World Academy of Science, Engineering and Technology
International Journal of Computer and Systems Engineering
Vol:4, No:7, 2010

# A Subtractive Clustering Based Approach for Early Prediction of Fault Proneness in Software Modules

Ramandeep S. Sidhu, Sunil Khullar, Parvinder S. Sandhu, R. P. S. Bedi, Kiranbir Kaur

***Abstract*—**In this paper, subtractive clustering based fuzzy inference system approach is used for early detection of faults in the function oriented software systems. This approach has been tested with real time defect datasets of NASA software projects named as PC1 and CM1. Both the code based model and joined model (combination of the requirement and code based metrics) of the datasets are used for training and testing of the proposed approach. The performance of the models is recorded in terms of Accuracy, MAE and RMSE values. The performance of the proposed approach is better in case of Joined Model. As evidenced from the results obtained it can be concluded that Clustering and fuzzy logic together provide a simple yet powerful means to model the earlier detection of faults in the function oriented software systems.

***Keywords*—**Subtractive Clustering, Fuzzy Inference System, Fault Proneness.



## I. Introduction

MANY 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. A limited amount of valuable work in this area has been carried out previously. Despite this it is difficult to identify a reliable approach to identifying fault-prone software components. Prediction of fault-prone modules provides one way to support software quality engineering through improved scheduling and project control. Quality of software is increasingly important and testing related issues are becoming crucial for software. Although there is diversity in the definition of software quality, it is widely accepted that a project with many defects lacks quality. Methodologies and techniques for predicting the testing effort, monitoring process costs, and measuring results can help in increasing efficiency of software testing. Being able to measure the fault-proneness of software can be a key step towards steering the software testing and improving the effectiveness of the whole process.

Ramandeep S. Sidhu is doing M.Tech. CSE, RIEIT, Rail Majra, Punjab, India.

Sunil Khullar is associated with RIEIT, Rail Majra, Punjab, India

Dr. R. P.S. Bedi is working as Joint Registrar, Punjab Technical University, Jalandhar, India

Parvinder S. Sandhu is associated with Rayat-Bahra Institute of Engineering & Bio-Technology, Sahauran, Mohali (India).

Kiranbir Kaur is associated with Guru Nanak Dev University, India

Predictive modeling is the process by which a model is created or chosen to try to best predict the probability of an outcome. The objective of a fault-proneness model is to identify faulty classes and focus testing effort on them.

Fault-proneness of a software module is the probability that the module contains faults. A correlation exists between the fault-proneness of the software and the measurable attributes of the code (i.e. the static metrics) and of the testing (i.e. the dynamic metrics). Early detection of fault-prone software components enables verification experts to concentrate their time and resources on the problem areas of the software system under development. Early lifecycle data includes metrics describing unstructured textual requirement and static code metrics. Various researches show that use of static code metrics (such as Halstead complexity, Cyclomatic complexity, McCabe's complexity etc.) to measure quality is inefficient. The use of single features of software to predict faults is uninformative. Fenton offers an example where the same program functionality is achieved using different programming language constructs resulting in different static measurements for that module [1]. Fenton uses this example to argue the uselessness of static code attributes. However, where single features fail, combinations can succeed [2]. Hence combinations of static features extracted from requirements and code can be good predictors for identifying modules that actually contains fault.

## II. Clustering

As a broad subfield of Fault Prediction, clustering is concerned with the design and development of algorithms and techniques that allow division of data in to different groups. Clustering means to assign a set of observations in to different groups (known as clusters), so that the observations are same in some sense. At a general level, there are two types of clustering: distance based and conceptual clustering. Distance based clustering divides the data in to subsets on the basis of distance. Conceptual clustering, cluster the data on the basis of the similar concept the data will have.

An important component of a clustering algorithm is the distance measure between data points. If the components of the data instance vectors are all in the same physical units then it is possible that the simple Euclidean distance metric is sufficient to successfully group similar data instances. It is the ordinary distance between two points that one would measure with a ruler, which can be proven by repeated application of the Pythagorean theorem. The major focus of clustering

research is to extract information from data automatically, by computational and statistical methods. Hence, clustering is closely related to data mining and statistics.

Many clustering methods aim at finding a single partition of the collection of items into clusters. However, obtaining a hierarchy of clusters can provide more flexibility and other methods rather focus on this. A partition of the data can be obtained from a hierarchy by cutting the tree of clusters at some level. Most clustering methods were developed for numerical data, but some can deal with categorical data or with both numerical and categorical data [3].

The degree of membership of a data item to a cluster is either in [0, 1] if the clusters are fuzzy or in {0, 1} if the clusters are crisp. For fuzzy clusters, data items can belong to some degree to several clusters that don't have hierarchical relations with each other. This distinction between fuzzy and crisp can concern both the clustering mechanisms and their results. Crisp clusters can always be obtained from fuzzy clusters. Clusters can be seen either as distant compact sets or as dense sets separated by low density regions. Unlike density, compactness usually has strong implications on the shape of the clusters, so methods that focus on compactness should be distinguished from methods that focus on the density. Clustering denotes changes in a system that enables a system to do the same task more efficiently the next time. Clustering is a method of unsupervised learning, in which one seeks to determine how the data are organized [3]. Clustering algorithms can be:

### *A. Hierarchical*

A hierarchical algorithm creates a hierarchy of clusters which may be represented in a tree structure called a dendrogram. The root of the tree consists of a single cluster containing all observations, and the leaves correspond to individual observations. In hierarchical clustering algorithm, a valid metric may be used as a measure of similarity between pairs of observations. Algorithms for hierarchical clustering are generally either agglomerative, in which one starts at the leaves and successively merges clusters together; or divisive, in which one starts at the root and recursively splits the clusters [4].

### *B. Partitional*

Partitional algorithms typically determine all clusters at once. These algorithms divide data in to independent clusters on the basis of distance measures [4]. A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset.

K-Means is an unsupervised clustering technique used to classify data in to K clusters. It is partitional clustering approach, each cluster is associated with a centroid (center point), each point is assigned to the cluster with the closest centroid, Number of clusters, K, must be specified [5].

Fuzzy C-Means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters. It processes n vectors in p-space as data input, and uses them, in conjunction with first order necessary conditions for minimizing the FCM objective functional, to obtain estimates for two sets of unknowns. FCM clustering is used to build fuzzy rule bases for fuzzy systems design; and there are numerous applications of FCM in virtually every major application area of clustering [6].

### *C. Spectral*

Spectral clustering techniques make use of the spectrum of the similarity matrix of the data to perform dimensionality reduction for clustering in fewer dimensions [4]. The main requirements that a clustering algorithm should satisfy are scalability; dealing with different types of attributes; discovering clusters with arbitrary shape; minimal requirements for domain knowledge to determine input parameters; ability to deal with noise and outliers; insensitivity to order of input records; high dimensionality; interpretability and usability [7].

Clustering techniques create applications that are rugged, self-adapting, easier to maintain and often more fault tolerant than conventional systems. An adaptive feedback loop can tailor a system to changes in enterprise policies and make it more resilient. Clustering deals with the issue of how to build programs that improve their performance at some task through clustered data [8].

In this present thesis work, Subtractive clustering based Fuzzy Inference technique is experimented on different models and comparative analysis is performed for the prediction of faults in software systems.

## III. Methodology Proposed

The proposed methodology will consist of the following steps:

- First of all, find the requirement phase and structural code attributes of software systems.
- Select the suitable metric values as representation of statement
- Collect the metric data of requirement phase and structural code attributes
- Perform the join of the structural and requirement metric data and obtain the combined data.
- Analyze, refine metrics and normalize the metric values.
- Find the suitable algorithm for clustering of the software components into faulty/fault-free systems.

Clustering can be a very effective technique to identify natural groupings in data from a large data set, thereby allowing concise representation of relationships embedded in the data. In our study, clustering allows us to group software modules into faulty and non-faulty categories hence allowing for easier understandability.

Fuzzy logic is an effective paradigm to handle imprecision. It can be used to take fuzzy or imprecise observations for inputs and yet arrive at crisp and precise values for outputs. Also, the Fuzzy Inference System (FIS) is a simple and

commonsensical way to build systems without using complex analytical equations.

Here, fuzzy logic will be employed to capture the broad categories identified during clustering into a Fuzzy Inference System (FIS). The FIS will then act as a model that will reflect the relationship between the different input parameters [12].

### A. Subtractive clustering

Subtractive clustering, [13], is a fast, one-pass algorithm for estimating the number of clusters and the cluster centers in a set of data.

This means that the computation is now proportional to the problem size instead of the problem dimension. However, the actual cluster centers are not necessarily located at one of the data points, but in most cases it is a good approximation, especially with the reduced computation this approach introduces.

Since each data point is a candidate for cluster centers, a density measure at data point xi is defined as:

$$D_i = \sum_{j=1}^{n} \exp\left( - \frac{\left\| X_i - X_j \right\|^2}{\left( r_a / 2 \right)^2} \right) \quad (1)$$

Where *ra* is a positive constant representing a neighborhood radius. Hence, a data point will have a high density value if it has many neighboring data points.

The first cluster center xc1 is chosen as the point having the largest density value Dc1. Next, the density measure of each data point xi is revised as follows:

$$D_i = D_i - D_{c_1} \exp\left( - \frac{\left\| X_i - X_{c_1} \right\|^2}{\left( r_b / 2 \right)^2} \right) \quad (2)$$

Where *br* is a positive constant which defines a neighborhood that has measurable reductions in density measure. Therefore, the data points near the first cluster center xc1 will have significantly reduced density measure. After revising the density function, the next cluster center is selected as the point having the greatest density value. This process continues until a sufficient number of clusters are obtained [11].

### B. Fuzzy Inference System

FIS is composed of inputs, outputs and rules. Each input and output can have any number of membership functions. The rules dictate the behavior of the fuzzy system based on inputs, outputs and membership functions. Fuzzy Inference System is constructed to capture the position and influence of each cluster in the input space.

Each input attribute and output attribute has as many membership functions as the number of clusters that subtractive clustering algorithm has identified. The number of rules will be equals the number of clusters.

If there are three rules generated then significance of the first rule is that it succinctly maps cluster 1 in the input space to cluster 1 in the output space. Similarly the other two rules map cluster 2 and cluster 3 in the input space to cluster 2 and cluster 3 in the output space. If a data point is closer to the first cluster, or in other words having strong membership to the first cluster, is fed as input to fuzzy inference system then rule1 will fire with more firing strength than the other two rules. Similarly, an input with strong membership to the second cluster will fire the second rule will with more firing strength than the other two rules and so on.

The output of the rules (firing strengths) are then used to generate the output of the FIS through the output membership functions [12].

## IV. Implementation

Implementing the model and test the performance of the model using following criteria:

The comparisons are made on the basis of the more accuracy and least value of MAE and RMSE error values. Accuracy value of the prediction model is the major criteria used for comparison. The mean absolute error is chosen as the standard error. The technique having lower value of mean absolute error is chosen as the best fault prediction technique.

### A. Mean Absolute Error

Mean absolute error, MAE is the average of the difference between predicted and actual value in all test cases; it is the average prediction error [9]. The formula for calculating MAE is given in equation shown below:

$$\frac{\left|a_1 - c_1\right| + \left|a_2 - c_2\right| + ... + \left|a_n - c_n\right|}{n} \quad (3)$$

### B. Root Mean Squared Error

RMSE is frequently used measure of differences between values predicted by a model or estimator and the values actually observed from the thing being modeled or estimated [9]. It is just the square root of the mean square error as shown in equation given below:

$$\sqrt{\frac{\left(a_1 - c_1\right)^2 + \left(a_2 - c_2\right)^2 + ... + \left(a_n - c_n\right)^2}{n}} \quad (4)$$

The mean-squared error is one of the most commonly used measures of success for numeric prediction. This value is computed by taking the average of the squared differences between each computed value and its corresponding correct value. The root mean-squared error is simply the square root of the mean-squared-error. The root mean-squared error gives the error value the same dimensionality as the actual and predicted values.

## V. Results And Discussion

So, the real-time defect data sets used are taken from the NASA's MDP (Metric Data Program) data repository, available online at [10]. The CM1 data is obtained from a spacecraft instrument, written in C, containing approximately 505 modules. The PC1 data is collected from a flight software system coded in C, containing 1107 modules.

After polishing of data the metrics used are product requirement metrics and product module metrics and the combination of requirement and module metrics.

The output of a software module is considered to be Error count. Error count metric defines the number of defects that can occur in a module. A zero error count for a software module specifies that module is error free and a non-zero number in this metric specifies the number of faults that can occur in a module and that module is said to be fault prone.

Figure 1 shows the CM1 Graphical representation of details of the output of software of the same metric from CM1 dataset.

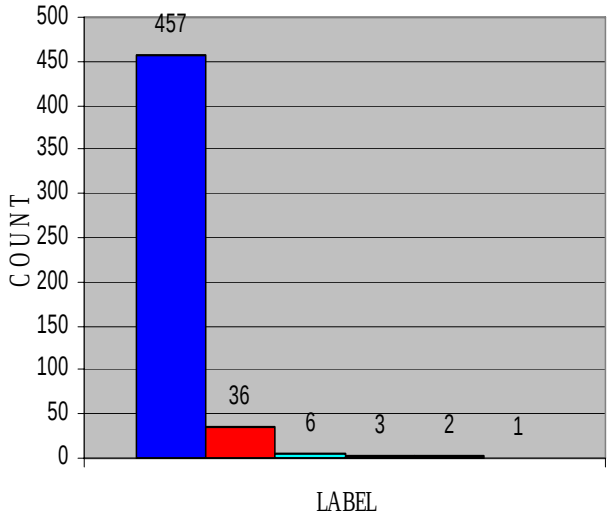


Fig. 1 CM1 Graphical details of the output of software Product Module metric

Figure 2 shows the CM1 graphical details of the output of software of combined metrics extracted from product module metrics and product requirement metrics where label i.e. error count is meant for output and is equal to the number of errors and the count tells the number of occurrences of that label in the CM1 data set.

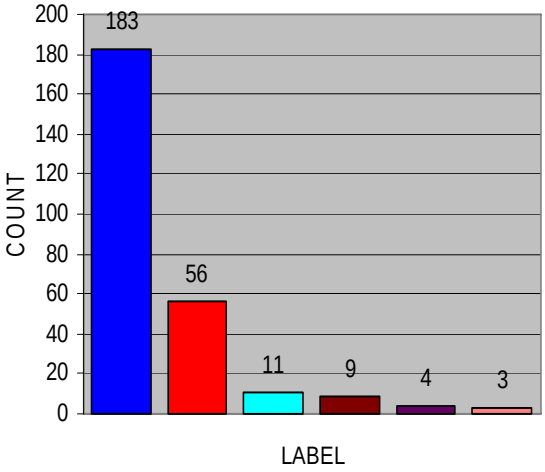


Fig. 2 CM1 Graphical details of the output of software of Combination metrics

Figure 3 shows the PC1 graphical details of the output of software Product Requirement metric where label and the count tells the number of occurrences of that label in the PC1 data set.

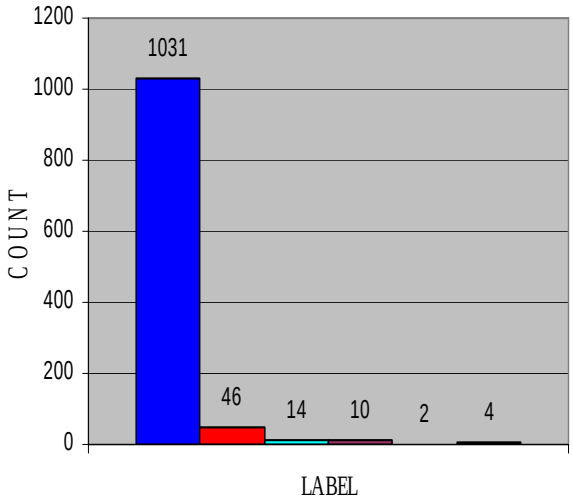


Fig. 3 PC1 Graphical details of the output of software Product Module metric

Figure 4 shows the PC1 details of the output of software of combination of product module metric and product requirement metric where label and the count tells the number of occurrences of that label in the PC1 data set.

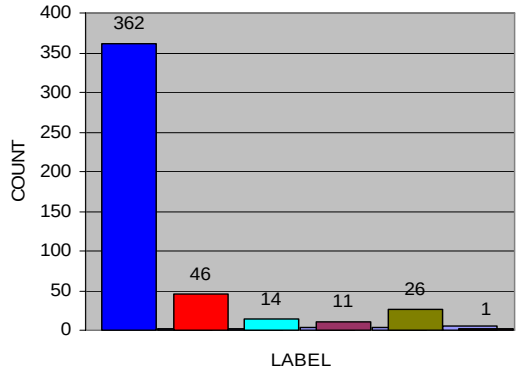


Fig. 4 PC1 Graphical details of the output of software of Combination metric

The next step is to use subtractive clustering algorithm based fuzzy Inference system approach for classification of the software components into faulty/fault-free categories. The proposed algorithm is applied on both CM1 and PC1 datasets and performance is measured on both Code based and Joined Model as shown in table 1.

TABLE I (A) PERFORMANCE OF THE SUBTRACTIVE CLUSTERING BASED FUZZY INFERENCE SYSTEM

| **Performance Criteria** | **DataSet** | | | |
|---|---|---|---|---|
| | **PC1** | | **CM1** | |
| | *Code Model* | *Joined Model* | *Code Model* | *Joined Model* |
| *Accuracy* | 92.8765 | 96.0168 | 88.9558 | 100 |
| *MAE* | 0.0694 | 0.0199 | 0.1024 | 2.4 e-016 |
| *RMSE* | 0.2618 | 0.0998 | 0.3137 | 1.03 e-015 |

## VI. CONCLUSION

Predicting faults early in the software life cycle can be used to improve software process control and achieve high software reliability. So, in this study, subtractive clustering based fuzzy inference system approach is used for early detection of faults in the function oriented software systems. This approach has been tested with real time defect datasets of NASA software projects named as PC1 and CM1. Both the code based model and joined model of the datasets are used for training and testing of the proposed approach.

In case of PC1 dataset, the joined or combine model of requirement metrics and code based metrics is showing better prediction capability with 96.01%, 0.0199 and 0.0998 as Accuracy, MAE and RMSE values.

Similarly, in case of CM1 dataset, the joined or combine model of requirement metrics and code based metrics is again showing better prediction capability with 100%, 2.4e-16 and 1.03 e-15 as Accuracy, MAE and RMSE values respectively. So, the performance of the proposed approach is better in case of CM1 dataset's Joined Model.

As evidenced from the results obtained it can be concluded that Clustering and fuzzy logic together provide a simple yet powerful means to model the earlier detection of faults in the function oriented software systems.

## REFERENCES

[1] Fenton N.E. and Pfleeger S.L. (1997), "Software Metrics: A Rigorous and Practical Approach". PWS publishing Company: ITP, Boston, MA, 2nd edition, pp.132-145.
[2] Jiang Y., Cukic B. and Menzies T. (2007), "Fault Prediction Using Early Lifecycle Data". ISSRE 2007, the 18th IEEE Symposium on Software Reliability Engineering, IEEE Computer Society, Sweden, pp. 237-246.
[3] Nizar Grira, Michel Crucianu, Nozha Boujemaa, "Unsupervised and Semi-supervised Clustering: a Brief Survey", A Review of Machine Learning Techniques for Processing Multimedia Content, Report of the MUSCLE European Network of Excellence (6th Framework Programme), 2005 URL: www.rocq.inria.fr/~crucianu/src/BriefSurveyClustering.pdf
[4] http://en.wikipedia.org/wiki/Cluster_analysis
[5] Toon Calders, "Data Mining Clustering", URL: wwwis.win.tue.nl/~tcalders/teaching/.../slides/DM09-07-Clustering.pdf
[6] http://www.scholarpedia.org/article/Fuzzy_C-means_cluster_analysis
[7] home.dei.polimi.it/matteucc/Clustering/tutorial_html/
[8] scianta.com/technology/machinelearning.htm
[9] Challagulla, V.U.B. , Bastani, F.B. , I-Ling Yen , Paul,( 2005) "Empirical assessment of machine learning based software defect prediction techniques", 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems, WORDS 2005, 2-4 Feb 2005, pp. 263-270.
[10] http://mdp.ivv.nasa.gov
[11] Khaled Hammouda, "A Comparative Study of Data Clustering Techniques", SYDE 625: Tools of Intelligent Systems Design. Course Project. Unpublished Aug 2000
[12] www.mathworks.com
[13] S. Chiu, "Fuzzy Model Identification Based on Cluster Estimation," J. of Intelligent & Fuzzy Systems, Vol. 2, No. 3, 1994.