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

Yasutaka Kamei Akito Monden Shuji Morisaki Ken-ichi Matsumoto
Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara 630-0192, Japan
{yasuta-k, akito-m, smrs, matumoto}@is.naist.jp

ABSTRACT
This paper proposes a fault-prone module prediction method that combines association rule mining with logistic regression analysis. In the proposed method, we focus on three key measures of interestingness of an association rule (support, confidence and lift) to select useful rules for the prediction. If a module satisfies the premise (i.e. the condition in the antecedent part) of one of the selected rules, the module is classified by the rule as either fault-prone or not. Otherwise, the module is classified by the logistic model. We experimentally evaluated the prediction performance of the proposed method with different thresholds of each rule interestingness measure (support, confidence and lift) using a module set in the Eclipse project, and compared it with three well-known fault-proneness models (logistic regression model, linear discriminant model and classification tree). The result showed that the improvement of the F1-value of the proposed method was 0.163 at maximum compared to conventional models.

Categories and Subject Descriptors
D.2.9 [**Software Engineering**]: Management– *Software quality assurance*

General Terms
Management, Reliability, Experimentation

Keywords
Fault-prone module prediction, empirical study, association rule mining, logistic regression analysis

1. INTRODUCTION
Identification of fault-prone modules, which may need rework and/or comprehensive testing, is an important issue in software quality assurance [3][5][6]. Various multivariate modeling techniques applicable to fault-prone module prediction have been proposed, including linear discriminant analysis, logistic regression analysis and classification tree. Particularly, this paper targets the logistic regression analysis, which is one of the commonly used modeling techniques [2][5]. Given a module, a logistic regression model computes the probability that the module has a fault (i.e. fault-prone) based on its module metrics. Since fault injection is a stochastic event, it is natural to use such a probabilistic model for fault-prone module detection.

On the other hand, prediction using association rule mining has also been proposed as a non model-based (rule-based) method [6]. Association rule mining aims to discover patterns of co-occurrences of attributes in a dataset. For example, an association rule "(10 ≤ cyclomatic number < 30) and (10 ≤ fan-in < 25) ⇒ fault prone" indicates that a module is fault-prone if its cyclomatic number is between 10 and 30, and its fan-in is between 10 and 25. A large number of such rules are mined from a past project's module dataset. The advantage of association rule mining is that various types of faulty modules can be characterized by a large set of rules, while model-based methods rely on a single model. In addition, to increase the prediction performance, we can select rules based on interestingness measures of a rule such as support and confidence (Section 2). The disadvantage is that not all the modules are predictable because for some modules, there would be no rule that matches the modules' metrics values, while model-based methods can predict all modules.

In this paper, we propose a hybrid faulty module prediction method combining association rule mining and logistic regression analysis. We focus on three key measures of interestingness of an association rule (support, confidence and lift) to select useful rules for the prediction. If a module satisfies the premise (i.e. the condition in the antecedent part) of one of the selected rules, the module is classified as either fault-prone or not-fault-prone by the rule. Otherwise, the module is classified by the logistic regression analysis. If there exist two or more rules, then the module is classified by the majority of rules' conclusion. To our knowledge, no study has reported such hybrid methods of rule-based approach (association rule mining) and model-based approach (logistic regression analysis).

This paper experimentally evaluates the prediction performance of the proposed method with different thresholds of each interestingness measure (support, confidence, and lift) using a module set in the Eclipse project, and compares it to that of three well-known fault-proneness models (logistic regression model, linear discriminant model and classification tree).

2. THE PROPOSED METHOD: A HYBRID FAULTY MODULE DETECTION
We propose a hybrid faulty module prediction method using association rule mining and logistic regression analysis. We focus on three key measures of interestingness of a rule (support, confidence and lift) [1] to select rules that are likely to contribute to fault-prone module prediction. These measures are described as follows.

Let $I = \{I_1, I_2, ..., I_m\}$ be a set of items where each I_k ($1 \le k \le m$) is an item and m is the number of unique items. An association rule is denoted by an expression $A \Rightarrow B$, where $A \subset I$, $B \in I$, $A \cap B =$

ESEM'08, October 9–10, 2008, Kaiserslautern, Germany.

Table 1. Source code metrics of the dataset in Eclipse

	Metrics		Metrics
m_1	LOC executable	m_2	Total methods LOC
m_3	Nested block depth	m_4	Cyclomatic complexity
m_5	# of parameters	m_6	# of attributes
m_7	# of methods	m_8	# of overridden methods
m_9	# of children	m_{10}	# of statistic attributes
m_{11}	# of statistic methods	m_{12}	SIX (Specialization Index)
m_{13}	LCOM (Lack of Cohesion of Methods)	m_{14}	WMC (Weighted Methods per Class)
m_{15}	DIT (Depth of Inheritance Tree)		

ϕ. Let a database D be $\{T_1, T_2, ..., T_n\}$ where $T_i \subseteq I$ is called a transaction, n is the number of transactions.

Support: Support is an indicator of rule frequency. It is expressed as support ($A \Rightarrow B$), and is support ($A \Rightarrow B$) = a/n, where $a = |\{T \in D | A \subset T \cap B \subset T\}|$.

Confidence: Confidence is the probability that consequent B will follow antecedent A. It is expressed as confidence ($A \Rightarrow B$), and is confidence ($A \Rightarrow B$) = a/b, where a is defined as in Support and $b = |\{T \in D | A \subset T\}|$.

Lift: Lift is an indicator of the contribution antecedent A makes to consequent B. It is expressed as lift ($A \Rightarrow B$), and is lift ($A \Rightarrow B$) = confidence ($A \Rightarrow B$) * (n / c), where $c = |\{T \in D | B \subset T\}|$.

The detailed procedure of the proposed method is described as follows. A logistic regression model L is built and a set of association rules R is mined from a fit dataset. Here, to apply association rule mining to the fit dataset, ratio scale and interval scale variables are converted to ordinal scale variables beforehand. Each quantitative variable is divided into d equal interval parts as with Morisaki et al. [4] (in this paper, d was set to 5).

Next, given a threshold $\theta_{support}$ (or $\theta_{confidence}$ or θ_{lift}) of an interestingness measure (support, confidence or lift), we select rules R' whose support (or confidence or lift) is greater than the threshold from the rules R. These rules are used to predict modules before considering using logistic regression analysis.

Then, given a target module (to be predicted), we inspect all the rules in R' if there exists a rule whose premise (i.e. the condition in the antecedent part) is satisfied by the module. If not, the module is classified by the logistic regression model L. Otherwise, if the module satisfies only one rule, then the module is classified by this rule. If there exist two or more rules, then the module is classified by the majority of rules' conclusion.

However, in our method, it is not clear which interestingness measure (support, confidence or lift) is the most appropriate for the rule selection. Also, proper thresholds for these measures are unknown. Therefore, we clarify these points through experimental evaluation in Sections 3.

3. EVALUATION

3.1 Overview

In this experiment, we experimentally evaluated the prediction performance of the proposed hybrid method with different thresholds of each rule interestingness measure (support, confidence and lift), and compared it with three well-known fault-proneness models (logistic regression model, linear discriminant model and classification tree). We collected module metrics data and fault data from the Eclipse project by using Gyimothy's approach [3].

Table 2. Condition for collecting bug reports

Product	Platform
Status of faults	Resolved, Verified, Closed
Resolution of faults	Fixed
Severity	Except Enhancement
Priority	All

Table 3. Summary of datasets in Eclipse

Version	# of faulty modules	# of not faulty modules	% of faulty modules
3.0 (fit)	793	3,659	17.8
3.1 (test)	859	4,415	16.3

When we used the support or the confidence as a threshold, we changed the value of the threshold $\theta_{support}$ or the threshold $\theta_{confidence}$ by 0.1 (0.1, 0.2, 0.3 …). And, as for the lift, we change the value of the threshold θ_{lift} from 1.5 to 2.0, 2.5, 2.7 and 2.9 since the maximum lift value in this experiment was less than 3.0. In this experiment, we used a prototype tool "NEEDLE" implemented by Morisaki et al. [4] to derive the rule set R.

We used three commonly used criteria, recall, precision and F1-value, to evaluate the prediction performance. Recall is the ratio of correctly predicted fault-prone modules to actual fault-prone modules, and precision is the ratio of actual fault-prone modules to the modules predicted as fault-prone. F1-value is a harmonic mean of recall and precision. Larger F1-value indicates better prediction performance.

3.2 Dataset

The target software is Eclipse, one of the most famous open development platforms. In this experiment, we used a module dataset of "Platform" of Eclipse in Version 3.0 as fit and a module dataset in Version 3.1 as test. We measured metrics of modules and collected bug reports as follows. Here, a module is a Java file (*.java).

First, we collected source files of each version and measured source code metrics using the Eclipse Metrics plug-in[1]. In this experiment, for the construction of fault-proneness models, 15 metrics were used as predictor variables and the existence of a fault (no fault or more than one fault) was used as an objective variable (Table 1). Then, based on the condition shown in Table 2, we collected bug reports to determine whether each module was faulty or not from Bugzilla[2], which was provided by the developer community of Eclipse. Finally, in this paper, we associated faults, modules and versions by using Gyimothy's approach [3]. Table 3 shows a statistics summary of datasets collected from Eclipse using the procedure above.

3.3 Results

The prediction performance (F1-value) of the hybrid method for each measure (support, confidence and lift) and that of three fault-proneness models are shown in Table 4. As shown in Table 4, the improvements of F1-values were 0.163 at maximum compared to

[1] http://sourceforge.net/projects/metrics/
[2] https://bugs.eclipse.org/bugs/

Table 4. Prediction performance of each method

	Precision	*Recall*	*F1-value*	% of the classified modules by rules
LRA	0.574	0.176	0.269	-
LDA	0.580	0.135	0.219	-
CT	0.567	0.173	0.266	-
Hybrid method θ_{supp} = 0.1, 0.2, …, 0.8	0.000	0.000	0.000	100.00
Hybrid method θ_{conf} = 0.1, 0.2, …, 1.0	0.000	0.000	0.000	100.00
Hybrid method θ_{lift} = 1.5	0.231	**0.877**	**0.365**	61.87
Hybrid method θ_{lift} = 2.0	0.298	**0.705**	**0.419**	38.36
Hybrid method θ_{lift} = 2.5	0.376	**0.509**	**0.432**	21.71
Hybrid method θ_{lift} = 2.7	0.439	**0.320**	**0.370**	10.77
Hybrid method θ_{lift} = 2.9	**0.583**	**0.179**	**0.274**	2.03

linear discriminant analysis that was the best performance in the three fault-proneness models. The following characteristics were found for each measure.

Support: As shown in Table 4, regardless of the threshold $\theta_{support}$, the prediction performance of the hybrid method was the worst (F1-value = zero). In the hybrid method, all modules were classified as not fault-prone by association rules (i.e. the logistic regression model was not used).

Confidence: As with the support, regardless of the threshold $\theta_{confidence}$, the prediction performance of the hybrid method was worse than that of linear discriminant analysis. In the hybrid method, all modules were classified as not fault-prone by association rules.

Lift: Regardless of the threshold θ_{lift}, the prediction performance of the hybrid method was the best. When the threshold θ_{lift} = 2.5, the F1-value of the hybrid method was the best (0.432), while that of linear discriminant analysis was 0.269, that of logistic regression analysis was 0.219, and that of classification tree was 0.266.

The F1-value of the hybrid method increased while the threshold $\theta_{lift} \leq 2.5$, and it decreased while $2.5 < \theta_{lift}$. The best performance was achieved when the percentage of modules classified by the association rules was about 20%.

3.4 Discussion

The prediction performance of the hybrid method using the support or the confidence was worse than that of three fault-proneness models. While Song et al. [6] used the support in rule-ranking strategy for predicting defect associations, the support alone did not contribute to the prediction performance in our method. This was because the support does not indicate the probability that the consequent will follow the antecedent. On the other hand, the confidence indicates the probability; however, the confidence also did not work well because it does not consider the percentage of faulty modules in the fit dataset. Many module datasets in the field are actually imbalanced, i.e. there exists a large difference between the number of fault-prone modules and not-fault-prone modules. In our experiment, the percentage of fault-prone modules was about 17.8%. Therefore, for example, "confidence = 70%" for a rule "xxx ⇒ faulty" is meaningful but it is not for a rule "xxx ⇒ not faulty." As a result the confidence did not contribute to the prediction.

The prediction performance of the hybrid method with the lift was better than that of three conventional fault-proneness models. Regardless of the threshold of the lift (in Table 4), the proposed method was better than the logistic model. This indicates that most of association rules selected by the lift contributed to improving the performance of the logistic model.

4. CONCLUSION

We experimentally evaluated the prediction performance of the proposed hybrid method by using module set in Eclipse project. Our major findings include the following:

- The improvement of the F1-value of the hybrid method was 0.163 at maximum compared to three well-known fault-proneness models (linear discriminant model, logistic regression model and classification tree).
- The lift was the most suitable measure to select useful association rules in the proposed method compared to other measures (support and confidence).
- The proposed method performed best when the percentage of the classified modules by rules was about 20%.

The major limitation of this paper is that we used only a single dataset. Our future work is to confirm our results using other datasets. We also plan to combine association rule mining with other models (such as the linear model and classification tree).

5. ACKNOWLEDGMENTS

This work is being conducted as a part of StagE Project, the Development of Next Generation IT Infrastructure, supported by Ministry of Education, Culture, Sports, Science and Technology and Grant-in-Aid for Japan Society for the Promotion of Science (JSPS) Fellows (Research No:20009220).

6. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In Proc. Int'l Conf. on Management of Data, pages 207-216, 1993.

[2] V. R. Basili, L. C. Briand, and W.L. Melo. A Validation of Object-Oriented Design Metrics as Quality Indicators. IEEE Trans. Softw. Eng., 22(10):751-761, 1996.

[3] T. Gyimothy, R. Ferenc, and I. Siket. Empirical Validation of Object-Oriented Metrics on Open Source Software for Fault Prediction. IEEE Trans. Softw. Eng., 31(10):897-910, 2005.

[4] S. Morisaki, A. Monden, H. Tamada, T. Matsumura, and K. Matsumoto. Mining Quantitative Rules in a Software Project Data Set. IPSJ Journal, 48(8):2725-2734, 2007.

[5] J. C. Munson, and T. M. Khoshgoftaar. The Detection of Fault-prone Programs. IEEE Trans. Softw. Eng., 18(5): 423-433, 1992.

[6] Q. Song, M. Shepperd, M. Cartwright, and C. Mair. Software Defect Association Mining and Defect Correction Effort Prediction. IEEE Trans. Softw. Eng., 32(2):69-82, 2006.