A New Approach to Find Predictor of Software Fault Using Association Rule Mining

Dipti Kumari#1, Kumar Rajnish#2

Department of Computer science and engineering,
BIT, MESRA, Ranchi, Jharkhand-835215, INDIA
1kumari_dipti0511@yahoo.co.in
2krajnish@bitmesra.ac.in

Abstract-In this paper, we use a new method to find the best predictor of software fault using association rule mining. The method, first of all select all the association rules having confidence greater than 40% and support greater than 30% using Apriori algorithm. After that our aim is to select top 'n' association rules out of a pool of 'k' association rules based on heuristic analysis. The method ranks association rules giving weight to a larger set of parameters than used by standard methods. The role of correlation has been emphasized in this method which also tries to eliminate issues faced in incorporating correlation, support and confidence expressively into a single fitness function. A least square regression analysis has been done to establish the best rules in a set of "good" rules and allows for pruning of misleading rules that are often suggested by standard algorithms like the Apriori method. Furthermore, we investigate which OO-metrics are related to each other by best rules. The metrics on the antecedent part make sure the occurrence of the consequent part metrics. So, those OO-metrics which are present in the rule at antecedent part in most of the rules can be used as best predictor in software fault. It is found that applying this method results in both accurate and comprehensible rule sets as well as best predictor of fault.

Keywords: Software engineering, Defect prediction, Data mining, Association rule, support, confidence, correlation, lift, cosine.

I. INTRODUCTION

Software quality [1] is considered of great importance in the software engineering field. On the other hand, building software of high quality is very expensive. Thus, in order to increase the efficiency and use of quality assurance and testing, software defect prediction is used to identify defect-prone modules in a forthcoming version of a software system and help to allocate optimized required effort on those modules [2].So, developing software fault prediction model with best predictor will help very much in this perspective. But, finding the best predictor i.e. OO-metric is also a typical task. For solving this type of problem when software has more than thousands and thousands modules and their respective more than hundreds and hundreds OO-metrics. In this scenario Data mining helps very much. As we know Data mining refers to extracting or "mining" knowledge from large amounts of data. Data mining is one of the most important tools which extracts, manipulates data and establishes a pattern which helps in decision making[3].The branch of data mining that deals in discovery of interesting associations and correlations between item sets in transactional and relational databases is called frequent pattern mining. The most important frequent pattern mining application is mining association rules. In 1993, R. Agrawal and R. Srikant first introduced the association rule mining [4]. Association rule mining (ARM) is a very popular and well researched method for discovering relationship between variables in large databases [5]. Association rules are the rules that correlate the presence of one set of items with that of another set of items. It extracts frequent item sets, interesting rules and discovers the relationship among items in transactional database or in other data repositories [6]. ARM generates the best association rules which qualify the minimum support threshold and minimum confidence threshold. Association rule can be used to improve decision making in various areas such as: market basket strategy, process mining, protein sequences, logistic regression, medical diagnosis, bio-medical literature, web search, CRM of credit card business etc. Many researchers have shown that selecting the right objective measures is a very important factor to be considered [7]. A. Silberschatz and A.Tuzhilin proposed an approach about the interestingness pattern [8]. Many algorithms have been proposed to generate frequent item sets: Apriori algorithm, Éclat and FP-Growth. The Apriori algorithm is an iterative level-wise algorithm which is used to find frequent pattern in data [9]. Improved Apriori algorithm [10-12] removes the unnecessary transactional records from the database which reduces scan time in large amount and also reduces the redundant generation of sub items during pruning the candidate set. However, improved mining algorithms performance and its complexity is subject to research area, as they have to deal with the large set of data items. Recent works involve different usage of correlation measure [13-14]. Also, there have been some soft computing approaches using algorithms like genetic algorithm, ant colony optimization, etc [15-18]. There

are not so many works has been done in the field of Software fault prediction using association mining [19-21].This work assesses the traditional way of frequent pattern mining using Apriori algorithm and introduces the concept of F-measure by using the notion of correlation i.e., association rule is generated by considering three factors, support, confidence and correlation: A=>B [support, confidence, correlation]. Correlation is calculated by using the "Lift" measure. F-measure is the linear summation of the support, confidence and correlation of each rule with the unknown coefficient α, β, and γ. The values of unknown coefficient are generated by using the Least Square Regression. According to F-measure values, best association rules will be generated. Higher the F-measure value, better the association rule will be. This paper uses this approach for the problem of finding the predictor of software fault i.e. OO-metrics. The fault prediction based on the idea of discovering best association rules within a dataset. Best association rules helps to find the OO-metrics on which other OO-metrics are dependent. So, those OO-metrics which are found in most of the rules and also found in the antecedent are taken as the predictor. And use it for predicting the software fault for Eclipse version. The results obtained by evaluating the classification model by applying this association rule mining for defect prediction is promising and indicate the potential of our proposal.

The rest of the paper is organized as follows: In Section 2, we discuss about Dataset. In section 3, we describe about the research methodology. In section 4, we describe the approach used by the study. In Section 5, we present our experimental results. Finally, in Section 6, we summarize our work and findings by conclusion and future scope.

II. EXPERIMENTAL DESIGN

The objective of the study is to find the software metrics which can be used as best predictor of software fault in OO system. The following steps are followed to achieve the objective:

- To select the software metrics.
- To collect the data- the metrics as well as errors data.

In this study we identify best metrics for fault prediction by analyzing the association rule mining between metrics in Eclipse- a widely used industrial-strength system. We chosen Eclipse because it is Open-Source System and the error data are also obtainable .Furthermore, there are several versions of Eclipse available for analysis. We collected the software metrics from three releases of Eclipse (Versions 2.0, 2.1, and, 3.0) and error data from [22] [23]. In the following section, we present how we selected and collected the software metrics in the study.

A. The Selection of Software Metrics

The selection of software metrics was a difficult task because there are many available metrics. We used two criteria in our selection process:

- The set of metrics cover all aspects of OO design.
- We have to be able to collect the metrics by using automated tool.

Finally, we selected 17 class level Object-Oriented metrics which are discussed in Table 1 as follows:

TABLE I. Selected Object-Oriented Metrics description

Metrics used	Description
NOS	Total number of Java statements (alternative to lines of code)
UWCS	Unweighted Class size
CC	Class Complexity
RFC	Total Response For Class
NLOC	Total Lines of Code in the class
EXT	No. of External Methods called
MPC	Message passing coupling (MPC) value
LMC	No. of local methods called
TCC	Total Cyclomatic Complexity
PACK	No. of packages imported
NOM	Number of Methods
LCOM2	Lack of Cohesion of methods
INST	No. of instance variables declared
CBO	CBO (Coupling Between Objects)
MAXCC	Maximum Cyclomatic Complexity
FOUT	Fan Out (Efferent Coupling)
AVCC	Average Cyclomatic complexity

These metrics cover all aspects of class level OO design due to this reason they are belonging to coupling, cohesion, inheritance, class complexity and class-size metrics. We used JHAWK [24] automated tool metric to collect these metrics from the Eclipse source code [25]. JHAWK compiled the source code and give output as each module name and their set of OO metrics. In the next section, we describe how we collected the error data.

B. Collection of Error Data

From [26] where Eclipse bug data set are freely available, we collected the error data from three official releases of the Eclipse project (Versions 2.0, 2.1, and 3.0). Pre release bug data are used for study and multinomial categorization has been done on the pre release error data.

Multinomial Categorization: In this we divide the error severity into 4 classes.

For classification our followed steps are:

- We find the descriptive statistics of pre error data. From that we are able to know the min, different number of occurrences of error (nonzero) and max value of error data in all classes of every versions of Eclipse.
- After that, we again find the descriptive statistics of (Min, 25Q, Mean, 75Q and Max) the different occurrences of number of errors (from min (nonzero) to max).
- Based on that we classified class error data into one of five categories that are defined as follows:
  - Nominal: class containing error in the range Min<=error<25Q
  - Low: class containing error in the range 25Q<=error<Mean
  - Medium: class containing error in the range Mean<=error<75Q
  - High: class containing error in the range 75Q<=error<Max

III. RESEARCH METODOLOGY

In this section, the steps taken to analyze the selected OO-metrics as a best predictor of software fault for classes taken for analysis are described in following stages: (i) data preprocessing, (ii) Association Rule Mining, (iii) Apriori Algorithm, (iv) ROC curve Analysis and (v) Metrics Evaluation

A. Data Preprocessing

We are using Apriori Algorithm for finding the best predictor of software fault. This algorithm takes dataset in the form of nominal values. So, we have to change the quantitative value of data to nominal value. Some preprocessing has to be done on the dataset so as to make it compatible for using the Apriori algorithm.

1) Profile range development of different OO-metrics: In this section we use a technique for define a multinomial profile range of all OO-software metrics.

Let D is a dataset containing N Classes and having training patterns like (O1,O2,...,Oj,...,Ok) of k OO-metrics. The OO-metric Oj have n values v1j,v2j,...,vNj).Oj_min and Oj_max denote the minimum and maximum value of OO-metric Oj. When in the dataset the OO-metric has numeric value, then following steps are performed for profile range development.

Step 1. Sort the values of OO-metric Oj in ascending order.

Step 2. Perform K-means clustering algorithm for clustering the quantative values of the OO-metric Oj into 4 clusters (c1, c2, c3 and c4) where, ci_min and ci_max denote the minimum and maximum value of ith cluster (ci).

Step 3. Find the cluster centers(cc1,cc2,...,cci,...,cck) of k clusters(c1,c2,...,ci,...,ck).

Step 4. Two boundary points (lowest and highest) of every cluster gives the range value falling in "Nominal", "Low", "Mid" and "High" categories. One fact we have to keep in mind that at least more than two data comes in one cluster. Otherwise we, have to decrease the no. of categories.

B. Association Rule mining

Relationship between the data is called association. Association rule shows attribute value conditions which occur most frequently in the given dataset. In general, Association rules are expressed in the form X→Y, where X and Y are item sets (collection of items) representing the antecedent and the consequent part of the rule and both X and Y do not intersect each other (disjoint), they do not have common items. Association rule may have more than one item in antecedent (X) and consequent(Y) part. The complexity of rules depends upon the number of items it contains. Association rule mining (ARM) finds interesting associations and correlation among the data in a given dataset [27]. Support and confidence are two measures or rule interestingness.

The strength of association rule depends upon following factors:-

- Support or prevalence: - It is simply the number of transactions that contain all the items in the antecedent and consequent parts of rule. Thus, the rule has support S in dataset D, if S% of the transactions in D contains both X and Y i.e. (X ∪ Y).

$$\mathrm{Supp}(X \rightarrow Y) = P(X \cup Y)$$

- Confidence or predictability: - It is a ratio of the number of transactions that contain all items in the consequent as well as in the antecedent (namely, support) to the number of transactions that contain all items in antecedent. A rule is said to hold on D, if the confidence of the rule is greater than or equal to confidence threshold. Thus, the rule has confidence C, if C% of the transactions in dataset D that contain X also contains Y.

$$\mathrm{Conf}(X \rightarrow Y) = \frac{\mathrm{Supp}(X \cup Y)}{\mathrm{Supp}(X)}$$

- Correlation: It finds the actual relationship between two or more items whether it is negatively or positively associated. It measures the strength of the implication between X and Y. It prunes out the large number of negatively associated rules. Thus, actual interesting association rules are generated based upon support, confidence and correlation value. Correlation is measured by one of the correlation measure such as Lift after getting the strong association rules.

$$\mathrm{Lift}(X, Y) = \frac{\mathrm{Conf}(X \rightarrow Y)}{\mathrm{Supp}(Y)} = \frac{P(X \cup Y)}{P(X) \times P(Y)}$$

It is the simple correlation measure of how much better the rule is doing. If P (XUY) = P (X).P(Y) then the occurrence of an item set X is independent of the occurrence of an item set Y, else they are correlated or dependent. If the resulting value is less than1, then X and Y are negatively correlated. If it is greater than 1, then X and Y are positively correlated, meaning that the occurrence of one will implies the occurrence of the other and if resulting value is equal to 1 , then X and Y are independent mean there is no correlation between them. It is also referred as the lift of association rule X→Y, as it tend to lift the occurrence of an item with the other items.

*1) Disadvantage of Association rule:*Many researchers have given the drawbacks of association rules in their paper [28]:-

- Discovering too many association rules: The traditional association rules mining (ARM) algorithms were very simple and efficient. However, ARM algorithms generate a large number of association rules and it does not give the actual information that the rules generated are relevant or not.
- Strong rules generated can be misleading and uninteresting: The traditional ARM algorithm is based upon a support-confidence framework. A large number of association rules are generated by using low support thresholds. Although minimum support and minimum confidence threshold helps to prune out

a good number of rules, many rules found are still not interesting to the users. This truly happens when mining for long patterns or when mining at low support thresholds.

- Does not consider effect of correlation: The traditional ARM algorithm does not measure the strength of the correlation and implication between X and Y. It does not give any information about negative association among items which leads to unwise decisions based on rules.

In this paper, we have used three parameters: - support, confidence and correlation in order to remove the drawbacks of association rules to a large extent. It also gives negative correlation which is not identified by the traditional ARM following support-confidence framework.

C. *Apriori Algorithm*

It is also known as level-wise algorithm. It was introduced by R.Agrawal and R.Srikant in 1994[29]. It is the most popular algorithm for mining frequent item sets for Boolean association rules. Apriori consists of two important steps: the first step is to find the frequent item sets among the given number of transactions, and second step is to extract the rules from the mined frequent item sets. It requires the prior knowledge of frequent item sets. It uses the downward closure property. Apriori algorithm uses the bottom-up search method, moving towards upward level-wise in the lattice. Before reading the database at every level, it prunes out the infrequent sets. If there is any item set which is infrequent, then its superset should not be tested /generated.

Steps of this Algorithm are as follows:

Step 1. Initially scan the database DB to accumulate the count for each item and retain those that satisfy minimum support, to generate frequent 1-itemset.

Step 2. Frequent k-item sets is used to generate (k+1) candidate item sets.

Step 3.Test the candidates against DB.

Step 4.Terminate when no candidate set can be generated or it is unlikely to be frequent (fails to meet the minimum support threshold).

In this paper, Apriori algorithm is used to generate association rules using the support and confidence threshold.

D. Least Square Regression

The objective [30] of Least square Regression consists of adjusting the parameters of a model function to best fit a data set. A simple data set consists of n points (data pairs) (x_i,y_i), i = 1, ..., n, where x_i is an independent variable and y_i is a dependent variable whose value is found by observation. The model function has the form f(x,β), where m adjustable parameters are held in the vector β. The goal is to find the parameter values for the model which "best" fits the data. The least squares method finds its optimum when the sum, S, of squared residuals

$$S = \sum_{i=1}^{n} r_i^2$$

is a minimum. A residual is defined as the difference between the actual value of the dependent variable and the value predicted by the model.

$$r_i = y_i - f(x_i, \beta)$$

An example of a model is that of the straight line in two dimensions. Denoting the intercept as β_0 and the slope as β_1, the model function is given by

$$f(x, \beta) = \beta_0 + \beta_1 x$$

To calculate F-measure, we take the Support, Confidence and the correlation. The formula for F-measure is:

$$F - measure = k + \alpha * Support + \beta * Confidence + \gamma * Correlation$$

Where, the weights α , β , γ and the constant k are derived by least square regression analysis. Constraint used in F-measure: $(\alpha + \beta + \gamma = 1)$ and $(\gamma > \alpha > \beta)$.

Reason for applying this constraint is that we are giving more stress on correlation after that support and then confidence, to get the best rule so that we are able to find the best predictor of software fault.

E. ROC Curve

ROC is a diagnostic accuracy test [31]. The ROC method can be used to assess the quality of the information provided by the classification of classes into a binary category using a metric. To plot the ROC curve, we need to define two variables: one binary (i.e., 0 or 1) and another continuous. In our study, we are using the Multinomial categorization. The classes in the Multinomial categorization should be considered one by one, i.e., we need to plot the ROC curve for each category (Nominal, Low, Medium, and High) leaving the No-error category as the option. The continuous variable in both categorizations is the metric used in the

study. There are four possible outcomes from a binary classifier. From P positive instances and N negative instances. The four outcomes can be formulated in a 2×2 contingency table or confusion matrix, as follows:

TABLE II. Confusion Matrix

		Actual value		Total
		P	n	
Prediction Outcome	p'	True Positive(TP)	False Positive(FP)	P'
	n'	False Negative(FN)	True Negative(TN)	N'
Total		P	N	

The area under ROC curve ranges between 0 and 1—it measures the classification performance of using the selected metrics value to put classes into Error (flag alarm) or No-error (don't flag alarm) categories. The graph below shows three ROC curves representing excellent, good, and worthless tests plotted on the same graph. The accuracy of the test depends on how well the test separates the group being tested into those with and without the error in classes. Accuracy is measured by the area under the ROC curve. An area of 1 represents a perfect test; an area of .5 represents a worthless test. A rough guide for classifying the accuracy of a diagnostic test is the traditional academic point system:

The general rule to evaluate the classification performance is to find the area under the curve (AUC) [32]:

- AUC=0.5 means no good classification;
- 0.5<AUC<0.6 means poor classification;
- 0.6≤AUC<0.7 means fair classification;
- 0.7≤AUC<0.8 means acceptable classification;
- 0.8≤AUC<0.9 means excellent classification;
- AUC≥0.9 means outstanding classification.

For calculating AUC, we have used IBM SPSS statistics Version 19.

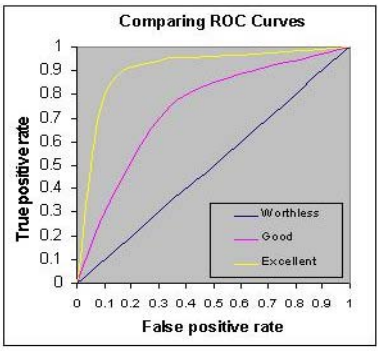

Fig1: Compairing ROC curve for calculating AUC[32]

The selected metrics values should have a classification performance falls at least within the acceptable range. Therefore, the metrics that have AUC within the acceptable (or higher) range will be considered valid; otherwise, we conclude that we could not select that metrics as predictor. The ROC analysis is very effective for data with skewed distribution and unequal classification error costs [32] and is suitable for analyzing our data because our data is not normally distributed and somewhat skewed. Using ROC curve, we will evaluate that selected OO-metrics using our new method are best predictor of software fault or not.

IV. PROPOSED ALGORITHM

Step 1: Generate Association rules with the minimum support (30%) and minimum confidence (40%) using "Apriori Algorithm".
Step2: From generated rule select out all those rules having confidence >=0.95
Step 2: For all selected rules. Find the support, confidence and correlation of each rule and also
Find F-measure=α. support + β. confidence + γ. correlation + k by applying the multivariate least square regression Analysis.
Step4: Rank the association rules according to the F-measure value, higher value better the association rule.
Step5: Match the best top "n" association rule generated by using support, confidence and correlation (F-measure) with the association rule generated by the support-confidence.
Step6: Use a heuristic to "Maximize the match number of association rule". It gives the best association rules.
Step7: The best association rules are analyzed for finding the best predictor for predicting the fault prone module in other version of same project .Those OO-metrics which are present in the rule at antecedent part in most of the rules can be used as best predictor in software fault.
Step 8: Evaluation of the selected metrics are done using ROC curve has done.

By integrating the correlation with support confidence, it generates the best and interesting rules. F-measure value of each rule is sorted in descending order in terms of higher value to lower value. Compare the unsorted F-measure of rule with the sorted F-measure. Sorted F-measure gives the optimal association rules.

A. Heuristic Employed

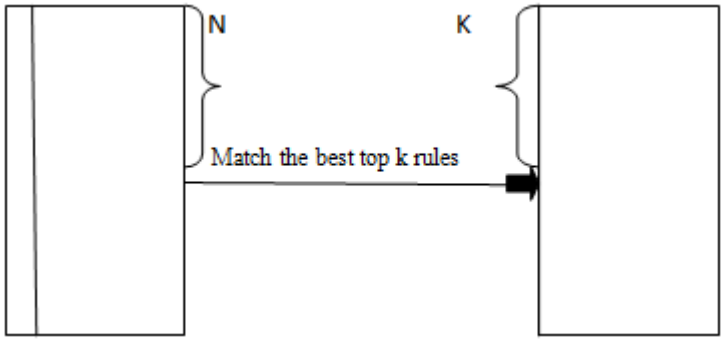

Fig. 2. Match top "N" association rule with top "K" Association rules in order to maximize the match number

In Fig. 2, Left side shows: Association Rules with support and confidence (Apriori order, Old Sequence). Right side shows: Association Rules with support, confidence and correlation (F-measure order, New Sequence). To find the appropriate value of α, β and γ (the coefficients), the least square regression analysis heuristic is as follows: "Generate a new sequence of rules based on the F-measure values with the aim of maximizing the total number of matches out of top K rules between the old and new sequence, subject to the constraints mentioned before". The corresponding values of α, β and γ are the ones used in our further steps, i.e. the "appropriate values".

V. RESULT AND DISCUSSION

The objective of this paper is to discover the interesting association rules by considering all the three parameters:- support, confidence and correlation. Thus, all three of them contribute to the result F-measure. We attach three coefficients α, β and γ to these three parameters, which as a weight for the individual parameters contribution to the value of F-measure. Thus, it prunes out the weak association rule which tends to creep into the top n association rules Apriori algorithm is used to generate association rules on the basis of support-confidence framework. Support = 30% and confidence = 40% is used, thus all the association rules which qualify these two threshold, will be generated.

We are taking Eclipse2.0 dataset for finding the association rule between OO-metrics for getting the best predictor of fault using this rule. First of all we convert the dataset column value from numerical value to nominal value by K-means clustering method and provide profile range value for each metric through four clusters. Table 3 shows the all four clusters ranges of all the 17 metrics.

TABLE III. Clusters of the Metrics for Defining Range in Form of Min and Max of Each Cluster

OO-metric	Nominal			Low			Mid			High		
	C1_min	C1_max	CC1	C2_Min	C2_max	CC2	C3_min	C3_max	CC3	C4_min	C4_max	CC4
NOS	0	98	50	99	215	158	216	383	285.5	384	3582	441.5
UWCS	0	30	16	31	65	49	66	121	88	123	1646	133
CC	0	50	26	51	111	82	112	200	149.5	201	1839	228
RFC	0	43	22	44	94	70	95	158	127	160	596	194
NLOC	0	116	58.5	117	255	185.5	256	463	338.5	465	5200	517.5
EXT	0	32	17	33	69	52	70	116	93.50	118	325	142
MPC	0	32	17	33	69	52	70	116	93.50	118	325	142
LMC	0	10	6	11	22	17.50	23	36	30.50	37	194	45.50
TCC	0	45	23.5	46	100	74	101	186	134	188	1222	204.5
PACK	0	15	8.50	16	33	25.50	34	57	46	58	146	70
NOM	0	21	11.5	22	47	35.50	48	78	63.50	79	596	95.50
LCOM2	0	96	48	97	226	151.5	227	550	276.5	561	41126	423.5
INST	0	18	10	19	42	31	43	84	56	86	1050	85
CBO	0	9	5.50	10	21	16.50	22	38	29.50	40	76	44
MAXCC	0	10	6	11	23	18	24	40	32.50	41	229	49.50
FOUT	0	6	4	7	14	11.50	15	22	19.50	24	69	28.50
AVCC	-	-	-	.00	4.60	2.67	4.62	11.64	6.5	12.5	26.17	17.40

After getting the min and max value of each cluster for all the 17 metrics. we have done some transformation in input data in the form that for each metric we have assumed four cluster on the condition that each cluster must contain more than two value(except AVCC all metrics have 4 clusters).Four clusters name are Nominal, Low, Mid and High for every metrics. We specify range for each cluster based on the min and max value respective to that cluster in all the taken metrics.

After Preprocessing done on the dataset we make the dataset compatible to use it for association Mining using WEKA tool. We have chosen only those rules whose confidence factor is greater and equal to confidence 0.95. We got total 52 rules having confidence greater and equal to 0.95.After that we calculate the support, confidence and correlation of each selected rules. After that find the F-measure of each selected rules. After matching the rules in both ways first sorting by confidence and second by F-measure. And then select the top 20 rules for analysis. The selected rules are shown in Table 4. We study all the 20 rules.

TABLE IV. Shows the Top Matched 20 Association Rules in Confidence Order (Apriori Order) for Eclipse2.0 from WEKA 3.6

1. nom UWCS=nominal 2153 ==> nomINST=nominal 2140 <conf:(0.99)>
2. nom UWCS=nominal nomLMC=nominal 2128 ==> nomINST=nominal 2115 <conf:(0.99)>
3. nom NOM=nominal 2186 ==> nomLMC=nominal 2170 <conf:(0.99)>
4. nom NOM=nominal nomINST=nominal 2120 ==> nomLMC=nominal 2104 <conf:(0.99)>
5. nom UWCS=nominal 2153 ==> nomLMC=nominal 2128 <conf:(0.99)>
6. nom UWCS=nominal nomINST=nominal 2140 ==> nomLMC=nominal 2115 <conf:(0.99)>
7. nomMAXCC=nominal 2199 ==> nomAVCC=low 2165 <conf:(0.98)>
8. nom LCOM2=nominal 2164 ==> nomLMC=nominal 2129 <conf:(0.98)>
9. nom UWCS=nominal 2153 ==> nomLMC=nominal nomINST=nominal 2115 <conf:(0.98)>
10. nomLMC=nominal nomCBO=nominal 2175 ==> nomFOUT=nominal 2120 <conf:(0.97)>
11. nomCBO=nominal nomAVCC=low 2198 ==> nomFOUT=nominal 2135 <conf:(0.97)>
12. nom UWCS=nominal 2153 ==> nom NOM=nominal 2091 <conf:(0.97)> lift:(1.16)
13. nom NOM=nominal 2186 ==> nomINST=nominal 2120 <conf:(0.97)> lift:(1.03)
14. nomLMC=nominal nom NOM=nominal 2170 ==> nomINST=nominal 2104 <conf:(0.97)>
15. nomINST=nominal nomCBO=nominal 2231 ==> nomFOUT=nominal 2163 <conf:(0.97)>
16. nomCBO=nominal nom MULTINOMIAL=nominal 2217 ==> nomFOUT=nominal 2149 <conf:(0.97)>
17. nomCBO=nominal 2342 ==> nomFOUT=nominal 2270 <conf:(0.97)>
18. nom LCOM2=nominal 2164 ==> nomINST=nominal 2095 <conf:(0.97)>
19. nomLMC=nominal nomCBO=nominal 2175 ==> nomINST=nominal 2103 <conf:(0.97)>
20. nomLMC=nominal nomFOUT=nominal 2225 ==> nomINST=nominal 2144 <conf:(0.96)>

From the above table we found that all the top 20 rules revolve around only 8 metrics among 17 metrics: UWCS, INST, LMC, NOM, AVCC, LCOM2, CBO and FOUT. 9 metrics are deleted from analysis. According to our objective we want to find all those metrics as predictors using association mining. But in our problem we are finding frequent software metrics in every class. From observation we found that if any metric is found in antecedent part of the relation and other metrics comes in consequent part. Then it means there is no need to use both of the metrics in the relation for developing fault prediction model. Because they can share same type of information in prediction of fault. After giving more focus on the generated top 20 rules, we found that:

TABLE V. Metrics Selected as Predictor for Prediction of Fault

Rule	Include	Exclude
1&2	UWCS(1) , LMC(1)	INST(1)
3&4	NOM(1),INST(1)	LMC(1)
5&6	UWCS(2),INST(2)	LMC(2)
7	MAXCC(1)	AVCC(1)
8	LCOM2(1)	LMC(3)
9	UWCS(3)	LMC(4),INST(2)
10	LMC(2), CBO(1)	FOUT(1)
11	CBO(2), AVCC(1)	FOUT(2)
12	UWCS(4)	NOM(1)
13	NOM(2)	INST(3)
14	LMC(3), NOM(3)	INST(4)
15	INST(3), CBO(3)	FOUT(3)
16 &17	CBO(4)	FOUT(4)
18	LCOM2(2)	INST(5)
19	LMC(4), CBO(5)	INST(6)
20	LMC(5), FOUT(1)	INST(7)

In Table 5 the number beside the metric shows the number of times metric comes in antecedent part in "Include" column and in Consequent part in "Exclude' column. If the number of metrics in "Exclude column is greater than or equal to "Include" column then that metric is finally excluded otherwise included for fault prediction. We finally excluded"INST","AVCC" and "FOUT". So, we got only 6 metrics for prediction. For knowing the classification capability of the metrics, we draw the ROC curve; we got the fair result for nominal category. But acceptable and excellent results for all the three categories: low, mid and high for all three version of Eclipse.

TABLE VI. Support Confidence Correlation Value and F-measure Sorted Value of Top 20 Association Rules Among n Rules (Eclipse2.0)

Rules	Confidence	correlation	Support	F-measure
1	0.97	1.16	0.3107	0.9764
2	0.98	1.12	0.3143	0.9583
3	0.99	1.09	0.3127	0.9470
4	0.99	1.09	0.3225	0.9458
5	0.99	1.08	0.3143	0.9417
6	0.99	1.08	0.3162	0.9414
7	0.98	1.08	0.3164	0.9371
8	0.99	1.06	0.3143	0.9316
9	0.99	1.06	0.3180	0.9311
10	0.97	1.06	0.3151	0.9227
11	0.97	1.06	0.3173	0.9224
12	0.97	1.06	0.3214	0.9219
13	0.98	1.05	0.3217	0.9212
14	0.97	1.06	0.3295	0.9210
15	0.97	1.06	0.3373	0.9201
16	0.97	1.03	0.3113	0.9080
17	0.97	1.03	0.3125	0.9079
18	0.97	1.03	0.3127	0.9079
19	0.97	1.03	0.3151	0.9076
20	0.96	1.03	0.3186	0.9027

From table 6, it clearly seen that the series of rules are in the order of F-measure value.Higher the F-measure value, better the association rule will be. The analysis is done for each selected rules on Eclipse2.0 dataset and the analysis is recorded in table 5 in the form of include and exclude column. The selected metrics are used as predictor. The discrimination ability of each selected metric of all three version of eclipse is recorded in the table 8 by AUC value. The ROC curve are also drawn for all three version of Eclipse of all error types(i. e. nominal, low, mid and high) in fig. 3 to fig. 14.The proposed algorithm efficiently gives the actual best association rule of the dataset, depending upon F-measure value. The same process was employed for two next versions of Eclipse and we found that the selected metrics by our algorithm are really giving promising results.

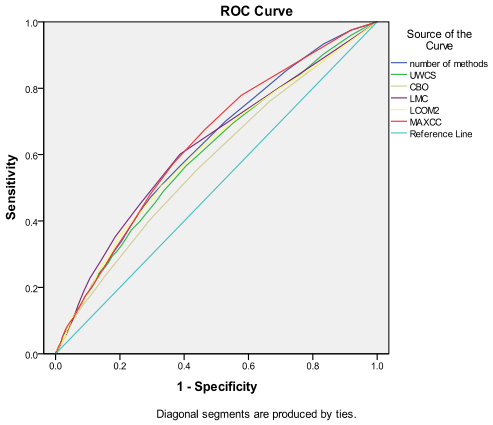

Fig3: ROC curve for Nominal type error in Eclipse2.0

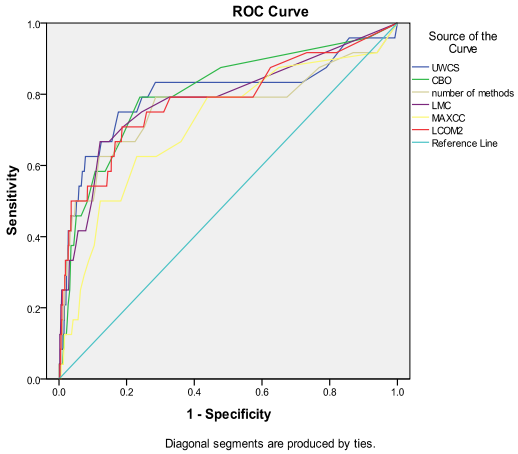

Fig4:ROC curve for Low type error in Eclipse2.0

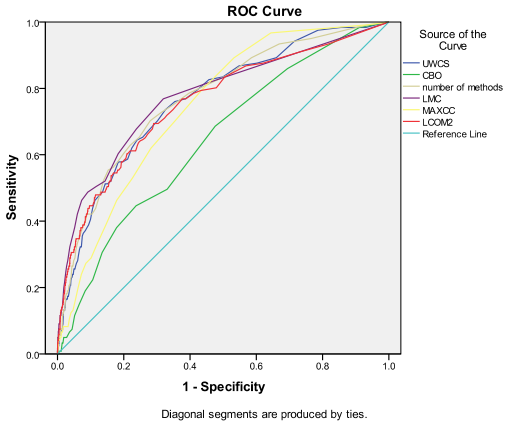

Fig5:ROC curve for Mid type error in Eclipse 2.0

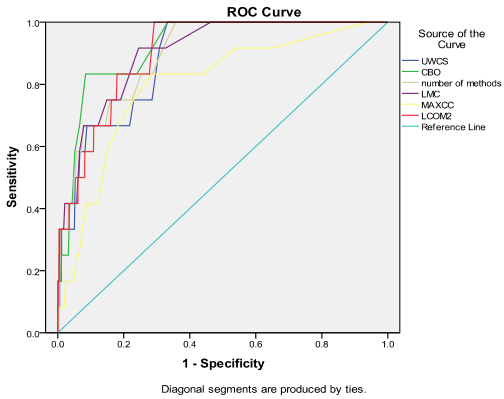

Fig6: ROC curve for High type error in Eclipse 2.0

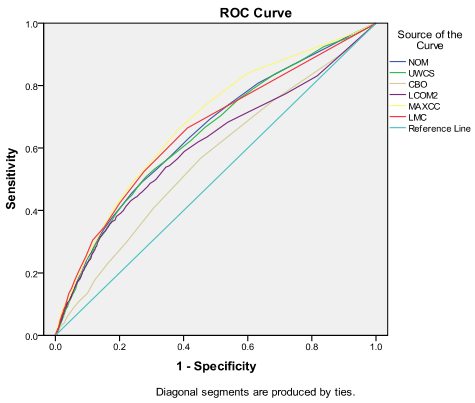

Fig7:ROC curve for Nominal type error in Eclipse2.1

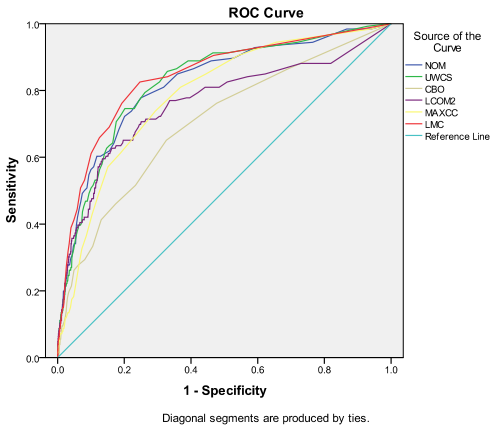

Fig8:ROC curve for Low type error in Eclipse2.1

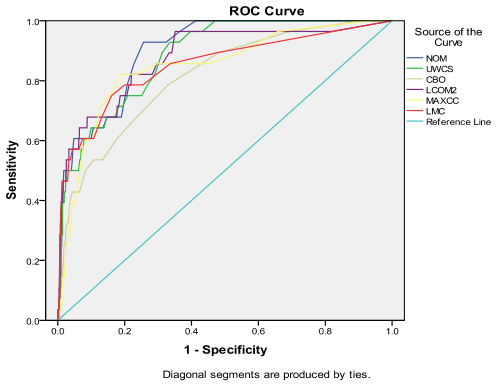

Fig9:ROC curve for Mid type error in Eclipse2.1

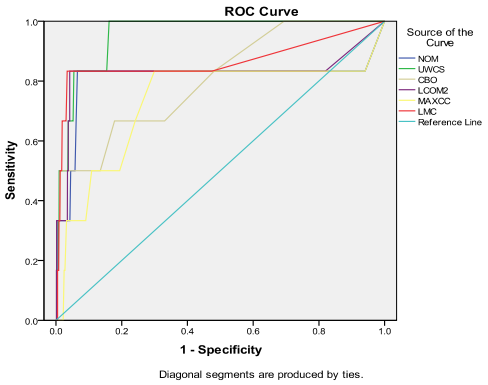

Fig10:ROC curve for High type error in Eclipse2.1

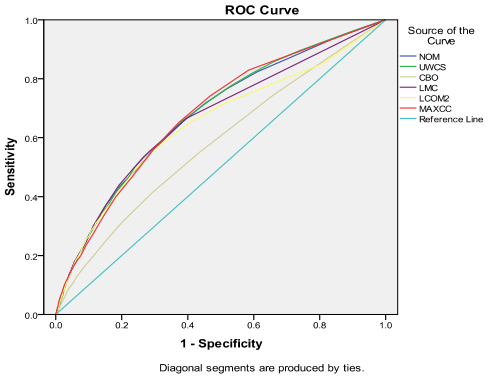

Fig11:ROC curve for Nominal type error in Eclipse3.0

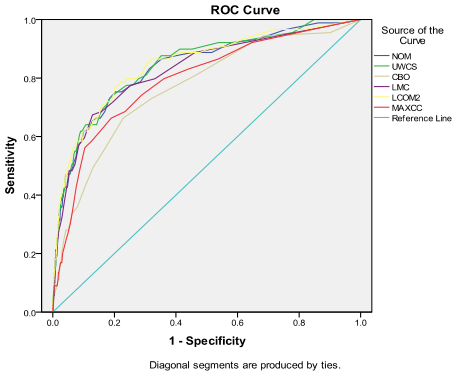

Fig12:ROC curve for Mid type error in Eclipse3.0

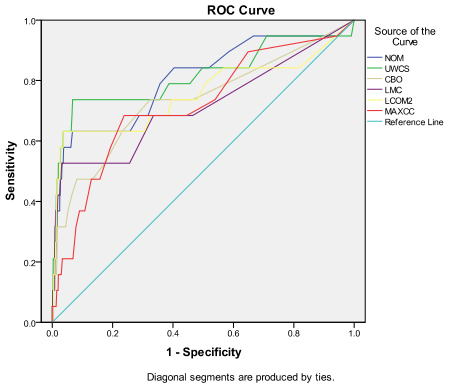

Fig13:ROC curve for Mid type error in Eclipse3.0

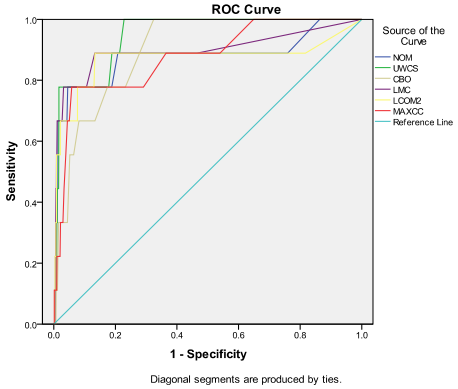

Fig14:ROC curve for High type error in Eclipse3.0

TABLE VIII. AUC result for all three version of Eclipse for all error types

Version	Selected Metrics	Nominal AUC	Low AUC	Mid AUC	High AUC
Eclipse2.0	UWCS	.603	.764	.799	.881
	CBO	.578	.645	.805	.920
	NOM	.623	.771	.762	.895
	LMC	.622	.775	.786	.901
	MAXCC	.635	.745	.715	.798
	LCOM2	.641	.756	.783	.900
Eclipse2.1	UWCS	.649	.821	.902	.809
	CBO	.644	.825	.882	.955
	NOM	.563	.704	.813	.806
	LMC	.608	.761	.878	.829
	MAXCC	.679	.785	.848	.732
	LCOM2	.655	.841	.851	.864
Eclipse3.0	UWCS	.678	.838	.804	.876
	CBO	.681	.845	.817	.946
	NOM	.579	.767	.734	.898
	LMC	.665	.827	.716	.897
	MAXCC	.646	.846	.756	.870
	LCOM2	.680	.792	.718	.876

VI. CONCLUSION AND FUTURE SCOPE

In this paper, we conducted association rule mining and statistical analysis to see whether we could classify the Software module into Multinomial categorization. In Multinomial categorization, we investigate whether metrics selected through association rule mining could classify the modules into one of the four categories (nominal-impact error, low-impact error, medium-impact error and high impact error).We belief that if we are able to classify the modules using selected metrics, we can use these metrics in practice to classify modules in OO-design to different error-risk categories. This paper also presented the heuristics to rank the association rules by considering three parameters: support, confidence and correlation. This proposed method will generate a best association rules as it can weed out the relatively weaker association rules and the actual best association rules will be easily noticed and identified in the original dataset. Therefore, for those databases which contain large numbers of transactions, our algorithm can efficiently give the actual best association rule of the database. We also found that the selected predictor from this method also give promising result in predicting the multinomial fault of Eclipse software, which is shown by the ROC curve. So, it is very useful for the market strategies, such as in the supermarket example the sales manager can recommend the relevant related products to the customers. Any field in which association rules are required will benefit from this methodology. Like: - Business Solutions, Industrial Solutions, and in any other case where we want to make a better choice.

REFERENCES

[1] N.J. Pizzi, A fuzzy classifier approach to estimating software quality, Inform. Sci. 241 (2013) 1–11.

[2] R. hua Chang, X. Mu, L. Zhang, Software defect prediction using non-negative matrix factorization, J. Softw. 6 (11) (2011) 2114–2120.

[3] A. Sharma, N. Tivari, (Aug-2012), A Survey of Association Rule Mining Using Genetic Algorithm, International Journal of Computer Applications and Information Technology, Vol. 1,Issue-2, ISSN: 2278-7720, pp.1-8.

[4] R. Agrawal, T. Imielinski, A. N. Swami, (May-1993), Mining association rules between sets of items in large databases. In Proceeding of the ACM SIGMOD International Conference on Management of Data, Washington D.C, pp. 207-216.

[5] M. Renuka Devi, A. Babysarojini, (Aug-2012), Applications of Association Rule Mining in Different Databases, Journal of Global Research in Computer Science, Vol.3, pp. 30-34.

[6] L. Fang, Q. Qizhi, (2012), The Studying on the Application of Data Mining based on Association Rules, International Conference on Communication Systems and Network Technologies, Rajkot, India pp.477-480.

[7] P. N. Tan, V. Kumar, and J. Srivastava (2002), Selecting the right interestingness measure for association patterns,Information Systems, Vol.29, pp.293-313.

[8] A Silberschatz, A Tuzhilin(1994),What makes patterns interesting in knowledge discovery systems,Knowledge and Data Engineering, IEEE Transactions on 8 (6), 970-974.

[9] Shweta, K. Garg, (June-2013), Mining Efficient Association Rules Through Apriori Algorithm Using Attributes and Comparative Analysis of Various Association Rule Algorithms, International Journal of Advanced Research in Computer Science and Software Engineering, Vol.3, Issue-6, pp. 306-312.

[10] R. Santhi, K. Vanitha, (April-2012), An Effective Association Rule Mining in Large Database, International Journal of Computer Application and Engineering Technology, Vol.1(2), ISSN: 2277-7962,pp.72-76.
[11] M. Dhanda, S. Guglani, G. Gupta, (Sep-2011), Mining Efficient Association Rules Through Apriori Algorithm Using Attributes, International Journal of Computer Science and Technologies, Vol.2 ,Issue 3, pp.342-344.
[12] J. Singh, H. Ram, J. Sodhi, (Jan-2013), Improving Efficiency of Apriori Algorithm Using Transaction Reduction, International Journal of Scientific and Research Publications, Vol.3 (1),ISSN: 2250-3153, pp.1-4.
[13] Jun-Sese, S. Morishita, (Aug-2002), Answering the Most Correlated N Association Rules Efficiently. In Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery, Helsinki, Finland, pp.410-422.
[14] H.S. Anand , S.S. Vinodchandra , (Mar-2013), Applying Correlation Threshold on Apriori Algorithm, IEEE International Conference on Emerging trends in Computing, Communication and Nanotechnology, Tirunelveli, India, pp. 432-435.
[15] S. Ghosh, S. Biswas, D. Sarkan, P.P. Sarkar, (Oct-2010), MiningFrequent Itemsets Using Genetic Algorithm, International Journal of Artificial Intelligence and Applications, Vol.1 , No.4, pp. 133-143.
[16] B. Rani, S. Aggarwal, (Dec-2013), Optimization of Association Rule Mining Techniques using Ant Colony Optimization, International Journal of Current Engineering and Technology, Vol.3,No.-5, pp.1804-1808.
[17] P. Mandrai, R. Barskar, (July-2013), A Novel Algorithm for Optimization of Association Rule with Karnagh Map and Genetic Algorithm, 4th International Conference on Computing, Communications and Network Technologies, Tiruchengode, India, pp.1-7.
[18] Kannika Nirai Vaani M, E. Ramaraj, (Feb-2013), An Integrated Approach to derive effective rules from Association Rule Mining using Genetic Algorithm, In Proceedings of the International Conference on Pattern Recognition, Informatics and Mobile Engineering, Salem, pp. 90-95.
[19] Gabriela Czibula, Zsuzsanna Marian, Istvan Gergely Czibula,Software defect prediction using relational association rule mining,Information Sciences 264 (2014) 260–278
[20] Qinbao Song, Martin Shepperd, Michelle Cartwright, and Carolyn Mair,Software Defect Association Mining and Defect Correction Effort Prediction,IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 32, NO. 2, FEBRUARY 2006
[21] Ching-Pao Chang a, Chih-Ping Chu a, Yu-Fang Yeh b,Integrating in-process software defect prediction with association mining to discover defect pattern,Information and Software Technology 51 (2009) 375–384
[22] Eclipse bug data (for archived releases):http://www.st.cs.uni-sb.de/softevo/bug-data/eclipse(Accessed 20 November 2012)
[23] A. Schroter, T. Zimmermann, R. Premraj, and A. Zeller(2006)'If your bug database could talk...' ,in Proceedings of the 5th International Symposium on Empirical Software Engineering. Volume II: Short Papers and Posters, 2006, pp. 18-20.
[24] JHAWK metrics referencehttp://www.virtualmachinery.com/jhawkreferences.html(Accessed March 2013)
[25] Eclipse source code (for archived releases):http://archive.eclipse.org/eclipse/downloads/(Accessed 3 December 2012)
[26] Eclipse bug data (for archived releases):http://www.st.cs.uni-sb.de/softevo/bug-data/eclipse(Accessed20 November 2012)
[27] B. Ramasubbareddy, A.Govardhan, A. Ramamohanreddy, (Nov-2010), Mining Positive and Negative Association Rules,International Journal of Recent Trends in Engineering andTechnology, Vol.4, pp.151-155.
[28] E. Garcia, C. Romero, S. Ventra, T. Calders, (2007), Drawbacksand Solutions of applying association rule mining in learning management systems, In Proceedings of International Workshop on Applying Data Mining in e-learning, Crete, Greece, pp.-15-25.
[29] R. Agrawal, R. Srikant, (Sep-1994), Fast algorithms for mining association rules. In Proceedings of the 20th International Conference on Very Large Data Bases, Santiago, Chile, pp.487-499.
[30] Hosmer D and Lemeshow S.(2000)Applied Logistic Regression, Wiley-Interscience,2nd edn,,New York NY.
[31] Zweig M and Campbell G.(1993)Receiver-operating characteristic (ROC) plots: A fundamental evaluation tool in clinical medicine, Clinical Chemistry 1993; 39(4):561–577.
[32] Fawcett T. (2004)ROC graphs: Notes and practical considerations for researchers,Technical Report, HP Laboratories, Page Mill Road, Palo Alto, CA, 2004; 38.

AUTHORS

Mrs. Dipti Kumari Completed M.Tech. (CSE) from Birla Institute of Technology, Mesra, Ranchi, Jharkhand, India in the year 2010. Currently, she is pursuing PhD on Software Fault Prediction. Her Research area is Object-Oriented Metrics, Software Engineering, Software Fault Prediction, Programming Languages, Database Management System and Object-Oriented Software fault prediction.

Dr. Kumar Rajnish is an Assistant Professor in the Department of Information Technology at Birla Institute of Technology, Mesra, Ranchi, Jharkahnd, India. He received his PhD in Engineering from BIT Mesra, Ranchi, Jharkhand, India in the year of 2009. He received his MCA Degree from MMM Engineering College, Gorakhpur, State of Uttar Pradesh, India. He received his B.Sc Mathematics (Honours) from Ranchi College Ranchi, India in the year 1998. He has 23 Research Publications. His Research area is Object-Oriented Metrics, Object-Oriented Software Engineering, Software Quality Metrics, Programming Languages, and Database System.