Software Defect Prediction in Large Space Systems through Hybrid Feature Selection and Classification

Shomona Jacob[1] and Geetha Raju[2]
[1]SSN College of Engineering, affiliated to Anna University, Chennai, Tamilnadu, India
[2]College of Engineering, Guindy, Anna University, Chennai, Tamilnadu, India

Abstract: *Data mining and machine learning techniques have been used in several scientific applications including software fault predictions in large space systems. State-of the-art research revealed that existing space systems succumb to enigmatic software faults leading to critical loss of life and capital. This article presents a novel approach to solve this issue of overlooking software faults by utilizing both features selection and classification techniques to accurately predict software defects in aerospace systems. The main objective was to identify the preeminent feature selection and prediction technique that enhanced the software fault prediction accuracy with the optimal set of features. The investigations affirmed that a novel hybrid feature selection method revealed the most optimal set of predictive features although no particular predictive technique was suitable to predict faults in all space system datasets. Besides, the exploration of data mining techniques in fault prediction on the NASA Lunar space system software data clearly portrayed the improved fault prediction accuracy (~82% to ~98%) with the feature set selected by the proposed Hybrid Feature Selection method. Also, the random sub sampling method revealed an improved mean Matthew's Correlation Coefficient (MCC) and accuracy ranging from ~0.7 to ~0.9 and ~86% to ~98% respectively. This we believe generates further scope for future investigations on the most contributing space system features for fault prediction thus enabling design of aerospace systems with minimal faults and enhanced performance.*

Keywords: *Classification, data mining, hybrid feature selection, NASA datasets, prediction, software defects.*

Received November 21, 2013; accepted June 12, 2014

1. Introduction

Software source code defect prediction has been an economically important field in software engineering for more than 20 years [10]. A defective module in software causes high repair and development cost and reduces quality of the software [2]. The growing demand for higher operational efficiency and safety in defence systems has resulted in a growing interest in fault-detection techniques [1, 3, 4, 19, 21, 23]. Hence, this research aimed at evolving a suitable and less complex software fault prediction framework that could yield higher accuracy in fault prediction with minimum number of optimal system features. Data mining [7, 25] is the task of analyzing data from various perspectives and consolidating/summarizing the data into relevant and meaningful information. Data mining techniques viz, feature selection and classification have proved very effective in predicting biological defects, irregularities in clinical data and revealing significant medical facts that raised interest in exploring such avenues for drug therapy and clinical decision making. Feature selection [7, 9, 17] is the method of deciding on a subset of important features for building reliable learning models. Classification [20] is a data analysis technique that is used to distinguish important data classes/categories. This paper aims at identifying the optimal and minimal set of software features that could predict the fault-proneness of software in aerospace systems with improved accuracy. The performance measures used to evaluate the proposed approach include the Matthew's Correlation Coefficient (MCC) [20, 21], accuracy, sensitivity and specificity.

Software errors are usually not found until the late stages of the development cycle, when it turns expensive to return and fix them [2, 8, 14, 23]. Addressing these errors is highly essential failing which, software developers build a reputation for delivering faulty products or, create life-critical situations when the software is part of larger systems or devices, such as defence equipments or medical treatment plants [3]. Hence, detecting and predicting fault-proneness in software systems (aerospace systems) to improve the quality of software utilized in designing defence equipments was the rationale for this research.

Several papers on mining software faults through prediction techniques have been proposed in literature [4, 15, 18]. Some of the papers discussed include methods for fault prediction such as size and complexity metrics, multivariate analysis, and multi-co-linearity using Bayesian belief networks. NB [7, 22, 23] is widely used for building classifiers. When developing a defect predictor, the probability of each class is calculated, given the attributes extracted from a module, using metrics such as Halstead and McCabe ones etc., (i.e., metrics that are relevant to predicting faulty modules). Menzies *et al*. [15] developed predictors with Naïve Bayes (NB) classifier for fault characteristics. They discovered more predictive power in combined or hybrid predictors than in the mono metrics. They found that NB was the best faulty model predictor reported so far.

Vandecruys *et al*. [23] used the Ant Colony Optimization (ACO) algorithm, and the max-min ant system to develop the AntMiner+model that classifies the dataset into either faulty or non-faulty modules. This algorithm achieved a predictive accuracy that was competitive to other methods. Predictors that were built using the previous techniques, suffered from high possible errors in assigning records to the correct class. NB provides high number of incorrectly classified modules [11]. As a result, many algorithms were built [5, 13, 18] to overcome the significant drawbacks of NB. One of those algorithms that demonstrated the accuracy of NB technique was Lazy Bayes Rules (LBR) [18] . However, LBR had high computational overheads. A group of researchers conducted manual software reviews to find defective modules [5]. They found that approximately 60 percent of defects could be detected manually. Reviews and inspections found over 50% of the defects in artefacts, regardless of the lifecycle phase applied.

Twala [23] worked on four publicly available NASA datasets and stated the NB classifier to yield more robust software fault prediction while most ensembles with a decision tree classifier as one of its components also achieved higher accuracy rates according to their study. Evidence records that most of the ensembles improved the prediction accuracy of the baseline classifiers Decision Tree (DT), K-Neighbours (k-NN), Naïve Bayes Classifier (NBC) and Vector Machines Classifiers (SVM). Surprisingly, most of the ensembles with NBC as one of its components did not perform as good as when NBC was just a single classifier. In addition, the overall performance of feature selection for all the ensembles was very poor [23]. According to the above study, it appeared that there was currently no reasonable data to model software fault prediction. Secondly, method-level metrics appeared to be dominant in software fault prediction with class-level metrics being hardly utilised.

This paper placed focus on a recent article [23] on NASA datasets using ensemble classifiers. We chose this paper for three main reasons: The paper is recent and the data is publicly available; the accuracy reported by ensemble techniques revealed great scope for improvement; and design of more accurate fault prediction techniques could greatly enhance the quality of software currently being used in defence systems. This research focussed on three main objectives: Utilizing feature selection techniques to identify the optimal set of software features for fault prediction; identify a suitable predictive technique that yields maximum accuracy in classification; and formulate a software fault prediction framework for space systems. The proposed methodology and the space system dataset utilized in this research are detailed in the subsequent section.

The rest of the paper is organized as follows: Section 2 describes the data mining framework and investigations. Section 3 presents the experimental results. Section 4 discusses the improvements claimed by the current research findings while section 5 concludes the paper with a clear idea of possible extensions to this work.

2. Materials and Methods

The publicly available datasets of the NASA MDP repository was utilized for this research. NASA's Metrics Data Program (MDP) Repository [14, 15, 16] is a database that stores problem, product, and metrics data. The primary goal of this data repository is to provide project data to the software community. In doing so, the MDP collects artefacts from a large NASA dataset, generates metrics on the artefacts, and then generates reports that are made available to the public at no cost. The main characteristics of the data are tabulated in Table 1.

Table 1. Desciption of the NASA aerospace system datasets.

Data Set	Attributes	Instances	Language	Description
CM1	38	344	C	NASA spacecraft instrument
JM1	22	9593	C	Real time predictive ground system
KC3	40	200	Java	Satellite-image data
MW1	38	264	C	Zero-gravity experiment related to combustion
PC1	38	759	C	Flight software for earth orbiting satellite
PC2	37	1585	C	Dynamic simulator for altitude control systems
PC3	38	1125	C	Flight software for earth orbiting satellite
PC4	37	1399	C	Flight software for earth orbiting satellite

The eight NASA datasets (CM1, JM1, MW1, KC3, PC1, PC2, PC3 and PC4) contain static code measures [14, 16, 23] (LOC, Halstead, MaCabe etc.,) along with their defect rates in numeric form. The metrics are based on product's size, complexity and vocabulary.

2.1. Software Fault Prediction Methodology

The methodology proposed in this paper for software defect prediction comprises of two phases: Training phase; and validation phase. The former involves data pre-processing, feature selection and classification of the training data. The latter phase comprises of validating the performance of the classifiers investigated in this study using cross-validation and random sampling techniques and ranking the performance of the classifiers based on the classification accuracy and MCC. The computational framework for software defect prediction using data mining techniques is portrayed in Figure 1.

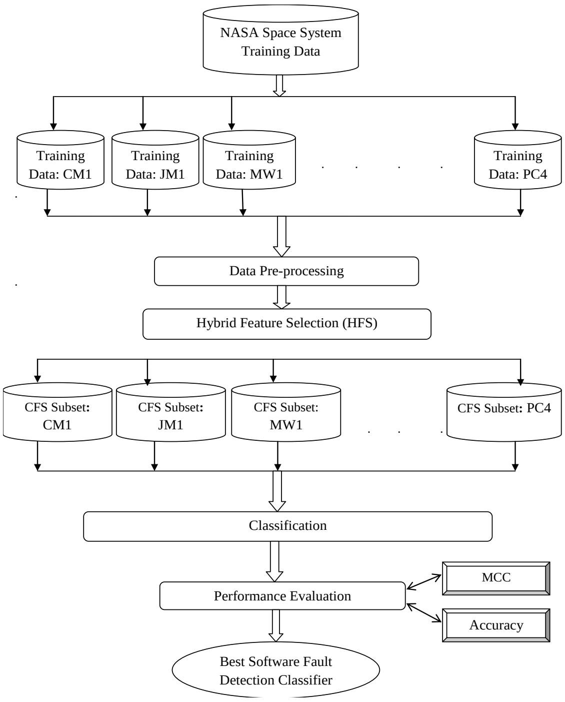

Figure 1. Proposed software fault prediction framework.

2.2. Data Pre-Processing

The data pre-processing phase [20, 21] comprised of data cleaning and transformation for easy and efficient processing on software tools for software prediction. The attributes of each space system dataset were loaded onto Excel spreadsheets and saved as Comma Separated Version (CSV) files for execution on WEKA data mining suite [25]. Missing values were eliminated from further processing. This phase resulted in the clean training data for further processing using feature selection and classification algorithms

2.3. Hybrid Feature Selection

The authors of this research paper attempted to investigate the feature selection capability of their novel HFS method [20] (proposed to mine biological data) in order to extract contributing features for software defect prediction. This phase involved executing the Hybrid Feature Selection (HFS) method proposed by the authors Ramani and Jacob [19] that attempted to automate the process of finding the minimal and optimal set of features, by combining the ranking feature selection algorithms with feature subset selection methods yielding features highly correlated to the class and least correlated to each other. Since both the ranking (Gain Ratio Criterion) and subset selection methods (Correlation Feature Subset) were utilized to obtain the optimal feature set, this was termed the Hybrid Feature Selection strategy.

The information gain ratio was calculated as the ratio between the Information Gain (InfoG) and the Intrinsic Value (IntrinV), according to Equation 1.

$$IGRatio(r,f\ ) = InfoG\ /\ IntrinV \tag{1}$$

The attributes were then ranked in the descending order of the gain ratio score and were used for the CFS Subset selection method. The CFS criterion [6] is defined as follows:

$$CFS = \underset{s_K}{MAX}\left[\frac{r_{cf1} + r_{cf2} + \dots + r_{cfk}}{\sqrt{k + 2(r_{f1f2} + \dots + r_{fifj} + \dots + r_{fkf1})}}\right] \tag{2}$$

Where r_{cfi} and r_{fifi} variables were referred to as correlations. The attributes that portrayed a high correlation to the target class and least relevance to each other were chosen as the best subset of attributes.

2.4. Classification

The main objective of classification [8, 12, 13, 18] is to accurately predict the target class for each record. The best performing classification algorithms in this study are briefly explained in the following sub-sections.

2.4.1. Bayesian Belief Network Learning Algorithm

A Bayesian network [19, 20, 21] over a set of variables *U* was a network structure Bs, a Directed Acyclic Graph (DAG) over the set of variables U and a set of probability tables given by [19]:

$$B_P = \{p(u \mid pa(u)) \mid u \in U\} \tag{3}$$

Where *pa*(*u*) was the set of parents of u in B_S and the network represented a probability distribution given by:

$$P(U) = \prod_{u \in U} p(u \mid pa(u)) \tag{4}$$

The inference made from the Bayesian Network was to allocate the category with the maximum probability. The simple estimator with the K2 local search method using Bayes Score was utilized for the execution of the algorithm.

2.4.2. Nearest-Neighbour Algorithm

The Nearest-Neighbour Algorithm (NNA) [1, 10, 11, 13] was also investigated to build the prediction model for NASA space system data. NNA calculates similarities between the test sample and all the training samples. In the current study, the distance between vector p_x and p_y is defined as following [13]:

$$D(p_x, p_y) = 1 - \frac{p_x \bullet p_y}{\|p_x\| \bullet \|p_y\|} \tag{5}$$

In Equation 5 *px.py* denotes the inner product of p_x and p_y. ||*p*|| denotes the module of vector *p*. The smaller the

$D(p_x.p_y)$ is, the more similar p_x to p_y is. In NNA, given a vector p_t and training set $P=\{p_1, \ldots, p_n, ..., p_N\}$, p_t will be designated to the same class of its nearest neighbour p_n in P, i.e., the vector having the smallest $D(p_n, p_t)$. NN algorithms have three defining general characteristics [1, 13]; a similarity function, a typical instance selection function and a classification function.

2.4.3. Ensemble Classifier

AdaBoost [5, 11, 21, 25], a meta-learning ensemble classifier combines a series of ‘k’ learned models with the aim of creating a composite model. Initially, Adaboost assigned each training instance an equal weight that equalled 1/number of training instances. A number of iterations were executed wherein, instances from the dataset were sampled by weight to form the training set. A classifier model was derived and its error rate was computed with the training set that later served as the test set. The instance weights were adjusted according to the error-rate. For each class, the sum of the weights of each classifier that assigned class ‘c’ to an instance ‘X’ was determined. The class with the highest sum was considered as the category of the instance X. The performance evaluation methods and parameters are briefed about in the subsequent section.

2.4.4. Jack-knife Cross-Validation Method

In Jack-knife cross-validation [21], each one of the statistical samples in the training dataset was in turn singled out as a test sample and the predictor was trained by the remaining samples. The following indexes were adopted to test our proposed predictors.

$$\Re_{ACC} = \frac{TP+TN}{TP+FP+TN+FN} \tag{6}$$

$$\Re_{MCC} = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP+FN)\times(TN+FP)\times(TP+FP)\times(TN+FN)}} \tag{7}$$

$$\Re_{SEN} = \frac{TP}{TP+FN} \tag{8}$$

$$\Re_{SPE} = \frac{TN}{TN+FP} \tag{9}$$

Where $\Re_{MCC}$ reflected the Mathews Correlation Coefficient; $\Re_{ACC}$ reflected the accuracy, i.e., the rate of correctly predicted records, $\Re_{SEN}$ reflected the sensitivity, i.e., the rate of defective records correctly predicted; $\Re_{SPE}$ reflected the specificity, i.e., the rate of non-defective records that were correctly predicted. TP, TN, FP and FN denoted the number of true positives, true negatives, false positives and false negatives, respectively.

3. Experimental Results

The performance of the HFS and classification algorithms was evaluated on the WEKA machine-learning toolkit [25]. The results are discussed in two sections. The first section reveals the results of the HFS method while the latter section describes the performance of the classification algorithms.

3.1. HFS Method

The HFS method was executed on all the eight NASA datasets and was found to reduce the feature set size to nearly one-third of the original data set. However, the ten-fold cross-validation technique was used to evaluate the predictor performance on the JM1 dataset in view of the massive size of the data. The performance of the proposed HFS algorithm was further evaluated as described in the ensuing section. The feature set size and the description of the NASA datasets are tabulated in Table 2.

Table 2. Feature set of NASA datasets pre- and post- feature selection.

Dataset	Entire Feature Set (EFS) Size	HFS Feature Set Size	HFS Selected Features
CM1	38	8	Loc_Comments, Cyclomatic_Density, Loc_Executable, Halstead_Content,Num_Unique_Operands, Num_Unique_Operators,Percent_Comments, Loc_Total
JM1	22	7	Loc_Blank,Loc_Code_And_Comment,Loc_Comments,Cyclomatic_Complexity,Halstead_Content,Halstead_Volume,Loc_Tot
KC3	40	4	Loc_Blank,Branch_Count,Loc_Code_And_Comment Normalized_Cylomatic_Complexity
MW1	38	8	Loc_Blank,Loc_Comments,Edge_Count, Halstead_Content, Modified_Condition_Count,Node_Count, Num_Unique_Operands, Number_Of_Lines
PC1	38	10	Loc_Blank,Loc_Code_And_Comment,Loc_Comments,Cyclomatic_Density, Loc_Executable, Parameter_Count, Halstead_Content, Node_Count, Normalized_Cylomatic_Complexity, Num_Unique_Operands
PC2	37	5	Loc_Comments,Cyclomatic_Density,Halstead_Content, Modified_Condition_Count,Percent_Comments
PC3	38	7	Loc_Blank,Loc_Code_And_Comment,Loc_C,Per_Comments, Halstead_Content, Halstead_Length, Num_Unique_Operands,
PC4	37	4	Loc_Code_And_Comment,Condition_Count,Essential_Complexity, Percent_Comments

3.2. Performance of Prediction Algorithms

A comparison of seven classification algorithms (BN-Bayesian Network; NB-Naïve Bayes; AD-Adaboost; NN-Nearest-Neighbour; RF-Random Forest; RT-Random Tree; J48-Decision Tree) was performed on the NASA datasets. The comparative results of the predictor performances before and after feature selection are tabulated in Table 3.

Table 3. Comparison of predictor performance on NASA datasets

Dataset	Feature Selection	Measures	BN	NB	AD	NN
CM1	EFS[1]	Accuracy	66.6	82.6	87.8	77.9
		MCC	0.211	0.219	0	0.011
	HFS[2]	Accuracy	82.8	85.5	87.8	80.8
		MCC	0.269	0.263	0	0.003
JM1	EFS	Accuracy	70.7	81.4	81.7	77.1
		MCC	0.247	0.226	0	0.223
	HFS	Accuracy	75.2	81.2	81.7	76.4
			0.266	0.277	0	0.203
KC3	EFS	Accuracy	77.5	78.5	84	75.5
		MCC	0.094	0.231	0.399	0.123
	HFS	Accuracy	79	81	83.5	78.5
		MCC	0.126	0.268	0.374	0.214
MW1	EFS	Accuracy	81.4	81.8	84.8	83.7
		MCC	0.304	0.31	-0.07	0.155
	HFS	Accuracy	87.1	85.6	84.8	83.7
		MCC	0.384	0.373	-0.07	0.127
PC1	EFS	Accuracy	70.2	88.5	92	89.9
		MCC	0.276	0.274	0	0.287
	HFS	Accuracy	75.1	88.7	92	90.6
		MCC	0.219	0.288	0	0.323
PC2	EFS	Accuracy	86	95.5	98.5	98
		MCC	0.156	0.078	-0.07	-0.01
	HFS	Accuracy	96.2	95.8	99	98.4
		MCC	0.186	0.114	0	0.125
PC3	EFS	Accuracy	65.1	32.6	87.6	85.7
		MCC	0.271	0.124	0	0.308
	HFS	Accuracy	74.8	82.4	87.6	84.4
		MCC	0.33	0.293	0	0.291
PC4	EFS	Accuracy	74.5	87.3	88.2	86.6
		MCC	0.346	0.364	0.283	0.398
	HFS	Accuracy	79.3	88.6	89.3	87.4
		MCC	0.462	0.401	0.378	0.434

The tabulated results clearly reveal the improvement in software defect prediction accuracy on the space system datasets even in the presence of the reduced feature set, with the feature set being reduced to nearly one–third of the original feature set size.

Moreover, in terms of computational complexity, the nearest neighbor algorithm proved to be executing in minimum time closely followed by the Bayesian approaches. In order to prove the unbiased nature of the results and to better reflect the strength of the chosen feature set and the predictive power of the formulated fault prediction framework, the calculations were also done on many randomly sampled balanced sets and the results on the trials reported as mean accuracy and MCC in Table 5 and the optimal predictor performance is graphically portrayed in Figure 2.

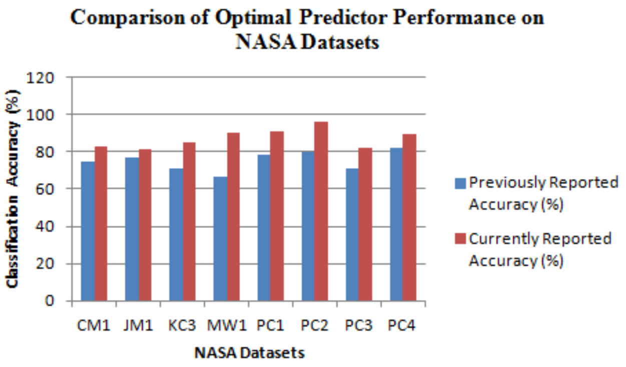

Figure 2. Optimal Predictor Performance on the NASA Datasets

The comparative results of the decision tree predictor performances' are tabulated in Table 4.

Table 4. Comparison of decision tree predictors' performance on NASA datasets.

Dataset	Feature Selection	Measures	RF	RT	J48
CM1	EFS	Accuracy	86.3	82	82.3
		MCC	0.05	0.193	0.109
	HFS	Accuracy	86.9	82	85.5
		MCC	0.072	0.176	-0.05
JM1	EFS	Accuracy	80.7	76.1	79.9
		MCC	0.269	0.207	0.211
	HFS	Accuracy	80.2	75.1	81.9
		MCC	0.26	0.177	0.166
KC3	EFS	Accuracy	82.5	77	77.5
		MCC	0.262	0.204	0.212
	HFS	Accuracy	81.5	77	85
		MCC	0.295	0.204	0.449
MW1	EFS	Accuracy	87.9	85.6	88.6
		MCC	0.154	0.216	0.212
	HFS	Accuracy	87.5	84.1	90
		MCC	0.176	0.039	0.455
PC1	EFS	Accuracy	90.9	88.5	90.1
		MCC	0.184	0.195	0.199
	HFS	Accuracy	91.4	88.9	90.5
		MCC	0.31	0.24	0.226
PC2	EFS	Accuracy	98.9	98.1	99
		MCC	-0.0	-0.01	0
	HFS	Accuracy	98.9	97.9	99
		MCC	-0.00	-0.01	0
PC3	EFS	Accuracy	87.6	84	85.4
		MCC	0.275	0.27	0.2
	HFS	Accuracy	87.8	85.2	87.6
		MCC	0.295	0.308	0
PC4	EFS	Accuracy	90.6	87.6	88.6
		MCC	0.543	0.434	0.465
	HFS	Accuracy	89.3	88.8	88.8
		MCC	0.503	0.493	0.36

The classifiers were chosen based on their performance on the original dataset.

Table 5. Predictor performance on randomly sampled HFS datasets.

Dataset	Classifier	Mean Accuracy	Mean MCC	Mean Sensitivity	Mean Specificity
CM1	BN	86.367	0.73	0.8637	0.766
JM1	BN	86.983	0.72	0.869	0.505
KC3	J48	91.5	0.83	0.915	0.789
MW1	J48	96.28	0.93	0.962	0.705
PC1	NN	98.23	0.97	0.982	0.92
PC2	BN	98.9	0.96	0.989	0.352
PC3	RT	97.58	0.95	0.975	0.892
PC4	RF	98.13	0.96	0.981	0.925

4. Discussions

Precise prediction of software faults in space systems is very valuable to engineers, especially those dealing with software development processes. This is important for minimizing cost and improving effectiveness of the software testing process. The results of the proposed methodology on the eight NASA space system datasets suggest that the Bayesian and Decision Tree approaches could be successfully applied in software fault prediction with HFS feature sets yielding overall significant increase in prediction performance.

4.1. HFS Method vs Feature Ranking Approaches

The HFS method combines the power of both ranking and feature subset selection approaches. The algorithm

[1] Entire Feature Set
[2] Hybrid Feature Selection Feature Set

automatically defines the number of features in the extracted feature subset. This is an improvement over the feature ranking algorithms that generate a rank of all the features based on a predefined criterion. The number of features to be selected for classification has to be decided by the user who sets the threshold for feature selection. This may often result in more number of features being selected for classification and may lead to extensive time being consumed before the optimal feature set is identified.

4.2. Comparison to Previous Work

The improvements put forth by this research analysis in comparison to previous work is reported in Table 6 based on the results of Song *et al*. [22, 23] who have reported on fault prediction in NASA space system datasets.

Table 6. Comparison of predictor performance to previous work.

S.No	NASA Dataset	Previously Reported Accuracy (%)	Currently Reported Accuracy (%)
1	CM1	74.9	82.8
2	JM1	76.6	81.2
3	KC3	70.8	85
4	MW1	66.5	90
5	PC1	78.7	90.6
6	PC2	79.7	96.2
7	PC3	71.1	82.4
8	PC4	82.2	89.3

However, the previous work did not report on the MCC measure of the predictor techniques. The comparisons clearly reveal the improved classification performance with comparison to previous work, with reduced computational complexity. The optimal feature sets identified by this research generates further scope for design investigations on the detected software space system attributes for fabrication of improved and fault-free space systems.

This research has achieved three main objectives: The utilization of feature selection techniques has unearthed the relevance of the most contributing properties in space system software for fault prediction; reduction in the number of features for prediction greatly minimized the computational complexity in terms of time and memory requirements; and the obtained classification accuracy and MCC is much higher compared to the previous reports on the NASA datasets with the MCC (stated to be more precise in ranking the predictor techniques on unbalanced binary class datasets) being reported for the first time on NASA space system datasets.

5. Conclusions

The goal of fault prone modules' prediction using data mining techniques aims at improving the software development process. This enables the software manager to effectively allocate project resources toward those modules that require more effort. This will eventually enable the developers to fix the bugs before delivering the software product to end users. This research placed focus on identifying the optimal set of predictive features in NASA space system datasets to enable design of fault-free space systems for utilization in defence purposes. This research has revealed the most contributing features for fault-prediction in space system software with the highest reported accuracy thus far, consequently paving way for further investigations on the possible design enhancements for space systems.

References

[1] Alhutaish R. and Omar N., "Arabic Text Classification Using K-Nearest Neighbour Algorithm," *The International Arab Journal of Information Technology*, vol. 12, no. 2, pp. 1-6, 2014.

[2] Compton P., Edwards G., Kang B., Malor R., Menzies T., Preston P., Srinivasan A., and Sammut S., "Ripple Down Rules: Possibilities and Limitations," *in Proceeding of the 6th Knowledge Acquisition for Knowledge-Based Systems Workshop*, pp.6-1-6-20, Canada, 1991.

[3] Fenton N. and Neil M., "Critique of Software Defect Prediction Models," *IEEE Transactions on Software Engineering*, vol. 25, no. 5, pp. 679-685, 1999.

[4] Fenton N. and Ohlsson N., "Quantitative Analysis of Faults and Failures in a Complex Software System," *IEEE Transactions on Software Engineering*, vol. 26, no. 8, pp. 797-814, 2000.

[5] Gaines B. and Compton P., "Induction of Ripple-Down Rules Applied to Modeling Large Databases," *Journal of Intelligent Information Systems*, vol. 5, no. 3, pp. 211-228, 1995.

[6] Halstead M., *Elements of Software Science*, Elsevier, 1977.

[7] Han J. and Kamber M., *Data Mining Concepts and Techniques*. Second edition, Morgan Kaufman Publishers, 2006.

[8] Hassan A. and Holt R., "Guest Editors. Introduction: Special Issue on Mining Software Repositories," *IEEE Transactions on Software Engineering*, vol. 31, no. 6, pp. 1-20, 2005.

[9] Jacob S. and Ramani G., "Design and Implementation of a Clinical Data Classifier: A Supervised Learning Approach," *Public Library of Science*, vol. 8, no. 2, pp. 16-26, 2013.

[10] Kagdi H., Collard M., and Maletic J., "A Survey and Taxonomy of Approaches for Mining Software Repositories in the Context of Software Evolution," *Journal of Software Maintenance and Evolution*: *Research and Practice*, vol. 19, no. 2, pp. 77-131, 2007.

[11] Lo D., Khoo S., and Liu C., "Efficient Mining of Iterative Patterns for Software Specification Discovery," *in Proceeding of the 13th ACM*

International Conference on Knowledge Discovery and Data Mining, San Jose, pp. 460-469, 2007.

[12] Martens D., Backer M., Haesen R., Vanthienen J., Snoeck M., and Baesens B., "Classification with Ant Colony Optimization," *IEEE Transactions on Power Systems*, vol. 11, no. 5, pp. 651-655, 2007.

[13] Martin B., *Instance-Based learning: Nearest Neighbour With Generalization*, University of Waikato, 1995.

[14] McCabe T., "A Complexity Measure," *IEEE Transactions Software Engineering*, vol. 2, no. 4, pp. 308-320, 1976.

[15] Menzies T., Greenwald J., and Frank A., "Data Mining Static Code Attributes to Learn Defect Predictors," *IEEE Transactions on Software Engineering*, vol. 33, no. 1, pp. 2-13, 2007.

[16] Metric Data Program MDP, http://mdp.ivv.nasa.gov, Last Visited 2014.

[17] Nada V. and Lavrac N., "Feature SubsetSelection in Association Rules Learning Systems," *in Proceeding of Slovenian Electrical and Computer Science Conference*, pp. 301-304, 1999.

[18] Najadat H. and Izzat A., "Enhance Rule Based Detection for Software Fault Prone Modules," *International Journal of Software Engineering and Its Applications*, vol. 6**,** no**.** 1, pp. 75-86, 2012.

[19] Ramani R. and Jacob S., "Improved Classification of Lung Cancer Tumors Based on Structural and Physicochemical Properties of Proteins Using Data Mining Models," *Public Library of Science*, vol. 8, no. 3, pp. 58772, 2013.

[20] Ramani R. and Jacob S., "Prediction of P53 Mutants (Multiple Sites) Transcriptional Activity Based on Structural (2D and 3D) Properties," *Public Library of Science*, vol. 8, no. 2, pp. 55401, 2013.

[21] Ramani R., Kumar S., and Jacob S., "Predicting Fault-Prone Software Modules Using Feature Selection and Classification Through Data Mining Algorithms," *in Proceeding of IEEE International Conference on Computational Intelligence and Computing Research*, Coimbatore, pp. 1-4, 2012.

[22] Song Q., Jia Z., Shepherd M., Ying S., and Liu J., "General Software Defect-Proneness Prediction Framework," *IEEE Transactions on Software Engineering*, vol. 37, no. 3, pp. 356-370, 2011.

[23] Twala B., "Predicting Software Faults in Large Space Systems using Machine LearningTechniques," *Defence Science Journal*, vol. 61, no. 4, pp. 306-316, 2011.

[24] WEKA data mining toolkit, http://www.cs.waikato.ac.nz/~ml/weka, Last Visited 2014.

[25] Witten I. and Frank E., *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, 2005.

Geetha Raju is Associate Professor, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, Chennai, India. She has more than 15 years of teaching and research experience. Her areas of specialization include Data mining, Bioinformatics, Social Networks, Evolutionary Algorithms and Network Security. She has over 50 publications in International Conferences, Journals and books to her credit.

Shomona Jacob is Associate Professor, Department of Computer Science and Engineering, SSN College of Engineering, Chennai, India. She completed her Ph.D in the area of Biological and Clinical Data Mining at Anna University, Chennai. She has more than 25 publications in International Conferences and Journals to her credit. Her areas of interest include Data Mining, Bioinformatics, Machine Learning, and Artificial Intelligence.