ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Special Issue 2, April 2014

Predicting the Software Fault Using the Method of Genetic Algorithm

Mrs.Agasta Adline[1], Ramachandran .M[2]

Assistant Professor (Sr.Grade), Easwari Engineering College, Chennai, Tamil Nadu, India.[1]

PG student, Software Engineering, Easwari Engineering College, Chennai, Tamil Nadu, India.[2]

ABSTRACT — Software metrics and fault data belonging to a previous software version are used to build the software fault prediction model for the next release of the software. However there are certain cases when previous fault data are not present. In other words predicting the fault-proneness of program modules when the fault labels for modules are unavailable is a challenging task frequently raised in the software industry. There is need to develop some methods to build the software fault prediction model based on supervised learning which can help to predict the fault–proneness of a program modules when fault labels for modules are not present. One of the methods is use of classification techniques. Supervised techniques like classification may be used for fault prediction in software modules, more so in those cases where fault labels are not available. In this study, we propose a Genetic algorithm based software fault prediction approach for classification.

KEYWORDS: fault proneness, genetic algorithm, supervised techniques, classification.

I.INTRODUCTION

Software quality models are useful tools toward achieving the objectives of a software quality assurance initiative. A software quality model can be used to identify program modules that are likely to be defective. Subsequently, the limited resources allocated for software quality inspection and improvement can be targeted toward only those program modules, achieving cost-effective resource utilization. A software quality estimation model allows the software development team to track and detect potential software defects relatively early on during development, which is critical to many high-assurance systems. Software fault is a mistake in the coding that may lead to software to behave not in the intended way and may result in error and hence software failure. A software fault is a possibly recoverable error that occurred because of a programming error. Such faults are usually detected by hardware, and sent to the appropriate software handler for processing Fault prediction will give one more chance to the development team to retest the modules or files for which the defectiveness probability is high. By spending more time on the defective modules and no time on the non-defective ones, the resources of the project would be utilized better and as a result, the maintenance phase of the project will be easier for both the customers and the project owners. The perfect prediction of where faults are likely to occur in code can help direct test effort, reduce costs and improve the quality of software. In software development practice, however, various practical issues limit the availability of defect data for modules in the training data. For example, an organization may have not recorded or collected software defect data from previous releases or similar projects. In addition, since the organization may not have experience developing a similar system, the use of software measurement and defect data of previous projects for modeling purposes is inappropriate. In current times, where globalization of technology has gained momentum, distributed software development is not uncommon. Under such conditions, software defect data may not be collected by all development sites depending on the organizational structure and resources of individual sites. The absence of defect data or quality-based class labels from the training data prevents following the commonly use supervised learning approach to software quality modeling

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Special Issue 2, April 2014

.

II. RELATED WORK

Shi Zhonget.al. (2008).[11] developed Expert-Based Software Measurement Data Analysis with Clustering Techniques. In which software quality estimation problems, one typically constructs software quality classification or software fault prediction models using software metrics and fault data from a previous system re-lease or similar software project developed previously. Such models are then used to predict the fault-proneness of software modules that are currently under development. This allows for early tracking and detecting of potential software faults, which is critical in many high-assurance telecommunication and medical software systems. There are two main challenges in building an accurate software quality estimation model: (a) the presence of “noisy" data instances usually degrade trained models; (b) the absence of software quality measurements (fault- proneness labels) renders the construction of a classification model impossible. Clustering is naturally proposed as an exploratory data analysis tool to address these two challenges. C. Catal, et.al. (2009).[9] Marmara Res. Center, Inf. Technol. Inst., Kocaeli developed Software Fault Prediction of Unlabeled Program Modules, the quality of software components should be tracked continuously during the development of high-assurance systems such as telecommunication infrastructures, medical devices, and avionics systems. Quality assurance group can improve the product quality by allocating necessary budget and human resources to low quality modules identified with different quality estimation models. Recent advances in software quality estimation yield building defect predictors with a mean probability of detection of 71 percent and mean false alarms rates of 25 percent. Software quality estimation is not only interested in reliability, but also the other quality characteristics such as usability, efficiency, maintainability, functionality, and portability. However, some researchers prefer using the term software quality estimation for the software fault prediction modeling studies. Software metrics are used as independent variables and fault data are regarded as dependent variable in software fault prediction models. C. Catal, et.al. (2009).[7] Marmara Res. Center, Inf. Technol. Inst., Kocaeli developed Clustering and Metrics Thresholds Based Software Fault Prediction of Unlabeled Program Modules, which explains despite the amount of effort spent in the design and application of fault prediction models, software fault prediction research area still poses great challenges. Unfortunately, none of the techniques developed within last 15 years have achieved widespread applicability in the software industry due to several reasons including the lack of software tools to automate this prediction process, the unwillingness to collect the fault data, and the other practical problems. Laszlo, M. ; Mukherjee, S., (2006) [10] developed A Genetic Algorithm Using Hyper-Quad-trees for Low-Dimensional K-means Clustering. A GA describes a process that mimics evolution in nature. It maintains a population of individuals or chromosomes that evolves over successive generations. Each chromosome stores a set of genes whose values, called alleles, collectively determine the chromosome’s fitness or likelihood to reproduce or otherwise contribute to the next generation. During each generation, various genetic operators like selection, crossover, mutation, and replacement are applied to the current population to produce the individuals comprising the next generation. Parvinder S. Sandhu, Sunil Khullar,et.al,(2010) [17] developed A study on early prediction of fault proneness in software module using genetic algorithm.it describes A software fault prediction is a proven technique in achieving high software reliability. 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.

A. Existing Algorithm Disadvantage

In Existing Software prediction using K-Means Clustering Algorithm.In K-Means has some drawbacks:

1) The user has to initialize the number of clusters which is very difficult to identify.
2) It requires selection of the suitable initial cluster centers.
3) Very sensitive to noise.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Special Issue 2, April 2014

III. PROPOSED GENETIC ALGORITHM

B. Overview

Genetic algorithm are inspired by Darwin's theory about evolutions. solution to a problem solved by genetic algorithm is evolved. algorithm is started with a set of solutions called populations.solution from one population are taken from a new population.This is motivated by a hope,that the new population will be better than the old one.this is repeted until some condition is satisfied.In software development practice, however, various practical issues limit the availability of defect data for modules in the training data. For example, an organization may have not recorded or collected software defect data from previous releases or similar projects. In addition, since the organization may not have experience developing a similar system, the use of software measurement and defect data of previous projects for modeling purposes is inappropriate.

In current times, where globalization of technology has gained momentum, distributed software development is not uncommon. Under such conditions, software defect data may not be collected by all development sites depending on the organizational structure and resources of individual sites. The absence of defect data or quality-based class labels from the training data prevents following the commonly use supervised learning approach to software quality modeling. Consequently, the task of software quality estimation or labeling program modules as fault prone (fp) or not fault prone (nfp) falls on the software engineering expert. The process of labeling each program module one at a time is a laborious, expensive, and time-consuming effort. We propose a supervised classification scheme to aid the expert in the labeling process. In classification we use genetic algorithm for fault classification. In the computer science field of artificial intelligence, a genetic algorithm (GA) is a search heuristic that mimics the process of natural selection. This heuristic (also sometimes called a metaheuristic) is routinely used to generate useful solutions to optimization and search problems.

Genetic algorithms are improvement algorithms primarily based on natural biology and choice mechanisms. To apply genetic algorithms to a selected drawback, it's to be decomposed in atomic units that correspond to genes. Then individuals are often designed with correspondence to a finite string of genes, and a collection of people is termed a population. A criterion must be defined: a fitness function F that, for each individual among a population, gives F(x), the worth that is that the quality of the individual regarding the matter we wish to unravel. Once the matter is outlined in terms of genes, and fitness operate is on the market, a genetic formula computed following the method represented figure one. As Associate in nursing accommodative search technique, genetic algorithms have been wont to notice solutions to several NP-complete problems and are applied in several areas and may notice

1) Choose an initial population.
2) Evaluations of fitness operate.
3) Reproduction.
4) Crossover.
5) Mutation
6) Stopping criteria: x range of generations, a given fitness worth reached.

A better solution. Genetic formula uses 3 operators: reproduction, crossover and mutation.

1) *Initial population:*

Initial population is made arbitrarily. Every of those strings represent one possible resolution to the search problem.

2) *Fitness evaluation:*

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Special Issue 2, April 2014

Fitness values is evaluated through some applicable live. Once the fitness of the complete population has been determined, It should confirm whether or not or not the termination criteria have been glad. If the standards not glad then we tend to continue genetic operation of replica, crossover and mutation.

3) *Reproduction:*

This operator copies the people that go to participate in crossover and chosen per the fitness values. The selection is often seen as spinning a wheel wherever every individual features a slot propositional to its fitness worth. We tend to spin the wheel as over and over because the variety of the individual, and so we've a replacement population that's about to participate in crossover.

4) *Crossover:*

This module combination of choice and crossover. Choice module selects best fitness values action at law from set of fitness values from previous module.
Crossover combines current and parent action at law if parent action at law is best optimum one. Crossover provides best optimum action at law with previous parent answer.

5) *Mutation:*

Mutation rearranging all optimized resolution the for fitness checking and store it for more process. It reduces the quality of fitness checking and replica method if current resolution isn't met our criteria.

IV. ALGORITHM ADVANTAGES

6) Automatically find the initial cluster centers.
7) Generate desired number of initial centers.
8) Reduce the error rate of fault prediction.
9) Genetic algorithms are used in search and optimization, such as finding the maximum of a function over some domainspace.

V. EXPERIMENTAL DESIGN

C. System Architecture

System architecture is the conceptual model that defines the structure and/or behavior of the system. It provides a way in which products can be procured, systems can be developed an architectural overview of the overall system. The system architecture for the proposed system is given in Fig. 1.

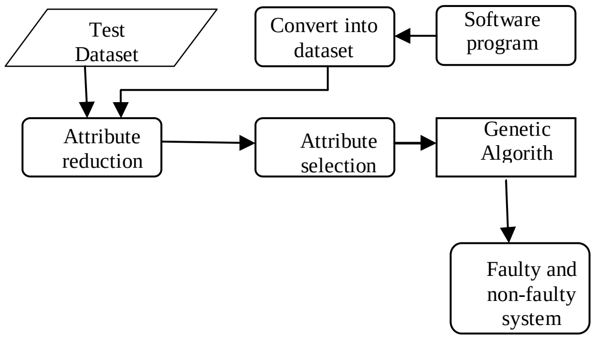


Fig. 1 System architecture

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Special Issue 2, April 2014

D. Dataset conversion

From above figure 1.1 explains the unique operator, unique operand , total lines of code ,total number of operator , total number of operators , number of distinct operators, number of distinct operands we have to find the halsted measurable.

- η_1 = the number of distinct operators
- η_2 = the number of distinct operands
- N_1 = the total number of operators
-

N_2 = the total number of operands

From these numbers, several measures can be calculated:

- Program vocabulary: $\eta = \eta_1 + \eta_2$
- Program length: $N = N_1 + N_2$
- Calculated program length: $\hat{N} = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$
- Volume: $V = N \times \log_2 \eta$
- Difficulty : $D = \frac{\eta_1}{2} \times \frac{N_2}{\eta_2}$
- Effort: $E = D \times V$

TABLE 1. AR3 SAMPLE DATASET

total loc	blank loc	comment loc	code com. loc	exe. Loc	unique opn.	unique opr.	tot. opn	tot. opr
307	116	44	5	147	138	23	245	366
3	0	0	0	3	4	6	6	8
268	72	22	0	174	125	23	337	484
11	2	0	0	9	10	4	15	17
9	2	0	0	7	7	4	11	13
10	2	0	0	8	7	4	13	15
5	0	0	0	5	5	3	5	9
28	5	1	0	22	18	12	49	53
26	6	0	0	20	16	11	41	44
15	4	0	0	11	11	5	19	22
23	5	1	0	17	18	9	25	36
10	4	0	0	6	9	6	24	23

E. Atribute selection

Get all Attribute in our dataset

1) Attribute Selection is the technique of selecting a subset of relevant features for building robust learning models.
2) We take all attributes into our process it takes so much time for processing and increase the work burden.
3) So, we reduce the total number of attributes and consider high relevance attributes only.
4) Calculate the relevance of an attribute using Attribute Evaluation.

F. weka tool

Weka (Waikato Environment for Knowledge Analysis) is a popular suite of machine learning software written in Java, developed at the University of Waikato, New Zealand. Weka is free software available under the GNU General Public License.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

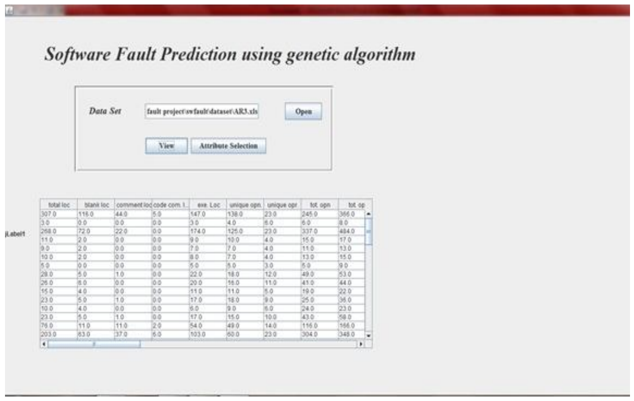
Vol. 3, Special Issue 2, April 2014

The Weka workbench contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy access to this functionality.
After loaded a data file, click "Classify"

1) *Choose a classifier,*
   Under "Classifier": click "choose", then a drop-down menu appears, Click "trees" and select "J48" – a decision tree algorithm
2) *Select a test option*
   Select "percentage split" with default ratio 66% for training and 34% for testing
3) *Click "Start" to train and test the classifier.*
   The training and testing information will be displayed in classifier output window.

VI. IMPLIMENTATION

The basic input to the system is C or Java program where the code is converted in to datasets using halsted, and the most important thing is data sets cannot be processed with all attributes only some attributes can be used by weka tool. The main purpose of using weka tool is to improve the efficiency as well its speed. Initially a threshold value will be fixed in genetic algorithm the threshold is fixed using logical regression. After that classification is done to the datasets based on their threshold value. If the threshold value is less it is non-faulty and if the threshold value is high it is faulty.



fault project\swfault\dataset\AR3.xls

total loc	blank loc	comment loc	code com. l...	exe. Loc	unique opn.	unique opr.	tot. opn	tot. op
307.0	116.0	44.0	5.0	147.0	138.0	23.0	245.0	366.0
3.0	0.0	0.0	0.0	3.0	4.0	6.0	6.0	8.0
268.0	72.0	22.0	0.0	174.0	125.0	23.0	337.0	484.0
11.0	2.0	0.0	0.0	9.0	10.0	4.0	15.0	17.0
9.0	2.0	0.0	0.0	7.0	7.0	4.0	11.0	13.0
10.0	2.0	0.0	0.0	8.0	7.0	4.0	13.0	15.0
5.0	0.0	0.0	0.0	5.0	5.0	3.0	5.0	9.0
28.0	5.0	1.0	0.0	22.0	18.0	12.0	49.0	53.0
26.0	6.0	0.0	0.0	20.0	16.0	11.0	41.0	44.0
15.0	4.0	0.0	0.0	11.0	11.0	5.0	19.0	22.0
23.0	5.0	1.0	0.0	17.0	18.0	9.0	25.0	36.0
10.0	4.0	0.0	0.0	6.0	9.0	6.0	24.0	23.0
23.0	5.0	1.0	0.0	17.0	15.0	10.0	43.0	58.0
76.0	11.0	11.0	2.0	54.0	49.0	14.0	116.0	166.0
203.0	63.0	37.0	6.0	103.0	60.0	23.0	304.0	348.0

Fig. 1 Browse Dataset and View the Attributes

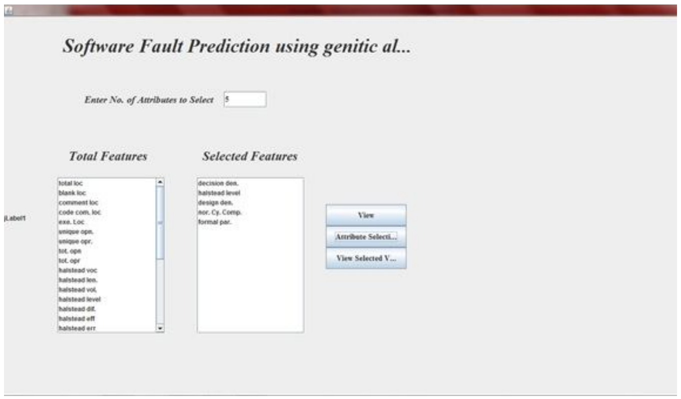
The above fig 1 shows the attribute selection level. First give the path of the dataset where it store in the system then click the view button it show the dataset attributes with values.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Special Issue 2, April 2014


Software Fault Prediction using genitic al...

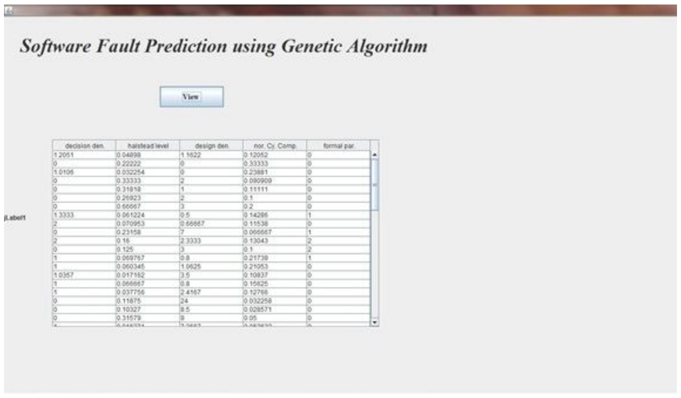
Enter No. of Attributes to Select: 5

Total Features:
total loc
blank loc
comment loc
code com. loc
exe. Loc
unique opn.
unique opr.
tot. opn
tot. opr
halstead voc
halstead len.
halstead vol.
halstead level
halstead dif.
halstead eff
halstead err

Selected Features:
decision den.
halstead level
design den.
nor. Cy. Comp.
formal par.

Fig. 2 Select the Few Attributes

In Fig. 2 by entering the number of attributes to be selected, weka tool derives the number of important attributes needed. In Fig. 3 it shows the selected attribute and their values.


Software Fault Prediction using Genetic Algorithm

decision den.	halstead level	design den.	nor. Cy. Comp.	formal par.
1.2051	0.04898	1.1622	0.12052	0
0	0.22222	0	0.33333	0
1.0106	0.032254	0	0.23881	0
0	0.33333	2	0.090909	0
0	0.31818	1	0.11111	0
0	0.26923	2	0.1	0
0	0.66667	3	0.2	0
1.3333	0.061224	0.5	0.14286	1
2	0.070953	0.66667	0.11538	0
0	0.23158	7	0.066667	1
2	0.16	2.3333	0.13043	2
0	0.125	3	0.1	2
1	0.069767	0.8	0.21739	1
1	0.060345	1.0625	0.21053	0
1.0357	0.017152	3.5	0.10837	0
1	0.066667	0.8	0.15625	0
1	0.037756	2.4167	0.12766	0
0	0.11875	24	0.032258	0
0	0.10327	8.5	0.028571	0
0	0.31579	9	0.05	0

Fig. 3 Selected Attributes with Values

VII. ALGORITHM EXAMPLE

G. Initial population:

1) Here gene of individual is modeled as a “0” or “1”.

Gj =[gj1,gj2……..gjn] where gji={0,1} ,1<=j<=k and 1<=i<=n
if gij=1, it means test case ti tested block bj
if gij=0, it means test case ti not tested block bj

2) Here all the coverage information are initial population. For Example:
s1 s2 s3 s4 s5 s6 s7
tc1={1, 1, 1, 1, 0, 0, 0}

H. Fitness evaluation:

The fitness value for an individual is combination of its coverage and its cost. The fitness value for individual calculated by ,

F(ti)=(Σk j=1gj) *wj/c(ti)

I. Cross over:

Let "m" be the size of individuals in a population and lets select an integer "i" at random between 1 and m-1,then from two individual ind1 and ind2. Then create two new individual ind3 and ind4,one made of the "i" first genes from ind1 and the "m-i" last genes of ind2,and the other made of the "i" first genes if ind2 and "m-i" last genes of ind1 .
For example:
Selected one:ind1=[1,1,1,1,0,0,0] ind2=[1,1,1,0,1,0,0,1] New one : ind3=[1,1,1,1,0,0,1] ind4=[1,1,1,0,1,0,0,0]

J. Mutation:

Based on gene model, mutation operator consists in changing a "1" in to "0" in vice versa. Here mutation operator chooses a gene at random in an individual. This gene chosen by mutation.
Example: Individual gene : [1,1,1,1,0,0,0] Output of mutation: [1,1,1,0,0,0,0]

K. Stopping criteria:

An answer is found that satisfies minimum criteria Fastened variety of generations reached. Then should be given execution condition satisfies the coverage criteria.

VIII. CONCLUTION

Standard dataset is collected from AR3, which contains large number of attributes and by using these large number of attribute values accurate value cannot be obtained, to overcome this drawback weka tool is used to reduce the attributes. From the reduced attributes classification of faulty and non-faulty records is made.

IX. FUTURE WORK

More number of iterations occurs in genetic algorithm, which consumes more time, in future this drawback can be overcomed. In Case any updations on genetic algorithm in future the number of iterations can be improved which obviously improves the efficiency of the system.

REFERENCES

[1] Bhattacherjee .V, P.K. Mohanti, and S. Kumar, (2009) 'Complexity Metrics for Analogy Based Effort Estimation' IEEE Trans. Software Eng., vol. 16, no. 11, pp. 1293-1306.
[2] Bhattacherjee. V and P.S. Bishnu, (2010) 'Unsupervised Learning Approach to Fault Prediction in Software Module' IEEE Trans. Software Eng., vol.18, pp. 210-215.
[3] Bhattacherjee.V and P.S. Bishnu, (2011)'Software Fault Prediction Using KMedoids Algorithm' IEEE Trans. Software Eng., vol. 26, no. 11, pp. 93-130.
[4] Bishnu P.S. and V. Bhattacherjee, (2009) 'Outlier Detection Technique Using Quad Tree' IEEE Trans. Software Eng., vol. 32, pp. 129-130.
[5] Bishnu P.S. and V. Bhattacherjee, (2011)'Application of K-Medoids with kd-Tree for Software Fault Prediction' IEEE Trans. Software Eng., vol. 22, no. 10, pp.321-452.
[6] Catal . C, U. Sevim, and B. Diri, (2009) 'Software Fault Prediction of Unlabeled Program Modules' IEEE Trans. Software Eng., vol. 16, no. 32, pp. 78-130.
[7] Catal, C. ; Marmara Res. Center, Inf. Technol. Inst., Kocaeli ; Sevim, U. ; Diri, B., (2009) 'Clustering and Metrics Thresholds Based Software Fault Prediction of Unlabeled Program Modules 'IEEE Trans. Software Eng., vol. 14, no. 22, pp. 56-152.
[8] Catal. C, U. Sevim, and B. Diri, (2009) 'Clustering and Metrics Thresholds Based Software Fault Prediction of Unlabeled Program Modules' IEEE Trans. Software Eng., vol. 10, no. 32, pp. 270-320
[9] Catal. C, U. Sevim, and B. Diri, (2009) 'Clustering and Metrics Threshold Based Software Fault Prediction of Unlabeled Program Modules' IEEE Trans. Software Eng., vol. 29, no. 21, pp. 320-417.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Special Issue 2, April 2014

[10] Laszlo, M. ; Mukherjee, S., (2006) 'A Genetic Algorithm Using Hyper-Quadtrees for Low-Dimensional K-means Clustering' IEEE Trans. Software Eng., vol. 10, no. 43, pp. 320-458.
[11] Shi Zhong ; Khoshgoftaar, T.M. ; Seliya, N.,(2008)'Expert-Based Software Measurement Data Analysis with Clustering Techniques' IEEE Trans. Software Eng., vol. 56, no. 31, pp. 1298-1389.
[12] Shi Zhong, Taghi M. Khoshgoftaar, and NaeemSeliya, (2004) 'Shi Zhong, Taghi M. Khoshgoftaar, and NaeemSeliya' IEEE Trans. Software Eng., vol. 16, no. 32, pp. 93-106 .
[13] Steinley. D and M.J. Brusco, (2007)'Initializing K-Means Batch Clustering: A Critical Evaluation of Several Techniques' IEEE Trans. Software Eng., vol. 32, no. 21, pp. 89-123.
[14] Stephen J. Redmond ,(2005) 'A method for initialising the K-means clustering algorithm using kd-trees' IEEE Trans. Software Eng., vol. 16, no. 11, pp. 1243-1342.
[15] Taghi M. Khoshgoftaar and Naeem Seliya,(2003) 'Software Quality Classification Modeling Using the SPRINT Decision Tree Algorithm' IEEE Trans. Software Eng., vol. 18, pp. 1283-1330.
[16] Zhong . S, T.M. Khoshgoftaar, and N. Seliya, (2004)'Analyzing Software Measurement Data with Clustering Techniques' IEEE Trans. Software Eng., vol. 34, no. 43, pp. 89-116.
Parvinder S. Sandhu, Sunil Khullar,et.al,(2010) [17] developed 'A study on early prediction of fault porneness in software module using genetic algorithm' IEEE Trans. Software Eng., vol. 21, no. 21, pp. 54-243.