Global Journals LaTeX JournalKaleidoscope™
Artificial Intelligence formulated this projection for compatibility purposes from the original article published at Global Journals. However, this technology is currently in beta. *Therefore, kindly ignore odd layouts, missed formulae, text, tables, or figures.*

Analysis of Data Mining Based Software Defect Prediction Techniques

Naheed Azeem[1] and Shazia Usmani[2]

[1] Federal Urdu University of Arts, Science and Technology

Received: 15 July 2011 Accepted: 10 August 2011 Published: 23 August 2011

Abstract

Software bug repository is the main resource for fault prone modules. Different data mining algorithms are used to extract fault prone modules from these repositories. Software development team tries to increase the software quality by decreasing the number of defects as much as possible. In this paper different data mining techniques are discussed for identifying fault prone modules as well as compare the data mining algorithms to find out the best algorithm for defect prediction.

Index terms— Defect prediction, Data Mining, algorithms

1 INTRODUCTION

oftware life cycle is a human activity, so it is impossible to produce the software without defects. To deliver a defect free software it is imperative to predict and fix the defects as many as possible before the product delivers to the customer.

Software repositories have lots of information that is useful in assessing software quality. Data mining techniques and machine learning algorithms can be applied on these repositories to extract the useful information.

The aim of this research is to explore the different issues and problems in the area of defect prediction as well as provide the solutions to improve the product quality via defect prediction mechanism.

In this survey four type of research issues, formulated as questions, need to be addressed to understand the problems of defect prediction mechanism based on data mining techniques.

2 Research questions:

-How can we resolve the problem of ceiling effects as well as imbalanced and highly skewed datasets? -What software repositories and datasets should be mined for defect prediction? -How can we get better results in identifying defects from large features and high level software modules? -How machine learning algorithms and data mining techniques can be proved more effective in defect extraction from repository? -Is there any good data mining technique that performs the best in all situations?

The remainder of this paper begins with a background and description. (Section 2), followed by Issues and problems regarding data mining in defect prediction and its solution(section 3) , future work and open issues are discussed in section 5and finally summarizes the paper (section 4).

3 II. BACKGROUND AND DESCRIPTIONS

A software defect is an error, flaw, mistake, failure, or fault in a computer program or system that produces an incorrect or unexpected result, or causes it to behave in unintended ways **??**24]. Software defects are expensive in terms of quality and cost. Moreover, the cost of capturing and correcting defects is one of the most expensive software development activities. It will not be possible to eliminate all defects but it is possible to minimize the number of defects and their severe impact on the projects. To do this a defect management process needs to be implemented that focuses on improving software quality via decreasing the defect density. A little investment in

defect management process can yield significant returns. a) Software Defect Prediction Software defect prediction is the process of locating defective modules in software. To produce high quality software, the final product should have as few defects as possible. Early detection of software defects could lead to reduced development costs and rework effort and more reliable software. So, the study of the defect prediction is important to achieve software quality.

The most discussed problem is software defect prediction in the field of software quality and software reliability. As Boehm observed finding and fixing a problem after delivery is 100 times more expensive than fixing it during requirement and design phase.

4 ISSUES AND PROBLEMS

Software prediction model only works well when enough amount of data is available in software repository within the organization to initially feed the model. Extraction of defects from software bug repository accurately is not done without a good data mining model. There is a need of good data mining model to predict the software defects from a bug repository. a) Highly skewed and imbalanced datasets -Existing prediction models based on un sampling as well as training dataset does not contain any information about number of fault per module and distribution of fault among modules [3]. -Data mining algorithms lack of business knowledge and hit a performance ceiling effect when cannot extract the additional information that related to software metrics with fault occurrence [16]. -Fit datasets are usually imbalanced that cause the degradation of defect prediction models [22]. -Highly skewed dataset is considered as the main cause of unsatisfactory prediction result. However the results of more balanced dataset are also unsatisfactory [23].

b) Early life cycle and multiple dataset -Early life cycle data cannot be useful in identifying fault prone modules [9,20]. -No change in defect predictions results when different software repositories are mined [11]. -Single classifier is technically unfit to make use of all the features. However the problems of combing different classifier still remain unresolved [14].

c) Large number of features and high level software modules -Most of the machine leading algorithms are not capable of extracting defects from the database that store continuous features [7]. -Supervised learning are useful for defect prediction at same logical levels but it is not suitable for high level software modules [8]. -Existing classifier based defect prediction model are insufficient accurate for practical use and use of a large number of features [13].

5 d) Accurate defect prediction model

-There is a need of accurate defect prediction model for large-scale software system which is more robust to noise [2]. -Traditional decision tree are used in classification of defective and non-defective modules. However traditional decision trees induction method contain several disadvantages [4]. -There is a need of good data mining model to predict the software defects from a bug repository [5]. -Data transformation can improve the performance of software quality models [21].

6 IV. APPROACHES AND METHODOLOGIES

a) Sampling effect on imbalanced datasets An oversampling method is proposed that using the number of fault per modules and distribution of fault among modules. Two prediction models Naïve Bayes and Logistic regression are applied to two dataset from NASA MDP project .Sampling and over sampling method are used. The result of T test and the nonparametric method of Wilcoxon test showed that oversampling method significant influence on the prediction of both LR and NB model [3].

Author in [16] proposed a human-in-the-loop CBR tool that ad business knowledge to the data mining algorithm. CBR build better prediction model that detect the lower bound on the number of instances. Using three sub sampling techniques (over, under and micro sampling) to find the lower bound the number of training instances. Naive Bayes and j48 methods are used in case of over and under sampling and Naive Bayes is used in case of micro sampling.

Another technique used Sampling method to improve the performance of defect prediction models when data sets are imbalanced [22].Four sampling methods (random over sampling, synthetic minority over sampling, random under sampling and one-sided selection) applied to four fault-proneness models(linear discriminant analysis, logistic regression analysis, neural network and classification tree) by using two module sets of industry legacy software. A method SimBoost is used to handle the software defect prediction problem when high skewed datasets are used, with a fuzzy based classification. A novel method SimBoost is applied on the NASA project dataset to reduced the effects of skewed dataset but the prediction on more balanced dataset are still not accurate. So, fuzzy classification was used to accurate the prediction result [23].

7 b) Effect of early life cycle and multiple dataset

Most of the researchers raise the issue that relying on single data source can limit the accuracy of defect prediction models. However, a combination of different data sources is better to utilize in order to built more accurate fault prediction models.

Both papers [9,20] analyzed that early lifecycle data can be highly useful in defect prediction. In [9] a hybrid Defect prediction models consisting of K-means clustering and C 4.5 are built. Requirement metrics and code metrics and the combination of both requirement and code metrics are applied on these models. Compare the result of models on three NASA projects i.e. CM1, JM1 and PC1. Result shows requirement metric plays an important role in identifying defects. While in the paper [20] author built a Defect prediction models using requirement metrics and code metrics and the combination of both requirement and code metrics. Compare the result of models on three NASA projects. Result shows requirement metric plays an important role in identifying defects.

Author [11] claimed that Defect prediction results improve significantly with different data sources. Three repositories static analysis, version control and release management are used for defect prediction. Learning algorithm ID3, J48 and SVM are used to assess the accuracy of different data sources.

A method is proposed to build a software quality model using multiple learners induced on multiple training datasets to take advantage of their respective biases.

Seventeen classifier models were used on seven NASA datasets. Multiple classifiers were combined by majority voting of experts. Four classification scenarios were used to evaluate the result [14].

8 c) Large number of features and high level software modules

The paper [7] proposed a new data mining model to predict the software bug estimation more accurately. This technique used an entropy based splitting criteria and minimum description stopping criteria (decide when to stop discretization).The binary discretization was always select the best cut point and was applied recursively.

Author investigated that a novel Multi-instance learning technique is much better in identifying defects for high level software modules. Four multi-instance learning algorithm i.e. Statistical learner, Set Kernel, Citation KNN (k Nearest Neighbor) and MI EM-DD (Expectation-Maximization version of Diverse Density) are investigated against three supervised learners Naïve Bayes, Multi-layer Perceptron and logistic Regression [8].

A feature selection algorithm is proposed in [13] that decrease the number of features used by a machine learning classifier for fault prediction. Perform a feature selection process using gain ratio to reduce the set of features in an iterative form. These reduced features are then used to train the two classification model i.e. Naïve Bayes and SVM. Finally the performance of two classifier are assessed in terms of overall prediction accuracy, buggy precision, recall, Fmeasure, and ROC area under curve (AUC).

9 d) Need of accurate defect prediction model

The paper [2] present a software defect prediction model based on random forest which is more robust to outliers and noise than other classifiers and beneficial for large-scale software system. They applied Random forest on five different data set of NASA project using two machine learning tools WEKA and See5. Finally they compare the accuracy of random forest with other statistical methods such as logistic regression and discriminant analysis.

Earlier studies have addressed the use of evolutionary decision tree in classification of defective and non-defective modules. But in [4] author used Evolutionary decision tree in a multi population genetic algorithm. SAEDT is applied on promise dataset using software metrics. The result shows better generalization and higher accuracy.

In [5] a two step data mining model is proposed to predict software bug estimation. In first step, a weighted similarity model is used to match the summary and description of new bug from the previous bug in the bug repository. In the second step calculate the duration of all the bugs and the average is calculated.

The authors [21] criticized that data transformation can improve defect prediction model. They proved it with four data transformation methods applied on ten software quality models on nine dataset from MDP. The performances of models are compared through different test i.e. the Friedman test, the Nemenyi test and the Wilcoxon test.

10 e) Need of a Consistent data mining technique

This paper focused on using and comparing the performance of different machine learning algorithms to build a prediction models based on source code measures and history data. Confusion matrix may be inappropriate for evaluation criteria. Nine different machine learning algorithms are used to build prediction models for a java legacy system to identify the fault prone modules. Compare the performance of each model using confusion matrix and cost sensitive criteria [1].

11 September

In [6], author Evaluate the performance of five data mining algorithm named J48, CART, Random Forest, BFTree and Naïve Bayesian classifier (NBC). The performances of algorithms are evaluated using WEKA tool on software metric dataset KC2 from NASA database. Cross validation test are applied to verify the results. Result shows that performance of algorithm is depends on various factors like problem domain , nature of dataset etc.

Another comparison is done in [10]. This paper compares the three most used data mining techniques. The performance of J48 is better than ONER and ONER is better than Naïve Bayes. Two datasets having 1212

modules and 101 modules was used to evaluate the performance of three machine learning algorithms i.e. J48 , ONER and Naïve Bayes with the help of WEKA tool.10 fold cross validation was applied to confirm the result.

Performances of five classifier prediction model based only on the size of modules measured in LOC are evaluated. Data sets from NASA MDP are used to evaluate the performance of trial defect prediction model based only on the size of modules measured in LOC. Compare the performance of five classifier including Naive Bayes, Logistic Regression, CART decision tree learner, bagging and random forest.When model is evaluated using AUC it shows surprising well results while evaluated using proposed performance measure, the result becomes worst [15].

Researchers [17] evaluate the performance of different fault prediction techniques on different real time software data sets. But no particular technique that prove consistently accurate. Seven different learning methods are applied on Real time data sets from NASA MDP repository to predict the fault prone modules. Also different methods are trained to combine with statistical method PCA. Assess the performance of machine learning algorithm.

MCLP method to build a better prediction model and assess the performance by comparing with other classification algorithm was reported in [8]. Different method are used for generating prediction model include C4.5, Decision Tree, Support Vector Machine (SVM), Neural Network(NN) and Multiple Criteria Linear Programming (MCLP) and applied to data set taken from NASA MDP. Assess the performance of the prediction models based on accuracy, probability of detection (PD), and probability of false alarm (PF).

While in [19], author proposed an ideal a software defect management system based on data mining techniques and data mining models. Proposed methodology of this paper based on three data mining techniques classification, clustering and association rule with two specific data mining models Bayesian Network and Probabilistic Relational Model.

V.

12 CONCLUSIONS

Defects can assess in directing the software quality assurance measures as well as improve software management process if developers find and fix them early in the software life cycle.

Effective Defect prediction is based on good data mining model. In this we surveyed different data mining algorithms used for defect prediction. We also discuss the performance and effectiveness of data mining algorithms. This survey also has showed that all the issues for selecting a data mining technique for defect prediction and their provided solutions have been discussed.

Our most important finding is that there is no single data mining technique that is more powerful or suitable for all type of projects. In order to select a better data mining algorithm, domain expert must consider the various factors like problem domain, type of data sets, nature of project, uncertainty in data set etc. Multiple classifiers were combined by majority voting of experts to get more accurate result.

Our findings indicate that early life cycle data can be highly effective in defect prediction. However, a combination of different data sources can utilize to get better prediction results.

Another finding of this paper is Sampling method are useful to improve performance when dataset are highly skewed. Data transformation cannot improve the performance of defect prediction. Integration of discretization method with classification algorithm improves the defect prediction accuracy by transforming the continuous features into discrete features. However different techniques are applied in identifying defects for large features and high level software modules. and find out whether the cost sensitive learning algorithms can be used to build better defect prediction models.

13 VI. FUTURE WORK AND OPEN ISSUES

Figure 1:

199 1 2 3 4 5

[1]© 2011 Global Journals Inc. (US) Global Journal of Computer Science and Technology Volume XI Issue XVI Version I 3

[2]© 2011 Global Journals Inc. (US) Global Journal of Computer Science and Technology Volume XI Issue XVI Version I 4

[3]© 2011 Global Journals Inc. (US) Global Journal of Computer Science and Technology Volume XI Issue XVI Version I 5

[4]September

[5]© 2011 Global Journals Inc. (US) Global Journal of Computer Science and Technology Volume XI Issue XVI Version I 6 2011 September This page is intentionally left blank

[Euromicro Conference on Software Engineering and Advanced Applications] , *Euromicro Conference on Software Engineering and Advanced Applications* p. .

[Sandhu et al. ()] 'A Model for Early Prediction of Faults in Software Systems'. P S Sandhu , R Goel , A S Brar , J Kaur , S Anand . *The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, 2010. p. .

[Li and Reformat ()] 'A practical method for the software fault-prediction'. Z Li , M Reformat . `24.en.wikipedia.org/wiki/Software_bug25.www.cs.umd.edu/projects/SoftEng/ESEG/papers/82.78.pdf` *IEEE International Conference on Information Reuse and Integration*, 2007. IRI 2007. p. .

[Singh and Verma ()] 'An Investigation of the Effect of Discretization on Defect Prediction Using Static Measures'. P Singh , S Verma . *IEEE International Conference on Advances in Computing, Control, and Telecommunication Technologies*, 2009. p. .

[Jiang et al. ()] 'Can data transformation help in the detection of fault-prone modules?'. Y Jiang , B Cukic , Menzies . *Proceedings of the 2008 workshop on Defects in large software systems*, (the 2008 workshop on Defects in large software systemsSeattle, Washington) July 20-20, 2008. p. .

[Singh ()] 'Comparing the effectiveness of machine learning algorithms for defect prediction'. P Singh . *International Journal of Information Technology and Knowledge Management* 2009. p. .

[Arisholm et al.] 'Data Mining Techniques for Building Fault-proneness Models in Telecom Java Software'. E Arisholm , L C Briand , M Fuglerud . *Proceedings of The 18th IEEE International Symposium on Software Reliability*, (The 18th IEEE International Symposium on Software Reliability) p. .

[Challagulla et al. ()] 'Empirical Assessment of Machine Learning based Software Defect Prediction Techniques'. V U B Challagulla , F B Bastani , I Yen , R A Paul . *Proceedings of the 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems*, (the 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems) 2005. p. .

[Jiang et al. ()] 'Fault Prediction using Early Lifecycle Data'. Y Jiang , B Cukic , T Menzies . *Proceedings of The 18th IEEE International Symposium on Software Reliability*, (The 18th IEEE International Symposium on Software Reliability) 2007. p. .

[Menzies et al. ()] 'Implications of ceiling effects in defect predictors'. T Menzies , B Turhan , A Bener , G Gay , B Cukic , Y Jiang . *Proceedings of the 4th international workshop on Predictor models in software engineering*, (the 4th international workshop on Predictor models in software engineeringLeipzig,Germany) 2008. p. .

[Metrics for Module Defects Identification International Journal of Computer and Information Science ()] 'Metrics for Module Defects Identification'. *International Journal of Computer and Information Science* 2008. p. .

[Gayatri et al. ()] 'Performance Analysis of Data Mining Algorithms for Software Quality Prediction'. N Gayatri , S Nickolas , A V Reddy , R Chitra . *International Conference on Advances in Recent Technologies in Communication and Computing*, 2009. p. .

[Huang and Zhu] 'Predicting Defect-Prone Software Modules at Different Logical Levels'. P Huang , J Zhu . *International Conference on Research Challenges in Computer Science, 2009. ICRCCS '09*, p. .

[Zhao et al. ()] 'Predicting Software Defects using Multiple Criteria Linear Programming'. X Zhao , Y Liu , S Yong . *Proceedings of the International Symposium on Intelligent Information Systems and Applications (IISA'09)*, (the International Symposium on Intelligent Information Systems and Applications (IISA'09)) 2009. p. .

[Nagwani and Verma] 'Predictive Data Mining Model for Software Bug Estimation Using Average Weighted Similarity'. N K Nagwani , S Verma . *2010 IEEE 2nd International Advance Computing Conference*, p. .

[Shivaji et al.] 'Reducing Features to Improve Bug Prediction'. S Shivaji , E J Whitehead , R Akella , S Kim . *24th IEEE/ACM International Conference on Automated Software Engineering, ASE'09*, p. .

[Chen et al. ()] 'Research on software defect prediction based on data mining'. Y Chen , X Shen , P Du , B Ge . *The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, 2010. p. .

[Mende and Koschke ()] 'Revisiting the Evaluation of Defect Prediction Models'. T Mende , R Koschke . *Proceedings of the 5th International Conference on Predictor Models in Software Engineering*, (the 5th International Conference on Predictor Models in Software Engineering) 2009.

[Guo et al. ()] 'Robust prediction of fault-proneness by random forests'. L Guo , Y Ma , B Cukic , H Singh . *Proceedings of the 15th International Symposium on Software Reliability Engineering*, (the 15th International Symposium on Software Reliability Engineering) 2004. p. .

[Khoshgoftaar et al. ()] *Software quality analysis by combining multiple projects and learners*, T M Khoshgoftaar , P Rebours , N Seliya . 2009. Springer. 17 p. .

[Shafi et al. ()] 'Software quality prediction techniques: A comparative analysis'. S Shafi , S M Hassan , A Arshaq , M J Khan , S Shamail . *4th International Conference on Emerging Technologies*, 2008. ICET 2008. p. .

[Kamei et al. ()] 'The effect of over and under sampling on fault-prone module detection'. Y Kamei , A Monden , S Matsumoto , T Kakimoto , K Matsumoto . *First International Symposium on Empirical Software Engineering and Measurement*, 2007.ESEM 2007. p. .

[Li and Leung] 'Using the Number of Faults to Improve Fault-Proneness Prediction of the Probability Models'. L Li , H Leung . *Proceedings of the 2009 WRI World Congress on Computer Science and Information Engineering*, (the 2009 WRI World Congress on Computer Science and Information Engineering) 07 p. .

[Ramler et al. ()] *What Software Repositories Should Be Mined for Defect Predictors?*, R Ramler , S Larndorfer , T Natschläger . 2009. p. 35.