International Journal of Research in Engineering, Science and Management
Volume-1, Issue-5, May 2018
www.ijresm.com

Overview of Fault Prediction Using Data Mining Techniques and Software Metrics

T. Siva Kumar

Student, Department of MCA, Dr.BR Ambedkar University, Srikakulam, India

***Abstract*: Software fault prediction method used to improve the quality of software. Finding and fixing software fault is difficult, and needs significant effort. Data mining techniques are used to discover many hidden factors regarding software. The fault prediction is a very important task to minimize cost of the software project and also used in the analysis of software quality. Fault prediction systems predict faults by using software metrics and data mining techniques. Software fault prediction models are built based on a different set of metrics and faulty data of previous software release to build fault prediction models, which is called supervised learning approaches. There are some other methods like clustering, which could be used when there are no previous available data; these methods are known as unsupervised learning approaches. There are many software fault prediction techniques are available. This paper presents the survey on software fault prediction models.**

Key Words: fault prediction, software metrics, classification, clustering, association rule mining.

I. Introduction

A fault, by definition, is a structural imperfection in a software system that may lead to the system's eventually failing. In other words, it is a physical characteristic of the system of which the type and extent may be measured using the same ideas used to measure the properties of more traditional physical systems. People making errors in their tasks introduce faults into a system. These errors may be errors of commission or errors of omission.

A software fault refers to a defect in a system. An error is inconsistency between the observed performance of a system and its specified performance. A software failure occurs when the delivered product deviates from correct service and perform unexpected behaviour from user requirements. A software fault or error may not necessarily cause a software failure. Fault detection is recognizing that a problem has occurred, even if you don't know the reason. Faults may be detected by a variety of quantitative or qualitative approaches. This includes many of the multivariable, model-based approaches. Fault diagnosis is investigating one or more root causes of problems to the point where corrective action can be taken. This is also referred to as "fault isolation", especially when need to show the distinction from fault detection. A "fault" or "problem does not have to be the result of a complete failure of a software product. In a process plant, root causes of non-optimal operation might be hardware failures but problems might also be caused by poor choice of operating targets, poor feedstock quality or human error.

The following are the major classes of software faults:

1. Syntactic faults: interface faults and parameter faults called as syntactic faults.
2. Semantic faults: inconsistent behavior and incorrect results called as semantic faults.
3. Service faults: QoS faults, SLA (Service Level Agreement) related faults, and real-time violations are called service faults.
4. Communication / interaction faults: time out and service unavailability is called communication or interaction faults.
5. Exceptions: I/O related exceptions and security-related exceptions are called exception faults.

Data mining is one of the evolution techniques in information technology. It can be named as "knowledge mining from data". Before storing data into data warehouse or any type of databases, there is important to perform some data pre-processing steps. The pre-processing steps are data cleaning, integration, selection, transformation, pattern evaluation and knowledge presentation [1]. Data mining includes forecasting what may happen in future, classifying things into groups by recognizing patterns, clustering things into groups based on their attributes and associating what events are likely to occur together. Data mining process is reliable process and repeatable process by the people with small quantity of data mining skills. Data mining have two types of learning technique such as supervised and unsupervised learning technique. The class label of each training tuple is known is referred as supervised learning. Unsupervised learning represents the class label of each training tuple not known in advance [1].

II. Literature Survey

Koru and Liu (2005)[2] built fault prediction models by using J48,V K-Star, and Random Forests on public NASA datasets and they used method and class level metrics. F-measure was selected as performance evaluation metric. KC1 dataset has method level metrics and they converted them into class level ones by using minimum, maximum, average and sum operations. Therefore, 21 method level metrics were converted into 84 (21 _ 4 = 84) class level metrics. They stated that large modules had higher F-measure values for J48, K-Star and random forests algorithms. F-measure was 0.65 when they applied class level metrics and when they chose method level metrics, F-measure was 0.40. Therefore, they stated that class level metrics improved the model performance, but detection of faults was at class level instead of model level.

Khoshgoftaar, Seliya, and Sundaresh (2006)[3] applied case based reasoning by using 24 product and four execution metrics on a large telecommunications system to predict software

faults. Performance evaluation metrics were average absolute error and average relative error. They reported that case based reasoning works better than multivariate linear regression and correlation based feature selection and stepwise regression model selection did not improve the performance of models. When CBR was used with Mahalanobis distance, the best performance was achieved. When principal component analysis is applied to remove the metrics' correlations, CBR with city block distance approach provided better results than CBR with Mahalanobis approach.

Gao and Khoshgoftaar (2007)[4] investigated the performance of Poisson regression, zero-inflated poisson regression, negative binomial regression model, Zero-Inflated negative binomial, and Hurdle regression (HP1, HP2, HNB1, HNB2) techniques on two embedded software applications which configure the wireless telecommunications products by using five file level metrics for software fault prediction. Performance evaluation metrics were Pearson's chi square measure, information criteria, average absolute error (AAE), and average relative error (ARE). They reported that model based on Zero-Inflated negative binomial technique performs better than the other algorithms according to the information criteria and chi square measures. Model based on HP2 technique was the best one when AAE and ARE parameters were used.

Riquelme, Ruiz, Rodríguez, and Moreno (2008) [5] investigated two balancing techniques with two classifiers, Naive Bayes and C4.5, on five public datasets from PROMISE repository for software fault prediction. They reported that balancing techniques improve the AUC measure, but did not improve the percentage of correctly classified instances. Performance evaluation metrics were AUC and percentage of correctly classified instances. Sampling metrics were resample implementation of WEKA and SMOTE.

Chang, Chu, and Yeh (2009) [6] proposed a fault prediction approach based on association rules to discover fault patterns. They reported that prediction results were excellent. The benefit of this method is the discovered fault patterns can be used in causal analysis to find out the causes of faults.

Arisholma, Briand, and Johannessen (2010) [7] evaluated fault-proneness models on a large Java legacy system project. They reported that modelling technique has limited affect on the prediction accuracy, process metrics are very useful for fault prediction, and the best model is highly dependent on the performance evaluation parameter. They proposed a surrogate measure of cost-effectiveness for assessment of models. Adaboost combined with C4.5 provided the best results and techniques were used with default parameters.

III. Data Mining Techniques for Fault Prediction

There are various data mining techniques used for predictions which are discussed below.

1. *Regression:* It is a statistical process to evaluate the relationship among variables. It analyses the relationship between the dependent or response variable and independent or predictor variables. The relationship is expressed in the form of an equation that predicts the response variable as a linear function of predictor variable. Linear Regression: Y=a+bX+u
2. *Association Rule Mining:* It is a method for discovering interesting relationships between variables in large databases. It is about finding association or correlations among sets of items or objects in database. It basically deals with finding rules that will predict the occurrence of item based on the occurrence of other items.
3. *Clustering:* Clustering is a way to categorize a collection of items into groups or clusters whose members are similar in some way. It is task of grouping a set of items in such a way that items in the same cluster are similar to each other and dissimilar to those in other clusters.
4. *Classification:* It consists of predicting a certain outcome based on a given input. Classification technique use input data, also called training set where all objects are already tagged with known class labels. The objective of classification algorithm is to analyze and learns from the training data set and develop a model. This model is then used to classify test data for which the class labels are not known.

a) *Neural Networks:* Neural Networks are the nonlinear predictive models which can learn through training and resemble biological neural networks in structure. A neural network consists of interconnected processing elements called neurons that work together in parallel within a network to produce output.
b) *Decision Trees:* A decision tree is a predictive model which can be used to represent both classification and regression models in the form a tree structure. It refers to a hierarchical model of decisions and their consequences. It is a tree with decision nodes and leaf nodes. A decision node has two or more branches. Leaf nodes represent a classification or decision.
c) *Naive Bayes:* It is based on Bayes theorem with independence assumption between predictors. Naive Bayes Classifier is based on the assumption that the presence or absence of a particular feature of a class in not related to the presence or absence of any other feature.
d) *Support Vector Machines:* SVM are based on the concept of decision planes that define decision boundaries. A decision plane is the one that separates between a set of objects having different class membership. SVM is primarily a classifier method that performs classification task by constructing hyper plane in a multidimensional space that separates cases of different class labels. It supports both regression and classification.
e) *Case Based Reasoning:* Case based reasoning means solving new problems based on the similar past problems and using old cases to explain new situations. It works by comparing new unclassified records with known examples and patterns. A simple example of a case based learning algorithm is k-nearest neighbor algorithm. It is simple algorithm that stores all available cases and classifies new cases based on a similarity measure i.e. distance function.

IV. Software Fault Prediction Metrics

1. *Lines of Code :* This metric calculate the faults by

a) The total number of lines
b) The number of blank lines in module
c) The number of lines of comments in a module
d) Lines of executable code
e) The number of lines which contain both code and comment in a module

2. *Cyclomatic complexity:* The complexity of software can be correlated with the complexity of the graph.

a) McCabe proposed the cyclomatic number. V(G) which is equal to the number of linearly independent paths through a program in its graphs representation to indicate the software complexity.
b) The V(G) for a program control graph G, is given by:

$$V(G) = E - N + P$$

c) Design Complexity: Design complexity measures the amount of interaction between the modules in a system.
d) *Essential complexity:* Essential Complexity (eV (G)) is a measure of the degree to which a module contains unstructured constructs.

This metric measures the degree of quality of the code. It is used to predict the maintenance effort and to help in the modularization process.

3. Halstead Metrics: Halstead metrics are computed statically from the code and was introduced by Halstead in 1977s [11] Metrics applicable to several aspects of program. The metrics are defined as follows. The following token counts are used to compute the various Halstead metrics

The metrics are defined as follows.
n_1 = the number of distinct operators
n_2 = the number of distinct operands
N_1 = the total number of operators
N_2 = the total number of operands
Halstead length content
$N = N_1 + N_2$
Halstead volume metric
Volume metric is a measure of the storage
Volume required to represent the program.
$V = N. \log_2, n$
Where $n = n_1+n_2$
Number of Faults
$Faults=V/S_0$

V. Conclusion

Software fault prediction is the process of tracing defective components in software prior to the start of testing phase. Occurrence of defects is inevitable, but we should try to limit these defects to minimum count. Defect prediction leads to reduced development time, cost, reduced rework effort, increased customer satisfaction and more reliable software.

References

[1] Jiawei Han and Michline Kamber, " Data Mining concepts and techniques", Morgan Kaufmann publishers.
[2] Koru, A. G., & Liu, H. (2005). An investigation of the effect of module size on defect prediction using static measures. In Workshop on predictor models in software engineering (pp. 1–5). Missouri: St. Louis.
[3] Khoshgoftaar, T. M., Seliya, N., & Sundaresh, N. (2006). An empirical study ofpredicting software faults with case based reasoning. Software Quality Journal, 14(2), 85–111.
[4] Gao, K., & Khoshgoftaar, T. M. (2007). A comprehensive empirical study of count models for software fault prediction. IEEE Transactions for Reliability, 56(2), 223–236.
[5] Riquelme, J. C., Ruiz, R., Rodríguez, D., & Moreno, J. (2008). Finding defective modules from highly unbalanced datasets. Actas taller sobre el apoyo a la decisión en ingeniería Del software (ADIS-JISBD) (pp. 67–74).
[6] Chang, C., Chu, C., & Yeh, Y. (2009). Integrating in-process software defect prediction with association mining to discover defect pattern. Information and Softwar Technology, 51(2), 375–384.
[7] Arisholma, E., Briand, L. C., & Johannessen, E. B. (2010). A systematic and comprehensive investigation of methods to build and evaluate fault prediction models. Journal of Systems and Software, 83(1), 2–17.