Software Fault Prediction Methods: A Brief Survey

Sajna P[1], Shahad P[2]

[1]MEA Engineering College, State Highway 39, Nellikkunn-Vengoor, Perinthalmanna, Malappuram
[2]MEA Engineering College, State Highway 39, Nellikkunn-Vengoor, Perinthalmanna, Malappuram

Abstract: ***Software fault prediction is a valuable exercise in software quality assurance to best allocate limited testing resources .One strategy is to process and analyze previous generated data to predict future failures. This makes it extremely important in software development to develop quality and fault free software. In this paper, we discuss Data mining techniques for software defect prediction. Various classifiers are used to classify faulty or non-faulty modules. Then we compare our methods and find which method is better. The quality of software datasets can be improved by data preprocessing. In empirical studies, we chose datasets from real-world software projects, such as Eclipse and NASA.***

Keywords: Data Mining, Fault Detection, Prediction Algorithms, Data preprocessing

1. Introduction

Software fault prediction is a hot research topic in software engineering. It can allocate the limited test resources effectively by predicting the fault proneness of software modules. Classification is one of the prevalent methods used for software fault prediction. Its main task is to first categorize software modules that is represented by a set of software metrics, into two classes: fault-prone modules (FP), or non-fault-prone modules (NFP).

A software defect is a mistake, a default, an error, a failure or a fault in a computer program or a system which produces incorrect or unexpected results, or which causes unintended behavior. The forecast of software defects is the localization of the faulty modules process in the software. It allows to improve the quality of the software and the effectiveness of tests by building predictive models from code attributes to allow quick identification of the vulnerable modules, helps us to plan, monitor and control and predict the density of defects and to better understand and control the quality of the software. The result of prediction of software defects, i.e. the number of defects in a software system, can be used as an important measure for the developer of software and can be used to control the software process.

Detecting software faults prior to system development may reduce software maintenance costs. Early software fault prediction helps to improve software quality and to achieve high software reliability. Defect Prediction Models locate error causing software system to ensure quality assurance activities like tests/code reviews. Most current models assume that quality assurance cost for all models are the same. When effort is considered, many classifiers performance is almost the same as randomly chosen modules.

In recent years, researchers have found that the quality of software datasets had serious effect on the performance of predicting software faults. Issues concerned in data quality include biased datasets, noise a large number of features, and class imbalance. One is the high dimensionality problem caused by too many unnecessary features, and the other is the class imbalance problem caused by superfluous instances of some classes. The former can be solved by feature selection, which selects a small fraction of the features by removing irrelevant or redundant ones. The latter is handled by instance sampling (or reduction), which samples a subset of the instances from majority classes. Both have been proven useful by previous experimental studies. In this paper, we will discuss the techniques of Data mining for prediction of software defects. Data mining is a process of data analysis from various perspectives and summarizes it into useful information. It helps users to understand the substance of the relationships between the data.

2. Literature Review

2.1 K-means clustering Algorithm

K-Means clustering is a non-hierarchical clustering procedure in which elements are moved between sets of clusters until the desired set is reached [1]. Zhong et al. [2], [3] applied grouping techniques and an expert-based approach to the problem of software defect prediction. As in this study, the data are divided into two groups according to whether they are free of defects or defects. Distance measurement is an important step in grouping that will determine how the similarity of the two elements is calculated. K-Canberra means clustering [4] uses Canberra. This approach yields results that show better and more accurate results compared to traditional K-means clustering.

In [5], the K-means clustering algorithm with the Sorensen distance function is used with the K-means clustering algorithm and the modified algorithm is called the K-Sorensen-means clustering algorithm. The Sorensen distance is a standardization method that considers space as a grid similar to the distance from the city block. K-Sorensen means is more accurate and likely to predict more faults than K-Canberra means clustering. The Sorensen Distance function is illustrated below,

$$S_j^t = \left\{ x_j : \frac{\| x_j - m_j \|}{\| x_j + m_j \|} \; for \; all \; j = 1..k \right\} \quad (1)$$

In [6] Quad Tree based algorithm is applied for predicting faults in program modules. The clusters obtained by Quad Tree-based algorithm were found to have maximum gain values. Quad Trees are applied for finding initial cluster centers for K-Means algorithm. A Quad Tree in two dimensional spaces is a 4-way branching tree that represents recursive decomposition of space using separators parallel to the coordinate axis. The overall error rates of this prediction approach are compared to other existing algorithms and are found to be better in most of the cases [7]. In [8] proposed Hyper Quad-tree, which will be given an input to the K-Means algorithm to lowers the fault rate and effective software fault prediction. Hyper Quad-Trees are expected to give improved cluster centers than the Quad-tree. Hyper Quad Trees algorithm is applied for finding the cluster centers which will be an input to the K-Means clustering algorithm. Hyper Quad Tree based K-Means algorithm offer better cluster center and lowers the Fault ratio as well as errors in a given data set as compared to the Quad Tree based K-means algorithm.

2.2 Naive Bayes Software Defect Prediction Model

Na¨ıve Bayes(NB) is a very effective machine learning method. A NB model treats defect prediction as binary classification, it trains and constructs predictor by analyzing historical data of software modules, based on predictor it will make decision whether new module has defects or not [9].

In [10] proposed Naive Bayes Prediction (NBP) model chooses software module to be object unit of training and prediction. Let A= a1,a2,...,an be set of metrics attribute set, there is a vector M: (a1,w1),(a2,w2),...,(an,wn) to denote a software module, where 'a' is metrics and 'w' is weight of 'ai' . We train the classifier using module data sets with category tag, and compute the defective probability of a new module which will make an alert when it exceeds a threshold.

If we define the category notion of software module is C∈(Cd,Cn) where Cd is defective category and Cn is non-defective category. Then according to Bayesian theory, the probability that software module is defective will be computed by,

$$P[C_d \mid M] = \frac{P[M \mid C_d] \times P[C_d]}{\sum_{c \in C} P[C] \times P[M \mid C]} \qquad (2)$$

Classifier will classify software module M to Cd when the ratio greater than threshold. Estimation of P[M |c] is a key problem of NBP model .To solve this problem we use Multi-variants Gauss Naïve Bayes [10]. When attribute value of metrics is real-valued, Multi-variants Gauss Naïve Bayes (MvGNB) estimates P[M|c] using ,

$$P[M \mid c] = \prod_{i=1}^{n} g(w_i; \mu_{i,c}, \sigma_{i,c}) \qquad (3)$$

Where we supposed that each attribute follows a normal distribution 'g' in each category 'c', and the mean (μ) and typical deviation (σ) of each distribution are estimated from the training data sets. In[11]derived the Weighted NaïveBayes method, and then describe three heuristics for feature weight assignment. Used three heuristics in order to estimate the weights of features based on their relative importance. Two novel heuristics are introduced for this purpose. We have evaluated our approach on Weighted Naïve Bayes predictor, which is an extension of standard Naïve Bayes. To the best of our knowledge, the weighted features approach is a novel one in defect prediction literature. We observed linear methods for feature weighting lack the ability to improve the performance of Naïve Bayes

2.3 Support Vector Machine model

SVM are useful tools for performing data classification. and have been successfully used in applications. SVM constructs an N-dimensional hyperplane that optimally separates the data set into two categories. The purpose of SVM modeling is to find the optimal hyperplane that separates clusters of vector in such a way that cases with one category of the dependent variable on one side of the plane and the cases with the other category on the other side of the plane [12]. The support vectors are the vectors near the hyperplane. The SVM modeling finds the hyperpalne that is oriented so that the margin between the support vectors is maximized. When the given points are separated by a nonlinear region, SVM classifier handles this by using a kernel function in order to map the data into a different space when a hyperplane can be used to do the separation. Decision stump achieves slightly higher classification accuracy, the precision and f measure is much lower. Elish et al. [12] and [13] stated that the performance of Support Vector Machines (SVM) is generally better than, or at least is competitive against the other statistical and machine learning models in the context of four NASA datasets. In [14] Proposed ensemble SVM. The bagging algorithm creates an ensemble of models for a learning scheme where each model gives an equally weighted prediction. In this study, the Support Vector Machine is constructed and 10-fold cross validation technique is applied and evaluated error rate from the mean square error. Secondly, bagging is performed with Support Vector Machine to obtain a very good generalization performance.

2.4 Decision tree model

The Decision tree is one of the classification techniques which is done by the splitting criteria. The decision tree is a flow chart like a tree structure that classifies instances by sorting them based on the feature (attribute) value[15][16]. Each node in a decision tree represents a feature in an instance to be classified. All branches denote an outcome of the test, each leaf node hold the class label. The instances are classified from starting based on their feature value. Decision tree generates the rule for the classification of the data set. Decision trees use feature values for the classification of instances. A feature in an instance that has

to be classified is represented by each node of the decision tree, while the assumption values taken by each node is represented by each branch. The classification of instances is performed by following a path through the tree from root to leaf nodes by checking feature values against rules at each node. The root node is the node that best divides the training data.

Three basic algorithms are widely used that are ID3, C4.5, and CART [18].ID3 is an iterative Dichotomer 3. It is an older decision tree algorithm introduced by Quinlan Ross in 1986. The basic concept is to make a decision tree by using the top-down greedy approach. C4.5 is the decision tree algorithm generated by Quinlan. It is an extension of ID3 algorithm.. C4.5 algorithm is widely used because of its quick classification and high precision. CART stands for Classification Regression Tree introduced by Bremen. The property of CART is that it is able to generate the regression tree.

The C4.5 can be referred as the statistic Classifier. This algorithm uses gain radio for feature selection and to construct the decision tree[17]. It handles both continuous and discrete features. C4.5 algorithm is widely used because of its quick classification and high precision. C4.5 based technique that uses information entropy to build the decision tree. At each node of the decision tree, a rule is chosen by C4.5 such that it divides the set of training samples into subsets effectively. The C4.5 algorithm is an inductive supervised learning system which employs. Decision trees to represent a quality model. C4.5 is a descendent of another induction program[17]

2.5 KNN model

The K-Nearest Neighbor (NN) is the simplest method of machine learning. It is a type of instance base learning in which object is classified based on the closest training example in the feature space[19][1]. It implicitly computes the decision boundary however it is also possible to compute the decision explicitly. So the computational complexity of K NN is the function of the boundary complexity .The k-NN algorithm is sensitive to the local structure of the data set. The special case when k = 1 is called the nearest neighbor algorithm. The best choice of k depends upon the data set; larger values of k reduce the effect of noise on the classification but make boundaries between classes less distinct. The various heuristic techniques are used to select the optimal value of K. KNN has some strong consistent results. As the infinity approaches to data, the algorithm is guaranteed to yield an error rate less than the Bayes error rate.

Nearest neighbour (a.k.a., lazy-learning) techniques are another category of statistical techniques. Nearest neighbor learners take more time in the testing phase, while taking less time than techniques like decision trees, neural networks, and Bayesian networks during the training phase .In this paper, we study the KNN nearest neighbor technique. KNN considers the K most similar training examples to classify an instance. KNN computes the Euclidean distance to measure the distance between instances. We find K = 8 to be the best-performing K value of the five tested options (i.e., 2, 4, 6, 8, and 16).

3. Discussion

Prediction performance measures

In order to evaluate performance of these models, we compared prediction results. We choose the AUC measure to evaluate the prediction performance. The results demonstrate the potential of our approach in enhancing the prediction performance of the classifiers built thereafter.

3.1 Confusion Matrix

A confusion matrix is a visualization tool that reports the number of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN). TN represents the fault free modules correctly classified. FP refers to fault free modules incorrectly labelled as faulty modules. FN corresponds to faulty modules incorrectly labelled as fault free modules. TP refers to modules that are correctly classified as faulty modules [5].

3.2 ROC curve

AUC-ROC is used as a performance metrics [5][12][19]. A receiver operating characteristic (ROC) curve can be represented equivalently by plotting the probability of detection (PD) vs. probability of false alarms (PF).ROC curves can be beneficial for finding accuracy of predictions.ROC curve is divided in two different regions defined as follows. Risk incompatible region with high PD and high PF, is beneficial for safety critical systems as identification of faults is more important than validating false alarms. Cost incompatible region defines low PD and low PF, this region is beneficial for the organizations having limited Verification Validation budgets. Negative region with low PD high PF is also preferred for some of the software projects.ROC analysis can easily avoid this risk.AUC is area under the ROC curve. Higher AUC values indicate that the classifier is on average more to the upper left region of the graph.

3.3 Accuracy

Accuracy is also known as correct classification rate. It can be find out as the ratio of the number of modules correctly predicted to the total number of modules. The accuracy is calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

3.5 Results

The comparison results are shown in Table 1 and Table 2. In Table 1 we can understand SVM shows more accurate than k-means and Naïve Bayes software defect prediction models. SVM has higher ROC value so SVM is good classifier for predicting software faults.

In Table 2 shows advantages and disadvantages of each model. In this Table we can clearly understand which model is best for software fault prediction.

Table 1: Comparison Results of Algorithms

Algorithm	Accuracy (%)	AUC(avg)
K-means clustering algorithm	67	0.63
NBP model	69	0.68
SVM model	70	0.70
DT Model	68	0.66
KNN Model	60	0.61

Table 2: Advantages and Disadvantages of Algorithms

Algorithm	Advantage	Disadvantage
K-means clustering algorithm	Fast and cheap and Easy to understand Achieve high software reliability.	Less Accuracy Very sensitive to noise. Inefficient clustering
NBP model	Very effective in the case of micro sampling. Highly scalable Robust to irrelevant features	High complexity. Difficult to estimate probability Matrix
SVM model	SVM models is feasible and adaptable. The SVM method yielded good AUC Value	Models can be quite sensitive in the case of over-fitting. Lack of transparency results.
DT Model	Comprehensive Nature ,Flexible Implicitly perform feature selection Relatively little effort from users for data preparation	May suffer from overfitting Does not easily handle nonnumeric data Pruning is necessary
KNN Model	The cost of the learning process is zero Best and most widely used Complex concepts can be learned using simple procedures	The model cannot be interpreted It is computationally expensive when the dataset is very large Performance depends on the number of dimensions

3. Conclusion

In this paper, we have discussed that how data mining techniques are used for software defect prediction. Defect prediction is based on good data mining model. In this we surveyed different data mining algorithms used for defect prediction. We also discuss the performance and effectiveness of data mining algorithms. This survey also has showed that all the issues for selecting a data mining technique for defect prediction. Our most important finding is that there is no single data mining technique that is more powerful or suitable for all type of projects. In order to select a better data mining algorithm, domain expert must consider the various factors like problem domain, type of data sets, nature of project, uncertainty in data set etc. Multiple classifiers were combined by majority voting of experts to get more accurate result.

References

[1] Han, Jiawei, Jian Pei, and Micheline Kamber. Data mining: concepts and techniques. Elsevier, 2011.
[2] Zhong, Shi, Taghi M. Khoshgoftaar, and Naeem Seliya. "Unsupervised Learning for Expert-Based Software Quality Estimation." HASE. 2004.
[3] Zhong, Shi, Taghi M. Khoshgoftaar, and Naeem Seliya. "Analyzing software measurement data with clustering techniques." IEEE Intelligent Systems 19.2 (2004): 20-27.
[4] Jiang, Yue, Bojan Cukic, and Tim Menzies. "Cost curve evaluation of fault prediction models." Software Reliability Engineering, 2008. ISSRE 2008. 19th International Symposium on. IEEE, 2008.
[5] Kaur, Deepinder, et al. "A clustering algorithm for software fault prediction." Computer and Communication Technology (ICCCT), 2010 International Conference on. IEEE, 2010.
[6] Bishnu, Partha S., and Vandana Bhattacherjee. "Software fault prediction using quad tree-based k-means clustering algorithm." IEEE Transactions on knowledge and data engineering 24.6 (2012): 1146-1150.
[7] Patil, Swapna M., and R. V. Argiddi. "Study Of Fault Prediction Using Quad Tree Based K-Means Algorithm And Quad Tree Based EM Algorithm."
[8] Varade, Swati, and Madhav Ingle. "Hyper-quad-tree based k-means clustering algorithm for fault prediction." International Journal of Computer Applications 76.5 (2013).
[9] Lessmann, Stefan, et al. "Benchmarking classification models for software defect prediction: A proposed framework and novel findings." IEEE Transactions on Software Engineering 34.4 (2008): 485-496.
[10] Wang Tao, LI Wei-hua "Naïve Bayes Software Defect Prediction Model"School of ComputerScience and Technology,Northwestern Polytechnical University, Xi'an, China.water@snnu.edu.cn
[11] Turhan, Burak, and Ayse Basar Bener. "Software Defect Prediction: Heuristics for Weighted Naïve Bayes." ICSOFT (SE). 2007.
[12] Elish, Karim O., and Mahmoud O. Elish. "Predicting defect-prone software modules using support vector machines." Journal of Systems and Software 81.5 (2008): 649-660.
[13] Singh, Yogesh, Arvinder Kaur, and Ruchika Malhotra. "Software fault proneness prediction using support vector machines." Proceedings of the world congress on engineering. Vol. 1. 2009.
[14] Shanthini, A., G. Vinodhini, and R. M. Chandrasekaran. "Bagged SVM Classifier for Software Fault Prediction." International Journal of Computer Applications 62.15 (2013).
[15] Chen, Mike, et al. "Failure diagnosis using decision trees." Autonomic Computing, 2004. Proceedings. International Conference on. IEEE, 2004.
[16] Dietterich, Thomas G. "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization." Machine learning 40.2 (2000): 139-157.

[17] Sharma, Seema, Jitendra Agrawal, and Sanjeev Sharma. "Classification through machine learning technique: C4. 5 algorithm based on various entropies." International Journal of Computer Applications 82.16 (2013).
[18] Quinlan, J. Ross. "Induction of decision trees." Machine learning 1.1 (1986): 81-106.
[19] Wilson, D. Randall, and Tony R. Martinez. "Reduction techniques for instance-based learning algorithms." Machine learning 38.3 (2000): 257-286.
[20] Bradley, Andrew P. "The use of the area under the ROC curve in the evaluation of machine learning algorithms." Pattern recognition 30.7 (1997): 1145-1159.

References

Sajna P. received the B.Tech degrees in Computer Science and Engineering from MEA Engineering college Perinthalmanna, Malappuram During 2009-2013 . Right now she is pursuing her M. Tech degree in computer science at MEA Engineering College, Kerala from 2015 to 2017. Her research interests lie in Data Mining.

Shahad P, Assistant Professor, MEA Engineering College, Perinthalmanna. Received the B.Tech degree in Computer Science & Engineering from Maharaja Prithvi Engineering College Coimbatore. and M.Tech degree in Computer Science from College of Engineering, at MEA Engineering College Perinthalmanna. His research interests include Data Mining.