Indian Journal of Communications Technology and Electronics (IJCTE)
Vol.2.No.2 2014 pp 69-74.
available at: www.goniv.com
Paper Received : 05-03-2014
Paper Accepted: 24-05-2014
Paper Reviewed by: 1. John Arhter 2. Hendry Goyal
Editor: Prof. P.Hendry

SOFTWARE FAULT PREDICTION MODELS USING AUGMENTED BAYESIAN NETWORK CLASSIFIERS

R.Nithya Rani[1], M.Mathina Kani[2]
Department of Computer and Communication Engineering[1], Department of Computer Science Engineering[2]
Sethu Institute of Technology, Affiliated to Anna University,
Kariapatti, Tamilnadu, 626106- India
nithireeta@gmail.com mathinakani@yahoo.co.in

ABSTRACT

Software metrics may be used in fault prediction models to improve software quality by predicting fault location. There are lots of different software metrics discovered and used for defect prediction in the literature. Instead of dealing with so many metrics, it would be practical and easy if we could determine the set of metrics that are most important and focus on them more to predict defectiveness. We use Bayesian networks to determine the probabilistic influential relationships among software metrics and defect proneness. Furthermore, the applicability of the Markov blanket principle for feature selection, which is a natural extension to Bayesian Network theory, is investigated. The results, both in terms of the Area Under Curve and recently introduced H-measure are rigorously tested using the statistical framework of Demsar.It is concluded that simple and comprehensible networks with less nodes can be constructed using Augmented Bayesian Network Classifiers. Furthermore, it is found that the aspects of comprehensibility and predictive performance need to be balanced out and also the development context is an item which should be taken into account during model selection.

Keywords: Software fault prediction, Bayesian networks, Data mining, Classification, Comprehensibility, Input query, Output query.

1. INTRODUCTION

Developing a defect free software system is very difficult and most of the time there are some unknown bugs or unforeseen deficiencies even in software projects where the principles of the software development methodologies were applied carefully. Due to some defective software modules, the maintenance phase of software projects could become really painful for the users and costly for the enterprises, that is why predicting the defective modules or files in a software system prior to project development is a crucial activity, since it leads to a decrease in the total cost of the project and an increase in overall project success rate.

Defect prediction will give one more chance to the development team to retest the modules or files for which the defectiveness probability is high. By spending more time on the defective modules and no time on the non-defective ones, the resources of the project would be utilized better and as a result, the maintenance phase of the project will be easier for both the customers and the project owners. Although there is diversity in the definition of software quality, it is widely accepted that a project with many defects lacks quality. Methodologies and techniques for predicting the testing effort, monitoring process costs and measuring results can help in increasing efficiency of software testing. Being able to measure the fault-proneness of software can be a key step towards steering the software testing and improving the effectiveness of the whole process. This motivates the use of software fault prediction models, which provide an upfront indication of whether code is likely to contain faults. Predictive modeling is the process by which a model is created or chosen to try to predict the probability of an outcome. The objective of a fault proneness model is to identify faulty classes and focus

testing effort on them. Furthermore, due to experience that specific subset of predictors from the suite which seem to show a significant relationship to fault proneness differs from system to system, therefore to predict software quality accuracy is always to be a hard work. We use Area under Curve (AUC) obtained from Receiver Operating Characteristics Curve (ROC) in order to evaluate the performance of the machine learning classification techniques. ROC analysis provides optimal cut off point that provides balance between number of classes predicted as faulty and number of classes predicted as non faulty.

Bayesian network is a graphical representation that shows the probabilistic causal or influential relationships among a set of variables that we are interested in. There are a couple of practical factors for using Bayesian networks. First, Bayesian networks are able to model probabilistic influence of a set of variables on another variable in the network. Given the probability of parents, the probability of their children can be calculated. Second, Bayesian networks can cope with the missing data problem. This aspect of Bayesian networks is very important for defect prediction since some metrics might be missing for some modules in defect prediction data sets. Looking at the defect prediction problem from the perspective that all or an effective subset of software or process metrics must be considered together besides static code measures. Bayesian network model is a very good candidate for taking into consideration several process or product metrics at the same time and measuring their effect.

In this paper, we build a Bayesian network among metrics and defectiveness, to measure which metrics are more important in terms of their effect on defectiveness and to explore the influential relationships among them. As a result of learning such a network we find the defectiveness probability of the whole software system, the order of metrics in terms of their contribution to accurate prediction of defectiveness, and the probabilistic influential relationships among metrics and defectiveness.

2. ARCHITECTURE OF DATA MINING

2.1 DATA BASE, DATAWARE HOUSE, WWW, OTHER INFO

This is one or a set of databases, data warehouses, spreadsheets, or other kinds of information repositories. Data cleaning and data integration techniques may be performed on the data.

2.2 DATABASE OR DATA WARE HOUSE SERVER

The database or data ware house server is responsible for fetching the relevant data, based on the user's data mining request.

2.3 KNOWLEDGE BASE

This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns. Such knowledge can include concept hierarchies, used to organize attributes or attribute values into different levels of abstraction. Knowledge such as user beliefs, which can be used to assess a pattern's interestingness based on its unexpectedness, may also be include.

2.4 DATA MINING ENGINE

This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association and correlation analysis, classification, prediction, cluster analysis, outlier analysis, and evolution analysis.

2.5 PATTERN EVALUATION MODEL

This component typically employs interestingness measures and interacts with the data mining modules so as to focus the search toward interesting patterns. It may use interestingness thresholds to filter out discovered patterns.

2.6 USER INTERFACE

This module communicates between users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task providing information to help focus the search, and performing exploratory data mining based on the intermediate data mining results.

3. RELATED WORKS

The prediction of fault in the software is being studied from various viewpoints in order to estimate the software reliability of the individual components and also the probability of failure each time a software component is being executed. The main purpose of these models is to predict the fault in the software at the cheaper rate.

Some researchers suggest the software fault prediction as both machine learning based and statistical based approaches. They investigate the previous studies from metrics, methods, datasets, performance evaluation metrics and experimental results in an easy and effective manner. The aspect of feature subset selection by using a generic backward input selection

wrapper is investigated. The results are subjected to rigorous statistical testing and indicate the ordinary least squares regression in combination with logarithmic transformation performs best. Another key finding is that by subset of highly predictive attributes such as project size, development and environmental related attributes, typically a significant increase in estimation accuracy can be obtained.

More recently, predicting defect prone software components are an economically important activity and so has received a good deal of attention. A general framework for software defect prediction that supports unbiased and comprehensive comparison between competing prediction components has been empirically illustrated. The defect predictor builds models according to the evaluated learning schemes and predicts software defects with new data according to the constructed model.

Researchers have a adopted algorithms to evaluate the performance metrics within a range of specific parameters such as speed of learning, over fitting avoidance and their accuracy. Besides these parameters we have included their benefits and limitations to unveil their hidden features and provide more details regarding their performance. We have found the Augmented Bayesian Network classifiers is the best as compared with other algorithms that can be used for the prediction of the fault in the software.

4. AUGMENTED BAYESIAN NETWORK CLASSIFIERS

A Bayesian Network is a directed acyclic graph (DAG), composed of E edges and V vertices which represent joint probability distribution of a set of variables. In this notation each vertex represents a variable and each edge represents the causal or associational influence of one variable to its successor in the network.

Let X=$\{X1, X2\ldots.Xn\}$ be n variables taking continuous or discrete values. The probability distribution of Xi is shown as P (Xi/axi) where axi's represent parents of Xi if any. When there are no parents of Xi, then it is a prior probability distribution and can be shown as P (Xi).

The joint probability distribution of X can be calculated using chain rule.

$$P(X) = P(X_1|X_2, X3\ldots Xn_) P(X_2, X3\ldots Xn_)$$

$$= P(X_1|X_2,\ldots,.X_n)P(X_2|X_3,\ldots,X_n)P(X_3,\ldots,X_n)$$

$$= P(X_1|X_2,\ldots,X_n)P(X_2|X_3,\ldots,X_n)\ldots P(X_{n-1}|X_n)P(X_n)$$

$$=\pi_{i=1} P(X_i|X_{i+1}\ldots Xn)$$

Given the parents of Xi, other variables are independent from Xi, so we can write the joint probability distribution as

$$P(X) = \pi_{i=1} P(X_i|ax_i)$$

On the other hand, Bayes' rule is used to calculate the posterior probability of Xi in a Bayesian network based on the evidence information present. We can calculate probabilities either towards from causes to effects P (Xi|E) or from effects to causes (P(E|Xi)). Calculating probability of effects from causes is called causal inference whereas calculating probability of causes from effects is called diagnostic inference. Figure 1 shows a sample Bayesian network and conditional probability tables. Assume that we would like to investigate the effect of using experienced developers (ED) and applying unit testing methodology (UT) on defectiveness (FP). Furthermore, each variable can take discrete values of on/off, that is developers are experienced or not, unit testing used or not used.

In Bayesian network structure learning, the search space is composed of all of the possible structures of directed acyclic graphs based on the given variables (nodes). Normally, it is very difficult to enumerate all of these possible directed acyclic graphs without a heuristic method. Because, when the number of nodes increases, the search space grows exponentially and it is almost impossible to search the whole space. Given a data set, the K2 algorithm proposed by cooper and Herskovits, heuristically searches for the most probable Bayesian network structure. Based on the ordering of the nodes, the algorithm network. If addition of a certain node Xj to the set of parents of node Xi does not increase the score of the Bayesian network,K2 stops looking for parents of node Xi further. Since the ordering of the nodes in the Bayesian network is known, the search space is much smaller compared to the entire space that needs to be searched without a heuristic method. Furthermore, a known ordering ensures that there will be no cycles in the Bayesian network, so there is no need to check for cycles too. K2 algorithm takes a set of n nodes; an initial ordering of the n nodes, the maximum number of parents of any node denoted by u and a database D of m cases as input and outputs a list of parent nodes for every node in the network. For every node in the network, the algorithm finds the set of parents with the highest probability taking into consideration the upper bound u for the maximum number of parents a node can have.

5. METHDOLOGY

In this paper we predict the fault in the software by constructing the Augmented Bayesian network classifiers and then finding out the Area under Curve and the recently introduced H-measure value.

The proposed system consists of following modules.

Preprocessing

Bayesian Network
Construction Markov
Blanket Feature Selection
Area Under Curve
H-measure

The datasets are being selected and then preprocessing step is done which is then undergone the Markov Blanket Feature Selection .The Bayesian network classifier is constructed to eliminate all the dependent variables. The K2 algorithm adopts a bottom-up search strategy, assuming equal prior probabilities for all possible for all possible network structures and considers all variables one by one, assuming some ordering in the variables. The Max-Min Hill-Climbing (MMHC) measure is asymptotically following a x2 distribution with appropriate degrees of freedom under the null hypothesis of conditional independence, which allows calculation of a p-value indicating the probability of falsely rejecting this null hypothesis. The Area under Curve and H-measure value is being evaluated to find out the probabilistic influential relationships among the variables and then finally the software fault is being predicted.

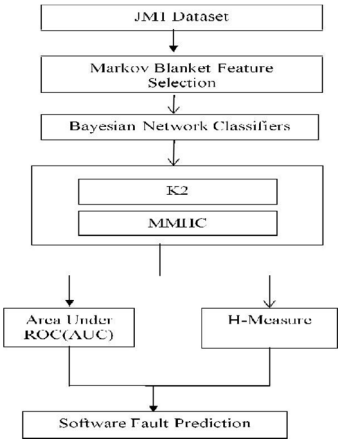

Fig 1. Dataflow diagram for software fault prediction model

A.PREPROCESSING

A first important step in each data mining exercise is pre-processing the data. Each observation in the datasets consists of a unique ID, several static code features and an error count. First, the data used to learn and validate the models are selected and thus the ID as well as attributes exhibiting zero variance are discarded. Observations with a total line count of zero are deemed logically incorrect and are removed. Each of the datasets is randomly partitioned into two disjoint sets, i.e. training and test set consisting of respectively 2/3 and 1/3 of the observations, using stratified sampling in order to preserve the class distribution.

B.BAYESIAN NETWORK CONSTRUCTION

Bayesian networks are directed acyclic graphs whose nodes represent random variables in the Bayesian sense: they may be observable quantities, latent variables, unknown parameters or hypotheses. Edges represent conditional dependencies; nodes which are not connected represent variables which are conditionally independent of each other. Each node is associated with a probability function that takes as input a particular set of values for the node's parent variables and gives the probability of the variable represented by the node. For example, if the parents are Boolean variables then the probability function could be represented by a table of 2m entries, one entry for each of the 2m possible combinations of its parents being true of false. Similar ideas may be applied to undirected and possibly cyclic graphs such are called Markov networks.

C.MARKOV BLANKET FEATURE SELECTION

The use of Markov blanket based feature selection approach provides a natural solution to this issue. The Markov blanket (MB) of a node y is the union of y's parents, y's children and the parents of y's children and is the minimal variable subset conditioned on which all other variables are independent of y. In other words, no other variables than those contained in the MB of y need to be observed to predict the value of y. For instance, the value of x can be ignored when predicting the value of y as it is the child of a parent of y and thus is no part of the MB of y. The HITON algorithm is used for the Markov blanket feature selection, which adopts the same test of conditional independence as the Max-Min Hill (MMHC) algorithm.

D.AREA UNDER CURVE

The single point metrics such as the Area under the ROC curve (AUC) were proposed. Let F1(S) be the probability density function of the scores for the classes and F1(S) the corresponding cumulative distribution function. The AUC can be regarded as a measure of aggregated classification performance as it in some

sense average performance over all possible thresholds. Moreover, the AUC has an interesting statistical interpretation in the sense that it is the probability that a randomly chosen positive instance will be ranked higher than a randomly chosen negative instance against the opposite type of misclassification.

E.H-MEASURE

If the additional knowledge of the likely values of c is available, Hand proposes using symmetric beta distribution. As no specific costs have been specified in the fault prediction literature, the H-measure will be calculated with these default values. H-measure on the other hand, has the benefit of explicitly, balancing the losses arising from classifying fault-prone as not fault-prone instances against the opposite type of misclassification. In order to keep the participants motivated, it is advised to release early and often thus the cost of missing defects is perhaps lower than the cost of delays due to unnecessary testing. As such, the robustness of the H-measure with respect to changes in the software development context is investigated. As no specific costs have been specified in the fault prediction literature, the H- measure will be calculated with these default values. However, depending on the context , it can be argued that misclassifying a faulty instance as non fault-prone is more serious.

6. IMPLEMENTATION

In order to achieve the objectives and benefits expected from the proposed system it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for education and training is more and more important. Once the implementation plan is decided, it is essential that the user of the system is made familiar and comfortable with the environment. A documentation providing the whole operations of the system is being developed. The first maintenance activity occurs because it is unreasonable to assume that software testing will uncover all latent errors in a large software system. During the use of any large program, errors will occur and be reported to the developer. The second activity contributes to a definition of maintenance occurs because of the rapid change that is encountered in every aspect of computing. The third activity that may be applied to a definition of maintenance occurs when a software package is successful. As the software is used, recommendations for new capabilities, modifications to existing functions and general enhancement are received from users. The fourth maintenance activity occurs when software is changed to improve future maintainability or reliability or to provide a better basis for future enhancements.

7. EXPERIMENTAL EVALUATION

The various induced models are being evaluated in terms of their classification performance and comprehensibility. A variety of performance measures has been used to gauge the strength of the classifiers. Augmented Bayesian Network Classifiers whose ROC curve lies above the ROC curve of the second classifier is superior and the point (0, 1) corresponds to perfect classification.

$$= \int \quad ( ) \ ( ) \ .$$

The AUC can be regarded as a measure of aggregated classification performance as it in some averages performance as it in some sense averages performance over all possible thresholds. H-measure has the benefit of explicitly, balancing the losses arising from classifying fault-prone as not fault-prone instances against the opposite type of misclassification. As such, the robustness of the H-measure with respect to changes in the software development context is investigated.

8. CONCLUSION AND FUTURE WORK

The broad goal of our research is to build a model to analysis the causal relation between evaluable metrics and software quality in software development. Then enhance software development efficiency by exploring the dependence between them. In this paper, we apply Augmented Bayesian Network Classifiers to solve the problem of classifying software modules as defect-free or non-defect-free. The Augmented Bayesian Network Classifier model provides a robust mechanism to include diverse sources of data into the analysis. The machine learning models predicted in this paper can help the testing community to focus the resources on the faulty parts of the software. The developers can also consider the software design and hence take necessary corrective actions. The fault prediction models can help the testers in planning and allocating resources in early phases of software development.

In future, besides only use one project's data is not convictive enough, dataset in different software project which focus on different functions tends to present weight of each

matrix.

REFERENCES

[1]Karel Dejaeger, Thomas Verbraken and Bart Baesens, "Toward Comprehensible Software Prediction Models Using Bayesian Network Classifiers", vol.39, 2013

[2]C.Catal, "Software Fault Prediction: A Literature Review and Current Trends, "Expert Systems with Applications, vol.38,pp. 4626-4636, 2011.

[3]K.Dejaeger, W.Verbeke, D.Martens, and B.Baesens, "Data Mining Techniques for software Effort Estimation :A Comparative Study,"IEEE Trans. Software Eng., vol. 38,no.2,pp. 375-397, Mar./Apr. 2011.

[4]P.Flach, J.Hernandez-Orallo and C.Ferri, "A Coherent interpretation of AUC as a Measure of Aggregated Classification Performance,"Proc. 28th Int'l Conf.Machine Learning, 2011.

[5]T.Menzies, Z.Milton, B.Turhan, B.Cukic, Y.Jiang and A.Bener, "Defect Prediction from Static Code Features: Current Results, Limitations, New Approaches," Automated Software Eng., pp.1-33, 2010.

[6]M.Baojun, K.Dejaeger, J.Vanthienen and B.Baesens, "Software Defect Prediction Based on Association Rule Classification, "Proc. Int'l Conf.Electronic-Buisness Intelligence, pp.396-402, 2010.

[7]Y.Jiang and B.Cukic, "Misclassification Cost-Sensitive Fault Prediction Models," Proc. Fifth Int'l Conf. Predictor Models in Software Eng., 2009.

[8]C.Catal and B.Diri, "A Systematic Review of Software Fault Prediction Studies," Expert Systems with Applications, vol.36, no.4, pp. 7346-7354, 2009.

[9]S.Ali and K.Smith, "On Learning Algorithm Selection for Classification," Applied Soft Computing, vol.6, no.2, pp.119-138, 2006.

[10]D.Chickering, C.Meek and D.Heckerman, "Large-Sample Learning of Bayesian Networks is NP-Hard," J.Machine Learning Research, vol.5, pp.1287-1330, 2004.