ERA -An Enhanced Ripper Algorithm to Improve Accuracy in Software Fault Prediction

N.Vinothini #[1](Research Scholar)
School Of Computer Science and Engineering,
Bharathidasan University, Tiruchirappalli,
Tamil Nadu, India.
senthavinocs@gmail.com #[1]

Dr. E. George Dharma Prakash Raj *[2](Assistant Professor)
School Of Computer Science and Engineering,
Bharathidasan University, Tiruchirappalli,
Tamil Nadu, India.
georgeprakashraj@yahoo.com *[2]

Abstract

The data mining Techniques are used and applied in the various fields of science. The software mining is an application of data mining, which describes investigation into the examination of software Repositories, such as software assurance, reuse, fault, effort or cost prediction and detection of incomplete changes. Fault in software system is a deficiency that causes software failure and also affects software quality, cost and time. Software quality assurance is major vital to increase the level of the developed software. In existing system different types of rule based classification Algorithms are used for the fault prone modules, In Proposed , A new rule based classification algorithm ERA(Enhanced Ripper Algorithm) has be introduced. The ERA is the enhanced form of Ripper algorithm, in this algorithm we classify the software modules through Rule's and it is mainly designed to minimize the Rule set length. For that we improve classification accuracy. Which include the attribute selection, to improve accuracy and software efficiency. In this research Enhanced ripper algorithm applied on publicly available datasets of NASA Repository and predicted the fault modules.

Keywords: data mining, software mining, software efficiency, software defect, Ripper, NASA Dataset, Rule extraction

I. INTRODUCTION

Data mining refers to extracting or ―"mining" knowledge from large amounts of data. Knowledge Discovery in Data is the non-trivial process of identifying valid, novel, potentially helpful and ultimately understandable patterns in data. Data mining consists of more than collection and managing data; it also includes analysis and prediction. People are frequently do mistakes while analysing or, possibly, when trying to set up relationships between multiple features.

Classification is a model finding process that is used for portioning the data into different classes according to several constrains. In other words we can say that classification is process of generalizing the data according to different instances An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The term "classifier" sometimes also refers to the mathematical function, implemented by a classification algorithm that maps input data to a category. Many classification and prediction methods have been proposed by researchers in machine learning, pattern recognition, and statistics. Most algorithms are memory resident, typically assuming a small data size. Recent data mining research has built on such work, developing scalable classification and prediction techniques capable of handling large disk-resident data.

Software mining is the application of data mining techniques to improve and support the management activities of software projects. The term software mining describes a broad class of investigations into the examination of software repositories, such as; software quality, fault, effort, or cost prediction, software reuse, and detection of incomplete changes.

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 softwareThis Research focuses on the classification of software modules into either faulty or correct modules through the use of ERA-Enhanced Ripper Algorithm. The aim of this paper is to apply the ERA-Enhanced Ripper Algorithm on some publicly

available datasets of the NASA software repository in order to classify the software modules into either fault or not fault prone modules. The selected datasets are: PC1, CM1, KC1, and KC2.

II. RELATED WORK

Several papers are discussed about using mining for software fault prediction [1, 2, 4, 5, 6, 7, 8, 9, 10, 11]. Some of those papers discussed methods for fault prediction such as size and complexity metrics, multivariate analysis, and multi-co-linearity using various data mining algorithms. Pradeep Singh, Shrish Verma [1] proposed, Clustering based classification allows production of comprehensible models of software faults exploiting symbolic learning algorithms. To evaluate this approach we perform an extensive comparative analysis with benchmark results of software fault prediction for the same data sets. Sonali Agarwal and Divya Tomar sona [2] proposed a feature selection based Linear Twin Support Vector Machine (LSTSVM) model to predict defect prone software modules. F-score, a feature selection technique, is used to determine the significant metrics set which are prominently affecting the defect prediction in a software modules. The efficiency of predictive model could be enhanced with reduced metrics set obtained after feature selection and further used to identify defective modules in a given set of inputs. P.A. Selvaraj and Dr.P. Thangaraj [4] proposed to investigate the classification accuracy of Support Vector Machine (SVM) for software defect prediction using different kernels. W. Nor Haizan W. Mohamed, Mohd Najib Mohd Salleh, Abdul Halim Omar [5] proposed decision tree based software fault prediction .decision tree is one of the most popular and efficient technique in data mining. Some decision tree algorithms may produce a large structure of tree size and it is difficult to understand. Furthermore, misclassification of data often occurs in learning process. Therefore, a decision tree algorithm that can produce a simple tree structure with high accuracy in term of classification rates a need to work with huge volume of data. Pruning methods have been introduced to reduce the complexity of trees structure without decrease the accuracy of classification. One of pruning methods is the Reduced Error Pruning (REP). In data modelling, J48 and REPTree generate tree structure as an output while PART, Ridor and JRip generate rules. In additional J48, REPTree and PART using REP method for pruning while Ridor andJRip using improvement of REP method, namely IREP and RIPPER methods. This paper shown performance of J48 and REP Tree are competitive in producing better result [5]. Hassan Najadat. [6] Proposed that a modification on RIDOR (Ripple Down Rule) algorithm that is they improved the effectiveness of RIDOR algorithm and that algorithm is refereed as Enhanced RIDOR algorithm. This enhanced algorithm learns defect prediction using mining static code attributes. These attributes are used to propose a new defect predictor with high accuracy and low error rate [6]. A. Okutan O. T. Yıldız used Bayesian networks [7] to determine probabilistic influential relationships between software metrics and defect proneness. Two metrics - NOD for developer's number and LOCQ for source code quality - were defined. They were extracted through an inspection of the source code repositories of selected Promise data repository data sets. At the end of modelling, software system's defect proneness probability, most effective metrics set, and relationships between metrics and defects was understood. Meenakshi P.C, Meenu S, Mithra M, and Leela Rani P [8] applied Expectation Maximization (EM) algorithm and Quad tree concept for predicting faulty. Akalya devi.C, Kannammal. K.E and Surendiran.B, [9] proposed a hybrid feature selection method which gives the better prediction than the traditional method. For evaluating the performance of software fault prediction models they used accuracy, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) [9]. Shanthini.A and Chandrasekaran.RM [10] focused on high performance fault predictors that are based on machine learning algorithm. They used Method level metrics and Class level metrics for one type of data set. Support Vector Machine (SVM) provides the best prediction performance in terms of precision, recall and accuracy. Method level metrics are suitable for both procedural and object oriented programs. Class level metrics are only suitable for object oriented programs. They used four types of classifiers are: Naïve Bayes, K – Star, Random Forest and SVM. Their future work is to predict the software models based on some other machine learning algorithm [10]

III. Methodology

PC1, KC1, KC2, CM1 dataset are available from PROMISE software dataset repository [3]. All these dataset are used for software defect prediction. This paper used these datasets. This dataset contains several software metrics, software **metric** is a measure of some property of a piece of software or its specifications. such as Line of Code, number of operands and operators, Design complexity, Program length, effort and time estimator and various other metrics. Each data set is comprised of a number of software modules (cases), each containing the corresponding number of defects and various software static codes attributes. The dataset includes McCabe and Halstead features code extractors. The measures are module based. The defect detectors are assessed as follows:

a = Classifier predicts no defects and module actually has no error.

b= Classifier predicts no defects and module actually has error.

c = Classifier predicts some defects and module actually has no error.

d = Classifier predicts some defects and module actually has error.

A. Rule-Based Classication

Rules are a good way of representing information or bits of knowledge. A rule-based classier uses a set of IF-THEN rules for classication. An IF-THEN rule is an expression of the form-

IF condition **THEN** conclusion

B. Enhanced Ripper Algorithm

Software quality assurance is necessary to increase the level of confidence in the developed software and reduce the overall cost for developing software projects. The problem addressed in this research is the prediction of fault prone modules using ERA-Enhanced Ripper Algorithm. Predicting fault prone modules allows the software managers to allocate more testing and resources to such modules. This can also imply a good investment in better design in future systems to avoid building error prone modules.

Enhanced Ripper Algorithm follows common procedure of ripper algorithm, ERA is a classification algorithm designed to generate rules set directly from the training dataset. A new rule associated with a class value will cover various attributes of that class .The algorithm was designed to be fast and effective. Enhanced Ripper Algorithm was mainly designed for improving the accuracy in software fault prediction; in this algorithm we reduced length of rule set,because,increased rule set length is not applicable for all types of classification, it depends upon the type of dataset's or classification. if the rule set length is maximum and number of attributes is minimum looping occurs, and it will be time consuming, it's also affects accuracy of classification, proposed algorithm solves this problem. In this algorithm use attribute selection method which is the process of selecting a subset of relevant features for use in good software model construction. This method facilitates identification of software fault modules.

Step 1: S = {X, C} represents the training set, where X = {x1, x2… xk} represents the instances and C = {c1, c2… ck} represents the class-label associated with each instance.

Step 2: The classes c1… ck are sorted in the order of least prevalent class to the most frequent class. This is done by counting the number instances associated with each class. The instances associated with the least prevalent class are separated into SPos subset whilst the remaining instances are grouped into Sneg subset.

Step 3: Create a new Rule Set{}, define the length of rule set, the Spos not in rule set, then split the Growpos,growneg,prone pos,prone neg.

Step 4: two rules are created one for Grow, and another one for pruning.

Step 5: IREP is invoked (with SPos and Sneg subsets passed as parameters) to find the rule set that splits least prevalent class from the other classes.

Step 6: Initialise an empty rule set R.

Step 7: SPos and Sneg are split into growing positive Gpos and growing negative Gneg subsets as well as pruning positive Ppos and negative Pneg subsets. Growing positive subsets contains instances that are associated with the least prevalent class. Growing negative subset contains instances associated with the remaining classes. This is similar to the Ppos and Pneg subsets.

Step 8: A new rule is created by growing Gpos and Gneg. This is done by iteratively adding conditions that maximize the information gain criterion until the rule cannot cover any negative instances from the growing dataset.

Step 9: The new rule is pruned for optimization of the function

C. Attribute Selection

In the above algorithm mainly focused on four sets of attributes, the are total number of operator, total number of operand, n,d, attridutes in KC1,KC2,CM1,PC1 dataset ,because these type of attributes are very important in software quality assurance, Software quality assurance is necessary to increase the level of confidence in the developed software and reduce the overall cost for developing software projects. the total number of operator and operand is very important in software making, any defects in these kinds of attributes will affect software efficiency. Operator should also maintain a detailed understanding of the job processing requirements and data flow for all applications systems. -Attribute d denotes difficulty measure. The difficulty measure is related to the difficulty of the program to write or understand the result. Discussion part describe the overall accuracy, precision,f-measure and recall of this attributes.

IV. IMPLEMENTATION PROCESS

Step1: Load the Software defect dataset from PROMISE repository.

Step2: Perform pre-processing of the dataset

Step 3: select (or) software metrics

Step 4: then generate rule set for all choosing software metrics, rule set was generated based on static code attribute value

Step5: then apply the confusion matrix methodology

Step6: predicting accuracy, true positive, true negative, false positive, false negative, precision, recall, f-measure.

Step7: Then plot the ROC.

Step7: Compare the attribute selection result with overall results of the attributes

Step 8: Select the feature subset showing highest accuracy.

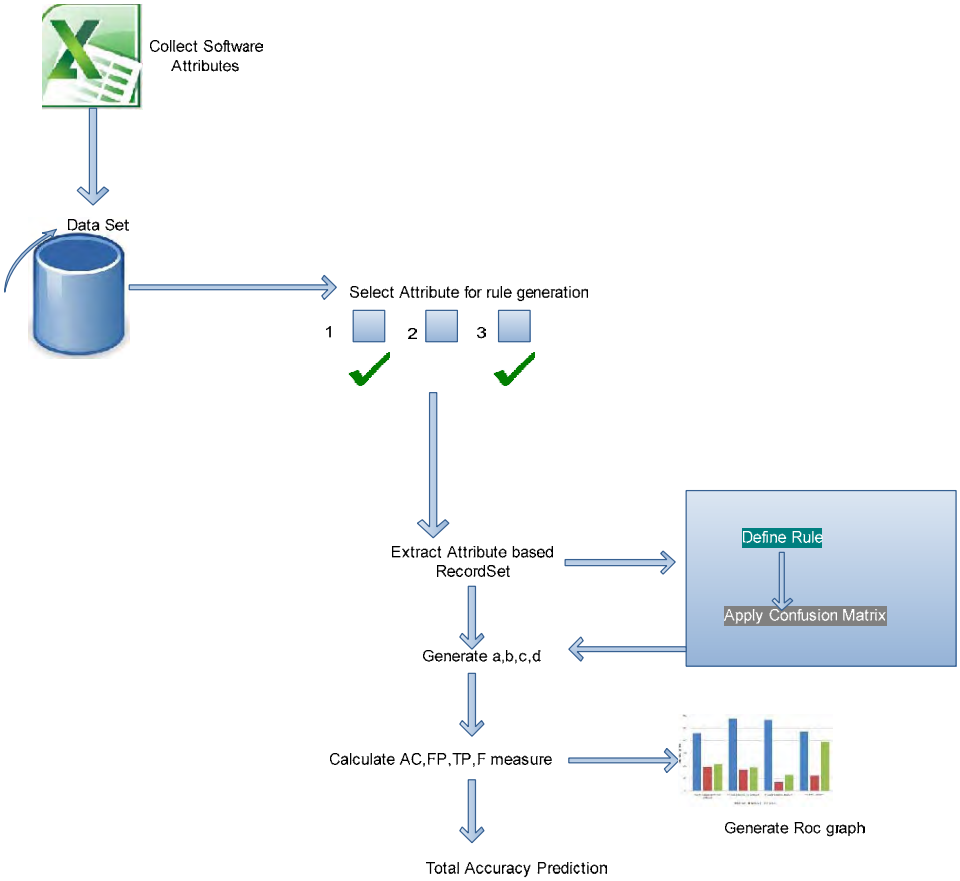

Fig 1: proposed model

V. PROPOSED MODEL PERFORMANCE EVALUATION

The performance of proposed model is measured with the help of confusion matrix which store the results of classifier in the form of actual and predicted class as indicated in Table 1. Using confusion matrix, we can estimate accuracy, specificity, Precision and F-measure which further utilized for performance evaluation of proposed model.

Accuracy: Accuracy is also referred as "correct classification rate" and is measured by taking the ratio of correctly prediction to the total prediction made by the software defect prediction model and is formulated as:

Accuracy= (TP+TN)/(TP+ FN +FP+ +TN)

Recall: Recall, also called true positive rate, is estimated by calculating the % of correctly identified not-defective software modules and is formulated as:

Recall= TP/ (TP+FN)

Precision: Sometime it is also referred as correctness and is measured by taking the proportion of correctly recognized defect free modules and total predicted not-defective software modules by classifier and is formulated as:

Precision= TP/ (TP+FP)

F-Measure: It is measured by taking the harmonic mean of precision and sensitivity and is calculated as

F-Measure= (2 *Recall*Precision)/ (Recall + Precision)

Table 1: Confusion Matrix Result

Actual Class	**Predicted Class**	
Defective	**Defective**	**Not Defective**
	True Positive (TP)	**False Positive (FP)**
Not Defective	**False Negative (FN)**	**True Negative (TN)**

Table 2: Result of proposed algorithm

Dataset	**Accuracy**	**Precision**	**Recall**	**F-measure**
KC1	**88.25%**	**0.967**	**0.865**	**0.915**
KC2	**87.5%**	**0.89**	**0.89**	**0.885**
CM1	**93.25%**	**0.85**	**0.94**	**0.898**
PC1	**94%**	**0.89**	**0.942**	**0.915**

The table 2 describes the classification accuracy, precision, recall, F-measure of the selected attributes in KC1, KC2, CM1, PC1 dataset.

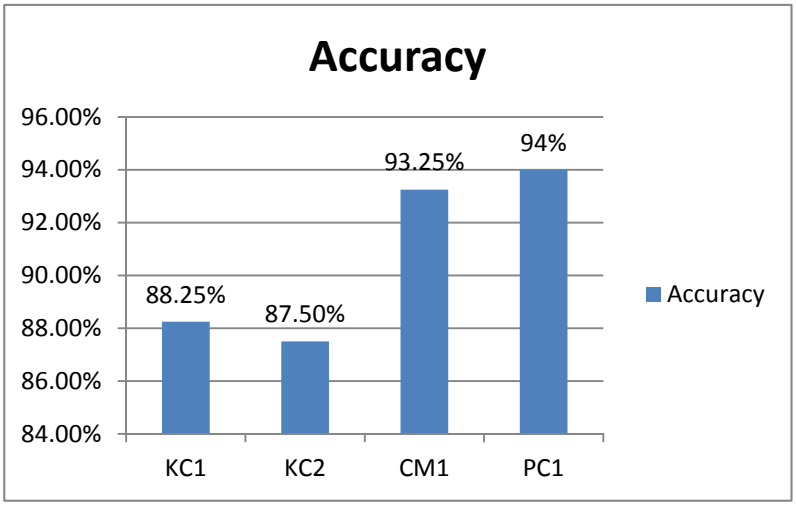

Fig 2: Classification accuracy of KC1, KC2, CM1and PC1 Dataset

VI.CONCLUSION

The main objective of fault prone modules‟ prediction using data mining is to improve the software development process”. This enables the software manager to effectively allocate project resources toward those modules that require more effort. This will eventually enable the developers to fix the bugs before delivering the software product to end users. This research proposed a Enhanced Ripper algorithm for software defect prediction. It improves the accuracy of the prediction.

Advantages

- ✓ Easy to interpret
- ✓ Easy to generate
- ✓ Can classify new instances rapidly
- ✓ Performance comparable to decision trees
- ✓ It produce the accurate result

VII . REFERENCES

[1] Pradeep Singh, Shrish Verma “An Efficient Software Fault Prediction Model using Cluster based Classification” international Journal of Applied Information Systems (IJAIS) Volume 7– No. 3, May 2014. ISSN: 2249-0868.

[2] Sonali Agarwal and Divya Tomar “A Feature Selection Based Model for Software Defect Prediction” International Journal of Advanced Science and Technology (IJAST) Vol.65 (2014), pp.39-58, ISSN: 2005-4238 IJAST.

[3] Software Defect Dataset, PROMISE REPOSITORY, http://promise.site.uottawa.ca/SERepository/datasets-page.html, (2013) December 4.

[4] P.A. Selvaraj1 and Dr.P. Thangaraj” Support Vector Machine for Software Defect Prediction” International Journal of Engineering & Technology Research (IASTER) Volume 1, Issue 2, October-December, 2013, ISSN Online: 2347-4904.

[5] W. Nor Haizan W. Mohamed, Mohd Najib Mohd Salleh, Abdul Halim Omar “A Comparative Study of Reduced Error Pruning Method in Decision Tree Algorithms” International Conference on Control System, Computing and Engineering, 23 - 25 Nov. 2012, Penang, Malaysia ,978-1-4673-3141-8©2012.

[6] Hassan Najadat and Izzat Alsmadi “Enhance Rule Based Detection for Software Fault Prone Modules “International Journal of Software Engineering and Its Applications (IJSEIA) Vol. 6, No. 1, January, 2012.ISSN-1738-9984.

[7] A. Okutan O. T. Yıldız, “Software defect prediction using Bayesian networks”, Empirical Software Engineering, (2012), pp. 1-28. DOI 10.1007/s10664-012-9218-8

[8] Meenakshi P.C, Meenu S, Mithra M, Leela Rani P,” Fault Prediction using Quad Tree and Expectation Maximization Algorithm”, International journal of Applied Information System (IJAIS) Volume 2– No.4, May 2012 , ISSN : 2249-0868.

[9] Akalya devi.C, Kannammal. K.E and Surendiran.B,” A Hybrid Feature Selection Model for Software Fault Prediction”, International Journal on Computational Sciences & Applications (IJCSA) Vo2, No.2, April 2012. DOI: 10.5121/ijcsa.2012.2203.

[10] Shanthini. A Chandrasekaran.RM,” Applying Machine Learning for Fault Prediction Using Software Metrics”, International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), Volume 2, Issue 6, June 2012, ISSN: 2277 128X.