society 基盤ソフトウェアの総合開発」の委託に基づいて行われた．

参考文献

[1] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Proc. of 1993 ACM SIGMOD Int'l Conf. on Management of Data*, pp. 207–216, Washington, D.C., USA, 1993.

[2] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Information Systems*, Vol. 22, No. 1, pp. 5–53, 2004.

[3] Taghi M. Khoshgoftaar and Edward B. Allen. Modeling software quality with classification trees. In *Recent Advances in Reliability and Quality Engineering*, pp. 247–270, Singapore, 1999. World Scientific.

[4] Paul L. Li, James Herbsleb, Mary Shaw, and Brian Robinson. Experiences and redsults from initiating field defect prediction and product test prioritization efforts at ABB inc. In *Proc. 28th Int'l Conf. on Software Engineering*, pp. 413–422, Shanghai, China, 2006.

[5] John C. Munson and Taghi M. Khoshgoftaar. The detection of fault-prone programs. *IEEE Trans. Softw. Eng.*, Vol. 18, No. 5, pp. 423–433, 1992.

[6] NASA/WVU IV&V Facility. Metrics Data Program. `http://mdp.ivv.nasa.gov/`.

[7] Niclas Ohlsson and Hans Alberg. Predicting fault-prone software modules in telephone switches. *IEEE Trans. Softw. Eng.*, Vol. 22, No. 12, pp. 886–894, 1996.

[8] Qinbao Song, Martin Shepperd, Michelle Cartwright, and Carolyn Mair. Software defect association mining and defect correction effort prediction. *IEEE Trans. Softw. Eng.*, Vol. 32, No. 2, pp. 69–82, 2006.

[9] Yasunari Takagi, Osamu Mizuno, and Tohru Kikuno. An empirical approach to characterizing risky software projects based on logistic regression analysis. *Empirical Software Engineering*, Vol. 10, No. 4, pp. 495–515, 2005.