Software Qual J (2010) 18:3–35
DOI 10.1007/s11219-009-9079-6

Empirical validation of object-oriented metrics for predicting fault proneness models

Yogesh Singh · Arvinder Kaur · Ruchika Malhotra

Published online: 1 July 2009

Abstract Empirical validation of software metrics used to predict software quality attributes is important to ensure their practical relevance in software organizations. The aim of this work is to find the relation of object-oriented (OO) metrics with fault proneness at different severity levels of faults. For this purpose, different prediction models have been developed using regression and machine learning methods. We evaluate and compare the performance of these methods to find which method performs better at different severity levels of faults and empirically validate OO metrics given by Chidamber and Kemerer. The results of the empirical study are based on public domain NASA data set. The performance of the predicted models was evaluated using Receiver Operating Characteristic (ROC) analysis. The results show that the area under the curve (measured from the ROC analysis) of models predicted using high severity faults is low as compared with the area under the curve of the model predicted with respect to medium and low severity faults. However, the number of faults in the classes correctly classified by predicted models with respect to high severity faults is not low. This study also shows that the performance of machine learning methods is better than logistic regression method with respect to all the severities of faults. Based on the results, it is reasonable to claim that models targeted at different severity levels of faults could help for planning and executing testing by focusing resources on fault-prone parts of the design and code that are likely to cause serious failures.

Keywords Metrics · Object-oriented · Software quality · Empirical validation · Fault prediction · Receiver operating characteristics analysis

Y. Singh · A. Kaur · R. Malhotra (✉)
University School of Information Technology, GGS Indraprastha University, Delhi 110403, India
e-mail: ruchikamalhotra2004@yahoo.com

Y. Singh
e-mail: ys66@rediffmail.com

A. Kaur
e-mail: arvinderkaurtakkar@yahoo.com

1 Introduction

As the complexity and the constraints under which software is developed are increasing, it is usual to produce software with severe faults. One way to deal with this problem is to predict important quality attributes such as fault-proneness, effort, productivity, amount of maintenance modifications and reliability during early phases of software development. Software metrics (Aggarwal et al. 2005; Briand et al. 1998, 1999; Bieman and Kang 1995; Cartwright and Shepperd 1999; Chidamber and Kamerer 1991, 1994; Harrison et al. 1998; Henderson-Sellers 1996; Hitz and Montazeri 1995; Lake and Cook 1994; Li and Henry 1993; Lee et al. 1995; Lorenz and Kidd 1994; Tegarden et al. 1995) can be used in predicting these quality attributes. However, software faults vary considerably with respect to their severity. The value of severity quantifies the impact of the fault on the software operation. The IEEE standard states, "Identifying the severity of an anomaly is a mandatory category as is identifying the project schedule, and project cost impacts of any possible solution for the anomaly" (IEEE Std. 1044-1993 1994). A failure caused by a fault may lead to a whole system crash or an inability to open a file (El Emam et al. 1999; Aggarwal et al. 2009). The former failure is more severe than the latter; thus, not all types of failures are the same. Lack of determination of severity of faults was one of the main criticisms of the approaches to fault prediction in the study by Fenton and Neil (Fenton and Neil 1999). Therefore, there is a need to develop prediction models that can be used to identify classes that are prone to have serious faults. The software practitioners can use the model predicted with respect to high severity of faults to focus the testing on those parts of the system that are likely to cause serious failures. Empirical studies have been carried out in the past of fault proneness models (Basili et al. 1996; Binkley and Schach 1998; Briand et al. 2000; Cartwright and Shepperd 1999; Chidamber et al. 1998; Harrison et al. 1998; Li and Henry 1993; Yu et al. 2002; Khoshgoftaar et al. 2002; Phadke and Allen 2005; Zhou and Leung 2006; Aggarwal et al. 2006b; Olague et al. 2007; Pai 2007; Menzies et al. 2007; Aggarwal et al. 2009). Though these studies have raised the need to study fault proneness models at different severity levels of faults, yet less work has been done in this important area.

Several methods such as linear and logistic regression methods and machine learning methods [such as decision tree (DT) and Artificial Neural Network (ANN)] have been proposed in the literature. There are few studies of machine learning methods for fault prediction using OO metrics. Most of the prediction models in the literature are built using statistical methods. There are many machine learning methods and there is a need to compare the results of various machine learning methods as they give different results. ANN and DT methods have seen an explosion of interest over the years, and are being successfully applied across a range of problem domains such as finance, medicine, engineering, geology, and physics. Indeed, these methods are being introduced to solve the problems of prediction, classification, or control (Duman 2006; Porter and Selby 1990; Dreiseitl and Ohno-Machado 2002; Eftekhar et al. 2005; Marini et al. 2008). It is natural for software practitioners and potential users to wonder, "Which classification technique is best?", or more realistically, "What methods tend to work well for a given type of data set?" More data based empirical studies, which are capable of being verified by observation or experiment are needed. The evidence gathered through these empirical studies is today considered to be the most powerful support possible for testing a given hypothesis (Aggarwal et al. 2009). Hence, conducting empirical studies to compare regression and machine learning methods is necessary to build an adequate body of knowledge in order to draw strong conclusions leading to widely accepted and well-formed theories.

Thus, there is a need for both (1) building fault proneness models at different severity levels of faults and (2) empirically comparing the results of regression and machine learning methods. Now we briefly describe the work done in this study. In this paper, we investigate the following issues:

- How accurately and precisely do the OO metrics predict high severity faults?
- Which OO metrics are related to fault proneness of classes with regard to high severity faults?
- How accurately and precisely do the OO metrics predict medium severity faults?
- Which OO metrics are related to fault proneness of classes with regard to medium severity faults?
- How accurately and precisely do the OO metrics predict low severity faulty?
- Which OO metrics are related to fault proneness of classes with regard to low severity faults?
- Is the performance of machine learning methods better than logistic regression methods?

In this work, we investigate the performance of the fault proneness predictions using the OO design metrics suite given by Chidamber and Kemerer (1994), with particular focus on how accurately these metrics predict fault proneness at different severity levels of faults. In the study conducted by Zhou and Leung (2006), faults were only categorized into two severity levels: high and low, in order to simplify the process of fault-categorization. All failures are not the same and failures should be categorized in the same severity level only if they have the same impact (Laird and Brennan 2006). We need to differentiate the different types of failures by defining the appropriate severity levels of faults (Laird and Brennan 2006). The models constructed by taking all the severity levels given in the data set (NASA 2004) would represent real life fault severity as given in the impact classification scheme in IEEE Std. 1044-1993 (1994) and Lovin and Yaptangco (2006). Detailed classification and analysis of fault data can help the software practitioner obtain more detailed information regarding potentially problematic areas to provide more specific and more valuable feedback to the software development process for focused quality improvement (Tian 2005). Hence, in this study we have categorized faults with respect to all the severity levels given in the NASA data set in order to improve the effectiveness of the categorization and provide meaningful, correct and detailed analysis of fault data.

First the classes predicted faulty with respect to high severity faults will be tested. Then the classes predicted as faulty with respect to medium severity faults will be tested and this process will continue while the resources are available. However, in the case of faults categorized into only two severity levels the classes with faults at medium and low severity level are treated as having equal level of severity which is incorrect as the classes with medium severity level have more severe faults than the classes at low severity level (Laird and Brennan, 2006; Bazman 2006). Hence, classes with higher severity level of faults should be tested and fixed before testing and fixing the classes at lower severity level of faults (Horch 2003). Categorizing the faults according to different severity levels helps prioritize the fixing of faults (Afzal 2007). The software practitioners can select from the list of prioritized classes by using models predicted at high, medium, and low severity levels of faults, as available resources will allow. This would help software engineers narrow down the testing effort to focus their resources on different severity levels of faults. The investigation of models at multiple severity levels will allow the software practitioners to reduce and assess the failures of a specified severity before deploying the software. Further, the NASA data set categorizes faults into different severity levels and combining faults with medium and low severity into one level will not be a true representation of categories of fault severity given in the data set. The detailed analysis performed would also provide a comparative analysis of models predicted with respect to high, medium, and low severity levels of faults.

Hence, we categorized faults into three levels: high severity, medium severity and low severity. The performance of logistic regression and two machine-learning methods (DT and ANN) was evaluated in the study for predicting the fault proneness in the classes. The validation of the methods is carried out using receiver operating characteristic (ROC) analysis. To obtain a balance between the number of classes predicted as fault prone, and number of classes predicted as not fault prone, we use ROC curves. We also analyze the performance of the models by calculating the area under the curve from an ROC curve (El Emam et al. 1999). In order to perform the analysis we validate the performance of these methods using public domain KC1 NASA data set (NASA 2004) The data set is available on www.mdp.ivv.nasa.gov. The 145 classes in this data were developed using C++ language.

The paper is organized as follows: The related work is summarized in Sect. 2. Section 3 summarizes the metrics studied and gives hypothesis to be tested in the study. Section 4 describes sources from which data is collected and presents the research methodology for detection of outliers. The results of the study are given in Sect. 5 and the model is evaluated in Sect. 6. Section 7 presents threats to validity of the models and the conclusions of the research are presented in Sect. 8.

2 Related work

Based on a study of eight medium-sized systems, developed by students it was (Basili et al. 1996) found that several of the (Chidamber and Kemerer 1994) metrics were associated with fault proneness.

Tang et al. (1999) analyzed (Chidamber and Kemerer 1994) OO metrics suite on three industrial applications developed in C++. They found none of the metrics examined to be significant except RFC and WMC. El Emam et al. (2001) examined a large telecommunication application developed in C++ and found that class size i.e. SLOC has a confounding effect of most OO metrics on faults.

Briand et al. (2000) have extracted 49 metrics to identify a suitable model for predicting fault proneness of classes. The system under investigation was a medium sized C++ software system developed by undergraduate/graduate students. The eight systems under study consisted of a total of 180 classes. They used univariate and multivariate analysis to find the individual and combined impact of OO metrics and fault proneness. The results showed all metrics except NOC (which was found related to fault proneness in an inverse manner) to be significant predictors of fault proneness. Another study by Briand et al. (2001) used a commercial system consisting of 83 classes. They found the DIT metric related to fault proneness in an inverse manner and the NOC metric to be an insignificant predictor of fault proneness.

Yu et al. (2002) chose eight metrics and they examined the relationship between these metrics and the fault-proneness. The subject system was the client side of a large network service management system developed by three professional software engineers. It was written in Java, and consisted of 123 classes and around 34,000 lines of code. First, they examined the correlation among the metrics and found four highly correlated subsets. Then, they used univariate analysis to find out which metrics could detect faults and which could not. They found that metrics (CBOin, RFCin, and DIT) were not significant while the other metrics were significant predictors but to a different extent.

Gyimothy et al. (2005) empirically validated (Chidamber and Kemerer 1994) metrics on open source software for fault prediction. They employed regression (linear and logistic regression) and machine learning methods (neural network and decision tree) for model prediction. The results indicated that NOC was not a significant predictor of fault proneness but all the other metrics were found significant in LR analysis.

Olague et al. (2007) validated OO metrics on versions of open source agile software. They found WMC, CBO, RFC, and LCOM metrics to be very significant, while DIT was found insignificant in two versions of the system. The NOC metric was found to be insignificant in one version while less significant in the other two versions.

Zhou and Leung (2006) validated the same data set as ours to predict fault proneness models with respect to two categories of faults: high and low. They categorized faults with severity rating 1 as high severity faults and faults with other severity levels as low severity faults. They did not consider the faults, which originated from design and COTS/OS. Our approach differs from Zhou and Leung as we categorized faults into three severity levels: high, medium, and low severity of faults. The medium severity level of faults is more severe than low severity level of faults. Hence, the classes having faults of medium severity level must be given more attention compared with the classes with low severity level of faults. In the study conducted by Zhou and Leung, the classes were not categorized into medium and low severity level of faults. Further, the faults produced from the design were not taken into account. We also analyzed two different machine learning methods (ANN and DT) for predicting fault proneness models and evaluated the performance of these models using ROC analysis. Pai (2007) used the same data set using a Bayesian approach to find the relationship between software product metrics to fault content and fault proneness. They did not categorize faults at different severity levels and mentioned that a natural extension to their analysis is severity classification using Bayesian network models hence their work is not comparable with ours.

3 The variables and research hypothesis

In this section, we present the summary of the metrics studied in this paper (Sect. 3.1), and hypotheses to be tested in our work (Sect. 3.2).

3.1 Dependent and independent variables

The binary dependent variable in our study is fault proneness. The goal of our study is to explore empirically the relationship between OO metrics and fault proneness at the class level. Fault proneness is defined as the probability of fault detection in a class (Briand et al. 2000; Pai 2007; Aggarwal et al. 2009). We use logistic regression and machine learning methods, which are based on predicting probabilities. Our dependent variable will be predicted, based on the faults, found during the software development life cycle. The metrics given by Chidamber and Kemerer (1994) are summarized in Table 1. These metrics are explained with practical applications in Aggarwal et al. (2005, 2006a).

3.2 Hypotheses

In this section, research hypotheses are presented.

Table 1 Metrics studied (Chidamber and Kemerer 1994)

Metric	Definition
Coupling between objects (CBO)	CBO for a class is a count of the number of other classes to which it is coupled and vice versa
Response for a class (RFC)	A count of methods implemented within a class and the number of methods accessible to an object class due to inheritance
Lack of cohesion (LCOM)	For each data field in a class, the percentage of the methods in the class using that data field; the percentages are averaged then subtracted from 100%
Number of children (NOC)	The NOC is the number of immediate subclasses of a class in a hierarchy
Depth of inheritance (DIT)	The depth of a class within the inheritance hierarchy is the maximum number of steps from the class node to the root of the tree and is measured by the number of ancestor classes
Weighted methods per class (WMC)	The WMC is a count of sum of complexities of all methods in a class
Source lines of code (SLOC)	The number of lines of code

3.2.1 Hypothesis set A

We tested the hypotheses given below to find the individual effect of each OO metric on fault proneness at different severity levels of faults.

CBO Hypothesis: A class with high import or export coupling is more likely to be fault-prone than a class with less import or export coupling. (Null hypothesis: A class with high import or export coupling is less likely to be fault-prone than a class with high import or export coupling).

RFC Hypothesis: A class with a high number of methods implemented within a class and the number of methods accessible to an object class due to inheritance is more likely to be fault-prone than a class with a low number of methods implemented within a class and the number of methods accessible to an object class due to inheritance. (Null hypothesis: A class with a high number of methods implemented within a class and the number of methods accessible to an object class due to inheritance is less likely to be fault-prone than a class with a low number of methods implemented within a class and the number of methods accessible to an object class due to inheritance).

LCOM Hypothesis: A class with less cohesion is more likely to be fault-prone than a class with high cohesion. (Null hypothesis: A class with less cohesion is less likely to be fault-prone than a class with high cohesion).

NOC Hypothesis: A class with a greater number of descendants is more likely to be fault-prone than a class with fewer descendants. (Null hypothesis: A class with more descendants is less likely to be fault-prone than a class with fewer descendants).

DIT Hypothesis: A class with a large depth in inheritance tree is more likely to be fault-prone than a class with a small depth in inheritance tree. (Null hypothesis: A class with a large depth in inheritance tree is less likely to be fault-prone than a class with a small depth in inheritance tree).

WMC Hypothesis: A class with a large number of methods weighted by complexities is more likely to be fault-prone than a class with a fewer number of methods weighted by complexities. (Null hypothesis: A class with a large number of methods weighted by complexities is less likely to be fault-prone than a class with a fewer number of methods weighted by complexities).

SLOC Hypothesis: A class with a larger size i.e., more information is more likely to be fault-prone than a class with a smaller size. (Null hypothesis: A class with a larger size i.e., more information is less likely to be fault-prone than a class with a smaller size).

3.2.2 Hypothesis set B

We tested the hypotheses given below to compare the performance of regression and machine learning methods at different severity levels of faults:

H1 hypothesis LR models do not outperform models predicted using DT. (Null Hypothesis: LR models do outperform models predicted using DT).

H2 hypothesis LR models do not outperform models predicted using ANN. (Null Hypothesis: LR models do outperform models predicted using ANN).

H3 hypothesis ANN models do not outperform models predicted using DT. (Null Hypothesis: ANN models do outperform models predicted using DT).

4 Empirical data collection and outlier analysis

This study makes use of the public domain data set KC1 from the NASA Metrics Data Program (NASA 2004, PROMISE). The NASA data repository stores the data, which is collected and validated by the Metrics Data Program (MDP 2006, http://sarpresults.ivv.nasa.gov/ViewResearch/107.jsp). The data in KC1 was collected from a storage management system for receiving/processing ground data, which was implemented in the C++ programming language. Fault data for KC1 was collected since the beginning of the project (storage management system) but that data can only be associated back by 5 years (MDP 2006, http://sarpresults.ivv.nasa.gov/ViewResearch/107.jsp). This system consists of 145 classes that comprises 2,107 methods, with 40 K lines of code. KC1 provides both class-level and method-level static metrics. At the method level, 21 software product metrics based on product's complexity, size and vocabulary are given. At the class level, values of 10 metrics are computed including six metrics given by Chidamber and Kemerer (1994). The seven OO metrics are taken in our study (see Sect. 3.1) for analyses. In KC1, six files provide association between class/method and metric/defect data. In particular, there are four files of interest, the first representing the association between classes and methods, the second representing association between methods and defects, the third representing association between defects and severity of faults and the fourth representing association between defects and specific reasons for closure of the error report.

We first associate defects with each class according to their severities. The value of severity quantifies the impact of the defect on the overall environment with 1 being most severe to 5 being least severe as decided in data set KC1. The defect data from KC1 is collected from information contained in error reports. An error either could be from the source code, COTS/OS, design or is actually not a fault. We take into account defects produced from the source code, COTS/OS and design. We further processed the data by removing all the faults that had "not a fault" keyword used as the reason of closure of error report. This reduced the number of faults from 669 to 642. Out of 145 classes, 59 were faulty classes i.e. classes with at least one fault and the rest were non-faulty.

Table 2 Distribution of faults and faulty classes at high, medium and low severity levels

Level of severity	Number faulty of classes	% Of faulty classes	Number of faults	% Of distribution of faults
High	23	15.56	48	7.47
Medium	58	40.00	449	69.93
Low	39	26.90	145	22.59

In this study, we categorized the faults as high, medium, or low severity. Faults with severity rating 1 were classified as high severity faults. Faults with severity rating 2 were classified as medium severity faults and faults with severity rating 3, 4, 5 as low severity faults as at severity rating 4 no class was found to be faulty and at severity rating 5 only one class was faulty. Faults at severity rating 1 require immediate correction for the system to continue to operate properly (Zhou and Leung 2006).

Table 2 summarizes the distribution of faults and faulty classes at high, medium, and low severity levels in the KC1 NASA data set after preprocessing of faults in the data set. High severity faults were distributed in 23 classes (15.56%). There were 48 high severity faults (7.47%), 449 medium severity faults (69.93%) and 145 low severity faults (22.59%). As shown in Table 2, the majority of the classes are faulty at severity rating medium (58 out of 59 faulty classes). In Fig. 1a, b, and c show the distribution of high severity faults, medium severity faults, and low severity faults. It can be seen from Fig. 1a that 22.92% of classes with high severity faults contain one fault, 29.17% of classes contain two faults and so on. In addition, the maximum number of faults (449 out of 642) are covered at medium severity (see Fig. 1b).

Data points, which are located in an empty part of the sample space, are called outliers. Outlier analysis is done to find data points that are unusual and removing them may be important. Univariate and multivariate outliers were found in our study. To identify multivariate outliers, we calculate the Mahalanobis Jackknife distance for each data point . Mahalanobis Jackknife is a measure of the distance in multidimensional space of each observation from the mean center of the observations (Aggarwal et al. 2009; Hair and Anderson 2006). Details on outlier analysis can be found in Belsley et al. (1980) and Barnett and Price (1995). The input metrics were normalized using min–max normalization. Min–max normalization performs a linear transformation on the original data

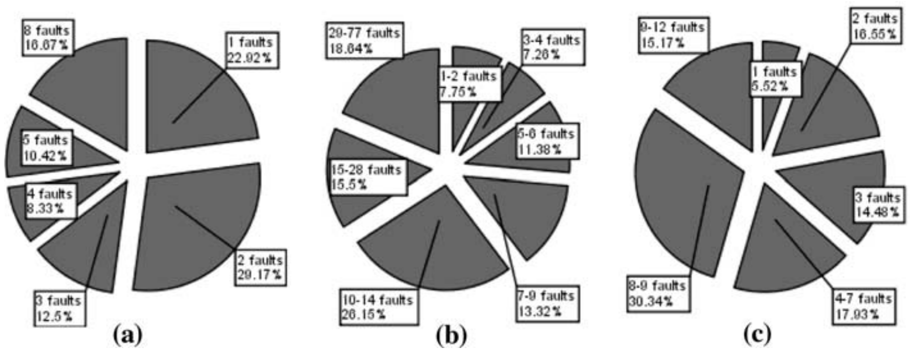

Fig. 1 Distribution of high, medium, and low severity faults in the classes **a** high **b** medium **c** low severity faults

(Han and Kamber 2001). Suppose that minA and maxA are the minimum and maximum values of an attribute A. It maps value v of A to v' in the range 0–1 using the formula:

$$v' = \frac{v - \min A}{\max A - \min A} \tag{1}$$

5 Analysis results

In this section, we described the analyses performed to find the relationship between OO metrics and fault proneness of the classes. We employed both the univariate and multivariate logistic regression analysis. The univariate analysis is used to find the individual effect of each metric on fault proneness. The multivariate analysis is used to find the combined effect of metrics on fault proneness. Besides these techniques, we also applied two machine-learning techniques to find the effect of OO metrics on fault proneness of the classes. We employed the DT and ANN methods. These methods are rarely applied to this area. Both the methods can be used to predict the individual effect of each metric on fault proneness (same as univariate analyses) and the combined effect of metrics on fault proneness (same as multivariate analyses). The models predicted were applied to 145 classes. The following measures are used to evaluate the performance of each predicted fault proneness model in all the above sub sections:

- The *sensitivity and specificity* of the model are calculated to predict the correctness of the model. The percentage of classes correctly predicted to be fault prone is known as the sensitivity of the model. The percentage of non-occurrences correctly predicted i.e. classes predicted not to be fault prone is called specificity of the model. Ideally, both the sensitivity and specificity should be high to predict faulty and non faulty classes.
- *Completeness*: Completeness is defined as the number of faults in classes classified fault-prone, divided by the total number of faults in the system.
- *Precision*: Precision is defined as the number of classes that are predicted correctly, divided by the total number of classes.
- *Receiver Operating Characteristic* (*ROC*) *analysis*: The performance of the outputs of the predicted models were evaluated using ROC analysis. The ROC curve, which is defined as a plot of sensitivity on the y-coordinate versus its 1-specificity on the x coordinate, is an effective method of evaluating the quality or performance of predicted models (El Emam et al. 1999). While constructing ROC curves, we selected many cutoff points between 0 and 1, and calculated sensitivity and specificity at each cut off point. The optimal choice of the cutoff point (that maximizes both sensitivity and specificity) can be selected from the ROC curve (El Emam et al. 1999). Hence, by using the ROC curve one can easily determine optimal cutoff point for a predicted model.
- In order to predict the accuracy of the model it should be applied to different data sets. We therefore performed a k-cross validation of the models (Stone 1974). The data set is randomly divided into k subsets. Each time one of the k subsets is used as the test set and the other k-1 subsets are used to form a training set. Therefore, we get the fault proneness for all the k classes.

5.1 Descriptive statistics and correlation analysis

Each table presented in the following subsections show “min”, “max”, “mean”, “median”, “std dev”, “25% quartile” and “75% quartile” for all metrics considered in this study.

Table 3 Descriptive statistics for metrics

Metric	Min.	Max.	Mean	Median	SD	Percentile (25%)	Percentile (75%)
CBO	0	24	8.32	8	6.38	3	14
LCOM	0	100	68.72	84	36.89	56.5	96
NOC	0	5	0.21	0	0.7	0	0
RFC	0	222	34.38	28	36.2	10	44.5
WMC	0	100	17.42	12	17.45	8	22
LOC	0	2,313	211.25	108	345.55	8	235.5
DIT	0	6	1	1	1.26	0	1.5

Table 4 Correlations among metrics

Metric	CBO	LCOM	NOC	RFC	WMC	LOC	DIT
CBO	1						
LCOM	0.256	1					
NOC	−0.03	−0.028	1				
RFC	0.386	0.334	−0.049	1			
WMC	0.245	0.318	0.035	**0.628**	1		
LOC	**0.572**	0.238	−0.039	**0.508**	**0.624**	1	
DIT	0.4692	0.256	−0.031	**0.654**	0.136	0.345	1

The following observations are made from Table 3:

- The size of a class measured in terms of lines of source code ranges from 0 to 2,313.
- The values of DIT and NOC are low in the system, which shows that inheritance is not much used in all the systems; similar results have also been shown by others (Chidamber et al. 1998; Briand et al. 2000; Cartwright and Shepperd 1999).
- The LCOM measure, which counts the number of classes with no attribute usage in common, has high values (upto 100) in KCl data set.

We calculated the correlation among metrics, which is an important static quantity as shown in Table 4. Gyimothy et al. (2005) and Basili et al. (1996) also calculated the correlation among metrics. Hopkins calls a correlation coefficient value between 0.5–0.7 large, 0.7–0.9 very large, and 0.9–1.0 almost perfect (Hopkins 2003). Thus, in Table 4, the correlated values with correlation coefficient greater than 0.5 are shown in bold. The correlation coefficients shown in bold are significant at 0.01 level. In this data set, WMC, LOC, DIT metrics are correlated with RFC metric. Similarly, the WMC and CBO metrics are correlated with LOC metric. Therefore, it shows that these metrics are not totally independent and represents redundant information.

5.2 Logistic regression (LR) analyses

In this section we present the research methodology followed (Sect. 5.2.1), results of univariate analysis (Sect. 5.2.2) and results of multivariate analysis (Sect. 5.2.3).

5.2.1 Research methodology

Logistic Regression (LR) is used to predict the dependent variable (fault proneness) from a set of independent variables (OO metrics) to determine the percent of variance in the dependent variable explained by the independent variable [a detailed description is given by Basili et al. (1996), Hosmer and Lemeshow (1989) and Aggarwal et al. (2009)]. LR is of two types (Hosmer and Lemeshow 1989): (a) Univariate LR; (b) Multivariate LR. Univariate LR is a statistical method for dealing with the formulation of the mathematical model depicting relationship among variables, which can be used for the values of each independent variable. This technique is used to test the hypotheses given earlier.

Multivariate LR is done to construct a prediction model for the fault proneness of classes. In this method, metrics are used in combination. The multivariate LR formula can be defined as (Aggarwal et al. 2009):

$$\text{prob}(X_1, X_2, \ldots, X_n = \frac{e^{(A_0 + A_1X_1 + \cdots + A_nX_n)}}{1 + e^{(A_0 + A_1X_1 + \cdots + A_nX_n)}} \tag{2}$$

where X_i, $i = 1, 2, \ldots, n$ are the independent variables. *Prob* is the probability of detecting faults in a class.

In LR, two stepwise selection methods—forward selection and backward elimination—are used (Hosmer and Lemeshow 1989). The forward stepwise procedure examines the variables that are selected one at a time for entry at each step. The backward elimination method includes all the independent variables in the model. Variables are deleted one at a time from the model until a stopping criteria is fulfilled.

Multicollinearity refers to the degree to which any variable effect can be predicted by the other variables in the analysis. As multicollinearity rises, the ability to define any variable's effect is diminished. Thus, the interpretation of the model becomes difficult, as the impact of individual variables on the dependent variable can no longer be judged independently from the other variables (Aggarwal et al. 2009; Belsley et al. 1980). Thus, a test of multicollinearity was performed on the fault proneness model predicted in Sect. 5.2.2. Let X_1, X_2, …, X_n be the covariates of the model predicted. The Principal-Component Method (or PC method) is a standard technique used to find the interdependence, among a set of variables. The factors summarize the commonality of these variables and factor loadings represent the correlation between the variables and the factor. The PC method maximizes the sum of squared loadings of each factor extracted in turn (Aggarwal et al. 2009). The PC method is applied to these variables to find the maximum eigenvalue, $e_{\max}$ and minimum eigenvalue, $e_{\min}$. The conditional number is defined as$\lambda = \sqrt{e\max/e\min}$. If the value of the conditional number exceeds 30 then multicollinearity is not tolerable (Belsley et al. 1980).

The following statistics are reported for each significant metric:

- *Odds ratio*: The odds ratio is the probability of the event divided by the probability of the nonevent. The event in this study has a fault and the nonevent is the probability of not having a fault. An odds ratio with a value of two means that the dependent variable is multiplied by two when the independent variable increases by one unit.
- *Maximum likelihood estimation* (*MLE*) *and coefficients* (A_i*s*): MLE is a statistical method for estimating the coefficients of a model. The likelihood function measures the probability of observing the set of dependent variable values. MLE involves finding the coefficients that make the log of the likelihood function as large as possible. The larger

the value of the coefficients, the larger the impact of the independent variables (OO metrics) on the predicted fault proneness (Aggarwal et al. 2009).

- *The statistical significance*: Statistical significance measures the significance level of the coefficient. The larger the statistical significance the lower the estimated impact of the independent variables (OO metrics).
- *The R^2 Statistic*: It is the proportion of the variance in the dependent variable that is explained by the variance of the independent variables. The higher is the value of R^2, the higher is the effect of the independent variables and more is the accuracy of the model. However, as stated in (Zhou and Leung 2006; Briand et al. 2000), "we should not interpret the value of R^2 in logistic regression using the usual heuristics for linear regression R^2s since they are built upon very different formulas". As a result, high R^2s are rare in logistic regression.

In our analysis, if more than one fault is found in a given class, each fault detection event is treated as a separate observation. The number of faults are taken into account when we construct our logistic regression models. Details of LR analysis can be found in Aggarwal et al. (2009).

5.2.2 Univariate analysis results

We treated a class as faulty if it contained at least one fault. Tables 5, 6, 7, 8 provides the coefficient (B), standard error (SE), statistical significance (sig), odds ratio (exp(B)), and R^2 statistic for each measure. The metrics with a significant relationship to fault proneness, that is, below or at the significance (named as Sig. in Tables 5, 6, 7, 8) threshold of 0.01 are shown in bold (see Tables 5, 6, 7, 8). Table 5 presents the results of univariate analysis for predicting fault proneness with respect to High Severity Faults (HSF). From Table 5, we can see that five out of seven metrics were found to be very significant (Sig. < 0.01). However, NOC and DIT metrics were not found to be significant. The LCOM metric was significant at 0.05 significance level. The value of R^2 statistic was highest for SLOC and CBO metrics.

Table 6 summarizes the results of univariate analysis for predicting fault proneness with respect to Medium Severity Faults (MSF). Table 6 shows the values of R^2 statistic is the highest for the SLOC metric. All the metrics except DIT were found to be significant. NOC has a negative coefficient, which implies that classes with higher NOC value are less fault-prone.

Table 7 summarizes the results of univariate analysis for predicting fault proneness with respect to Low Severity Faults (LSF). Again, it can be seen from Table 7 that the value of R^2 statistic is highest for the SLOC metric. The results show that four out of seven metrics were found to be very significant. The LCOM, NOC and DIT metrics were not found to be significant.

Table 5 Univariate analysis using LR method for HSF

Metric	B	SE	Sig.	Exp (B)	R^2
CBO	0.145	0.028	**0.0001**	1.156	0.263
WMC	0.037	0.011	**0.0001**	1.038	0.180
RFC	0.016	0.004	**0.0001**	1.016	0.160
SLOC	0.003	0.001	**0.0001**	1.003	0.268
LCOM	0.015	0.006	**0.0170**	1.015	0.100
NOC	−18.256	5,903.250	0.9980	0.000	0.060
DIT	0.036	0.134	0.7840	1.037	0.001

Table 6 Univariate analysis using LR method for MSF

Metric	*B*	SE	Sig.	Exp (*B*)	R^2
CBO	0.276	0.030	**0.0001**	1.318	0.375
WMC	0.065	0.011	**0.0001**	1.067	0.215
RFC	0.025	0.004	**0.0001**	1.026	0.196
SLOC	0.010	0.001	**0.0001**	1.110	0.392
LCOM	0.009	0.003	**0.0050**	1.009	0.116
NOC	−1.589	0.393	**0.0001**	0.204	0.090
DIT	0.058	0.092	0.5280	1.060	0.001

Table 7 Univariate analysis using LR method for LSF

Metric	*B*	SE	Sig.	Exp (*B*)	R^2
CBO	0.175	0.025	**0.0001**	1.191	0.290
WMC	0.050	0.011	**0.0001**	1.052	0.205
RFC	0.015	0.004	**0.0001**	1.015	0.140
SLOC	0.004	0.001	**0.0001**	1.004	0.338
LCOM	0.004	0.003	0.2720	1.004	0.001
NOC	−0.235	0.192	0.2200	0.790	0.002
DIT	0.148	0.099	0.1340	1.160	0.005

Table 8 Univariate analysis using LR method for USF

Metric	*B*	SE	Sig.	Exp (*B*)	R^2
CBO	0.274	0.029	**0.0001**	1.315	0.336
WMC	0.068	0.019	**0.0001**	1.065	0.186
RFC	0.023	0.004	**0.0001**	1.024	0.127
SLOC	0.011	0.002	**0.0001**	1.011	0.389
LCOM	0.008	0.003	**0.0100**	1.008	0.013
NOC	−0.674	0.185	**0.0001**	0.510	0.104
DIT	0.086	0.091	0.3450	1.089	0.001

Table 8 summarizes the results of univariate analysis for predicting fault proneness. The results show that six out of seven metrics were found to be very significant when the faults were not categorized according to their severity i.e. Ungraded Severity Faults (USF). The DIT metric was not found to be significant and the NOC metric has a negative coefficient. This shows that the NOC metric is related to fault proneness but in an inverse manner.

Thus, the SLOC metric has the highest R^2 value for all the severity of faults, which shows that it is the best predictor. The CBO metric has the second highest R^2 value. The values of the R^2 statistic are more important as compared to the value of Sig. as they show the strength of the correlation.

5.2.3 Multivariate LR results

In this section, we summarize the results obtained from four multivariate fault prediction models using the LR method. The first one is for high severity faults, the second one is for medium severity faults, the third one is for low severity faults and the fourth one is for ungraded severity faults. The multivariate analysis is used to find the combined effect of OO metrics on fault proneness. We attempted to use the backward elimination method. However, the results obtained were poorer (i.e. the values of R^2 and log likelihood statistic were low) than for the model obtained from the forward stepwise procedure. We therefore used the forward stepwise procedure in this study. The conditional number is below 30 for all models predicted at high, low, medium, and ungraded faults. This implies that the multicollinearity of the predicted model is tolerable (see Sect. 5.2.1). Tables 9, 12, 14, and 17 provide the coefficient (B), standard error (SE), statistical significance (sig), odds ratio (exp(B)) for metrics included in the model. Tables 10, 13, 15, and 18 shows the sensitivity, specificity, precision, completeness, and cutoff point of each metric and the model predicted.

High Severity Fault (*HSF*) *prediction model*: One outlier was found influential and was removed from the analysis. In the univariate analysis, five out of seven metrics (except NOC and DIT) were found related to fault proneness. Table 9 shows that two metrics SLOC and CBO were included in the multivariate model.

Since, we did not wish to select an arbitrary cutoff value and in order to obtain a balance between the number of classes predicted as fault prone and not fault prone, the cutoff point of the model is computed using the ROC analysis method (Hanley and McNeil 1982). The SLOC metric has the highest sensitivity (65.2%) and completeness (75%) values. The CBO (60.9% sensitivity) and RFC (60.4% sensitivity) metrics have the next highest values of sensitivity, but the completeness of the RFC (60.4%) is greater than the CBO (58.3%) metric. However, in case of the DIT and NOC metrics all the classes were predicted to be non faulty; hence the results are not shown. Thus testing all the classes will be a large waste of the testing resources. As shown in Table 10, the cut off point for the model predicted is 0.22. The value of R^2 of the predicted model is 0.301.

Table 11a presents the classification results of the fault proneness model predicted using HSF. The sensitivity of the model is 69.5%, specificity of the model is 73.7%, and the completeness of the model is 79.16%.

Medium Severity Fault (*MSF*) *prediction model*: In the univariate analysis, six out of seven metrics (except DIT) were found related to fault proneness. Table 12 shows that four metrics SLOC, CBO, LCOM, and NOC were included in the multivariate model. The sign of the coefficient of NOC metric is negative. The same results were found in the univariate analysis. The sign of the coefficient of the LCOM metric is also negative, though it was positive in the univariate analysis. This is due to suppressor relationships among independent variables commonly observed in multivariate logistic regression analysis (Briand et al. 2000).

The SLOC and CBO metrics have the highest values of sensitivity and completeness. However, in case of DIT metric all the classes were predicted to be non faulty hence the results are not shown.

Table 9 High severity faults model statistics

Metric	*B*	SE	Sig.	Exp (*B*)
CBO	0.102	0.033	0.002	1.107
SLOC	0.001	0.001	0.007	1.001
Constant	−2.541	0.402	0.000	0.079

Log likelihood = −79.358

Table 10 Sensitivity, specificity, precision and completeness for HSF

Metric	Sensitivity	Specificity	Precision	Completeness	Cutoff
CBO	60.90	59.80	60.00	58.30	0.24
WMC	56.50	63.90	62.70	70.80	0.24
RFC	60.40	57.10	57.90	60.40	0.24
SLOC	65.20	68.00	67.50	75.00	0.21
LCOM	56.50	58.20	57.90	68.75	0.32
NOC	–	–	–	–	–
DIT	–	–	–	–	–
Model	69.50	73.70	73.10	79.16	0.22

Table 11 Predicted correctness of model (a) HSF (b) MSF

	Predicted	
Observed	Po $\leq$ 0.25 Not faulty	Po $>$ 0.25 Faulty
(a) HSF		
Not faulty	90	32
Faulty	7 (10)	16 (38)
	Predicted	
Observed	Po $\leq$ 0.84 Not faulty	Po $>$ 0.84 Faulty
(b) MSF		
Not faulty	63	24
Faulty	18 (82)	40 (367)

Table 12 Medium severity faults model statistics

Metric	*B*	SE	Sig.	Exp (*B*)
CBO	0.190	0.038	0.0001	1.209
LCOM	−0.011	0.004	0.009	0.989
NOC	−1.070	0.320	0.001	0.343
SLOC	0.004	0.002	0.006	1.004
Constant	−0.307	0.340	0.367	0.736

Log likelihood = −63.24

Table 13 show that the cut off point for the model predicted is 0.84. The value of R^2 of the predicted model is 0.512. Table 11b shows that the classification results of the predicted model with respect to MSF. The sensitivity, specificity, and completeness values of the model are 68.9, 72.4, and 81.7%, respectively. Thus, the accuracy of the model is better than the model predicted with respect to HSF.

Low Severity Fault (*LSF*) *prediction model*: The results in Sect. 5.2.2 shows that four out of seven metrics (CBO, WMC, RFC, and SLOC) were found related to fault proneness. Table 14 shows that all of these metrics were included in the multivariate model. The sign of the coefficient of the LCOM and RFC metrics is negative, though it was positive in the

Table 13 Sensitivity, specificity, precision and completeness for MSF

Metric	Sensitivity	Specificity	Precision	Completeness	Cutoff
CBO	63.80	70.10	67.58	79.30	0.84
WMC	58.60	60.90	60.00	71.30	0.75
RFC	60.30	56.30	57.93	74.80	0.77
SLOC	67.20	67.80	67.57	80.20	0.70
LCOM	50.00	49.00	48.96	76.80	0.84
NOC	94.80	15.90	47.58	98.40	0.70
DIT	–	–	–	–	–
Model	68.90	72.40	70.78	81.70	0.84

Table 14 Low severity faults model statistics

Metric	*B*	SE	Sig.	Exp (*B*)
CBO	0.167	0.041	0.001	1.137
RFC	−0.034	0.010	0.001	0.971
WMC	0.047	0.018	0.028	1.039
SLOC	0.003	0.001	0.001	1.003
Constant	−1.447	0.371	0.005	0.354

Log likelihood = −66.81

Table 15 Sensitivity, specificity, precision and completeness for LSF

Metric	Sensitivity	Specificity	Precision	Completeness	Cutoff
CBO	66.70	65.10	66.47	72.90	0.55
WMC	59.00	57.50	57.89	61.10	0.48
RFC	56.40	57.50	57.20	66.40	0.54
SLOC	66.70	72.60	71.00	79.20	0.45
LCOM	48.70	50.00	49.65	57.60	0.59
NOC	–	–	–	–	–
DIT	51.30	46.20	47.50	55.60	0.55
Model	79.40	65.50	69.26	79.80	0.58

univariate analysis. As mentioned above, this is due to suppressor relationships among independent variables.

Table 15 shows that the SLOC and CBO metrics have the highest values of sensitivity and completeness. However, in case of the NOC metric all the classes were predicted to be non faulty. The value of R^2 of the predicted model is 0.464. The sensitivity and specificity of the model is 79.4% and 65.5%, respectively. The completeness of the model is 79.8% (see Table 16a). Thus, the accuracy of the model is the same as the model predicted with respect to the MSF but is better than the fault proneness model predicted with respect to the HSF.

Ungraded Severity Fault (*USF*) *prediction model*: In the univariate analysis, six out of seven metrics (CBO, WMC, RFC, LCOM, NOC, and SLOC) were found related to fault proneness. Table 17 shows that five metrics, the SLOC, CBO, RFC, WMC, and LCOM were included in the multivariate model. Similar to the results of univariate analysis, the

Table 16 Predicted correctness of model (a) LSF (b) USF

	Predicted	
Observed	$Po \leq 0.54$ Not faulty	$Po > 0.54$ Faulty
(a) LSF		
Not faulty	82	24
Faulty	8 (29)	31 (115)
	Predicted	
Observed	$Po \leq 0.84$ Not faulty	$Po > 0.84$ Faulty
(b) USF		
Not faulty	62	24
Faulty	17 (122)	42 (520)

Table 17 Ungraded severity fault model statistics

Metric	*B*	SE	Sig.	Exp (*B*)
CBO	0.195	0.040	0.0001	1.216
LCOM	−0.010	0.004	0.007	0.990
NOC	−0.749	0.199	0.0001	0.473
RFC	−0.016	0.006	0.006	0.984
SLOC	0.007	0.002	0.0001	1.007
Constant	0.134	0.326	0.680	1.144

Log likelihood = −63.387

Table 18 Sensitivity, specificity, precision and completeness for USF

Metric	Sensitivity	Specificity	Precision	Completeness	Cutoff
CBO	76.30	68.60	71.70	74.69	0.88
WMC	62.70	60.50	61.40	67.31	0.81
RFC	66.20	62.80	64.10	59.65	0.88
SLOC	72.90	68.60	70.34	82.40	0.76
LCOM	49.20	47.70	48.29	82.90	0.88
NOC	93.20	16.30	47.59	98.00	0.84
DIT	–	–	–	–	–
Model	71.10	72.10	71.70	81.10	0.84

sign of the coefficient of the NOC metric is also negative in the multivariate model. The sign of the coefficient of the LCOM and RFC metrics is negative, though it was positive in the univariate analysis. As mentioned above, this is due to suppressor relationships among independent variables.

Again, from Table 18 we see that the SLOC and CBO metrics have the highest values of sensitivity and completeness. However, in case of the NOC metric all the classes were predicted to be non faulty.

The value of R^2 of the predicted model is 0.501. Table 16b shows that with a cutoff point of 0.84, the model achieved 71.1% sensitivity, 72.1% specificity, and 81.1% completeness. Thus, the precision and completeness of the model is almost the same as the model predicted with respect to the MSF but it is better than the fault proneness model predicted with respect to the HSF.

Discussion: The values of the performance measures corresponding to SLOC and CBO are higher than the values of the performance measures corresponding to the WMC, RFC, LCOM, NOC, and DIT metrics. Hence, software developers can use the SLOC and CBO metrics in earlier phases of software development to measure the quality of the systems and predict which classes with higher severity need extra attention. This can help management focus resources on those classes that are likely to cause serious failures. Also, if required, developers can reconsider design and thus take corrective actions. The models predicted in the previous section could be of great help for planning and executing testing activities. For example, if one has the resources available to inspect 26% of the code, one should test 26% of the classes predicted with more severe faults. If these classes are selected for testing one can expect maximum severe faults to be covered. The models can be applied by software practitioners to assess the level of product quality at various severity levels i.e. high, medium, and low. The lower the probability of fault proneness predicted by a model at each severity level, the higher is the probability of delivering a product with less severe faults. Thus, the software practitioner can measure the acceptable level of risk at a specified severity level before deploying the software.

The sensitivity (69.5%) and completeness (79.16%) of the model predicted with respect to the HSF are slightly lower than the models predicted using the MSF, LSF, and USF. However, the precision of model predicted with respect to the HSF is the best. As mentioned in Sect. 2, Zhou and Leung (2006) categorized the faults into two levels: faults with severity rating 1 as high severity faults and faults at other severity rating as low severity faults. The results of Zhou and Leung (2006) showed that the prediction usefulness of the metrics was limited with respect to the HSF. The model predicted with respect to HSF shows less value of correctness (31.25%) and completeness (58.82%). However, our results show that when faults produced from all the phases were taken into account and an appropriate cutoff point is chosen using ROC analysis the model predicted with respect to HSF is better and more acceptable. The completeness of the model predicted using medium and low severity faults in our analysis is higher than the model predicted with respect to low severity faults in Zhou and Leung (2006). Thus, the models predicted with respect to medium and low severity faults are more useful than the model predicted with respect to low severity faults in Zhou and Leung (2006).

5.3 Machine learning methods

In this section, we present the results for two machine learning methods: Artificial Neural Networks (ANN) and Decision Trees (DT). First, we summarize the methodology and architecture used for the DT and ANN methods and then we present the results of the univariate and multivariate analysis at different severity levels of faults.

5.3.1 Research methodology used in DT analysis

In the DT method (Porter and Selby 1990), each node of the tree is associated with an independent variable. The tree is traversed during classification from the root until a leaf node is reached. Each leaf node is associated with a classification value. Chi-squared

Automatic Interaction Detection (CHAID) algorithm is used in the DT method. At each step, CHAID chooses the independent variable that has the strongest interaction with the dependent variable. Categories of each predictor are merged if they are not significantly different with respect to the dependent variable.

5.3.2 Architecture used in ANN analysis

The network used in this work belongs to Multilayer Feed Forward networks and is referred to as M-H-Q network with M source nodes, H nodes in hidden layer and Q nodes in the output layer (Aggarwal et al. 2007). The input nodes are connected to every node of the hidden layer but are not directly connected to the output node. Thus, the network does not have any lateral or shortcut connection.

The ANN repetitively adjusts different weights so that the difference between desired output from the network and actual output from the ANN is minimized. The network learns by finding a vector of connection weights that minimizes the sum of squared errors on the training data set. The summary of the ANN architecture used in this study is shown in Table 19. The ANN was trained by standard error back propagation algorithm at a learning rate of 0.005, having the minimum square error as the training stopping criterion.

The input layer has one unit for each input variable. Each input value in the data set is normalized within the interval [0, 1] using min–max normalization (see Sect. 4). We performed the univariate analysis to find out the metrics that are most important in predicting fault proneness. Then we find the combined effect of OO metrics using the backward elimination method with metrics selected in the univariate analysis. The backward elimination method includes all the independent variables in the model. Variables are deleted one at a time from the model until a stopping criteria is fulfilled.

We use one hidden layer as what can be achieved in function approximation with more than one hidden layer can also be achieved by one hidden layer (Khoshgaftaar et al. 1997). There is one unit in the output layer. The output unit with value greater than a threshold (cutoff point) indicates the class selected by the network is fault prone, otherwise it is not.

Due to the nonlinear nature of the ANN, the statistical tests for parameter significance that are used in the LR cannot be applied here. Instead, we used ROC analysis (Hanley and McNeil 1982) to heuristically assess the importance of input variables for the classification result.

Table 19 ANN summary

Architecture	
Layers	3
Input units	7
Hidden units	15
Output units	1
Training	
Transfer function	Tansig
Algorithm	Back propagation
Training function	TrainBR

5.3.3 *Univariate and multivariate analysis results*

The results of the univariate and multivariate analysis are presented with respect to high, medium, low, and ungraded severity faults. Tables 20, 21, 22, and 23 show the sensitivity, specificity, precision, completeness, and cutoff point of each metric and the model predicted.

High Severity Fault (*HSF*) *prediction model*: In the DT analysis of all the metrics, the SLOC metric has the highest sensitivity (82.6%) and completeness (90.47%) values. The CBO metric have the next highest values of sensitivity (78.3%) and completeness (88%).

In the ANN analysis, the SLOC and LCOM metrics have the highest sensitivity (73.9%) values. The NOC and DIT metrics predicted all classes to be non faulty using both the DT and ANN methods.

The sensitivity, specificity, and completeness of the DT model are 69.5, 90, and 70.8%, respectively. The ANN model has sensitivity 73.9%, specificity 71.31%, and completeness 87.5%.

Medium Severity Fault (*MSF*) *prediction model*: In case of the DT analysis, the SLOC metric has sensitivity 91.4% and the CBO metric has 81%. The completeness of the SLOC (95.4%) and CBO (89.75%) are also high. Similar results were shown using the LR analysis.

As shown in the ANN analysis, the CBO metric has 74.6% sensitivity and completeness 88.65%. The completeness of the SLOC (86.34%) is also high. Similar results were shown using the LR and DT analysis. However, in case of the NOC metric all the classes were predicted to be non faulty for the DT and ANN methods, hence the results are not shown.

Table 21 shows that the DT model achieved 89.7% sensitivity, 71.3% specificity, 97.5% completeness. The ANN model has 74.1% sensitivity, 74% specificity, and completeness 85.07%.

Low Severity Fault (*LSF*) *prediction model*: The results of the DT analysis show that the sensitivity (94.9%) of the SLOC metric is high, but the specificity (52.8%) and precision

Table 20 Sensitivity, specificity, precision, and completeness for HSF

Metric	Method	Sensitivity	Specificity	Precision	Completeness	Cutoff
CBO	DT	78.30	69.70	71.50	88.00	0.26
	ANN	60.90	68.90	67.60	70.80	0.20
WMC	DT	47.20	82.00	76.45	50.00	0.35
	ANN	65.10	63.90	64.10	79.16	0.15
RFC	DT	17.40	95.90	83.40	14.20	0.48
	ANN	69.60	73.80	73.12	72.90	0.18
SLOC	DT	82.60	52.50	57.27	90.47	0.22
	ANN	73.90	77.00	76.51	75.00	0.18
LCOM	DT	78.30	75.40	75.85	73.80	0.27
	ANN	73.90	80.30	79.28	78.00	0.25
NOC	DT	–	–	–	–	–
	ANN	–	–	–	–	–
DIT	DT	–	–	–	–	–
	ANN	–	–	–	–	–
Model	DT	69.50	90.00	84.68	70.80	0.21
	ANN	73.90	71.31	71.70	87.50	0.18

Table 21 Sensitivity, specificity, precision, and completeness for MSF

Metric	Method	Sensitivity	Specificity	Precision	Completeness	Cutoff
CBO	DT	81.00	64.40	71.00	89.75	0.44
	ANN	74.60	73.30	73.63	88.65	0.41
WMC	DT	22.40	92.00	64.10	49.40	0.50
	ANN	61.00	61.00	60.73	74.53	0.41
RFC	DT	34.10	96.60	70.00	34.07	0.89
	ANN	62.70	67.40	65.26	69.60	0.40
SLOC	DT	91.40	60.90	73.10	95.50	0.50
	ANN	72.90	77.90	75.70	86.34	0.52
LCOM	DT	60.30	72.40	67.50	72.80	0.46
	ANN	61.00	72.10	67.39	81.10	0.43
NOC	DT	–	–	–	–	–
	ANN	–	–	–	–	–
DIT	DT	20.70	88.50	61.40	23.80	0.50
	ANN	66.10	47.70	54.13	77.50	0.39
Model	DT	89.70	71.30	78.64	97.50	0.60
	ANN	74.10	74.00	73.79	85.07	0.40

Table 22 Sensitivity, specificity, precision, and completeness for LSF

Metric	Method	Sensitivity	Specificity	Precision	Completeness	Cutoff
CBO	DT	82.10	69.80	73.09	84.00	0.29
	ANN	74.40	76.40	75.85	76.05	0.36
WMC	DT	47.20	91.50	79.30	50.60	0.67
	ANN	76.90	71.70	73.10	87.32	0.27
RFC	DT	30.60	94.30	74.50	34.60	0.65
	ANN	61.50	59.40	59.97	70.42	0.27
SLOC	DT	94.90	52.80	64.10	88.60	0.27
	ANN	79.40	71.40	73.57	83.80	0.33
LCOM	DT	59.00	68.90	66.20	37.30	0.34
	ANN	56.40	73.60	68.90	45.07	0.29
NOC	DT	–	–	–	–	–
	ANN	–	–	–	–	–
DIT	DT	15.40	94.00	72.80	34.00	0.68
	ANN	53.80	55.70	55.20	77.46	0.28
Model	DT	90.30	81.10	83.40	94.30	0.50
	ANN	71.70	72.64	72.41	79.31	0.36

(64.1%) are low. In the DT analysis, it is also shown that sensitivity (82.1%), precision (73.09%) and completeness (84%) values of the CBO metric are high. The DIT metric show very poor sensitivity (15.4%), although the precision value is 72.8%. The RFC metric also has poor sensitivity (30.6%) and completeness (34.6%). The completeness of the LCOM metric (37.3%) is also low.

Table 23 Sensitivity, specificity, precision, and completeness for USF

Metric	Method	Sensitivity	Specificity	Precision	Completeness	Cutoff
CBO	DT	78.00	64.00	69.60	72.70	0.83
	ANN	73.30	72.90	72.89	82.28	0.43
WMC	DT	79.70	52.30	63.40	83.40	0.84
	ANN	65.00	64.70	64.58	77.74	0.43
RFC	DT	86.40	26.70	51.00	63.70	0.77
	ANN	61.07	65.90	63.91	74.45	0.40
SLOC	DT	93.20	58.10	72.40	75.70	0.69
	ANN	75.00	70.60	72.21	90.90	0.46
LCOM	DT	62.70	77.90	71.70	69.90	0.85
	ANN	61.70	71.80	67.41	80.80	0.44
NOC	DT	17.00	98.80	65.02	–	0.80
	ANN	–	–	–	–	–
DIT	DT	18.60	94.20	63.40	50.30	0.92
	ANN	65.00	58.80	61.08	74.13	0.41
Model	DT	91.50	87.20	88.90	94.00	0.78
	ANN	72.80	79.06	76.55	88.16	0.41

Again, it can be seen in the ANN analysis that the sensitivity of the SLOC metric (79.4%) is high, but the specificity, precision, and completeness are low. The sensitivity (76.9%), precision (73.1%) and completeness (87.32%) values of the WMC metric are high. The DIT metric show poor sensitivity (53.8%), although the completeness is 77.46%. The completeness of the LCOM metric is less (45.07%). The NOC metric predicts all classes as non faulty in the DT and ANN analysis.

Table 22 shows that the sensitivity, specificity, and completeness of the DT model are 90.3%, 81.1%, and 94.30%, respectively. Similarly, the ANN model achieved sensitivity 71.7%, specificity 72.64%, and completeness 79.31%.

Ungraded Severity Fault (*USF*) *prediction model*: In the DT and ANN analysis, the SLOC and CBO metrics have the highest values of sensitivity and completeness (see Table 23). However, in case of the NOC metric again all the classes were predicted to be non faulty.

The DT model achieved 91.5% sensitivity, 87.2% specificity, and completeness 94%. The sensitivity, specificity, and completeness of the ANN model are 72.8, 79.06 and 88.16%, respectively.

Discussion: The results of the machine learning models show that the SLOC metric has the best sensitivity and completeness at all the severity of faults. The CBO has a competitive prediction performance to the SLOC. The worst prediction performance is of the models corresponding to the NOC and DIT metrics.

The results of the DT analysis show that the model predicted using the HSF has lowest values of sensitivity and completeness. The completeness of the model predicted with regard to the MSF is highest. This shows that the model predicted with respect to the MSF will classify correctly most of the classes with high number of faults. The results of the ANN analysis show that the model predicted with respect to the HSF has higher value of completeness (87.5%) than the models predicted with respect to the MSF and LSF. Thus, this means that classes that are actually containing high severity faults will undergo thorough and necessary inspection.

In Zhou and Leung (2006) three machine learning methods were evaluated. The completeness of models predicted using these methods with respect to the HSF was less than 48%. Thus in our study, the values of performance measures of the machine learning models with respect of the HSF are higher and more acceptable than the values of the performance measures of machine learning models in the study conducted by Zhou and Leung.

5.4 Discussion and validation of hypothesis set A

In this section, we validate our hypothesis set A stated in Sect. 3.2.1. At the same time, we also compare our results with those of previous studies as shown in Table 24.

CBO Hypothesis was found to be significant in our LR analysis for all severities of faults. Sensitivity of both the LR and ANN models are the same. The machine learning methods confirmed the findings of the regression analysis as the values of sensitivity, precision, and correctness for the CBO metric is high. It was also found a significant predictor in all studies except Tang et al. (1999) and El Emam et al. 2001.

All of our models found the CBO metric to be significant predictor of fault proneness; hence, we rejected the null hypothesis and accepted the alternative hypothesis.

RFC Hypothesis was found to be significant in our LR analysis for all severities of faults. Sensitivity of the ANN and DT models is also high. Similar results were shown by (Basili et al. 1996; Briand et al. 2000; El Emam et al. 2001; Zhou and Leung 2006; Olague et al. 2007; Gyimothy et al. 2005). Tang et al. (1999) found it significant at the 95% level. Yu et al. (2002) also found the RFC metric significant predictor but their method of calculating the RFC metric was different.

Hence, we rejected the null hypothesis and accepted the alternative hypothesis for the RFC metric.

LCOM Hypothesis was found to be significant in our LR analysis (except for faults predicted with respect to low severity), contradicting the results of (Basili et al. 1996), where the LCOM was shown to be insignificant. Zhou and Leung (2006), Olague et al. (2007) and Gyimothy et al. (2005) also found the LCOM metric to be very significant predictor of fault proneness. Yu et al. (2002) calculated the LCOM in a totally different way therefore, we could not compare our results with theirs.

Hence, we rejected the null hypothesis and accepted the alternative hypothesis.

NOC Hypothesis We found the NOC metric not to be significant with respect to the LSF and HSF in our LR analysis. However, the NOC metric was found inversely related to fault proneness with respect to MSF and USF i.e. the larger the value of the NOC the less is the probability of fault detection. The results of the DT and ANN also predicted all classes to be non faulty for all severity levels of faults. Briand et al. (2001), Gyimothy et al. 2005 and Tang et al. (1999) found the NOC not to be a significant predictor of fault proneness. Basili et al. (1996), Briand et al. (2000), and Zhou and Leung (2006) found the NOC metric to be significant but they found that the larger the value of NOC, the lower the probability of fault proneness. According to (Yu et al. 2002), the NOC metric was a significant predictor of fault proneness and they found that the greater the number of children in a class, the more fault prone it is.

We thus accepted the null hypothesis for the NOC metric and rejected the alternative hypothesis. Most of the studies that examined this metric found either the NOC metric to

Table 24 Results of different validations

Metric	Our results				Basili et al. (1996)	Tang et al. (1999)	Briand et al. (2000)	Briand et al. (2001)	El Emam et al. (2001)		Yu et al. (2002)	Gyimothy et al. (2005)	Zhou et al. (2006)		Olague et al. (2007)		
Lang. used	C++				C++	C++	C++	C++	C++	C++	Java	C++	C++	C++	Java		
Method used	LR, ML (DT, ANN)				LR	LR	LR	LR	LR	LR	OLS	LR, ML (DT, ANN)	LR, ML (NNage, RF, NB)		LR		
Type of Data	NASA data set				Univ.	Comm.	Univ.	Comm.	Comm.		Comm.	Open source	NASA Data set		Open Source		
Fault Severity taken?	Yes				No	No	No	No	No		No	No	Yes		No		
	HSF	MSF	LSF	USF					#1	#2			LSF/USF	HSF			
WMC	++	++	++	++	+	+	+	++	+	0	++	++	++	++	++	++	++
DIT	0	0	0	0	++	0	++	−−	0	0	0	+	0	0	0	−−	0
RFC	++	++	++	++	++	+	++	++	++	0	+	++	++	++	++	++	++
NOC	0	−−	0	−−	−−	0	−	0			++	0	−−		0	0	0
CBO	++	++	++	++	+	0	++	++	+	0	+	++	++	++	++	++	0
LCOM	++	++	0	++	0							+	+	++	++	++	++
SLOC	++	++	++	++			++		++	++		++	++	++			

++, Denotes metric is significant at 0.01; +, denotes metric is significant at 0.05; −−, denotes metric is significant at 0.01 but in an inverse manner; −, denotes metric is significant at 0.05 but in an inverse manner; 0, denotes that metric is not significant. A blank entry means that our hypothesis is not examined or the metric was calculated in a different way. LR, logistic regression; OLS, ordinary least square; ML, machine learning; DT, decision tree; ANN, artificial neural network; RF, random forest; NB, Naïve Bayes; LSF, low severity fault; USF, ungraded severity fault; HSF, high severity fault; MSF, medium severity faults; #1, without size control; #2, with size control; comm., commercial; univ., university be not related to fault proneness or negatively related to fault proneness. The conclusion is that the NOC metric is a bad predictor of fault proneness. Hence, perhaps more attention (e.g., through walkthroughs and inspections) is given during development to the classes on which other classes depend (Briand et al. 2000).

DIT Hypothesis was not found to be significant in our univariate LR analysis. On the other hand (Briand et al. 2001) found it to be significant but in an inverse manner. Our finding is similar to those given by (Yu et al. 2002; Tang et al. 1999; El Emam et al. 2001; Zhou and Leung 2006). Basili et al. (1996) and Briand et al. (2000) found the DIT metric to be a significant predictor of fault proneness. Gyimothy et al. 2005 found the DIT metric to be a less significant predictor of fault proneness. Our DT results showed very small values of sensitivity and precision for medium, low, and ungraded severities of faults. For high severity of faults, the DT and ANN methods predicted all classes as non faulty. The ANN method showed low values of sensitivity for medium, low and ungraded severities of faults. The completeness value of the DIT is worse with respect to all the severities of faults. Table 24 shows that most of the studies found that the DIT metric is not related to fault proneness. The class may have fewer ancestors in most of the studies and further investigation is needed.

We thus accepted the null hypothesis for the DIT metric and rejected the alternative hypothesis.

WMC Hypothesis was found to be significant in our LR and ANN analysis. On the other hand, Basili et al. (1996) found it less significant. In the study conducted by (Yu et al. 2002), the WMC metric was found to be a significant predictor of fault proneness. Similar to our regression and DT and ANN results, (Briand et al. 2000; Gyimothy et al. 2005; Olague et al. 2007; Zhou and Leung 2006) also found the WMC metric as one of the best predictors of fault proneness. Rest of the studies found it to be significant predictor but at the 95% level.

All of our three models found the WMC metric to be a significant predictor of fault proneness; hence, we rejected the null hypothesis and accepted the alternative hypothesis.

SLOC Hypothesis was found to be significant in the LR analysis. It was also found significant in all the studies that examined them. Hence, we reject the null hypothesis and accept the alternative hypothesis.

In Table 25, we summarize the results of the hypothesis stated in Sect. 3.2.1 with respect to each severity of faults. The LCOM metric was found significant at the 95% level in the study conducted by Zhou and Leung (2006) with regard to low and ungraded

Table 25 Summary of hypothesis

Metric	Hypothesis accepted/rejected			
Severity of Faults	HSF	MSF	LSF	USF
RFC	√	√	√	√
CBO	√	√	√	√
LCOM	√	√	×	√
DIT	×	×	×	×
NOC	×	×	×	×
WMC	√	√	√	√
LOC	√	√	√	√

√, means the hypothesis is accepted; ×, means that the hypothesis is rejected severity levels of faults. However, in our study the LCOM metric was found significant at the 99% level with respect to the HSF, MSF, and USF severity levels of faults and was not found significant with respect to the LSF.

6 Model evaluation using ROC analysis

In this section, we present the results of the model evaluation and summarize the results of hypothesis set B presented in Sect. 3.2.2.

6.1 Model evaluation

The accuracy of the models predicted is somewhat optimistic since the models are applied on the same data set from which they are derived. To predict accuracy of the model it should be applied on different data sets thus we performed 10-cross validation of the LR, DT, and ANN models following the procedure given in Sect. 5. For the 10-cross validation, the classes were randomly divided into 10 parts of approximately equal (14 partitions of ten data points each and 1 partition of five data points each). We summarized the results of cross validation of predicted models via the LR approach in Table 26. In Tables 26 and 27, Model I is predicted with respect to the HSF, Model II is predicted with respect to the MSF, Model III is predicted with respect to the LSF and Model IV is predicted with respect to the USF.

Table 26 summarizes the results of 10-cross validation of models using the DT, and ANN methods.

We did not wish to select an arbitrary cutoff point and in order to obtain a balance between the number of classes predicted as fault prone and not fault prone, the cutoff point of the predicted model is computed using ROC analysis. Area Under the ROC Curve (AUC) is a combined measure of sensitivity and specificity. In order to compute the accuracy of the predicted models, we use the area under the ROC curve. The Standard Error (SE) for ROC curves was determined according to the method proposed by Hanley and McNeil (1982).

In case of the LR method, the AUC of model predicted with regard to the HSF is 0.686, which is, less than the AUC of the model predicted using the MSF (0.754), LSF (0.664) and the USF (0.753) (see Fig. 2). In the case of DT and ANN, the same trend is seen. All these models performed best at MSF as can be seen from their AUC values.

Table 26 Results of 10-cross validation of models

	Logistic regression			
	Model I	Model II	Model III	Model IV
Cutoff	0.25	0.77	0.49	0.37
Sensitivity	64.60	70.70	61.50	69.50
Specificity	66.40	66.70	64.20	68.60
Precision	66.21	68.29	63.48	68.96
Completeness	59.81	74.14	71.96	79.59
AUC	0.686	0.754	0.664	0.753
SE	0.044	0.041	0.053	0.039

Table 27 Results of 10-cross validation of models

	Decision tree				Artificial neural network			
	Model I	Model II	Model III	Model IV	Model I	Model II	Model III	Model IV
Cutoff	0.21	0.60	0.50	0.78	0.18	0.44	0.31	0.46
Sensitivity	69.50	89.70	90.30	91.50	65.20	71.20	71.80	71.70
Specificity	90.00	71.30	81.10	87.20	68.90	70.90	70.80	71.80
Precision	84.68	78.64	83.40	88.90	68.30	71.81	71.06	71.50
Completeness	70.80	97.50	90.60	94.00	85.70	82.48	78.32	81.30
AUC	0.766	0.888	0.875	0.835	0.722	0.809	0.765	0.809
SE	0.040	0.026	0.026	0.032	0.054	0.034	0.045	0.305

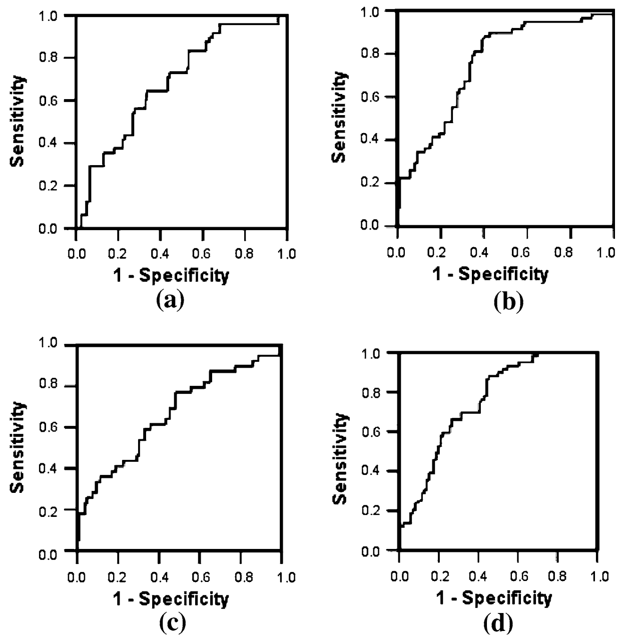

Fig. 2 ROC curve for **a** Model I **b** Model II **c** Model III **d** Model IV using LR method

The ROC curve for the LR model with respect to the high, medium, low, and ungraded severity of faults is shown in Fig. 2. In Figs. 3 and 4, the ROC curves for the DT and ANN models are presented.

6.2 Validation of hypothesis B

The results presented in Sect. 6.1 show that both the DT and ANN methods predict faulty classes with better accuracy and higher AUC than models predicted using LR method at all severity levels of faults. Table 27 show that models predicted using the DT method outperformed the models predicted using the ANN method. Thus, models predicted using DT showed the best results. Table 28 summarizes the results of the hypothesis stated in Sect.

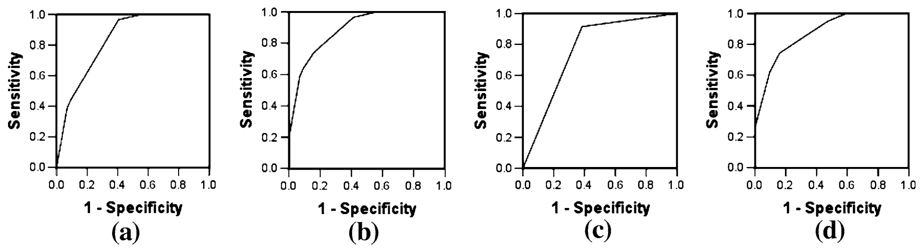

Fig. 3 ROC curve for **a** Model I **b** Model II **c** Model III **d** Model IV using tree method

Table 28 Summary of hypothesis

Severity of Faults	Hypothesis accepted/rejected		
	H1	H2	H3
HSF	√	√	√
MSF	√	√	√
LSF	√	√	√
USF	√	√	√

3.2.2, in which √ means that the hypotheses are supported. In Table 28, hypothesis is accepted (or rejected) of the models predicted with respect to all the severity levels. The basis of comparison are the values of performance measures given in Sect. 6.1. In line with other predictive models, likewise findings of this study need to be externally validated. Although LR is widely used method in the literature, our results show that performance of the models predicted using the machine learning methods is better as compared to the model predicted using the LR method. Therefore, it appears that the models predicted using the machine-learning methods might lead to development of optimum prediction models for constructing fault prone models. In (Zhou and Leung 2006), the machine learning methods performed worse than the LR models. The models predicted with the machine learning methods (DT and ANN) in our study have higher prediction performance in terms of sensitivity, and completeness than the models predicted using machine learning methods (NNage, Random Forest, Naïve Bayes) in the study conducted by Zhou and Leung (2006). Pai (2007) did not categorize faults according to severity, but the sensitivity, specificity, and precision of their predicted model using Bayesian framework was less than the model predicted in our study with respect to USF.

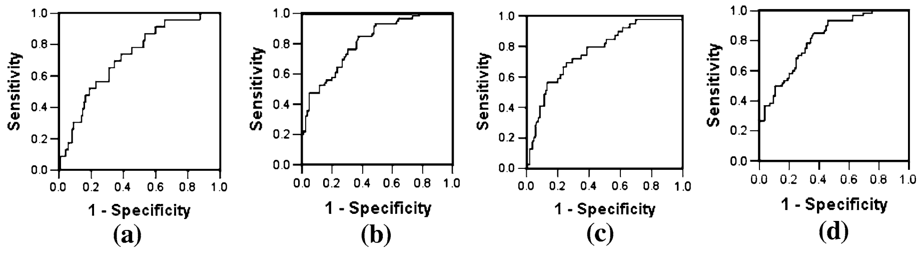

Fig. 4 ROC curve for **a** Model I **b** Model II **c** Model III **d** Model IV using ANN method

7 Threats to validity

The study has many limitations that are common with most of the empirical studies in the literature. However, it is necessary to repeat them here.

In the KC1 data set the fault severity rating are subjective and may be inaccurate. Thus, the generalisability of the results is possibly limited.

The usefulness of OO metrics for predicting fault proneness models also depends on the programming language (e.g. Java) being used. Thus, similar studies with different data sets are required to be carried out in order to establish the acceptability of the model.

Our conclusions are pertinent to only dependent variable fault proneness, as it seems to be the most popular dependent variable in empirical studies. We do not claim anything about the validity of the chosen OO metrics in this study when the dependent variable changes to maintainability or effort.

While these results provide guidance for future research on the impact of metrics on fault proneness at different severity levels, further validations are needed with different systems to draw stronger conclusions.

8 Conclusions and future work

The goal of our research is to find the effect of OO metrics on fault proneness at different severity levels of faults. We also empirically analyze and compare the performance of regression and machine learning methods at different severity levels of faults. The faults were categorized as: high, medium, and low severity faults.

Based on public domain NASA data set KC1, we analyzed the performance of predicted models using the ROC analysis. We analyzed the OO design metrics given by Chidamber and Kemerer. Thus, the main contributions of this paper are summarized as follows: first, we performed the analysis of public domain NASA data set (NASA 2004), therefore analyzing valuable data in an important area where empirical studies and data are limited. Second, we took the severity of faults into account while predicting fault proneness of the classes. Third, besides the common statistical methods, we applied machine-learning methods (DT and ANN) to predict the effect of OO metrics on fault proneness and evaluated these methods based on their performance. However, since our analysis is based on only one data set, this study should be replicated on different data sets to generalize our findings. Our main results are summarized as follows:

- The CBO, WMC, RFC, and SLOC metrics were found to be significant across all severity of faults. The LCOM metric was not found significant with respect to the LSF. The DIT metric was not found to be significantly related to fault proneness across any severity of faults. The NOC metric is not found significant when HSF and LSF were taken into account. The results of machine learning methods (DT and ANN) show that the usefulness of the NOC metric is poor. The DIT metric also shows very poor results with regard to the HSF. This may be due to the fact that the number of ancestors of each class is small in the data set.
- The model predicted with respect to high severity faults has lower accuracy than other models predicted at medium and low severities. The fault proneness models predicted with respect to medium severity faults showed the best result using all the methods.
- The DT and ANN models outperformed the LR model, although all methods yielded good AUC using ROC analysis. This study confirms that construction of models using machine learning methods is feasible, adaptable to OO systems and useful in predicting

fault prone classes. While research continues, practitioners and researchers may apply machine learning methods for constructing the model to predict faulty classes.

As in all empirical studies, the relationship we established is valid only for a certain population of systems. In this case, we can roughly characterize this population as "object-oriented, large-sized systems."

More similar type of studies must be carried out with different data sets to give generalized results across different organizations. We plan to replicate our study to predict models based on machine learning algorithms such as genetic algorithms. We may carry out cost benefit analysis of the models that will help to determine whether a given fault proneness model would be economically viable.

References

Afzal, W. (2007). *Metrics in software test planning and test design processes*. Ph.D. Disseration.

Aggarwal, K. K., Singh, Y., Kaur, A., & Malhotra, R. (2005). Software reuse metrics for object-oriented systems. In *Proceedings of the Third ACIS Int'l Conference On Software Engineering Research, Management and Applications* (*SERA '05*), 48–55.

Aggarwal, K. K., Singh, Y., Kaur, A., & Malhotra, R. (2006a). Empirical study of object-oriented metrics. *Journal of Object Technology, 5*(8), 149–173.

Aggarwal, K. K., Singh, Y., Kaur, A., & Malhotra, R. (2006b). Investigating the effect of coupling metrics on fault proneness in object-oriented systems. *Software Quality Professional, 8*(4), 4–16.

Aggarwal, K. K., Singh, Y., Kaur, A., & Malhotra, R. (2007). *Application of artificial neural network for predicting fault proneness models*. International conference on information systems, technology and management (ICISTM 2007), March 12–13, New Delhi, India.

Aggarwal, K. K., Singh, Y., Kaur, A., & Malhotra, R. (2009). Empirical analysis for investigating the effect of object-oriented metrics on fault proneness: A replicated case study. *Software Process: Improvement and Practice, 16*(1), 39–62. doi:10.1002/spip.389s.

Barnett, V., & Price, T. (1995). *Outliers in statistical data*. London: Wiley.

Basili, V., Briand, L., & Melo, W. (1996). A validation of object-oriented design metrics as quality indicators. *IEEE Transactions on Software Engineering, 22*(10), 751–761. doi:10.1109/32.544352.

Bazman's Testing Pages. (2006). A website containing articles and white papers on software testing. http://members.tripod.com/~bazman/classification.html?button7=Classification+of+Errors+by+Severity, November 2006.

Belsley, D., Kuh, E., & Welsch, R. (1980). *Regression diagnostics: Identifying influential data and sources of collinearity*. New York: Wiley.

Bieman, J., & Kang, B. (1995). Cohesion and reuse in an object-oriented system. In *Proceedings of the ACM Symposium on Software Reusability* (*SSR'94*), 259–262.

Binkley, A., & Schach, S. (1998). Validation of the coupling dependency metric as a risk predictor. In *Proceedings of the International Conference on Software Engineering*, 452–455.

Briand, L., Daly, W., & Wust, J. (1998). Unified framework for cohesion measurement in object-oriented systems. *Empirical Software Engineering, 3*(1), 65–117. doi:10.1023/A:1009783721306.

Briand, L., Daly, W., & Wust, J. (1999). A unified framework for coupling measurement in object-oriented systems. *IEEE Transactions on Software Engineering, 25*(1), 91–121. doi:10.1109/32.748920.

Briand, L., Daly, W., & Wust, J. (2000). Exploring the relationships between design measures and software quality. *Journal of Systems and Software, 51*(3), 245–273. doi:10.1016/S0164-1212(99)00102-8.

Briand, L., Wüst, J., & Lounis, H. (2001). Replicated Case Studies for Investigating Quality Factors in Object-Oriented Designs, Empirical Software Engineering. *International Journal (Toronto, Ont.), 6*(1), 11–58.

Cartwright, M., & Shepperd, M. (1999). An empirical investigation of an object-oriented software system. *IEEE Transactions on Software Engineering, 26*(8), 786–796. doi:10.1109/32.879814.

Chidamber, S., Darcy, D., & Kemerer, C. (1998). Managerial use of metrics for object-oriented software: An exploratory analysis. *IEEE Transactions on Software Engineering, 24*(8), 629–639. doi:10.1109/32.707698.

Chidamber, S., & Kamerer, C. (1991). Towards a metrics suite for object oriented design. In *Proceedings of the Conference on Object-Oriented Programming: Systems, Languages and Applications* (*OOPSLA'91*). SIGPLAN Notices, 26(11), 197–211.

Chidamber, S., & Kamerer, C. (1994). A metrics suite for object-oriented design. *IEEE Transactions on Software Engineering, 20*(6), 476–493. doi:10.1109/32.295895.

Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: A methodology review. *Journal of Biomedical Informatics, 35*, 352–359. doi:10.1016/S1532-0464(03)00034-0.

Duman, E. (2006). *Comparison of decision tree algorithms in identifying bank customers who are likely to buy credit cards*. Seventh international Baltic conference on databases and information systems, Kaunas, Lithuania, July 3–6, 2006.

Eftekhar, B., Mohammad, K, Ardebili, H., Ghodsi, M., & Ketabchi, E. (2005). Comparision of artificial neural network and logistic regression models for prediction of mortality in head truma based on initial clinical data. *BMC Medical Informatics and Decision Making, 5*(3), 3. doi: 10.1186/1472-6947-5-3.

El Emam, K., Benlarbi, S., Goel, N., & Rai, S. (1999). *A validation of object-oriented metrics*. Technical report ERB-1063, NRC.

El Emam, K., Benlarbi, S., Goel, N., & Rai, S. (2001). The confounding effect of class size on the validity of object-oriented metrics. *IEEE Transactions on Software Engineering, 27*(7), 630–650. doi:10.1109/32.935855.

Fenton, N., & Neil, M. (1999). A critique of software defect prediction models. *IEEE Transactions on Software Engineering, 25*(3), 1–15.

Gyimothy, T., Ferenc, R., & Siket, I. (2005). Empirical validation of object-oriented metrics on open source software for fault prediction. *IEEE Transactions on Software Engineering, 31*(10), 897–910. doi:10.1109/TSE.2005.112.

Hair, J., Anderson, R., & Tatham, W. (2006). *Black multivariate data analysis*. London: Pearson Education.

Han, J., & Kamber, M. (2001). *Data mining: Concepts and techniques*. India: Harchort India Private Limited.

Hanley, J., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic ROC curve. *Radiology, 143*, 29–36.

Harrison, R., Counsell, S. J., & Nithi, R. V. (1998). An evaluation of MOOD set of object-oriented software metrics. *IEEE Transactions on Software Engineering, 24*(6), 491–496. doi:10.1109/32.689404.

Henderson-Sellers, B. (1996). *Object-oriented metrics, measures of complexity*. Englewood Cliffs, NJ: Prentice Hall.

Hitz, M., & Montazeri, B. (1995). Measuring coupling and cohesion in object-oriented systems. In *Proceedings of the International Symposium on Applied Corporate Computing*, Monterrey, Mexico.

Hopkins, W. G. (2003). *A new view of statistics*. Sport Science. http://www.sportsci.org/resource/stats/.

Horch, J. (2003). *Practical guide to software quality management* (2nd ed.). London: Artech House.

Hosmer, D., & Lemeshow, S. (1989). *Applied logistic regression*. New York: Wiley.

IEEE Std. 1044-1993. (1994). IEEE standard classification for software anomalies.

Khoshgaftaar, T. M., Allen, E. D., Hudepohl, J. P., & Aud, S. J. (1997). Application of neural networks to software quality modeling of a very large telecommunications system. *IEEE Transactions on Neural Networks, 8*(4), 902–909. doi:10.1109/72.595888.

Khoshgoftaar, T., Geleyn, E., Nguyen, L., & Bullard, L. (2002). Cost-sensitive boosting in software quality modeling. In *Proceedings of 7th IEEE International Symposium on High Assurance Systems Engineering*, 51–60.

Laird, L., & Brennan, M. (2006). *Software measurement and estimation: A practical approach*. NJ: Wiley.

Lake, A., & Cook, C. (1994). Use of factor analysis to develop OOP software complexity metrics. In *Proceedings of the 6th Annual Oregon Workshop on Software Metrics*, Silver Falls, Oregon.

Lee, Y., Liang, B., Wu, S., & Wang, F. (1995). Measuring the coupling and cohesion of an object-oriented program based on information flow. In *Proceedings of the International Conference on Software Quality*, Maribor, Slovenia.

Li, W., & Henry, S. (1993). Object-oriented metrics that predict maintainability. *Journal of Systems and Software, 23*(2), 111–122. doi:10.1016/0164-1212(93)90077-B.

Lorenz, M., & Kidd, J. (1994). *Object-oriented software metrics*. Englewood Cliffs, NJ: Prentice-Hall.

Lovin, C., & Yaptangco, T. (2006). *Best practices: Measuring the success of enterprise testing*. Dell Power Solutions. pp. 101–103.

Marini, F., Bucci, R., Magri, A. L., & Magri, A. D. (2008). Artificial neural networks in chemometrics: History, examples and perspectives. *Microchemical Journal, 88*(2), 178–185. doi:10.1016/j.microc.2007.11.008.

Menzies, T., Greenwald, J., & Frank, A. (2007). Data mining static code attributes to learn defect predictors. *IEEE Transactions on Software Engineering, 32*(11), 1–12.

NASA. (2004). *Metrics data repository*. http://www.mdp.ivv.nasa.gov.

Olague, H., Etzkorn, L., Gholston, S., & Quattlebaum, S. (2007). Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes. *IEEE Transactions on Software Engineering, 33*(8), 402–419. doi:10.1109/TSE.2007.1015.

Pai, G. (2007). Empirical analysis of software fault content and fault proneness using Bayesian methods. *IEEE Transactions on Software Engineering, 33*(10), 675–686. doi:10.1109/TSE.2007.70722.

Phadke, A., & Allen, E. (2005). Predicting risky modules in open-source software for high-performance computing. In *Proceedings of Second International Workshop on Software Engineering for High Performance Computing System Applications*, 60–64.

Porter, A., & Selby, R. (1990). Empirically guided software development using metric-based classification trees. *IEEE Software, 7*(2), 46–54. doi:10.1109/52.50773.

Promise. http://promisedata.org/repository/.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series A (General), 36*, 111–147.

Tang, M. H., Kao, M. H., & Chen, M. H. (1999). An empirical study on object-oriented metrics. In *Proceedings of Metrics*, 242–249.

Tegarden, D., Sheetz, S., & Monarchi, D. (1995). A software complexity model of object-oriented systems. *Decision Support Systems, 13*(3–4), 241–262. doi:10.1016/0167-9236(93)E0045-F.

Tian, J. (2005). *Software quality engineering: Testing, quality assurance, and quantifiable improvement*. NJ: Wiley.

Yu, P., Systa, T., & Muller, H. (2002). Predicting fault-proneness using OO metrics: An industrial case study. In *Proceedings of Sixth European Conference on Software Maintenance and Reengineering*, Budapest, Hungary, 99–107.

Zhou, Y., & Leung, H. (2006). Empirical analysis of object-oriented design metrics for predicting high severity faults. *IEEE Transactions on Software Engineering, 32*(10), 771–784. doi:10.1109/TSE.2006.102.

Author Biographies

Yogesh Singh is a professor with the University School of Information Technology, Guru Gobind Singh Indraprastha University, Delhi, India. He is also Controller of Examinations with the Guru Gobind Singh Indraprastha University, Delhi, India. He was founder Head (1999–2001) and Dean (2001–2006) of University School of Information Technology, Guru Gobind Singh Indraprastha University. He received his master's degree and doctorate from the National Institute of Technology, Kurukshetra. His research interests include software engineering focusing on planning, testing, metrics, and neural networks. He is coauthor of a book on software engineering, and is a Fellow of IETE and member of IEEE. He has more than 200 publications in international and national journals and conferences.

Arvinder Kaur is a Reader with the School of Information Technology. She obtained her doctorate from Guru Gobind Singh Indraprastha University, Delhi, India and her master's degree in computer science from Thapar Institute of Engineering and Technology. Prior to joining the school, she worked with Dr. B. R. Ambedkar Regional Engineering College, Jalandhar and Thapar Institute of Engineering and Technology. Her research interests include software engineering, object-oriented software engineering, software metrics, microprocessors, operating systems, artificial intelligence, and computer networks. She is also a lifetime member of ISTE and CSI. She is also a member of ISTE,CSI, and ACM. Kaur has published 45 research papers in national and international journals and conferences. Her paper titled "Analysis of object oriented Metrics" was published as a chapter in the book *Innovations in Software Measurement* (Shaker-Verlag, Aachen 2005).

Ruchika Malhotra is a research scholar with the University School of Information Technology, Guru Gobind Singh Indraprastha University, Delhi, India. She received her master's degree in software engineering from the University School of Information Technology, Guru Gobind Singh Indraprastha University, Delhi, India. Her research interests are in improving software quality, statistical and adaptive prediction models for software metrics, neural nets modeling, and the definition and validation of software metrics. She has more than 25 publications in international journals and conferences. Her paper titled "Analysis of object oriented Metrics" was published as a chapter in the book Innovations in Software Measurement (Shaker -Verlag, Aachen 2005). She can be contacted by e-mail at ruchikamalhotra2004@yahoo.com.