International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org
Volume 4, Issue 6, June 2015 **ISSN 2319 - 4847**

Software Defect Prediction for Quality Improvement Using Hybrid Approach

[1]Pooja Paramshetti , [2]D. A. Phalke

D.Y. Patil College of Engineering, Akurdi, Pune. Savitribai Phule Pune University

ABSTRACT

In today's computing environment, software systems have become increasingly complex and versatile. Therefore it is necessary to continuously identify and correct software design defects. Software defect prediction plays an important role in improving software quality and it helps to reduce time and cost for software testing. The Software defect prediction is a method which predicts defects based on historical database. Different machine learning techniques are used to predict software defects from historical databases. The paper mainly focuses on generating accurate rules for software defect prediction System. For this purpose, K-means clustering technique is used for discretization. Then association rule mining is applied to generate rules in large volumes of data using Apriori algorithm. Software defect prediction system has been experimented on open sources NASA defect dataset. It contains software metric data and error data at the function or method level. In this paper comparison of proposed approach with existing approaches is discussed and the results show that proposed method is generated only interesting rules therefore it is effective for software defect prediction.

Keywords: *Software defect prediction, Software metrics, k-means clustering technique, Apriori algorithm.*

1. INTRODUCTION

A Software defect is a condition in any software product which does not meet a software requirement or end user expectation. In other words, a defect is an error or bug in coding or logic that causes a program to malfunction or to produce incorrect unexpected results. Software defect prediction is the main process of locating defective modules in software. For producing high quality software, the delivered final product should have as few defects as possible. Early detection of software defects could lead to reduced development cost, time and rework effort and more reliable software. Therefore the defect prediction is important to achieve good software quality. Software defect prediction metrics play the most important role to build a statistical defect prediction model. The defect prediction models can be used by the software organizations during the early phases of software development to identify defect modules. The software organizations can use this subset of metrics amongst the available large dataset of software metrics. Software metrics used for developing the defect prediction models. Many researchers have used different methods to establish the relationship between the static code metrics and defect prediction.

1.1 Software metrics

Software metric is a measure of some property of a piece of software [16]. They are often used to assess the ability of software to achieve a predefined goal. In software include complexity, coupling, and cohesion related metrics can be measured during the software development phases such as design or coding and it also used to evaluate the quality of software. Software metrics can be categorized into two kinds those are code and process metrics. Code metrics includes size, Hastead, McCabe, CK and OO metrics have used absolute use frequency of code metrics is higher than process metrics [13].

1.1.1 Cyclomatic Complexity

Cyclomatic Complexity measures the structural of program complexity of the source code. Cyclomatic Complexity calculating the number of different code paths in the flow of the program. A program with complex control flow will require more tests to achieve good code coverage and less maintainable [16].

1.1.2 Halsteads Product Metrics

The measures were developed by the late Maurice Halstead metrics determining a quantitative measure of complexity directly from the operators and operands in the module and also program vocabulary, program length [13].

1.1.3 Product Metrics

In Product Metrics contains the lines of code (LOC) indicates the approximate number of lines in the code. Design metrics computed from requirements or design document before the system has been implemented. Object oriented metrics help identify defects, and it also allow developers to see directly how to make their classes and objects simpler *[16]*.

According to our knowledge in existing system lot of unnecessary rules are generated using Apriori algorithm. Because of that software manager or user get confused therefore more defects occur in software. In order to solve this problem hybridized approach is adopted which gives only interesting rules. For that NASA defect dataset is used and K-means

clustering is used for converting continuous values into discretized form. Then form three clusters for generating accurate rules and the purpose of generating interesting rules using hybrid approach is to assist managers in improving the software process through analysis of the reasons some defects frequently occur together. If the analysis leads to the identification of a process problem, managers have to come up with a corrective action therefore software build good quality and reduced cost and time.

The next section describes the literature survey of existing work. Section 3 describes the proposed work along with algorithm. Section 4 describes dataset used for implementation. Section 5 describes results. The last section consists of the conclusion and future scope.

2. RELATED WORK

Software defect prediction is the most popular research area in these predicting defects using software metrics and data mining techniques recently several research centers started new projects. In this paper, categorized according to the different data mining techniques.

Cagatay Catal [1] studied various papers in year 1990 to 2009 those are as following: they used classification trees with method level metrics on two software systems of NASA and Hughes Aircraft and also applied logistic regression, classification trees. Evett et al. predicted quality based on genetic programming system. They applied fuzzy subtractive clustering method to predict the number of faults and then, they used different module order modeling to classify the modules into faulty or non-faulty classes. They stated that process metrics is not improving the classification accuracy and such a model does not provide acceptable results. They used principal component analysis for first step that is feature selection and then applied fuzzy nonlinear regression to predict defects on a large telecommunications system developed with Protel language. They reported that fuzzy nonlinear regression method is an encouraging technology for early defect prediction. They observed that support vector machine performed better than quadratic discriminate analysis and classification tree. They focused on the high-performance defect predictors based on machine learning such as Random Forests algorithms.

2.1 Software Defect Prediction Based on Classification Techniques

Ezgi Erturk et al. [9] proposed a new method Adaptive Neuron Fuzzy Inference System (ANFIS) for the software fault prediction. Then for performing experiment they used PROMISE Software Engineering Repository dataset, and McCabe metrics are selected because they comprehensively address the programming effort. The results achieved were 0.7795, 0.8685, and 0.8573 for the SVM, ANN and ANFIS methods, respectively.

Mie Thet Thwin [6] have used two kinds of neural network techniques. The first one focuses on predicting the number of defects in a class and the second on predicting the number of lines changed per class. Two neural network models are used which are: Ward neural network and General Regression neural network (GRNN). They have performed the analysis result on the NASA dataset.

David Gray et al. [14] have focused on classification analysis rather than classification performance, it was decided to classify the training data rather than having some form of tester set. It involves a manual analysis of the predictions made by Support vector machine classifiers using data from the NASA Metrics Data Program repository. Ensemble classifier also gives better result for classifying software defects [4]. The purpose was to gain insight into how the classifiers were separating the training data.

Ruchika Malhotra [5] have analyzed and compared the statistical and six machine learning methods for software fault prediction. These methods (Decision Tree, Artificial Neural Network, Cascade Correlation Network, Support Vector Machine, Group Method of Data Handling, and Gene Expression Programming) are empirically validated to find the relationship between the static code metrics and the defects occurs in a module. They compared the models predicted using the regression and the machine learning methods. They have used two publicly available data sets AR1 and AR6 and among them decision tree gives best prediction result.

Ahmet Okutan [12] have used Bayesian networks to determine the probabilistic influential relationships among software metrics and defect proneness. The software metrics used in Promise data repository, define two more metrics, i.e. number of developers for the number of developers and lack of coding quality for the source code quality.

2.2 Software Defect Prediction Based on Association rule Techniques

Alina Campan et al.[11] they proposed a novel algorithm for the discovery of interesting any length of ordinal association rules in defect data sets. Datasets that contain several software metrics with similar or comparable domains of values are frequent in data mining.

Gabriela Czibula et al.[3] they proposed a supervised method for detecting software entities with architectural defects, based on relational association rule mining. They performed eexperiments on open source software are cconducted in order to detect defective classes in object oriented software systems for example the WinRun4J is a windows native launcher for Java implementation.

Qinbao Song et al. [20] they calculate defect association, defect isolation effort, defect correction effort on SEL defect data consisting of more than 200 projects over more than 15 years. They compared the defect correction effort prediction method with other types of methods like PART, C4.5, and Naive Bayes and show that accuracy has been

improved by at least 23 percent. They have explored the impact of support and confidence levels on prediction accuracy, false negative rate, false positive rate, and the number of rules as well. They found that higher support and confidence levels may not result in higher prediction accuracy, and a sufficient number of rules is a precondition for high prediction accuracy.

2.3 Software Defect Prediction Based on Regression

Kamei et al.[2] proposed a defect prone module prediction method that combines association rule mining with logistic regression. They have predicted performance of their algorithm method with different thresholds of each rule interestingness measure support, confidence and lift using a module set in the Eclipse project.

Yuan Jiang, Ming Li et al.[9] have addressed two practical issues first, it is rather difficult to collect a large amount of labeled training data for learning a well-performing model and second, in a software system there are usually much less defective modules than defect free modules, therefore learning techniques would have to be conducted over an imbalanced data set therefore they proposing a novel semi-supervised learning approach named Random Committee with Under Sampling (Rocus). This method incorporates recent advances in disagreement-based semi-supervised learning with under-sampling strategy for imbalanced data.

Above approaches have not used hybrid approach that is k-means clustering with Apriori algorithm for generating accurate rules regarding, they just focused on the relation association rule. This work focuses on improving performance of rule generation for software defect prediction. As in original work Apriori algorithm is used, it returns a large amount of results. Applying K-means algorithm in preprocessing step on results of defect prediction improve accuracy.

3. IMPLEMENTATION DETAILS

3.1 System Overview

In the proposed system, input is training dataset including software metrics and their values. First preprocessing is done, in which all the values in continues form are converted into discrete form by using k-means clustering techniques. Then applying Apriori algorithm with minimum support threshold and minimum confidence threshold, after that rules are generated and in such way software defect are predicted. The architecture of the software defect prediction proposed system is shown in Figure 1. The main objective is to locate defective modules in software, for producing high quality software so that the final product should be of good quality. In the first phase, all software metrics are discretized into three values low, medium, high. The way in which dataset is pre-processed can now be used for building the association rule. Finally, focusing on identifying relations between two software metrics, relations that would be relevant for deciding if a software entity is or not defective. After the relations are defined, the interesting association rules are discovered in the training datasets with minimum support and confidence.

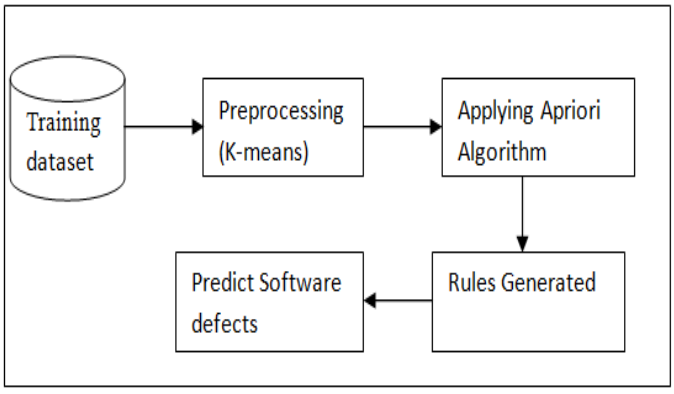

Figure 1 System architecture for software defect prediction

3.2 Algorithm

3.2.1 K-means

The clustering algorithm sorts the software metrics values into three different values in discrete form. For this performed following steps:

1. Randomly select C as a cluster center
2. Calculate the distance between each data point and cluster center.
3. Assign data point to the cluster where distance from cluster center is minimum of all cluster centers.
4. Recalculate distance between number of data points and new obtained cluster center.
5. Recalculate distance between number of data points and new obtain cluster center.

3.2.2 Apriori

The algorithm is used to mine all frequent item sets in database and generate rules. For this performed following steps :

1. It finds all frequent item sets. To find frequent item sets where k-item sets are used to generate k+1 item sets.
2. In this each k-item set must be greater than or equal to minimum support threshold to be frequency if not then it is candidate item sets. It finds frequent item set using candidate generation.
3. It implies an iterative approach known as level wise search where k item sets level 1 are used to explore (k+1) item sets level 2 i.e. L1 is used to find L2, L2 is used to find L3 and so on.
4. It generates strong association rules from the frequent item sets.

The algorithm performs multiple passes over the data set *R*. In the first pass, it calculates the support and confidence of the any length rules and determines which of them are interesting, i.e. decide minimum support and confidence requirement. In every subsequent pass over the data, start with a seed set of interesting rules, found in the previous pass. Then this set to generate new possible interesting rules, called candidate rules, and then computes the actual support and confidence of these candidates during the scan of the data. At the end of this step, keep the rules that are deemed interesting, which will be used in the next iteration. The process ends when no new interesting rules were found in the latest iteration.

4. DATASET

The publicly available National Aeronautics and Space Administration (NASA) datasets have been extensively used for finding software defect research. The NASA defect data sets are easy to understand and comparable. The data set is provided by the NASA IV and V Metrics Data Program contains software metrics and associated error data at the method level. The data repository records are stored and organized which has been collected and validated by the Metrics Data Program [8]. The Promise Data Repository 2 has served as an important role for making software engineering data sets publicly available [17]. The database uses unique numeric identifiers or values to describe the individual error records and software entries. The repository contains all metric data in terms of product metrics, object oriented class metrics, requirement metrics and defect association metrics.

Table 2 Dataset description [8]

Dataset	Language	Attribute	System
CM1	C	38	NASA spares craft instrument
PC1	C	38	Flight software
KC1	C++	22	Storage management

5. RESULTS

In this section, the comparison between proposed system and the existing system is discussed. In the proposed work NASA defect dataset is used and it is build on java language. The rules are generated with different minimum support thresholds and different minimum confidence thresholds. Figure 2 shows the comparison of rules generated for existing system and proposed system. From this analysis it is observed that in existing system lot of unnecessary rules are generated because of that software manager or user get confused therefore more defects occur in software. In order to solve this problem hybridized approach is adopted which gives only interesting rules. For each data set, the average number of rules decreases as the minimum support increases from 20 percent to 40 percent, and this decrease in rules is very sharp when minimum support exceeds 30 percent. Figure 3 shows the comparison of rules generated by minimum confidence. It is observed that for each data set, the average number of rules decreases as minimum confidence increases from 30 percent to 50 percent.

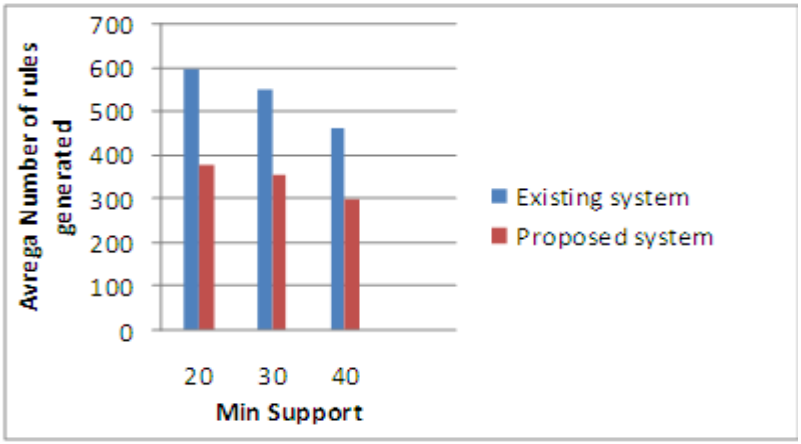

Figure 2 Rules are generated for minimum support

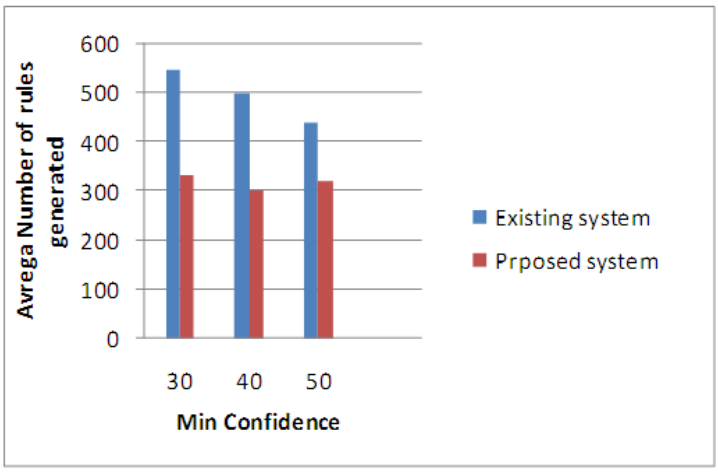

Figure 3 Rules are generated for minimum confidence

6. CONCLUSION AND FUTURE WORK

In this paper, association rule discovery for detecting software entities that are likely to be defective in software systems. This approach is useful to evaluate software defects. When a problem arises due to the increasing complexity of a program, then solutions are being submitted by finding software defect. The main feature that distinguishes our approach from others is using a k-means and Apriori method. It is possibly the best algorithm for the software defect problem. Standard dataset have been used for experimental purpose. The focus is to improve the quality and feasibility of the software. In our scenario, the result heavily depends on the accuracy of rules generation and based on that it will predict the software defects. The results show that proposed system generating only interesting rules which is more useful for predicting defects in software.

Our future research focuses on elaborating with machine learning or fuzzy techniques that will further improve accuracy of predicting software defects.

Acknowledgment

We would like to thank the NASA Software Engineering Laboratory (SEL) for providing the defect data for this analysis and publishers, researchers for making their resources available and teachers for their guidance. Lastly, we would also like to extend a heartfelt gratitude to friends and family members who have inspired, guided and assisted us in performing this work.

References

[1]. C. Catal, Software fault prediction: “A literature review and current trends, Expert systems with applications”, vol. 38, no. 4, pp. 4626-4636, 2011.

[2]. Y. Kamei, A. Monden, S. Morisaki, and K.-i. Matsumoto, “A hybrid faulty module prediction using association rule mining and logistic regression analysis”, in Proceedings of the Second ACM-IEE international symposium on Empirical software engineering and measurement, pp. 279-281, ACM, 2008.

[3]. G. Czibula, Z. Marian, and I. G. Czibula, “Detecting software design defects using relational association rule mining,” Knowledge and Information Systems, pp. 1-33, 2012.

[4]. Tao WANG, Weihua LI, Haobin SHI, Zun LIU, “Software Defect Prediction on Classifier Ensemble”, Journal of Information & Computational Sciences, 8:16(2011) 4241-4254.

[5]. R. Malhotra, “Comparative analysis of statistical and machine learning methods for predicting faulty modules,”Applied Soft Computing, vol. 21pp. 286-297 2014.

[6]. M. M. T. Thwin and T.-S. Quah, “Application of neural networks for software quality prediction using object-oriented metrics,”Journal of systems and software, vol. 76, no. 2, pp. 147-156, 2005.

[7]. D. Radjenovic, M. Hericko, R. Torkar, and A. Zivkovi, Software fault prediction metrics: A systematic literature review, Information and Software Technology, vol. 55, no. 8, pp. 1397-1418 ,2013.

[8]. M. Shepperd, Q. Song, Z. Sun, and C. Mair, “Data quality: some comments on the nasa software defect datasets,” Software Engineering, IEEE Transactions on, vol. 39, no. 9, pp. 1208 -1215, 2013.

[9]. E. Erturk and E. A. Sezer, “A comparison of some soft computing methods for software fault prediction,” Expert Systems with Applications, 2014.

[10]. M. Shepperd, D. Bowes, and T. Hall, “Researcher bias: The use of machine learning in software defect prediction,” Software Engineering, IEEE Transactions on, vol. 40, no. 6, pp. 603-616, 2014.

[11]. Campan, G. Serban, T. M. Truta, and A. Marcus, “An algorithm for the discovery of arbitrary length ordinal association rules.,” DMIN, vol. 6, pp. 107-113, 2006.

[12]. Okutan, Ahmet, and Olcay Taner Yıldız. "Software defect prediction using Bayesian networks." Empirical Software Engineering 19.1 (2014) 154-181

[13]. J. Nam, Survey on software defect prediction," 2010.

[14]. D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, "Software defect prediction using static code metrics underestimates defect-proneness," in Neural Networks (IJCNN), International Joint Conference on, pp. 1-7, IEEE, 2010.

[15]. Naidu, M. Surendra, and N. GEETHANJALI. "Classification of Defects in Software using Decision Tree Algorithm." International Journal of Engineering Science and Technology (IJEST) 5.06 (2013).

[16]. E. E. Mills, Software metrics, 2000.

[17]. Dataset available: http://promise.site.uottawa.ca/SERepository/datasets

[18]. Pooja Paramshetti, Mrs D.A. Phalke. "Survey on software defect prediction using machine learning tecniques" IJSR 2014.

[19]. G. Czibula, Z. Marian, and I. G. Czibula, "Software design defects using relational association rule mining," Knowledge and Information Systems, 2014.

[20]. Qinbao Song, Martin Shepperd, Michelle Cartwright, Carolyn Mair, "Software Defect Association Mining and Defect Correction Effort Prediction" , IEEE transaction on software engineering VOL. 32, NO. 2, Feb 2006