VOL. 13, NO. 5, MARCH 2018 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

CLASSIFYING SOFTWARE FAULTS THROUGH BOOLEAN CLASSIFICATION MODEL BASED ON DISCRETIZED DEFECT DATASETS

Pooja Kapoor[1], Deepak Arora[1] and Ashwani Kumar[2]
[1]Department of Computer Science and Engineering, Amity School of Engineering & Technology, Amity University, India
[2]IT and Systems, Indian Institute of Management, Lucknow, India
E-Mail: inkhanna@gmail.com

ABSTRACT

Identifying software defects has always been one of the major concerns of software designers across the entire software industry. The software faults which were not determined previously during its testing phase can lead to a complete failure of entire software. With the advent of latest technologies, the software becomes more distributed and complex in nature, the requirement of predicting such faults at an earlier stage of software development process has become essential requirement. In this research work authors have proposed and implemented the idea of Discretization of metric values in order to get better classification results. Authors have generated Boolean functions based on project metrics values, so that these values could be confined in the domain of classifying and predicting software faults. Authors have also checked the performance of their proposed model by considering seventeen different software projects and their versions taken from promise repository of NASA namely *Jedit*, *lucene, tomcat, velocity, xerces and xalan*. The results gained after applying the proposed Boolean classifier are found better and more promising in terms of its accuracy and precision. Authors have also found results of their experiment satisfactory when compared with the available literature. This study claims visible increase in accuracy, as compared to other classifiers considered in the study like *Naïve Bayes, Random Forest, Perceptron, KNN and SOM.*

Keywords: object oriented software metrics, discretization, boolean classifier.

1. INTRODUCTION

In classification process, a classification model is generated using training data and same is then evaluated using test data. Different assesment models like prediction accuracy, precesion, recall etc. are used to check the fit of the classification model formed. The current work, is intended to build a classification model based on the boolean data generated using the discretization method, applied on software defect data set [1]. The boolean data exhibits a pattern and to analyse such patterns, authors have suggested a boolean classifier. The classifier model suggested in the current research work, has been further optimized using K-Map reduction.The complete dataset is clustered to identify the patterns. The identified patterns are then used to generate the boolean function for classification model. The fit of the proposed classification model, is checked using test data. The assesment model used in the study is prediction accuracy. The study conclude a significant increase in the accuracy of the proposed classifier as compared to Naive Bayes classifier. Many studies found in the literature has established a co-relation between the software metrics and the final software quality. Majority of work has been done using CK metric suite. CK metric suite has been proposed by Chidamber, Kemerer [2]. Authors have suggested six metircs for Object Oriented systems and established a relation of CK metrics with quality requirements. The study used linguistic terms of each metrics in metric suite to predict the functional quality requirements. Kapoor et al. has proposed, that values of CK metrics can predict the occurence of fault at design level. The study suggested a threshold range for each metrics in CK suite. In this study authors have evaluated eighteen differnt defect data set and found that if the over all distribution of the CK metrics is within the threshold range it ensures less occurence of faults [3]. Kapoor *et al.*, has proposed a discretization method, in order to increase the overall efficiency of the existing CK Metrics that is prescribed and widely used for quality assurance for Object Oriented System develeopment [3]. Using the proposed threshold based discretization method, CK metric suite defect data, is converted in categorical form {0,1}. Eighteen different defect data sets are considered for the study, taken from NASA repositories. The study concluded a increase in classification efficiency of classifiers like Naive Bayes and the Voted Percepton in an significant manner [1]. The study suggests that by using certain threshold value of metrics, the efficiency of CK metrics in prediction of software quality can be increased towards predicting software defects at an early stages of its development. The rest of this paper is organized as follows: In section 2, Background is discussed. In section 3, the proposed Algorithm for Boolean classifier is discussed. In section 4, results of the proposed methodology are discussed and compared using the various statistical measures. Section 5 finally presents the conclusions of the study.

2. BACKGROUND

Many study in literature used Boolean function in various domain of engineering and science.Pawalak and skowron presented a method to use Boolean reasoning along with rough set. The study explained the use of

Boolean reasoning in various domains like pattern recognition, machine learning, and conflict analysis [4]. Rademacher-Walsh transformation can be used to classify Boolean functions,which enable Boolean functions to be synthesized in an elegantway by using a universal threshold logic gate [5]. Another study suggested an approach to use Boolean system to check the reliability of complex system and suggested a measure how Boolean system can be re converted to a probalistic model [6]. Karnaugh map, is a simple method used for mininizing the Boolean equation.Though it is applicable for small design problems but is very efficient and simple method for boolean equation reduction. K-Map (Karnaugh map) is a grid like structure for representation of the Truth Table data. Such representation is helpful in reducing the number of inputs and gates, required to solve a specific logic problem. For any boolean function **f** comprising of minterms or maxterms, a K-Map can be drawn. A boolean function defines a mapping $\{\mathbf{0, 1}\}^{\mathbf{n}} \rightarrow \{\mathbf{0, 1}\}$. K-Map uses grouping method for min or max term reduction. It is a graphical tool that could be used in many other domains for mapping and error code generation [7]. The study presented, use of Karnaugh Map in error detection and error correction, as an extension to hamming code. The study suggested that, K-Map representation has made error corection and detection easy and simple [12]. A K-Map is a geometrical structure that do not require advanced mathematics for generation of codes [8-9]. The study concluded that encoding and decoding done using K-Map approach can be used for parallel processing. Another study proposed a way where a Karnaugh Map is used as statistical tool to analyse the health care data [10]. The study used a filtering method to convert the linguistic statements used for analysis of twenty five different studies related to clinical coding data accuracy in Boolean form. K-Map is used to drive research function or expresssion. The expression is further used for analyzing the trends in data of differnt studies related to clinical coding data accuracy [4]. The study presented a way how K-Map can be used in simplifying the overview of meta-analysis and systemic reviews.

3. PROPOSED METHODOLOGY/ALGORITHM

For current work authors have considered the Boolean data set [1]. The data set is then converted or re-written in Truth Table form. A Truth Table is an arrangement of columns and rows to represent logic in such a way that column contains the input signal and the output column represents the expected output for the input provided. The Truth Table used for study has five input columns as *wmc, dit, noc, cbo, rfc labeled as A,B,C,D,E* and output column Y which contains the defect status.{1 for fault and 0 for no fault }.The entire data set in form of truth table is clustered together for data segmentation or pattern identification. The extracted data is then used to build the Boolean classification model. The Boolean model generated can be used for data classifier. *Algorithm 1* describes the steps used for generating the Boolean function used for classification.

Algorithm 1:

Step 1: Reduce the data set using attributed reduction technique

Step 2: Data set S is sorted data in any order according to the attributes A = $\{a_1, a_2, a_3.....a_m\}$ of data set S, Where m is number of features/attributes in S

Step 3: The data set S is grouped into item set B = $\{b_1, b_2,.....b_n\}$ where $n<=2^m$

Step 4: Each element of set B has been assigned an output class C = {0, 1}, c_i=maximum (Output of tuples bj)

Step 5: An optimized Boolean function is than generated from item set B using K map reduction.

3.1Classification model

In the current research work a Boolean classification model has been proposed with the help of K map reduction. Table-1 represents the Boolean function generated for seventeen different systems under study. Truth Table is generated from defect data set and function is optimized using K- Map. The same has been verified using LOGIC FRIDAY, a freeware available for generating Boolean functions from the Boolean data.

Table-1. Represents the twelve systems under study and their Boolean Classification model generated using Algorithm 1

S. No.	System under study	Boolean function (Y)
1	Jedit 3.2	$Y_{jedit3.2}$= ~BC+ ~A~DE
2	Jedit 4.0	$Y_{jedit\ 4.0}$=~A ~B +~B ~D~ E+ A B ~C ~D
3	Jedit 4.1	$Y_{jedit\ 4.1}$= ~A +~B ~E
4	Jedit 4.2	$Y_{jedit\ 4.2}$= ~A~B C+~A B~C
5	Jedit 4.3	$Y_{jedit\ 4.3}$ =ABCD~E
6	Lucene2.0	$Y_{lucene\ 2.0}$ = ~A E + ~ C E + B E + B C D +A C D + ~B ~C D + A B ~ C ~D
7	Lucene2.2	$Y_{lucene\ 2.2}$ =~ B + ~A~C + ~ D E + C D
8	Lucene2.4	$Y_{lucene\ 2.4}$ =~ B ~D + ~ B E + ~A C + C E + B D + A ~B
9	Tomcat	Y_{tomcat} = ~A B~C~E +~A~B~C~D E + A~ B~C D E
10	Velocity 1.4	$Y_{velocity\ 1.4}$ = ~ A+ ~ E + C
11	Velocity 1.6	$Y_{velocity\ 1.6}$ = ~ C ~ D+ ~ B E + ~ D E+ A C E+A B ~ D
12	Xalan 2.4	$Y_{xalan\ 2.4}$ = ~A~B~D~E +~A~C~D~E + A~B~C~E
13	Xalan 2.5	$Y_{xalan\ 2.5}$ = ~A~E +~A~C~D +~B D E + B~ D~E +~A~ C D E
14	Xalan 2.6	$Y_{xalan\ 2.6}$ = ~C~E +~D~E +~A~C~D + A~C D +~A~ B C E
15	Xerces 1.2	$Y_{xerces\ 1.2}$ = D E +~A~B~ E +~A~C~D +~A C~D~E
16	Xerces 1.3	$Y_{xerces1.3}$ = ~B~C +~A ~C~D E
17	Xerces 1.4	$Y_{Xerces\ 1.4}$ = ~B + ~C + D + ~A~E + A E1

The defect data set that contains CK metrics and independent variable as occurrence of fault, exhibits a pattern that is identified using data segmentation and identified patterns are then converted in form of Boolean functions. Boolean function for $Y_{jedit\ 4.1}$= ~ A +~ B ~ E, specifies that for classification we would be requiring three dependent variables A(*wmc*), B(*dit*), E(*rfc*). Here wmc is weighted methods per class, dit is Depth of inheritance and rfc is response for class.

4. RESULT ANALYSIS AND DISCUSSION

Performance of the proposed Boolean classifier is verified using the classification accuracy model. Accuracy is defined as the closeness of a measured value to a standard or known value. Authors used Naive Bayes, Perceptron, KNN, SOM and Random forest classifiers for performance analysis for proposed Boolean classifier. For analysis purpose authors have considered seventeen different defect set in Table-2. The Boolean function is generated for each system under the study using the proposed algorithm (*Algorithm 1*); Table-1 lists all the Boolean models generated using the proposed algorithm. These seventeen Boolean modes are then used for classification of their respective test defect data set. Table-2 represents the accuracy obtained after running the proposed classifier and other classifiers. For the analysis of the Boolean models, authors have considered classifiers from a wide range of pre established classifier families. Naive Bayes is a well-known probabilistic classifier. Similarly random Forest is a rule based classifier and Perceptron is a single layer neural network classifier. For the experimental purpose these classifiers are run using WEKA 3.6 tool. It is clear from the Table-2 that using Boolean classifier, authors found better accuracy for many systems considered in the study. The detailed description is as follows:

Table-2. Accuracy comparison of proposed Boolean classifier with five different available classifiers using seventeen different defect dataset.

	Boolean Classifier	Naive bayes	Random forest	Perceptron	KNN	SOM
Jedit 3.2	55	72.05	72	73.1	73.1	74
Jedit 4.0	78.7	77.45	79	78.1	78.75	77
Jedit 4.1	79.4	79.16	76.9	77.24	76.9	79
Jedit 4.2	88	86.6	86.3	87.4	86.9	86
Jedit 4.3	79.6	96.9	97.7	97.7	97.7	97
Lucene 2.0	69.2	63.5	62	63	62.5	65
Lucene 2.2	61.5	55	54.2	57	55	57
Lucene 2.4	62.9	63.2	57.6	59.7	58.5	57
Tomcat	90.2	90.7	91	90.6	90.9	91
Velocity 1.4	75.5	74.4	72.9	75.5	75.5	0.75
Velocity 1.6	69.4	64.1	64.1	66.3	67.2	0.65
Xalan 2.4	84	84.5	83.8	84.5	84.7	84
Xalan 2.5	59	54.6	58.1	54.6	59.2	55
Xalan 2.6	64.4	59.7	62.9	60	63.2	61
Xerces 1.2	74.7	80.2	83.6	83.1	82.5	83
Xerces 1.3	84.7	83.2	83.6	84.1	84.5	84
Xerces 1.4	68.8	70.4	74.8	73.1	75	74

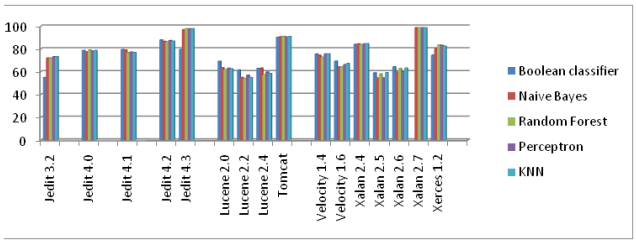


Figure-1. Bar graphs that provides the compartive analysis of the proposed Boolean classifer and other classifiers used in the study.

Jedit 4.2, Lucene 2.0, lucene 2.2, xalan 2.6, xerces 1.3 are the defect set/systems for which current work obtain better accuracy using Boolean classification model (Figure-1). The proposed classifier has achieved 3-6% increase in accuracy, compared to many classifiers considered in the study like Naïve Bayes, Random forest, Perceptron, SOM. Figure-1 presents comparative graph, for most of the other systems the accuracy obtained using proposed classifier model is comparable to other known and well recognized classifiers. For few cases like jedit 3.2, jedit 4.3, xerces 1.4, accuracy obtained is less as compared to the other classifiers. Table-3 represents the performance analysis of proposed Boolean classifier, using precision and recall. Beside accuracy, Precision and recall are the basic measure to analyze the performance of the classifier [13]. Precision defines ratio of the correct prediction out of the total predicted values for a class.

$$precision(n\ class) = \frac{TN}{(TN+FN)} \quad (1)$$

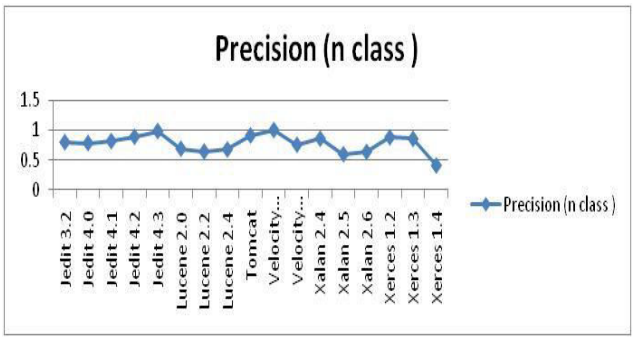
$$precision\ (y\ class\ ) \frac{TP}{(TP+FP)} \quad (2)$$

Equation (1) provides precision value of *n class*. It says ratio of true negative values (TN) from total number of instances predicted as in negative class (TN+FN). Precision analysis is important in case where there is imbalance in number of instance of output class of the training data set. In such case classification accuracy can come out to be Very high, but value of precision will decide how well the classifier can classify each class. As in case of velocity 1.4 in Table 3, precision for *n class* is recorded as 1. This signifies 100% correct prediction for *n class* from total prediction for *n class* (TN=1, FN=0). But in similar case precision for *y class* is .75 *i.e.* only 75 % prediction for y class is correct, of total prediction for *y class*. Equation (1) provides precision value of *y class*. Table-3 also presents the recall calculated for each of the seventeen systems under the study. Recall is the ratio of instances that are actually correct, were also predicted correct.

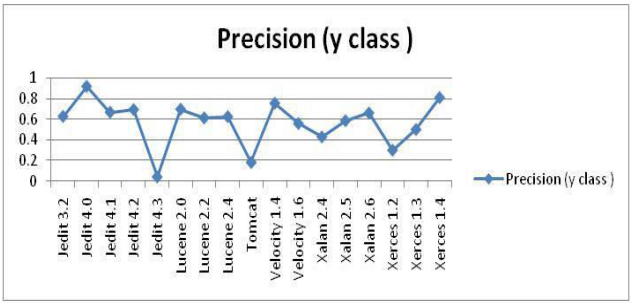
Table-3. Performance analysis of proposed Boolean classifier using precision and recall for n class using seventeen different defect dataset.

System	TN	TP	FP	FN	Recall (n class)	Precision (n class)	Precision (y class)
Jedit 3.2	151	52	31	38	0.82967	0.798942	0.626506
Jedit 4.0	230	11	1	64	0.995671	0.782313	0.916667
Jedit 4.1	218	30	15	49	0.935622	0.816479	0.666667
Jedit 4.2	315	9	4	39	0.987461	0.889831	0.692308
Jedit 4.3	388	4	93	7	0.806653	0.982278	0.041237
Lucene 2.0	80	55	24	36	0.769231	0.689655	0.696203
Lucene 2.2	18	134	85	10	0.174757	0.642857	0.611872
Lucene 2.4	21	193	116	10	0.153285	0.677419	0.624595
Tomcat	772	2	9	75	0.988476	0.911452	0.181818
Velocity 1.4	1	147	48	0	0.020408	1	0.753846
Velocity 1.6	120	39	31	39	0.794702	0.754717	0.557143
Xalan 2.4	601	9	12	101	0.980424	0.856125	0.428571
Xalan 2.5	271	204	145	183	0.651442	0.596916	0.584527
Xalan 2.6	371	199	103	212	0.7827	0.636364	0.65894
Xerces 1.2	299	30	70	41	0.810298	0.879412	0.3
Xerces 1.3	380	4	4	65	0.989583	0.853933	0.5
Xerces 1.4	72	333	79	104	0.476821	0.409091	0.8082

Figure-2 (a) (b) represents the graph to compare the precision of *n class* and precision of y *class* for seventeen, different data set considered in the study. The significance of each of these performance measures like precision and recall of n class and y class, are highly case dependent. In present study the objective is to analyses the predictive capabilities of available Object Oriented software metrics like CK metrics. The major concern of authors here is prediction of No fault occurrence (*n class* i.e. No fault) at the design level. Figure-3 represents the comparative analysis of precision and recall for *n class.* As in Figure-3, high value of precision and low recall ensures confident prediction about *n class* occurrence, where *n class* indicates NO occurrence of fault.



(a)



(b)

Figure-2(a) (b). Represents the comparison of precision *n class: (a)* and precision *y class: (b)*, for seventeen different defect data set.

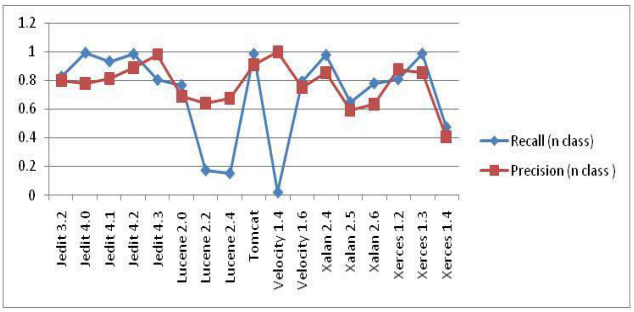


Figure-3. Represents the comparison of precision *n class* and recall n *class*, for seventeen, different defect data set.

As in Figure-3, Jedit4.0, Jedit 4.1, Jedit 4.2, lucene 2.0, tomcat and xalan 2.4 precision is higher than recall. A high recall ensures ratio of total prediction for *n class* out of actual n class. To avoid missing, *n class* occurrence recall should be high. With proposed Boolean classifier Jedit 4.3, lucene 2.2, lucene 2.4, velocity 2.4 has achieved better recall value (Figure-3).

5. CONCLUSION

The Boolean classifier is a predictive classifier that is used in this research work to predict the class label of unknown data. Accuracy defines how well the model suits in assigning the correct labels to unseen data. The classification model proposed in the current study is a logical statement that describes relation between dependent and independent variables of the defect dataset. The study obtained a remarkable improvement in proposed model classification accuracy, on average 3-6% increase in accuracy, compared to the other classifiers used in the available literature. Jedit 4.0, Jedit 4.2, Lucene 2.0, Lucene 2.2, Velocity 1.6, Xalan 2.5, Xalan 2.6, Xerces 1.3 are the systems for which proposed Boolean classifier have been tested and authors found better results in classification accuracy over other classifiers. The study also identified few systems for which proposed model has not shown very promising results. The results could be improved with the inclusion of few external factors like development environment, experience of development teams etc. In this research authors have also considered the performance measure like precision and recall for better performance analysis of the proposed classifier. The precision values for proposed Boolean classifier are also found satisfactory in most of classification results, which better chances of classification acceptances.

REFERENCES

[1] P. Kapoor, D. Arora, Ashwani. 2017. An Approach for Improving Classification Accuracy using Discretized Software Defect Data, In proc of Springer's International Conference ICACNI 2017.

[2] S.R. Chidamber, C.F. Kemerer. 1994. A metrics suite for Object Oriented Design. IEEE Transactions on Software Engineering. 20(6): 476-493.

[3] Kapoor P., Arora D., Kumar A. 2017. Effects of Mean Metric Value over CK Metrics Distribution towards Improved Software Fault Predictions. In: Bhatia S., Mishra K., Tiwari S., Singh V. (eds) Advances in Computer and Computational Sciences. Advances in Intelligent Systems and Computing. Vol. 553. Springer, Singapore.

[4] Pawlak, Zdzisław, and AndrzejSkowron. 2007. Rough sets and Boolean reasoning. Information sciences. 177(1): 41-73.

[5] C. R. Edwards. 1975. The Application of the Rademacher-Walsh Transform to Boolean Function Classification and Threshold Logic Synthesis. in IEEE Transactions on Computers. C-24(1): 48-62.

[6] Bennett, R.G. 1982. Analysis of Reliability Block Diagrams by Boolean Techniques. Reliability. IEEE Transactions on. R-31. 159-166.

[7] Hassan Abdul Wahab and Ahmad Kamal Hassan. 2016. A Karnaugh Map Based Approach towards Systemic Reviews and Meta-Analysis. Springer Plus. 5: 371. PMC. Web. 7 September.

[8] Muller David E. 1954. Application of Boolean algebra to switching circuit design and to error detection. Transactions of the IRE Professional Group on Electronic Computers. 3: 6-12.

[9] 1955. Application of Boolean Algebra to Switching Circuit Design and to Error Detection by D. E. Muller Review by: Raymond J. Nelson The Journal of Symbolic Logic. 20(2): 195 Published by: Association for Symbolic Logic Stable.

[10] T. M. Cover. 1965. Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition. in IEEE Transactions on Electronic Computers. EC-14(3): 326-334.

[11] S. Golomb. 1959. On the Classification of Boolean Functions. in IRE Transactions on Circuit Theory. 6(5): 176-186.

[12] Tabandeh M. 2012. Application of Karnaugh map for easy generation of error correcting codes. Scientia Iranica. 19(3): 690-695.

[13] Ting Kai Ming. 2011. Precision and recall. Encyclopedia of machine learning. Springer US. pp. 781-781.