

International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 2, Issue 10, May 2014)

## Software Fault Estimation using Fuzzy C-Means and Neuro-Fuzzy Classification

Jyoti Nagpal M. Tech. Scholar, Computer Science Dept. Poornima college of Engineering, Jaipur (Rajasthan.), India jyotibhugra2010@gmail.com

Abstract -The fault-proneness of a software module is the probability that the module contains faults and a software fault is a defect that causes software failures in an executable project. Early detection of fault prone software components enables verification experts to concentrate their time and resources on the problem areas of the software systems under development. In this paper, the performance comparison of a Software Fault Prediction System is done using two techniques; the first technique is Fuzzy C-Means clustering and another one is hybrid method which is combination of Fuzzy c-means clustering and Neural Networks approach (Neuro-Fuzzy). Both of the methods have been performed with the real time data set named PC1, taken from NASA MDP software projects. The performance is recorded on the basis of accuracy, net reliability, and **RMSE and MAE values.** 

*Keywords* – Fault-Proneness, Fuzzy C-Means, Neural Networks, Neuro-Fuzzy, NASA MDP.

#### I. INTRODUCTION

Faults are major problem in software systems that need to be resolved. Fault is a flaw that results in failure. We should have to know the clear difference between bug, fault and failure. Failure is deviation of software actions from the expected outcomes. A fault in software is a flaw that results in failure. Bug occurs when specified requirements of the software do not conform. There are many number of software having number of faults are delivered to the market [1].

A fault is a defect, an error in source code that causes failures when executed. A fault prone software module is the one containing more number of expected faults. Accurate prediction of fault prone modules enables the verification and validation activities focused on the critical software components.

A software fault is a defect that causes software failure in an executable product. For each execution of the software program where the output is incorrect, we observe a failure. Software engineers distinguish software faults from software failures. Dr. Ajay Khuteta Assosiate professor, Computer Science Dept. Poornima college of Engineering, Jaipur (Rajasthan.), India khutetaajay@poornima.org

Faults in software systems continue to be a major problem. Various systems are delivered to users with excessive faults. This is despite a huge amount of development effort going into fault reduction in terms of quality control and testing. It has long been recognized that seeking out fault-prone parts of the system and targeting those parts for increased quality control and testing is an effective approach to fault reduction. An inadequate amount of valuable work in this area has been carried out previously. Regardless of this it is difficult to identify a reliable approach to identifying faultprone software components. Using software complexity measures, the techniques build models, which categorize components as likely to contain faults or not.

Till now there are proposed numerous methods for data clustering methods. The algorithms provide a satisfying measure for the classification and mining of data. The software fault prediction is also now using the data clustering techniques because of the features and the functions they are expected to deliver. The clustering techniques till now have solved many purposes yet the satisfying result could not be guaranteed. In this research work, we have tried to modify the previous algorithms for the better results. We do not say that it is the end of research in this segment but it will definitely provide the new researchers with the scope to bring new considerations that could serve the future demands.

The main objective of this paper is to design a Software Fault Prediction System using Fuzzy cmeans clustering approach and a hybrid technique (combination of Fuzzy c-means clustering and Neural Networks approach). The results after classification of software fault data come in terms of certain efficiency parameters like Accuracy, Net Reliability, Mean Absolute Error, and Root Mean Squared Error in order to compare both approaches.



## International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 2, Issue 10, May 2014)

### II. SOFTWARE FAULT PREDICTION

A software system is the composition of number of modules dependent on each other. Any module with faults in its functionality adverse the output and lowers its reliability. In this scenario, the detection of faulty modules in early stage (development stage) is mandatory to minimize faults in operation phase. Hence, the systems are classified in two categories i.e. with faulty/nonfaulty modules in their testing phase. This classification diverts the focus to neutralize faulty sections to achieve high reliability and accuracy.

A software fault or error is reason of failure in execution stage. The error message at each stage of executing the program indicates the fault in programming. Generally speaking the errors are the logical errors in programming of software. Out of the heavy researches explored in this section a limited amount of valuable work is illustrated in literature section. The prediction models of software fault proneness technique estimate the amount of faulty modules in a program. The software metrics are the attributes for process, execution and product of the software system. Various other attributes like defect density, normalized work, fault proneness, maintain ability, reusability etc. determines the quality of software. Differennt fault prediction techniques are stated below:

**Decision Tree:** Decision trees are great and standard tools for classification and prediction. It produces classifiers in a form of structure of tree where each leaf node represents decision node. In this technique, classification starts from root of the tree and continues to move down until leaf node is reached. Classification helps in classifying faulty and non-faulty modules. Prediction helps in predicting faulty and non-faulty modules. Decision tress helps in developing fault prediction models that predicts faults.

*Neural Network:* Neural Network helps in recognizing patterns from the data set. An artificial neural network is composed of many artificial neurons that are interconnected together according to specific network architecture. The goal of the neural network is to transform the inputs into meaningful outputs. Adaptive Resonance Neural Network is generally used for defect prediction in software systems. It helps in identifying faulty modules very excellently. The benefit of using this technique is that it assists in decreasing effort and cost of developing software.

*Clustering approach:* Density Based Clustering is a clustering algorithm. It can be used to estimate the number of faulty and non-fault modules in software system. Clusters are defined as areas of higher density.

**Bagging method:** It creates base learners on many data subsets that are uniformly sampled from the original data, and then uses a linear combination to aggregate them. It is also referred as Bootstrap Aggregating. Combination technique can be majority voting. It also helps in identifying faulty and non-faulty modules with data sets that suffers from imbalance problem. This method can increase the performance of the defect data predictions.

*Naïve Bayes:* It is a classifier based on Bayes theorem used in software fault prediction. It resolves the several difficulties like spam classification (to predict whether email is spam or not), medical diagnosis (given list of symptoms, predict whether patient has cancer or not) and so on. This method can be used to predict faulty and non-faulty modules.

## III. METHODOLOGY

# 1. Find the structural code and requirement attributes

The first step is to find the structural code and requirement attributes of software systems i.e. software metrics. The real time defect data sets are taken from the NASA's MDP (Metric Data Program) data repository, [online] Available: http://mdp.ivv.nasa.gov.innamed as PC1 dataset which is collected from a flight software from an earth orbiting satellite coded in C programming language, containing 1107 modules and only 109 have their requirements specified. PC1 has 320 requirements available and all of them are associated with program modules. All these data sets varied in the percentage of defect modules, with the PC1 dataset containing the least number of defect modules.

## 2. Select the suitable metric values as representation of statement

The Suitable metric values used are fault and without fault attributes, we set these values in database create in MATLAB R2010 A as 0 and 1. Means 0 for data with fault and 1 for data without fault. The metrics in these datasets (NASA MDP dataset) describe projects which vary in size and complexity, programming languages, development processes, etc. When reporting a fault prediction modelling experiment, it is important to describe



## International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 2, Issue 10, May 2014)

the characteristics of the datasets. Each data set contains twenty-one software metrics, which describe product's size, complexity and some structural properties. We use only fault and without attributes to classify the selected NASA MDP PC1 dataset. Also the product metrics and product module metrics available in dataset which can also be use are the product requirement metrics are as follows:

- Module
- Action
- Conditional
- Continuance
- Imperative
- Option
- Risk\_Level
- Source
- Weak\_Phrase

The product module metrics are as follows:

- 1. Module
- 2. Loc\_Blank
- 3. Branch\_Count
- 4. Call\_Pairs
- 5. LOC\_Code\_and\_Comment
- 6. LOC\_Comments
- 7. Condition\_Count
- 8. Cyclomatic\_complexity
- 9. Cyclomatic\_Density
- 10. Decision\_Count
- 11. Edge\_Count
- 12. Essential\_Complexity
- 13. Essential\_Density
- 14. LOC\_Executable
- 15. Parameter\_Count
- 16. Global\_Data\_Complexity
- 17. Global\_Data\_Density
- 18. Halstead\_Content
- 19. Halstead\_Difficulty
- 20. Halstead\_Effort
- 21. Halstead\_Error\_EST
- 22. Halstead\_Length
- 23. Halstead\_Prog\_Time
- 24. Halstead\_Volume
- 25. Normalized\_Cyclomatic\_Complexity
- 26. Num\_Operands
- 27. Num\_Operators
- 28. Num\_Unique\_Operands
- 29. Num\_Unique\_Operators
- 30. Number\_Of\_Lines
- 31. Pathological\_Complexity
- 32. LOC\_Total

Figure 1 and 2 show flow diagrams for Fuzzy cmeans clustering approach and hybrid (Neuro-Fuzzy) approach respectively.



- Net Reliability
- Root Mean Squared Error

Figure 1: Flow diagram for Fuzzy C-means clustering Approach



Initially data is passed to Fuzzy C-Means clustering block in order to classify data according to attributes (true / false attributes)

Visit Database Site

The classified data from Fuzzy C-Means clustering block is given to Neural Network block to train neural network for these attributes in order to classify any fault in the system

Result in terms of:

- Accuracy
- Mean Absolute Error
- Net Reliability
- Root Mean Squared Error

Figure 2: Flow diagram for hybrid (Neuro-Fuzzy) technique

In this paper we have developed a software fault prediction module using two methods:

• Fuzzy c-means clustering (FCM) approach.



## International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 2, Issue 10, May 2014)

• Hybrid technique or Neuro-Fuzzy (Fuzzy c-means clustering (FCM) +Neural Network) technique.

PC1 software fault database is used available at NASA's research website. In the first method Fuzzy c-means clustering approach is used to detect any fault present in the data. In the hybrid method, Fuzzy c-means clustering approach is used to classify data according to attributes. The classified data from Fuzzy C-Means clustering is trained by neural network in order to predict any fault in the system.

## **Fuzzy C-Means Clustering**

Fuzzy C-Means iteratively moves the cluster centers to the "right" location within a data set. Objective function based fuzzy clustering algorithms such as the fuzzy c-means (FCM) algorithm has been used extensively for different tasks such as pattern recognition, data mining, and image processing and fuzzy modeling.

In general, cluster analysis refers to a broad spectrum of methods which try to subdivide a data set X into c subsets (clusters) which are pairwise disjoint, all nonempty, and reproduce X through union. The clusters then are termed a hard (i.e., non-fuzzy) c-partition of X.

## Parameters of the FCM Algorithm

*Number of Clusters:* The number of clusters c is the most important parameter, in the sense that the remaining parameters have less influence on the resulting partition. When clustering real data without any a priori information about the structures in the data, one usually has to make assumptions about the number of underlying clusters. The chosen clustering algorithm then searches for c clusters, regardless of whether they are really present in the data or not. Two main approaches to determining the appropriate number of clusters in data can be distinguished:

Α. Validity measures: Validity measures are scalar indices that assess the goodness of obtained partition. Clustering the algorithms generally aim at locating wellseparated and compact clusters. When the number of clusters is chosen equal to the number of groups that actually exist in the data, it can be expected that the clustering algorithm will identify them correctly. When this is not the case, misclassifications appear, and the clusters are not likely to be well separated and compact. Hence, most cluster validity

measures are designed to quantify the separation and the compactness of the clusters.

B. Iterative merging or insertion of clusters: The basic idea of cluster merging is to start with a sufficiently large number of clusters, and successively reduce this number by merging clusters.

Fuzziness Parameter: The weighting exponent m is a rather important parameter as well, because it significantly influences the fuzziness of the resulting partition.

Termination Criterion: The FCM algorithm stops iterating when the norm of the difference between U in two successive iterations is smaller than the termination parameter  $\epsilon$ . For the maximum norm  $max_{ik}(|\mu_{ik}^{(l)} - \mu_{ik}^{(l-1)}|)$ . The usual choice is  $\epsilon = 0.001$ , even though  $\epsilon = 0.01$  works well in most cases, while drastically reducing the computing times.

*Norm-Inducing Matrix:* The shape of the clusters is determined by the choice of the matrix A in the distance measure. A common choice is A = I, which gives the standard Euclidean norm:

$$D_{ik}^{2} = (z_{k} - v_{i})^{T} (z_{k} - v_{i})$$
(2)

Where  $v_i$  are ordinary means of the clusters.

Let  $\{x_1, x_2, ..., x_N\}$  be a set of N data objects represented by n-dimensional feature vectors.

$$\mathbf{x}_k = [x_{1k}, \dots, x_{nk}]^T \in \mathbb{R}^n \tag{3}$$

A set of N feature vectors is then denoted as a data matrix of  $n \times N$ .

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1N} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nN} \end{bmatrix}$$
(4)

A fuzzy clustering algorithm partitions the data X into M fuzzy clusters, forming a fuzzy partitioning. A fuzzy partition can be conveniently represented as a matrix, U, whose elements  $u_{ik} \in [0; 1]$  represents the membership degree of  $x_k$  in cluster '*i*'. Hence, the *i*<sup>th</sup> row of U contains values of the *i*<sup>th</sup> membership function in the fuzzy partition. Objective function based fuzzy clustering algorithms minimize an objective function of the type

$$I(X; U, V) = \sum_{i=1}^{M} \sum_{k=1}^{N} (u_{ik})^{m} d^{2}(\mathbf{x}_{k}, \mathbf{v}_{i})$$
(5)

Where,

$$V = [\mathbf{v}_1, \dots, \mathbf{v}_M]^T \in \mathbb{R}^n \tag{6}$$

International Journal Of Digital Application & Contemporary Research

## International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 2, Issue 10, May 2014)

is an M-tuple of cluster prototypes which have to be determined, and  $m \in (1; \infty)$  is a weighting exponent which determines the fuzziness of the clusters in order to avoid the trivial solution, constraints must be forced on U.

$$\sum_{i=1}^{M} u_{ik} = 1, \forall k \tag{7}$$

$$0 < \sum_{i=1}^{N} u_{ik} < N, \forall i \tag{8}$$

These constraints imply that the sum of each column of U is 1. Further, there may be no empty clusters, but the distribution of membership among the M fuzzy subsets is not constrained. The prototypes are typically selected to be idealized geometric forms such as linear varieties (e.g. FCV algorithm) or points (e.g. GK or FCM algorithms). When point prototypes are used, the general form of the distance measure is given by

$$d^{2}(\mathbf{x}_{k},\mathbf{v}_{i}) = (\mathbf{x}_{k} - \mathbf{v}_{i})^{T} A_{i}(\mathbf{x}_{k} - \mathbf{v}_{i})$$
(9)

Where the norm matrix  $A_i$  is a positive definite symmetric matrix. The FCM algorithm uses the Euclidian distance measure, i.e.  $A_i = I \forall_i$ , while the GK algorithm uses the Mahalonibisdistance, i.e.  $A_i = P_i^{-1}$  with  $P_i$  the covariance matrix of cluster i, and the additional volume constraint  $|A_i| = \rho_i$ .

The FCM algorithms are best described by recasting conditions in matrix-theoretic terms [3]. Towards this end, let U be a real  $c \times N$  matrix,  $U = [u_{ik}]$ . U is the matrix representation of the partition {  $Y_i$ } in the situation

$$u_i(\mathbf{y}_k) = u_{ik} = \begin{cases} 1; & \mathbf{y}_k \in Y_i \\ 0; & otherwise \end{cases}$$
(10)

$$\sum_{i=1}^{M} u_{ik} > 0 \quad for \ all \ i \tag{11}$$

$$\sum_{i=1}^{M} u_{ik} = 1 \quad for \ all \ k \tag{12}$$

In equation (10),  $u_i$  is a function such that:  $u_i: Y \rightarrow i$  $\{0, 1\}$ . In conventional models,  $u_i$  is the characteristic function of,  $Y_i$ : in fact,  $u_i$  and  $Y_i$ determine one another, so there is no harm in labelling u; the i<sup>th</sup> hard subset of the partition (It is unusual, of course, but is important in terms of understanding the term "fuzzy set"). Conditions of equations (11) and (12) are equivalent, so U is termed a hard c-partition of Y. Generalizing this idea, we refer to U as a fuzzy c-partition of Y when the elements of U are numbers in the unit interval [0, 1] that continue to satisfy both equations (11) and (12). The basis for this definition are c functions  $u_i: Y \to \{0, 1\}$  whose values  $u_i(y_k) \in$ [0,1] are interpreted as the grades of membership of the  $y_k$ s in the "fuzzy subsets"  $u_i$  of Y.

## Neural Networks

Neural Network approach contains following steps:

- Neural network creation
- Configuration
- Training



Figure 3: An example of a simple feed forward network [4]

Feed-forward ANNs (figure 3) as the name implies allow signals to travel in one way only; from input to output layer. There is no feedback loops or recurrent loops i.e. the output of any layer will not affect that output of the same layer. Feed-forward ANNs is also tend to be a straight forward networks that is associated with inputs outputs. They are highly used in pattern recognition and classification.

#### The Network layers

The general type of neural network consists of three groups of layers, or three groups of units: first one is a layer of "input" units which is always is connected to a second layer i.e. layer of "hidden" units, which is finally connected to a layer of "output" units. Figure 7 shows the representation of all layers of neural network.

## The Learning Process

There are basically two major categories of learning methods used for neural networks; Supervise learning methods and unsupervised learning method. In this research work we perform simulation of neural network under supervised learning mechanism. Supervised learning which work as an external teacher or guide, so that each output unit is told to perform what should be desired response to the respected input signals. Global information may be required during learning process. Error convergence is the main concern issue of supervise learning, i.e. the minimization of error between the desired and computed unit values of network. Here the main aim is to find a set of weights which minimizes or reduce the error up to precise level.



## International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 2, Issue 10, May 2014)

#### **Transfer Function**

The whole behavior of Neural Network totally depends on both the weights and the input-output function i.e. transfers function which is specified in the all units. There are basically three categories of Transfer Functions:

- Linear (or ramp)
- Threshold
- Sigmoid

For linear units or for the linear transfer function, the output activity is directly proportional to the total weighted output units. For threshold units or for threshold transfer function, the output unit outputs are set at one of two levels, which totally depending on whether the total input of output unit is greater than or less than some predefined threshold value. For sigmoid units or for sigmoid transfer function, the output varies or changes continuously but not linearly as the inputs of input unit changes.

#### IV. SIMULATION AND RESULTS

Simulation is carried out using MATLAB 2010a:



Figure 4: Graphical User Interface (GUI) for proposed work



Figure 5: Input PC1 dataset with attributes (fault and without fault)



Figure 6: Input PC1 dataset with fault attributes when separating fault attributes from input data



separating without fault attributes from input data



Figure 8: Data Input (PC1 database without attributes)



International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 2, Issue 10, May 2014)



Figure 9: Membership function plot





Figure 10: Membership function plot after Fuzzy logic

Figure 11: Membership function plot after Neuro-Fuzzy approach

Evaluation Parameter	Fuzzy C- Means Clustering	Hybrid (Adaptive Neuro-Fuzzy) technique
Accuracy	75	87
Net Reliability	60.07	47.20
Mean Absolute Error (MAE)	0.25	0.13
Root Mean Squared Error (RMSE)	0.0833	0.0194

Table 1: Performance comparison for Fuzzy c-means and proposed (Adaptive-Neuro-Fuzzy) technique

The disadvantage of the methods based on genetic algorithms is that the number of fuzzy sets must be informed. Genetically optimized fuzzy clustering method [9] have high computational complexity and large time spending.

#### V. CONCLUSION

In this paper, a software Fault Prediction System is implemented using Fuzzy C-Means clustering and hybrid (Neuro-Fuzzy) techniques. Performance of both the techniques is discussed in the comparative analysis in order to predict level of impact of faults in NASA's public domain defect dataset. Predicting faults in the software life cycle can be used to improve software process control and achieve high software reliability. On the basis of evaluation parameters, it was found that the hybrid technique shows more accuracy and less errors as compared to Fuzzy C-Means clustering method.

### REFERENCES

- Jaakkola T., and Haussler D., "Exploiting generative models in discriminative classifiers", In Advances in Neural Information Processing Systems 1, MIT Press, pp. 487–493, 1998.
- [2] Kazama J., and Tsujii J., "Evaluation and extension of maximum entropy models with in equality constraints", Proceedings of 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP2003), pp. 137–144, 2003.
- [3] Fuzzy C-means (FCM) algorithm, online available at: <u>http://hayoungkim.tistory.com/20</u>.
- [4] Christos Stergiou and Dimitrios Siganos, "Neural Networks" Report online available at: http://www.doc.ic.ac.uk/~nd/surprise\_96/journal/vol 4/cs11/report.html.
- [5] G. Pai, "Empirical analysis of Software Fault Content and Fault Proneness Using Bayesian



## International Journal of Digital Application & Contemporary research Website: www.ijdacr.com (Volume 2, Issue 10, May 2014)

Methods", IEEE Transactions on software Engineering, 33(10), pp. 675-686, 2007.

- [6] Arvinder Kaur, Ruchika Malhotra, "Application of Random Forest in Predicting Fault-Prone Classes", 2008 International Conference on Advanced Computer Theory and Engineering ICACTE 2008, Pukhet, pp. 37-43, 2008.
- [7] Anil Kumar Singh, Rajkumar Goel and Pankaj Kumar, "Comparative Analysis of Accuracy Prediction using Fuzzy C-Means and KNN Clasiffier", International Journal of Digital Application & Contemporary Research (IJDACR), ISSN: 2319-4863, Vol. 2, Issue 7, February 2014.
- [8] Aditi Sanyal, Balraj Singh, "A Systematic Literature Survey on Various Techniques for Software Fault Prediction", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), ISSN: 2277 128X, Vol. 4, Issue 1, January 2014.
- [9] Saurabh Bhattacharya, Dr. Sourabh Rungta and Naresh Kar, "Software Fault Prediction using Fuzzy Clustering & Genetic Algorithm", International Journal of Digital Application & Contemporary Research (IJDACR), ISSN: 2319-4863, Vol. 2, Issue 5, December 2013.
- [10] Kriti Purswani, Pankaj Dalal, Dr. Avinash Panwar and Kushagra Dashora, "Software Fault Prediction Using Fuzzy C-Means Clustering and Feed Forward Neural Network", International Journal of Digital Application & Contemporary Research (IJDACR), ISSN: 2319-4863, Vol. 2, Issue 1, July 2013.
- [11] R. Sathyaraj, S. Prabu, "A survey Quality based Object Oriented Software Fault Prediction", International Journal of Engineering and Technology (IJET), Vol. 5 No 3 Jun-Jul 2013.
- [12] Nurudeen Sherif, Nurudeen Mohammed, "Using Fuzzy Clustering and Software Metrics to Predict Faults in large Industrial Software Systems" IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, ISSN: 2278-8727, Volume 13, Issue 6, PP 32-36, Jul. - Aug. 2013.
- [13] Karpagavadivu. K, Maragatham. T, Dr. Karthik. S, "A Survey of Different Software Fault PredictionUsing Data Mining Techniques Methods", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), ISSN: 2278 – 1323, Vol.1, Issue 8, October 2012.
- [14] Atul Bisht, Amanpreet Singh Brar and Parvinder S. Sandhu, "Prediction of Faults in Open Source Software Systems Using FCM", International Conference on Computer Graphics, Simulation and Modeling (ICGSM'2012), Pattaya (Thailand), July 28-29, 2012.
- [15] Amandeep Kaur, Arjan Singh, Baljit Singh, "Design of Hybrid Neural Network Model for Quality Evaluation of Object Oriented Software Modules", International Journal of Engineering Research and Development (IJERD), ISSN: 2278-067X, Vol. 2, Issue 5, July 2012.
- [16] Supreet Kaur, and Dinesh Kumar, "Software Fault Prediction in Object Oriented Software Systems Using Density Based Clustering Approach". International Journal of Research in Engineering and Technology (IJRET), ISSN: 2277-4378, Vol. 1, No. 2, March 2012.
- [17] Cagatay Catal, "Performance Evaluation Metrics for Software Fault Prediction Studies", Istanbul Kultur University, Department of Computer Engineering, Atakoy Campus, 34156, Istanbul, Turkey

- [18] Parvinder S. Sandhu, Sheena Singh, Neha Budhija, "Prediction of Level of Severity of Faults in Software Systems using Density Based Clustering" 2011 International Conference on Software and Computer Applications IPCSIT vol.9 IACSIT Press, Singapore, 2011.
- [19] Giuseppe Scanniello, Carmine Gravino, Andrian Marcus, Tim Menzies, "Class Level Fault Prediction Using Software Clustering". Online available at: http://www2.unibas.it/gscanniello/Clustering and Fault Prediction.
- [20] Neeraj Mohan, Parvinder S. Sandhu, and Hardeep Singh, "Impact of Faults in Different Software Systems: A Survey" World Academy of Science, Engineering and Technology, 2009.
- [21] Thomas J. Ostrand and Elaine J. Weyuker, "A Tool for Mining Defect Tracking Systems to Predict Fault-Prone Files", 1st international workshop on mining software repositories, pp. 85-89, 2005.
- [22] Brian Randell, "Facing Up to Faults", The Computer Journal, Vol. 43, January 2000.
- [23] Supreet Kaur, Dinesh Kumar, "Quality Prediction of Object Oriented Software Using Density Based Clustering Approach", IACSIT International Journal of Engineering and Technology, Vol.3, No.4, August 2011.