A Study of Redundant Metrics in Defect Prediction Datasets

Jirayus Jiarpakdee, Chakkrit Tantithamthavorn, Akinori Ihara, Kenichi Matsumoto
Graduate School of Information Science, Nara Institute of Science and Technology, Japan.
{jirayus.jiarpakdee.jc6,chakkrit-t,akinori-i,matumoto}@is.naist.jp

***Abstract*—Defect prediction models can help Software Quality Assurance (SQA) teams understand their past pitfalls that lead to defective modules. However, the conclusions that are derived from defect prediction models without mitigating redundant metrics issues may be misleading. In this paper, we set out to investigate if redundant metrics issues are affecting defect prediction studies, and its degree and causes of redundancy. Through a case study of 101 publicly-available defect datasets of systems that span both proprietary and open source domains, we observe that (1) 10%-67% of metrics of the studied defect datasets are redundant, and (2) the redundancy of metrics has to do with the aggregation functions of metrics. These findings suggest that researchers should be aware of redundant metrics prior to constructing a defect prediction model in order to maximize internal validity of their studies.**

***Index Terms*—Software quality assurance, defect prediction models, redundant metrics.**

I. Introduction

Defect prediction models can help Software Quality Assurance (SQA) teams understand their past pitfalls that lead to defective modules. For example, Hassan [2] studies the impact of complexity of code changes on software quality. Shihab et al. [7] investigate the impact of code and process metrics on post-release defects. Such knowledge can help SQA teams chart quality improvement plans to avoid such past pitfalls.

There have been a variety of techniques to study the impact of software metrics (i.e., explanatory variables) and software quality. For example, Tantithamthavorn et al. [9] use Breiman's score of importance metrics of random forest models to analyze the impact of various software metrics on defect-proneness. Thongtanunam et al. [11] use an ANOVA analysis of regression models to study the impact of code ownership and review-aware ownership metrics on software quality.

Interpreting defect prediction models without mitigating redundant metrics issues (i.e., a phenomenon that one metric is highly correlated with one another) may be misleading or produce spurious relationship, which likely lead to missteps in practice. For example, Mason et al. [1] show that redundant metrics can inflate or deflate the variance of coefficients of regression models. Tantithamthavorn et al. [10] show that redundant metrics introduce interference to impact analyses, such as ANOVA, leading to misleading conclusions.

While redundant metrics issue may impact the interpretation of defect prediction models, a literature survey by Shihab [6] shows that as few as 37% of defect prediction studies that published during 2000-2011 mitigate redundant metrics prior to constructing a defect prediction model.

In this paper, we set out to investigate if redundant metrics issues are affecting defect prediction studies, and its degree and causes of redundancy. We apply three detection techniques (i.e., variance inflation factors, variable clustering, and redundancy analysis) to detect redundant metrics on 101 publicly-available datasets of systems that span both proprietary and open source domains.

II. Case Study Design

In this section, we divide our study design into two parts: data collection and data analysis.

A. Data Collection

In order to combat any potential bias and better generalize our conclusions, we select to study 101 publicly-available defect datasets that are gathered by our recent work [8]. The 101 defect datasets are gathered from systems that span both proprietary and open source domains, which vary in size, metrics, defective ratio, and research groups.

B. Data Analysis

To detect redundant metrics, we apply three commonly-used detection techniques, i.e., variance inflation factors, variable clustering, and redundancy analysis. We apply these techniques to each of our studied datasets. We describe each technique below.

Variance Inflation Factors (VIF) computes the variance inflation factor for each metric. A VIF score for a single metric is calculated from the R-squared value of regression models of that metric against all other metrics. A VIF score of 1 indicates that there is no correlation between the studied metric and the remaining metrics. However, when the score exceeds the threshold of 5, the metric is considered likely redundant [5], and the mitigation is required. We use the implementation of variance inflation factors using the `vif` function that is provided by the `rms` R package.

Variable Clustering (Varclus) is a hierarchical clustering to determine the correlation between metrics. Highly correlated metrics are grouped together. In our study, we select Spearman rank correlation as our measure as it is resilient to data that is not normally distributed. As suggested by prior work [3, 4, 11], we use the threshold of 0.7 to identify highly redundant groups when their values are higher than the threshold. We use the implementation of variable clustering using the `varclus` function that is provided by the `Hmisc` R package.

Redundancy Analysis (Redun) determines how well each metric can be predicted using the remaining metrics. Metrics are dropped in stepwise fashion, removing the most predictable metric at each step. The process will be stopped by two conditions. The first condition is when the leftover metrics have no metrics that can be predicted with an adjusted R-squared that is greater than a threshold. The second condition is when removing a metric would cause a previously dropped metric no longer be predicted at the threshold. In this study, we use the default threshold of adjusted R-squared value of 0.9. We use the implementation of redundancy analysis using the `redun` function that is provided by the `Hmisc` R package.

III. Case Study Results

10%-67% of metrics of the 101 public defect datasets are redundant. We find that the results of variable clustering analyses produce 1-8 groups of redundant metrics (i.e., a group of metrics which are highly correlated with each other).

On the other hand, we observe that the results of the VIF analysis of three defect datasets (i.e., `ar3`, `ar4`, and `ar5` of NASA datasets) cannot be produced. After we manually investigate the three problematic datasets, we find that one metric is linearly proportional to another metric — i.e., the `branch_count` metric is two times of the `decision_count` metric. Additionally, we also observe that the results of the redundancy analysis of eight defect datasets (i.e., `ckjm, forrest-0.6, nieruchomosci, skarbonka, systemdata, szybkafucha, tomcat, and zuzel`) cannot be produced. We observe that some metrics (e.g., `noc` and `ce`) of the eight defect datasets are constant at zero.

Our deeper investigation shows that the redundancy of metrics has to do with the aggregation functions of metrics. In `eclipse-2.0, 2.1`, and `3.0` datasets, some metrics that are calculated at the class- or method-level (e.g., method lines of code (`MLOC`)) are often aggregated to file-level using the maximum, summation, or average functions. Thus, the aggregated metrics (i.e., `MLOC_max, MLOC_sum`, and `MLOC_avg`) are redundant — i.e., these three aggregated metrics are derived from the same source of metric (i.e., `MLOC`). Additional results are publicly-available online, https://github.com/jirayusjiar/ISSRE2016_Result.

IV. Conclusions

Defect prediction models can help SQA teams understand their past pitfalls that lead to defective modules. However, the conclusions that are derived from defect prediction models without mitigating redundant metrics may be misleading. Yet, little is known if redundant metrics issues are affecting defect prediction studies.

In this paper, we set out to investigate if redundant metrics issues are affecting defect prediction studies, and its degree and causes of redundancy. Through a case study of 101 publicly-available defect datasets of systems that span both proprietary and open source domains, we made the following observations:

– 10%-67% of metrics of the 101 public defect datasets are redundant.
– Our deeper investigation shows that the redundancy of software metrics has to do with the aggregation functions of metrics.

These findings lead us to conclude that software metrics of the 101 public defect datasets are redundant — which may produce misleading conclusions, suggesting that researchers should be aware of redundant metrics prior to constructing a defect prediction model in order to maximize internal validity of their studies.

V. Acknowledgement

This work was supported by the JSPS Program for Advancing Strategic International Networks to Accelerate the Circulation of Talented Researchers: Interdisciplinary Global Networks for Accelerating Theory and Practice in Software Ecosystem and the Grant-in-Aid for JSPS Fellows (No. 16J03360) and for Young Scientists (B) (No. 16K16037).

References

[1] W. D. P. Charlotte H. Mason, "Collinearity, power, and interpretation of multiple regression analysis," *Journal of Marketing Research*, vol. 28, no. 3, pp. 268–280, 1991.

[2] A. E. Hassan, "Predicting faults using the complexity of code changes," in *Proceedings of the 31st International Conference on Software Engineering (ICSE)*, 2009, pp. 78–88.

[3] H. C. Kraemer, G. A. Morgan, N. L. Leech, J. A. Gliner, J. J. Vaske, and R. J. Harmon, "Measures of clinical significance," *Journal of the American Academy of Child & Adolescent Psychiatry*, vol. 42, no. 12, pp. 1524–1529, 2003.

[4] S. McIntosh, Y. Kamei, B. Adams, and A. E. Hassan, "The Impact of Code Review Coverage and Code Review Participation on Software Quality: A Case Study of the Qt, VTK, and ITK Projects," in *Proc. of the Working Conference on Mining Software Repositories (MSR)*, 2014, pp. 192–201.

[5] R. M. Obrien, "A caution regarding rules of thumb for variance inflation factors," *Quality & Quantity*, vol. 41, no. 5, pp. 673–690, 2007.

[6] E. Shihab, "An exploration of challenges limiting pragmatic software defect prediction," Ph.D. dissertation, Queen's University, 2012.

[7] E. Shihab, Z. M. Jiang, W. M. Ibrahim, B. Adams, and A. E. Hassan, "Understanding the impact of code and process metrics on post-release defects: a case study on the eclipse project," in *Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2010, p. 4.

[8] C. Tantithamthavorn, S. McIntosh, A. Hassan, and K. Matsumoto, "An empirical comparison of model validation techniques for defect prediction models," *IEEE Transactions on Software Engineering (TSE)*, p. To appear, 2016.

[9] C. Tantithamthavorn, S. McIntosh, A. E. Hassan, A. Ihara, and K. Matsumoto, "The impact of mislabelling on the performance and interpretation of defect prediction models," in *proceedings of the International Conference on Software Engineering (ICSE)*, vol. 1, 2015, pp. 812–823.

[10] C. Tantithamthavorn, S. McIntosh, A. E. Hassan, and K. Matsumoto, "Comments on "Researcher Bias: The Use of Machine Learning in Software Defect Prediction"," *IEEE Transactions on Software Engineering (TSE)*, 2016.

[11] P. Thongtanunam, S. McIntosh, A. E. Hassan, and H. Iida, "Revisiting code ownership and its relationship with software quality in the scope of modern code review," in *Proceedings of the 38th International Conference on Software Engineering (ICSE)*, 2016, pp. 1039–1050.