Support Vector Machine with K-fold Cross Validation Model for Software Fault Prediction

International Journal of Pure and Applied Mathematics
Volume 118 No. 20 2018, 321-334
ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue
ijpam.eu

Support Vector Machine with K-fold Cross Validation Model for Software Fault Prediction

K Srujan Raju[1.1], M.Ramakrishna Murty[2], M Varaprasad Rao[1], Suresh C. Satapathy[3]
[1]CMRTC, Hyderabad, Telangana, India
ksrujanraju@gmail.com, vpr_m@yahoo.com

[2]ANITS, Visakhapatnam, AP, India
ramakrishna.malla@gmail.com

[3]PVPSIT, Vijayawada, AP, India
sureshsatapathy@gmail.com

Abstract: The most important goal of the software industry is to build high quality software products. Defective software modules lead to software system failure. The aim of reliable software is to minimize the number of failures that occur when software program runs. Software fault prediction is an important area to develop quality software. By using essential prediction metrics and historical fault data one can predict the software faults. An efficient software fault prediction model makes test effort easier, and at the same time improves quality of software and its consistency. In the literature various techniques have been proposed for fault prediction methods based on various metrics. But unfortunately none of the models satisfy cost reduction and software reliability.

This paper proposes a methodology for software fault prediction using support vector machine along with K-fold cross validation procedure. The proposed methodology is suitable for all software metrics for fault prediction with empirical validations used for standard data set in terms of cost reduction and software reliability. A short and pregnant description of the content and intent of the article. Please try to avoid mathematical symbols and special characters as much as possible. Make sure the heading of the article, the names of the authors and their affiliations are formatted as shown above. State the names of the city and country in the affiliations.

Keywords: Software fault, support vector machines, software metrics, software fault prediction

1 Introduction

In present days in the process of software development dealing with software faults is a foremost important task for software quality and reliability [2]. Software fault can be defined as it is a defect in the software code causes software failure while execute the code.

There are different types of faults viz., semantic faults, syntactic faults, communication faults, run time faults, communication faults and exceptions in the system modules. These faults lead to the software failure and affect the quality of the software. Therefore predict software faults in the early phases of software development process leads to develop quality software as well cost effective [1]. The main objective of the software fault prediction is to find faulty software modules before to delivery to the testing phase in the development process. Fault prediction just before the testing phase cannot useful, because most of the vital modules developed by that time and high amount of money already involved and need more efforts required to change the previous phases of the work[3][4]. Hence, the prediction of software faults at the initial stages of the software development improves the quality of the software and reduces the project cost. In general few software modules are reason for failure, which are having faults; identify and make them fault free in the early stages of the software development for produce quality software with less cost [5]. In the process of fault prediction, identify the fault-prone software modules and separate them from non-fault prone modules. In the prediction process also identify the relationship between software modules, which can control the damage of other non-fault modules. Recent past software fault prediction is an attractive area because it provides quality of the product and cost effective. The prediction details will share with the quality assurance team to fine down the testing pains to the module having more number of faults. Machine learning methods are playing crucial role for fault prediction apart from the traditional methods.

It is very difficult to make fault-free software because of the problem complexity, human thinking behaviours, and other resource constrains. Software failure is common and results in undesirable consequences, which can badly affect reliability, safety of the system. In general there are many modules in the software system, but a small number of modules are responsible for maximum failures [6]. Early detection of faulty modules makes the software cost effective but it also helps in the reduction of number of faults in the specified software module. Predict a fault in a software module is attractive because it provides the probability that number of faults will occur in the specified software module. Therefore predict and classify the software modules based on faulty and non-faulty after the coding phase in the software development life cycle. In the software fault prediction process prediction metrics play major role.

Fault Prediction Metrics

The metrics for the software fault prediction is classified into different groups. The important groups are class-level metrics, method-level metrics file-level metrics, and component-level metrics etc. [9] Method level metrics attentive on programming concepts and are acquired from the source code. According to the study the most frequently used metrics are CK. CK metrics measure exclusive aspects of the object oriented approach. These metrics also measure complexity of the design. Even no thresholds defined for the CK metrics [10]. Still, they can be used recognizing outlaying values.CK metrics have been correlated to fault-proneness, productivity, rework effort, design effort and maintenance while measuring the code. The utmost used suit of CK metrics are NOC(Number of children), DIT(Depth of Inheritance Tree), RFC(Response for Class), LCOM(Lack of Cohesion in Methods), CBO (Coupling between object classes) and WMC(Weighted Method for Class). Software metrics play a significant role at the process and final output levels in the software development.

2 Literature Review

The study related to software fault prediction is summarized in this section. Many research papers studied few of them mentioned here. In this study found that different categories of methods applied to predict software faults with using statistical methods, machine learning tools, soft computing techniques. As per the study, most of the researchers used basic statistical methods used for fault prediction and also observed few studies used machine learning methods. The development is moving towards evolutionary and swarm intelligent methods used to predict software faults.

Authors like [Afzal, Khoshgoftaar (11)] Afzal et al, Rathore and kumar and Khoshgoftaar et al investigated on genetic programming, decision tree regression and multilayer preceptor respectively, how these methods helped in the fault prediction process. [Alsmadi ,Najadat (11)] Alsmadi ,Najadat el al, intended to predict faulty modules to find out relationships among them [18]. They believed that attributes with high correlations among them could negatively influence each other.

In the recent studies on the software fault prediction have mentioned interestingly, either proposed new methods to increase the performance using new parameters and feature selection approaches. We found in some studies on new classification models constructed on relational association rule mining for software defect prediction. The multi dimensionality data set may affect the performance of the model, due to its size and complexity. The studies shows that to decrease the multi-dimensionality of the input, the relation between the features were analysed through Spearman's rank correlation coefficient in the pre-processing step helps to find the relationship between valid feature values.

[Khoshgoftaar, (11)] Khoshgoftaar et al, (2000) projected software quality by means of case-based reasoning and eight metrics for a command-control-communication technique developed. Several researchers proposed different various metrics, applied them on different classification models and compare metrics enactment in terms of F-measure, precision and recall.
In the literature it is also found there are separate metrics applied for object-oriented software models. Briand et al. supported an observed validation of object oriented measures to comprehend their relationships with fault prone modules using single variant and multivariate regression models[7][8].

In depth of the study, found that building fault prediction models using different learning methods are applied such as supervised, unsupervised. Study also revealed that most of the models build using supervised learning method rather than un-supervised. Supervised learning approach efficiency is based on the data set, because model can be trained through trained data [12][13]. Due to this some of the researcher also considered a semi-supervised approach for software fault prediction with constrained fault data set. [Catal (09)] C. Catal et al, (2009) have introduced an early fault prediction model by software metrics, and also examined the effects of data size, metrics, and feature selection techniques in the process of software fault prediction. Feature selection plays vital role in the process of fault prediction, which can remove irrelevant, noisy and inconsistent features from the dataset [16][19]. In the literature found that [Akalya, (12)C. Akalya Devi et al, (2012), mentioned how feature selection helps to enhance performance of learning models used for fault prediction.

3 Methodology

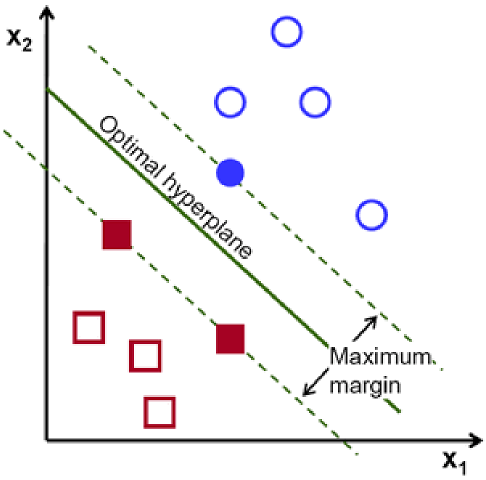
In this paper used various methodologies to implement proposed work. In the proposed work implemented feature extraction method, Support Vector Machines with K-fold cross validation method. The details of the methodologies are given in below sub-sections.

3.1 Feature extraction

Feature extraction also called as attribute selection, which play vital role in the classification model. Feature extraction is used to find appropriate features used for classification model to perform better. Here in this case the data set contains many attributes, which are all may not relevant for the fault prediction application. Attribute relevance analysis used to find either attributes are positively relevant, negatively relevant or independent [17]. To integrate feature selection method along with classification is called as wrapper approach. Feature selection wrapped with classifier due to dependency of each method. Most of the cases wrapper method performs better than the filter methods due to feature extraction process is optimized for classification method. But wrapper methods are performed poorly for high dimensional datasets in terms of time complexity [20].

3.2 Support Vector Machines

Support Vector Machine was invented by Vladimir N.Vapnik and Chervonenkis in the year 1963. It is a supervised machine learning algorithm which can be used for classification [11]. In this plot each data item in the n-dimensional space. Perform classification through the hyper-plane. The hyper-pane differentiates two classes in well separable manner. It means support vector machine is a discriminative classifier. Support vector machine operation is based on the discovery of the hyper-plane that provides the largest minimum distance to the training samples [19][20]. The optimal straight line or hyper-line maximizes the margin of the training sample. SVM classifier is having its own advantages like it works clear margin of separation, it works effectively in higher dimensional spaces and memory efficient. According to the software fault prediction application, it consider it as two class problem like faulty-modules and non-faulty models. The decision boundary should be as far distance from the data of two classes as possible.



For maximize margin treated with m as shown in the Fig 1and find the distance between the origin and the line $w^t x$=k, here k is k/||w||. Here m= 2/(||w||).

Equations of three parallel lines for the two class problem consider here.

$$w^T x + b = -1 \text{ – (1)}$$

$$w^T x + b = 1 \text{ – (2)}$$

$$w^T x + b = 0 \text{ – (3)}$$

Let us consider data set (x¬1,….xn) and let yi belongs to (1,-1) be the class labels of the data set. The decision boundary must classify all points exactly by using following equation (4).

yi(W^T xi+b)≥1 ∀i- (4)

The decision boundary can be initiated by resolving through constraint optimization problem.

$$minimize\ \frac{1}{2}||w||^2\text{- (5)}$$

The equation (5) can be performing subjected to equation (4) as shown in the fig1.

3.3 K-fold Cross Validation

Different cross validation methods are available in the literature for sample selection as training data set. The k-fold cross validation method subdivided actual samples into k equal sized subsamples. Each subsample is taken as the validation data for testing the classification model and repeat the process k times. The advantage with this method is over repeated random sub-sampling as training and validation for each for validation at least once. Here k is the unfixed parameter will be chosen by the user [14][15].

4 Proposed methodology

In the proposed model initially take input as data set, which is normally incomplete, noisy and inconsistency. Therefore apply pre-processing methods to eliminate above mentioned qualities in the data set. Generally in this data set concern applied feature extraction method to find appropriate attributes for the next process step.

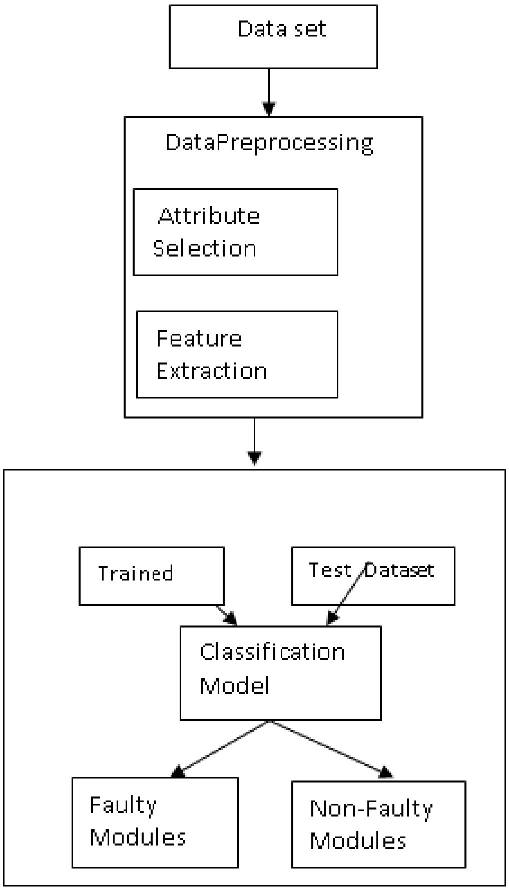


Fig 2: Proposed model working procedure

The proposed methodology follow the below simple steps to predict the software faults.

Input: Defected dataset ‘s’

Step 1: Pre-process the data set using feature extraction method.

Step 2: Apply K-fold cross validation process for divide the data set effectively for model training and testing.

Step 3: Use SVM classifier to classify.

4.1 Dataset Detail

Data set gathered from promise data repository, in this web link,.promise.site.uottawa.ca/SERepository/datasets-page.html[15]. Available data sets are namely CM1, JM1, KC1, KC2, PC1, and DATATRIEVE above mentioned web link. The details of the above mentioned data listed in the following table 1.

Table 1: Data set details used for experimentation

Name of the data set	Programming Language	Number of attributes	Number of instances	Percentage of defect modules
CM1	C	22	498	9.7%
JM1	C	22	10885	19.35%
KC1	C++	22	2109	15.55%
KC2	C++	22	522	20.53%
PC1	C	22	1109	6.99%
Data Trieve	C	9	130	8.46%

There are six data sets details mentioned in the table 1 related to the C and C++ programing language. The attributes of the first five data sets contains 22, last data set contains 9 attributes. Each data having good number of instances indicate software modules. According to the data set defect percentage also mentioned in the table1. The data set is having different software metrics, which can helps to find the faults in the existing software modules. In table 2 mentioned complete attribute details data set mentioned in the table1, except ‘DataTrieve’, which contain only 9 attributes. All data sets do not have missing values.

Table 2 :Details of attribute information of dataset, except Datatrieve

S.No.	Name of the attribute	Type of the metric	Description of the attributes
1	loc	McCabe	Line count of code
2	v(g)	McCabe	Cyclomatic complexity
3	ev(g)	McCabe	essential complexity
4	iv(g)	McCabe	design complexity
5	n	Halstead	total operators + operands
6	v	Halstead	Volume
7	l	Halstead	program length
8	d	Halstead	Difficulty

9	i	Halstead	Intelligence
10	e	Halstead	Effort
11	b	Halstead	Effort estimation
12	t	Halstead	Time estimator
13	loCoed	Halstead	line count
14	loComment	Halstead	count of lines of comments
15	LoBlank	Halstead	count of blank lines
16	LoCodeAndComment	--	Number of codes and comments
17	uniq_op	BaseHalstead	Unique operators
18	uniq_opnd	BaseHalstead	Unique operands
19	total_op	BaseHalstead	Total operators
20	total_opnd	BaseHalstead	Total operands
21	branchCount	--	Number of flow graphs
22	Problems	Goal field	Module is defected or Not

5 Experimentation process

Proposed methodology is implemented using object oriented programing language through Java, on hardware configuration of the processor of i3 with 4GB of RAM and operating system windows7. The data set, which mentioned in the table 1 is taken as input to the proposed model and perform experimentation with the following evolution parameters. The metrics mentioned in the section 1 applied on the data set KC1 and KC2, because these data sets related to object oriented programing. Experimentation performed on the above mentioned data set to evaluate the parameters viz. accuracy, sensitivity, completeness, Cut-off and F-measure. The above mentioned parameters are described below.

Accuracy is the correctness of the classifier on the given data set. It is measured classifier prediction correctness on the software defects. Classifier error rate may go increase the accuracy of the classifier go decrease.

Sensitivity is a positive rate, estimated on non-defective software modules. The sensitivity is measure through percentile. The software module does not have any defective then the sensitivity of the module is 100%.

Specificity is measured, how the classifier classifies defective modules. It also measured through percentage.

Precision denotes correctness of the classifier, how it measures by the proportion of recognized non-defect software modules and total predicted with non-defect software modules.

F-measure is a measure by calculating mean of precision and sensitivity.

6 Results Discussion

Experimentation performed on the data set mentioned in the table1, section 4.1. There are six different data sets listed, four related to the ‘c’ programming and two related to the ‘c++’, which is the object oriented programing language (KC1, KC2). CK-metrics used in the experimentation for the dataset KC1 and KC2 due to the data set related to object-oriented programming language. The metrics are NOC(Number of children), DIT(Depth of Inheritance Tree), RFC(Response for Class), LCOM(Lack of Cohesion in Methods), CBO (Coupling between object classes) and WMC(Weighted Method for Class).

The experimentation process is divided into two phases. In the first phase experimentation performed on KC1 and KC2 data set on the CK-matrices, which are mentioned above. In first phase of experiment, evolution parameters are considered were accuracy, sensitivity, specificity, precision and completeness.

Table 3: Classification model performance of KC1 data set.

Name of the CK-Metric	Accuracy	Sensitivity	Specificity	Precision	F-Measure
NOC	89.28	86.78	87.89	89.01	87.29

DIT	97.57	96.39	95.76	**96.90**	96.56
RFC	91.98	90.94	89.91	92.01	89.98
LCOM	95.67	93.78	93.64	94.82	95.07
CBO	**99.21**	**98.91**	**97.92**	95.97	**98.16**
WMC	89.78	85.98	88.18	86.48	86.77

Table 4: Classification model performance of KC2 data set.

Name of the CK-Metric	Accuracy	Sensitivity	Specificity	Precision	F-Measure
NOC	88.98	86.98	86.96	87.00	88.28
DIT	96.79	**97.98**	94.96	**95.94**	**97.06**
RFC	92.08	92.04	88.96	92.10	88.87
LCOM	94.87	91.89	94.04	93.89	94.17
CBO	**98.61**	97.91	**95.97**	94.90	96.80
WMC	88.49	85.97	86.38	87.08	85.82

In the second phase of experimentation performed on the data set CM1, JM1, PC1, DataTrieve with different metrics on the same evolution parameters. All the data sets contain 22 kinds of feature attributes except DataTrieve. In the experimentation process considered four McCabe measures feature, which can play vital role to consider faulty software modules. Feature generation performed based on the combination of different attributes. The combination procedure performed randomly while executing the proposed model. First phase of the experiments are shown in the Table 3 and 4 of KC1 and KC2 data set. According to the table 3 CBO performed well for both the data sets with all performance parameters.

Table 5: Classification performance of CM1 data set.

Feature Number	Accuracy	Sensitivity	Specificity	Precision	F-Measure
Feature1	84.58	81.48	86.96	85.60	85.87
Feature2	93.79	90.39	94.96	**93.85**	**96.86**
Feature3	93.08	89.71	88.96	89.87	85.82
Feature4	91.79	92.19	94.04	91.94	91.57
Feature5	**94.60**	**98.78**	**95.97**	93.67	89.87
Feature6	83.49	86.73	86.38	86.78	87.80
Feature7	86.78	76.89	76.81	75.34	79.23

Feature8	79.89	81.67	86.09	67.78	81.02
Feature9	79.01	76.89	78.02	80.14	76.92
Feature10	87.82	89.63	86.81	84.98	84.78
Feature 11	78.91	86.89	79.53	87.01	81.98
Feature 12	82.36	78.71	80.75	79.70	78.78

Table 6: Classification performance of JM1 data set.

Feature Number	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**F-Measure**
Feature1	85.18	82.08	85.91	84.70	83.77
Feature2	92.87	91.09	95.76	**94.83**	**95.76**
Feature3	91.85	88.01	87.90	88.07	84.72
Feature4	90.93	91.92	95.13	92.54	92.07
Feature5	**95.09**	**97.86**	**96.87**	94.57	90.77
Feature6	84.04	88.33	85.28	88.08	88.08
Feature7	85.89	77.08	77.71	76.24	78.22
Feature8	80.09	82.57	85.19	66.08	82.20
Feature9	80.07	75.79	77.22	81.04	75.82
Feature10	86.28	88.52	87.71	83.88	83.08
Feature 11	79.19	87.88	78.43	86.11	80.78
Feature 12	81.68	77.01	81.85	78.07	77.08

Table 7: Classification performance of PC1 data set.

Feature Number	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**F-Measure**
Feature1	83.08	82.08	85.06	84.50	84.07
Feature2	92.89	91.29	93.06	92.05	**96.96**
Feature3	94.28	88.51	89.06	88.77	84.72
Feature4	92.68	93.09	**95.24**	92.04	91.47
Feature5	**93.97**	**97.88**	94.37	**92.57**	88.77
Feature6	82.59	85.83	85.28	87.48	86.90

Feature7	85.68	75.79	77.91	74.74	78.43
Feature8	75.69	82.77	87.49	68.88	82.32
Feature9	78.11	77.69	77.32	82.24	77.82
Feature10	86.02	88.53	87.71	83.68	85.68
Feature 11	77.21	87.99	78.63	86.21	82.88
Feature 12	81.26	77.61	81.85	78.80	79.68

Table 8: Classification performance of DataTrievedata set.

Feature Number	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**F-Measure**
Feature1	82.28	81.18	84.06	82.40	83.47
Feature2	91.09	92.19	92.26	**93.15**	**97.06**
Feature3	95.18	89.41	88.56	87.87	83.02
Feature4	91.78	92.29	94.14	93.24	92.37
Feature5	**92.87**	**96.78**	**96.27**	91.17	89.67
Feature6	81.49	84.73	84.08	88.38	87.80

Table 9: Comparative Classification performance of DataTrievedata set.

Name of the data set	**Actual Percentage of defect modules**	**Proposed model produced defect percentage**	**Difference**	**Accuracy of the classifier**
CM1	9.7%	10.00%	0.3	97.89%
JM1	19.35%	19.30%	0.5	99.54%
KC1	15.55%	16.10%	0.55	96.58%
KC2	20.53%	20.54%	0.01	99.95%
PC1	6.99%	7.02%	0.03	99.57%
Data Trieve	8.46%	9.01%	0.55	93.89%

Results are proving that the proposed classifier accuracy above 97% for CM1 data set, 99.54% for JM1 data set , 96.58% for KC1 data set 99.95% for KC2 data set,99.57 for PC1 data set and 93.89 for the DataTreive data set.

7 Conclusion

In this work proposed support vector machine based classifier for the software fault prediction. The bench mark data set used from the promise data repository, which is available in the web. The data set can be distributed based on the 10-fold cross validation method. The experimental results proved that the proposed model performed best in terms of classification accuracy.The proposed SVM based used as binary classifier to classify the software modules either faulty or non-faulty. This work will be extended to hybridize with other classification model to get 100% accuracy.

References

[P.S. Bishnu, 12] Bishnu and V. Bhattacherjee, Member, IEEE "Software Fault Prediction Using Quad Tree-Based K-Means Clustering Algorithm"IEEETransactions on Knowledge and Data Engineering, Vol. 24, No. 6, June 2012

[M. Shepperd, 14] Shepperd, D. Bowes, and T. Hall, "Researcher bias: The use of machine learning in software defect prediction," Software Engineering, IEEE Transactions on, vol. 40, no. 6, pp. 603-616, 2014.

[Danijel Radjenovic, 13] Radjenovic, Marjan Hericko, Richard Torkar, Ales Zivkovic "Software fault prediction metrics: A systematic literature review" Information and software technology Elsevier 55(2013) 1397-1418.

[M.Unterkalmsteriner, 11] Unterkalmsteriner, T.Gorschek,A.Islam,C. Cheng, R.Permadi,R. Feldt "Evolution and measurement of software process improvement- a systematic literature review" ,IEEE Transaction on software Engineering ,X,1-29,2011.

[R.Karthikeyan, 14] C.Jothi Kumar, "Improved Reputation System for Wireless Sensor Networks (WSNS)", International Journal of Innovations in Scientific and Engineering Research (IJISER), Vol.1, No.3, pp.191-195, 2014.

[J.Ratzinger, 07] Ratzinger, M.Pinzger, H.Gall, EQ-Mine: Prdicting short-term defects for software evolution, Fundamental Approaches to software engineering, pp. 12-26, 2007.

[Y.Jiang, 07] Jiang, B.Cukic,T.Menzies, "Fault prediction using early lifecycle data", The 18th IEEE International Symposium on software Reliability(ISSRE-07),pp. 237-246.

[T.Kamiya, 99] Kamiya, S.Kusumoto, K.Inoue, "Prediction of fault-proneness at early phase in object-oriented development" ,Proceedings of the 2nd IEEE international Symposium on Object Oriented Real time Distributed Computing ISORC99,pp 253-258,1999.

[R.Satnawi,08] Satnawi, W.Li, "The effectiveness of software metrics in identifying error prone classes in post-release software evaluation process" Journal of systems and software vol. 81,pp 1868-1882,2008.

[EzgiiErturk, 15] EzgiiErturk, Ebru Akapinar Sezer "A comparision of some soft computing methods for software fault prediction" Expert system with applications ,Elsevier, pp1872-1879, vol.42, 2015.

[Chidamber Shyam, 98] Shyam, Kemerer Chris, Darcy David, "Managerial use of Metrics for Object-Oriented Software: an Exploratory Analysis", IEEE Transactions on software Engineering, August 1998.

[M. Ramakrishna Murty, 11] Ramakrishna, J.V.R.Murthy, Prasad Reddy P.V.G.D "Text document Classification Based on a Least Square Support Vector Machines with Singular Value Decomposition" International journal of Computer Application (IJCA),ISBN 978-93-80864-56-6, DOI 10.5120/3312-4540,Vol. 27 –No. 7, , pp 21-26,August 2011.

[Yuming, Z, 06] Yuming, and Hareton, L. "Empirical analysis of Object-Oriented Design Metrics for predicting high severity faults" IEEE Transactions on Software Engineering, 32(10), 771-784,2006.

[Gyimothy, T., 05] Gyimothy, Ferenc, R., Siket, I. "Empirical validation of object-oriented metrics on open source software for fault prediction", IEEE Trans. Software Engineering, 31 (10), 897 – 910, 2005.

[Olague, H., 07] Olague, Etzkorn, L., Gholston, S., and Quattlebaum, S. "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, 2007.

[Dataset, 13] Software Defect Dataset, Promise Repository, http://promise.site.uottawa.ca/SERepository/datasetspage.html, December 4, 2013.

[Emam, K. E., 01] Emam, Benlarbi, S., Goel, N. and Rai, S.N. "Comparingcase based reasoning classifiers for predicting high-risk softwarecomponents", Journal System Software, vol. 55, no. 3, pp. 301–320, 2001.

[Zhang, M. L.; 09] Zhang, Peña, J. M.; Robles, V. "Feature Selection forMulti-label Naive Bayes Classification. // InformationSciences", 179, 19(2009), pp. 3218-3229.

[Vandecruys, O.; 08] Vandecruys, Martens, D.; Baesens, B.; Mues, C.; DeBacker, M.; Haesen, R. "Mining Software Repositories for Comprehensible Software Fault Prediction Models" Journal of Systems and Software, 81, 5(2008), pp. 823-839.

[Xi Tan, 11] Tan, XinPeng, SenPan,Wenyun Zhao, "Assessing software quality by program Clustering and Defect Prediction" 18th Working Conference on Reverse Engineering 2011.

[Yue Jiang, J. L., 09] Jiang, "Variance Analysis in Software Fault Prediction Models," IEEE 20th International Symposium on Software Reliability Engineering, 2009.