

Mining of Defect using Apriori and Defect Correction Effort Prediction

J. Priyadarshin

Sree Sastha Institute of Engineering and Technology
Chembarambakkam, Chennai-602103

Abstract-Organizations face many problems that impede rapid development of software systems critical to their operations and growth. The challenge in any software product development lies in minimizing the number of defects. Occurrence of defects is the greatest contributor to significant increases in product costs due to correction and rework time. Most defects are caused by process failures rather than human failures. Identifying and correcting process defects will prevent many product defects from re-occurring. Much current software defect prediction work focuses on number of defects remaining in a software system. This project presents association rule mining based methods to predict defect associations and defect correction efforts. This is to help developers detect software defects and assist project managers in allocating testing resources more effectively. For defect association prediction, the accuracy is very high and false-negative rate is very slow. Likewise for the defect correction effort prediction, the accuracy for both defect isolation effort prediction and defect correction effort prediction are also high. The impact of support and confidence levels on prediction accuracy, false-negative rate, false-positive rate and the number of rules is evaluated and found that higher support and confidence levels result in higher prediction accuracy and sufficient number of rules is a pre-condition for high prediction accuracy.

I. INTRODUCTION

The success of a software system depends not only on cost and schedule, but also on quality. Current defect prediction work focuses on estimating the number of defects remaining in software systems with code metrics, inspection data, and process-quality data by statistical approaches, capture-recapture (CR) models and detection profile methods (DPM). The prediction result, which is the number of defects remaining in a software system, can be used as an important measure for the software developer, and can be used to control the software process.

The prediction of defect (type) associations and corresponding defect correction effort is to be done. This is answering the following questions:

1. For the given defect(s), what other defect(s) may co-occur?
2. In order to correct the defect(s), how much effort will be consumed?

Defect type data to predict software defect associations that are the relations among different defect types such as: If defects a and b occur, then defect c also will occur. This is formally written as $a \wedge b \Rightarrow c$. The defect associations can be used for three purposes:

First, find as many related defects as possible to the detected defect(s) and, consequently, make more-effective corrections to the software. For example, consider the situation where there are classes of defect a, b, and c and suppose the rule $a \wedge b \Rightarrow c$ has been obtained from a historical data set, and the defects of class a and b have been detected occurring together, but no defect of class c has yet been discovered. The rule indicates that a defect of class c is likely to have occurred. If the result is positive, the search can be continued so, if rule $a \wedge b \wedge c \Rightarrow d$ holds as well, then the same thing for defect d is done.

Second, help evaluate reviewers' results during an inspection. For example, if rule $a \wedge b \Rightarrow c$ holds but a reviewer has only found defects a and b, it is possible that the missed defect c. Thus, a re-inspection should be done for completeness.

Third, 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. At the same time, for each of the associated defects, effort required to isolate and correct is predicted.

Association rule mining aims to discover the patterns of co-occurrences of the attributes in a database. An association rule is an expression $A \Rightarrow C$, where A (Antecedent) and C (Consequent) are sets of items. The meaning of such rules is quite intuitive: Given a database D of transactions, where each transaction $T \in D$ is a set of items, $A \Rightarrow C$ expresses that whenever a transaction T contains A, then T also contains C with a specified confidence. The rule confidence is defined as the percentage of transactions containing C in addition to A with regard to the overall number of transactions containing A.

II. RESEARCH METHOD

A. GENERAL METHOD

The objective of the study is to discover software defect associations from historical software engineering data sets, and help determine whether or not a defect(s) is accompanied by other defect(s).

The data set are processed and obtain three data sets: the defect data set, the defect isolation effort data set, and the defect correction effort data set. Then, for each of these data sets, randomly extract five pairs of training and test data sets

B. DATA SOURCE AND DATA EXTRACTION

The Data in the database consists of tables that provide data on the projects’ software characteristics, changes and errors during all phases of development, and effort and computer resources used. In the Data the defects are divided into six types, Table 1 contains the details.

**TABLE 1
 Defect Type Description**

Defect type	Description
Computational defect	Cause a computation to erroneously evaluate a variable’s value. These defects could be equations that are incorrect not because of the incorrect use of a data structure within the statement but by miscalculation.
Data value defect	A result of the incorrect use of data structure .Examples of this defects errors are the use of incorrect subscripts for an array, the use of the wrong variable in an equation , the use of the wrong unit of measurement, or the inclusion of an incorrect declaration of a variable.
Internal interface defect	That were associated with internal structured of a module
External interface defect	That were associated with structures existing outside the module’s local environment but
Initialization defect	Results from an incorrectly initialized variable, failure to reinitialize a variable, or because a necessary initialization is missing, failure to initialize or reinitialize a data structure properly upon a module entry/exit .
Logical/Control structure defect	Cause an “incorrect path” in a module to be taken. Such a control defect might be a conditional statement causing control to be passed to an incorrect path.

In addition, the effort used to correct defects falls into four categories: One Hour or Less, One Hour to One Day, One Day to Three Days, and More Than Three Days. For the purpose of defect association prediction, use SQL to extract defect data from different tables of the data and obtain the basic defect data set.

The defect data is very simple, and consists of defect types and the corresponding dates on which the need for change was determined. Then, follow the sliding window approach, that is, two subsequent

defects a and b are part of one transaction if they are at most one day apart and belong to the same project, to infer the transactions needed for the association rule mining Moreover,

placing defects into one transaction according to the selected sliding window just means

They co-occur during the given time window, it does not imply they must be dependent. However, if the placement of defects is coincidental they tend not to form association rules.

For the purpose of defect correction effort prediction, use SQL to extract defect data and the corresponding isolation and correction effort data from data and obtain two further data sets: the defect isolation effort data set and the defect correction effort data set. Both consist of five attributes (table 2).For both the defect association and defect correction effort predictions, the data set D is randomly split into five mutually exclusive subsets D1, D2, . . . and D5 of equal size, and $U_i = \bigcup_{D_j \in \{D_1, D_2, \dots, D_5\}} D_j$ ($i=1,2,3,4,5$), and D_t as training set s and test sets are used, respectively.

**TABLE 2
 Defect Effort Data Description**

Attribute	Description
Defect type	Type of defect as in table 1
Effort	Used to isolate or correct the defect
Typo	Flag that indicates the typographical error
Code left out	Indicates the omission error that is forgetting some entity within a module
Bad Code Added	Flag that indicates the commission error that is result of incorrect executable statement

C. ANALYSIS APPROACH

The five-fold cross-validation method as the overall analysis approach is used. That is, for each D of the defect data set, the defect isolation effort data set, and the defect correction effort data set, the inducer is trained and tested a total of five times. Each time $t \in \{1,2,\dots,5\}$, it is trained on $D \ominus D_t$ and tested on D_t .

The association rule mining method, to learn rules from the training data sets is used. For defect association prediction, the rule learning is straightforward, while for defect correction effort prediction; it is more complicated because the consequent of a rule has to be defect correction effort. Considering the target of association rule mining is not predetermined and classification rule mining has only one predetermined target, the class, integrate these two techniques to learn defect correction effort prediction rules by focusing on a special subset of association rules whose consequents are restricted to the special attribute, the effort. Once the rules, are obtained rank them, and use them to predict the defect associations and defect correction effort with the corresponding test data sets. The predictions of defect associations and defect correction effort are both on length first strategy.

D. RULE DISCOVERY AND DEFECT/EFFORT PREDICTION

In this section, first introduce the basic concepts of association rule mining. Then, the rule-ranking strategy used for the purpose of defect association and defect correction effort predictions is presented. After that, the methods of defect association prediction and defect correction effort prediction are based on the association rule mining method is given.

Association Rule Discovery

Association rule mining searches for interesting relationships, e.g., frequent patterns, associations, correlations, or potential causal structures, among sets of objects in databases or other information repositories. The approach is data rather than hypothesis driven. The interestingness of an association rule is measured by both support and confidence, which respectively reflect the usefulness and certainty of the rule. It must be stressed that even rules that discover with high levels of support and high confidence do not necessarily imply causality. However, such rules would obviously stimulate further research through the postulation of models that can be empirically evaluated.

Let $I = \{I_1, I_2, \dots, I_m\}$ be a set of attribute values, called items. A set $A \subseteq I$ is called an item set. Let a database D be a multiset of I . Each $T \in D$ is called a transaction. An association rule is an expression $A \Rightarrow C$, where $A \subset I, C \subset I$, and $A \cap C = \Phi$. A is referred as the antecedent of the rule, and C as the consequent of the rule. The rule $A \Rightarrow C$, has support $\text{Supp}(A \Rightarrow C)$ in D , where the support is defined as $\text{Supp}(A \Rightarrow C) = \text{Supp}(A \cup C)$. That means $\text{Supp}(A \Rightarrow C)$ percent of the transactions in D contain $A \cup C$, and $\text{Supp}(A) = |\{T \in D \mid A \subseteq T\}|/|D|$ is the support of A that is the fraction of transactions T supporting an item set A with respect to database D . The number of transactions required for an item set to satisfy minimum support is referred to as the minimum support count. A transaction $T \in D$ supports an item set $A \subset I$ if $A \subset T$ holds. The rule $A \Rightarrow C$ holds in D with confidence $\text{Conf}(A \Rightarrow C)$, where the confidence is defined as $\text{Conf}(A \Rightarrow C) = \text{Supp}(A \cup C)/\text{Supp}(A)$. That means $\text{Conf}(A \Rightarrow C)$ percent of the transactions in D that contain A also contain C . The confidence is a measure of the rule's strength or certainty while the support corresponds to statistical significance or usefulness. Association rule mining generates all association rules that have a support greater than minimum support $\text{min: Supp}(A \Rightarrow C)$, in the database, i.e., the rules are frequent. The rules must also have confidence greater than minimum confidence $\text{min: Conf}(A \Rightarrow C)$, i.e., the rules are strong. The process of association rule mining consists of these two steps:

Find all frequent item sets, where each ASC of these item sets must be at least as frequently supported as the minimum support count.

Generate strong rules from the discovered frequent item sets, where each $(A \Rightarrow C)$ of these rules must satisfy $\text{min: Supp}(A \Rightarrow C)$ and $\text{min: Conf}(A \Rightarrow C)$

Rule-Ranking Strategy

Before prediction, rank the discovered rules according to the length-first strategy. The length-first strategy was used for two reasons. First, for the defect association prediction, the length-first strategy enables us to find out as many defects as possible that coincide with known defect(s), thus preventing errors due to incomplete discovery of defect associations. Second, for the defect correction effort prediction, the length-first strategy enables us to obtain more-accurate rules, thus improving the effort prediction accuracy.

Specifically, the length-first rule-ranking strategy is as follows:

Rank rules according to their length. The longer the rules, the higher the priority.

If two rules have the same length, rank them according to their confidence values. The greater the confidence values, the higher the priority. The more-confident rules have more predictive power in terms of accuracy; thus, they should have higher priority.

If two rules have the same confidence values, rank them according to their support values. The higher the support values, the higher the priority. The rules with higher support value are more statistically significant, so they should have higher priority.

If two rules have the same support value, rank them in alphabetical order.

E. DEFECT ASSOCIATION PREDICTION

As the first step of defect association prediction, use the association rule mining method to find defect association rules from the defect data set. Although the discovery of defect association rules is straightforward, the implementation requires certain modifications to the data set.

In order to predict the defects that occurred independently, add a NULL to the transactions with only one defect. With this modification, the association rules mining method is able to find rules like Defect $\{a\} \Rightarrow$ Defect $\{\text{NULL}\}$, which means defect a occurred independently. The next step is to predict whether or not a k -defect will occur with others. The prediction begins by ranking the discovered rules according to the strategy.

The measures used to evaluate the defect association prediction method are prediction accuracy, false-negative rate, and false-positive rate, which are defined as follows:

Let G be a given original defect set, R be the real defect set, and P be the predicted defect set. The prediction accuracy of P is defined as:

$$\text{Accuracy}(P) = \frac{|(R \ominus G) \cap (P \ominus G)|}{|(R \ominus G)|}$$

where, if $G \equiv R \equiv P$ or $R \subset P$, $\text{Accuracy}(P) = 1$

The false-negative rate and false-positive rate to present the prediction error of defect associations is used. The false-negative rate FN denotes how many defects that are not predicted to occur along with the given set G but actually do, and is defined as follows:

$$\text{FN}(P) = \frac{|R \ominus R \cap P|}{|(R \ominus G)|}$$

where, if $G \equiv R \equiv P$ or $R \equiv P$, $\text{FN}(P) = 0$.

The false-positive rate FP denotes how many defects are predicted to occur along with the given set G but actually do not. It is defined as follows:

$$\text{Accuracy}(P) = \frac{|(R \ominus G) \cap (P \ominus G)|}{|(R \ominus G)|}$$

where, if $G \equiv R \equiv P$ or $P \subset R$, $\text{FP} = 0$

F. DEFECT CORRECTION EFFORT PREDICTION

Because the intent of defect correction effort prediction is predetermined and the association rule mining has no special goal, the association rules mining based discovery of defect correction effort prediction rules is not straightforward. Here, the defect correction effort includes both defect isolation effort and defect correction effort. the purpose of defect correction effort prediction, the constraint-based association rule mining method is used. Specifically, the procedure of association rule mining is as follows:

Compute the frequent item sets that occur together in the training data set at least as frequently as a predetermined min: Supp. The item sets mined must also contain the effort labels.

Generate association rules from the frequent item sets, where the consequent of the rules is the effort. In addition to the min: Supp threshold, these rules must also satisfy a minimum confidence min: Conf

Input: Rules- the output of Rule Ranking RR;

G – a given K-defect set

Output : Associations – the predicated defect set that can be co-occurred with G.

PreAssoc ← G; // the candidate associations

for each rule $r \in$ Rules do

if \exists PreAssoc \in Antecedent \textcircled{R} then

PreAssoc ← PreAssoc \cup Consequent(r); // combining end if

end for

Associations ← Longest(PreAssoc) \ominus G;

Once the rules are obtained, rank them according to the approach presented in Section 3.2. Then, for each element of

the given defect and its attributes, scan the rules one by one and identify the rule whose antecedent contains the element. After that, merge the antecedent of the corresponding rule with the element, generate an element set, and obtain the corresponding effort. For the element set, continue the scan until no more rules fit. At this point, obtain the most likely candidate effort and the corresponding similarity for the element. The process is repeated until all the elements of the given defect and its attributes have been checked. Finally, compare the similarities of all the candidates; the most similar candidate's effort is the defect correction effort for the given defect and its attributes.

CONCLUSIONS

This paper help developers detect software defects and project managers improve software control and allocate their testing resources effectively. From this, defect data and the corresponding defect isolation and correction effort data is extracted. For each of the three data sets, randomly generate five training data sets and a corresponding five test data sets. After that, apply the association rule mining method. The impact of support and confidence levels on prediction accuracy, false negative rate, false positive rate, and the number of rules are explored. The conclusions from a single data set study, is that our results suggest that association rule mining is an attractive technique to the software engineering community due to its relative simplicity, transparency, and seeming effectiveness in constructing prediction systems.

REFERENCES

- [1] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," Proc. ACM SIGMOD Conf. Management of Data, May 1993.
- [2] K. Ali, S. Manganaris, and R. Srikant, "Partial Classification Using Association Rules," Proc. Third Int'l Conf. Knowledge Discovery and Data Mining, pp. 115-118, 1997.
- [3] I.S. Bhandari, "Attribute Focusing: Machine-Assisted Knowledge Discovery Applied to Software Production Process Control," Proc. Workshop Knowledge Discovery in Databases, July 1993.
- [4] B. Liu, W. Hsu, and Y. Ma, "Integrating Classification and Association Rule Mining," Proc. Fourth Int'l Conf. Knowledge Discovery and Data Mining, pp. 80-86, 1998.
- [5] F. Padberg, T. Ragg, and R. Schoknecht, "Using Machine Learning for Estimating the Defect Content after an Inspection," IEEE Trans. Software Eng., vol. 30, no. 1, pp. 17-28, 2004.